



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Desarrollo de un Bot de
Trading**



Presentado por Sergio Rebollo Ortega
en Universidad de Burgos — 10 de junio
de 2024

Tutor: Jose Manuel Galán ordax

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	iv
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	2
A.3. Estudio de viabilidad	32
Apéndice B Especificación de Requisitos	39
B.1. Introducción	39
B.2. Objetivos generales	40
B.3. Catálogo de requisitos	41
B.4. Especificación de requisitos	44
Apéndice C Especificación de diseño	71
C.1. Introducción	71
C.2. Diseño de datos	71
C.3. Diseño arquitectónico	73
Apéndice D Documentación técnica de programación	79
D.1. Introducción	79
D.2. Estructura de directorios	79
D.3. Manual del programador	80
Apéndice E Documentación de usuario	87

E.1. Introducción	87
E.2. Requisitos de usuarios	87
E.3. Instalación	88
E.4. Manual del usuario	88
Apéndice F Anexo de sostenibilización curricular	95
F.1. Introducción	95
Bibliografía	97

Índice de figuras

A.1. Burndown Sprint 1	4
A.2. Burndown Sprint 2	8
A.3. Burndown Sprint 3	10
A.4. Burndown Sprint 4	13
A.5. Burndown Sprint 5	16
A.6. Burndown sprint 6	19
A.7. Burndown sprint 7	22
A.8. Burndown Sprint 8	25
A.9. Burndown Sprint 9	28
A.10.Impuestos	33
B.1. Frecuencia casos de uso	44
B.2. DIAGRAMA DE CASOS DE USO	45
C.1. Modelo-Vista-Presentador	73
C.2. Diagrama de Paquetes	74
C.3. Diagrama de clases	75
D.1. Runbot.bat	82
D.2. Run _{bot} .sh	84
E.1. Interfaz Bot	88
E.2. Interfaz error	89
E.3. Interfaz error 2	90
E.4. Interfaz correcta	91
E.5. Resultados Bot	92

Índice de tablas

A.1. Detalles del Sprint 1	3
A.2. Tiempos Sprint 1	4
A.3. Etiquetas Sprint 1	5
A.4. Detalles del Sprint 2	6
A.5. Tiempos Sprint 2	7
A.6. Etiquetas Sprint 2	8
A.7. Detalles Sprint 3	9
A.8. Tiempos Sprint 3	10
A.9. Etiquetas Sprint 3	11
A.10.Detalles Sprint 4	12
A.11.Tiempos Sprint 4	13
A.12.Etiquetas Sprint 4	14
A.13.Detalles Sprint 5	15
A.14.Tiempos Sprint 5	16
A.15.Etiquetas Sprint 5	17
A.16.Detalles Sprint 6	18
A.17.Tiempos Sprint 6	19
A.18.Etiquetas Sprint 6	20
A.19.Detalles Sprint 7	21
A.20.Tiempos Sprint 7	22
A.21.Etiquetas Sprint 7	23
A.22.Detalles Sprint 8	24
A.23.Tiempos Sprint 8	25
A.24.Etiquetas Sprint 8	26
A.25.Detalles Sprint 9	27
A.26.Tiempos Sprint 9	28
A.27.Etiquetas Sprint 9	29
A.28.TOTAL	36

A.29.Licencias	37
B.1. Caso de uso: Importar Datos	48
B.2. Caso de uso 2: Almacenar Datos	49
B.3. Caso de uso 3: Transformar Velas Japonesas	50
B.4. Caso de uso 4: Definir características de las velas japonesas	51
B.5. Caso de uso 5: Convertir la información en un DataFrame	52
B.6. Caso de uso 6: Cálculo de indicadores técnicos	53
B.7. Caso de uso 7: Calcular RSI	54
B.8. Caso de uso 8: Calcular SMA	55
B.9. Caso de uso 9: Calcular Fibonacci	56
B.10.Caso de uso 10: Ejecución y gestión de órdenes de trading	57
B.11.Caso de uso 11: Ejecutar órdenes automatizadas	58
B.12.Caso de uso 12: Gestionar riesgo	59
B.13.Caso de uso 13: Registrar operaciones	60
B.14.Caso de uso 14: Interactuar con la interfaz de usuario	61
B.15.Caso de uso 15: Configurar bot	62
B.16.Caso de uso 16: Elección del activo a operar	63
B.17.Caso de uso 17: Elección del lotaje	63
B.18.Caso de uso 18: Control del Bot	64
B.19.Caso de uso 19: Manejo de múltiples hilos y procesos	64
B.20.Caso de uso 20: Ejecutar tareas en paralelo	65
B.21.Caso de uso 21: Sincronizar hilos	66
B.22.Caso de uso 22: Gestionar errores y documentación	67
B.23.Caso de uso 23: Manejar errores del sistema	68
B.24.Caso de uso 24: Registrar errores	69
B.25.Caso de uso 25: Documentar el sistema	70

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En esta primera sección explicaremos cómo ha sido la planificación inicial del proyecto del bot de trading y su evolución junto con el desarrollo del plan de viabilidad económica.

Como se mencionó anteriormente en la memoria del proyecto, la intención original era utilizar la herramienta Zenhub como herramienta en la primera reunión con mi tutor, esta herramienta nos iba a permitir tener un excelente control y poder visualizar el avance de las diversas tareas que hemos ido implementando a lo largo del proyecto. Sin embargo, debido a la expiración de mi licencia de Zenhub y que el proyecto está realizado en dos fases, es decir, el año pasado y el presente año he tenido que ir adaptando el control de las tareas sin la herramienta de Zenhub.

Cada tarea se ha asignado a un milestone, que agrupaba todas las tareas de un mismo sprint, para realizar así un seguimiento más completo. Se han creado nuevas etiquetas, además de las que nos ofrece Github por defecto, para poder así clasificar y proporcionar información sobre las diversas tareas.

En este proyecto hemos aplicado una metodología SCRUM, aunque como hemos comentado no hemos podido aplicarla en su totalidad debido a los diferentes problemas que nos han surgido. A pesar de todos los obstáculos se ha intentado mantener la metodología al máximo.

- Se han intentado programar sprints cada dos semanas manteniendo el tiempo para realizar las tareas asignadas a ese sprint, excepto el intervalo de tiempo de un año a otro en el que no seguí el plan.

- Se mantuvieron reuniones al principio en la primera fase del proyecto donde estaba arrancando, analizando errores y posibles mejoras. En la segunda fase no ha habido reuniones ya que estaba todo el proyecto en la parte final.
- Se han mantenido revisiones por cada sprints para ver si he realizado los objetivos adaptando los tiempos y las tareas para los próximos sprints corrigiendo posibles errores.

A.2. Planificación temporal

Esta sección explicara los diferentes sprints en los que hemos organizado el proyecto para poder llevarlo a cabo, se explicara la duración del sprints, las tareas que se han llevado a cabo y se realizara un gráfico para mostrar la evolución de este. Los gráficos de evolución de los sprints se han realizado en Excel, también se muestra una tabla en la que se recogen las diversas tareas que ocurren durante el sprint con su correspondiente etiqueta.

Sprint 1 - (1/3/23 - 14/3/23)

El primer sprint del proyecto consistirá en realizar la primera reunión con el profesor para iniciar los primeros pasos a la hora de realizar el proyecto, definiendo así objetivos de este, diversas pautas para tener en cuenta y una serie de consejos de cara al desarrollo del Bot de Trading.

En la reunión se puntualizó las primeras tareas que debíamos tener en cuenta para iniciar el proyecto, estas tareas son las siguientes:

- **Seleccionar gestor de tareas:** Tarea que implica la elección del gestor de referencias que se va a utilizar para organizar las referencias bibliográficas del proyecto.
- **Configurar entorno y componentes:** Instalar y preparar el ecosistema necesario para el desarrollo del proyecto, esto implica instalación y configuración de software, sistemas operativos, y otras herramientas necesarias para el desarrollo del proyecto.
- **Instalar herramientas de memoria:** Como vamos a organizar y almacenar los datos durante el proyecto.

- **Seleccionar herramienta para documentar:** Elección de la plataforma que vamos a utilizar para redactar nuestra memoria y anexos en este caso será mediante Látex.

Sprint 1	
Fecha inicio	1/3/23
Fecha Final	14/3/23
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	53,00 %
Horas de desarrollo disponibles	16
Horas diarias de desarrollo	1,6

Tabla A.1: Detalles del Sprint 1

En la anterior ilustración [A.1](#) mostramos el inicio y el fin de este primer sprint, los días totales que se trabajaron, exceptuando el fin de semana, en total fueron 10. Los desarrolladores a lo largo de todo el sprint serán uno, y la carga del equipo en este caso ha sido del 50 %. Las horas de desarrollo disponibles serán 16 horas, esto se calcula mediante la fórmula 'Días Laborales * N° de Desarrolladores * Horas Laborales diarias * Carga del equipo'. Después tenemos las horas diarias de desarrollo que se calculan mediante la división de las horas de desarrollo disponibles entre los días laborales.

Sprint 1		
Time	Esperado	Real
1/3/23	16	16
2/3/23	15	16
3/3/23	14	15
4/3/23	13	14
5/3/23	11	14
6/3/23	10	13
7/3/23	9	11
8/3/23	8	10
9/3/23	7	10
10/3/23	6	7
11/3/23	5	5
12/3/23	3	5
13/3/23	2	2
14/3/23	1	1

Tabla A.2: Tiempos Sprint 1

En esta tabla [A.2](#) exponemos como serían las horas esperadas a lo largo de las dos semanas en las que se realizó el Sprint.

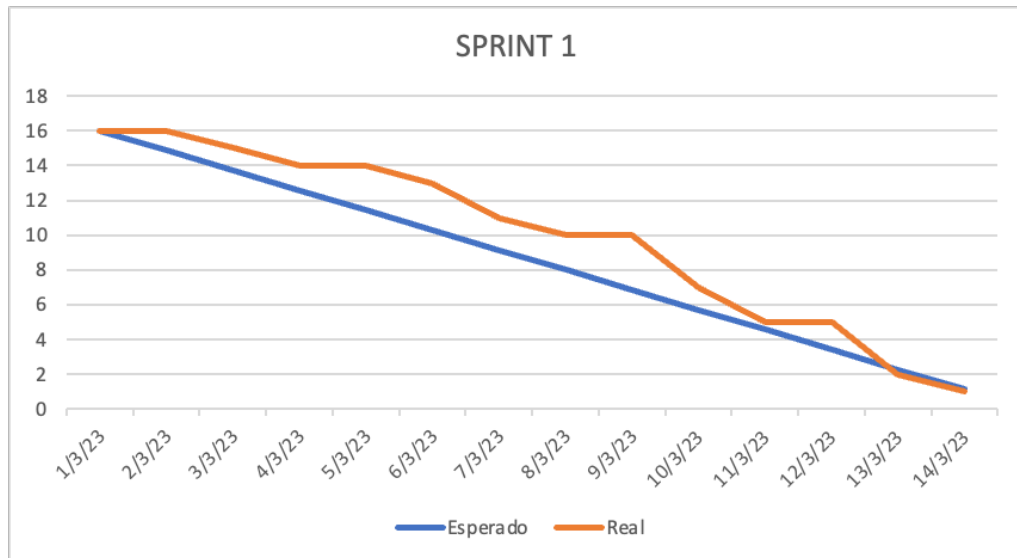


Figura A.1: Burndown Sprint 1

En la anterior figura A.1 se muestra el Burndown del Sprint 1, representa de forma gráfica el avance de las tareas a medida que va avanzando el tiempo pudiendo observar así una comparativa idónea.

La línea naranja indica como en verdad hemos ido avanzando en comparación con lo que se esperaba del avance del proyectos.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Seleccionar gestor de referencias							
Configurar entorno y componentes							
Instalar herramientas de memoria							
Seleccionar herramienta para documentar							
Primera reunión con el tutor							

Tabla A.3: Etiquetas Sprint 1

Por último en la tabla A.1, se muestra la tabla de tareas con sus correspondientes etiquetas.

Sprint 2 - (21/03/2023 – 03/04/2023)

En este segundo sprint no se empezó nada más acabar el primero por problemas externos, sino que se empezó seis días después. En este sprint se empieza a entrar en materia del contenido en el que se va a basar el proyecto, buscando información teórica y práctica.

Las tareas para realizar en este sprint son las siguientes:

- **Buscar estudios relacionados:** Búsqueda de estudios relacionados con el TFG, su situación actual y ejemplos de bots de trading en la actualidad.
- **Información de análisis técnico:** Se ha realizado una búsqueda exhaustiva de diversas fuentes de información sobre análisis técnico, creando un resumen preliminar de los datos recopilados. Durante este proceso, se han anotado meticulosamente las referencias en un bloc de notas, registrando las fuentes de donde se extrajo la información que se planea utilizar en el trabajo.
- **Estructurar memoria TFG:** Se ha realizado una estructura de cómo se va a organizar la memoria del TFG, con un índice preliminar.
- **Ver videos sobre bibliotecas de Python acerca del trading:** Obtener información de las bibliotecas y de cómo se va a orientar la estructura del bot de trading.

Sprint 2	
Fecha inicio	21/3/23
Fecha Final	3/4/23
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	60,00 %
Horas de desarrollo disponibles	18
Horas diarias de desarrollo	1,8

Tabla A.4: Detalles del Sprint 2

En esta tabla A.4 se muestran las fechas, la carga de trabajo y las horas de desarrollo que se debería de trabajar en el proyecto a lo largo del sprint 2.

Sprint 2		
Time	Esperado	Real
21/3/23	18	13
22/3/23	17	13
23/3/23	15	12
24/3/23	14	10
25/3/23	13	7
26/3/23	12	7
27/3/23	10	7
28/3/23	9	6
29/3/23	8	5
30/3/23	6	4
31/3/23	5	4
1/4/23	4	3
2/4/23	3	2
3/4/23	1	1

Tabla A.5: Tiempos Sprint 2

La tabla A.5 muestra la comparativa entre lo esperado y lo que se ha trabajado realmente a lo largo de este sprint.

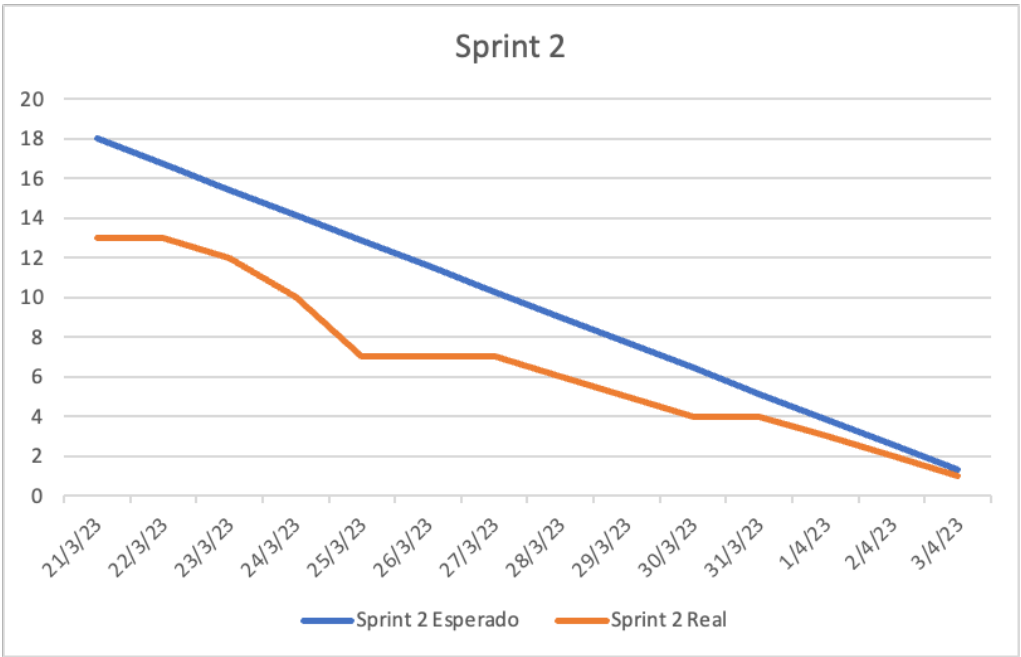


Figura A.2: Burndown Sprint 2

La figura A.2 muestra el Burndown del sprint número 2 en el que se observa gráficamente la evolución de los tiempos ilustrados en la Tabla A.6.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Buscar Estudios Relacionados							
Información de análisis técnico							
Estructurar memoria TFG							
Ver videos sobre bibliotecas de trading							

Tabla A.6: Etiquetas Sprint 2

La tabla A.6 muestra las tareas con sus correspondientes etiquetas.

Sprint 3 - (16/04/2023 – 23/04/2023)

Este sprint es más corto, con una duración de 1 semana, debido a la carga de trabajo de otras asignaturas. Además, comienza 10 días después de finalizar el segundo sprint.

En este sprint se obtiene información necesaria para poder empezar a programar en el siguiente sprint y se descarga un software que permite obtener los datos históricos del mercado.

Las tareas realizadas en este sprint son:

- **Descargar y obtener información de Metatrader 5:** Descargar Metatrader 5, una plataforma que permite obtener información histórica de los datos de los activos que se van a analizar.
- **Mejorar la memoria del TFG:** Añadir información al TFG de manera continua para reducir la carga de trabajo al final del proyecto. Repasar la información obtenida e ir añadiendo más si es preciso.

Sprint 3	
Fecha inicio	16/4/23
Fecha Final	23/4/23
Días Totales	5
Días Festivos	0
Días Laborales	5
Desarrolladores	1
Carga del equipo	60,00 %
Horas de desarrollo disponibles	9
Horas diarias de desarrollo	1,8

Tabla A.7: Detalles Sprint 3

En la tabla [A.7](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 3.

Sprint 3		
Time	Esperado	Real
16/4/23	9	6
17/4/23	8	6
18/4/23	8	5
19/4/23	7	5
20/4/23	6	4
21/4/23	6	3
22/4/23	5	2
23/4/23	5	1

Tabla A.8: Tiempos Sprint 3

La tabla A.8 muestra la comparativa entre el tiempo esperado y lo que se ha trabajado realmente en este sprint.

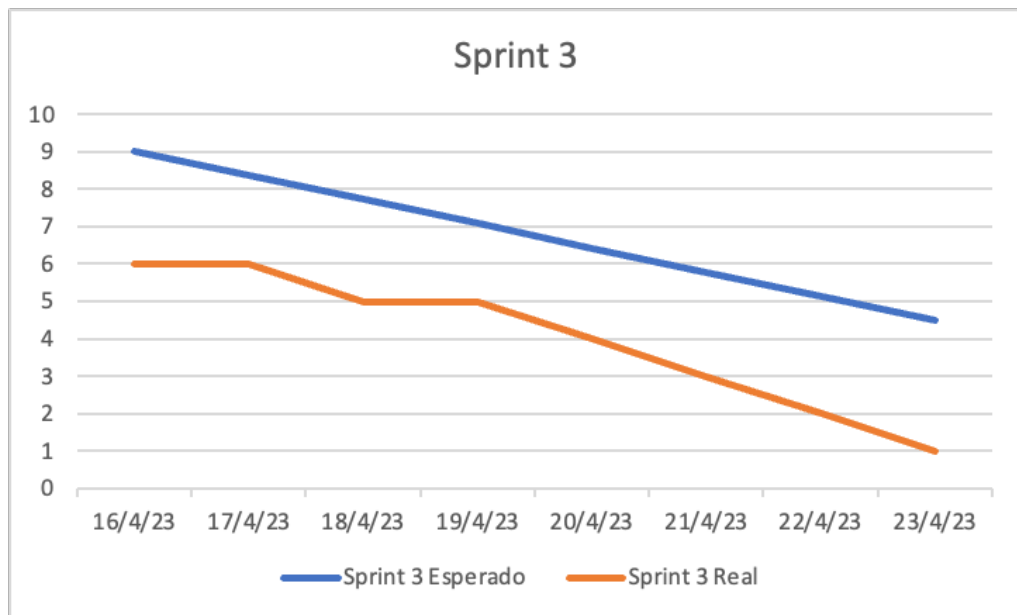


Figura A.3: Burndown Sprint 3

En la figura A.3 se observa el Burndown del sprint número 3 que muestra gráficamente la evolución de la tabla A.8.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Descargar y informarse de Metatrader 5							
Mejorar memoria TFG							

Tabla A.9: Etiquetas Sprint 3

La tabla A.7 muestra las tareas con sus correspondientes etiquetas.

Sprint 4 - (10/05/2023 – 23/05/2023)

En este sprint, al igual que en el anterior, no se pudo realizar de manera continua debido a problemas externos. Este sprint se enfoca en comenzar a programar y a estructurar el bot de trading.

Las tareas llevadas a cabo en este sprint son:

- **Programar Bot de trading en Python, estrategia chartista:**
Estructurar y empezar a programar el código del bot de trading. Se planificará cómo se va a estructurar y en qué dirección se va a trabajar para lograrlo.

Sprint 4	
Fecha inicio	10/5/23
Fecha Final	23/5/23
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	90,00 %
Horas de desarrollo disponibles	27
Horas diarias de desarrollo	2,7

Tabla A.10: Detalles Sprint 4

En la tabla A.10 se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 4.

La tabla A.11 muestra la comparativa entre el tiempo esperado y lo que se ha trabajado realmente en este sprint.

Sprint 4		
Time	Esperado	Real
10/5/23	27	27
11/5/23	25	26
12/5/23	23	23
13/5/23	21	21
14/5/23	19	18
15/5/23	17	14
16/5/23	15	10
17/5/23	14	10
18/5/23	12	9
19/5/23	10	6
20/5/23	8	2
21/5/23	6	1
22/5/23	4	1
23/5/23	2	0

Tabla A.11: Tiempos Sprint 4

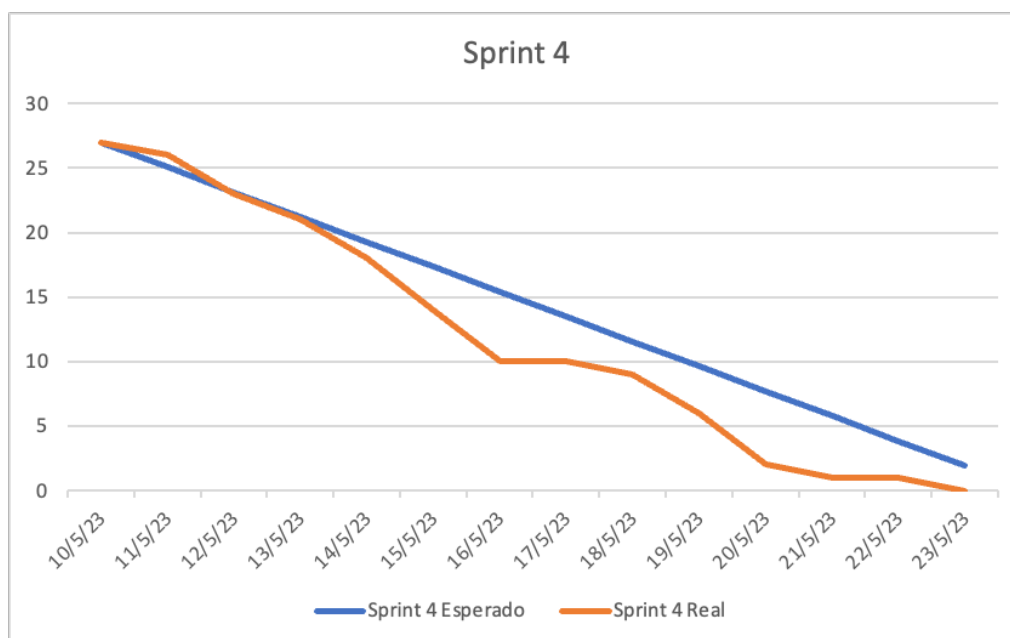


Figura A.4: Burndown Sprint 4

En la figura A.4 se observa el Burndown del sprint número 4 que muestra gráficamente la evolución de la tabla A.11.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Programar Bot de trading en Python							
Reunión con el tutor							

Tabla A.12: Etiquetas Sprint 4

La tabla A.12 muestra las tareas con sus correspondientes etiquetas.

Sprint 5 - (9/2/24 - 22/2/24)

Tras dejar el TFG en el año 2023 debido a no aprobar dos asignaturas, continuamos con el sprint número 5 retomando el hábito de trabajo. En este marco de trabajo empezamos a documentar el código y a realizar las clases que estructuramos anteriormente.

Las tareas a realizar en este sprint son:

- **Documentación del código del Bot de trading, de la estrategia chartista:** Se comienza a documentar parte del código.
- **Memoria del TFG:** Continuar mejorando la memoria del proyecto.
- **Clase VELAS_1 - Método constructor:** Programación de la clase destinada a almacenar la información de las velas (información de los mercados financieros).
- **Clase VELAS_1 - Método set - Método get:** Métodos de la clase VELAS_1 para obtener y establecer la información de la bolsa.
- **Método obtener datos:** Método para pasar la información de la bolsa a un dataframe de pandas para poder manejar estos datos después.

Sprint 5	
Fecha inicio	9/2/24
Fecha Final	22/2/24
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	65,00 %
Horas de desarrollo disponibles	20
Horas diarias de desarrollo	2,0

Tabla A.13: Detalles Sprint 5

En la tabla [A.13](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 5.

En la tabla [A.14](#) se muestra la comparativa entre lo esperado y lo que se ha trabajado realmente en este sprint.

Sprint 5		
Time	Esperado	Real
9/2/24	17	17
10/2/24	16	16
11/2/24	15	15
12/2/24	13	14
13/2/24	12	14
14/2/24	11	13
15/2/24	10	11
16/2/24	9	10
17/2/24	7	10
18/2/24	6	7
19/2/24	5	5
20/2/24	4	5
21/2/24	2	2
22/2/24	1	1

Tabla A.14: Tiempos Sprint 5

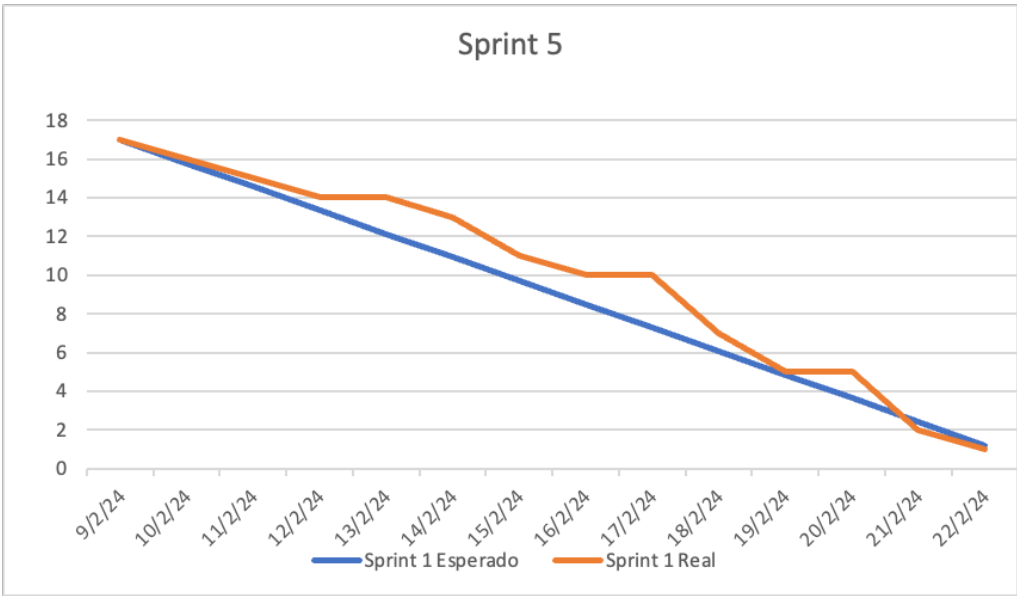


Figura A.5: Burndown Sprint 5

En la ilustración A.5 se muestra Burndown del sprint número 5 que muestra gráficamente la evolución de la tabla anterior.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Documentación del código del bot							
Memoria del TFG							
Clase VELAS1, Método constructor							
Clase VELAS1, Método set, Método get							
Método obtener datos							

Tabla A.15: Etiquetas Sprint 5

La tabla A.15 muestra las tareas con sus correspondientes etiquetas.

Sprint 6 - (24/2/24 - 9/3/24)

Sprint destinado únicamente a programar las clases que faltan del proyecto. Las tareas son las siguientes:

- **Clase VELAS - Método `update_candles`:** Método para cargar y actualizar las velas desde el archivo CSV sobre los datos de la bolsa.
- **Clase VELAS - Método `thread_candles`:** Función principal del hilo encargado de cargar y actualizar las velas.
- **Clase RSI - Método `thread_rsi` y Método `parameters_RSI`:** Métodos para cargar y actualizar los datos de la herramienta RSI según avanza el mercado.
- **Clase SMA - Método `thread_sma` y Método `parameters_sma`:** Clase con los métodos para cargar y actualizar la información de la bolsa para que podamos cargar la línea de tendencia de 200 periodos.
- **Clase Fibonacci:** Método que permite obtener los niveles importantes de Fibonacci.

Sprint 6	
Fecha inicio	24/2/24
Fecha Final	9/3/24
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	80,00 %
Horas de desarrollo disponibles	24
Horas diarias de desarrollo	2,4

Tabla A.16: Detalles Sprint 6

En la tabla [A.16](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 6.

Sprint 6		
Time	Esperado	Real
24/2/24	24	27
25/2/24	22	24
26/2/24	21	22
27/2/24	19	18
28/2/24	17	15
29/2/24	15	14
1/3/24	14	12
2/3/24	12	10
3/3/24	10	10
4/3/24	9	9
5/3/24	7	5
6/3/24	5	5
7/3/24	3	2
8/3/24	2	1

Tabla A.17: Tiempos Sprint 6

En la tabla A.17 se muestra la comparativa entre lo esperado y lo que se ha trabajado realmente en este sprint.

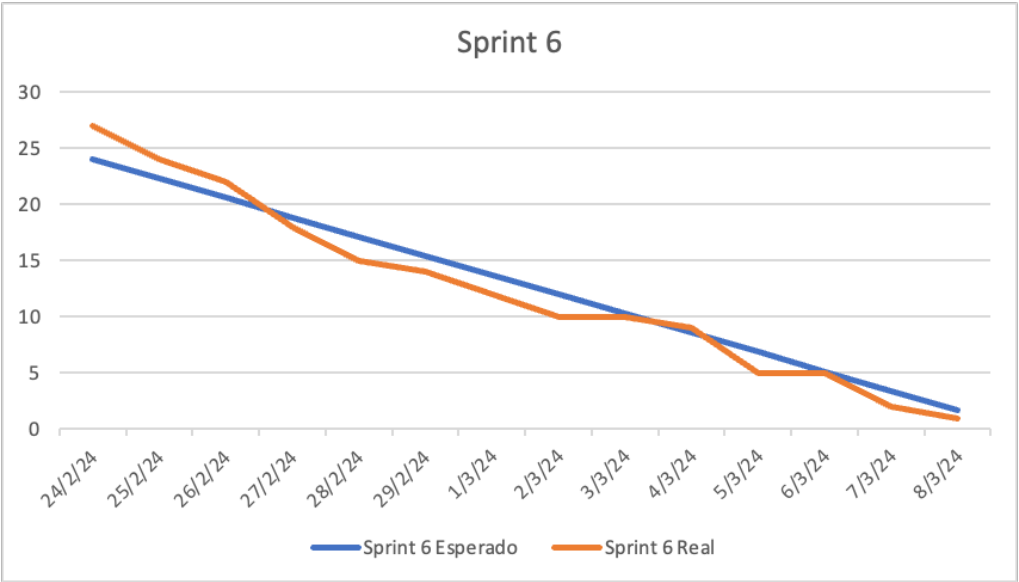


Figura A.6: Burndown sprint 6

En la ilustración A.6 se muestra Burndown del sprint número 5 que muestra gráficamente la evolución de la tabla anterior.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Clase VELAS, Método updatecandles							
Clase VELAS, Método threadcandles							
Clase RSI							
Clase SMA							
Clase Fibonacci							

Tabla A.18: Etiquetas Sprint 6

La tabla A.18 muestra las tareas con sus correspondientes etiquetas.

Sprint 7 - (19/3/24 - 1/4/24)

Este sprint, al igual que el sprint número 6, está dedicado únicamente a la programación de código. Las tareas realizadas son las siguientes:

- **Clase interfaz:** Clase para mostrar o introducir los datos del bot de forma más estética.
- **Clase main:** Clase que se encarga de controlar y iniciar el bot.
- **Clase botCSV:** Clase destinada a guardar en un CSV los datos que el bot obtiene tras realizar las operaciones.
- **Clase Bot:** Clase que maneja los hilos que se van a crear para controlar las herramientas utilizadas a la hora de realizar las operaciones.
- **Clase ORDER:** Descripción y programación de los métodos que comprenderán la clase ORDER.

Sprint 7	
Fecha inicio	19/3/24
Fecha Final	1/4/24
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	90,00 %
Horas de desarrollo disponibles	27
Horas diarias de desarrollo	2,7

Tabla A.19: Detalles Sprint 7

En la tabla [A.19](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 7.

En la tabla [A.20](#) se muestra la comparativa entre lo esperado y lo que se ha trabajado realmente en este sprint.

Sprint 7		
Time	Esperado	Real
19/3/24	27	38
20/3/24	25	35
21/3/24	23	30
22/3/24	21	27
23/3/24	19	26
24/3/24	17	22
25/3/24	15	20
26/3/24	14	14
27/3/24	12	10
28/3/24	10	10
29/3/24	8	9
30/3/24	6	7
31/3/24	4	4
1/4/24	2	1

Tabla A.20: Tiempos Sprint 7

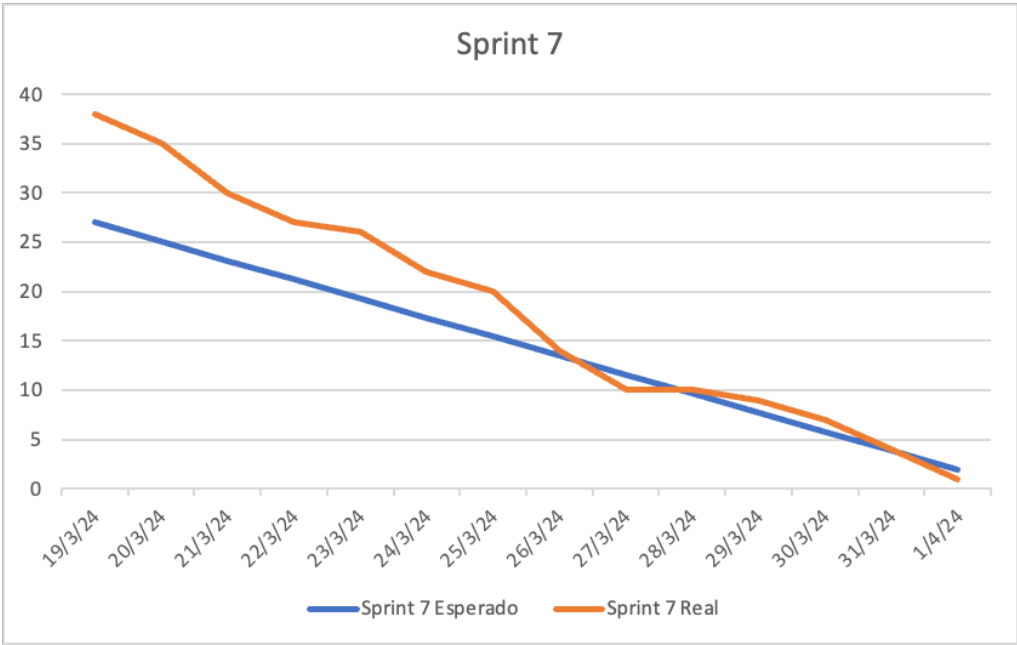


Figura A.7: Burndown sprint 7

En la ilustración A.7 se muestra Burndown del sprint número 5 que muestra gráficamente la evolución de la tabla anterior.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Clase interfaz							
Clase main							
Clase BotCSV							
Clase Bot							
Clase ORDER							

Tabla A.21: Etiquetas Sprint 7

La tabla A.21 muestra las tareas con sus correspondientes etiquetas.

Sprint 8 - (1/4/24 - 14/4/24))

En este penúltimo sprint, se realizarán tareas de programación y documentación, y se enviarán borradores al tutor para mostrarle el avance del proyecto.

Las tareas de este sprint son las siguientes:

- **Documentación del código:** Documentar las clases que tenemos y sus métodos, junto con sus variables, para que los usuarios puedan entender lo que hemos programado.
- **Tratamiento de errores – Clases:** En esta tarea se realiza el manejo de los errores de las clases.
- **Reunión con el tutor:** Mostrar avances al tutor.

Sprint 7	
Fecha inicio	1/4/24
Fecha Final	14/4/24
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	65,00 %
Horas de desarrollo disponibles	20
Horas diarias de desarrollo	2,0

Tabla A.22: Detalles Sprint 8

En la tabla [A.22](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 8.

Sprint 7		
Time	Esperado	Real
1/4/24	20	14
2/4/24	19	14
3/4/24	17	13
4/4/24	16	13
5/4/24	14	13
6/4/24	13	10
7/4/24	11	10
8/4/24	10	9
9/4/24	9	8
10/4/24	7	7
11/4/24	6	6
12/4/24	4	3
13/4/24	3	2
14/4/24	1	1

Tabla A.23: Tiempos Sprint 8

En la tabla A.23 se muestra la comparativa entre lo esperado y lo que se ha trabajado realmente en este sprint.

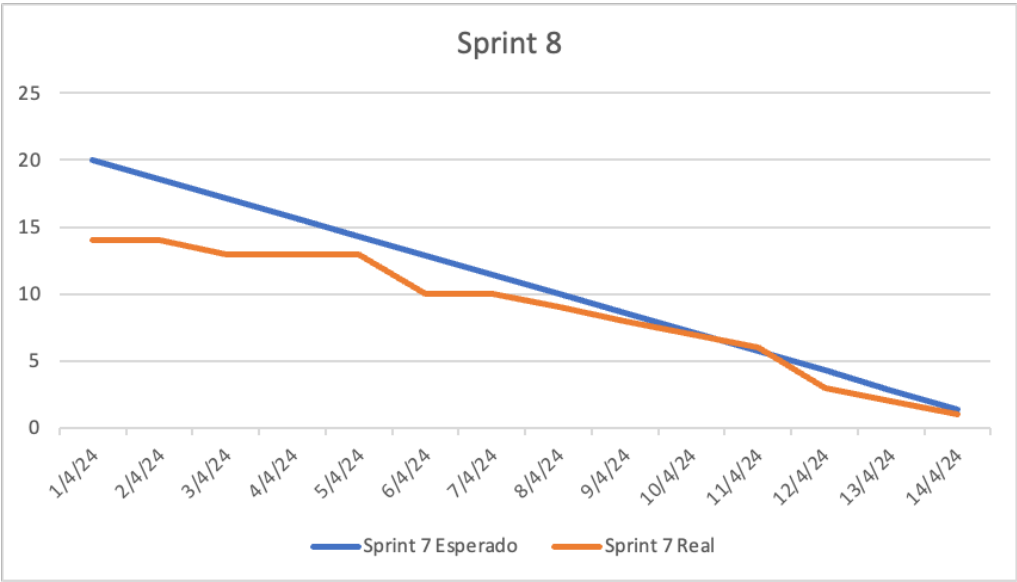


Figura A.8: Burndown Sprint 8

En la ilustración A.8 se muestra Burndown del sprint número 5 que muestra gráficamente la evolución de la tabla anterior.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Documentación del código							
Tratamiento de errores - CLASES							
Reunión con el tutor							

Tabla A.24: Etiquetas Sprint 8

La tabla A.24 muestra las tareas con sus correspondientes etiquetas.

Sprint 9 - (1/5/24 - 14/4/24))

El Sprint 9 está dedicado a la evaluación y mejora de la funcionalidad del bot de trading, así como a la preparación de la documentación final y la adaptación del bot para su ejecución en diferentes sistemas operativos. Las tareas realizadas en este sprint son las siguientes:

- **Análisis de resultados del bot de trading:** Esta tarea implica la revisión y evaluación del desempeño del bot de trading, analizando los resultados obtenidos de las operaciones recientes para identificar áreas de mejora y ajustar las estrategias según sea necesario.
- **Ficheros para la ejecución del Bot en diferentes sistemas:** Se desarrollan y adaptan los ficheros necesarios para que el bot de trading pueda ser ejecutado en diferentes sistemas operativos, asegurando su compatibilidad y funcionamiento óptimo en cada uno.
- **Realizar anexos y añadir los errores:** Esta tarea consiste en la elaboración de los anexos finales para la memoria del TFG, incluyendo una sección detallada que documente los errores encontrados durante el desarrollo del proyecto y cómo fueron solucionados, proporcionando un recurso valioso para futuras referencias y mejoras.

Sprint 9	
Fecha inicio	1/5/24
Fecha Final	14/5/24
Días Totales	10
Días Festivos	0
Días Laborales	10
Desarrolladores	1
Carga del equipo	70,00 %
Horas de desarrollo disponibles	21
Horas diarias de desarrollo	2,1

Tabla A.25: Detalles Sprint 9

En la tabla [A.25](#) se muestran las fechas, la carga de trabajo y las horas de desarrollo que se deberían trabajar en el proyecto a lo largo del sprint 9.

Sprint 9		
Time	Esperado	Real
1/5/24	21	19
2/5/24	20	16
3/5/24	18	12
4/5/24	17	12
5/5/24	15	10
6/5/24	14	10
7/5/24	12	10
8/5/24	11	8
9/5/24	9	8
10/5/24	8	7
11/5/24	6	6
12/5/24	5	6
13/5/24	3	3
14/5/24	2	1

Tabla A.26: Tiempos Sprint 9

En la tabla A.26 se muestra la comparativa entre lo esperado y lo que se ha trabajado realmente en este sprint.

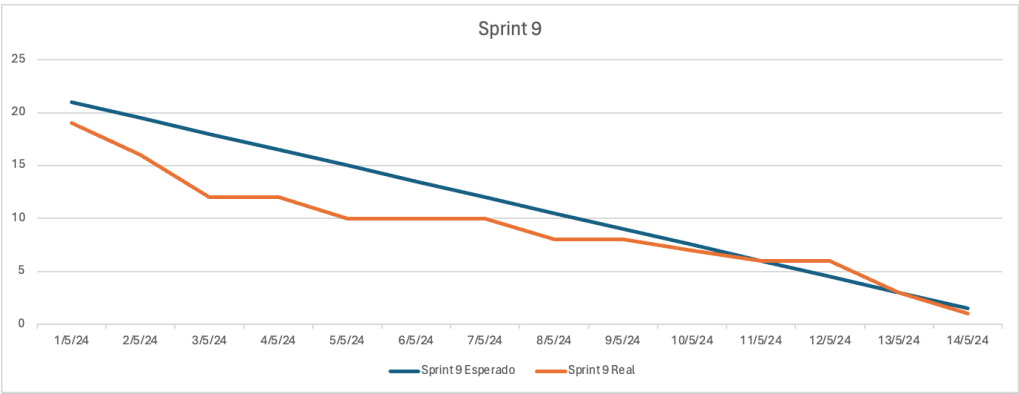


Figura A.9: Burndown Sprint 9

En la ilustración A.9 se muestra Burndown del sprint número 5 que muestra gráficamente la evolución de la tabla anterior.

tareas/etiquetas	Bug	Documentation	Duplicate	Enhancement	Info	Meeting	Programation
Análisis de resultados del bot de trading.							
Ficheros para la ejecución del Bot.							
Añadir log de errores							
Realizar anexos							

Tabla A.27: Etiquetas Sprint 9

La tabla [A.27](#) muestra las tareas con sus correspondientes etiquetas.

Issues

Tarea
1. Seleccionar gestor de referencias
2. Configurar entorno y componenets
3. Instalar herramientas de memoria
4. Seleccionar herramienta para documentar
5. Primera reunión con el tutor
6. Buscar Estudios Relacionados
7. Información de análisis técnico
8. Estructurar memoria TFG
9. Ver videos sobre bibliotecas de Python de trading
10. Descargar y obtener información de Metatrader 5 .
11. Mejorar memoria TFG
12. Programar Bot de trading en Python, estrategia chartista
13. Reunión con el tutor
14. Documentación del código del bot de trading.
15. Memoria del TFG
16. Clase VELAS_1 - Método constructor
17. Clase VELAS_1 - Método set - Método get
18. Método obtener datos
19. Clase VELAS - Método update_candles
20. Clase VELAS - Método thread_candles
21. Clase RSI - Método thread_rsi y Método parameters_RSI
22. Clase SMA - Método thread_sma y Método parameters_sma
23. Clase Fibonacci
24. Clase interfaz
25. Clase main
26. Clase BotCSV
27. Clase Bot
28. Clase ORDER
29. Documentación del código
30. Tratamiento de errores - CLASES
31. Reunión con el tutor
32. Análisis de resultados del bot de trading.
33. Ficheros para la ejecución del Bot.
34. Añadir log de errores
35. Realizar anexos

Tarea	Sprint	Inicio	HORAS	Fin
1	1	1/3/23	4	14/3/23
2	1	1/3/23	4	14/3/23
3	1	1/3/23	4	14/3/23
4	1	1/3/23	4	14/3/23
5	1	1/3/23	0,5	14/3/23
6	2	21/3/23	4	3/4/23
7	2	21/3/23	6	3/4/23
8	2	21/3/23	1	3/4/23
9	2	21/3/23	2	3/4/23
10	3	16/4/23	3	23/4/23
11	3	16/4/23	3	23/4/23
12	4	10/5/23	27	23/5/23
13	4	10/5/23	0,5	23/5/23
14	5	9/2/24	4	22/2/24
15	5	9/2/24	6	22/2/24
16	5	9/2/24	2,5	22/2/24
17	5	9/2/24	3	22/2/24
18	5	9/2/24	1,5	22/2/24
19	6	24/2/24	3	9/3/24
20	6	24/2/24	2,5	9/3/24
21	6	24/2/24	3	9/3/24
22	6	24/2/24	3	9/3/24
23	6	24/2/24	15,5	9/3/24
24	7	19/3/24	6	1/4/24
25	7	19/3/24	2	1/4/24
26	7	19/3/24	4	1/4/24
27	7	19/3/24	12	1/4/24
28	7	19/3/24	14	1/4/24
29	8	1/4/24	5	14/4/24
30	8	1/4/24	8	14/4/24
31	8	1/4/24	1	14/4/24
32	9	1/5/24	4	14/5/24
33	9	1/5/24	3	14/5/24
34	9	1/5/24	4	14/5/24
35	9	1/5/24	8	14/5/24

A.3. Estudio de viabilidad

En esta sección se va a medir el cálculo medio de los costes y beneficios que podría generar el proyecto en un hipotético caso de realizarse en un caso empresarial.

En el segundo apartado se expondra las librerías y herramientas que se han utilizado con sus correspondientes licenciadas para realizar el estudio de la viabilidad legal del proyecto.

Viabilidad económica

Los costes de las empresas se van a dividir en apartados, estos apartados son los siguientes:

- Empleados
- Hardware
- Software

Empleados

Primero se calcula el coste de los empleados, que van a participar el proyecto, en concreto el único empleado es el alumno que ha desarrollado el proyecto y los tutores que han controlado este proceso.

En el sistema universitario español el crédito vale 25 horas de trabajo, el trabajo de fin de grado tiene 12 créditos en la Universidad de Burgos, por lo que serán 300 horas de trabajo en 7 meses de marzo de 2023 a mayo de 2023 y de febrero de 2024 a mayo de 2024. Supone un total de 10,71 horas de trabajo semanal.

Una estimación del salario bruto de un informático junior con jornada completa es de 18000 € en 14 pagas, más o menos un sueldo de 8 € la hora. Realizando el cálculo obtenemos que el alumno cobra:

$$11\text{horas/semana} \times 8 \text{ €/hora} \times 4\text{semanas/mes} = 352 \text{ euros al mes.}$$

Pero la empresa debe de pagar impuestos y se tiene que añadir. Esto lo podemos consultar en el siguiente [enlace](#)

La ilustración de la siguiente página muestra los tipos de cotización de una empresa por horas, etc.

TIPOS DE COTIZACIÓN (%)				
CONTINGENCIAS	EMPRESA	TRABAJADORES	TOTAL	Accidentes de Trabajo y Enfermedades Profesionales
Comunes	23,60	4,70	28,30	Tarifa Primas establecida en la disposición adicional cuarta Ley 42/2006, de 28 de diciembre, PGE 2007, en la redacción dada por la Disposición Final Quinta del RDL 28/2018 de 28 de diciembre (BOE del 29) siendo las primas resultantes a cargo exclusivo de la empresa
Horas Extraordinarias Fuerza Mayor	12,00	2,00	14,00	
Resto Horas Extraordinarias	23,60	4,70	28,30	
Mecanismo Equidad Intergeneracional (MEI)	0,58	0,12	0,7	

(1) Exoneración en la cotización por contingencias comunes (salvo IT), desempleo, fondo garantía salarial y formación profesional, prevista en el art. 152 del RD Legislativo 8/2015:

- Aplicable durante el año 2024 a trabajadores cuenta ajena o a socios trabajadores o de trabajo de cooperativas, que continúen trabajando tras haber alcanzado la edad de 65 años si se acreditan 38 o más años de cotización o 66 años y 6 meses cuando se acrediten menos de 38 años de cotización.
- Tipo de cotización aplicable durante el año 2024 por IT por contingencias comunes: 1,55 %, del que el 1,30% será a cargo de la empresa, y el 0,25 % a cargo del trabajador.

(2) Los contratos de duración determinada por tiempo inferior a 30 días tendrán una cotización adicional a cargo del empresario que se abonará a su finalización, y que durante el año 2024 tendrá un importe de 31,22 €. Esta cotización adicional no se aplicará a los contratos por sustitución, a los contratos para la formación y el aprendizaje ni a los contratos de formación en alternancia.

DESEMPLEO	EMPRESA	TRABAJADORES	TOTAL
Tipo General: Contratación indefinida, incluidos los contratos indefinidos a tiempo parcial y fijos discontinuos, contratación de duración determinada en las modalidades de contratos de formación en alternancia, para la formación y aprendizaje, formativo para la obtención de la práctica profesional adecuada al nivel de estudios, de relevo, interinidad y contratos realizados con trabajadores que tengan reconocido un grado de discapacidad no inferior al 33%	5,50	1,55	7,05
Contrato duración determinada a tiempo completo o a tiempo parcial	6,70	1,60	8,30

	EMPRESA	TRABAJADORES	TOTAL
FOGASA	0,20		0,20

	EMPRESA	TRABAJADORES	TOTAL
FORMACIÓN PROFESIONAL	0,60	0,10	0,70

Figura A.10: Impuestos

En el caso de nuestro proyecto, estos impuestos aplicarían, como se observa en la figura A.10:

1. - 23,6 % contingencias comunes
2. - 5,5 % de desempleo
3. - 0,2 % FOGASA
4. - 0,6 % en formación profesional

En total el coste del alumno mensual:

$$\blacksquare 352\text{€/mes} / 1 - (0,236 + 0,055 + 0,002 + 0,006) = 502 \text{ €/mes}$$

A parte del alumno encargado de realizar el proyecto tenemos dos profesores con amplios conocimientos, debido a este factor el precio de la contratación debe ser mayor, los profesores cobrarán 30 €/hora. Más o menos los tutores guiarán al alumno durante los sprints, el trabajo de los profesores comprenderá al menos 1 hora a la semana.

Realizamos el cálculo.

$$\blacksquare 1 \text{ horas/semana} \times 30 \text{ €/hora} \times 4 \text{ semanas/mes} = 120 \text{ euros al mes.}$$

Este coste se multiplica por dos al haber dos tutores para el proyecto, por lo que serían 240 euros brutos al mes más los impuestos:

$$\blacksquare 240 \text{ €/mes} / 1 - (0,236 + 0,055 + 0,002 + 0,006) = 342 \text{ €/mes}$$

La empresa encargada de realizar el proyecto paga al mes un coste de 844 euros. El proyecto se ha realizado durante 7 meses, pero con intervalos por lo que aproximadamente serían 6 meses completos, los cálculos se realizan teniendo en cuenta 14 pagas, por lo tanto, el coste total de la empresa será de 6752 euros.

Hardware

En la realización del proyecto ha hecho falta un equipo de desarrollo de código, este equipo ha costado alrededor de 1400 euros, que se van a amortizar en 4 años:

Los costes de hardware:

- $1400 \text{ €} / 48 \text{ meses} \times 6 \text{ meses} = 175 \text{ €}$

Software

sistema operativo de los equipos. El sistema operativo es Windows 10 Home, el cual cuesta 145 euros. Para la documentación del proyecto se ha utilizado el paquete Office, el cual cuesta 8,80 euros al mes.

Dado que la amortización es de 4 años:

El coste del sistema operativo Windows 10 Home:

- $145 \text{ €} / 48 \text{ meses} \times 6 \text{ meses} = 18,125 \text{ €}$

El coste del paquete Office para la documentación, considerando que el proyecto dura 6 meses y cuesta 8,80 euros al mes:

- $8,80 \text{ €} \times 6 \text{ meses} = 52,8 \text{ €}$

- $52,8 \text{ €} / 48 \text{ meses} \times 6 \text{ meses} = 6,6 \text{ €}$

TOTAL, costes de software:

- $6,6 \text{ €} + 18,125 \text{ €} = 24,725 \text{ €}$

TOTAL

Al total de los costes se añadirán unos costes adicionales, como por ejemplo internet y luz. Estos costes se establecerán en un 10 % del total del proyecto.

Tipos	Costes
Empleados	6.752,00 €
Software	24,10 €
Hardware	175,00 €
Costes adicionales	69,51 €
TOTAL	6.951,10 €

Tabla A.28: TOTAL

Beneficios

Al ser un proyecto meramente educacional, no se espera obtener beneficios por su comercialización.

La única forma de obtener beneficios sería a través de donaciones que posibles clientes o usuarios puedan realizar para futuras mejoras.

Viabilidad legal

En el estudio de viabilidad legal se mostraran las licencias de todas las librerías y herramintetas que se han utilizado a lo largo de la realización del Bot de trading. En este apartado se ha establecido que tipo de licencia tendrá el proyecto.

En el estudio de viabilidad legal se mostrarán las licencias de todas las librerías y herramientas que se han utilizado a lo largo de la realización del bot de trading. En este apartado se ha establecido qué tipo de licencia tendrá el proyecto.

Se han empleado las siguientes librerías:

Módulo	Versión	Licencia
appdirs	1.4.4	MIT
beautifulsoup4	4.12.2	MIT
certifi	2022.12.7	Mozilla Public License 2.0 (MPL 2.0)
cffi	1.15.1	MIT
charset-normalizer	3.1.0	MIT
constants	0.6.0	BSD-3-Clause
cryptography	40.0.2	BSD or Apache License 2.0
finta	1.3	MIT
frozendict	2.3.7	BSD-3-Clause
html5lib	1.1	MIT
idna	3.4	BSD
lxml	4.9.2	BSD
MetaTrader5	5.0.44	Custom License
multitasking	0.0.11	MIT
numpy	1.26.4	BSD-3-Clause
pandas	2.2.2	BSD-3-Clause
pycparser	2.21	BSD
python-dateutil	2.8.2	Simplified BSD
pytz	2023.3	MIT
requests	2.29.0	Apache 2.0
scipy	1.10.1	BSD
six	1.16.0	MIT
soupsieve	2.4.1	MIT
ta	0.10.2	MIT
tzdata	2023.3	MIT
urllib3	1.26.15	MIT
webencodings	0.5.1	BSD
yfinance	0.2.18	Apache 2.0

Tabla A.29: Licencias

Como se puede observar, todas las herramientas empleadas son de uso libre y no imponen ninguna restricción. Por ello, y dado que la aplicación es de carácter divulgativo y educativo, se ha decidido que la licencia del proyecto será MIT, permitiendo su uso a todos los niveles.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En esta sección se establecerán los objetivos generales del proyecto y las funcionalidades que debe incluir la aplicación. Asimismo, se detallarán los requisitos y casos de uso previamente definidos para el proyecto. La organización de la explicación se distribuirá de la siguiente manera:

- Objetivos generales: los fines funcionales del proyecto.
- Requisitos: se enumerarán las distintas funcionalidades menores necesarias para cumplir con los objetivos.
- Especificación de requisitos: descripción y casos de uso de la aplicación.

B.2. Objetivos generales

Como el proyecto ha sido propuesto por mí no tiene versiones anteriores, entonces los objetivos para tener en cuenta son totalmente nuevos y no de versiones anteriores con las que podamos hacernos una idea para poder empezar. Los objetivos son los siguientes:

- Comprender la estructura en la que se basa el Bot de trading, en mi caso el manejo de la concurrencia de los hilos.
- Familiarizarse con las bibliotecas destinadas al trading algorítmico.
- Comprender los parámetros y variables de las que dependen los mercados financieros (análisis técnico, análisis fundamental, indicadores, velas...).
- Crear una estrategia de trading mediante los conocimientos adquiridos.
- Crear el sistema automatizado que realice las operaciones según las configuraciones realizadas.
- Diseñar e implementar una interfaz sencilla e intuitiva para el usuario.
- Documentar y recopilar las operaciones que hemos realizado con diferentes activos.
- Documentar y explicar el proceso mediante el cual hemos realizado el Bot de trading.
- Analizar la rentabilidad del Bot de trading.
- Enumerar y determinar que mejoras podemos implementar en futuras versiones para que el Bot de trading avance.

B.3. Catálogo de requisitos

REQUISITOS FUNCIONALES

Los requisitos funcionales del Bot de trading son los siguientes:

- R.F-1 Gestión de datos de trading: Importar y procesar datos de mercado desde archivos CSV. Almacenar y gestionar información histórica y en tiempo real de precios en formato de velas japonesas.
 - R.F-1.1 Importación de datos: Capacidad para leer datos de mercado desde archivos CSV.
 - R.F-1.2 Almacenamiento de datos: Almacenar datos históricos y en tiempo real de precios, volumen, etc.
 - R.F-1.3 Manejo de velas japonesas: Transformar datos brutos en formato de velas japonesas para análisis.
 - R.F-1.3.1 Definir las características de las velas japonesas: Definir los atributos de las velas japonesas donde guardaremos la información de cada momento por el que pasa la bolsa de valores.
 - R.F-1.3.1 Convertir la información en un Dataframe: Pasar la información recogida del CSV a la estructura de datos que manejaremos a lo largo del proyecto.
- R.F-2 Cálculo de indicadores técnicos: Calcular y programar los diferentes indicadores técnicos que vamos a utilizar en nuestra estrategia de trading.
 - R.F-2.1 RSI (Índice de Fuerza Relativa): Calcular el RSI para identificar condiciones de mercado extremas.
 - R.F-2.2 SMA (Media Móvil Simple): Utilizar la SMA para determinar tendencias del mercado.
 - R.F-2.3 Niveles de Fibonacci: Aplicar niveles de Fibonacci para soportes y resistencias.
- R.F-3 Ejecución y gestión de órdenes de trading: Ejecución de órdenes mediante la estrategia programada con su correspondiente manejo del riesgo y la documentación de cada operación para posteriormente analizarlas.

- R.F-3.1 Operaciones Automatizadas: Ejecutar automáticamente órdenes de compra y venta basadas en condiciones predefinidas.
- R.F-3.2 Gestión de Riesgo: Administrar el riesgo asociado a cada operación.
- R.F-3.3 Registro de Operaciones: Documentar todas las operaciones en un archivo CSV.
- R.F-4 Interfaz de usuario: Crear una interfaz de usuario que sea fácil e intuitiva para el usuario.
 - R.F-4.1 Configuración del Bot: Interfaz gráfica para configurar parámetros del Bot.
 - R.F-4.1.1 Elección del activo a operar: Elección de los activos disponibles dentro de nuestra carpeta de Históricos de divisas.
 - R.F-4.1.2 Elección del lotaje: Elección del lotaje con el que queremos que el Bot opere, es decir, las cantidades.
 - R.F-4.2 Control del Bot: Facilitar el inicio, la pausa y la detención del bot desde la interfaz.
- R.F-5 Manejo de múltiples hilos y procesos: Programar la ejecución del Bot de trading mediante la coordinación de hilos y su ejecución en paralelo.
 - R.F-5.1 Multitarea: Ejecución paralela de tareas de análisis y trading.
 - R.F-5.2 Sincronización de Hilos: Coordinar la interacción entre diferentes hilos de ejecución.
- R.F-6 Robustez y manejo de errores: Programar correctamente el manejo de errores de cada clase del Bot de trading.
 - R.F-6.1 Estabilidad del Sistema: Mantener operativo el Bot ante errores.
 - R.F-6.2 Registro de Errores: Crear logs detallados para la depuración.
- R.F-7 Documentación del proyecto: Documentar detalladamente cada clase de nuestro proyecto junto con sus correspondientes métodos.

REQUISITOS NO FUNCIONALES

El desarrollo de un Bot de Trading implica no solo la implementación de funcionalidades que permitan realizar operaciones en el mercado financiero, sino también la garantía de que el sistema cumpla con una serie de criterios de calidad esenciales para su correcto funcionamiento y aceptación por parte de los usuarios. Estos criterios, conocidos como requisitos no funcionales, abarcan aspectos como el rendimiento, la confiabilidad, la seguridad, la escalabilidad, la usabilidad, y la mantenibilidad del sistema. A continuación, se detallan los requisitos no funcionales que deben ser considerados en el desarrollo del Bot de Trading:

- **Rendimiento:** El sistema debe ser capaz de dar un tiempo de respuesta mínimo y mantener la optimización de los recursos.
- **Confiabilidad y Disponibilidad:** La aplicación debe funcionar de forma continua sin fallos críticos y debe tener la capacidad de recuperarse fácilmente de los fallos que pueda tener.
- **Seguridad:** Debe mantener bajo protección los datos y la información crítica que pueda estar en peligro.
- **Escalabilidad:** Capacidad de manejar un aumento en el volumen de datos o en la complejidad de las operaciones. Facilidad para añadir nuevas funcionalidades o integraciones.
- **Usabilidad:** La interfaz de usuario debe ser clara y fácil de utilizar, además de proveer guías o manuales para los usuarios.
- **Mantenibilidad y Soporte:** Código bien documentado y estructurado para permitir modificaciones y actualizaciones. Capacidad para actualizar el sistema fácilmente.

B.4. Especificación de requisitos

En este apartado se describirá los casos de uso de forma individual y detallada con el correspondiente diagrama de casos de uso general que engloba a todos los casos de uso del sistema junto con los actores. La frecuencia indicada para cada caso de uso se definirá de la siguiente manera:

1. **Baja:** Frecuencia de caso de uso que se usara menos del 30 % de las veces que la aplicación se ejecute
2. **Media:** Frecuencia de caso de uso que se usara entre el intervalo del 30 % y el 60 % de las veces que la aplicación se ejecute.
3. **Alta:** Frecuencia de caso de uso que se usara entre el intervalo del 60 % y el 90 % de las veces que la aplicación se ejecute.
4. **Muy alta:** Frecuencia de caso de uso que se usara más de un 90 % de las veces que se ejecuta la aplicación.

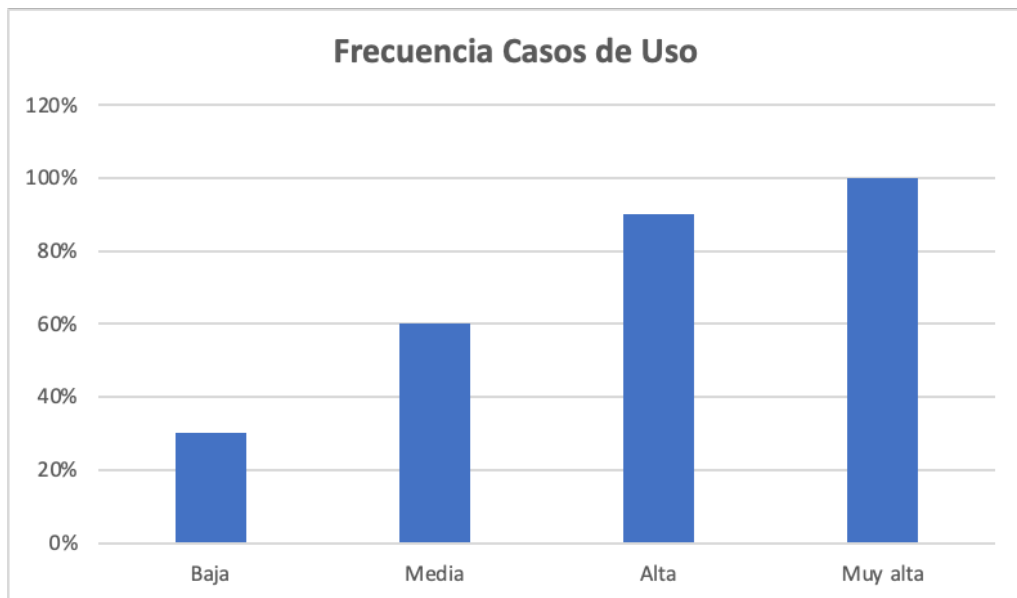


Figura B.1: Frecuencia casos de uso

DIAGRAMA DE CASOS DE USO

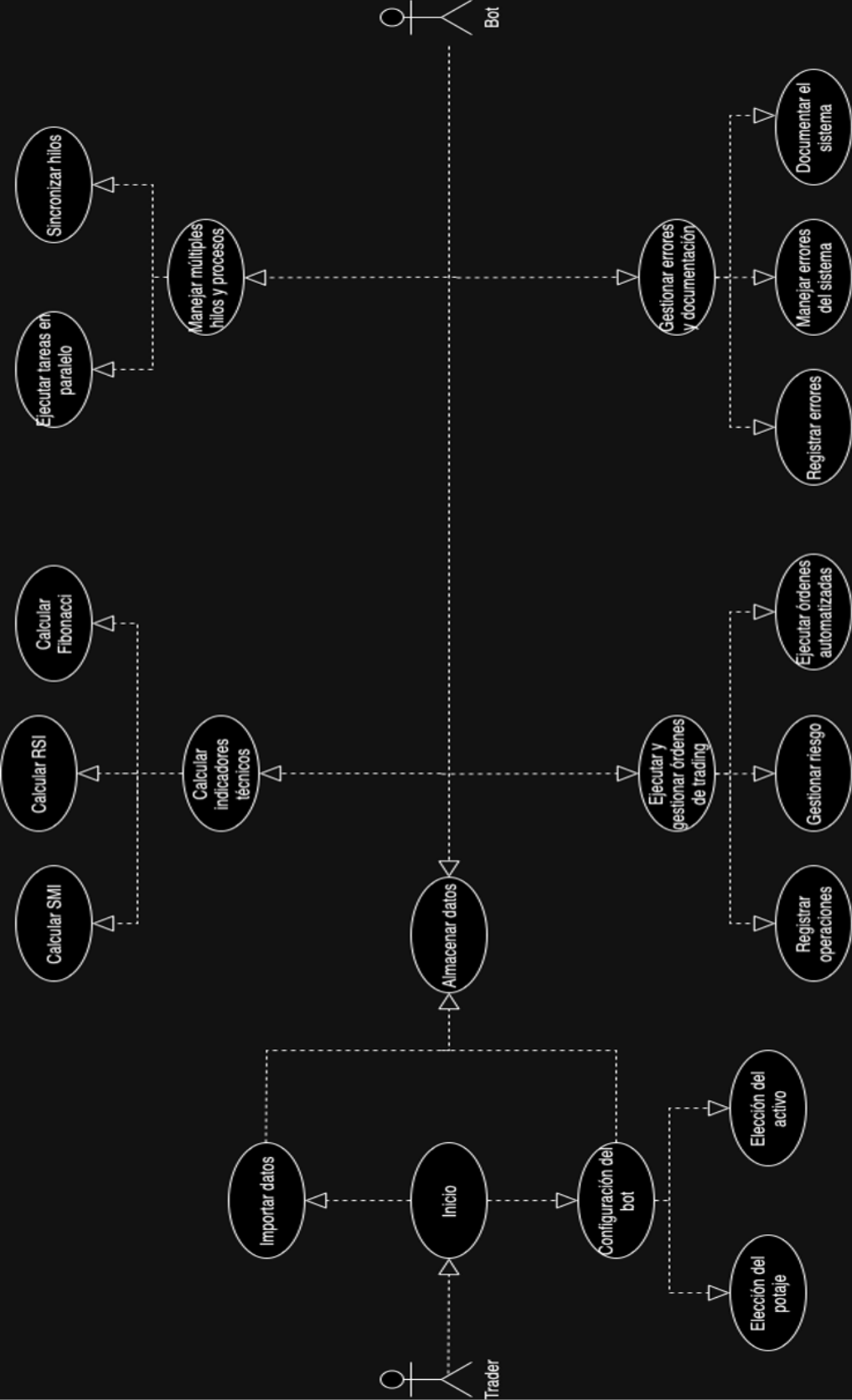


Figura B.2: DIAGRAMA DE CASOS DE USO

ACTORES

En esta aplicación solo disponemos de un único actor que será el trader, es decir, el usuario es quien se encarga de configurar el bot de trading y monitorearlo. El usuario se encarga de seleccionar el activo a operar, la temporalidad y los lotes con los que se desea operar.

Casos de Uso Relacionados con el Usuario (Trader):

■ Importar Datos (CU-1.1)

- El usuario importa datos de mercado desde archivos CSV para su análisis.

■ Transformar a Velas Japonesas (CU-1.3)

- El usuario puede necesitar configurar o iniciar este proceso, que transforma los datos brutos a formato de velas japonesas para facilitar el análisis técnico.
- Subcasos:
 - Definir Características de Velas Japonesas (CU-1.3.1)
 - Convertir la Información en un DataFrame (CU-1.3.2)

■ Interactuar con la Interfaz de Usuario (CU-4)

- El usuario configura parámetros del bot, como el activo a operar y el lotaje, y controla el inicio, la pausa y la detención del bot desde la interfaz.
- Subcasos:
 - Configurar Bot (CU-4.1)
 - Elección del Activo a Operar (CU-4.1.1)
 - Elección del Lotaje (CU-4.1.2)
 - Control del Bot (CU-4.2)

Casos de Uso Relacionados con el Sistema (Bot) y Administrador del Sistema:

■ Almacenar Datos (CU-1.2)

- El sistema almacena automáticamente los datos en tiempo real y datos históricos en una base de datos segura, gestionada por el sistema sin intervención directa del usuario.
- **Calcular Indicadores Técnicos (CU-2)**
 - Operaciones completamente automáticas que se ejecutan dentro del sistema para apoyar las decisiones de trading.
 - Subcasos:
 - **Calcular RSI (CU-2.1)**
 - **Calcular SMA (CU-2.2)**
 - **Aplicar Fibonacci (CU-2.3)**
- **Ejecutar y Gestionar Órdenes de Trading (CU-3)**
 - Estos procesos se gestionan internamente por el sistema basándose en algoritmos predeterminados.
 - Subcasos:
 - **Ejecutar Órdenes Automatizadas (CU-3.1)**
 - **Gestionar Riesgo (CU-3.2)**
 - **Registrar Operaciones (CU-3.3)**
- **Manejar Múltiples Hilos y Procesos (CU-5)**
 - La coordinación y ejecución en paralelo de tareas son manejadas internamente por el sistema para asegurar eficiencia y estabilidad.
 - Subcasos:
 - **Ejecutar Tareas en Paralelo (CU-5.1)**
 - **Sincronizar Hilos (CU-5.2)**

CASOS DE USO

Caso de uso 1	Importar Datos	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-1.1	
Descripción	El trader importa datos de mercados desde archivos CSV para su análisis	
Precondición	Deben existir archivos CSV accesibles y correctamente formateados que contengan los datos de mercado deseados.	
Acciones	Paso	Acción
	1	El trader elige el archivo CSV que desea importar desde su sistema.
	2	El sistema verifica que el archivo esté en el formato correcto y contenga los datos estructurados adecuadamente.
	3	El sistema lee los datos del archivo CSV.
Postcondiciones	Los datos importados están correctamente almacenados y listos para ser utilizados en análisis y cálculos posteriores dentro del bot de trading.	
Excepciones	Paso	Acción
	1	El archivo CSV no existe.
	2	El archivo CSV no está en el formato esperado.
	3	El sistema no puede leer el archivo debido a permisos de acceso restringidos o errores en el sistema de archivos.
Importancia	Alta	
Frecuencia	Baja	

Tabla B.1: Caso de uso: Importar Datos

Caso de uso 2	Almacenar Datos	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-1.2	
Descripción	Almacenamiento de datos, que implica el almacenamiento de datos históricos.	
Precondición	Los datos tienen que estar importados.	
Acciones	Paso	Acción
	1	Verificación de datos.
	2	Procesamiento de datos.
	3	Inserción de datos en el almacenamiento.
	4	Validación de almacenamiento.
Postcondiciones	Los datos deben estar almacenados correctamente.	
Excepciones	Paso	Acción
	1	Datos incompletos.
	2	Datos con diferente formato.
Importancia	Muy alta	
Frecuencia	Baja	

Tabla B.2: Caso de uso 2: Almacenar Datos

Caso de uso 3	Transformar Velas Japonesas	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-1.3, R.F-1.3.1, R.F-1.3.2	
Descripción	Transformar datos brutos en formato de velas japonesas para análisis.	
Precondición	Datos del mercado importados y listos para ser procesados.	
Acciones	Paso	Acción
	1	Recibir los datos.
	2	Procesamiento de los datos.
	3	Transformación de los datos.
Postcondiciones	Los datos tienen que estar almacenados en el DataFrame en formato de velas japonesas.	
Excepciones	Paso	Acción
	1	Datos incompletos.
	2	Error de procesamiento.
Importancia	Muy alta	
Frecuencia	Alta	

Tabla B.3: Caso de uso 3: Transformar Velas Japonesas

Caso de uso 4	Definir características de las velas japonesas	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-1.3.1	
Descripción	Definir los atributos de las velas japonesas como el cierre, apertura, máximo, mínimo, etc.	
Precondición	Los atributos y características deben estar bien definidas.	
Acciones	Paso	Acción
	1	Definir atributos.
	2	Documentar los atributos.
Postcondiciones	Los atributos están definidos en el sistema con su correspondiente documentación.	
Excepciones	Paso	Acción
	1	Definiciones inconsistentes.
Importancia	Muy alta	
Frecuencia	Baja	

Tabla B.4: Caso de uso 4: Definir características de las velas japonesas

Caso de uso 5	Convertir la información en un DataFrame	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-1.3.2	
Descripción	Convertir los datos para que sean almacenados en un DataFrame.	
Precondición	Datos listos para ser convertidos.	
Acciones	Paso	Acción
	1	Creación del DataFrame.
	2	Inserción de datos.
	3	Validación de datos.
Postcondiciones	El DataFrame está completamente construido y listo para ser utilizado.	
Excepciones	Paso	Acción
	1	Fallo en la creación del DataFrame.
	2	Fallo al almacenar los datos.
Importancia	Muy alta	
Frecuencia	Alta	

Tabla B.5: Caso de uso 5: Convertir la información en un DataFrame

Caso de uso 6	Cálculo de indicadores técnicos	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-2, R.F-2.1, R.F-2.2, R.F-2.3	
Descripción	El bot calcula los datos suficientes para poder utilizar los indicadores técnicos en la estrategia.	
Precondición	Datos de las velas tienen que estar cargados.	
Acciones	Paso	Acción
	1	Cálculo del RSI.
	2	Cálculo del SMA.
	3	Cálculo de los niveles de Fibonacci.
Postcondiciones	Datos de los indicadores técnicos cargados.	
Excepciones	Paso	Acción
	1	Datos erróneos de las velas.
	2	Datos erróneos de los indicadores.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.6: Caso de uso 6: Cálculo de indicadores técnicos

Caso de uso 7	Calcular RSI	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-2.1	
Descripción	Calcular el indicador técnico RSI.	
Precondición	Datos de las velas listos para ser utilizados.	
Acciones	Paso	Acción
	1	Calcular los datos necesarios para obtener el RSI.
	2	Almacenar sus resultados.
Postcondiciones	Datos del RSI correctos en su cálculo.	
Excepciones	Paso	Acción
	1	Datos erróneos de las velas.
	2	Datos erróneos en la carga del RSI.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.7: Caso de uso 7: Calcular RSI

Caso de uso 8	Calcular SMA	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-2.2	
Descripción	Cálculo del indicador técnico SMA de 200 periodos.	
Precondición	Datos de las velas listos para ser utilizados.	
Acciones	Paso	Acción
	1	Calcular los datos necesarios para obtener el SMA.
	2	Almacenar sus resultados.
Postcondiciones	Datos del SMA correctos y listos para ser usados.	
Excepciones	Paso	Acción
	1	Datos erróneos de las velas.
	2	Datos erróneos en la carga del SMA.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.8: Caso de uso 8: Calcular SMA

Caso de uso 9	Calcular Fibonacci	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-2.3	
Descripción	Calcular los niveles de Fibonacci que son importantes.	
Precondición	Datos de las velas listos para ser utilizados.	
Acciones	Paso	Acción
	1	Calcular los datos necesarios para obtener los niveles de Fibonacci.
	2	Almacenar sus resultados.
Postcondiciones	Datos de Fibonacci correctos y listos para ser usados.	
Excepciones	Paso	Acción
	1	Datos erróneos de las velas.
	2	Datos erróneos en la carga del SMA.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.9: Caso de uso 9: Calcular Fibonacci

Caso de uso 10	Ejecución y gestión de órdenes de trading	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-3, R.F-3.1, R.F-3.2, R.F-3.3	
Descripción	El bot, mediante la estrategia que se le ha implementado, ejecutará, gestionará y registrará las operaciones que detecte.	
Precondición	Cálculo de los indicadores técnicos.	
Acciones	Paso	Acción
	1	Ejecutar las órdenes de forma automática.
	2	Gestión del riesgo.
	3	Registrar las operaciones realizadas.
Postcondiciones	Órdenes ejecutadas correctamente.	
Excepciones	Paso	Acción
	1	Cálculo incorrecto de los indicadores técnicos.
	2	Fallo en la ejecución.
	3	Fallo en el registro.
	4	Exceso de riesgo.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.10: Caso de uso 10: Ejecución y gestión de órdenes de trading

Caso de uso 11	Ejecutar órdenes automatizadas	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-3.1	
Descripción	El bot ejecuta órdenes de forma automática.	
Precondición	Estrategias y parámetros configurados de forma correcta.	
Acciones	Paso	Acción
	1	Detección de oportunidades del mercado.
	2	Ejecución de órdenes.
Postcondiciones	Las órdenes se han ejecutado correctamente.	
Excepciones	Paso	Acción
	1	Fallo en la ejecución.
	2	Desviaciones en los precios.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.11: Caso de uso 11: Ejecutar órdenes automatizadas

Caso de uso 12	Gestionar riesgo	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-3.2	
Descripción	El bot gestiona el riesgo de las operaciones realizadas.	
Precondición	Las políticas de riesgo se han establecido.	
Acciones	Paso	Acción
	1	Evaluación del riesgo.
	2	Ajuste de operaciones.
Postcondiciones	Riesgo controlado.	
Excepciones	Paso	Acción
	1	Exceso de riesgo.
	2	Cambios de mercado rápidos.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.12: Caso de uso 12: Gestionar riesgo

Caso de uso 13	Registrar operaciones	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-3.3	
Descripción	El bot registra las operaciones realizadas en un CSV.	
Precondición	Sistema de registro de operaciones operativo.	
Acciones	Paso	Acción
	1	Documentación de detalles de la operación.
	2	Almacenamiento de datos.
Postcondiciones	Historial de operaciones documentado.	
Excepciones	Paso	Acción
	1	Fallo de registro.
	2	Datos corruptos o incompletos.
Importancia	Alta	
Frecuencia	Muy alta	

Tabla B.13: Caso de uso 13: Registrar operaciones

Caso de uso 14	Interactuar con la interfaz de usuario	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-4, R.F-4.1, R.F-4.1.1, R.F-4.1.2, R.F-4.2	
Descripción	Interacción del usuario con la interfaz del bot de trading.	
Precondición	Interfaz de configuración accesible.	
Acciones	Paso	Acción
	1	Elección del lotaje.
	2	Elección del activo a operar.
	3	Iniciar bot.
Postcondiciones	Bot configurado con todas las condiciones.	
Excepciones	Paso	Acción
	1	Datos inválidos.
	2	Selecciones inválidas.
Importancia	Alta	
Frecuencia	Baja	

Tabla B.14: Caso de uso 14: Interactuar con la interfaz de usuario

Caso de uso 15	Configurar bot	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-4.1	
Descripción	Configuración del bot de trading con el lotaje a operar y el tipo de activo.	
Precondición	Acceso a la interfaz por parte del usuario.	
Acciones	Paso	Acción
	1	Acceso a la configuración.
	2	Configurar el lotaje.
	3	Elegir el activo a operar.
Postcondiciones	Bot configurado con todas las condiciones.	
Excepciones	Paso	Acción
	1	Acceso denegado.
	2	Datos inválidos.
Importancia	Moderada	
Frecuencia	Baja	

Tabla B.15: Caso de uso 15: Configurar bot

Caso de uso 16	Elección del activo a operar	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-4.1.1	
Descripción	El usuario elige el activo con el que el bot operará.	
Precondición	Interfaz del usuario accesible.	
Acciones	Paso	Acción
	1	Listado de activos disponibles.
	2	Selección de un activo.
Postcondiciones	Activo seleccionado.	
Excepciones	Paso	Acción
	1	Especificar el lotaje.
	2	Validación del lotaje.
Importancia	Moderada	
Frecuencia	Muy baja	

Tabla B.16: Caso de uso 16: Elección del activo a operar

Caso de uso 17	Elección del lotaje	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-4.1.2	
Descripción	El usuario elige el lotaje con el que el bot operará.	
Precondición	Interfaz del usuario accesible.	
Acciones	Paso	Acción
	1	Especificar lotaje.
	2	Validación lotaje.
Postcondiciones	Lotaje configurado.	
Excepciones	Paso	Acción
	1	Valor del lotaje no válido.
Importancia	Moderada	
Frecuencia	Muy baja	

Tabla B.17: Caso de uso 17: Elección del lotaje

Caso de uso 18	Control del Bot	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-4.2	
Descripción	El usuario controla el funcionamiento del bot.	
Precondición	El bot debe de estar configurado.	
Acciones	Paso	Acción
	1	Iniciar bot.
	2	Detener bot.
Postcondiciones	Estado del bot modificado.	
Excepciones	Paso	Acción
	1	Fallo de operación.
Importancia	Moderada	
Frecuencia	Baja	

Tabla B.18: Caso de uso 18: Control del Bot

Caso de uso 19	Manejo de múltiples hilos y procesos	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-5	
Descripción	Manejo de los hilos y procesos en el bot de trading.	
Precondición	Recursos adecuados para manejar hilos y procesos.	
Acciones	Paso	Acción
	1	Ejecutar tareas en paralelo.
	2	Sincronizar hilos.
Postcondiciones	Tareas y procesos completados de forma simultánea.	
Excepciones	Paso	Acción
	1	Fallo al ejecutar tareas en paralelo.
	2	Fallo al sincronizar hilos.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.19: Caso de uso 19: Manejo de múltiples hilos y procesos

Caso de uso 20	Ejecutar tareas en paralelo	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-5.1	
Descripción	El bot ejecuta varias tareas en paralelo.	
Precondición	Sistema capaz de realizar multitareas.	
Acciones	Paso	Acción
	1	Identificación de tareas.
	2	Creación de hilos.
	3	Ejecución en paralelo.
	4	Monitorización de hilos.
Postcondiciones	Tareas completadas simultáneamente.	
Excepciones	Paso	Acción
	1	Sobrecarga de recursos.
	2	Fallo de hilos.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.20: Caso de uso 20: Ejecutar tareas en paralelo

Caso de uso 21	Sincronizar hilos	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-5.2	
Descripción	Sincronización de los hilos con los que se ejecuta el bot de trading.	
Precondición	Hilos en ejecución.	
Acciones	Paso	Acción
	1	Definición de puntos de sincronización.
	2	Implementación de mecanismos de sincronización.
	3	Control de condiciones de carrera.
	4	Revisión y ajuste de sincronización.
Postcondiciones	Ejecución coordinada de tareas.	
Excepciones	Paso	Acción
	1	Errores de sincronización.
	2	Inconsistencia de datos.
Importancia	Muy alta	
Frecuencia	Muy alta	

Tabla B.21: Caso de uso 21: Sincronizar hilos

Caso de uso 22	Gestionar errores y documentación	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-6, R.F-6.1, R.F-6.2, R.F-6.3	
Descripción	Gestionar y documentar los errores que se produzcan en el bot de trading.	
Precondición	El bot de trading debe de estar activo.	
Acciones	Paso	Acción
	1	Manejar errores del sistema.
	2	Registrar errores.
	3	Documentar el sistema.
Postcondiciones	Estabilidad del sistema restaurada.	
Excepciones	Paso	Acción
	1	Fallo a la hora de manejar los errores.
	2	No registra los errores.
	3	El sistema no esté documentado.
Importancia	Baja	
Frecuencia	Baja	

Tabla B.22: Caso de uso 22: Gestionar errores y documentación

Caso de uso 23	Manejar errores del sistema	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-6.1	
Descripción	Manejar los errores que se produzcan a lo largo de la ejecución del sistema.	
Precondición	El bot debe estar funcionando.	
Acciones	Paso	Acción
	1	Detección de errores.
	2	Clasificación de errores.
	3	Resolución de errores.
	4	Recuperación del sistema.
Postcondiciones	Estabilidad del sistema después del manejo de errores.	
Excepciones	Paso	Acción
	1	Fallo de recuperación.
	2	Errores no detectados.
Importancia	Moderada	
Frecuencia	Baja	

Tabla B.23: Caso de uso 23: Manejar errores del sistema

Caso de uso 24	Registrar errores	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-6.2	
Descripción	Registrar los errores que se detecten en el bot de trading.	
Precondición	Sistema de registro de errores configurado.	
Acciones	Paso	Acción
	1	Captura de la información del error.
	2	Almacenamiento de logs.
	3	Notificación de errores.
Postcondiciones	Registro completo de errores.	
Excepciones	Paso	Acción
	1	Fallo en el registro de los errores.
Importancia	Moderada	
Frecuencia	Baja	

Tabla B.24: Caso de uso 24: Registrar errores

Caso de uso 25	Documentar el sistema	
Versión	1.0	
Autor	Sergio Rebollo Ortega	
Requisitos asociados	R.F-6.3	
Descripción	Documentar el sistema para que resulte más fácil de realizar cambios en el futuro.	
Precondición	Existencia del bot de trading.	
Acciones	Paso	Acción
	1	Creación de la documentación.
	2	Actualización continua.
	3	Accesibilidad a la documentación.
Postcondiciones	Documentación completa.	
Excepciones	Paso	Acción
	1	Información desactualizada.
	2	Acceso restringido.
Importancia	Baja	
Frecuencia	Moderada	

Tabla B.25: Caso de uso 25: Documentar el sistema

Apéndice C

Especificación de diseño

C.1. Introducción

En esta sección se van a definir los datos empleados por la aplicación, como están estructurados y almacenados.

C.2. Diseño de datos

Estas variables corresponden a los datos que introducirá el usuario a través de la interfaz gráfica.

Variables Bot ejecución

- **CSV datos históricos:** CSV con todos los datos correspondientes al activo sobre el cual va a actuar el bot de trading. Además, este fichero se desglosa en secciones las cuales almacenará el bot más adelante. Este fichero lo elegirá el usuario de entre una ristra de datos históricos de activos.
- **Lotaje:** Dato introducido por el usuario a través del cual el bot utilizará para realizar sus operaciones.

DataFrame temporal.

En este DataFrame temporal el bot almacena los datos referidos a los históricos de activos desglosados en la apertura, el cierre, máximo, mínimo, el volumen, y la fecha de la vela.

- **Time_last_candle:** Contendrá el tiempo de la vela del activo financiero.
- **Open:** Contendrá el precio de apertura de la vela en concreto.
- **Close:** Contendrá el precio de cierre de la vela.
- **High:** Almacenará el precio más alto de la vela.
- **Low:** Contendrá el precio mínimo de la vela.
- **Volume:** Contendrá el volumen que se ha manejado en el mercado a lo largo del ciclo de vida de la vela.

Operaciones

CSV donde se almacenarán los datos de las operaciones que está realizando el bot de trading.

- **Tipo_operacion:** El tipo de operación que ha realizado el bot, es decir, si es de compra o de venta.
- **Fecha_apertura:** Fecha de apertura de la operación.
- **Fecha_cierre:** Fecha de cierre de la operación.
- **Precio_apertura:** Precio de apertura de la operación.
- **Precio_cierre:** Precio de cierre de la operación.
- **Dinero_ganado_perdido:** Beneficio o pérdida de las operaciones realizadas por el bot de trading.

Indicadores Técnicos

Datos de los indicadores técnicos para su uso en la toma de decisiones del bot de trading.

- **RSI:** Dato actualizado del nivel de RSI en cada vela del mercado, actualizándose continuamente para detectar posibles entradas.
- **SMA:** Dato actualizado del nivel de SMA en cada vela del mercado, actualizándose continuamente para detectar posibles entradas.
- **Fibonacci:** Precios almacenados que corresponden a los niveles de Fibonacci que hemos calculado, actualizándose continuamente.

C.3. Diseño arquitectónico

En esta sección se explica la organización del proyecto, es decir, la estructura que tiene a nivel de paquetes y cómo sus clases se conectan entre ellas.

Modelo-Vista-Presentador (MVP)

El patrón MVP se estructura en tres componentes: Modelo, Vista y Presentador, cada uno con tareas claramente definidas.

- **Modelo:** En el contexto del bot de trading, el modelo gestiona los datos de trading, realiza cálculos de indicadores técnicos y ejecuta las órdenes de trading según las estrategias definidas.
- **Vista:** Es la interfaz de usuario donde los usuarios interactúan con el sistema. Para este bot de trading, esto incluye interfaces para configurar parámetros de trading.
- **Presentador:** Actúa como intermediario entre la vista y el modelo. Recoge las entradas de la vista, las calcula y luego actualiza la vista. El presentador también maneja la lógica de control que responde a entradas del usuario y eventos del sistema.



Figura C.1: Modelo-Vista-Presentador

Diagrama de paquetes

Estructura que sigue el bot de trading en cuanto a los paquetes que componen el proyecto y los subpaquetes que forman los paquetes.

- **/DiagramasClases/**: Carpeta donde se han realizado los diagramas de clases.
- **/HistóricosDivisas/**: Carpeta donde se guardan todos los datos históricos de los activos con los que el bot puede operar.
- **/Operaciones/**: Carpeta donde el bot guarda los resultados de las operaciones que se han realizado con los datos históricos de los activos.
- **/src/**: Carpeta que contiene todos los archivos de código necesarios para el correcto funcionamiento del proyecto.

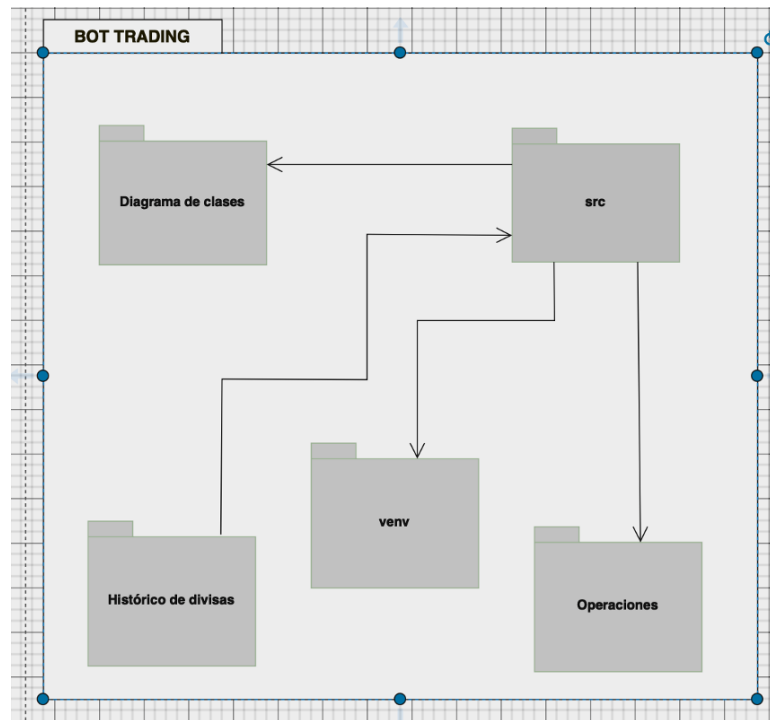


Figura C.2: Diagrama de Paquetes

Diagrama de clases

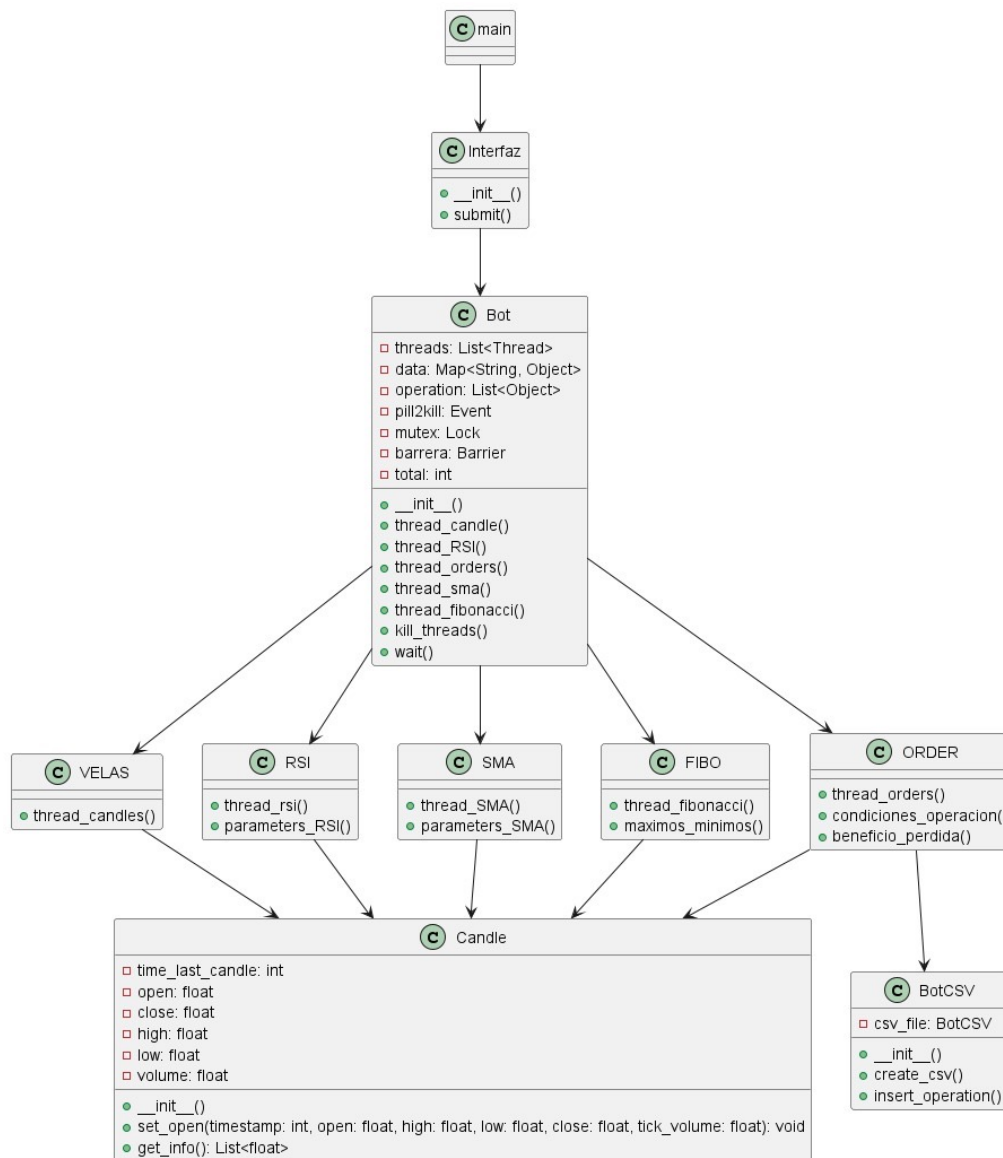


Figura C.3: Diagrama de clases

Este diagrama de clases representa la estructura que se utilizará para el Bot de trading, que opera basado en análisis técnico del mercado.

Clase Principal:

- **main:** Es la clase principal que inicia el sistema. Su papel es probablemente instanciar y lanzar la interfaz y el bot.

Interfaz de Usuario:

- **Interfaz:** Esta clase gestiona la interfaz de usuario. Tiene dos métodos:
 - `__init__()`: Es el constructor de la clase, que inicializa una nueva instancia de la interfaz.
 - `submit()`: Este método se utilizaría para enviar comandos o datos desde la interfaz de usuario al sistema, posiblemente para iniciar el trading o definir parámetros.

Bot de Trading:

- **Bot:** Es la clase central que coordina las operaciones de trading. Tiene varios atributos y métodos:
 - Atributos como listas de threads (hilos), mapas para datos, eventos, bloqueos y barreras para la sincronización de hilos.
 - `__init__()`: Constructor de la clase.
 - Métodos `thread_*`: Parecen ser funciones que inician hilos de ejecución para diferentes tareas del bot, como `thread_candle()` para obtener datos de velas de precios, `thread_RSI()` para calcular el índice de fuerza relativa, etc.
 - `kill_threads()`: Un método para detener todos los hilos de ejecución.
 - `wait()`: Probablemente para pausar la ejecución hasta que cierta condición se cumpla.

Componentes del Análisis Técnico:

■ **VELAS:**

- `thread_candles()`: Método para recoger datos del mercado en forma de velas.

■ **RSI:**

- `thread_rsi()`: Método para calcular el Índice de Fuerza Relativa, un indicador técnico.
- `parameters_RSI()`: Método o atributo para los parámetros que configuran el cálculo del RSI.

■ **SMA:**

- `thread_SMA()`: Método para calcular la Media Móvil Simple.
- `parameters_SMA()`: Método o atributo para los parámetros que configuran el cálculo de la SMA.

■ **FIBO:**

- `thread_fibonacci()`: Método para analizar o aplicar niveles de retroceso de Fibonacci.
- `maximos_minimos()`: Método o atributo para identificar máximos y mínimos, importantes para el trazado de Fibonacci.

■ **ORDER:**

- `thread_orders()`: Método para gestionar órdenes de compra o venta.
- `condiciones_operacion()`: Método o atributo que define las condiciones bajo las cuales se ejecutarán las órdenes.
- `beneficio_perdida()`: Método o atributo para calcular la ganancia o pérdida de las operaciones comerciales.

Clase Candle:

■ **Atributos:**

- `time_last_candle`: Entero que representa el tiempo de la última vela.

- open, close, high, low, volume: Atributos de tipo flotante que representan el precio de apertura, cierre, máximo, mínimo y volumen de la vela, respectivamente.

■ **Métodos:**

- `__init__()`: Constructor de la clase.
- `set_open()`: Método para establecer los valores de una nueva vela.

Clase BotCSV:

■ **Atributos:**

- `csv_file`: Cadena que representa el nombre del archivo CSV donde se registrarán las operaciones.

■ **Métodos:**

- `__init__()`: Constructor de la clase que inicializa el archivo CSV.
- `create_csv()`: Método para crear un nuevo archivo CSV.
- `insert_operation()`: Método para insertar operaciones en el archivo CSV.

El diagrama muestra las relaciones entre clases con líneas que conectan las clases. Por ejemplo, Bot está conectado a VELAS, RSI, etc., lo que significa que Bot utiliza estas clases para realizar sus operaciones.

Apéndice D

Documentación técnica de programación

D.1. Introducción

En esta sección de los anexos, correspondiente al apéndice D, se explican los detalles a tener en cuenta para una posible implementación por parte de un usuario interesado en probar la estrategia de inversión. Este apartado se elabora con el fin de proporcionar al cliente una guía concisa para utilizar un nuevo proyecto, mejorar el proyecto existente o implementarlo con un activo específico, tal como se mencionó anteriormente.

D.2. Estructura de directorios

Repositorio Bot de Trading

- **/DiagramasClases/**: Carpeta donde se han realizado los diagramas de clases.
- **/HistoricosDivisas/**: Carpeta donde se guardan todos los datos históricos de los activos con los que el bot puede operar.
- **/Operaciones/**: Carpeta donde el bot guarda los resultados de las operaciones que se han realizado con los datos históricos de los activos.
- **/src/**: Carpeta que contiene todos los archivos de código necesarios para el correcto funcionamiento del proyecto.

- **/Log.errors/**: Carpeta donde se guardarán los posibles errores que puedan ocurrir al ejecutar el bot de trading.
- **/venv/**: Carpeta que pertenece al entorno virtual del proyecto, donde se guardarán las bibliotecas específicas con sus versiones para el proyecto.
- **/requirements.txt**: Fichero para instalar en un momento determinado todas las bibliotecas de las que depende el proyecto con sus respectivas versiones.
- **/run_bot.bat**: Archivo para ejecutar el proyecto en un entorno virtual con un sistema operativo Windows.
- **/run_bot.sh**: Archivo para ejecutar el proyecto en un entorno virtual con un sistema operativo Linux o Mac.

Repositorio GitHub

- **/LICENSE**: Fichero en el que se especifica la licencia del proyecto, en este caso, una licencia MIT.
- **/README.md**: Descripción de las funcionalidades del bot de trading y de las implementaciones que se intentarán realizar en el proyecto a lo largo de su duración.
- **/Repositorio Bot de Trading**: Estructura explicada en el apartado anterior.

D.3. Manual del programador

Instalación y Ejecución del Bot de Trading en Local

A continuación, se explicará paso a paso cómo instalar y ejecutar el Bot de Trading en local.

Ejecución

Primero de todo, se descarga el repositorio en el sistema y se leen todos los ficheros para comprender bien su funcionamiento. Después de descargar el repositorio, cuando se tenga todo preparado, se abrirá la terminal del sistema y se comprobará la versión de Python con el siguiente comando:

`Python --version`

Este comando devolverá la versión de Python que contiene el sistema con el que se está ejecutando el proyecto. Es muy recomendable utilizar la versión de Python indicada, ya que puede que el proyecto no funcione con otras versiones.

El siguiente paso es ejecutar el fichero `.bat` llamado `run_bot.bat`. Este fichero ejecutará directamente el Bot de Trading sin necesidad de que el nuevo usuario tenga que realizar ninguna instalación aparte de la versión de Python utilizada. El fichero `.bat` es para los sistemas Windows. Además de este fichero, en el repositorio se encuentra el fichero `run_bot.sh`, destinado a los sistemas que utilizan Linux o macOS.

run_bot.bat

Se explicará el contenido del fichero `.bat` para comprender lo que se ha realizado para poder ejecutar el proyecto.

1. REM

Comentario: Guardar la ruta del directorio donde se encuentra el `.bat`

2. set SCRIPT_DIR= % dp0

Comentario: El comando `REM` es una etiqueta de comentario en los archivos por lotes de Windows. Después, se guarda la ruta del fichero `.bat` en la variable `SCRIPT_DIR`.

3. REM Navegar al directorio del script

4. cd /d "%SCRIPT_DIR%"

Comentario: Con el comando `cd /d "%SCRIPT_DIR%"` se navega hasta la ruta que contenga la variable mencionada.

5. REM Activar el entorno virtual

6. call venv\Scripts\activate.bat

Comentario: El siguiente comando `call venv\Scripts\activate.bat` ejecuta el script `activate.bat` para activar el entorno virtual de Python ubicado en `venv\Scripts`.

7. REM Instalar las dependencias (si es necesario)

8. pip install -r requirements.txt

Comentario: El comando `pip install -r requirements.txt` instala las referencias necesarias para el proyecto que se encuentran en el fichero `requirements.txt`. Si ya están instaladas, no ocurrirá nada.

9. cd src

Comentario: Se cambia el directorio de trabajo a `src`.

10. python main.py

Comentario: Ejecuta el script principal del proyecto.

11. cd ..

Comentario: Se cambia al directorio padre.

12. deactivate

Comentario: Se desactiva el entorno virtual.

```
@echo off
REM Guardar la ruta del directorio donde se encuentra el .bat
set SCRIPT_DIR=%~dp0

REM Navegar al directorio del script
cd /d "%SCRIPT_DIR%"

REM Activar el entorno virtual
call venv\Scripts\activate.bat

REM Instalar las dependencias (si es necesario)
pip install -r requirements.txt

cd src

REM Ejecutar el bot de trading (asegúrate de cambiar 'main.py' al archivo principal de tu bot)
python main.py

REM
cd ..

REM Desactivar el entorno virtual
deactivate

pause
```

Figura D.1: Runbot.bat

En la figura [D.1](#) se observa la descripción del código que se ha seguido paso por paso anteriormente

run_bot.sh

A continuación, se explicará paso por paso el script `run_bot.sh` que se utiliza para ejecutar el Bot de Trading en un entorno Unix (Linux o macOS).

1. `#!/bin/bash`

Comentario: Esta línea indica que el script debe ser ejecutado por el intérprete de Bash.

2. Guardar la ruta del directorio donde se encuentra el script `SCRIPT_DIR= (cd "$(dirname BASH_SOURCE[0]) > /dev/null pwd)"`

Comentario: Estas líneas guardan la ruta del directorio donde se encuentra el script en la variable `SCRIPT_DIR`. Esto se hace utilizando una combinación de comandos para obtener el directorio del script independientemente de dónde se ejecute.

3. Navegar al directorio del script `cd "SCRIPT_DIR"`

Comentario: Con este comando, se navega al directorio que contiene el script, asegurando que todos los comandos posteriores se ejecuten en el contexto correcto.

4. Activar el entorno virtual `source venv/bin/activate`

Comentario: Este comando activa el entorno virtual de Python ubicado en `venv/bin/activate`. Esto garantiza que el script utilice las dependencias y la versión de Python especificadas en el entorno virtual.

5. Instalar las dependencias (si es necesario) `pip install -r requirements.txt`

Comentario: Este comando instala las dependencias necesarias para el proyecto, que están listadas en el fichero `requirements.txt`. Si las dependencias ya están instaladas, el comando no realiza ningún cambio.

6. `cd src`

Comentario: Se cambia el directorio de trabajo a `src`, que es donde se encuentra el código fuente del proyecto.

7. **Ejecutar el bot de trading** (asegúrate de cambiar 'main.py' al archivo principal de tu bot) `python main.py`

Comentario: Este comando ejecuta el script principal del proyecto, `main.py`. Es importante asegurarse de que `main.py` sea el archivo correcto a ejecutar para iniciar el bot de trading.

8. `cd ..`

Comentario: Se regresa al directorio padre (`SCRIPT_DIR`) desde el directorio `src`.

9. **Desactivar el entorno virtual** `deactivate`

Comentario: Este comando desactiva el entorno virtual de Python, retornando al entorno de sistema original.

10. `read -p "Presiona cualquier tecla para salir..."`

Comentario: Finalmente, este comando muestra un mensaje en la terminal y espera a que el usuario presione cualquier tecla antes de cerrar la ventana de terminal. Esto es útil para asegurarse de que el usuario pueda ver cualquier mensaje o resultado final antes de que la terminal se cierre.

```
#!/bin/bash
# Guardar la ruta del directorio donde se encuentra el script
SCRIPT_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" &> /dev/null && pwd )"

# Navegar al directorio del script
cd "$SCRIPT_DIR"

# Activar el entorno virtual
source venv/bin/activate

# Instalar las dependencias (si es necesario)
pip install -r requirements.txt

cd src

# Ejecutar el bot de trading (asegúrate de cambiar 'main.py' al archivo principal de tu bot)
python main.py

cd ..

# Desactivar el entorno virtual
deactivate

read -p "Presiona cualquier tecla para salir..."
```

Figura D.2: `Run_bot.sh`

En la figura [D.2](#) se observa la descripción del código que se ha seguido paso por paso anteriormente

Documentación de usuario

E.1. Introducción

En esta sección se explicarán los conceptos necesarios para la ejecución del proyecto por parte de un usuario que desee utilizarlo. Se detallarán los procesos que debe seguir para instalar el Bot de Trading y se proporcionará un manual de usuario.

E.2. Requisitos de usuarios

El Bot de Trading se accederá únicamente de forma local a través del equipo del usuario que desee utilizarlo.

Es decir, en Windows se accederá ejecutando el fichero 'run_bot.bat' desde la terminal o haciendo doble clic en el archivo desde la interfaz gráfica en la carpeta donde se encuentre el fichero.

En Linux o macOS, se ejecutará navegando hasta el fichero 'run_bot.sh' desde la terminal. Se deben dar permisos de lectura, escritura y ejecución para poder ejecutar el fichero 'run_bot.sh' sin problemas, y por último, ejecutarlo introduciendo los siguientes comandos:

1. `chmod +rwx run_bot.sh`
2. `./run_bot.sh`

E.3. Instalación

Para realizar la instalación de forma local del Bot de Trading se seguirán los pasos comentados y detallados en el manual del programador.

E.4. Manual del usuario

En el último apartado de esta sección se explicará toda la interfaz gráfica de la que dispone el Bot de Trading. Va a ser un proceso sencillo ya que no consta de mucha interfaz gráfica debido a que su ejecución es muy simple.

Interfaz Bot de Trading

En la figura E.1 se muestra la interfaz de configuración del Bot de Trading. Esta interfaz permite al usuario especificar dos parámetros fundamentales antes de iniciar el bot:



Figura E.1: Interfaz Bot

1. **Lotaje:** Un campo de entrada de texto donde el usuario puede introducir el tamaño del lote que desea utilizar para las operaciones de trading.
2. **Seleccionar Histórico:** Un menú desplegable que permite al usuario seleccionar el archivo histórico de datos que se utilizará para realizar las operaciones de trading. Este archivo contiene los datos históricos necesarios para el análisis y la ejecución de las estrategias de trading.

Finalmente, se incluye un botón etiquetado como **Iniciar Bot**", el cual, al ser presionado, inicia la ejecución del bot de trading con los parámetros especificados. Esta interfaz proporciona una manera sencilla y directa para que el usuario configure y ejecute el bot de trading.

En la figura E.2 se muestran dos ventanas de la interfaz de usuario del Bot de Trading. La primera ventana corresponde a la configuración inicial del bot, mientras que la segunda es una advertencia emergente.

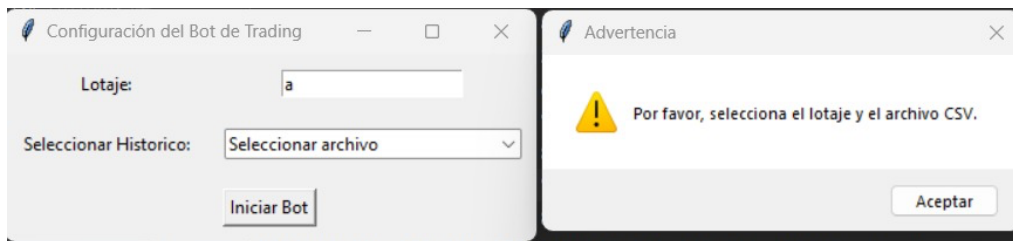


Figura E.2: Interfaz error

■ Ventana de Configuración del Bot de Trading:

- **Lotaje:** Un campo de entrada de texto donde el usuario debe introducir el tamaño del lote para las operaciones de trading. En la imagen, se observa que el campo está incompletamente llenado con la letra 'a'.
- **Seleccionar Histórico:** Un menú desplegable que permite al usuario seleccionar un archivo histórico de datos (en formato CSV) para realizar las operaciones de trading.
- **Botón 'Iniciar Bot':** Un botón que, al ser presionado, intenta iniciar el bot de trading con los parámetros proporcionados por el usuario.

■ Ventana de Advertencia:

- **Mensaje de Advertencia:** Una ventana emergente que aparece cuando el usuario no ha proporcionado todos los datos necesarios. El mensaje advierte al usuario con el texto: "Por favor, selecciona el lotaje y el archivo CSV."
- **Botón 'Aceptar':** Un botón que permite al usuario cerrar la ventana de advertencia después de leer el mensaje.

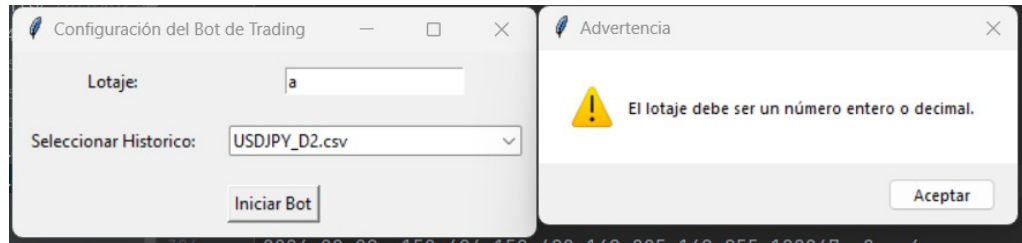


Figura E.3: Interfaz error 2

- **Ventana de Configuración del Bot de Trading:**

- **Lotaje**
- **Seleccionar Histórico**
- **Botón 'Iniciar Bot'**

- **Ventana de Advertencia:**

- **Mensaje de Advertencia:** Una ventana emergente que aparece cuando el usuario ha proporcionado un valor incorrecto para el lotaje. El mensaje advierte al usuario con el texto: 'El lotaje debe ser un número entero o decimal.'
- **Botón 'Aceptar':** Un botón que permite al usuario cerrar la ventana de advertencia después de leer el mensaje.

Esta interfaz de usuario está diseñada para asegurar que todos los parámetros necesarios sean proporcionados antes de iniciar el bot, evitando errores en la ejecución del programa.



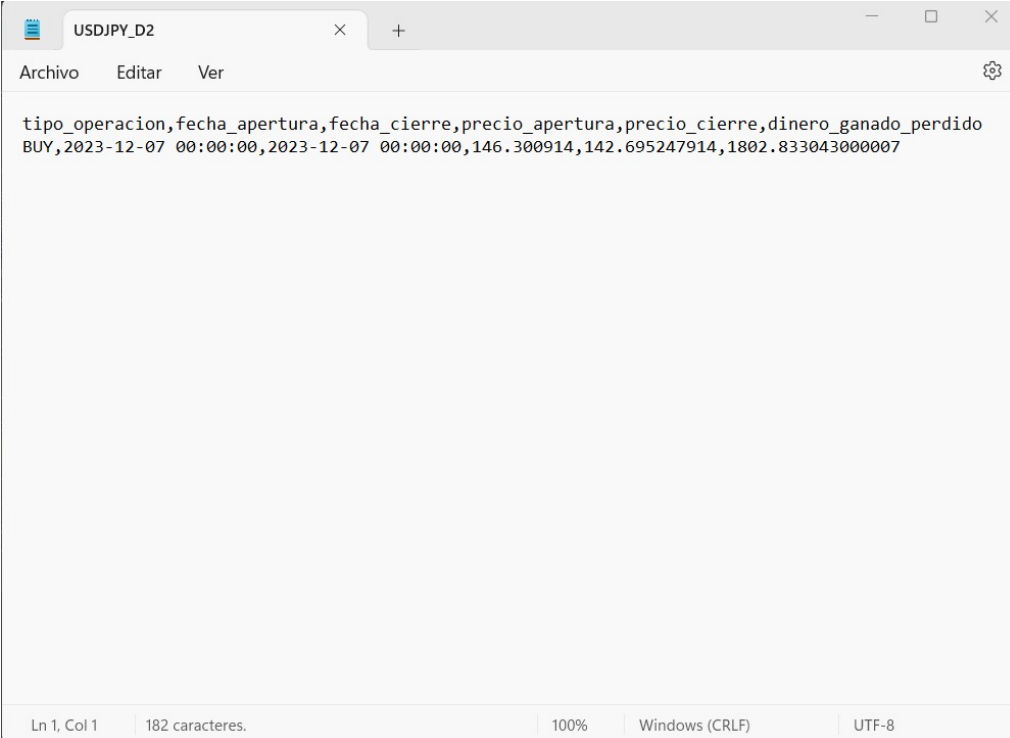
Figura E.4: Interfaz correcta

En la figura E.4 se muestra la ventana de configuración del Bot de Trading con los parámetros correctos.

■ **Ventana de Configuración del Bot de Trading:**

- **Lotaje:** Un campo de entrada de texto donde el usuario debe introducir el tamaño del lote para las operaciones de trading. En la imagen, se observa que el campo está correctamente llenado con el valor '0.05'.
- **Seleccionar Histórico:** Un menú desplegable que permite al usuario seleccionar un archivo histórico de datos (en formato CSV) para realizar las operaciones de trading. En la imagen, se ha seleccionado el archivo 'USDJPY_D2.csv'.
- **Botón 'Iniciar Bot':** Un botón que, al ser presionado, inicia el bot de trading con los parámetros proporcionados por el usuario.

Tras iniciar el bot de Trading con los parámetros correctos, realizara operaciones hasta que termine y muestre el resultado en un fichero formato notepad, se muestra en la Figura E.5



```
tipo_operacion,fecha_apertura,fecha_cierre,precio_apertura,precio_cierre,dinero_ganado_perdido
BUY,2023-12-07 00:00:00,2023-12-07 00:00:00,146.300914,142.695247914,1802.833043000007
```

Figura E.5: Resultados Bot

El contenido del archivo incluye las siguientes columnas:

- **tipo_operacion:** Indica el tipo de operación realizada (por ejemplo, "BUY").
- **fecha_apertura:** Fecha y hora en que se abrió la operación.
- **fecha_cierre:** Fecha y hora en que se cerró la operación.
- **precio_apertura:** Precio del activo en el momento de la apertura de la operación.
- **precio_cierre:** Precio del activo en el momento de cierre de la operación.
- **dinero_ganado_perdido:** Cantidad de dinero ganada o perdida en la operación.

En la imagen se puede observar una fila de datos con la siguiente información, :

- **tipo_operacion:** 'BUY'
- **fecha_apertura:** '2023-12-07 00:00:00'
- **fecha_cierre:** '2023-12-07 00:00:00'
- **precio_apertura:** '146.300914'
- **precio_cierre:** '142.695247914'
- **dinero_ganado_perdido:** '1802.833043000007'

Esta información muestra la operación que el bot ha realizado en este tramo y que ha salido ganadora. El ejemplo puesto esta realizado en un periodo de tiempo corto por eso solo realiza una operación.

Anexo de sostenibilización curricular

F.1. Introducción

En un mundo cada vez más consciente de los desafíos ambientales y sociales, la integración de la sostenibilidad en la formación académica es esencial. Este enfoque se vuelve particularmente relevante en disciplinas como la tecnología financiera, donde las decisiones y herramientas pueden tener amplios impactos económicos, sociales y ambientales. El desarrollo de un bot de trading en mi TFG no solo refleja avances tecnológicos, sino también un compromiso con prácticas sostenibles, demostrando cómo la tecnología puede ser una fuerza positiva para el cambio sostenible.

Durante mi formación, adquirí competencias clave en sostenibilidad, las cuales apliqué en el desarrollo de mi bot de trading. Primero, implementé algoritmos que consideran factores ESG (ambientales, sociales y de gobernanza) para tomar decisiones de inversión, lo que alinea las operaciones del bot con prácticas de inversión sostenible. Además, diseñé el sistema para ser energéticamente eficiente, reduciendo su huella de carbono. Este enfoque no solo optimiza el rendimiento del bot desde una perspectiva financiera, sino que también promueve responsabilidad ambiental y social.

Este proyecto ha sido una oportunidad significativa para reflexionar sobre el papel que la tecnología y las finanzas pueden jugar en la promoción de un futuro sostenible. Integrar principios de sostenibilidad en el diseño y operación de tecnología financiera ha profundizado mi comprensión de la responsabilidad ética como desarrollador. Los desafíos, aunque numerosos,

reafirmaron mi creencia en la necesidad de soluciones tecnológicas que prioricen la sostenibilidad. Esta experiencia ha moldeado mi visión profesional y reforzado mi compromiso de seguir contribuyendo a la innovación sostenible.

La integración de la sostenibilidad en el desarrollo de mi bot de trading no solo ha enriquecido mi proyecto de TFG, sino que también ha contribuido a mi desarrollo profesional y personal. Este enfoque ha demostrado ser no solo viable, sino esencial para crear soluciones financieras que sirvan a la sociedad de manera integral. Al mirar hacia el futuro, estoy comprometido a continuar explorando y promoviendo la implementación de prácticas sostenibles en tecnología, asegurando que la innovación financiera avance en armonía con nuestros valores éticos y ambientales.

Bibliografía
