



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Optimización de Riego
Deficitario Controlado en
cultivos Leñosos Mediante
Algoritmos de Machine
Learning**



Presentado por Rodrigo Castroviejo Ausucua
en Universidad de Burgos — 13 de febrero
de 2026

Tutor: Carlos Cambra Baseca
Co-tutora: Antonia Maiara Marques Do
Nascimento

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Estructura de Epics	2
A.3. Planificación temporal	9
A.4. Estudio de viabilidad	27
Apéndice B Especificación de Requisitos	35
B.1. Introducción	35
B.2. Objetivos generales	35
B.3. Catálogo de requisitos	36
B.4. Especificación de requisitos	37
Apéndice C Especificación de diseño	45
C.1. Introducción	45
C.2. Diseño de datos	46
C.3. Diseño arquitectónico	54
C.4. Diseño procedimental	61
Apéndice D Documentación técnica de programación	67
D.1. Introducción	67
D.2. Estructura de directorios	67

D.3. Manual del programador	69
Apéndice E Documentación de usuario	81
E.1. Introducción	81
E.2. Requisitos de usuarios	81
E.3. Instalación	81
E.4. Manual del usuario	82
Apéndice F Anexo de sostenibilización curricular	99
F.1. Introducción y Justificación	99
F.2. Contribución a los Objetivos de Desarrollo Sostenible (ODS)	100
F.3. Reflexión Personal y Compromiso Ético	101
F.4. Conclusión	101
Bibliografía	103

Índice de figuras

A.1. Actividades que comprende el Epic: Importar datos	2
A.2. Actividades que comprende el Epic: EDA inicial	3
A.3. Actividades que comprende el Epic: EDA selección features	4
A.4. Actividades que comprende el Epic: Creación y uso de modelos	5
A.5. Actividades que comprende el Epic: Creación base aplicación web	6
A.6. Actividades que comprende el Epic: Cálculo necesidad hídrica cultivo	7
A.7. Actividades que comprende el Epic: Generación datos sintéticos	8
A.8. Actividades que comprende el Epic: Implementación predicción web	9
A.9. Tablero Sprint 1	10
A.10. Tablero Sprint 2	11
A.11. Tablero Sprint 3	13
A.12. Tablero Sprint 4	16
A.13. Tablero Sprint 5	18
A.14. Tablero Sprint 6	19
A.15. Tablero Sprint 7	21
A.16. Tablero Sprint 8	24
A.17. Número de commits durante el desarrollo del proyecto	27
C.1. Implementación de la clase User en Python.	47
C.2. Implementación de la clase Dataset en Python.	48
C.3. Diagrama de la base de datos realacional	49
C.4. Estructura carpeta migraciones	50
C.5. Dataframe Mini1 combinado anualmente	51
C.6. Dataframe Mini2 combinado anualmente	52
C.7. Dataframe de clima combinado anualmente	52
C.8. Dataframe combinado original	52

C.9. Dataframe diario combinado e interpolado	53
C.10. Dataframe Mini1 combinado anualmente	53
C.11. Dataframe final con datos sintéticos físicos agregados	54
C.12. Estructura funcionamiento de Docker	55
C.13. Esquema específico de mis contenedores y su comunicación	56
C.14. Separación estructura entrenamiento/datos	57
C.15. Estructura API REST aplicada a mi aplicación	58
C.16. Ejemplo de implementación de SARIMA siguiendo la estructura propuesta	60
C.17. Diagrama de secuencia de proceso de entrenamiento	62
C.18. Diagrama de secuencia de proceso de Registro de un usuario . .	63
C.19. Diagrama de secuencia de proceso de subida de un dataset . .	64
C.20. Diagrama de secuencia de proceso de descarga de predicciones .	65
D.1. Índice de contenidos EDA.ipynb	75
D.2. Clase de configuración de parámetros de entrenamiento	76
D.3. Implementación de sonarQube a través de GitHub Actions . . .	79
D.4. Dashboard de sonarQube del repositorio del proyecto	80
E.1. Botón de ejecutar todas las celdas	82
E.2. Botón de reiniciar sesión y ejecutar todas las celdas	83
E.3. Pantalla principal de la aplicación	84
E.4. Pantalla registro	85
E.5. Pantalla principal de la aplicación	86
E.6. Pantalla inicio de sesión	87
E.7. Pantalla principal de la aplicación	88
E.8. Menú desplegable del perfil del usuario	88
E.9. Pantalla principal de la aplicación	89
E.10. Menú desplegable	90
E.11. Pantalla de subida de datasets	90
E.12. Pantalla principal de la aplicación	92
E.13. Menú desplegable	92
E.14. Selección de datos de entrenamiento	93
E.15. Selección de modelos a entrenar	93
E.16. Selección de parámetros avanzados de entrenamiento	94
E.17. Pop-up de confirmación de entrenamiento	95
E.18. Consola entrenamiento y opciones sobre esta	97

Índice de tablas

A.1. Resumen de costes de hardware y consumo energético.	30
A.2. Coste del software utilizado en el proyecto	30
A.3. Desglose de costes de infraestructura y servicios.	31
A.4. Resumen de licencias de software de terceros.	32
B.1. CU-1 Registro de usuario.	38
B.2. CU-2 Inicio de sesión de usuario.	39
B.3. CU-3 Cierre de sesión de usuario.	40
B.4. CU-4 Subir archivos de datos.	41
B.5. CU-5 Configurar y Entrenar Modelo.	42
B.6. CU-6 Ejecutar Predicción RDC.	43
B.7. CU-7 Descargar Resultados.	44
D.1. Mapeo de rutas de autenticación.	77
D.2. Flujo técnico del módulo de predicción.	78

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este plan se detalla la planificación temporal, las metas de cada sprint y los estudios de viabilidad económica y legal que sustentan el proyecto.

El objetivo principal de este proyecto es optimizar el rendimiento económico del cultivo del almendro mediante un sistema de riego deficitario controlado, enmarcado en la Agricultura 5.0 [1]. Utilizando datos de sensores en campo y algoritmos de minería de datos desarrollados en Python, se predice con precisión las necesidades hídricas para reducir el consumo de agua sin mermar significativamente la producción. Los resultados y la operativa del modelo se integran en una aplicación web intuitiva que facilita la visualización, el entrenamiento de modelos y la realización de predicciones, promoviendo así una gestión del agua más eficiente, sostenible y rentable.

Además, se abordan aspectos fundamentales como la viabilidad económica y la viabilidad legal, asegurando el cumplimiento de normativas y la correcta gestión de la propiedad intelectual y datos personales.

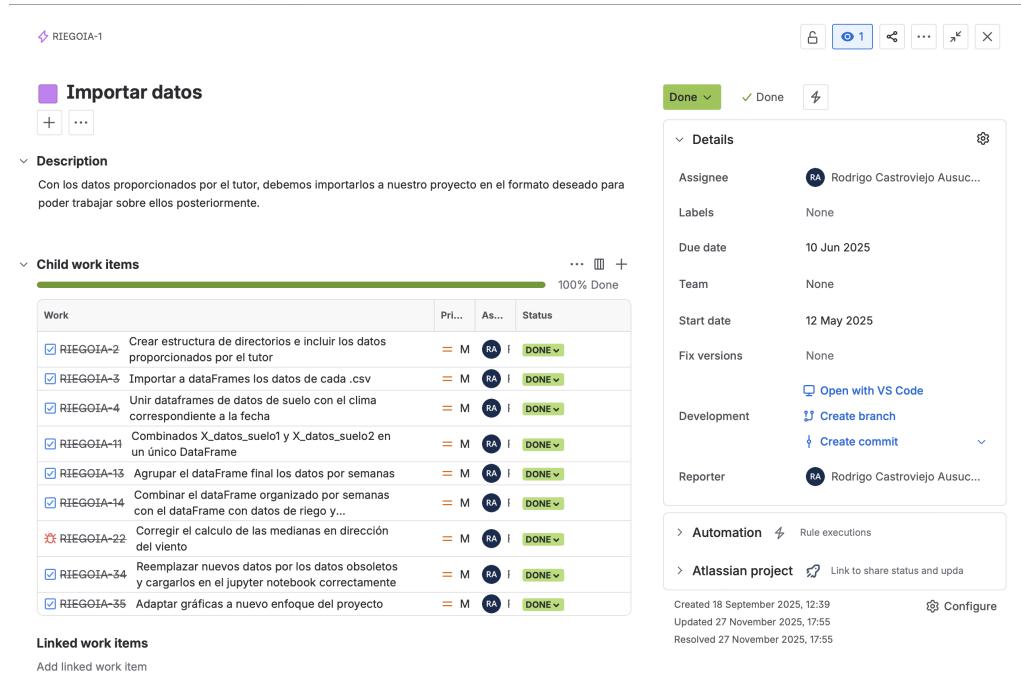
Este plan de proyecto se estructura en fases o sprints, cada una de las cuales permite iterar, ajustar y validar de manera progresiva los objetivos definidos. Este enfoque garantiza que, a lo largo del proceso, la aplicación cumpla con los requisitos funcionales y técnicos establecidos.

A.2. Estructura de Epics

Para la organización del desarrollo, se han definido diversos Epics en Jira que agrupan las funcionalidades y etapas clave del sistema. Un Epic representa un bloque de trabajo significativo que se desglosa en tareas específicas para facilitar el seguimiento y la implementación técnica [6]. A continuación, se detallan los bloques que componen el proyecto:

Epic 1: Importar datos

Este bloque inicial se centra en la preparación de la infraestructura de datos. Comprende desde la importación de los datos, a su procesado para combinarlos en un único DataFrame.



The screenshot shows a Jira Epic titled "Importar datos" (Import Data). The epic has a status of "Done" and is assigned to "Rodrigo Castroviejo Ausuc...". It was created on 18 September 2025, 12:39 and updated on 27 November 2025, 17:55. The epic has 100% Done. The child work items are:

Work	Pri...	As...	Status
RIEGOIA-2 Crear estructura de directorios e incluir los datos proporcionados por el tutor	= M	RA I	DONE
RIEGOIA-3 Importar a dataFrames los datos de cada .csv	= M	RA I	DONE
RIEGOIA-4 Unir dataframes de datos de suelo con el clima correspondiente a la fecha	= M	RA I	DONE
RIEGOIA-41 Combinados X_datos_suelo1 y X_datos_suelo2 en un único DataFrame	= M	RA I	DONE
RIEGOIA-43 Agrupar el DataFrame final los datos por semanas	= M	RA I	DONE
RIEGOIA-44 Combinar el DataFrame organizado por semanas con el DataFrame con datos de riego y...	= M	RA I	DONE
RIEGOIA-22 Corregir el cálculo de las medianas en dirección del viento	= M	RA I	DONE
RIEGOIA-34 Reemplazar nuevos datos por los datos obsoletos y cargarlos en el jupyter notebook correctamente	= M	RA I	DONE
RIEGOIA-35 Adaptar gráficas a nuevo enfoque del proyecto	= M	RA I	DONE

Linked work items: Add linked work item

Figura A.1: Actividades que comprende el Epic: Importar datos

[Enlace a Epic 1 en Jira](#)

Epic 2: EDA inicial

El Análisis Exploratorio de Datos (EDA) inicial comprende las tareas de validación y comprensión estadística de la información importada. En

esta fase se realizan verificaciones de calidad (búsqueda de nulos), análisis de varianzas y distribuciones, así como la generación de visualizaciones preliminares para identificar patrones y anomalías antes de aplicar modelos de aprendizaje automático.

The screenshot shows a Jira Epic titled 'EDA inicial' (purple icon). The 'Details' panel on the right shows the assignee as 'Rodrigo Castroviejo Ausuc...' (RA), due date as '09 Jun 2025', and labels as 'None'. The 'Work' table below lists five child work items, all of which are marked as 'DONE' with a green status indicator. The 'Activity' section at the bottom shows a comment from 'RA' and a 'Looks good!' button.

Work	Pri...	As...	Status
RIEGOIA-7	= M	RA I	DONE
RIEGOIA-8	= M	RA I	DONE
RIEGOIA-9	= M	RA I	DONE
RIEGOIA-10	= M	RA I	DONE
RIEGOIA-12	= M	RA I	DONE

Figura A.2: Actividades que comprende el Epic: EDA inicial

[Enlace a Epic 2 en Jira](#)

Epic 3: EDA selección features

Una vez comprendidos los datos, este Epic se enfoca en la optimización del conjunto de variables. El trabajo abarca la aplicación de técnicas de ingeniería de variables y reducción de dimensionalidad para mejorar la eficiencia de los modelos futuros.

The screenshot shows a Jira Epic titled "EDA selección de features". The "Description" section contains a bulleted list of tasks: "Implementar Matriz de correlación para ver cuanto esta cada atributo relacionado con la variable objetivo", "Identificación preliminar de features relevantes (mediante la matriz de correlación)", "Selección de mejores atributos usando Random Forest (no finalizado)", and "Preparación informe con las justificaciones sobre la elección de los atributos". The "Child work items" section lists six sub-tasks, each with a checkbox, priority (M), assignee (RA), and status (DONE). The "Details" panel on the right shows the assignee as "Rodrigo Castroviejo Ausuc...", labels as "None", due date as "None", team as "None", start date as "17 Jun 2025", and fix versions as "None". It also includes links to "Open with VS Code", "Create branch", "Create commit", and an "Atlassian project".

Work	Pri...	As...	Status
<input checked="" type="checkbox"/> RIEGOIA-18 Implementar matriz de correlación de atributos respecto a la variable objetivo	= M	RA I	DONE ✓
<input checked="" type="checkbox"/> RIEGOIA-19 Analizar matriz correlación y sacar primeras conclusiones	= M	RA I	DONE ✓
<input checked="" type="checkbox"/> RIEGOIA-20 Implementar random forest, para la selección de los mejores atributos (scikit-learn)	= M	RA I	DONE ✓
<input checked="" type="checkbox"/> RIEGOIA-21 Elaborar informe final	= M	RA I	DONE ✓
<input checked="" type="checkbox"/> RIEGOIA-22 Decidir que Scalers (o no) usar sobre las variables numéricas	= M	RA I	DONE ✓
<input checked="" type="checkbox"/> RIEGOIA-38 Hacer ingeniería de atributos y reducir dimensionalidad	= M	RA I	DONE ✓

Figura A.3: Actividades que comprende el Epic: EDA selección features

Enlace a Epic 3 en Jira

Epic 4: Creación y uso de modelos

Este Epic constituye el núcleo predictivo del sistema. Se orienta a la investigación y despliegue de diversos algoritmos de minería de datos, desde modelos estadísticos tradicionales hasta redes neuronales avanzadas para series temporales multivariantes.

CREACIÓN Y USO DE MODELOS

Description

Este EPIC esta enfocado a la investigación, selección y desarrollo de los modelos, comenzando con el desarrollo de los modelos más sencillos. Posteriormente, se planea implementar modelos más avanzados que se determinaran durante el estudio de los algoritmos disponibles para series temporales multivariante.

Child work items

Work	Priority	Assignee	Status
RIEGOIA-24 Estudiar los algoritmos de minería de datos a aplicar	Medium	Rod...	DONE
RIEGOIA-25 Aplicar el algoritmo Random Forest sobre los datos	Medium	Rod...	DONE
RIEGOIA-29 Generar informe acerca de los algoritmos escogidos para los...	Medium	Rod...	DONE
RIEGOIA-37 Aplicar el algoritmo decidido SARIMA/SARIMAX	Medium	Rod...	DONE
RIEGOIA-37 Aumentar la granularidad de el DataFrame de semanal a diaria	Medium	Rod...	DONE
RIEGOIA-39 Reducir el horizonte de predicción de los modelos actualmente...	Medium	Rod...	DONE
RIEGOIA-41 Crear sistema para guardar modelos entrenados	Medium	Rod...	DONE
RIEGOIA-48 Implementar LSTM	Medium	Rod...	DONE

Details

Assignee: Rodrigo Castroviejo Ausuc...
 Labels: None
 Due date: None
 Team: None
 Start date: None
 Fix versions: None
 Development: Open with VS Code, Create branch, Create commit
 Reporter: Rodrigo Castroviejo Ausuc...
 Automation: Rule executions
 Atlassian project: Link to share status and update
 Created 19 September 2025, 13:24
 Updated 10 minutes ago
 Resolved 10 minutes ago

Figura A.4: Actividades que comprende el Epic: Creación y uso de modelos

1

[Enlace a Epic 4 en Jira](#)

Epic 5: Creación base aplicación web

En este bloque se establecen los cimientos de la interfaz de usuario. Comprende la investigación de las herramientas de desarrollo web, el diseño de la arquitectura backend (basada en Flask) y la creación de un modelo funcional básico de la plataforma que servirá de soporte para las funcionalidades de visualización y predicción finales.

Figura A.5: Actividades que comprende el Epic: Creación base aplicación web

[Enlace a Epic 5 en Jira](#)

Epic 6: Cálculo necesidad hídrica cultivo

Este Epic integra los conocimientos agronómicos con los datos del sistema. Se enfoca en el desarrollo de la lógica necesaria para calcular la necesidad hídrica deficitaria final sobre las instancias predichas por los modelos.

The screenshot shows a Jira Epic card for the epic 'Calculo necesidad hídrica cultivo'. The card has a purple header and a 'In Progress' status indicator. It includes sections for 'Description' (with a text area containing project details), 'Child work items' (a table showing five sub-tasks: RIEGOIA-40 to RIEGOIA-45, all assigned to 'Rod...' and marked as 'DONE'), and 'Details' (a panel showing project metadata like assignee, labels, and start date). A 'Linked work items' section is also present.

Work	Priority	Assignee	Status
RIEGOIA-40 Calculo de formula ETO	Medium	Rod...	DONE
RIEGOIA-42 Cálculo y ajuste de Kc en las diferentes fases	Medium	Rod...	DONE
RIEGOIA-43 Construcción final de la Gráfica Kc a lo largo del tiempo	Medium	Rod...	DONE
RIEGOIA-44 Cálculo ETc y aplicación a instancias predichas mediante modelos	Medium	Rod...	DONE
RIEGOIA-45 Cálculo necesidad de riego deficitario final diaria cultivo sobre instancias...	Medium	Rod...	TO DO

Figura A.6: Actividades que comprende el Epic: Cálculo necesidad hídrica cultivo

[Enlace a Epic 6 en Jira](#)

Epic 7: Generación datos sintéticos

Para robustecer el entrenamiento de los modelos, este Epic se centra en la aplicación de técnicas de simulación y generación de datos artificiales. Mediante estrategias como el bootstrapping estacional y perturbaciones controladas, se busca ampliar el volumen de datos para que los algoritmos de Machine Learning identifiquen mejor los patrones de estacionalidad y ciclos del cultivo.

Generación datos sintéticos

Description
Se implementaran varias soluciones para generar datos sintéticos para luego tener más ciclos de estacionalidad para que nuestros algoritmos de ML aprendan los patrones.

Child work items

Work	Priority	Assignee	Status
RIEGOIA-49 Implementar estrategia generación datos sintéticos mediante simulació...	Medium	Rod...	DONE
RIEGOIA-50 Implementar bootstrapping estacional con perturbación para generar dato...	Medium	Rod...	DONE
RIEGOIA-51 Comparar gráficamente rendimiento ambos modelos de generación de...	Medium	Rod...	DONE
RIEGOIA-53 Importar datos sintéticos guardados en .csv a Dataframe	Medium	Rod...	DONE

Linked work items
Add linked work item

Activity

Add a comment...

Looks good! Need help? This is blocked... Can you clarify...? >

Pro tip: press **M** to comment

Details

Assignee: Rodrigo Castroviejo Ausuc... (RA)

Labels: None

Due date: None

Team: None

Start date: 27 Nov 2025

Fix versions: None

Development: 2 commits 2 months ago 1 pull request MERGED

Reporter: Rodrigo Castroviejo Ausuc... (RA)

Automation: Rule executions

Atlassian project: Link to share status and upda

Created: 27 November 2025, 17:53
Updated: 10 December 2025, 13:28
Resolved: 10 December 2025, 13:28

Figura A.7: Actividades que comprende el Epic: Generación datos sintéticos

[Enlace a Epic 7 en Jira](#)

Epic 8: Implementación predicción web

El objetivo del último Epic es la integración final de las estructuras de entrenamiento y predicción desarrolladas en los cuadernos de trabajo dentro de la aplicación web, permitiendo la visualización de los resultados de los modelos de forma accesible para el usuario final.

Desarrollo y unir resultados predicción con aplicación web

Description
Este EPIC esta centrado en la implementación de las estructuras de entrenamiento y predicción ,desarrolladas en el Jupyter notebook, en la aplicación web.

Child work items

Work	Priority	Assignee	Status
Crear página web muestra de resultados de modelos	Medium	Rod...	IN PROGRESS

Linked work items
Add linked work item

Activity
All Comments History Work log

Automation
Rule executions

Atlassian project Link to share status and update

Created 10 December 2025, 11:42 Updated 18 minutes ago Configure

Figura A.8: Actividades que comprende el Epic: Implementación predicción web

[Enlace a Epic 8 en Jira](#)

A.3. Planificación temporal

La aplicación ha seguido una metodología de SCRUM y KANBAN, con sprints de 2 semanas de duración. A continuación, se detallan los sprints en los que se ha desarrollado el proyecto:

Sprint 1

Fecha: 2025/05/12 - 2025/05/26 (2 semanas)

Objetivos

Establecer la base del proyecto y pensar en la planificación de este, tanto en las herramientas, como los pasos que debemos seguir en el desarrollo del proyecto (algoritmos que queremos usar, como enfocar esos algoritmos a nuestro conjunto de datos, etc).

Tareas planificadas para el sprint

Type	Key	Summary	Status	Sprint	Story point estimate
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-2	Crear estructura de directorios e incluir los datos proporcionados por el tutor	DONE	(Tablero Sprint 1)	1
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-3	Importar a dataFrames los datos de cada .csv	DONE	(Tablero Sprint 1)	3
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-4	Unir dataframes de datos de suelo con el clima correspondiente a la fecha	DONE	(Tablero Sprint 1)	3

Figura A.9: Tablero Sprint 1

Retrospective Review

Se han finalizado a tiempo todas las tareas planificadas. Sin embargo, el epic “Importar datos” sigue en progreso, se continuará haciendo tareas relacionadas con este epic en el siguiente sprint.

Adjunto referencias a los commits de github en los que se realizan las diferentes historias.

- Historia 2: Crear estructura de directorios e incluir los datos proporcionados por el tutor [Enlace a commit en GitHub](#)
- Historia 3: Importar a dataFrames los datos de cada .csv [Enlace a commit en GitHub](#)
- Historia 4: Unir dataframes de datos de suelo con el clima correspondiente a la fecha [Enlace a commit en GitHub](#)

Planes a futuro, posibles mejoras, observaciones...

La hoja de ruta para los próximos sprints prioriza la evolución hacia una arquitectura de software más robusta y organizada, facilitando el mantenimiento a largo plazo. Asimismo, se plantea el inicio de un análisis preliminar sobre el conjunto de datos, orientada a generar el conocimiento necesario que sustente la toma de decisiones técnicas y funcionales en las etapas posteriores del desarrollo.

Decisiones a adoptar

Modularización del entorno de desarrollo: Establecer la división del código en múltiples ficheros para evitar la saturación de los documentos

de trabajo. Lo desarrollado hasta la fecha se refactorizará en un script de Python (.py) para optimizar la eficiencia y velocidad de carga de datos en comparación con el formato interactivo.

Segregación de responsabilidades: El análisis exploratorio de datos (EDA) se llevará a cabo de forma independiente en un *Jupyter Notebook*, separando la lógica de preprocesamiento de la lógica de visualización y exploración inicial.

Sprint 2

Fecha: 2025/05/26 - 2025/06/09 (2 semanas)

Objetivos

Avanzar con el tratamiento de los datos para conseguir unificarlos en un dataframe con el intervalo temporal deseado (1 semana) para posteriormente tratarlo con algoritmos de minería de datos especializados en series temporales.

También en este sprint, se quiere avanzar con el análisis exploratorio preliminar que nos sirva para tomar las primeras decisiones del tratamiento de los datos.

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOIA-6	Refactorizar el código en dos ficheros loader.py y EDA.py para separar funciones	DONE	Tablero Sprint 2	3
<input checked="" type="checkbox"/>	RIEGOIA-5 ↳ RIEGOIA-7	Analisis comparativo de medias y varianzas entre 'df_datos_suelo1' y 'df_datos_suelo2' p...	DONE	Tablero Sprint 2	1
<input checked="" type="checkbox"/>	RIEGOIA-5 ↳ RIEGOIA-8	Verificación de valores nulos/missing: No se encontraron datos faltantes en ningún Data...	DONE	Tablero Sprint 2	1
<input checked="" type="checkbox"/>	RIEGOIA-5 ↳ RIEGOIA-9	Obtener estadísticos principales de series temporales mediante funciones integradas en libr...	DONE	Tablero Sprint 2	1
<input checked="" type="checkbox"/>	RIEGOIA-5 ↳ RIEGOIA-10	Generar visualizaciones iniciales (series, boxplots) para ver distribución de los datos visualm...	DONE	Tablero Sprint 2	5
<input checked="" type="checkbox"/>	RIEGOIA-5 ↳ RIEGOIA-12	Analizar distribución de los dataFrames X_datos_suelo1 y 2 para ver si podemos combinarlos	DONE	Tablero Sprint 2	3
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-11	Combinados X_datos_suelo1 y X_datos_suelo2 en un único DataFrame	DONE	Tablero Sprint 2	3

Figura A.10: Tablero Sprint 2

Retrospective Review

NO se han finalizado a tiempo todas las tareas planificadas. Se han finalizado 7 de 8, quedando sin finalizar “Agrupar el DataFrame final los datos por semanas”. Ha finalizado el epic “EDA inicial”. El epic “Importar datos” sigue desarrollándose.

- RIEGOIA-6: Refactorizar el código en dos ficheros loader.py y EDA.ipynb para separar funciones [Enlace a commit en GitHub](#)
- RIEGOIA-7: Análisis comparativo de medias y varianzas entre 'df_datos_suelo1' y 'df_datos_suelo2' para evaluar diferencias. [Enlace a commit en GitHub](#)
- RIEGOIA-8: Verificación de valores nulos/missing: No se encontraron datos faltantes en ningún DataFrame. [Enlace a commit en GitHub](#)
- RIEGOIA-9: Obtener estadísticos principales de series temporales mediante funciones integradas en librería pandas. [Enlace a commit en GitHub](#)
- RIEGOIA-10: Generar visualizaciones iniciales (series, boxplots) para ver distribución de los datos visualmente. [Enlace a commit en GitHub](#)

Planes a futuro, posibles mejoras, observaciones...

En las siguientes etapas, el objetivo principal es dejar listo el DataFrame y organizar mejor el código para que sea más fácil trabajar con él a medida que el proyecto crezca. Queremos centrarnos en el análisis exploratorio de los datos para entender bien qué variables son las más importantes y así conseguir que los modelos predigan con la mayor precisión posible. Por último, daremos prioridad a dejar bien definido todo el preprocesamiento de los datos, de forma que el paso a la fase de probar y elegir los algoritmos de aprendizaje automático sea mucho más sencillo y fluido.

Decisiones a adoptar

Reorganización y refactorización del código: Dividir el desarrollo en varios ficheros. Lo implementado hasta el momento se migrará a un archivo de script de Python (.py) para mejorar la eficiencia en la carga y procesamiento de datos. Por otro lado, el análisis exploratorio de datos (EDA) se mantendrá en un *Jupyter Notebook* independiente para aprovechar su capacidad de visualización interactiva.

Unificación del conjunto de datos: Establecer el formato y la metodología para integrar el *DataFrame* de riego con el actual. El objetivo es consolidar una base de datos única, consistente y depurada que sirva como fuente de verdad para el entrenamiento de los modelos.

Metodología de selección de atributos: Fijar los criterios para identificar las variables con mayor impacto sobre la variable objetivo. Esta selección se basará en técnicas de minería de datos que se documentarán detalladamente en un informe técnico adjunto, justificando la eliminación o inclusión de cada variable.

Estrategia de modelado predictivo: Definir la hoja de ruta para evaluar diferentes algoritmos de aprendizaje automático. La elección de estos modelos se orientará específicamente a la naturaleza del cultivo del almendro y a la precisión requerida en la predicción de necesidades hídricas.

Sprint 3

Fecha: 2025/06/09 - 2025/06/23 (2 semanas)

Objetivos

Realizar todos los pasos previos, para al final del sprint, comenzar con la construcción y aplicación de modelos a los datos.

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-13	Agrupar el DataFrame final los datos por semanas	DONE	Tablero Sprint 3	8
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-14	Combinar el DataFrame organizado por semanas con el DataFrame con datos de riego y prec...	DONE	Tablero Sprint 3	5
<input checked="" type="checkbox"/>	RIEGOIA-16	Reorganizar y refactorizar notebook para una mayor limpieza	DONE	Tablero Sprint 3	1
<input checked="" type="checkbox"/>	RIEGOIA-17 ↳ RIEGOIA-18	Implementar matriz de correlación de atributos respecto a la variable objetivo	DONE	Tablero Sprint 3	1
<input checked="" type="checkbox"/>	RIEGOIA-17 ↳ RIEGOIA-19	Analizar matriz correlación y sacar primeras conclusiones	DONE	Tablero Sprint 3	3
<input checked="" type="checkbox"/>	RIEGOIA-17 ↳ RIEGOIA-20	Implementar random forest, para la selección de los mejores atributos (scikit-learn)	DONE	Tablero Sprint 3	3
<input checked="" type="checkbox"/>	RIEGOIA-17 ↳ RIEGOIA-21	Elaborar informe final	DONE	Tablero Sprint 3	5
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-22	Corregir el cálculo de las medianas en dirección del viento	DONE	Tablero Sprint 3	1
<input checked="" type="checkbox"/>	RIEGOIA-17 ↳ RIEGOIA-23	Decidir que Scalers (o no) usar sobre las variables numéricas	DONE	Tablero Sprint 3	3
<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-25	Aplicar el algoritmo Random Forest sobre los datos	DONE	Tablero Sprint 3	5

Figura A.11: Tablero Sprint 3

Retrospective Review

NO se han finalizado a tiempo todas las tareas planificadas. Se han finalizado 9 de 11, quedando sin finalizar “Estudiar los algoritmos de minería de datos a aplicar” y “Aplicar el algoritmo XGBoost sobre los datos”. Ha finalizado el epic “EDA selección de features”.

- RIEGOIA-13: Agrupar el DataFrame final los datos por semanas [Enlace a commit en GitHub](#)
- RIEGOIA-14: Combinar el DataFrame organizado por semanas con el DataFrame con datos de riego y precipitaciones [Enlace a commit en GitHub](#)
- RIEGOIA-16: Reorganizar y refactorizar notebook para una mayor limpieza. [Enlace a commit en GitHub](#)
- RIEGOIA-18: Implementar matriz de correlación de atributos respecto a la variable objetivo. [Enlace a commit en GitHub](#)
- RIEGOIA-19: Analizar matriz correlación y sacar primeras conclusiones [Enlace a commit en GitHub](#)
- RIEGOIA-20: Implementar random forest, para la selección de los mejores atributos (scikit-learn). [Enlace a commit en GitHub](#)
- RIEGOIA-21: Elaborar informe final.
- RIEGOIA-22: Corregir el calculo de las medianas en dirección del viento. [Enlace a commit en GitHub](#)
- RIEGOIA-23: Decidir que Scalers (o no) usar sobre las variables numéricas. [Enlace a commit en GitHub](#)

Planes a futuro, posibles mejoras, observaciones...

En las próximas semanas el objetivo es dejar listos los modelos de predicción y ver cómo mostrar esos resultados de forma clara. La idea es pasar de las pruebas iniciales a tener algo que funcione de verdad, no solo obteniendo buenos números en los algoritmos, sino pensando ya en cómo diseñar una web para que cualquier usuario pueda consultar las predicciones fácilmente. Al final, queremos recoger todo este aprendizaje en un informe que explique bien qué modelos han funcionado mejor y qué pasos recomendariamos seguir en el futuro para este tipo de cultivos.

Decisiones a adoptar

Elección de algoritmos: Decidir qué modelos de aprendizaje automático vamos a usar, basándonos en cuáles se adaptan mejor a nuestro

problema. Prepararé un informe sencillo explicando por qué hemos elegido estos y no otros, comparando sus puntos fuertes y débiles.

Ajuste de los datos: Para que los modelos funcionen lo mejor posible, he concretado cómo voy a preparar los datos. Esto incluye decidir el rango de tiempo que vamos a usar y aplicar técnicas de escalado y normalización para que los algoritmos lean los datos correctamente.

Hoja de ruta para los modelos: Fijar el orden en el que vamos a desarrollar cada modelo y cómo vamos a medir su éxito. Usar métricas de rendimiento claras para poder comparar unos con otros de forma justa.

Contenido del informe final: Definir qué puntos debe tener el informe de resultados. La idea es que no solo se quede en conclusiones técnicas, sino que también dé consejos prácticos sobre lo que hemos descubierto.

Planificación de la web: Decidir empezar ya con el diseño y la estructura de la aplicación web. Aunque el grueso del desarrollo vendrá más adelante, queremos tener claro desde ahora cómo se mostrarán los resultados y qué opciones tendrá el usuario para interactuar con los modelos.

Sprint 4

Fecha: 2025/09/23 - 2025/10/07 (2 semanas)

El hueco temporal entre el último sprint y este se debe a que llegó el verano y viendo que no iba a conseguir entregar en la anterior convocatoria, pausé el desarrollo hasta la vuelta al ciclo académico.

Objetivos

Finalizar de construir los modelos de minería de datos con los que vamos a predecir la variable objetivo. Acompañado de esto, generar un informe con los resultados de los diferentes modelos e ir pensando/implementando, base de la página web que mostrará los resultados del proyecto.

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-24	Estudiar los algoritmos de minería de datos a aplicar	DONE	(2) Tablero Sprint 4	5
<input checked="" type="checkbox"/>	RIEGOIA-31 ↳ RIEGOIA-30	Investigar como hacer la página web e ir trazando un plan con las tecnologías que vamos a ...	DONE	(2) Tablero Sprint 4	5
<input checked="" type="checkbox"/>	RIEGOIA-31 ↳ RIEGOIA-32	Crear base estructura backend en Flask	DONE	(2) Tablero Sprint 4	3

Figura A.12: Tablero Sprint 4

Retrospective Review

NO se ha finalizado a tiempo todas las tareas planificadas. Se han finalizado 3 de 7. El sprint se ha centrado mayoritariamente en dar los primeros pasos en la creación de la aplicación web. Desde investigar las tecnologías y hacer un plan inicial de desarrollo, a implementar toda la base funcional de la aplicación web. Las tareas que se han quedado pendientes han sido las correspondientes a implementación de los algoritmos para procesar los datos, ya que se han dejado de lado temporalmente por avanzar en el desarrollo de la aplicación web. También se han quedado pendientes tareas a futuro sobre el desarrollo de la web. Hasta ahora, se ha implementado una base funcional con flask sobre docker.

Lo ultimo implementado ha sido la gestión de usuarios en la aplicación web.

- **RIEGOIA-24:** Estudiar los algoritmos de minería de datos a aplicar
[Enlace a commit en GitHub](#)
- **RIEGOIA-30:** Investigar como hacer la página web e ir trazando un plan con las tecnologías que vamos a usar. [Enlace a commit en GitHub](#)
- **RIEGOIA-32:** Crear base estructura backend en Flask.
 - ↳ [Commit 6b0828e](#): Estructura base y Dockerfile.
 - ↳ [Commit f3a5dc0](#): Creación templates HTML/Jinja2 + docker-compose.yml
 - ↳ [Commit 0e876eb](#): Adaptación proyecto a postgresSQL.
 - ↳ [Commit a433046](#): Añadida gestión de usuarios y cambios para adaptar app a ello.

Planes a futuro, posibles mejoras, observaciones...

De cara a las próximas semanas, la prioridad absoluta es conseguir que los modelos de predicción dejen de ser pruebas aisladas y pasen a ser funcionales.

Queremos dedicar el grueso del esfuerzo a que los algoritmos estén bien definidos y listos para integrarse. Al mismo tiempo, no queremos descuidar la experiencia del usuario; por eso, empezaremos a dar forma a la parte personal de la web con los perfiles de usuario y a pensar en cómo conectar de forma robusta la entrada de datos con los procesos de los modelos. La idea es dejar los retoques estéticos más minuciosos para cuando la base técnica sea sólida, asegurándonos primero de que todo el flujo de información, desde que se suben los datos hasta que se obtiene un resultado, funcione sin errores.

Decisiones a adoptar

Diseño del perfil de usuario: Hemos decidido planificar la creación de la página de perfil para el siguiente sprint. El objetivo es definir qué información se mostrará y qué opciones de edición tendrá el usuario, manteniendo una línea visual que encaje con lo que ya tenemos construido.

Selección técnica de algoritmos: Vamos a cerrar definitivamente la lista de algoritmos que utilizaremos. Para que la decisión quede bien fundamentada, redactaré un informe técnico que explique por qué estos modelos son los más aptos en cuanto a precisión y viabilidad para nuestro proyecto.

Priorización del desarrollo de modelos: La decisión principal es centrar la mayor parte del tiempo y recursos del próximo sprint en tener modelos que ya funcionen. Para avanzar rápido, hemos acordado posponer las tareas estéticas secundarias o detalles visuales que consumen mucho tiempo y no son críticos para el funcionamiento actual.

Sprint 5

Fecha: 2025/10/07 - 2025/10/21 (2 semanas)

Objetivos

Diseño de perfil de usuario: Planificar la estructura y generar rutas aplicación web manteniendo un diseño visual coherente con lo realizado hasta el momento.

Selección de algoritmos: Redacción de informe justificativo.

Desarrollo de modelos: Priorizar modelos funcionales, posponiendo tareas secundarias.

Tareas planificadas para el sprint



Figura A.13: Tablero Sprint 5

Retrospective Review

Debido a complicaciones de tiempo, este sprint, no se terminó casi nada de lo que tenía previsto. Se finalizó únicamente la creación de la base de la página web. Detalles estéticos sobre desplegables y apariencia de la página principal y el navbar que esta contiene.

- RIEGOIA-33: Crear primer modelo funcional muy básico de la página web. [Enlace a commit en GitHub](#)

Planes a futuro, posibles mejoras, observaciones...

Dada la reunión con el tutor del TFG, los resultados de esa reunión determinarán que realizar a futuro. Mantenemos los planes del anterior sprint, ya que no se ha realizado avance significativo en este sprint.

Decisiones

No se han tomado decisiones debido a la reunión pendiente que cambie el rumbo del proyecto.

Sprint 6

Fecha: 2025/10/23 - 2025/11/06 (2 semanas)

Objetivos

Importar los nuevos datos de suelo y clima de años anteriores y construir predicción a 1 año basado en dichos datos.

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-29	Generar informe acerca de los algoritmos escogidos para los modelos	DONE	🔗 Tablero Sprint 6	3
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-34	Reemplazar nuevos datos por los datos obsoletos y cargarlos en el jupyter notebook correct...	DONE	🔗 Tablero Sprint 6	11
<input checked="" type="checkbox"/>	RIEGOIA-1 ↳ RIEGOIA-35	Adaptar gráficas a nuevo enfoque del proyecto	DONE	🔗 Tablero Sprint 6	5

Figura A.14: Tablero Sprint 6

Retrospective Review

Durante este sprint se ha centrado el trabajo en adaptar la base de datos y los notebooks de análisis al nuevo formato de los archivos de entrada, sustituyendo los antiguos CSV por hojas Excel (.xlsx) y ajustando el módulo loader.py para soportar múltiples hojas y diferentes estructuras de datos. Esto permitió unificar la información proveniente de clima y suelo en un único flujo de trabajo, manteniendo la compatibilidad con los análisis y visualizaciones previas.

Asimismo, se realizaron tareas de limpieza y depuración de datos, eliminando instancias con valores nulos y reorganizando el código en celdas más claras dentro de los notebooks, añadiendo descripciones y mejorando la legibilidad general. También se incorporó un nuevo gráfico para visualizar la “agua útil a través del espacio”.

Posteriormente, se actualizó el cuaderno de cálculo de la necesidad de agua, retirando secciones obsoletas y analizando la distribución de las variables mediante rangos intercuartílicos (*IQR*) para determinar la conveniencia de aplicar distintos métodos de normalización.

Finalmente, se implementaron los primeros modelos de predicción, comenzando con *SARIMA* y *SARIMAX*, logrando ejecutar y evaluar sus resultados mediante métricas *MAE*, *MAPE* y *RMSE*. A pesar de las correcciones aplicadas (como el manejo de ceros en los datos de test o la supresión de warnings), los resultados iniciales fueron catastróficos, lo que deja claro que estos modelos requieren ajustes significativos.

El sprint concluye, por tanto, con la infraestructura de datos estabilizada y los primeros experimentos de modelado completados, aunque aún no se ha logrado corregir la desastrosa primera predicción obtenida, tarea que quedará pendiente para el próximo ciclo de trabajo.

- **RIEGOIA-29** Generar informe acerca de los algoritmos escogidos para los modelos.

- **RIEGOIA-34:** Reemplazar nuevos datos por los datos obsoletos y cargarlos en el jupyter notebook correctamente.
 - ↪ [Commit b17efe5](#): Reemplazar archivos .csv obsoletos por archivos Excel (.xlsx) en loader.py
 - ↪ [Commit 6e22d2e](#): Adaptación EDA.ipynb a nuevo formato datos
 - ↪ [Commit 86608d6](#): Solución de problemas al combinar DataFrames y refactorización código.
- **RIEGOIA-35:** Adaptar gráficas a nuevo enfoque del proyecto. [Enlace a commit en GitHub](#)

Planes a futuro, posibles mejoras, observaciones...

Tras analizar el rendimiento de los modelos actuales, hemos detectado que para ganar precisión en el cálculo de la evapotranspiración (ET_0) necesitamos ser más específicos con los datos. La idea es pasar de una visión general a una granularidad diaria, lo que nos permitirá captar mejor los cambios del entorno. Queremos simplificar los modelos eliminando variables que solo aportan ruido y acortar el tiempo que intentamos predecir; en lugar de buscar un horizonte a un año vista, nos centraremos en los próximos cuatro meses para reducir errores. Con estos ajustes de "limpieza" enfoque, esperamos resolver los problemas técnicos de convergencia que nos estaban dando los modelos más complejos, como el *VAR* o el *SARIMAX*, logrando así predicciones mucho más fiables y robustas.

Decisiones a adoptar

Cambio en la escala temporal: Adoptar la granularidad diaria como base para cálculos y modelados.

Optimización de variables: Aplicar una reducción de dimensionalidad descartando aquellas variables que no influyan directamente en la fórmula FAO-56. De esta forma, simplificamos la complejidad del modelo sin perder información relevante.

Reajuste del horizonte de predicción: Limitar la ventana de predicción a un máximo de cuatro meses. Hemos decidido priorizar la exactitud en el corto y medio plazo antes que intentar cubrir un año entero con un margen de error acumulativo demasiado alto.

Refactorización de modelos estadísticos: Mantener y mejorar el uso de los modelos *SARIMA*, *SARIMAX* y *VAR*. Una vez aplicados los cambios en los datos (granularidad y menos variables), procederemos a reentrenarlos para solucionar los fallos de convergencia detectados anteriormente.

Sprint 7

Fecha: 2025/11/07 - 2025/11/21 (2 semanas)

Objetivos

- Adoptar la granularidad diaria como base para futuros cálculos y modelados.
- Reducir el número de variables manteniendo únicamente las que influyen en la ET_0 .
- Limitar el horizonte de predicción a 4 meses, priorizando precisión sobre extensión temporal.
- Mantener y mejorar los modelos SARIMA, SARIMAX y VAR, reentrenándolos con los nuevos ajustes de datos y parámetros.
- Calcular ET_0 y ET_c , con el ajuste correspondiente de K_c .

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-28	Aplicar el algoritmo decidido SARIMA/SARIMAX	DONE	Tablero Sprint 7	8
<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-37	Aumentar la granularidad de el DataFrame de semanal a diaria	DONE	Tablero Sprint 7	5
<input checked="" type="checkbox"/>	RIEGOIA-27 ↳ RIEGOIA-39	Reducir el horizonte de predicción de los modelos actualmente implementado de 12 a 4 meses	DONE	Tablero Sprint 7	8
<input checked="" type="checkbox"/>	RIEGOIA-46 ↳ RIEGOIA-40	Calculo de formula ETO	DONE	Tablero Sprint 7	5
<input checked="" type="checkbox"/>	RIEGOIA-46 ↳ RIEGOIA-42	Calculo y ajuste de K_c en las diferentes fases	DONE	Tablero Sprint 7	13

Figura A.15: Tablero Sprint 7

Retrospective Review

Durante este sprint se ha centrado el trabajo en refinar la granularidad temporal de los datos, cambiando a una frecuencia diaria para mejorar la precisión en el análisis temporal, lo que requirió la eliminación de columnas con valores NaN resultantes de este ajuste y la actualización de todas

las referencias en los notebooks correspondientes. Asimismo, se optimizó el preprocesamiento de datos mediante un análisis automatizado que determinó el tipo de scaler más adecuado para cada variable, y se incorporó un estudio de estacionalidad que identificó un patrón mensual, el cual fue integrado en la configuración de los modelos.

Posteriormente, se resolvió una incidencia crítica relacionada con un índice no válido debido a frecuencia irregular, aplicando interpolación para completar los días faltantes y asegurando así una secuencia temporal uniforme con diferencia de un día entre todas las instancias.

En paralelo, se implementó la metodología FAO-56 Penman-Monteith para el cálculo diario de las necesidades hídricas del cultivo, adaptando los coeficientes de cultivo (K_c) a las condiciones locales y al sistema de riego por goteo. Esto incluyó la definición de las etapas de crecimiento, la extracción de valores K_c de referencia del artículo FAO-56, y el cálculo específico de K_{cni} , K_{cmid} y K_{cend} considerando variables como intervalos entre riegos, profundidad de infiltración, humedad relativa mínima y velocidad del viento.

Finalmente, se refactorizó el código para eliminar duplicaciones, creando funciones reutilizables para el cálculo de K_c con validación de entradas, y se construyó la gráfica de evolución temporal de K_c utilizando matplotlib, además de centralizar las constantes y parámetros frecuentes en un diccionario para mejorar el mantenimiento y la claridad del código.

El sprint concluye con la base de datos estabilizada en granularidad diaria, la metodología científica para el cálculo de necesidades hídricas completamente implementada y adaptada localmente, y una base de código más robusta y mantenible, sentando las bases para el desarrollo del sistema de recomendación de riego en los próximos ciclos.

- **RIEGOIA-37:** Aumentar la granularidad de el DataFrame de semanal a diaria.
 - ↪ [Commit 49c3d06](#): Adaptación a granularidad diaria y ajuste de estacionalidad en modelos.
 - ↪ [Commit b7c81db](#): Resolución incidencia Not Valid index
- **RIEGOIA-39:** Reducir el horizonte de predicción de los modelos actualmente implementado de 12 a 4 meses.
- **RIEGOIA-40:** Calculo de formula ET0. [Enlace a commit en GitHub](#)
- **RIEGOIA-42:** Calculo y ajuste de K_c en las diferentes fases.

- **Commit 04d8817**: Adaptación de Kc_{ni} a nuestras condiciones y variedad.
- **Commit 876fc27**: Adaptación Kc_{mid} a nuestras condiciones locales.
- **Commit 1fcd3ba**: Calculo Kc end a nuestras condiciones de cultivo locales
- **Commit 1b5caa4**: Solución periodo sobre el que se calcula RH_{min} y $u2$ en Kc mid.

Planes a futuro, posibles mejoras, observaciones...

El foco principal para las próximas etapas es garantizar la estabilidad técnica de nuestros modelos estadísticos (SARIMA, VAR y SARIMAX), resolviendo los errores de convergencia que han surgido al trabajar con grandes volúmenes de datos. Para lograrlo, vamos a simplificar los modelos eliminando cualquier variable que no sea estrictamente necesaria para el cálculo de la evapotranspiración. Una vez que la lógica de predicción sea robusta, empezaremos a dar el salto visual del proyecto: el diseño de la interfaz web. El objetivo es que los resultados dejen de ser solo datos técnicos y pasen a mostrarse de forma clara y útil para el usuario final a través de gráficas y métricas intuitivas.

Decisiones a adoptar

Optimización de la dimensionalidad: Se ha decidido reducir el número de variables de entrada, conservando únicamente aquellas con impacto directo en el cálculo de la ET_0 . Esto reducirá la complejidad computacional y evitará errores en el entrenamiento.

Estabilización de modelos: Priorizar la resolución de los problemas de convergencia en SARIMA, VAR y SARIMAX. Realizaremos una revisión profunda de los parámetros y de los criterios de parada de los algoritmos para asegurar que los modelos finalicen sus procesos correctamente.

Arquitectura del frontend: Iniciar formalmente el modelado de la plataforma web. Hemos acordado definir una arquitectura que no solo muestre las predicciones, sino que incluya componentes específicos para visualizar series temporales y sugerencias prácticas de riego.

Panel de visualización: Planificar el diseño de módulos dedicados a métricas de rendimiento y gráficos comparativos, facilitando que cualquier usuario pueda interpretar el éxito y la precisión de los modelos entrenados.

Sprint 8

Fecha: 2025/11/25 - 2025/12/09 (2 semanas)

No comenzamos el sprint 8 nada más finalizar el anterior, hay 4 días vacios entremedias debido a un viaje.

Objetivos

MÁXIMA IMPORTANCIA: Resolver los problemas de convergencia y dimensionalidad detectados en los modelos SARIMA, VAR y SARIMAX.

Aplicar una reducción de dimensionalidad eliminando variables irrelevantes para el cálculo de la ET_0 , reduciendo así la complejidad combinatoria en los modelos.

Comenzar a modelar pagina web, para mostrar resultados de los modelos entrenados.

Tareas planificadas para el sprint

<input checked="" type="checkbox"/>	RIEGOJA-17 t _o RIEGOJA-38	Hacer ingeniería de atributos y reducir dimensionalidad	DONE	(2) Tablero Sprint 8	3
<input checked="" type="checkbox"/>	RIEGOJA-27 t _o RIEGOJA-41	Crear sistema para guardar modelos entrenados	DONE	(2) Tablero Sprint 8	3
<input checked="" type="checkbox"/>	RIEGOJA-46 t _o RIEGOJA-44	Calculo ET _c y aplicación a instancias predichas mediante modelos	DONE	(2) Tablero Sprint 8	1
<input checked="" type="checkbox"/>	RIEGOJA-27 t _o RIEGOJA-48	Implementar LSTM	DONE	(2) Tablero Sprint 8	9
<input checked="" type="checkbox"/>	RIEGOJA-52 t _o RIEGOJA-49	Implementar estrategia generación datos sintéticos mediante simulación basada en física ag...	DONE	(2) Tablero Sprint 8	3
<input checked="" type="checkbox"/>	RIEGOJA-52 t _o RIEGOJA-50	Implementar bootstrapping estacional con perturbación para generar datos sintéticos	DONE	(2) Tablero Sprint 8	3
<input checked="" type="checkbox"/>	RIEGOJA-52 t _o RIEGOJA-51	Comparar gráficamente rendimiento ambos modelos de generación de datos sintéticos	DONE	(2) Tablero Sprint 8	8
<input checked="" type="checkbox"/>	RIEGOJA-52 t _o RIEGOJA-53	Importar datos sintéticos guardados en .csv a Dataframe	DONE	(2) Tablero Sprint 8	3

Figura A.16: Tablero Sprint 8

Retrospective Review

Durante este periodo, se consolidó la identificación y validación de variables exógenas críticas para el cálculo de necesidades hídricas, integrando parámetros climáticos y radiación solar.

Sobre esta base, se desarrolló un sistema de generación de datos sintéticos mediante simulación física-estadística y *bootstrap* estacional perturbado, permitiendo la creación de *datasets* híbridos validados gráficamente para el entrenamiento de modelos.

En cuanto a la arquitectura del software, se completó la migración a *pipelines* de `scikit-learn` mediante *wrappers* personalizados y `ColumnTransformer`, garantizando la persistencia y trazabilidad del modelo.

Finalmente, se integraron con éxito redes LSTM y modelos SARIMAX, corrigiendo errores críticos en la gestión de índices temporales (`pd.DateTime`) y la compatibilidad de atributos en el flujo de predicción.

- **RIEGOIA-38:** Hacer ingeniería de atributos y reducir dimensionalidad. [Enlace a commit en GitHub](#)
- **RIEGOIA-49:** Implementar estrategia generación datos sintéticos mediante simulación basada en física agrícola. [Enlace a commit en GitHub](#)
- **RIEGOIA-50:** Implementar bootstrapping estacional con perturbación para generar datos sintéticos. [Enlace a commit en GitHub](#)
- **RIEGOIA-51:** Comparar gráficamente rendimiento ambos modelos de generación de datos sintéticos. [Enlace a commit en GitHub](#)
- **RIEGOIA-53:** Importar datos sintéticos guardados en .csv a Dataframe. [Enlace a commit en GitHub](#)
- **RIEGOIA-48:** Implementar LSTM.
 - [Commit be06d4d](#): Reorganización en pipelines de modelos ya implementados.
 - [Commit 7fad370](#): Implementado LSTM y solucionado problemas con carga de predicciones.
- **RIEGOIA-41:** Crear sistema para guardar modelos entrenados. [Enlace a commit en GitHub](#)
- **RIEGOIA-41:** Calculo ETc y aplicación a instancias predichas mediante modelos

Planes a futuro, posibles mejoras, observaciones...

Iniciar el desarrollo de la plataforma web para transformar los resultados de los modelos en información visual e interactiva. Paralelamente, comenzaré la redacción de la memoria.

Decisiones a adoptar

Aplicación web: Implementar visualización de predicciones en aplicación web con múltiples opciones para que el usuario pueda interactuar con los resultados, filtrar según necesidades y descargar la información que desee.

Escritura de la memoria: Priorizar la documentación técnica dedicando el 50 % del tiempo disponible a la redacción de la memoria del proyecto.

Visión general final

El proyecto ha avanzado progresivamente, aumentando la cantidad de horas dedicadas los últimos meses previos a la fecha límite de entrega.

Commits over the last year of [rodrigocastroviejo/Optimizacion-del-Riego-Deficitario-Controlado-en-cultivo-de-Almendro-mediante-mineria-de-datos](#)

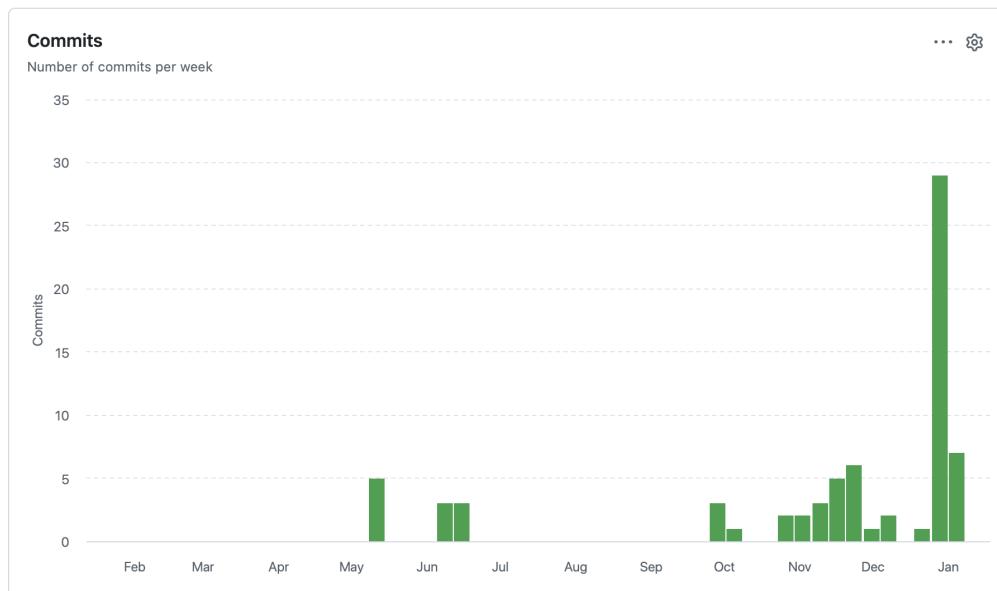


Figura A.17: Número de commits durante el desarrollo del proyecto

A.4. Estudio de viabilidad

Viabilidad económica

Para determinar si financieramente el proyecto es viable, tenemos que calcular todos los recursos usados durante el desarrollo del proyecto, podemos dividir estos en tres secciones:

Coste de personal

Vamos a realizar un desglose de las horas trabajadas. El proyecto se ha llevado en 3 fases diferenciadas en el que se ha variado la cantidad de horas dedicadas a ello.

- Fase inicial: Se comenzó el proyecto el 12/05/2025, y se trabajó intensivamente hasta el 2/06/2025, fecha que coincidió con el comienzo de mi periodo de prácticas, por lo que junto con el periodo de vacaciones de verano, se dejó congelado el proyecto hasta la vuelta al ciclo

académico. Durante esta fase, dediqué 5-6h diarias a el desarrollo del TFG, haciendo cálculos, 16 días x 5.5h diarias. 88h fueron destinadas durante esta fase inicial.

- Fase 2: Comienza con el inicio del ciclo académico presente, registro los primeros progresos el 2025/09/23. En esta fase progresé adecuadamente, dedicándole 2-3h diarias. Podemos dar por concluida esta fase al entrar en diciembre. Este periodo comprende 48 días laborables. Haciendo cuentas sale: 48 x.2.5h/día = 120h.
- Fase final: Con la convocatoria acercándose, aumento a 6-7h diarias el tiempo dedicado al proyecto, este periodo comprende del 1/12/25 a 13/02/2026. Esto hace un total de 56 días laborables. Haciendo un total de 364h.

$$T_{total} = 88 \text{ h} + 120 \text{ h} + 364 \text{ h} = 572 \text{ h} \quad (\text{A.1})$$

Se toma como sueldo medio de un ingeniero informático junior en España un salario bruto de 20.400 €/año [11]. Suponiendo una jornada laboral completa de 8h, la cantidad de horas realizadas al año ascienden a 1.800h [12].

Deberemos calcular los costes adicionales, como la cotización a la seguridad social, que toma la empresa por el trabajador, aumentando el coste final de este. El coste total mensual para la empresa se calcula con la siguiente fórmula:

$$C_{empresa} = S_{bruto} \times 0,334 \quad (\text{A.2})$$

Aplicamos dicha formula a nuestro caso:

$$C_{empresa} = 20,400 \times 0,334 \quad (\text{A.3})$$

[10]

Finalmente, sobre el coste total del trabajador para la empresa, calculamos el coste para el total de horas estimadas que ha llevado el proyecto con la siguiente fórmula:

$$C_{humano} = T_{total} \times \frac{S_{bruto}}{J_{anual}} \quad (\text{A.4})$$

Aplicamos a nuestro caso:

$$C_{humano} = 572 \times \frac{20,400}{1,800} = 6,482,67 \text{ €} \quad (\text{A.5})$$

Coste de hardware

Se ha usado el MacBook Pro 15"(2016) como hardware utilizado en el proyecto. Su valor actual se estima en 0 € dado que su vida útil contable está completa, aunque el valor de mercado de segunda mano podría situarse entre 200–300 €.

Otro factor económico a tener en cuenta, es el consumo de luz del ordenador, ya que para tareas intensas computacionalmente, como el entrenamiento y predicción de modelos, el consumo eléctrico del ordenador se dispara frente a un uso normal.

El MacBook Pro de 15 pulgadas (modelo 2016) presenta un consumo eléctrico medio-alto que oscila aproximadamente entre 25 y 60 W durante un uso exigente. Este rango corresponde a tareas como compilación de código, multitarea avanzada donde el procesador y la GPU trabajan de forma simultánea. En situaciones de carga máxima, el consumo puede acercarse al límite del adaptador de corriente, situado en 87 W, aunque este valor solo se alcanza de forma puntual. [4]. Usaremos de media el punto intermedio del consumo medio-alto sugerido por Apple.

Considerando que el tiempo total de ejecución del proyecto ha sido de 572 horas, el cálculo del consumo energético total (E) se realiza mediante la siguiente expresión:

$$E = \frac{P \cdot t}{1000} = \frac{42,5 \text{ W} \cdot 572 \text{ h}}{1000} = 24,31 \text{ kWh} \quad (\text{A.6})$$

Donde P es la potencia media en vatios y t el tiempo en horas. Tomando como referencia el precio del megavatio hora en España en 2025, situado en 65,52 €/MWh [9], el coste total de electricidad (C_{elec}) se calcula de la siguiente forma:

$$C_{elec} = \left(\frac{E}{1000} \right) \times \text{Coste MWh} = \left(\frac{24,31}{1000} \right) \times 65,52 \text{ €} \approx 1,59 \text{ €} \quad (\text{A.7})$$

A continuación, se presenta el desglose final de los costes asociados al hardware utilizado:

Concepto	Detalle	Subtotal
Amortización de Hardware	MacBook Pro 15"(Vida útil 0)	0,00 €
Consumo Eléctrico	572 horas de uso intensivo	1,59 €
Total Hardware		1,59 €

Tabla A.1: Resumen de costes de hardware y consumo energético.

Coste de software

Todo el software usado es gratuito, sin embargo, en un contexto empresarial con equipos más grandes, tanto Docker [7], Jira atlassian [5] y GitHub [8] ofrecen planes de suscripciones adoptados para empresas en las que aumentan el límite de usuarios por equipo y aportan otros beneficios. No tendremos en cuenta estos ya que no han sido necesarios para el desarrollo del proyecto.

Tabla A.2: Coste del software utilizado en el proyecto

Software	Precio (€)
Docker	0
GitHub	0
Jira	0
VS Code	0
Python	0
Librerías python	0
MacTex	0
Total	0

Costes de infraestructura / servicios (Cloud / Hosting)

El despliegue del sistema se ha realizado mediante *self-hosting* en hardware local, utilizando una infraestructura de *tunneling* para la exposición del servicio. Este enfoque ha permitido prescindir de servicios de *hosting* gestionado, minimizando los costes operativos.

A continuación, se detallan los costes asociados a la infraestructura:

Como se observa, el único gasto directo ha sido la adquisición del dominio por un periodo de un año, aprovechando el plan gratuito de Cloudflare para la conectividad segura.

Concepto	Proveedor / Método	Coste
Hosting y Computación	Hardware propio	0,00 €
Infraestructura de Tunneling	Cloudflare (Free Plan)	0,00 €
Nombre de Dominio	Registro anual	7,50 €
Total		7,50 €

Tabla A.3: Desglose de costes de infraestructura y servicios.

Rendimiento económico

El desarrollo del sistema de predicción se fundamenta en la capacidad de transformar la eficiencia hídrica en rentabilidad financiera directa. Basándose en los datos expuestos en la sección de mejoras económicas, la viabilidad del proyecto se justifica mediante el análisis del retorno de inversión frente a un escenario real de aplicación.

Análisis de Amortización (Payback)

El coste total de ejecución del desarrollo asciende a **6491,76 €**. Para evaluar su amortización, se plantea usar el análisis del documento 'Análisis económico del cultivo de almendro en riego deficitario controlado (RDC)' [20], en el que se reporta que usando RDC, obtenemos un incremento del 11,27% en el beneficio neto total. Estimando un beneficio neto base de 30.000 € para la explotación de referencia, el beneficio adicional generado por la optimización del sistema es:

$$B_{extra} = 30,000 \text{ €} \times 0,1127 = 3,381 \text{ €/año}$$

El periodo de recuperación de la inversión (PRI) para cubrir los 6491,76 € del proyecto se calcula como:

$$PRI = \frac{6491,76 \text{ €}}{3,381 \text{ €/año}} \approx 1,92 \text{ años}$$

Conclusión de Viabilidad La implementación del software de predicción no solo mejora la eficiencia del uso del agua, sino que garantiza la viabilidad financiera del desarrollo. Con un periodo de amortización inferior a 2 campañas agrícolas, el proyecto se posiciona como una inversión de bajo riesgo y alta rentabilidad ante el escenario actual de escasez de recursos hídricos.

Viabilidad legal

Durante el desarrollo de este proyecto, se ha hecho uso de diversas bibliotecas y herramientas externas, cada una bajo su propia licencia de uso. Considerar los términos de estas licencias resulta esencial al momento de definir la licencia final del software, ya que pueden imponer condiciones específicas sobre la distribución, modificación o reutilización del código.

En la siguiente tabla, se indican las licencias de cada una de las herramientas software usadas.

Librería / Herramienta	Licencia	Uso Comercial
Python (Core y Standard Libs)	PSF License	Sí
NumPy, Pandas, SciPy, Joblib	BSD 3-Clause	Sí
Scikit-learn, Statsmodels	BSD 3-Clause	Sí
Matplotlib, Seaborn	Matplotlib / BSD	Sí
TensorFlow / Keras	Apache 2.0	Sí
Flask-SQLAlchemy	BSD 3-Clause	Sí
Flask-Migrate, Flask-Login, Openpyxl	MIT	Sí

Tabla A.4: Resumen de licencias de software de terceros.

El sistema desarrollado se apoya en cuatro modelos de licenciamiento principales. Todas ellas se clasifican como licencias "permisivas", lo que permite su integración en proyectos comerciales sin forzar la apertura del código fuente del software final.

Descripción de las Licencias Utilizadas

- **PSF (Python Software Foundation) [18]**: Es una licencia libre de tipo permisivo, similar a la BSD, pero específica para el lenguaje Python y sus librerías estándar. Permite la modificación y distribución tanto en formato fuente como binario.
- **MIT [16]**: Considerada la licencia más sencilla y popular. Otorga permiso ilimitado para copiar, modificar y vender el software, siempre que se mantenga el aviso de copyright original.
- **BSD (Berkeley Software Distribution) [15]**: En su variante de 3 cláusulas utilizada en este proyecto, es muy restrictiva en cuanto a la protección de la imagen de los autores (no permite usar sus nombres con fines publicitarios sin permiso), pero muy abierta en cuanto al uso del código.

- **Apache 2.0 [3]:** Una licencia moderna que, además de los permisos habituales, incluye una concesión de derechos de patente y términos específicos sobre la protección de marcas registradas.

La aplicación se presenta como una herramienta de apoyo a la decisión, dejando la responsabilidad final de la ejecución del riego en manos del técnico o agricultor. Esto es de vital importancia para evitar responsabilidad en caso de un fallo y perdida de la cosecha en consecuencia.

Apéndice B

Especificación de Requisitos

B.1. Introducción

Antes de proceder al diseño técnico y al desarrollo, es fundamental realizar una especificación detallada de los requisitos que el sistema debe cumplir. Este proceso permite definir de forma precisa las funcionalidades clave de la herramienta, como la predicción de las necesidades hídricas, la gestión de modelos de aprendizaje automático y la visualización de los resultados, sentando así las bases para un desarrollo estructurado y orientado a los objetivos del proyecto.

B.2. Objetivos generales

El objetivo principal de este proyecto es optimizar el rendimiento económico del cultivo del almendro mediante un sistema de riego deficitario controlado utilizando datos de sensores en campo y algoritmos de minería de datos que predicen con precisión las necesidades hídricas para reducir el consumo de agua sin mermar significativamente la producción, promoviendo así una gestión del agua más eficiente, sostenible y rentable.

Los objetivos generales del proyecto son los siguientes:

- **Mejorar el rendimiento económico del cultivo:** Al reducir el consumo de millones de litros de agua, reducimos gastos. Sin embargo, gracias a el uso inteligente del agua y aprovechando las diferentes fases fenológicas, no mermamos la producción, aumentando el rendimiento económico.

- **Facilitar la vida del agricultor:** Al proporcionarle una guía exacta con la cantidad de riego a realizar, facilitamos el seguimiento de un plan de riego que tradicionalmente, tendría que realizar manualmente el agricultor.
- **Demostrar la aplicabilidad de la tecnología en el campo:** Un objetivo claro desde el principio ha sido conseguir una exitosa implementación de los modelos de minería de datos probando la cabida de la tecnología en el campo, en linea con el movimiento de la agricultura 5.0 [1].

B.3. Catálogo de requisitos

Requisitos funcionales

RF-01. Gestión y preprocesamiento: El sistema debe importar archivos de datos, realizar un análisis previo y normalizar las variables para su tratamiento.

RF-02. Entrenamiento de modelos: Permitir al usuario entrenar diferentes modelos (SARIMA, VAR, etc.) utilizando conjuntos de datos específicos cargados desde la web.

RF-03. Predicción de riego: Generar predicciones semanales de riego basadas en los modelos entrenados, integrando los cálculos de la fórmula FAO-56.

RF-04. Aplicación de política RDC: Calcular automáticamente la dosis de riego óptima aplicando las reglas de Riego Deficitario Controlado sobre las predicciones.

RF-05. Visualización de resultados: Presentar métricas de rendimiento y gráficos comparativos de las predicciones a través de una interfaz interactiva.

RF-06. Carga de datos externa: Proporcionar una pasarela para que el usuario suba sus propios archivos de datos.

RF-07. Serialización y persistencia: Permitir el guardado y la carga posterior de modelos entrenados para realizar predicciones sin necesidad de reentrenamiento.

Requisitos No funcionales

RNF-01. Usabilidad: Interfaz intuitiva que permita a usuarios agrícolas gestionar modelos y visualizar gráficos de riego sin formación técnica previa.

RNF-02. Portabilidad: Aplicación accesible vía navegador web mediante despliegue en la nube, eliminando la necesidad de instalaciones locales de software.

RNF-03. Eficiencia: El procesamiento y entrenamiento de los modelos en Python debe realizarse en tiempos optimizados para garantizar la fluidez de la web.

RNF-04. Robustez de los datos: Capacidad de manejar y validar diversos formatos de entrada para asegurar la correcta ejecución de los algoritmos de predicción.

RNF-05. Mantenibilidad: Código estructurado de forma modular para facilitar la actualización de las fórmulas de riego o la inclusión de nuevos modelos estadísticos.

RNF-06. Disponibilidad: El sistema debe garantizar el acceso continuo a los datos históricos y a las recomendaciones de riego almacenadas en la plataforma.

B.4. Especificación de requisitos

CU-1	Registro de usuario
Versión	1.0
Autor	Alumno
Requisitos asociados	
Descripción	Permite a un nuevo usuario dar de alta una cuenta en la aplicación para acceder a las funciones de entrenamiento y predicción.
Precondición	El usuario debe acceder a la dirección URL de la aplicación y no tener una sesión activa.
Acciones	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de 'Registrarse'. 2. El sistema presenta un formulario solicitando datos (nombre de usuario, correo electrónico y contraseña). 3. El usuario cumplimenta los campos y envía el formulario. 4. El sistema valida que los datos sean correctos y que el usuario no exista previamente. 5. El sistema cifra la contraseña y almacena el nuevo perfil en la base de datos. 6. El sistema redirige al usuario a la pantalla de inicio de sesión o accede automáticamente.
Postcondición	Los datos del usuario quedan almacenados en el sistema y este puede autenticarse.
Excepciones	<ul style="list-style-type: none"> ■ El usuario ya existe en la base de datos. ■ Los datos introducidos no cumplen con el formato requerido (ej. contraseña débil o email inválido).
Importancia	Alta

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

CU-2 Inicio de sesión de usuario	
Versión	1.0
Autor	Alumno
Requisitos asociados	
Descripción	Permite a un usuario registrado identificarse en el sistema para acceder a las funcionalidades privadas de gestión de modelos y datos.
Precondición	<ul style="list-style-type: none"> ■ El usuario debe estar previamente registrado en el sistema. ■ El usuario no debe tener una sesión activa.
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la página de login de la aplicación web. 2. El usuario introduce sus credenciales (nombre de usuario y contraseña). 3. El usuario pulsa el botón de 'Iniciar sesión'. 4. El sistema verifica la existencia del usuario y la validez de la contraseña. 5. El sistema crea una sesión de usuario activa. 6. El sistema redirige al usuario al panel principal.
Postcondición	<ul style="list-style-type: none"> ■ El usuario obtiene acceso a las áreas restringidas de la aplicación. ■ Se mantiene la identidad del usuario durante la navegación.
Excepciones	<ul style="list-style-type: none"> ■ Credenciales incorrectas (usuario no existe o contraseña errónea). ■ Campos obligatorios vacíos en el formulario.
Importancia	Alta

Tabla B.2: CU-2 Inicio de sesión de usuario.

CU-3	Cierre de sesión de usuario
Versión	1.0
Autor	Alumno
Requisitos asociados	
Descripción	Permite al usuario finalizar su sesión activa en la aplicación, restringiendo nuevamente el acceso a las funciones privadas.
Precondición	El usuario debe estar autenticado en el sistema y tener una sesión activa.
Acciones	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de 'Cerrar sesión' en la interfaz. 2. El sistema recibe la petición y procede a la invalidación del token o sesión activa del servidor. 3. El sistema elimina las cookies de sesión del navegador del usuario. 4. El sistema redirige al usuario a la página de inicio pública.
Postcondición	<ul style="list-style-type: none"> ■ La sesión del usuario queda destruida. ■ El acceso a las funcionalidades de entrenamiento y predicción queda bloqueado hasta un nuevo inicio de sesión.
Excepciones	Ninguna.
Importancia	Media

Tabla B.3: CU-3 Cierre de sesión de usuario.

CU-4	Subir archivos de datos
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-01, RF-06
Descripción	Permite al usuario cargar ficheros con datos recolectados por los sensores para su posterior uso en las fases de entrenamiento de modelos o generación de predicciones de riego.
Precondición	El usuario debe haber iniciado sesión en la aplicación.
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de 'Subida de Archivos'. 2. El usuario pulsa sobre el botón 'Subir archivo'. 3. El usuario selecciona el archivo en su explorador de ficheros (formato .xlsx) que contiene los datos de la finca. 4. El usuario confirma la subida del archivo pulsando el botón 'Cargar'. 5. El sistema valida que el formato del archivo sea compatible y que la estructura de columnas sea la esperada. 6. El sistema almacena el archivo en el servidor y lo asocia al perfil del usuario. 7. El sistema muestra un mensaje de éxito notificando que el archivo está disponible.
Postcondición	El archivo queda almacenado y disponible para ser seleccionado en los procesos de entrenamiento y predicción.
Excepciones	<ul style="list-style-type: none"> ■ El archivo supera el tamaño máximo permitido. ■ El formato del archivo es incompatible. ■ Los datos del archivo no cumplen con la estructura requerida para el modelo de IA.
Importancia	Alta

Tabla B.4: CU-4 Subir archivos de datos.

CU-5	Configurar y Entrenar Modelo
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-02, RF-07
Description	Permite al usuario configurar los parámetros de entrenamiento, seleccionar un conjunto de datos y ejecutar el proceso de aprendizaje para generar un modelo capaz de predecir necesidades de riego.
Precondición	<ul style="list-style-type: none"> ■ El usuario ha iniciado sesión. ■ Existen archivos de datos cargados previamente en el sistema (CU-4).
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de 'Entrenamiento' de la aplicación. 2. El usuario selecciona el conjunto de datos (archivo) de entre los disponibles. 3. El usuario configura los parámetros del modelo. 4. El usuario pulsa el botón 'Iniciar Entrenamiento'. 5. El sistema realiza el preprocesamiento de los datos y ejecuta el entrenamiento del modelo de forma asíncrona. 6. El sistema muestra un mensaje de finalización y redirige a la ruta de predicción una vez terminado el proceso.
Postcondición	<ul style="list-style-type: none"> ■ Se genera un nuevo modelo entrenado y se guarda de forma serializada. ■ El modelo queda habilitado para realizar predicciones.
Excepciones	<ul style="list-style-type: none"> ■ Error en el entrenamiento debido a parámetros inconsistentes. ■ Fallo de memoria o tiempo de ejecución excedido en el servidor.
Importancia	Alta

Tabla B.5: CU-5 Configurar y Entrenar Modelo.

CU-6	Ejecutar Predicción RDC
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-03, RF-04, RF-05
Descripción	El sistema utiliza un modelo entrenado para predecir la necesidad hídrica y aplica la política de Riego Deficitario Controlado (RDC) para optimizar el suministro de agua semanal.
Precondición	<ul style="list-style-type: none"> ■ El usuario ha iniciado sesión. ■ Existe al menos un modelo entrenado y guardado (CU-5). ■ Se ha subido un conjunto de datos nuevos para predecir (CU-4).
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de 'Predicción'. 2. El usuario selecciona el modelo entrenado que desea utilizar. 3. El usuario selecciona el archivo de datos recientes de la finca. 4. El usuario pulsa el botón 'Ejecutar Predicción'. 5. El sistema carga el modelo serializado y procesa los nuevos datos. 6. El sistema calcula la necesidad de riego teórica y aplica el factor de corrección según la política RDC. 7. El sistema muestra los resultados en pantalla mediante gráficas y tablas de riego semanal.
Postcondición	El sistema genera una recomendación de riego optimizada lista para su visualización y descarga.
Excepciones	<ul style="list-style-type: none"> ■ El modelo seleccionado no es compatible con la estructura de los nuevos datos cargados. ■ Error en el motor de cálculo al aplicar la lógica RDC.
Importancia	Muy Alta

Tabla B.6: CU-6 Ejecutar Predicción RDC.

CU-7	Descargar Resultados
Versión	1.0
Autor	Alumno
Requisitos asociados	
Descripción	Permite al usuario exportar los resultados de las predicciones de riego y las métricas de los modelos en un formato de archivo local para su consulta externa o registro.
Precondición	El usuario debe haber generado resultados de predicción previamente y encontrarse en la vista de visualización (CU-6).
Acciones	<ol style="list-style-type: none"> 1. El usuario selecciona la opción 'Descargar' o 'Exportar' dentro de la pantalla de resultados. 2. El sistema solicita al usuario que elija el formato de salida (ej. CSV, Excel o PDF). 3. El usuario confirma la acción pulsando el botón de descarga. 4. El sistema recupera la información de la estructura de datos 'PREDICTION_RESULTS' y genera el archivo correspondiente. 5. El sistema inicia la transferencia del archivo a través del navegador del usuario.
Postcondición	El usuario dispone de un archivo local con la planificación de riego RDC.
Excepciones	<ul style="list-style-type: none"> ■ Error en la generación del archivo por parte del servidor. ■ Interrupción de la conexión de red durante la descarga.
Importancia	Media

Tabla B.7: CU-7 Descargar Resultados.

Apéndice C

Especificación de diseño

C.1. Introducción

El diseño del sistema se ha concebido bajo premisas de modularidad, escalabilidad y robustez, garantizando que la infraestructura soporte de manera eficiente. A continuación, se describen los pilares fundamentales que componen esta arquitectura:

- **Persistencia y Gestión de Datos:** Se ha implementado un sistema de gestión de bases de datos relacionales basado en **PostgreSQL**. Esta decisión responde a una visión estratégica de futuro, priorizando su capacidad de escalabilidad y fiabilidad ante el previsible aumento de la densidad de datos y usuarios. Para facilitar la interacción con la base de datos y abstraer la lógica de persistencia, se emplea el **ORM SQLAlchemy**, integrando la gestión de usuarios y datasets de forma cohesiva dentro de la POO.
- **Arquitectura de Servicios:** La aplicación web sigue un paradigma de diseño **API REST**, permitiendo una comunicación desacoplada entre el cliente y el servidor. Para garantizar la portabilidad y la consistencia en los diferentes entornos de ejecución, los servicios se han desplegado mediante **contenedores Docker**, orquestados a través de **Docker Compose**. Este enfoque facilita enormemente el despliegue y la replicación de este sistema en otros entornos de forma plug and play.
- **Estructura de Modelado:** En el ámbito de la computación, se ha adoptado una arquitectura de **pipelines de Scikit-learn**. Para ase-

gurar la compatibilidad y flexibilidad del sistema, se han desarrollado **wrappers** específicos que adaptan los modelos de predicción a estas estructuras secuenciales, optimizando así los procesos de entrenamiento, predicción y evaluación.

C.2. Diseño de datos

Esta sección detalla la organización y el modelado de la información del sistema, abarcando desde la persistencia relacional para la gestión de la aplicación hasta las estructuras dinámicas empleadas en el procesamiento de datos de sensores y el seguimiento de procesos de aprendizaje automático.

Aplicación web

Diseño de datos de usuario y datasets

Para la gestión de la persistencia de datos de la plataforma, se ha diseñado un esquema relacional implementado en **PostgreSQL**. La elección de una base de datos externa frente a estructuras de datos volátiles en memoria (como diccionarios de Python) responde a la necesidad de garantizar la persistencia, integridad y concurrencia de la información, permitiendo que el sistema sea escalable y profesional.

El modelado de datos se ha realizado mediante el **ORM SQLAlchemy**, lo que permite interactuar con la base de datos a través de objetos de Python, facilitando el mantenimiento y la legibilidad del código. El esquema se divide principalmente en dos entidades relacionadas:

- **Entidad User:** Gestiona la información de acceso de los usuarios. Esta clase hereda de `UserMixin` para integrarse de forma nativa con la librería **Flask-Login**, facilitando la gestión de sesiones y el control de acceso. Por seguridad, no se almacenan contraseñas en texto plano, sino que se utilizan funciones de *hashing* mediante **Werkzeug**.

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True,
    nullable=False)
    email = db.Column(db.String(120), unique=True,
    nullable=False)
    password_hash = db.Column(db.String(255), nullable=False)

    datasets = db.relationship("Dataset", backref="user",
    lazy=True)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash,
        password)
```

Figura C.1: Implementación de la clase User en Python.

- **Entidad Dataset:** Almacena los metadatos de los archivos subidos por los usuarios (ruta en disco, nombre y fecha). Existe una relación de *uno a muchos* (**one-to-many**) entre el usuario y sus datasets, permitiendo que cada usuario gestione su propio histórico de datos de forma aislada.

```
class Dataset(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    filename = db.Column(db.String(120), nullable=False)
    uploaded_at = db.Column(db.DateTime, default=datetime.
        utcnow)
    filepath = db.Column(db.String(200), nullable=False)  #
    ruta en disco
    user_id = db.Column(db.Integer, db.ForeignKey("user.id"),
        nullable=False)
```

Figura C.2: Implementación de la clase Dataset en Python.

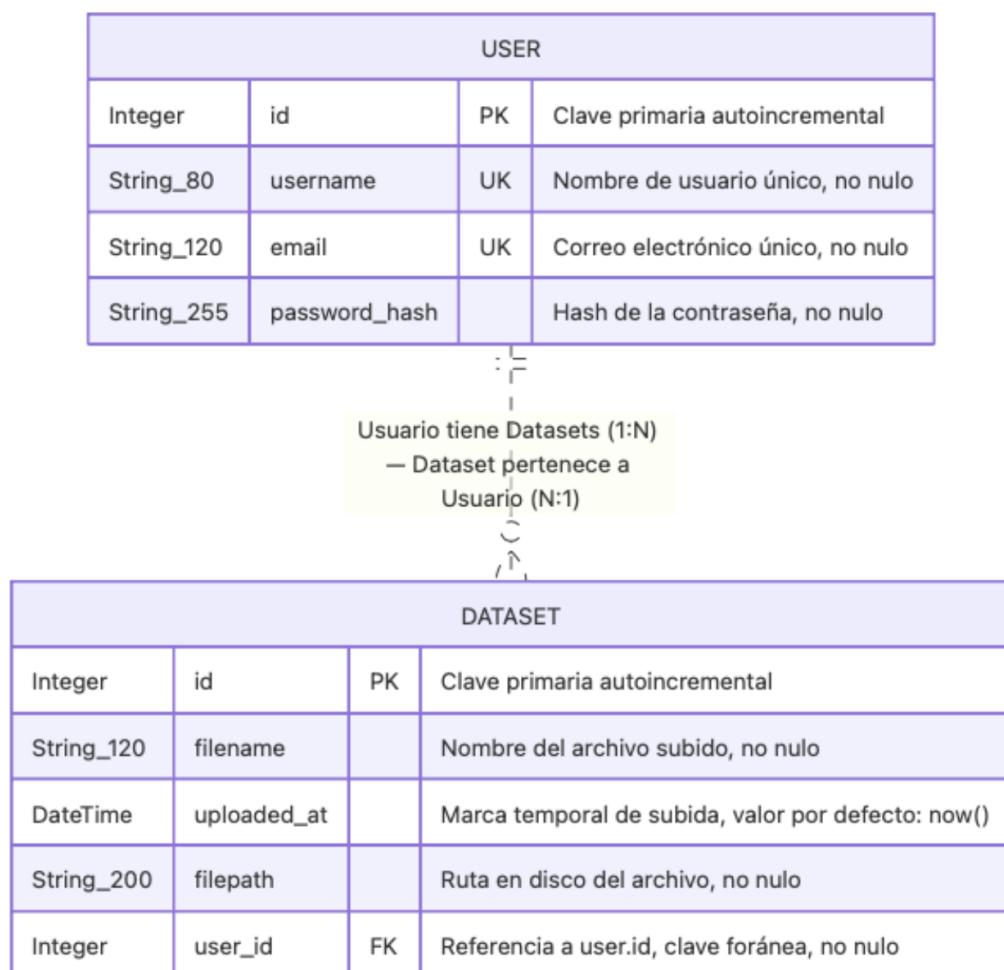


Figura C.3: Diagrama de la base de datos relacional

Para la evolución del esquema, se ha incorporado **Alembic** como herramienta de migraciones. Esto permite llevar un control de versiones de la estructura de la base de datos, facilitando cambios futuros sin pérdida de información y garantizando un flujo de trabajo *production-ready*.

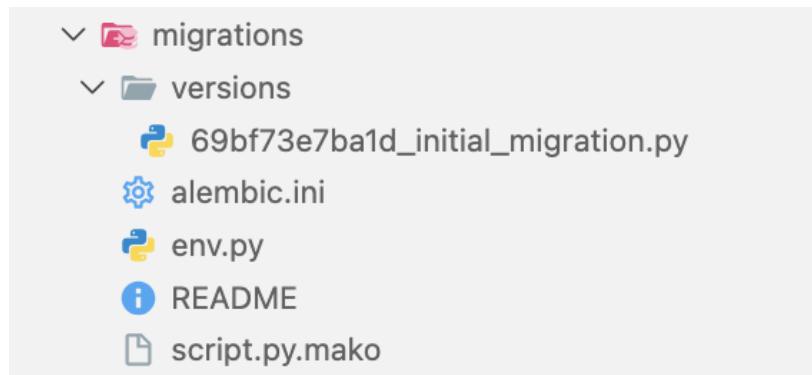


Figura C.4: Estructura carpeta migraciones

Finalmente, aunque el servicio se despliega de forma aislada en un contenedor Docker, su comunicación con la aplicación Flask está optimizada mediante redes virtuales de Docker, asegurando un entorno de ejecución seguro y portable.

Diseño estructuras de seguimiento de progreso de entrenamiento/predicción

Debido a que las tareas de entrenamiento y predicción son procesos computacionalmente costosos y de larga duración, se ha diseñado un sistema de comunicación entre el *backend* y el *frontend* para proporcionar actualizaciones de estado en tiempo real.

Como se detalla en el apartado de *Creación de rutas de seguimiento de progreso* Sección 5.4, el diseño evolucionó desde un almacenamiento en la sesión de Flask —limitado por su naturaleza síncrona y su restricción de tamaño de 4KB— hacia una solución de persistencia desacoplada. El diseño final se consolida en el módulo `state.py`, donde se definen las siguientes estructuras globales:

- **PREDICTION_PROGRESS**: Un diccionario encargado de mapear el identificador del proceso con su porcentaje de ejecución actual, permitiendo consultas asíncronas desde la interfaz.

- **PREDICTION_RESULTS:** Una estructura destinada a albergar los objetos resultantes de las predicciones, incluyendo los gráficos generados que excedían las capacidades de la sesión estándar.

Aunque el uso de variables globales en `state.py` resuelve eficazmente los problemas de concurrencia y límites de memoria detectados durante el desarrollo, se reconoce como un punto de mejora para la escalabilidad del sistema. En una fase de producción más avanzada, estas estructuras deberían migrarse a un sistema de mensajería o una base de datos en memoria como **Redis**, gestionada por trabajadores de tareas asíncronas como **Celery**. No obstante, dadas las restricciones temporales del proyecto, se priorizó esta implementación por su robustez y simplicidad, permitiendo centrar los esfuerzos en la optimización de los modelos de riego.

Datos de sensores y modelado de DataFrames

Los datos de origen consisten en series temporales capturadas por estaciones agroclimáticas y sensores de humedad de suelo, con una frecuencia de registro de 15 minutos. Como se desarrolla extensamente en [Sección 5.2](#), la información se presenta originalmente en múltiples ficheros anuales (3) diferenciados por su ubicación física (grupos mini1 y mini2) y naturaleza (datos climáticos). Los datos fueron concatenados por años para formar un fichero único de cada tipo. Posteriormente, los datos de suelo (mini1 y mini2) tras un estudio fueron combinados siguiendo la media como estrategia. Finalmente, el dataframe de datos de suelo combinado y el de clima, son unidos en un gran dataframe.

	Fecha	Contenido volumétrico (10 cm)	Contenido volumétrico (20 cm)	Contenido volumétrico (40 cm)	Contenido volumétrico (60 cm)	Agua (%)	Temperatura en suelo (10 cm)	Temperatura en suelo (20 cm)	Temperatura en suelo (40 cm)	Temperatura en suelo (60 cm)
0	2023-01-01 00:00:00	14.0	29.5	30.7	33.9	66.58074	10.8	10.8	11.4	11.4
1	2023-01-01 00:15:00	14.0	29.5	30.7	33.9	66.58074	10.7	10.8	11.4	11.4
2	2023-01-01 00:30:00	14.0	29.5	30.7	33.9	66.58074	10.6	10.8	11.4	11.4
3	2023-01-01 00:45:00	14.0	29.5	30.7	33.9	66.58074	10.6	10.8	11.4	11.4
4	2023-01-01 01:00:00	13.9	29.5	30.7	33.9	66.543074	10.6	10.8	11.4	11.4

89883	2025-10-08 22:00:00	28.7	27.6	31.1	32.7	67.52517	18.6	18.8	18.4	17.9
89884	2025-10-08 23:00:00	28.7	27.6	31.1	32.7	67.52517	18.6	18.8	18.4	17.9
89885	2025-10-08 23:15:00	28.6	27.6	31.1	32.7	67.52517	18.4	18.7	18.4	17.9
89886	2025-10-08 23:30:00	28.6	27.6	31.1	32.7	67.52517	18.3	18.6	18.4	17.9
89887	2025-10-08 23:45:00	28.6	27.6	31.1	32.7	67.52517	18.3	18.6	18.4	17.9

Figura C.5: Dataframe Mini1 combinado anualmente

Fecha	Contenido volumétrico (0 cm)	Contenido volumétricos (20 cm)	Contenido volumétricos (40 cm)	Contenido volumétricos (60 cm)	Agua útil	Temperatura en suelo (10 cm)	Temperatura en suelo (20 cm)	Temperatura en suelo (40 cm)	Temperatura en suelo (60 cm)
0 2023-01-01 00:00:00	20.3	25.0	21.9	21.7	47.13496	9.9	10.6	11.4	11.6
2023-01-01 00:30:00	20.3	25.0	21.9	21.7	47.13496	9.9	10.6	11.4	11.6
3 2023-01-01 00:45:00	20.3	25.0	21.9	21.7	47.13496	9.9	10.6	11.4	11.6
4 2023-01-01 01:00:00	20.3	25.0	21.9	21.7	47.13496	9.9	10.6	11.4	11.6
83380 2025-05-08 00:45:00
83380 2025-05-08 00:45:00	25.4	24.4	25.8	18.8	53.813496	18.7	19.2	18.7	18.7
83380 2025-05-08 00:45:00	25.4	24.4	25.8	18.8	53.813496	18.6	19.1	18.7	18.7
83380 2025-05-08 00:45:00	25.4	24.4	25.8	18.8	53.813496	18.5	19.1	18.7	18.7
83382 2025-10-08 23:30:00	25.3	24.4	25.8	18.8	53.813496	18.4	19.1	18.2	18.7
83383 2025-10-08 23:45:00	25.3	24.4	25.8	18.8	53.813496	18.4	19.0	19.2	18.7

Figura C.6: Dataframe Mini2 combinado anualmente

Fecha	Índice de calor	Humedad relativa	Presión de vapor	Radiación solar	Temperatura	Punto de rocío	Temperatura del bulbo húmedo	Velocidad del viento	Racha de viento máxima	Dirección del viento	Humectación en hoja (porcentaje)	Presión atmosférica	Precipitaciones
0 2023-01-01 00:00:00	11.6	81.0	14.160000	0.0	11.6	1720000	7.280329	10.0	17.0	180.0	0.0	940.6	0.0
1 2023-01-01 00:15:00	11.2	52.0	13.810000	0.0	11.2	1620000	7.038901	7.0	10.0	225.0	0.0	940.8	0.0
2 2023-01-01 00:30:00	11.1	58.0	13.720000	0.0	11.1	3.070000	7.050979	7.0	12.0	180.0	0.0	940.8	0.0
3 2023-01-01 00:45:00	11.7	54.0	14.250000	0.0	11.7	2.820000	7.046474	13.0	18.0	180.0	0.0	940.7	0.0
4 2023-01-01 01:00:00	11.3	56.5	13.830000	0.0	11.3	2.780000	7.049861	11.0	20.0	180.0	0.0	940.8	0.0
86998 2025-10-08 22:45:00	15.4	80.0	17.965096	0.0	15.4	11937903	13.891570	6.0	10.0	22.5	0.0	939.7	0.0
86997 2025-10-08 22:45:00	15.3	80.0	17.944727	0.0	15.3	11.840528	13.779991	6.0	12.0	315.0	0.0	940.0	0.0
86998 2025-10-08 22:45:00	15.2	80.0	17.944727	0.0	15.2	11.840528	13.779991	6.0	12.0	450.0	0.0	940.1	0.0
86999 2025-10-08 23:00:00	15.2	80.0	17.734981	0.0	15.2	11.743854	13.284523	6.0	12.0	0.0	0.0	940.2	0.0
97000 2025-10-08 23:45:00	15.2	81.0	17.734981	0.0	15.2	11.932791	13.329255	6.0	15.0	9.0	0.0	940.1	0.0

Figura C.7: Dataframe de clima combinado anualmente

Fecha	Contenido volumétrico (0 cm)	Contenido volumétrico (20 cm)	Contenido volumétrico (40 cm)	Contenido volumétrico (60 cm)	Agua útil	Temperatura en suelo (10 cm)	Temperatura en suelo (20 cm)	Temperatura en suelo (40 cm)	Temperatura en suelo (60 cm)	Índice de calor	Punto de rocío	Temperatura del bulbo húmedo	Velocidad del viento	Racha de viento máxima	Dirección del viento	Humectación en hoja (porcentaje)	Presión atmosférica	Radiación solar	Precipitaciones	Humedad relativa mínima
2023-01-01 00:00:00	27.048438	27.232812	26.251063	27.794271	56.814266	11.004167	10.950208	11.048333	11.480208	12.891667	6.075147	9.020438	13.861867	20.531260	176.484373	2.770783	937.698675	5820.0	3.0	42.0
2023-01-01 00:15:00	27.048456	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 00:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 00:45:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 01:00:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 01:15:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 01:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 01:45:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 02:00:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 02:15:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 02:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 02:45:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 03:00:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 03:15:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 03:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 03:45:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 04:00:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 04:15:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 04:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 04:45:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 05:00:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 05:15:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.869271	11.054602	11.953648	12.895028	4.751562	5.908668	4.039750	7.322997	198.063795	0.500000	942.697708	4620.0	1.0	68.0
2023-01-01 05:30:00	27.048457	27.236456	26.228644	27.770833	56.926798	10.421675	10.8692													

Fecha	Contenido volumétrico (10 cm)	Contenido volumétrico (20 cm)	Contenido volumétrico (30 cm)	Contenido volumétrico (40 cm)	Aqua-útil	Temperatura en suelo (10 cm)	Temperatura en suelo (20 cm)	Temperatura en suelo (30 cm)	Temperatura en suelo (40 cm)	Índice de color	Punto de rocío	Temperatura del bulbo húmedo	Velocidad del viento	Ratio de viento máxima	Dirección del viento	Humedad en hoja (porcentaje)	Presión atmosférica	Radiación solar	Precipitaciones	Humedad relativa mínima	
2023-01-01	77048418	27233812	26.35163	27.97437	58.814868	11.00497	10.951208	11.458333	11.891907	-	4.67447	9.020438	13.85487	20.512150	176.484875	217.088	937.698676	5826.0	3.0	42.0	
2023-01-02	
2023-01-03	77048958	27234548	26.228848	27.70538	58.826758	10.425875	10.885271	11.564062	11.953848	-	4.751982	9.059888	13.85529	20.697950	176.484875	186.09750	0.500000	942.617708	4622.0	1.0	68.0
2023-01-04	16.934074	27234449	26.170553	27.70598	58.284717	7.348938	8.566489	10.481383	11.201064	-	5.016598	9.058484	13.79787	16.638350	179.7987	165.38350	0.595748	942.270715	3837.0	0.0	83.0
2023-01-05	16.819065	26.938021	26.036512	27.70984	58.993025	6.419229	7.303646	9.194271	10.313064	-	5.032042	9.04022	12.79987	13.53333	174.699375	2.052083	932.208835	3282.0	0.2	97.0	
2023-01-06	16.848437	26.936671	25.992708	27.84797	55.650979	6.493729	7.750896	8.73547	9.793572	0.539983	0.527997	0.539968	1.333333	2.700417	91.874000	5.562290	948.170642	3054.0	0.0	99.0	
2023-01-07	
2023-01-08	273.0729	26.34479	28.75521	25.805042	61.986537	18.602771	18.428125	18.606538	18.661852	18.423438	18.000501	15.796206	18.027981	4.687100	181.426250	0.000000	940.115642	14464.0	0.0	44.0	
2023-01-09	26.677083	25.438642	28.399867	25.687500	57.731826	17.705279	18.264370	18.819087	18.635617	18.037570	17.322381	10.142863	18.886833	7.802063	78.760000	0.000000	942.435417	15550.0	0.0	47.0	
2023-01-10	272.18979	26.346075	28.779107	25.647356	62.337020	16.951075	17.371154	18.273646	18.603333	18.435258	17.723663	14.020000	18.021000	4.002100	110.390025	0.116583	942.463542	19573.0	0.0	28.0	
2023-01-11	29.2709833	28.032813	29.769979	26.309250	60.298430	17.345750	17.650521	18.184958	18.167050	14.869887	17.97627	0.520683	17.876750	81.328105	0.000000	941.244865	14898.0	0.0	37.0		
2023-01-12	27.379521	26.016071	28.738642	25.921875	62.813956	18.143750	18.186448	18.433468	18.749792	9.803297	12.884810	2.145683	4.739563	7.499005	0.966333	930.396875	13752.0	1.2	38.0		

Figura C.9: Dataframe diario combinado e interpolado

- Ingeniería de variables y reducción de dimensionalidad:** El volumen original de atributos fue reducido a un total de 7 variables críticas. Esta selección no es arbitraria, sino que responde estrictamente a las necesidades de la fórmula de **Monteith-Penman** [2] para el cálculo de la evapotranspiración. Entre estas, destaca la creación de la columna *Humedad relativa mínima*, derivada mediante el análisis de los picos de humedad diarios del conjunto original.

Fecha	Humedad relativa	Temperatura	Velocidad del viento	Presión atmosférica	Radiación solar	Precipitaciones	Humedad relativa mínima
2023-01-01	58.583333	12.891667	13.854167	937.696875	5826.0	3.0	42.0
2023-01-02	87.062500	6.855208	4.093750	942.617708	4620.0	1.0	68.0
2023-01-03	96.882979	0.775532	0.691489	949.270213	3837.0	0.0	83.0
2023-01-04	99.197917	0.633333	1.729167	952.208333	3282.0	0.2	97.0
2023-01-05	99.916667	0.539583	1.333333	948.170642	3054.0	0.0	99.0
...
2025-10-04	63.895833	18.047917	1.927083	940.113542	14664.0	0.0	44.0
2025-10-05	70.437500	13.093750	3.895833	942.435417	15558.0	0.0	47.0
2025-10-06	67.520833	13.386458	1.625000	942.463542	15573.0	0.0	28.0
2025-10-07	64.229167	14.877083	0.520833	941.246875	14898.0	0.0	37.0
2025-10-08	66.125000	16.794792	2.145833	939.396875	13752.0	1.2	39.0

Figura C.10: Dataframe Mini1 combinado anualmente

Finalmente, para solventar la escasez de ciclos estacionales reales, se ha diseñado un sistema de expansión del *dataset* mediante la generación de **datos sintéticos**. Utilizando el DataFrame original como base, se han generado 5 años adicionales de registros mediante las funciones *generar_datos_fisicos_pasado* y *bootstrap_estacional_perturbado_pasado*.

Fecha	Humedad relativa	Temperatura	Velocidad del viento	Presión atmosférica	Radiación solar	Precipitaciones	Humedad relativa mínima
2019-01-02	90.128868	6.007436	5.795502	938.634024	3367.231362	0.000000	75.808482
2019-01-03	84.395362	4.485541	7.065581	936.690247	3412.164152	0.000000	78.063128
2019-01-04	91.437993	6.384803	5.575320	940.942082	2126.165240	0.282648	75.156331
2019-01-05	86.913963	5.879959	5.972289	941.624613	4463.718728	0.000000	81.287777
2019-01-06	88.892900	7.266199	4.608092	938.412480	2982.080895	2.275458	68.211840
...
2025-10-04	63.895833	18.047917	1.927083	940.113542	14664.000000	0.000000	44.000000
2025-10-05	70.437500	13.093750	3.895833	942.435417	15558.000000	0.000000	47.000000
2025-10-06	67.520833	13.386458	1.625000	942.463542	15573.000000	0.000000	28.000000
2025-10-07	64.229167	14.877083	0.520833	941.246875	14898.000000	0.000000	37.000000
2025-10-08	66.125000	16.794792	2.145833	939.396875	13752.000000	1.200000	39.000000

Figura C.11: Dataframe final con datos sintéticos físicos agregados

Este diseño destaca por su escalabilidad, ya que el sistema permite ampliar el horizonte temporal de entrenamiento simplemente ajustando el parámetro `años_generar` en la lógica del preprocesamiento, lo que otorga a la aplicación la capacidad de adaptarse a escenarios de entrenamiento más profundos si la capacidad de cómputo lo permite.

C.3. Diseño arquitectónico

Este apartado detalla la estructura técnica y lógica del sistema, definiendo la organización de sus componentes y las tecnologías que garantizan un funcionamiento eficiente, escalable y modular.

En la siguiente sección se explica el diseño arquitectónico seguido, que define la organización global del software, estableciendo cómo interactúan el motor de inteligencia artificial, la base de datos y la interfaz de usuario.

Estructura Docker

Para garantizar la portabilidad y la homogeneidad del entorno de ejecución, se ha adoptado una arquitectura basada en contenedores utilizando **Docker**. Este enfoque permite aislar los servicios, gestionar las dependencias de Python de forma estricta y facilitar la réplica del proyecto en cualquier sistema sin conflictos de configuración.

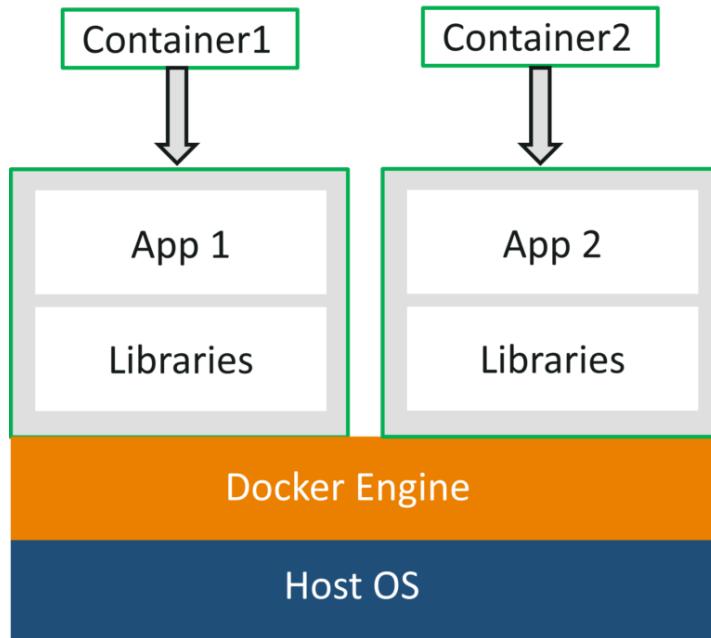


Figura C.12: Estructura funcionamiento de Docker

El despliegue se gestiona mediante `docker-compose`, un despliegue simplificado de ambos servicios (backend/frontend y base de datos) en un solo comando.

Detalles de la implementación: El sistema se compone de dos contenedores principales interconectados:

- **Contenedor de Aplicación (Flask):** Construido a partir de una imagen de Python mediante un *Dockerfile* específico, donde se instalan las librerías de procesamiento de datos y el servidor web.
- **Contenedor de Base de Datos (PostgreSQL):** Utiliza una imagen oficial de Postgres para el almacenamiento de datos de sesión y dataframes [C.2](#).

Ambos contenedores están conectados a través de una red de tipo *bridge*. El contenedor de Flask se comunica con la base de datos a través del puerto 5432, aprovechando la resolución de nombres interna de Docker para asegurar una conexión estable y segura entre servicios.

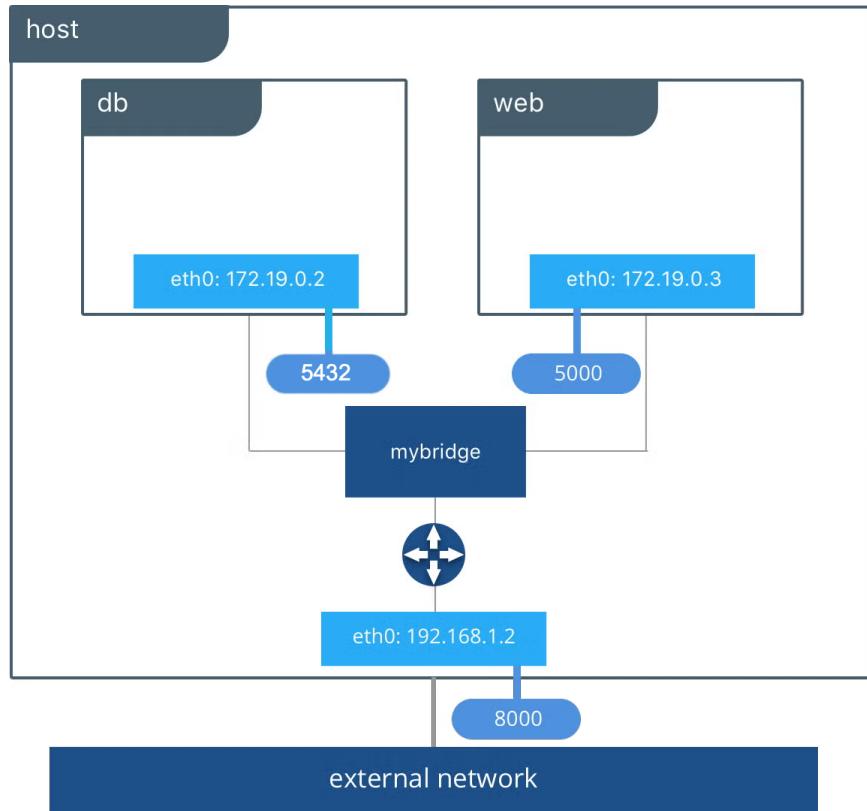


Figura C.13: Esquema específico de mis contenedores y su comunicación

Separación de la importación de datos

Debido al gran volumen de información manejado, que supera las 80,000 instancias por cada sensor, se ha decidido desacoplar el proceso de carga de datos del flujo principal de ejecución.

Esta lógica se ha implementado en un fichero independiente denominado `loader.py`, el cual es referenciado en el *Jupyter Notebook*. El objetivo de esta separación es reducir significativamente el tiempo de carga desde los ficheros originales `.xlsx` y permitir una carga de datos más ágil en memoria.

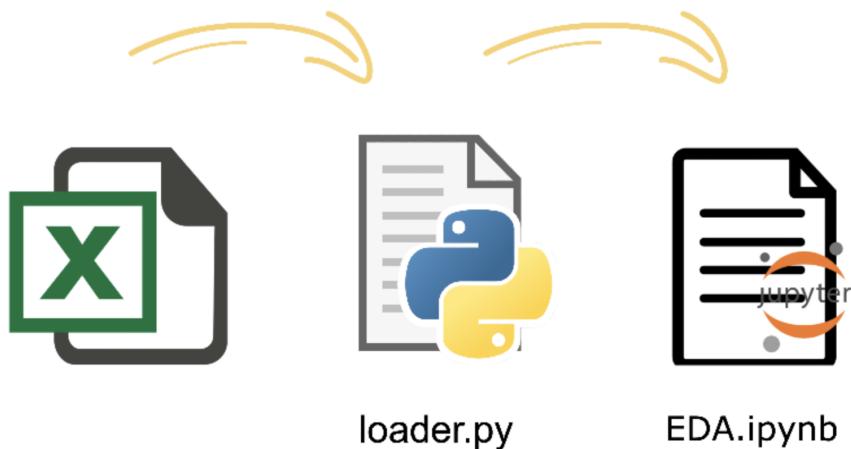


Figura C.14: Separación estructura entrenamiento/datos

API REST

La interacción entre la interfaz web y el núcleo de predicción se rige bajo una arquitectura de estilo **API REST**. Esta estructura organiza la comunicación mediante métodos estándar (GET, POST, etc.), lo que permite que el cliente web solicite entrenamientos o predicciones de forma asíncrona [19].

Los beneficios de aplicar este diseño en la aplicación incluyen:

- **Escalabilidad:** Permite que la lógica de predicción de riego pueda ser consumida por otros clientes o dispositivos en el futuro.
- **Independencia:** El frontend puede ser modificado o actualizado sin afectar al motor de Machine Learning en Python.
- **Mantenibilidad:** Facilita la depuración de errores al tener rutas claras para cada funcionalidad (subida de datos, visualización y ejecución de modelos).

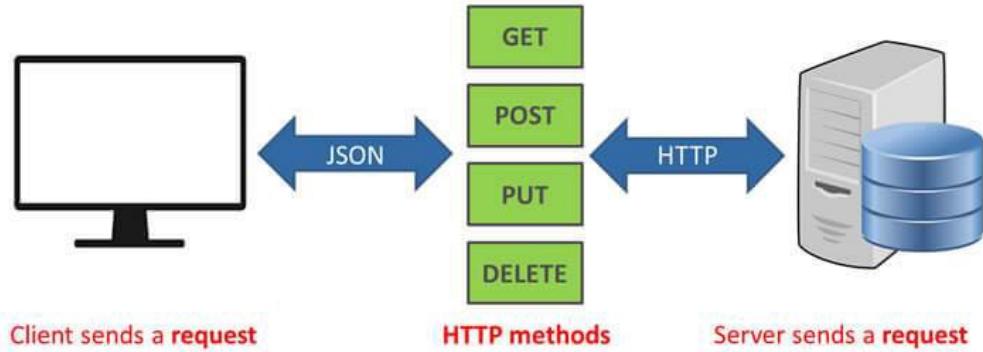


Figura C.15: Estructura API REST aplicada a mi aplicación

Reestructuración arquitectura entrenamiento y evaluación

Durante el desarrollo de LSTM, mi notebook no estaba preparado para la fácil inclusión de nuevos algoritmos de diferentes librerías a las ya incluidas en el notebook. Se realizaba una división de los datos en conjuntos de test y entrenamiento, escalado general de los datos, y no se establecían pipelines para que el proceso fuese secuencial exponiéndonos a errores de fuga de datos(data leakage), y alto riesgo de error humano al cambiar algo, ya que debo aplicar cambios en múltiples lugares.

Reestructuramos el proceso de entrenamiento, predicción y evaluación para adaptar el modelo de scikit-learn [17] de Pipelines.

Realizamos Wrappers personalizados a aquellos modelos provenientes de librerías fuera del entorno scikit-learn para adaptar el modelo de Pipelines (SARIMA, SARIMAX, VAR y LSTM) heredando de las clases:

- **BaseEstimator** [21]: Clase base fundamental que proporciona los métodos estándar `get_params()` y `set_params()` para la gestión de

hiperparámetros. Al heredar de esta clase, los *wrappers* personalizados adquieren compatibilidad nativa con herramientas de scikit-learn como `GridSearchCV`, `RandomizedSearchCV` y `Pipeline`, permitiendo que modelos externos se integren transparentemente en el ecosistema.

- **TransformerMixin** [22]: *Mixin* que implementa automáticamente el método `fit_transform()` combinando `fit()` y `transform()`. Esta clase es esencial para crear transformadores personalizados que puedan encadenarse en *pipelines*, manteniendo la consistencia del patrón `fit/transform`. Los *wrappers* que implementan preprocesamiento específico deben heredar de esta clase para garantizar interoperabilidad completa.

```

class SarimaModel(BaseEstimator, TransformerMixin):
    """Wrapper de SARIMA para scikit-learn"""

    def __init__(self, column: str, order: tuple = (1,1,1),
                 seasonal_order: tuple = (1,1,1,30), trend: str = 'c'):
        self.column = column
        self.order = order
        self.seasonal_order = seasonal_order
        self.trend = trend
        self.model = None
        self.fitted_model = None

    def fit(self, X, y=None):
        # Asegurarse de que X es una Serie temporal
        if isinstance(X, pd.DataFrame):
            series = X[self.column]
        else:
            series = X
        self.model = SARIMAX(series,
                             order=self.order,
                             seasonal_order=self.seasonal_order,
                             trend=self.trend,
                             enforce_stationarity=False,
                             enforce_invertibility=False)

        self.fitted_model = self.model.fit(disp=False)
        return self

    def predict(self, X, n_periods: int = 240):
        # Predicción para n periodos futuros
        forecast = self.fitted_model.get_forecast(steps=n_periods)
        return forecast.predicted_mean

    def get_params(self, deep=True):
        return {'column': self.column,
                'order': self.order,
                'seasonal_order': self.seasonal_order,
                'trend': self.trend}

    def set_params(self, **parameters):
        for parameter, value in parameters.items():
            setattr(self, parameter, value)
        return self

```

Figura C.16: Ejemplo de implementación de SARIMA siguiendo la estructura propuesta

Aprovechamos esta reestructuración para guardar los modelos y poder cargarlos desde archivo evitando tener que reentrenar tras reiniciar el notebook.

C.4. Diseño procedimental

Esta sección detalla el diseño procedimental del sistema, describiendo la lógica de interacción y el flujo de ejecución entre componentes durante las diferentes llamadas de la aplicación web. Cabe destacar que, a diferencia de la plataforma web, el Jupyter Notebook no contiene componentes diferenciados, por lo que no existen llamadas internas que detallar en este análisis.

Entrenamiento de modelos

1. El usuario accede a la sección de 'Entrenamiento' de la aplicación.
2. El backend recibe la petición a la ruta correspondiente y redirige a la página necesaria.
3. El frontend carga los archivos disponibles y los muestra en la interfaz.
4. El usuario selecciona el conjunto de datos de entre los ficheros disponibles y configura los parámetros específicos del modelo. Finalmente, El usuario pulsa el botón 'Iniciar Entrenamiento' para lanzar el proceso.
5. El sistema realiza el preprocesamiento de los datos y ejecuta el entrenamiento del modelo de forma asíncrona. Mientras, envía actualizaciones asíncronas sobre el estado del proceso mientras el frontend las muestra en pantalla. Al finalizar el entrenamiento, el backend notifica al frontend para que este redirija al usuario a la pantalla de predicción.

Nota: No se especifica el flujo de predicción dada su similitud con el proceso de entrenamiento a nivel de llamadas entre componentes.



Figura C.17: Diagrama de secuencia de proceso de entrenamiento

Gestión de usuarios: Registro

1. El usuario selecciona la opción de 'Registrarse' en la interfaz.
2. El backend procesa la petición y devuelve la página de registro que contiene el formulario correspondiente.
3. El usuario cumplimenta los campos requeridos y realiza el envío del formulario.
4. El backend valida la integridad de los datos y verifica que el usuario no exista previamente en la base de datos. Se cifra la contraseña y almacena el nuevo perfil de usuario en la base de datos de forma segura.
5. El sistema redirige al usuario a la pantalla de inicio de sesión o le concede acceso automático a la plataforma.

Nota: Se detalla el proceso de registro por su mayor complejidad al crear nuevas entradas en la base de datos; el resto de procesos de gestión de usuarios omiten su descripción debido a su similitud técnica.

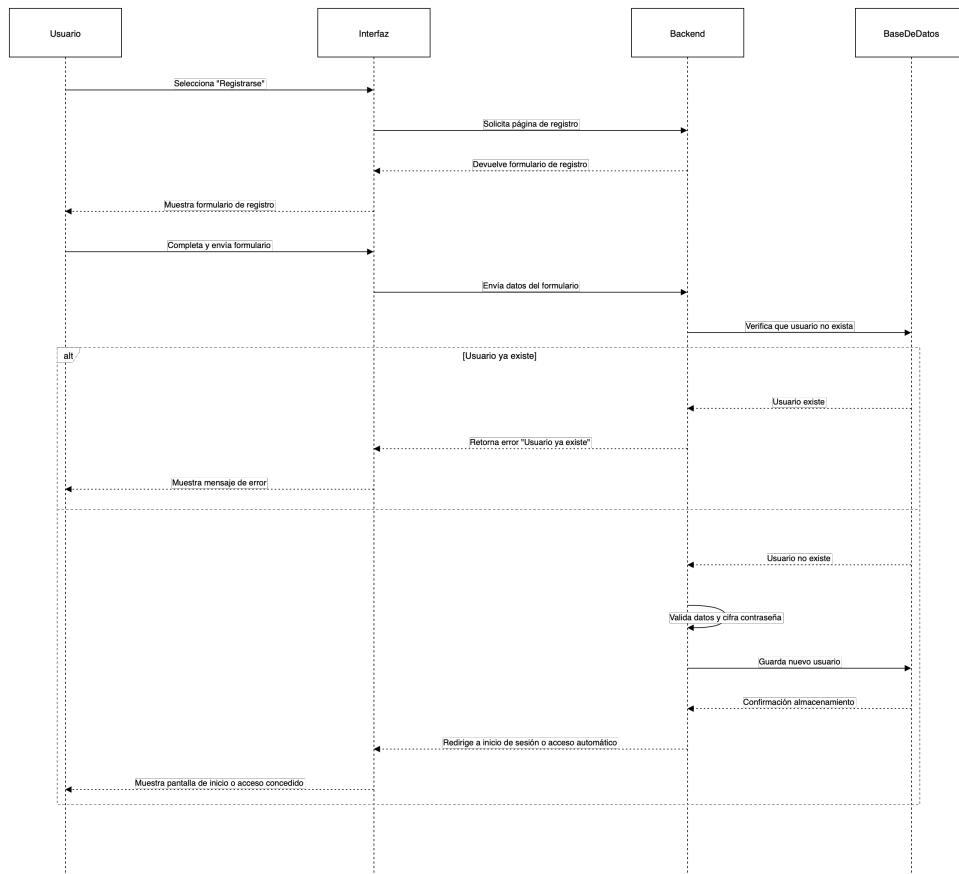


Figura C.18: Diagrama de secuencia de proceso de Registro de un usuario

Subir dataset

1. El usuario accede a la sección de 'Subida de Archivos' de la aplicación.
2. El backend recibe la petición en la ruta correspondiente y redirige a la página de subida de datos.

3. El usuario selecciona el archivo en su explorador de archivos en formato .xlsx que contiene los datos de la finca y confirma la subida del archivo pulsando el botón 'Cargar'.
4. El backend valida que el formato del archivo sea compatible y que la estructura de columnas sea la esperada por el sistema. En caso afirmativo, almacena el archivo en la base de datos, lo asocia al perfil del usuario y muestra un mensaje de éxito notificando que el archivo ya está disponible.

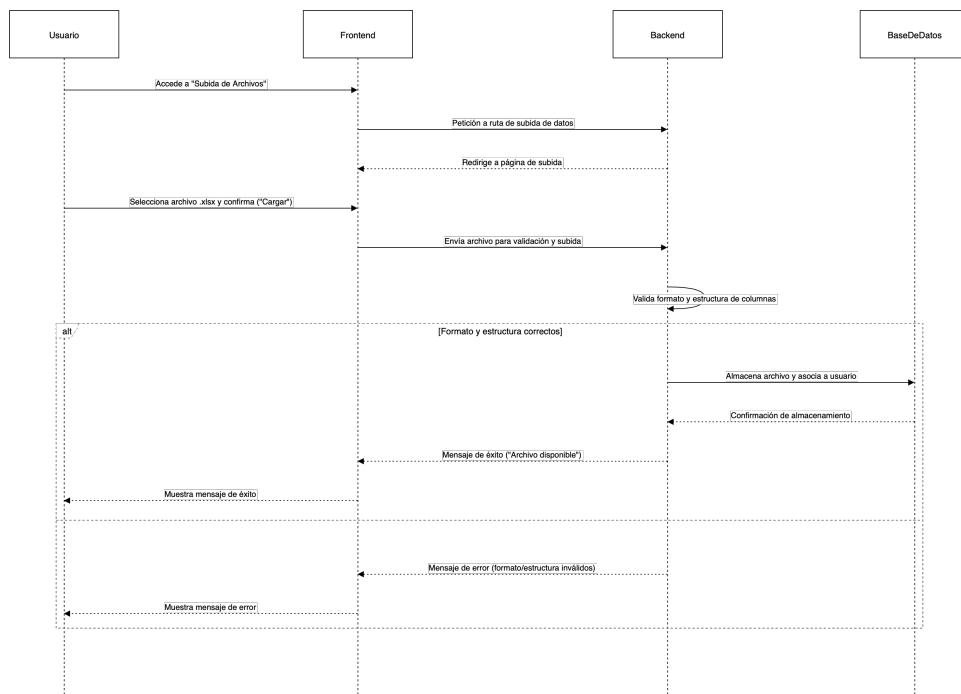


Figura C.19: Diagrama de secuencia de proceso de subida de un dataset

Descargar resultados

Una vez finaliza la predicción, nos encontramos con la posibilidad de descargar los resultados obtenidos, las diferentes llamadas que se realizan entre componentes se describe a continuación:

1. El usuario selecciona la opción de 'Descargar' o 'Exportar' dentro de la pantalla de resultados de predicción y elige el formato de salida deseado (por ejemplo, CSV, Excel o PDF).
2. El backend recupera la información de la estructura de datos PREDICTION_RESULTS **C.2** y genera el archivo en el formato seleccionado.
3. El sistema inicia la transferencia del archivo a través del navegador del usuario para su almacenamiento local.

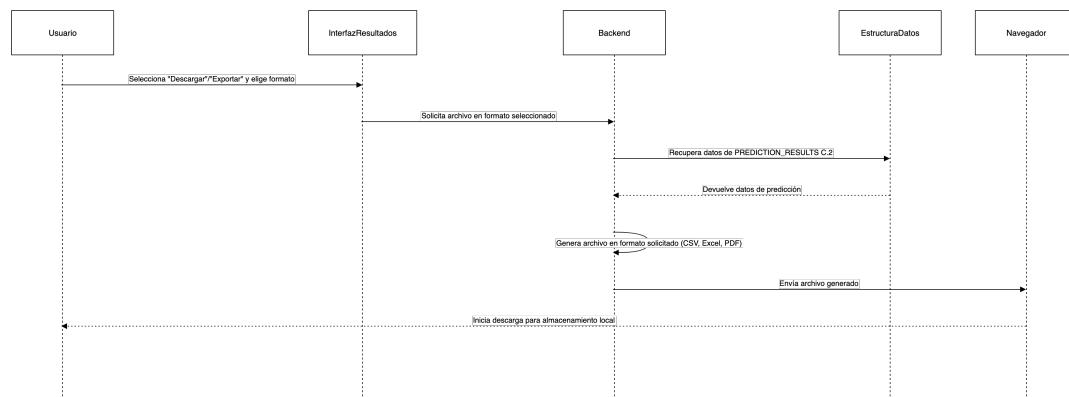


Figura C.20: Diagrama de secuencia de proceso de descarga de predicciones

Apéndice D

Documentación técnica de programación

D.1. Introducción

Esta documentación técnica tiene como objetivo servir de guía para entender, replicar y mantener el sistema desarrollado. A lo largo de estas páginas se explica cómo está organizado el proyecto, detallando la estructura de archivos, el código y la configuración necesaria para poner en marcha tanto el Jupyter Notebook como la plataforma web. Se detallan pautas para ajustar los parámetros de predicción, gestionar la base de datos y resolver posibles errores, proporcionando así el contexto necesario para que el software pueda crecer, integrarse en nuevos entornos o servir de base para investigaciones más profundas en el ámbito de la agricultura de precisión.

D.2. Estructura de directorios

Estructura sistema de predicción

A continuación se detalla la jerarquía de archivos y carpetas del sistema de minería de datos:

- **model/** – Directorio principal del sistema de predicción de riego.
 - **EDA.ipynb** – Análisis exploratorio de datos y estudio inicial de las series temporales.

- **data/** – Datos utilizados en el entrenamiento y evaluación del modelo.
 - ◊ **climate_data/** – Datos climáticos históricos relevantes para la predicción.
 - ◊ **soil_data/** – Información relacionada con las características del suelo.
 - * **mini1/** – Sensores ubicados en una zona específica de la finca.
 - * **mini2/** – Sensores ubicados en otra zona de la finca.
 - ◊ **datos_entrenamiento/** – Conjuntos de datos procesados para el entrenamiento del modelo que incluyen los datos sintéticos.
 - ◊ **models/** – Modelos entrenados serializados (.pkl)
 - ◊ **results/** – Resultados de predicciones.
 - ◊ **old_obsolete_data/** – Datasets obsoletos con menor longevidad.
- **loader.py** – Script encargado de la carga y preprocesado de datos.

Estructura aplicación web

A continuación se detalla la jerarquía de archivos y carpetas de la aplicación web:

- **my_flask_app/** – Directorio raíz de la aplicación web desarrollada con Flask.
 - **Dockerfile** – Definición de la imagen Docker para la aplicación.
 - **docker-compose.yml** – Orquestación de los servicios del sistema.
 - **requirements.txt** – Dependencias Python necesarias para la ejecución.
 - **run.py** – Punto de entrada de la aplicación Flask.
 - **uploads/** – Almacenamiento temporal de archivos subidos.
 - **app/** – Módulo principal que contiene la lógica de la aplicación.
 - ◊ **__init__.py** – Inicialización del módulo Flask.
 - ◊ **config.py** – Configuración global de la aplicación.

- ◊ **routes.py** – Definición de rutas y controladores HTTP.
- ◊ **state.py** – Gestión del estado interno.
- ◊ **ml_models.py** – Modelos de aprendizaje automático.
- ◊ **models.py** – Modelos de persistencia.
- ◊ **model_registry.py** – Registro de los modelos para poder ser deserializados por pickle.
- ◊ **train_models.py** – Entrenamiento de modelos.
- ◊ **auxiliary_prediction_functions.py** – Funciones auxiliares de la ruta de predicción.
- ◊ **progress_tracker.py** – Clase con las herramientas para realizar un seguimiento del progreso de entrenamiento/predicción
- ◊ **models/** – Modelos entrenados almacenados.
- ◊ **static/** – Recursos estáticos de la aplicación web.
 - * **js/** – Scripts JavaScript del cliente (*prediction_progress.js*, *train_progress.js*, *training.js*).
 - * **style.css** – Estilos CSS de la aplicación.
- ◊ **templates/** – Plantillas HTML renderizadas por Flask.
 - * **base.html** – Plantilla base.
 - * **index.html** – Página principal.
 - * **login.html** – Autenticación.
 - * **register.html** – Registro de usuarios.
 - * **upload.html** – Subida de datos.
 - * **prediction.html** – Interfaz de predicción.
 - * **prediction_progress.html** – Progreso de predicción.
 - * **training.html** – Entrenamiento de modelos.
 - * **train_progress.html** – Progreso de entrenamiento.
- **migrations/** – Control de versiones de la base de datos.
 - ◊ **alembic.ini** – Configuración de Alembic.
 - ◊ **versions/** – Historial de migraciones.
 - * **69bf73e7ba1d_initial_migration.py** – Migración inicial.

D.3. Manual del programador

Este manual está diseñado para facilitar la configuración, el despliegue y la comprensión técnica del proyecto. El código fuente completo está disponible

para su clonación desde el siguiente repositorio: [Enlace al Repositorio de GitHub](#).

Advertencia sobre el almacenamiento: El repositorio de github no incluye los modelos preentrenados necesarios para el funcionamiento inmediato. Tras el entrenamiento de estos, el repositorio local puede alcanzar más de **25GB** dependiendo de el número de modelos entrenados, la cantidad de datos, etc. Sin estos modelos, el peso total del proyecto se reduce drásticamente a menos de **100MB**.

La versión suministrada al tribunal en los USB dispone de una serie de modelos preentrenados de base.

Compilación, instalación y ejecución del proyecto

Esta guía parte de la base de que se dispone de una copia local del proyecto obtenida mediante el proceso de clonado mencionado anteriormente [Sección D.3](#).

Ejecución Jupyter Notebook

Ejecución en maquina local

Aunque existe una versión web desplegada del entorno Jupyter notebook([enlace al jupyter notebook desplegado](#)), la ejecución local es preferible si se dispone de hardware con alto poder de computación para acelerar las predicciones y re-entrenamientos. Puede encontrar una guía sobre uso de la versión web en la [Sección E.1](#).

Requisitos

Es indispensable contar con el kernel de Jupyter Notebook instalado. Para ello, existen dos vías principales:

- **Anaconda:** Instalación del entorno completo (incluye software adicional no estrictamente necesario).
- **Integración en IDE:** Configuración del kernel en Visual Studio Code u otros editores ligeros.

Se recomienda consultar guías oficiales para la [instalación del kernel en VSCode](#) o la [descarga de Anaconda](#). Una vez instalado, el entorno detectará automáticamente si existe una GPU disponible para optimizar los tiempos de ejecución.

Pasos de ejecución

1. Abra el proyecto en su editor (VSCode/Jupyter).
2. **Importante:** Asegúrese de haber importado la carpeta raíz o la carpeta `model` íntegramente, ya que el archivo `EDA.ipynb` depende del módulo `loader.py` para la carga de datos.
3. Navegue hasta la carpeta `model` y abra el archivo `EDA.ipynb`.
4. Seleccione la opción 'Ejecutar todo'.

Tenga en cuenta que el entrenamiento es un proceso intensivo; tiempos de 30 minutos o más son completamente normales. Si desea re-ejecutar partes específicas, puede utilizar los controles individuales por celda:

Ejecución en local sobre Docker

Si no desea instalar ninguna dependencia para correr el notebook, se proporciona también los archivos de docker preconfigurados para levantar un contenedor con el servicio en un puerto y ser accesible desde cualquier navegador con solo un par de comandos.

Requisitos

- Un navegador web actualizado (se recomienda Google Chrome o Safari).
- **Docker** instalado y en funcionamiento. Puede consultar la [guía oficial de instalación](#) según su sistema operativo.

Pasos de ejecución

Siga estos comandos en su terminal para levantar el entorno:

1. Acceda a la carpeta del proyecto jupyter:

```
cd model
```

2. Construya y levante los servicios mediante Docker Compose:

```
docker-compose up -d
```

Asegúrese de que el demonio de Docker esté corriendo antes de ejecutar este comando [25].

3. Una vez finalizado el despliegue, la aplicación será accesible en: <http://localhost:8888>

Ejecución de la aplicación web

Aunque existe una versión desplegada en producción ([enlace a la aplicación web](#)), la ejecución local es preferible si se dispone de hardware con alto poder de computación para acelerar las predicciones y re-entrenamientos. Puede encontrar una guía sobre uso de la versión web en la sección de ??.

Requisitos

- Un navegador web actualizado (se recomienda Google Chrome o Safari).
- **Docker** instalado y en funcionamiento. Puede consultar la [guía oficial de instalación](#) según su sistema operativo.

Pasos de ejecución

Siga estos comandos en su terminal para levantar el entorno:

1. Acceda a la carpeta de la aplicación Flask:

```
cd myflaskapp
```

2. Construya y levante los servicios mediante Docker Compose:

```
docker-compose up -d
```

Asegúrese de que el demonio de Docker esté corriendo antes de ejecutar este comando [25].

3. Una vez finalizado el despliegue, la aplicación será accesible en: <http://localhost:5000>

Reseteo de la Base de Datos

En caso de que necesite resetear las migraciones de la base de datos para limpiar el entorno, deberá eliminar los volúmenes de Docker asociados o ejecutar los comandos de Flask-Migrate dentro del contenedor, asegurando que la estructura de tablas se sincronice nuevamente desde cero.

Cómo debugear

El contenedor ha sido configurado específicamente tanto en el backend como en el archivo `docker-compose.yml` para permitir la depuración en tiempo real. Para acceder a los registros (logs) del sistema:

1. Con los servicios levantados, identifique el nombre del contenedor:

```
docker ps
```

2. Localice la columna '*NAME*' y copie el nombre del contenedor correspondiente al backend.

3. Visualice los logs en tiempo real con el siguiente comando:

```
docker logs -f <nombre_del_contenedor>
```

Aspectos relevantes del proyecto

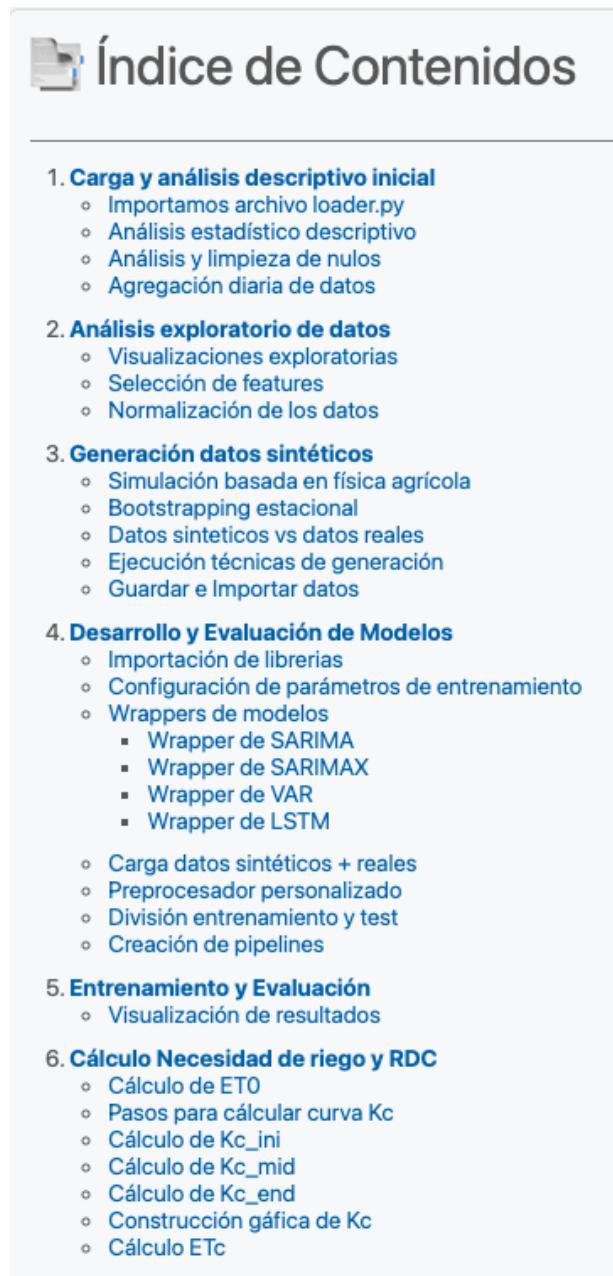
Secciones divididas código jupyter notebook

El jupyter notebook puede ser dividido en las siguientes secciones:

1. **Carga y análisis descriptivo:** Preparación inicial de los datos mediante el uso de scripts de carga, inspección de métricas estadísticas básicas, tratamiento de valores ausentes y remuestreo temporal a frecuencia diaria.
2. **Análisis exploratorio de datos (EDA):** Identificación de patrones mediante gráficos, selección de variables con mayor poder predictivo y escalado de los datos para asegurar la convergencia de los modelos.
3. **Generación de datos sintéticos:** Ampliación del dataset mediante simulaciones biofísicas y *bootstrapping* para mejorar la robustez del entrenamiento.

4. **Desarrollo y Evaluación de Modelos:** Configuración del entorno de trabajo, definición de hiperparámetros y encapsulamiento (*wrappers*) de algoritmos estadísticos (SARIMA, SARIMAX, VAR) y de aprendizaje profundo (LSTM).
5. **Entrenamiento y Evaluación:** Ejecución de los modelos sobre el conjunto de datos combinado y representación gráfica de las métricas de error para comparar el rendimiento.
6. **Cálculo de Necesidad de riego y RDC:** Aplicación de fórmulas agro-nómicas para determinar la evapotranspiración de referencia (ET_0) y la curva del coeficiente de cultivo (K_c) para optimizar el consumo de agua.

Se ha realizado un índice dentro del jupyter notebook con enlaces a las diferentes secciones para facilitar la navegación como se muestra a continuación.



1. Carga y análisis descriptivo inicial
◦ Importamos archivo loader.py
◦ Análisis estadístico descriptivo
◦ Análisis y limpieza de nulos
◦ Agregación diaria de datos
2. Análisis exploratorio de datos
◦ Visualizaciones exploratorias
◦ Selección de features
◦ Normalización de los datos
3. Generación datos sintéticos
◦ Simulación basada en física agrícola
◦ Bootstrapping estacional
◦ Datos sintéticos vs datos reales
◦ Ejecución técnicas de generación
◦ Guardar e Importar datos
4. Desarrollo y Evaluación de Modelos
◦ Importación de librerías
◦ Configuración de parámetros de entrenamiento
◦ Wrappers de modelos
▪ Wrapper de SARIMA
▪ Wrapper de SARIMAX
▪ Wrapper de VAR
▪ Wrapper de LSTM
◦ Carga datos sintéticos + reales
◦ Preprocesador personalizado
◦ División entrenamiento y test
◦ Creación de pipelines
5. Entrenamiento y Evaluación
◦ Visualización de resultados
6. Cálculo Necesidad de riego y RDC
◦ Cálculo de ETO
◦ Pasos para calcular curva Kc
◦ Cálculo de Kc_ini
◦ Cálculo de Kc_mid
◦ Cálculo de Kc_end
◦ Construcción gáfica de Kc
◦ Cálculo ETc

Figura D.1: Índice de contenidos EDA.ipynb

Configuración modelos en el jupyter y ajuste horizonte

En la siguiente Clase de la sección 'Creación pipelines modelos', encontramos las configuraciones tanto de ajuste del tamaño del conjunto de datos de entrenamiento como los parámetros con los que se ejecutan cada modelo.

```

# Celda 3: Configuración y parámetros
class Config:
    """Clase de configuración para el proyecto"""

    # Parámetros de modelos
    SARIMA_ORDER = (1, 1, 1)
    SARIMA_SEASONAL_ORDER = (1, 1, 1, 30)
    VAR_MAXLAGS = 15

    # Parámetros de predicción
    PREDICTION_HORIZON = 240 # 8 meses ≈ 240 días
    TEST_SIZE = 180 # 6 meses para test

    # Rutas para guardar modelos
    MODEL_PATH = base_path / "models"
    RESULTS_PATH = base_path / "results"

    @classmethod
    def setup_directories(cls):
        os.makedirs(cls.MODEL_PATH, exist_ok=True)
        os.makedirs(cls.RESULTS_PATH, exist_ok=True)

# Configurar directorios
Config.setup_directories()

```

Figura D.2: Clase de configuración de parámetros de entrenamiento

Se advierte que si no se tiene un conocimiento experto en dichos algoritmos y datos, no se muevan cambien los parámetros de serie.

Relación rutas aplicación con lógica backend

Este apartado establece la relación directa entre la interfaz visual descrita en el manual de usuario y la lógica de control implementada en el backend.

Gestión de Sesiones y Usuarios

El sistema utiliza la extensión Flask-Login para la gestión de estados. Las rutas encargadas de la autenticación son:

Pantalla de Usuario	Ruta (Endpoint)	Función en Backend
Pantalla Principal	/	<code>index()</code> : Renderiza la página de bienvenida.
Registro de Usuario	/register	<code>register()</code> : Gestiona el método GET (formulario) y POST (creación de usuario en <code>db.session</code>).
Inicio de Sesión	/login	<code>login()</code> : Valida credenciales contra la base de datos y establece la sesión.
Cerrar Sesión	/logout	<code>logout()</code> : Invalida el token de acceso mediante <code>logout_user()</code> .

Tabla D.1: Mapeo de rutas de autenticación.

Gestión de Datos (Datasets)

La subida de archivos está restringida mediante el decorador `@login_required`.

- **Ruta:** /upload (upload_file)
- **Lógica técnica:** El backend valida que la extensión sea `.csv` (aunque el manual de usuario mencione `.xlsx`, el backend actual aplica `ALLOWED_EXTENSIONS = {"csv"}`). Los archivos se almacenan en la carpeta `uploads` mediante `secure_filename`.
- **Persistencia:** Se crea un registro en la tabla `Dataset` vinculado al `current_user.id`.

Entrenamiento de Modelos

El proceso de entrenamiento es la funcionalidad más compleja y se divide en tres fases técnicas:

1. **Configuración (/entrenamiento):** Renderiza la interfaz de selección de parámetros y modelos (SARIMA, VAR, LSTM).

2. **Ejecución (/entrenamiento/proceso):** Recibe vía POST los hiperparámetros y el archivo seleccionado. Instancia la clase `Config` y ejecuta `train_and_save()`.
3. **Monitorización (/entrenamiento/progreso):** Renderiza la consola de logs. El cliente realiza peticiones asíncronas a `/api/progreso_entrenamiento` para obtener el estado del `Progress_tracker`.

Predicción y Cálculo RDC

Esta sección relaciona la generación de resultados con las librerías de análisis de datos:

Acción de Usuario	Ruta Backend	Procesamiento Interno
Acceso a Predicción	<code>/prediccion</code>	Renderiza <code>prediction.html</code> .
Iniciar Cálculo	<code>/prediccion/proceso</code>	Ejecuta <code>make_predictions</code> y <code>calculate_irrigation</code> .
Ver Resultados	<code>/prediccion/resultados</code>	Recupera datos de <code>PREDICTION_RESULTS</code> y genera los diccionarios para tablas.
Descarga de Datos	<code>/descargar_predicciones</code>	Genera un objeto <code>io.BytesIO</code> con formato Excel (.xlsx) usando <code>openpyxl</code> .

Tabla D.2: Flujo técnico del módulo de predicción.

APIs de Soporte

Para alimentar los elementos dinámicos de la interfaz (como los selectores de archivos o listas de modelos), se han implementado rutas de tipo API que retornan objetos JSON:

- `/api/archivos_datos`: Escanea el directorio de subidas y devuelve metadatos de los CSV disponibles.
- `/api/modelos_disponibles`: Lista los archivos `.pkl` presentes en la ruta definida por `MODELS_PATH`.

Pruebas del sistema

El proceso de 'testing' del proyecto se ha centrado en la evaluación del código de la aplicación web mediante herramientas estáticas en linea.

Análisis de código estático con SonarQube Cloud

El código fuente de la aplicación ha sido sometido a **herramientas de análisis de código estático** para garantizar las buenas prácticas, la mantenibilidad y la detección temprana de vulnerabilidades. Para este propósito, se ha implementado la plataforma **SonarQube Cloud**, integrada en el flujo de integración continua (CI) del proyecto.

La integración se ha realizado aprovechando las capacidades de *GitHub Actions*, permitiendo que el análisis se ejecute de forma automatizada ante cualquier cambio en el repositorio.

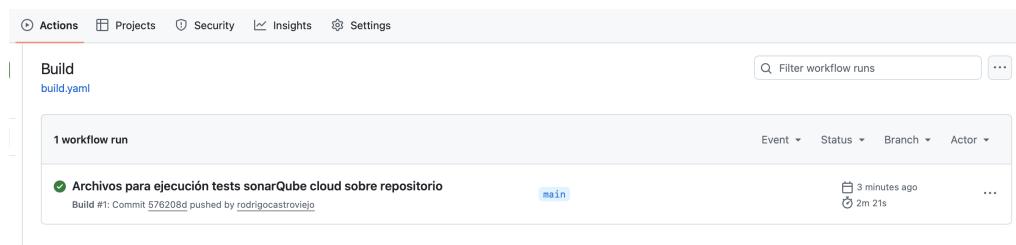


Figura D.3: Implementación de sonarQube a través de GitHub Actions

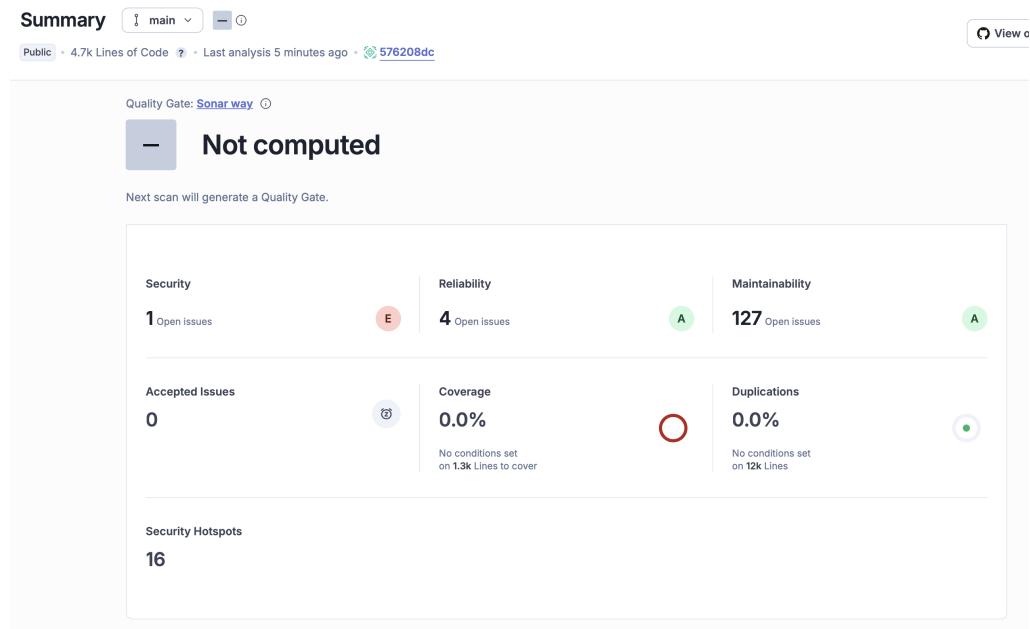


Figura D.4: Dashboard de sonarQube del repositorio del proyecto

Esta doble aproximación, combinando métricas cuantitativas de desempeño de los modelos con análisis de código, asegura que tanto la **calidad de los modelos** como la **robustez de la aplicación** cumplen con los estándares necesarios para su posible despliegue en plataformas agrícolas reales.

Apéndice E

Documentación de usuario

E.1. Introducción

Este manual detalla como usar las dos herramientas desarrolladas, sin embargo, se centrará en el funcionamiento de la aplicación web, al ser más accesible/fácil de usar para un usuario no técnico.

Pese a ser más complicada su instalación y comprensión, se han buscado alternativas de software más accesible para ejecutar el jupyter notebook evitando la instalación de software y facilitando el acceso sin comprometer la experiencia.

Por ello, se mostrará como ejecutar el jupyter notebook en la versión web desplegada de jupyter notebook con mi proyecto preconfigurado. Este no requiere de ninguna instalación.

E.2. Requisitos de usuarios

El software necesario para ejecutar ambas herramientas es el siguiente:

- Conexión a internet
- Navegador

E.3. Instalación

No requiere ningun tipo de instalación.

Como se ha comentado, se ha buscado mediante alternativas, buscar la forma más accesible de ofrecer los productos software.

Se pueden acceder a ambos servicios por las siguientes URLs:

- **Enlace a cuaderno de entrenamiento en Jupyter Notebook web**
 - **Token acceso:** tribunalUBU
- **Enlace a aplicación web**

Si se desea ejecutar localmente ambos servicios, se puede encontrar una guía en la [Sección D.3](#)

E.4. Manual del usuario

Jupyter notebook

Abrimos el [enlace a servicio Jupyter Notebook web](#) proporcionado, introducimos el token de acceso [E.3](#), seleccionamos el archivo `EDA.ipynb` y seleccionamos el botón de 'Ejecutar todas las celdas'.

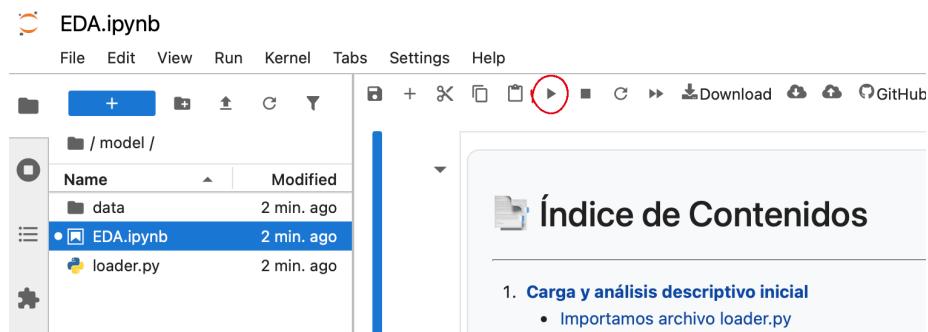


Figura E.1: Botón de ejecutar todas las celdas

Esperamos a que finalice. Es un proceso largo, no preocuparse si el entrenamiento llega a tomar 30 mins o más, son tiempos normales de ejecución.

En caso de que queramos reiniciar el entorno borrando el estado del cuaderno y reseteando las variables, seleccionaremos en el desplegable del botón 'Ejecutar todas', 'Reiniciar sesión y ejecutar todo'. Se recomienda esta opción ante un fallo/comportamiento extraño del cuaderno.

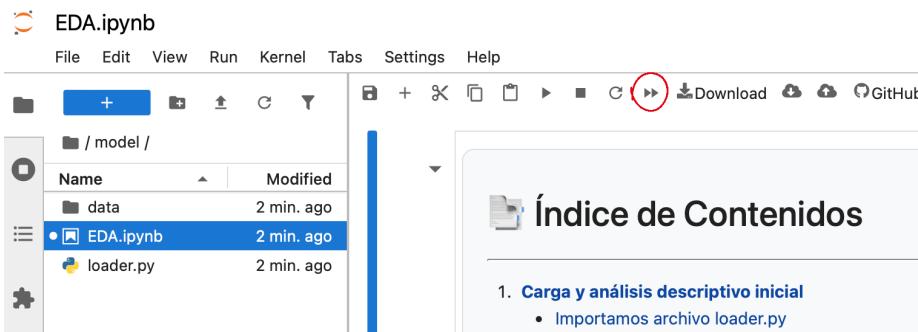


Figura E.2: Botón de reiniciar sesión y ejecutar todas las celdas

Si se desea volver a ejecutar únicamente una celda del notebook, podemos hacer **Shift + Enter** con la celda seleccionada.

En caso de que se quiera realizar una anotación, el usuario puede modificar una celda markdown o crear una nueva donde realizar el comentario en notación markdown. Pueden encontrar información acerca de como escribir en formato markdown en el [siguiente enlace](#).

Aplicación web

En este apartado se describen todas las acciones que el usuario puede realizar en la aplicación.

Registro

Para dar de alta un nuevo perfil en el sistema, proceda con las siguientes acciones:

1. **Navegación:** Acceda a la sección de registro(botón 'Registrarse').
2. **Cumplimentación de datos:** Proporcione la siguiente información en el formulario:
 - Identificador o nombre de usuario.
 - Dirección de correo electrónico.
 - Clave de acceso.
3. **Confirmación:** Presione el botón de envío('Registrarse') para procesar el alta.

Resultados del proceso:

- **Registro exitoso:** El sistema lo redirigirá al panel principal para iniciar su sesión.
- **Error en los datos:** Si existe información incompleta o no válida, se desplegará una notificación con las correcciones necesarias.

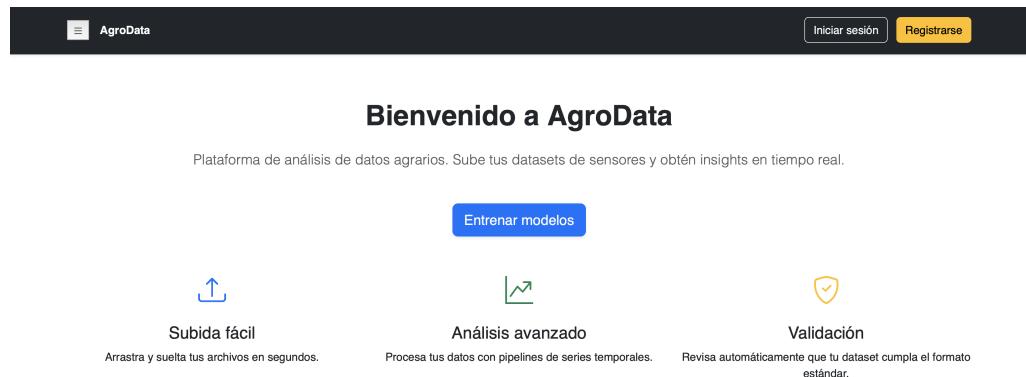


Figura E.3: Pantalla principal de la aplicación

Crear cuenta

Usuario

Email

Contraseña

Registrarse

Figura E.4: Pantalla registro

Iniciar sesión

Se necesita estar previamente registrado para poder iniciar sesión, en caso de que no lo este, siga la sección en la que se indica como registrarse en la aplicación [A.1](#).

Para acceder a su cuenta en la aplicación, siga estos pasos:

1. Clickar en el enlace de la web
2. Clickar sobre el botón de 'Inicio de sesión' en la parte superior derecha
3. Suministrar el nombre de usuario en el campo habilitado.
4. Introducir la contraseña en el espacio de seguridad.
5. Pulsar el botón de inicio de sesión (Entrar).

En caso de que las credenciales sean introducidas incorrectamente, saldrá una alerta notificando al usuario de este hecho. Reintroduce las credeciales de nuevo correctamente.

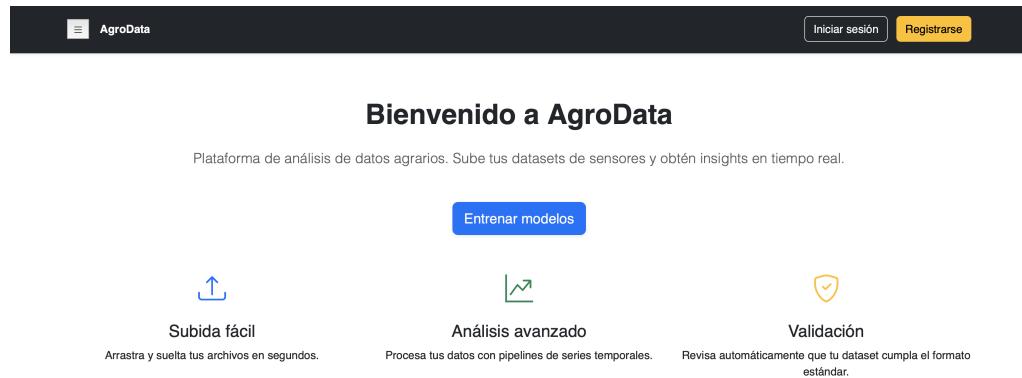


Figura E.5: Pantalla principal de la aplicación

Iniciar sesión

Usuario

Contraseña

Entrar

Figura E.6: Pantalla inicio de sesión

Cerrar la sesión de usuario

Condición de visibilidad: La opción de desconexión y el acceso al perfil solo estarán disponibles y visibles si el usuario ha completado previamente el proceso de autenticación E.4. En estado de navegación anónima, estos elementos permanecen ocultos.

Para salir de la plataforma de forma segura, realice los siguientes pasos:

1. **Localización del perfil:** Identifique el icono de usuario situado en el margen superior derecho de la interfaz, el cual está presente de forma persistente en todas las pantallas de la aplicación web.
2. **Interacción con el menú:** Haga clic sobre dicho icono para desplegar el menú de opciones personales.
3. **Cierre de sesión:** Seleccione la opción etiquetada como “*Cerrar sesión*”.

- **Efecto del sistema:** Una vez confirmada la acción, el sistema invalidará el token de acceso actual y redirigirá al usuario automáticamente a la pantalla de bienvenida o inicio de sesión.

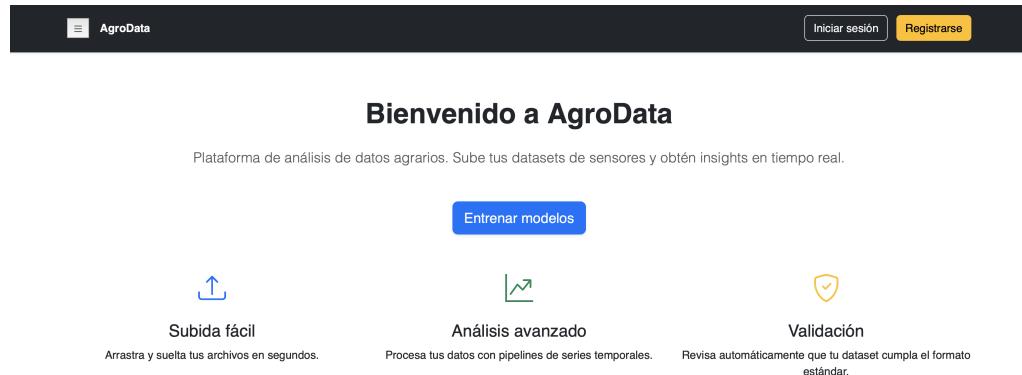


Figura E.7: Pantalla principal de la aplicación

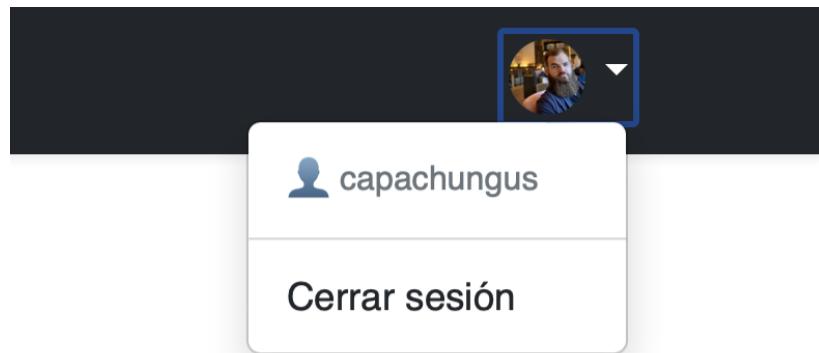


Figura E.8: Menú desplegable del perfil del usuario

Subir un dataset

Requisito previo: El acceso a esta funcionalidad está restringido exclusivamente a usuarios autenticados. Si intenta acceder sin haber iniciado

sesión, el sistema lo redirigirá automáticamente a la interfaz de acceso (véase la sección E.4 sobre cómo iniciar sesión).

Para realizar la carga de un conjunto de datos en la plataforma, siga este procedimiento:

1. **Despliegue del menú de navegación:** Accione el menú desplegable situado en el extremo superior izquierdo de la interfaz principal.
 2. **Selección de funcionalidad:** Dentro de las opciones disponibles, elija la entrada etiquetada como *Subir dataset*.
 3. **Localización del archivo:** Presione el botón de 'seleccionar archivos' para abrir el explorador de su sistema operativo.
 4. **Carga del documento:** Identifique y seleccione el archivo de interés desde su almacenamiento local (asegúrese de que posea la extensión **.xlsx**).
 5. **Ejecución del proceso:** Pulse el botón de 'Subir' para procesar la integración del dataset en la aplicación.
- **Nota técnica:** El sistema solo validará archivos con formato de hoja de cálculo compatible para garantizar la integridad de los datos procesados.

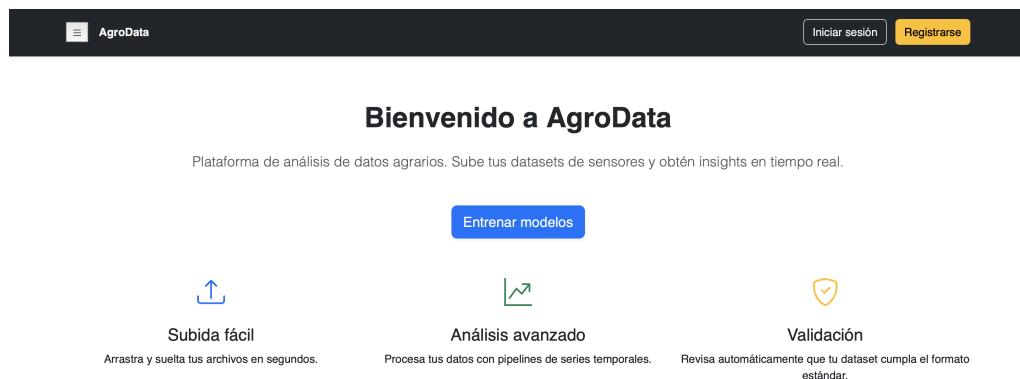


Figura E.9: Pantalla principal de la aplicación

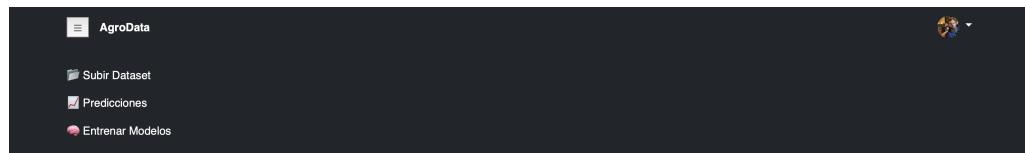


Figura E.10: Menú desplegable

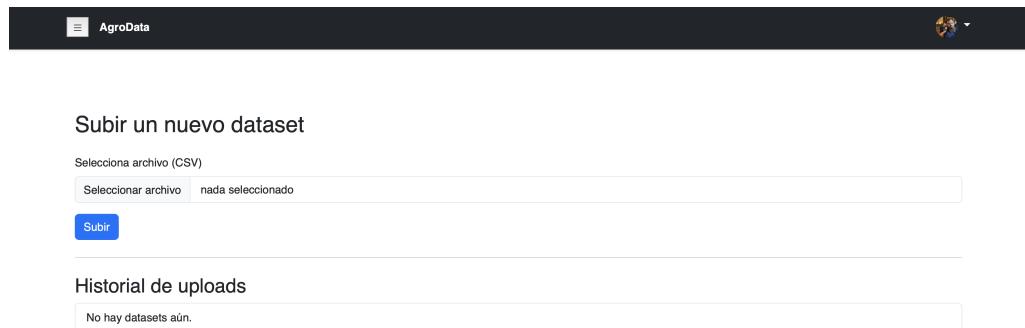


Figura E.11: Pantalla de subida de datasets

Configuración y Ejecución del Entrenamiento

Requisito de acceso: Esta funcionalidad requiere autenticación previa. Si el usuario intenta acceder de forma anónima, el sistema lo redirigirá a la interfaz de inicio de sesión (consulte la sección [E.4](#) para más detalles).

Acceso a la funcionalidad

Puede acceder a la sección de entrenamiento de dos formas:

- Desplegando el menú superior izquierdo y seleccionando *Entrenar modelos*.
- Accionando el botón central *Entrenar modelos* ubicado en el panel principal.

Configuración del proceso (3 pasos)

1. **Selección de Datos de Entrenamiento:** Elija uno de los datasets cargados previamente. El sistema marcará la elección con un indicador circular azul sobre la tarjeta del archivo.
 - **Ajuste del Conjunto de Prueba:** Defina el tamaño del set de test mediante el control deslizante o el cuadro numérico.
 - **Visualización Dinámica:** El gráfico circular de *Distribución* se actualizará en tiempo real para mostrar la proporción entre los datos de *Train* y *Test* según los días seleccionados.
2. **Selección de Modelos a Entrenar:** Identifique los algoritmos deseados (como SARIMA, VAR, SARIMAX o LSTM).
 - **Advertencia de Memoria:** La ejecución simultánea de múltiples modelos puede demandar hasta 20GB de RAM. Si la capacidad del sistema es insuficiente, el proceso de entrenamiento se interrumpirá por error.
3. **Parámetros Avanzados:** Permite ajustar las variables específicas de cada modelo seleccionado.
 - **Recomendación:** Dado que estos ajustes requieren conocimientos técnicos profundos, se aconseja mantener los valores por defecto a menos que se sea un usuario experto.

Ejecución y Confirmación

1. **Inicio:** Tras configurar los pasos anteriores, pulse *Iniciar entrenamiento*. Si desea revertir los cambios en los parámetros, puede usar la opción *Restablecer por defecto*.
2. **Validación (Pop-up):** Se desplegará una ventana emergente con el resumen de la configuración.
3. **Confirmación final:** Seleccione “*Sí, iniciar entrenamiento*” para comenzar el proceso o *Cancelar* para realizar ajustes adicionales.

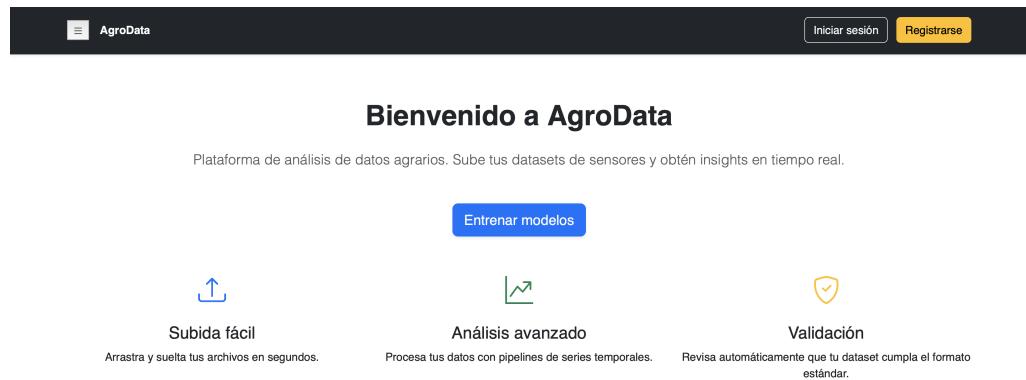


Figura E.12: Pantalla principal de la aplicación

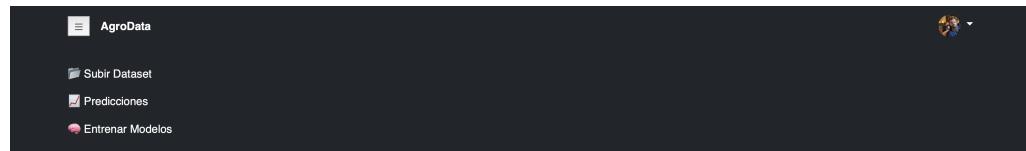


Figura E.13: Menú desplegable

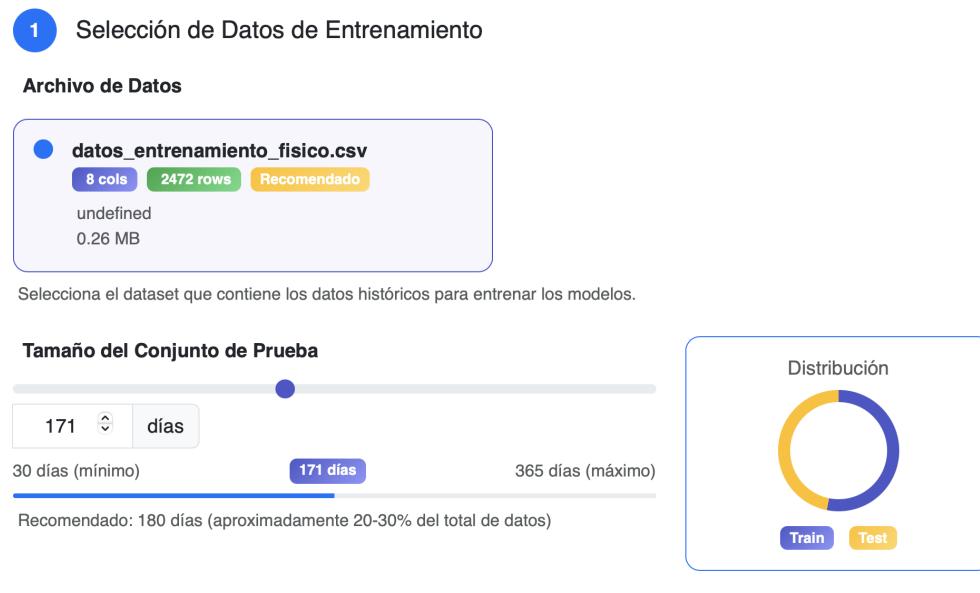


Figura E.14: Selección de datos de entrenamiento

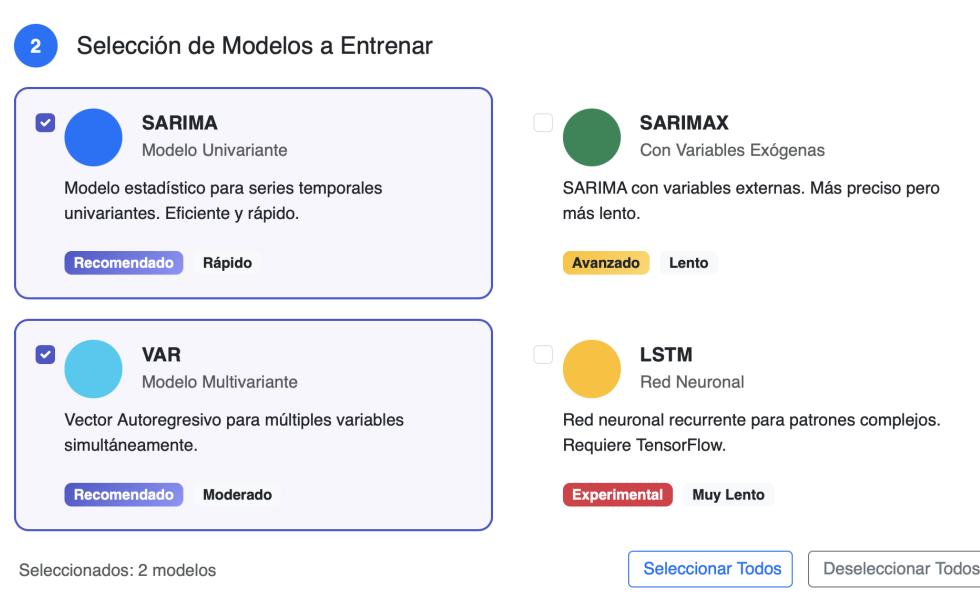


Figura E.15: Selección de modelos a entrenar

3 Parámetros Avanzados Mostrar/Ocultar

Parámetros SARIMA/SARIMAX ^

p (AR) <input type="text" value="1"/> ^	d (I) <input type="text" value="1"/> ^	q (MA) <input type="text" value="1"/> ^
Orden autorregresivo	Orden de diferenciación	Orden de media móvil
P (AR estacional) <input type="text" value="1"/> ^	D (I estacional) <input type="text" value="1"/> ^	Q (MA estacional) <input type="text" value="1"/> ^
s (Estacionalidad) <input type="text" value="30"/> ^	Período estacional en días	

Parámetros VAR ▽

Figura E.16: Selección de parámetros avanzados de entrenamiento

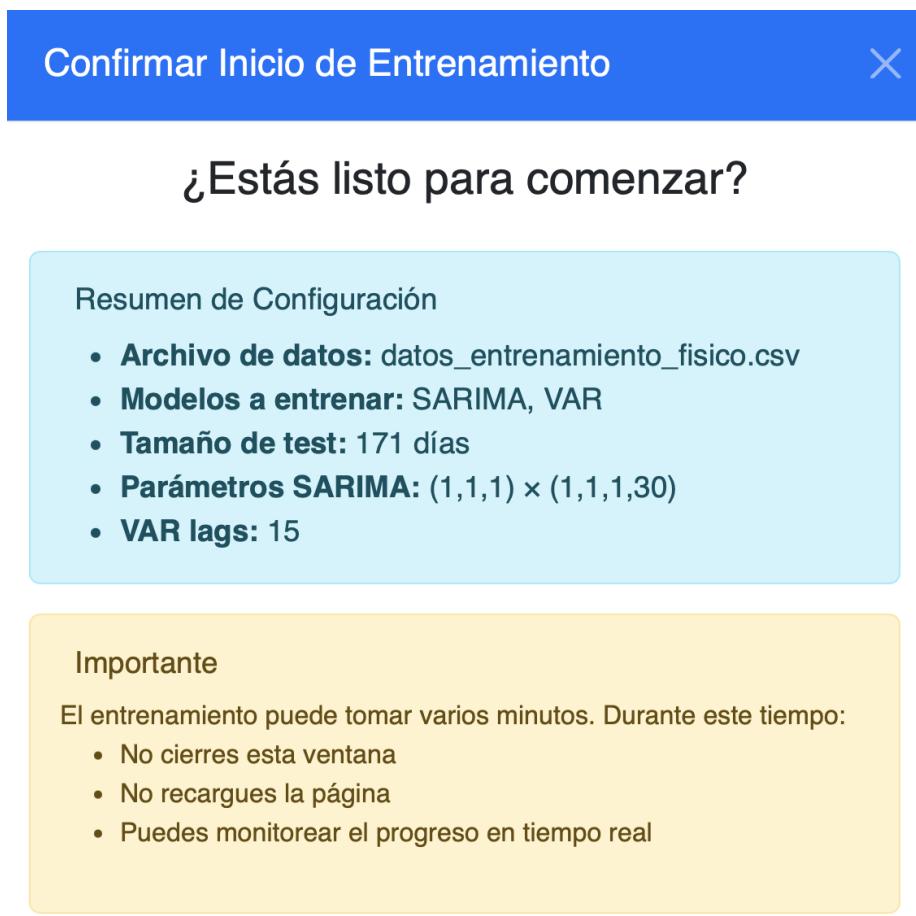


Figura E.17: Pop-up de confirmación de entrenamiento

Monitoreo del Progreso de Entrenamiento

Una vez confirmado el inicio del proceso, el sistema desplegará una interfaz dedicada al seguimiento en tiempo real de la ejecución. Esta pantalla permite supervisar la evolución del entrenamiento y el estado del backend a través de los siguientes elementos:

1. **Indicadores de Estado:** En la parte superior se visualizan métricas críticas que incluyen:
 - **Porcentaje de avance:** Representación numérica del progreso total del entrenamiento.
 - **Cronómetro de ejecución:** Tiempo transcurrido desde el inicio del proceso.
 - **Contador de pasos:** Relación de pasos completados frente al total programado.
2. **Consola de Salida:** Situada en la parte central, esta terminal muestra los registros (logs) detallados generados por el servidor.
 - **Actualización en tiempo real:** Los mensajes informativos se refrescan automáticamente cada 2 segundos para reflejar el estado actual del entrenamiento.
 - **Navegación:** El área de texto permite realizar *scroll* manual para revisar eventos anteriores en el historial de la sesión.
3. **Controles de la Consola:** En el margen superior derecho de la terminal se ubican dos utilidades de gestión:
 - **Limpiar:** Acción para vaciar el contenido actual de la consola de salida.
 - **Auto-scroll:** Botón conmutador que activa o desactiva el desplazamiento automático hacia el mensaje más reciente.

Nota: Se recomienda mantener activado el *Auto-scroll* para un seguimiento fluido, desactivándolo únicamente si se requiere inspeccionar minuciosamente una línea de registro específica sin interrupciones por nuevos mensajes.



Figura E.18: Consola entrenamiento y opciones sobre esta

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción y Justificación

El presente proyecto de fin de grado nace de la intersección entre la innovación tecnológica y la necesidad urgente de gestionar los recursos naturales de forma eficiente. En un contexto global marcado por la crisis climática y la creciente escasez hídrica, la agricultura —especialmente en cultivos intensivos como el del almendro en la finca de 'Canduela'— se enfrenta al reto crítico de mantener la productividad sin comprometer la integridad de los ecosistemas locales ni agotar las reservas de agua dulce.

La implementación de sistemas de *Internet of Things* (IoT) y el desarrollo de modelos predictivos basados en Python no solo responden a una necesidad técnica de optimización de procesos, sino que se alinean profundamente con la Agenda 2030 de las Naciones Unidas [13]. Este anexo detalla cómo el diseño y la ejecución de este trabajo contribuyen específicamente a los Objetivos de Desarrollo Sostenible (ODS) 2, 12 y 13 [14], integrando la sostenibilidad como un pilar fundamental de la ingeniería informática.

F.2. Contribución a los Objetivos de Desarrollo Sostenible (ODS)

ODS 2: Hambre cero

Aunque el ODS 2 suele asociarse a la ayuda alimentaria directa, una de sus metas fundamentales es asegurar la sostenibilidad de los sistemas de producción de alimentos y aplicar prácticas agrícolas resilientes que aumenten la productividad.

Mi proyecto impacta en este objetivo al transformar la agricultura tradicional en una agricultura de precisión. Al predecir con exactitud la cantidad de riego necesaria mediante políticas de Riego Deficitario Controlado (RDC), se garantiza la salud del cultivo incluso en condiciones de estrés hídrico. Esto permite que la producción de almendras sea estable y predecible en el tiempo, protegiendo la viabilidad económica de las explotaciones agrícolas. Al optimizar el uso del agua, se asegura la capacidad de producir alimentos de calidad sin degradar el suelo ni agotar los acuíferos, elementos críticos para la seguridad alimentaria a largo plazo y la nutrición de una población mundial en constante crecimiento.

ODS 12: Producción y consumo responsables

El núcleo de este trabajo es la eficiencia operativa. El ODS 12 insta a la gestión ecológicamente racional de los recursos naturales y a la reducción del desperdicio.

La herramienta desarrollada permite transitar de un modelo de 'riego por calendario o intuición' a un modelo de 'riego por necesidad real comprobada'. Al integrar datos de sensores IoT, el modelo evita el sobre-riego. La interfaz web construida facilita que el usuario final tome decisiones basadas en evidencia técnica, fomentando una cultura de consumo responsable donde cada litro de agua se utiliza de forma estratégica para maximizar la producción por unidad de recurso invertido, minimizando así la huella hídrica de la explotación.

ODS 13: Acción por el clima

La crisis climática se manifiesta de forma predominante a través de la alteración del ciclo hidrológico. Las sequías son cada vez más frecuentes, prolongadas y severas, especialmente en la península ibérica [23]. Por tanto,

la gestión del agua ya no es solo una cuestión de costes, sino de supervivencia del sector primario.

Este proyecto constituye una medida directa de adaptación al cambio climático. Al proporcionar tecnología que permite a los agricultores ser resilientes ante la irregularidad de las precipitaciones, se está mitigando el impacto socioeconómico de los fenómenos climáticos extremos. Además, existe una correlación directa entre el ahorro hídrico y el ahorro energético: menos agua bombeada se traduce en un menor consumo de electricidad o combustibles fósiles para los sistemas de riego, reduciendo la huella de carbono total de la finca.

F.3. Reflexión Personal y Compromiso Ético

Como futuro profesional del ámbito tecnológico, la elaboración de este proyecto me ha permitido comprender que la ingeniería no debe evaluarse exclusivamente por su precisión algorítmica o su eficiencia computacional, sino por su impacto socioambiental tangible. El desarrollo de este software me ha obligado a considerar variables biológicas y climáticas que a menudo quedan fuera del espectro puramente informático.

La sostenibilidad no ha sido un añadido secundario en este trabajo, sino el motor que ha guiado el diseño de la solución. La elección de herramientas de código abierto como Python y la creación de una plataforma web accesible buscan democratizar el acceso a la tecnología punta para el agricultor medio, ayudando a cerrar la brecha digital en el sector rural. He aprendido que la optimización de un recurso tan vital como el agua requiere una visión holística que combine la ciencia de datos con el respeto profundo por los ciclos de la naturaleza.

F.4. Conclusión

En conclusión, este Trabajo Fin de Grado demuestra que la digitalización del sector agrario es una herramienta indispensable para alcanzar las metas globales de sostenibilidad. El modelo predictivo y la plataforma de visualización no solo cumplen con el objetivo técnico de optimizar el riego en la finca 'Canduela', sino que proponen un camino hacia una agricultura inteligente, resiliente y, sobre todo, responsable.

Bibliografía

- [1] Agrozapiens. La revolución de la agricultura 5.0: Innovación y sostenibilidad para el futuro. <https://agrozapiens.com/la-revolucion-de-la-agricultura-5-0-innovacion-y-sostenibilidad-para-el-futuro/>, s.f. Visitado el 8 de febrero de 2026.
- [2] Richard G. Allen, Luis S. Pereira, Dirk Raes, and Martin Smith. *Crop Evapotranspiration: Guidelines for Computing Crop Water Requirements*. Number 56 in FAO Irrigation and Drainage Paper. Food and Agriculture Organization of the United Nations, Rome, 1998.
- [3] Apache Software Foundation. Apache license, version 2.0, 2004. Accedido el 5 de febrero de 2026.
- [4] Apple Inc. Macbook pro (15 pulgadas, 2016) — especificaciones técnicas, 2026. Consultado el 15 de enero de 2026.
- [5] Atlassian. Jira pricing – precios de jira software, standard, premium y enterprise, 2026. Consultado el 15 de enero de 2026.
- [6] Coca-Cola Andina. Epics (Épicas) - espacio de conocimiento jira y confluence, 2024. Accedido el 5 de febrero de 2026.
- [7] Docker, Inc. Docker pricing – planes y precios de suscripción, 2026. Consultado el 15 de enero de 2026.
- [8] GitHub, Inc. Github pricing – plans for github free, team, enterprise, 2026. Consultado el 15 de enero de 2026.
- [9] Grupo ASE. El precio de la electricidad subió un 4,2 % en 2025 pero apunta una tendencia a la baja en 2026, 2026. Publicado el 5 de enero de 2026, consultado el 15 de enero de 2026.

- [10] Impulsa Empresa. Calculadora de coste de un trabajador para la empresa en 2025, 2024. Consultado el 15 de enero de 2026.
- [11] Jobted. Sueldo del ingeniero informático en españa, 2026. Consultado el 15 de enero de 2026.
- [12] Kelio. Horas laborales en 2026: ¿cuántas horas hay que trabajar al año?, 2025. Publicado el 11 de noviembre de 2025, consultado el 15 de enero de 2026.
- [13] Naciones Unidas. La Agenda para el Desarrollo Sostenible. <https://www.un.org/sustainabledevelopment/es/development-agenda/>, s.f. Accedido el 9 de febrero de 2026.
- [14] Naciones Unidas. Objetivos de Desarrollo Sostenible. <https://www.un.org/sustainabledevelopment/es/sustainable-development-goals/>, s.f. Accedido el 9 de febrero de 2026.
- [15] Open Source Initiative. The bsd 3-clause license, 2026. Accedido el 5 de febrero de 2026.
- [16] Open Source Initiative. The mit license, 2026. Accedido el 5 de febrero de 2026.
- [17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python, 2011. Versión 1.3.2.
- [18] Python Software Foundation. History and license, 2026. Accedido el 5 de febrero de 2026.
- [19] Red Hat. ¿qué es una api rest?, 2022.
- [20] Pascual Romero. Análisis económico del cultivo de almendro en riego deficitario controlado (rdc). *imida. es*, 2005.
- [21] Scikit-learn developers. *sklearn.base.BaseEstimator — scikit-learn 1.8.0 documentation*, 2024. Accedido el 5 de febrero de 2026.
- [22] Scikit-learn developers. *sklearn.base.TransformerMixin — scikit-learn 1.8.0 documentation*, 2024. Accedido el 5 de febrero de 2026.
- [23] Sergio Martín Vicente Serrano. La evolución de los estudios sobre sequías climáticas en españa en las últimas décadas. *Geographicalia*, (73):7–34, 2021.

- [24] The pandas development team. *pandas.DataFrame.interpolate — pandas 2.2.2 documentation*, 2024. Accedido el 5 de febrero de 2026.
- [25] Windows Day. Cómo comprobar si el demonio de docker se está ejecutando y cómo iniciararlo, 2024. Accedido el 5 de febrero de 2026.