Universidad de Cádiz
Escuela Superior de Ingeniería

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA

Privatización y Usabilidad de Datos Sintéticos: Un análisis práctico

AUTOR/A: Francisco Javier Molina Rojas

Puerto Real, junio de 2024

Universidad de Cádiz
Escuela Superior de Ingeniería

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA

Privatización y Usabilidad de Datos Sintéticos: Un análisis práctico

DIRECTOR: Juan Manuel Dodero Beardo
CODIRECTORA: Mercedes Rodríguez García
AUTOR/A: Francisco Javier Molina Rojas

Puerto Real, junio de 2024

Declaración personal de autoría

Francisco Javier Molina Rojas con DNI 45386606Q, estudiante del Grado en Ingeniería Informática en la Escuela Superior de Ingeniería de la Universidad de Cádiz, como autor de este documento académico titulado Privatización y Usabilidad de Datos Sintéticos: Un análisis práctico y presentado como Trabajo Final de Grado en Ingeniería Informática.

DECLARO QUE

Es un trabajo original, que no copio ni utilizo parte de obra alguna sin mencionar de forma clara y precisa su origen tanto en el cuerpo del texto como en su bibliografía y que no empleo datos de terceros sin la debida autorización, de acuerdo con la legislación vigente. Asimismo, declaro que soy plenamente consciente de que no respetar esta obligación podrá implicar la aplicación de sanciones académicas, sin perjuicio de otras actuaciones que pudieran iniciarse. En Puerto Real, a 6 de marzo de 2024

Fdo: Francisco Javier Molina Rojas

Agradecimientos

A mi madre y a mi padre, por siempre confiar en mí, animarme a seguir formandome y estar siempre a mi lado.

A mi hermano y a mi pareja Isabel, por aguantarme, acompañarme y hacerme reir cuando lo necesitaba.

A mi prima Miriam, por ayudarme en todo lo que hago y estar siempre pendiente de mí.

A mis tutores Juan Manuel y Mercedes, por ofrecerme realizar esta investigación con ellos y ayudarme siempre que lo he necesitado.

A mis compañeros Juan, Mari, Blanca, Adriana, Isabel, Jorge, Duván, Ale, Paula y Manu, por acompañarme a lo largo de la carrera y por apoyarme siempre.

A mis amigos, por ayudarme a desconectar y animarme.

A Ian y a María, por todas las clases de inglés recibidas que tanto me han ayudado a lo largo de mi vida.

A mi profesora Blanca, por apoyarme a no dejar bachillerato y enseñarme que con esfuerzo siempre se puede.

A todos los docentes de la carrera, por todo el conocimiento que me han proporcionado.

Resumen

En los últimos tiempos, se han incrementado las tecnologías que hacen uso de la inteligencia artificial entrenada por grandes cantidades de datos. Este auge ha hecho que la demanda de datos se incremente exponencialmente. Estos son necesarios para el correcto funcionamiento de las inteligencias artificiales.

Las compañías han reaccionado a este evento y, en la actualidad, se esfuerzan en recoger la mayor cantidad de estos datos para posteriormente explotarlos y obtener rentabilidad.

El problema aparece cuando estos datos contienen información de carácter personal y pueden simbolizar un peligro para la privacidad de las personas. Esta condición crea un obstáculo a la hora de difundir los datos, afectando directamente al desarrollo de la tecnología mencionada anteriormente.

Sobre este contexto se presentan diversas soluciones al problema de la privacidad destacando especialmente los datos sintéticos. Estos son datos generados artificialmente, es decir, no están asociados a ninguna persona por lo que garantizan la protección de la privacidad; y a pesar de no ser idénticos a los reales conservan una utilidad estadística similar.

A raíz de lo anterior, presentamos nuestro proyecto; este presenta un aplicativo web que genera datasets sintéticos a partir de un dataset original en formato CSV. Para ello, hace uso de la librería Sythetic Data Vault de Python. Con este aplicativo, los usuarios pueden anonimizar cualquier dataset utilizando uno de los sintetizadores que ofrece SDV.

Este aplicativo es de código abierto y tiene como objetivo principal permitir a todo tipo de usuario generar datos sintéticos sin importar la experiencia del mismo. De esta manera aseguraremos el acceso sencillo y gratuito a una herramienta de anonimización de datos.

Palabras clave: Dataset, Sintético, Datos, Synthetic Data Vault, Aplicativo web

Índice general

Índice de figuras

Índice de tablas

Parte I

Prolegómeno

1. Introducción

1.1. Motivación

En la actualidad, es muy costoso encontrar servicios o software gratuitos que no recopilen ni hagan uso de nuestros datos. En una sociedad donde el "Big Data" y la inteligencia artificial está a la orden del día, uno de los activos más valiosos y preciados son los datos de los usuarios. Estos datos son usados para diversas actividades, como el testeo de software o el entrenamiento de modelos de inteligencia artificial.

Las tecnologías basadas en inteligencia artificial nos brindan nuevas oportunidades en numerosos aspectos de nuestro día a día. Sin embargo, para que los resultados de estas sean óptimos, es necesario disponer de grandes cantidades de datos para entrenarlas (Gröger, 2021).

A raíz de esta necesidad de datos nacen varios problemas. En ocasiones, estos datos puede contener información sensible ligada a una persona. El uso o compartición de esta información puede atentar contra la privacidad de ellas (Farayola et al., 2024). Al ser estos datos privados no pueden ser distribuidos y, a su vez, esto puede causar un problema de falta de datos.

Debemos intentar dar solución a estos problemas para poder asegurar la privacidad de los usuarios.

1.2. Alcance y Objetivos

El principal objetivo del proyecto es el desarrollo de un aplicativo web de código abierto. Este permitirá a los usuarios generar datos sintéticos a través de un dataset en formato CSV y evaluar la calidad estadística del mismo.

El proyecto permitirá a los usuarios elegir y configurar distintos sintetizadores de datos. Por último, este tendrá una interfaz minimalista y sencilla para que usuarios sin experiencia puedan usarlo sin problema.

Se investigará en las tecnologías más recientes, con el objetivo de seleccionar la más adecuada para el proyecto.

1.3. Glosario de términos

CSV: Comma-Separated Values

IA: Inteligencia Artificial

SDV: Synthetic Data Vault

1.4. Organización del documento

En la primera parte, daremos una introducción al proyecto para posteriormente centrarnos en la gestión del mismo; especificaremos las metodologías, tecnologías, planificaciones, costes, riesgos y la organización.

En la segunda parte, expondremos detalladamente el desarrollo del proyecto; desde la toma de requisitos, análisis, diseño, construcción, pruebas y despliegue del sistema. Concretamente:

- En la sección de requisitos se especificarán los objetivos del sistema y los requisitos funcionales, no funcionales y de información.
- En la sección de análisis se realizarán los diferentes modelos en base a los requisitos recogidos en el anterior apartado.
- En la sección de diseño se mostrarán los diagramas de modelo C4, donde podemos observar los diferentes componentes del proyecto junto a la relación de estos. Además se determinarán los diferentes detalles técnicos del código usados.
- En la sección de construcción se definen los ficheros que componen al sistema.
- En la sección de pruebas se establecen las pruebas realizadas al sistema.
- En la sección despliegue se explica cómo se ha desplegado el sistema en un servidor de producción.

Para terminar, en la última parte se muestran las conclusiones obtenidas después de desarrollar el proyecto, además de los objetivos alcanzados y propuestas de posibles mejoras.

2. Antecedentes

2.1. Contexto

Existen múltiples problemas relacionados con la utilización de datos personales para los propósitos mencionados anteriormente, los que más destacan son:

- Problema de privacidad: los usuarios pueden sentir que su privacidad está comprometida.
- Problema de desarrollo: a veces la falta de datos o la poca distribución de estos causan problemas en los desarrollos o investigaciones.
- Problema de distribución de datos: con el desarrollo de soluciones de machine learning, los investigadores ahora más que nunca, tienen necesidad de obtener y compartir datos, pero estos al tener información personal no pueden ser compartidos.

Antes de presentar las soluciones, es necesario definir una serie de conceptos clave que nos permitan comprender la diferencia entre cada una de estas:

- Utilidad estadística: qué tan útil (estadísticamente) es un dataset. Podemos asumir que un dataset tendrá mayor utilidad estadística cuanto más se asemeje a los datos originales.
- Nivel de privacidad: qué tan privado es un dataset. Cuanto más dificil sea identificar a un individuo del dataset, mayor nivel de privacidad tendrá este.

En el pasado, se han propuesto diferentes soluciones para solventar estos problemas, destacando la técnica de "k-anonimato" (El Emam, 2008) y la técnica de "privacidad diferencial" (Dwork, 2006). La técnica de "k-anonimato" consiste resumidamente en la aplicación de diferentes acciones de anonimización a los datasets que contienen información de los usuarios. Mientras que, la técnica de "privacidad diferencial" consiste principalmente en la adición de ruido a ciertas columnas. Estas técnicas no son excluyentes, por lo que pueden combinarse para obtener mejores resultados en la anonimización, pero se deben tener en cuenta los defectos de estas.

El uso de datos sintéticos (Bellovin et al., 2019) y (Jordon et al., 2022) es una novedosa y prometedora técnica. Actualmente existen varios tipos de algoritmos y librerías que permiten obtenerlos y explotarlos.

El principal objetivo de esta investigación consiste en intentar brindar una solución mediante la generación de datos sintéticos; y en especial, usando la librería Synthetic Data Vault (SDV) de python.

2.2. Estado de la técnica

Examinando detenidamente la técnica "k-anonimato"podemos observar que posee dos fases:

- Supresión: se eliminan las columnas que pueden asociar o identificar a cualquier miembro de la población, como por ejemplo, nombres o los DNI.
- Generalización: los valores individuales de una o varias columnas son sustituidos por valores más generales.

Con respecto al nivel de privacidad, la técnica protege la identificación directa de los indiviuos, sin embargo, puede ser vulnerable en ocasiones en las que los atacantes poseen información de los usuarios. Si hablamos de la utilidad estadística, podemos ver como suprime columnas y generaliza valores, perdiendo mucha de su utilidad.

Por otro lado, la técnica de "privacidad diferencial" combina datos reales con datos falsos siguiendo una distribución probabilística. Lógicamente, cuanto más ruido se haya introducido mayor privacidad tendrá el dataset, pero a su vez, menor utilidad y exactitud.

Por último, los datos sintéticos consiguen combinar una protección en la privacidad de los usuarios al crear datos que no pertenecen a nadie y, a su vez, consiguen tener una distribución semejante a la de los datos originales, por lo que tienen una utilidad ideal.

Tabla 2.1

Comparación de soluciones

Solución	Privacidad	Utilidad
K-anonimato	Adecuada, pero vulnerable	Reducida
Privacidad diferencial	Depende de la distribución	Datos aleatorios, reducida
Datos sintéticos	Adecuada, datos sin dueño	Adecuada, distribución similar

A raíz de ser una tecnología prometedora, realizaremos una exploración sobre las soluciones existentes que usan y ofrecen los datos sintéticos.

Analizando las soluciones de software privativo podemos encontrar numerosos SAAS (Sofware as a Service), como por ejemplo, Mostly AI (Mostly AI, 2017) e IBM watsonx (IBM watsonx, 2023). Estos permiten la generación y tratamiento de los datos sintéticos.

En cuanto a las soluciones de software libre, analizaremos por una parte las librerías que brindan herramientas para la generación de datos sintéticos y, por otra parte, las soluciones software creadas para profesionales no desarrolladores que quieren incluir esta tecnología en sus proyectos.

Existe un amplio abanico de librerías que pueden proporcionar distintos instrumentos para la generación de datos sintéticos. Una de las más novedosas, funcionales y objeto de este trabajo, es Synthethic Data Vault (Patki et al., 2016) desarrollada inicialmente por el laboratorio de datos para IA del MIT en 2016. Esta librería ofrece

una considerable cantidad de sintetizadores de datos, además posee herramientas para poder realizar una evaluación sobre los datos obtenidos.

Si hablamos de las soluciones sofware libres que podemos encontrar, observamos que existen generadores para distintos tipos de tecnologías como por ejemplo, PeopleSansPeople (Erfanian Ebadi et al., 2022), un generador de datos sintéticos para la visión por computador centrada en el ser humano. Este nos permite generar datasets sintéticos, que posteriormente, sirven para la generación de imágenes, etiquetas y escenas 3D usadas para el reconocimiento de personas.

El trabajo realizará una exploración de la librería SDV y desarrollará un aplicativo web de código abierto que permita a los desarrolladores generar datasets sintéticos, usando diferentes sintétizadores configurables a partir un dataset de entrada en formato CSV. También permitirá evaluar los datos generados.

3. Plan de gestión de proyecto

3.1. Metodología de desarrollo

El desarrollo del proyecto ha consistido en las siguientes actividades:

- Investigación, recopilación de información y documentación del proyecto: incluye aquellas tareas que consisten en recaudar, analizar información de interés y redactar la documentación.
- Implementación: recoge todas las tareas relacionadas con la creación y la implementación del software.
- Testeo: abarca las tareas que tienen como objetivo probar y examinar el software.
- Despligue: contiene todas las acciones que tienen como objetivo desplegar el aplicativo en un servidor de producción.

  Cada tarea posee una prioridad y una estimación aproximada de tiempo de realización, lo que permite usar una metodología de desarrollo ágil.

3.2. Tecnologías

Para la construcción de la solución, he utilizado el lenguaje de programación Python; debido a que es uno de los lenguajes más versátiles y posee una gran cantidad de librerías que permiten implementar aplicativos web, gestionar datasets con extensión CSV y generar datos sintéticos.

Para implementar el aplicativo web, he decidido usar Flask junto a Jinja2.[1] Este es un framework minimalista que permite el desarrollo rápido de aplicaciones web.

Para gestionar los datasets, he elegido usar la librería Pandas[2]. Esta permite manipular y gestionar datos de diferentes formatos, incluyendo los ficheros CSV.

Para la creación de gráficos estadísticos, he decidido utilizar la librería Matplotlib[3], que permite generar y almacenar este tipo de gráficos.

Por último, para generar datos sintéticos he optado por usar la librería SDV[4] por los motivos expuestos anteriormente.

[1] Flask y Jinja2 - https://flask.palletsprojects.com/es/main/templating/
[2] Pandas - https://pandas.pydata.org/
[3] MatPlotLib - https://matplotlib.org/
[4] Synthetic Data Vault - https://docs.sdv.dev/sdv

3.3. Planificación del proyecto

Desde un inicio, se realizó un diagrama Gantt 3.1 estimando la duración de las tareas que componen el proyecto, estableciendo un plazo de entrega en unos aproximados 4 meses.

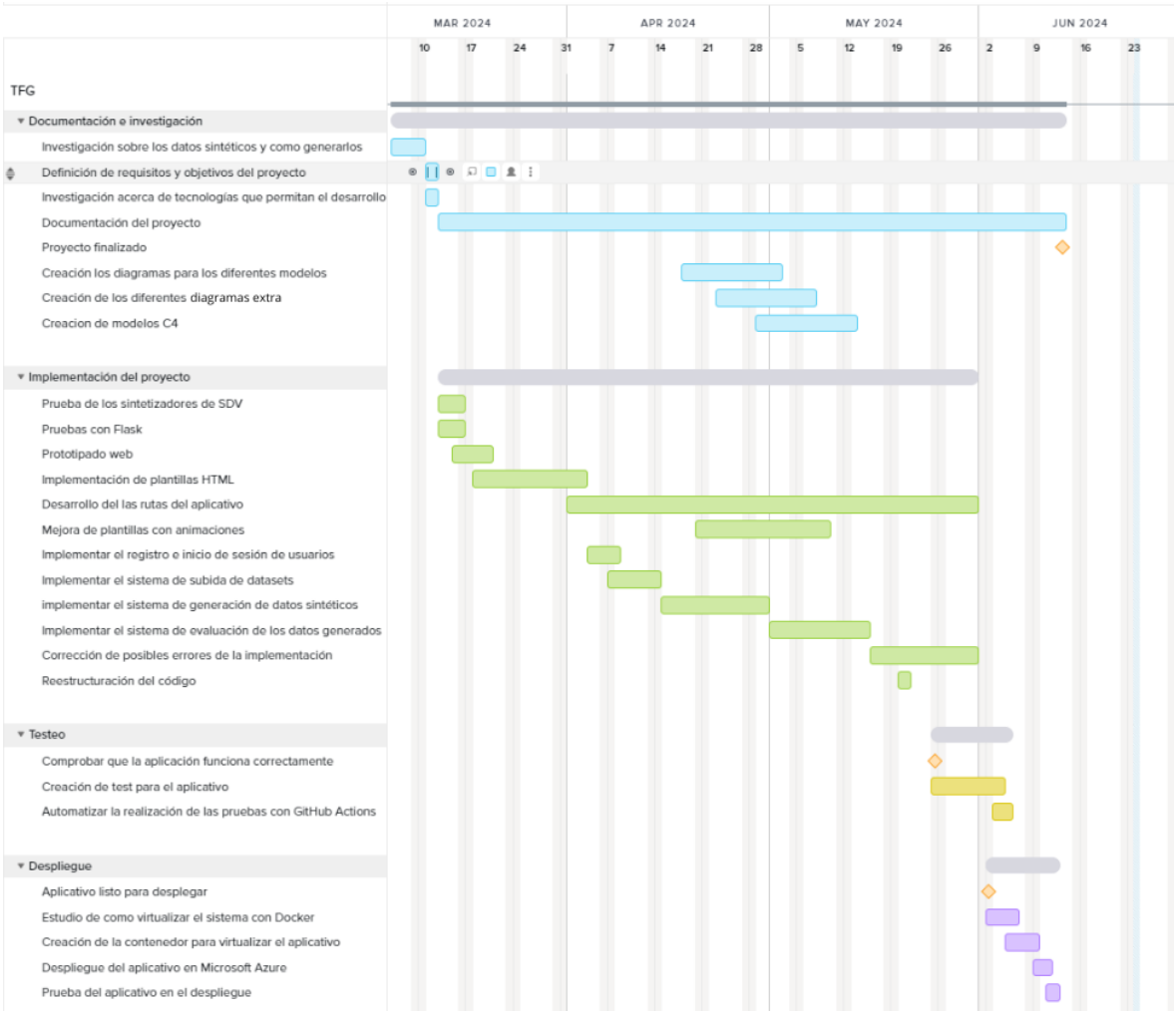

Figura 3.1: Diagrama gantt - estimación de duración de las tareas

Finalmente, debido a la realización de las prácticas mientras se desarrollaba el proyecto, varias tareas tuvieron que ser pospuestas y/o alargadas. Esto puede visualizarse en el siguiente diagrama 3.2

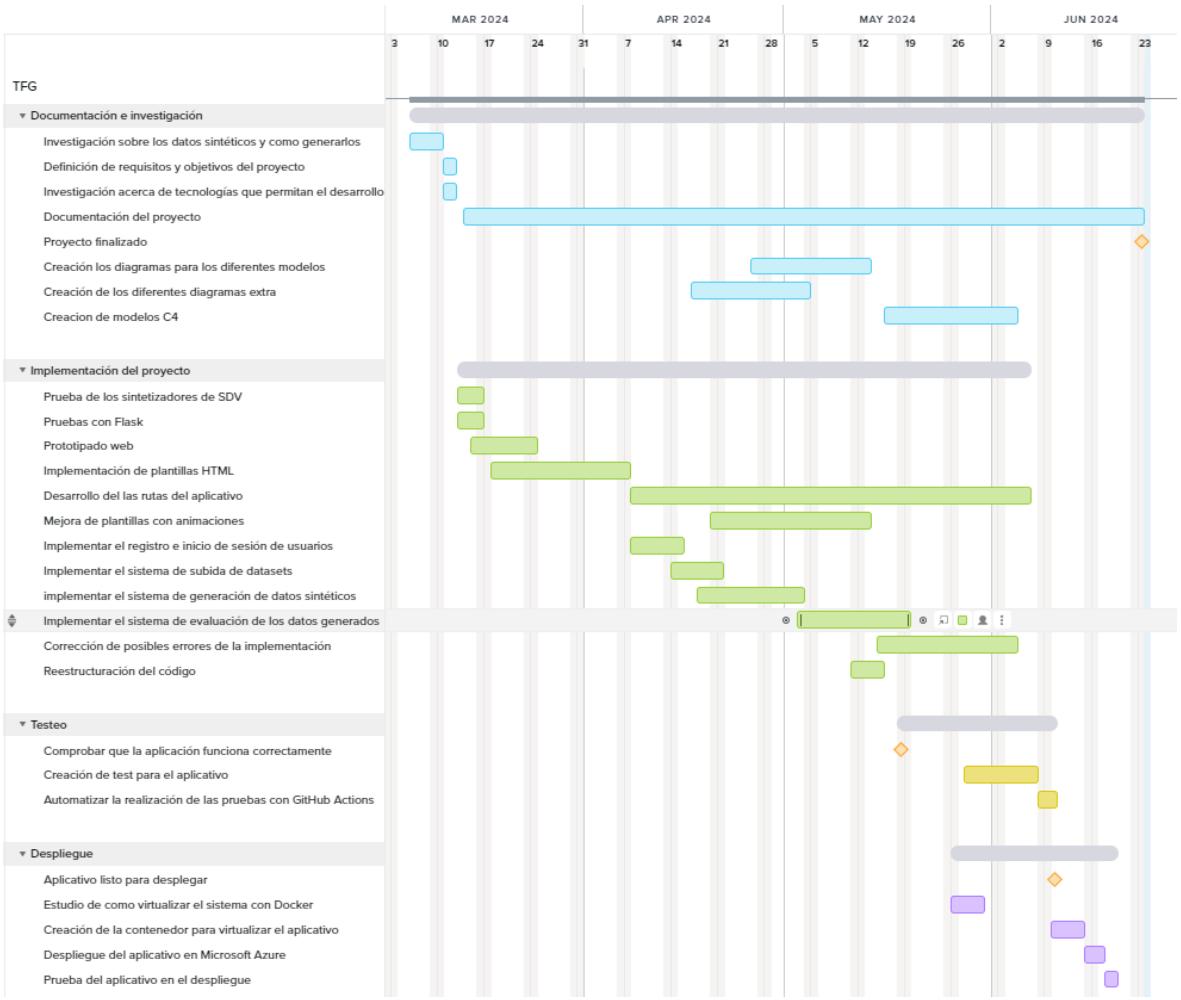

Figura 3.2: Diagrama gantt - duración real de las tareas

3.4. Organización

Este documento y proyecto ha sido desarrollado por el alumno Francisco Javier Molina Rojas, bajo la supervisión del Dr. Juan Manuel Dodero Beardo y la Dra. Mercedes Rodríguez García.

Para la realización de las tareas del proyecto, he tenido que asumir un amplio conjunto de roles relacionados con la ingeniería de software; desde la recogida de requisitos funcionales y no funcionales, hasta la realización de tests.

Para la fase de diseño y prototipado he usado la herramienta Figma[5] para poder visualizar y especificar un diseño a seguir. En la fase de implementación para programar he usado principalmente el IDE Pycharm, mientras que para el control de versiones he decidido crear un repositorio en Github[6]; y todo esto lo he realizado con mi ordenador personal.

[5]Figma - https://www.figma.com/
[6]Repositorio de Github - https://github.com/Javivi-MR/Synthetic-Data-Generator

Por último, para la realización de tests, he usado las librerías de python unittest y selenium[7].

3.5. Costes

Introduciremos los costes con la siguiente tabla:

Tabla 3.1

Desglose de costes

Concepto	Coste
Técnico Investigador Graduado	6860€
Costes indirectos (10 %)	686€
Lincencias software	0€
Total	7546€

Especificando en cada coste:

- Técnico Investigador Graduado: coste basado en la normativa presupuestaria de la UCA (Universidad de Cádiz) visible en el Capítulo VI (UCA, 2022) por 4 meses.
- Costes indirectos: contiene costes derivados del personal y proyectos.
- Licencias de software: algunas de las herramientas usadas para hacer el proyecto eran gratuitas, si bien otras tenían un coste asociado a la licencia. Sin embargo, al ser estudiante, pude obtener una licencia sin coste para estas herramientas.

3.6. Riesgos

En la siguiente tabla especificaré cada riesgo junto a su probabilidad de ocurrencia y el impacto que tendría:

Tabla 3.2

Tabla de riesgos

Riesgo	Probabilidad	Impacto
Definir erróneamente los objetivos	Baja	Alto
Incumplimiento de requisitos	Alta	Bajo
Incumplimiento del plazo de entrega	Medio	Medio

Describiendo individualmente cada riesgo:

- Definir erróneamente los objetivos: durante la definición del proyecto algunos de los objetivos pueden no quedar bien definidos, lo que puede causar confusiones y problemas durante el desarrollo.

[7]Selenium - https://www.selenium.dev/documentation/

- Incumplimiento de requisitos: finalizando el proyecto puede que alguno de los requisitos que definimos con anterioridad no queden satisfechos, lo que puede llevar a incongruencias.
- Incumplimiento del plazo de entrega: no es posible acabar el proyecto dentro del límite establecido.

El Plan de mitigación a seguir es el siguiente:

- Definir erróneamente los objetivos: en etapas tempranas del proyecto se tendrán reuniones con los tutores para definir adecuadamente todos los objetivos.
- Incumplimiento de requisitos: al definir los requisitos se tendrán en cuenta el tiempo de realización de cada uno y cómo puede afectar al desarrollo, esto con el objetivo de poseer los que sean necesarios e indispensables.
- Incumplimiento del plazo de entrega: realizar una planificación realista, valorando la longitud aproximada de cada tarea.

3.7. Gestión de la configuración

Como se ha explicado anteriormente, el control de versiones se hará a través de la plataforma GitHub con acceso libre. El objetivo de esto es hacer que la distribución y descarga del software sea más sencilla.

3.8. Aseguramiento de calidad

Para asegurar la calidad del software producido, contaré con la ayuda del propio IDE PyCharm y de la extension SonarLint[8], que asistirán en las tareas de corrección de la sintaxis del código; y también asegurarán que el estándar de Python PEP-8[9] se cumpla para tener un código limpio, seguro y de calidad.

[8]SonarLint - https://www.sonarsource.com/products/sonarlint/
[9]PEP-8 - https://peps.python.org/pep-0008/

Parte II

Desarrollo

4. Requisitos del Sistema

4.1. Objetivos del Sistema

Los objetivos que debe cumplir el sistema son:

- Ser sencillo de desplegar: en cuanto a las acciones necesarias para poder realizar el despliegue del sistema, deben de ser breves y concisas.
- Poseer una interfaz gráfica web que sea: limpia, minimalista, accesible y funcional.
- Permitir a los usuarios privatizar datos: a través de generación de datos sintéticos para datasets en formato CSV.
- Permitir a los usuarios evaluar la calidad estadística de los datasets generados.

4.2. Requisitos funcionales

El sistema debe permitir a los usuarios realizar las siguientes funciones:

- Registrarse e iniciar sesión en el sistema: el registro e inicio de sesión requerirán de un nombre de usuario y una contraseña.
- Subir datasets en formato CSV. Se habilitará un formulario donde los usuarios podrán subir datasets en formato CSV. Estos deberán tener cabecera y los nombres de las columnas no deberán tener caracteres especiales.
- Generar datasets sintéticos, pudiendo especificar el número de filas de este y pudiendo seleccionar uno de los sintetizadores de datos que ofrece la librería SDV. Además se permitirá cambiar todos los parámetros modificables que ofrezca el sintetizador.
- Permitir comparar los datos sintéticos generados con los originales. El fin es mostrar la usabilidad estadística del dataset generado.

4.3. Requisitos no funcionales

El sistema debe cumplir los siguientes requisitos:

- La interfaz del sistema debe ser: intuitiva, usable, accesible, limpia y atractiva para el usuario.
- El sistema debe ser: funcional y fácil de desplegar. Cualquier usuario, independientemente de su experiencia, debe poder desplegar el sistema.
- La implementación del sistema debe ser eficiente y debe estar optimizada.

- El sistema debe ser de código abierto y debe estar localizado en un repositorio público y documentado, facilitando la participación de personas externas para mejorar y modificar este.

4.4. Requisitos de información

La información del sistema que se debe gestionar se divide en las siguientes categorías:

- Usuarios. Su información a gestionar es:
  - Identificador.
  - Nombre de usuario.
  - Contraseña.
  - Datasets asociados al usuario.
- Datasets originales. La información a gestionar es:
  - Identificador.
  - Nombre del dataset.
  - Path al dataset.
  - Usuario propietario del dataset.
- Datasets Generados. La información a gestionar es:
  - Dataset original asociado.
  - Valor/similitud estadística.
  - Propiedades estadísticas:
    - Puntuación global de similitud.
    - Media (solo en columnas con valores numéricos continuos).
    - Varianza (solo en columnas con valores numéricos continuos).
    - Covarianza (solo en pares de columnas con valores numéricos continuos).
    - Moda (solo en columnas con valores discretos).
    - Puntuación de similitud de cada columna.
    - Covarianza entre pares de columnas numéricas.
    - Puntuación de similitud entre pares de columnas.
    - Datos de regresión entre pares de columnas numéricas: coeficiente de Pearson, recta de regresión y coeficiente de determinación.

5. Análisis del Sistema

5.1. Modelo Conceptual

A partir de los requisitos de información expuestos en la sección 4.4, obtenemos el diagrama conceptual que permite ver las relaciones entre los diferentes objetos del sistema.

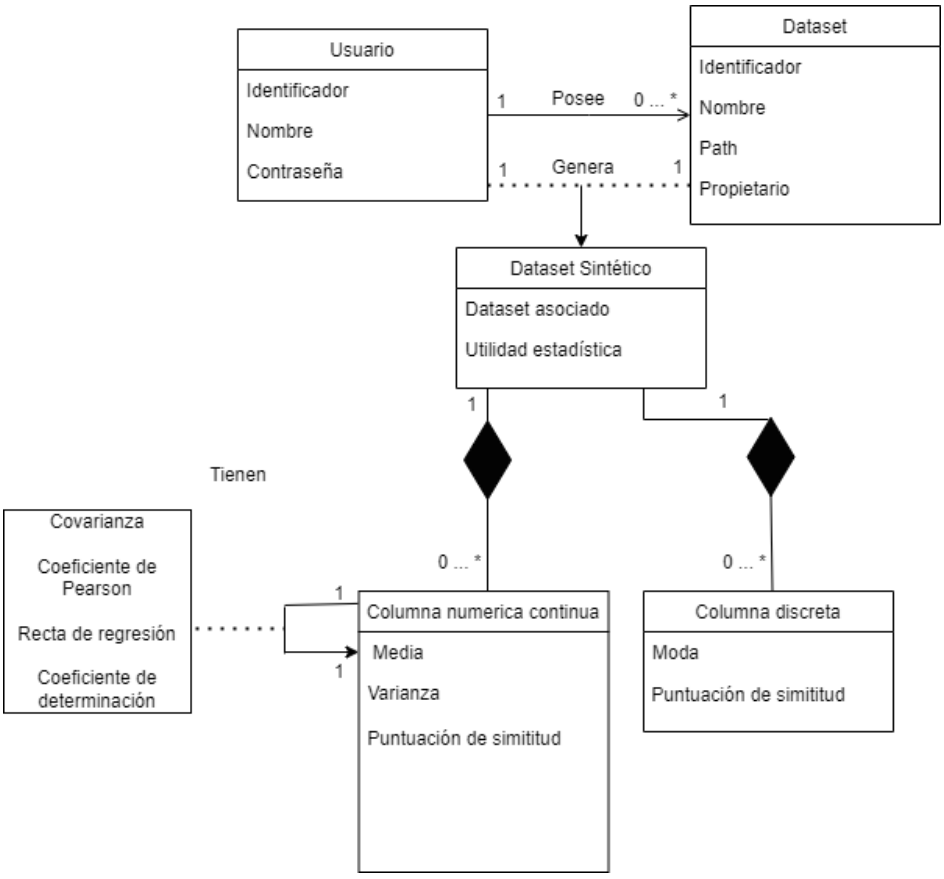

Figura 5.1: Diagrama conceptual de datos

5.2. Modelo de Casos de Uso

A partir de los requisitos funcionales descritos con anterioridad, expondremos los dos casos de uso más relevantes. El resto de casos de uso pueden ser consultados en el anexo C.

Nombre del CU:	Subir dataset
Creado por:	Francisco Javier Molina Rojas
Fecha	10/05/2024
Descripción	Cuando un usuario está registrado, puede subir un dataset al sistema para posteriormente usarlo.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. El usuario debe poseer un dataset en formato csv. 3. El sistema está iniciado y hay espacio disponible.
Postcondiciones	1. El sistema hace una copia local del dataset subido y lo asocia en la base de datos al usuario.
Flujo	1. El usuario, una vez se ha registrado o iniciado sesión en el sistema, hace clic en "Generate". 2. El usuario hace clic en el botón "Upload a dataset". 3. El usuario hace click en "Seleccionar archivo". 4. El usuario selecciona un dataset en formato csv. 5. El usuario hace click en "Submit". 6. El sistema realiza una copia local del dataset y almacena una asociación del dataset y el usuario en la base de datos.
Flujo alternativo	4a. En el paso 4, el usuario no selecciona ningún archivo. 5. El usuario hace click en "Submit". 6. El sistema notifica al usuario que no ha seleccionado ningún archivo y le permite rectificar. 4b. En el paso 4, el usuario selecciona un archivo que no tiene formato csv. 5. El usuario hace click en "Submit". 6. El sistema le muestra al usuario un mensaje de error junto al motivo de este (en este caso, el fichero seleccionado no tiene formato csv). 4c. En el paso 4, El usuario introduce un fichero csv, pero este no tiene cabecera (nombre de las columnas) o esta tiene caracteres especiales. 5. El usuario hace click en "Submit" 6. El sistema le muestra al usuario un mensaje de error junto al motivo de este.
Excepciones	5a. En el paso 5, el usuario cierra la aplicación. 6. El sistema no generará los datos sintéticos.
Requisitos	Para poder realizar este caso de uso se tienen que cumplir las siguientes condiciones: 1. El usuario debe estar registrado en el sistema. 2. El sistema debe tener espacio suficiente.

Figura 5.2: Caso de uso Subir dataset

Nombre del CU:	Generar datos sintéticos
Creado por:	Francisco Javier Molina Rojas
Fecha	10/05/2024
Descripción	Cuando un usuario está registrado y ha subido al menos un dataset, puede generar datos sintéticos de alguno de sus datasets.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. El usuario debe tener al menos un dataset subido al sistema. 3. El sistema está iniciado y hay espacio disponible.
Postcondiciones	1. El sistema genera datos sintéticos en formato csv.
Flujo	1. El usuario, una vez se ha registrado o iniciado sesión en el sistema, hace clic en "Generate". 2. El usuario decide qué dataset quiere usar. 3. El usuario selecciona un número de filas. 4. El usuario selecciona y configura un sintetizador de datos. 5. El usuario hace click en "Generate". 6. El sistema genera los datos sintéticos del dataset usando el sintetizador y el número de filas seleccionadas. Para finalizar lo redirige
Flujo alternativo	3a. – 4a. El usuario introduce un número de filas o un número de etapas (si el sintetizador de datos lo permite) 5. El usuario hace click en "Generate" 6. El sistema detecta el número negativo, avisa al usuario y le permite rectificar.
Excepciones	5a. En el paso 5, el usuario cierra la aplicación. 6. El sistema no generará los datos sintéticos.
Requisitos	Para poder realizar este caso de uso se tienen que cumplir las siguientes condiciones: 1. El usuario debe estar registrado en el sistema. 2. El usuario debe haber subido al menos un dataset. 3. El sistema debe tener espacio suficiente.

Figura 5.3: Caso de uso Generar datos sintéticos

5.3. Modelo de Interfaz de Usuario

La interacción del usuario con el sistema se realizará mediante la navegación, de diferentes páginas web. En el siguiente diagrama de navegación mostramos cómo el usuario puede moverse por el sistema.

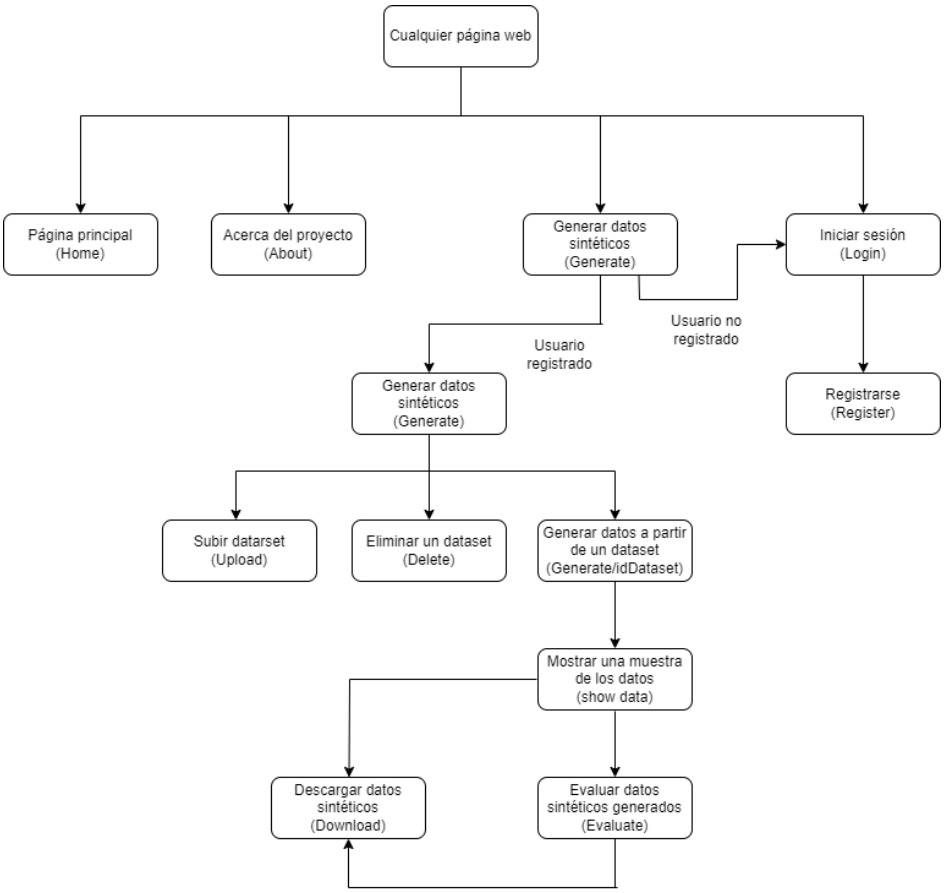

Figura 5.4: Diagrama de navegación

5.4. Modelo de Comportamiento

A partir de los casos de uso definidos en la sección 5.2, mostramos los siguientes diagramas de secuencia de operaciones.

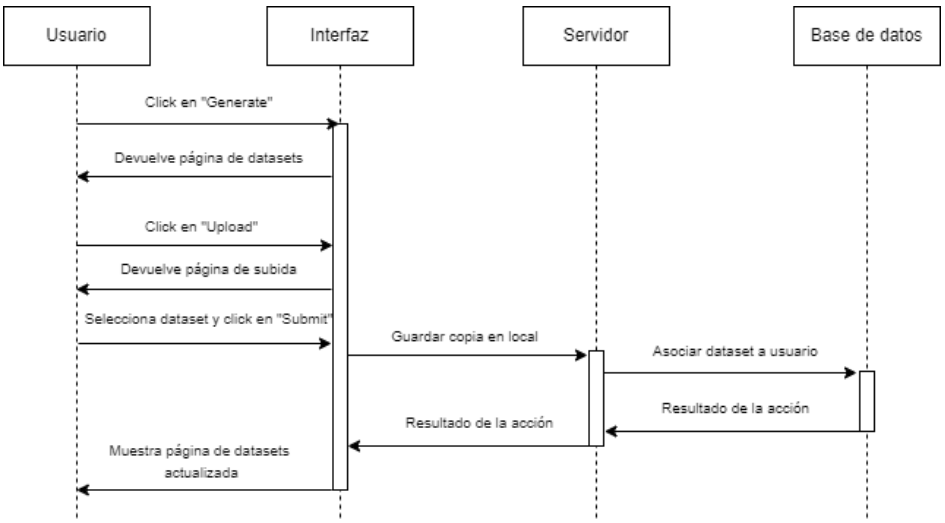

Figura 5.5: Diagrama de secuencia "Subir dataset"

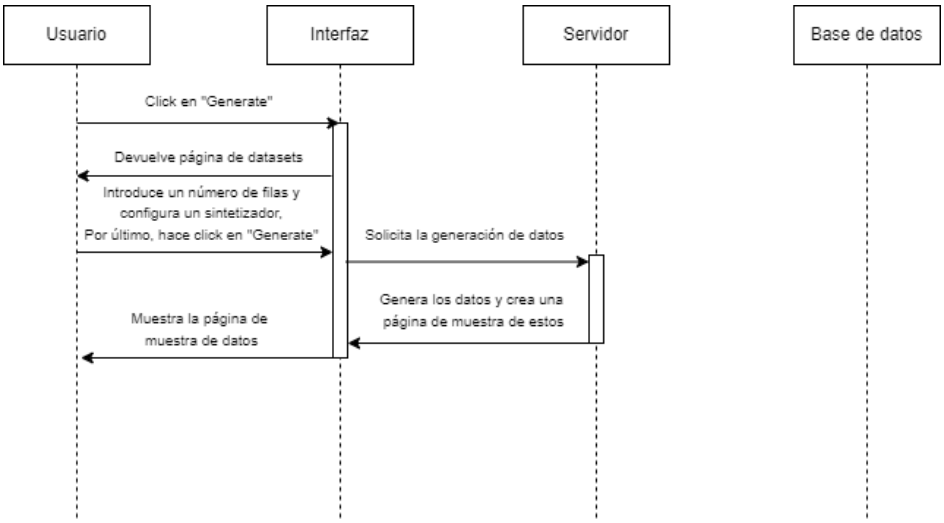

Figura 5.6: Diagrama de secuencia "Generar datos sintéticos"

El resto de diagramas de secuencia pueden ser consultados en el anexo D.

6. Diseño del Sistema

6.1. Arquitectura del Sistema

La arquitectura del sistema está compuesta principalmente por un script en lenguaje Pyhton que tiene como objetivo iniciar el servidor y este permite a los usuarios interactuar con el sistema. Esta interacción se realizará mediante páginas web usando la librería Flask.

La gestión y manipulación de los datasets se llevará a cabo mediante la librería Pandas, mientras que la generación de datos sintéticos y su posterior evaluación se realizarán con la ayuda de la librería SDV.

6.1.1. Diseño de alto nivel

Al ser un sistema web, utilizamos el patrón Layers (capas), en el que cada capa utilizará funciones que ofrecen las capas inferiores a esta. Como podemos ver el siguiente diagrama de contexto 6.1, el sistema está compuesto por personas y 3 sistemas distintos que abarcan diferentes tipos de tareas.

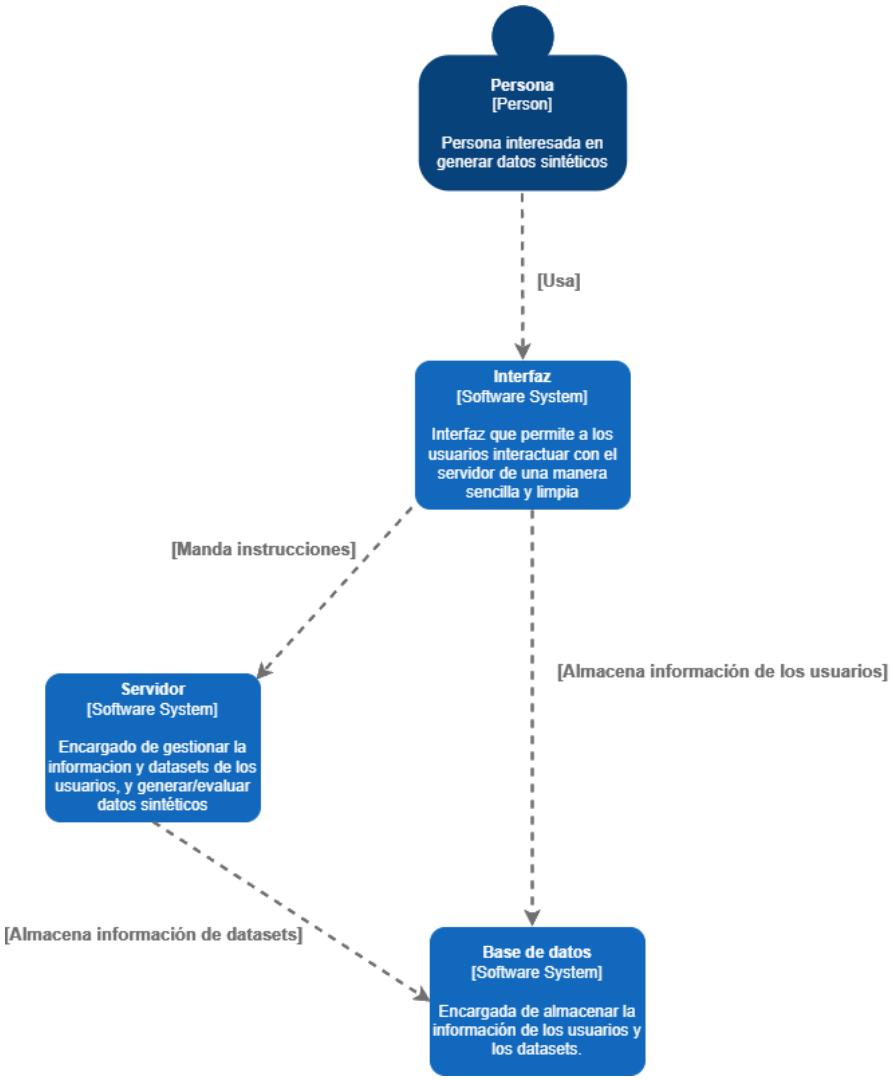

Figura 6.1: Diagrama de contexto

La primera capa a distinguir es la capa de interfaz, que es la que permite al usuario interactuar con el sistema subyacente. Esta integra un servidor WSGI, que la librería Flask de Python genera automáticamente. Además el aplicativo de Flask B.1 ofrece la navegación entre páginas web, mientras que el modulo de sesión B.2 nos permite gestionar los registros e inicios de sesión de los usuarios.

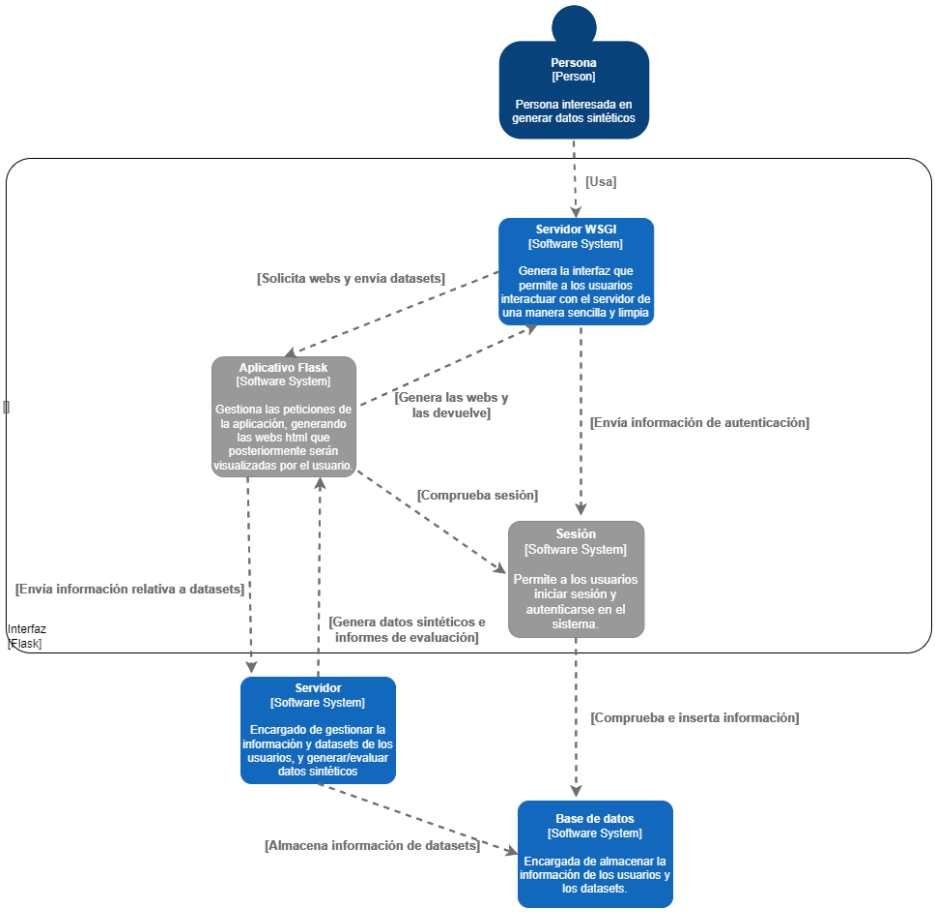

Figura 6.2: Diagrama de contexto - Interfaz

Otra capa que es importante es la capa de servidor, es la encargada de generación de datos sintéticos y la posterior evaluación de estos. Esta integra los siguientes componentes:

- Gestor de procesos: gestiona los diferentes procesos que pueda solicitar el usuario.

- Sintetizadores de datos B.4: proporcionados por SDV, son los encargados de generar los datos sintéticos y configurables por el usuario.

- Evaluador B.3: proporcionado por SDV, es el encargado de realizar la comparación entre los datos originales y los datos sintéticos.

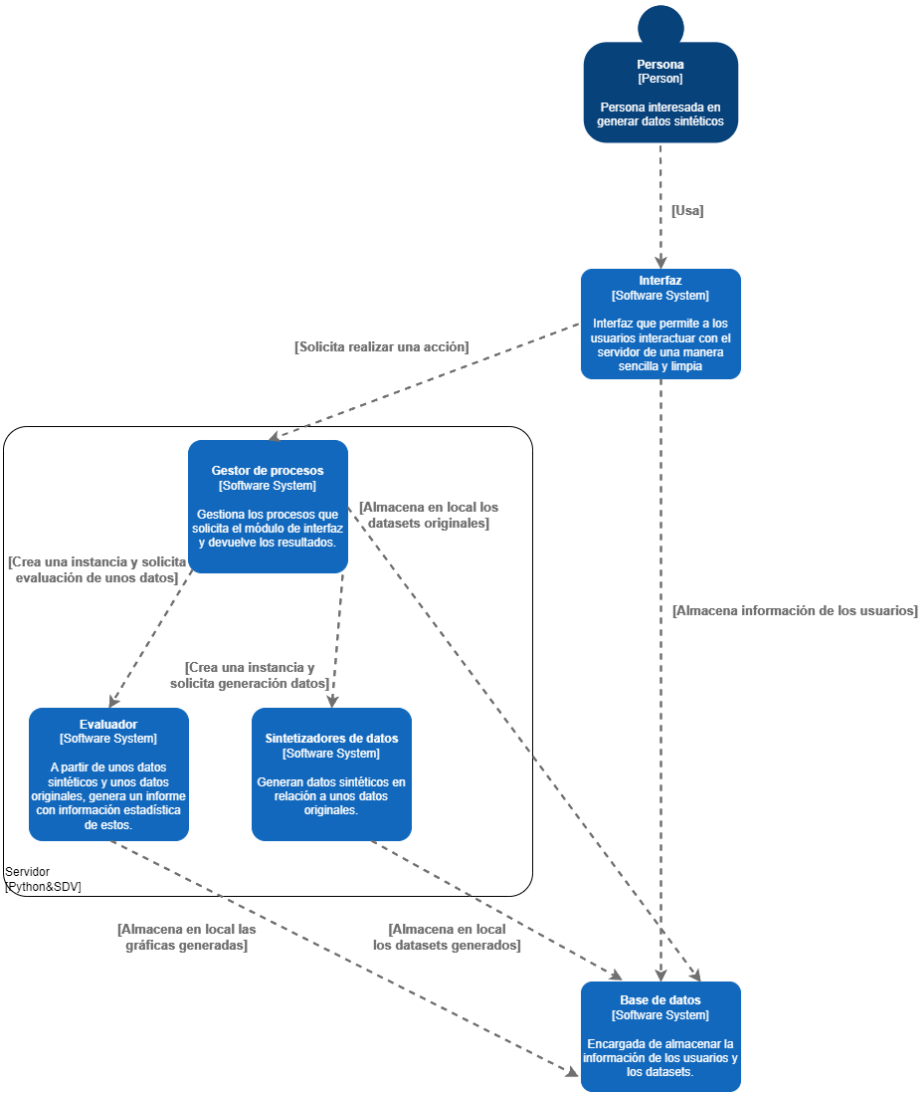

Figura 6.3: Diagrama de contexto - Servidor

Por último, la capa de base de datos incluye dos componentes principales:

- Base de datos SQLite: en la que se almacena la información relativa a los usuarios y los datasets asociados a estos.

- Archivos locales: donde se almacenan los propios datasets subidos por los usuarios, los datos sintéticos generados y los gráficos generados por el servidor.

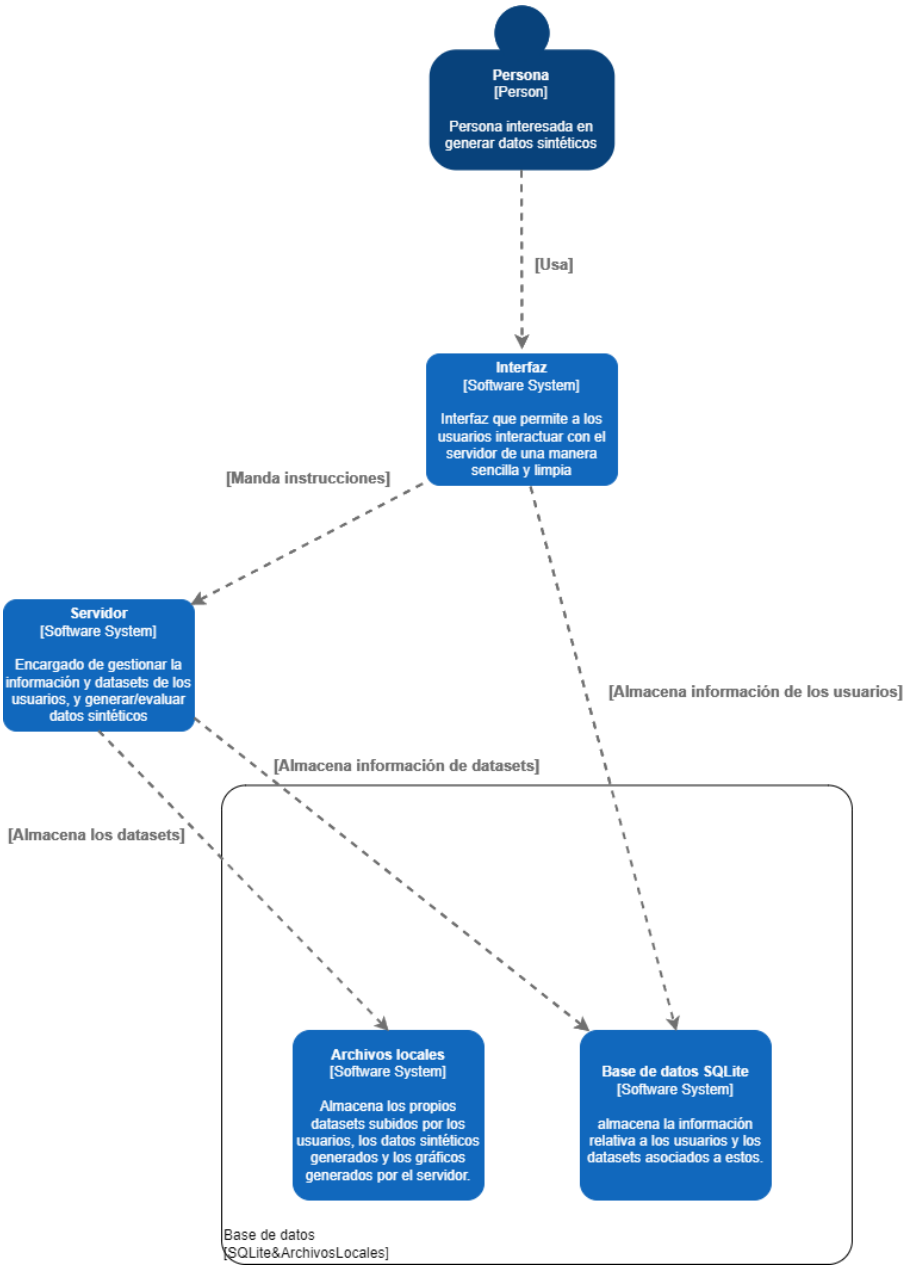

Figura 6.4: Diagrama de contexto - Base de datos

El resto de diagramas de modelo C4, junto a la descripción de estos puede ser consultados en el anexo B.1

6.1.2. Diseño detallado

En esta sección se expondrán en detalle las diferentes funciones que incluye el aplicativo web desarrollado.

Generación de datos sintéticos:

La función principal del sistema es la generación de datos sintéticos; para generar datos sintéticos es necesario usar sintetizadores de datos.

Un sintetizador de datos es un objeto que a través del uso de algoritmos estadísticos y/o deep learning, genera datos sintéticos a partir de unos datos originales.

El sistema permite utilizar todos los sintetizadores gratuitos que ofrece SDV y, a la vez, permite configurar todos los parámetros de estos.

Similitudes de los sintetizadores:

Los sintetizadores que vamos a exponer comparten varias propiedades. Si hablamos sobre sus parámetros configurables, todos comparten el siguiente:

- metadata: metadatos provenientes del dataset. La librería Pandas nos permite obtener estos cuando cargamos en memoria cualquier dataset. Este parámetro es obligatorio.

A su vez, los sintetizadores comparten los siguientes métodos:

- Método fit(df): tiene como objetivo entrenar al modelo. Recibe como parámetro de entrada un objeto Dataframe de la librería Pandas con los datos originales.
- Método sample(num_rows): usado para generar los datos sintéticos. Recibe como parámetro de entrada el número de filas que debe tener el Dataframe generado.

Gaussian Copula Synthesizer:

Sintetizador que mediante el uso de métodos estadísticos entrena a un modelo de machine learning que, posteriormente, permite la generación de datos sintéticos.

Parámetros configurables propios:

- enforce_min_max_values: permite añadir una restricción para que los valores numéricos generados estén en un rango entre el valor mínimo y máximo de los datos originales. Es opcional y por defecto es True.
- enforce_rounding: permite añadir una restricción para que los valores númericos generados posean el mismo número de decimales que los originales. Es opcional y por defecto es True.
- numerical_distributions: toma por entrada un diccionario donde las entradas son nombres de columnas, y los valores son una distribución numérica (por defecto no se especifica ninguna). Es opcional y permite ajustar la generación de los datos de una columna a una de estas distribuciones:
    - norm: normal.
    - beta: beta.
    - truncnorm: normal truncada.
    - uniform: uniforme.
    - gamma: gamma
    - gaussian_kde: densidad de kernel gaussiana.

- default_distribution: toma una de las anteriores distribuciones numéricas y las aplica al resto de columnas que no hayan especificado una distribución en numerical_distributions. Es opcional y por defecto usa la distribución Beta.

CTGAN Synthesizer:

Sintetizador que utiliza métodos de deep learning basados en GAN (red generativa antagónica) para entrenar a un modelo y posteriormente generar datos.

Parámetros configurables propios:

- enforce_rounding: permite añadir una restricción para que los valores númericos generados posean el mismo número de decimales que los originales. Es opcional y por defecto es False.
- epochs: número de etapas que debe superar la red GAN. Es opcional y por defecto es 300.
- verbose: permite visualizar en consola los resultados de cada etapa. Es opcional y por defecto es False.
- cuda: permite usar la GPU (si es posible) para agilizar los cálculos. Es opcional y por defecto es True.

TVAE Synthesizer:

Sintetizador que utiliza técnicas de redes neuronales basadas en codificador automático variacional (VAE) para entrenar a un modelo y posteriormente generar datos.

Parámetros configurables propios:

- enforce_min_max_values: permite añadir una restricción para que los valores numéricos generados estén en un rango entre el valor mínimo y máximo de los datos originales. Es opcional y por defecto es True.
- enforce_rounding: permite añadir una restricción para que los valores númericos generados posean el mismo número de decimales que los originales. Es opcional y por defecto es True.
- epochs: número de etapas que debe superar la red VAE. Es opcional y por defecto es 300.
- verbose: permite visualizar en consola los resultados de cada etapa. Es opcional y por defecto es False.
- cuda: permite usar la GPU (si es posible) para agilizar los cálculos. Es opcional y por defecto es True.

Copula GAN Synthesizer:

Sintetizador que utiliza una mezcla de métodos estadísticos y de deep learning basados en GAN para entrenar a un modelo y posteriormente generar datos.

Parámetros configurables propios:

- enforce_min_max_values: permite añadir una restricción para que los valores numéricos generados estén en un rango entre el valor mínimo y máximo de los datos originales. Es opcional y por defecto es True.
- enforce_rounding: permite añadir una restricción para que los valores númericos generados posean el mismo número de decimales que los originales. Es opcional y por defecto es True.
- numerical_distributions: toma por entrada un diccionario, donde las entradas son nombres de columnas y los valores son una distribución numérica (por defecto no se especifica ninguna). Es opcional y permite ajustar la generación de los datos de una columna a una de las distribuciones anteriormente descritas.
- epochs: número de etapas que debe superar la red GAN. Es opcional y por defecto es 300.
- verbose: permite visualizar en consola los resultados de cada etapa. Es opcional y por defecto es False.
- cuda: permite usar la GPU (si es posible) para agilizar los cálculos. Es opcional y por defecto es True.

Fast ML:

Este sintetizador se encuentra obsoleto, se recomienda usar el sintetizador "Gaussian Copula" en su lugar.

Este sintetizador no posee parámetros configurables propios.

Evaluación y comparación de datasets

Cuando usamos un sintetizador y generamos datos sintéticos, podemos evaluar la calidad estadística de estos comparándolos con los datos originales. SDV posee un apartado de evaluación que permite obtener información acerca de la calidad de los datos generados.

Destacan los siguientes métodos:

- QualityReport(): devuelve un objeto informe vacío.
- report.generate(real_data, synthetic_data, metadata): inicializa un informe. Recibe por parámetro de entrada los datos reales y sintéticos (en objetos Dataframe de pandas) y los metadatos de estos.
- report.get_score(): devuelve una puntuación del 0 al 1 que resume la calidad de los datos, siendo 0 la más baja y 1 la más alta.
- report.get_properties(): devuelve un Dataframe con la puntuación de semejanza en columnas individuales y entre pares de columnas.
- get_details(property_name): recibe una cadena con el nombre de la propiedad (Column Shapes para columnas individuales y Column Pair Trends para pares de columnas), de la que se quiere obtener detalles. Devuelve un Dataframe que contiene las columnas del dataset, las puntuaciones que han obtenido y las métricas usadas.

- get_visualization(property_name): recibe una cadena con el nombre de la propiedad que se quiere visualizar. Devuelve una figura (Plot) en la que se muestran las columnas del dataset junto al la puntuación obtenida.

6.2. Parametrización del software base

El sistema permite la configuración de algunos parámetros a través del archivo *config.py*. Donde, es posible modificar:

- WORKERS: número máximo de hilos que el aplicativo web usará.
- DATASET_PATH: ruta hacia la carpeta donde quiere que se guarden los datasets subidos.
- PLOT_PATH: ruta hacia la carpeta donde quiere que se guarden las figuras generadas.
- SYNTHETIC_PATH: ruta hacia la carpeta donde quiere que se guarden los datasets sintéticos generados.
- DATA_BASE_URI: dirección de conexión con la base de datos SQLite.
- SECRET_KEY: variable de protección para el aplicativo web. Debe ser privado.

6.3. Diseño Físico de Datos

El almacenamiento de datos en el sistema se realiza en dos apartados diferentes.

Para almacenar la información de los usuarios y las referencias a los datasets que poseen, se usa una base de datos relacional SQLite[1]; debido a que es un motor de bases de datos ligero y no hay necesidad de almacenar una gran cantidad de datos.

Además, el sistema de archivos locales que proporciona el sistema operativo, almacena los archivos que necesita el aplicativo.

6.3.1. Sistema de archivos locales

Con la configuración inicial del sistema, existen 3 directorios almacenados en el directorio *static*:

- data: directorio donde se almacenan los datasets en formato CSV, que los usuarios suben al sistema. Son almacenados para posteriormente poder ser usados para la generación de datos sintéticos.
- synthetic: directorio donde se almacenan los datasets sintéticos que se generan, con el objetivo de que el usuario pueda descargarlos.

[1]SQLite - https://www.sqlite.org/

- plots: directorio donde se almacenan los gráficos de comparación de columnas entre los datos originales y sintéticos. Si un usuario solicita evaluar unos datos, estos gráficos pueden ser consultados en el aplicativo.

6.3.2. Base de datos

Para poder usar la base de datos desde el aplicativo web de Flask, utilizamos la herramienta SQLAlchemy. Esta nos permite gestionar de forma sencilla las tablas, los datos, las conexiones y peticiones que se realicen.

Con la configuración inicial del sistema, al ejecutarlo, se crea archivo en el directorio *instance* con el nombre *database.db*. En este se almacenarán los datos y las tablas.

El esquema de la base de datos es secillo y está compuesto por dos tablas:

- User: almacena la información de los usuarios (nombre y contraseña) y asigna un identificador único con un entero que se incrementa.
- Dataset: almacena información del dataset (nombre, path hacia él y usuario que lo ha subido) y asigna un identificador único con un entero que se incrementa.

6.4. Diseño de la Interfaz de Usuario

6.4.1. Flask y Jinja2

Para diseñar la interfaz inicialmente se usó Figma[2], un editor gráfico con el que se crearon prototipos, que posteriormente servirán como una base para implementar la interfaz.

Inicialmente, planteamos crear una aplicación de escritorio. Buscando una herramienta que me permitiese crear una GUI (Interfaz Gráfica de Usuario), encontré Tkinter.

Tkinter es una librería de python que proporciona un kit de herramientas para la creación de GUI, disponible para sistemas UNIX y windows.

Esta idea fue descartada en una etapa temprana del desarrollo y sustituida por la idea de crear un aplicativo web. Los motivos para hacer este cambio son los siguientes:

- Incompatibilidades: un aplicativo web será más compatible que una aplicación de escritorio. Siempre que un servidor mantenga disponible dicha aplicación web, será accesible desde cualquier navegador sin importar su sistema operativo ni componentes.

[2]Figma - https://www.figma.com/

- Disponibilidad: la naturaleza de los aplicativos web (alojados en un servidor) hace que estén disponibles a todas horas para todos los dispositivos sin necesidad de iniciar o instalar un programa o software.
- Rendimiento: los aplicativos web suelen ofrecer un rendimiento superior a los de escritorio.
- Tendencias: en un mundo donde la conexión a Internet está al alcance de cada vez más personas, hace que se tienda a la creación de más aplicaciones webs.

El aplicativo web fue realizado con Flask. Este framework minimalista, incluye un motor de plantillas html llamado Jinja2; este facilita, agiliza y permite gestionar el proceso de creación de archivos html, introduciendo los conceptos de herencia, condicionales y usos de variables.

A partir de aquí, implementamos los distintos apartados del aplicativo web usando como inspiración los prototipos anteriormente expuestos y siguiendo un estilo minimalista y accesible.

6.4.2. Elementos comunes

Barra superior

Siguiendo el estándar de la gran mayoría de webs, todas las páginas creadas tendrán en la parte superior una barra de navegación (top bar).

Esta barra facilita a los usuarios la navegación por la página web haciendo que sea más accesible.

En la parte izquierda, encontramos el logotipo (escudo de la universidad de Cádiz), que sirve como hipervínculo hacia la web principal y el nombre del aplicativo web (Synthetic Data Generator).

En la parte derecha, encontramos texto con hipervínculos hacia las diferentes secciones de la web. Estos son los siguientes:

- Main: enlace hacia la página principal.
- Login: enlace hacia la página de inicio de sesión.
- Generate: enlace hacia la página de generación de datos sintéticos.
- About: enlace hacia la página de información del proyecto.

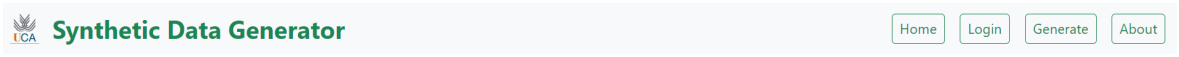

Figura 6.5: Barra de navegación superior

Barra inferior

Siguiendo con el anterior estándar, todas las webs tendrán una barra inferior (footer bar).

En esta se incluye una vez más el logotipo de la universidad de Cádiz, el nombre de la aplicación, y el año de creación.

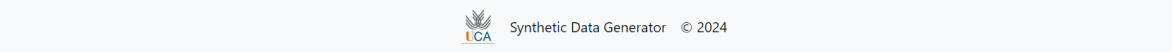

Figura 6.6: Barra inferior

6.4.3. Página principal

Página de inicio del sistema. Orientada a presentar la herramienta al usuario.

Es accesible desde el inicio de la aplicación, desde el nombre e icono de la barra superior y desde el apartado "Home".

Esta página contiene una portada (imagen con texto de introducción) y una sección de FAQ (preguntas frecuentes) con diferentes cuestiones acerca de la aplicación y sus objetivos.

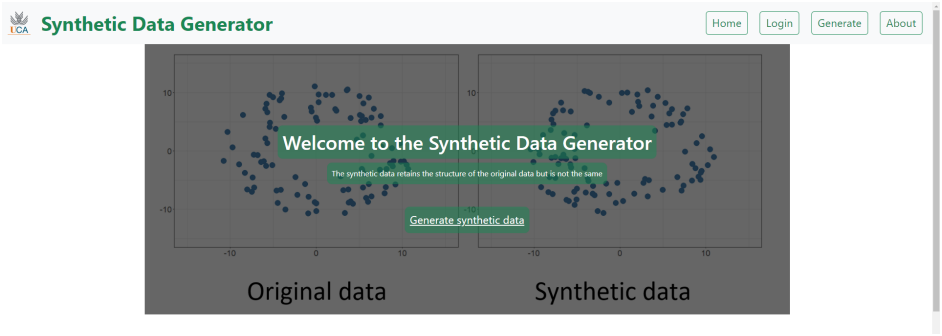

Figura 6.7: Página principal - Portada

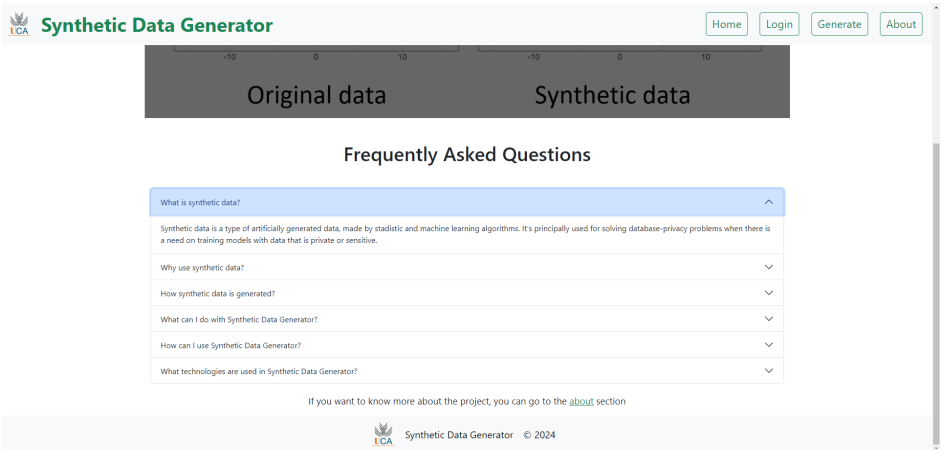

Figura 6.8: Página principal - FAQ

6.4.4. Página de información

Página con información acerca del proyecto y del creador.

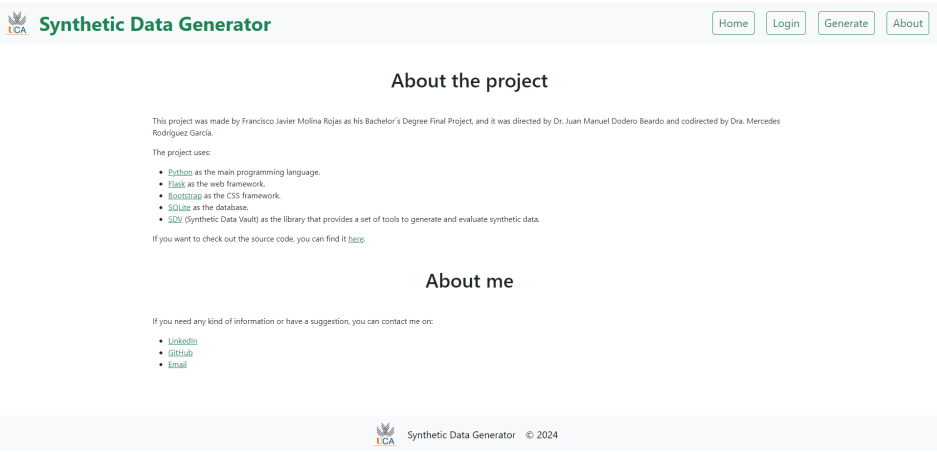

Figura 6.9: Página de información

6.4.5. Página de inicio de sesión - registro

Para acceder a la página de inicion de sesión, se debe pulsar en el apartado "Login"de la barra de navegación superior.

En ella, nos encontramos un sencillo formulario de dos campos: username (nombre de usuario) y password (contraseña).

Si el usuario olvidase su contraseña, podría dirigirse al vinculo "Have you forgotten your password?". En esta sección se le mostraría un mensaje con la acción a realizar.

Si el usuario no dispone de una cuenta en el aplicativo, podría crearla siguiendo el vínculo "Don't have an account? Register here". En esta sección encontramos un formulario similar al de inicio de sesión, pero esta vez tiene como objetivo el registro del usuario.

Si un usuario intenta hacer un registro con un nombre de usuario existente en la base de datos, el sistema notificará de este fallo al usuario.

Si un usuario intenta iniciar sesión con unas credenciales no válidas (ya sea nombre de usuario y/o contraseña no válidas), el sistema notificará de este fallo al usuario.

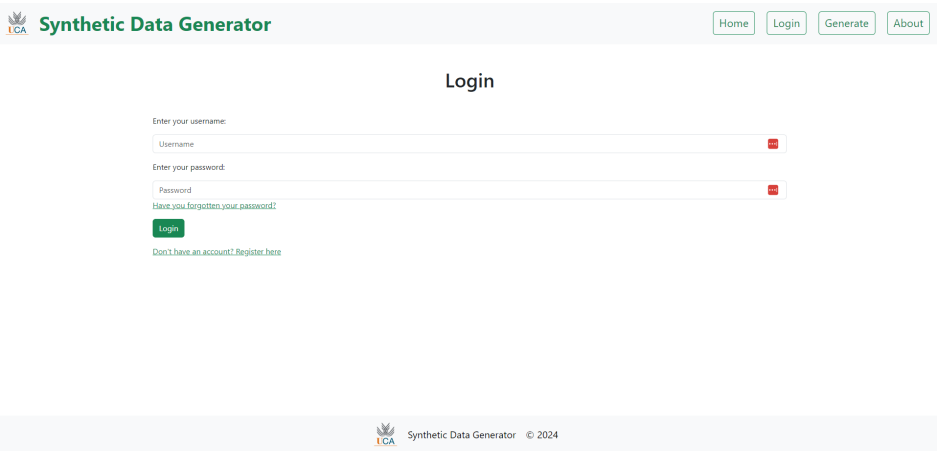

Figura 6.10: Página de inicio de sesión

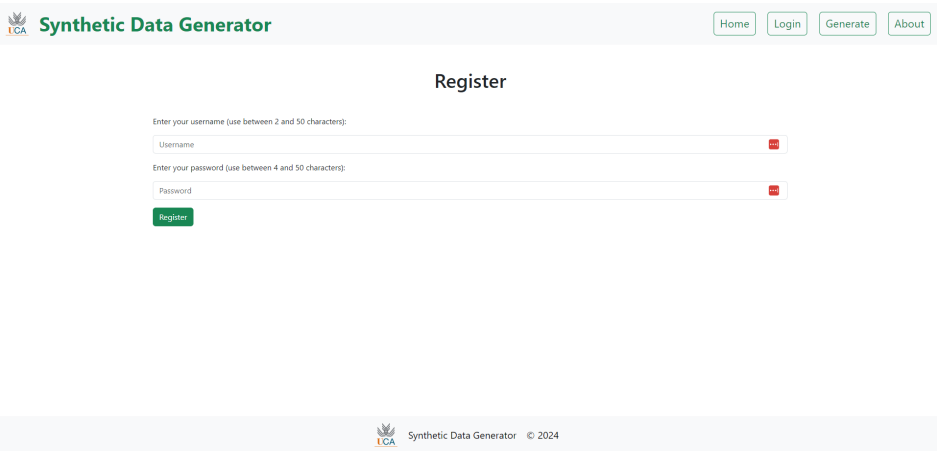

Figura 6.11: Página de Registro

6.4.6. Página de generación

Para poder acceder a la página de generación, el usuario debe haber iniciado sesión.

Usuario sin datasets subidos

Si el usuario se encuentra sin datasets subidos, el sistema mostrará un aviso indicando que suba uno para poder comenzar a generar datos sintéticos.

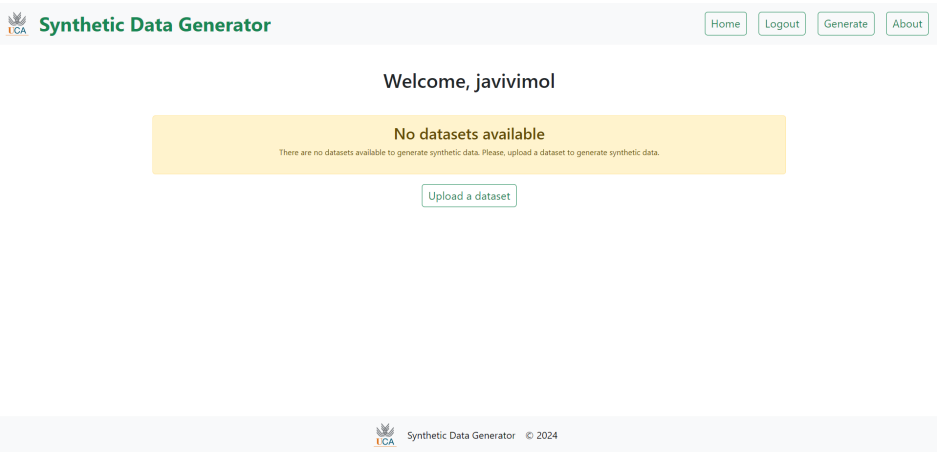

Figura 6.12: Página de generación - usuario sin datasets

Cuando el usuario desea subir un dataset, puede hacerlo haciendo click en el botón "Upload a dataset".

Haciendo esto, la página nos redirige a otra donde el usuario puede insertar un documento y subirlo al sistema.

Si el documento subido no tiene extensión CSV, la página mostrará un mensaje de error indicando al usuario que el documento debe ser un archivo CSV.

Si el documento subido tiene extension CSV, pero no posee cabezera (nombre de las columnas) o esta posee caracteres especiales, la página mostrará un mensaje indicando que el archivo no es válido dado a alguno de estos motivos.

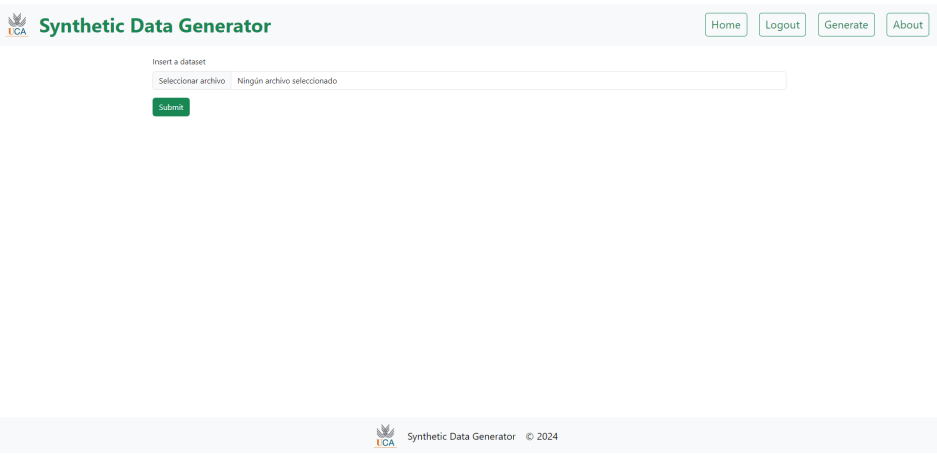

Figura 6.13: Página de subida de datasets

Usuario con datasets subidos

Si el usuario sube un documento con extensión CSV, el sistema nos redirigirá a la página de generación nuevamente; pero esta vez mostrará el dataset subido junto a las diferentes acciones que podemos hacer con este.

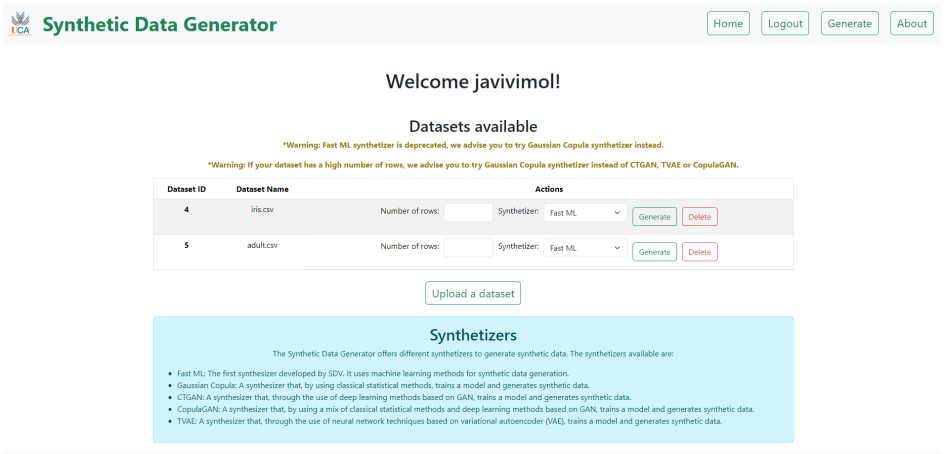

Figura 6.14: Página de generación - usuario con datasets

Cada sintetizador tiene una serie de características configurables. Seleccionando cada uno de ellos, el usuario es capaz de modificarlo según su preferencia.

Fast_ML

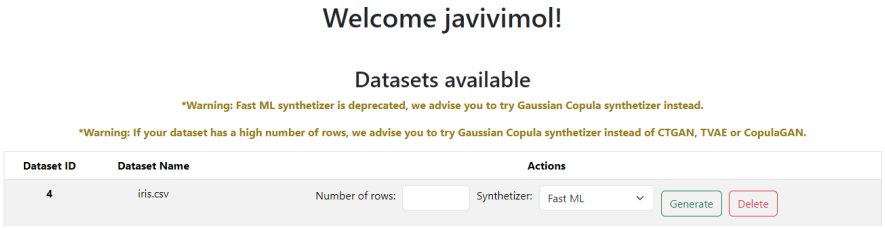

Figura 6.15: Opciones de configuración - Fast-ML

Gaussian Copula

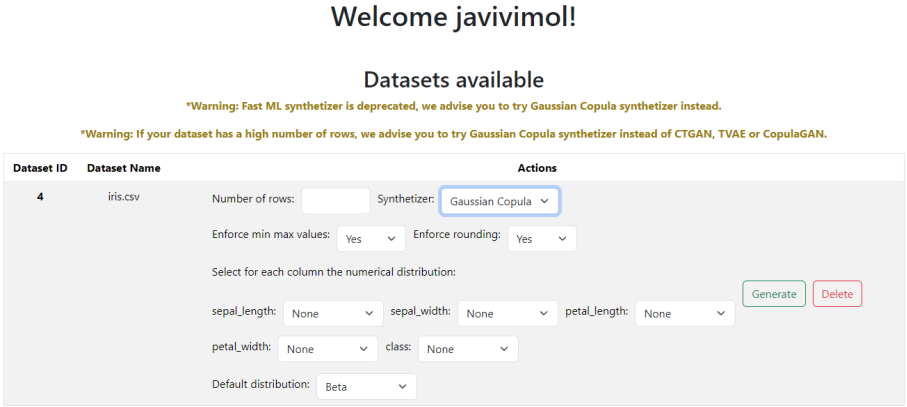

Figura 6.16: Opciones de configuración - Gaussian Copula

CTGAN

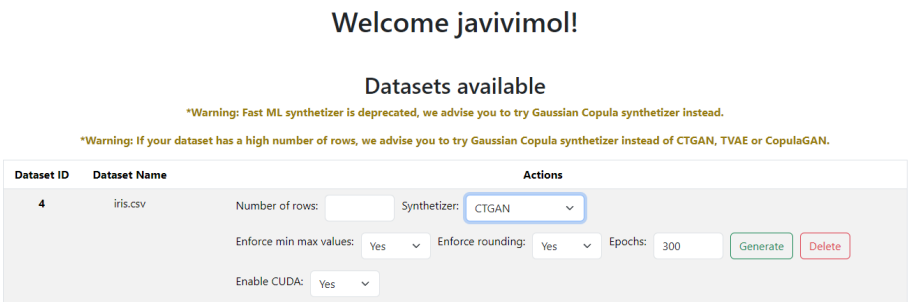

Figura 6.17: Opciones de configuración - CTGAN

TVAE

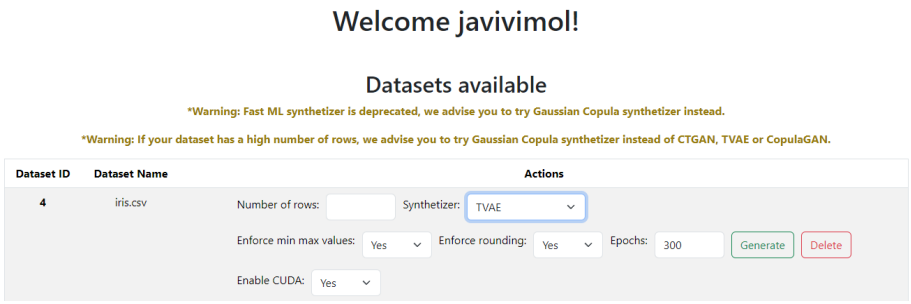

Figura 6.18: Opciones de configuración - TVAE

Copula GAN

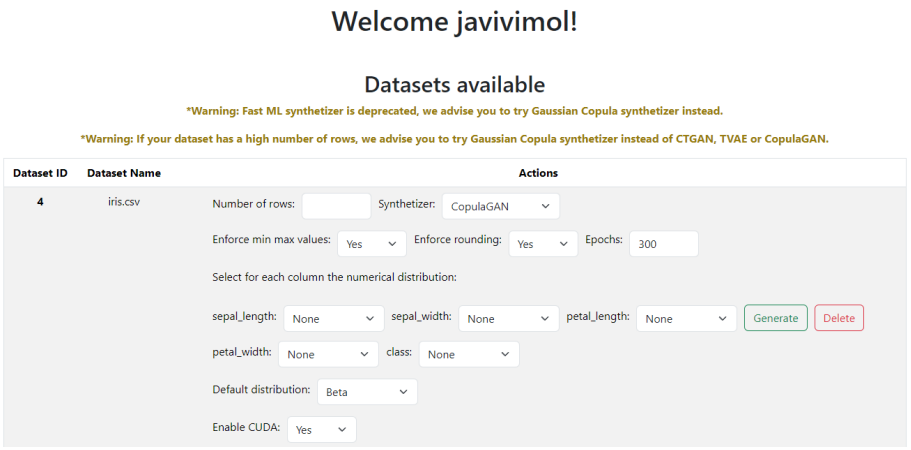

Figura 6.19: Opciones de configuración - Copula GAN

Manejo de errores

Si el usuario introduce un número de filas o etapas menor o igual a cero, se le notificará el error sin cambiarlo de página para que pueda continuar con la configuración del sintetizador.

Muestra de datos

Cuando el usuario elige un sintétizador y genera datos, el sistema mostrará una vista previa de las cinco primeras columnas de los datos originales y de los datos sintéticos generados. Además, se le dará la opción de descargar los datos o de evaluarlos.

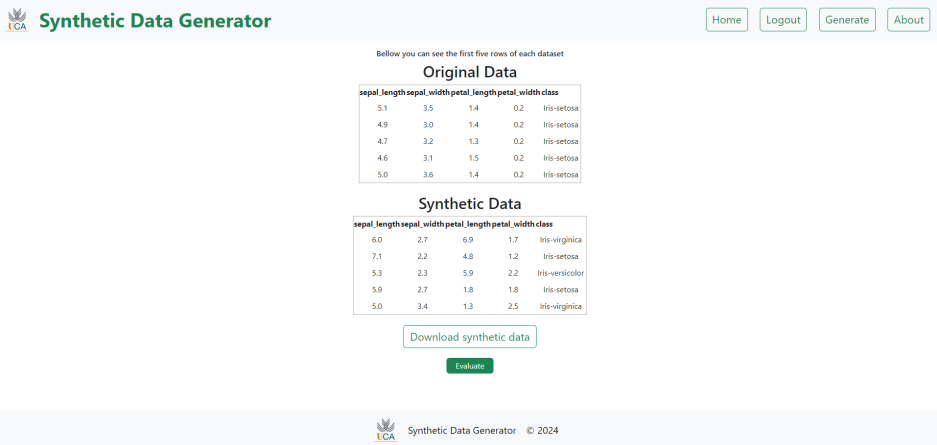

Figura 6.20: Página de muestra de datos

Evaluación de datos

Si el usuario presiona el botón para evaluar los datos, el sistema generará un informe. Este informe contendrá información estadística y gráficos, y con estos se podrá evaluar la calidad de los datos generados.

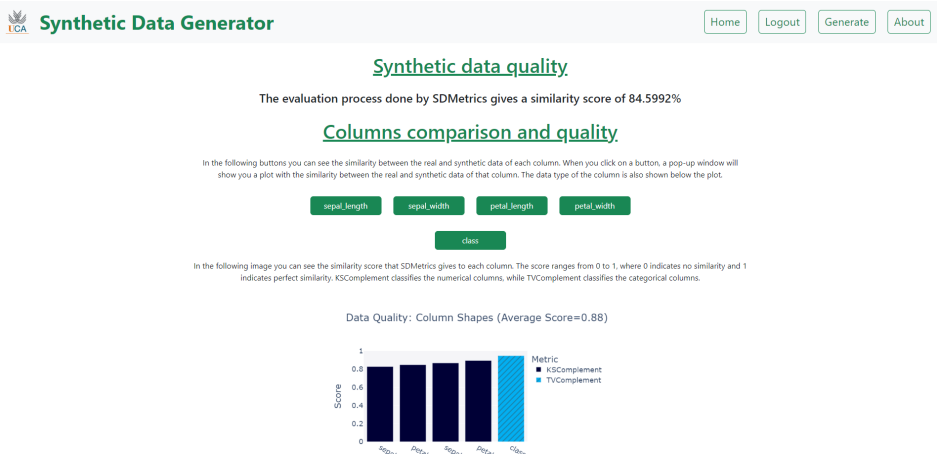

Figura 6.21: Página de evaluación

Puntuación de similitud

Lo primero que muestra el informe es una puntuación de similitud general que ofrece SDMetrics (librería para evaluar los datos sintéticos). Esta se expresa como porcentaje.

Figura 6.22: Puntuación de similitud

Comparación y similitud de columnas

Lo primero que mostramos son botones para cada columna, y cuando estos se pulsan se mostrará una información que dependerá de la clase a la que pertenece cada columna:

- Columnas numéricas: gráfico de distribución (frecuencias), tipo de dato, media y desviación estándar.

- Columnas discretas: gráfico de barras (frecuencias), tipo de dato y moda.

Además, se mostrará un gráfico de columnas donde se puede observar la puntuación de similitud para cada columna.

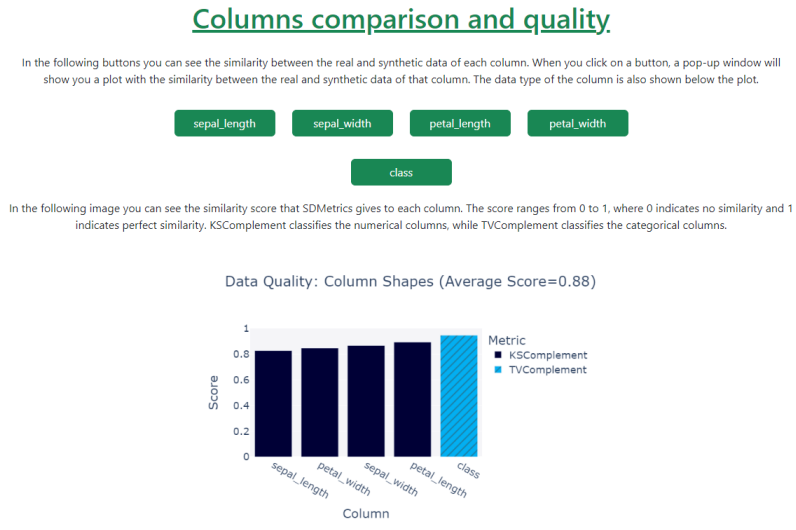

Figura 6.23: Comparación de columnas

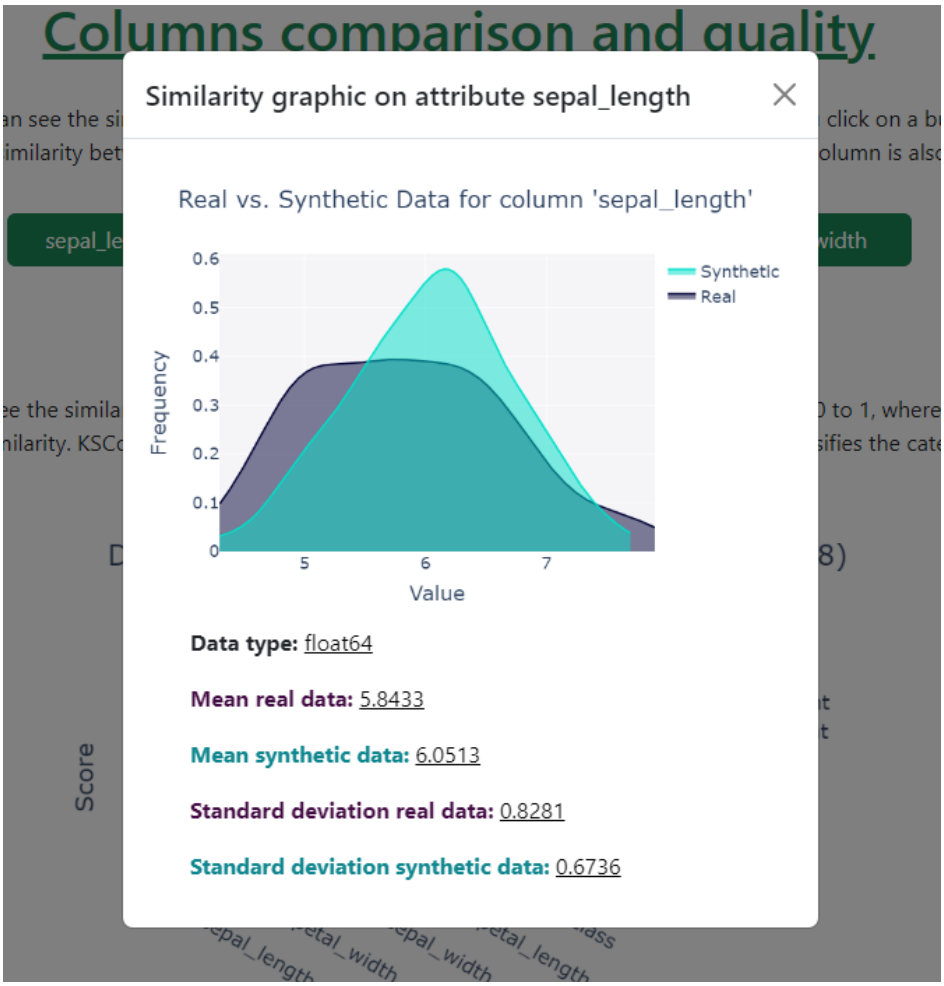

Figura 6.24: Información - columna numérica

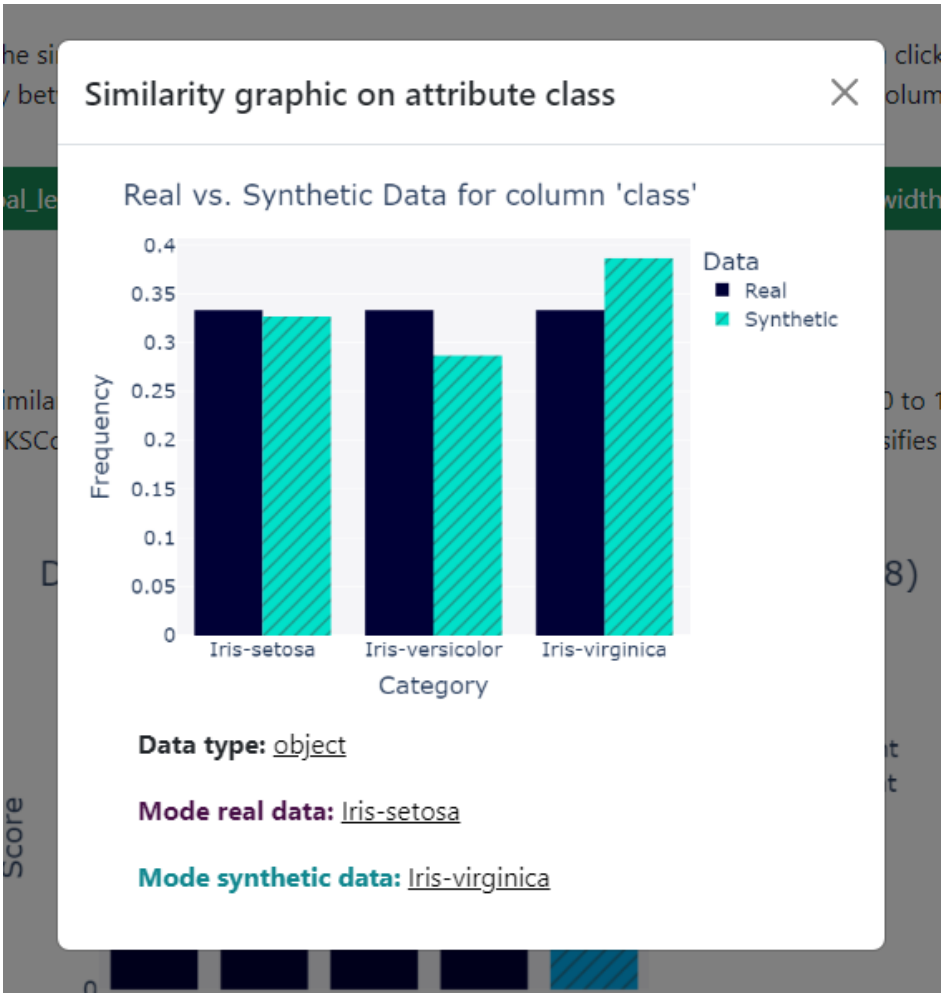

Figura 6.25: Información - columna discreta

Comparación entre pares de columnas numéricas

En la sección de covarianza, siguiendo el mismo formato anterior, mostraremos botones con los pares de columnas numéricas. Pulsando estos, podremos visualizar una gráfica de puntos (mostrando la dispersión de los datos) junto a la covarianza de las columnas.

Además, mostraremos la puntuación de similitud entre los datos originales y los sintéticos y la correlacion entre pares, usando una imagen con una matriz de tonalidades.

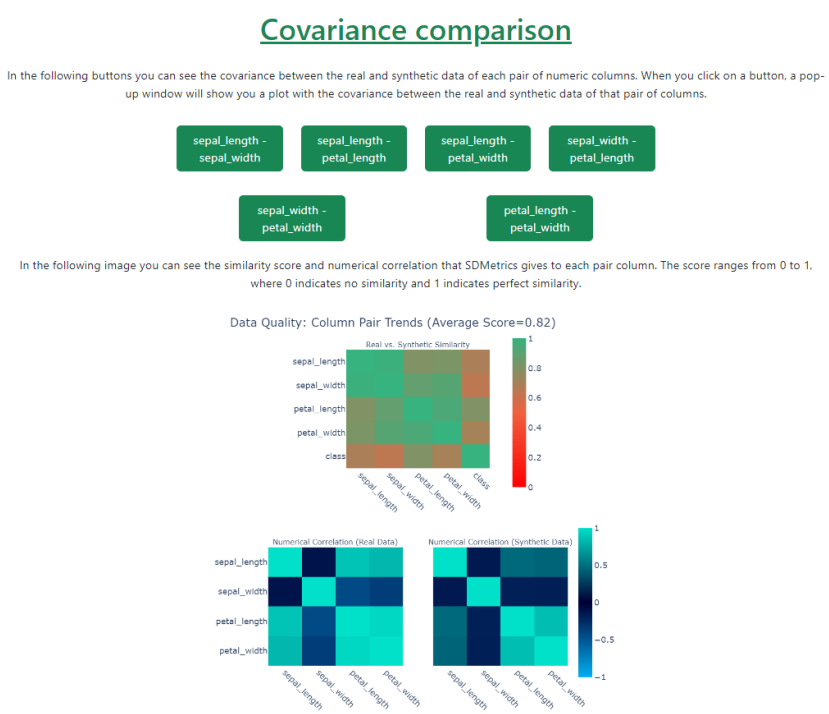

Figura 6.26: Comparación entre pares de columnas numéricas - covarianza

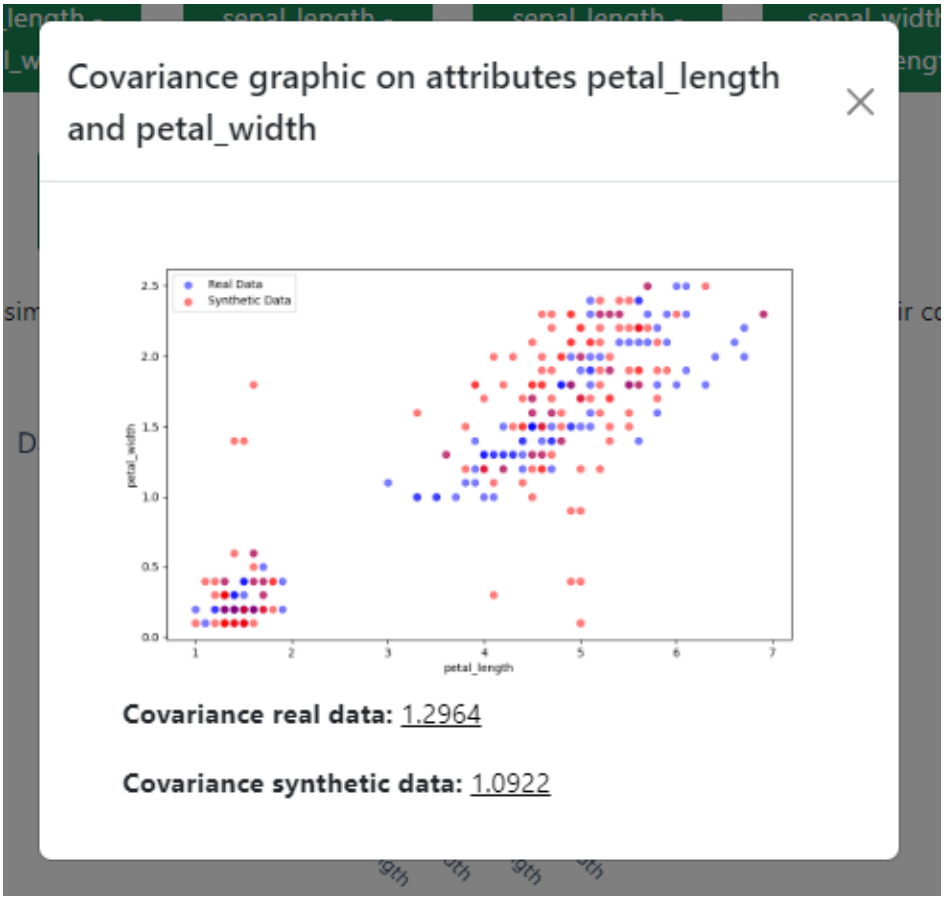

Figura 6.27: Información - covarianza

Por último, en la sección de regresión, expondremos la dependencia entre los pares de columnas. Para ello, mostraremos botones que al ser pulsados permitiran consultar: la gráfica de regresión, el coeficiente de Pearson, la recta formada, y el coeficiente de determinación.

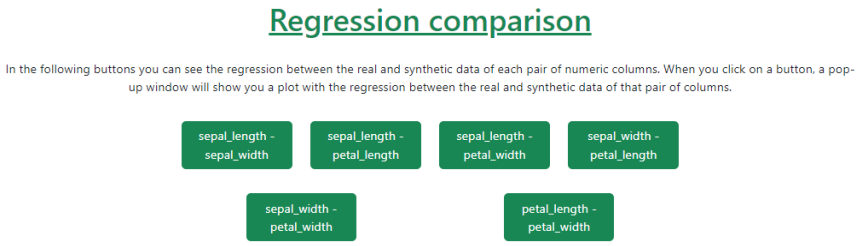

Figura 6.28: Comparación entre pares de columnas numéricas - regresión

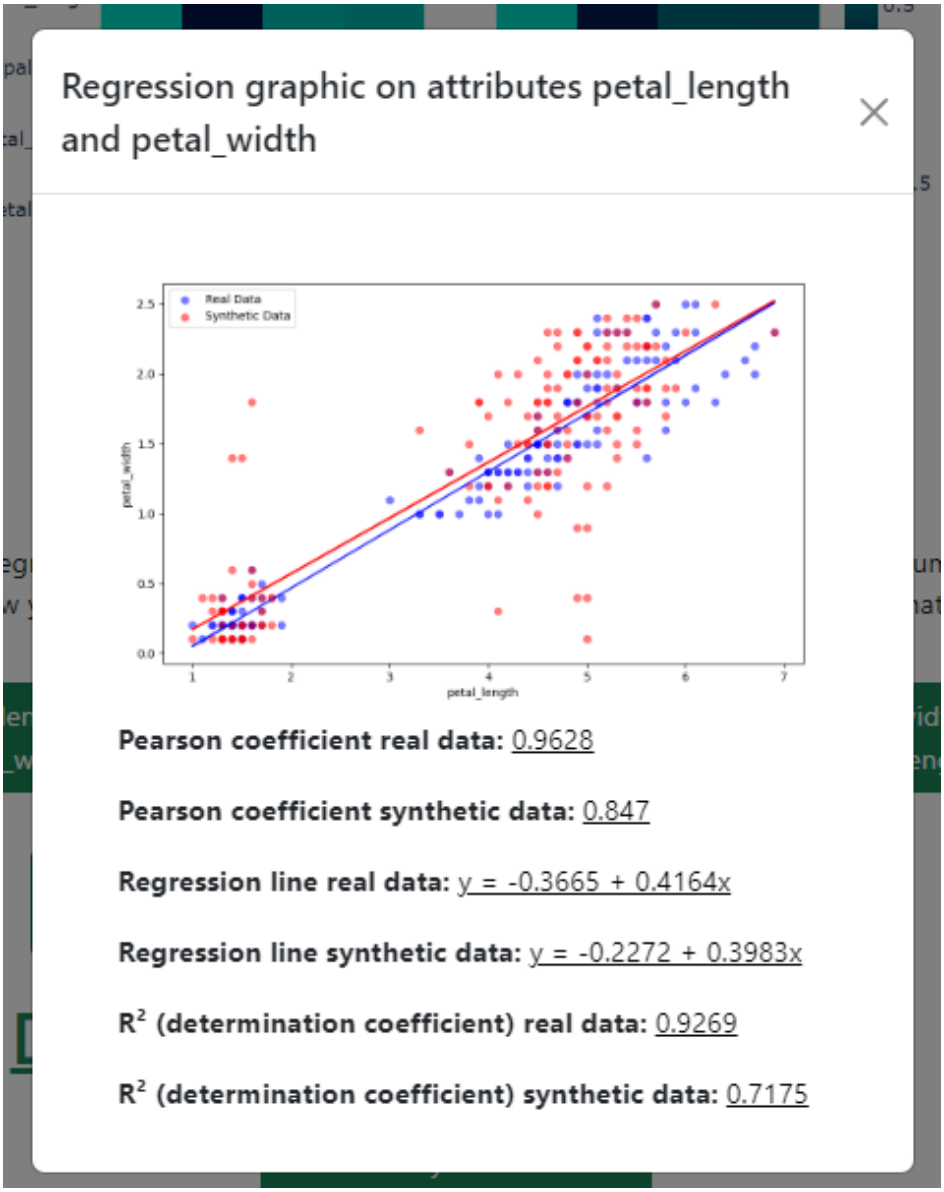

Figura 6.29: Información - regresión

Descarga y generación

La página de evaluación permitirá también descargar el dataset.

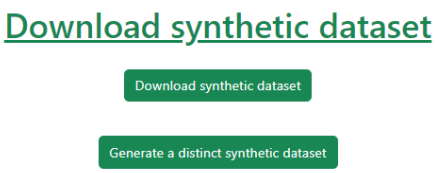

Figura 6.30: Sección de descarga

El resto de capturas que incluyen páginas de información adicional y errores, pueden ser consultadas en el anexo B.2

7. Construcción del Sistema

7.1. Entorno de Construcción

Para desarrollar el aplicativo web, el lenguaje principal ha sido Python debido a su flexibilidad, capacidades de uso y a que posee una gran cantidad de herramientas y librerías muy potentes.

El principal IDE que se ha usado para el desarrollo ha sido Pycharm. Este es un IDE con licencia de pago, pero gracias a ser estudiante de la Universidad de Cádiz, tuve acceso de forma gratuita.

Pycharm ofrece la posibilidad de asociar un proyecto a un repositorio de Github[1]. Esto facilita la sincronización con la nube y permite tener siempre una copia actualizada.

Por último, para la creación de los diagramas se han usado diferentes aplicaciones como draw.io[2] y visual paradigm online[3].

7.2. Código Fuente

El aplicativo web de Flask está contenido completamente en el directorio App. Dentro de este, podemos encontrar 7 ficheros de python.

Los ficheros representan el motor del sistema y cada uno tiene una función específica:

- main.py: es el fichero principal para ejecutar la aplicación. Depende del resto de ficheros para poder funcionar.
- app.py: fichero encargado de crear una instancia de la aplicación y base de datos, además de configurarla.
- config.py: fichero que contiene varias variables de configuración modificables por el usuario.
- forms.py: fichero que define los formularios que se usarán en la aplicación.
- models.py: fichero que define los modelos de la base de datos que serán usados.
- routes.py: fichero que define las rutas (URLs) a las que los usuarios pueden acceder. Es el encargado también de crear instancias de los sintetizadores, generar datos sintéticos con ellos y realizar informes de evaluación.

[1] Repositorio de GitHub - https://github.com/Javivi-MR/Synthetic-Data-Generator
[2] draw.io - https://app.diagrams.net/
[3] visual paradigm online - https://online.visual-paradigm.com/

- utils.py: fichero que contiene funciones usadas por el resto de ficheros.

Además de los ficheros, en el directorio *App* nos encontramos con 3 subdirectorios que tienen distintas funciones:

- instance: directorio en el que se almacena la base de datos SQLite. Si el directorio no la tuviera, el servidor la crearía y la almacenaría ahí.
- static: directorio en el que se almacenan diferentes archivos (generados por el servidor o no) que posteriormente serán usados por el aplicativo web. Dentro de este existen 5 subdirectorios:
  - css: directorio donde se almacenan diferentes hojas de estilo.
  - data: directorio donde se almacenan los datasets en CSV que suben los usuarios.
  - images: directorio donde se almacenan las imágenes estáticas que muestra el aplicativo.
  - plots: directorio donde se almacenan los gráficos que se generan automáticamente al evaluar los datos.
  - synthetic: directorio donde se almacenan los datasets sintéticos generados en CSV.
- templates: directorio donde se almacenan las diferentes plantillas que usa el servidor. Las plantillas son las siguientes:
  - index.html: plantilla de la página de inicio.
  - base.html: plantilla que contiene elementos comunes para todas las plantillas.
  - about.html: plantilla de la página de información acerca del proyecto.
  - error.html: plantilla que sirve para generar diferentes páginas de error.
  - evaluate.html: plantilla que genera las páginas de informes de evaluación.
  - forgot.html: plantilla de la página de "has olvidado tu contraseña".
  - generate.html: plantilla que genera las páginas que permiten la generación de datos sintéticos.
  - login.html: plantilla de la página de inicio de sesión.
  - register.html: plantilla de la página de registro de usuarios.
  - showdata.html: plantilla de la página de muestra de datos.
  - upload.html: plantilla para la página de subida de datasets.

7.2.1. Detalles de la hoja de estilos

La utilización del framework bootstrap[4] agilizó la construcción y decoración de las páginas webs. Además se usó sweetalert2[5] para embellecer las alertas.

7.2.2. Manejo de datasets y creación de gráficos

Para gestionar la información de los datasets de forma eficiente, se ha usado la clase Dataframe de la librería Pandas. Esta nos permite agilizar el proceso de carga y acceso a los datos de datasets en formato CSV, siendo idónea para el aplicativo.

Además, para la creación dinámica de gráficos se ha utilizado la librería Matplotlib. Esta permite crear diferentes tipos de gráficos estadísticos, que pueden ser útiles para determinar la usabilidad de un dataset sintético. SDV también integra funcionalidades con esta librería.

7.3. Scripts de Base de datos

Principalmente existe un script que gestiona dos aspectos diferentes relacionados a la base de datos. Este script siempre se ejecutará al iniciarse el sistema y se encuentra en el archivo *app.py*.

7.3.1. Creación de la base de datos

Si el sistema no posee un fichero *database.db* en el directorio *instance*, el script creará la base de datos. Esta acción automatiza la tarea de tener que inicializar la base de datos y crea las tablas con los datos definidos en el fichero *models.py*.

Esta acción ayuda a los usuarios que solo quieren usar la herramienta y no tienen experiencia con bases de datos.

7.3.2. Eliminación de información residual

Si el sistema detecta que existe un dataset que está asociado a un usuario inexistente o no se encuentra disponible en el sistema, eliminará la referencia de este en la base de datos.

El objetivo de esta acción es reducir lo máximo posible el coste en memoria que posee la base de datos.

[4]Bootstrap - https://getbootstrap.com/
[5]Sweetalert2 - https://sweetalert2.github.io/

8. Pruebas del Sistema

8.1. Estrategia

Hemos decidido seguir el enfoque de integración continua para la realización de pruebas en el software. Este método tiene como objetivo automatizar el proceso de testear el aplicativo cada vez que se realize cualquier tipo de actualización.

Para llevarlo a cabo, realizamos las siguientes acciones:

1. Programar las pruebas que el software debe pasar cada vez que lo actualicemos.
2. Subir todo el código junto a las pruebas a un repositorio en Github.
3. Crear un workflow junto a un fichero yml. En este programar que cada vez que se actualice el repositorio, se realicen las intrucciones necesarias para ejecutar el código de pruebas. Finalmente, añadir una instrucción para generar un Issue si se falla en alguna prueba.

Estas pruebas son ejecutables también desde un servidor local, usando puertos locales del equipo para alojar la aplicación de flask. Para ejecutar las pruebas en local, ejecuta primero *main.py*, para inciar el servidor y posteriormente ejecuta *tests.py*. Este archivo usa la librería unittest para facilitar la codificación de las pruebas.

Además de las anteriores pruebas, se testearán manualmente distintos aspectos de la aplicación para asegurar que sea accesible, eficiente, intuitiva y usable.

El dataset usado para las pruebas es *iris* (Fisher, 1988) del repositorio UC Irvine Machine Learning[1] y está disponible en el directorio *examples*.

8.2. Pruebas Unitarias

Las pruebas unitarias pueden ser observables dentro de la clase *TestApp*, los métodos con nombre *test_unit_x...*.

El objetivo de estas pruebas es testear funciones y funcionalidades individuales que el aplicativo usa, encontrando en el proceso posibles errores y faltas de optimización.

Las pruebas unitarias programadas y realizadas cumplen las siguientes funciones:

- Comprobar que el sistema construye correctamente los directorios indicados en la configuración.
- Comprobar si la base de datos permite la insersión de usuarios y datasets, además de la encriptación de la contraseña.

[1] UCI Machine Learning Repository - https://archive.ics.uci.edu/

- Comprobar si los cálculos (media, desviación típica...) que mostramos en la página de evaluación estadística, son correctos.
- Comprobar si los datasets generados cumplen con un mínimo de calidad.
- Comprobar si los datasets subidos tienen cabecera y usan como separador la coma.

8.3. Pruebas de Integración

Desde una etapa temprana de desarrollo se realizaron diferentes módulos y cada vez que se añadía uno, se verificaba que su integración con el resto fuera correcta y no produjese ningún tipo de incompatibilidad.

Esto permite detectar y localizar posibles errores en la implementación del aplicativo, haciendo que el software sea versátil y se ahorre tiempo en el desarrollo.

8.4. Pruebas de Sistema

Con el objetivo de comprobar si el sistema cumple con los requisitos establecidos en el capítulo 4, realizaremos distintas pruebas que exponemos en las siguientes subsecciones:

8.4.1. Pruebas Funcionales

Además de realizar interacciones manuales con el aplicativo, hemos querido automatizar estas pruebas utilizando la librería selenium[2]. Esta nos permite simular la interacción del usuario con el aplicativo web.

Los tests podemos observarlos en el fichero *tests.py*, en los métodos *test_func_x...* de la clase *TestApp*.

Las pruebas se han realizado según los flujos normales y alternativos explicados en la sección 5.2. La realización de ellas ha servido para comprobar que no existen flujos inesperados y corregirlos si se detectaban.

8.4.2. Pruebas No Funcionales

Para comprobar que el sistema es accesible y usable se han utilizado distintos navegadores, dispositivos y diferentes tamaños de ventanas para acceder y usarlo.

También, hemos comprobado que el sistema puede gestionar varias conexiones y sesiones que suceden simultáneamente, dando una respuesta rápida y eficiente.

[2]Selenium - https://www.selenium.dev/documentation/

Por último, hemos realizado distintas pruebas de edición de URLs y peticiones HTTP, para asegurar la integridad de la aplicación y para comprobar que no es posible acceder a la información que pertenece a otro usuario.

8.5. Pruebas de Aceptación

Para esta sección, realizamos la instalación del sistema en un servidor de producción final y lo pondremos a prueba. Para ello:

1. Creamos una cuenta gratuita de estudiantes en Microsoft Azure[3] y en esta, creamos un apartado para aplicación web.
2. Añadimos al proyecto un servidor de WSGI proporcionado por Gunicorn[4]
3. Creamos un Docker[5] para virtualizar el sistema.
4. Usamos Azure Client para subir y vincular nuestro Docker al aplicativo y dejamos que se inicie el sitema.
5. Una vez iniciado, realizamos las pruebas pero esta vez en el entorno de producción final.

[3]Microsoft Azure - https://azure.microsoft.com/es-es
[4]Gunicorn - https://gunicorn.org/
[5]Docker - https://www.docker.com/

9. Despliegue del Sistema

9.1. Arquitectura Física

La arquitectura física del sistema está formada por un servidor que aloja y ofrece el aplicativo web. El equipo podría ofrecer el servicio a todos los usuarios de la red, permitiendo conectarse desde una URL. Para un uso individual, se ofrecen dos posibles maneras de ejecutar el aplicativo:

1. Virtualización: el sistema del usuario debe tener instalada la aplicación Docker[1] y con esta construir el sistema.
2. Ejecución con Python: el sistema del usuario debe tener instalado python 3.9 junto a todas las dependencias disponibles en el fichero *requirements.txt*.

Los requisitos hardware del sistema aumentarán a medida que se incremente el número de usuarios, debido a que crece la exigencia y el número de peticiones. Recomendamos los siguientes componentes:

- Procesdor: Intel Xeon E-2314
- Memoria RAM: 16 GB.
- Memoria principal: SSD 1 TB.

Con el fin de que el proyecto pueda ser fácilmente probado, se ha utilizado el plan gratuito de Microsoft Azure para desplegar el aplicativo online de forma pública. Está disponible en el siguiente enlace `https://syntheticdatagen1.azurewebsites.net/`. Este servicio tiene recursos limitados, por lo que recomendamos usar pequeños datasets para ponerlo a prueba. Un ejemplo puede ser *iris* (Fisher, 1988) que posee 150 filas y 5 columnas.

9.2. Instrucciones de despliegue

En esta sección, se dictarán las instrucciones de instalación del sistema sobre la infraestructura física descrita anteriormente.

9.2.1. Virtualización

Para facilitar lo máximo posible la instalación del sistema a los usuarios más inexpertos, se ha implementado un contenedor Docker. Este contenedor virtualizará el aplicativo completamente.

[1] Docker - `https://www.docker.com/`

Para ello, hemos usado la herramienta Docker-Compose, que consiste en realizar la configuración del contenedor en dos archivos principales *Dockerfile* y *docker-compose.yml*

El fichero *Dockerfile* inicia el sistema con una imagen base de Python 3.9, para posteriormente instalar paquetes adicionales como *gcc* y *musl-dev*. Después, los archivos locales son copiados al contenedor y se instalan todas las dependencias del proyecto especificadas en *requirements.txt*, mediante el gestor de paquetes de python. Para finalizar, gracias al servidor WSGI que proporciona Gunicorn[2], el aplicativo web se ejecuta como servicio.

El fichero *docker-compose.yml* es un archivo de configuración y nos permite configurar los puertos a usar y asignar volúmenes.

No es necesaria la instalación de dependencias para la base de datos SQLite, debido a que la librería SQLAlchemy ya proporciona métodos para la manipulación de esta.

Para poder construir y lanzar el sistema, solo es necesario tener instalado Docker en tu dispositivo.

9.2.2. Ejecución con Python

Para ejecutar el sistema con python, deberás poseer la version 3.9 instalada en tu dispositivo y, además, deberás instalar todos las dependencias disponibles en *requirements.txt*. Por último, ejecutando el fichero *main.py*, te permitirá acceder al aplicativo desde localhost.

9.2.3. Requisitos previos

Para los usuarios del sistema se sugiere el uso de navegadores web actuales. Se recomienda el uso de las siguientes versiones:

- Google Chrome 100 o superior.
- Mozilla Firefox 100 o superior.
- Microsoft Edge 98 o superior.
- Opera 80 o superior.
- Safari 13 o superior.

Para la realizar la instalación del sistema, se requiere de:

- Python 3.9 Debe contar con las dependencias mencionadas en el anexo E.
- Servidor que permita desplegar aplicaciones WSGI. Se recomienda Gunicorn.
- Docker 19 o superior.

[2]Gunicorn - https://gunicorn.org/

9.2.4. Inventario de componentes

El proyecto incluye los siguientes componentes:

- Aplicativo web para la generación de datasets sintéticos.
- Documento de requisitos con todas las dependencias del aplicativo.
- Servidor WSGI de Gunicorn.

9.2.5. Procedimientos de instalación

Para poder clonar el aplicativo puedes seguir este enlace: https://github.com/Javivi-MR/Synthetic-Data-Generator.

Para realizar la instalación por docker, solo debes situarte con una consola en el directorio raíz del proyecto y realizar el siguiente comando: *docker compose up*. Con esto se iniciará el proceso de virtualización y en breves minutos podrás acceder al sistema.

Para realizar la instalación completa, debes instalar python 3.9 junto a todas las dependencias. Para hacerlo, después de instalar python, sitúate en el directorio raiz del proyecto y ejecuta el siguiente comando: *pip install -r requirements.txt*. Por último, dirígete al fichero *main.py* y ejecútalo. Podrás acceder al aplicativo por localhost.

9.2.6. Pruebas de implantación

Con el objetivo de verificar que el sistema se ha instalado correctamente, realiza las siguientes acciones:

1. Accede al aplicativo mediante el uso de un navegador web.
2. Navega a la sección de Login y posteriormente selecciona la opción de registrarme.
3. Rellena el formulario con un nombre de usuario y una contraseña. Después de esto serás redirigido a la sección de Login.
4. En la sección de Login, introduce tu nombre de usuario y contraseña. Comprueba que se te redirige a la sección principal de la web y ahora el botón de login se ha convertido en uno de Logout.
5. Dirígete a la sección de Generate y sube un dataset con formato CSV, con cabecera y separado por comas. Comprueba que has vuelto a la sección Generate y ahora aparece.
6. Genera datos sintéticos usando cualquiera de los sintetizadores.
7. Descarga los datos generados.
8. Evalua los datos y comprueba que los gráficos se generan de forma correcta.

9. Vuelve a la sección Generate y comprueba que no puedes generar datos si introduces números negativos en la sección de filas o etapas.
10. Vuelve a la sección Generate y borra el dataset subido. Comprueba que vuelves a estar sin datasets.
11. Vuelve a subir un dataset, pero esta vez, selecciona un dataset que no tenga cabecera. Comprueba que te muestra error.
12. Vuelve a subir un dataset, pero esta vez, selecciona un dataset que no use la coma de separador. Comprueba que te muestra error.

Si ejecutas las anteriores acciones y todas se cumplen, el sistema se ha instalado de forma correcta y puede ser usado con normalidad.

9.3. Instrucciones para la operación del sistema y mantenimiento del nivel de servicio

En cuanto a política de backups, es recomendable realizar una copia de seguridad de la base de datos con cierta frecuencia. Todo sistema es vulnerable ante ataques informáticos y para poder recuperar el servicio lo antes posible y minimizar las pérdidas de datos, es recomendable realizar backups de la base de datos periódicamente.

La información (con la configuración inicial del proyecto) se almacena en el fichero *database.db*, dentro de los directorios */App/Instance*. Nuestra recomendación es automatizar la creación de una copia de seguridad completa semanal y una copia incremental acumulativa diaria.

Parte III

Epílogo

10. Conclusiones

10.1. Objetivos alcanzados

Se ha conseguido crear un aplicativo web que permite: el registro de usuarios, iniciar sesión, subir datasets en formato CSV (con cabecera), usar los sintetizadores que ofrece SDV para generar datos sintéticos y poder evaluar la calidad de estos. El aplicativo ofrece una plataforma que permite conocer y experimentar con los datos sintéticos. Este ha sido diseñado para usuarios de todos los niveles de experiencia.

Hemos hecho que el aplicativo sea sencillo de instalar y usar, permitiendo que, las personas que quieran probarlo puedan hacerlo a través de Microsoft Azure y las que quieran instalarlo en su sistema solo tengan que seguir unos breves pasos.

Por último, se ha logrado que la interfaz del sistema sea simple, usable, accesible e intuitiva.

10.2. Lecciones aprendidas

Desde el inicio del proyecto, tuve que investigar y recopilar información acerca de la actual problemática de privacidad. He conseguido aprender sobre las distintas soluciones que se han propuesto al problema y, en especial, he profundizado mis conocimientos en la solución de datos sintéticos. Esta solución a pesar de no ser la *bala de plata*[1] que elimine el problema, es una de las más prometedoras y puede representar un paso hacia el camino de la solución definitiva.

En el desarrollo del proyecto, he aprendido a utilizar diferentes tecnologías. Destacando algunas de ellas, Flask[2] es el framework que me ha permitido construir la aplicación y gestionar las solicitudes REST; SDV[3] es una de las librerías más conocidas para la generación de datos sintéticos y, gracias a esta, he podido usar los sintetizadores que ofrece, para permitir a los usuarios generar datos sintéticos.

Finalmente, en el despliegue del aplicativo, he conseguido virtualizar el sistema utilizando Docker[4] y, además, he aprendido a desplegar este en un servidor de producción de Microsoft Azure[5].

[1] Bala de plata o en inglés, Silver bullet - Solución definitiva a un problema
[2] Flask - https://flask.palletsprojects.com/en/3.0.x/
[3] Synthetic Data Vault - https://docs.sdv.dev/sdv
[4] Docker - https://www.docker.com/
[5] Microsoft Azure - https://azure.microsoft.com/es-es

10.3. Trabajo futuro

El proyecto tiene diversas áreas que pueden ser mejoradas. Algunas pueden ser las siguientes:

- Una posible mejora del sistema de subida de datasets sería permitir al usuario subir datasets utilizando un enlace que redirigiese a un fichero CSV con cabecera.
- El actual sistema permite generar datos sintéticos para datasets en formato CSV, es decir, generar datos de tablas individuales. Sin embargo, SDV ofrece sintetizadores para múltiples tablas que poseen relaciones, como en una base de datos relacional. Sería interesante ampliar la funcionalidad del aplicativo añadiendo la posibilidad de poder generar datos sintéticos de múltiples tablas.
- En relación con el anterior punto, SDV ofrece también sintetizadores para tablas individuales pero con información secuencial, como por ejemplo, la recogida de datos de sensores. Añadir esto al aplicativo haría que tuviese una mayor funcionalidad.
- Existen técnicas de generación de datos sintéticos que SDV no abarca. Por ejemplo, tenenemos los modelos CART[6], redes neuronales recurrentes y redes bayesianas (Pathare et al., 2023). Podríamos buscar librerías en python que ofrezcan las anteriores técnicas y añadirlas al aplicativo.
- Varios estudios indican que el uso de redes GAN[7] pueden revelar información de los datos con las que fueron entrenadas implícitamente (Xie et al., 2018) y (Jordon y Yoon, 2019). Estos proponen como solución el uso de una red DPGAN[8]. Creo que añadir esta tecnología al proyecto haría que la calidad de privacidad de los datos generados fuera más robusta.

[6] Árboles de clasificación y regresión.
[7] GAN - Generative Adversarial Network
[8] DPGAN - Differentially Private Generative Adversarial Network

Información sobre Licencia

Información sobre Licencia

"*Privatización y Usabilidad de Datos Sintéticos: Un análisis práctico*.[es] un software de código abierto distribuido bajo la Licencia Pública General de GNU versión 3.0[9] .

Esto significa que deben cumplirse los siguientes derechos y obligaciones:

- Tienes derecho a usar, copiar, modificar, fusionar, publicar y ditribuir este software
- Si realizas una modificación del software, debe de estar distribuida bajo la licencia GPL - 3.0
- Si distribuyes este software, debes proporcionar acceso al código fuente completamente.

[9]GPL - 3.0 - https://www.gnu.org/licenses/gpl-3.0.html

Bibliografía

Bellovin, S. M., Dutta, P. K., y Reitinger, N. (2019). Privacy and synthetic datasets. 22:1.

Dwork, C. (2006). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., y Wegener, I., editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg. Springer Berlin Heidelberg.

El Emam, K., . D. F. K. (2008). Protecting privacy using k-anonymity. *Journal of the American Medical Informatics Association*, 15:627–637.

Erfanian Ebadi, S., Dhakad, S., Vishwakarma, S., Wang, C., Jhang, Y.-C., Chociej, M., Crespi, A., Thaman, A., y Ganguly, S. (2022). Psp-hdri+: A synthetic dataset generator for pre-training of human-centric computer vision models. In *First Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022*.

Farayola, O. A., Olorunfemi, O. L., y Shoetan, P. O. (2024). Data privacy and security in it: A review of techniques and challenges. *Computer Science amp; IT Research Journal*, 5(3):606–615.

Fisher, R. A. (1988). Iris. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C56C76.

Gröger, C. (2021). There is no ai without data. *Commun. ACM*, 64(11):98–108.

IBM watsonx (2023). `https://www.ibm.com/watsonx`.

Jordon, J., Szpruch, L., Houssiau, F., Bottarelli, M., Cherubin, G., Maple, C., Cohen, S. N., y Weller, A. (2022). Synthetic data – what, why and how?

Jordon, J. y Yoon, J. (2019). PATE-GAN: GENERATING SYNTHETIC DATA WITH DIFFERENTIAL PRIVACY GUARANTEES.

Mostly AI (2017). `https://mostly.ai/`.

Pathare, A., Mangrulkar, R., Suvarna, K., Parekh, A., Thakur, G., y Gawade, A. (2023). Comparison of tabular synthetic data generation techniques using propensity and cluster log metric. *International Journal of Information Management Data Insights*, 3(2):100177.

Patki, N., Wedge, R., y Veeramachaneni, K. (2016). The synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410.

UCA (2022). Tabla de presupuestos capítulo vi.

Xie, L., Lin, K., Wang, S., Wang, F., y Zhou, J. (2018). Differentially private generative adversarial network.

Parte IV

Anexos

A. Manual de usuario

A.1. Introducción

Este aplicativo permite a los usuarios subir datasets en formato CSV con cabecera. A partir de estos, el usuario puede usar y configurar diferentes sintetizadores para generar datos sintéticos. Posteriormente, el usuario podrá evaluar estos datos para comprobar su usabilidad estadística.

A.2. Instalación

Para probar el software no es necesario realizar ninguna descarga o instalación, simplemente acceder al enlace disponible en la sección 9.1. Desde ahí el usuario podrá usar el sistema sin limitaciones.

En el caso de querer instalar y desplegar el aplicativo de forma local, el usuario tendría que seguir las instrucciones de la sección 9.2. Para realizar la virtualización, el usuario deberá tener la aplicación docker instalada y configurada. Los ficheros que construyen el contenedor son los siguientes:

Dockerfile:

```
FROM python:3.9

RUN apt-get update && apt-get install -y \
    gcc \
    musl-dev

COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt

COPY . .

EXPOSE 50505

WORKDIR /App

ENTRYPOINT ["gunicorn", "-b", "0.0.0.0:50505", "app:app"]
```

docker-compose.yml:

```
version: '3.9'

services:
```

```
  web:
    build: .
    ports:
      - "50505:50505"
    volumes:
      - .:/code
```

A.3. Uso del sistema

En esta sección expondremos las acciones que podemos realizar en el sistema:

A.3.1. Inicio del sistema

Al acceder a la aplicación te situarás en el inicio del sistema. En él verás una breve portada junto a preguntas y respuestas acerca de los datos sintéticos.

Usando la barra superior podrás acceder a la sección principal nuevamente, a la sección de inicio de sesión, a la sección de generar datos (solamente si has iniciado sesión) y a la sección de acerca del proyecto.

A.3.2. Inicio de sesión y registro

Al acceder a la sección de inicio de sesión, visualizarás un formulario donde podrás introducir tus credenciales para acceder al sistema.

En el caso de que el usuario no tenga cuenta en el sistema, deberá dirigirse a la sección de registro justo abajo del formulario. En esta, podrá introducir sus datos para darse de alta en el sistema y ahora sí, podrá usar el inicio de sesión.

A.3.3. Generar

Cuando el usuario ha iniciado sesión, puede acceder a la sección de generar. En esta sección los usuarios podrán visualizar los datasets que han subido (en el caso de no tener subido ninguno, le aparecerá un mensaje indicándole que suba uno para poder usar el sistema) y podrán generar datasets sintéticos configurando el sintetizador que les aparece a la derecha.

A.3.4. Subir dataset

Al presionar el botón *Upload*, los usuarios visualizarán en pantalla un formulario que pide un documento. El dataset subido deberá estar en formato CSV, poseer cabezeras y los nombres de las columnas no pueden contener caracteres especiales (a excepción de los puntos (.), los guiones bajos (_) y los guiones (-)).

A.3.5. Muestra de datos

Al configurar un sintetizador, introducir un número de filas y pulsar el botón *Generate*, el usuario podrá visualizar por pantalla una muestra de 5 filas de los datos originales y los datos sintéticos. Además, podrá descargar el dataset sintético generado y evaluar la calidad de estos datos.

A.3.6. Evaluación de datos

Al presionar el botón *Evaluate*, el sistema generará un informe que mostrará la calidad de los datos. Este informe contendrá la siguiente información:

- Puntuación de similitud global.
- Gráficos de distribución y estadísticas de cada columna.
- Gráficos con la puntuación de similitud de cada columna.
- Gráficos de dispersión y covarianza de los pares de columnas numéricas.
- Matrices de tonalidades, muestran la puntuación de similitud entre pares y la correlación numerica.
- Gráficos de regresión y información relativa de los pares de columnas numéricas.

Por último, se permitirá al usuario descargar los datos generados y volver a generar otro dataset.

B. Diagramas de diseño

B.1. Diagramas del modelo C4

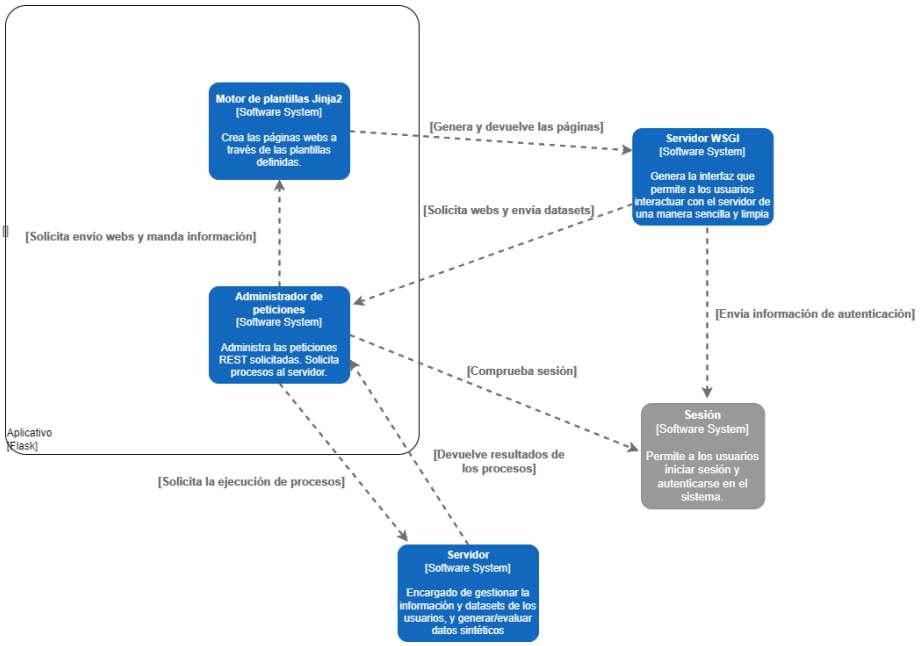

Figura B.1: Diagrama de contexto - Aplicativo Flask

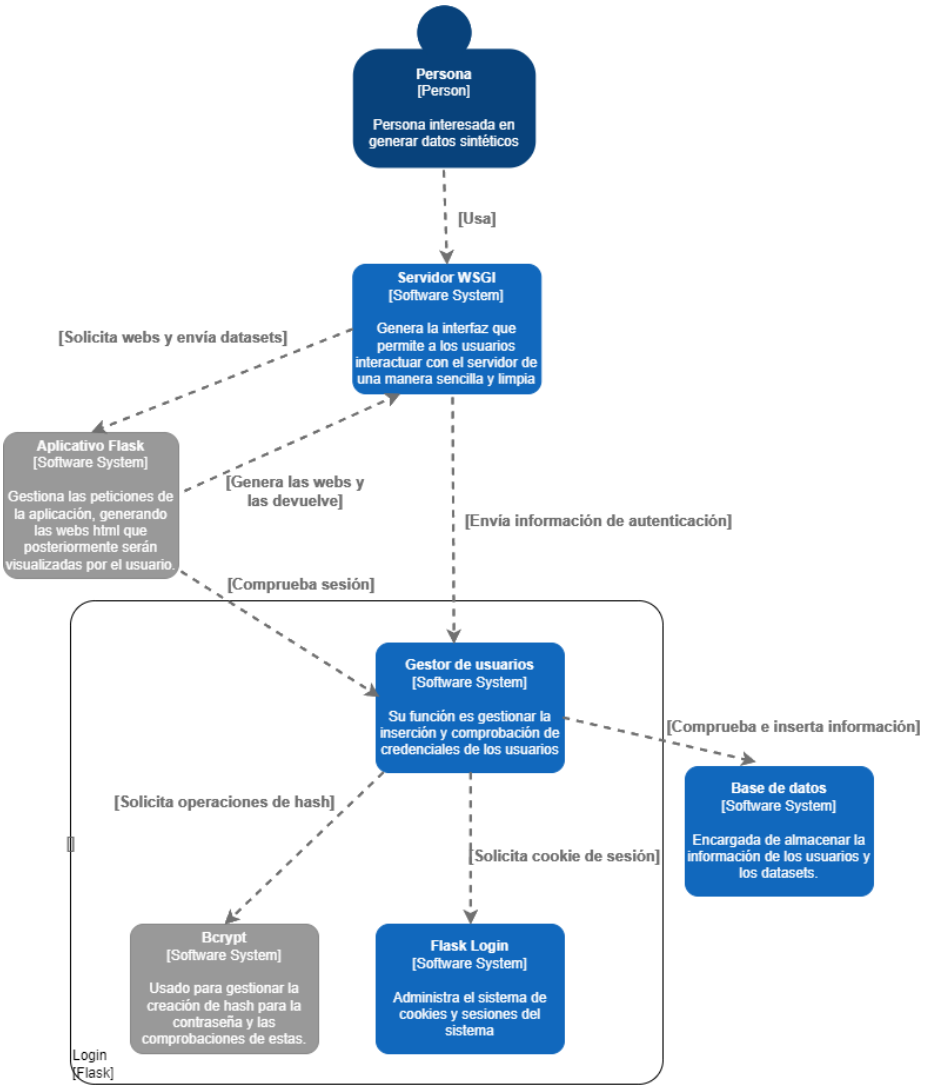

Figura B.2: Diagrama de contexto - Módulo de sesión

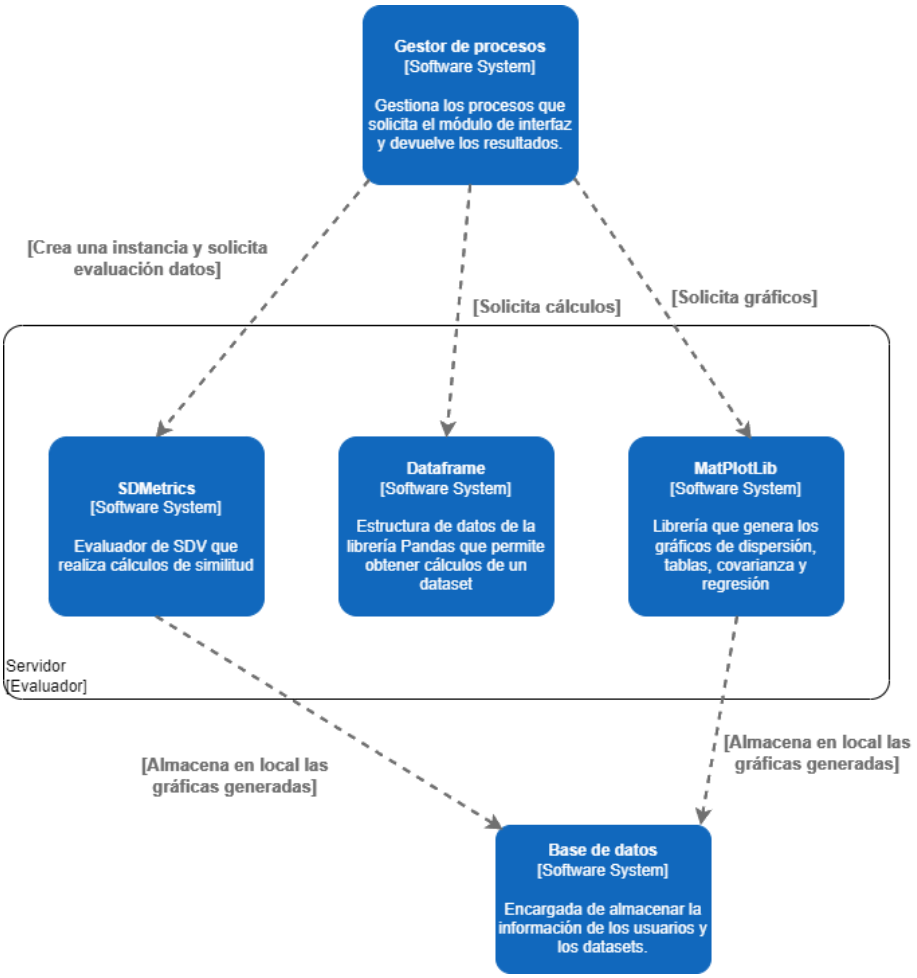

Figura B.3: Diagrama de contexto - Módulo de evaluador

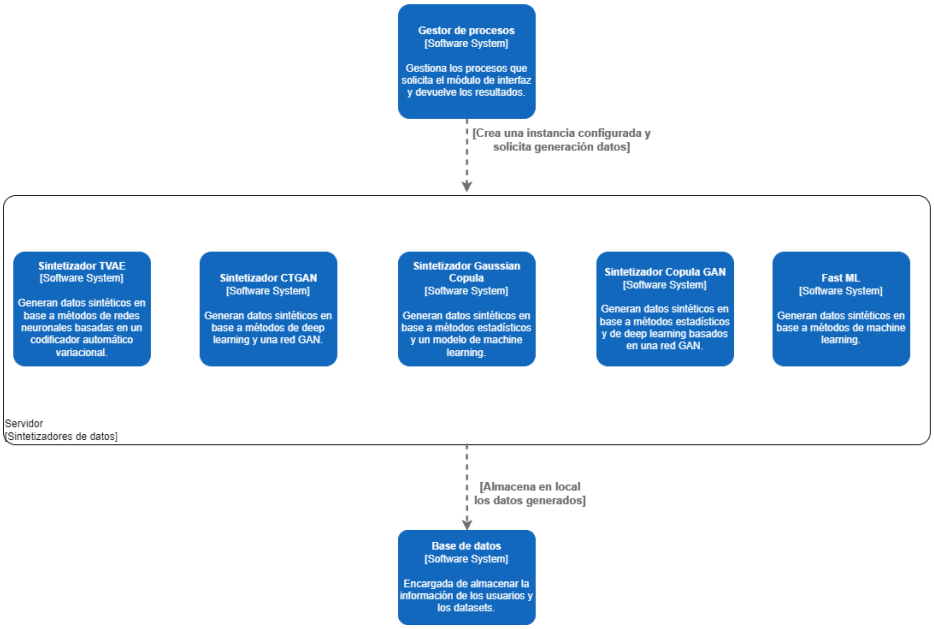

Figura B.4: Diagrama de contexto - Módulo de sintetizadores

B.2. Páginas del sistema

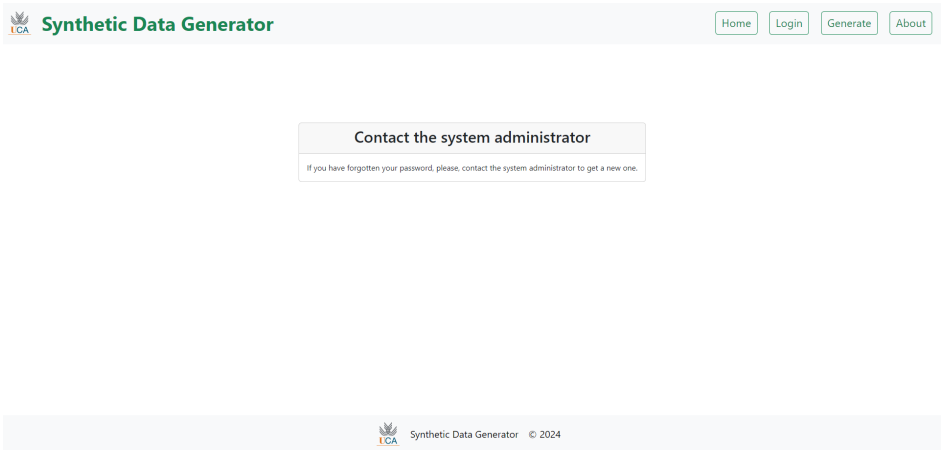

Figura B.5: Página de has olvidado tu contraseña

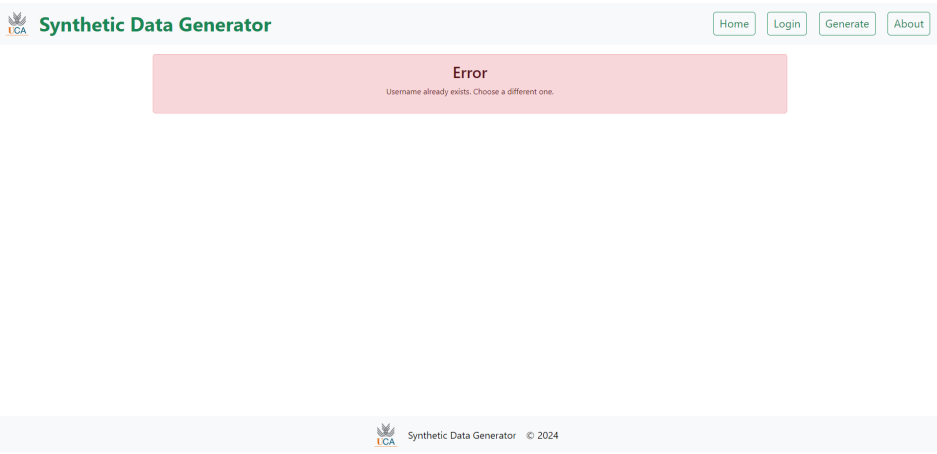

Figura B.6: Error: nombre de usuario ya registrado

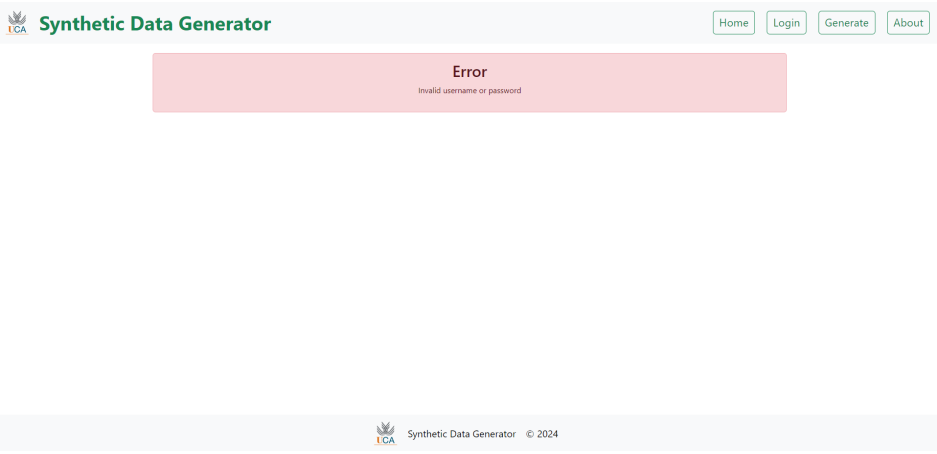

Figura B.7: Error: nombre de usuario y/o contraseñas inválidos

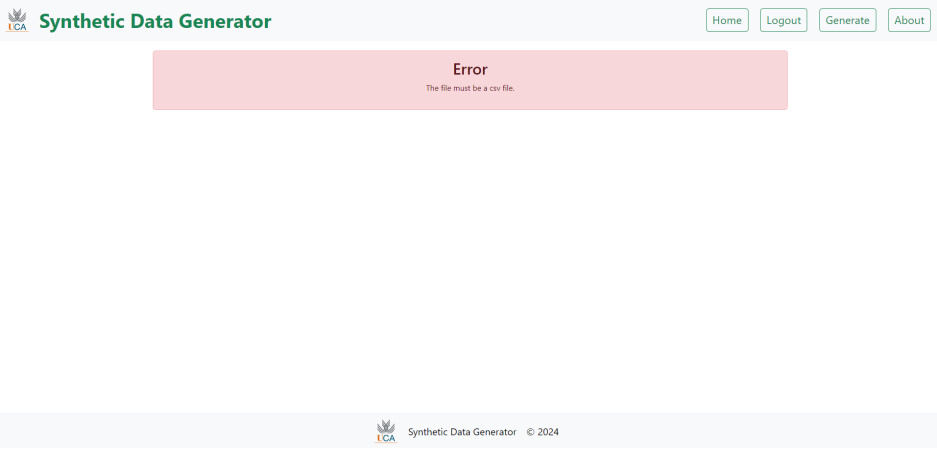

Figura B.8: Error: Tipo de archivo inválido

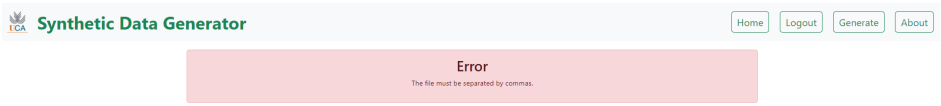

Figura B.9: Error: Fichero CSV sin separadores ’,’.

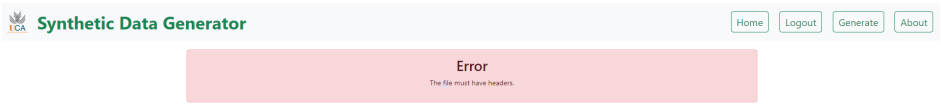

Figura B.10: Error: Fichero CSV sin cabecera.

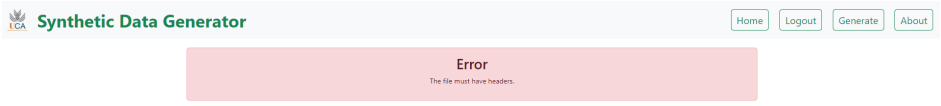

Figura B.11: Error: Fichero CSV con caracteres especiales en los nombres de las columnas.

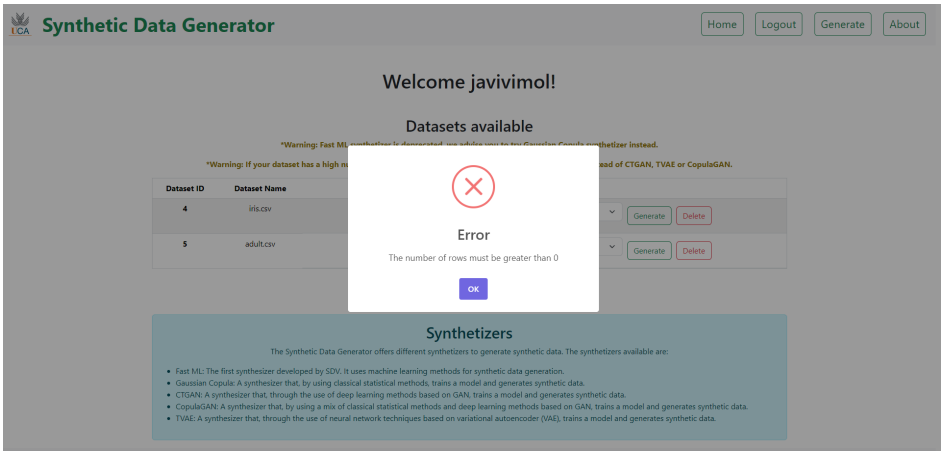

Figura B.12: Error: Número de filas o etapas negativo

C. Diagramas de casos de uso

C.1. Registro de usuarios

Nombre del CU:	Registrarse
Creado por:	Francisco Javier Molina Rojas
Fecha	11/05/2024
Descripción	Para poder generar datos, el usuario tiene que iniciar sesión en el sistema. Si no tiene una cuenta, debe registrarse para poder iniciar sesión.
Actores	Usuario y sistema.
Precondiciones	1. El usuario no se encuentra registrado en el sistema.
Postcondiciones	1. El usuario queda registrado en el sistema
Flujo	1. El usuario accede al apartado Login utilizando la barra superior. 2. El usuario hace clic en la frase "Dont have an account? Register here." 3. El usuario introduce un nombre de usuario y una contraseña. El 4. sistema recoge los datos del usuario, verifica que no exista un usuario con el mismo nombre en el sistema, crea un hash con su contraseña y registra en la base de datos al usuario junto a su contraseña con hash. 5. El sistema devuelve al usuario a la pantalla de Login para que pueda iniciar sesión.
Flujo alternativo	4a. En el paso 4, el sistema detecta que existe un usuario registrado con el mismo nombre de usuario. 5. El sistema devuelve al usuario una pantalla de error donde se le comunica que ya existe un usuario registrado con ese nombre.
Excepciones	4b. En el paso 4, el usuario cierra la aplicación. 6. El sistema no registrará los datos del usuario
Requisitos	Para poder realizar este caso de uso se tiene que cumplir la siguiente condición: 1. El sistema debe tener suficiente espacio en la base de datos.

Figura C.1: Caso de uso Registrarse en el sistema

C.2. Iniciar sesión

Nombre del CU:	Iniciar sesión
Creado por: Fecha	Francisco Javier Molina Rojas 11/05/2024
Descripción	Cuando un usuario ha realizado el registro en el sistema, pueden iniciar sesión para poder usarlo.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema.
Postcondiciones	1. La sesión del usuario se inicia.
Flujo	1. El usuario hace click en la sección de Login de la barra principal. 2. El usuario rellena con sus datos el formulario de inicio de sesión. 3. El usuario hace click en el botón de "Login". 4. El sistema verifica que el usuario exista y que el hash de la contraseña introducida coincide con la almacenada. 5. El sistema proporciona la cookie de inicio de sesión al usuario y lo redirige a la página principal.
Flujo alternativo	2a. En el paso 2, el usuario introduce uno o ambos datos de forma errónea (usuario o contraseña). 3. El usuario hace click en el botón de "Login". 4. El sistema comprueba que el usuario o contraseña son incorrectos y muestra al usuario el error por pantalla.
Excepciones	4a. En el paso 4, el usuario cierra la aplicación. 5. El sistema no generará los datos sintéticos.
Requisitos	Para poder realizar este caso de uso se tiene que cumplir la siguiente condición: 1. El usuario debe estar registrado en el sistema.

Figura C.2: Caso de uso Iniciar sesión en el sistema

C.3. Eliminar dataset

Nombre del CU:	Eliminar dataset subido
Creado por: Fecha	Francisco Javier Molina Rojas 11/05/2024
Descripción	Cuando un usuario está registrado y ha subido al menos un dataset, puede eliminar datasets.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. El usuario debe tener al menos un dataset subido al sistema. 3. El sistema está iniciado.
Postcondiciones	1. El sistema elimina los datos asociados al dataset que el usuario desea eliminar
Flujo	1. El usuario, hace click en la sección "Generate" de la barra superior. 2. El usuario decide que dataset quiere eliminar. 3. El usuario hace click en el botón "Delete" asociado a dicho dataset. 4. El sistema suprime la referencia de dicho dataset en la base de datos y lo elimina del almacenamiento del sistema.
Flujo alternativo	-
Excepciones	-
Requisitos	Para poder realizar este caso de uso se tienen que cumplir las siguientes condiciones: 1. El usuario debe estar registrado en el sistema. 2. El usuario debe haber subido al menos un dataset.

Figura C.3: Caso de uso Eliminar dataset del sistema

C.4. Descargar datos

Nombre del CU:	Descargar datos sintéticos
Creado por:	Francisco Javier Molina Rojas
Fecha	12/05/2024
Descripción	Cuando un usuario ha generado los datos o ha realizado la evaluación de estos, el sistema permite descargarlos.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. El usuario debe haber subido al menos un dataset al sistema. El 3. usuario debe haber generado datos sintéticos y debe situarse en la página de muestra de datos o en la de evaluación.
Postcondiciones	1. El usuario descarga los datos generados.
Flujo	1. El usuario hace click en el botón de “Download”.
Flujo alternativo	-
Excepciones	-
Requisitos	Para poder realizar este caso de uso se tiene que cumplir la siguiente condición: 1. El sistema debe tener el espacio suficiente.

Figura C.4: Caso de uso Descargar datos sintéticos

C.5. Evaluar datos

Nombre del CU:	Evaluar datos sintéticos
Creado por:	Francisco Javier Molina Rojas
Fecha	13/05/2024
Descripción	Cuando un usuario ha generado los datos, el sistema le permite evaluar la calidad estadística de estos.
Actores	Usuario y sistema.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. El usuario debe haber subido al menos un dataset al sistema. El 3. usuario debe haber generado datos sintéticos y debe situarse en la página de muestra de datos.
Postcondiciones	1. El usuario puede visualizar un informe donde se muestra la calidad de los datos generados.
Flujo	1. El usuario hace click en el botón de “Evaluate”. 2. El sistema genera un informe de evaluación, genera los gráficos pertinentes y redirige al usuario a la página de informe.
Flujo alternativo	-
Excepciones	1a. Después de pulsar el botón, el usuario cierra la aplicación 2. El sistema no generará el informe.
Requisitos	Para poder realizar este caso de uso se tienen que cumplir las siguientes condiciones: 1. El sistema debe tener el espacio suficiente. 2. El usuario debe haber generado datos sintéticos y debe situarse en la página de muestra de datos.

Figura C.5: Caso de uso Evaluar datos sintéticos

D. Diagramas de comportamiento

D.1. Registrar usuario

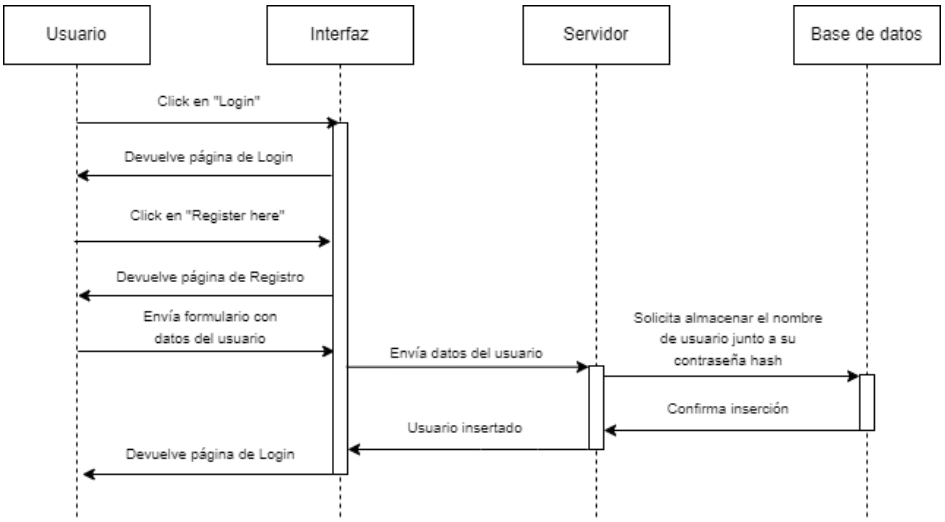

Figura D.1: Diagrama de secuencia de Registro de usuario

D.2. Iniciar sesión

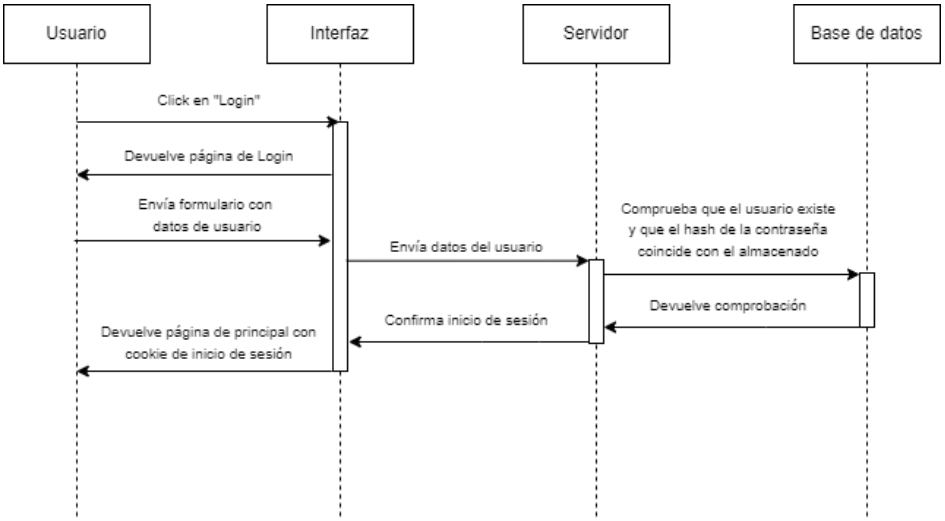

Figura D.2: Diagrama de secuencia de inicio de sesión

D.3. Eliminar dataset del sistema

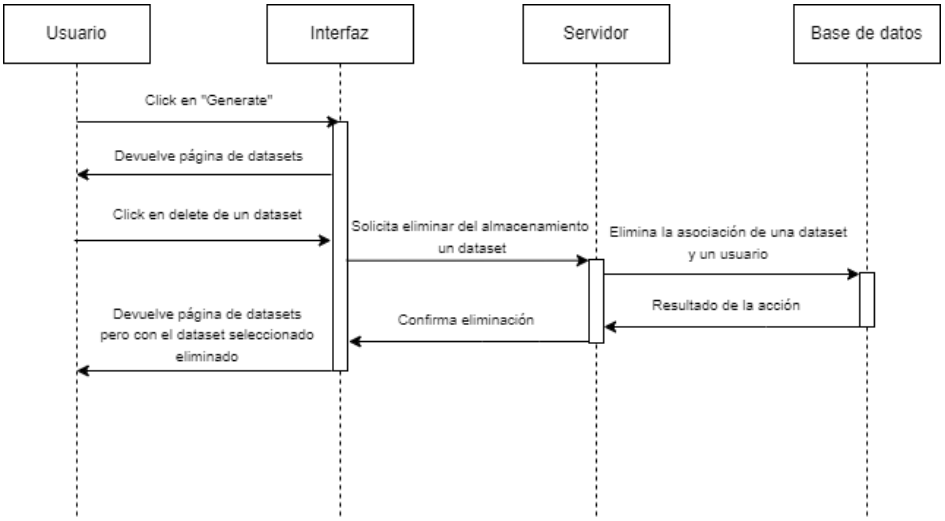

Figura D.3: Diagrama de secuencia de eliminar dataset del sistema

D.4. Descargar datos sintéticos

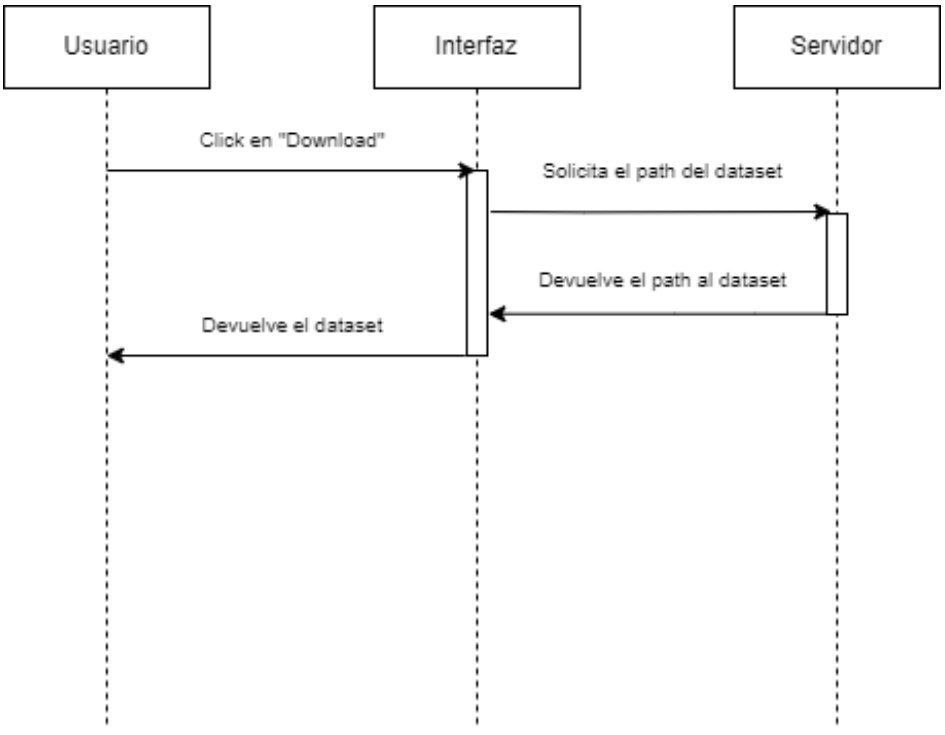

Figura D.4: Diagrama de secuencia de descargar datos sintéticos

D.5. Evaluar datos sintéticos

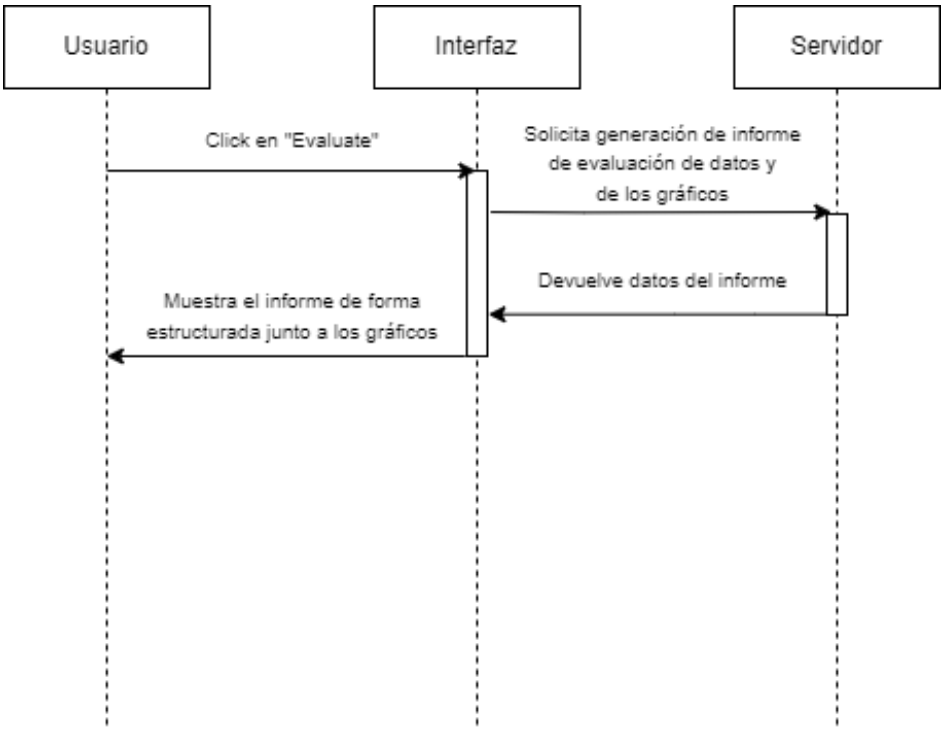

Figura D.5: Diagrama de secuencia de Evaluar datos sintéticos

E. Dependencias del aplicativo

Tabla E.1

Tabla de dependencias I

Librería	Compatibilidad	Versión
attrs	exacta	23.2.0
bcrypt	exacta	3.2.0
blinker	exacta	1.7.0
boto3	exacta	1.34.45
botocore	exacta	1.34.45
certifi	exacta	2024.6.2
cffi	exacta	1.16.0
charset-normalizer	exacta	3.3.2
click	exacta	8.1.7
cloudpickle	exacta	2.2.1
colorama	exacta	0.4.6
contourpy	exacta	1.2.0
copulas	exacta	0.9.2
ctgan	exacta	0.9.0
cycler	exacta	0.12.1
deepecho	exacta	0.5.0
exceptiongroup	exacta	1.2.1
Faker	exacta	19.13.0
filelock	exacta	3.13.1
Flask	exacta	3.0.2
Flask-Bcrypt	exacta	1.0.1
Flask-Login	exacta	0.6.3
Flask-SQLAlchemy	exacta	3.1.1
Flask-WTF	exacta	1.2.1
fonttools	exacta	4.49.0
fsspec	exacta	2024.2.0
graphviz	exacta	0.20.1
greenlet	exacta	3.0.3
gunicorn	exacta	21.2.0
h11	exacta	0.14.0

Tabla E.2

Tabla de dependencias II

Librería	Compatibilidad	Versión
idna	exacta	3.7
importlib-metadata	exacta	7.0.1
importlib-resources	exacta	6.1.1
itsdangerous	exacta	2.1.2
Jinja2	exacta	3.1.3
jmespath	exacta	1.0.1
joblib	exacta	1.3.2
kaleido	exacta	0.2.1
kiwisolver	exacta	1.4.5
MarkupSafe	exacta	2.1.5
matplotlib	exacta	3.8.3
mpmath	exacta	1.3.0
networkx	exacta	3.2.1
numpy	exacta	1.26.4
outcome	exacta	1.3.0.post0
packaging	exacta	23.2
pandas	exacta	2.2.0
pillow	exacta	10.2.0
plotly	exacta	5.19.0
pycparser	exacta	2.21
pyparsing	exacta	3.1.1
PySocks	exacta	1.7.1
python-dateutil	exacta	2.8.2
python-dotenv	exacta	1.0.1
pytz	exacta	2024.1
rdt	exacta	1.9.2
requests	exacta	2.32.3
s3transfer	exacta	0.10.0
scikit-learn	exacta	1.4.1.post1
scipy	exacta	1.12.0

Tabla E.3

Tabla de dependencias III

Librería	Compatibilidad	Versión
sdmetrics	exacta	0.13.0
sdv	exacta	1.10.0
selenium	exacta	4.21.0
six	exacta	1.16.0
sniffio	exacta	1.3.1
sortedcontainers	exacta	2.4.0
SQLAlchemy	exacta	2.0.27
sympy	exacta	1.12
tenacity	exacta	8.2.3
threadpoolctl	exacta	3.3.0
torch	exacta	2.2.0
tqdm	exacta	4.66.2
trio	exacta	0.25.1
trio-websocket	exacta	0.11.1
typing_extensions	exacta	4.9.0
tzdata	exacta	2024.1
urllib3	exacta	1.26.18
webdriver-manager	exacta	4.0.1
Werkzeug	exacta	3.0.1
wsproto	exacta	1.2.0
WTForms	exacta	3.1.2
zipp	exacta	3.17.0