UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Aplicación para la monitorización de operaciones en el mercado de valores

Trabajo Fin de Grado

Grado en Ciencia de Datos

AUTOR/A: Guiral Ortuño, Paloma

Tutor/a: Busquets Mataix, José Vicente

CURSO ACADÉMICO: 2022/2023

Resumen

La bolsa de valores es un mercado donde se negocian una variedad de instrumentos financieros. Los inversores se dedican a comprar y vender estos instrumentos, pudiendo ganar dinero si el precio de estos sube o perder dinero si el precio baja, con lo que podemos afirmar que la inversión implica riesgos y por lo tanto es importante tanto investigar antes de invertir como mantener posteriormente un control sobre las operaciones realizadas. Para mantener ese control posterior, en este trabajo de fin de grado se desarrolla una aplicación con el lenguaje de programación Python que permite trasladar diariamente de manera automática las operaciones que realiza un inversor mediante un bróker online a una plataforma de gestión de portfolios, la cual permite crear carteras personalizadas según el objetivo de análisis del usuario. En este gestor de portfolios creamos tres carteras, donde en cada una de ellas vamos a controlar las posiciones abiertas, las posiciones de venta en largo y las posiciones de venta en corto respectivamente, a partir de la creación de tres algoritmos que permiten insertar las operaciones realizadas diariamente en cada una de las tres carteras de manera adecuada. Con esto se consigue no solo tener un control de las acciones que se poseen, sino también de las que se vendieron en un pasado para poder recomprarlas en el momento adecuado si fuera oportuno.
Como complemento y para ayudar a los inversores a realizar transacciones de forma sencilla se crea un chatbot capaz de ejecutar operaciones de compra y venta a partir de consultas en lenguaje natural.

Palabras clave: Big Data, Bolsa de valores, Python, Inteligencia Artificial

Resum

La borsa de valors és un mercat on es negocien una varietat d'instruments financers. Els inversors es dediquen a comprar i vendre aquests instruments, podent guanyar diners si el preu d'aquests puja o perdre diners si el preu baixa, amb el que podem afirmar que la inversió implica riscos i per tant és important tant investigar abans d'invertir com mantindre posteriorment un control sobre les operacions realitzades. Per a mantindre aqueix control posterior, en aquest treball de fi de grau es desenvolupa una aplicació amb el llenguatge de programació Python que permet traslladar diàriament de manera automàtica les operacions que realitza un inversor mitjançant un bróker en línia a una plataforma de gestió de portfolios, la qual permet crear carteres personalitzades segons l'objectiu d'anàlisi de l'usuari. En aquest gestor de portfolios creem tres carteres, on en cadascuna d'elles controlarem les posicions obertes, les posicions de venda en llarg i les posicions de venda en curt respectivament, a partir de la creació de tres algorismes que permeten inserir les operacions realitzades diàriament en cadascuna de les tres carteres de manera adequada. Amb això s'aconsegueix no sols tindre un control de les accions que es posseeixen, sinó també de les que es van vendre en un passat per a poder recomprarles en el moment adequat si fora oportú.
Com a complement i per a ajudar els inversors a realitzar transaccions de manera senzilla es crea un chatbot capaç d'executar operacions de compra i venda a partir de consultes en llenguatge natural.

Paraules clau: Big Data, Borsa de valors , Python, Intel·ligència Artificial

Abstract

The stock exchange is a market where a variety of financial instruments are traded. Investors are engaged in buying and selling these instruments, and can make money if

the price rises or lose money if the price falls, so we can say that investment involves risks and therefore it is important both to investigate before investing and subsequently maintain control over the operations performed. In order to maintain this subsequent control, in this thesis we develop an application with the Python programming language that allows to automatically transfer daily operations performed by an investor through an online broker to a portfolio management platform, which allows to create customized portfolios according to the user's analysis objective. In this portfolio manager we create three portfolios, where in each one of them we will control the open positions, the long sell positions and the short sell positions respectively, from the creation of three algorithms that allow to insert the daily operations in each one of the three portfolios in an appropriate way. This allows not only to have control of the shares owned, but also of those sold in the past in order to be able to repurchase them at the right time if appropriate.
As a complement and to help investors to make transactions in a simple way, a chatbot capable of executing purchase and sale operations based on natural language queries is created.

Key words: Big Data, Stock market, Python, Artificial Intelligence

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

La bolsa de valores es un mercado donde se compran y venden acciones y otros instrumentos financieros, como bonos o fondos de inversión. Las acciones representan una parte de propiedad en una empresa, lo que significa que si la empresa tiene éxito, el valor de la acción puede aumentar y los inversores pueden obtener ganancias al vender sus acciones a un precio más alto que el que pagaron por ellas. Por otro lado, si la empresa no tiene éxito, el valor de la acción puede disminuir y los inversores pueden perder dinero [1][2][3]. Los inversores pueden invertir en bolsa para hacer crecer su dinero, pero es importante tener en cuenta que la inversión en la bolsa conlleva riesgos, por ello es importante hacer una investigación cuidadosa antes de invertir y a su vez llevar cierto control sobre las operaciones de compraventa que ejecutan.

Para llevar el control de las operaciones que un usuario realiza, creamos una aplicación con el lenguaje de progrmación Python [4] que nos permite monitorizarlas diariamente y poder tomar decisiones a partir del análisis de los valores de estas. Para llevar a cabo esta monitorización de compras y ventas vamos a hacer uso de dos plataformas.
Por una parte necesitamos un bróker [5], en nuestro caso Interactive Brokers [6], que es una entidad financiera autorizada y regulada que ejecuta las órdenes de los inversores en los mercados.
Por otra parte, necesitamos una plataforma de gestión de portfolios, en nuestro caso investing.com [7], que nos permita crear carteras donde anotar posiciones según nuestros objetivos de análisis para poder mantenerlas controladas.

En investing.com tendremos tres cartera con objetivos de análisis diferente donde anotaremos las operaciones de la forma en que corresponda. Estas carteras son:

- Cartera 1. Es la única cartera donde tenemos información real de la cuenta de trading del usuario, las posiciones que el usuario tiene actualmente abiertas.
  El inversor puede tener controlado el estado y los beneficios que está obteniendo de las acciones que tiene actualmente en su posesión.

- Cartera 2: Cartera ficticia, orientada a analizar operaciones en largo, consistentes en que el inversor compra un activo con la expectativa de que su precio aumente en el futuro para obtener ganancias.
  Para comprender el valor de esta cartera, hay que tener en cuenta que los inversores, suelen tener como objetivo mantener un cierto número de acciones de cada activo.
  Por lo tanto, aquí se anotan la cantidad vendida de cada activo financiero para hacer un seguimiento de esas cantidades, que nos gustaría volver a comprar en algún momento, y así decidir el mejor momento para hacer la recompra, donde se obtengan mayores beneficios.

Por ejemplo, el inversor vende unas acciones a un determinado precio, pero quiere comprar de nuevo en un futuro, entonces necesitan tener anotado a qué precio las vendió, para volver a comprarlas cuando el precio disminuya.

- Cartera 3: Cartera ficticia, orientada a analizar operaciones en corto, consistentes en pedir acciones prestadas para venderlas al precio actual con el compromiso de comprarlas más adelante para devolverlas. Lo que quiere el inversor, es recomprar esas acciones que tiene que devolver a un precio menor para obtener ganancias.
  En esta cartera se anotan esas ventas en corto, para seguir la evolución de precios de esas acciones y saber cuando es un buen momento de volver a comprarlas a un precio menor.

El funcionamiento básico de la aplicación consiste en la conexión con la API de Interactive Brokers [8], para descargar las operaciones que realiza el usuario. Estos movimientos se trasladan a cada una de las carteras de investing.com, mediante tres algoritmos creados, haciendo uso de la librería selenium [9] que nos permite interacturar con la página web.

En último lugar y aunque no forma parte del núcleo central del proyecto, aprovechando el auge y la gran ayuda que propociona la inteligencia artificial, se ha desarollado un chatbot que le permite al usuario realizar operaciones de compra-venta en su cuenta de Interactive Brokers mediante consultas en lenguaje natural. Esta funcionalidad añade una capa adicional de interacción y facilita la realización de transacciones de manera más intuitiva.
Haciendo uso de la herramienta de ChatCompletion de OpenAI [10] [11], el chatbot es capaz de comprender las consultas expresadas en lenguaje cotidiano y así generar y ejecutar el código Python, que hace uso de las funcionalidades que proporciona la API de Interactive Brokers, para realizar la operación deseada.

Todo el código desarrollado se pone a disposición de cualquier usuario que quiera utilizar las funcionalidades creadas mediante un repositorio público de GitHub [12].

1.1 Motivación

La tecnología y la Ciencia de Datos tienen el potencial de revolucionar la forma en que los inversores interactúan con el mercado de valores. Por un lado, la aplicación de monitorización que se desarrollada a lo largo de este documento les permitirá tener un mayor control sobre sus operaciones de compraventa, ahorrando tiempo y esfuerzo en el seguimiento de las mismas. Por otro lado, el chatbot es un gran avance para realziar operaciones de forma mucho más sencilla y directa.
Todo esto es de ayuda para todos los inversores, pero quizá especialmente para inversores sin mucha experiencia, a la hora de extraer información valiosa para la toma de decisiones.
Este proyecto se enmarca en el avance y actualización de un mercado que lleva tantos años existiendo y que necesita de la tecnología para seguir evolucionando.

Desde un enfoque personal, desde antes de empezar mis estudios universitarios, siempre me ha llamado la atención el mundo de la economía y sus diferentes ramas, especialmente la bolsa de valores, aunque nunca he llegado a hacer un estudio profundo y teórico sobre el tema. A lo largo de mis estudios en Ciencia de Datos he tenido la oportunidad de recibir varios cursos relacionados con la economía, empresa o finanzas, los cuales despertaron gran interés en mí y me ayudaron a saber de qué formas puedo aportar valor a la economía con mis conocimientos tecnológicos. Además, en el último

año, y especialmente a raíz de mis prácticas de empresa, he desarrollado un gran interés por el campo de la inteligencia artificial, teniendo la oportunidad actualmente de poder aplicarla en el mundo de las finanzas y así transformar la forma en que los inversores interactúan con el mercado.
Siempre he sido consciente de que el seguimiento y control de las operaciones de compra-venta es una tarea importante y necesaria para cualquier inversor, por lo que me resulta emocionante y desafiante contribuir de manera significativa al avance y actualización de la bolsa de valores creando una aplicación que permita a los inversores llevar este control de sus operaciones y contribuir a mejorar su experiencia.

1.2 Objetivos

A continuación se enumeran la serie de objetivos que se pretende conseguir con la realización del proyecto:

1. Desarrollar una aplicación utilizando el lenguaje de programación Python que permita la monitorización de operaciones en el mercado de valores, realizando la migración de datos de las operaciones realizadas mediante el broker online a un gestor de portfolios.
2. Permitir la simulación de estrategias de operación en largo y en corto, lo que incluye el análisis de la cantidad vendida y la posibilidad de recompra en el futuro a un precio más bajo.
3. Ejecución automática diaria de la aplicación, asegurando que la información en el gestor de portfolios esté actualizada y funcione de manera continua.
4. Facilitar la realización de inversiones, sobre todo a inversores principiantes, con el desarrollar un chatbot basado en inteligencia artificial que permita a los usuarios realizar operaciones de compra-venta en su cuenta de Interactive Brokers mediante consultas en lenguaje natural.
5. Garantizar que la aplicación sea accesible para terceros, proporcionando acceso a datos y código en repositorios abiertos o internos.

1.3 Impacto esperado

La aplicación resultante de este trabajo proporcionará una solución eficiente y accesible para el seguimiento y análisis de las inversiones en el mercado de valores. Los usuarios se beneficiarán de una mayor comprensión, control y optimización de sus operaciones financieras, lo que les permitirá tomar decisiones más informadas y alcanzar sus objetivos de inversión de manera efectiva.

Se han identificado diferentes usuarios y para cada uno de ellos vamos a explicar el impacto de forma más concreta:

1. Inversionistas:

    - Mayor control y seguimiento de sus operaciones en el mercado de valores.
    - Toma de decisiones informadas gracias a la gran cantiadad de noticias financieras y gráficos generados.

- Análisis según diferentes estrategias de inversión, como operar en largo o en corto.
- Maximización de beneficios.

2. Empresas de gestión financiera:
   - Permite monitorear las operaciones de sus clientes y brindarles un servicio personalizado.
   - Mejora en la evaluación del rendimiento de los activos.
3. Investigadores:
   - Acceso a datos históricos y en tiempo real para realizar análisis y estudios sobre el mercado de valores.
   - Posibilidad de desarrollar nuevas estrategias de inversión y probar su efectividad utilizando datos reales.
4. Usuarios sin experiencia en bolsa:
   - Familiarización con conceptos clave, practicar estrategias de inversión y mejorar sus conocimientos en el mercado de valores.
   - Facilidad para la ejecución de operaciones.

1.4 Metodología

La metodología propuesta para el desarrollo de este Trabajo de Fin de Grado se basa en un enfoque ágil e iterativo [13], el cual ha sido seleccionado debido a su capacidad para adapartar la forma de trabajo a cambios durante desarrollo de aplicaciones y la implementación de técnicas de inteligencia artificial.

La metodología se estructura en diversas etapas y se realiza de manera iterativa, lo que permite una mayor flexibilidad y ajuste continuo. A continuación, se describen las diferentes etapas:

1. **Investigación inicial**. Investigación sobre el mercado de valores, los conceptos financieros relevantes y las estrategias de inversión para comprender los desafíos y oportunidades que enfrentan los inversores.
2. **Funcionalidades a implementar**. A partir del análisis de las necesidades de los usuarios, se establecen los objetivos y se definen las funcionalidades necesarias para cumplirlos.
3. **Tecnologías para el desarrollo**. Establecimiento de las tecnologías necesarias para interactuar con las plataformas empleadas. Además, se planificará la implementación de la funcionalidad de simulación de estrategias de inversión en largo y corto.
4. **Desarrollo e implementación**. Implementación de los programas necesarios para la monitorización y el chatbot. Desarrollo teórico e implementación en Python de los algoritmos necesarios para crear los programas, incluyendo el desarrollo de los módulos y componentes requeridos para el correcto funcionamiento y cumplimiento de los requisitos.
5. **Pruebas y evaluación**. Pruebas para verificar que todas las funcionalidades se implementen correctamente.

6. **Mejora continua**. A medida que se avance en el desarrollo, se realizarán iteraciones para detectar posibles errores y se realizarán ajustes según sea necesario.

7. **Documentación y redacción de la memoria**. Durante todo el proceso de desarrollo, se documentarán cada una de las etapas mediante el uso de un trabelo de Trello. De esta forma, al finalizar todo el desarrollo, se realizará la redacción de la memoria explicando los pasos seguidos, siguiendo la estructura de redacción propuesta por la Escuela de Informática [14].

1.5 Estructura de la memoria

El desarrollo de este trabajo de fin de grado comienza en el capítulo actual con una introducción sobre su desarrollo, indicando la pretensión de realizar principalmente una aplicación para la monitorización de operaciones en la bolsa de valores y por otra parte, como complemento, un chatbot capaz de ejecutar transacciones.
Se indica también tanto la motivación para su realización como los objetivos específicos que se pretenden alcanzar.

A continuación, en los Capítulos 2 y 3, se comentan los trabajos existentes con finalidades similares al actual para así hacer un análisis del problema y concretar el espacio de conocimiento que vamos a llenar.

Como paso previo al desarrollo del trabajo en sí, se comenta el contexto teórico en el Capítulo 4, explicando de qué trata la bolsa de valores y la monitorización.

Para explicar la realización de la aplicación de monitorización de operaciones se comienza por comentar en el Capítulo 5 las plataformas de bróker y gestor de portfolios elegidas entre todas las disponibles, explicando la estructura y lo que nos ofrece cada una.
Una vez elegidas las plataformas con las que se trabajará, se concretan todas las herramientas tecnologías empleadas para poder interactuar con ellas en el Capítulo 6.
En el Capítulo 7 se propone el esquema que se va a seguir para conectar ambas plataformas y realizar la descarga y carga de las operaciones.
En los siguientes capítulos se explica con detalle los programas necesarios que se deben desarrollar y cómo se realizan. En primer lugar, el Capítulo 8 habla de la descarga de la información del bróker, y a continuación, el Capítulo 9 habla de cómo introducir las operaciones al gestor de portfolios con las funciones adecuadas.

En el Capítulo 10 se explica el desarrollo del Chatbot, comentando tanto la herramienta usada para tratar el lenguaje natural como la contrucción de la aplicación web y su interfaz para su uso.

Una vez finalizada la explicación de los dos programas realizados, en el Capítulo 11 se detallan los pasos a seguir por cualquier usuario que quiera hacer uso de ellos, a partir del código presente en un repositorio público de GitHub.

El trabajo finaliza con las conlusiones en el Capítulo 12 y trabajos que se plantean realizar en un futuro en el Capítulo 13.

Este documento también cuenta con anexos que el lector puede consultar en el caso de necesitar ampliar la información. El Anexo A cuenta con la explicaciones en detalle sobre la generación del código para tratar con el gestor de portfolios, el Anexo B es una introducción de la API del bróker online y finalmente en el Anexo C se comentan los Objetivos de Desarrollo Sostenible de la Agenda 2023 que guardan realción con el trabajo de final de grado.

CAPÍTULO 2
Estado del arte

En el presente capítulo, se explorarán y analizarán los trabajos y artículos existentes que han sido desarrollados previamente en el campo temático del proyecto relacionados con el TFG, ampliando así nuestro conocimiento sobre el campo de estudio y preparando el terreno para el desarrollo del proyecto propuesto.

Una de las necesidades por las que surje el interés en la monitorización de las operaciones realizadas es por la rapized con la que ocurren los cambios en la bolsa. Esta necesidad de que los inversores puedan monitorear sus inversiones de manera rápida se trata en el artículo [15], donde se presenta P-Stock, una aplicación diseñada para monitorear las actividades que ocurren en el mercado de valores, la cual rastrea las actividades del mercado de valores y utiliza los datos recopilados para calcular los valores de las acciones en la cartera de un inversionista, todo en tiempo real.
Gracias a algortimos de machine learning se pueden crear directamente portfolios diversificados con alta probabilidad de generar beneficios. Esto es lo que ocurre en en artículo 'Clustering Indian stock market data for portfolio management' [16], donde se usa el Clustering par la clasificación de acciones en grupos (clusters) y a partir de ahí construir portfolios diversificados.

Por otra parte, es de gran importancia para realizar un completo análisis previo del mercado y así poder hacer una mejor gestión de las posiciones, desarrollar modelos predictivos que a partir de datos históricos del mercado nos ayuden a hacer una predicción del precio futuro de los activos, como ocurre en los artículos 'Predicting Stock Market Trends by Recurrent Deep Neural Networks' [17] o 'Adaptive Stock Market Portfolio Management and Stock Prices Prediction Platform for Colombo Stock Exchange of Sri Lanka' [18].

Centrándonos en otra área, en el mundo de las finanzas está teniendo un crecimiento elevado el uso de inteligencia artificial y chatbots para ofrecer ayuda a usuarios [19], siendo un ejemplo de esto el Bank of America [20], el cual ha desarrollado un asistente virtual que ayuda a los clientes con información sobre saldos de cuentas, detalles de transacciones e incluso asesoramiento financiero. También ha sido implementado por otros bancos estadounidenses como JP Morgan Chase [21] que tiene un chatbot dedicado a revisar y extraer información de documentos legales o Goldam Sachs [22] cuyo chatbot ofrece asesoramiento de inversión personalizado basado en los datos, preferencias y perfiles de riesgo de los clientes.

Como vemos, el procesamiento del lenguaje natural (NLP) abre un gran abanico de posibilidades para la creación de aplicaciones dentro de las finanzas, muchas de ellas

abordadas en el artículo 'Natural Language Processing in Accounting, Auditing and Finance: A Synthesis of the Literature with a Roadmap for Future Research' [23].

A parte del procesamiento de lenguaje natural, encontramos otras áreas de la inteligencia artificial que se pueden aplicar al análsis de la bolsa de valores, como puede ser el reconocimiento de voz, área usada en el trabajo desarrollado en el artículo 'An Intelligent Stock Market Automation with Conversational Web Based Build Operate Transfer (BOT)' [24] cuyo objetivo es construir un chatbot de reconocimiento de voz para predecir los precios futuros intra día de las acciones.

2.1 Crítica al estado del arte

En la actual sección se analizan y evalúan los trabajos previamente presentados en la ETSINF (Escuela Técnica Superior de Ingeniería Informática) de la Universidad Politécnica de Valencia y tiene como objetivo identificar posibles fallos o aspectos no abordados que justifiquen la necesidad de llevar a cabo el presente Trabajo de Fin de Grado (TFG).

Comenzamos por comentar los trabajos encontrados que están relacionados con el análisis de operaciones financieras o gestión de operaciones y portfolios. En primer lugar, siendo la idea más cercana a nuestro proyecto, tenemos 'Diseño de un portal web de gestión de carteras de acciones' [25], cuyo objetivo es la creación de una aplicación Web que permitira al usuario simular carteras o fondos de inversión y crear operaciones bursátiles con títulos que cotizan en el Ibex 35, es decir, la creación de un gestor de portfolios.

Por otra parte tenemos el trabajo 'Virtual Stock Trading' [26], donde se realiza la creación de un software web donde los usuarios se registran y se realiza una simulación de negociación de acciones. Aunque se puedan realizar operaciones de compra-venta, estas transacciones son ficticias, teniendo como principal objetivo servir para la comprensión del funcionamiento del mercado de valores.

Por último, como herramientas que ayudan a hacer un análisis de operaciones, tenemos los proyectos 'Predicting stock prices using long short-term memory models' [27] y 'Forecasting exchange rates using recurrent neural networks' [28] los cuales usan distintos algoritmos para predecir el valor futuro de distintos activos financieros a partir de datos históricos.
Estas técnicas son un proceso previo a la realización de operaciones, las cuales podrían ser perfectas herramientas complementarias para nuestro proyecto.

Continuamos con los trabajos que pueden tener relación con la otra herramienta desarrollada en el presente TFG, el chatbot capaz de ejecutar operaciones a partir de lenguaje natural y generar el correspondiente código Python.
Los trabajos en los que podemos encontrar la implementación de un chatbot son 'Desarrollo de un chatbot para la recomendación de eventos o lugares de interés' [29] y 'Desarrollo de un chatbot para la resolución de dudas sobre devonfw' [30]. Sin embargo, ningo de los chatbots creados mantiene ningún tipo de relación con el mundo de las finanzas ni son capaces de generar scripts de código a partir del lenguaje natural.

2.2 Propuesta

Una vez comentados todos los trabajos existentes que guardan relación con el actual proyecto, ya podemos concretar cuál es espacio de conocimiento que se pretende llenar con este trabajo.

La idea principal que se pretende alcanzar en este proyecto no es solo la creación de carteras con distintos objetivos de análisis, sino la integración en un mismo programa de las cuentas del usuario de la plataforma del bróker donde ejecuta sus operaciones con la del gestor de portfolios. De esta forma, el usuario no tendrá que introducir las transacciones que vaya realizando de forma manual a las carteras del gestor, sino que se crean unos algoritmos que descargan las operaciones hechas cada día y las introducen de la forma adecuada a cada cartera de seguimiento, todo esto de forma automática cada día, la cual es una funcionalidad muy destacable.

Además, a diferencia de otros trabajos previos , en lugar de crear una web de gestor de portfolios, utilizamos una plataforma ya existente hacer los seguimientos, aprovechándonos de sus distintas funcionalidades como la actualización de precios en tiempo real, la generación de gráficas y las noticias financieras relacionadas con las posiciones.

Por otra parte, se pretende avanzar en la implementación y usos de la inteligencia artificial en las finanzas. Queremos ir más allá de una simple asistencia teórica para el inversor (como podría ser la consulta de precios o estados de activos), pretendiendo llegar al punto de poder ejecutar operaciones.
Para ello, se integra el bróker online con herramientas de procesamiento de lenguaje natural, consiguiendo que una transacción se realice en la cuenta del usuario en el bróker simplemente a partir de una orden en forma de texto, además de proporcionarle al usuario el código de programación completo con el que se ha ejecutado dicha transacción.

CAPÍTULO 3
Análisis del problema

La problemática principal abordada en este proyecto es la necesidad de los inversores de controlar y monitorizar sus operaciones en el mercado de valores. La bolsa de valores es un entorno dinámico y complejo en el que las transacciones ocurren rápidamente, lo que dificulta el seguimiento manual de las operaciones y el análisis de su desempeño.

Además, los inversores se enfrentan a riesgos inherentes a la inversión en la bolsa de valores. El valor de las acciones puede fluctuar considerablemente, lo que puede resultar en ganancias o pérdidas para los inversores. Por lo tanto, es crucial que los inversores realicen un seguimiento cuidadoso de sus operaciones, tanto de las posiciones abiertas como de las cerradas, y tomen decisiones informadas para minimizar los riesgos y maximizar los beneficios.

Por otra parte, para los inversores principiantes, puede resultar abrumador comprender y seguir el funcionamiento de la bolsa de valores, así como realizar transacciones de manera eficiente y efectiva. Se necesita una solución que simplifique el proceso de inversión y proporcione orientación a los inversores en la toma de decisiones.

En las siguientes secciones vamos a comentar más en detalle los espectos de eficiencia y legalidad.

3.1 Análisis energético o de eficiencia algorítmica

El programa debe descargar datos de una plataforma, procesarlos adecuadamente y volver a cargarlos en otra plataforma. Por todo ello es inevitable que el proceso conlleve tiempo, pero para garantizar una experiencia fluida a los usuarios se analiza el código creado para poder optimizarlo todo lo posible y reducir el consumo de recursos.

3.2 Análisis del marco legal y ético

Este trabajo accede y procesa datos para extraer conocimiento a partir de ellos. Tanto la descarga como la carga de datos se realiza a partir la cuenta del usuaria en distintas plataformas, que únicamente se realiza si el usuario proporciona sus credenciales de manera voluntaria, para que con esta autorización sus datos puedan ser accedidos y manipulados.

En relación al uso de la inteligencia artificial, es importante destacar que su implementación en el contexto de inversiones financieras no viola ninguna ética. Su utilización

se ha llevado a cabo con el propósito de optimizar y agilizar los procesos financieros, sin comprometer los principios éticos fundamentales.

3.3 Solución propuesta

El desarrollo de una aplicación que permita la monitorización de operaciones en el mercado de valores y facilite la gestión de carteras, integrando la conexión de un bróker junto con una plataforma de gestión de portfolios, ofrece una oportunidad para abordar las limitaciones mencionadas y mejorar la experiencia de inversión de los usuarios.

Una solución automatizada permitiría el seguimiento en tiempo real de las operaciones, la actualización automática de los precios de los activos y el análisis de rendimiento de las inversiones. Asimismo, las plataformas de gestión de portfolios actuales ofrecen herramientas de análisis avanzadas, como gráficos y métricas de rendimiento, para facilitar la toma de decisiones informadas.

Además del control de las posciones abiertas en cada momento, desarrollar algoritmos que permitan mantener la información de las posiciones cerradas también controladas en carteras, permite que inversor tenga una mejor visión de todas las estrategias que puede seguir según sus intereses o lo que sea más beneficioso en cada momento temporal.

La integración de un chatbot basado en inteligencia artificial proporcionaría una interfaz intuitiva y accesible para los inversores, permitiéndoles realizar transacciones de manera más eficiente.

En resumen, existe una oportunidad para desarrollar una solución que aborde las limitaciones actuales en el monitoreo y control de operaciones financieras en la bolsa de valores. Al proporcionar automatización, análisis avanzado e interacción intuitiva, esta solución tiene el potencial de mejorar la experiencia de inversión y ayudar a los inversores a tomar decisiones más informadas y rentables.

3.4 Plan de trabajo

El desarrollo del presente Trabajo de Fin de Grado se lleva a cabo siguiendo un plan de trabajo estructurado y organizado. Para gestionar las tareas y el progreso del proyecto, utilizamos el software Trello, una herramienta de gestión de proyectos basada en tableros [31].

En Trello, creamos un proyecto específico para el TFG, donde establecemos diferentes tableros para abordar las distintas áreas de trabajo. Estos tableros incluyen:

- Memoria: Este tablero se centra en las tareas relacionadas con la elaboración de la memoria del proyecto. Aquí registramos las actividades necesarias para redactar y revisar cada sección de la memoria, incluyendo los capítulos que consideramos necesarios incluir a medida que hacemos nuevas implementaciones y pautas a seguir en la redacción.

- Investing: En este tablero llevamos a cabo las tareas relacionadas con la gestión de portfolios utilizando la plataforma investing.com. Mantenemos apuntada toda la documentación que hemos seguido, como los algoritmos desarrollados para crear la aplicación que migra los datos.

- IBKR: Este tablero se enfoca en las tareas relacionadas con la conexión y uso de la API de Interactive Brokers. Aquí también registramos toda la documentación descargada, las distintas partes código necesarios para descargar operaciones o pasos para descargar los software necesarios, entre otros.
- Ideas: En este tablero registramos todas las ideas y propuestas, como nuevas funcionalidades, mejoras o cambios que surjan a lo largo del desarrollo del TFG.
- Dudas: Se apunta las preguntas o dudas que van surgiendo mientras avanzamos, para poder resolverlas durante las sesiones de tutoría.

Cada tarea en Trello se marca con un estado que indica su progreso, como 'por hacer', 'en proceso' o 'hecho'. Además, asignamos prioridades a las tareas para establecer su importancia relativa en el desarrollo del proyecto. En el caso de tareas extensas, indicamos una estimación aproximada del tiempo que estimamos necesario para completarlas.

Además, dentro de cada tarea podemos añadir comentarios que proporcionan información relevante sobre la tarea en cuestión. Esto puede incluir referencias a fuentes de información utilizadas para resolver la tarea o cualquier otro detalle que sea útil para el desarrollo del proyecto.

El plan de trabajo se realiza de manera iterativa, permitiendo ajustes y modificaciones a medida que avanzamos en el proyecto, para así conseguir llevar un seguimiento constante de las tareas y corregir problemas a tiempo.

CAPÍTULO 4
La bolsa de valores y la monitorización

4.1 La bolsa de valores

4.1.1. ¿Qué es?

La bolsa de valores es un mercado es un mercado físico o virtual [32] donde se compran y venden tanto renta variable (acciones) como renta fija (deuda) y otros instrumentos de inversión, siendo el caso de las acciones el más conocido.
Una acción es un tipo de valor financiero que representa la propiedad de una pequeña parte de una empresa y los inversores que compran estas acciones se convierten en propietarios parciales de la empresa [3].
Otras formas de invertir en la bolsa de valores es a través de los fondos de inversión — formados por un grupo de inversores que aportan su dinero en un fondo común — o también los bonos, que son deuda emitida por empresas o gobiernos, y los futuros y opciones, que son contratos que otorgan el derecho a comprar o vender un activo en el futuro a un precio determinado [33].

La bolsa pone en contacto empresas que buscan financiación, por una parte, y ahorradores (particulares u organizaciones) que quieren invertir su dinero en busca de una rentabilidad.
Estas compañías tienen dos motivos principales para salir a bolsa: obtener capital para poder impulsar su crecimiento y generar liquidez para sus accionistas privados, además de ganar prestigio y credibilidad.
Una vez que se ha adquirido ese trocito minoritario de una empresa hay dos formas de que el comprador obtenga beneficios: [2]

- Reparto de dividendos. Algunas empresas reparten sus beneficios (si los hay) a los accionistas de manera proporcional a su participación.
- Mayor valor de la participación. La empresa puede aumentar su valor en bolsa, de manera que hay más gente que quiere comprar esas acciones, que ahora son más caras, y los beneficios de venderlas son mayores.

4.1.2. Elementos básicos

Después de haber hecho un explicación sencilla en el apartado anterior sobre qué es la bolsa y su funcionamiento básico, podemos proceder a ver en detalle los elementos básicos necesarios para realizar operaciones en el mercado [2].

Instrumentos financieros

En la bolsa se compran y venden diversos tipos de valores, que son instrumentos financieros que representan derechos de propiedad, de crédito o de participación en una empresa o en una entidad.
Existen una gran cantidad de productos financieros, incluyendo una variedad de derivados. Sin embargo, para obtener una visión general y simplificada del mercado, nos enfocaremos en aquellos productos que son más comúnmente utilizados. [33]

- **Acciones**. Son el producto estrella del mercado bursátil y se consideran la opción por excelencia para aquellos que desean iniciarse en este ámbito, debido a su sencillez de comprensión y análisis. Además, es el producto más accesible a través de cualquier bróker, gestoras o banco.
  Las acciones son unidades de propiedad en una o más compañías. El titular de acciones, denominado accionista, tiene el derecho a obtener una parte de los beneficios de la compañía en caso de que se paguen dividendos, así como derechos de voto. Estas acciones se puede comprar y vender libremente en el mercado bursátil
  Las diversas Bolsas de valores facilitan el intercambio de acciones que cotizan de forma pública.[3]

- **Bonos**. Los utilizan los Gobiernos y las compañías para obtener dinero mediante préstamos de inversionistas. Por lo general, los bonos se emiten con el fin de recaudar fondos para proyectos específicos. A cambio, el emisor del bono se compromete a devolver la inversión, con intereses, durante un período determinado.[34]

- **Fondos de inversión**. Un fondo de inversión es un instrumento de ahorro en el que los partícipes entregan su dinero para que un equipo de gestores profesionales lo inviertan en una serie de activos financieros, empleando sus conocimiento y experiencia en los mercados [35].

- **Futuros**. Contrato en el que dos partes acuerdan comprar o vender un activo subyacente (como una materia prima, una divisa o un índice bursátil) a un precio determinado en una fecha futura acordada de antemano.

- **Opciones**. Contrato que otorga al comprador el derecho, pero no la obligación, de comprar o vender un activo subyacente a un precio determinado en una fecha determinada o antes de ella.

- **Forex**. Es el mercado global descentralizado en el que se negocian las monedas de diferentes países. Los inversores pueden ganar dinero en el Forex al especular sobre el movimiento de los tipos de cambio entre las distintas monedas.

Trader

Es cualquier individuo o empresa que opera en los mercados financieros por su propia cuenta, es decir, el vendedor o comprador de activos financieros (acciones, bonos, divisas, opciones, etc.). Un trader compra y vende mientras controla en tiempo real las fluctuaciones de los precios del mercado de valores, siendo el principal objetivo la generación de ganancias. [36]

El trader lanza órdenes de compra o venta en el mercado, pero no tiene acceso gratuito a estos mercados, necesita un intermediario para dárselo y ejecutar sus órdenes. Este es el bróker. Por lo que la relación entre el trader y el bróker es la siguiente: [5]

- El trader es el cliente del bróker. Él toma las decisiones de inversión.
- El trader lanza las órdenes en el mercado, el bróker las ejecuta.
- El trader recoge las ganancias o asume la pérdida de las órdenes lanzadas en los mercados. El bróker recibe su comisión, sin importar el resultado de la orden ejecutada.
- El trader toma sus decisiones de trading en función del análisis técnico o fundamental que realiza y su estilo.

Los traders se suelen definir en función del tiempo que mantengan sus operaciones abiertas: [36]

- Traders a largo plazo: posiciones abiertas durante años.
- Swing traders: posiciones abiertas durante días y semanas.
- Day traders: abren posiciones dentro del mismo día, evitando mantener posiciones abiertas por períodos prolongados debido al alto riesgo.
- Scalpers: operan a a muy corto plazo, generalmente usan un marco temporal de quince minutos.

Por lo tanto, para ser trader solo se necesita lo siguiente: conexión a internet, un ordenador, un bróker y una cuenta en una plataforma de trading.

Bróker

Un bróker es una entidad financiera autorizada y regulada, cuya remuneración consiste en comisiones, que ejecuta las órdenes de los traders en los mercados, ofreciendo a sus clientes un amplio acceso a los mercados financieros de todo el mundo. Por lo tanto, actúa como intermediario y facilitador entre un vendedor y un comprador, siendo responsable de la seguridad de los fondos de sus clientes. [5]

Además de lo anterior, el bóker proporciona al trader tanto la plataforma de trading — donde coloca en los mercados las órdenes de compra y venta de los clientes — para operar como una gran colección de herramientas de análisis, historial de precios, gráficos, indicadores de análisis etc.

Hace unos años los bancos tenían prácticamente el monopolio de servicios de este tipo pero, nacieron los "brokers online" ofreciendo la posibilidad de invertir sin límites ni fronteras, además de reclamar comisiones mucho más asequibles.
Por lo tanto, actualmente, los brokers suelen ser empresas en forma de plataformas online en las que te puedes abrir una cuenta directamente y comenzar a operar en bolsa. [37]

4.2 La monitorización de la bolsa de valores

4.2.1. ¿Para qué sirve?

Como ya se ha comentado anteriormente, el objetivo del proyecto es la monitorización de operaciones en el mercado de valores.

El objetivo de realizar operaciones con instrumentos financieros en el mercado de valores es generar beneficios económicos. Pero esto no es tan simple como, por ejemplo, comprar una acción de Netflix hoy y venderla mañana. Para ganar dinero con el mercado de valores, antes de realizar cualquier movimiento hay que hacer un estudio detallado de la situación del activo que queremos comprar.
Y no solo antes de realizar la compra, sino cuando ya poseemos el activo, debemos seguir analizando el mercado para saber si nos conviene mantenerlo o es preferible venderlo.
Por último, el estudio de un activo no acaba cuando lo vendemos. Cuando tenemos un acción y la vendemos, es necesario anotar todos los datos de esa venta, para así poder realizar un seguimiento y en el futuro saber si nos interesa volver a comprarla o por el contrario hicimos bien en venderla en su momento y actualmente no nos interesa más. Por ejemplo, el inversor vende unas acciones a un determinado precio, pero quiere comprar de nuevo si bajan, entonces necesitan saber a qué precio vendió y si es rentable comprar de nuevo. Esto es especialmente valioso cuando se ejecutan muchas operaciones al mes sobre los mismos valores.

Hacer un seguimiento de las operaciones en bolsa de la forma comentada anteriormente puede ser muy beneficioso para mejorar resultados de inversión y gracias a una mayor cantidad de información tomar mejores decisiones de inversión. Podemos comentar de forma más concreta algunas de las formas en que puede ayudar este seguimiento detallado al trader:

- Podemos medir el rendimiento de nuestra cartera actual.
- Revisar las operaciones pasadas nos permite identificar patrones en las inversiones que hemos hecho, para así tanto identificar errores que hayamos cometido y ajustar estrategias en el futuro como identificar las operaciones que han dado beneficios y poder repetir esa estrategia.
- Al utilizar una plataforma de gestión de portfolios, vemos todas nuestras inversiones en un solo lugar, lo que te permitirá tener una visión general de tu cartera para tomar las decisiones. Esto es de gran utilidad porque, quizá, el inversor hace operaciones desde distintas plataformas, y lo ideal es ver todos tus movimientos en un solo portfolio.

4.2.2. Elementos básicos para monitorizar las operaciones

A continuación vamos a explicar cómo vamos a realizar este seguimiento y monitorización de operaciones en el mercado de valores.

Para monitorizar las operaciones, en primer lugar, debemos ser capaces de realizar operaciones en el mercado de valores. Nosotros como clientes, como bien se ha explicado en el capítulo anterior, solo podemos realizar esas operaciones de compra y venta mediante un **bróker** que nos proporciona una plataforma para operar.
Por lo tanto, una vez escogido el bróker con el que vamos a trabajar durante este proyecto, podremos empezar a operar en bolsa.

La siguiente parte es la de seguimiento y monitorización de esas operaciones que estamos realizando. Para llevar esto a cabo, trasladaremos la información de todas las operaciones que realizamos a diversas carteras, donde podemos añadir la información que nosotros deseemos. Para esto haremos uso de una **plataforma de gestión de portfolios**, la cual nos va a permitir seguir y gestionar las inversiones en una sola plataforma. Estas plataformas ofrecen una serie de herramientas y funcionalidades para ayudar a los inversores a analizar, gestionar y optimizar sus carteras de inversión.

Existen diversas plataformas de este tipo y cada una va a contar con unas características, pero podemos nombrar algunas herramientas que suelen ser comunes en todas ellas:

- Permiten a los inversores mantener un registro de todas sus inversiones en una sola plataforma.
- Proporcionan herramientas y análisis para ayudar a los inversores a evaluar su cartera, desde el análisis de rendimiento hasta el análisis de riesgo, pasando por la evaluación de las posiciones individuales.
- Posibilidad de configurar alertas para recibir notificaciones cuando se produzcan cambios importantes en el mercado o en alguna de las posiciones de la cartera.
- Permiten evaluar y gestionar los riesgos de su cartera, desde la diversificación hasta el análisis de riesgo.

En el próximo capítulo escogemos con qué platformas vamos a trabajar, para posteriormente poder llevar a cabo los pasos para conseguir la monitorización.

CAPÍTULO 5
Plataformas de compra y gestión

5.1 Opciones de plataformas

Ya sabemos que necesitamos, por una parte un bróker online, y por otra parte un gestor de portfolios. A continuación vamos a enumerar distintas plataformas existentes, comentando sus características y qué nos pueden ofrecer, para después hacer una valoración global y esocoger aquellas con las que realizaremos el proyecto.

5.1.1. Brókers online

Según varias páginas web especializadas en inversiones [38] [39] [40], entre los mejores brókers online encontramos XTB, DEGIRO, Freedom24, Interactive Brokers, eToro o Naga, entre otros.
Sin embargo, lo ideal es que la plataforma escogida tenga API, para facilitar el acceso y descarga de datos. Por lo tanto, vamos a centrarnos en buscar información y escoger directamente entre brókers online que cuenten con API.
Entre algunos de los bróker con API encontramos los siguientes [41] [42]: Alpaca, Interactive Brokers, TD Ameritrade, Tradier, AvaTrade y Freedom24.

5.1.2. Plataformas de análisis financiero y gestión de portfolio

Aunque nuestro objetivo principal sea la creación de carteras, queremos que la plataforma escogida nos pueda ofrecer también una buena variedad de noticias financieras y análisis. Alguns de las plataformas que cuentan con estas carterísticas son: Yahoo Finanzas, Google Finance, investing.com, Morningstar.

Cabe destacar, que ninguna de las plataformas con las características deseadas nos ofrece API.

5.2 Plataformas escogidas para el proyecto y su estructura

De entre las plataformas disponibles, escogemos para el desarrollo de nuestro proyecto como bróker a **Interactive Brokers** y como gestor de portfolios a **investing.com**.

A continuación, por una parte explicamos las razones y comentamos las caterísticas de éstas y por otra parte mostraremos sus estructuras.

5.2.1. Interactive Brokers

Escogemos Interactive Brokers (IBKR), entre otras cosas, por ser el bróker que ofrece API que además se encuntra entre los mejores brókers online en general en varios rankins.

Este corredor de bolsa ofrece tarifas bajas, una plataforma de trading avanzada, acceso global a mercados, una amplia variedad de productos de inversión, servicio al cliente de alta calidad y herramientas de gestión de riesgos avanzadas. [6]
Estas características le ofrecerán al usuario de nuestra aplicación de monitorización una gran experiencia ya que puede realizar gran variedad de operaciones.

IBKR nos ofrece la posibilidad de abrir una cuenta simulada, llamada "Paper Trading Account", en lugar de una real, lo que nos va a permitir practicar y experimentar con la plataforma de trading y los productos de inversión sin arriesgar dinero real.
Este tipo de cuenta proporciona a los usuarios un saldo virtual de $1 millón para que lo utilicen en sus operaciones de trading.

Además, ofrece una amplia gama de recursos educativos y herramientas de formación como vídeos explicativos o tutoriales, como una introducción a la plataforma, herramientas de trading, explicación de cada tipo de activo o análisis de portfolio, siendo el curso que más nos ha servido el del uso de la API, llamado 'Python TWS API' [8], donde te explican mediante una serie de vídeos y desde cero el uso de la API.

Estructura de la plataforma

Para realizar transacciones con IBKR, la empresa nos proporciona diferentes plataformas. Entre estas plataformas encontramos: Trader Workstation (TWS), IBKR Mobile, WebTrader, API de IBKR, Client Portal.
Trader Workstation (TWS) es la plataforma de trading principal de IBKR, con amplia gama de herramientas y funciones avanzadas.
Client Portal está diseñado para ser fácil de usar y proporciona funciones esenciales. La forma de acceder es a través de un navegador web.
Otra forma de realizar operaciones es a través de la *API* que ofrece Interactive Brokers y que admite varios lenguajes de programación.

5.2.2. investing.com

Las cuatro plataformas que comentamos en el apartado anterior son bastante completas y parecidas entre sí. En primer lugar descartamos Google Finance, ya que es la menos completa y, aunque, tanto Yahoo Finance como MorningStar ofrecen mucha información, decidimos trabajar en este proyecto haciendo uso de investing.com ya que, a parte ofrecer más información que las otras dos, nos ha sido recomendada.

Estructura de la plataforma

Es muy importante realizar un análisis detallado de la estructura de la página web de investing.com para saber qué apartados contienen la información que necesitamos y la forma de acceder a ellos, ya que vamos a acceder a ella constantemente para trasladar cada una de las operaciones que hagamos mediante el bróker.

- En la página incial encontramos varios elementos, como un resumen de los principales índices bursátiles o cotizaciones en tiempo real, que podemos ver la Figura 5.1.
  Por otra parte, como vemos en la Figura 5.2, también se muestran noticias sobre eventos económicos importantes programados para el día o la semana o anuncios de bancos centrales.

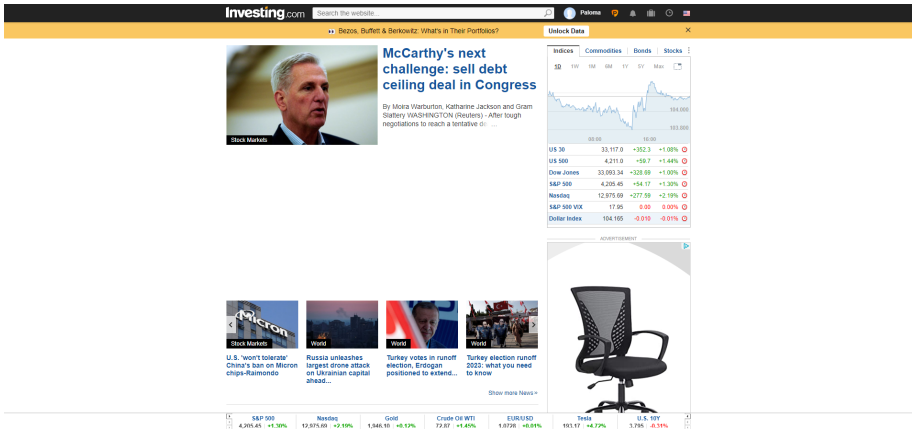


Figura 5.1: Página principal de investing.com - Parte 1

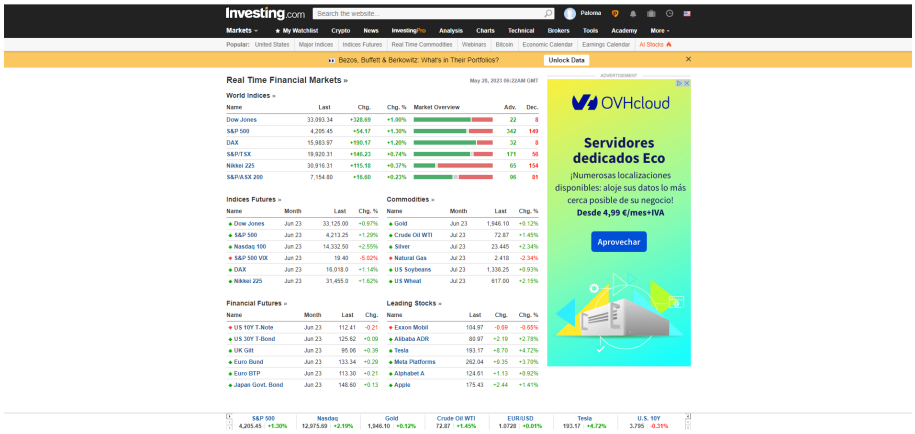


Figura 5.2: Página principal de investing.com - Parte 2

- El cuadro de texto superior nos permite buscar una variedad de elementos relacionados con los mercados financieros y la información económica.
  En la Figura 5.3 se muestra una búsqueda de la empresa META, y en la Figura 5.4 el resultado, que contiene diversas noticias relacionadas con la empresa, así como gráficos, estadísticas y resúmenes.

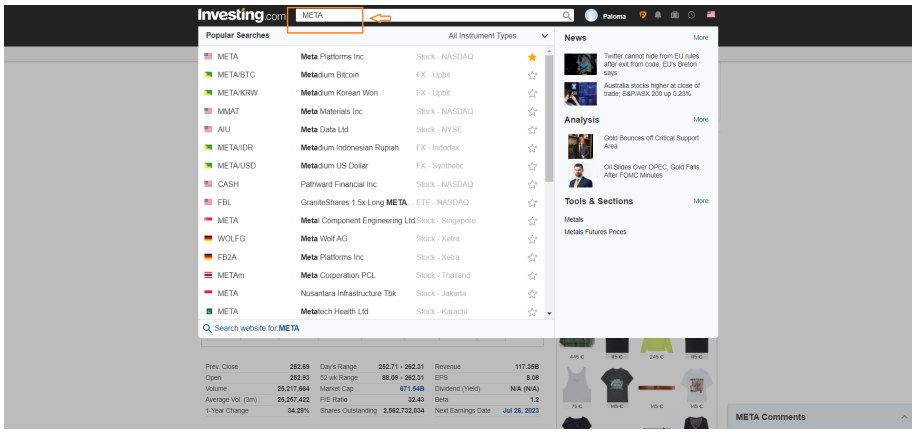


Figura 5.3: Buscador superior investing.com

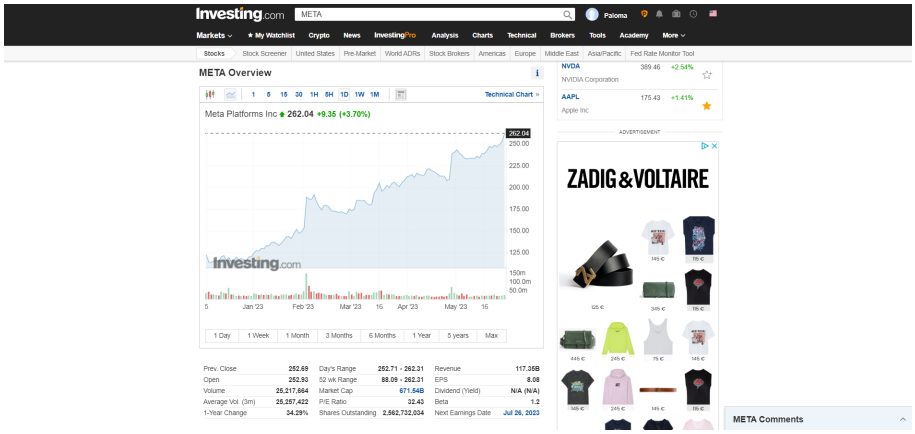


Figura 5.4: Resultado búsqueda activo financiero investing.com

- En la parte superior tenemos el apartado de 'My Watchlist' donde tendremos nuestras carteras de seguimiento.

  Dentro de este apartado, como muestra la figura 5.6, encontramos todas las carteras creadas, cada una con las posiciones que tenga, la opción para poder crear una nueva cartera y la opción de importar directamente un archivo con posiciones a una cartera.

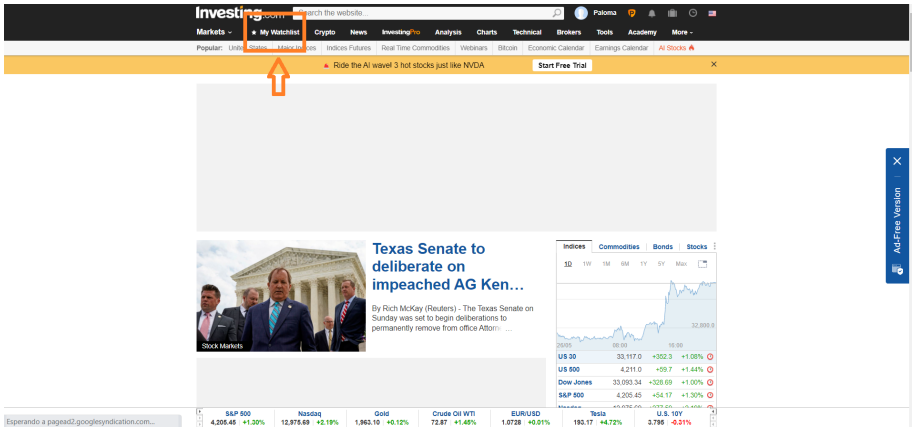


Figura 5.5: Botón de importar watchlist en investing.com

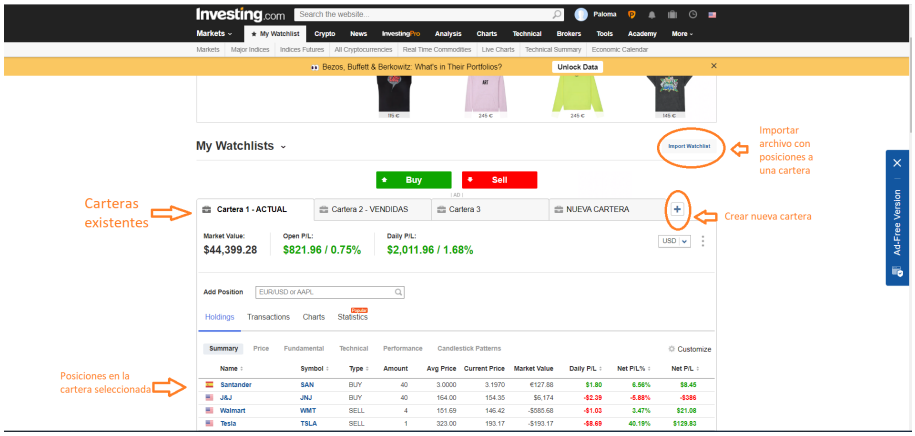


Figura 5.6: Sección de gestión de carteras en investing.com

A continuación vamos a mostrar con más detalle los apartados comentados:[7]

- Una cartera se crea simplemente mediante el símbolo '+' superior, introduciendo el nombre que le queremos dar y el tipo de cartera, que son dos:
  - 'watchlist'. Es una lista personalizada de activos financieros que un inversor desea seguir de cerca.
  - 'holdings'. Los holdings se refieren a las posiciones abiertas de un inversor en activos financieros, por lo que este tipo sirve para hacer el seguimiento de los activos que poseemos y por lo tanto al añadir un activo debemos especificar como mínimo información como: tipo de acción (compra o venta), fecha de la acción, cantidad y precio.
- Mediante el recuadro de 'Add Position' podemos introducir manualmente posiciones a nuestras carteras indicando los detalles de la posición.
  La Figura 5.7 muestra que en primer lugar hay que buscar el activo financierao que deseemos añadir.

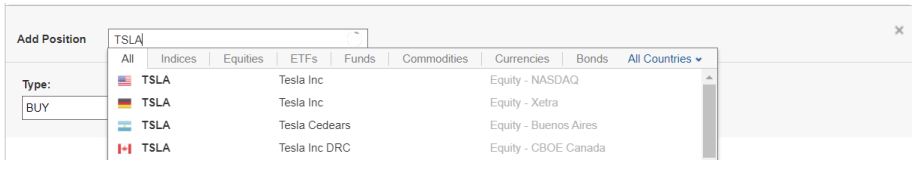


Figura 5.7: Añadir nueva posición en cartera de investing.com - Parte 1

La Figura 5.8 muestra el siguiente paso, que es rellenar los detalles de la posición mediante las variables, que son las siguientes:

- **Type**: El tipo de posición que estás agregando. Puede ser 'Compra' o 'Venta'. Indica si estás comprando o vendiendo un activo.
- **Date**: La fecha en la que realizaste la transacción.
- **Amount**: La cantidad de activos que estás comprando o vendiendo.
- **Price**: El precio al que estás comprando o vendiendo cada unidad del activo. Indica el precio por unidad en la moneda de tu elección, como dólares o euros.
- **Commission**: La comisión o tarifa asociada a la transacción.

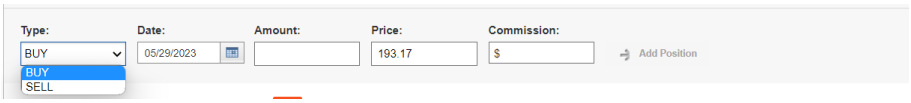
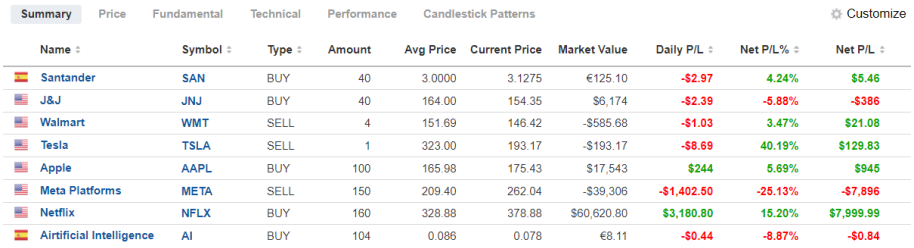


Figura 5.8: Añadir nueva posición en cartera de investing.com - Parte 2

- En la Figura 5.9 se muestra la tabla que contiene las posiciones de la cartera seleccionada. Para cada posición se muestran actualizados en cada momento los siguientes valores:
  - **Name**: El nombre del activo en el que tienes una posición. Por ejemplo, si se tratan de acciones de una empresa, aquí se muestra el nombre de la empresa.
  - **Symbol**: El símbolo o código abreviado del activo.
  - **Type**: El tipo de posición que tienes en el activo. Puede ser 'Compra' o 'Venta', indicando si compraste o vendiste el activo.
  - **Amount**: La cantidad de activos que el usuario posee en la posición.
  - **Avg. Price**: El precio promedio al que se compraron los activos en la posición.
  - **Current Price**: El precio actual del activo en el mercado. Indica el valor del activo en el momento en que se muestra la tabla.
  - **Market Value**: El valor total de la posición en el mercado. Se calcula multiplicando la cantidad de activos por el precio actual.
  - **Daily P/L**: La ganancia o pérdida diaria en la posición. Representa la variación de valor desde el día anterior.
  - **Net P/L %**: La ganancia o pérdida neta como un porcentaje. Muestra el cambio porcentual en el valor de la posición desde que el usuario la abrió.
  - **Net P/L**: La ganancia o pérdida neta en términos monetarios. Indica la diferencia entre el valor actual de la posición y el costo inicial.

  La diferencia de color en los valores de las últimas tres variables se utiliza para resaltar de manera visual si el valor es positivo o negativo. Es decir, los valores en verde para indican ganancias o rendimientos favorables. Por otro lado, los valores se presentan en rojo cuando hay que destacar pérdidas.



Summary | Price | Fundamental | Technical | Performance | Candlestick Patterns | Customize

Name	Symbol	Type	Amount	Avg Price	Current Price	Market Value	Daily P/L	Net P/L%	Net P/L
Santander	SAN	BUY	40	3.0000	3.1275	€125.10	-$2.97	4.24%	$5.46
J&J	JNJ	BUY	40	164.00	154.35	$6,174	-$2.39	-5.88%	-$386
Walmart	WMT	SELL	4	151.69	146.42	-$585.68	-$1.03	3.47%	$21.08
Tesla	TSLA	SELL	1	323.00	193.17	-$193.17	-$8.69	40.19%	$129.83
Apple	AAPL	BUY	100	165.98	175.43	$17,543	$244	5.69%	$945
Meta Platforms	META	SELL	150	209.40	262.04	-$39,306	-$1,402.50	-25.13%	-$7,896
Netflix	NFLX	BUY	160	328.88	378.88	$60,620.80	$3,180.80	15.20%	$7,999.99
Airtificial Intelligence	AI	BUY	104	0.086	0.078	€8.11	-$0.44	-8.87%	-$0.84

Figura 5.9: Tabla resumen posiciones de la cartera de investing.com

- En la tabla de posiciones de la cartera, es importante tener en cuenta que cada posición puede tener más de una fila, conocidas como "detail rows". Esto se debe a que una posición puede estar compuesta por diferentes transacciones o acciones relacionadas. Por ejemplo, si compraste acciones de una empresa en diferentes momentos o a diferentes precios, cada compra se mostrará en una fila separada dentro de la posición correspondiente.
  Estas "detail rows"proporcionan un desglose más detallado de las transacciones y acciones que componen cada posición, lo que permite un seguimiento más preciso de las ganancias, pérdidas y cambios en el valor de la cartera.
  A continuación, en la Figura 5.10, se muestra un ejemplo visual de cómo se ven las “detail rows” en la tabla del resumen de posiciones:

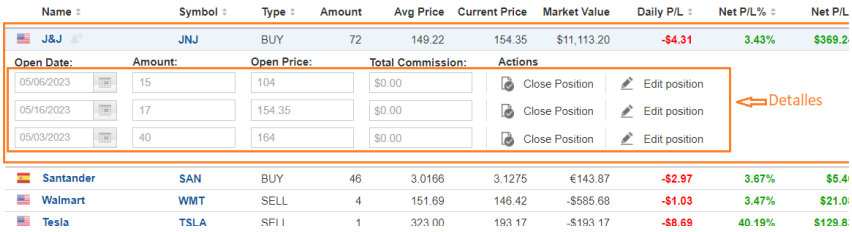


Figura 5.10: Tabla resumen con posiciones detalladas de la cartera de investing.com

- Es importante destacar que, aunque se refieran al mismo activo, las posiciones de compra y venta se consideran distintas en la tabla del resumen de posiciones, como ocurre en la Figura 5.11.
  Esto significa que cada posición de compra y cada posición de venta se representan como entradas separadas en la tabla, incluso si se trata del mismo activo.

Name	Symbol	Type	Amount	Avg Price	Current Price	Market Value	Daily P/L	Net P/L%	Net P/L
J&J	JNJ	SELL	7	154.35	154.35	-$1,080.45	$0.41	0.00%	$0
J&J	JNJ	BUY	72	149.22	154.35	$11,113.20	-$4.31	3.43%	$369.24

Figura 5.11: Posición de compra y de venta de un activo en la cartera de investing.com

- Dentro de cada posición en la cartera, hay dos botones para administrar las posiciones: ‘Close Position’ y ‘Edit Position’, como se veía en la Figura 5.10 anteriormente.
  - **Close Position**: te permite cerrar la posición, indicando la intención de vender esa posición totalmente o solo una parte de ella.
    Como se muestra en la Figura 5.12, debemos especificar la fecha y precio de venta, así como la cantidad vendida.
  - **Edit Position**: te permite realizar modificaciones en una posición existente, ya sea por ajuste de estrategia o por una introducción errónea de los datos.

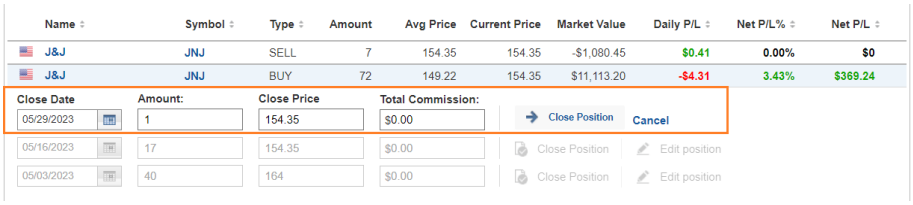


Figura 5.12: Cierre de posición en la cartera de investing.com

- Una forma más directa de añadir varias posiciones al mismo tiempo es mediante el botón de 'import my watchlist', donde subimos un archivo csv que contiene las posiciones que queremos añadir con las variables que las describen. Para conseguirlo, al pulsar el botón tenemos que seguir tres pasos:
  - Elegir el archivo csv donde están las posiciones que queremos importar, la cartera donde las queremos importar y el tipo de divisa.

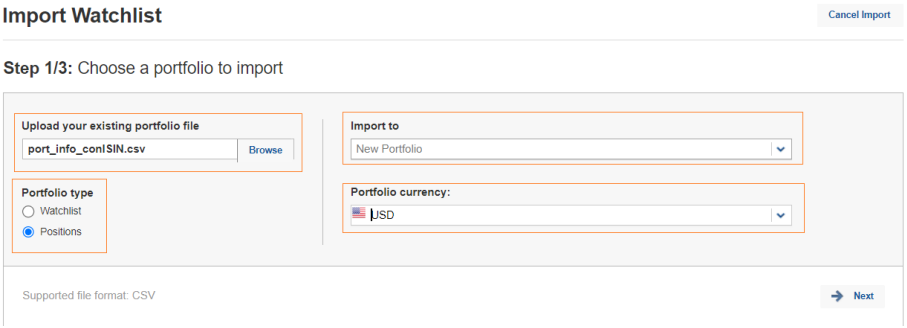


Figura 5.13: Importar posiciones a cartera de investing.com - Elegir portfolio

  - Según los valores de las posiciones en el csv, se completan automáticamente los campos. Ahora es el momento de comprobar que los campos son corretos y elegir las posiciones que efectivamente queremos añadir.

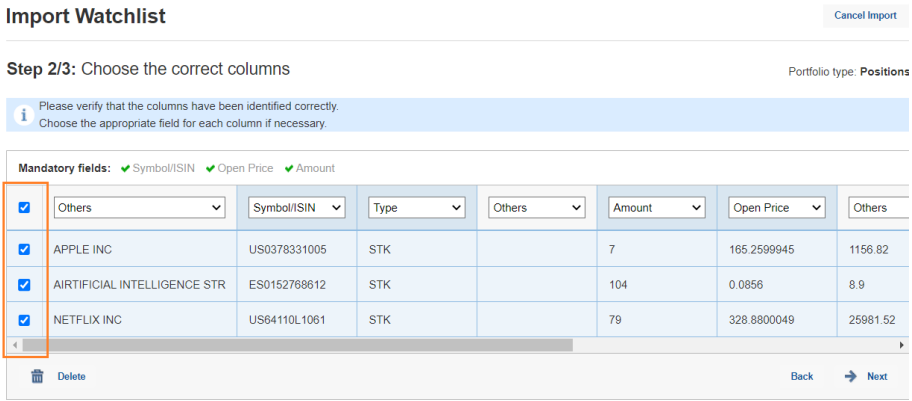


Figura 5.14: Importar posiciones a cartera de investing.com - Elegir columnas correctas

- Por último, nos muestra el formato definitivo en el que se van a añadir las posiciones y confirmamos.

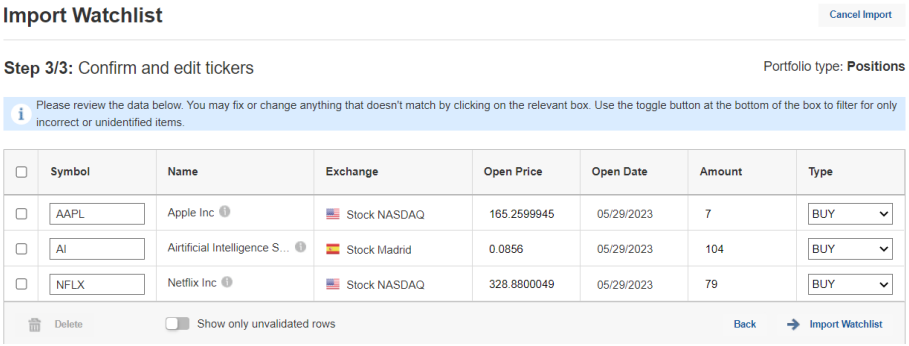


Figura 5.15: Importar posiciones a cartera de investing.com - Confirmación

CAPÍTULO 6
Tecnologías empleadas para el acceso a los datos

La creación del programa se realiza mediante el uso del lenguaje de programación Python. Por otra parte, se necesita concretar las herramientas a utilizar tanto en la descarga de datos del bróker como en la carga de las posiciones en las diferentes carteras. Estas herramientas dependen de los instrumentos tecnológicos que nos proporcionen las plataformas escogidas, Interactive Brokers e investing.com para interactuar con ellas. En el capítulo anterior ya hemos comentado solo IBKR nos ofrece una API para interactuar, Por lo que la forma para acceder a los datos de cada plataforma va a ser la siguiente:

- Interactive Brokers. Nos descargaremos los datos de las operaciones realizadas en nuestra cuenta mediante la **TWS API** (Interactive Brokers Trader Workstation Application Programming Interface).

- investing.com. No proporciona ninguna herramienta para intercambiar información con la plataforma, por lo que haremos uso de la herramienta **Selenium WebDriver** para interactuar con la página web y en algunas ocasiones también haremos uso de **Web Scraping** para extraer información contendida en la página.

En las siguientes secciones comentamos de forma más técnica de qué se tratan las herramientas y tecnologías comentadas.

6.1 Python

Usaremos este lenguaje para la creación de todos los programas necesarios y para comunicarnos con cualquier API o página web, ya que es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML) [4], que además nos va a ofrecer una gran cantidad de bibliotecas y herramientas con los que poder desarrollar programas de descarga y subida de datos entre diferentes plataformas.

En cuanto a las librerías principales utilizadas, se ha hecho uso de las librerías `ibapi` para interactuar con la TWS API de Interactive Brokers y de las librerías `selenium` y `bs4` para automatizar la interacción con páginas web.

6.2 ¿Qué es una API?

Una API (Application Programming Interface) es un conjunto de reglas y protocolos que permiten a diferentes aplicaciones comunicarse y compartir datos entre sí[43]. En lugar de acceder a los datos de una aplicación directamente, la API permite que otra aplicación solicite los datos a través de una interfaz bien definida.
En nuestro caso, usamos la API de una plataforma a través de un programa de Python, así el programa se comunica con la plataforma y solicita los datos que necesita.

6.2.1. TWS API

La TWS API es una interfaz de programación de aplicaciones proporcionada por Interactive Brokers que permite a los desarrolladores crear aplicaciones personalizadas para interactuar con la plataforma de Trader Workstation. Esta API sigue el concepto general de una API, que es un conjunto de reglas y protocolos que permiten la comunicación entre diferentes aplicaciones.

Con la TWS API, es posible automatizar diversas funcionalidades de la plataforma TWS, como la ejecución de órdenes, la recepción de datos de cartera y mercado, y la consulta de información detallada sobre instrumentos financieros.

Al utilizar la TWS API en conjunto con un programa en Python, podemos establecer una comunicación entre el programa y la plataforma TWS. El programa puede enviar solicitudes a la API para obtener datos específicos y recibir respuestas correspondientes.[44]

Para utilizar la TWS API y recuperar datos de una cuenta, es necesario tener la aplicación de escritorio Trader Workstation abierta. Esto se debe a que la API se comunica con la TWS para acceder a los datos y funciones relacionadas con la cuenta y las operaciones financieras. La TWS actúa como el intermediario entre la API y los servidores de IBKR, proporcionando la conexión y la autenticación necesarias.

Esta subsección proporciona una descripción básica de la TWS API, y para obtener información más detallada y ejemplos específicos, se remite al Anexo B, donde se proporciona una introducción completa a la TWS API de Interactive Brokers, incluyendo la instalación, configuración, acceso al código fuente y los componentes esenciales de los programas de la API.

6.3 Selenium

¿Qué es Selenium?

Selenium es una herramienta de automatización de pruebas de software para aplicaciones web que nos permite simular acciones del usuario en un navegador.
Es una herramienta de código abierto y es compatible con varios lenguajes de programación, como Java, Python, C#, Ruby o JavaScript. [9]
Centrándonos en el lenguaje Python, este cuenta con la librería `selenium` [45], con la que se pueden crear scripts de Python que utilicen la API de Selenium para controlar un navegador web y automatizar acciones como hacer clic en botones, llenar formularios, y navegar por páginas web. Para utilizar esta librería es necesario en primer lugar que el programador investigue el HTML de la página web que se desea automatizar, y luego utilizar Selenium para interactuar con los esos elementos HTML.

Esta forma de acceder a una página web es arriesgada y no es la preferida debido a que en cualquier momento la página puede cambiar su estructura HTML, y por lo tanto el código creado para accederla ya no nos serviría.

6.3.1. Inicialización y configuración

Para hacer uso de selenium en Python, en primer lugar es necesario tener instalada la librería 'Selenium' [46].

Después de haber completado la instalación, Selenium necesita un navegador web instalado en el sistema para poder funcionar, como pueden ser Chrome, Firefox, Edge, Internet Explorer o Safari. En el caso específico de nuestro proyecto, hemos elegido trabajar con Chrome, por la alta compatibildiad de este navegador con Selenium y la familiarización con él.

Una vez escogido el navegador, debemos descargar y configurar el controlador asociado a ese navegador, el cual nos permitirá manejarlo [47].En nuestro caso es el controlador de Chrome ('ChromeDriver), que podemos encontrar en
https://sites.google.com/a/chromium.org/chromedriver/downloads [48].

Teniendo el controlador inicializado, ya podemos abirir una página web.

Al incio de cualquier script relacionado con la plataforma de investing.com se realiza la inicalización comentada para poder acceder e interactuar con la página.

6.4 Web Scraping

El web scraping es un conjunto de prácticas utilizadas para extraer automáticamente — o «scrapear» — datos de la web, a partir de su código HTML [49].
Para realizar Web Scraping con Python, hacemos uso de la librería Beautiful Soup [50], que nos permite extraer información de contenido en formato HTML o XML [51].

Haremos uso del web scrapping, mediante la librería Beautiful Soup, cuando sea necesario analizar y extraer elementos del código HTML de la página web de investing.

CAPÍTULO 7
Estrategia a seguir para conseguir la monitorización

Como ya se ha comentado en los anteriores capítulo, la idea a desarrollar es reflejar en la plataforma de gestión de portfolios, en nuestro caso investing.com, las transacciones que realiza un inversor.

En este capítulo se exponen las diferentes carteras que vamos a crear para reflejar las operaciones que se realicen, cuáles son sus objetivos de análisis y cómo se plasma cada operación realizada en IBKR en cada una de las carteras.
Posteriormente, se comenta esquemáticamente cómo realizar la migración de datos entre plataformas, que se comentará en detalle en los Capítulos 8 y 9.

7.1 Carteras necesarias en nuestro gestor de portfolio

Se ha decido crer en la plataforma de gestión de portfolios **tres carteras** diferentes, en las cuales reflejaremos todos los movimientos que el usuario haga mediante en bróker.
Cada una de estas carteras tiene diferentes objetivos de análisis y dependiendo de ellos, el inversor deberá poner más atención en una cartera u otra.

A continuación vamos a explicar la información que va a reflejar cada una de las tres carteras y su utilidad.

7.1.1. Cartera 1

Esta cartera es un fiel reflejo de lo que el inversor tiene en su cartera de IBKR, es decir, contiene las posiciones que el usario tiene abiertas en el momento.
Así, podremos ver de un solo vistazo, por una parte indicadores de las ganancias y pérdidas globales, y por otra parte indicadores concretos para cada una de las posiciones, tanto la información relativa a los datos de compra y venta como una actualización constante de los precios y pérdidas y ganancias.
De esta forma, gracias a todos esos indicadores, es más directo analizar el desempeño de las posiciones de inversión, y por lo tanto evaluar el rendimiento de las inversiones y tomar decisiones informadas sobre cuándo comprar, vender o mantener posiciones.

Reflejar las operaciones de IBKR en la Cartera 1

En la siguiente tabla 7.1, mostramos un ejemplo sencillo, para un determinado activo, de cómo se reflejan las operaciones de IBKR en la Cartera 1.
Los valores de la tabla reflejan la cantidad de acciones que se compran o venden del activo. Por otra parte, el símbolo positivo (+) refleja una operación de compra y el símbolo negativo (-) una operación de venta.

Operación IBKR	**Cartera 1**	
	Operación	**Resultado posición**
0	0	0
+5	+5	5
+10	+10	15
+20	+20	35
-15	-15	20
+15	+15	35
+25	+25	60
-20	-20	40
+10	+10	50
-10	-10	40
+30	+30	70
-100	-100	-30

Tabla 7.1: Reflejo de las operaciones de IBKR en la Cartera 1

En conclusión, aquí apuntaremos las posiciones que el inversor tiene actualmente abiertas.

7.1.2. Cartera 2

Esta cartera, de caracter ficticio y meramente informativo, está diseñada para usuarios que orientan su inversión a operar en **largo**, la cual es una estrategia en la que el inversor compra un activo con la expectativa de que su precio aumente en el futuro para obtener ganancias. El uso de esta estrategia se suele utilizar cuando se tiene una perspectiva optimista sobre el rendimiento futuro de un activo.

En la siguiente Figura 7.1 mostramos esto de forma gráfica, a partir del gráfico Candlestick de un activo donde se muestra la evolución de los precios del activo a lo largo del tiempo. Concretamente vemos en el eje X el precio del activo y en el eje Y el tiempo. En este ejemplo, el inversor ha comprado unas acciones del activo en Octubre a unos 130$, y le gustaría vender cuando suben de precio, por ejemplo en Enero a un valor de entorno 155$.

Ejemplo operación en largo:
Supongamos que la empresa XYZ está siendo negociada a $130 dólares. Después de realizar un análisis exhaustivo, determinas que existe una alta probabilidad de que el precio aumente en el futuro, por lo tanto, decides comprar 10 acciones a $130 dólares cada una. El costo total de la operación sería de (10 * $130) = $1,300 dólares.

Posteriormente, el precio de la empresa XYZ sube a $155 dólares por unidad, y decides vender las 10 acciones a este nuevo precio. Como resultado, obtienes un total de (10 * $155) = $1,550 dólares. En consecuencia, has obtenido una ganancia de $250 dólares.

Sin embargo, en un escenario alternativo, supongamos que el precio no sube, sino que disminuye a $100 dólares por unidad. Si decidieras vender las 10 acciones a este precio, tendrías un total de (10 * $100) = $1,000 dólares. En este caso, habrías experimentado una pérdida de $300 dólares.[52]

Figura 7.1: Ejemplo de operación en largo

¿Cómo esta cartera aporta valor al inversor?

Muchos inversores, suelen tener un objetivo en mente: mantener un cierto número de acciones de un activo determinado.

Por ejemplo, supongamos que un inversor está posicionado en largo y posee 100 acciones de Tesla. A medida que el precio de las acciones de Tesla comienza a subir, el inversor puede decidir tomar ganancias y vender una parte de sus acciones, digamos 20 de ellas. Esto le permite obtener beneficios y reducir su exposición al riesgo.
Sin embargo, es importante destacar que el inversor aún mantiene un interés en mantener un número constante de acciones de Tesla en su cartera. Por lo tanto, si el precio de las acciones de Tesla disminuye en el futuro, el inversor puede aprovechar la oportunidad y recomprar esas 20 acciones a un precio más bajo. Al hacerlo, el inversor puede aumentar su posición inicial de 100 acciones nuevamente, pero ahora habrá ganado dinero debido a la diferencia entre el precio de venta y el precio de recompra.

La utilidad de la Cartera 2 recae en que en ella vamos a apuntar todos los datos de esas operaciones de venta en largo, por lo que podremos ver la evolución posterior de esas acciones, con lo que se irán actualizando las pérdidas o ganancias que tendríamos respecto al momento el que hicimos la venta, y así poder decidir el buen momento para recomprar a un precio más bajo.

Reflejar las operaciones de IBKR en la Cartera 2

A continuación explicamos el procedimiento utilizado para trasladar las operaciones realizadas en IBKR a la Cartera 2 para reflejar las posiciones en largo.

En la Cartera 2 solo queremos tener apuntadas la cantidad de acciones que hemos vendido y el precio al que las vendimos, hasta que consigamos recuperarlas comprándolas de nuevo, para poder hacer el seguimiento y análisis de su evolución de precio.
Por lo tanto, la **casuística** es la siguiente:

- Los tipos de operaciones (compra/venta) que se realizan en IBKR se reflejan en la Cartera 2 de manera opuesta. Es decir - en el caso de que se tengan que añadir - las operaciones de compra de IBKR, se añaden en la cartera como tipo 'venta' y las operaciones de venta de IBKR, se añaden en la cartera como tipo 'compra'.
- Todas las operaciones de venta que realicemos en IBKR, las apuntaremos en la Cartera 2, ya que queremos hacerles un seguimiento.
- Las operaciones de compra que realicemos en IBKR solo las apuntaremos en la Cartera 2 en el caso de que se trate de una recompra de acciones que vendimos.

Con las siguientes tablas, mostraremos el mismo ejemplo que en la sección anterior en la Tabala 7.1 para la Cartera 1, pero esta vez trasladando también a la Cartera 2 las operaciones de la forma en que proceda en cada caso.

En la tabla 7.2 vemos que mientras realizamos operaciones de compra de acciones, no apuntamos esa informaciónen la Cartera 2, ya que no es el objetivo de esta cartera hacer un seguimiento de acciones que poseemos.
Sin embargo, cuando el usuario vende 15 acciones, sí que reflejamos esa cantidad en la Cartera 2 haciendo una compra de esa cantidad de acciones.

Operación IBKR	**Cartera 1**	**Cartera 2**	
		Operación	**Resultado posición**
0	0	-	0
+5	5	-	0
+10	15	-	0
+20	35	-	0
-15	20	+15	15

Tabla 7.2: Reflejo de las operaciones de IBKR en el número de acciones en Cartera 2. Parte 1

Continuando con el ejemplo en la Tabla 7.3, ahora hacemos una operación de compra de 15 acciones en IBKR, es decir, recompramos las 15 que habíamos vendido antes. Como es una operción de recompra la apuntamos como venta, por lo que ahora se cierra la posición y ya no tenemos acciones de ese activo a las que hacer seguimiento, porque ya las hemos recuperado.

Operación IBKR	**Cartera 1**	**Cartera 2**	
		Operación	**Resultado posición**
0	0	-	0
+5	5	-	0
+10	15	-	0
+20	35	-	0
-15	20	+15	15
+15	35	-15	0

Tabla 7.3: Reflejo de las operaciones de IBKR en el número de acciones en Cartera 2. Parte 2

En la Tabla 7.4, para finalizar el ejemplo, aparecen más operaciones que siguen la misma lógica que las dos anteriores.
Hay que destacar que en la penúltima operación solo se apunta una venta de 20 acciones en la Cartera 2 y no 30, porque es el número de acciones que queremos recuperar.

Operación IBKR	**Cartera 1**	**Cartera 2**	
		Operación	**Resultado posición**
0	0	-	0
+5	5	-	0
+10	15	-	0
+20	35	-	0
-15	20	+15	15
+15	35	-15	0
+25	60	-	0
-20	40	+20	20
+10	50	-10	10
-10	40	+10	20
+30	70	-20	0
-100	-30	+100	100

Tabla 7.4: Reflejo de las operaciones de IBKR en el número de acciones en Cartera 2. Parte 3

Este era un ejemplo general para mostrar el comportamiento con cualquier activo. Suponiendo que se trataba de acciones de la compañía Johnson & Johnson (J&J), en la Figura 7.2 se muestra cómo quedaría reflejado el resultado de todas las operaciones realizadas en la plataforma de investing.com, donde hemos acabado con 100 acciones en nuestra Cartera 2.

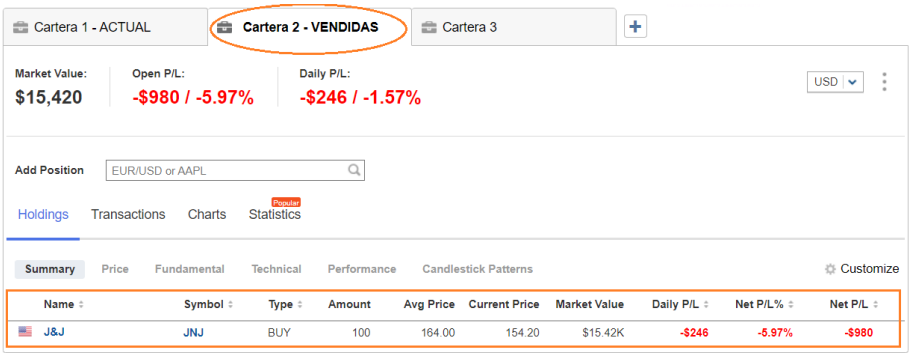


Figura 7.2: Ejemplo de posición en Cartera 2

Interpretación de las posiciones de la Cartera 2

A continuación se muestra cómo se vería en la Cartera 2, poniendo como ejemplo una posición, tanto una bajada de valor del activo como una subida:

- Si la cartera me indica que he tenido pérdidas en el activo y la información sobre pérdidas y ganancias se muestran en color rojo, es porque el precio actual del activo es menor del precio al que lo vendimos en IBKR. Por lo tanto, este es un buen momento para hacer la recompra, ya que podemos conseguir el mismo número de acciones a un precio menor.

  En la figura 7.3 vemos cómo el precio unitario actual de las acciones, es menor al precio que apuntamos en la Cartera 2, es decir, el precio de la venta de esas acciones desde IBKR.
  Las variables Daily P/L, Net P/L % y Net P/L nos indican exactamente la pérdida diaria y la pérdida neta. Los valores del ejemplo los podemos traducir como que si recompramos ahora las 100 acciones que vendimos en su momento, estaríamos ganando el valor de la variable 'Net P/L', en este caso ganaríamos 965$, lo que en términos de porcentaje es un 5.88 % de ganancias.

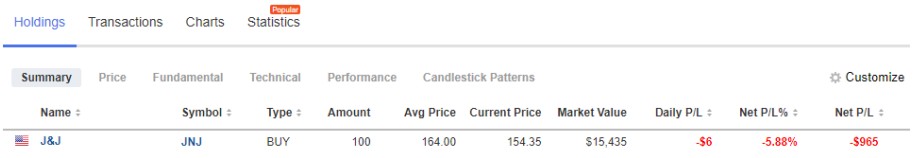


Figura 7.3: Reflejo de ganancias en la Cartera 2

- Si la cartera me indica que he tenido ganancias en el activo y la información sobre pérdidas y ganancias se muestran en color verde, es porque el precio actual del activo es mayor del precio al que lo vendimos en IBKR. Por lo tanto, este no es un buen momento para hacer la recompra, ya que el número de acciones que teníamos antes ahora están a un precio mayor.

  Este otro ejemplo lo podemos ver en la Figura 7.4, donde el precio unitario actual de las acciones es mayor que el precio al que las vendimos en IBKR, y nuevamente, las variables Daily P/L, Net P/L % y Net P/L nos indican exactamente la ganancia diaria y la ganancia neta.

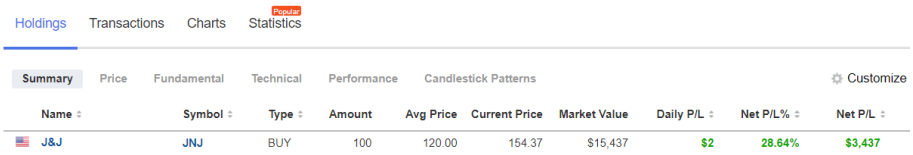


Figura 7.4: Reflejo de pérdidas en la Cartera 2

7.1.3. Cartera 3

Esta cartera, de caracter ficticio y meramente informativo, está diseñada para usuarios que orientan su inversión a operar en **corto**, los cules predicen que el mercado tendrá una tendencia a la baja, no compran acciones, sino que las piden prestadas, normalmente, a otro inversor que esté vendiendo o a un bróker. Así, las pide prestadas para venderlas al precio actual con el compromiso de comprarlas más adelante para devolverlas.
El objetivo del inversor es recomprar esas acciones que tiene que devolver a un precio menor para obtener ganancias. [52]

En la siguiente Figura 7.5 mostramos esto de forma gráfica, a partir del gráfico Candlestick de un activo donde se muestra la evolución de los precios del activo a lo largo del tiempo. Concretamente vemos en el eje X el precio del activo y en el eje Y el tiempo. En este ejemplo, el inversor ha vendido unas acciones prestadas del activo en Abril a unos 155$, y le gustaría volveras a comprar cuando bajen de precio, por ejemplo en Julio a un valor de entorno 125$.

Figura 7.5: Ejemplo de operación en corto

Ejemplo operación en corto:
Imagina que realizas un análisis y predices que el precio de la empresa XYZ va a experimentar una disminución. En este momento se está comercializando a unos $155 dólares. Decides tomar una posición en corto y tomas prestadas 10 unidades de acciones de un bróker o de otro inversionista dispuesto a venderlas.

Al final del día, efectivamente, el precio de la empresa ha caído a $125 dólares. Decides cerrar tu posición en corto y procedes a comprar las mismas 10 unidades, esta vez a $125 dólares cada una.

El costo inicial de tomar prestadas las 10 acciones fue de (10 * $155) = $1,550 dólares. Ahora, el costo de comprar las 10 acciones a $125 dólares cada una es de (10 * $125) = $1,250 dólares. Como resultado, obtienes una ganancia de $300 dólares.

Sin embargo, en un escenario contrario, supongamos que el precio aumenta a $175 dólares. En este caso, al estar obligado a comprar las mismas 10 acciones para devolverlas, habrías incurrido en una pérdida en la transacción. [52]

¿Cómo esta cartera aporta valor al inversor?

Esta cartera le permite al usuario tener controladas en cada momento las acciones que en un momento tomó prestadas para vender, y por lo tanto debe volver a comprarlas para devolverlas.
Así, se puede ver el dinero que estaríamos ganando o perdiendo al recomprar las acciones respecto al precio al que hicimos la venta, y decidir el momento más conviniente para hacer la recompra.

Reflejar las operaciones de IBKR en la Cartera 3

A continuación se explica el procedimiento utilizado para reflejar las posiciones en corto en la Cartera 3.

En la Cartera 3, se filtran y añaden únicamente aquellas posiciones que corresponden a operaciones en corto. Para hacer este filtro tenemos que tener en cuenta dos cosas:

- La Cartera 1 tiene las posiciones abiertas en cada momento.
- En una cartera de investing.com, cuando una posición se etiqueta como tipo "SELL", significa que se trata de una posición corta.

Por lo tanto, filtraremos las posiciones cortas uniendo ambas informaciones. Es decir, en primer lugar miraremos la Cartera 1 y en el caso de que un activo tenga posición en corto, es decir, de tipo 'Sell', añadiremos esa misma posición a la Cartera 3.

En la siguiente tabla se muestra un ejemplo de esta idea, donde como en las tablas anteriores, los números negativos son operaciones tipo venta (operaciones en corto):

Cartera 1	Cartera 3
0	0
5	0
15	0
-5	-5
2	0

Tabla 7.5: Reflejo de las operaciones de IBKR en el número de acciones en Cartera 3.

Interpretación de las posiciones de la Cartera 3

A continuación se muestra cómo se vería en la Cartera 3, poniendo como ejemplo una posición, tanto una bajada de valor del activo como una subida:

- Si el precio actual es menor que el precio de venta, significa que el valor del activo en el mercado es actualmente inferior al precio al que vendiste. Es decir, puedes recomprar esas acciones que tienes que devolver por un precio más bajo en comparación con el precio al que las vendiste.
  Las variables de pérdidas y ganacias estarían reflejadas en color verde.

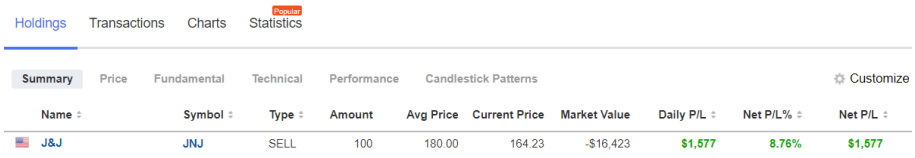


Figura 7.6: Reflejo de ganancias en la Cartera 3

- Si el precio actual es mayor que el precio de venta, significa que el valor del activo en el mercado es actualmente superior al precio al que vendiste. Es decir, si decides recomprar las acciones ahora deberías pagar por ellas un precio mayor en comparación al precio al que las vendiste, lo cual te ocasiona pérdidas.
  Las variables de pérdidas y ganacias estarían reflejadas en color rojo.

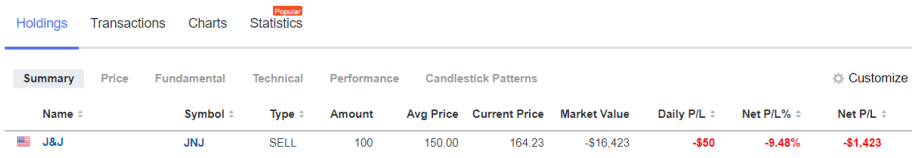


Figura 7.7: Reflejo de pérdidas en la Cartera 3

Como los inversores dividen su capital en invertir en varios activos al mismo tiempo, dependiendo de la estrategia que tenga en cada momento con cada uno de esos activos se fijará en una cartera u otra.
Los inversores alcistas - que piensan que el precio de un activo financiera va a subir - prestarán atención a la cartera 2, mientras que los inversores bajistas - que piensan que el precio de un activo financiera va a bajar - prestarán atención a la cartera 3.
Al ofrecer un seguimiento de operaciones tanto en el lado largo como en el corto, el usuario puede ajustar rápidamente sus estrategia según las condiciones del mercado y capitalizar diferentes escenarios. Esta flexibilidad permite aprovechar las fluctuaciones del mercado y proporcionando adaptación a las condiciones económicas cambiantes.

7.2 Funcionalidades a implementar

Una vez ha quedado claro cuáles son los objetivos de análisis mediante las tres carteras que crearemos en el gestor de portfolios, podemos proceder a mostrar cómo sería el proceso de descarga de las operaciones realizadas y posterior introducción de esas operaciones a las distintas carteras, es decir, cómo vamos a relacionar e integrar la información que nos ofrecen ambas plataformas.

Funcionalidad general de la aplicación

La Figura 7.8 muestra el esquema a seguir para la creación de la aplicación. Esta figura muestra claramente las dos plataformas utilizadas, por una parte Interactive Brokers y por otra parte investing.com, y su forma de relacionarse.

El pograma estaría dividido en dos fases o etapas. La primera etapa corresponde a la de inicialización de las carteras y la segunda etapa ya corresponde a la monitorización diairia de las operaciones que se van ejecutando cada día.

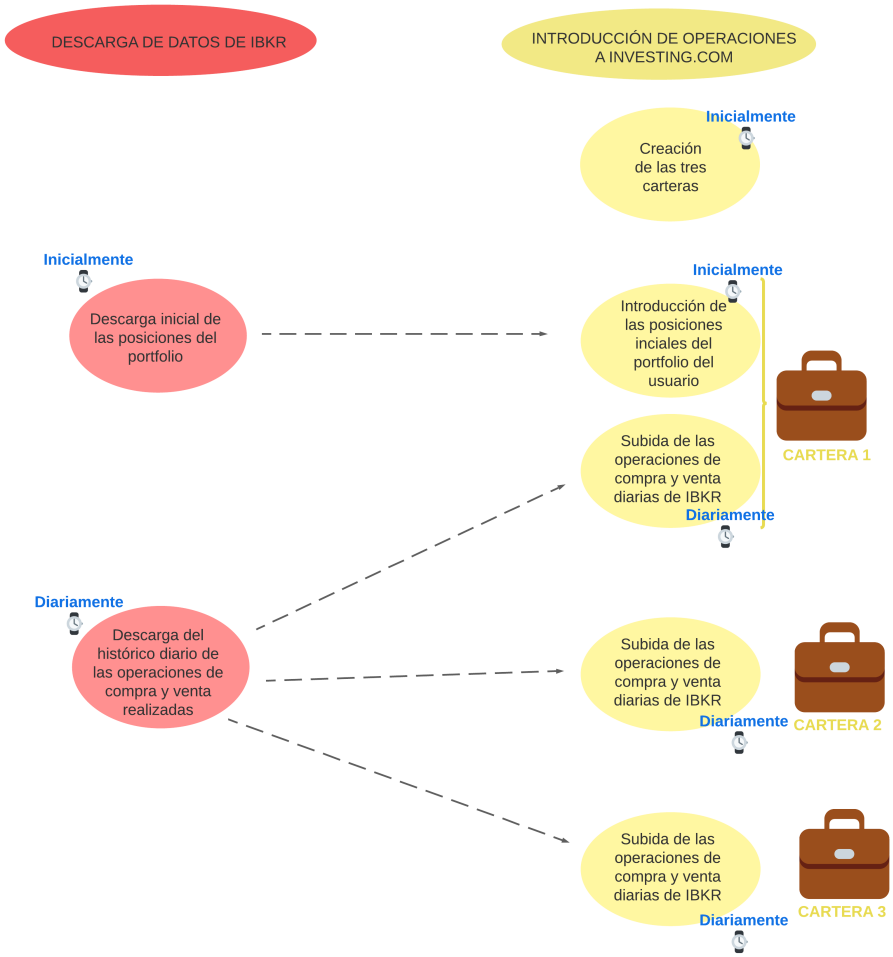


Figura 7.8: Diagrama descarga e introducción de datos

Fase 1 - Inicialización de las carteras

Este es el paso previo a poder añadir diariamente las operaciones.
Se basa en la creación de las tres carteras de investing.com y la inicialización de la Cartera 1 con las posiciones que pudiera tener abiertas actualmente el usuario.

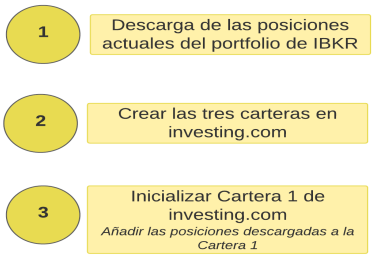


Figura 7.9: Pasos de inicialización previos a la monitorización diaria

Fase 2 - Monitorización diaria de las transacciones realizadas

Una vez ya tengamos las carteras de investing.com inicilizadas, se podrá ejecutar diariamente la monitorización, migrando las operaciones hechas en IBKR durante el día anterior a cada una de las carteras según corresponda. La aplicación para realizar esta monitorización seguirá la siguiente estructura:

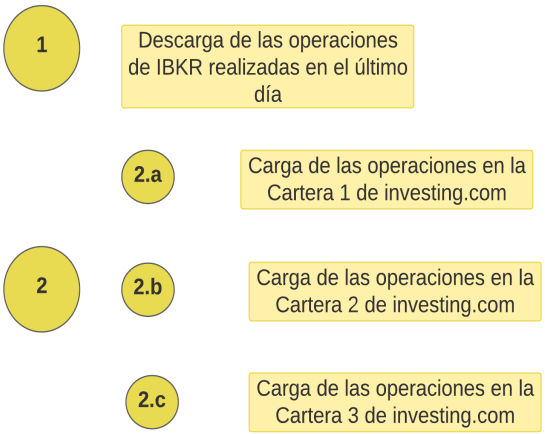


Figura 7.10: Estructura de la aplicación para la monitorización diaria de operaciones

La explicación completa de cómo realizar las diferentes descarga de datos de IBKR se expone en el Capítulo 8, y la explicación de los algoritmos creados para que se realice de forma automática la introuducción de posiciones a investing.com según el tipo de cartera se realiza en el Capítulo 9.

CAPÍTULO 8
Descarga de datos de IBKR

En este capítulo se comenta cómo se realiza la descarga de datos de la cuenta de Interactive Brokers de un usuario.
En primer lugar se trata la recuperación de las posciones del portfolio inicial y en segundo lugar se trata la recuperación del histórico diario de operaciones realizadas.

8.1 Descarga de las posiciones del portfolio de IBKR

En esta sección se explica cómo se realiza la descarga de las posiciones que tiene el usuario en su portfolio de Interactive Brokers y su posterior almacenamiento en un archivo csv.

Recordamos que estos datos de las posiciones que tiene abiertas el usuario sirven para la inicialización de la Cartera 1.

Acceso a los datos del portfolio e información de la cuenta

Vamos a definir la clase con la que descargar la información de nuestro portfolio de IBKR.
Para conseguir esto hay varias funciones diferentes en la API que pueden ser utilizadas [53]:

- **ReqAccountUpdates**. Devuelve información tanto de la posición como de la cuenta para una cuenta especificada. Sólo puede utilizarse con una única cuenta a la vez.
- **ReqPositions**. Para suscribirse a actualizaciones de posiciones de hasta 50 subcuentas simultáneamente.
- **ReqPositionsMulti** En el caso de que haya un gran número de subcuentas.
- **reqAccountSummary**. Para suscribirse a las actualizaciones de varias cuentas a la vez.

Nuestro caso es el más sencillo, solo vamos a necesitar pedir y recibir información de una cuenta, por lo que haremos uso de la función `ReqAccountUpdates`.

Cuando se llama a la función para solicitan datos del portfolio a la API, inicialmente se devuelve una lista completa de todos los tipos de datos que coinciden con la consulta y, a continuación, se envían actualizaciones si hay una operación o si el valor de la cuenta ha cambiado en el periodo de 3 minutos.

Para utilizar las funciones necesarias de la API y recuperar la información del portfolio del usuario de IBKR, vamos a crear en Python una clase a la que vamos a llamar `PortfolioApp`.
Dentro de esta clase definiremos varias funciones necesarias para obtener la información de cada posición del portfolio. Hay que destacar que vamos a referirnos a cada activo que poseamos como "contrato", ya que en el contexto de IBKR el término "contrato" se refiere a un acuerdo para comprar o vender un instrumento financiero en un intercambio.

Las funciones que creamos en la clase 'PortfolioApp' son las siguientes:

- `__init__ (self)`
  Función de inicialización de la clase. Aquí inicializamos las variables que utlizaremos para almacenar la información de cada posición del portfolio, que son:
  - *symbol* El símbolo del contrato. Por ejemplo, AAPL para Apple Inc.
  - *sectype* El tipo de seguridad del contrato. Acciones, opciones, futuros, forex, etc.
  - *exchange* Bolsa de valores en la que se cotiza el contrato.
  - *currency* La cantidad actual de contratos de ese tipo en la cuenta del usuario.
  - *position* El precio actual de mercado del contrato.
  - *marketPrice* El valor de mercado actual de la posición. Se calcula multiplicando la cantidad de contratos por el precio de mercado actual.
  - *marketValue* El costo promedio ponderado de la posición. Es el precio promedio al que el usuario ha adquirido los contratos.
  - *averageCost* La ganancia o pérdida no realizada de la posición. Es la diferencia entre el valor de mercado actual y el costo promedio ponderado.
  - *unrealizedPNL* La ganancia o pérdida no realizada de la posición. Es la diferencia entre el valor de mercado actual y el costo promedio ponderado.
  - *realizedPNL* La ganancia o pérdida realizada de la posición. Es la diferencia entre el precio de venta y el precio de compra de los contratos que se han vendido.
  - *accountName* El nombre de la cuenta del usuario en la que se encuentra la posición.

  Cada una de esas variables la incializamos como una lista vacía, para ir añadiendo en ellas la información de cada posición del portfolio.

- `error(self, reqId, errorCode, errorString, errorObj)`
  Función que se llama cuando se produce un error durante la comunicación con la API de IBKR y nos muestra por pantalla la explicación del error.

- `nextValidId(self, orderId)`
  Función que se llama cuando se recibe un ID válido de la API de IBKR. Llama a la función start().

- `updatePortfolio(self, contract: Contract, position: float, marketPrice: float, marketValue: float, averageCost: float, unrealizedPNL: float, realizedPNL: float, accountName: str)`
  Esta función se llama cuando se actualiza la información de la cuenta del usuario. Almacena la información de cada posición del portfolio en las variables definidas en la función de inicialización.

- `updateAccountTime(self, timeStamp: str)`
  Es una callback function mediante la cual la API de IBKR enviará periódicamente actualizaciones del portfolio. Por lo tanto, si hay nuevas posiciones en el portfolio se añadirán también a los atributos correspondientes de la instancia de la clase PortfolioApp.

- `start(self)`
  Función que se llama para comenzar a recibir actualizaciones del portfolio del usuario. Llama a la función `reqAccountUpdates()` de la API de IBKR.

- `stop(self)`
  Función que se llama para detener la recepción de actualizaciones del portfolio del usuario. Llama a la función `reqAccountUpdates()` de la API de IBKR con el argumenta 'False', y se desconecta de la API.

Una vez tenemos bien definidas todas las funciones de la clase `PortfolioApp()`, podemos ejecutar la aplicación para comenzar a recibir la información del portfolio. La ejecución se hace siguiendo los siguientes pasos:

1. Creamos una instancia de la clase `PortfolioApp()`.

2. Conectamos la instancia de app a la dirección IP 127.0.0.1, puerto 7497, que es la dirección para para conectarnos a la plataforma de IBKR.
   La IP 127.0.0.1 es la dirección de localhost, lo que significa que la conexión se realiza en la misma máquina donde se está ejecutando el programa.
   El puerto 7497 es el puerto predeterminado utilizado por la TWS API para la conexión en tiempo real.

3. Llamamos al método `start()` e iniciamos la aplicación, lo que significa que comienza a recibir y procesar los datos de la conexión con IBKR.

Guradar los datos descargados en un archivo csv

Queremos tener guarada toda la información de las posiciones extraídas en un fichero, y en nuestro caso la iremos guardando en un fichero csv, para su posterior carga en el gestor de portfolios.

Cada observación del csv es una posición del portfolio, y las variables son las diferentes características de esa posición que hemos extraído. El archivo csv con la información recopilada quedaría de esta forma:

name	ISIN	sectype	exchange	position	marketPrice	marketValue	averageCost	unrealizedPNL	realizedPNL	accountName
APPLE INC	US0378331005	STK		7	1652599945	115682	15982714285	3803	0	DU6828925
AIRTIFICIAL INTELLIGENCE STR	ES0152768612	STK		104	856	89	1681905	-859	-308	DU6828925
NETFLIX INC	US64110L1061	STK		79	3288800049	2598152	32471176455	32929	0	DU6828925

Figura 8.1: Archivo csv con el portfolio recuperadas de IBKR

8.2 Descarga del histórico diario de compras y ventas de IBKR

En esta sección se explica cómo se realiza la descarga de los datos históricos diaros de las operaciones realizadas en la cuenta de Interactive Brokers del usuario y su posterior almacenamiento en un archivo csv.

Acceso al histórico de operaciones de la cuenta

En este caso, como ya se ha mencionado, se descargan las operaciones diarias, pero el período del histórico se puede modificar dentro de la aplicación de escritorio Trader Workstation, pudiendo elegir la recuperación desde hoy hasta los últimos 7 días.

Para utilizar las funciones necesarias de la API y recuperar la información histórica, vamos a crear en Python una clase llamada 'MyTrades_BuySell_App'. Se definen, al igual que en la sección anterior 8.1, las funciones `error`, `nextValidId`, `start` y `stop`.
Por otra parte, se llama a la función de la TWS API que nos permite recibir los detalles de ejecución de las operaciones realizadas en la cuenta de IBKR, que es `execDetails` [54].
Esta función tiene los siguientes parámetros:

- reqId: Es el ID de la solicitud para los detalles de ejecución.
- contract: Es un objeto Contract que contiene información sobre el contrato del instrumento financiero involucrado en la ejecución.
- execution: Es un objeto Execution que contiene los detalles de ejecución de la operación, como el precio, la cantidad, el tipo de operación (compra o venta), el tiempo de ejecución.

Al invocar la función `execDetails`, se procesan los detalles de ejecución de las operaciones de compra y venta, extrayendo los detalles que hemos considerado relevantes para la ejecución y almacenándolos en variables en la instancia de la clase.
Los detalles que extramos, algunos ya mencionados en la sección anteior, son los siguientes:

- *conID* El ID del contrato
- *symbol* El símbolo del contrato.
- *sectype* El tipo de seguridad del contrato.
- *exchange* Bolsa de valores en la que se cotiza el contrato.
- *currency* La divisa.
- *action* Tipo de operación. 'BOT' indica operación de compra y 'SLD' indica operación de venta.
- *quantity* Cantidad ejecutada.
- *executionPrice* Precio de ejecución.
- *executionTime* Fecha de ejecución.

Una vez tenemos bien definidas todas las funciones de la clase 'MyTrades_BuySell_App', podemos ejecutar la aplicación para comenzar a recibir la información del portfolio. La ejecución se hace siguiendo los siguientes pasos:

1. Creamos una instancia de la clase `PortfolioApp()`.
2. Conectamos la instancia de app a la dirección IP 127.0.0.1, puerto 7497, que es la dirección para para conectarnos a la plataforma de IBKR.
   La IP 127.0.0.1 es la dirección de localhost, lo que significa que la conexión se realiza en la misma máquina donde se está ejecutando el programa.
   El puerto 7497 es el puerto predeterminado utilizado por la TWS API para la conexión en tiempo real.

3. Llamamos al método `start()` e iniciamos la aplicación, lo que significa que comienza a recibir y procesar los datos de la conexión con IBKR.

Guradar los datos descargados en un archivo csv

Una vez tenemos los detalles de las operaciones almacenados en las variables comentadas anteriormente, guardaremos toda esa información obtenida en un fichero csv. De esta forma tendremos guardada la información de forma permanente para luego poder volcarla según corresponda en la plataforma de gestión de portfolios

Cada observación del csv es una operación realizada, y las variables son los detalles extraídos de dichas operaciones. El archivo csv con la información recopilada quedaría de esta forma:

action	name	ISIN	symbol	quantity	execPrice	execTime
BOT	JOHNSON & JOHNSON	US4781601046	JNJ	5	16414	20230519 13:53:12 US/Eastern
BOT	JOHNSON & JOHNSON	US4781601046	JNJ	10	16414	20230519 13:53:12 US/Eastern
BOT	JOHNSON & JOHNSON	US4781601046	JNJ	20	16414	20230519 13:53:12 US/Eastern
SLD	JOHNSON & JOHNSON	US4781601046	JNJ	15	16414	20230519 13:53:12 US/Eastern

Figura 8.2: Archivo csv con las operaciones recuperadas de IBKR

CAPÍTULO 9

Subida de las operciones de IBKR a las carteras de investing.com

En el Capítulo 7, discutimos de manera teórica cómo queremos reflejar las operaciones de IBKR en nuestras carteras de investing.com. Ahora, en este capítulo, daremos vida a esa teoría al presentar los algoritmos de progrmación que cumplen los pasos necesarios y también indicaremos a qué apartados debemos acceder para realizar las operaciones.

Como ya se ha comentado, la forma de realizar los siguientes pasos en la plataforma de investing, va a ser a través de utilizar programación en Python junto con la biblioteca Selenium.

Todos los procesos tendrán lugar dentro del apartado de 'My Watchlist' de la página web, que es donde se gestionan los portfolios.

El código Python creado para acceder e interactuar con la página web de investing.com y así subir las operaciones utilizando tal lógica se explica en detalle en el Anexo A.

9.1 Creación de las tres carteras en investing.com

En primer lugar, debemos realizar la creación de las tres carteras donde guardaremos información, mediante el icono '+' -como muestra la Figura 5.6 - de la barra superior. Elegimos el tipo de carteras, que será 'holdings', ya que queremos hacer el seguimiento de los activos que poseemos, y les damos nombre: Cartera 1, Cartera 2 y Cartera 3.

El apartado de mis carteras de seguimiento quedarían de la siguiente manera después de la inicialización:

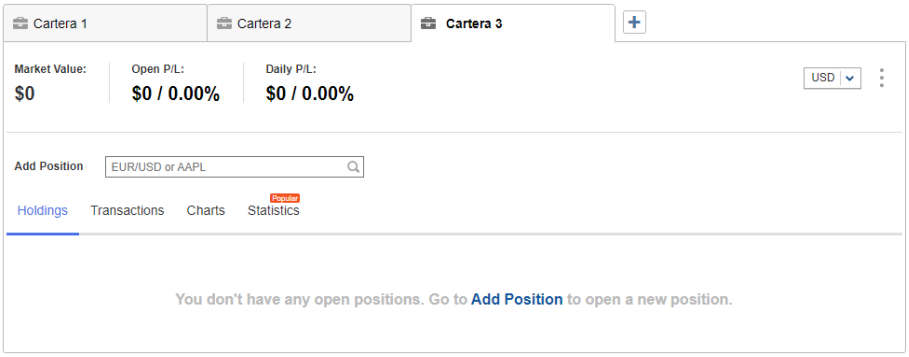


Figura 9.1: Inicialización carteras en investing.com

9.2 Carga de las posiciones iniciales a la Cartera 1

Antes de iniciar este proceso de monitorización de operaciones, puede que el usuario tenga ya posiciones existentes en el portfolio de su cuenta de IBKR. Estas posiciones reales deben estar reflejadas en la Cartera 1, como parte su inicialización, antes de comenzar el proceso de volcar diariamente las transacciones que se realicen.

En primer lugar debemos realizar la descarga del portfolio de IBKR del usuario, tal y como se explicó en la Sección 8.1. Una vez tenemos las posciones almacenadas en un archivo csv, podemos de forma sencilla añadir a la Cartera 1 todas esas posiciones mediante la carga del fichero csv en el apartado de 'Import Watchlist' (ver Figura 5.6).

Una vez seguidos los pasos necesarios para importar el fichero -explicados en la Subsección 5.2.2- ya tendremos nuestra Cartera 1 inicializada con las posiciones con las que partimos, para posteriormente actualizarlas diariamente.

9.3 Subida de las operaciones diarias a las tres carteras

A continuación se explican los pasos y condiciones seguidos para añadir cada transacción realizada según las condiciones que cumpla a cada una de las tres carteras.
Vamos a dividir las metodologías de inserción de operaciones según el tipo de operación realizada, es decir, según si se ha hecho una compra o una venta.
Las operaciones que se añaden son las descargadas del histórico diario, que guardamos en un fichero .csv en la Sección 8.2

Para poder seguir las explicaciones es muy importante tener clara la estructura de la plataforma y sus funcionalidades, todo ello explicado en la Subsección 5.2.2.

9.3.1. CARTERA 1:

Operaciones de VENTA de IBKR

Para cada operación de venta que queramos añadir, mirar si ese activo ya lo teníamos en nuestra Cartera 1 y es de tipo'BUY':

- Ya existía en nuestra cartera.
  Significa que vamos a cerrar posiciones (vender acciones) de un activo que ya teníamos

  Pinchamos en el activo para desplegar sus posiciones y pinchamos en el botón de 'Close Position' indicando la cantidad que hemos vendido y los

otros detalles.
Hay que iterar todas las posiciones del activo e ir vendiendo shares, hasta haber vendido toda la cantidad que queríamos.

Posibilidad: Vender más cantidad de la que poseemos, es decir, una parte de la venta es venta en largo y la otra parte es venta en corto.
En el caso en el que al acabar de iterar aún nos queden acciones por vender, significa que ahora toca hacer venta en corto, es decir, vender acciones que no poseemos. Por lo tanto, con esa cantidad de acciones que nos quedan por vender añadiremos una posición de tipo compra mediante el apartado de Add Position.

- No existía en nuestra cartera.
  Significa que vamos a vender algo que no teníamos (venta en corto).

  Tenemos que añdir la posición como tipo venta, desde el apartado de Add Position.

Operaciones de COMPRA de IBKR

Para cada operación de compra que queramos añadir, mirar si ese activo ya lo teníamos en nuestra Cartera 1 y es de tipo'SELL':
Comprobamos en primer lugar si existía el activo de tipo 'SELL' ya que eso indica una venta en corto. Esas cantidades son acciones que en realidad no teníamos (prestadas) y tenemos que recomprar. Por lo que si hacemos una compra de ese activo ahora, iría destinado a recomprar esas acciones.

- Ya existía en nuestra cartera con tipo 'SELL':

  Debemos hacer la recompra de esas acciones prestadas. Esto se consigue cerrando posiciones, mediante Close Position
  Hay que iterar todas las posiciones del activo e ir comprando shares, hasta haber comprado toda la cantidad que queríamos.

  Posibilidad. Cuando ya he recomprado toda la cantidad me siguen quedando acciones por comprar.
  El resto de las acciones que tenemos que comprar, las compraremos añadiendo esa posición mediante Add Position de tipo compra.

- No existía en nuestra cartera con tipo 'SELL':
  No tenemos que hacer recompra, simplemente comprar las acciones de forma normal.

  Añadimos la compra de esas acciones añadiendo la posición de tipo compra mediante el apartado Add Position.

A continuación, en la Figura 9.2, se muestra un diagrama de flujo con las condiciones explicadas anteriormente, para entender el proceso de forma más visual:

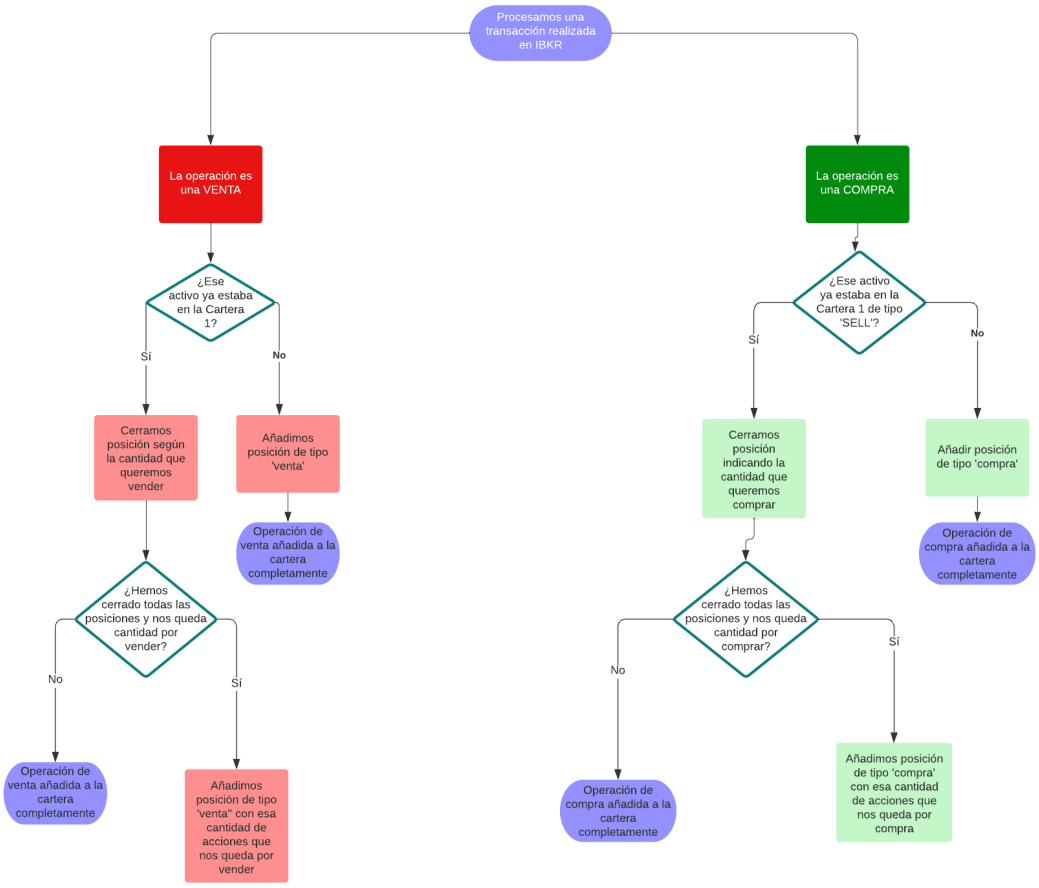


Figura 9.2: Diagrama de flujo para añadir las operaciones a la Cartera 1

9.3.2. CARTERA 2:

Recordamos que las operaciones se añadirán con el tipo inverso al de la transacción real.

Operaciones de VENTA de IBKR

Las operaciones de venta de IBKR se añaden siempre a la cartera 2, mediante el apartado de Add Position de tipo compra.

Operaciones de COMPRA de IBKR

Para cada operación de compra que queramos añadir, mirar si ese activo ya lo teníamos en nuestra Cartera 2:
Si el activo ya está en la Cartera 2 significa que lo vendimos en el pasado.

- Ya existía en nuestra cartera:
  Vamos a reflejar la recompra de esas acciones que vendimos en el pasado.

  Hay que iterar todas las posiciones del activo y cerrar la posición, mediante Close Position, según la cantidad correspondiente a las acciones que recompramos.

- No existía en nuestra cartera:
  No es una recompra.

  No hacemos nada.

A continuación, en la Figura 9.2, se muestra un diagrama de flujo de esta explicación:

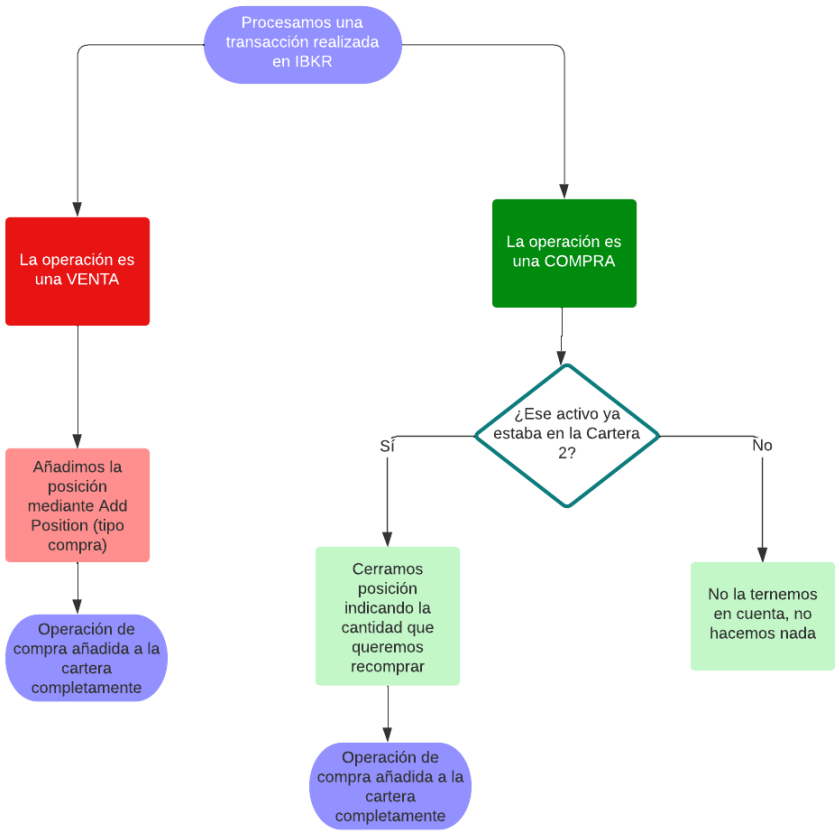


Figura 9.3: Diagrama de flujo para añadir las operaciones a la Cartera 2

9.3.3. CARTERA 3:

Para introducir las posiciones en corto en la Cartera 3 no nos fijamos directamente en las operaciones descargadas de IBKR, sino en la Cartera 1 donde previamente esas operaciones de IBKR ya han sido reflejadas.

Los pasos a seguir para añadir las posiciones cortas a la Cartera 3 son los siguientes:

1. Guardar las posiciones de tipo SELL de la Cartera 1 con toda su infromación, obteniéndolas mediante técnicas de web scraping.

2. Eliminar la Cartera 3 y volverla a crear.

3. Añadir a la Cartera 3 todas las posiciones tipo SELL que hemos obtenido de la Cartera 1.

Ya que las actualizaciones de carteras se hacen diariamente, para optimizar el proceso en términos de programación, se ha decidido borrar la Cartera 3 cada día para volver a crearla vacía y volcar las posiciones cortas de la Cartera 1 actualizdas de ese mismo día.

Otra forma que se ha planteado sin tener que eliminar la Cartera 3 y volverla a crear cada día es, a partir de las posiciones en corto de la Cartera 1 que hemos obtenido, ir iterando cada posición de la Cartera 3 para ver si esa posición corta estaba ya añadida o no y dependiendo del caso, dejar la posición tal como estaba, borrarla o añadir una posición nueva.
Esta manera de operar ha sido descartada por su alto coste al tener que tener que iterar todas las posiciones.

A continuación, en la Figura 9.4, se muestra un diagrama de flujo de los pasos a seguir:

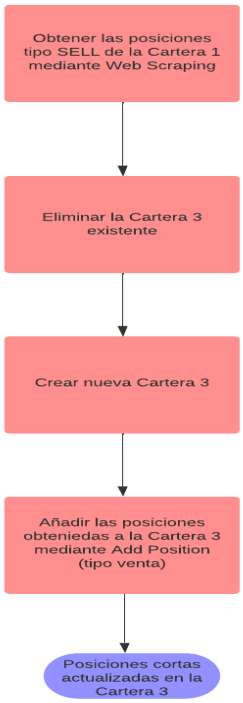


Figura 9.4: Diagrama de flujo para añadir las operaciones a la Cartera 3

CAPÍTULO 10
Chatbot para operaciones de compra y venta

En el último año nos hemos encontrado con el boom de la Inteligencia Artificial y la creación diaria de nuevas aplicaciones de inteligencia artificial han transformado diversos aspectos de la vida cotidiana, incluyendo las finanzas personales. Concretamente en el ámbito de nuestro proyecto y con las plataformas que estamos trabajando también podemos aplicar esta inteligencia artificial para ayudar a los usuarios que desean realizar operaciones a través de su bróker online.

La idea que vamos a desarrollar en este apartado es la creación de un chatbot que permita al usuario introducir mediante lenguaje natural la orden de compra o venta que quiera realizar con su cuenta de IBKR y ejecutarla.

Como ya sabemos, la plataforma de Interactive Brokers nos permite realizar una gran cantidad de operaciones mediante la ejecución de códigos Python, gracias a su API.
Por lo tanto, la forma en la que nuestro chatbot realizará las operación será a partir de la 'traducción' de la consulta del usuario a código Python.

Para realizar ese proceso haremos uso de la API de OpenAI[11], la cual es una empresa de investigación en inteligencia artificial, de la forma que explicaremos a continuación.

10.1 Procesamiento del Lenguaje Natural (PLN)

El Procesamiento del Lenguaje Natural (PLN) es una rama de la inteligencia artificial que se ocupa de la investigar la manera de comunicar las máquinas con las personas mediante el uso de lenguas naturales[55]. En la actualidad podemos encontrar técnicas de PLN implementadas en multitud de productos como pueden ser los asistentes de voz, traducción automática o chatbots y asistencia al cliente, todos ellos proporcionando mayores facilidades al usuario final.

En el caso que vamos a desarrollar, el hecho de que un usuario pueda realizar operaciones de compra y venta en bolsa a través de una simple frase, hace mucho más rápido y sencillo el trabajo de los trades, sobre todo de los principiantes. Así, el usuario se puede ahorrar el tener que entrar en la página web del broker online y el tener que lidiar con páginas complicadas y con gran cantidad de información y datos si únicamente desea realizar operaciones sencillas.

10.2 OpenAI y ChatCompletion

OpenAI es una empresa de investigación en inteligencia artificial que ha llevado a cabo el desarrollo de herramientas de procesamiento de lenguaje natural avanzado que permiten a las máquinas comprender y generar texto de manera más parecida a como lo haría un humano. Esta empresa posee múltiples herramientas que nos permiten crear nuestras propias aplicaciones.

Para la creción de nuestro chatbot vamos a hacer uso de la herramienta 'Chat completions', la cual utiliza técnicas de aprendizaje profundo para aprender a generar respuestas para la entrada del usuario, pudiendo ser entrenado en diferentes tipos de datos para adaptarse a diferentes casos de uso.

10.2.1. API de OpenAI

Podemos hacer uso de las herramientas que OpenAI pone a nuestra disposición a través de Python gracias a la API que ofrecen.
Para poder conectarnos a esta API, lo primero que tenemos que hacer es tener una cuenta en OpenAI y solicitar una API key para poder acceder a las funciones del servicio. Además, por otra parte debemos instalar la librería 'openai'.
Una vez instalada la librería y con la API key en nuestro poder, podemos utilizar la función `openai.ChatCompletion.create()` para crear nuestro chatbot personalizado. Podemos pasarle los parámetros necesarios para que el modelo de chatbot procese la conversación que queremos simular.

10.2.2. La función de ChatCompletion y sus parámetros

La función `openai.ChatCompletion.create()` tiene varios parámetros que permiten personalizar la respuesta generada por el modelo de lenguaje[10].
Los parámetros que vamos a usar para generar nuestro modelo de chatbot son:

- **model**. Especifica el modelo de lenguaje que se utilizará para generar la respuesta. Nosotros usaremos el modelo "gpt-3.5-turbo", que es uno de los modelos más avanzados y potentes que ofrece OpenAI, adecuado para aplicaciones que requieren una alta precisión y una capacidad de respuesta rápida.

- **messages**. Es el input principal. Es un array donde indicamos el rol ('system' y 'user') y el contenido (el contenido del mensaje). Esos dos roles consisten en lo siguiente:

  - El rol de 'system': es la parte del chatbot que es responsable de procesar la entrada del usuario y generar una respuesta. Aquí especificamos mediante texto cómo queremos que se comporte nuestro chatbot, qué habilidades tiene, cómo debe responder en función de las preguntas del usuario, ejemplos de conversaciones y todas las especificaciones que creamos necesarias para que la respuesta generada sea la que queremos.

  - El rol de 'user': es la consulta a la que el chatbot va a responder.

10.3 Fases para la ejecución de las operaciones

Dividiremos la creación de nuestro chatbot en tres fases. También podemos decir que el chatbot está compuesto por dos sub-chatbots.
Podemos esquematizar el funcionamiento así:

- El usuario introduce una consulta en lenguaje natural de la operación que desea realizar:
  1. Usamos la herramienta de ChatCompletion para generar el código Python que ejecuta esa orden. Para esto creamos un system llamado 'AIBroker' al que le damos las explicaciones necesarias para realizarlo, tanto información sobre la TWS API como ejemplos.
  2. Usamos la herramienta de ChatCompletion para generar una explicación de lo que hace el código generado en el paso anterior. Para esto creamos un system llamado 'TWSAPIexplanator' al que le damos las explicaciones necesarias para realizarlo y ejemplos de ejecución.
  3. Ejecutamos el código Python.

Todo esto se hace mientras que el usuario tiene la aplicación de escitorio Trader Workstation abierta en su ordenador, para que la API pueda relacionar qué cuenta está realizando la operación.

A continuación explicamos con más detalle las distintas fases.

10.3.1. Generación de código Python a partir de la consulta en lenguaje natural

En esta fase transformamos la consulta introducida por el usuario a un código Python que, a través de la TWS API, realiza esa misma operación.

Esto, lo conseguimos haciendo uso de la herramienta ChatCompletion. En el rol de 'user' se introduce la consulta del usuario, es decir, la operación que quiere realizar. Por otro lado, en el rol del 'system', para que la herramienta de sea capaz de crear el código de forma adecuada hemos creado un system al que hemos llamdo 'AIBroker' al cual le hemos explicado mediante texto todas las habilidades y capacidad que tiene. Algunas de estas capaciadades son:

- Es un experto de Python e Interactive Brokers.
- Tiene conocimiento de las posibles operaciones, funciones y clases de la TWS API.
- Las respuestas que proporciona deben ser solo de código Python.
- Genera el codigo Python necesario para realizar la operación de compra o venta que pide el usuario.
- Se le proporcionan también varios ejemplos de ejecución con operaciones de compra y de venta sencillas.

En el script de nuestra aplicación Python, esto se implementa en una función a la que hemos llamado `crear_codigo_operacion(usuario_input)`, que toma como input la consulta del usuario y devuelve como output código Python.

10.3.2. Explicación de la operación que realiza el código Python generado

En esta fase se genera una explicación concisa de la operación que se realiza en el código generado en el paso anterior.
Esta fase es de gran importancia, ya que sirve de doble autenticación. Puede ser que el código generado esté incorrecto y no sea exactamente la operación que nosotros queremos realizar.

Esto, lo conseguimos haciendo uso de la herramienta ChatCompletion. En el rol de 'user' se introduce el código Python generado en el paso anterior. Por otro lado, en el rol del 'system', para que la herramienta sea capaz de crear el código de forma adecuada hemos creado un system al que hemos llamdo 'TWSAPIexplanator', al cual le hemos explicado mediante texto todas las habilidades y capacidad que tiene. Algunas de estas capaciadades son:

- Es un experto de Python e Interactive Brokers.
- Tiene conocimiento de las posibles operaciones, funciones y clases de la TWS API.
- Las respuestas que proporciona deben ser solo en lenguaje natural.
- Devuelve una explicación corta y concisa sobre la operación que realiza el código Python, no explica todo el código.
- Se le proporcionan también varios ejemplos de ejecución con operaciones de compra y de venta sencillas.

En el script de nuestra aplicación Python, esto se implementa en una función a la que hemos llamado `explicacion_codigo(code_input)`, que toma como input el código de la anterior función y devuelve como output la explicación en texto.

10.3.3. Ejecución de la operación

En esta fase no se hace uso de ninguna herramienta de OpenAI, únicamente consiste en ejecutar el código generado.

10.4 Implementación en Flask

Flask es un framework escrito en Python que permite crear aplicaciones de forma sencilla y rápida[56]. En esta sección, se describe cómo se han integrado las funciones de Python anteriormente descritas en una aplicación Flask con una interfaz de usuario, la cual proporciona a los usuarios una forma sencilla de interacción con el chatbot y les permite ejecutar las operaciones de compra y venta en IBKR.
Para crear la interfaz, utilizamos la función `render_template` que es una función que se encarga de presentar una plantilla en el texto realizado con HTML[57].

10.4.1. Aplicación Flask y rutas

Para crear la aplicación Flask utilizamos la biblioteca Flask de Python. Esta aplicación recibe la entrada del usuario a través de un formulario HTML y procesa la entrada utilizando las funciones desarrolladas anteriormente.

En una primera sección de la app tenemos definidas las dos funciones explicadas en en apartado anterior: `crear_codigo_operacion(usuario_input)` y `explicacion_codigo(code_input)`.

En la siguiente sección se definen las dos rutas principales de la aplicación. La primera es la ruta raíz / y la siguiente es la ruta `/results`.

- Ruta raíz **" / "**
  Carga el archivo index.html que muestra la página principal de la aplicación, con el botón de inserción de texto.
- Ruta **"results "**
  Se utiliza para procesar las consultas del usuario y mostrar los resultados en la página web.

  Mediante las funciones creadas comentadas, al recibir la consulta en la ruta raíz, se genera el código Python, la aplicación y se ejecuta la transacción.

  En la interfaz queremos mostrar tanto el código como la explicación, por lo que, finalmente se devuelve, mediante la función `render_template`, la página `results.html` con esos resultados.

10.4.2. Templates

Vamos a explicar con más detalle los templates HTML creados que usa la aplicación web para que el usuario pueda enviar una consulta de operación de compra o venta a través de un formulario y mostrar los resultados en una página de resultados.

El primer archivo, `index.html`, es la página de inicio que contiene el formulario donde el usuario introduce su consulta. Se basa en un botón de input de texto y debajo un botón de submit, para que se empiece a generar el resultado.
Una vez generado el resultado, además de ejecutarse la venta o compra de IBKR, se mostrará en la siguiente página la información relacionada, `results.html`.

El segundo archivo, `results.html`, es la página de resultados que muestra los resultados de la consulta del usuario, dividia en dos secciones:

- En una primera caja se incluye la explicación del código utilizado para realizar la operación.
- En una segunda caja un botón para mostrar el código completo en el caso de que el usuario deseara visualizarlo. Esto es una gran herramienta para ayudar al usuario a introducirse a la programación y que tenga estos scripts como base por si deseara editarlos incluyendo cambios.

En la siguiente sección, podremos ver las interfaces que se generan con esos documentos HTML.

10.4.3. Cómo ejecutar la aplicación Flask

La aplicación Flask creada se ejecuta de forma local, es decir, se ejecuta en la propia máquina del usuario que vaya a ejecutarla, en lugar de estar en un servidor en línea. Para acceder a la aplicación, debes abrir un navegador y escribir la siguiente dirección en la barra de direcciones: http://127.0.0.1:5000.

Pero, para que se pueda establecer la conexión entre el navegador y la aplicación Flask a través de la dirección comentada. Debemos ejecutar la app introduciendo este

comando en la terminal de nuestro editor de código: `flask app run`. Una vez ejecutado el comando, Flask iniciará el servidor y la aplicación estará disponible en la dirección comentada, http://127.0.0.1:5000.

En esta dirección http://127.0.0.1:5000:

- 'http' es el protocolo de comunicación utilizado para transferir datos entre el navegador y la aplicación.
- '127.0.0.1' es la dirección IP especial de localhost, la cual hace referencia al servidor en el propio equipo. [58]
- El número '5000' se refiere al puerto en el que se está ejecutando la aplicación Flask. Es, por defecto, el servidor que viene con Flask.[59]

Al ingresar la dirección 'http://127.0.0.1:5000' en el navegador se muestra la interfaz de la aplicación, donde ya podemos a ejecutar consultas.

10.5 Ejemplo de ejecución

Vamos a mostrar, a través de capturas de pantalla, un ejemplo de ejecución la aplicación chatbot creada. Para este ejemplo vamos a realizar la compra de 5 acciones de la empresa META.

1. Iniciamos la aplicación Flask mediante el comando `flask run` en la línea de comandos. Además, nos indica en qué dirección está corriendo: http://127.0.0.1:5000.

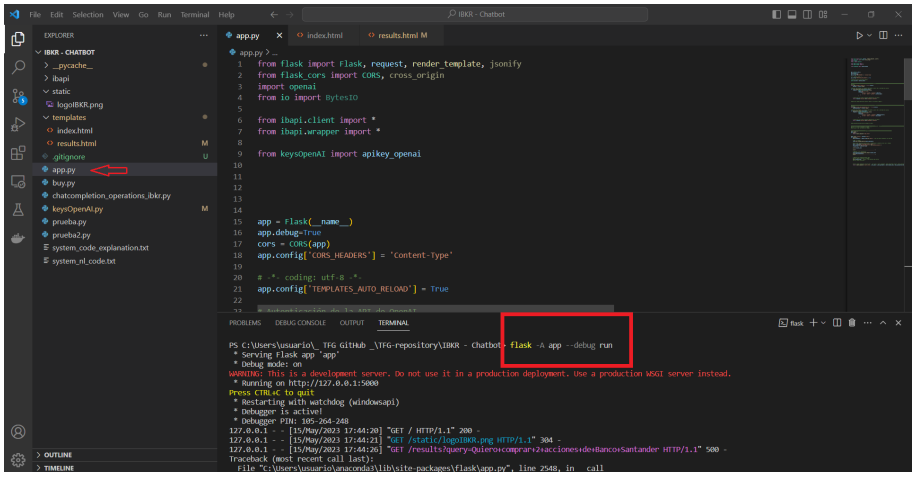


Figura 10.1: Inicio de la aplicación flask

2. Ya tenemos disponible la aplicación en el servidor local. Accedemos a la página principal de la aplicación mediante la dirección.

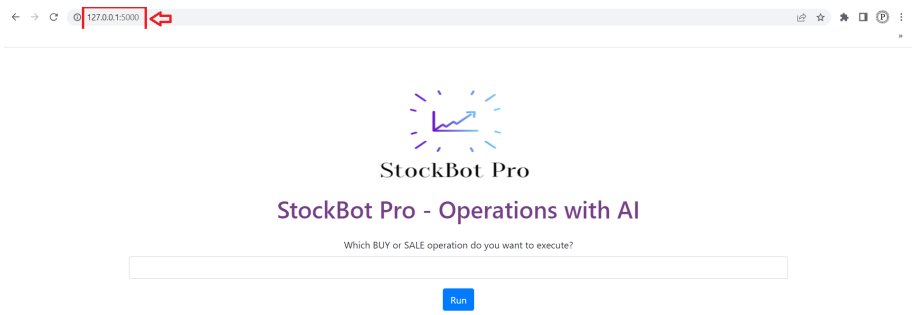


Figura 10.2: Ventana incial chatbot

3. Introducimos la compra que deseamos realizar en el botón de entrada de texto y ejecutamos la consulta mediante el botón 'Run'.

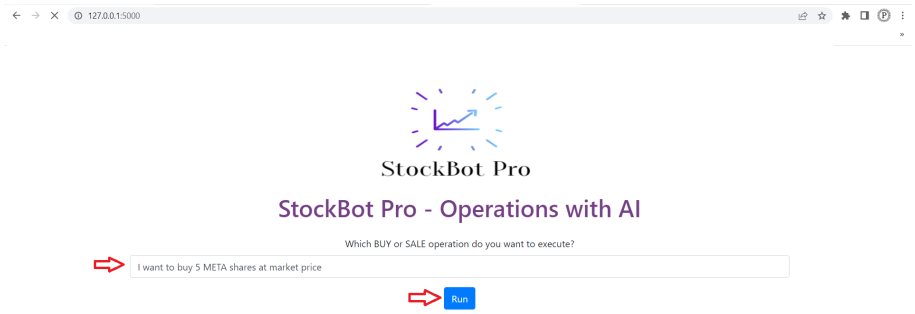


Figura 10.3: Introducir y ejecutar operación

4. Se realiza la conulta y en el navegador se nos muestra la explicación y un botón por si quisiéramos observar el código Python utlizado.

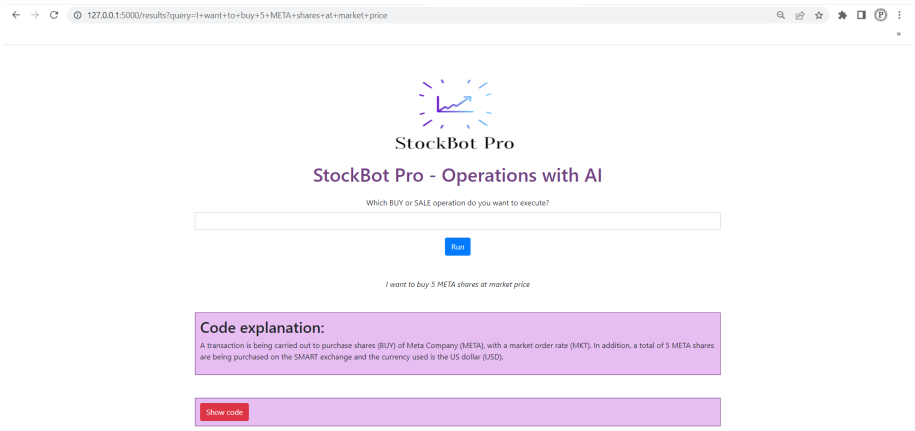


Figura 10.4: Resultados consulta

5. Al pinchar en el botón de 'Show code' se nos muestra el código utilziado.

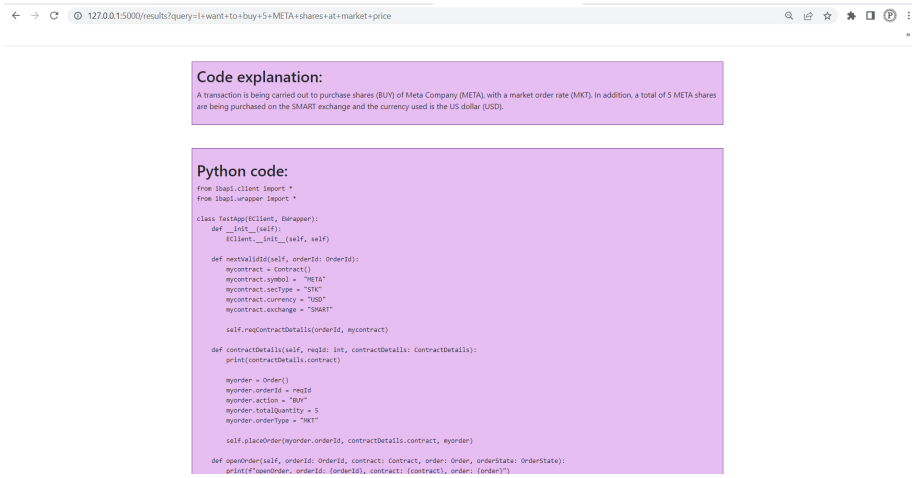


Figura 10.5: Muestra de código utilizado en la operación - Parte 1

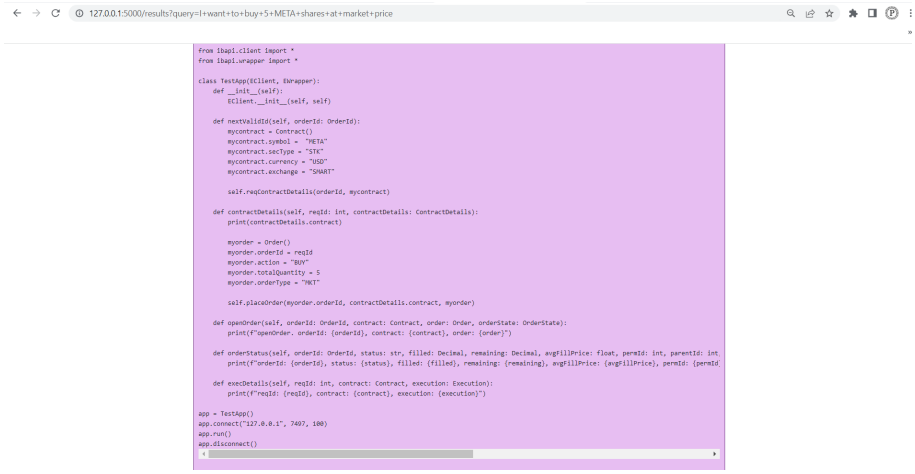


Figura 10.6: Muestra de código utilizado en la operación - Parte 2

6. Comprobamos mediante la terminal como mediante la aplicación de escritorio de Trader Workstation que la operación se ha realizado.

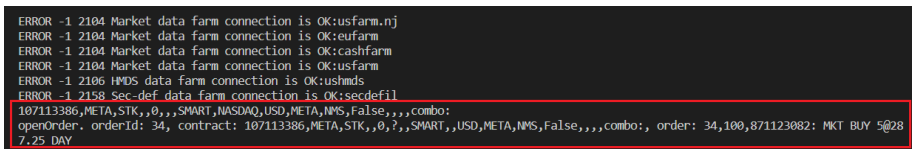


Figura 10.7: Datos de la operación realizada en la terminal

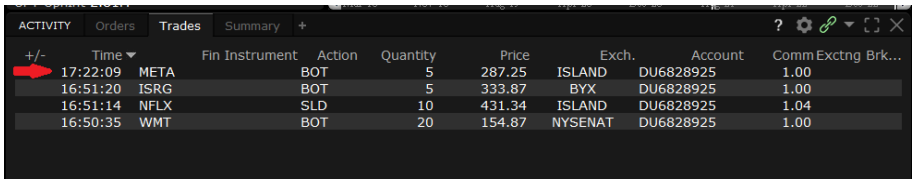


Figura 10.8: Datos de la operación realizada en TWS

CAPÍTULO 11
Reproducibilidad de la aplicación

En los capítulos anteriores se ha explicado el desarrollo de los algoritmos, funciones y programas necesarios para crear las dos funcionalidades de este proyecto, por una parte la aplicación de monitorización de operaciones del mercado de valores y por otra parte el chatbot ejecutor de operaciones.

En este capítulo se va a comentar detalladamente cómo a partir de todos los scripts de Python creados, una tercera persona que quisiera utilizar las aplicaciones desarolladas puede ejecutarlas de manera local en su ordenador.

Todo el código desarrollado está disponible públicamente en este repositorio de Git Hub.

Prerrequisitos generales

Para realizar el despligue con los pasos que se comentan en las siguientes secciones, en primer lugar es necesario que el usuario cumpla con los siguientes prerrequisitos en su ordenar:

- Tener Python instalado.
- Tener descargado un editor de código que nos permita abrir los scripts de Python, como podría ser Visual Studio Code.
- Clonar el repositorio de GitHub donde se encuentra el código necesario, al que se puede acceder aquí.
- Tener descargada la aplicación de escritorio Trader Workstation.

En el repositorio de GitHub tenemos dos carpetas diferentes, 'Aplicación monitorización operaciones' y 'Chatbot para transacciones', que hacen referencia a los dos programas diferentes desarrollados.

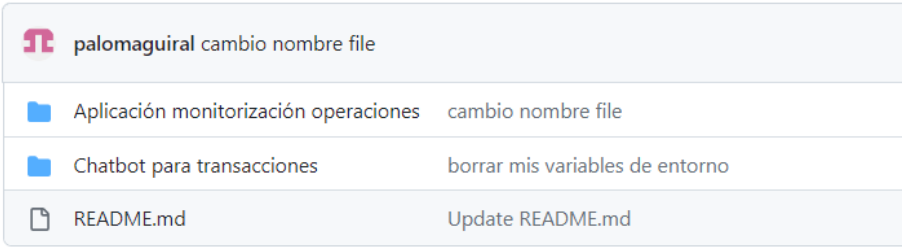


Figura 11.1: Estructura principal del repositorio de GitHub

Se explica en los siguientes apartados el despligue cada uno de estos programas de manera separada.

11.1 Aplicación monitorización operaciones

Dentro de la carpeta 'Aplicación monitorización operaciones' encontramos todo lo necesario para mantener la monitorización de las operaciones que realice el usuario en su bróker.

Para realizar los pasos que se explican a continuación es necesario estar dentro de esta carpeta 'Aplicación monitorización operaciones'.

Prerrequisitos

- Estar registrado en Interactive Brokers e investing.com.
- Tener abierta la aplicación de escritorio Trader Workstation durante las ejecuciones.
- Instalar las dependencias presentes en el fichero 'requirements.txt'.

Este programa está divido en un programa para inicilizarlo en la primera ejecución (carpeta 'primera ejecución') y otro programa para hacer la monitorización diaria de las operaciones (carpeta 'ejecución diaria'). Se explica el funcionamineto de ambos programas por separado.

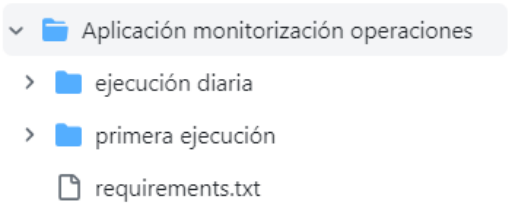


Figura 11.2: Estructura carpeta 'Aplicación monitorización operaciones' del repositorio de GitHub

11.1.1. Primera ejecución

Dentro de esta carpeta encontramos encontramos el código necesario para hacer la inicialización de la monitorización, creando las 3 carteras de investing e inicializando las posiciones de la cartera 1.

El script principal es `app_firstExecution.py`, y el otro script `dependencies.py` contiene las funciones creadas que son necesarias para su funcionamiento.

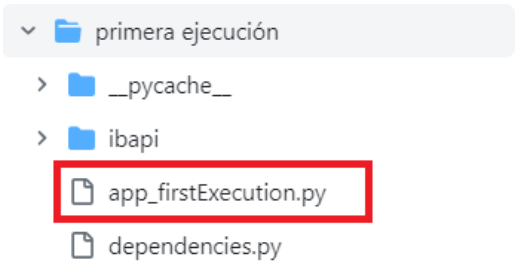


Figura 11.3: Estructura carpeta 'primera ejecución' del repositorio de GitHub y script principal

Los pasos a seguir son:

1. Crear un fichero .env donde se indiquen las credenciales del usuario de investing.com, de la siguiente forma:
   `USER_INVESTING = NOMBRE DE USUARIO`
   `PASSWORD_INVESTING = CONTRASEÑA`

2. Ejecutar el fichero `app_firstExecution.py`

11.1.2. Ejecución diaria

En esta carpeta se encuentra el código necesario para que diariamente se descarguen las operaciones realizadas IBKR y se carguen a las distintas carteras de investing.com. El script principal es `app_dailyExecution.py`, y el otro script `dependencies.py` contiene las funciones creadas que son necesarias para su funcionamiento.

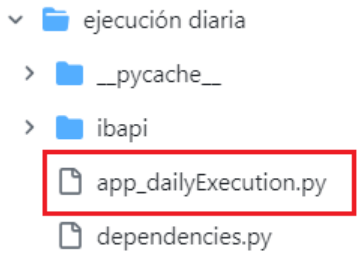


Figura 11.4: Estructura carpeta 'ejecución diaria' del repositorio de GitHub y script principal

Se explican los pasos necesarios para programar la ejecución diaria del programa en local, diferenciando según el sistema operativo del ordenador del usuario:

- Windows

  1. Abrir el 'Programador de tareas' de Windows
  2. Clic en 'Crear tarea básica' en el panel de la derecha. Se abrirá el asistente para crear tareas:

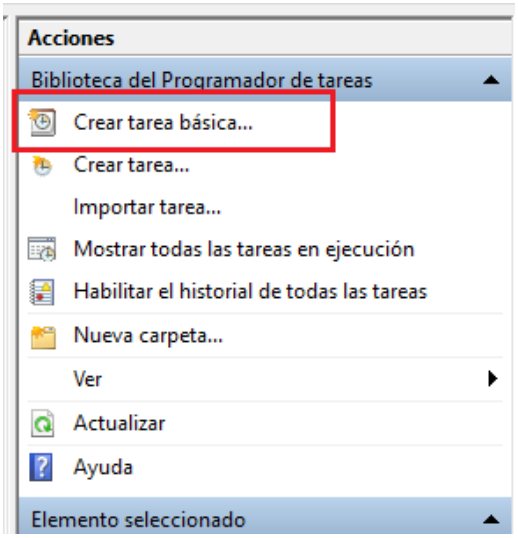


Figura 11.5: Programador de tareas de Windows - Crear tarea

a) Proporcionar un nombre y una descripción opcional para la tarea

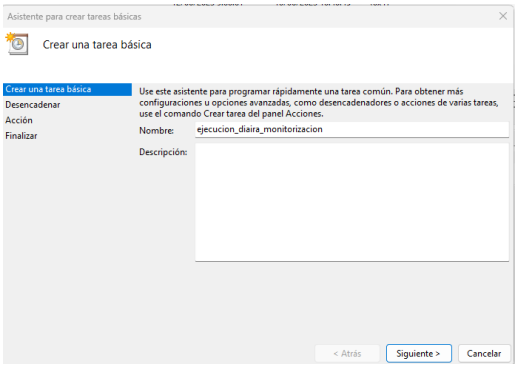


Figura 11.6: Programador de tareas de Windows - Dar nombre a la tarea

b) Elegir la frecuencia con la que se ejecuta la tarea. Elegimos 'Diariamente'.

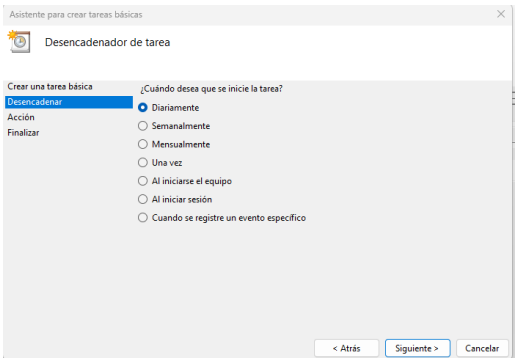


Figura 11.7: Programador de tareas de Windows - Frecuencia diaria

c) Configurar la fecha y la hora en la que se ejecutará la tarea.

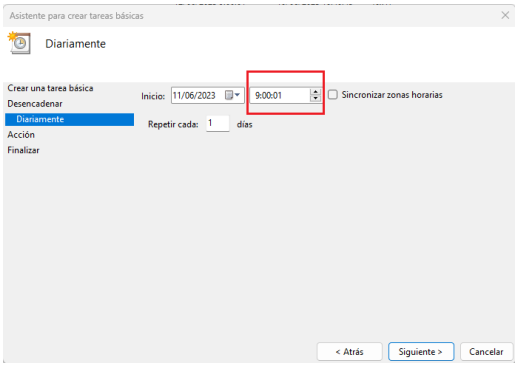


Figura 11.8: Programador de tareas de Windows - Hora

d) Elegir como acción 'Iniciar un programa'.

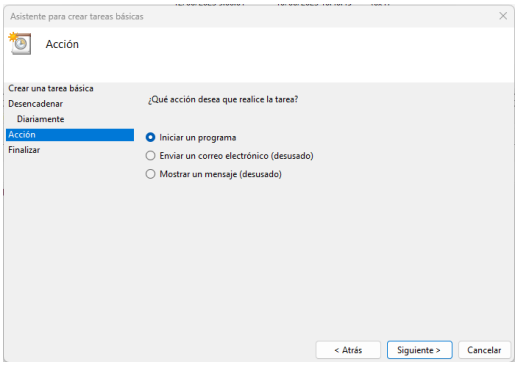


Figura 11.9: Programador de tareas de Windows - Iniciar programa

e) Indicamos el programa que queremos que se ejecute (`app_dailyExecution.py`) e indicamos también que queremos ejecutarlo con Python, en los siguientes campos:

- Programa o script: ruta al ejecutable de Python en nuestro ordenador (python.exe)
- Agregar argumentos: app_dailyExecution.py
- Iniciar en: ruta a la carpeta 'ejecución diaria'

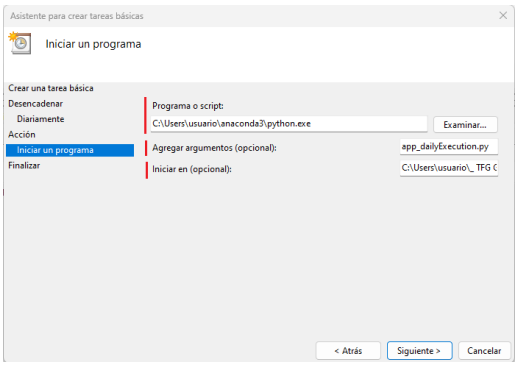


Figura 11.10: Programador de tareas de Windows - Programa a ejecutar

f) Finalizar.

- MacOS Se utiliza el programador de tareas llamado 'launchd' [60].
  1. Crear archivo plist (Property List), especificando la configuración de la tarea, como el intervalo de tiempo y el comando para ejecutar el script de Python.
  2. Guardar el archivo plist en el directorio ' /Library/LaunchAgents'.
  3. Carga la tarea en launchd utilizando el comando 'launchctl load /Library/-LaunchAgents/nombre_del_archivo.plist'
- Linux Se utiliza el programador de tareas cron [61].
  1. Abrir una terminal y ejecutar el comando 'crontab -e' para editar las tareas programadas
  2. Agregar una línea en el archivo cron con la sintaxis 'minuto hora * * * `ruta_al_interprete_python` y `ruta_al_script` '

11.2 Chatbot para ejecutar transacciones

Dentro de la carpeta 'Chatbot para transacciones' tenemos todo lo necesario para desplegar el chatbot que permite efectuar operaciones.
El script principal es la flask app,`app.py`, y los demás archivos o templates son necesarios para su funcionamiento.

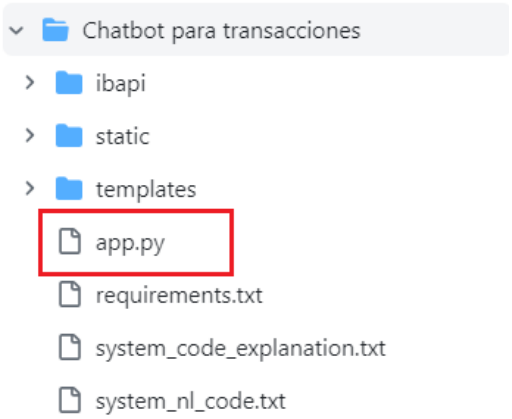


Figura 11.11: Estructura de la carpeta 'Chatbot para transacciones' del repositorio de GitHub y script principal

Para realizar los pasos que se explican a continuación es necesario estar dentro de esta carpeta 'Chatbot para transacciones'.

Prerrequisitos

- Estar registrado en Interactive Brokers.
- Tener abierta la aplicación de escritorio Trader Workstation durante las ejecuciones.
- Instalar las dependencias presentes en el fichero 'requirements.txt'.
- Estar registrado en OpenAI y obtener una API Key.

Los pasos a seguir para levantar el ChatBot en local son:

1. Crear un fichero .env donde se indique la API Key de OpenAI, así:
   `APIKEY_OPENAI = TU API KEY DE OPENAI`
2. Ejecutar el script 'app.py'. Esta ejecución se puede hacer desde la terminal con 'python ruta/app.py' o mediante el editor de código.
3. Acceder a la aplicación flask en el navegador web y visitando la URL 'http://localhost:5000/', ya que se estará ejecutando en local.

CAPÍTULO 12
Conclusiones

Una vez finalizado el desarrollo del trabajo la forma de poder extraer conclusiones es comprobar si se han cumplido los objetivos que nos propusimos y que quedan mencionados en la Sección 1.2.

Después de realizar un análisis global de todo lo realizado, podemos confirmar que se han cumplido todos los objetivos marcados.
Sin embargo, se han encontrado algunos problemas comentados a continuación en este mismo capítulo que, aunque no hayan imposibilitado el desarrollo del proyecto, hay que tener en cuenta.

El logro de las metas planteadas ha sido posible gracias a la aplicación de diferentes áreas de conocimiento, habiendo sido clave la itegración de las habilidades de programación, conocimientos de algorítmica y optimización de código y funciones.
Además, cabe destacar que el desarrollo no hubiera sido posible sin el aprendizaje de nuevas librerías, profundización en el conocimiento de teoría financiera o búsqueda de nuevas soluciones tecnológicas que antes eran desconocidas para el alumno.

12.1 Legado

La aplicación creada busca proporcionar a los traders una herramienta integral que les ayude a realizar un análisis más efectivo de sus inversiones y a mantener un control preciso sobre ellas, para así brindar una visión clara de sus inversiones e identificar patrones y tendencias. A partir de esos análisis podrán tomar la decisión más adecuada según sus estrategias y el momento de tiempo, con el objetivo final de maximizar los beneficios económicos.

Por otro lado, el desarrollo del chatbot, agrega facilidad y rapidez para ejecutar operaciones mediante consultas en lenguaje natural, además de poder al mismo tiempo servir como herramienta de apredizaje en progrmación en Python relacionada a las finanzas.

Además, se proporciona acceso al código fuente en un repositorio abierto de GitHub, lo que permite a terceros reproducir el análisis y obtener resultados similares con sus propias cuentas de trading.

12.2 Limitaciones y problemas encontrados

En esta sección se describen los aspectos específicos de la aplicación que presentan deficiencias o limitaciones conocidas y cómo afectan a su funcionamiento.

- Dependencia de una conexión a internet estable para la ejecución del programa.

  La estabilidad de la conexión a internet es crucial para garantizar el correcto funcionamiento y la obtención precisa de datos en el programa, tanto en la comunicación con Interactive Brokers a través de la TWS API como en la interacción con la plataforma de investing.com mediante Selenium. En cuanto a esto último, si la página web de investing.com carga de manera lenta debido a una conexión deficiente, es posible que Selenium no encuentre los elementos necesarios para realizar las acciones programadas, lo que puede provocar errores en la ejecución de la aplicación.

- Dependencia de la Trader Workstation (TWS) Desktop App para establecer la conexión con la TWS API, ya que es necesario tener instalada y ejecutándose la TWS Desktop App, que es la aplicación proporcionada por Interactive Brokers para acceder a sus servicios de trading.

- Confusión de activos al migrar operaciones de IBKR a la cartera de investing.com debido a nombres ligeramente diferentes.

  En el proceso de migración, aunque se dé en pocos casos, nos podemos encontrar con situaciones en las que el nombre de un activo en IBKR difiere ligeramente del nombre correspondiente en investing.com. Por ejemplo, en IBKR el activo puede tener el nombre "NH HOTEL GROUP SA", mientras que en investing.com se le conoce como "NH HOTELES".

  Esta diferencia en la nomenclatura puede generar problemas al realizar operaciones de cierre de posición en la cartera de investing.com. Al intentar cerrar una posición de un activo en particular, la aplicación buscará el nombre del activo en la tabla correspondiente de investing.com. Sin embargo, dado que el nombre utilizado en IBKR no coincide exactamente con el nombre en investing.com, la aplicación no detectará correctamente la posición que se desea cerrar.

  Este error solo es posible solucionarlo a la hora de añadir una posición, mediante el apartado de Add Position, ya que en lugar del nombre del activo buscamos su ISIN (identificación internacional dado a las acciones para facilitar las transacciones bursátiles y evitar confusiones) [62] para que no haya confusión.

12.3 Relación del trabajo desarrollado con los estudios cursados

La realización de este proyecto de fin de grado, se encuentra estrechamente relacionado con los conocimientos adquiridos a lo largo de mis estudios en el Grado en Ciencia de Datos. A través de él, he tenido la oportunidad de aplicar y combinar los diversos conceptos y habilidades que he adquirido durante mi formación académica.

Para lograr los objetivos marcados, he aplicado varios conocimientos clave de mi grado:

1. Programación: Uso de Python como lenguaje principal de programación para implementar de manera efectiva las funcionalidades necesarias tanto para acceder a las diferentes plataformas, como para crear las funciones y la aplicación creando los algoritmos necesarios para el manejo de las operaciones.

2. Acceso a Datos: Capacidad de acceder y extraer datos de diferentes fuentes.

3. Conexión entre Aplicaciones: Establecer conexión entre las aplicaciones utilizadas para volcar datos y darle valor a la información extraída.

4. Automatización de procesos: Creación de las funciones, clases y programas que permiten automatizar procesos como la extracción y carga de datos.

5. Uso de técnicas de Inteligencia Artificial y Procesamiento del Lenguaje Natural.

Además de los conocimientos técnicos específicos, el desarrollo de este proyecto ha requerido el uso de competencias transversales que he adquirido a lo largo de mis estudios, como:

1. Capacidad de Investigación: Para comprender el mercado de valores y las operaciones financieras, realicé una profunda investigación sobre los conceptos clave, las plataformas utilizadas y la conexión con las tecnologías.

2. Pensamiento Analítico: La capacidad de analizar y comprender los datos obtenidos de las operaciones financieras ha sido fundamental para entender qué información útil queremos proporcionar a los usuarios como ayuda a sus análisis.

3. Resolución de Problemas: Enfrentamiento a problemas técnicos y obstáculos que he tenido que superar utilizando técnicas de resolución de problemas y pensamiento crítico.

CAPÍTULO 13
Trabajos futuros

En este capítulo se presentan las mejoras de funcionalidades del trabajo realizado que se pretenden abordar y solucionar en un futuro.

13.1 Despliegue en producción de la aplicación

En un futuro, se pretende facilitar el despliegue de la aplicación en un entorno de producción, lo que permitirá a los usuarios ejecutarla de manera más sencilla y sin la necesidad de configurar su entorno local. El despliegue en producción tiene como objetivo proporcionar una experiencia más fluida y accesible para los usuarios, eliminando la necesidad de cumplir con prerrequisitos específicos y simplificando el proceso de instalación.

Estas mejoras facilitarán a los usuarios la ejecución de la aplicación sin tener que lidiar con la configuración de su entorno local, lo que les permitirá centrarse en utilizarla y aprovechar sus funcionalidades de manera más eficiente.

13.2 Aumento de funcionalidades del chatbot

Además de la posibilidad de realizar operaciones de compra y venta implementadas en el chatbot, es posible aumentar su funcionalidad y proporcionar a los usuarios una experiencia aún más completa, gracias a que la TWS API nos permite realizar gran variedad de consultas y operaciones. Algunas de las posibles mejoras y características que se podrían implementar en futuras versiones del chatbot son: consulta de información de activos en tiempo real, consultas de cartera y operaciones avanzadas como órdenes de stop-loss u órdenes de límite.

Es importante destacar que estas implementaciones adicionales requerirían tiempo y esfuerzo para entrenar adecuadamente el modelo del chatbot y garantizar su precisión en la ejecución de las operaciones.

Bibliografía

[1] Estrategias de inversión. ¿qué es y cómo funciona la bolsa de valores? https://www.estrategiasdeinversion.com/actualidad/noticias/bolsa-espana/que-es-y-como-funciona-la-bolsa-de-valores-n-578613, 2022. Fecha de acceso: 06 abril 2023.

[2] Patricia Ruiz Guevara. Guía básica para principiantes: ¿qué es eso de invertir en bolsa. https://www.santander.com/es/sala-de-comunicacion/dp/guia-basica-para-principiantes---que-es-eso-de-invertir-en-bolsa. Fecha de acceso: 5 abril 2023.

[3] Ismael de la Cruz. Qué es una acción en bolsa y cómo funciona. https://es.investing.com/academy/acciones/que-es-una-accion-en-bolsa-y-como-funciona/, 2022. Fecha de acceso: 10 abril 2023.

[4] Amazon AWS. ¿qué es python? https://aws.amazon.com/es/what-is/python/. Fecha de acceso: 5 Mayo 2023.

[5] Admirals. ¿qué es un broker? ¿cuáles son sus características? *Admirals Market*, Enero 2023.

[6] Jose V. Gascó. Interactive brokers españa opiniones 2023: seguridad, mercados, comisiones y review. *Rankia*, Abril 2023.

[7] David Fraile. Tutorial y análisis de investing: Cómo funciona y herramientas destacadas. https://youtu.be/dg0a47w_XyI, 2021. Fecha de acceso: Mayo 2023.

[8] IBKR Traders' AcademyIBKR Campus Finance Courses and Lessons. Python tws api. https://ibkrcampus.com/trading-course/python-tws-api/.

[9] Erik Anaya. ¿qué es selenium y para qué sirve? https://inmediatum.com/blog/piensa-digital/que-es-selenium-y-para-que-sirve/, 2020. Fecha de acceso: 9 Mayo 2023.

[10] OpenAI. Chat completions. https://platform.openai.com/docs/guides/chat. Fecha de acceso: Marzo 2023.

[11] OpenAI. Welcome to the openai platform. https://platform.openai.com/. Fecha de acceso: Marzo 2023.

[12] Ben Lutkevich. Definition github. https://www.techtarget.com/searchitoperations/definition/GitHub, 2023. Fecha de acceso: 10 Junio 2023.

[13] SANDRA GARRIDO SOTOMAYOR. Cómo usar cron para automatizar tareas en ubuntu 18.04. LasmetodologÃŋasÃągilesmÃąsutilizadasysusventajasdentrodelaempresa, 2021. Fecha de acceso: 8 Mayo 2023.

[14] Escuela Técnica Superior de Ingeniería Informática. Trabajo fin de grado de ciencia de datos - estructura y contenidos recomendados. https://www.inf.upv.es/www/etsinf/wp-content/uploads/2021/10/Estructura-y-Contenido-de-un-TFG-GCD_CAT02092021.pdf, 2022. Fecha de acceso: 5 Febrero 2023.

[15] Senanu R. Okuboyejo and Daniel O. Adeyoju. Design and implementation of a stock market monitoring system: A case study of the nigerian stock exchange. *International Journal of Multidisciplinary and Current Research*, page 5, Diciembre 2013.

[16] S.R. Nanda, B. Mahanty, and M.K. Tiwari. Clustering indian stock market data for portfolio management. *Expert Systems with Applications*, 37(12):8793–8798, 2010.

[17] Kazuhiro Seki Kuniaki Uehara Akira Yoshihara, Kazuki Fujikawa. Predicting stock market trends by recurrent deep neural networks. page 2, 2014.

[18] Samudith Nanayakkara, Ashen Wanniarachchi, and Dushyanthi Vidanagama. Adaptive stock market portfolio management and stock prices prediction platform for colombo stock exchange of sri lanka. *2021 5th SLAAI International Conference on Artificial Intelligence (SLAAI-ICAI)*, pages 1–6, Diciembre 2021.

[19] Matthew Cheung. How chatgpt and chatbots are transforming financial markets. https://ipushpull.com/blog/how-chatgpt-and-chatbots-are-transforming-financial-markets, 2023. Fecha de acceso: 4 Junio 2023.

[20] News Room Bank of America. Banck of america. company overview. https://newsroom.bankofamerica.com/content/newsroom/company-overview.html, 2023. Fecha de acceso: 8 Junio 2023.

[21] JPMorgan Chase Co. Who we are. https://www.jpmorganchase.com/about, 2023. Fecha de acceso: 8 Junio 2023.

[22] Goldman Sachs. About us. https://www.goldmansachs.com/about-us/, 2023. Fecha de acceso: 8 Junio 2023.

[23] Mark E. Hughes Ingrid E. Fisher, Margaret R. Garnsey. Natural language processing in accounting, auditing and finance: A synthesis of the literature with a roadmap for future research. https://doi.org/10.1002/isaf.1386, 2016. Fecha de acceso: 4 Junio 2023.

[24] Aryan Bajaj, N Preethi, Benny J Godwin, and Fr Jossy P George. An intelligent stock market automation with conversational web based build operate transfer (bot). *2022 International Conference on Trends in Quantum Computing and Emerging Business Technologies (TQCEBT)*, pages 1–6, 2022.

[25] A Poveda González. Diseño de un portal web de gestión de carteras de acciones. http://hdl.handle.net/10251/11985, 2011. Fecha de acceso: 7 Junio 2023.

[26] MM. García Martínez. Virtual stock trading. http://hdl.handle.net/10251/10200, 2010. Fecha de acceso: 7 Junio 2023.

[27] A. Navarro Hawach. Predicting stock prices using long short-term memory models. http://hdl.handle.net/10251/185563, 2022. Fecha de acceso: 7 Junio 2023.

[28] JJ. Cuéllar Abril. Forecasting exchange rates using recurrent neural networks. http://hdl.handle.net/10251/150383, 2020. Fecha de acceso: 7 Junio 2023.

[29] H. Jover Ibáñez. Desarrollo de un chatbot para la recomendación de eventos o lugares de interés. http://hdl.handle.net/10251/133836, 2019. Fecha de acceso: 7 Junio 2023.

[30] ÁJ. Egea Ruiz. Desarrollo de un chatbot para la resolución de dudas sobre devonfw. http://hdl.handle.net/10251/125890, 2019. Fecha de acceso: 7 Junio 2023.

[31] Daniella Terreros. Qué es trello, para qué sirve y cómo funciona. https://blog.hubspot.es/marketing/que-es-trello, 2022. Fecha de acceso: 24 Mayo 2023.

[32] Paula Nicole Roldán. Bolsa de valores - definición, qué es y concepto. https://economipedia.com/definiciones/bolsa-de-valores.html, 2017. Fecha de acceso: 06 abril 2023.

[33] Acciones y Valores. Productos que se negocian en bolsa. https://accionesyvalores.es/productos-se-negocian-bolsa/, 2018. Fecha de acceso: 12 abril 2023.

[34] BlackRock. ¿qué es un bono? https://www.blackrock.com/co/educacion/como-invertir-en-bonos. Fecha de acceso: 10 abril 2023.

[35] renta4banco. ¿cómo funciona un fondo de inversión? https://www.r4.com/fondos-de-inversion/como-funciona-un-fondo-de-inversion. Fecha de acceso: 11 abril 2023.

[36] Eva Blanco Garzón. Cómo aprender a ser un buen trade | rutinas y trucos. *Admiral Markets*, Septiembre 2022.

[37] Eva Robledo. Qué es un broker, tipos de broker y todo lo que debes saber. *Escuela profesional de traders*, 2022.

[38] Josep García. ¿qué broker online se adaptará mejor a tu perfil? *Mejor Banco*, Abril 2023.

[39] Jon P. Benito. Mejores brokers online para invertir en bolsa. *Robo Advisors*.

[40] Víctor Galán. Los 9 mejores brokers para trading de 2023. *Rankia*, Marzo 2023.

[41] Pedro Braz. Best brokers with api access: Api trading platforms reviewed. *Investing in the web*, Marzo 2023.

[42] Kimberley Johnson. Best brokers with api access in 2023. *wikijob*, Abril 2023.

[43] ¿qué es una api y cómo funciona? *Red Hat*, Enero 2023.

[44] IBKR Traders' AcademyIBKR Campus Finance Courses and Lessons. What is the tws api? https://ibkrcampus.com/trading-lessons/what-is-the-tws-api/.

[45] Python package index. selenium 4.8.3. https://pypi.org/project/selenium/, 23. Fecha de acceso: Marzo.

[46] Baiju Muthukadan. Selenium with python. https://selenium-python.readthedocs.io/. Fecha de acceso: 6 Junio 2023.

[47] Aprende Python. Selenium. https://aprendepython.es/pypi/scraping/selenium/. Fecha de acceso: 4 Junio 2023.

[48] Chromium. Chromedriver - webdriver for chrome. https://sites.google.com/a/chromium.org/chromedriver/downloads. Fecha de acceso: 20 Mayo 2023.

[49] Nombre Apellido. ¿qué es el web scraping? cómo extraer legalmente el contenido de la web. *Kinsta*, Diciembre 2020.

[50] Beautiful Soup. Beautiful soup documentation. https://beautiful-soup-4.readthedocs.io/en/latest/, 2023.

[51] Juan José Lozano Gómez. Web scraping con python. extraer datos de una web. guía de inicio de beautiful soup. *J2LOGO*, 2022.

[52] Sebastián Zuluaga. ¿qué es una posición en corto y largo en el trading? *MDC Trading Academy*, 2022.

[53] IBKR Traders' AcademyIBKR Campus Finance Courses and Lessons. Accessing portfolio data and account information. https://ibkrcampus.com/trading-lessons/python-account-portfolio/.

[54] Interactive Brokers. Executions and commissions. 2016.

[55] Antonio Moreno. Procesamiento del lenguaje natural ¿qué es?. inteligencia de cliente, pln. *Instituto de ingeniería de conocimiento*, 2022.

[56] ¿qué es flask?. aprendiendo qué es flask. *devCamp*, 2021.

[57] Redacción KeepCoding. ¿qué es render template de flask? *Keep Coding*, Agosto 2022.

[58] Juan Antonio Pascual Estapé. ¿qué es localhost o ip 127.0.0.1 y para qué se utiliza? *Computer Hoy*.

[59] Juan José Lozano Gómez. Tutorial flask – lección 1: La primera aplicación flask. *j2logo*, 2022.

[60] Support Apple. Gestión de scripts con launchd en terminal en el mac. https://support.apple.com/es-es/guide/terminal/apdc6c1077b-5d5d-4d35-9c19-60f2397b2369/mac, 2023. Fecha de acceso: 10 Junio 2023.

[61] Digital Ocean. Cómo usar cron para automatizar tareas en ubuntu 18.04. https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-ubuntu-1804-es, 2023. Fecha de acceso: 10 Junio 2023.

[62] Estrategias de inversion. Código isin (international standards identification number). https://www.estrategiasdeinversion.com/herramientas/diccionario/mercados/codigo-isin-international-standards-identification-t-1474. Fecha de acceso: 6 junio 2023.

[63] IBKR Traders' AcademyIBKR Campus Finance Courses and Lessons. Installing configuring tws for the api. https://ibkrcampus.com/trading-lessons/installing-configuring-tws-for-the-api/.

[64] IBKR Traders' AcademyIBKR Campus Finance Courses and Lessons. Essential components of tws api programs. https://ibkrcampus.com/trading-lessons/essential-components-of-tws-api-programs/.

APÉNDICE A

Detalles de acceso e interacciones con Investing.com utilizando Selenium

En este capítulo se comenta el código Python utilizado para la carga de operaciones en las carteras de investing.com, detallando las funciones creadas e interacción con la plataforma de investing.

A.1 Conexión con investing.com y configuración del controlador de Chrome

Funciones de Selenium

Selenium nos permite navegar o usar un navegador sin la participación humana y automatizar procesos a través de código, como escribir en una entrada de usuario e interactuar con el sitio web. **??**

Navegador web y Driver: Selenium necesita un navegador web instalado en el sistema para poder funcionar. Además de esto, también es necesario disponer un «webdriver» que permita manejar el navegador (a modo de marioneta).
Acceso: Una vez realizados esos pasos accedes a investing mediante su URL, utilizando el método .get()
Localización de elementos: Una vez dentro del sitio web ya podemos localizar elementos, como botones, inputs de texto, desplegables etc.

Interacciones: Selenium nos permite hacer clic en el lugar deseado, enviar texto por teclado, borrar una caja de entrada o manejar elementos de selección, entre otros: **??**

- Click: función `.click()`
- Enviar texto: función $.\texttt{send}_k eys('sdelquin')Borrarcontenido : funcin.\texttt{clear}()$
- Manejo de selects: `Select(lang)`

Acceso a investing.com con selenium

Todas las funciones o scripts que vamos a mencionar tienen que acceder a la plataforma investing.com para poder realizar sus operaciones. Este acceso se va a hacer siempre

de la misma manera, usando la librería Selenium, configurando el controlador de Chrome y estableciendo una conexión con la plataforma de inversiones mediante la automatización del proceso de inicio de sesión.

A.2 Creación de las tres carteras

Los pasos seguidos para crear las tres carteras de investing son:

1. Se define la función `crear_cartera_holdings` que recibe un parámetro `nombre_cartera`, que es el nombre que le queremos dar a la cartera que creamos.
2. Se eliminan todas las cookies del controlador utilizando el método `delete_all_cookies`.
3. Acceso al apartado 'My watchlist' utilizando el método `get` del controlador.
4. Se encuentra el botón de 'New portfolio' utilizando el método `find_element` del controlador y se hace clic en él.
5. Se encuentra el botón de tipo 'holdings' (tenencias) utilizando `By.CSS_SELECTOR` y se hace clic en él.
6. Se encuentra el campo de nombre de la cartera utilizando `By.ID` y se envía el nombre de la cartera ingresado mediante `send_keys`.
7. Se encuentra el botón 'Create' utilizando `By.ID` y se hace clic en él para crear la cartera.

A.3 Subida de las posiciones iniciales a la Cartera 1

Cargar el fichero csv donde guardamos las posiciones de la cuenta de IBKR:

1. Navega a la página de importación de CSV en la cartera de Investing.com.
2. Seleccionar el archivo CSV a importar y elegir el tipo de importación (en este caso, holdings).
3. Especificar el nombre de la cartera de destino para la importación.
4. Pasar a los seguientes dos pasos.
5. Confirmar la importación y cierra el controlador de Chrome.

A.4 Subida de las operaciones diarias a las tres carteras

Para estas funciones vamos a tener que sacar mucha información de la estructura HTML de la página. Esto es así, ya que por ejemplo, los identificadores para acceder a los botones o recuadros de texto usaremos siempre los mismos identificadores, en culaquier ejecución de la aplicación y para cualquier usuario, porque son elementos permanentes de la página y nunca cambiar. Pero, por el contrario, vamos a tener que sacar los identificadores de cada cartera, cada fila de la tabla de posiciones y cada fila en detalle, para poder acceder a tales apartados, ya que para cada ejecución y cuenta serán diferentes.

En primer lugar y antes de explicar cómo añadir operaciones a cada cartera, comentamos las funciones para extraer detalles e identificadores de las carteras y de las posciones:

- Recuperar información de las carteras (portfolios_Names_and_ID)

  1. Analizar el código HTML de la página utilizando BeautifulSoup.
  2. Encontrar el contenedor que contiene los nombres de las carteras.
  3. Encontrar todas las etiquetas `li` que representan información de cada cartera.
  4. Para cada etiqueta `li`:
     a) Obtener el nombre de la cartera (`title`).
     b) Obtener el ID de la cartera (`id`).
     c) Verificar el tipo de cartera analizando la clase de la etiqueta `span`.
     d) Agregar el nombre y el ID a la lista `existing_portfolios_positions`, como tuplas.
  5. Retornar la lista `existing_portfolios_positions` que contiene los nombres y los ID de las carteras de tipo "positions".

- Obtener qué posciones hay abiertas actualmente en una cartera (get_existing_positions(driver, id_cartera)).

  1. Utiliza BeautifulSoup para analizar el contenido HTML de la página y extraer la tabla que contiene las posiciones de la cartera.
  2. Identifica la tabla correcta utilizando el ID de la cartera.
  3. Itera sobre cada fila de la tabla para obtener la información relevante de cada posición.
  4. Para cada posición, crea un diccionario que contiene el nombre completo del activo y el ID de la fila.
  5. Agrega cada diccionario a una lista de posiciones existentes.
  6. Retorna la lista de posiciones existentes.

- Recuperar información de los detalles de una posición detail_rows(driver, id_row_position) Cuando vayamos a cerrar la posición de un activo, debemos iterar sus filas detalladas, por lo tanto debemos sacar sus identificadores.

  1. Obtener el código HTML de la página actual utilizando `driver.page_source`.
  2. Parsear el código HTML utilizando BeautifulSoup para poder realizar web scraping.
  3. Construir el ID de los detalles de las subfilas concatenando el prefijo 'details_' con el ID de la fila de la posición (`id_row_position`). Por ejemplo, si el ID de la fila de la posición es 'row_symbol_41609486_13994_39136031', el ID de los detalles sería 'details_row_symbol_41609486_13994_39136031'.
  4. Encontrar la etiqueta `tr` que tenga el ID de los detalles de las subfilas.
  5. Encontrar todas las etiquetas `div` dentro de la etiqueta `tr` de los detalles.
  6. Para cada etiqueta `div`:
     a) Obtener el ID de la subfila (`data-id`).
     b) Si el ID existe:
        1) Obtener el ID del precio concatenando el prefijo 'amount' con el ID de la subfila. Por ejemplo, si el ID de la subfila es 'row_symbol_41609486_13994_39136031", el ID del precio sería 'amountrow_symbol_41609486_13994_39136031'.
        2) Encontrar la etiqueta `input` con el ID del precio.

3) Obtener el valor del atributo `value` de la etiqueta `input`, que representa la cantidad de la subfila. *Nos guardamos también la cantidad de shares que tiene cada subfila, porque al añadir posiciones, muchas veces será necesario comprarla.*
4) Crear un diccionario (`d_row`) con las claves `'id_detail_row'` y `'amount_detail_row'`, y asignarles los valores correspondientes del ID de la subfila y la cantidad de la subfila.
5) Agregar el diccionario `d_row` a la lista `detailed_position_list`.

7. Retornar la lista `detailed_position_list`, que contiene información detallada de las subfilas de la posición.

Por otra parte, hay otras dos funciones que estarán presentes en el proceso de carga de operaciones en las carteras, las cuales nos sirven para automatizar los procesos de cerrar o añadir posiciones.

- Añadir posición (`add_position`)
  Esta función se encarga de añadir una posición a una cartera específica en Investing.com. Toma como argumentos el controlador del navegador (`driver`), la información de la operación (`operation`), el ID de la cartera (`id_cartera`), el tipo de operación (`type`) y opcionalmente la cantidad (`amount`).
  `operation` es la operación que descargada de IBKR que estamos procesando actualmente.

  1. Se accede a la página de la cartera en Investing.com.
  2. Se elimina una capa de superposición generada por una cookie para poder interactuar con los elementos de la página.
  3. Se hace clic en la cartera específica seleccionada mediante su ID.
  4. Se hace clic en el botón para añadir una posición en la cartera.
  5. Se introduce el nombre del activo en el campo de búsqueda y se selecciona el primer resultado.
  6. Se completan los campos requeridos, como el tipo de operación, la fecha, la cantidad y el precio.
  7. Se hace clic en el botón de guardar para añadir la posición a la cartera.

- Cerrar posición (`close_position`)
  Esta función toma como argumentos la fecha, la cantidad, el precio y el identificador de la fila de la posición detallada.

  1. Se utiliza driver.find_element junto con XPath y otros métodos para localizar y hacer clic en el botón 'Cerrar posición'.
  2. Se procesa la fecha para obtener el día y el mes en el formato deseado.
  3. Se localiza el campo de entrada de fecha y se hace clic en él para abrir el calendario.
  4. Se selecciona el mes y el día deseados haciendo clic en los elementos correspondientes dentro del calendario.
  5. Se localizan los campos de entrada de cantidad y precio y se ingresan los valores proporcionados.
  6. Por último, se hace clic en el botón 'Guarda' para cerrar la posición.

- Añadir las posiciones en corto a la Cartera 3 ($\texttt{add}_p osition_t oCartera3) Tomacomoargumentoseliddelacarte nombre, cantidad, precio, fechayeltipodeposicin, quepordefectoseponeatipoventa$.

  Tiene el mismo funcionamiento que la función `add_position`, pero es específica para añadir operaciones indicando cada parámetro de información por separado y siempre con tipo venta.

- Sacar la información de una posición detallada de la cartera (`detail_rows`). Recibe como argumento el id de la posición detallada y el nombre del activo de esa posición. Mediante web scraping, accede a la fila con ese id y extrae toda su información: cantidad, precio, fecha y nombre, devolviendo esto en formato de diccionario.

A.4.1. Cartera 1

Pasos para añadir operaciones a la Cartera 1:

1. Ir a la lista de seguimiento:
   - Abrir la página de la lista de seguimiento
   - Eliminar el elemento de consentimiento de cookies utilizando JavaScript
2. Acceder a la cartera:
   - Obtener el ID de la cartera deseada (mediante la función `portfolios_Names_and_ID`)
   - Hacer clic en la cartera deseada (Cartera 1)
3. Leer el hitórico de las operaciones realizadas en el útlimo día desde un archivo CSV
4. Procesar cada operación:
   - Obtener las posiciones existentes en la cartera, mediante la función $\texttt{get\_existing}_p ositions.Verifi$
   - Realizar acciones diferentes según el estado del activo:
     - Si el activo está en la cartera:
       - Acceder a la posición en la cartera
       - Obtener las filas de detalle de la posición (`detail_rows`)
       - Realizar acciones según la cantidad a vender:
         - Iterar sobre las filas de detalle y cerrar posiciones hasta alcanzar la cantidad deseada
         - Si queda cantidad por vender pero no hay más filas de detalle, agregar una nueva posición de venta en corto
     - Si el activo no está en la cartera:
       - Agregar una nueva posición, mediante la función `add_position` de venta en corto o compra normal según la acción de la operación

A.4.2. Cartera 2

Pasos para añadir operaciones a la Cartera 2:

1. Se obtiene el ID de la cartera 'Cartera 2' para realizar operaciones en esa cartera en particular.

2. Se accede a la cartera 'Cartera 2'.
3. Leer el hitórico de las operaciones realizadas en el útlimo día desde un archivo CSV
4. Para cada operación en el archivo CSV, realizar las siguientes acciones:
   a) Actualizar la lista de posiciones existentes en la cartera 'Cartera 2'.
   b) Comprobar si el activo de la operación de IBKR ya está presente en la cartera.
   c) Si se trata de una operación de venta, añadir una posición en la cartera.
   d) Si se trata de una operación de compra y el activo ya está en la cartera, cerrar una posición parcial o completa según la cantidad de compra.

A.4.3. Cartera 3

Pasos para añadir operaciones a la Cartera 3:

1. Scrapear las posiciones type SELL de la Cartera 1.
   En primer lugar se obtienen las filas de la Cartera 1 correspondientes a posiciones de tipo 'SELL' únicamente.

   Una vez tenemos las posiciones de tipo 'Sell', se itera cada una de esas posiciones para obtener las posciones detalladas, mediante la función `detail_rows`. Nos guardamos toda la infromación de cada posición detallada en una lista que usaremos luego. Esta infromación es: nombre, cantidad, precio y fecha.
2. Borrar la Cartera 3.
   Accedemos a la Cartera 3 y mediante el botón de opciones borramos por completo la cartera.
3. Crear Cartera 3 vacía.
   Volvemos a crear de nuevo la Cartera 3, que se creará vacía por su puesto, usando la función `crear_cartera_holdings`.
4. Añadir las posiciones scrapeadas a la Cartera 3 con type SELL.
   Accedemos a esa nueva Cartera 3 creada, para añadir las posiciones de venta en corto del día de hoy, es decir, las que hemos guardado en una lista en el primer paso.

   Iteramos cada una de las posicioens de venta en corto y la añadimos a la Cartera 3 con toda su información, usando la función `add_position_toCartera3`.

APÉNDICE B
Introducción a la TWS API de Interactive Brokers

En este capítulo vamos a explicar detalladamente todos los pasos realizados para llevar a cabo la monitorización de las operaciones, explicando las herramientas tecnológicas, lenguajes, librerías o software necesarios para realizar cada paso.

En primer lugar comentamos el proceso para descargarnos las operaciones que vamos realizando en IBKR.
En segundo lugar comentamos el proceso para, a partir de un archivo csv con datos de operaciones, subir esas operaciones a una cartera de investing.com.
En último lugar comentamos cómo podemos unir ambas partes, para que funcione todo como un único programa.

B.1 ¿Qué es la TWS API?

IBKR ofrece diferentes plataformas de traiding, incluyendo:

- Trader Workstation (TWS) es la aplicación de escritorio de IBKR.
- Client Portal es la plataforma web de trading.
- IBKR mobile, la aplicación móvil de trading.

Además de utilizar los softwares de negociación de IBKR comentados, hay otra forma por la que el usuario puede crear aplicaciones personalizadas para realizar órdenes en su cuenta de IBKR. Esta forma es mediante la conexión a la TWS API, que está disponible para todos los clientes.
La Trader Workstation API es una interfaz de código abierto para la TWS con la que crear aplicaciones personalizadas para automatizar funcionalidades de la plataforma de TWS, incluyendo entre otras, acciones como:

- Ejecución de órdenes
- Recepción de datos de cartera
- Recepción de datos de mercado
- Consulta de detalles de instrumentos financieros

La API de TWS en sí no proporciona nuevas funcionalidades no disponibles en TWS, sino que proporciona la capacidad de automatizar algunas acciones dentro de TWS desde software externo. Este código está disponible en lenguajes de programación generales como Python, Java, C# y C++. [44]

B.2 Instalación y configuración de TWS para la API

Descargamos la aplicación de software Trader Workstation (TWS), que será el software que se integre con nuestra API.
Esta aplicación de escritorio la descargamos gratuitamente de la página web, en el menú Trading >Platforms.

Una vez descargada la aplicación de escritorio, al abrirla debemos iniciar sesión con el usurio y contraseña que creamos en el apartado anterior, indicando que vamos a iniciar sesión con la cuenta de papel, es decir, con la cuenta simulada.

La primera vez que iniciamos sesión debemos habilitar una configuración para permitir la conexión a la API y otras configuraciones que afectan a la conexión inicial. Estas configuraciones se pueden encontrar de forma detallada en el tutorial de configuración [63].

B.3 Acceso al código fuente de TWS Python API

El código fuente de la API sirve para definir los mensajes disponibles que pueden intercambiarse entre la TWS y una aplicación API externa. Es decir, especifica tanto la forma y sintaxis permitida para crear códigos de Python en los que poder pedir informción a la API y que 'nos entienda', como el tipo de información que podemos pedir.

En la página principal de IBKR accedemos a Trading >API's menu y en apartado de Trader Worksation (TWS) API >Downloads & Resources, accedemos a 'Software: API de TWS' y procedemos a la descarga tras elegir nuestro sistema operativo.
En nuestro caso, al estar usando Windows, después de hacer la instalación de la API se guarda informaicón en dos directorios:

- C:/TWS API. La API es instalada aquí
- C:/TWS API/source/pythonclient/ibapi. Carpeta para el código fuente de Python. Proporciona clases y funciones para crear y enviar solicitudes a la API y recibir respuestas.
- C:/TWS API/samples/Python/Testbed. Carpeta con códigos de ejemplo en Python, para ayudar a los usuarios a empezar a trabajar con la API de TWS en Python.

B.4 Componentes esenciales de los programas TWS API

En cualquier programa API TWS siempre hay dos clases principales, EClient y EWrapper [64].

- EClient. Utilizado para los mensajes salientes que se envían desde el programa API a la TWS.
- EWrapper. Utilizado para manejar los mensajes entrantes desde el servidor de Interactive Brokers a través de TWS.

Por lo tanto, al crear los scrpits necesarios para nuestro programa, para que se puedan comunicar con la API, siempre deberemos importar esas clases.
Estas clases se encuentran definidas en el código fuente de Python, que hemos dicho que se guardaba por defecto en el directorio 'C:/TWS API/source/pythonclient/ibapi'.
De esta forma, comenzamos importando el módulo EClient y el módulo EWrapper. Después, creamos una clase para combinar los dos módulos y a continuación instanciamos la clase, ya que vamos a utilizar la programación orientada a objetos.
El código Python quedaría así:

```
1 from ibapi.client import * # importar el modulo EClient
2 from ibapi.wrapper import * # importar el modulo EWrapper
3
4 class TestApp(EClient, EWrapper): # clase que hereda de las
    clases                          # EClient y EWrapper
5     def __init__(self):
6         EClient.__init__(self, self)
```

Tras haber importado los módulos necesarios para hacer posible el intercambio de información, dentro de la clase creada para nuestra aplicación, crearemos las funciones pertinentes para que dicha aplicación ejecute las acciones necesarias.

ods$_e tsinf$

APÉNDICE C

Relación con los Objetivos de Desarrollo Sostenible de la Agenda 2030 de Naciones Unidas

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS 1. **Fin de la pobreza.**		X		
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**	X			
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades.**		X		
ODS 11. **Ciudades y comunidades sostenibles.**				X
ODS 12. **Producción y consumo responsables.**		X		
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**	X			

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

El Trabajo de Fin de Grado (TFG) 'Aplicación para la monitorización de operaciones en el mercado de valores', donde se desarrolla una aplicación que permite la monitorización y análisis de operaciones en el mercado de valores y la ejecución de transacciones mediante lenguaje natural,tiene una relación directa con varios Objetivos de Desarrollo Sostenible (ODS) de la Agenda 2030 de Naciones Unidas. A continuación, se destacan los ODS más relevantes para el proyecto.

En primer lugar, el TFG contribuye al logro del 'ODS 8: Trabajo decente y crecimiento económico'. La aplicación desarrollada permite a los inversores tener un mayor control y seguimiento de sus operaciones en el mercado de valores, lo que se traduce en una mejora de la eficiencia y la transparencia en el ámbito financiero. Al proporcionar herramientas para la toma de decisiones informadas, la aplicación ayuda a los usuarios a maximizar sus beneficios y, por lo tanto, a fomentar un crecimiento económico sostenible.

Además, el proyecto se relaciona con el 'ODS 9: Industria, innovación e infraestructura', demostrando el enfoque hacia la innovación tecnológica en el ámbito financiero. La utilización de la inteligencia artificial para desarrollar un chatbot que facilite la realización de transacciones de manera sencilla también muestra cómo la tecnología puede impulsar la eficiencia y la accesibilidad en los mercados financieros.

El TFG también se alinea con el 'ODS 17: Alianzas para lograr los objetivos'. Al proporcionar acceso al código creado mediante un repositorio abierto, se fomenta la colaboración y la cooperación entre los diferentes desarrolladores interesados en el mercado de valores.

Además de los ODS mencionados anteriormente, también se puede econtrar relación con otros objetivos de manera indirecta. Por ejemplo, el 'ODS 1: Fin de la pobreza' y el 'ODS 10: Reducción de las desigualdades', ya que una gestión eficiente de las inversiones puede contribuir a generar riqueza y reducir las desigualdades económicas. El 'ODS 12: Producción y consumo responsables' también se puede vincular, ya que analizar las operaciones puede promover prácticas financieras más responsables y sostenibles.

En resumen, el TFG .Aplicación para la monitorización de operaciones en el mercado de valores"del grado en Ciencia de Datos se relaciona directamente con los ODS 8 y 9, y de manera indirecta con otros objetivos como el ODS 1, 10 y 12. Además, el proyecto promueve la colaboración y la transparencia en el ámbito financiero, lo que es consistente con el ODS 17. A través de la mejora de la eficiencia, la accesibilidad y la toma de decisiones informadas, el trabajo ayuda a impulsar un crecimiento económico sostenible y a promover prácticas financieras responsables.