UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

GII-24.19 Contramedidas IoT mediante Reinforcement Learning Documentación Técnica

Presentado por César Rodríguez Villagrá
en Universidad de Burgos — 9 de junio de 2025
Tutor: nombre tutor

Índice general

Índice general i

Índice de figuras iii

Índice de tablas iv

Apéndice A Plan de Proyecto Software 1
A.1. Introducción 1
A.2. Planificación temporal 2
A.3. Estudio de viabilidad 6

Apéndice B Especificación de Requisitos 11
B.1. Introducción 11
B.2. Objetivos generales 11
B.3. Catálogo de requisitos 12
B.4. Especificación de requisitos 13

Apéndice C Especificación de diseño 19
C.1. Introducción 19
C.2. Diseño de datos 19
C.3. Diseño arquitectónico 22
C.4. Diseño procedimental 25

Apéndice D Documentación técnica de programación 27
D.1. Introducción 27
D.2. Estructura de directorios 27
D.3. Manual del programador 28

D.4. Compilación, instalación y ejecución del proyecto 31
D.5. Pruebas del sistema 32

Apéndice E Documentación de usuario **35**
E.1. Introducción 35
E.2. Requisitos de usuarios 35
E.3. Instalación 36
E.4. Manual del usuario 37

Apéndice F Anexo de sostenibilización curricular **39**
F.1. Introducción 39
F.2. Uso sostenible de recursos 39
F.3. Uso de tecnologías de código abierto 42
F.4. Consideraciones sobre la sostenibilidad social y económica . 42
F.5. Conclusiones 42

Bibliografía **43**

Índice de figuras

A.1. Gráfico de Burn up del proyecto 3
A.2. Diagrama de estado del proyecto 4
A.3. Diagrama de trabajo por tema del proyecto 5
A.4. Precio estimado de AWS 7
A.5. Costes de AWS en mayo 8

C.1. Esquema de la base de datos del proyecto 21
C.2. Clases de datos del proyecto 22
C.3. Diagrama de clases del proyecto 23
C.4. Matriz de dependencias del proyecto 24
C.5. Mapa de navegación de la aplicación web 25

D.1. Gráficas de SonarQube del proyecto 31

F.1. Ancho de banda ahorrado por el uso Cloudflare 41

Índice de tablas

B.1. Tabla de requisitos funcionales 12
B.2. Tabla de requisitos no funcionales 13
B.3. CU-1 Registro de usuarios. 14
B.4. CU-2 Inicio de sesión. 15
B.5. CU-3 Gestión de usuarios. 16
B.6. CU-4 Importación/Exportación de datos. 17
B.7. CU-5 Visualización de datos y documentación. 18

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apéndice se pretende mostrar el plan de proyecto software que se ha seguido durante el desarrollo del proyecto.

La implementación del plan de proyecto siguiendo una metodología ágil, ha permitido una mayor flexibilidad y adaptabilidad a los cambios que han surgido durante el desarrollo. En este caso, se ha utilizado principalmente la metodología de Kanban, debido a la flexibilidad que presenta en la organización y gestión del trabajo.

En cuanto a la estructura de este anexo, se explica la planificación temporal del proyecto y el estudio de viabilidad, tanto económica como legal.

Kanban

La metodología Kanban [2] se ha utilizado para gestionar el flujo de trabajo del proyecto. En el que se usa el tablero de Kanban predeterminado en los GitHub Projects, donde se han usado las siguientes columnas:

- **Backlog:** En esta columna se encuentran todas las tareas que se han propuesto para el proyecto, pero que aún no se han comenzado a desarrollar, limitado a 5 tareas.
- **Ready:** En esta columna se encuentran todas las tareas que estaban en el *Backlog*, y que están listas para ser desarrolladas.

- **In Progress:** En esta columna se encuentran todas las tareas que están en proceso de desarrollo, limitado a 3 tareas.

- **In Review:** En esta columna se encuentran todas las tareas que han sido completadas y están en revisión por parte de los autores del proyecto, limitado a 5 tareas.

- **Done:** En esta columna se encuentran todas las tareas que ya han sido completadas.

Todas las tareas que se han terminado quedan registradas en la columna *Done*, y se han ido moviendo a lo largo de las columnas según se ha ido avanzando en el desarrollo del proyecto. En cada tarea, si forma parte de desarrollo de una parte de código o documentación, quedan registrado en la tarea una rama de desarrollo de GitHub, y una pull request asociada a la tarea en la que se han incorporado los cambios realizados a la rama principal. En cada pull request se han incluido comentarios creados pro GitHub Copilot, en la que se ha incluido una descripción de los cambios realizados, y una lista de los archivos que han sido modificados, y una revisión de errores o propuestas de mejora para implementar antes de la incorporación de cambios.

A.2. Planificación temporal

Cada semana, si es necesario, se actualiza el tablero del proyecto. Se ha planificado cada semana una reunión de seguimiento del proyecto con los tutores, en la que se han revisado las tareas que se han realizado a lo largo de la semana. En la que se plantean propuestas de mejora y correcciones a las tareas realizadas.

Las reuniones semanales empezaron el día 6 de febrero de 2025 hasta el día 5 de junio de 2025, en que cada jueves a las 10:00, se han realizado las reuniones, a excepción de festivos y vacaciones.

Aclaración: Como la generación de esta documentación corresponde a una tarea aún en curso, en el diagrama aparece una tarea en estado .[En] progreso". No obstante, en el momento de la entrega final, todas las tareas estarán ubicadas en la columna "Done".

Burn up

El gráfico de Burn up muestra el progreso del proyecto a lo largo del tiempo, en el que se puede ver la cantidad de trabajo completado y el trabajo restante. En el eje horizontal se muestra el tiempo, y en el eje vertical se muestra la cantidad de trabajo completado. La línea morada representa el trabajo completado, y la línea verde representa el trabajo restante. El área entre las dos líneas representa el trabajo pendiente.

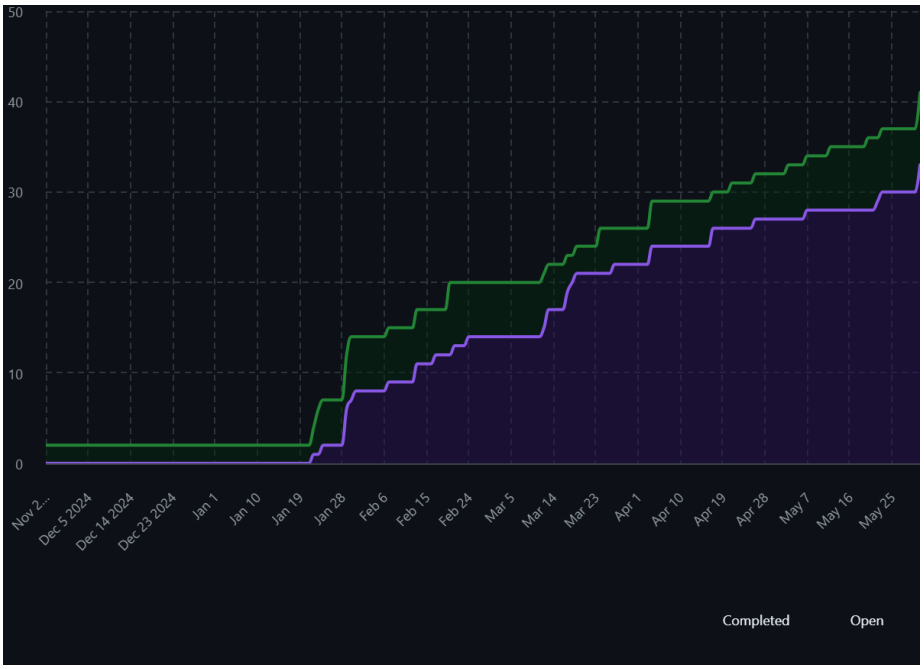

Figura A.1: Gráfico de Burn up del proyecto

Diagrama de estado

El diagrama de estado del proyecto muestra los diferentes estados en los que se encuentran actualmente el estado de las tareas del proyecto. Como el desarrollo del proyecto ya se ha terminado, el estado de las tareas se encuentra en "Done".

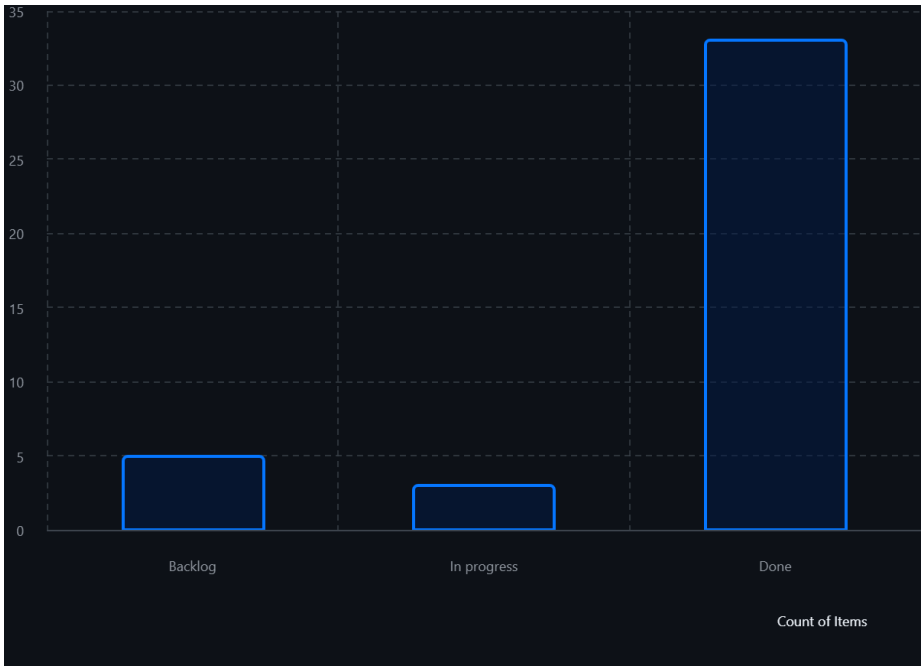

Figura A.2: Diagrama de estado del proyecto

Diagrama de trabajo por tema

El diagrama de trabajo por tema muestra los diferentes temas en los que se ha trabajado a lo largo del proyecto, y la cantidad de trabajo realizado en cada uno de ellos. En el eje horizontal se muestran los diferentes temas, y en el eje vertical se muestra la cantidad de trabajo realizado. Donde de izquierda a derecha se pueden ver: Lecturas de artículos, Procesos de CI, Desarrollo de la web, con la API y el frontend, Documentación, Desarrollo y entrenamiento del agente.

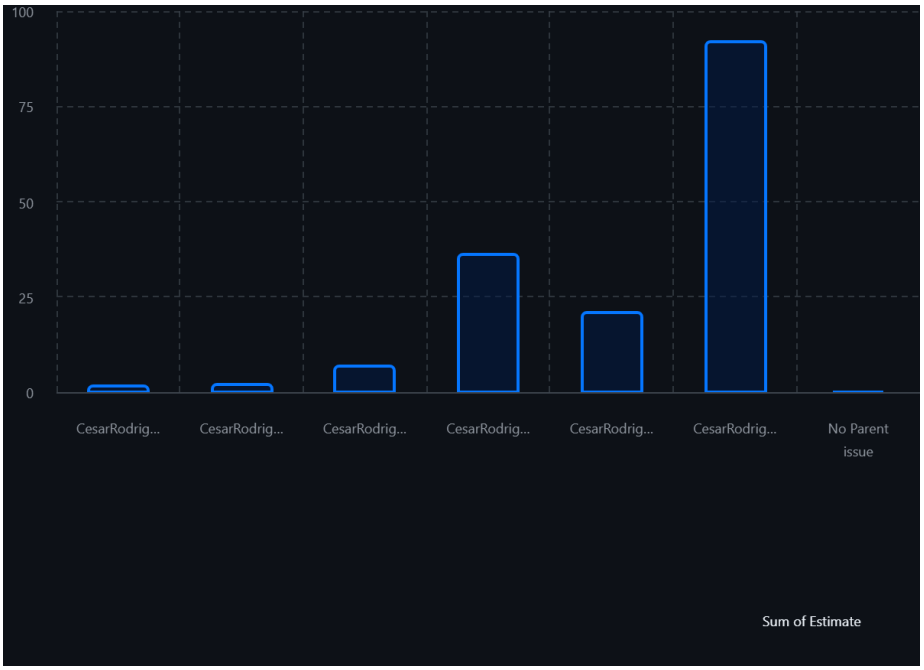

Figura A.3: Diagrama de trabajo por tema del proyecto

El orden de los temas que han requerido más trabajo ha sido el siguiente:

1. **Desarrollo y entrenamiento del agente #7:** El tema que más trabajo ha requerido ha sido el desarrollo y entrenamiento del agente, ya que ha requerido una investigación previa, tanto para la implementación del agente y entorno como por el uso de aprendizaje por refuerzo.

2. **Desarrollo de la web, con la api y el frontend #6:** La parte de desarrollo de la aplicación web también ha requerido un gran esfuerzo, ya que se ha tenido que desarrollar tanto el backend como el frontend, así como la integración con la API REST y la base de datos.

3. **Documentación #68:** Ya que se ha tenido que documentar tanto el código como el proyecto en general, así como la memoria y anexos del proyecto.

4. **Procesos de CI #47:** Los procesos de integración continua y calidad de código han sido claves en el inicio del proyecto para asegurar la calidad del código y la correcta ejecución de las pruebas, así como para facilitar el despliegue de la aplicación web.

5. **Lecturas de artículos #10:** La lectura de artículos ha sido necesaria para la investigación previa al desarrollo del proyecto, así como para la comprensión de los conceptos y técnicas utilizados en el proyecto, y la mejora de la función de recompensa del agente.

A.3. Estudio de viabilidad

Realizar un estudio de viabilidad es una parte importante del futuro de un proyecto, ya que permite evaluar si el proyecto es viable desde diferentes perspectivas. En este caso, se ha realizado un estudio de viabilidad económica y legal, en el que se pretende estudiar cúanto dinero hace falta para realizar las tareas del proyecto, y si se cumplen las normativas legales necesarias para su desarrollo y despliegue.

Viabilidad económica

Los costes se dividen principalmente en dos categorías: costes de personal y costes de infraestructura.

Los costes de personal incluyen los salarios de los integrantes del proyecto. En este caso, se ha considerado ún unico empleado, en el cual ha trabajado en el proyecto de forma activa durante más de 5 meses, con una dedicación aproximada de 17 horas semanales, en el que dan unas horas totales de aproximadamente 380 horas de trabajo totales. El salario estimado para el empleado con cargo de desarrollador junior es de 11,02 euros por hora [5], lo que da un coste total de personal de aproximadamente 4187 euros en el tiempo en el que se ha trabajado. En el que para un salario a tiempo completo serían aproximadamente 1790 euros al mes [5].

Los costes de infraestructura incluyen los gastos relacionados con la infraestructura necesaria para el desarrollo y despliegue del proyecto, como servidores, servicios en la nube y licencias de software. En este caso, debido a que ha utilizado en su mayoría servicios gratuitos, los costes de infraestructura son bajos. Los únicos servicios de pago que se han utilizado es el dominio de la web, comprado en Cloudflare, y el servicio de hosting de la web, que se ha realizado en AWS.

El dominio de la web, al ser *.com* tiene un coste de 10,44 dólares al año, lo que equivale a aproximadamente 9,49 euros al año.

Para calcular el precio estimado del hosting de la web, se ha utilizado la calculadora de precios de AWS [9], en el que se ha estimado un coste mensual

de aproximadamente 1,39 dólares al mes, con un coste anual de 16,68 dólares , lo que equivale a aproximadamente 14,64 euros al año, mostrado en la **Figura A.4**. Para la estimación del coste se ha tenido en cuenta el uso de una instancia t3.medium, que es de las más económicas, y el uso de 1 GB de almacenamiento en Amazon S3, que es el servicio de almacenamiento de AWS, que en este caso entra en la capa gratuita. Además, se ha tenido en cuenta el uso de 1 GB de transferencia de datos al mes, que es el límite gratuito de AWS. Es importante remarcar que se ha simulado con un uso de 1 hora al día, ya que la web no requiere un uso mayor.

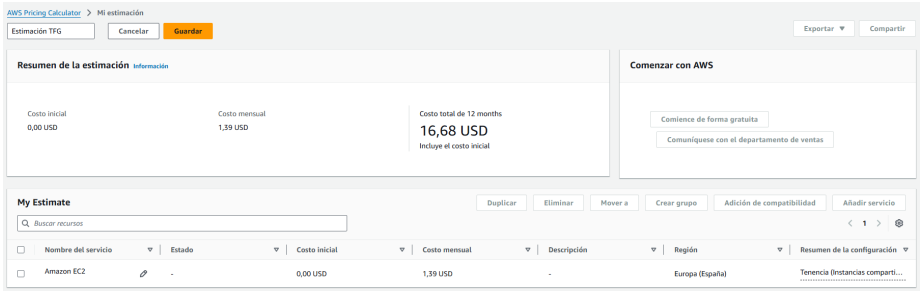

Figura A.4: Precio estimado de AWS

Como se muestra en la **Figura A.5**, el coste usado en mayo de 2025 ha sido de aproximadamente 0,60 dólares, lo que equivale a aproximadamente 0,53 euros, ya que el proyecto se ha desplegado a finales en mayo de 2025. Como se puede obvservar, el mayor coste ha sido el de VPC, que corresponde al uso de una IP fija, facilitando la dirección desde los DNS de Cloudflare, en el que en el plan gratuito incluye la IP fija mientras se use una instancia t3.medium, que es la más económica, que cumpla con los requisitos de la aplicación. Debido a que el proyecto ha estado con el servidor apagado para reducir costes derivó en coste por uso de la VPC. Para optimizar este costo se realizó un script en el servidor en el que cuando se enciende, hace una petición a la API de Cloudflare para actualizar los DNS, así eliminado la necesitadad de VPS, manteniendo los cambios de ip transparentes para el usuario.

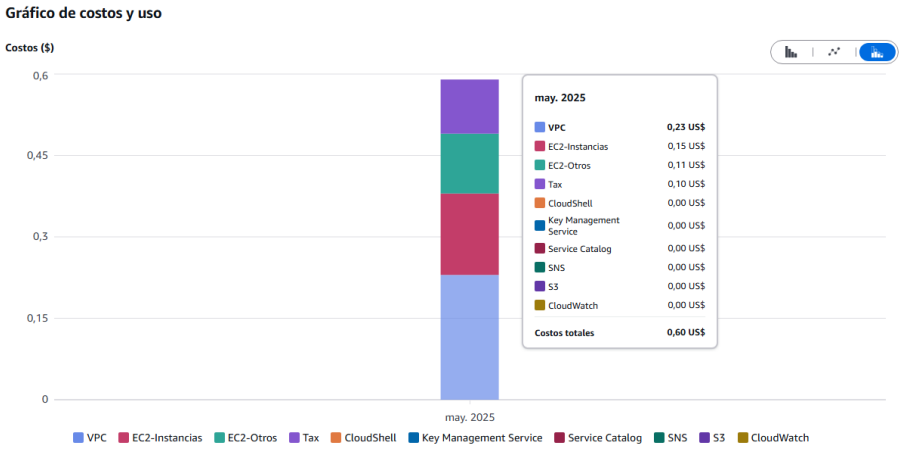

Figura A.5: Costes de AWS en mayo

Los costes totales del proyecto, sumando los costes de personal y de infraestructura, son aproximadamente 1792,02 euros, que se desglosan de la siguiente manera:

- **Costes de personal:** 1790 euros.
- **Costes de infraestructura:** 2,02 euros (9,49/12 euros del dominio + 1,23 euros de AWS).

En resumen, la viabilidad económica del proyecto se ha evaluado teniendo en cuenta los costes de personal y de infraestructura, determinando que el proyecto es viable económicamente, ya que los costes totales son bastante reducidos para el posible potencial del proyecto.

Viabilidad legal

Para la viabilidad legal del proyecto, se ha tenido en cuenta la normativa vigente en materia de protección de datos y propiedad intelectual. Las normativas aplicables más relevantes son:

- **Reglamento General de Protección de Datos (RGPD):** Este reglamento establece las normas para la protección de datos personales en la Unión Europea. [1]
- **Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales (LOPDGDD):** Esta ley complementa el RGPD

en España y establece normas adicionales para la protección de datos personales. [4]

Para analizar la viabilidad legal del proyecto, se debe tener en cuenta que el desarrollo y la implementación de un agente de aprendizaje por refuerzo encargado de la mitigación de ataques permitiendo o denegando el tráfico de red, debe garantizar que se cumplen las normativas vigentes.

En relación con la protección de datos personales del agente, cumple la normativa vigente, ya que no se recopilan ni procesan datos personales de los usuarios, ya que no obtiene información personal, ya que solo lee la capacidad de la cola, no accede a los datos de los paquetes. En la parte de la web, debido a que tampoco se recopilan datos personales de los usuarios, ya que en el registro de usuarios no se solicita información personal, y la única información que se almacena es el nombre de usuario y la contraseña, que se almacenan de forma segura en la base de datos, cumpliendo las mejores prácticas de seguridad, principalmente la autentificación y autorización de los usuarios, ya que utiliza Spring Security [17]. También usa servicios en la nube de terceros, como AWS [8], Cloudflare [11] y GitHub [12], que cumplen con las normativas de protección de datos y seguridad, ya que se utilizan para el despliegue de la aplicación web, la gestión de dominios y seguridad del tráfico de la web y el control de versiones del código fuente, respectivamente.

En cuanto a las licencias de los paquetes y liberías usadas, se han utilizado principalmente:

- **BSD-3-Clause:** Usada por más de 15 dependencias del proyecto.
- **BSD-2-Clause:** Usada por más de 13 dependencias del proyecto.
- **MIT:** Usada por 6 dependencias del proyecto.
- **Apache-2.0:** Usada por 5 dependencias del proyecto.

Todas estas licencias son compatibles con el proyecto, ya que permiten su uso y compercialización de la aplicación, siempre y cuando se cumplan las condiciones establecidas en cada licencia. En este caso, se ha cumplido con las condiciones de las licencias, ya que se ha incluido la información de las licencias en el proyecto, y no se ha modificado el código de las dependencias, ya que se han utilizado tal cual están en sus repositorios oficiales.

En resumen el proyecto cumple con la normativa vigente en materia de protección de datos y propiedad intelectual, permitiendo su desarrollo y despliegue sin problemas legales.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apéndice se presenta la especificación de requisitos del proyecto, que incluye los objetivos generales, el catálogo de requisitos y la especificación de requisitos. La especificación de requisitos es una parte fundamental del desarrollo de software, ya que permite definir de forma clara y precisa las necesidades y expectativas del cliente.

B.2. Objetivos generales

La realización de este proyecto tiene como objetivo principal el desarrollo de un agente inteligente capaz de mitigar ataques de denegación de servicio en redes de computadoras, especialmente en redes de internet de las cosas (IoT). Para lograr este objetivo, se han establecido los siguientes objetivos:

- Desarrollar un agente inteligente, y su entorno mediante aprendizaje por refuerzo que pueda mitigar ataques de denegación de servicio en redes.
- Hacer un estudio de los parámetros de la función de recompensa para optimizar el rendimiento del agente.
- Proporcionar una interfaz de usuario intuitiva para el usuario, que permita la visualización de estadísticas y gráficas relacionadas con el agente.

Número	Requerimiento	Descripción	Prioridad
RF1	Registro de usuarios	Se permitirá registrarse nuevos usuarios al sistema.	Alta
RF2	Inicio de sesión	Los usuarios registrados en el sistema podrán acceder con las credenciales que introdujo en el registro, pudiendo acceder a los modelos guardados previamente.	Alta
RF3	Panel de administración de usuarios	Solo los usuarios con permisos de administrador podrán acceder a la gestión de usuarios, pudiendo borrar, y editar los usuarios registrados en el sistema.	Media
RF4	Importación y exportación de datos	Se podrá importar y exportar los datos disponibles en la web.	Baja
RF5	Documentación del sistema	La web incluirá toda la información relacionada sobre el proyecto.	Media
RF6	Visualización de los datos del agente	Se podrán visualizar datos relacionados con el agente, incluyendo datos de entrenamiento como del agente actuando.	Alta

Tabla B.1: Tabla de requisitos funcionales

- Implementar un sistema que permita almacenar, consultar y eliminar modelos del agente.

B.3. Catálogo de requisitos

En el proyecto se han identificado los requisitos funcionales y no funcionales, desarrollados en la **Tabla B.1** y la **Tabla B.2** respectivamente. Estos requisitos son esenciales para el correcto funcionamiento del sistema y deben ser cumplidos para garantizar los objetivos propuestos.

Número	Requerimiento	Descripción	Prioridad
RNF1	Facilidad de uso	Las interfaces de usuario deben ser simples y adecuadas para todo tipo de usuarios.	Alta
RNF2	Rendimiento de la web	El tiempo de carga de la web no debe ser superior a 1 segundo para las operaciones más comunes.	Media
RNF3	Respuesta ante fallos	Si ocurre un fallo, debe tener el menor impacto posible en la web, mostrando un mensaje de error al usuario con el motivo del error.	Alta
RNF4	Seguridad	Un usuario solo debe poder acceder a sus modelos y solo los usuarios con permisos de administrador podrán hacer la gestión de usuarios.	Alta
RNF5	Minimizar los paquetes que se descartan por el agente	El agente deberá priorizar permitir el tráfico antes que denegarlo, intentando que los paquetes que se descartan sean los mínimos posibles.	Alta

Tabla B.2: Tabla de requisitos no funcionales

B.4. Especificación de requisitos

En esta sección se detallan los casos de uso del sistema, que describen las principales interacciones entre los usuarios y el sistema para cumplir con los requisitos funcionales identificados. Cada caso de uso incluye una descripción, precondiciones, acciones, postcondiciones, excepciones e importancia.

CU-1	**Registro de usuarios**
Versión	1.0
Autor	César Rodríguez Villagrá
Requisitos asociados	RF1
Descripción	Permite a un nuevo usuario crear una cuenta en el sistema proporcionando sus credenciales.
Precondición	▪ El usuario no debe estar registrado previamente. ▪ Disponibilidad del formulario de registro.
Acciones	1. El usuario accede al formulario de registro. 2. Introduce los datos requeridos. 3. El sistema valida los datos. 4. Si la validación es correcta, crea la cuenta y confirma el registro.
Postcondición	▪ El usuario queda registrado en el sistema. ▪ Puede iniciar sesión con sus credenciales.
Excepciones	▪ Datos inválidos o incompletos: muestra mensaje de error. ▪ Nombre de usuario ya registrado: notificación para usar otro nombre.
Importancia	Alta

Tabla B.3: CU-1 Registro de usuarios.

CU-2	**Inicio de sesión**
Versión	1.0
Autor	César Rodríguez Villagrá
Requisitos asociados	RF2, RF1
Descripción	Permite a usuarios registrados acceder al sistema introduciendo sus credenciales.
Precondición	▪ El usuario debe estar registrado. ▪ Disponibilidad del sistema.
Acciones	1. El usuario abre el formulario de inicio de sesión. 2. Introduce usuario y contraseña. 3. El sistema valida las credenciales. 4. Si son correctas, el usuario accede al sistema.
Postcondición	▪ El usuario está autenticado. ▪ Puede acceder a sus modelos.
Excepciones	▪ Credenciales incorrectas: muestra mensaje de error.
Importancia	Alta

Tabla B.4: CU-2 Inicio de sesión.

CU-3	**Gestión de usuarios**
Versión	1.0
Autor	César Rodríguez Villagrá
Requisitos asociados	RF3
Descripción	Los administradores pueden añadir, modificar o eliminar usuarios registrados en el sistema.
Precondición	▪ El usuario debe tener permisos de administrador. ▪ El sistema debe estar operativo.
Acciones	1. El administrador accede al panel de gestión de usuarios. 2. Puede consultar la lista de usuarios. 3. Selecciona acciones como crear, editar o eliminar usuarios. 4. El sistema aplica los cambios solicitados.
Postcondición	▪ La base de datos de usuarios queda actualizada. ▪ Cambios reflejados en futuras sesiones.
Excepciones	▪ Intento de acción sin permisos: no permite el acceso. ▪ Error en la base de datos: muestra una notificación de fallo.
Importancia	Media

Tabla B.5: CU-3 Gestión de usuarios.

CU-4	**Importación/Exportación de datos**
Versión	1.0
Autor	César Rodríguez Villagrá
Requisitos asociados	RF4
Descripción	Permite a los usuarios importar y exportar datos relacionados con los modelos y configuraciones.
Precondición	▪ Usuario autenticado. ▪ Formatos compatibles disponibles.
Acciones	1. Si el usuario quiere descargar el modelo, selecciona el modelo deseado. 2. El sistema procesa la operación. Utilizando el nombre adecuado. 3. Muestra resultado y confirma la acción.
Postcondición	▪ Datos importados o exportados correctamente.
Excepciones	▪ Archivo incompatible. ▪ Error durante la operación.
Importancia	Baja

Tabla B.6: CU-4 Importación/Exportación de datos.

CU-5	**Visualización de datos y documentación**
Versión	1.0
Autor	César Rodríguez Villagrá
Requisitos asociados	RNF1, RNF3
Descripción	Permite a los usuarios consultar la documentación y datos relacionados con el agente RL a través de la web.
Precondición	▪ Usuario autenticado. ▪ Disponibilidad de la web.
Acciones	1. El usuario accede al panel de visualización. 2. Consulta documentación y estadísticas disponibles. 3. Puede filtrar y buscar información relevante.
Postcondición	▪ Usuario obtiene la información solicitada.
Excepciones	▪ Fallo en la carga de datos. ▪ Error de conexión.
Importancia	Alta

Tabla B.7: CU-5 Visualización de datos y documentación.

Apéndice C

Especificación de diseño

C.1. Introducción

La especificación de diseño es una parte fundamental del desarrollo de software, ya que permite definir de forma detallada el comportamiento y la estructura del sistema a desarrollar. En este apéndice se presenta la especificación de diseño del proyecto, que incluye el diseño de datos, el diseño arquitectónico y el diseño procedimental.

C.2. Diseño de datos

Para la transferencia de datos entre el frontend y el backend se han utilizado las clases de transferencia de datos (DTOs) con ayuda de Thymeleaf. En cambio, para la transferencia de datos entre el backend y la API se ha utilizado JSON, ya que es un formato ligero y sencillo para la comunicación entre sistemas.

La motivación de usar DTOs es:

- **Reducción del acoplamiento:** Los DTOs permiten reducir el acoplamiento entre el frontend y el backend, lo que facilita la evolución y mantenimiento de la aplicación.
- **Seguridad y optimización de la transferencia de datos:** Los DTOs permiten transferir solo los datos necesarios entre el frontend y el backend, lo que reduce el tamaño de las peticiones y respuestas, además de proporcionar una capa adicional de seguridad, ya que por

ejemplo en el DTO de los usuarios no incluirá la contraseña de los mismos.

- **Uso como formularios:** Los DTOs permiten definir de forma más clara los distintos tipos de validaciones de datos, por ejemplo, en el caso de los formularios, se pueden definir las validaciones necesarias para cada campo del formulario, útil por ejemplo para los formularios de registro.

Para la gestión de los datos de la base de datos se ha utilizado una base de datos relacional, en este caso MySQL. Se ha diseñado un esquema de base de datos que incluye las siguientes tablas:

- **User**: Almacena información sobre los usuarios del sistema.
- **Role**: Define los roles de los usuarios.
- **TrainedModel**: Almacena los modelos de IA entrenados y datos relacionados.

Que se relacionan entre sí de la siguiente manera: **Figura C.1**

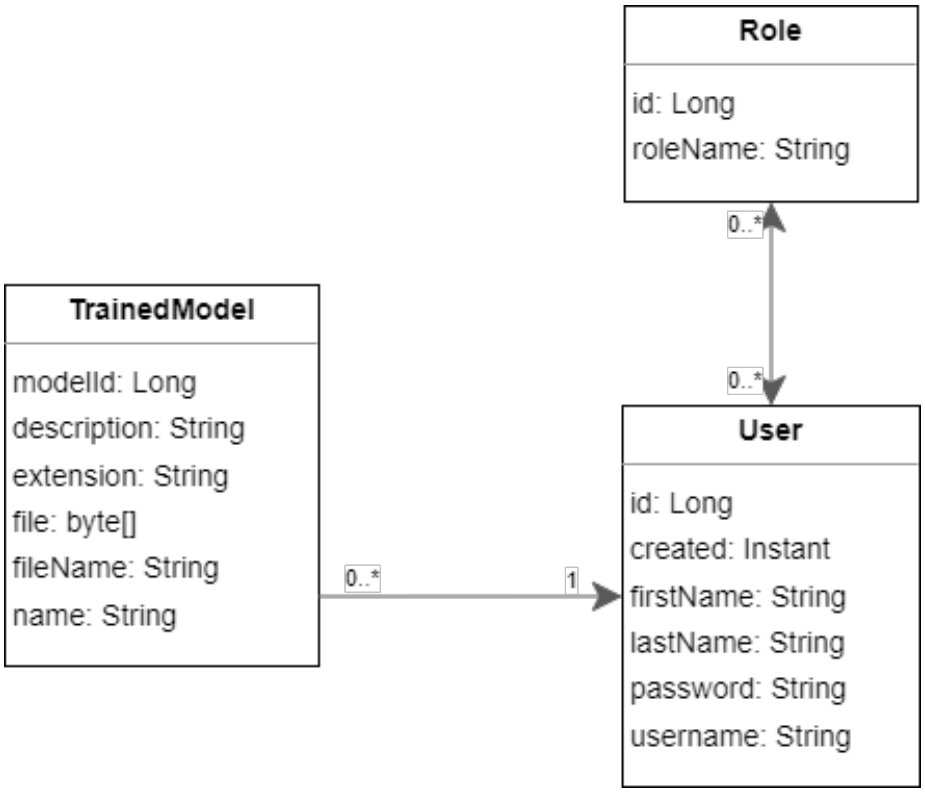

Figura C.1: Esquema de la base de datos del proyecto

Para el intercambio de datos se han definido las siguientes clases de datos:

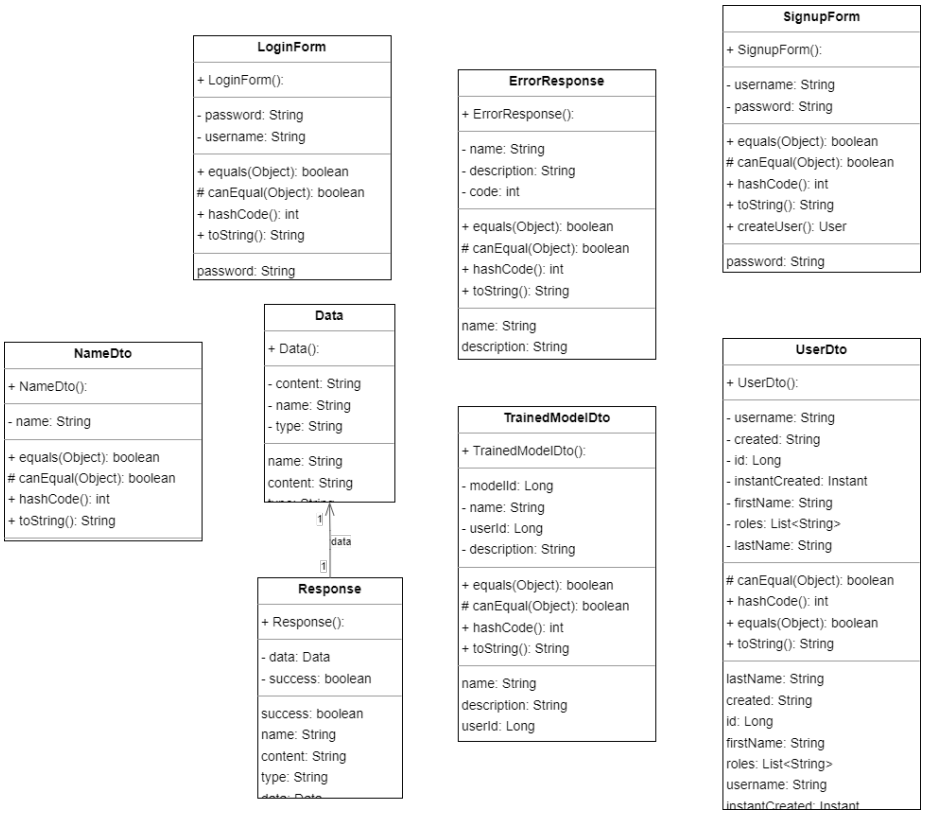

Figura C.2: Clases de datos del proyecto

En el que se muestran las clases, con sus contructores, atributos, métodos y propiedades, junto con su visibilidad y tipos.

C.3. Diseño arquitectónico

El diagrama de clases de alto nivel del proyecto muestra la estructura general del sistema, incluyendo las principales clases y sus relaciones. Este diagrama es una representación visual de cómo se organizan las clases en el sistema y cómo interactúan entre sí.

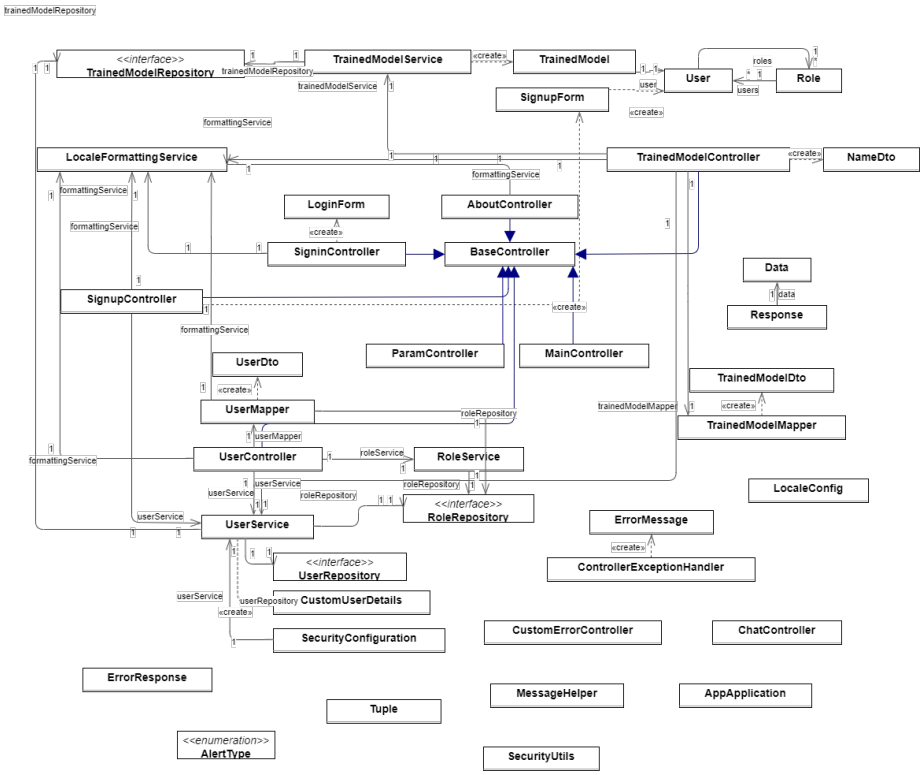

Figura C.3: Diagrama de clases del proyecto

Para facilitar la visualización de las dependencias, se ha utilizado una matriz de dependencias agrupadas por paquetes. En el que los paquetes que se muestran en las primera columna, En cada columna, los paquetes siguen el mismo orden que en la primera columna, y se marcan en cada celda cúantas dependencias tiene un paquete con otro.

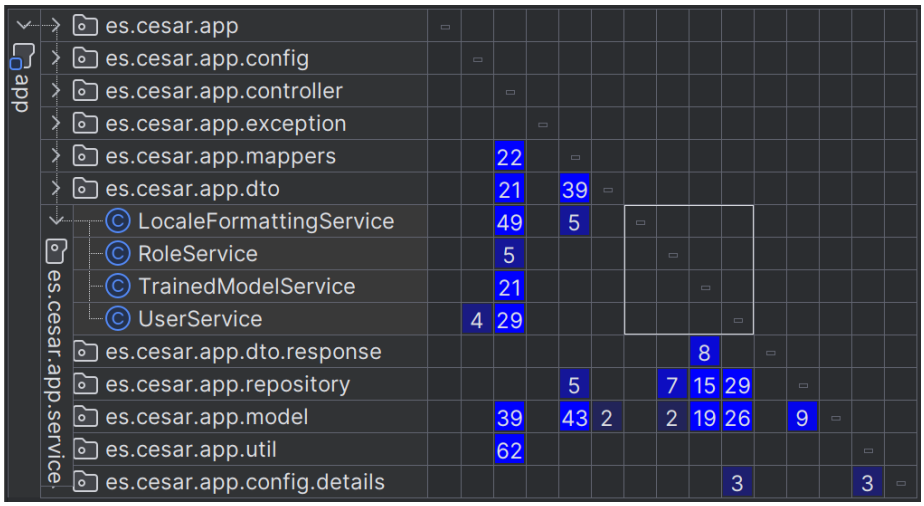

Figura C.4: Matriz de dependencias del proyecto

Como se puede observar en la matriz de la **Figura C.4**, el paquete es.cesar.app.service tiene muchas dependencias, por lo que solo en este paquetes se han incluido las depencias de las clases individuales. En este caso la mayor cantidad de dependencias corresponden con la columna del paquete de los controladores(tercera columna). Por ejemplo, la clase UserService tiene dependencias con los paquetes de los controladores (es.cesar.app.controller) y configuración (es.cesar.app.config), es decir el paquetes de configuración(4 referencias) y controladores(29 referencias) dependen de la clase UserService, pero no al revés.

En la parte de la aplicación web, se ha utilizado el patrón de diseño Modelo-Vista-Controlador (MVC) para separar la lógica de negocio, la presentación y el control de flujo de la aplicación. Este patrón permite una mayor modularidad, facilita el mantenimiento y mejora la evolución del sistema. El mapa de navegación es una representación visual de las diferentes vistas de la aplicación y cómo se relacionan entre sí. En este caso, el mapa de navegación muestra las diferentes páginas de la aplicación web y cómo se accede a ellas desde otras páginas.

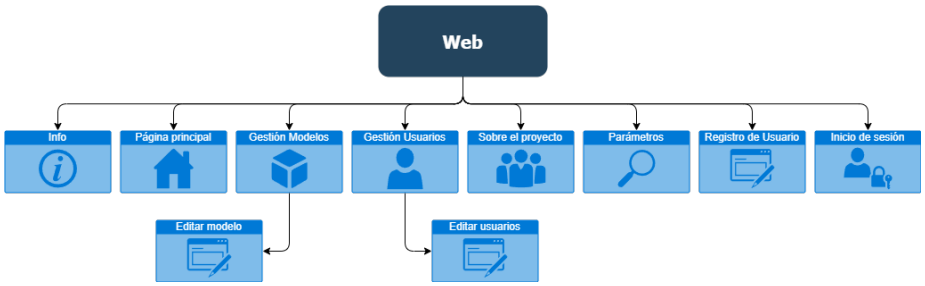

Figura C.5: Mapa de navegación de la aplicación web

La **Figura C.5** muestra la estructura de navegación de la aplicación web, incluyendo las diferentes vistas y cómo se conectan entre sí.

C.4. Diseño procedimental

El diseño procedimental de este proyecto se ha centrado en estructurar el código de forma modular, facilitando así el desarrollo, mantenimiento y la posible extensión del sistema. En el que, para conseguir esto, se han utilizado buenas prácticas de programación y patrones de diseño.

Módulos

Se ha realizado la división en 3 módulos:

- RL
- app
- docs

En el que cada módulo tiene el mismo nombre que en el repositorio. En el módulo de RL se encuentra todo lo relacionado con el aprendizaje por refuerzo, en el que proporciona en la carpeta principal lo relacionado con el entrenamiento y visualización de gráficas del modelo, y el sub-módulo de api, que proporciona una API REST para que desde la web pueda obtener datos de este módulo.

En el módulo app proporciona todo lo necesario de la aplicación web hecha con Spring Boot, en el que está subdividida en módulos, separando la configuración, los controladores, los archivos de transferencia de datos, los servicios etc.

En el módulo docs, proporciona todo lo relacionado con la generación de la documentación del proyecto, en el que encuentra los archivos base, y los archivos compilados.

Los módulos anteriores se unen mediante docker-compose (menos el de la documentación), creando un contenedor para la API, y otro para la aplicación web, junto a estos, también necesita un contenedor de una base de datos MySQL para persistir los datos de la web, garantizando un diseño modular y escalable.

Patrones de diseño aplicados

Para el diseño procedimental es importante el uso de patrones de diseño que faciliten la reutilización del código y la separación de responsabilidades. En este proyecto se han utilizado los siguientes patrones de diseño:

- **Método plantilla(template method):** Este patrón se ha utilizado para definir los generadores de paquetes de datos para el entrenamiento de los modelos de IA. Definiendo una vez la estructura del generador de paquetes, y sobrescribiendo en las subclases los pesos de los tamaños de los paquetes de datos.
- **Estrategia(strategy):** En los generadores de tráfico, este patrón permite que se puedan definir diferentes estrategias de generación de paquetes de datos, en este caso de tráfico benignos y DoS, que se pueden intercambiar fácilmente sin afectar al resto del sistema.
- **Máquina de estados(state machine):** Este patrón se ha utilizado para gestionar los diferentes estados del tráfico, en el que hay estado de ataque y normal, intercambiando en cada estado el generador de paquetes de datos correspondiente. Permitiendo que se puedan definir diferentes estados y modificar las transiciones entre ellos, facilitando la gestión del flujo de la aplicación.
- **Singleton:** Este patrón se ha utilizado para la clase de instanciación de la API, que asegura que solo haya una instancia de la clase en toda la aplicación.

El uso de estos patrones permite una mayor modularidad, reutilización del código y facilita el mantenimiento y evolución del sistema, asegurando una mayor calidad de código.

Apéndice D

Documentación técnica de programación

D.1. Introducción

Este apéndice contiene la documentación técnica de programación del proyecto, incluyendo la estructura de los directorios, manual del programador, cómo ejecutar el proyecto y las pruebas realizadas sobre el sistema. Esta documentación está destinada a las personas que quieran contribuir al proyecto o entender su funcionamiento interno.

D.2. Estructura de directorios

La estructura de directorios del proyecto que se encuentra en el repositorio de GitHub principalmente consta de los siguientes directorios:

- **app:** Directorio correspondiente al proyecto en Java con Spring Boot 3, que contiene todo el código y lógica de la aplicación web.
- **docs:** Directorio que contiene la documentación general del proyecto, que es donde se encuentra tanto los archivos fuente de LaTeX, como los pdfs de la memoria y el anexo.
- **RL:** Directorio que alberga el código fuente de la parte de Python, que contiene el código del agente de *reinforcement learning* y la implementación sencilla de una API REST para la comunicación entre el agente y la aplicación web.

- **sql:** Que contiene los archivos sql con los que se iniciará la base de datos.

A parte de estas carpetas, en el directorio principal se puede encontrar los siguientes archivos:

- **README.md:** Archivo que contiene una breve descripción del proyecto y las insignias (badges) de la calidad del código, la cobertura y el estado de las acciones de GitHub.
- **.gitignore:** Archivo de configuración de Git, que indica los archivos y carpetas deben ser ignorados por el control de versiones.
- **sonar-project.properties:** Archivo de configuración de SonarQube, que contiene la configuración del análisis de calidad del código.
- **docker-compose.yml:** Archivo de configuración de Docker Compose, que permite levantar todos los contenedores necesarios de Docker para el correcto funcionamiento del proyecto.
- **.env:** Archivo que contiene las variables de entorno necesarias para la ejecución del proyecto. El contenido de este archivo se desarrolla en más detalle en la subsección de Políticas de seguridad.
- **package.json:** Que es necesario para la ejecución de test con Jest, para los archivos de JavaScript.
- **pre_containers.sh:** Contiene unos comandos necesarios para la correcta ejecución de los contenedores en el servidor de AWS EC2.

D.3. Manual del programador

En la parte de gestión del repositorio GitHub, se ha realizado mediante un proyecto de GitHub en el que se ha realizado la gestión de tareas(issues) y la planificación de las mismas, en las que la mayoría se han convertido en ramas de desarrollo para añadir los cambios propuestos.

Acciones de GitHub

Las acciones de GitHub (GitHub Actions) son una herramienta de integración continua y entrega continua (CI/CD) que permite automatizar tareas en el flujo de trabajo de desarrollo de software. En este proyecto

se han utilizado para realizar pruebas automáticas del código, pruebas de calidad y la Compilación de la documentación de formato LaTeX a PDF. A continuación se describen las acciones implementadas:

- **Compilación de la documentación:** Se ha creado una acción que compila la documentación del proyecto en formato LaTeX a PDF. Esta acción se ejecuta cada vez que se realiza un push a la rama principal del repositorio en el que se han cambiado los archivos fuente de la propia documentación.
- **SonarQube:** Para las pruebas de calidad de código, se ha implementado SonarQube, que permite analizar el código en busca de errores, vulnerabilidades y problemas de calidad, según las reglas definidas en el proyecto. Cada vez que se añaden cambios a una incorporación de cambios o a la rama principal, se ejecuta una acción que analiza el código, ejecuta los test relacionados para obtener su cobertura y genera un informe de calidad. Este informe se puede consultar en la página de SonarCloud del proyecto y también se comenta en la subsección dedicada a SonarQube.
- **CodeQL:** GitHub CodeQL es una herramienta de análisis de código estático que permite detectar vulnerabilidades y errores en el código fuente. Cuando detecta un error suguiere una posible solución mediante Copilot. Esta acción se ejecuta por cada incorporación de cambios al repositorio.
- **Dependabot:** Dependabot es una herramienta de GitHub que ayuda a mantener las dependencias del proyecto actualizadas y seguras, en este caso solo está configurada para que actualice las dependencias si se ha detectado una vulnerabilidad en alguna de ellas. Dependabot crea automáticamente una solicitud de incorporación de cambios para corregirlas, sugiriendo la versión que corrige esa vulnerabilidad.

Políticas de seguridad

En el repositorio se ha añadido un conjunto de reglas(ruleset), en el su función es asegurar que todos los cambios se realicen de manera controlada. En la que se ha definido una restricción para que no se pueda realizar un borrado de código no autorizado, y que todos los cambios realizados en el código se realicen mediante una solicitud de incorporación de cambios(pull request), en la que automáticamente realiza una solicitud de revisión de código a copilot y se ejecutan las GitHub Actions ya descritas.

También hay una opción de poder reportar fallos de seguridad en el apartado de *security* si se han detectado en el código para que se puedan corregir lo más rápidamente posible. Los escaneos de seguridad que se realizan sobre el código del repositorio principalmente se enfocan en la detección de paquetes vulnerables, vulnerabilidades en código y de secretos (credenciales y claves de acceso que no deberían estar en el código fuente). Cabe destacar que el repositorio tiene en la carpeta raíz un archivo .env que contiene credenciales de acceso a la aplicación y este archivo no se debería subir al control de versiones. Sin embargo, en este caso se ha subido para que el código funcione correctamente, y que sea más fácil para un tercero ejecutarlo sin problemas, pero en un entorno de producción deberían estar como secretos de GitHub o solo disponer el archivo en el entorno local. Siendo altamente recomendable cambiar las credenciales cuando se vaya a desplegar el código en producción. Es importante destacar que el archivo .env no contiene el secreto de la clave de la API de Groq, para evitar que se pueda utilizar la API de Groq sin autorización, ya que esta clave es personal y no debería ser compartida públicamente. En el caso de que se quiera utilizar la funcionalidad del chat de ayuda de la web se debe acceder a la página web publicada en la nube accesible en `https://www.cesarrv.com`.

SonarQube

SonarQube es una herramienta de análisis de código estático que permite detectar errores, vulnerabilidades y problemas de calidad en el código fuente y proporcionar métricas de calidad de software. En el proyecto de SonarQube se pueden ver cómo se ha evolucionado en los problemas de calidad de código, en la cobertura y en las duplicaciones de código. Proporciona una forma muy visual de ver la calidad del código mediante insignias(badges) incluidas en el README del repositorio.

SonarQube trabaja con un conjunto de reglas que se deben cumplir para asegurar un mínimo de calidad de código del proyecto definida por el equipo de desarrollo. Estas reglas se pueden personalizar según las necesidades del proyecto, y se pueden añadir nuevas reglas o modificar las existentes. En este proyecto se han utilizado las reglas por defecto de SonarQube, ya que son suficientes para garantizar una buena calidad del código y principalmente porque son las únicas disponibles en el plan gratuito.

En la **Figura D.1** se pueden ver algunas de las gráficas que proporciona SonarQube, que en este caso, muestran la evolución de la calidad del código, con las métricas de cobertura de pruebas(primera gráfica) y los issues detectados sin resolver(segundo gráfico).

Figura D.1: Gráficas de SonarQube del proyecto

D.4. Compilación, instalación y ejecución del proyecto

Para la compilación e instalación del proyecto se puede hacer de forma general mediante Docker, ya que se ha creado un archivo *docker-compose.yml* que permite levantar todos los contenedores necesarios para el correcto funcionamiento del proyecto. Para ello, se debe tener instalado Docker y Docker Compose en el sistema.

En el caso de que se quiera ejecutar la parte del agente de aprendizaje por refuerzo, se deben seguir los mismos pasos descritos en la documentación de usuario en el apartado de instalación, que resume la instalación de las dependencias del proyecto en un entorno virtual, y un IDE asociado como Visual Studio Code [15] o Jupyter-Notebook.

Si se quisiese compilar la parte de la web se recomienda el uso del IDE IntelliJ IDEA [14], en el que requiere tener instalado una versión de Java 21 [3] y una versión compatible de Maven [6]. Para instalar las dependencias de Java se debe usar el siguiente comando:

```
mvn clean install -f app/pom.xml
```

Con esto ya se podrá editar y compilar la aplicación web. Para la ejecución del mismo se recomienda usar las herramientas que proporciona el IDE usado, o Docker.

Para la ejecución de pruebas se recomienda usar la GitHub Action asociada para evitar tener que instalar más dependencias. Si se quisiesen ejecutar en local, según el lenguaje se debería:

- **Python** Ejecutar el comando pytest, en el que se pase como argumento la carpeta que contiene el código python a probar.
- **Java** ejecutar el comando:

  ```
  mvn clean verify -f app/pom.xml
  ```

  Donde maven se encarga de ejecutar las pruebas disponibles.
- **Pruebas de Selenium** instalar la extensión web(o la configuración de las pruebas en el proyecto web) e importar el archivo WebTests.side ubicado dentro de la carpeta de tests de Spring.
- **JavaScript** Se necesita tener instalado con yarn el paquete de Jest. y ejecutarlo con yarn jest.

D.5. Pruebas del sistema

Los distintos tipos de pruebas que se han realizado en el proyecto son:

- **Pruebas unitarias:** Se han realizado pruebas unitarias tanto en el código de Java, Python y JavaScript, utilizando JUnit 5, Pytest y Jest respectivamente. Todas estas pruebas se ejecutan automáticamente cada vez que se realiza un push al repositorio, mediante las GitHub Actions descritas anteriormente, que envían los resultados a Sonar, pudiendo consultar la cobertura de estas en la página de SonarQube. En el caso de las pruebas unitarias de Java, se han realizado pruebas de integración con la base de datos, utilizando H2 como base de datos en memoria para evitar tener que realizar una conexión a una base de datos real. En el caso del agente de *reinforcement learning*, se han realizado pruebas unitarias para comprobar que los componentes del agente y entorno funcionan correctamente.
- **Pruebas funcionales:** Se han realizado pruebas funcionales a través de Selenium, que permite, a traves de Selenium IDE, grabar pruebas simulando la interacción del usuario con la web. Gracias a la función

de exportación de Selenium IDE las pruebas se pueden ejecutar tanto cargando el archivo WebTests.side, como ejecutando las pruebas de Java de la carpeta app/src/test/java/es/cesar/app/web (una vez ya iniciada la web) con un perfil de Spring Boot distinto a 'coverage', aceptando el perfil nativo. Se recomienda cargar el archivo WebTests.side en la extensión de Selenium IDE del navegador, ya que es más fácil de ejecutar y permite ver los resultados de las pruebas de forma más visual.

El objetivo de las pruebas es garantizar que la aplicación funciona correctamente y cumple con los requisitos funcionales y no funcionales definidos en la especificación del proyecto. La integración continua de estas pruebas permite detectar y corregir errores cada vez que si incorporan cambios al código, lo que mejora la calidad del software y reduce el riesgo de introducir errores en el sistema. Una alta cobertura de pruebas unitarias, como es en este caso, (superior al 90 %) proporciona una garantía de calidad, significando que la mayoría del código se prueba de forma automática, facilitando la evolución del software.

Apéndice E

Documentación de usuario

E.1. Introducción

La documentación de usuario es una parte fundamental en todo desarrollo de software, ya que permite a los usuarios comprender cómo utilizar el sistema, sus funcionalidades y características. En este caso, este apartado va a abordar los requisitos de usuario, la instalación del proyecto y el manual del usuario, proporcionando una guía para la correcta utilización del sistema desarrollado.

E.2. Requisitos de usuarios

Para la ejecución de la parte web del proyecto sólo es necesario tener instalado Docker y Docker Compose, ya que se ejecuta en contenedores de Docker. Para la instalación de Docker y Docker Compose se puede consultar la sección de Instalación. Para facilitar la descarga de los archivos fuente necesarios del proyecto, se recomienda instalar Git para poder clonar el repositorio de GitHub automáticamente. Simplemente se debe abrir una terminal en la carpeta donde se quiere descargar el proyecto y ejecutar git clone y la URL del repositorio. En este caso sería: **git clone https://github.com/CesarRodrigu/GII-24.19-contramedidas-IoT-mediante-reinforcement-learning.git**

En cualquier caso, si no se quiere instalar Git, se puede descomprimir la carpeta comprimida disponible para descargar de GitHub.

Se recomienda usar Python 3.12, y las versiones exactas de las dependencias ubicadas en el archivo de requerimientos, aunque probablemente en pueda funcionar en otras versiones.

Por lo que el usuario sólo debe tener un dispositivo compatible con Docker y Docker Compose (solo si se quiere ejecutar en local la web) y un navegador web moderno y un equipo compatible con Python, en todos los casos. También necesitará un entorno como Visual Studio Code o Jupyter-Notebook para poder ejecutar el notebook asociado al aprendizaje por refuerzo.

E.3. Instalación

Para la instalación del proyecto, se debe tener instalado Docker y Docker Compose. Para la instalación de Docker y Docker Compose, se recomienda la instalación de Docker Desktop, que para ello se puede consultar la documentación oficial de Docker. Una vez instalado, se debe abrir una terminal en la carpeta donde se ha descargado el proyecto y ejecutar el siguiente comando:

```
docker compose up -d
```

Con este simple comando se descarga, instala, compila y ejecuta todo el proyecto sin necesidad de realizar ninguna otra acción. Una vez ejecutado el comando y pasadas todas las tareas de Docker, se puede acceder a la aplicación web desde el navegador en la dirección **http://localhost:8081**. Cabe destacar que en la primera ejecución, Docker necesitará mucho tiempo para descargar e instalar todos los recursos necesarios, pero en las siguientes ejecuciones será mucho más rápido. Las dependencias de Bootstrap 5 y plotly.js no hace falta instalarlas, ya que la web cuenta con los archivos de Bootstrap necesarios y plotly.js se importa automáticamente desde la red.

Para la parte de aprendizaje por refuerzo, se necesita tener instalado Python [16] y las dependencias del archivo requirements.txt ubicado en la carpeta RL. Para esto, se deben ejecutar los siguientes comandos:

```
python -m venv venv
```

Para crear un entorno virtual donde tener los paquetes instalados. Una vez creado, se debe activar el entorno, en este caso, el comando es para Windows con cmd:

```
venv\Scripts\activate
```

Para activarlo en linux se puede hacer con el comando:

```
source venv/bin/activate
```

Por último para instalar todas las dependencias se debe de usar:

```
pip install -r RL/requirements.txt
```

Cabe destacar que para la instalación de StableBaselines3 los paquetes de dependencias cambian dependiendo del sistema operativo, en este caso se han recogido en entorno Windows [18].

E.4. Manual del usuario

El manual del usuario es un elemento importante para que los usuarios que no tienen conocimientos sobre el sistema puedan utilizarlo sin problemas en poco tiempo, por eso se han creado distintos elementos para facilitar al usuario la interacción con el sistema.

Para facilitar la presentación de la web, esta cuenta con un usuario por defecto con permisos de administrador, para poder comprobar la funcionalidad completa de la web.

El manual de usuario que recorre la mayoría de la funcionalidad de la web se encuentra en el repositorio de GitHub en la carpeta `docs/video/web.mp4`. Este video también se encuentra disponible en la web, en el apartado de ayuda(que se encuentra n el círculo abajo a la izquierda de la pantalla) en el que aparte del vídeo hay un chat en el que se puede hacer preguntas sobre el proyecto y un bot las responde. Para más documentación, se recomienda al usuario acceder al apartado *Sobre el proyecto* en el que se encontrará pdfs con documentación más concreta y extensa.

Para el entrenamiento de modelos se ha realizado otro vídeo disponible en la carpeta `docs/video/RL.mp4` donde se muestra cómo conseguir entrenar y visualizar un modelo entrenado. Si se quiere una mayor personalización del modelo se deben ajustar los parámetros de la función de recompensa, para eso, se puede modificar en la función reward en el archivo `RL/custom_env/router_env.py`.

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Durante el desarrollo del Trabajo de Fin de Grado, se han podido incluir diferentes aspectos relacionados con el desarrollo sostenible, no limitándose únicamente a los aspectos medioambientales, sino también a los sociales y económicos. La sostenibilidad ha sido un aspecto transversal en el proyecto, presente desde el diseño hasta la implementación del proyecto, considerando tanto el impacto inmediato como el impacto a largo plazo de las decisiones tomadas.

En primer lugar, se ha tenido en cuenta el impacto medioambiental de las tecnologías utilizadas, como el uso de contenedores Docker para la ejecución del proyecto. Esta tecnología permite una mayor eficiencia en el uso de recursos, incluyendo una mayor capa de seguridad y facilitando la escalabilidad de la aplicación, lo que contribuye a una reducción del consumo energético y de utilización de recursos. También se han utilizado servicios de terceros en la nube, comprometidos con la sostenibilidad, como es el caso de GitHub [13], Cloudflare [10] y AWS [7].

F.2. Uso sostenible de recursos

Uno de los aspectos más relevantes del proyecto es el uso eficiente de los recursos y la optimización de los mismos. Para ello, se han implementado

diversas estrategias que permiten reducir el consumo de recursos y minimizar el impacto ambiental.

En primer lugar, se ha optado por el uso de Cloudflare como proveedor de servicios de DNS y CDN, lo que permite mejorar las velocidades de carga, disminuyendo el ancho de banda y consultas al servidor de origen, ahorrando tiempo de cómputo. Cloudflare también se compromete con la sostenibilidad y ha implementado diversas iniciativas para reducir su huella de carbono, como el uso de energías renovables en sus centros de datos [10]. Cloudflare proporciona un informe sobre el impacto de carbono de los servicios usados, aunque no se ha podido obtener el informe para este proyecto en concreto, ya que no se ha alcanzado el umbral mínimo de uso de recursos para que se genere dicho informe. Aunque una de las métricas que sí se pueden obtener es el número del ancho de banda ahorrado, que en este caso se muestra en la **Figura F.1**.

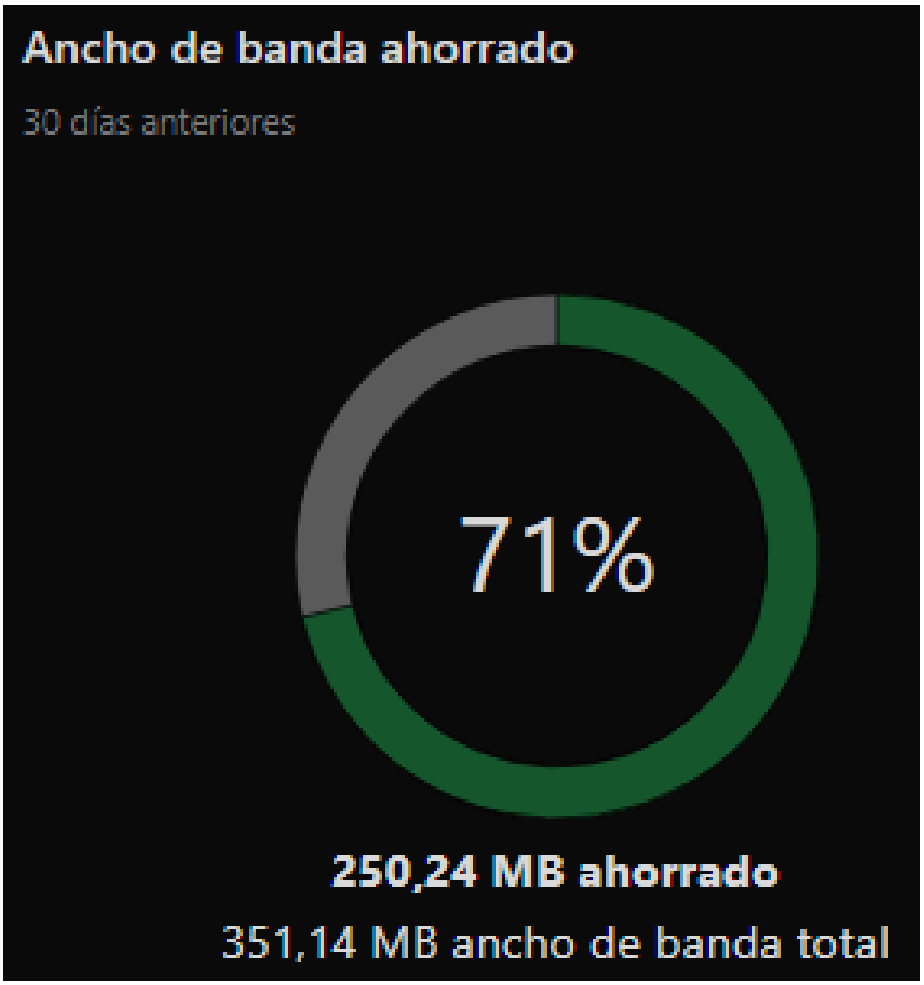

Figura F.1: Ancho de banda ahorrado por el uso Cloudflare

En el que se puede observar en la **Figura F.1** que se ha ahorrado un porcentaje considerable, que se traduce en un menor tiempo de cómputo, que deriva en una reducción del consumo energético, del impacto ambiental y reducciones de costes.

Además, se ha elegido un diseño de aplicación modular y escalable, que permite una fácil adaptación a futuros cambios y mejoras, así como una mejor gestión de los recursos individuales de cada módulo.

F.3. Uso de tecnologías de código abierto

El proyecto ha utilizado en su mayoría tecnologías de código abierto, que permiten una mayor sostenibilidad del proyecto, ya que promueven la transparencia, la colaboración, la auditabilidad del código y el acceso equitativo a la tecnología. [19]. Además, se permite contribuir al proyecto de forma abierta, facilitando la mejora continua del mismo y la posibilidad de mejorar el soporte a la comunidad, ya que se pueden realizar auditorías de seguridad y mejoras en el código por parte de la comunidad.

F.4. Consideraciones sobre la sostenibilidad social y económica

En los aspectos sociales y económicos, se ha tenido en cuenta la importancia de la accesibilidad y la inclusión en el desarrollo del proyecto. Por ejemplo, en la parte de la web se ha implementado un sistema de localización que permite que la aplicación sea accesible a un público más amplio, en el que de momento está disponible en español e inglés, pero se puede ampliar a otras regiones fácilmente. Además, se ha añadido la opción de cambiar el tema de la aplicación, para que sea más accesible a las personas, en el que se puede cambiar entre un tema claro y oscuro, para adaptarse a las preferencias del usuario y mejorar la experiencia de uso.

F.5. Conclusiones

En resumen, el proyecto ha tenido en cuenta diversos aspectos relacionados con la sostenibilidad, tanto medioambientales como sociales y económicos. Se han implementado diversas estrategias para reducir el impacto ambiental, optimizar el uso de recursos y promover la accesibilidad y la inclusión, como son el uso de tecnologías de código abierto y el uso de servicios de terceros comprometidos con la sostenibilidad.

Bibliografía

[1] Aepd. 2.- reglamento general de protecciÓn de datos (rgpd) | aepd.

[2] Julia Martins. ¿qué es la metodología kanban y cómo funciona? [2025] • asana, 1 2025.

[3] Oracle. Java downloads | oracle españa.

[4] Pdatos. ¿qué es la ley de protección de datos o lopdgdd? - pdatos.

[5] talent. junior developer salario en españa—salario medio.

[6] Apache Team. Installation – maven.

[7] AWS Team. Computación en la nube sostenible | amazon web services.

[8] AWS Team. Protección de datos y privacidad | aws.

[9] Cloudflare Team. Calculadora de precios de aws.

[10] CloudFlare Team. Diversidad, equidad e inclusión en cloudflare | cloudflare.

[11] CloudFlare Team. Privacidad y protección de datos | cloudflare.

[12] GitHub Team. Declaración de privacidad general de github - documentación de github.

[13] GitHub Team. Github social impact | the social impact team uses github's product, brand and employees to empower nonprofits and the greater social sector to make a positive & lasting contribution to the world.

[14] Jetbrains Team. Intellij idea – the ide for pro java and kotlin development.

[15] Microsoft Team. Visual studio code - code editing. redefined.

[16] Python Team. Download python | python.org.

[17] Spring Security Team. Features :: Spring security.

[18] StableBaselines3 Team. Installation — stable baselines3 2.6.1a1 documentation.

[19] Joan Francesc Canovas Tomàs. La sostenibilidad en sistemas open source - tecnología++.