



# MANUEL DE USUARIO

## J2A Coin

### AUTORES:

Javier Álvarez Páramo

Adrián Martín García

Andreea Tarabuta

## Índice

Descripción del sistema .....	3
Funcionalidad.....	3
Requisitos previos .....	3
Herramientas necesarias .....	3
Guía de inicio del software .....	5
Clonación del repositorio .....	5
Opción 1: Ejecutar mediante nuestro ordenador .....	5
<b>Ejecutar el servidor</b> .....	5
<b>Ejecutar el frontend</b> .....	6
Opción 2: Ejecutar mediante Docker-compose .....	6

## Descripción del sistema

J2ACoin es un software libre diseñado para aquellas personas interesadas en la compra o futura inversión en criptomonedas, pero con necesidad de realizar una suscripción para obtener la totalidad de sus funcionalidades. En lo que destaca de las múltiples plataformas, es la visualización de tres puntos importantes con un solo vistazo.

Este software integra el ranking de top 10 de las monedas más populares, un heatMap para una mejor comprensión de los datos y por último ofrecemos la posibilidad de su compra mediante unos Exchange otorgando una plena seguridad. A continuación, detallaremos más en profundidad sus funcionalidades, así como la estructura que presenta.

## Funcionalidad

Como ya hemos mencionado en los apartados anteriores, esta aplicación está enfocada en aquellas personas apasionadas y con un gran interés en las criptodivisas.

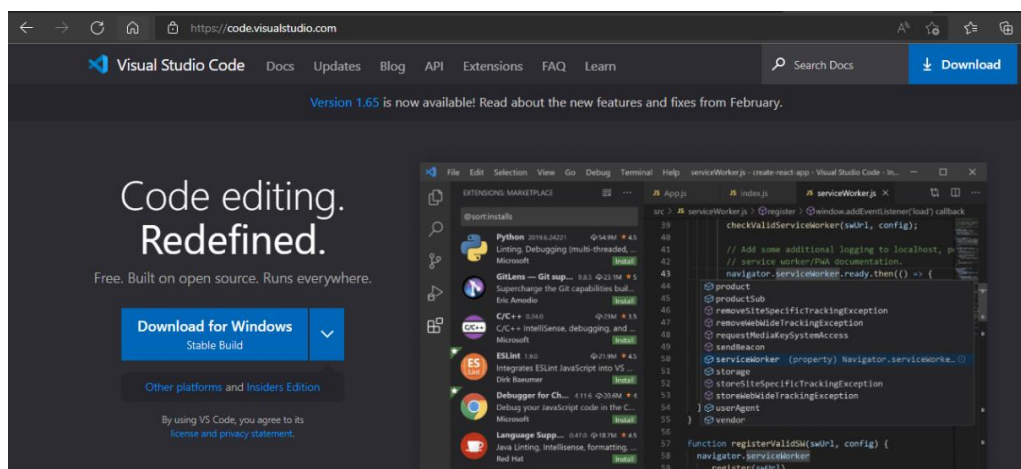
La principal funcionalidad que queremos otorgar aparte de una visión instantánea de las mejores criptomonedas de cada día, también se encuentra la plena seguridad de las páginas de compra. Es por ello, que J2ACoin ofrece la mejor fiabilidad a sus clientes tan solo con una suscripción de 2€/ mes recibiendo información privilegiada y una gran facilidad de realizar su compra.

## Requisitos previos

Como toda aplicación, debemos tener en cuenta unos requisitos previos antes de empezar a descargar la aplicación del repositorio de GitHub [1], para ello hace falta realizar las siguientes configuraciones para obtener un buen funcionamiento de la aplicación.

## Herramientas necesarias

El usuario debe haber instalado previamente un entorno de desarrollo para la manipulación de los datos, hay que tener en cuenta que el código esta implementado en



lenguaje Python. Por ello, una herramienta de fácil uso y recomendable es Visual Studio Code [2], además de ser una herramienta gratuita para la creación de aplicaciones modernas para Windows, Android e iOS, aplicaciones web y servicios en la nube.

El siguiente paso es instalar las dependencias necesarias para poder ejecutar la aplicación. La ejecución de J2ACoin se puede realizar de dos maneras diferentes que detallaremos a continuación.

### **Opción 1: Mediante la ejecución del comando yarn**

Este primer caso, necesitaremos tener instalado una versión de Node en nuestro ordenador. Para comprobar si satisfacemos este requisito, podemos ejecutar:

- **node --version**
- **npm --version**

Si ambos comandos nos devuelven una versión estaremos listos para instalar *yarn* con el siguiente comando:

- **npm install --global yarn**

Ahora sí, estamos listos para arrancar la aplicación.

### **Opción 2: Mediante la ejecución del comando docker-compose**

En este caso necesitamos tener instalado la plataforma de software de Docker. En primer lugar, comprobamos si lo tenemos instalado en nuestro ordenador mediante los siguientes comandos:

- **Docker --version**
- **Docker-compose --version**

Si ambos comandos nos devuelven una versión de los mismos, es que ya no hace falta realizar su instalación.

En caso de no obtener un resultado satisfactorio, debemos realizar las siguientes instalaciones:

- **Sudo apt install docker**
- **Sudo apt install docker-compose**

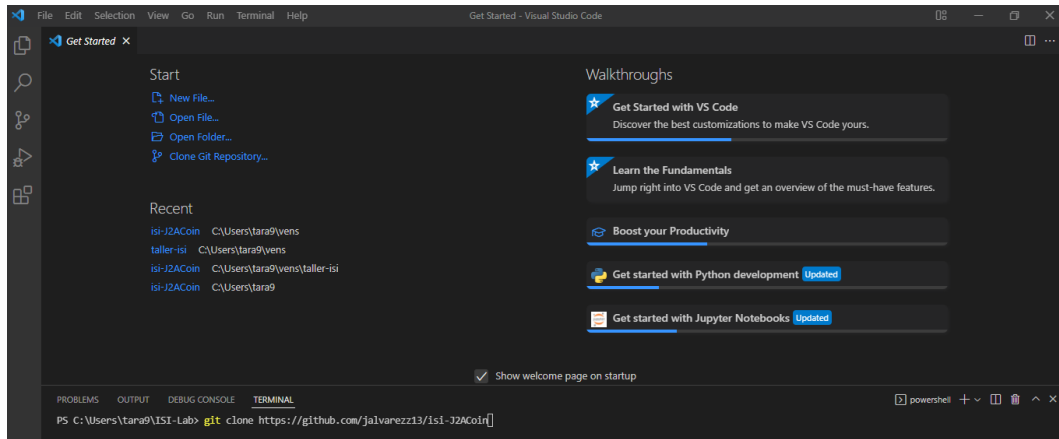
Y ya estaría listo para poder llevar a cabo el funcionamiento de la aplicación.

## Guía de inicio del software

### Clonación del repositorio

El siguiente paso, es realizar la clonación del repositorio de GitHub mediante el siguiente comando en una terminal *PowerShell* del Visual Studio Code:

➤ **git clone** <https://github.com/jalvarezz13/isi-J2ACoin>



### Opción 1: Ejecutar mediante nuestro ordenador

#### Ejecutar el servidor

Una vez descargado los archivos en nuestro ordenador, abrimos la carpeta en una ventana de Visual Studio Code. El siguiente paso, es descargar todas las librerías necesarias para ejecutar el archivo `server.py` que se ubica en el directorio `api`. Esto lo haremos mediante los siguientes comandos:

➤ **pip install -r requirements.txt**  
➤ **python3 server.py**

Si obtenemos como salida lo siguiente, el servidor habrá arrancado correctamente.

```
PS C:\Users\tara9\ISI-Lab\isi-J2ACoin> cd api
PS C:\Users\tara9\ISI-Lab\isi-J2ACoin\api> python server.py
* Serving Flask app 'server' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

### Ejecutar el frontend

Para la ejecución de la aplicación tenemos dos formas de realizarlo, una de forma mas directa y sencilla y otra mediante la descarga de módulos de yarn. A continuación, detallaremos ambos casos por partes:

Para ello, abrimos otra terminal, en este caso de tipo *cmd* y mediante el siguiente comando descargamos todos los módulos de la aplicación. Esto lo haremos con *yarn* que es el gestor de dependencias que utilizamos.

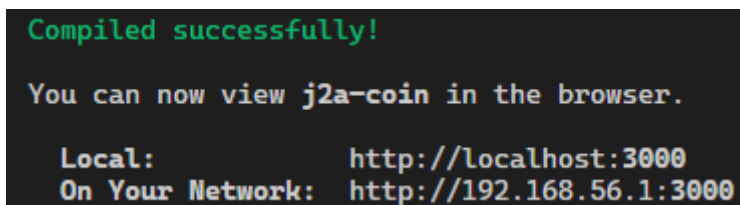
#### ➤ **yarn**

Esto puede tardar varios minutos, por lo tanto, *¡ten paciencia!*

Una vez, tengamos todas las dependencias de nuestro proyecto, procedemos a arrancarlo, mediante la ejecución del siguiente comando:

#### ➤ **yarn start**

Tras unos segundos, se abrirá nuestro navegador por defecto con una página cuya dirección es: <http://localhost:3000> y en la terminal aparecerá lo siguiente: ¡Perfecto! La aplicación está corriendo en local, correctamente.



```
Compiled successfully!  
  
You can now view j2a-coin in the browser.  
  
Local:      http://localhost:3000  
On Your Network:  http://192.168.56.1:3000
```

### Opción 2: Ejecutar mediante Docker-compose

Al igual que en el caso anterior, debemos abrir otra terminal de tipo *cmd* y ejecutar el siguiente comando:

#### ➤ **docker-compose up -d**

Tras su ejecución, se abrirá nuestro navegador por defecto con una página cuya dirección es: <http://localhost:3000> y en la terminal aparecerá lo siguiente: ¡Perfecto! La aplicación está corriendo en local, correctamente.