Convocatoria Colaboración Público-Privada CPP2021-009025

DICYME:

Dynamic Industrial CYberrisk Modelling based on Evidence

Iniciativa Conjunta de:

DENEXUS

Universidad Rey Juan Carlos

ENTREGABLE 3.4:

Documentación de los requisitos, casos de uso, arquitectura del sistema y técnicas de visualización y soporte a la toma de decisiones propuestas

Coordinadores:

Romy R. Ravines (DeNexus Tech)
Isaac Martín de Diego (Universidad Rey Juan Carlos)
Alberto Fernández Isabel (Universidad Rey Juan Carlos)

MINISTERIO DE CIENCIA E INNOVACIÓN

Financiado por la Unión Europea NextGenerationEU

Plan de Recuperación, Transformación y Resiliencia

AGENCIA ESTATAL DE INVESTIGACIÓN

Contenido

1 INTRODUCCIÓN Y OBJETIVOS

Este documento recoge la documentación técnica y funcional del sistema desarrollada hasta la versión v1.0.0, disponible en el repositorio del proyecto (*deriskGroup / DICYME Project · GitLab*) y detallado en el entregable E3.3. El contenido se centra en la definición estructurada de los requisitos, los casos de uso, la arquitectura de software y las técnicas aplicadas de visualización y soporte a la toma de decisiones.

Se abordan los distintos aspectos del sistema desde una perspectiva de ingeniería del software, incluyendo tanto elementos de análisis como de diseño e implementación. El sistema se apoya en los datos proporcionados por los módulos automáticos de extracción desarrollados en la Actividad 1 y en los resultados generados por los módulos de medida y estimación de probabilidad e impacto definidos en la Actividad 2.

Sobre esa base, se ha consolidado un conjunto de funcionalidades que permiten integrar los procesos de recogida de datos, análisis del ciberriesgo y apoyo a la toma de decisiones a través de una interfaz visual interactiva.

En las siguientes secciones se explica:

- La especificación de requisitos funcionales y no funcionales.
- La descripción de los principales casos de uso y flujos operativos.
- Las visualizaciones integradas y su rol en el análisis de los datos y el soporte a decisiones.
- La arquitectura del sistema, sus módulos principales y las decisiones de diseño adoptadas.
- Las instrucciones para el despliegue y configuración en distintos entornos.

2 ANÁLISIS DE REQUISITOS

En esta sección se enumeran y detallan los distintos requisitos del sistema sobre los cuales se ha ido introduciendo la funcionalidad y la lógica al desarrollo del proyecto. Los requisitos funcionales (RF) permiten establecer qué acciones debe realizar el sistema y cómo debe comportarse para cumplir con los objetivos planteados por el usuario o cliente. A continuación, se presentan en detalle.

2.1 Visualización de ciber incidentes

- RF1.1 - Filtros dinámicos: Se incluyen una serie de filtros (rangos de fechas, unidades de tiempo de agrupamiento y fuentes de datos) con los que refinar la visualización de los incidentes

- RF1.2 - Visualización línea temporal: La aplicación incluye la visualización de los incidentes filtrados a partir de una gráfica de línea temporal

- RF1.3 - Visualización diagrama de barras: Se incluye un diagrama de barras en función del tipo de variable seleccionada.

- RF1.4 - Representación tabla incidentes: La aplicación incluye en formato tabla los datos de los incidentes, así como todas sus observaciones y variables.

- RF1.5 - Pestaña de información: La aplicación incluye una sección con información relativa sobre los datos de incidentes

2.2 Visualización de datos de perfil de víctimas

- RF2.1 - Filtros dinámicos: Se incluyen una serie de filtros (rango de fechas y tipo de evento reportado) con el que se pueden refinar las muestras de las víctimas.
- - RF2.2 - Visualización mapa: Se incluye un mapa interactivo que muestra, por país, la cantidad de eventos reportados asociados a víctimas, así como el número total de eventos distintos registrados en cada país.
- - RF2.3 - Visualización diagrama de barras: Se incluye un diagrama de barras que muestra, por país, la cantidad de eventos registrados.
- - RF2.4 - Visualización línea temporal: Se incluye un diagrama de línea temporal que muestra por tipo de evento el número de eventos producidos a lo largo del tiempo.
- - RF2.5 - Representación tabla víctimas: La aplicación incluye en formato tabla los datos de los eventos reportados a víctimas, así como todas sus observaciones y variables.
- - RF2.6 - Pestaña de información: La aplicación incluye una sección con información relativa sobre los datos de perfil de víctimas.

2.3 Visualización de los datos IDS

- RF3.1 - Filtros dinámicos: Se incluyen una serie de filtros (tipo de instalación, tipo de alcance, indicador y los distintos niveles del indicador seleccionado) que permiten refinar las muestras de IDS
- RF3.2 - Visualización de métricas clave en tarjetas: La aplicación incluye una sección en la que se presentan distintos valores de métricas informativas. Muestran datos como fuentes de datos, número de instalaciones, número de redes y número de activos cibernéticos
- RF3.3 - Visualización de línea temporal: Se incluye una gráfica que representa mediante una serie temporal los valores de los indicadores seleccionados
- RF2.6 - Pestaña de información: La aplicación incluye una sección con información relativa sobre los datos de *Intrusion Detection Systems*, como son los nombres de los distintos indicadores.

2.4 Indicador de actores principales

- RF4.1 - Pestaña de resumen: La aplicación incluye una pestaña que ofrece un resumen con distintos gráficos y métricas sobre los datos de actores principales.
    - RF4.1.1 - Visualización de información básica en tarjetas: La aplicación incluye una sección en la que se presentan distinta información básica

acerca de los actores. Muestran datos como las bases de datos usadas para extraer los datos, la fecha de extracción y la cantidad de actores principales del *dataset*.
    - RF4.1.2 - Visualización diagrama de barras: Se incluyen una serie de diagrama que muestran el número de actores en función de una serie de variables:
        - Cantidad de actores en función de la categoría de actividad
        - Cantidad de actores en función de sus objetivos
        - Cantidad de actores en función de las regiones objetivo
        - Cantidad de actores en función de las industrias objetivo
    - RF4.1.3 - Visualización leyenda de NAICS e ISO: La aplicación incluye una leyenda que permite mostrar los distintos códigos NAICS con su respectiva categoría de industria y el código ISO junto con su respectivo país.
- RF4.2 - Pestaña de detalles: Se incluye una pestaña que ofrece detalles, métricas y gráficas acerca de la información de actores principales.
    - RF4.2.1 - Filtros dinámicos: Se incluyen una serie de filtros que permite filtrar la información a partir de la selección de actores principales, los países y las industrias objetivos.
    - RF4.2.2 - Visualización de los detalles en tarjetas: Se incluye una sección que incorpora información detallada del actor principal seleccionado en los filtros, como puede ser su descripción, el indicador calculado, la categoría de actividad, la puntuación de actividad, puntuación de capacidad, fecha de extracción, fecha de la primera actividad, fecha de última actividad, alias del actor, objetivos, países e industrias objetivos.
    - RF4.2.3 - Visualización mapa de calor: Se incluye un gráfico de mapa de calor que incluye distintos valores en función de las industrias y países objetivo seleccionados en los filtros. El valor de cada celda representa el valor del indicador de actor calculado para esa combinación país-industria.
- RF 4.3 - Pestaña de comparación de víctimas: La aplicación incluye una pestaña que permite comparar distintas víctimas del conjunto de datos cargado
    - RF4.3.1 - Filtros dinámicos: Se incluyen un filtro único de multiselección en el cual se pueden seleccionar múltiples entidades víctimas.
    - RF4.3.2 - Visualización histograma: Se incluye un histograma que representa la distribución de la puntuación del indicador de actores de amenaza para cada una de las entidades víctimas seleccionadas.
- RF 4.4 - Pestaña de víctimas potenciales: La aplicación incluye una pestaña que permite analizar y representar métricas informativas de víctimas potenciales a partir de la selección de datos característicos de la víctima.
    - RF4.4.1 - Filtros dinámicos: Se incluyen diversos filtros (país objetivo, industria objetivo y tipo de actividad del actor principal) que permite filtrar la información de las puntuaciones de actor principal
    - RF4.4.2 - Visualización métricas en cartas: Se incluye la representación en forma de tarjetas las distintas métricas calculadas en función de los filtros, como puede ser el número de actores principales, media del indicador de actor, valor medio del Q1, Q2, Q3 y Q4 del indicador de actor.

    - RF4.4.3 - Visualización diagrama de barras radial: Se incluye un diagrama de barras radial que muestra los 20 principales actores de amenaza según el indicador combinado.
    - RF4.4.4 - Visualización diagrama barras: Se incluye un gráfico de barras que permite representar el indicador de actor para cada uno de los distintos actores, ordenados de forma descendente.
- RF4.5 - Pestaña de información: La aplicación incluye una sección con información y explicaciones del indicador de actores principales.

2.5 Indicador de atractivo

- RF5.1 - Pestaña de comparación de víctimas: La aplicación incluye una pestaña que permite analizar hasta 3 víctimas en función del valor del indicador de atractivo
    - RF5.1.1 - Visualización atractivo en tarjetas: Se incluye información relativa al atractivo de las entidades seleccionadas en forma de tarjetas. Se muestra el valor de atractiva de cada una de las víctimas seleccionadas.
    - RF5.1.2 - Visualización gráfico de radar: Se muestra un gráfico de radas con todas las variables relevantes para el cálculo del atractivo como pueden ser el número de empleados, ganancias, ingresos, dispositivos, información crítica y reputación online.
    - RF5.1.3 - Visualización tabla: Se incluye una tabla que muestra la información de interés para cada entidad seleccionada, utilizada para calcular el valor de atractivo.
- RF5.2 - Pestaña de víctima potencial: La aplicación incluye una sección que permite calcular el valor de atractivo para una víctima potencial a partir del ingreso de información de la entidad.
    - RF5.2.1 - Filtros dinámicos: Se incluyen diversos filtros (país, categoría industria, ganancias, ingresos, número de empleados, cotización en bolsa, rentabilidad, reputación online, dispositivos visibles y fugas de información crítica) que permiten ser usados para calcular el valor de atractivo. El país y la categoría de la industria son campos obligatorios, mientras que el resto son campos opcionales donde en caso de no introducir un valor, se introduce un valor por defecto.
    - RF5.2.2 - Visualización del valor de atractivo: Se visualiza el valor de atractivo calculado a partir de los campos de los filtros introducidos.
    - RF5.2.3 - Visualización gráficos: Se incluyen diversos gráficos para cada uno de los campos de los filtros, donde para las variables numéricas se representa la información en un histograma, mientras que para las variables categóricas se representa la información en un diagrama de barras. Para mostrar estos diagramas se utiliza el conjunto de datos de perfil de víctimas.
- RF5.3 - Pestaña de información: La aplicación incluye una sección con información y explicaciones acerca del indicador de atractivo.

2.6 CVE2TTPs

- RF6.1 - Pestaña de estadísticas: La aplicación incluye una pestaña que contiene una serie de gráficos que representan las características del conjunto de datos de CVEs.
    - RF6.1.1 - Visualización mapa de calor empresas: Se incluye un mapa de calor que representa el número de CVEs en función del tipo de vulnerabilidad y táctica para empresas.
    - RF6.1.2 - Visualización mapa de calor empresas críticas: Se incluye un mapa de calor que representa el número de CVEs en función del tipo de vulnerabilidad y táctica para empresas críticas.
    - RF6.1.3 - Visualización diagrama barras años: Se incluye un diagrama de barras que reúne tanto el número de CVEs, técnicas utilizadas en empresas y empresas críticas en función del año.
    - RF6.1.4 - Visualización diagrama de barras CVSS: Se incluye un diagrama de barras que muestra el valor de puntuación CVSS en función de la táctica, tanto para empresas normales como para las críticas.
- RF6.2 - Pestaña de emparejamientos de CVEs: La aplicación incluye una pestaña que permite emparejar un CVE seleccionado con tácticas.
    - RF6.2.1 - Filtros dinámicos: Se incluyen diversos filtros (año del CVE y código del CVE) que permiten ser usados para obtener los emparejamientos para dicho CVE seleccionado.
    - RF6.2.2 - Visualización información CVE en tarjetas: Se incluye la representación de la información básica del CVE en una serie de tarjetas, como puede ser la descripción, el tipo o tipos, la puntuación del CVSS v2 y CVSS v3.
    - RF6.2.3 - Visualización de los emparejamientos: Se incluye una tabla que muestra las distintas tácticas asociadas al CVE seleccionado, donde también se muestra información como la técnica.
- RF6.3 - Pestaña de emparejamientos de técnicas: La aplicación incluye una pestaña que permite emparejar una técnica seleccionada con distintas vulnerabilidades.
    - RF6.3.1 - Filtros dinámicos: Se incluyen diversos filtros (matriz MITRE, táctica y técnica) que permiten ser usados para obtener los emparejamientos para dicha técnica seleccionada.
    - RF6.3.2 - Visualización información técnica en tarjetas: Se incluye la representación de la información básica de la técnica seleccionada en una serie de tarjetas, como puede ser el nombre y descripción.
    - RF6.3.3 - Visualización de los emparejamientos: Se incluye una tabla que muestra las distintas vulnerabilidades asociadas a la técnica seleccionada, donde adicionalmente se muestra información de cada vulnerabilidad obtenida.
- RF6.4 - Pestaña de información: La aplicación incluye una sección con información y explicaciones acerca del modelo CVE2TTPs.

2.7 Cyber Risk Quantification

- RF7.1 - Pestaña de selectores para la simulación: Se incluye una pestaña específica para incluir distintos selectores utilizados para la simulación, tanto para víctimas reales como potenciales. Los selectores comunes para los dos casos son: selector de archivo de CVEs, número de simulaciones, semilla, coste por hora forense, coste del equipamiento y valor de la vida estadística. En el caso de la vista de víctimas reales hay un selector adicional que es la entidad. Por el contrario, en el caso de víctimas potenciales se encuentran los siguientes selectores adicionales:
    - País
    - Categoría industria
    - Ganancias
    - Ingresos
    - Número de empleados
    - Cotización en bolsa
    - Rentabilidad
    - Reputación online
    - Dispositivos visibles
    - Fugas de información crítica
- RF7.2 - Pestaña de los resultados de la simulación: La aplicación incluye una pestaña mostrando los resultados obtenidos
    - RF7.2.1 - Visualización de histograma de eventos de pérdida: Se incluye un histograma que muestra los eventos de pérdida producidos.
    - RF7.2.2 - Visualización de la curva de excedencia anual: Se incluye una curva de excedencia de pérdida anualizada (LEC) donde se representa la exposición de pérdida en función de la probabilidad de que la pérdida supere cierto umbral
    - RF7.2.3 - Visualización del árbol de CRQ: Se incluye un árbol en el que se muestra en forma de nodos los valores de la simulación calculados, como lo son la pérdida primaria y secundaria, la frecuencia de eventos de amenaza, susceptibilidad, frecuencia de eventos de pérdida, magnitud de pérdida y el valor final de ciber riesgo.
    - RF7.2.4 - Visualización de los valores de los resultados: Se incluyen los valores de los resultados obtenidos, los mismos valores que los que se representan en el árbol de CRQ.
    - RF7.2.5 - Ajuste de los resultados: Se incluyen una serie de componentes que permiten ajustar los parámetros usados en la simulación, como es baseline, índice de incidentes, atractivo, índice de actor principal, puntuación de técnicas e índice de seguridad.
    - RF7.2.6 - Botones de control: Se incluyen una serie de botones que permiten realizar acciones sobre la simulación, como puede ser ejecutar de nuevo la simulación, resetear los valores de la simulación o descargar la simulación en un reporte PDF.

3 CASOS DE USO

Después de haber definido los requisitos funcionales de la aplicación, en este apartado se describe cómo interactúan los usuarios con el sistema para alcanzar sus objetivos. Para ello, se presentan una serie de diagramas de casos de uso (véanse la *Ilustración 1. Visualización del conjunto de datos de ciber incidentes*, la *Ilustración 2. Visualización del conjunto de datos de perfil de víctimas*, la *Ilustración 3. Visualización del conjunto de datos de IDS*, la *Ilustración 4. Indicador de actores de amenazas*, la *Ilustración 5. Indicador de atractivo*, la *Ilustración 6. CVE2TTPs* y la *Ilustración 7. Cyber Risk Quantification*) que permiten identificar las funcionalidades principales del sistema desde la vista del usuario. Estos diagramas proporcionan una visión general de las acciones que pueden realizar los usuarios, facilitando así la comprensión de los procesos que soporta el sistema. Los distintos diagramas que se presentan a continuación ilustran cómo se satisfacen uno o varios de los requisitos funcionales previamente presentados, mostrando de qué manera la interacción del usuario con la aplicación permite dar respuesta a dichos requirimientos.

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con la consulta de información de incidentes, la visualización de gráficos y el filtrado de datos. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF1.1, RF1.2, RF1.3, RF1.4 y RF1.5.

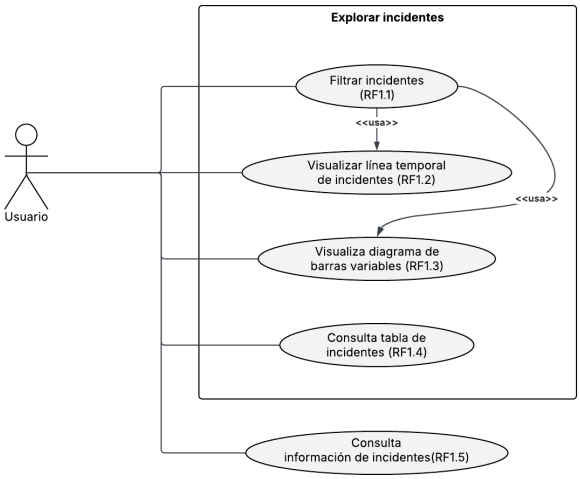

Ilustración 1. Visualización del conjunto de datos de ciber incidentes

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con la consulta de información de incidentes, la visualización de gráficos, tablas o mapas y el filtrado de datos. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF2.1, RF2.2, RF2.3, RF2.4, RF2.5 y RF2.6.

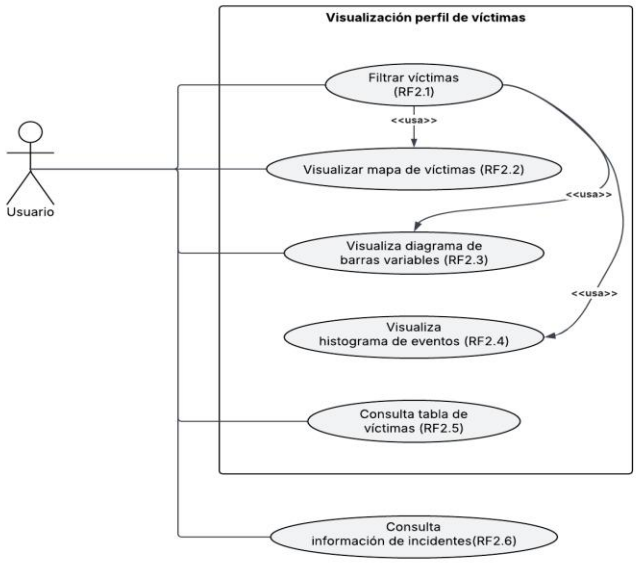

Ilustración 2. Visualización del conjunto de datos de perfil de víctimas

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con el filtrado de datos, visualización de métricas o gráficos y la consulta de información. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF3.1, RF3.2, RF3.3, RF3.4, RF3.5 y RF3.6.

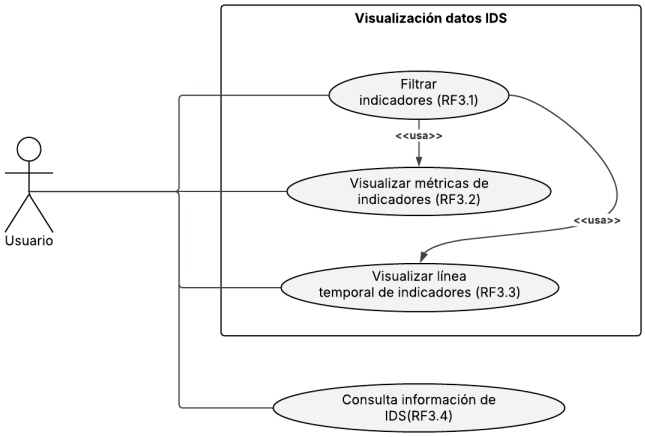

Ilustración 3. Visualización del conjunto de datos de IDS

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con el filtrado de datos, visualización de métricas, información o gráficos y la consulta de información de la pestaña de actores de amenaza. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF4.1.1, RF4.1.2, RF4.1.3, RF4.2.1, RF4.2.2, RF4.2.3, RF4.3.1, RF4.3.2, RF4.4.1, RF4.4.2, RF4.4.3, RF4.4.4 y RF4.5.

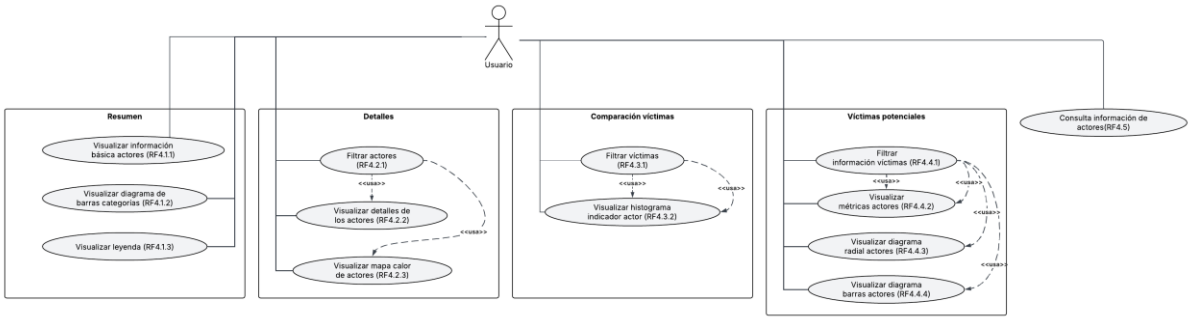

Ilustración 4. Indicador de actores de amenazas

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con el filtrado de datos, visualización de métricas, información, tablas o gráficos y la consulta de información de la pestaña de atractivo. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF5.1.1, RF5.1.2, RF5.1.3, RF5.2.1, RF5.2.2, RF5.2.3, RF5.3.

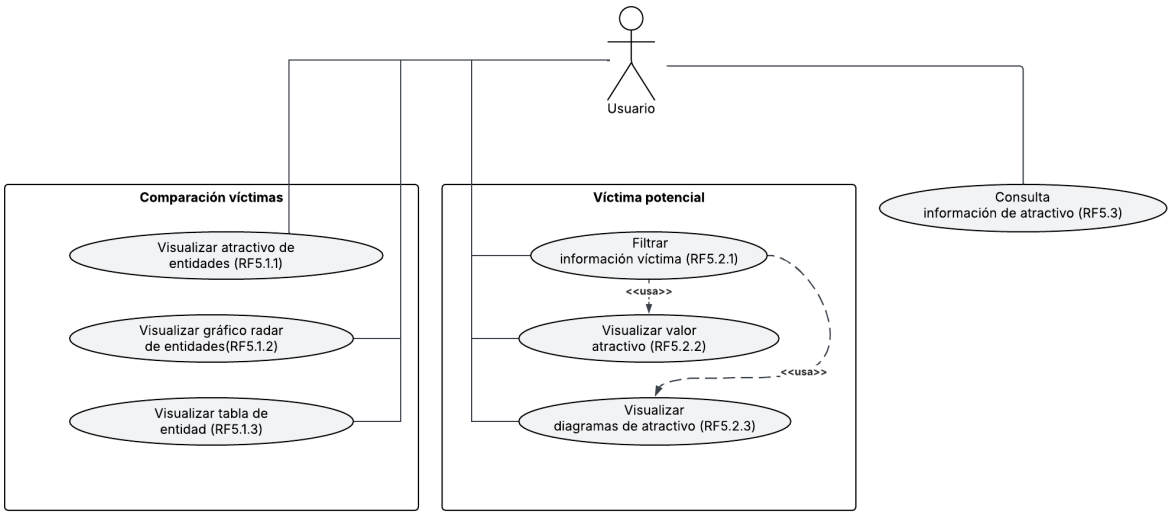

Ilustración 5. Indicador de atractivo

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con el filtrado de datos, visualización de métricas, información, tablas o gráficos y la consulta de información de la pestaña de CVE2TTPs. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF6.1.1, RF6.1.2, RF6.1.3, RF6.1.4, RF6.2.1, RF6.2.2, RF6.2.3, RF6.3.1, RF6.3.2, RF6.3.3 y RF6.4.

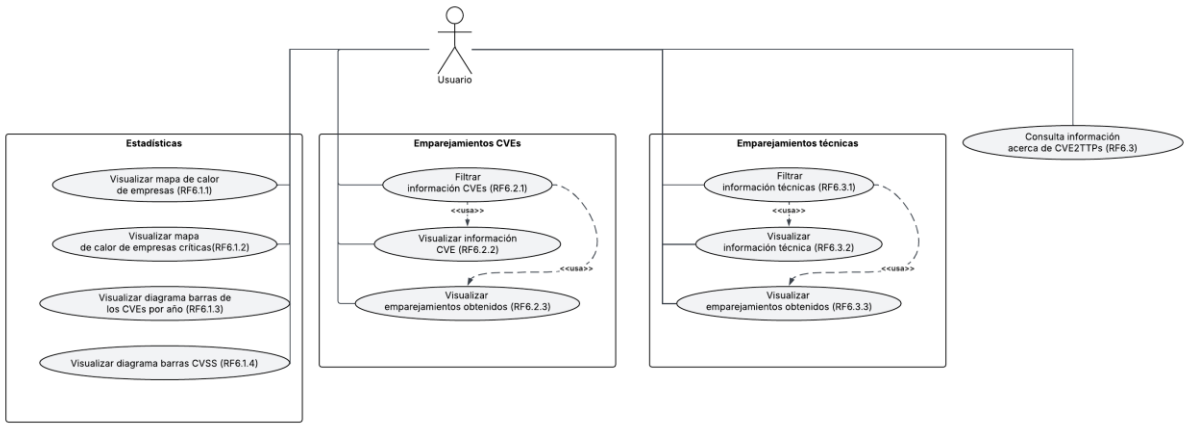

Ilustración 6. CVE2TTPs

El siguiente diagrama de caso de uso muestra la interacción del usuario con la aplicación para llevar a cabo determinadas acciones, en este caso relacionadas con el filtrado de datos, selección de datos, acciones con botones, consultas a agentes, visualización de métricas, información, tablas o gráficos y la consulta de información de la pestaña de modelos CRQ. Dichas interacciones permiten dar cumplimiento a los requisitos funcionales: RF7.1, RF7.2.1, RF7.2.2, RF7.2.3 RF7.2.4, RF7.2.5, RF7.2.6, RF7.2.7, RF7.2.8, RF7.3 y RF7.4.

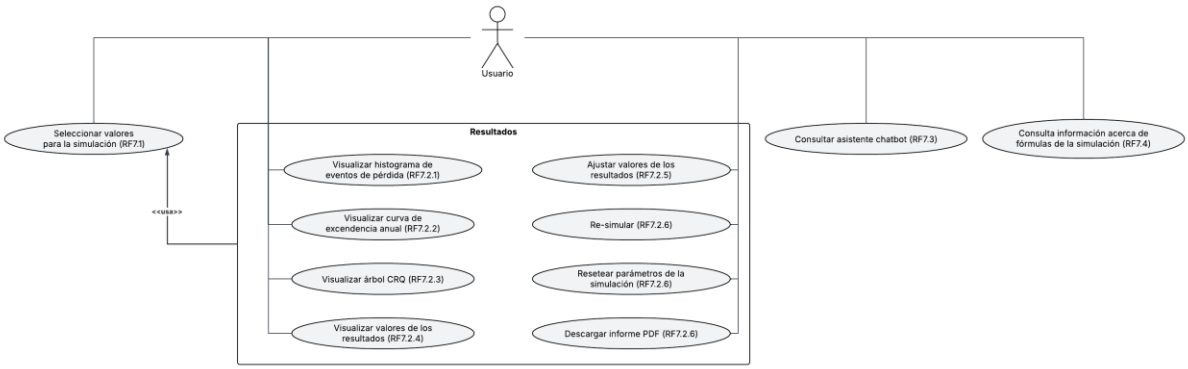

Ilustración 7. Cyber Risk Quantification

4 TÉCNICAS DE VISUALIZACIÓN Y SOPORTE A LA TOMA DE DECISIONES

En el desarrollo de la herramienta se implementaron diversas técnicas de visualización de datos con el objetivo de facilitar la interpretación de la información y mejorar la experiencia de usuario. La selección de los gráficos se basó en la naturaleza de los datos, su granularidad, y el tipo de análisis que se busca promover dentro del sistema. Se utilizó una distribución homogénea de elementos visuales para reforzar la usabilidad y evitar la sobrecarga cognitiva, asegurando una navegación fluida y una comprensión intuitiva de los resultados.

Para representar recuentos de variables categóricas o discretas se utilizaron gráficos de barras. Este tipo de visualización permite identificar rápidamente la frecuencia de

ocurrencia de diferentes categorías y comparar magnitudes de manera clara (véase la *Ilustración 8. Gráfico de barras para la cantidad de CVEs y técnicas predichas por el modelo CVE2TTPs por año*). En los casos en que se requirió representar proporciones relativas entre categorías, se optó por gráficos del 100%, facilitando la evaluación comparativa de la estructura interna de cada grupo en términos porcentuales.

La exploración de datos temporales se realizó mediante gráficos de series temporales, los cuales permiten detectar tendencias, estacionalidades y comportamientos anómalos a lo largo del tiempo. Estos gráficos fueron especialmente útiles en el análisis de evolución de indicadores y patrones de comportamiento dinámicos. Además, se emplearon tanto de series independientes como apiladas, que permiten ver las cantidades totales de los indicadores (por ejemplo, los incidentes a lo largo del tiempo).

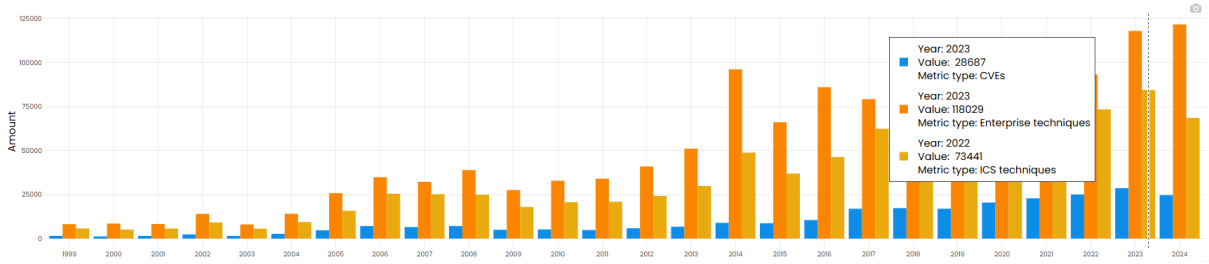

Ilustración 8. Gráfico de barras para la cantidad de CVEs y técnicas predichas por el modelo CVE2TTPs por año

También se incorporaron gráficos de densidad para analizar la distribución continua de variables numéricas, como el indicador de actores de amenazas. Esta técnica permitió observar la concentración de valores y detectar la presencia de multimodalidad o asimetrías, aspectos fundamentales en la caracterización de datos poblacionales o simulados. En este mismo apartado se introdujo un gráfico de barras circular que permite mostrar diversos aspectos del indicador: la altura de la barra muestra el valor final del indicador, mientras que el color denota la puntuación de actividad y un círculo en el eje vertical marca la puntuación del objetivo (*target*) (véase la *Ilustración 9. Gráfico de barras circular para el indicador de actores de amenaza*).

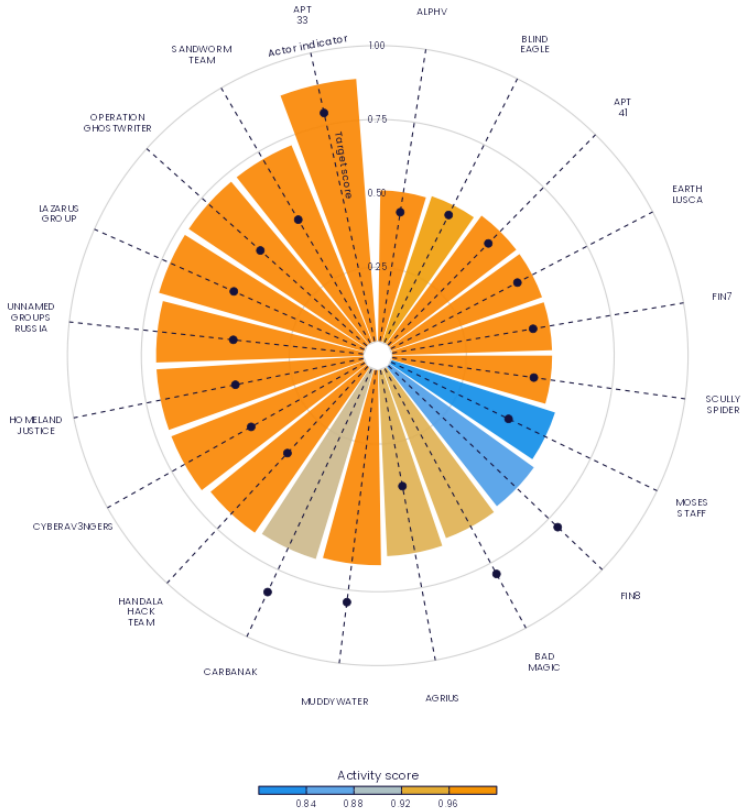

Ilustración 9. Gráfico de barras circular para el indicador de actores de amenaza

Complementariamente, para mostrar los resultados del modelo CVE2TTPs se diseñaron mapas de calor para representar matrices de datos, destacando mediante el uso del color la cantidad de relaciones observadas entre tácticas y tipos de vulnerabilidades, facilitando la identificación de patrones y relaciones clave. Asimismo, se integró un gráfico de pirámide que permite comparar los resultados para las matrices MITRE ATT&CK Enterprise e ICS (véase la *Ilustración 10. Gráfico de pirámide para comparar el CVSS medio por táctica en las matrices MITRE ATT&CK Enterprise e ICS*).

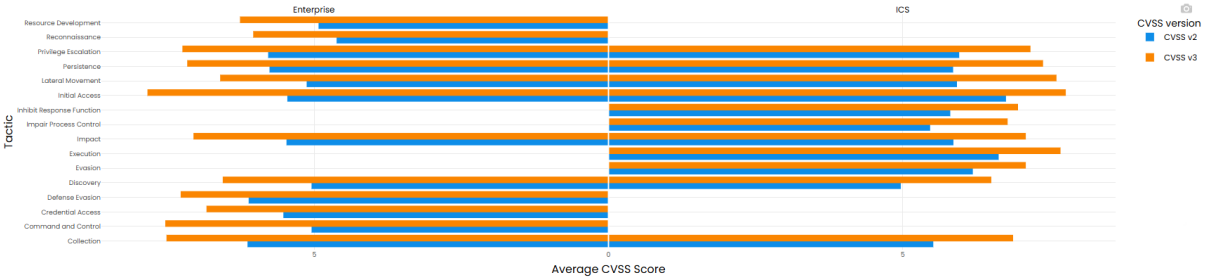

Ilustración 10. Gráfico de pirámide para comparar el CVSS medio por táctica en las matrices MITRE ATT&CK Enterprise e ICS

En contextos donde fue necesario mostrar la estructura jerárquica, como en la cuantificación del riesgo final, se integró la visualización mediante árboles, ya que sirven para modelar y visualizar la dinámica de la cuantificación y la combinación de los diferentes componentes para obtener una métrica final.

Adicionalmente, de manera transversal se incluyeron elementos visuales complementarios como los *value boxes*, los cuales presentan datos clave en formatos destacados para captar rápidamente la atención del usuario sobre métricas críticas o puntos de control. Esta estrategia visual ayuda a enfocar la interpretación en valores de interés sin necesidad de un análisis gráfico detallado.

Para enriquecer la experiencia del usuario, estas visualizaciones se desarrollaron como gráficos interactivos empleando librerías especializadas como Plotly, lo que permitió construir aplicaciones visuales ricas y adaptables a diferentes contextos analíticos (*Rich Interactive Application*, RIAs). El conjunto de estas técnicas permitió desarrollar una herramienta visualmente coherente, informativa y *responsive*, que se adapta a distintos tipos de análisis y necesidades de exploración, garantizando una interacción eficiente entre el usuario y los datos.

5 ARQUITECTURA DEL SISTEMA

Como se ha introducido en los entregables E3.1, E3.2 y E3.3, la arquitectura del sistema DICYME es modular, basada en microservicios. Esta modularidad se implementa mediante contenedores Docker, cada uno encapsulando un componente funcional independiente (la aplicación R Shiny, la API en Flask con Python, y el servidor proxy con Apache). En la ***Error! Reference source not found.*** se muestra cómo los distintos servicios se comunican entre sí a través de una red Docker interna, manteniendo una separación lógica y segura del resto de cargas de trabajo que se ejecutan en el servidor. Esta tipología aporta ventajas clave como la portabilidad del sistema, la escalabilidad horizontal (pudiendo replicar o distribuir contenedores según la carga) y la facilidad para el despliegue, mantenimiento de cada módulo de forma aislada y la incorporación de nuevas funcionalidades sin alterar los demás componentes. Además, se refuerza la seguridad mediante la gestión del tráfico externo a través de un contenedor proxy que canaliza y cifra las peticiones mediante HTTPS, empleando un certificado SSL facilitado por la URJC, ya que el servidor en el que se despliega y el dominio asociado pertenecen a esta entidad.

Por tanto, los usuarios tienen 2 vías de acceder al sistema: mediante HTTPS al puerto 3866, en el que se sirve la aplicación, o a través del puerto 3806, en cuyo caso el proxy redirecciona automáticamente las peticiones al puerto 3866 (HTTPS).

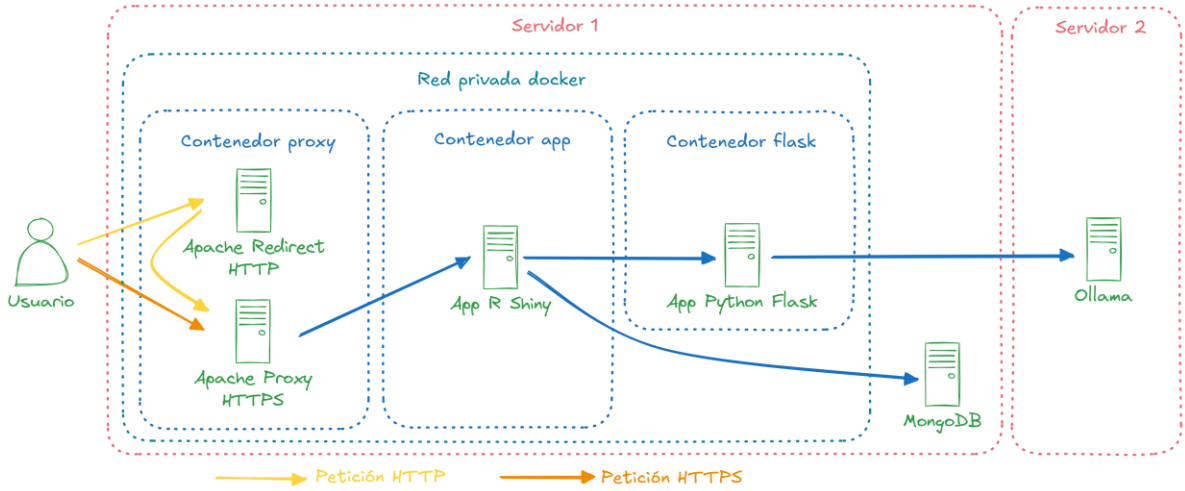

Ilustración 11. Diagrama de arquitectura de la aplicación y flujo de peticiones HTTP y HTTPS de un usuario

De manera independiente al grupo de contenedores se encuentra una instancia de Ollama en otro servidor, al cual se conecta la API en Flask para obtener las respuestas de los modelos LLM. Asimismo, está la base de datos MongoDB que proporciona almacenamiento persistente de los conjuntos de datos empleados en la aplicación.

Todo el proceso se encuentra securizado, a través de diferentes medidas:

- MongoDB tiene bloqueado el acceso en el firewall desde fuera de la red VPN de la URJC y las subredes de los despachos del equipo de la universidad [1]. También está restringido desde otras redes de Docker que no son las del sistema, que es exclusiva para este.
- Ollama, desplegado en otro servidor específico para este propósito, únicamente puede accederse desde el servidor en que se despliega la app (servidor 1). De esta manera, el sistema tiene comunicación directa. Para el desarrollo en local del sistema, el equipo levanta un puente SSH que crea la vía de comunicación con dicho servidor.
- El acceso HTTP está restringido, forzando HTTPS en todas las peticiones HTTPS.
- Todas las variables para la comunicación entre sistemas y autenticación (dirección de los servidores, credenciales de la base de datos, certificado SSL y clave privada del mismo, etc.) se configuran por medio de secretos de GitHub Actions, lo que garantiza que se excluyen del código y ni siquiera aparecen reflejados en los registros (*logs*) de los despliegues.
- El despliegue está automatizado con GitHub Actions, lo que asegura una entrega controlada y trazable, reduciendo errores humanos y tiempos de inactividad.
- Los *workflows* de GitHub Actions de terceros que se usan se especifican mediante hash en lugar de número de versión, denegando explícitamente todos

[1] El personal en teletrabajo y equipo de DeNexus puede acceder mediante VPN facilitada por la URJC.

salvo estos autorizados explícitamente en los ajustes de la organización de GitHub.
- *Dependabot* escanea semanalmente los workflows propios del despliegue para identificar dependencias actualizables, creando directamente una *Pull Request* que el equipo incorpora al proyecto tan pronto como es posible.

6 DESPLIEGUE DEL SISTEMA

El despliegue de la aplicación se puede hacer o bien mediante la automatización de los contenedores Docker, o bien levantando las aplicaciones de manera independiente. El despliegue automatizado maneja la gestión de las dependencias, librerías y demás configuración necesaria para el correcto funcionamiento del sistema completo e integrado.

En el primer caso, el *workflow* de GitHub Actions que orquesta el proceso de despliegue es `.github/workflows/deploy.yml`. Este realiza las siguientes etapas:

1. Validar que se han definido como secretos de GitHub Actions todas las variables definidas (`DB_HOST`, `DB_USERNAME`, `DB_PASSWORD`, `SSL_CERT`, `SSL_KEY`, `SERVER_USER`, `LLM_HOST`, `LLM_MODEL_NAME`, `LLM_USE_SSH_TUNNEL`, `LLM_SSH_PORT`, `LLM_SSH_USER`, `LLM_SSH_PASSWORD`, `LLM_REMOTE_BIND_IP`, `LLM_REMOTE_BIND_PORT`, `LLM_LOCAL_PORT` y `LLM_PORT`).
2. Con dichos secretos, crea el archivo `.Renviron` y `api_llm/config/.env`, incluyendo en cada uno las variables necesarias para el funcionamiento del contenedor de la aplicación Shiny en R y el de la API Flask en Python.
3. Crea el certificado SSL y su clave privada en `ssl/server.crt` y `ssl/server.key`, empleados por el contenedor de proxy HTTP.
4. Detiene los contenedores de Docker que estuviesen en ejecución (las versiones anteriores del sistema, si las hubiera), eliminando las imágenes y volúmenes asociados, y crea y despliega la nueva versión del sistema empleando Docker Compose, que define los tres contenedores, su red interna para comunicarse, los puertos del contenedor de proxy que se exponen al exterior, y el comportamiento en caso de reinicio del servidor *host*.
   La creación y ejecución de cada contenedor se define en los archivos `Dockerfile.app`, `Dockerfile.flask` y `Dockerfile.proxy`, que paso a paso instalan las librerías y dependencias necesarias para ejecutar cada uno de los tres contenedores que constituyen el sistema.

En la segunda opción, es preciso llevar a cabo el despliegue paso a paso. Para ello, hay que realizar lo siguiente:

- Aplicación Shiny:
  - Instalar la versión de R 4.4.1. Para otras versiones el sistema también puede funcionar, pero no se ha comprobado explícitamente.

    - Restaurar las dependencias del fichero renv.lock con el comando `renv::restore()`.
    - Instalar PhantomJS mediante `webshot::install_phantomjs()`.
    - Instalar una distribución de LaTeX como por ejemplo tinytex, ejecutando `quarto install tinytex`.
    - Crear el fichero .Renviron incluyendo las siguientes variables secretas:
        - MONGO_HOST: dirección IP o dominio de la base de datos.
        - MONGO_USER: usuario de la base de datos.
        - MONGO_PWD: contraseña de la base de datos.
        - FLASK_HOST: dirección IP o dominio de la API en Flask.
        - FLASK_PORT: puerto de la API en Flask.
    - Instalar el paquete R {dicymeviz}, que contiene el sistema: `devtools::install_local(".", build = TRUE, upgrade="never")`.
    - Ejecutar el paquete: `dicymeviz::app_run(.port = 1234, .host = '0.0.0.0')`.
- API en Flask:
    - Instalar Python 3.11. Para otras versiones el sistema también puede funcionar, pero no se ha comprobado explícitamente.
    - Instalar las dependencias especificadas en el archivo requirements.txt con el comando `pip install --no-cache-dir -r requirements.txt`.
    - Crear el archivo api_llm/config/.env con los siguientes secretos:
        - MONGO_HOST: dirección IP o dominio de la base de datos.
        - MONGO_USER: usuario de la base de datos.
        - MONGO_PWD: contraseña de la base de datos.
        - LLM_HOST: dirección IP o dominio de Ollama.
        - LLM_MODEL_NAME: nombre del modelo LLM en Ollama.
        - LLM_USE_SSH_TUNNEL: si usar túnel SSH o no (True/False).
        - LLM_PORT: si no se usa túnel SSH, el puerto en el que se sirve Ollama.
        - LLM_SSH_PORT: si se usa túnel SSH, el puerto al que realizar la conexión.

        - LLM_SSH_USER: si se usa túnel SSH, el usuario con el que realizar la conexión.
        - LLM_SSH_PKEY: si se usa túnel SSH, la ruta a la clave privada con la que realizar la conexión.
        - LLM_REMOTE_BIND_IP: si se usa túnel SSH, la IP remota a la que realizar la conexión.
        - LLM_REMOTE_BIND_PORT: si se usa túnel SSH, el puerto remoto al que realizar la conexión.
        - LLM_LOCAL_PORT: si se usa túnel SSH, el puerto local en el que se servirá el LLM.
    - Ejecutar la aplicación con el comando `python src/app.py`.
- Ollama instalado siguiendo el procedimiento apropiado, según su propia documentación oficial disponible en *https://github.com/ollama/ollama*. Las instrucciones de instalación en plataformas Linux son las siguientes: *https://github.com/ollama/ollama/blob/main/docs/linux.md*.
- MongoDB instalado siguiendo la documentación oficial dependiente de la plataforma específica, disponible en *https://www.mongodb.com/docs/manual/installation/*.
    - Los conjuntos de datos desarrollados en el proyecto y detallados en los entregables de la Actividad 1 (E1.1, E1.2 E1.3 y E1.4) y la Actividad 2 (E2.1, E2.2, E2.3 y E2.4). La aplicación usa la relación de bases de datos y colecciones detallada en la *Tabla 1. Relación de bases de datos y colecciones de MongoDB utilizadas por el sistema*..
- Comunicación cifrada mediante proxy web u otra solución según las condiciones y necesidades específicas del entorno.

Tabla 1. Relación de bases de datos y colecciones de MongoDB utilizadas por el sistema.

Base de datos	Colección
ActorScore	Actors
CVE2TTP	CVEMaster
IDS	Indicators
Incidentes	enrichedIncidents
	combinedIncidents

7 CONCLUSIONES

El sistema DICYME constituye una solución completa para la visualización, análisis y generación de descripciones automatizadas de conjuntos de datos relacionados con el riesgo cibernético. La integración de tecnologías estadísticas, de visualización interactiva y modelos LLM se ha abordado desde una perspectiva modular, orientada a la portabilidad, mantenibilidad y escalabilidad del sistema.

Desde el punto de vista funcional, DICYME permite al usuario explorar datos complejos de forma intuitiva a través de una interfaz web interactiva desarrollada con R Shiny. Esta interfaz se conecta de forma transparente con una API desarrollada en Python (Flask), que actúa como intermediaria entre la aplicación y el modelo de lenguaje alojado en un servidor independiente mediante Ollama. La incorporación de un modelo LLM permite generar descripciones textuales de variables, series temporales y patrones encontrados en los datos, facilitando la interpretación y reduciendo barreras técnicas para el análisis.

En cuanto a la arquitectura, se ha optado por un enfoque basado en microservicios desplegados mediante contenedores Docker, lo que garantiza una separación clara entre componentes, facilita el mantenimiento individual de cada servicio y mejora la reproducibilidad en distintos entornos. Esta arquitectura modular ha sido clave para asegurar la escalabilidad horizontal del sistema, permitiendo su adaptación a futuras cargas de trabajo sin reestructuración significativa.

La seguridad ha sido un eje central durante todo el desarrollo. La comunicación entre servicios está aislada mediante una red interna de Docker, y el tráfico externo está canalizado exclusivamente a través de un contenedor proxy que cifra las peticiones HTTPS mediante certificados SSL. Además, el acceso a los servicios externos, como la base de datos MongoDB y el servidor Ollama, está restringido mediante firewalls, redes VPN y/o túneles SSH, según el caso. Las credenciales y variables sensibles se gestionan a través de secretos de GitHub Actions, sin exponerse en código ni en logs.

En términos de despliegue, se ha implementado un pipeline CI/CD automatizado mediante GitHub Actions, lo que garantiza una entrega controlada, trazable y libre de errores manuales. El sistema puede desplegarse íntegramente mediante contenedores o de forma manual para entornos de desarrollo y depuración. En ambos casos, se ha documentado con precisión la configuración necesaria, tanto para la aplicación Shiny como para la API Flask, MongoDB y Ollama, asegurando así la reproducibilidad del sistema en nuevas instalaciones.

En conjunto, DICYME ejemplifica un diseño robusto, seguro y adaptable, aplicando buenas prácticas de desarrollo de software científico y de sistemas distribuidos. Su arquitectura modular, su enfoque en la seguridad y la automatización, y su capacidad de integrarse con modelos avanzados de lenguaje lo posicionan como una plataforma versátil para el análisis de microdatos en entornos académicos e institucionales. Además, establece una base tecnológica sólida para futuras extensiones, como la incorporación de nuevos tipos de visualizaciones, conectores de datos o modelos especializados por dominio.