Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Análisis de datos del Bitcoin

Bitcoin data analysis

Alejandro David Carrillo Padrón

La Laguna, 20 de agosto de 2019

Dña. **Rosa María Aguilar Chinea**, con N.I.F. 43.778.956-C Catedrática de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

D. **Jesús Miguel Torres Jorge**, con N.I.F. 43.826.207-Y profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Análisis de datos del Bitcoin"

ha sido realizada bajo su dirección por D. **Alejandro David Carrillo Padrón**, con N.I.F. 45939593-Z.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 20 de agosto de 2019

Agradecimientos

Quiero agradecer a todas las personas que me han acompañado a lo largo de la carrera, sin ellos no podría haber llegado hasta aquí.

Por último, agradecer a mis padres, sin su apoyo nada de esto hubiera sido posible.

¡Gracias!

Licencia

Resumen

El objetivo principal ha sido trabajar con las librería Bitcoin y Blockchain para implementar un sistema que permite tanto crear distintos tipos de cuenta y poder hacer transacciones entre ellas como recopilar y tratar la distinta información del Bitcoin para analizar las transacciones, el tamaño de bloque, el precio del Bitcoin entre otras opciones.

Para ello, se desarrolla la aplicación en Python 3.7.3 que permitirá crear una interfaz gráfica (Tinker) y hacer gráficas(matplotlib) para mostrar los datos tratados.

Palabras Clave: Bitcoin, Blockchain, tratamiento información, precio, Python.

Abstract

The main objective has been to work with the Bitcoin and Blockchain libraries to implement a system that allows both to create different types of accounts and to be able to make transactions between them as well as to collect and process the different Bitcoin information to analyze the transactions, the block size, the Bitcoin price among other options.

For this, the application is developed in Python 3.7.3 that will allow us to create a graphical interface (Tinker) and make graphs (matplotlib) to show the processed data.

KeyWords : Bitcoin, Blockchain, data manipulation data , Price, Python.

Índice general

Índice de figuras

1. Introducción

1.1. Motivación

La tecnología nacida del Bitcoin [1], la conocida Blockchain [2] está teniendo una gran importancia tanto a nivel social como económico. Tanto es así, que empresas como el Banco Santander, Alibaba y Facebook, entre otras muchas empresas [3], están implementando con Blockchain [2] distintas soluciones sus problemas. Por ejemplo al seguimiento de paquetes, a la gestión de los derechos intelectuales, a la protección de datos o a la administración de las cadena de suministro, entre muchos otros.

Por otra parte, el Bitcoin sigue en constante desarrollo y a día de hoy sigue siendo muy usado. Actualmente, empresas como Microsoft y Dell lo aceptan como medio de pago [4]. El repunte que ha sufrido en los últimos meses tanto en uso como en precio hace del Bitcoin un tema de actualidad. Por ello, en este proyecto se analizarán distintos puntos relacionados con el Bitcoin.

1.2. Antecedentes

1.2.1. Historia del Bitcoin

En 2008 Satoshi Nakamoto publicó un artículo en la lista de criptografía de Metzdowd.com donde describe el protocolo Bitcoin. El Bitcoin es una red p2p [5] que se utiliza como criptomoneda, utiliza el sistema Proof of work (PoW) [6] y usa el consenso entre nodos para verificar las transacciones. Se caracteriza por:

- Ser descentralizada, es decir, no está respaldado por ningún gobierno o banco central.
- Ser de código abierto.
- Anónimo o pseudónimo en cuanto a la participación en la red.
- Sin censura ni restricciones, es decir nadie puede prohibir o censurar transacciones.
- Los pagos son irreversibles.
- No existen intermediarios.
- Hasta ahora, ha sido imposible de falsificar o duplicar.
- Existe una cantidad límite de Bitcoins, 21 millones.

El Bitcoin es la primera criptomoneda. El 4 de enero de 2009 se registra el primer bloque de Bitcoin creando así la red y la emisión de los primeros Bitcoins. Muchos de los precursores de las criptomonedas como son Hal Finney, Wei Dai y Nick Szabo apoyaron y contribuyeron a este proyecto.

A partir de 2011, Electronic Frontier Foundation y posteriormente Wikileak empezaron a aceptar donaciones en Bitcoins. Con el aumento de la popularidad del Bitcoin distintas multinacionales como son Microsoft, Dell y Steam empiezan a aceptar el pago en Bitcoins, como ya habíamos mencionado.

Antes del lanzamiento de Bitcoin ya existían algunas tecnologías digitales de dinero basados en los protocolos ecash [7] de David Chaum y Stefan Brands.

Adam Back desarrollo Hashcash [8], la cual es una estrategia de verificación, que consiste en gastar un cierto tiempo de CPU si quieren enviar algo. Esta estrategia se creó en principio para evitar el spam, ya que el envío masivo de datos acarrearía un gran gasto de CPU. Esta estrategia finalmente se utilizó también en el Bitcoin.

Wei Dai y Nick Szabo crearon las primeras propuestas de criptomonedas.

Hal Finney desarrolló el Sistema "POW" (Proof-Of-Work system), que utiliza Hashcash. El sistema POW sirve para verificar que se está realizando un trabajo, esto se consigue mandando un problema moderadamente difícil pero factible al cliente, pero fácilmente verificable por el servidor.

Nick Szabo investigaba un mecanismo de control de la inflación basado en el mercado, creado así un antecedente al Blockchain.

A lo largo de su vida, el Bitcoin ha fluctuado con respecto al USD (valor de referencia) según ha ido madurando. Estas fluctuaciones no han sido constantes a lo largo del tiempo. Los fechas y intervalos más destacables son las siguientes:

Enero 2000 - Abril 2011	Su precio pasó de 0 a valer 1 USD.
Abril 2011 - Julio 2011	Su precio alcanzó 31 USD.
Julio 2011 - Abril 2013	En este periodo alcanzó un máximo precio de 266 USD.
Abril 2013- Enero 2017	Donde hubo grandes variaciones pero rompió la barrera de los 1.000 USD.
Enero 2017 - Septiembre 2017	Alcanzó el valor de 5.000 USD
12 Septiembre de 2017	Hubo una bajada hasta los 2.900 USD
El 17 Diciembre de 2017	Alcanzó su máximo histórico llegando a valer 19.783 USD.
22 Diciembre de 2017	El Bitcoin perdió un tercio de su valor llegando a valer 13.800 USD.
Diciembre 2017 - Marzo 2019	Su precio ha bajado hasta los 3.852 USD
Marzo 2019 – Agosto 2019	Su precio ha subido hasta los 10.000 USD aproximadamente.

1.2.2. Conceptos relacionados con el Bitcoin

El Bitcoin al ser una moneda virtual necesita de un monedero virtual también conocido como Wallet [9] .

Para poder generar un Wallet es necesario crear como mínimo una clave privada [10] y una clave pública [10] a partir de esta.

Dentro de las Wallets se distinguen dos tipos según quién posee las claves:

- Wallets en las cuales tú posees tu claves.
- Wallets en las cuales tú delegas la custodia de la claves a un tercero pero a través de una cuenta en una plataforma puedes hacer uso de ella.

De la misma manera las Wallets también se distinguen por propietarios:

- Wallets únicas las cuales se generan solo con una clave pública y tiene control total de la wallet
- Wallets Multi Signature[11] las cuales se generan con varias claves públicas y una única dirección, por lo tanto, para poder hacer una transferencia las distintas claves públicas deben llegar a un consenso.

Si a la clave pública se le aplica el algoritmo ECDSA[12] (Elliptic Curve Digital Signature Algorithm) se obtiene obtiene una dirección alfanumérica la cual es la dirección de tu Wallet.

Poseer tanto la clave privada como la clave pública de una dirección te hace dueño de todo los Bitcoin que se encuentran en ella.

Cuando haces una transferencia de Bitcoins de una cuenta a otra ocurre lo siguiente:

Se crea una transacción en la que se ingresa las claves públicas tanto del que quiere enviar como el que va a recibir y la cantidad que va a ser enviada. Después de esto coge dicha transacción y la firma con su clave privada y la envía a un nodo de la red Blockchain, este nodo también conocido como minero verifica que la información es real y puede ser llevada a cabo la añade a la cadena de bloques.

1.2.3. Blockchain.com

La pagina Blockchain.com es un explorador de bloques de como también es así como una billetera de criptomonedas.

Dentro de su página podremos observar distintos tipos de gráficas que hablan de la estadísticas de la moneda, detalles de bloque, información de extracción y actividad de la red.

Dentro de estas gráficas las más significativas son:

- El precio de mercado en USD
- El valor total de los Bitcoins que se encuentran en circulación en USD
- La mediana de tiempo para una transacción sea aceptada en un bloque
- La dificultad de minado

2. Objetivos

2.1. Introducción

El objetivo principal de este proyecto es implementar un software que nos permita tanto la creación de cuentas offline de Bitcoins, como poder visualizar el valor en tiempo real de los Bitcoins en distintas divisas y viceversa, así como ser capaz de visualizar distintas gráficas relacionadas con el Bitcoin como manipular las mismas para ver los datos los datos que nos interesan

2.2. Creación de cuentas

Una de las tareas a realizar en este proyecto radica en poder crear distintos tipos de cuenta: Las cuentas simples o single signature y cuentas múltiples o multi signature con la finalidad de ver las diferencias entre ellas y cómo funcionan

2.3. Tratamiento de datos

El objetivo de este apartado no solo consiste en adquirir conocimientos a través de diferentes recursos como pueden ser Blockchain.com sino también hacer uso de todas las herramientas disponibles para poder manipular a nuestro gusto todos estos datos con el fin de buscar específicamente aquello que necesitamos.

3. Desarrollo de Proyecto

3.1. Introducción

Una vez con el proyecto en mente tenía que elegir un lenguaje en el cual escribir mi programa y entre todas las posibles opciones decidí escoger Python[13] ya que es un lenguaje relativamente rápido, es ordenado, limpio, tiene una gran comunidad y está muy extendido por lo tanto sería fácil solventar un problema si tuviera dudas.

3.2. Metodología usada

Una vez se decidió en qué lenguaje se desarrollaría el programa, se planteó cómo hacerlo. Después de valorar varias alternativas, se escogió estructurarlo basándose en el Modelo-Vista-Controlador (MVC). Este modelo propone la construcción del programa en 3 componentes diferentes: el modelo, el controlador y la vista.

- El modelo se encarga tanto de almacenar los datos que tiene el programa como de la gestión del guardado de los datos.

- El controlador se encarga de recibir distintas órdenes del usuario a través de la vista. Además de comunicar entre sí al modelo y la vista, siendo el controlador la pasarela entre ellas.

- La vista es la representación visual de los datos, es decir ,la interfaz gráfica.

3.3. Librerías y herramientas utilizadas

Para desarrollar esta aplicación se han utilizado las siguientes librerías y herramientas que se detallaran a continuación:

- pip

Es un gestor de paquetes que se utiliza para instalar y administrar paquetes de

software escritos en Python. En este proyecto se ha utilizado para instalar todas las librerías.

Para instalar un paquete se debe utilizar el comando:

pip install (Paquete a instalar)

- tkinter

Es una librería gráfica la cual se considera un estándar para Python[14]. Esta librería es la que se ha utilizado para crear la interfaz gráfica del proyecto.

- Bitcoin

Librería de Python que permite crear cuentas de Bitcoins usando claves privadas y públicas.

- Blockchain

Librería que permite acceder a algunos datos de la página blockchain.com, los cuales se podrán manejar y tratar, para mostrar y manipular distintas gráficas acerca del Bitcoin.

- Pandas

Librería open source que permite manipular los datos que adquirimos para trabajar mejor con ellos [15].

- NumPy

Es una librería de python que proporciona funciones matemáticas de alto nivel para operar con matrices y vectores[16].

- Request

Liberia que permite tanto enviar como recibir información a través de internet[17].

- ssl

Librería de soporte para poder comunicar datos con distintas webs. En este proyecto será con webs que utilizan el protocolo https[18].

3.4. Interfaz de la aplicación

Una vez se inicia la aplicación, se puede observar la siguiente pantalla en la cual se observan distintas pestañas:

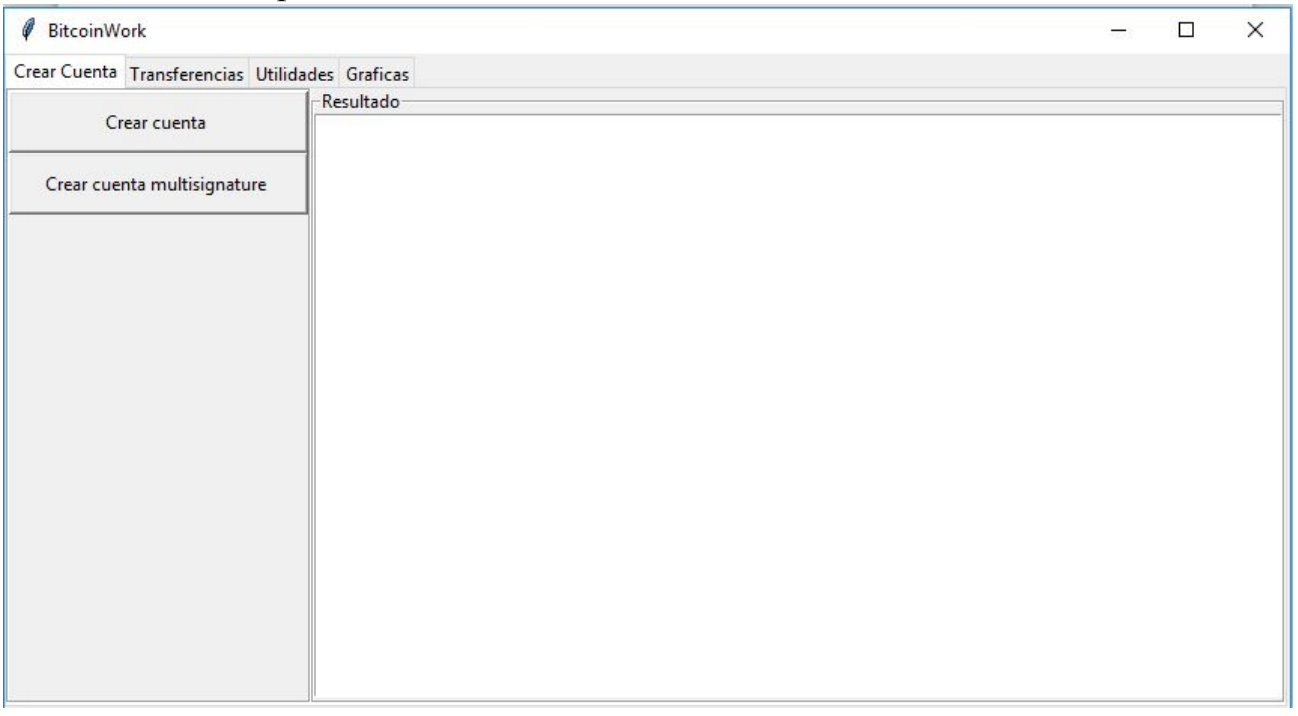

Figura 1: Interfaz gráfica Inicio

La pestaña principal muestra 2 botones:

- Crear cuenta.
- Crear cuenta multi signature.

Por como funcionan las cuenta multi signature, al seleccionar este botón aparece una nueva ventana tal que así:

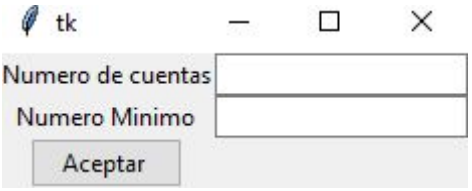

Figura 2: Interfaz gráfica crear cuenta

El número de cuentas es el número de cuentas asociadas a la dirección a crear. El número mínimo es el mínimo de claves públicas que deben de estar de acuerdo para realizar una transacción.

Cuando finalicemos cualquiera de las funciones anteriormente nombradas, nos aparecerá en el área de resultados las distintas variables que corresponden a cada cuenta, como son, la clave Privada, la clave pública y la dirección Bitcoin

En la pestaña Transferencias se ve de la siguiente manera:

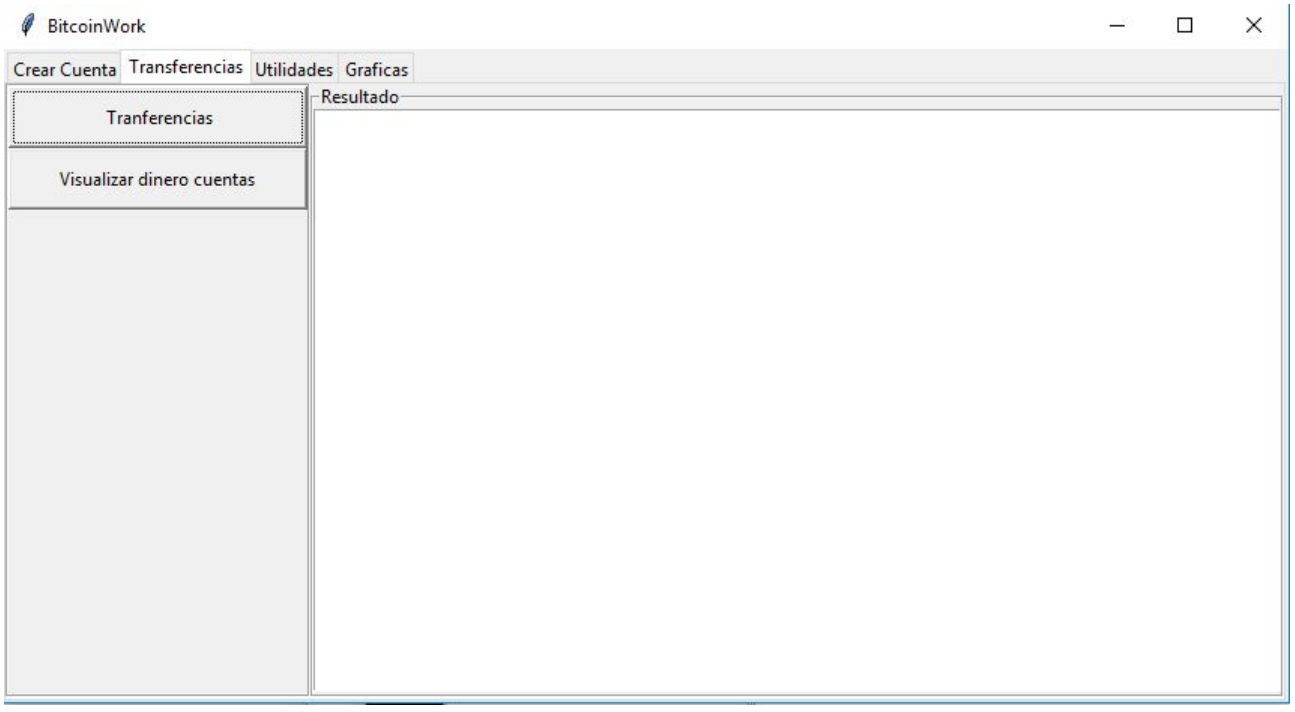

Figura 3: Interfaz gráfica Transferencias

Como se puede observar hay 2 botones: transferencia y visualizar dinero de cuentas.

- Transferencias

Al hacer clic en este botón se muestra la siguiente ventana:

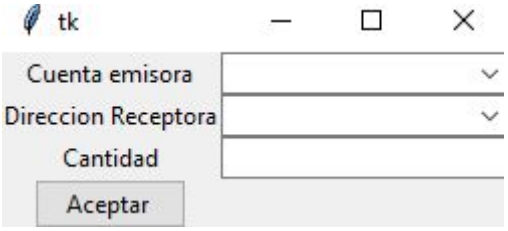

Figura 4: Interfaz gráfica transferencia envio

La cuenta emisora es la clave pública de la dirección que va a enviar los Bitcoins. Mientras que la dirección receptora es la dirección de la cuenta a la cual la persona quiere enviar Bitcoins. La cantidad son las unidades de Bitcoin a enviar.

- Visualizar dinero cuentas

Esta función muestra por pantalla tantos las cuentas creadas como sus correspondientes saldo. Por tanto, muestra en el área blanca definida el contenido de todas las cuentas generadas por el programa.

La pestaña utilidades, por otro lado, se muestra de esta manera:

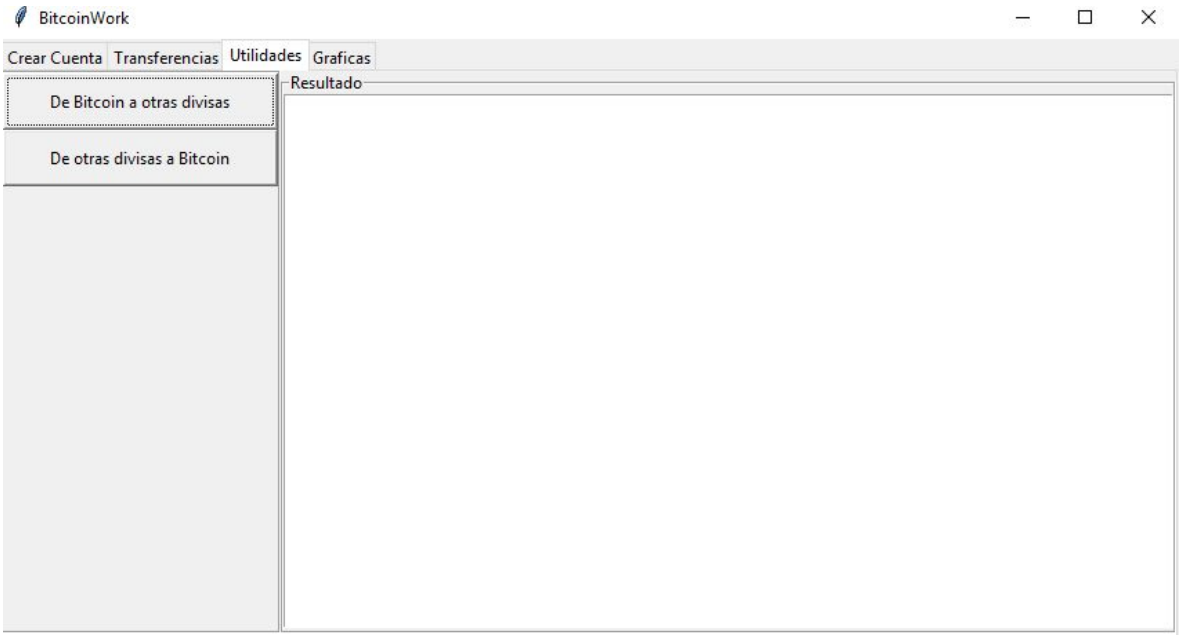

Figura 5: Interfaz gráfica Utilidades

Esta pestaña posee dos botones:

- De Bitcoin a otras divisas

Como bien indica su nombre, este botón permite pasar de Bitcoin a otras divisas. Al hacer clic se desplegará una nueva ventana:

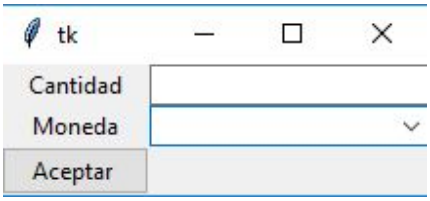

Figura 6: Interfaz gráfica inicial conversión

La cantidad es la cantidad de Bitcóin que quieres cambiar y la moneda es una combobox que permite seleccionar las monedas que están disponibles.

- De otras divisas a Bitcoins

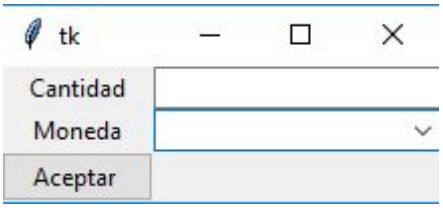

Figura 7: Interfaz gráfica inicial conversión 2

Como se puede observar, la ventana es la misma pero el concepto es distinto. En este caso, la cantidad se refiere a la cantidad de dinero que tienes y en moneda se debe especificar la abreviatura del nombre de la moneda que se posee.

La pestaña Gráficas se ve de la siguiente manera:

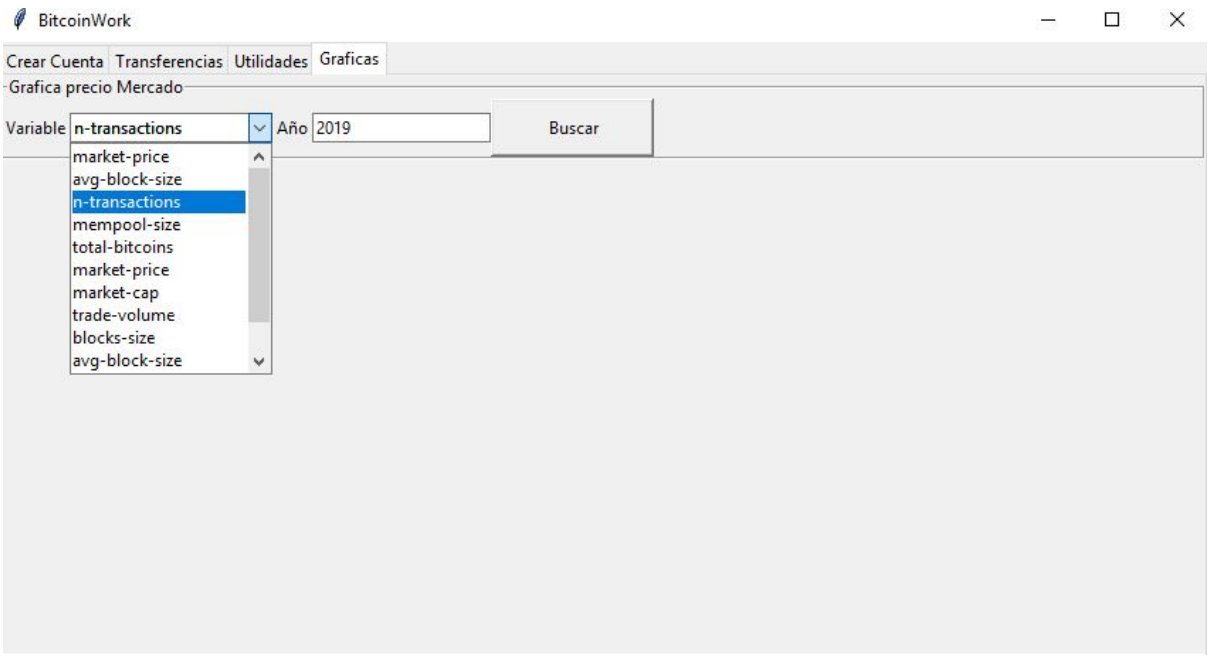

Figura 8: Interfaz gráfica Gráficos

Como podemos observar, hay una combo-box que contiene Los diferentes datos a analizar y por otro lado, tenemos un entrada donde especificamos el año final

a analiza

Al ejecutar un comando nos aparecerá una gráfica tal y como la siguiente expuesta:

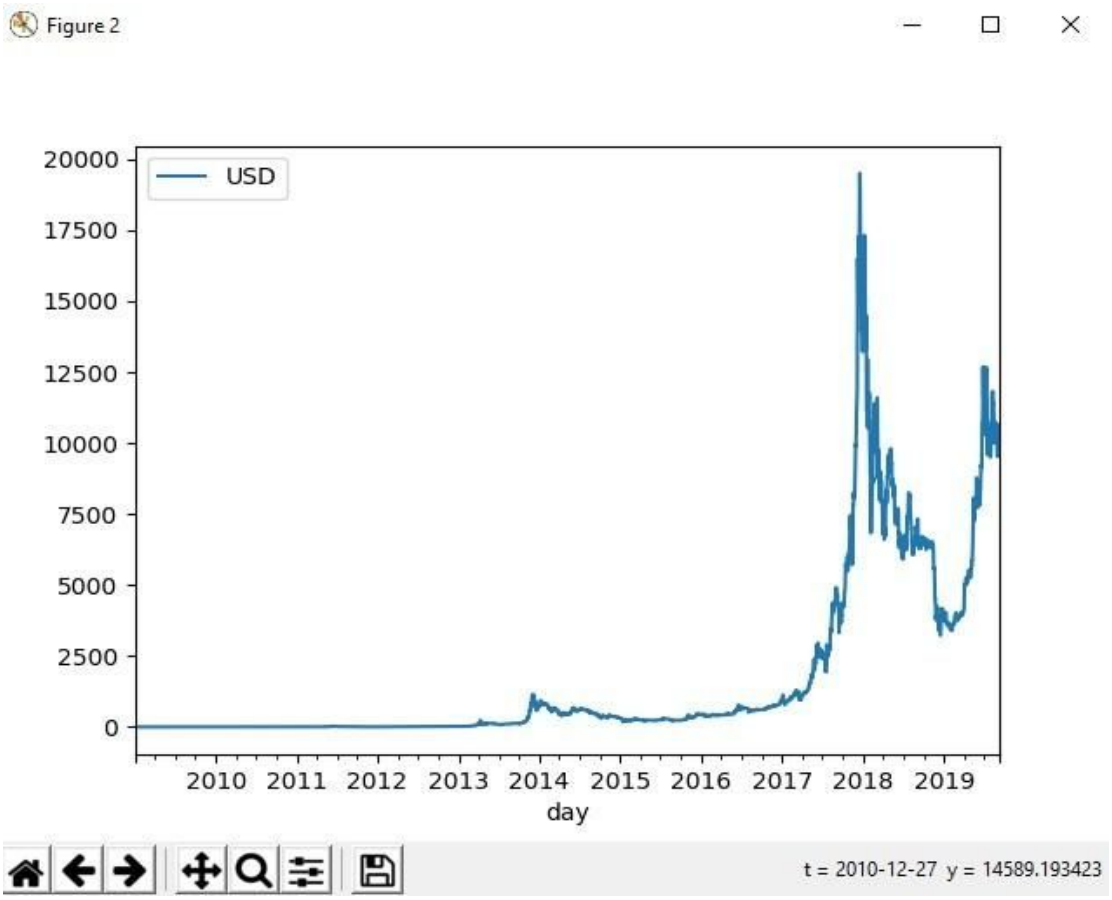

Figura 9: Interfaz gráfica MuestraGrafica

En este caso podemos observar, que la unidad mínima es el Dolar (USD) y, el dato mínimo es un día

3.5. Funcionamiento

Creación de cuenta

Al ejecutar la creación de cuenta se activa un trigger que está en el controlador:

```
self.view.Bcreateacc.bind("<Button>",self.TCreateAcc)
```

Figura 10: Trigger cuenta

Este trigger llama a la función TcreateAcc.

```
def TCreateAcc(self,event):
    acc = Block.createAcc(self.model)
    self.view.areacr.insert(
    tk.END, "Clave Privada: " + acc[2] + "\n\nClave publica: " + acc[0]  + "\n\nDireccion bitcoin: " + acc[1] + "\n\n")
```

Figura 11: Código TcreateAcc

Está a su vez llama a la función createAcc que se encuentra en modelo.

```
def createAcc(self):

    myPrivateKey = random_key()
    myPublicKey = privtopub(myPrivateKey)
    my_bitcoin_address = pubtoaddr(myPublicKey)
    self.ArrAcc.append([myPublicKey,my_bitcoin_address,0,myPrivateKey,1])
    return (myPublicKey,my_bitcoin_address,myPrivateKey)
```

Figura 12: Código createAcc

Se crea una clave privada y una clave pública que depende de la clave privada anteriormente creada. Posteriormente se crea una dirección Bitcoin a partir de la clave pública. Una vez hecho esto,se almacena la clave pública, la dirección de Bitcoin, la cantidad de Bitcoins que posee, la clave privada, y cuantas individuos hacen falta para poder realizar una transacción

Creación de cuenta multisignature

Al ejecutar la creación de cuenta multi signature se activa un trigger que está en el controlador:

```
self.view.Bcreatemultac.bind("<Button>",self.TCreateAccM)
```

Figura 13: Trigger cuenta múltiple

Este trigger llama a la función TcreateAccM

```
def TCreateAccM(self,event):
    self.view.auxWindow("Numero de cuentas","Numero Minimo")
    self.view.e3.bind("<Button>",self.TCreateAccM2)
    self.view.master.mainloop()
```

Figura 14: Codigo TcreateAccM

Esta función que genera una ventana auxiliar, la cual necesita del número de cuentas y, el número de cuentas mínimo para poder realizar una transferencia.Una vez se den estos y, se pulsa aceptar se activa el trigger TcreateAccM2 el cual recoge las

variables e1 (Número de cuentas) y e2 (Número mínimo).

```
def TCreateAccM2(self,event):
    a = int(self.view.e1.get())
    b = int(self.view.e2.get())
    #print(a,b)
    acc = Block.createMultAcc(self.model,int(a),int(b))
    for x in range(a):
        self.view.areacr.insert(tk.END,"\n\nCuenta " + str(x) + " " + acc[0][x])
    self.view.areacr.insert(tk.END,"\n\nCuenta Multiple  " + acc[1] + "/n")
    self.model.ShowMulAcc()
    self.view.master.destroy()
```

Figura 15: Código TcreateAccM2

Con estos datos se ejecuta la rutina de creación de cuentas múltiples, mostrada a continuación:

```
def createMultAcc(self,a,b):
    PublicandPrivate = []
    publicsKeys = []
    for x in range(a):
        myPrivateKey = random_key()
        myPublicKey1 = privtopub(myPrivateKey)
        PublicandPrivate.append([myPublicKey1,myPrivateKey])
        publicsKeys.append(myPublicKey1)
    my_multi_sig = mk_multisig_script(publicsKeys, b,a)
    my_multi_address = scriptaddr(my_multi_sig)
    for x in range(a):
        self.ArrAcc.append([publicsKeys[x],my_multi_address,0,PublicandPrivate[x][1],int(b)])
    return (publicsKeys,my_multi_address)
```

Figura 16: Código createMultAcc

En esta se puede observar la creación de direcciones privadas y públicas, igual al número de cuentas

Una vez se finaliza el proceso, se hace uso de la librería Bitcoin usando el comando mk_multisig_script, al cual se le pasa como parámetros todas las claves públicas que desees cuantas claves públicas son en total y el número mínimo para permitir una transferencia.

Una vez finalizado, se almacena la información para su futura utilización.

Transferencias

Antes de nada comentar que esta característica no está acabada, y es un añadido a el uso de las cuentas el cual tiene más un carácter “educativo” que formativo.

La intención de este apartado era simular cómo se comportaba la plataforma Blockchain a la hora de hacer transferencias y, así mostrar cómo funcionan estas.

Al ejecutar transferencias se activa un trigger que está en el controlador:

```
self.view.BsimpleTrans.bind("<Button>",self.Ttrans)
```

Figura 17: Trigger Transacciones

Este trigger llama a Ttrans

```
def Ttrans(self,event):
    aux = self.model.getAccs()
    aux2 = self.model.getDirs()
    self.view.auxWindow2("Cuenta emisora","Direccion Receptora",aux,aux2)
    self.view.e4.bind("<Button>",self.Ttrans2)
    self.view.master.mainloop()
```

Figura 18: Código Ttrans

El cual obtiene todas las claves públicas con getsAccs() y todas las direcciones con getDirs().

```
def getAccs (self):
    aux = []
    for x in range(len(self.ArrAcc)):
        aux.append(self.ArrAcc[x][0])
    return list(OrderedDict.fromkeys(aux))

def getDirs (self):
    aux = []
    for x in range(len(self.ArrAcc)):
        aux.append(self.ArrAcc[x][1])
    a = list(OrderedDict.fromkeys(aux))
    return a
```

Figura 19: Código getsAccs y codigo getDirs

Estas funciones recorrer el array que contiene las entradas de las múltiples cuentas creadas, y devuelven las claves públicas y las direcciones respectivamente como parámetros. Estos datos se obtienen para meterlos dentro de una combobox. Una vez finalizado la introducción de los datos, se ejecuta el trigger Ttras2.

La idea era que Ttras2 permitiera las transferencias de cuentas múltiples a simples y viceversa. Se consiguió hacer transferencias de cuentas simples a simples, y quedó planteado cómo se podría hacer de simple a múltiple, pero sin llegar a hacerlo funcionar.

Visualizar dinero cuentas

El objetivo de este apartado era mostrar el saldo de las distintas cuentas, que se creaban en el programa, debido a que el apartado de transferencias está incompleto y este apartado depende del anterior, no termina de mostrarse del todo bien. Aunque en caso de de arreglar el apartado anterior las modificaciones serán mínimas.

Al ejecutar visualizar dinero en cuentas se activa un trigger, el cual se encuentra en controlador:

```
self.view.BShowAcc.bind("<Button>",self.TShowAcc)
```

Figura 20: Trigger Mostrar cuentas

Esta función se llama a TshowAcc.

```
def TShowAcc(self,event):
    aux = self.model.Balances()
    for x in range(len(aux[0])):
        self.view.areaTr.insert(tk.END,'La cuenta ' + str(aux[0][x]) +' tiene '+ str(aux[1][x]) +' Bitcoins\n')
```

Figura 21: código TshowAcc

TshowAcc hace una llamada a la función balances que está dentro del modelo.

```
def Balances (self):
    aux = self.getDirs()

    aux2 = []
    for x in range(len(aux)):
        for y in range(len(self.ArrAcc)):
            if aux[x] == self.ArrAcc[y][1]:
                value = self.ArrAcc[y][2]
                aux2.append(value)
    print (aux2)
    return [aux,aux2]
```

Figura 22: código Balances

Balances recoge todas las direcciones que no están repetidas y, busca cada una de ellas en el array de cuentas para obtener su índice. Una vez se tiene el índice, se accede al array para obtener el balance de las diferentes direcciones . Unas vez se finaliza este proceso, se muestra por el área de la interfaz las direcciones de las cuentas y su correspondiente saldo.

De Bitcoins a otras divisas

Al ejecutar el botón de Bitcóin otras divisas se ejecuta el siguiente trigger

```
self.view.BtoO.bind("<Button>",self.TbtoO)
```

Figura 23: Trigger Bitcoin to Other

Dicho trigger llama a la función TbtoO la cual se encuentra en el controlador.

```
def TbtoO(self,event):
    aux = list(requests.get('https://blockchain.info/ticker').json().keys())
    self.view.auxWindow3("Cantidad","Moneda",aux)
    self.view.e3.bind("<Button>",self.TbtoO2)
    self.view.master.mainloop()
```

Figura 24: Codigo TbtoO

Esta función obtiene de Blockchain las distintas monedas disponibles. Luego de esto se crea una nueva ventana, a la cual se le pasa como parámetro todas las monedas factibles para asignarlos a una combobox.

Una vez se selecciona la cantidad de Bitcoins y, elegimos a qué moneda queremos, se activa el trigger TbtoO2.

```
def TbtoO2(self,event):
    aux = requests.get('https://blockchain.info/ticker').json()
    a = self.view.e1.get()
    b = self.view.e2.get()
    z = pd.DataFrame(aux)
    self.view.areaUt.insert(tk.END,str(a) + "Bitcoins ->" + str(float(a) *(z[b]['15m'])) + str(b) +"    \n")
    self.view.master.destroy()
```

Figura 25: Codigo TbtoO2

Este recoge los datos anteriormente dados y hace un request a Blockchain, para saber el valor aproximado que tiene esa cantidad de dinero. Sabiendo todo lo anterior realizamos una operación de conversión y mostramos en el área que cantidad de monedas corresponde a nuestros Bitcoins.

De otras divisas a Bitcoins

Al ejecutar el botón de otras divisas a Bitcoins se ejecuta el siguiente trigger

```
self.view.OtoB.bind("<Button>",self.TotoB)
```

Figura 26: Trigger Other to Bitcoin

Dicho trigger llama a la función TotoB la cual se encuentra en el controlador

```
def TotoB(self,event):
    aux = list(requests.get('https://blockchain.info/ticker').json().keys())
    self.view.auxWindow3("Cantidad","Moneda",aux)
    self.view.e3.bind("<Button>",self.TotoB2)
    self.view.master.mainloop()
```

Figura 27: Codigo Totob

Esta función obtiene exactamente lo mismo que TbtoO, diferenciándose en que llama a la función TotoB2 y ahora moneda es la abreviatura de la divisa que pasamos a Bitcoins.

Una vez se selecciona la cantidad y a que moneda tenemos se activa el trigger TotoB2

```
def TotoB2(self,event):
    a = self.view.e1.get()
    b = self.view.e2.get()
    print( exchangerates.to_btc(b, a))
    self.view.areaUt.insert(tk.END,str(a) + str(b) + " -> " + str(exchangerates.to_btc(b, a)) + " Bitcoins  \n")
    self.view.master.destroy()
```

Figura 28: Codigo Totob2

Esta función hace uso de la librería Blockchain para ejecutar exchangerates.to_btc, función a la cual, se le pasa como parámetro el dinero que tienes y la moneda en cuestión. Esta función te devuelve la conversión, esta conversión se imprime en el área de resultados.

Gráficas

Antes de poder ejecutar Gráficas, se debe seleccionar en el combobox que dato deseas analizar. Entre todos los disponibles se ha decidido añadir:

- market-price muestra el precio medio de Bitcóin en el mercado.
- avg-block-size muestra tamaño promedio del bloque en las últimas 24 horas.
- n-transactions enseña el número de transacciones que se hacen al día.
- mempool-size enseña total de transacciones a la espera de ser confirmadas.
- total-bitcoins visualiza Total de Bitcoins existentes.
- market-cap muestra el valor total de los Bitcoins en circulación.
- trade-volume enseña el valor total en dólares de las transacciones que se hacen.
- blocks-size visualiza el tamaño de la cadena de bloques en MB.
- avg-block-size muestra el tamaño promedio por bloque.
- n-transactions-per-block visualiza el número de transacciones por bloque.
- median-confirmation-time enseña la mediana de tiempo que tarda una transacción en ser aceptada en un bloque minado.

Una vez sabemos conocemos las distintas opciones disponibles, debemos elegir hasta qué año queremos analizar. Partiendo de la base que se empezó a desarrollar en el año 2000 debemos elegir un año superior a este. Una vez hecho esto nos mostrará los datos del intervalo.

Una vez lo ejecutamos se activará el trigger del botón

```
self.view.grafiB.bind("<Button>",self.TGrafics)
```

Figura 29: Trigger TGrafics

Este trigger llamara a la función TGráfics

```
def TGrafics(self,event):
    a = self.view.variableG.get()
    b = str(int(self.view.anioG.get()) - 2000)
    print('https://api.blockchain.info/charts/' + a +'?timespan='+ b +'years&format=json&sampled=false')
    d = requests.get('https://api.blockchain.info/charts/' + a +'?timespan='+ b +'years&format=json&sampled=false').json()
    df = pd.DataFrame(d['values'])
    df['x'] = pd.to_datetime(df['x'].astype(int), unit='s')
    df.columns = [d['period'], d['unit']]
    df.plot(x = d['period'], y = d['unit'])
    plt.show()
```

Figura 30: Codigo TGrafics

Esta función utiliza la variable a analizar y, el año hasta el cual, queremos analizar los datos. Por como trabaja la api de Blockchain, debemos quitarle al año escogido 2000 para darle a entender que queremos ver X años desde el nacimiento del Bitcoins, donde X es la resta entre, el valor introducido y 2000.

Una vez hecho todo esto, obtenemos gracias al request un Json, que contiene los datos en variables llamadas X e Y. Una vez tenemos los datos lo primero que debemos hacer es, modificar el valor de X de todas los valores del Json, Ya que están representados con Unix timestamp, para ello usamos un Data Frame. Una vez ya se encuentran modificamos los datos, modificamos el nombre de las distintas columnas con sus correspondientes parámetros y lo mostramos el resultado por pantalla usando plt.show().

4. Presupuesto

Para el desarrollo de este proyecto se utilizarán herramientas de software libre por lo cual los costes serán reducidos.

La partida de gastos más importante será la de personal, ya que sin un broker o analista financiero muy especializado este proyecto es inviable.

Costes directos		
Personal		
Programador/a	300 horas x 50 euros/hora	15.000 euros
Broker / Analista financiero	100 horas x 75 euros/hora	7.500 euros
Equipamiento		
Equipamiento informáticos	1.000 euros	1.000 euros
Costes indirectos		
Agua, luz, fotocopias, material de oficina, tóners, …	10% sobre el total de costes directos del proyecto	2.350 euros

Coste total de proyecto: 25.850 euros
Tiempo estimado de ejecución: 4 meses.

5. Conclusiones y líneas futuras

He obtenido los resultados que esperaba dentro de lo que quería. Teniendo en cuenta que no había trabajado nunca antes con muchas de las herramientas usadas, se podría decir que he adquirido conocimientos y he conseguido desenvolverme mejor; no solo con el Bitcoins en sí sino también con las tecnologías que lo rodean como pueden ser BlockChain, Hashes, Pow, claves privadas, claves públicas, Criptografía asimétrica entre otros. Sabiendo todas las tecnologías que hay detrás aún me queda por aprender pero he afianzado una base que me permitirá avanzar hacia un futuro.

Con respecto al futuro del proyecto se ha podido observar que tengo varias ideas en mente, una de esas ideas era implementar un "Blockchain offline" ya no solo para mostrar cómo funciona el Bitcóin por dentro si no también para poder ver como funciona el cifrado asimétrico que este tiene detrás haciendo de este proyecto un proyecto formativo para aquellas personas que conocen poco del tema.

Así mismo también se podría explorar la opción de convertir este proyecto en una cartera offline de distintas criptomonedas y integrar alguna solución que nos permita en el caso de que queramos poder hacer transferencias a otras cuentas.

También se podría explorar la idea de coger los distintos datos de Blockchain y añadirles modificaciones para permitir, por ejemplo, pasar una gráficas a escala algorítmica, o incluso a hacer uso de algoritmos para prever el precio del Bitcoin o se podría diseñar una IA que entrenandola te sepa decir que es lo siguiente que va a pasar con respecto al bitcóin.

Como se puede observar existen posibilidades ilimitadas para un proyecto como este ya que trabajamos con una tecnología nueva donde aún está todo por inventar.

5. Summary and Conclusions

I have obtained the results I expected within what I wanted. Taking into account that I had never worked before with many of the tools used on this project, I could say that I have acquired new knowledge and perform a better developing; not only with the Bitcoins, but also with the surrounding technologies such as: BlockChain, Hashes, Pow, private keys, public keys, asymmetric cryptography among others. Knowing all the technologies behind Bitcoins, I still had a lot to learn, but I have strengthened a consistent base that will allow me to move towards the future.

With regard to the future of the project, it has been observed that I have several ideas in mind. One of those ideas was to implement an *"offline Blockchain",* not only to show how Bitcoin works inside but also to be able to see how asymmetric encryption works behind, making this project an educational project for those who know little about the subject.

Likewise, it could also be explored the option of converting this project into a wallet of different cryptocurrencies and integrate multiples solutions that allows us to be able to do transfers to another accounts.

As well, you could explore the idea of taking the different Blockchain data and adding modifications to allow, for example, convert a graph to an algorithmic scale, or even to make use of algorithms to predict the price of Bitcoin or an IA. The IA could be trained to show you how is the next thing that will happen respect to Bitcoin.

As you can see, there are unlimited possibilities for a project like, inasmuch as we work with a new technology where everything is still to be invented.

Bibliografía

[1] Bitcoin - https://en.bitcoinwiki.org/wiki/Bitcoin

[2] Blockchain - https://en.wikipedia.org/wiki/Blockchain

[3] Empresas que usan Blockchain - https://101blockchains.com/es/empresas-implementando-blockchain/

[4] Empresas que aceptan Bitcoins - https://www.muypymes.com/2016/03/10/empresas-pagar-bitcoins

[5] p2p - https://www.digitalcitizen.life/what-is-p2p-peer-to-peer

[6] Proof of work - https://en.bitcoin.it/wiki/Proof_of_work

[7] Ecash - https://en.wikipedia.org/wiki/Ecash

[8] Hashcash - https://es.wikipedia.org/wiki/Hashcash

[9] Wallet - https://academy.bit2me.com/tipos-wallets-bitcoin/

[10] Clave privada y publica - https://medium.com/coinmonks/blockchain-public-private-key-cryptography-in-a-nutshell-b7776e475e7c

[11]Multi Signature - https://en.bitcoin.it/wiki/Multisignature

[12]ECDSA - https://es.wikipedia.org/wiki/ECDSA

[13] Python - https://www.python.org/

[14] Tinker - https://docs.python.org/3/library/tkinter.html

[15] Pandas - https://pandas.pydata.org/

[16] Numpy - https://numpy.org/

[17] Requests - http://sobretecnologia.org/la-importancia-del-bitcoin/

[18] SSL - https://docs.python.org/3/library/ssl.html