

2020

Voting App

Manual de usuario

Tabla de contenido

1. Configuración del entorno.....	3
1.1. Software necesario aprender sobre Voting App.....	3
1.1.1. Blockchain privada.....	3
1.1.2. Navegador	7
1.1.3. Software adicional.....	17
1.1.4. Datos adicionales para desarrolladores.	19
1.2. Despliegue de contratos	22
2. Usando Voting App	28
2.1. Registrando mesas de votación	28
2.2. Registrando candidatos	34
2.3. Realizando la Consulta de opinión.....	41
2.4. Consultando resultados	50
2.4.1. Resultados para el CEU	50
2.4.2. Resultados público en general.....	50
2.4.3. Auditando resultados.....	51

Introducción

Voting App es una aplicativo desarrollado como una prueba de concepto para simular procesos electorales, o simplemente Consultas de opinión como las que se llevan a cabo en la Universidad del Quindío. Esta solución permite soportar gran parte de lo necesario para un proceso electoral, sin embargo, debido a las dificultades encontradas en la tecnología que se está evaluando, este está especializado en la fase de votar y escrutinio únicamente. Además, cabe aclarar, que es una prueba de concepto y se la ha desarrollado como tal. Una vez realizada la aclaración, se procede a explicar cómo está estructurado el documento, de tal forma que el lector pueda sacar el máximo provecho de la información.

El capítulo 1 busca configurar el entorno de trabajo con las herramientas necesarias para poder hacer uso de Voting App sin percance alguno. Pensando en lo anterior, se sigue un paso a paso de cómo instalar el software necesario que será usado en el capítulo siguiente; instalado el software necesario, se puede configurar una Blockchain privada de forma local para hacer pruebas cuando se despliegue los contratos, que serán tratados más adelante en este mismo apartado. Una vez que se tiene una Blockchain privada lista para ser usada, se procede a configurar el navegador para que esté en la capacidad de trabajar con tecnologías descentralizadas. Posteriormente, se buscará integrarlo a la Blockchain con el fin de que Voting App pueda usar la Blockchain. Finalmente, como se había mencionado, se despliegan los contratos necesarios a través de un paso a paso.

El capítulo 2 está enfocado directamente en hacer uso del aplicativo, por ende, inicia explicando cómo registrar candidatos, qué opciones se deben habilitar, y dónde está cada una. Posteriormente se explica cómo registrar las mesas de votación, siendo de manera similar al anterior, necesita preparar los contratos para el registro de mesas y cómo terminar correctamente con este proceso. Acto seguido, se explica cómo se debe realizar la Consulta de opinión o jornada electoral; en esta sección se explica desde habilitar los contratos para votar, autorizar tokens para las mesas, autorizar votos, votar, cierre de la jornada electoral y habilitación del escrutinio público. Como consecuencia de habilitar el escrutinio público, existen 3 formas de consultar los resultados, una empleada por el Representante del Consejo Electoral (de ahora en adelante representante del CEU) y las otras dos para todo tipo de público. En estas dos últimas las personas pueden consultar los votos de dos maneras: consultando los resultados finales o auditando los votos obtenidos por cada candidato en cada mesa.

1. Configuración del entorno

Voting App se sustenta sobre tecnologías descentralizadas, y en este caso en particular, sobre una Blockchain de Ethereum de tipo privada; tecnologías que no son manipuladas directamente por los navegadores por el motivo de no estar diseñados para esto. Con el fin de hacer que este se pueda comunicar con la Blockchain privada que se ejecuta en memoria, se procede a realizar las configuraciones pertinentes para que esto se pueda dar satisfactoriamente. Estas configuraciones inician por instalar el software necesario siguiendo los pasos de la sección 1.1.

1.1. Software necesario aprender sobre Voting App

Esta sección es una explicación para las personas que deseen implementar esta aplicación en un computador personal, contiene información para preparar un entorno de trabajo para personas que no están involucradas en el desarrollo de software (en una sección siguiente se hablará de aquellas que buscan modificar el código del presente). Se encuentra dividido por apartados para conseguir información particular de una forma más rápida.

1.1.1. Blockchain privada

En primer lugar, se procede a instalar el software de Ganache en su [versión 2.0.1](#), el cual fue desarrollado por Truffle Suite. Este software permite simular una Blockchain real, pero de forma local y en memoria, por lo tanto, se tienen las mismas funcionalidades que la red principal de Ethereum la cual es pública, a diferencia de la que ofrece Ganache, siendo esta privada.

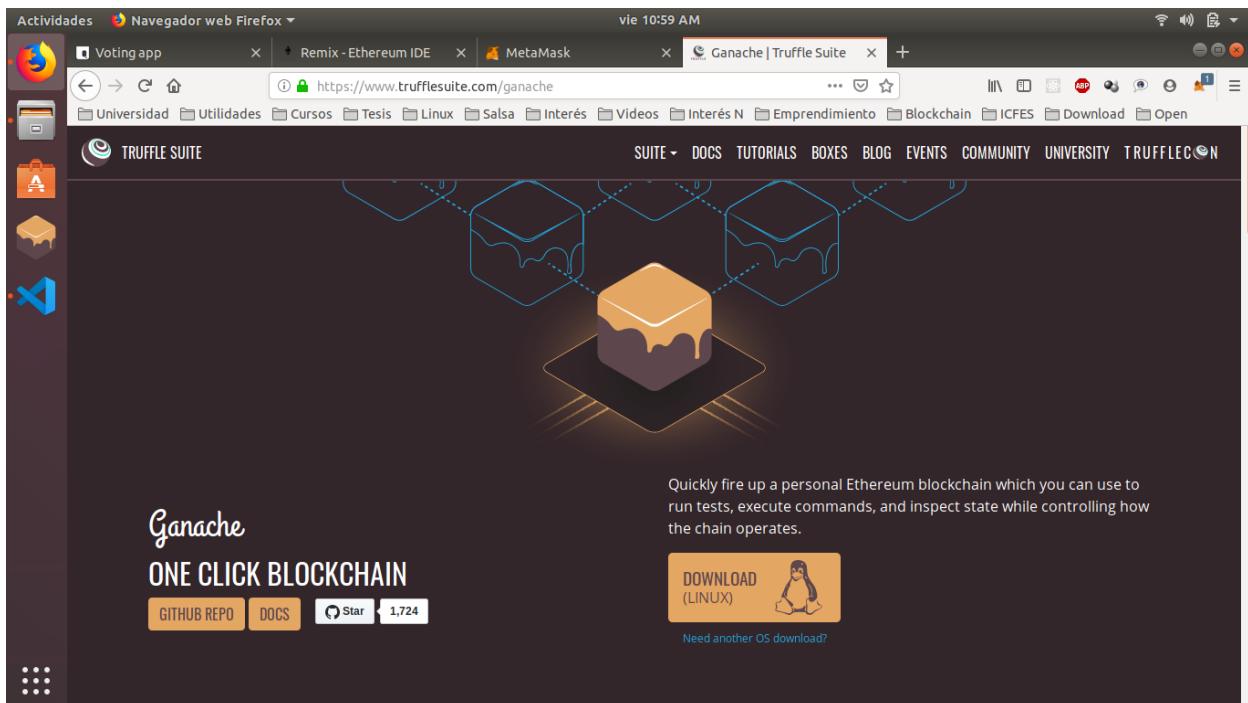


Figura 1: Página principal de Ganache. Fuente propia.

Ganache es empleado para la depuración del código de los diferentes contratos que se vayan a emplear en un proyecto, ya que el despliegue y uso de transacciones sobre la red pública principal de Ethereum, cuesta dinero real, usando Ganache no.

Para poder obtener el software de Ganache se debe ingresar a la página web de [Ganache](#), la cual tiene la apariencia, en el momento en que se trabajó este manual, como aparece en la *Figura 1*. En esta página se debe seleccionar el tipo de sistema operativo, que, en este caso, detecta automáticamente que es Linux, aunque también lo está para Windows.

Cuando se descargue el software y se dé doble clic sobre el mismo; si es el caso, acepte ejecutarlo como administrador si usa Windows. Al hacer esto, aparece una interfaz donde se tienen los diferentes Workspace, de clic en New workspace y Ganache lo llevará a un interfaz como la que se puede apreciar en la *Figura 2*. Si está en Linux descargue el archivo Ganache-2.0.1.AppImage y de doble clic luego de configurarlo para permitir iniciararlo como un ejecutable.

Nota: Si una vez creado el workspace, regresa a la interfaz principal de Ganache y aparece que no tiene saldo (0.00 ETH), vuelva a crearlo usando la opción Quickstart.

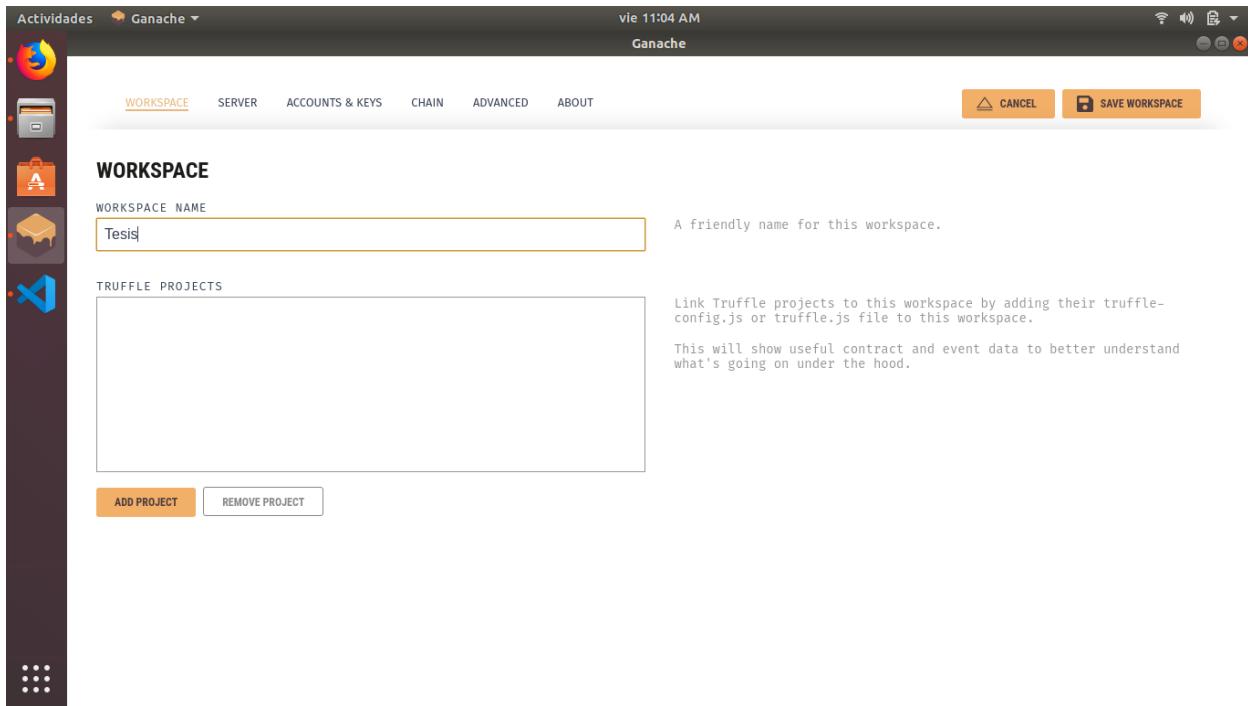


Figura 2: Interfaz de inicio de Ganache. Fuente propia.

Esta interfaz permite configurar el servidor de Ganache de forma personalizada, por lo tanto, se procede a configurar un nombre, puede ser cualquiera, en este caso particular se lo llamó Tesis al workspace que se va a crear. Ganache también es capaz de configurar un proyecto a partir de las configuraciones definidas en un archivo de javascript de Truffle (herramienta tratada más adelante), el cual se llama truffle-config.js y está localizado a partir del directorio

raíz del proyecto, en la dirección *tesis/truffle*. Sin embargo, en esta ocasión no se trabajó de esa manera.

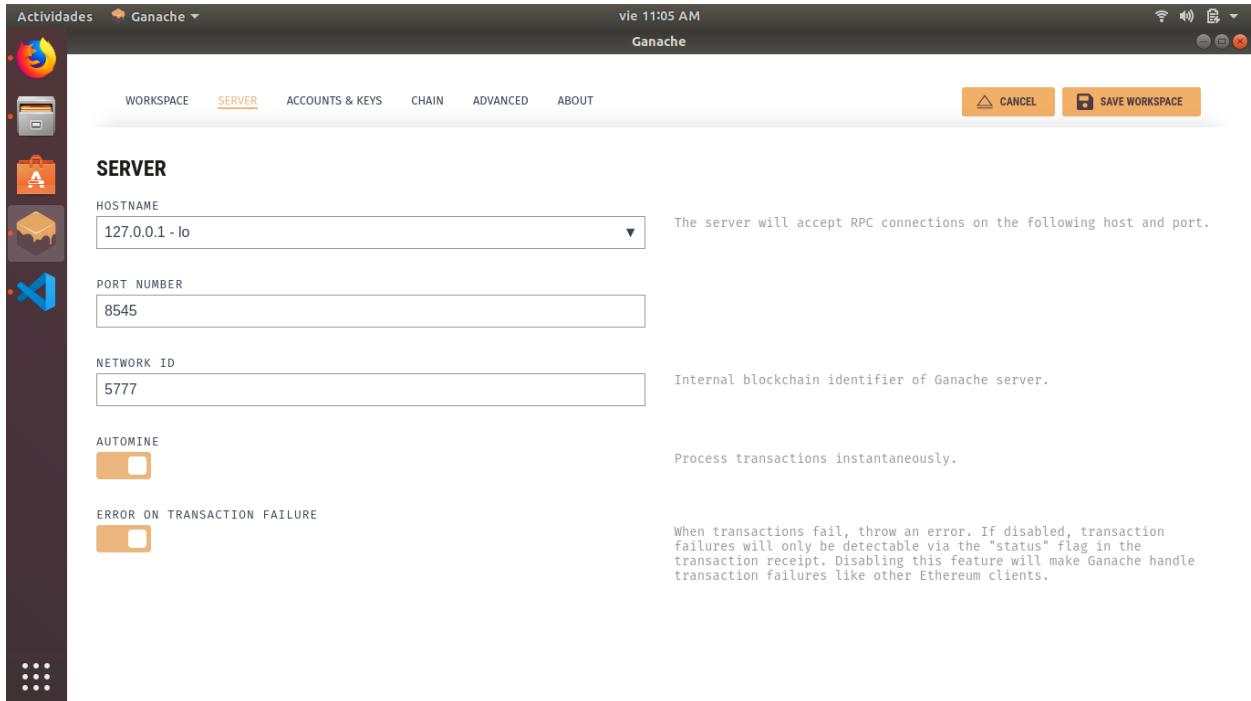


Figura 3: Configurando el servidor de Ganache. Fuente propia.

Como siguiente paso de la configuración del servidor, se debe añadir la IP del servidor donde estará la Blockchain, dejar la 127.0.0.1 para indicar que está de forma local; el puerto por el cual estará escuchando el servidor de Ganache, que en este caso se elige el 8545. El identificador es empleado para el caso de que se tengan varias Blockchain, poder acceder a una específica. Acto seguido, se debe de activar la opción de Autominado, para que los bloques con las transacciones sean añadidos a la blockchain de forma automática.

Hasta el momento se tiene diligenciado el nombre del proyecto y las configuraciones de red necesarias para que Truffle, Metamask y Web3JS (tecnologías tratadas más adelante) puedan conectarse satisfactoriamente a la Blockchain principal. Por lo tanto, es necesario sumar a lo anterior, el hecho de poder crear tantas cuentas como se necesite para hacer pruebas, por ejemplo. Si creó el workspace usando la opción Quickstart, no podrá cambiarlas, y no será necesario.

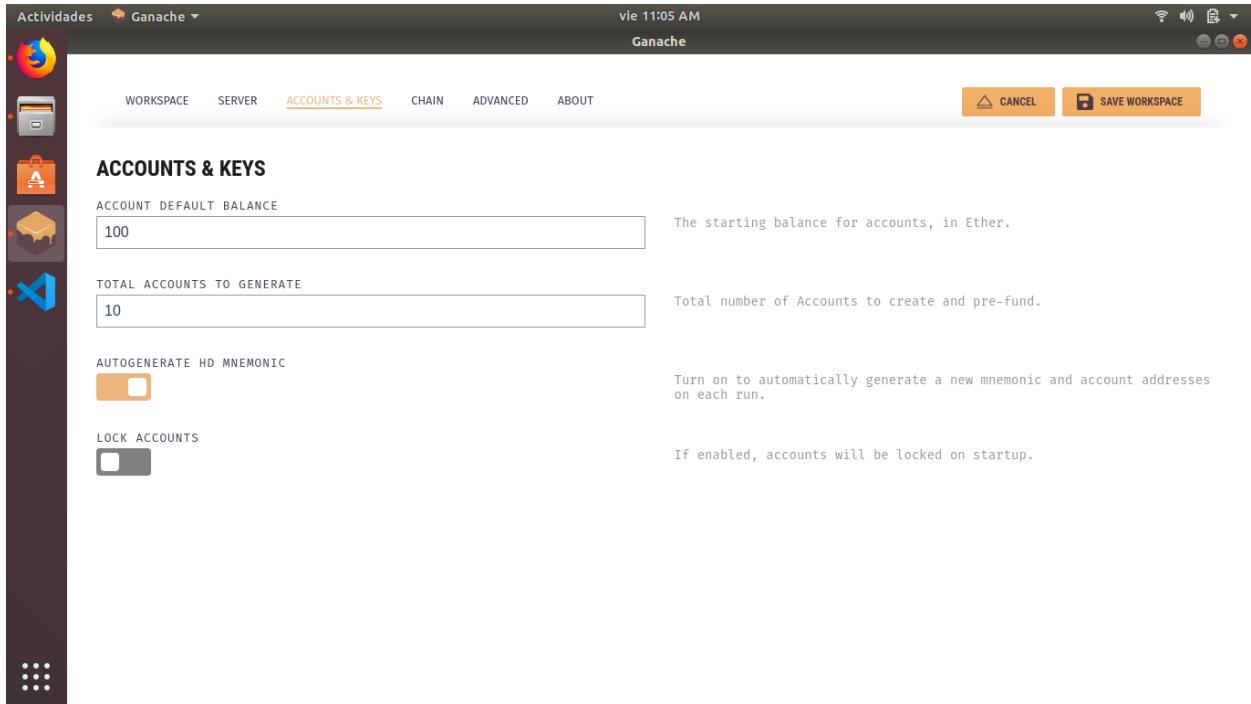


Figura 4: Configuraciones de las cuentas necesarias. Fuente propia.

En la imagen que se puede visualizar en la *Figura 4*, se aprecia las diferentes configuraciones en cuanto a las cuentas, es necesario tener precaución al hacerlo, ya que estos cambios no se pueden modificar una vez creadas, y es suficiente con dejar configurado como aparece en la imagen. Procediendo con la explicación de la imagen citada, se tiene la cantidad de ether (falso) que cada una de las cuentas tendrá como balance inicial, esto sirve para poder realizar transacciones, no se recomienda dejar en 0, ya que no podrá usar la aplicación y saltarán errores. En el campo siguiente, se tiene la cantidad total de cuentas que se desea crear dependiendo de las necesidades del proyecto. Finalmente, se debe habilitar la opción que autogenerar MNEMONIC. Las doce palabras que genera esta opción, permiten que la billetera (wallet) de cuentas de Ganache, pueda ser importada por otros gestores de wallets empleando dichas palabras, en este caso, serán importadas por Metamask.

Una vez se tienen estas configuraciones realizadas como se puede apreciar en las diferentes imágenes citadas y explicadas, se presiona el botón que se encuentra en la parte superior derecha que tiene la etiqueta SAVE WORKSPACE.

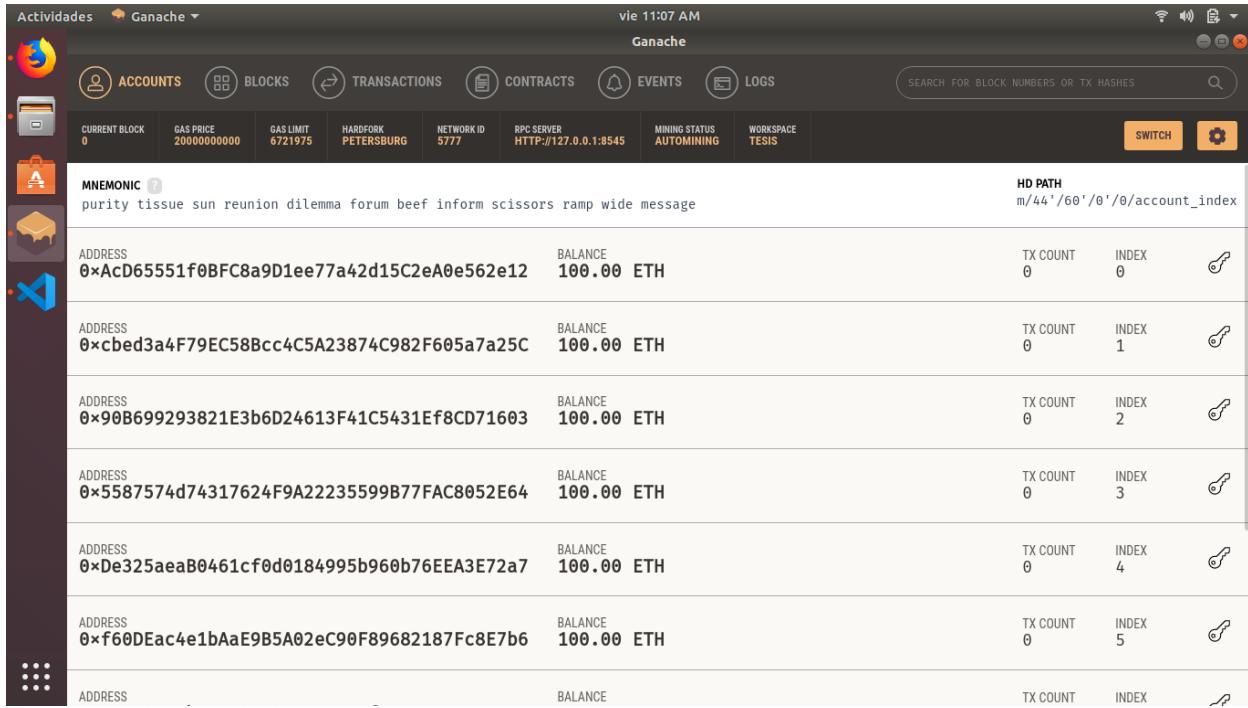


Figura 5: Interfaz principal luego de configurar la red Blockchain. Fuente propia.

Cuando se guardan la diferentes configuraciones que se han establecido en anteriores pasos, ganache configura una red Blockchain privada con dichas configuraciones. En la Figura 5 se puede apreciar un resumen de estas, se tiene una barra con el número de bloques añadidos a la red, las configuraciones por defecto del gas, identificador de la red, el servidor RPC (protocolo de comunicación), la opción de Autominado, el nombre del workspace actual, cambiar de workspace (SWITCH), modificar algunas opciones del workspace (ícono del engranaje) y la lista de cuentas que asignamos anteriormente.

Si todos los pasos se han seguido correctamente, debería tener una Blockchain privada totalmente operativa y lista para ser usada. Por precaución, cambie de workspace (usando el botón SWITCH que está en la esquina superior derecha), y vuelva y seleccione su workspace, si luego de hacer esto, las cuentas continúan teniendo 100.00 ETH, continúe con el tutorial. En caso contrario, vuelva a crear el workspace usando la opción Quickstart en lugar de New workspace. Esto es dado que en ocasiones, los saldos de las cuentas desaparecen repentinamente y la aplicación no funcionará.

1.1.2. Navegador.

Ahora es momento de seleccionar un navegador y puede ser uno de su preferencia, en este caso, se ha seleccionado Mozilla Firefox. Se debe tener precaución con la selección del navegador dado que debe ser compatible con el plugin Metamask, para conocer si lo es, puede ingresar a la página oficial de [Metamask](#).

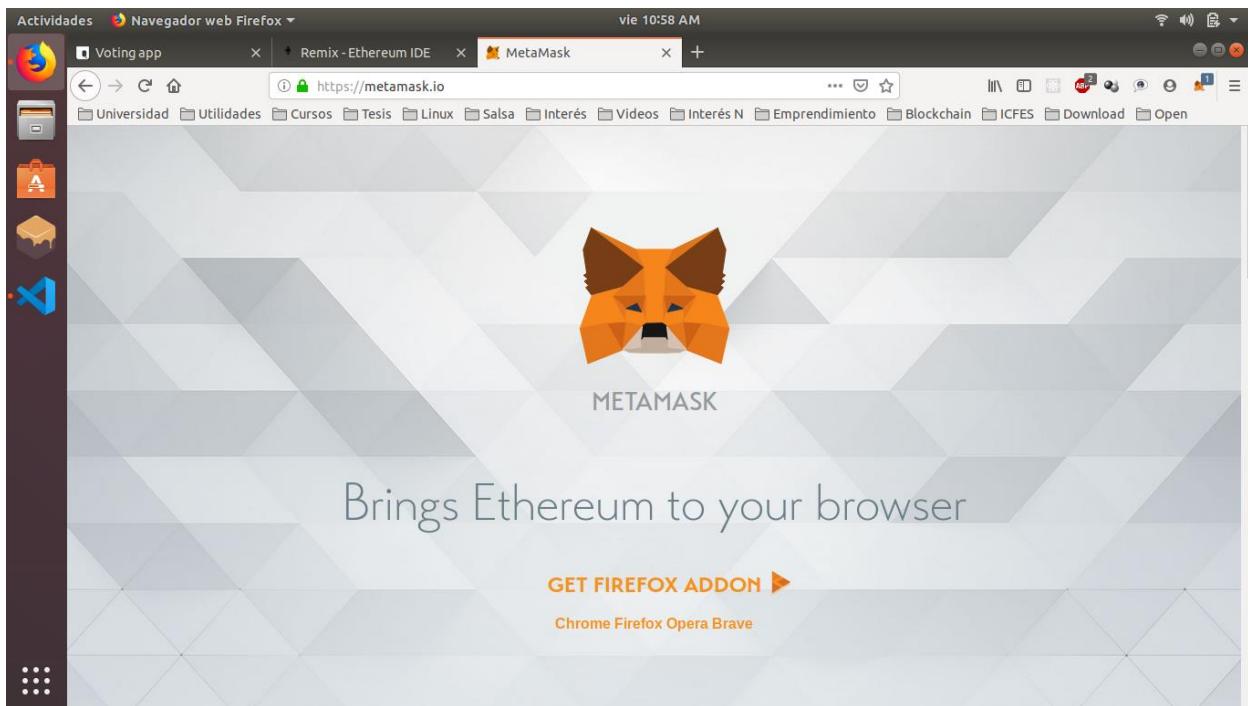


Figura 6: Página principal de Metamask. Fuente propia.

Con el fin de gestionar las cuentas que se tiene creadas en Ganache, se utilizará el plugin anteriormente mencionado el cual recibe el nombre de Metamask. Este plugin permite gestionar las cuentas y sus saldos, y ayuda al navegador utilizar tecnologías descentralizadas como las Dapps, las cuales son aplicaciones descentralizadas; por extensión, podrá comunicarse con la Blockchain a través de este plugin.

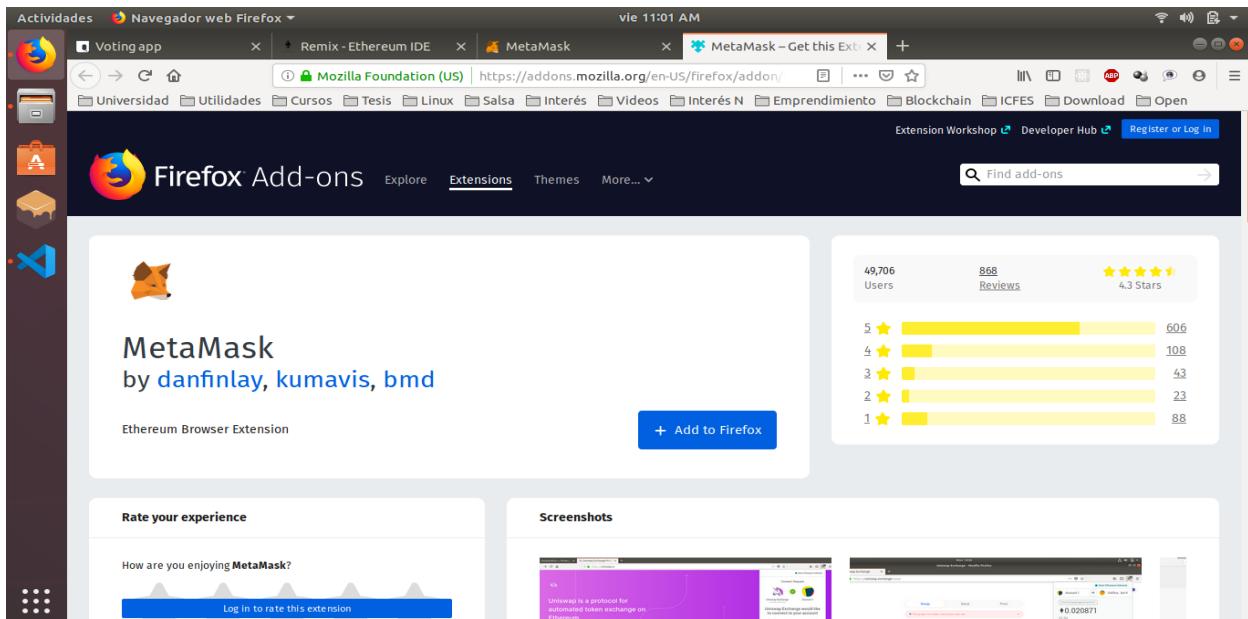


Figura 7: Instalando plugin Metamask. Fuente propia.

Como una herramienta necesaria para autenticar y usar las cuentas que se tienen en Ganache, se debe añadir Metamask al navegador que se emplee. Para tal fin, se debe realizar los pasos que se explican a partir de la *Figura 7*.

Con el fin de añadir Metamask al navegador, ingrese a la página oficial de [Metamask](#), seleccione el navegador adecuado (ver *Figura 6*) y este lo redirige a la tienda de complementos del navegador, si está siguiendo este manual con el navegador Firefox puede abrir el siguiente enlace que lo llevará hacia el [Plugin de Metamask](#) directamente.

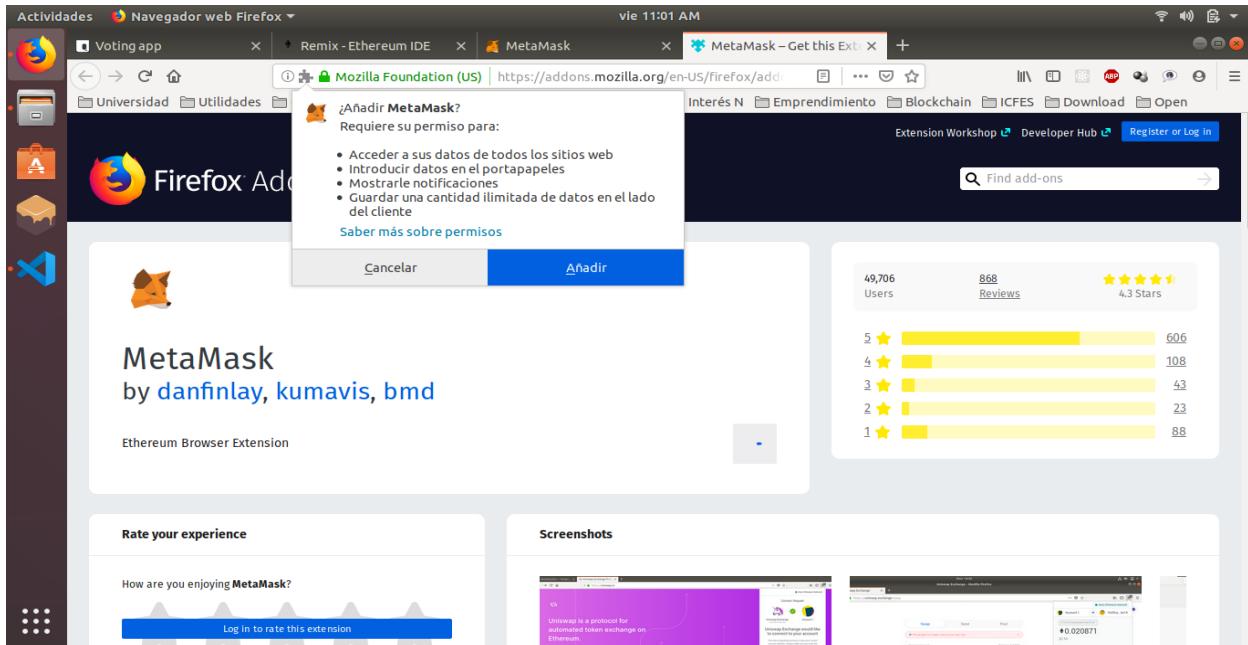


Figura 8: Autorizando instalación de Metamask. Fuente propia.

Una vez esté en la página que se presenta en la *Figura 7*, de clic sobre el botón que tiene la etiqueta Add to Firefox, mediante el cual Firefox comienza a añadir el plugin, y posteriormente le pregunta si desea confirmar la acción, tal como se aprecia en la *Figura 8*.

Acto seguido, se debe presionar el botón que tiene la etiqueta Añadir (ver *Figura 8*).

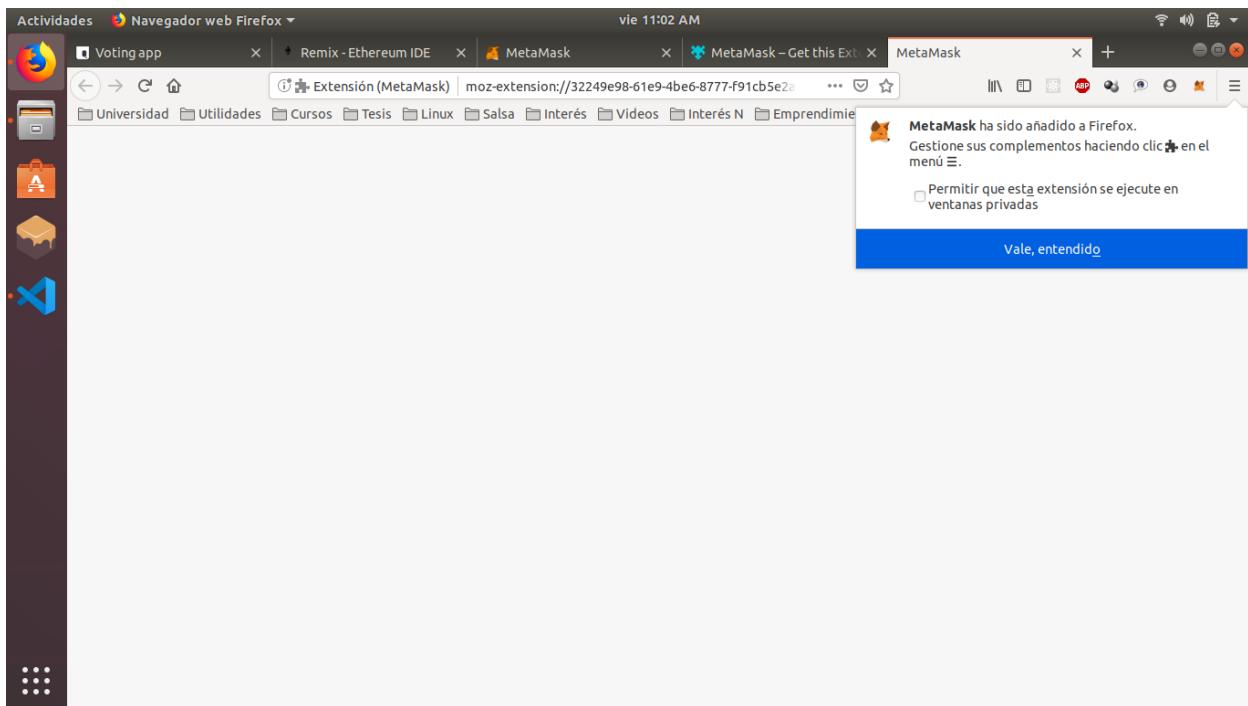


Figura 9: Confirmación de plugin añadido. Fuente propia.

Cuando se presente la página que se puede visualizar en la *Figura 9*, se presiona el botón que tiene la etiqueta de Vale, entendido. Esta acción carga una página en blanco y, cuando se da clic sobre el plugin Metamask, cargará la página de bienvenida de Metamask en el navegador en una nueva pestaña, al igual que puede ser observada en la parte superior derecha al dar clic sobre el ícono de Metamask, tal y como se puede apreciar en la

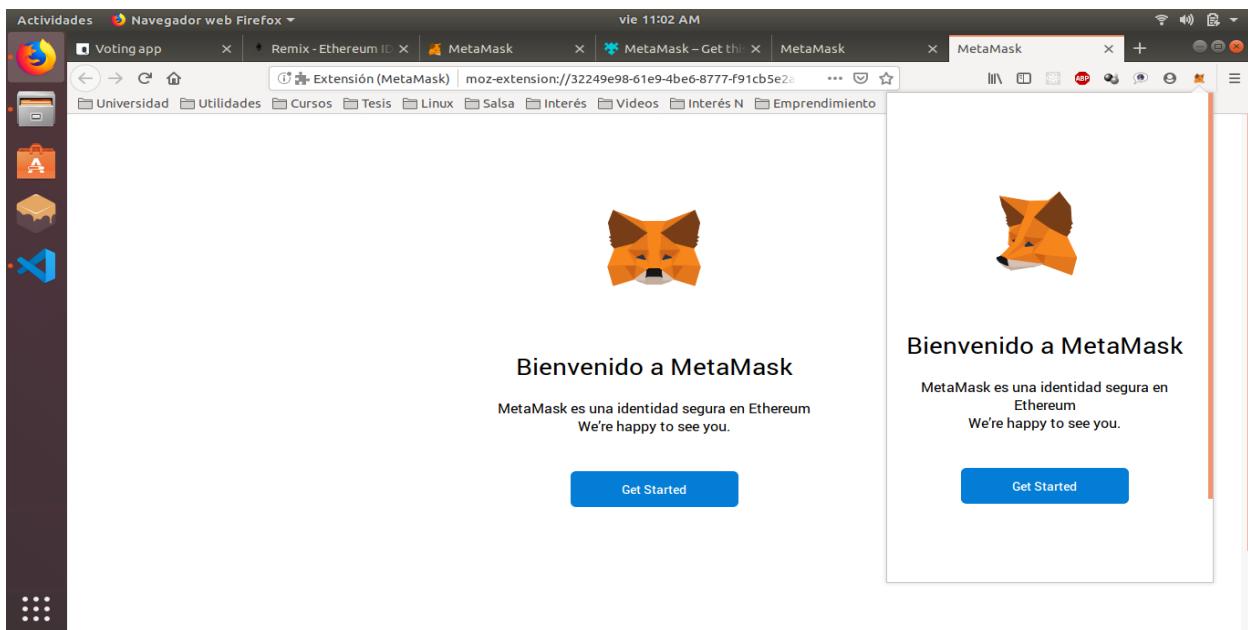


Figura 10: Página de bienvenida de Metamask. Fuente propia.

La página web que se puede ver en la Figura 10 es una buena señal, indica que todo ha sido realizado correctamente y que Metamask está listo para ser configurado para importar la wallet (billetera) con las cuentas de Ganache. Por lo que se presiona el botón que tiene la etiqueta de Get Started de la página web, para continuar con el proceso de importar la wallet de Ganache a Metamask.

Luego de presionar el botón ya mencionado, Metamask carga la página que se puede visualizar en la *Figura 11*.

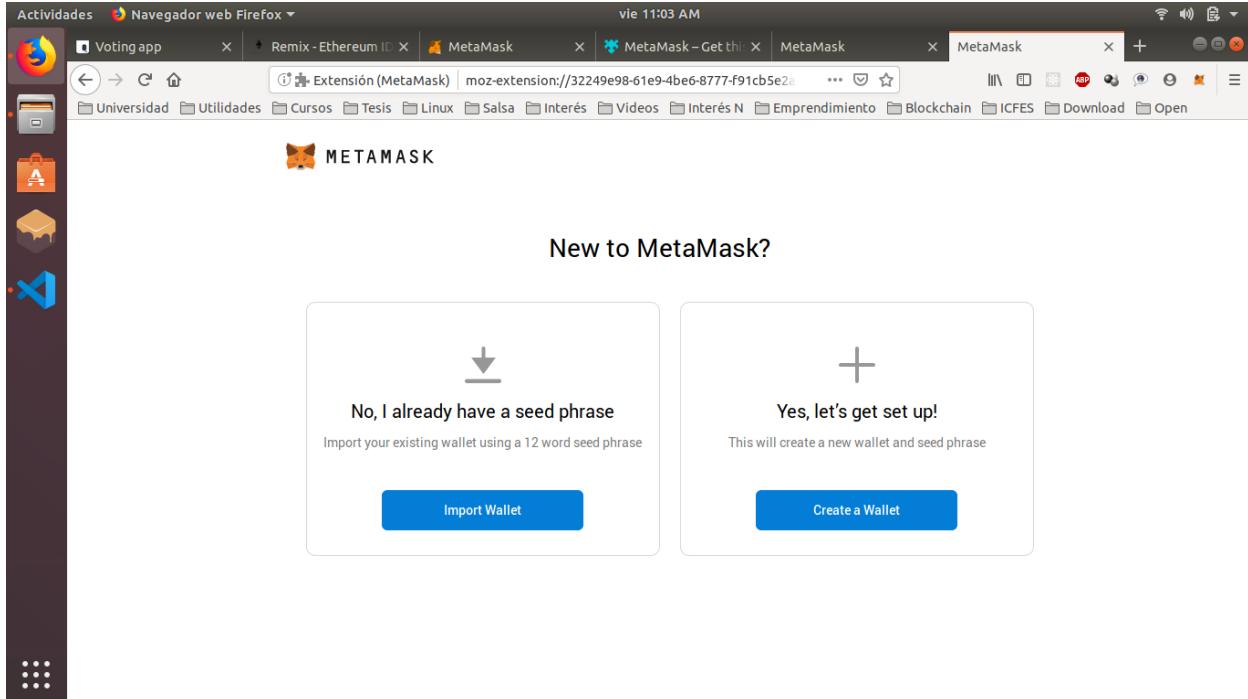


Figura 11: Asistente de configuración de Metamask. Fuente propia.

Como se mencionó anteriormente en la *Figura 4*, se activó la opción de generar una frase MNEMONIC de 12 palabras, la cual será empleada en este momento para importar la billetera (wallet) de Ganache. Para lograr esto, se debe usar la opción *No, I already have a seed phrase* y se presiona el botón que tiene la etiqueta Import Wallet, cuando se realice esta opción, Metamask cargará la página que se puede apreciar en la *Figura 12*.

Esta es una página que invita a las diferentes usuarios y desarrolladores, a contribuir en el mejoramiento de Metamask enviando datos que le son de utilidad para desarrollar versiones futuras más estables. También aclara qué cosas no hace Metamask con dicha información. Este punto, se deja decisión del lector, si desea contribuir o no a mejorar Metamask.

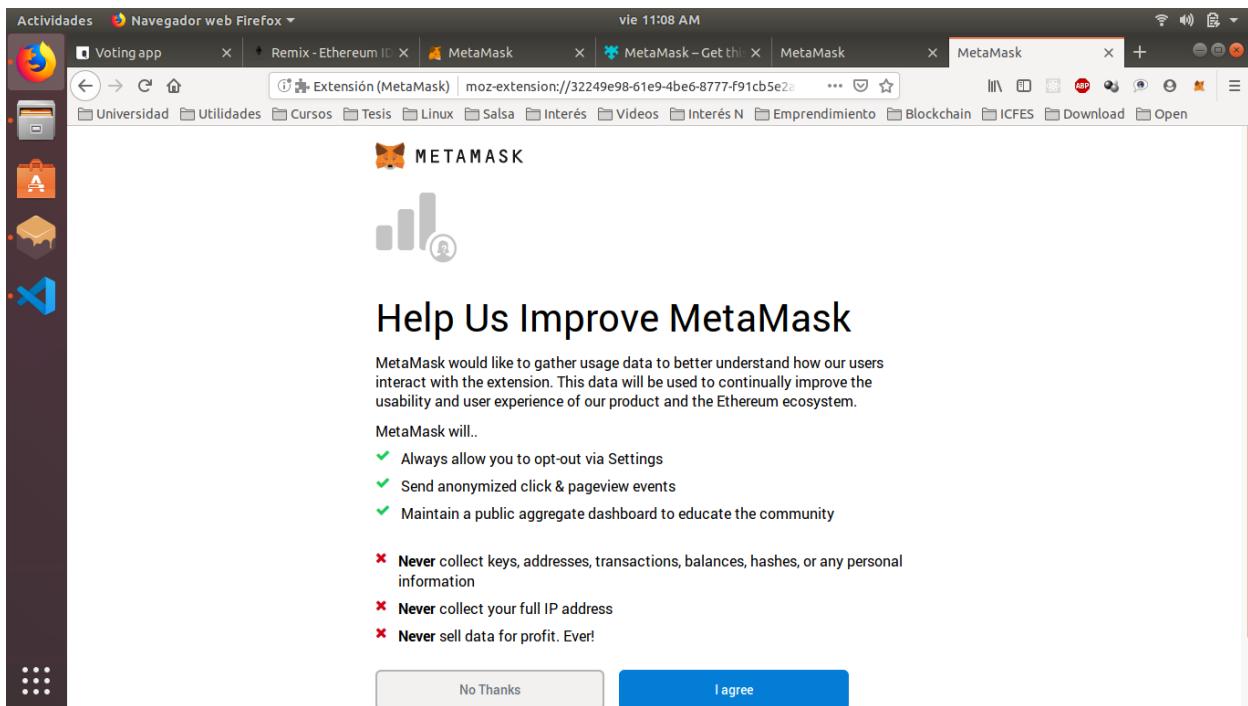


Figura 12: Página web de ayuda a mejorar Metamask. Fuente propia.

Continuando con la explicación para importar la wallet, una vez se acepta o no lo

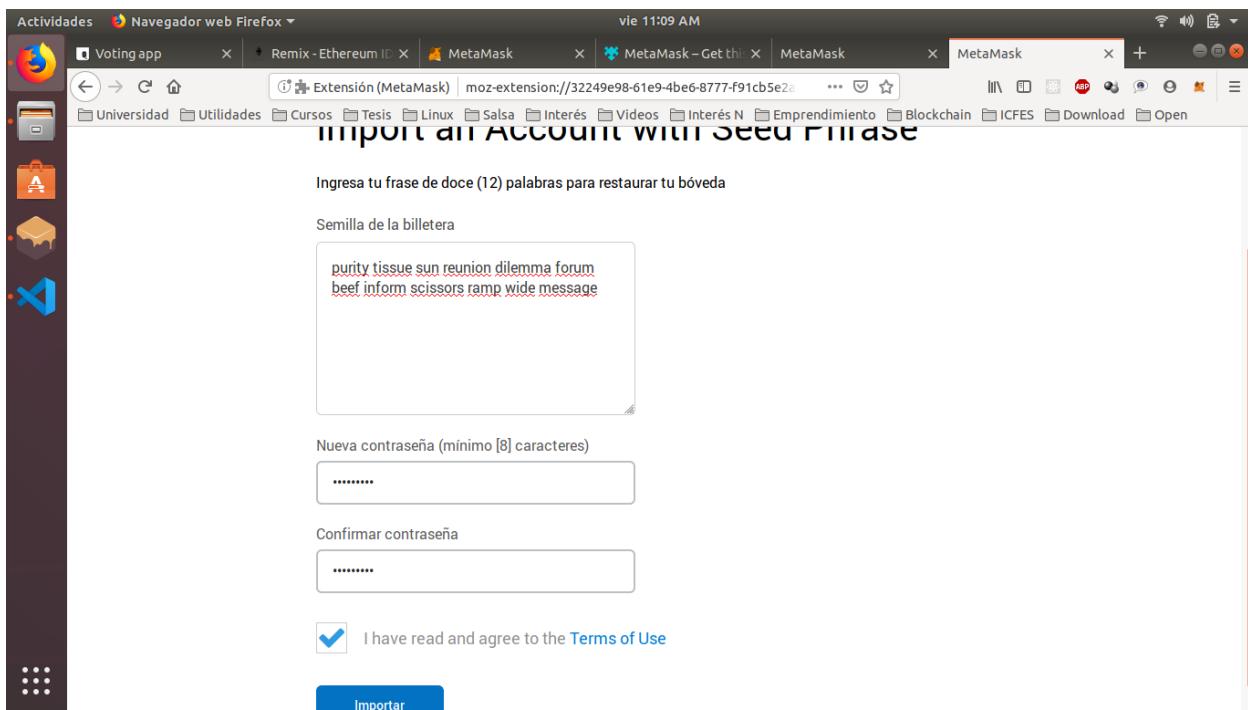


Figura 13: Añadiendo frase mnemonic de Ganache a Metamask. Fuente propia.

presentado en la Figura 12, Metamask cargará la siguiente página web que se puede ver en la Figura 13, donde se pide la frase y una contraseña segura para impedir el uso no autorizado

de las cuentas importadas. La frase mnemonic se la puede encontrar en la interfaz principal de Ganache tal y como se puede apreciar en la *Figura 14*.

The screenshot shows the Ganache application window. At the top, there are tabs for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, and LOGS. Below the tabs, there are several status indicators: CURRENT BLOCK (0), GAS PRICE (2000000000), GAS LIMIT (6721975), HARDFORK (PETERSBURG), NETWORK ID (5777), RPC SERVER (HTTP://127.0.0.1:8545), MINING STATUS (AUTOMINING), and WORKSPACE (TESIS). A search bar at the top right allows searching for block numbers or tx hashes. On the left, there is a sidebar with icons for Accounts, Blocks, Transactions, Contracts, Events, Logs, and a gear for settings. The main area displays a table of accounts. The first row of the table is highlighted in blue, containing the mnemonic phrase: "purity tissue sun reunion dilemma forum beef inform scissors ramp wide message". To the right of this phrase is its HD PATH: "m/44'/60'/0'/0/account_index". The table has columns for ADDRESS, BALANCE (100.00 ETH), TX COUNT (0), and INDEX (0, 1, 2, 3, 4, 5). Each account row also has a copy icon (a clipboard with a pencil).

ADDRESS	BALANCE	TX COUNT	INDEX
0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12	100.00 ETH	0	0
0xcbed3a4F79EC58Bcc4C5A23874C982F605a7a25C	100.00 ETH	0	1
0x90B699293821E3b6D24613F41C5431Ef8CD71603	100.00 ETH	0	2
0x5587574d74317624F9A22235599B77FAC8052E64	100.00 ETH	0	3
0xDe325aeaB0461cf0d0184995b960b76EEA3E72a7	100.00 ETH	0	4
0xf60DEac4e1bAaE9B5A02eC90F89682187Fc8E7b6	100.00 ETH	0	5
ADDRESS	BALANCE	TX COUNT	INDEX

Figura 14: Frase mnemonic de Ganache. Fuente propia.

Esta es una frase única (resaltada en azul en la *Figura 14*) que sirve como identificador de la wallet siendo generada por Ganache. Como precaución, no se debe compartir dicha frase, ya que, con ella, cualquiera podría acceder a las cuentas y posteriormente usarlas como si fuera el propietario. Una vez aclarado lo anterior, se procede a copiar dicha frase usando CTRL + C del teclado, para posteriormente pegarla en el campo que dice Semilla de la billetera en la *Figura 13*. Cuando se haya aceptado los términos y condiciones, se presiona sobre el botón Importar de la *Figura 13* que se puede visualizar en la imagen citada. Si la semilla es válida, Metamask automáticamente importará la wallet de Ganache a Metamask y mostrará la página web que se puede apreciar en la *Figura 15*.

La aparición de esta página web, indica que la wallet ha sido correctamente importada y que puede ser usada para importar cuentas de Ganache a través de la clave privada tal y como se mostrará más adelante.

De la figura anterior, se debe proceder a presionar el botón All Done para que permita el paso a la *Figura 16*, donde Metamask automáticamente se conectará a la red principal de Ethereum, aunque puede cambiarse la red como se hará más adelante.

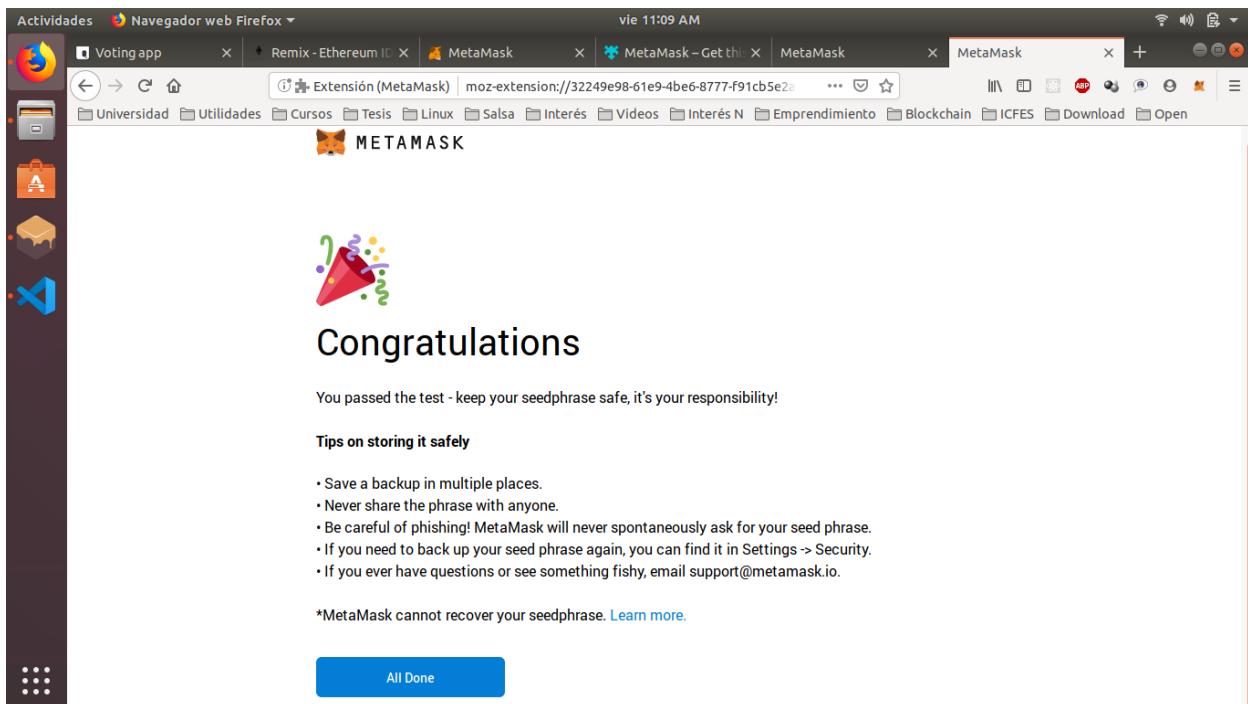


Figura 15: Importación correcta de la wallet a Metamask. Fuente propia.

En la Figura 16 se tiene un resumen de la cuenta que se tiene seleccionada en el momento, esto es la cantidad de ethers, tokens, cuenta usada, depositar/enviar ether o tokens, cambiar de red o de cuenta en el círculo que se encuentra en la parte superior derecha.

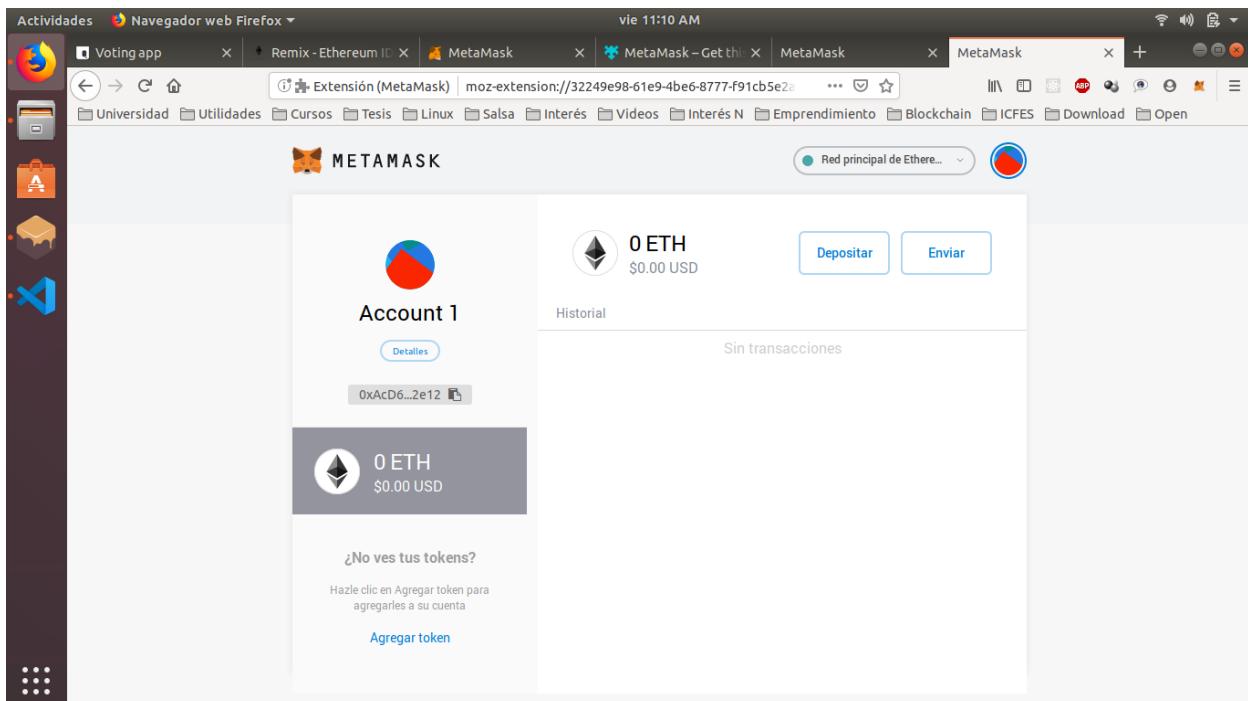


Figura 16: Metamask conectado a la red principal de Ethereum. Fuente propia.

Como siguiente paso, debemos cambiar a la red Blockchain privada que ya se configuró.

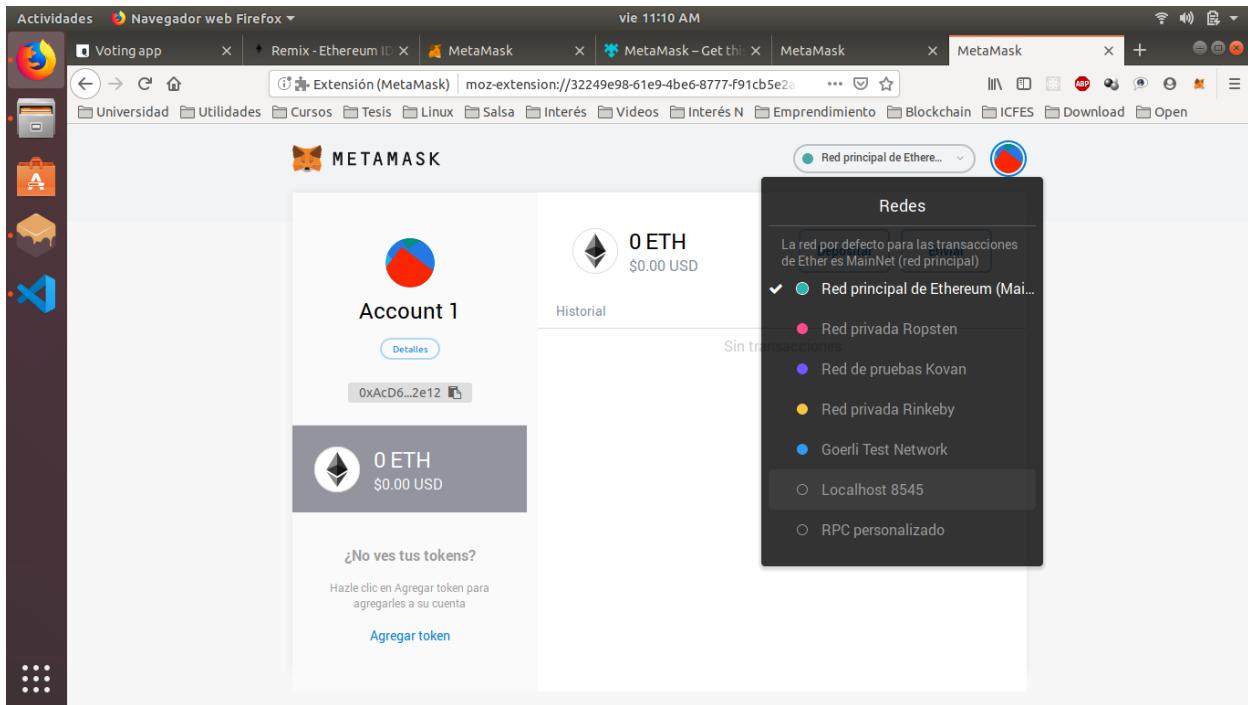


Figura 17: Conectando con la red de Ganache. Fuente propia.

Para cambiar la red, a la red Blockchain privada de Ganache, se debe presionar sobre la opción Red principal de Ethereum y seleccionar el ítem Localhost 8545 de la lista desplegable, esto indica que se tiene una red Blockchain local que se la puede ubicar en la IP 127.0.0.1 y que estará escuchando en el puerto 8545; lo cual fue configurado en la *Figura 3*. Cuando se selecciona esta opción, Metamask actualiza la página web con la nueva red, lo que muestra será similar a lo ilustrado en la *Figura 18*.

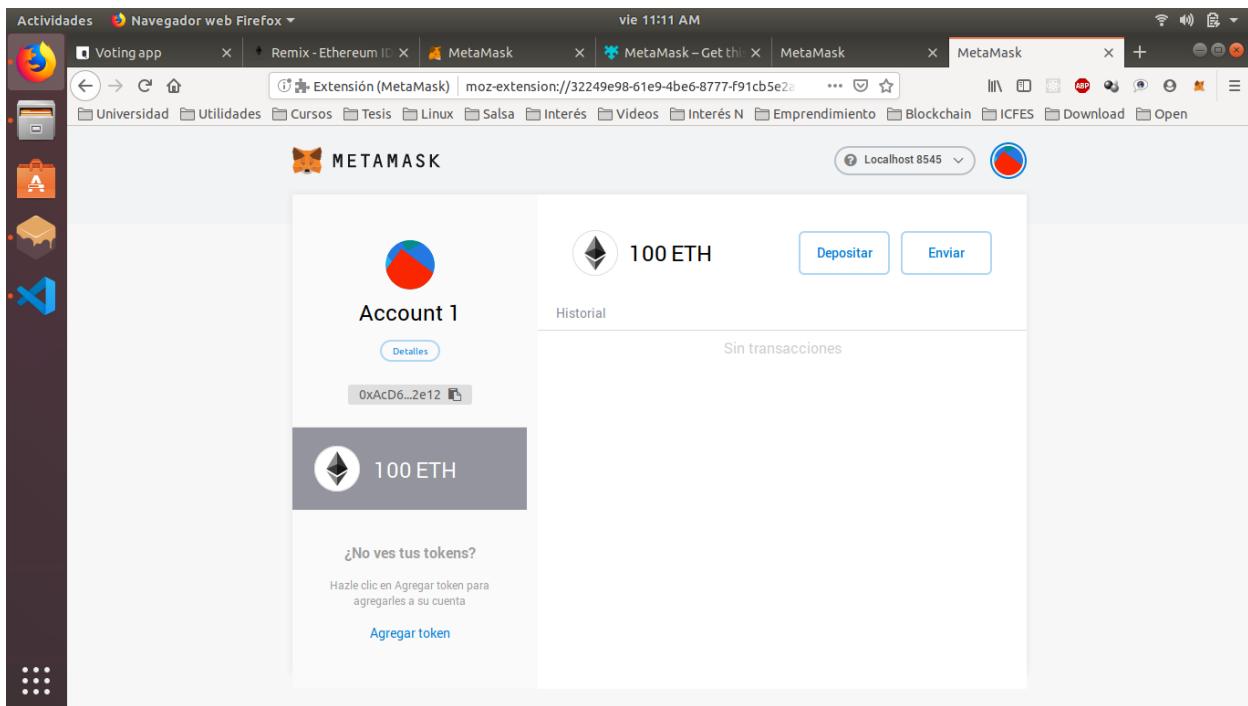


Figura 18: Conectado a la red Blockchain privada ofrecida por Ganache. Fuente propia.

Como posiblemente ya reparó en que la cuenta tiene 100 ethers a diferencia de la anterior, es porque Metamask siempre trabaja sobre la primera cuenta que tiene Ganache en su lista (ver Figura 5). En la Figura 19 se ofrece una mejor visión de lo que se acaba de mencionar.

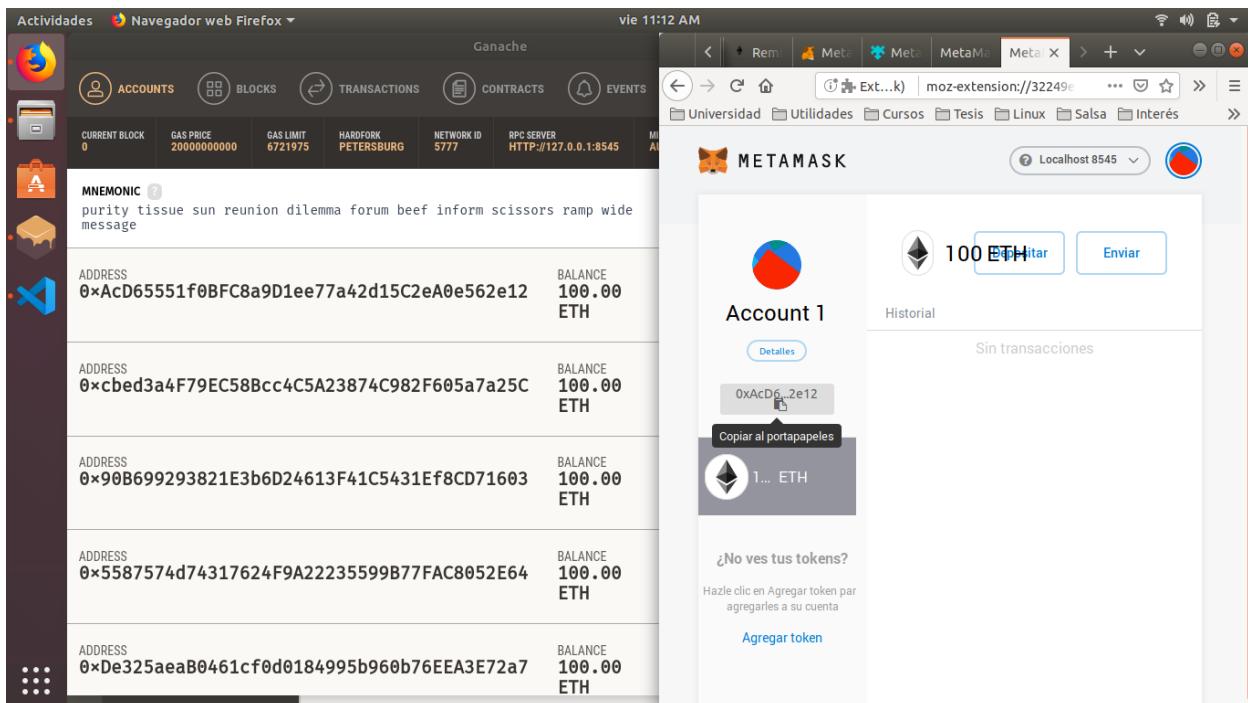


Figura 19: Comprobación de conexión entre Ganache y Metamask. Fuente propia.

Resumiendo, hasta este momento se tiene corriendo una Blockchain privada en Ganache y se ha configurado Firefox para que pueda hacer uso de tecnologías descentralizadas, donde se le ha añadido el plugin de Metamask para tal fin, y a su vez, este plugin ha sido conectado a la red Blockchain privada de Ganache al importar su wallet, si todo fue configurado correctamente, debería tener una página similar a la presentada en la *Figura 19*.

1.1.3. Software adicional.

Con fines de que tanto usuarios de Windows como GNU/Linux tengan la posibilidad de experimentar con la aplicación, se describe los pasos para instalar todo el software en cada uno de los sistemas operativos citados.

Ahora, es momento de instalar la herramienta necesaria para poder desplegar los contratos llamada Truffle, la cual ya ha sido nombrada en pasos anteriores. Truffle es una suite, que permite el desarrollo de contratos inteligentes, compilarlos, testearlos y desplegarlos a una red Blockchain. Es por estas razones que se hace necesaria para poder trabajar con la DApp (que se hablará más adelante) que se desarrolló. Como primer paso, se debe instalar unas herramientas previas para poder instalar este paquete (Truffle) a través de NPM (Node Package Manager) y usando NodeJs.

Como algunos paquetes que se instalarán necesitan clonar paquetes, también se debe tener instalado Git previamente. Como son herramientas documentadas en internet, no se explicará la instalación paso a paso, pero puede instalarlo usando este comando en GNU/Linux *sudo install git* o descargarlo desde [Git](#) para usuarios de Windows. Para instalar NodeJs en Windows puede dirigirse a la siguiente página de [NodeJs](#), es necesario que instale la versión 9.11.2 para no tener problemas de compatibilidad con las demás herramientas. Elija el tipo de sistema operativo y arquitectura (32 o 64 bits) que está usando y lo descarga e instala como otro programa del común. Si está usando GNU/Linux puede consultar [esta página](#) para instalar NodeJS a través de un PPA, teniendo siempre presente que instalará la versión 9.x. Cabe aclarar que antes debe instalar *curl* con el siguiente comando *sudo apt-get install curl*. Una vez instalado NodeJs, no es necesario instalar NPM, ya que este se instala junto con NodeJS. Una vez se ha instalado los anteriores programas, se procede a instalar Truffle desde la página de [Truffle](#).

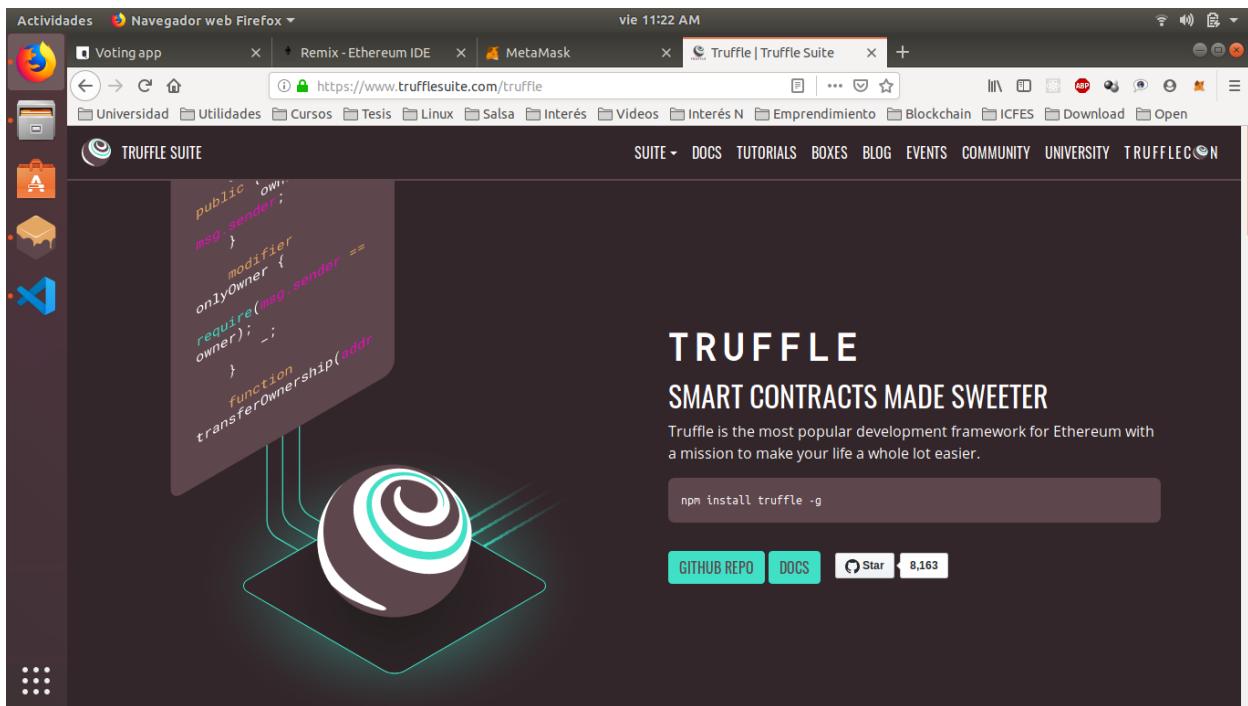


Figura 20: *Página principal de Truffle. Fuente propia.*

Truffle se instala a través de comandos usando NPM, para ello, se abre la ventana de CMD de Windows presionando las teclas Windows + R y escribiendo cmd en la ventana que se muestra y presionar la tecla Enter, o en GNU/Linux presionando las teclas CTRL + ALT +T, independientemente de si usa la consola de Windows o la terminal de GNU/Linux, escriba el comando que se puede apreciar en la Figura 20 y para evitar problemas de compatibilidad se agrega la versión usada en el presente, por lo tanto el comando quedaría de la siguiente forma primero debe darle permisos de super usuario usando *sudo su*, no es suficiente con *sudo*; luego instalar con el comando *sudo npm install -g truffle@5.0.2*. En GNU/Linux algunas veces suele generar errores el comando, por lo que se recomienda que lo instale como usuario root agregando la palabra *sudo* al inicio y en el final añadiendo el comando *--unsafe-perm=true* cuando indica que no tiene permisos para realizar la compilación. El comando para GNU/Linux quedaría de la siguiente forma con las modificaciones: *sudo npm install truffle@5.0.2 -g --unsafe-perm=true*. En Windows, solamente es necesario usar *npm install truffle@5.0.2 -g*.

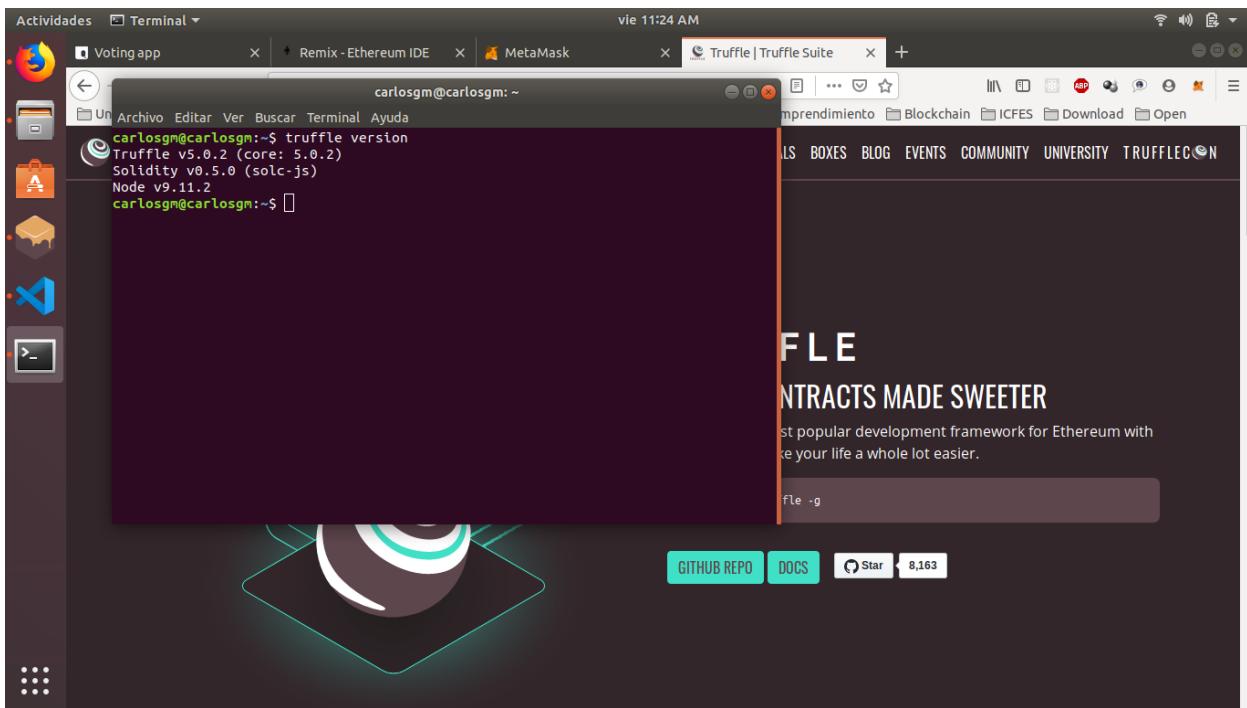


Figura 21: Versiones del software instalado. Fuente propia.

Una vez se ha instalado el software mencionado, se procede a ejecutar el comando *truffle version*, el cual luego de unos instantes, debería mostrar un mensaje parecido al que se presenta en la *Figura 21*. Si no obtiene resultados similares, revise nuevamente los pasos para tener el software adecuado.

1.1.4. Datos adicionales para desarrolladores.

Aquellas personas que deseen modificar el código propuesto, deben poseer conocimientos en estos temas particulares de cada una de las siguientes tecnologías:

- ReactJs: particularmente trabajar con componentes, estados, props, react-router, manipulación del DOM del navegador, manipulación del DOM virtual de ReactJs, ciclo de vida de un componente, importación y uso de paquetes instalados con NPM, sintaxis JSX, JavaScript, renderización de componentes, herencia, módulos, funciones anónimas, callbacks, sintaxis ECMAScript-6, ECMAScript-5, JSON, POO con JavaScript, entre otros.
- ReactJS – JavaScript
- NodeJS
- NPM
- Bootstrap: botones, listas, contenedores, objetos multimedia, márgenes, paddings, entre otros.
- Blockchain: tipos de Blockchain, bloque, nounce, transacción, minado, nodo, confirmación, encabezado de bloque, hash, identificador, dirección, consenso, prueba de trabajo, ataque del 51%, EDCSA, criptografía, clave pública, clave

privada, firmas digitales, ether, gas, gas límite, máquina virtual de Ethereum, árboles de Merkle, dificultad de minado, minero, transacción pendiente, transferencia de ether, tokens personalizados, ERC-20, ERC-721, eventos, contratos, registros, peso de un bloque, entre otros.

- Contratos usando Solidity: compilador, versiones del compilador, POO, Solidity, estados, calls, sends, transacciones, funciones, funciones anónimas, mappings, arrays, modificadores de visibilidad (public, external, internal, privada), modificadores, eventos, modificadores de accesibilidad de funciones (view, pure), address, variables globales (msg), payable, gas, gas límite, tokens, estados públicos, paquetes, importar contratos, llamados a contratos externos, retornar estados, transfer, emit, claves públicas, dirección propia, estándares, reglas de convención, documentación, constructor, memory, despliegue de contratos, storage, tipos de datos, ABI, entre otros.
- Ganache
- DApps
- Web3Js: proveedores, calls, sends, variables globales (ethereum), methods (eth, methods, Contract, givenProvider), entre otros.
- Truffle: compile, migrate, test, console, migraciones, entre otros.
- Metamask
- Visual Studio Code, Atom, Sublime text.
- IDE Remix

Las anteriores tecnologías son la base sobre la cual se ha implementado Voting App, aunque vienen otras involucradas que no hay necesidad de describirlas como HTML y css.

Para poder trabajar desarrollando en estas tecnologías, se debe preparar el entorno de la misma manera que se explicó anteriormente para personas no técnicas. Además, si está trabajando desde Windows, por lo que debe seguir los pasos de este comentario posteado por el usuario [@mousetraps](#) para evitar inconvenientes instalando paquetes de NodeJs. Los pasos son:

1. Instalar [Visual C++ Build Tools 2015](#)
2. Ejecutar en CMD el siguiente comando `npm config set python python2.7` para tener instalado Python 2.7.
3. Finalmente ejecutar el comando `npm config set msvs_version 2015 --global`

Luego, debe abrir la carpeta principal del proyecto llamada tesis con su editor de código de preferencia. Proceda a borrar la carpeta `/thesis/node_modules` si existe. Una vez ha realizado los ajustes necesarios, abra una terminal (puede ser la integrada en el editor) y ejecute el siguiente comando `npm run start` para iniciar el proyecto en modo desarrollo, en GNU/Linux suele generar errores de permisos por lo que se recomienda usar `sudo npm run start`. Esto abre

una pestaña en el navegador predeterminado y muestra la página principal de Voting App que es la que se puede apreciar en la *Figura 22*.

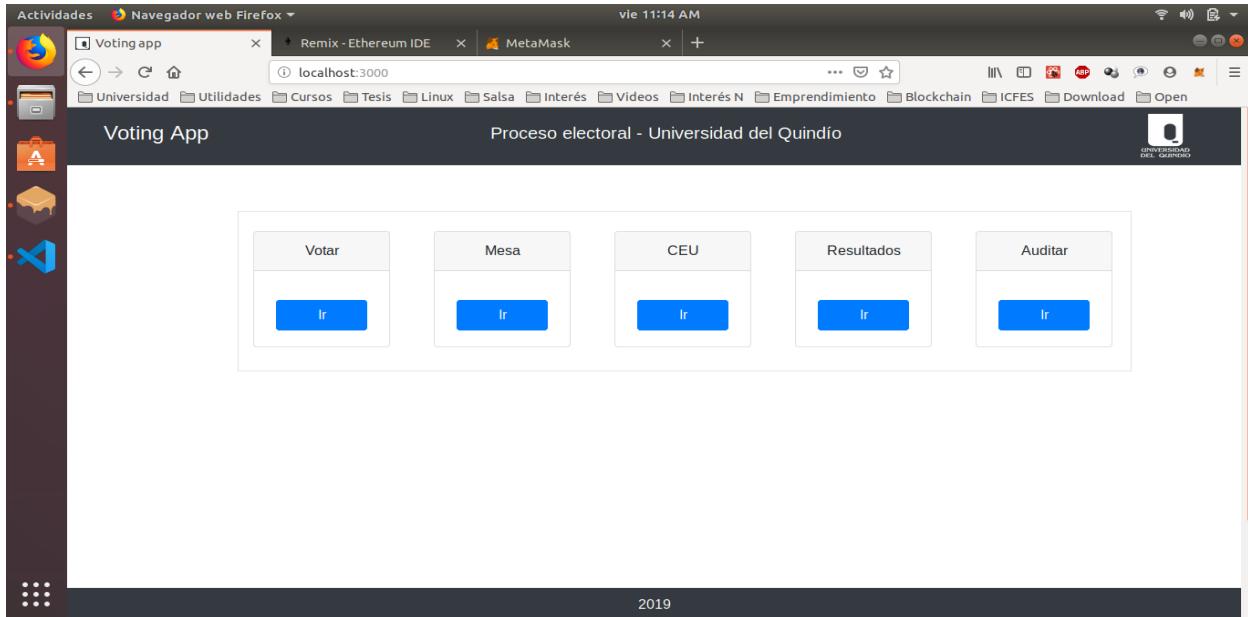


Figura 22: Página principal de Voting App. Fuente propia.

Si es la primera vez que la ejecuta o ha cerrado el navegador y lo ha abierto nuevamente, cuando Voting App necesite de información de la Blockchain, se dispara una ventana de Metamask preguntando si desea que este se integre a Voting App, petición a la cual se debe aceptar. Un ejemplo de esto se puede apreciar en la *Figura 23*.

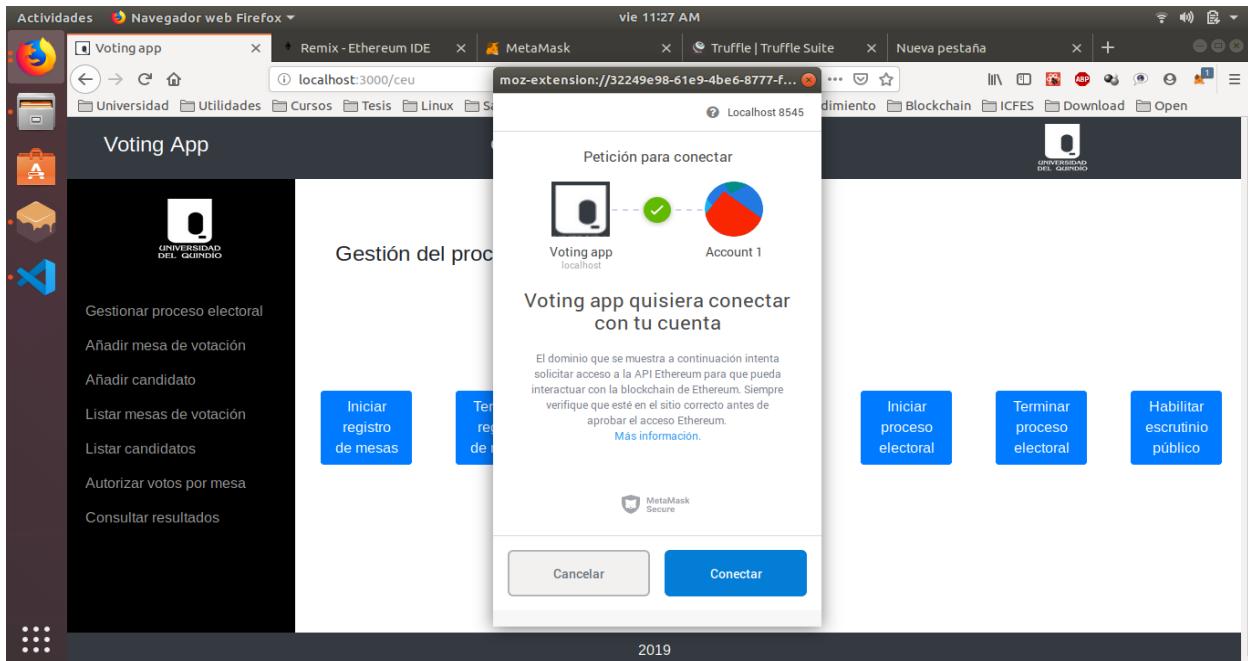


Figura 23: Mensaje de autorización Voting App se conecte con Metamask. Fuente propia.

Volviendo con la explicación para iniciar el proyecto en modo desarrollo, se abre una pestaña en el navegador, debido a que se ha configurado un servidor que escucha las modificaciones del código y refresca esos cambios al guardar en modo caliente, esto quiere decir, que no es necesario refrescar la página para ver cambios realizados en los archivos de ReactJs.

Si necesita construir una versión mínima y optimizada para ser desplegada en la raíz de un servidor web, puede usar el siguiente comando *npm run build* en una terminal que apunte a la raíz del proyecto. Esto creará una carpeta en la raíz del proyecto llamada *build* que es la que debe ser desplegada en el servidor. Si es el caso y no carga el contenido, revise la consola del navegador, en ocasiones no es capaz de cargar los script necesarios, para solucionar estos percances, edite el archivo *index.html* y busque el nombre del script, y antes de iniciar la ruta donde se encuentra, agregue un punto, ejemplo '*/ruta/archivo*', cambiar por '*./ruta/archivo*'.

Si realiza modificaciones a los contratos, debe compilarlos para que se actualicen los nuevos cambios en el código objeto, para ello se debe abrir una consola de Windows o terminal de GNU/Linux con permisos de superusuario y cambiar de directorio a la ruta */truffle*, luego se ejecutan en este último S.O. el comando *sudo truffle compile --all* (en Windows sin el *sudo*) para exigirle a Truffle que compile todos los contratos nuevamente. Este procedimiento se puede ver en la *Figura 24*.

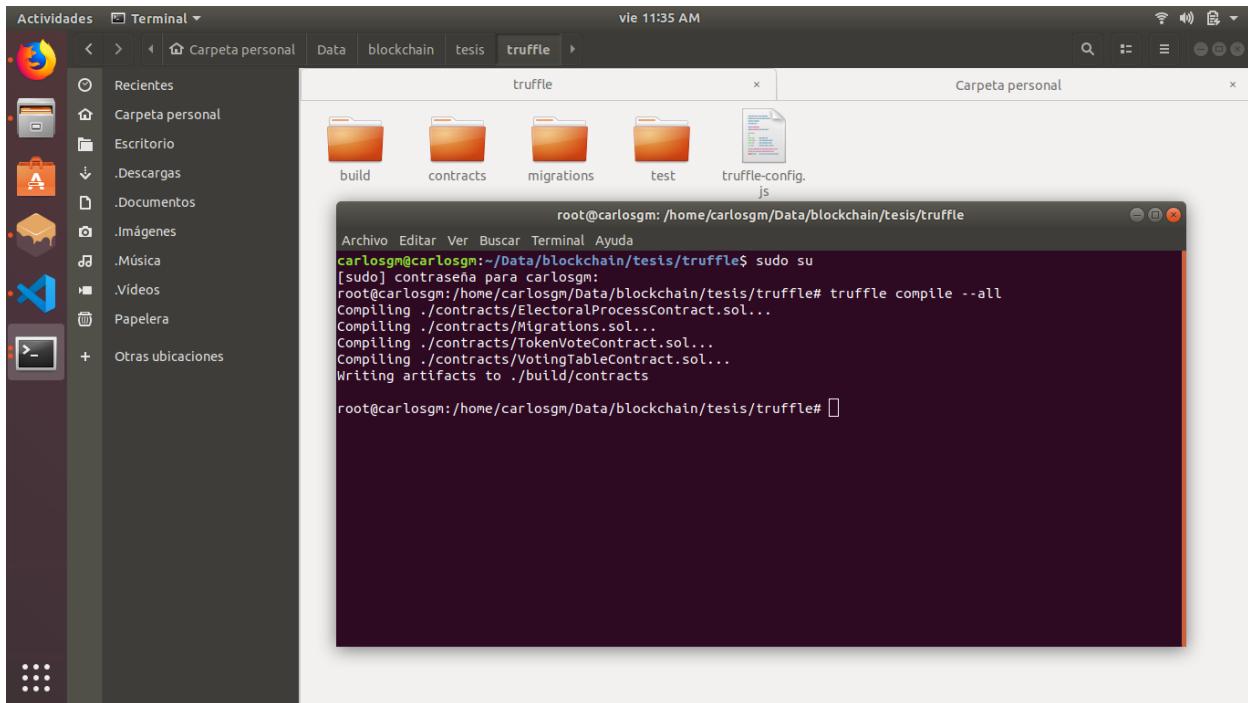


Figura 24: Compilando los Smart Contracts. Fuente propia.

1.2. Despliegue de contratos

Con el fin de desplegar los contratos, se debe realizar lo que se explica en esta sección, si no eres un desarrollador de software quien sigue este tutorial, puedes omitir lo relacionado a

compilar y actualizar propiedades ABI (Interfaz Binaria de la Aplicación). Con los contratos compilados, se procede a enviarlos a desplegarlos, para ello, sin cambiar de directorio, debe ejecutar el siguiente comando `sudo truffle migrate --reset`, la finalidad de este comando es desplegar los contratos en la Blockchain, y genera la salida en consola que se puede apreciar en las imágenes que empieza en la *Figura 25*.

```

Actividades Terminal vie 11:37 AM
root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle
Archivo Editar Ver Buscar Terminal Ayuda
root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle# truffle migrate --reset
⚠️ Important ⚠️
If you're using an HDWalletProvider, it must be Web3 1.0 enabled or your migration will hang.

Starting migrations...
=====
> Network name: 'development'
> Network id: 5777
> Block gas limit: 6721975

1_initial_migration.js
=====
Replacing 'Migrations'
-----
> transaction hash: 0x06ba3f042f25cb0d61c9be30ec66bf8091f9b31fb737bef21ac0b64ea11200af
> Blocks: 0 Seconds: 0
> contract address: 0x4de898579110a6983411253F2a591Cd29999859c
> account: 0xacd65551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.99430184
> gas used: 284908
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.00569816 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.00569816 ETH

2_initial_migration.js
=====
Replacing 'TokenVoteContract'
-----
```

Figura 25: Migrando contratos, parte 1/3. Fuente propia.

```

Actividades Terminal vie 11:38 AM
root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle
Archivo Editar Ver Buscar Terminal Ayuda
2_initial_migration.js
=====
Replacing 'TokenVoteContract'
-----
> transaction hash: 0x7917c1a5e37915d45461ea16fc0c88f376bb3006d883c4a9ceed48c667a2144a
> Blocks: 0 Seconds: 0
> contract address: 0xAFf497C47b794d69186407D7b1568d3EB84B97A1
> account: 0xacd65551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.96682566
> gas used: 1352345
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.0270469 ETH

Replacing 'ElectoralProcessContract'
-----
> transaction hash: 0xeed98146644728e395c9e16cbf29593408124417bd4548560e58fd72286b28e
> Blocks: 0 Seconds: 0
> contract address: 0xC4Bd7FE8E296405E86d0a56CE74502169F134f9
> account: 0xacd65551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.91150928
> gas used: 2765819
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.05531638 ETH

Replacing 'VotingTableContract'
-----
> transaction hash: 0xa49b7002357ee014e11ed4f5c4e56a0acab2fdada35949d8acb0c6fa29934cb
> Blocks: 0 Seconds: 0
> contract address: 0x9EB7Ee41FB9367cF30800D577DAF90B12518C08
> account: 0xacd65551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.87201824
> gas used: 1974552
> gas price: 20 gwei
```

Figura 26: Migrando contratos, parte 2/3. Fuente propia.

```

Actividades Terminal vie 11:38 AM
root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle

Archivo Editar Ver Buscar Terminal Ayuda
Replacing 'VotingTableContract'
-----
> transaction hash: 0xa49b7002357ee014e11ed4f5c4e56a0acab2fdada35949d8acb0c6fa29934cb
> Blocks: 0 Seconds: 0
> contract address: 0x9EB7Ee41F89367cF30800D5577DAF90B12518C08
> account: 0xAcD6551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.87201824
> gas used: 1974552
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.03949104 ETH

Replacing 'VotingTableContract'
-----
> transaction hash: 0xbd226b403ccf8f869ae01c1b567ffd719744fe4a67a4abaf4b0ac6401807c682
> Blocks: 0 Seconds: 0
> contract address: 0x314646C335A1a63511530474fbe50919617F3f76
> account: 0xAcD6551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.8325272
> gas used: 1974552
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.03949104 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.16134536 ETH

Summary
=====
> Total deployments: 5
> Final cost: 0.16704352 ETH
root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle#

```

Figura 27: Migrando contratos, parte 3/3. Fuente propia.

En este caso se desplegaron 5 contratos el correspondiente a las migraciones de Truffle, el contrato TokenVoteContract, ElectoralProcessContract y 2 contratos de VotingTableContract que se pueden apreciar estos dos últimos en la *Figura 27*.

Una vez se ha desplegado los contratos, se debe de actualizar los scripts que contienen los ABI de los contratos y las direcciones de los mismos. Estos scripts se encuentran, a partir de la raíz del proyecto, en la ruta */src/app/components/abi*. Las dos variables que estos tienen se actualizan desde dos fuentes distintas, una es con las direcciones de los contratos al momento de desplegar los contratos, que se hablará más adelante y la segunda es con los archivos json que genera la compilación de contratos usando Truffle, estos json se pueden ubicar en la dirección, a partir de la raíz del proyecto, en la ruta */truffle/build/contracts*. Estos JSON contienen una referencia al ABI del contrato en la tercera línea del código del objeto, de esta propiedad “abi”, se debe copiar todo su contenido hasta donde cierra el corchete, el cual terminará antes de acceder a la propiedad “bytecode” del mismo objeto.

```

src/app/components/abi/JS configTokenVoteContract.js
1 export const TODO_LIST_ADDRESS_TOKEN = '0x0B559c7D1Af87455B130dA3C135821a048307BeF';
2 export const TODO_LIST_ABI_TOKEN = [
3   {
4     "constant": true,
5     "inputs": [],
6     "name": "name",
7     "outputs": [
8       {
9         "name": "",
10        "type": "string"
11      },
12    ],
13    "payable": false,
14    "stateMutability": "view",
15    "type": "function",
16    "signature": "0x06fdde03"
17  },
18  {
19    "constant": true,
20    "inputs": [],
21    "name": "totalSupply",
22    "outputs": [
23

```

Compiled successfully!

You can now view **tesis** in the browser.

Local: http://localhost:3000/
On Your Network: http://192.168.0.102:3000/

Note that the development build is not optimized.

Figura 28: Ejemplo del script con el código ABI y su respectiva address. Fuente propia.

Este código copiado debe pegarlo en la variable constante del script (la cual contiene en el identificador la palabra ABI como se puede apreciar en la segunda línea de la *Figura 28*); localizado en la ruta /src/app/components/abi. Como se ha podido dar cuenta, existe un archivo JSON por cada script que se tiene en esta ruta, por lo tanto, debe realizar el mismo procedimiento por cada script. En la *Figura 28* se puede cómo debería quedar el script configTokenVoteContract.

```

root@carlosgm:/home/carlosgm/Data/blockchain/tesis/truffle
root@carlosgm: /home/carlosgm/Data/blockchain/tesis/truffle# truffle migrate --reset
⚠️ Important
If you're using an HDWalletProvider, it must be Web3 1.0 enabled or your migration will hang.

Starting migrations...
=====
> Network name: 'development'
> Network id: 5777
> Block gas limit: 6721975

1_initial_migration.js
=====

Replacing 'Migrations'
-----
> transaction hash: 0x06ba3f042f25cb6d61c9be30ec66bf8091f9b31fb737bef21ac0b64ea11200af
> Blocks: 0
> contract address: 0x4De898579110a6983411253F2a591Cd29999859c
> account: 0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12
> balance: 99.99430184
> gas used: 284908
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.00569816 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.00569816 ETH

2_initial_migration.js
=====

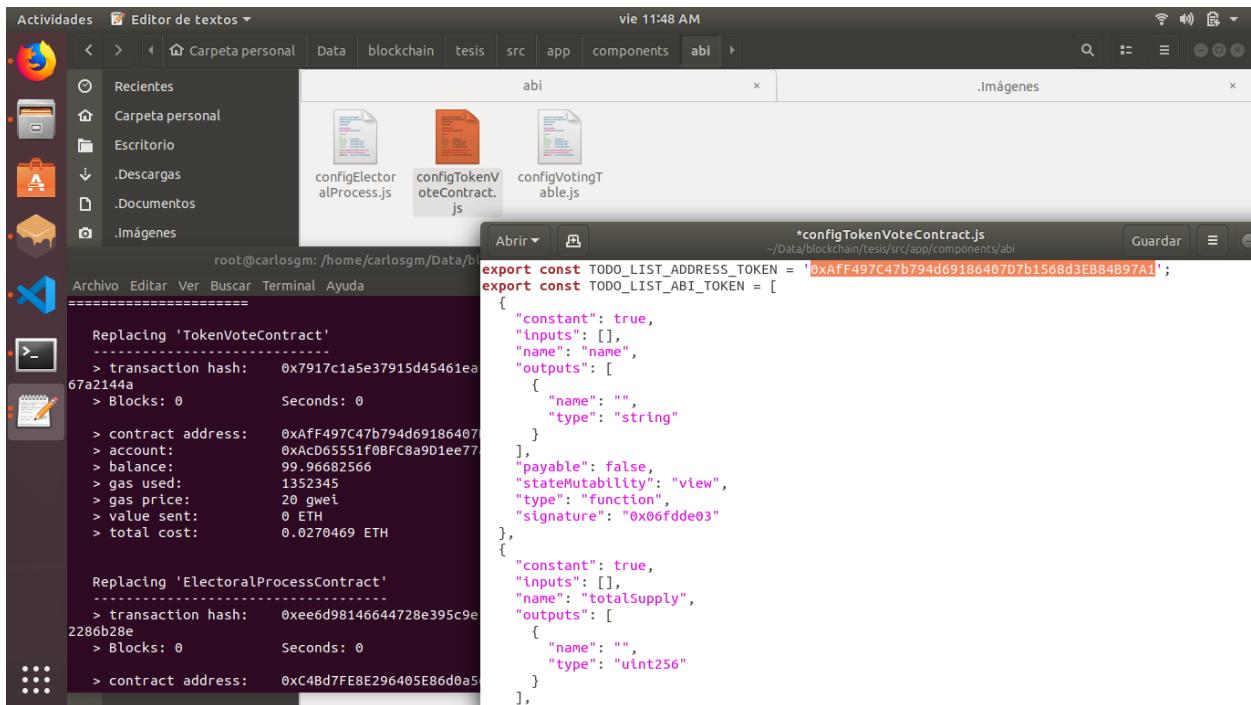
Replacing 'TokenVoteContract'

```

Figura 29: Dirección de un contrato desplegado. Fuente propia.

Para los contratos que se desplieguen se genera una dirección del contrato mediante la cual se puede llamarlo, un ejemplo de esta se puede apreciar en la *Figura 29*.

En la *Figura 28* se puede apreciar una variable constante la cual es una dirección del contrato que se tenía en ese momento, esa dirección se la ha tomado desde los datos que se plasmaron en la terminal cuando se desplegaron los contratos, para el caso del contrato de Truffle de nombre Migrations su dirección se encuentra resaltada en la *Figura 29*. Conforme se actualizó el ABI en los scripts, que es únicamente cada vez que se actualiza el código; se procede de la misma manera pero actualizando la constante de las direcciones que se puede apreciar en la *Figura 28*; a diferencia del anterior que se consultaba el JSON respectivo para tomar el ABI; para actualizar las direcciones de los scripts se consulta la terminal donde se desplegaron los contratos y se va actualizando estas en dichos scripts con las direcciones correspondientes de los contratos, tomando estas de donde se resaltó en la *Figura 29*. Este procedimiento se puede visualizar para algunos contratos en las siguientes imágenes:



The screenshot shows a Linux desktop environment with a terminal window and a code editor window. The terminal window is titled 'root@carlosgm:/home/carlosgm/Data/b...' and displays the output of a Truffle command, showing the deployment of two contracts: 'TokenVoteContract' and 'ElectoralProcessContract'. The code editor window is titled 'abi' and contains a file named 'configTokenVoteContract.js'. The ABI for the 'TOKEN' contract is being updated, with the constant 'TODO_LIST_ADDRESS_TOKEN' highlighted in red. The value of this constant is '0xAFF497C47b794d69186407b1568d3EB84B97A1'. The code editor interface includes tabs for 'Actividades', 'Editor de textos', 'Data', 'blockchain', 'tesis', 'src', 'app', 'components', and 'abi'. The status bar at the top indicates the date and time as 'vie 11:48 AM'.

```
root@carlosgm:/home/carlosgm/Data/b...
Archivo Editar Ver Buscar Terminal Ayuda
=====
Replacing 'TokenVoteContract'
-----
> transaction hash: 0x7917c1a5e37915d45461ea
67a2144a
> Blocks: 0 Seconds: 0
> contract address: 0xAFF497C47b794d69186407b1568d3EB84B97A1
> account: 0xAcD65551f0BFC8a901ee77
> balance: 99.96682566
> gas used: 1352345
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.0270469 ETH

Replacing 'ElectoralProcessContract'
-----
> transaction hash: 0xee6d98146644728e395c9e
2286b28e
> Blocks: 0 Seconds: 0
> contract address: 0xC4Bd7FE8E296405E86d0a5

export const TODO_LIST_ADDRESS_TOKEN = '0xAFF497C47b794d69186407b1568d3EB84B97A1';
export const TODO_LIST_ABI_TOKEN = [
  {
    "constant": true,
    "inputs": [],
    "name": "name",
    "outputs": [
      {
        "name": "",
        "type": "string"
      }
    ],
    "payable": false,
    "stateMutability": "view",
    "type": "function",
    "signature": "0x06fdde03"
  },
  {
    "constant": true,
    "inputs": [],
    "name": "totalSupply",
    "outputs": [
      {
        "name": "",
        "type": "uint256"
      }
    ],
    "payable": false,
    "stateMutability": "view",
    "type": "function",
    "signature": "0x0000000000000000000000000000000000000000000000000000000000000000"
  }
],
```

Figura 30: Actualizando dirección del script configTokenVoteContract. Fuente propia.

```

Actividades Editor de textos ▾ viernes 11:50 AM
Recientes Carpeta personal Data blockchain tesis src app components abi .Imágenes
root@carlosgm: /home/carlosgm/Data/blockchain/
Archivo Editar Ver Buscar Terminal Ayuda

Replacing 'VotingTableContract'
-----
> transaction hash: 0xa49b7002357ee014e11ed4f5c4e56a29934cb
> Blocks: 0 Seconds: 0
> contract address: 0x9EB7Ee41F89367cF30800D5577DAF90B12518C08
> account: 0xAcD6551f0BFC8a9D1ee77a42d15C20
> balance: 99.87201824
> gas used: 1974552
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.03949104 ETH

Replacing 'VotingTableContract'
-----
> transaction hash: 0xbd226b403ccf8f869ae01c1b567ffd1807c682
> Blocks: 0 Seconds: 0
> contract address: 0x314646C335A1a6351153D474cbe5092
> account: 0xAcD6551f0BFC8a9D1ee77a42d15C20

```

*configVotingTable.js

```

export const TODO_LIST_ADDRESS_TABLE = '0x9EB7Ee41F89367cF30800D5577DAF90B12518C08';
export const TODO_LIST_ADDRESS_TABLE2 = '0x314646C335A1a6351153D474cbe50919617F3f76';
export const TODO_LIST_ABI_TABLE = [
  {
    "inputs": [
      {
        "name": "_tableNum",
        "type": "uint256"
      },
      {
        "name": "adrElectoralProcessContract",
        "type": "address"
      },
      {
        "name": "_addressTokens",
        "type": "address"
      }
    ],
    "payable": true,
    "stateMutability": "payable",
    "type": "constructor",
    "signature": "constructor"
  },
  {
    "anonymous": false,
    "inputs": [
      {

```

JavaScript ▾ Anchura del tabulador: 8 ▾ Ln 1, Col 4

Figura 31: Actualizando dirección 1 del script configVotingTable. Fuente propia.

```

Actividades Editor de textos ▾ viernes 11:51 AM
Recientes Carpeta personal Data blockchain tesis src app components abi .Imágenes
root@carlosgm: /home/carlosgm/Data/blockchain/
Archivo Editar Ver Buscar Terminal Ayuda

Replacing 'VotingTableContract'
-----
> transaction hash: 0xbd226b403ccf8f869ae01c1b567ffd1807c682
> Blocks: 0 Seconds: 0
> contract address: 0x314646C335A1a6351153D474cbe5092
> account: 0xAcD6551f0BFC8a9D1ee77a42d15C20
> balance: 99.8325272
> gas used: 1974552
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.03949104 ETH

> Saving migration to chain.

> Saving artifacts
-----
> Total cost: 0.16134536 ETH

```

*configVotingTable.js

```

export const TODO_LIST_ADDRESS_TABLE = '0x9EB7Ee41F89367cF30800D5577DAF90B12518C08';
export const TODO_LIST_ADDRESS_TABLE2 = '0x314646C335A1a6351153D474cbe50919617F3f76';
export const TODO_LIST_ABI_TABLE = [
  {
    "inputs": [
      {
        "name": "_tableNum",
        "type": "uint256"
      },
      {
        "name": "adrElectoralProcessContract",
        "type": "address"
      },
      {
        "name": "_addressTokens",
        "type": "address"
      }
    ],
    "payable": true,
    "stateMutability": "payable",
    "type": "constructor",
    "signature": "constructor"
  },
  {
    "anonymous": false,
    "inputs": [
      {

```

JavaScript ▾ Anchura del tabulador: 8 ▾ Ln 2, Col 4

Figura 32: Actualizando dirección 2 del script configTokenVoteContract. Fuente propia.

Resumiendo, se han presentado las diferentes tecnologías que se usaron, iniciar el modo de desarrollo, producción, edición de contratos, compilación, despliegue, actualización de ABI y direcciones; una vez se realiza este procedimiento se pueden hacer uso de los cambios realizados

en el contrato, si no se han hecho cambios en el código del contrato no es necesarios compilarlos y actualizar el ABI, basta con desplegarlos nuevamente y actualizar las direcciones de los contratos en los scripts y ya estarán listos para ser usados.

La edición de los contratos la puede realizar en el IDE Remix que es web y persiste los cambios una vez se ha cerrado el navegador.

2. Usando Voting App

Ahora que se tiene configurado el respectivo entorno para poder trabajar con la página web (DApp) se procede a explicar cómo usarla, y de esto es lo que trata el siguiente apartado. Cabe aclarar que la autenticación de las personas no es trabajada en este proyecto, debido a su complejidad que da material para una tesis completa.

2.1. Registrando mesas de votación

Antes de poder registrar mesas de votación, se debe habilitar los contratos para este modo, con el fin de que permitan el registro de mesas únicamente, esto permite tener un mejor control para evitar autorizar más tokens a una mesa, por ejemplo, cuando estén votando. Esto ayuda a hacer el proceso un poco más ameno para los usuarios y transparente para los demás interesados en la jornada electoral.

La habilitación del Registro de mesas de votación, empieza por ingresar a la página principal de la aplicación y acceder por la opción del rol CEU, dicha interfaz es la que aparece en la *Figura 33*.

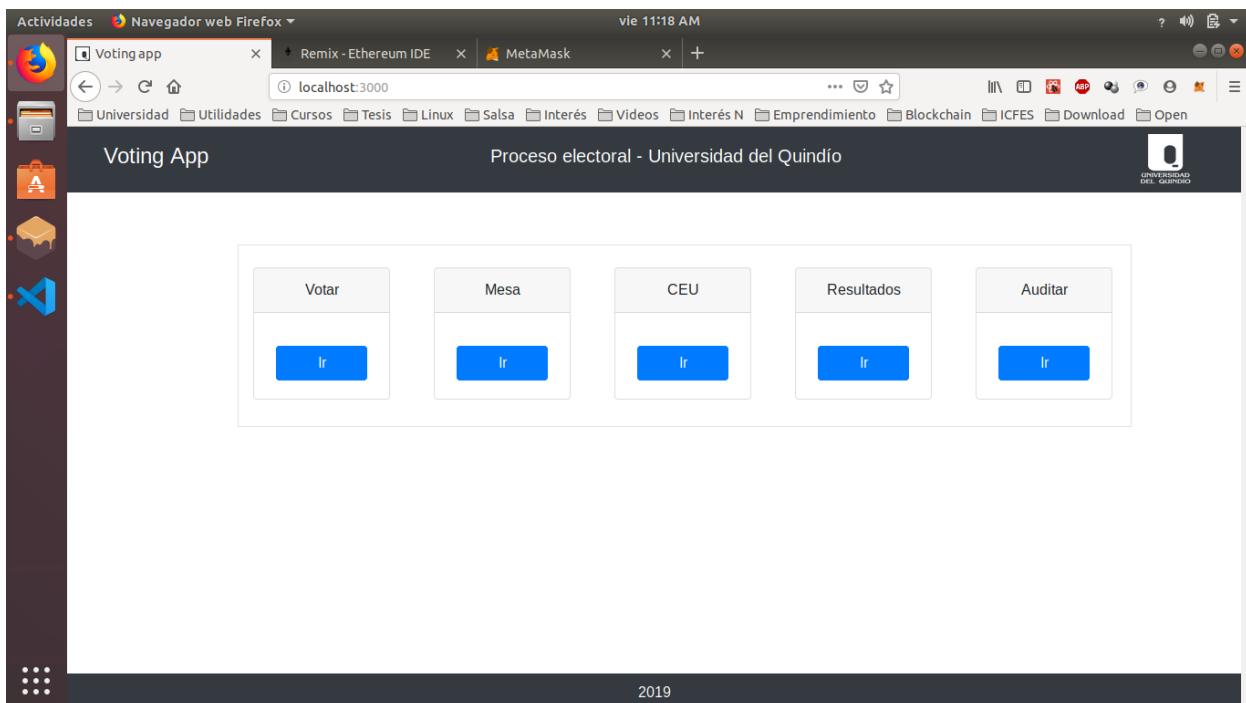


Figura 33: Página principal de Voting App (roles). Fuente propia.

Una vez que se ingresa, se presenta la interfaz que se puede visualizar en la *Figura 35*.

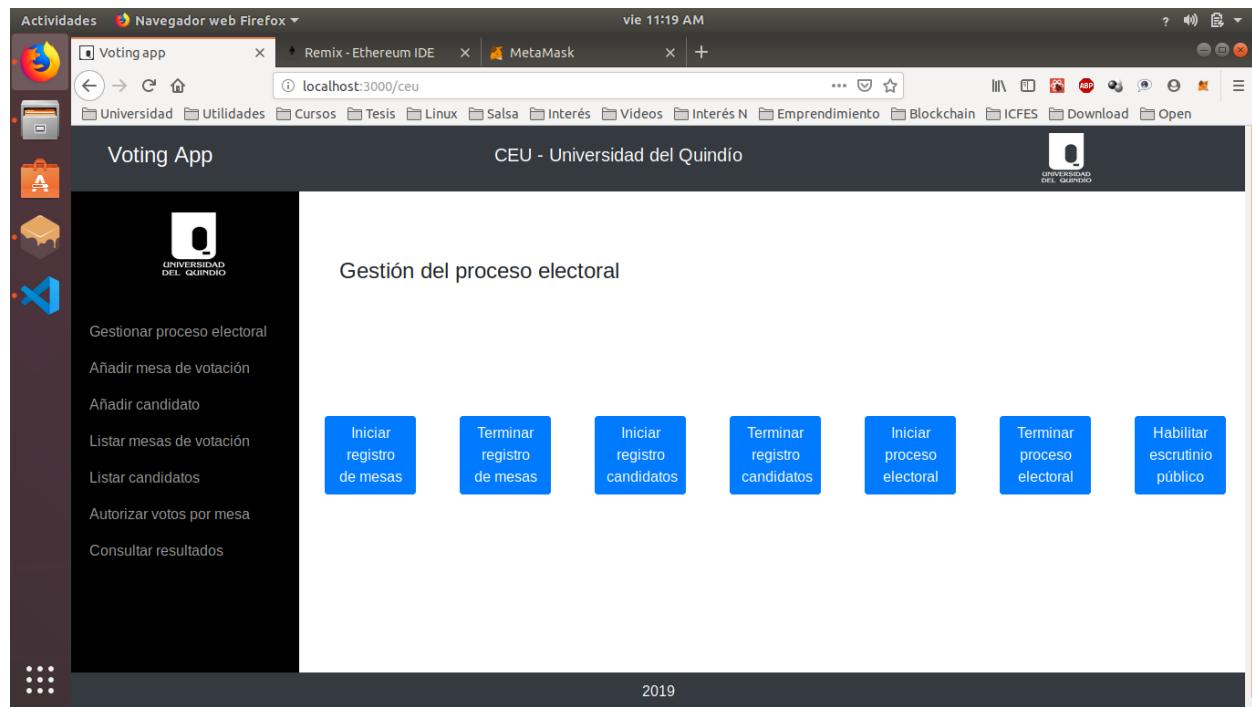


Figura 34: Página principal relacionada al rol CEU. Fuente propia.

Las diferentes opciones que se puede apreciar en la *Figura 34*, únicamente son usadas por el representante del Consejo Electoral Universitario (representante del CEU). Este, por medio de las opciones presentadas en la *Figura 34*, se limitará a habilitar etapas de votación, registrar mesas, candidatos, autorizar votos a las mesas y deshabilitar dichas etapas, principalmente. Aunque también puede listar mesas, candidatos y los resultados de la jornada electoral.

Para poder habilitar el registro de mesas de votación, se usa la opción *Iniciar registro de mesas*, que se puede apreciar al inicio del área de trabajo en la *Figura 35*, esto automáticamente llama a Metamask que es quien gestiona los saldos de las cuentas para indicarnos un valor aproximado de cuánto cuesta dicha transacción en ether (ETH), ya es cuestión del representante del CEU autorizar a Metamask a realizar el gasto o no. Si acepta, se presentará un aviso flotante en la parte derecha de la pantalla indicando el resultado de dicha transacción (autorización).

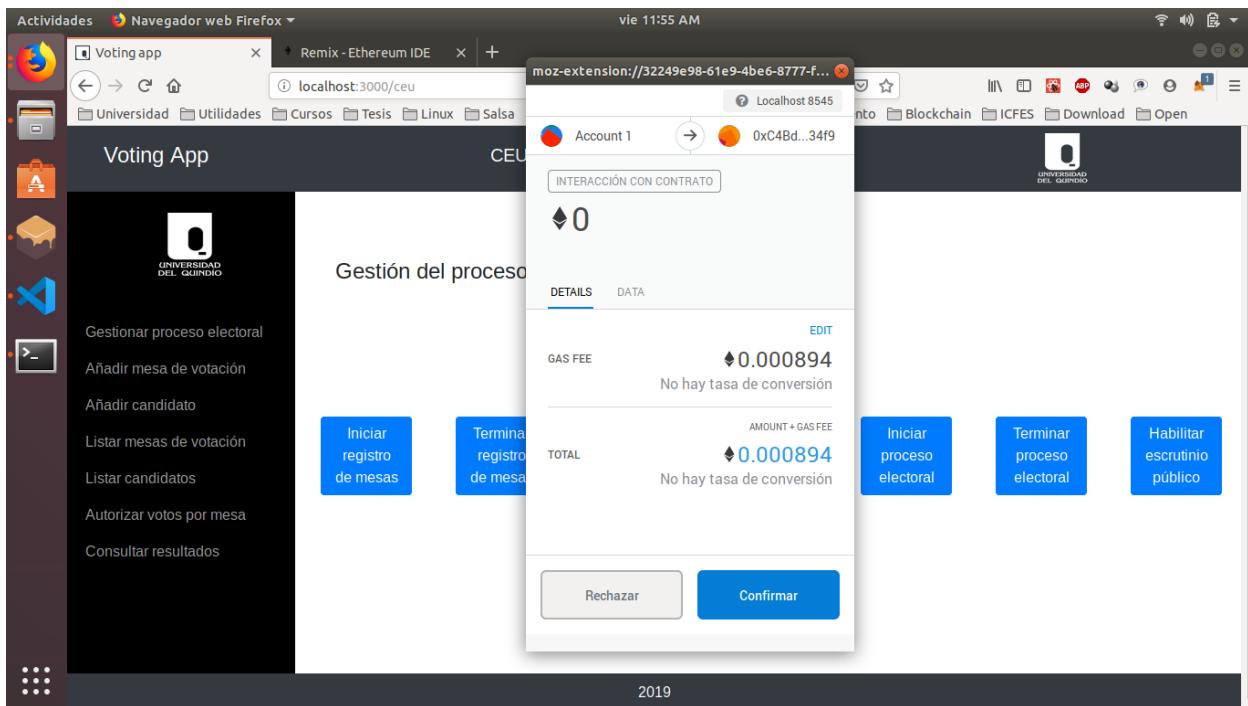


Figura 35: Autorizando registro de mesas de votación (CEU). Fuente propia.

Esta confirmación se puede apreciar en la Figura 36.

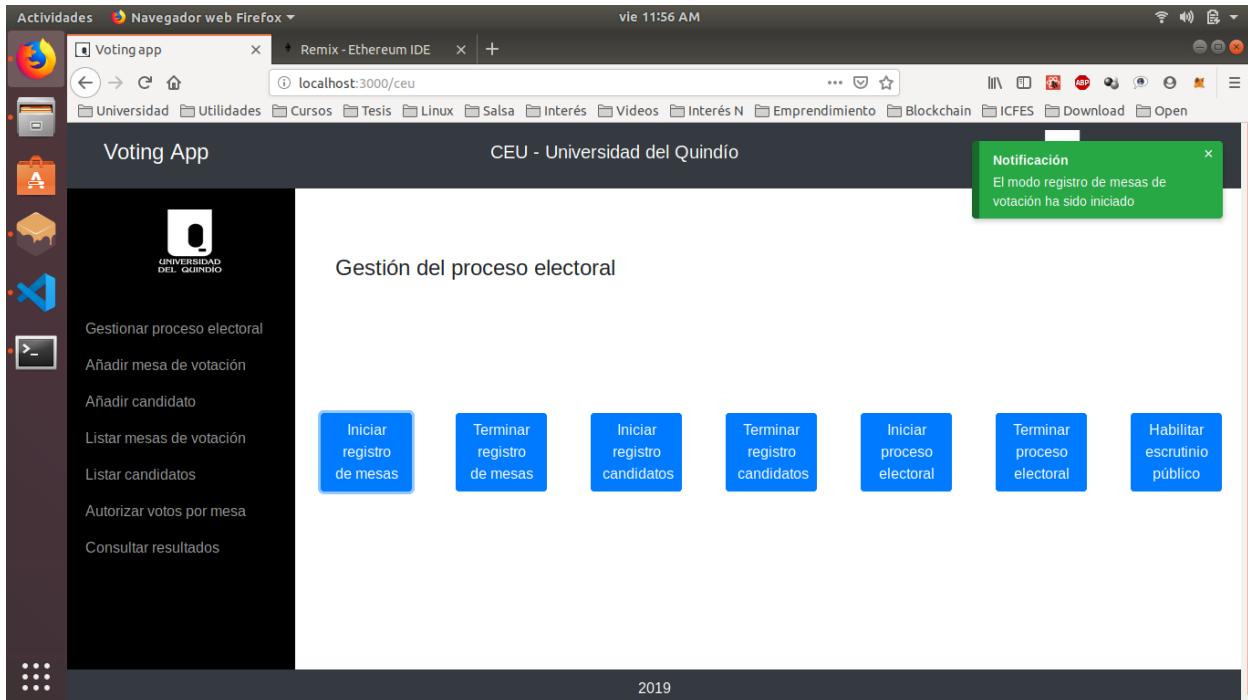


Figura 36: Notificación sobre el Inicio de registro de mesas de votación. Fuente propia.

Continuando con el tema de esta sección, se procede a describir cómo se puede registrar las diferentes mesas de votación, por lo que se usa la opción *Añadir mesa de votación* que se

presenta en la lista de ítems en la parte izquierda de la *Figura 36*. Al presionar sobre esta opción, se carga automáticamente el formulario correspondiente para que el representante pueda registrar tantas mesas como sean necesarias, con la finalidad de tener un control de las mesas que se habilitan y posteriormente poder auditar los votos obtenidos por cada candidato en cada mesa, tema que será objeto de discusión más adelante. El formulario para poder registrar las diferentes mesas de votación se puede apreciar en la *Figura 37*.

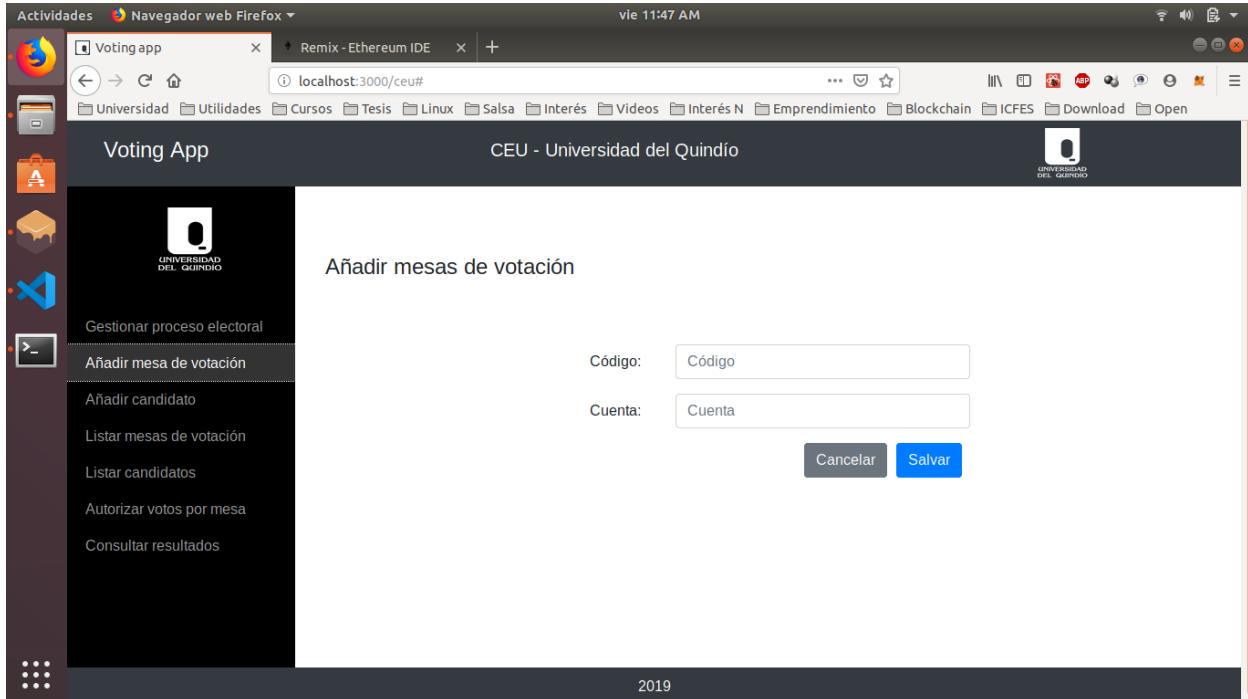


Figura 37: Formulario para el registro de mesas de votación. Fuente propia.

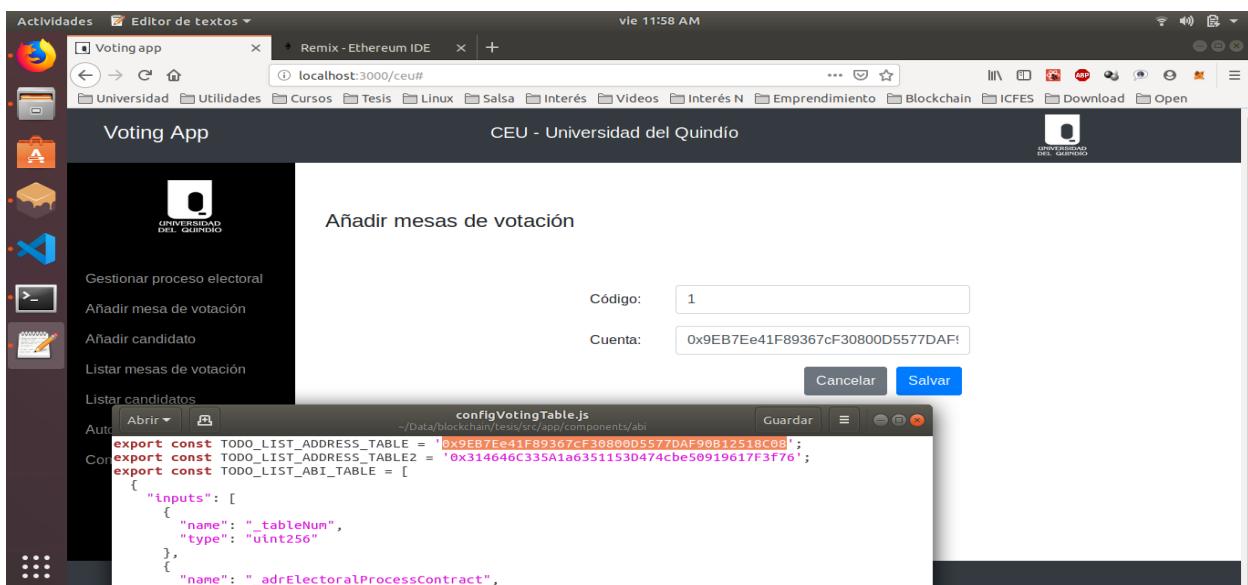


Figura 38: Registrando la mesa de votación con el código 1. Fuente propia.

Este únicamente contiene el código y la cuenta de la mesa, un ejemplo es el de la *Figura 38*.

Para el registro de las cuentas de las mesas se debe realizar la siguiente aclaración, estas a diferencia de las cuentas de los candidatos, no son aleatorias, estas direcciones como se mencionó anteriormente (ver Figura 29: Dirección de un contrato desplegado. Fuente propia. *Figura 29*), corresponden las direcciones de los contratos desplegados y que se encuentran en la Blockchain, de lo contrario, la mesa no podrá emitir votos, ya que no se le podrá autorizar los votos.

Una vez que se registre la mesa de votación, Voting App mostrará una notificación del estado del registro como se muestra en la *Figura 39*.

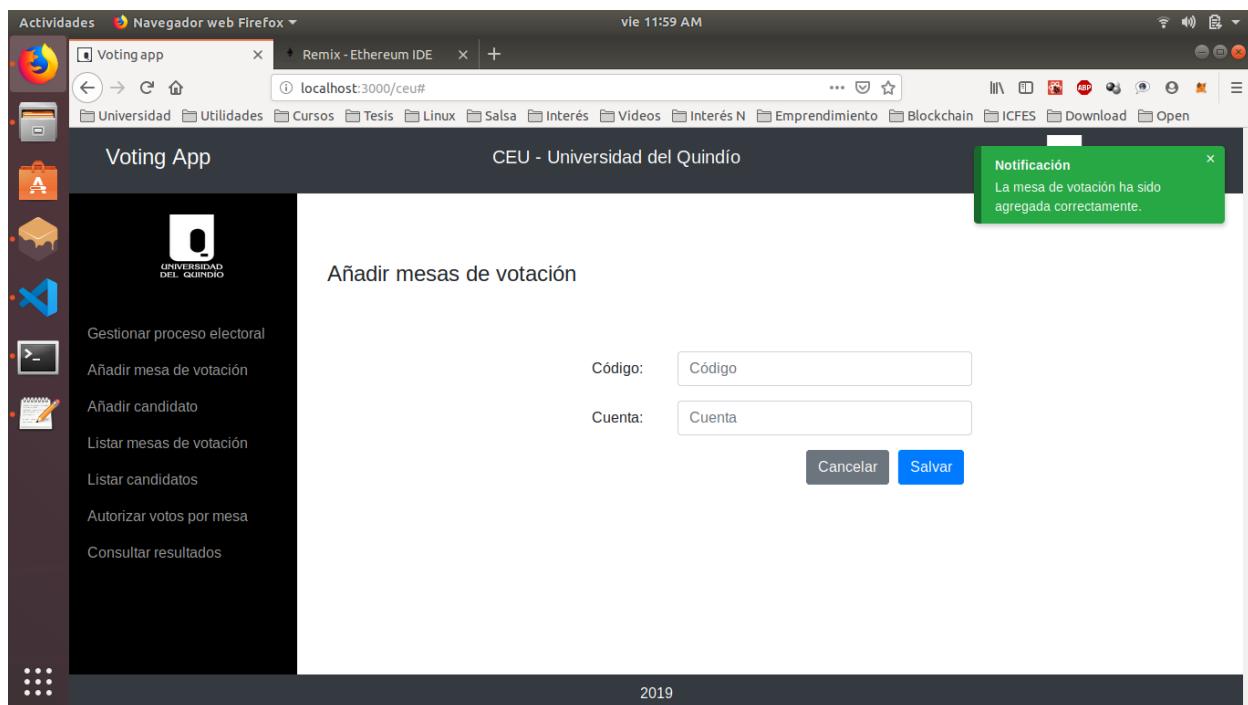


Figura 39: Confirmación registro de mesa de votación. Fuente propia.

En caso contrario, puede mostrar advertencias en color naranja o errores en color rojo. Para las advertencias puede ser cuando no está habilitados algún modo en especial, para el color rojo por lo menos que no tenga fondos suficientes, no esté autorizado, en ese caso revisar la cuenta activa en Metamask, entre otros errores.

Cuando esté seguro de haber registrado todas las mesas necesarias, podrá cerrar el modo registro de mesas de votación para evitar conflictos en el sistema si está habilitado un modo anterior al que se desee habilitar, para ello, en la parte izquierda de la *Figura 37* seleccione *Gestionar proceso electoral*, esto inmediatamente lo llevará la página que se puede apreciar en la *Figura 34*, donde debe seleccionar la opción *Terminar registro de mesas*, cuando realice esta

acción, Metamask pedirá confirmación para gastar ether en la transacción tal y como se puede apreciar en la *Figura 40*.

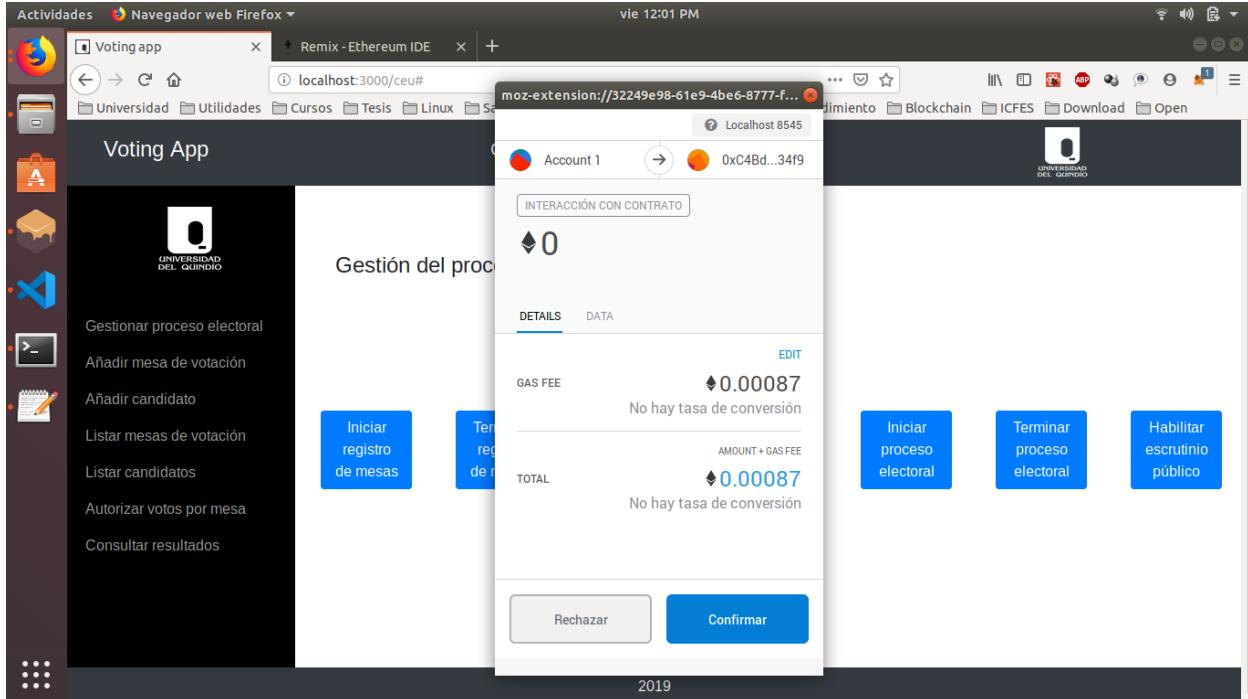


Figura 40: Terminando registro de mesas de votación. Fuente propia.

Al confirmar, debe esperar la notificación de que se ha realizado con éxito (ver Figura 41).

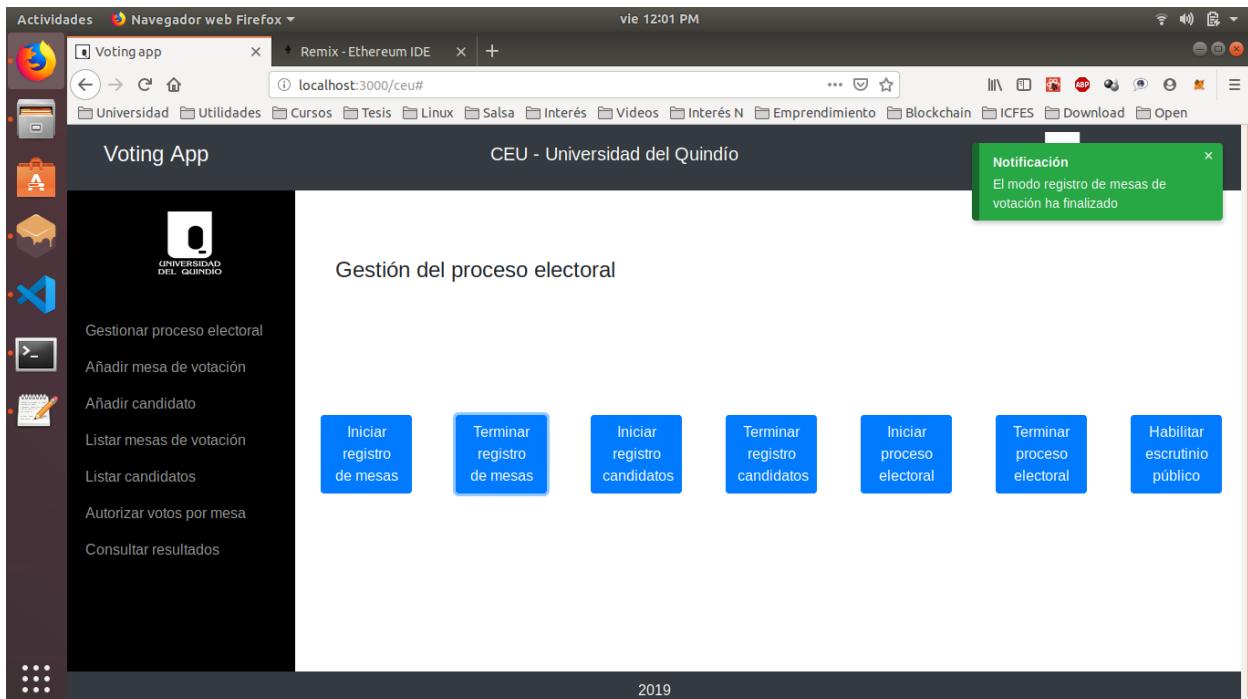


Figura 41: Confirmación terminación del modo registro de mesas. Fuente propia.

Ahora que se tienen todas las mesas registradas, las puede listar usando la opción *Listar mesas de votación* que se puede apreciar en la parte izquierda de la *Figura 41*. Esto desplegará una página como la presentada en la *Figura 42*.

The screenshot shows a Firefox browser window titled 'Voting app' with the URL 'localhost:3000/ceu#'. The main content area is titled 'Voting App' and 'CEU - Universidad del Quindío'. It displays a table titled 'Lista de mesas de votación inscritas' with two rows:

Mesa cód. 1	Mesa cód. 2
Número de votantes 0	Número de votantes 0
Cuenta: 0x9EB7Ee41F89367c F30800D5577DAF90 B12518C08	Cuenta: 0x314646C335A1a63 51153D474cbe50919 617F3f76

A green notification box in the top right corner says: 'Notificación' 'Se ha terminado de listar todas las mesas de votación.' On the left sidebar, under 'Gestionar proceso electoral', the option 'Listar mesas de votación' is selected. The footer of the page says '2019'.

Figura 42: Listando mesas de votación. Fuente propia.

Cabe aclarar, que no es necesario terminar el modo *registro de mesas de votación*, puede listarlas sin ningún problema con el modo activado.

2.2. Registrando candidatos

Para poder registrar candidatos, al igual que las mesas, también se debe de habilitar Voting App en este modo. Para esto, se debe ingresar a la página principal de Voting App (ver *Figura 33*), entrar en el rol de CEU y volverá a estar en la página que se muestra en la *Figura 34*. En esta página debe usar la opción *Iniciar registro de candidatos*, cuando se presione esta opción, Metamask pide confirmar la transacción tal y como se puede apreciar en la *Figura 43*.

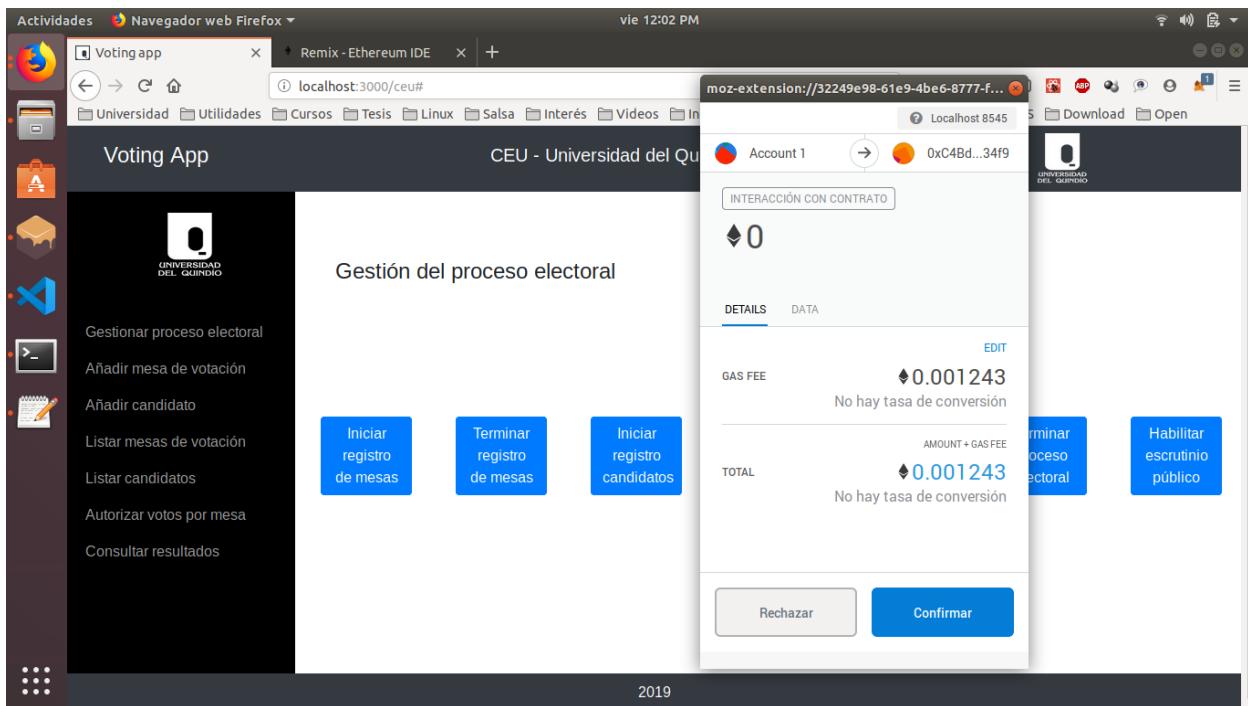


Figura 43: Iniciando registro de candidatos. Fuente propia.

Esto configura los contratos para que puedan aceptar el registro de candidatos (ver).

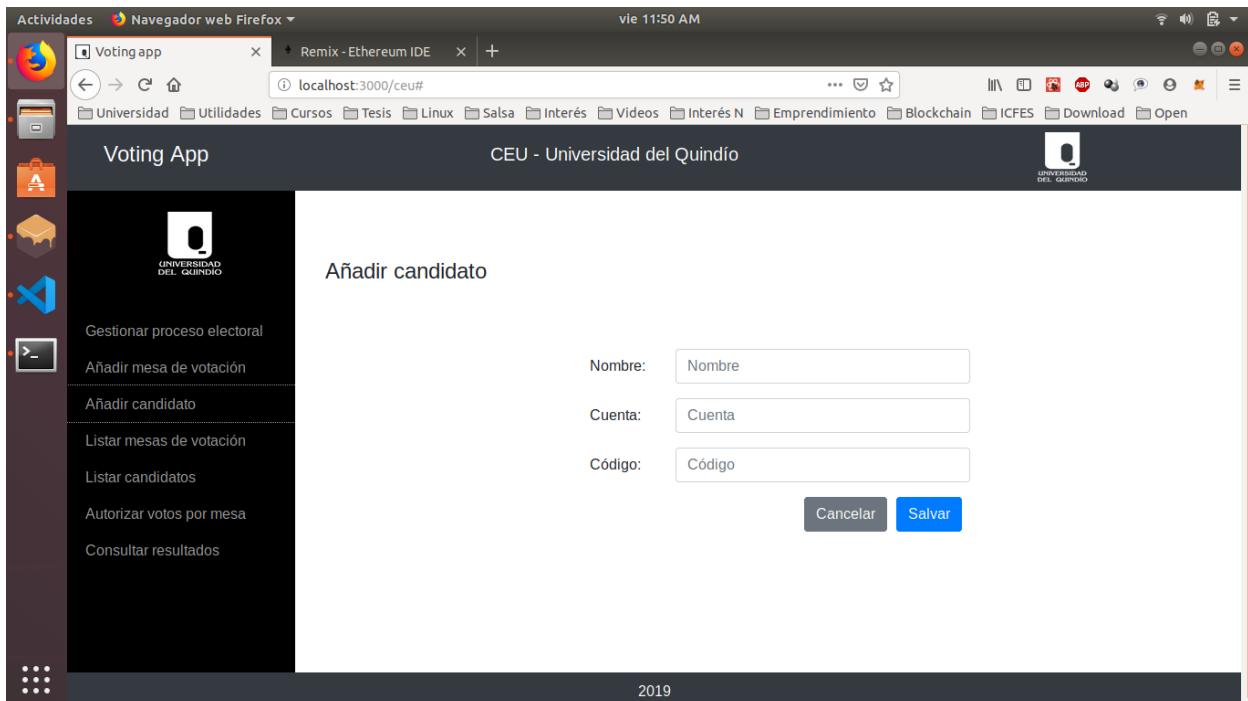


Figura 44: Interfaz para el registro de candidatos. Fuente propia.

Para registrar un candidato se debe tener a la mano el nombre, la cuenta y el código del tarjetón del mismo. A diferencia del registro de mesas de votación, la cuenta no debe ser una en

particular, puede ser cualquiera siempre y cuando se tenga la clave privada y pública de la misma. Para este ejemplo en particular, el autor se vale de las cuentas creadas en Ganache (ver *Figura 4*). Por lo tanto, se procede a la interfaz principal de Ganache como se muestra en la

The screenshot shows the Ganache interface with the following account details:

ADDRESS	BALANCE	TX COUNT	INDEX
0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12	99.83 ETH	12	0
0xcbed3a4F79EC58Bcc4C5A23874C982F605a7a25C	100.00 ETH	0	1
0x90B699293821E3b6D24613F41C5431Ef8CD71603	100.00 ETH	0	2
0x5587574d74317624F9A22235599B77FAC8052E64	100.00 ETH	0	3
0xDe325aeaB0461cf0d0184995b960b76EEA3E72a7	100.00 ETH	0	4
0xf60DEac4e1bAaE9B5A02eC90F89682187Fc8E7b6	100.00 ETH	0	5

Figura 45: Obteniendo una dirección para usarla como cuenta del candidato. Fuente propia.

Si desea conocer cuál es la clave privada de la cuenta seleccionada (clave pública) en la

The screenshot shows the Ganache interface with the following account details, where the address of the selected account is highlighted in blue:

ADDRESS	BALANCE	TX COUNT	INDEX
0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12	99.83 ETH	12	0
0xcbed3a4F79EC58Bcc4C5A23874C982F605a7a25C	100.00 ETH	0	1
0x90B699293821E3b6D24613F41C5431Ef8CD71603	100.00 ETH	0	2
0x5587574d74317624F9A22235599B77FAC8052E64	100.00 ETH	0	3
0xDe325aeaB0461cf0d0184995b960b76EEA3E72a7	100.00 ETH	0	4
0xf60DEac4e1bAaE9B5A02eC90F89682187Fc8E7b6	100.00 ETH	0	5

Figura 45, siendo necesaria para entregarla al candidato y sea él quien la use, puede presionar

sobre el ícono que tiene forma de llave en la parte derecha de la *Figura 45* del ítem correspondiente a la selección, esto desplegará una nueva ventana con otra cuenta la cual corresponde a la clave privada, se debe seleccionar y copiar únicamente con CTRL + C. Con estos datos, se procede a realizar el registro del candidato como en la

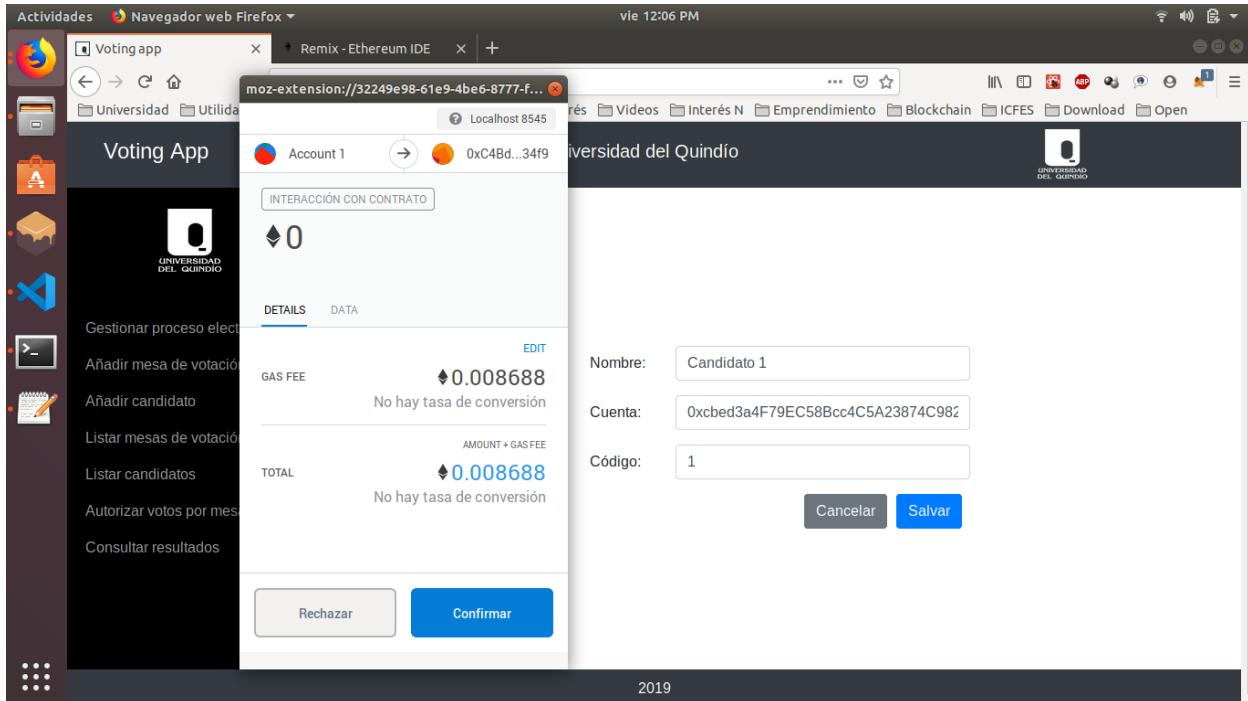


Figura 46: Registrando candidato con la cuenta seleccionada. Fuente propia.

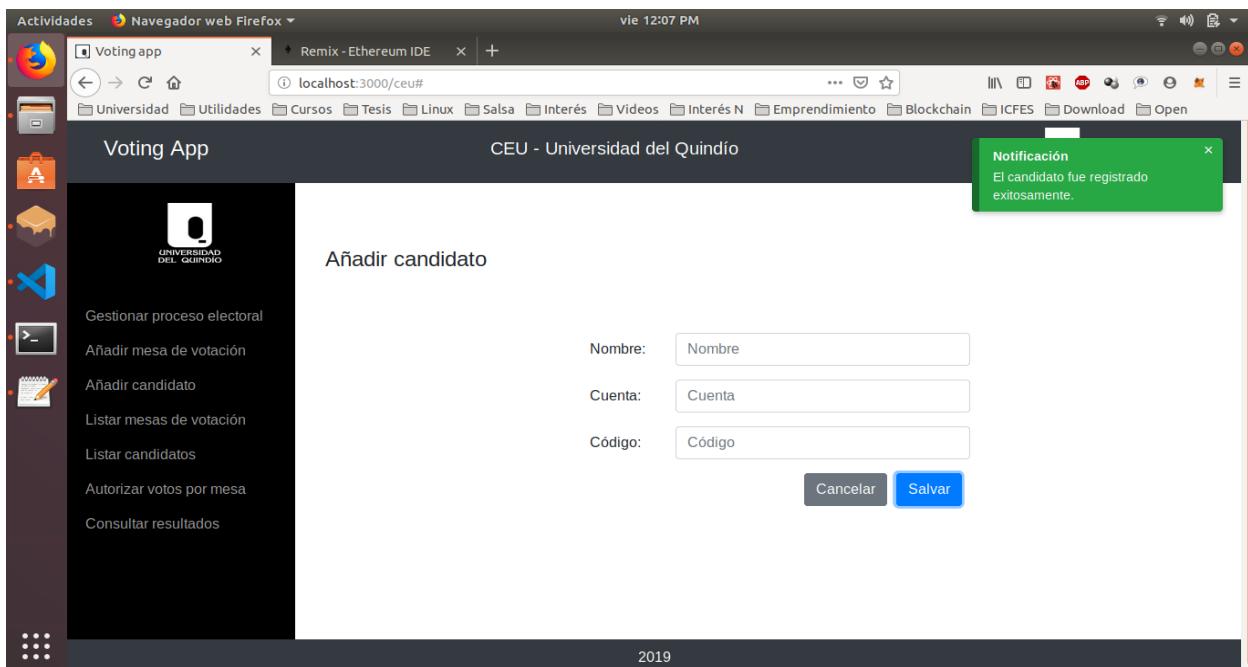


Figura 47: Confirmación del registro del candidato. Fuente propia.

Al presionar salvar, Metamask es lanzado para autorizar el gasto (ver *Figura 47*).

Para registrar el siguiente candidato, debe realizar los pasos desde la *Figura 45*

The screenshot shows the Ganache interface with the following details:

ADDRESS	BALANCE	TX COUNT	INDEX	KEY
0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12	99.83 ETH	12	0	🔑
0xcbed3a4F79EC58Bcc4C5A23874C982F605a7a25c	100.00 ETH	0	1	🔑
0x90B699293821E3b6D24613F41C5431Ef8CD71603	100.00 ETH	0	2	🔑
0x5587574d74317624F9A22235599B77FAC8052E64	100.00 ETH	0	3	🔑
0xDe325aeaB0461cf0d0184995b960b76EEA3E72a7	100.00 ETH	0	4	🔑
0xf60DEac4e1bAaE9B5A02eC90F89682187Fc8E7b6	100.00 ETH	0	5	🔑
ADDRESS	BALANCE	TX COUNT	INDEX	🔗

Figura 45

The screenshot shows a Firefox browser window with the following details:

- Voting app** tab is active.
- The address bar shows `localhost:3000/ceu#`.
- The main content area displays the "Voting App" interface with a sidebar menu and a "Añadir candidato" form.
- A Metamask extension overlay is visible on the right side of the screen, showing:
 - "Mis cuentas" section with "Account 1" (99.82684 ETH) checked.
 - "Importar cuenta ETH" button.
 - "Conectar Monedero Físico" button.
 - "Historial" section with two entries:
 - #10 - 8/16/2019 at 12:01: "Interacción Con Cont..." -0 ETH (CONFIRMADO)
 - #9 - 8/16/2019 at 12:00: "Interacción Con Cont..." -0 ETH (CONFIRMADO)
 - "Configuración" button.

Figura 48: Creando una clave con Metamask. Fuente propia.

Si no desea usar Ganache para generar cuentas para los candidatos, puede usar Metamask. Para ello debe hacer clic sobre el ícono del plugin que se puede apreciar en la *Figura 48* en la parte superior derecha, luego hacer clic sobre el círculo de color azul y rojo y posteriormente *Crear Cuenta*.

Una vez se haya añadido todos los candidatos o necesite comprobar si alguno ya está registrado de forma manual, puede listar los diferentes candidatos que ha registrado hasta el momento aún sin cerrar el modo de registro de candidatos. Para poder realizar esto, use la opción *Listar candidatos* que se puede ver en la parte izquierda en la *Figura 34*, lo cual desplegará una nueva página web con la lista de los candidatos tal y como se puede apreciar en la *Figura 49*.

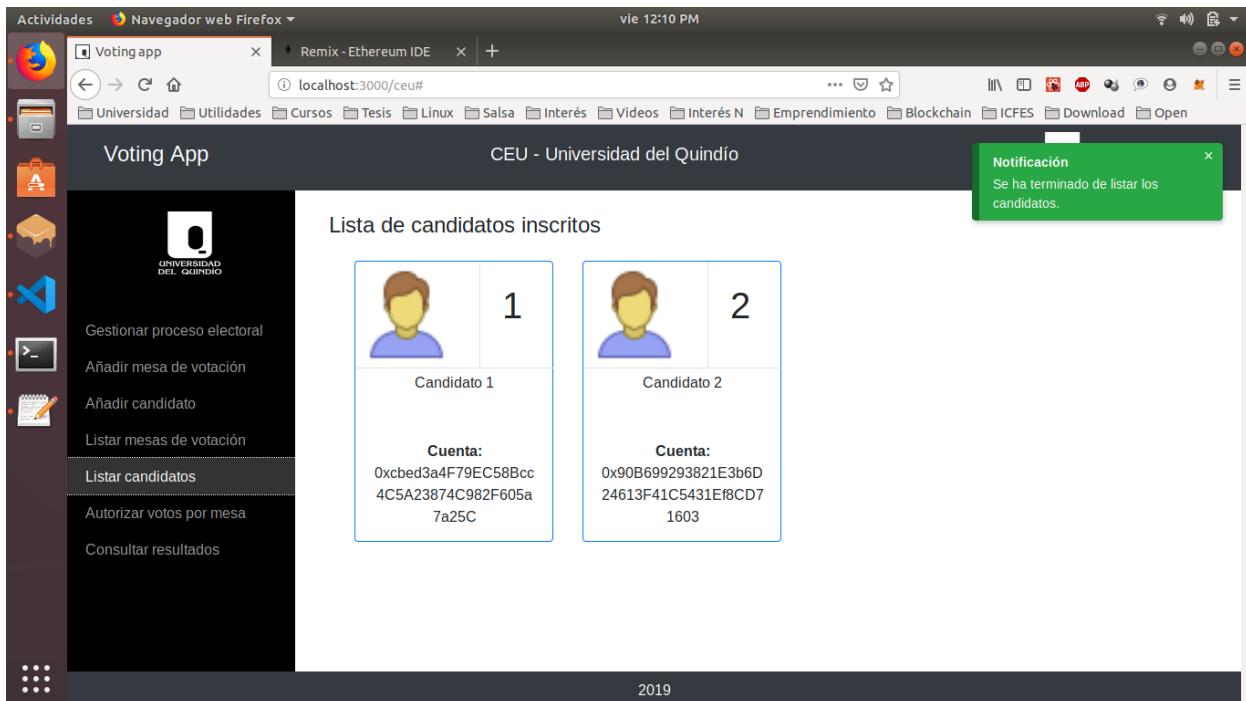


Figura 49: Listando candidatos registrados. Fuente propia.

Listar los diferentes candidatos no genera costos de ether, debido a que no se realizan modificaciones. Por lo tanto, se puede visualizarlos como se puede apreciar en la *Figura 49*, tantas veces como considere necesario.

Ahora que se tienen los diferentes candidatos debidamente registrados, puede terminar el modo registro de candidatos. Para ello, diríjase a la página que se puede ver en la *Figura 34* y seleccione la opción *Terminar registro de candidatos*, esto desplegará la página que se puede visualizar en la *Figura 50*, que una vez más, es lanzada por Metamask. Confirme el gasto de ether para poder realizar esta transacción y espere la confirmación que se puede visualizar en la *Figura 51*.

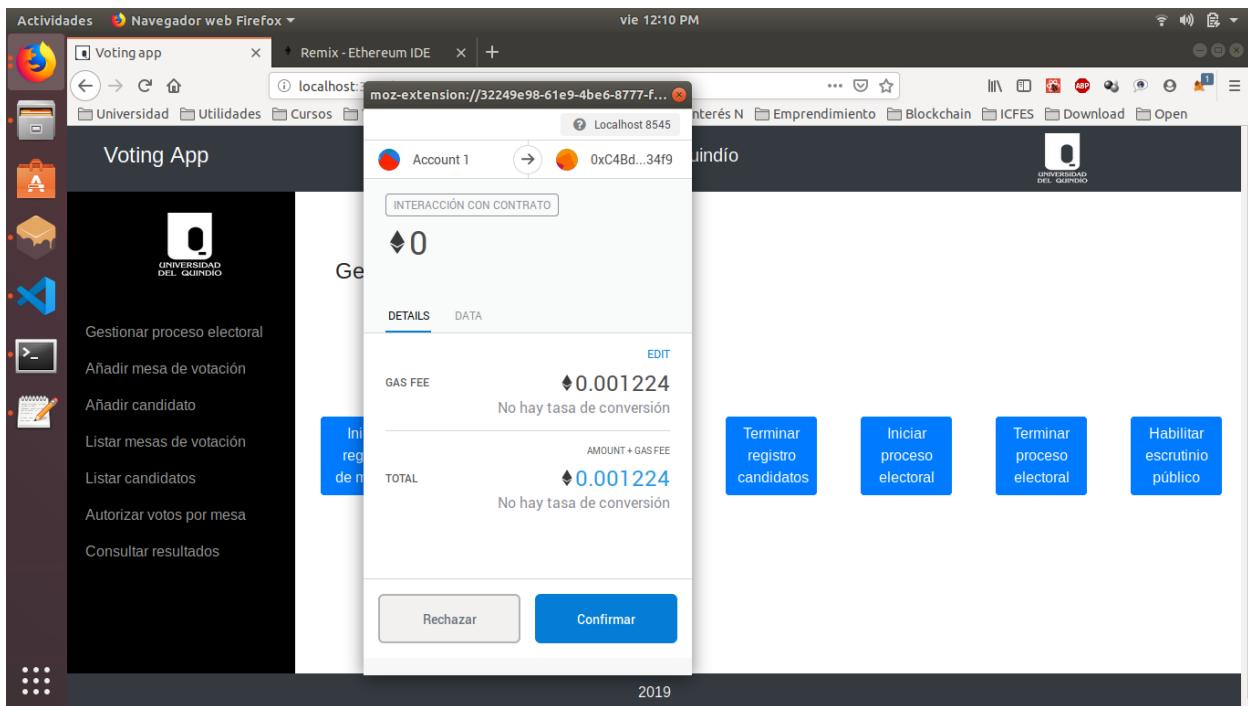


Figura 50: Terminando registro de candidatos. Fuente propia.

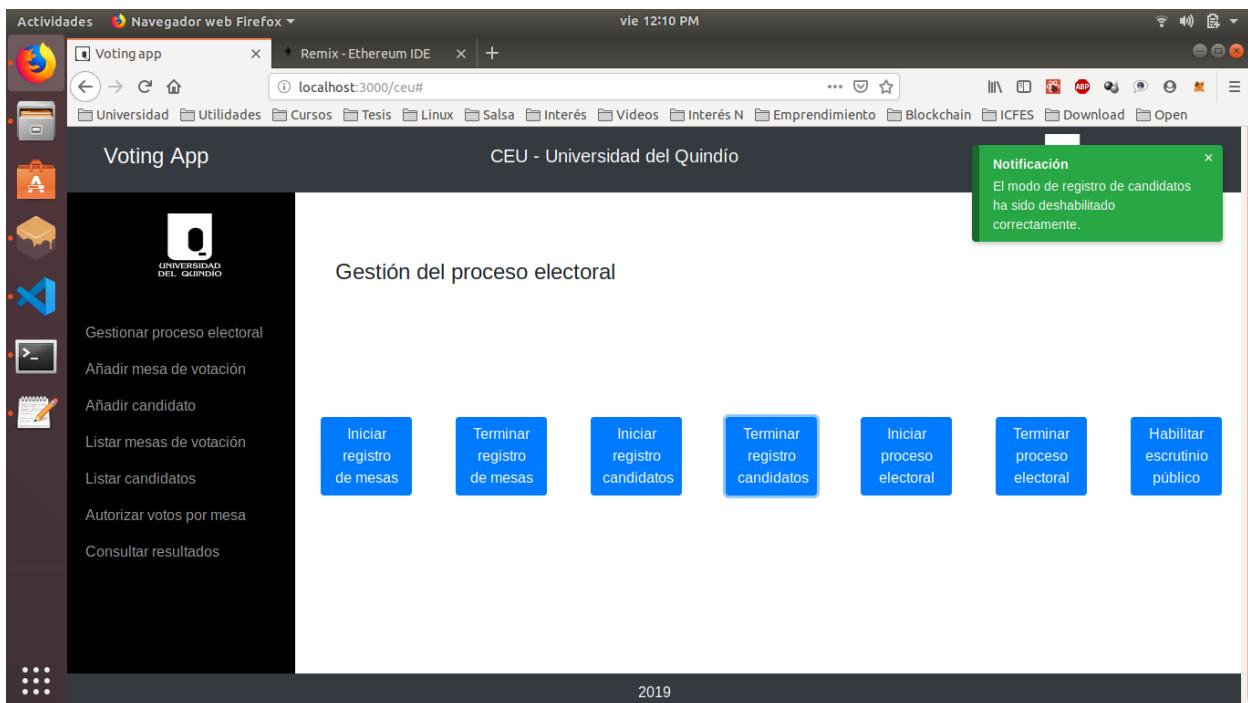


Figura 51: Confirmación modo registro de candidatos deshabilitado. Fuente propia.

Resumiendo, hasta el momento tenemos configurado un entorno de trabajo, registradas las diferentes mesas y candidatos; así que es momento de empezar la Consulta de opinión.

2.3. Realizando la Consulta de opinión

Para poder realizar la Consulta de opinión o jornada electoral, se debe autorizar los votos a las diferentes mesas de votación, para ello, se debe ingresar la página principal de Voting App y seleccionar entrar como CEU (ver *Figura 33*), una vez ingrese a través de este rol, se desplegará la página que se puede apreciar en la *Figura 34*. En esta página, debe ir a la opción *Autorizar votos por mesa* que está en la parte izquierda en la *Figura 34*. Esta acción desplegará la siguiente interfaz de usuario que se presenta en la *Figura 52*.

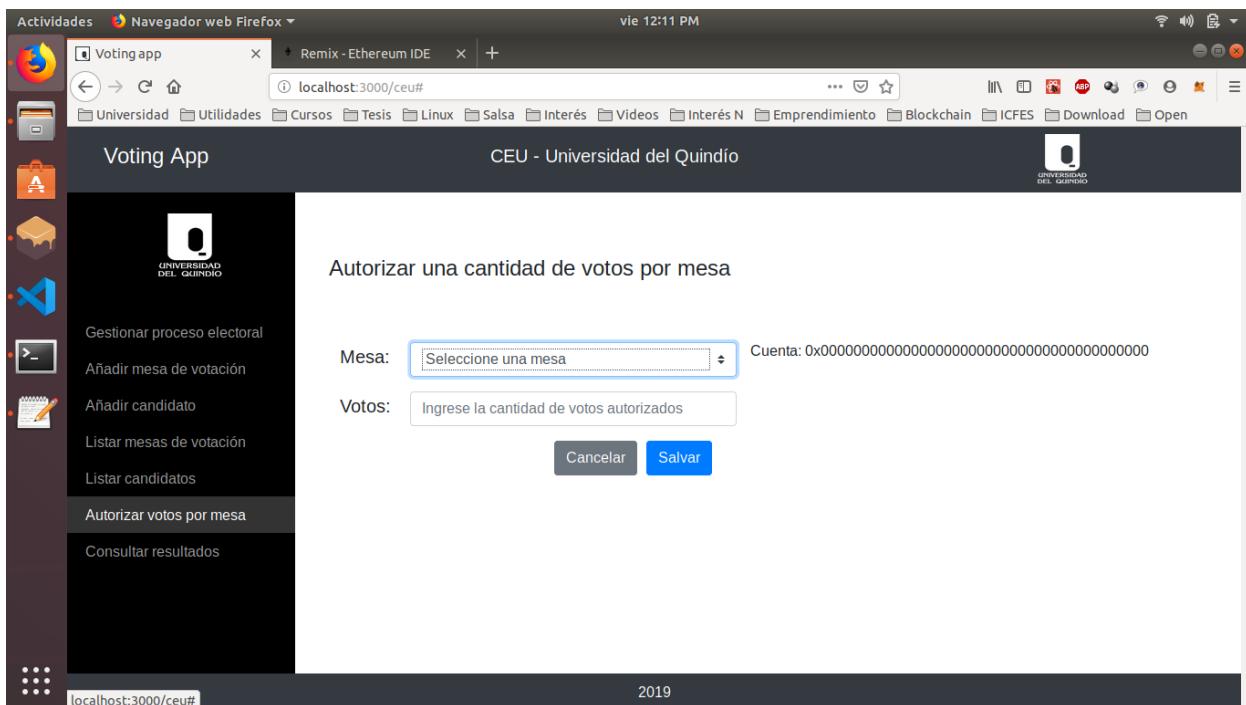


Figura 52: Interfaz de usuario para autorizar votos a las mesas. Fuente propia.

En esta sección debe seleccionar la mesa con el código a la cual desea autorizar votos. Cuando seleccione la mesa, para debe asegurarse de ser la mesa correcta, por lo tanto, se presenta la dirección de la misma a un lado. Luego ingrese la cantidad de votos que es igual a la cantidad de votantes inscritos en la misma. En la *Figura 53* se presenta un ejemplo de registro de la mesa con el código 1.

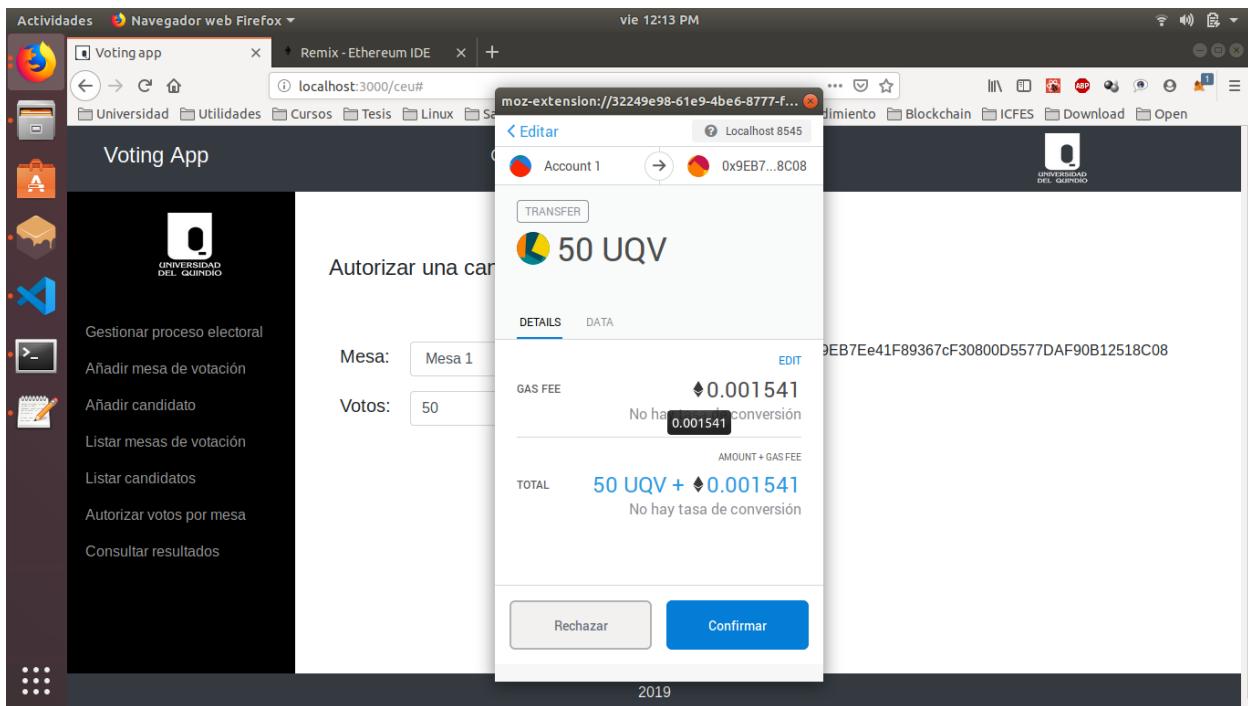


Figura 53: Autorizando 50 votos a la mesa con el código 1. Fuente propia.

Como podrá notar, la interfaz de Metamask tiene información adicional (50 UQV) a diferencia de la que se presenta en la *Figura 50*. Esto es debido a que se están transfiriendo un tipo particular de balotas a la mesa, que posteriormente al finalizar la jornada electoral, serán transferidas al candidato, esto con el fin de tener un mayor control sobre los votos y ser más transparentes en el proceso.

Cuando autorice los diferentes votos para la mesa, debe esperar la confirmación de que se han autorizado correctamente con la notificación presentada en la *Figura 54*.

Este procedimiento debe realizarlo con cada mesa que esté habilitada para votar, usando la página de mostrada en la *Figura 53*, de lo contrario, esta no podrá trabajar correctamente y no se emitirán votos desde la misma.

Como paso siguiente, se debe habilitar la jornada electoral ahora que las mesas cuentan con los respectivos votos. Para ello, cambie a la opción de *Gestionar proceso electoral*, que se puede apreciar en el lado izquierdo en la *Figura 54*, la cual cambiará a la página que se puede apreciar en la *Figura 34*. Una vez en esta página nueva, se debe elegir la opción *Iniciar proceso electoral*, la cual tiene la finalidad de configurar los contratos para poder votar, dado que esta opción está bloqueada por defecto al igual que el resto de modos. Como es de costumbre, Metamask pide confirmar la transacción tal y como se puede apreciar en la *Figura 55*.

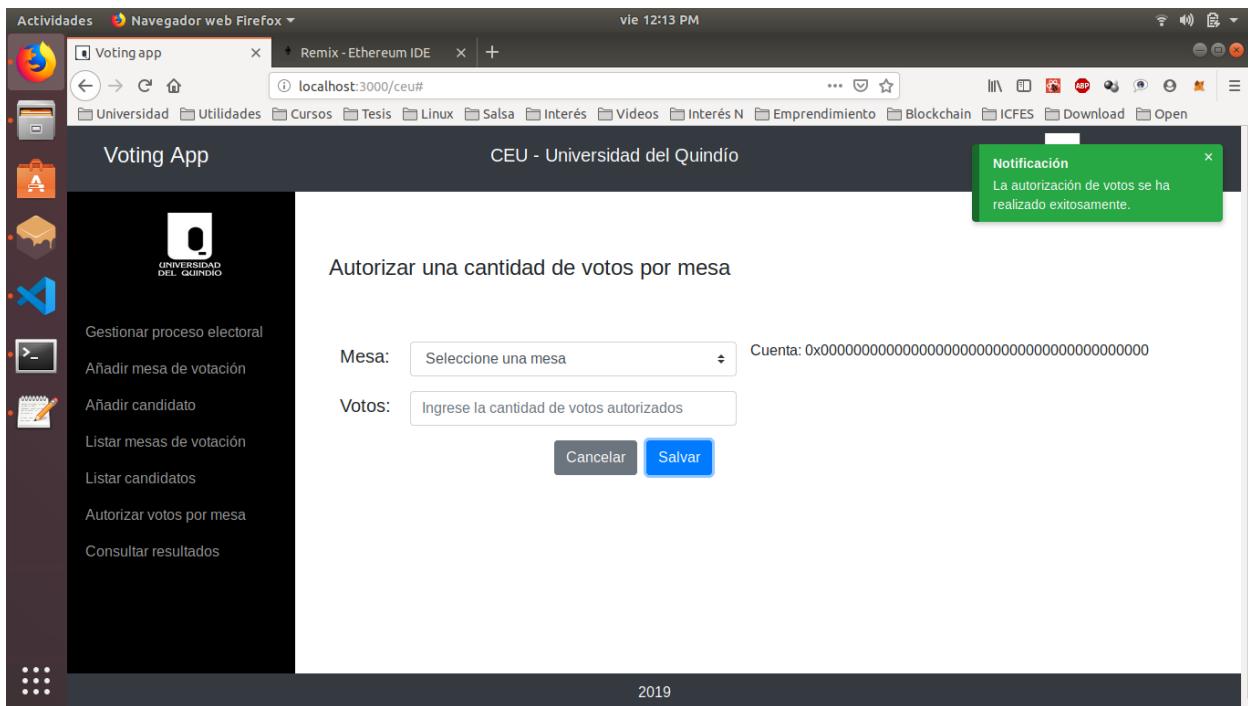


Figura 54: Confirmación de autorización de votos. Fuente propia.

Acto seguido, la mesa de votación debe dirigirse a la página principal de Voting App e ingresar por el rol *Mesa*, una vez ingrese podrá visualizar la página de la Figura 56.

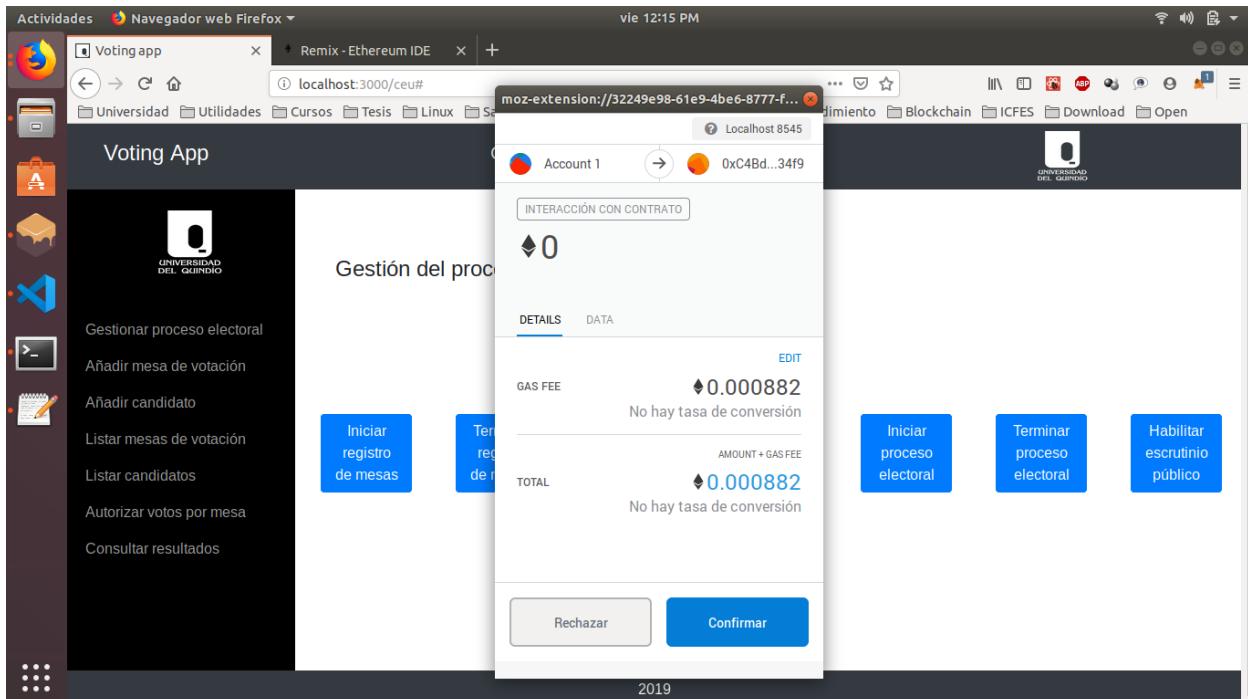


Figura 55: Confirmación para iniciar proceso electoral. Fuente propia.

La página tiene su cuenta, cantidad de votantes inscritos y autorizar voto (ver Figura 56).

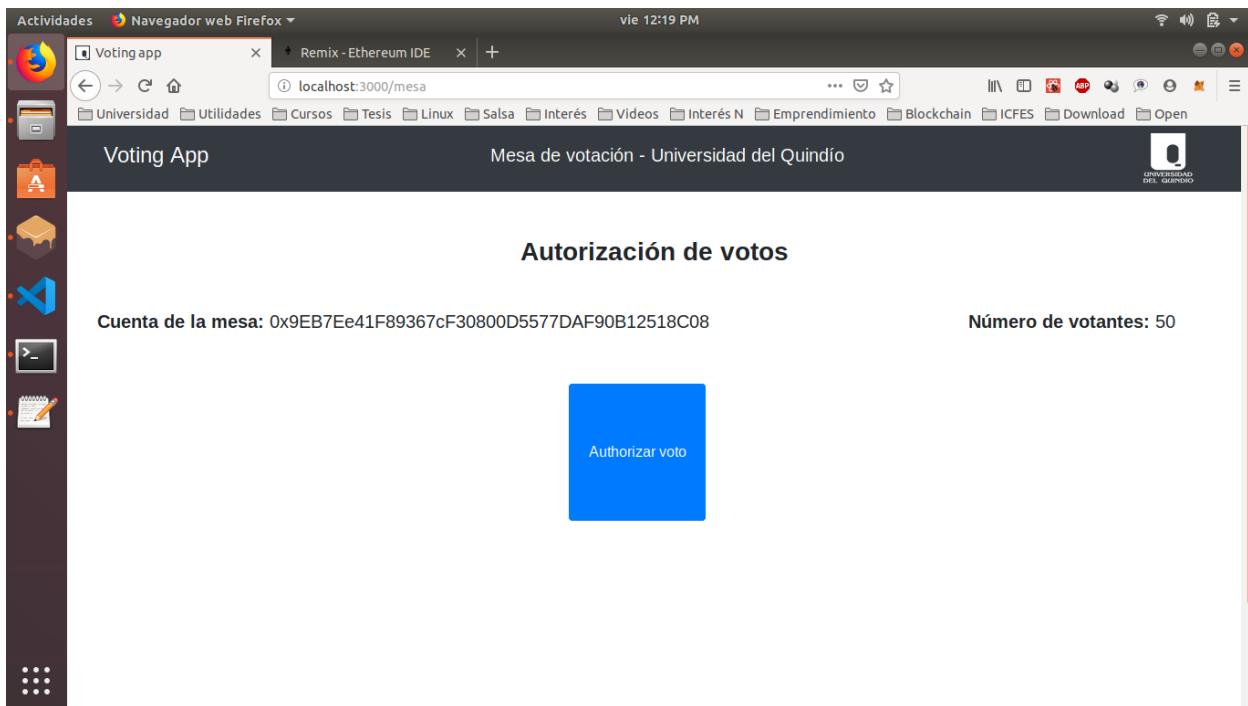


Figura 56: Página principal de la mesa de votación. Fuente propia.

Cuando un votante se acerca a la mesa, esta lo identifica manualmente a través de los listados que esta contiene en papel y le autoriza el voto, tal y como se puede ver en la *Figura 57*.

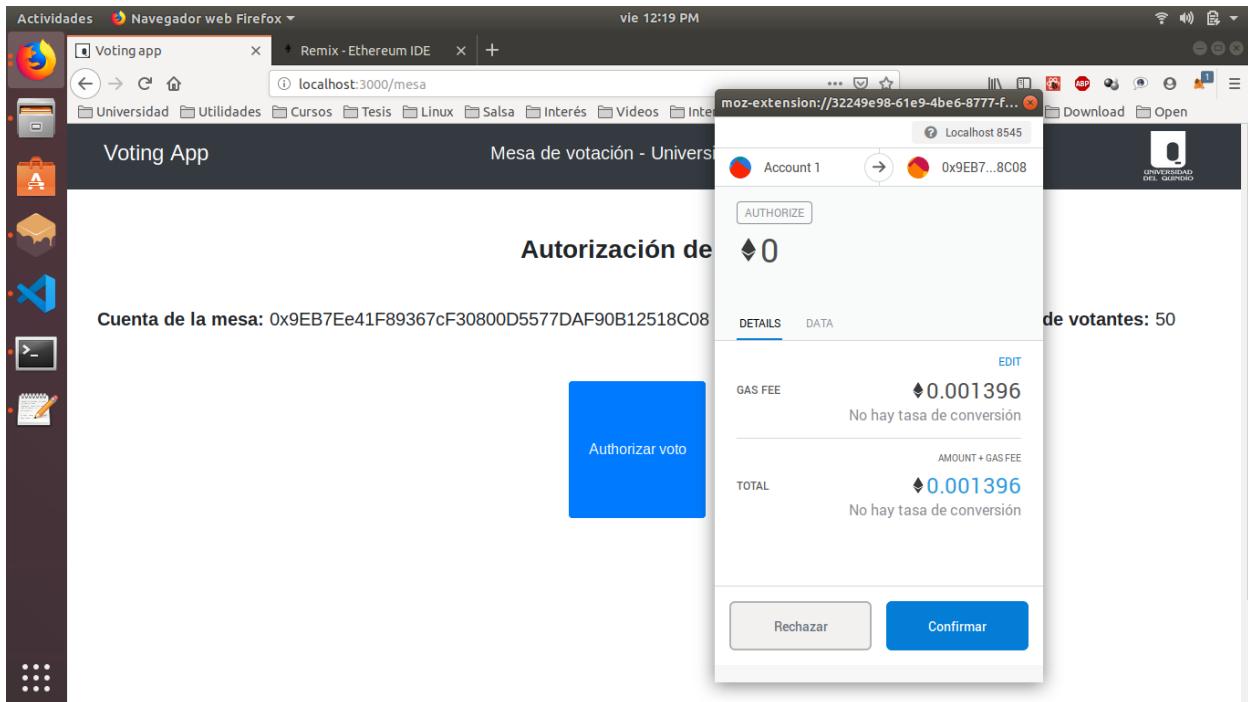


Figura 57: Autorizando voto. Fuente propia.

Esto autoriza al votante emitir un voto usando la cuenta de la mesa, por lo que no se puede ligar su elección de candidato al mismo. Se autoriza el gasto de ether para realizar la transacción y se debe esperar la notificación que se presenta en la *Figura 58*. Con esta confirmación de autorización exitosa, se le debe pedir al elector que se desplace hasta el cubículo donde tendrá una interfaz para votar, esta se puede apreciar en la *Figura 59*. A esta interfaz se puede llegar a partir de la página que se presenta en la *Figura 33*, e ingresando por la opción *Votar*.

Cuando el elector emite su voto, gasta la única oportunidad que tenía y no puede emitir más votos o cambiar su elección una vez emitido.

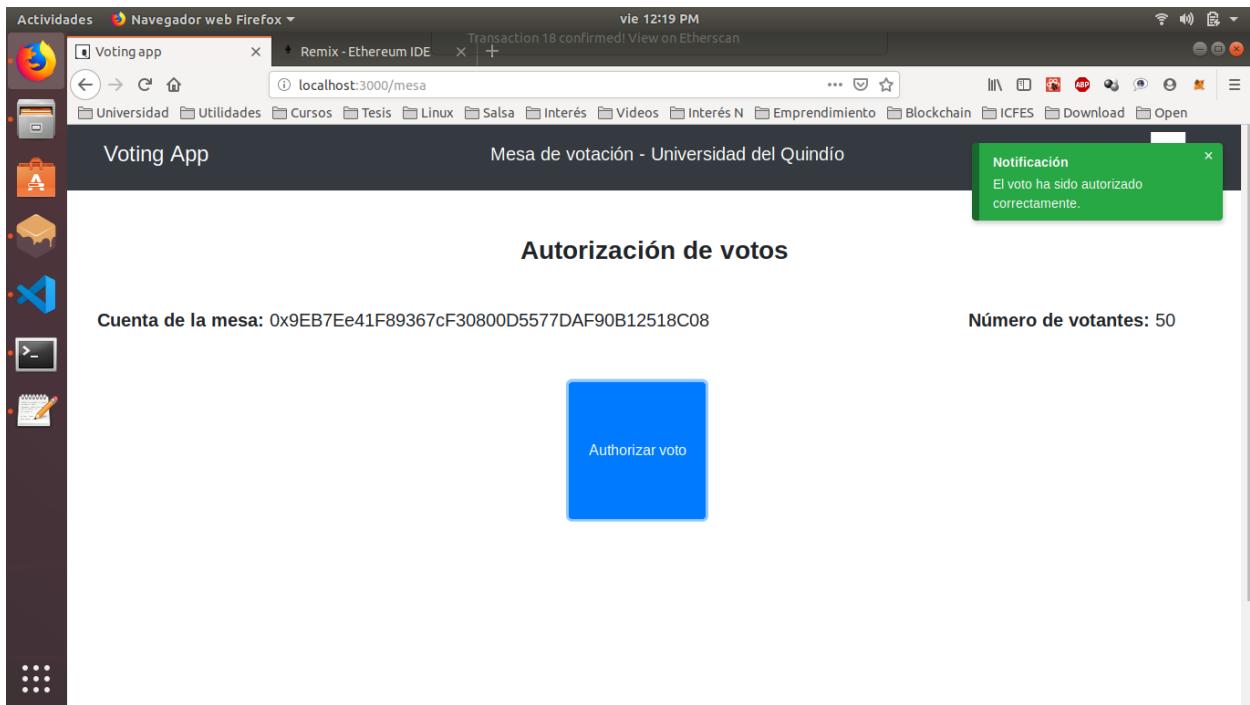


Figura 58: Confirmación autorización de voto para el elector desde la mesa. Fuente propia.

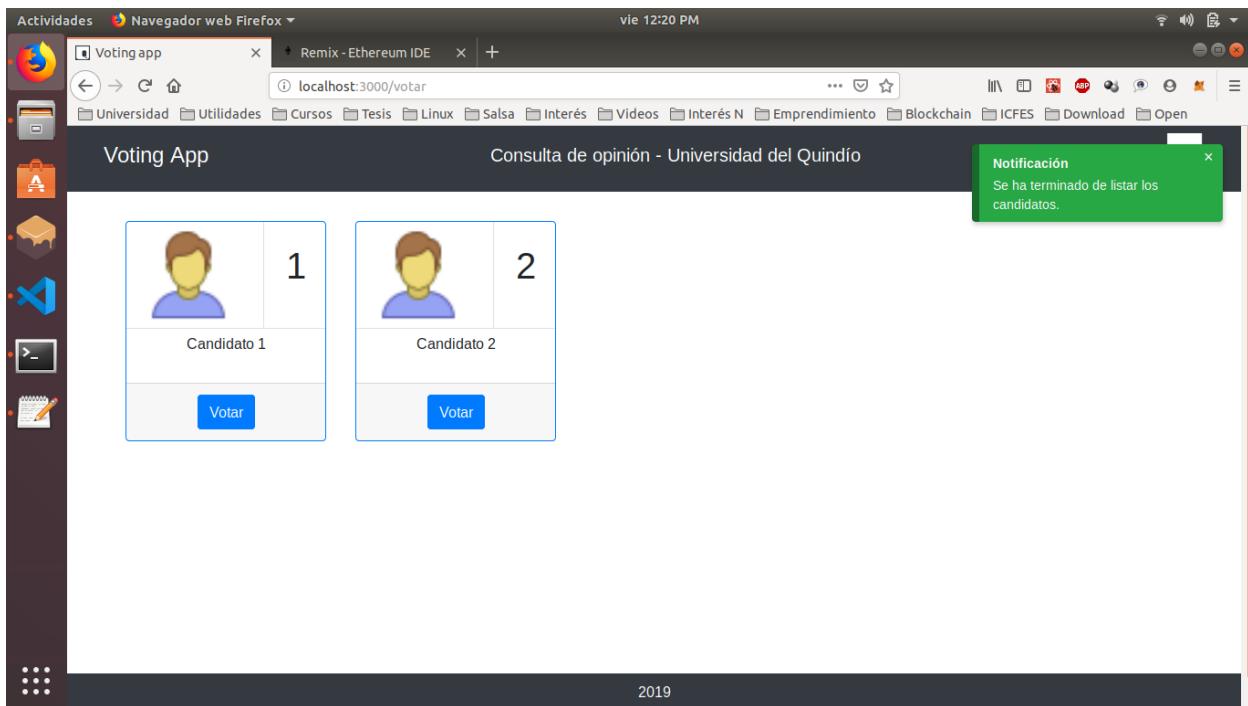


Figura 59: Listado de candidatos para elegir y votar. Fuente propia.

Cuando el votante presiona *Votar* por un candidato, se le pide confirmar (ver Figura 60).

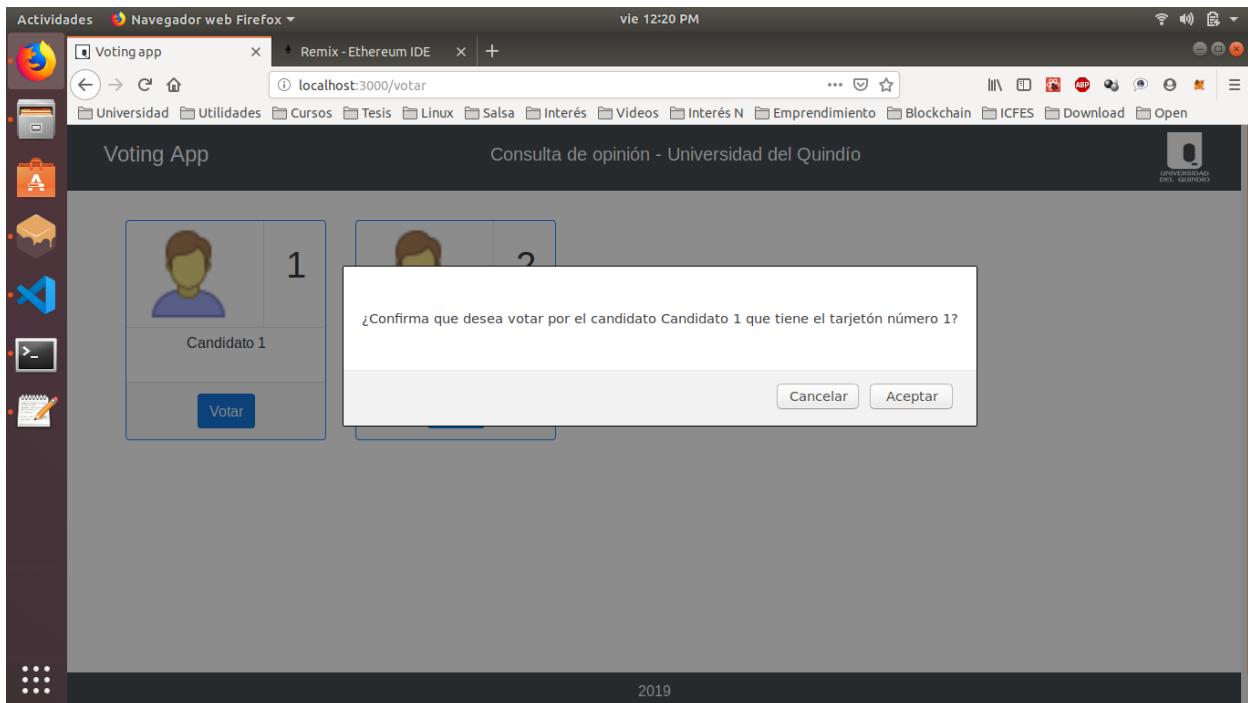


Figura 60: Confirmación de votar por el candidato seleccionado. Fuente propia.

Si confirma, Metamask pide confirmar el voto, como se puede ver en la Figura 61.

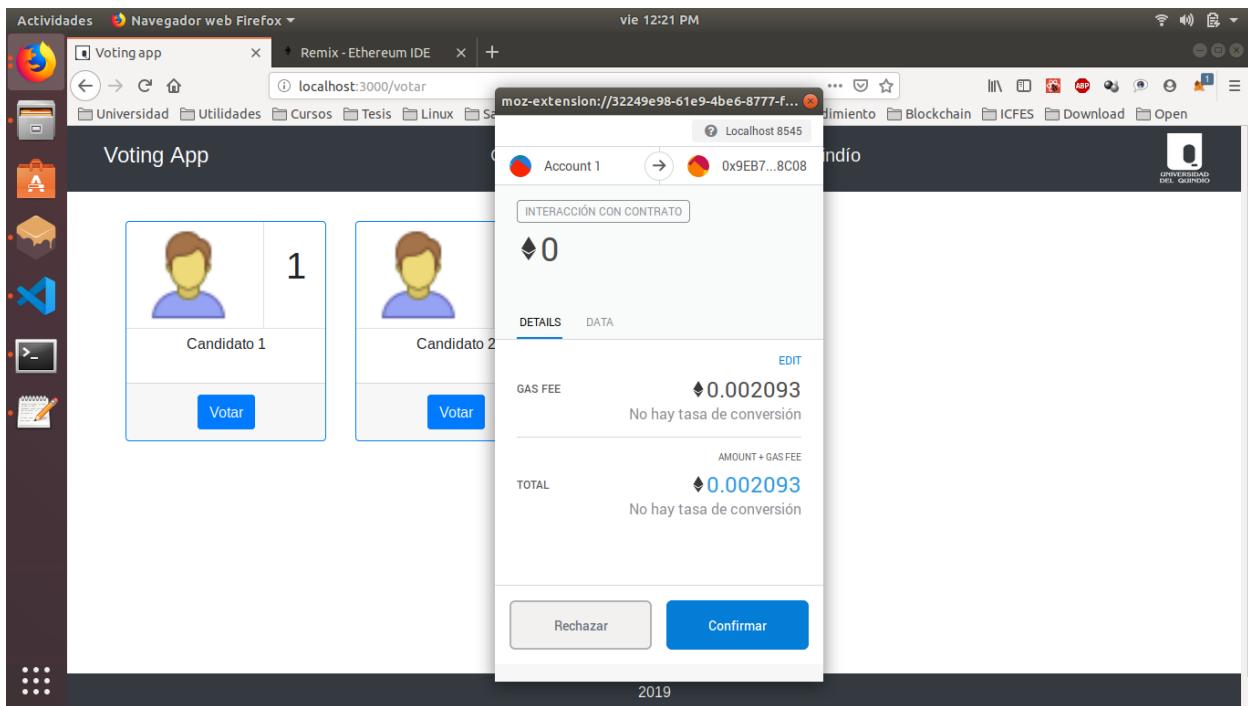


Figura 61: Confirmar transacción por parte del votante. Fuente propia.

El votante espera a que se emita su voto y Voting App le devuelva un error o la confirmación de que su voto fue guardado correctamente tal y como se puede apreciar en la Figura 62.

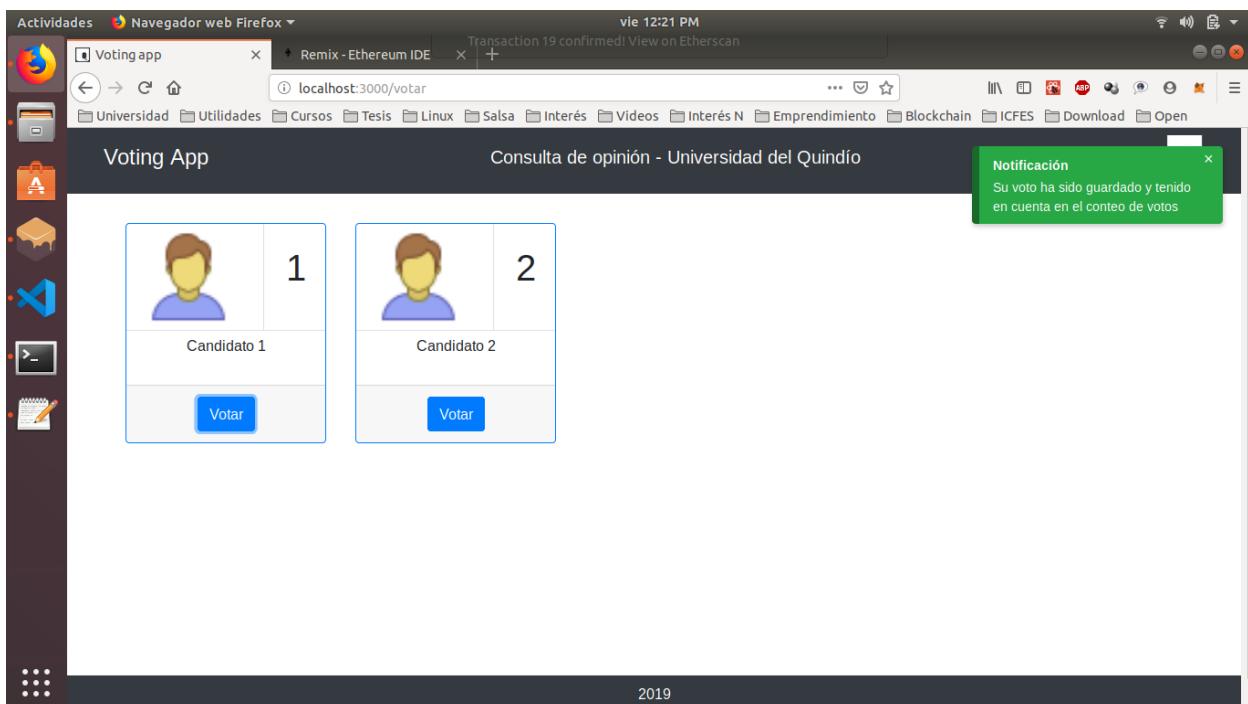


Figura 62: Confirmación de voto emitido correctamente. Fuente propia.

Para que otro elector pueda votar, se debe repetir el procedimiento desde la *Figura 56*.

Una vez que la jornada electoral concluya, en cuanto al tiempo, quien esté a cargo de gestionar la jornada (asumió el rol de CEU), debe ingresar a la página principal de Voting App (ver *Figura 33*) e ingresar a través del rol *CEU*; lo cual le cargará la página que se puede apreciar en la *Figura 34*. Una vez esté en la página que se menciona, debe terminar manualmente el proceso electoral, para lo cual, debe elegir la opción *Terminar proceso electoral*. Esto desactiva los contratos para que no puedan recibir más votos y rechacen cualquier intento. Una vez se presiona sobre esta opción, se debe esperar la notificación que se presenta en la *Figura 63*.

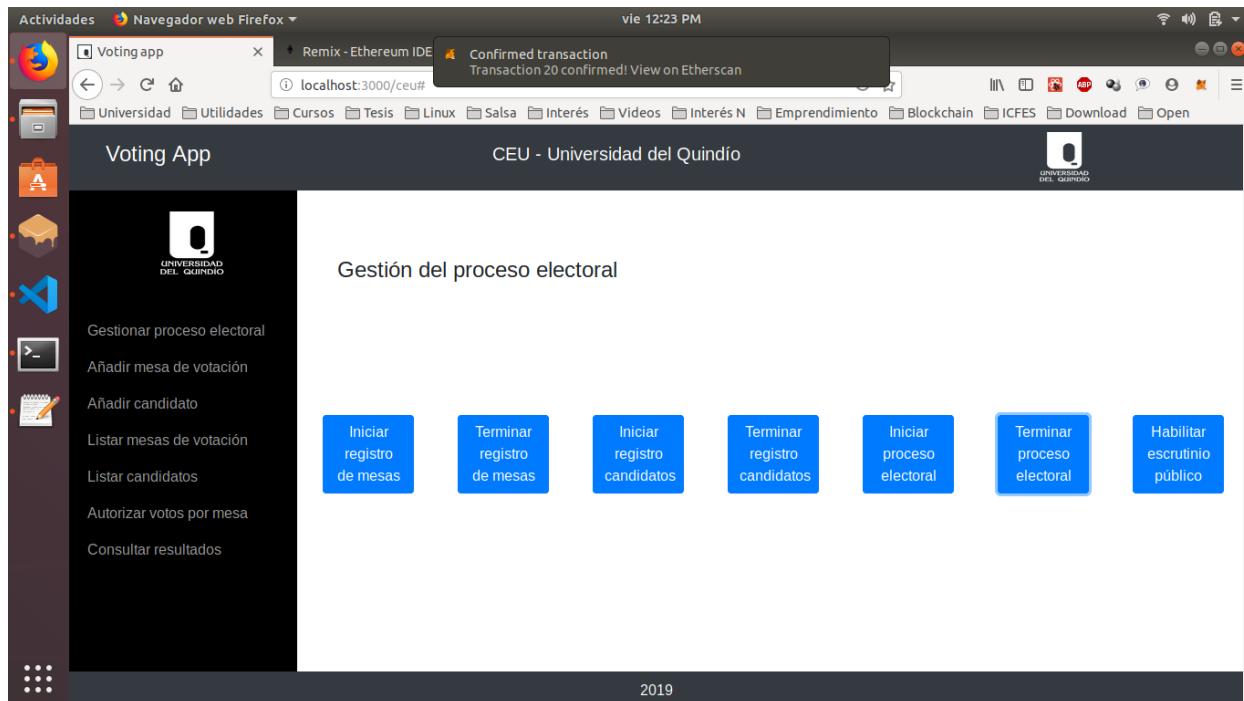


Figura 63: Confirmación de la transacción de cierre del proceso electoral. Fuente propia.

Acto seguido, debe habilitar el escrutinio público usando la página de la *Figura 34*.

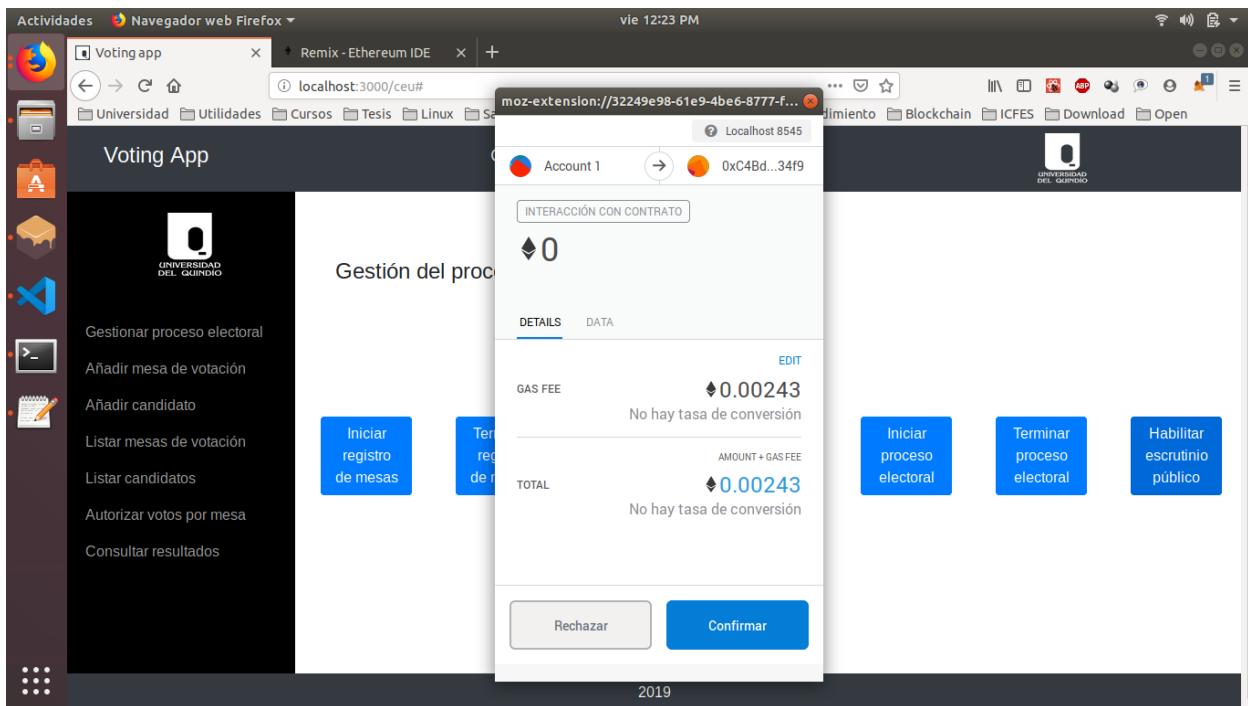


Figura 64: Autorizando transacción para habilitar el escrutinio público. Fuente propia.

Una vez confirma la transacción, espera la notificación que aparece en la Figura 65.

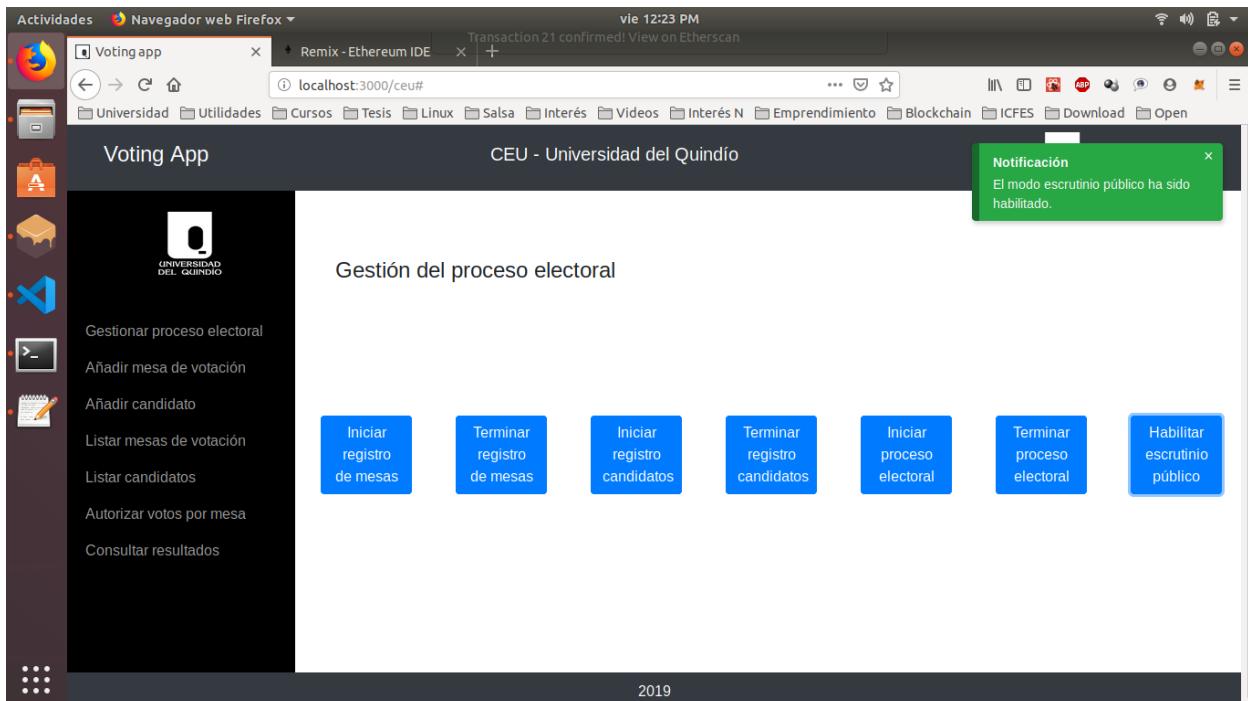


Figura 65: Confirmación exitosa, el escrutinio es público ahora. Fuente propia.

2.4. Consultando resultados

Una vez que el escrutinio es público, se lo puede consultar desde 3 perspectivas las cuales se presentarán en este apartado.

2.4.1. Resultados para el CEU

Quien esté gestionando el proceso electoral o quien asume el representante del CEU, para este paso particular de caso de estudio, puede consultar los resultados ingresando a la página principal de Voting App (ver *Figura 33*) e ingresar a través del rol CEU, que es donde se despliega la página que se puede apreciar en la *Figura 34*. Una vez en esta página, debe seleccionar la opción *Consultar resultados*, que es donde se listarán los diferentes candidatos con sus respectivos votos, tal y como se puede apreciar en la *Figura 66*.

The screenshot shows a Firefox browser window with the title 'Actividades Navegador web Firefox'. The address bar displays 'localhost:3000/ceu#'. The main content area is titled 'Resultados finales de la Consulta de opinión' for 'CEU - Universidad del Quindío'. It lists two candidates:

Candidato	Nº. votos
Candidato 1	1
Candidato 2	0

A green notification box in the top right corner says 'Notificación Se ha terminado de listar los candidatos.'. On the left sidebar, there are several options: 'Gestionar proceso electoral', 'Añadir mesa de votación', 'Añadir candidato', 'Listar mesas de votación', 'Listar candidatos', 'Autorizar votos por mesa', and 'Consultar resultados'. The date '2019' is at the bottom right of the page.

Figura 66: Consulta de resultados por parte del CEU. Fuente propia.

2.4.2. Resultados público en general

Otra forma de consultar los resultados y que es de interés para todas aquellas personas que de alguna forma participan en el proceso electoral, es a través de la página que se puede apreciar en la *Figura 67*.

Al igual que cualquier otro tipo de listados de los cuales se disponga para los diferentes usuarios de Voting App, esta opción no representa ningún cobro en ether a diferencia de las transacciones que se habían realizado en casos anteriores, esto es debido a que no se alteran registros dentro del sistema.

A parte de estas dos formas presentadas, existe una última forma mediante la cual se pueden consultar los resultados, aclarando que fue puesta intencionadamente en Voting App a través de interfaces gráficas de usuario, y es la que se presenta en la sección 2.4.3.

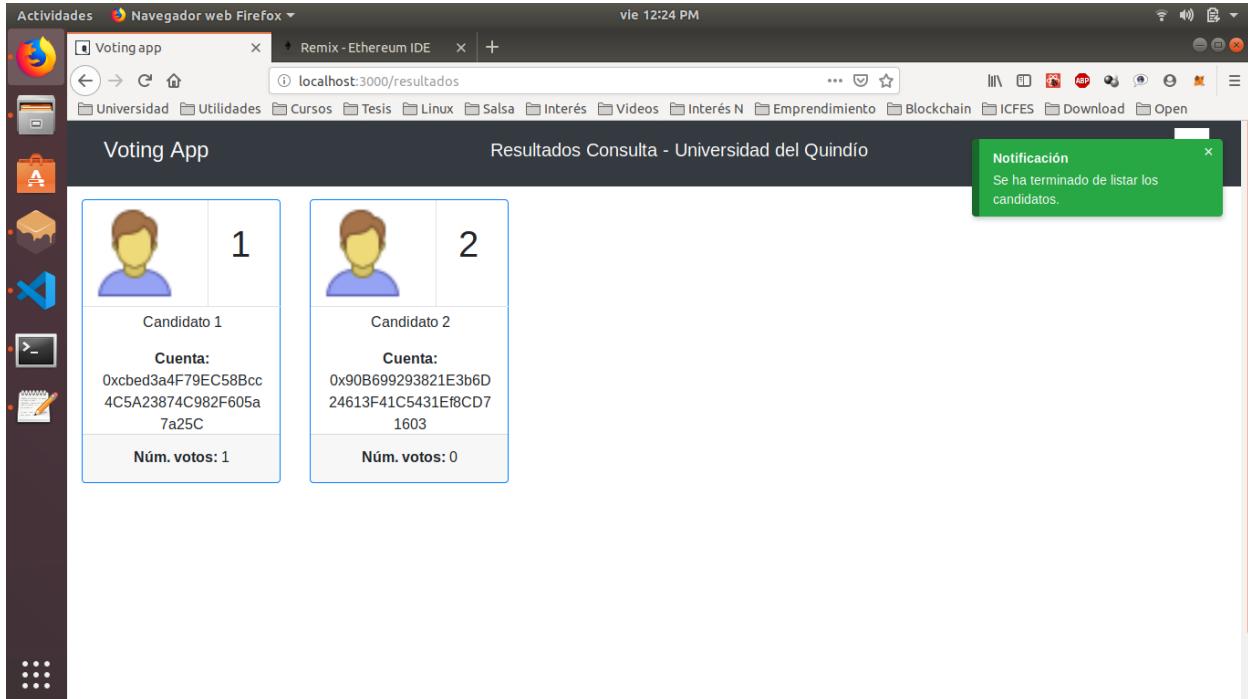


Figura 67: Consulta de resultados por parte del público en general. Fuente propia.

2.4.3. Auditando resultados

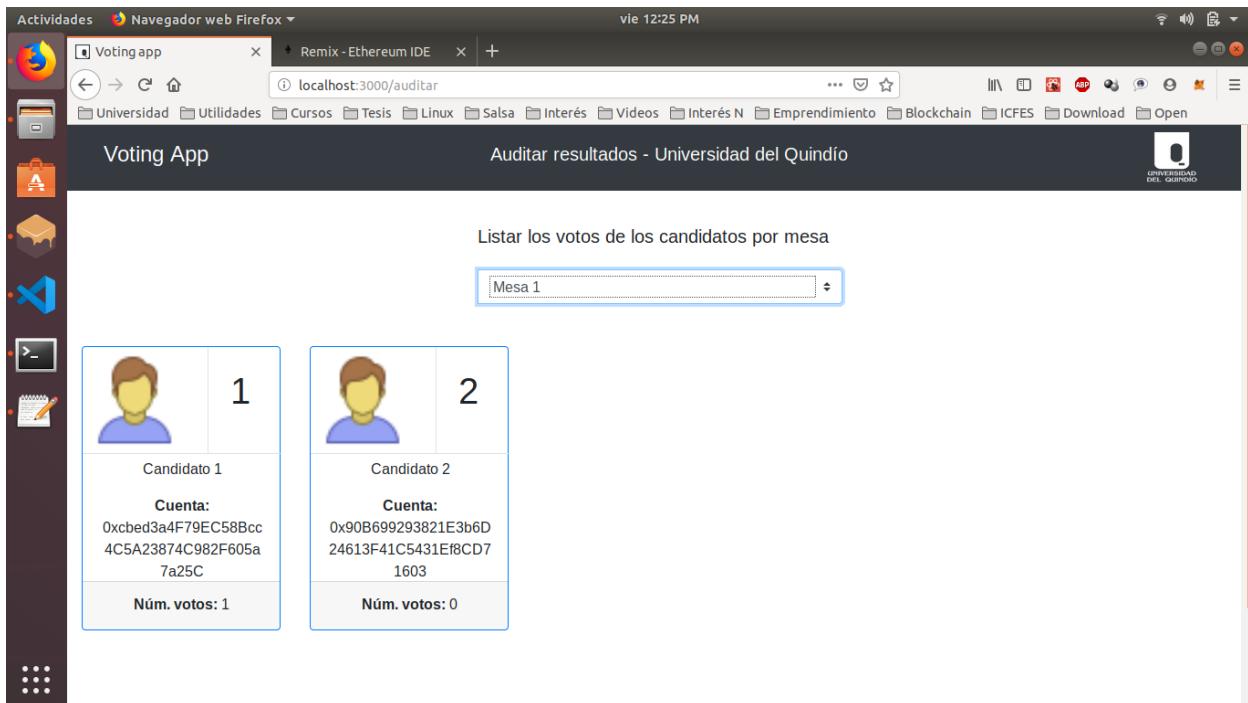


Figura 68: Auditando resultados de la mesa 1. Fuente propia.

Auditar los resultados implica consultar la cantidad de votos que ha tenido un candidato en cada una de las mesas que se disponen, esto se hace con la finalidad de realizar auditoría sobre los datos y así, generar una trazabilidad de los diferentes resultados de cada candidato en el escrutinio final.

Esto es todo en cuanto al manejo de Voting App, a continuación, se presentan algunos apuntes que pueden resultar útiles para evaluar la tecnología, para ello se empieza por saber, cuánto ether se ha gastado en la simulación que se realizó para escribir este manual, la cual se puede ver en la *Figura 69*.

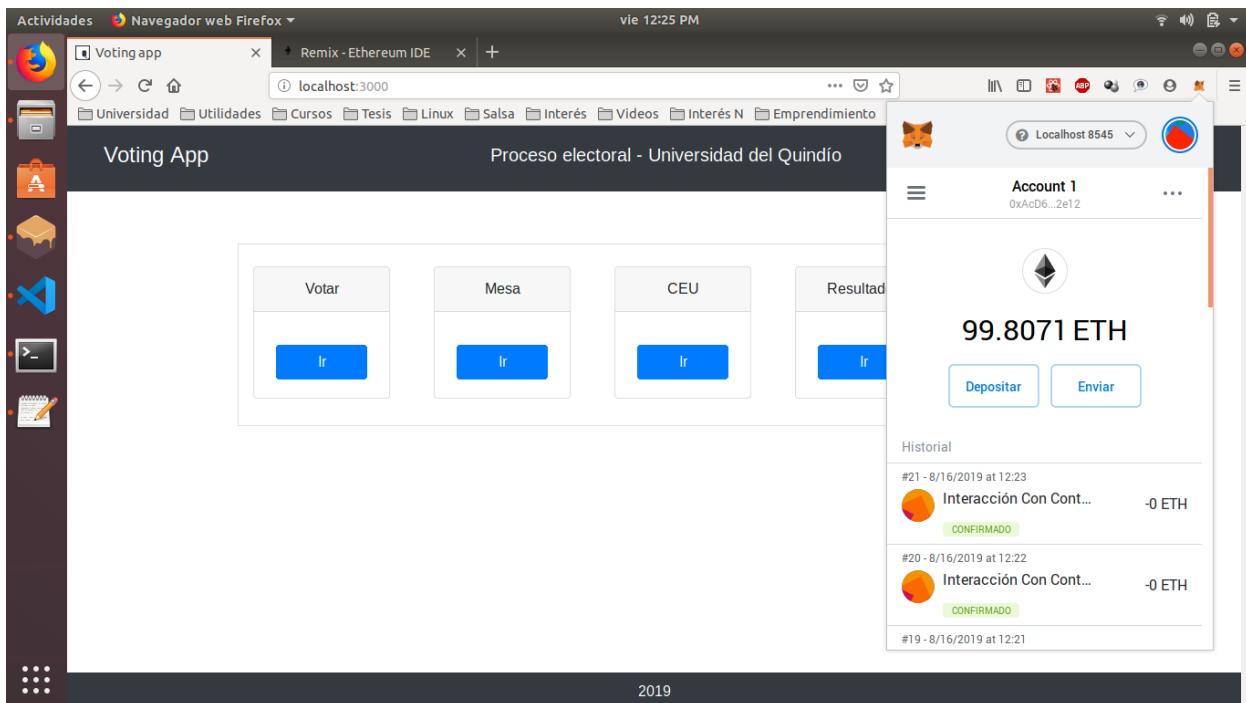


Figura 69: Ether gastado en este procedimiento. Fuente propia.

Este gasto fue dado por habilitar los diferentes modos, registrar mesas, candidatos, transferir tokens, autorizar votos por parte de la mesa y finalmente por emitir los diferentes votos. Ahora, en la *Figura 70* se puede visualizar cuántos bloques fueron minados.

Actividades Ganache

vie 12:26 PM Ganache

SEARCH FOR BLOCK NUMBERS OR TX HASHES

ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS	LOGS	
CURRENT BLOCK 22	GAS PRICE 2000000000	GAS LIMIT 6721975	HARDFORK PETERSBURG	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:8545	MINING STATUS AUTOMINING
WORKSPACE TESIS						
BLOCK 22	MINED ON 2019-08-16 12:23:53			GAS USED 80991		1 TRANSACTION
BLOCK 21	MINED ON 2019-08-16 12:22:57			GAS USED 128547		1 TRANSACTION
BLOCK 20	MINED ON 2019-08-16 12:21:39			GAS USED 39777		1 TRANSACTION
BLOCK 19	MINED ON 2019-08-16 12:19:37			GAS USED 46528		1 TRANSACTION
BLOCK 18	MINED ON 2019-08-16 12:16:13			GAS USED 29397		1 TRANSACTION
BLOCK 17	MINED ON 2019-08-16 12:14:38			GAS USED 36353		1 TRANSACTION
BLOCK 16	MINED ON 2019-08-16 12:13:22			GAS USED 51353		1 TRANSACTION
BLOCK 15	MINED ON 2019-08-16 12:10:51			GAS USED 40802		1 TRANSACTION
BLOCK 14	MINED ON 2019-08-16 12:09:23			GAS USED 245965		1 TRANSACTION
BLOCK	MINED ON			GAS USED		1 TRANSACTION

Figura 70: Bloques minados en Ganache para la simulación. Fuente propia.

También se pueden apreciar las diferentes transacciones en Ganache (ver Figura 71).

Actividades Ganache

vie 12:26 PM Ganache

SEARCH FOR BLOCK NUMBERS OR TX HASHES

ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS	LOGS	
CURRENT BLOCK 22	GAS PRICE 2000000000	GAS LIMIT 6721975	HARDFORK PETERSBURG	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:8545	MINING STATUS AUTOMINING
WORKSPACE TESIS						
TX HASH 0x8000f4d6d8c792d6cf718129ef00c40659f7932ff8b2ae6bf51f314fe2feba						CONTRACT CALL
FROM ADDRESS 0xAcd65551f0BFC8a9D1ee77a42d15C2eA0e562e12		TO CONTRACT ADDRESS 0xC4Bd7FE8E296405E86d0a56CDE74502169F134f9			GAS USED 80991	VALUE 0
TX HASH 0x39b36e4a6e4b024fc882d2233ba198dc07e03f9fd6f86ceca4476d9d8c7c915e						CONTRACT CALL
FROM ADDRESS 0xAcd65551f0BFC8a9D1ee77a42d15C2eA0e562e12		TO CONTRACT ADDRESS 0xC4Bd7FE8E296405E86d0a56CDE74502169F134f9			GAS USED 128547	VALUE 0
TX HASH 0x17e7224b46c8064e887024900956c4606bed7c65f8ae2b0feee8c837f4d500d4						CONTRACT CALL
FROM ADDRESS 0xAcd65551f0BFC8a9D1ee77a42d15C2eA0e562e12		TO CONTRACT ADDRESS 0x9EB7Ee41F89367cF30800D5577DAF90B12518C08			GAS USED 39777	VALUE 0
TX HASH 0x9b47233df453d1a02d3ad2386723048ae393b0f8011c6f42d308c9a5132386f0						CONTRACT CALL
FROM ADDRESS 0xAcd65551f0BFC8a9D1ee77a42d15C2eA0e562e12		TO CONTRACT ADDRESS 0x9EB7Ee41F89367cF30800D5577DAF90B12518C08			GAS USED 46528	VALUE 0

Figura 71: Transacciones realizadas por la página web y guardadas por Ganache. Fuente propia.

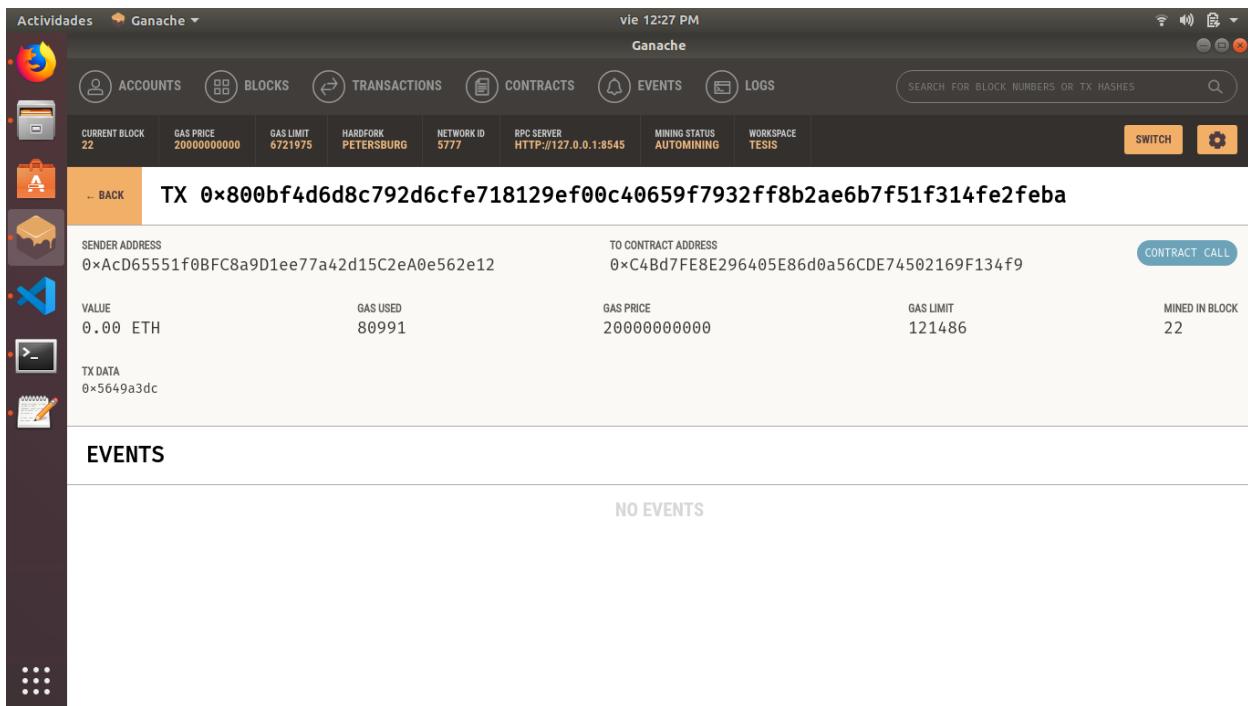


Figura 72: Detalles de una transacción de Ganache. Fuente propia.

Metamask también consulta estos datos y puede presentarlos, para ello, ver la

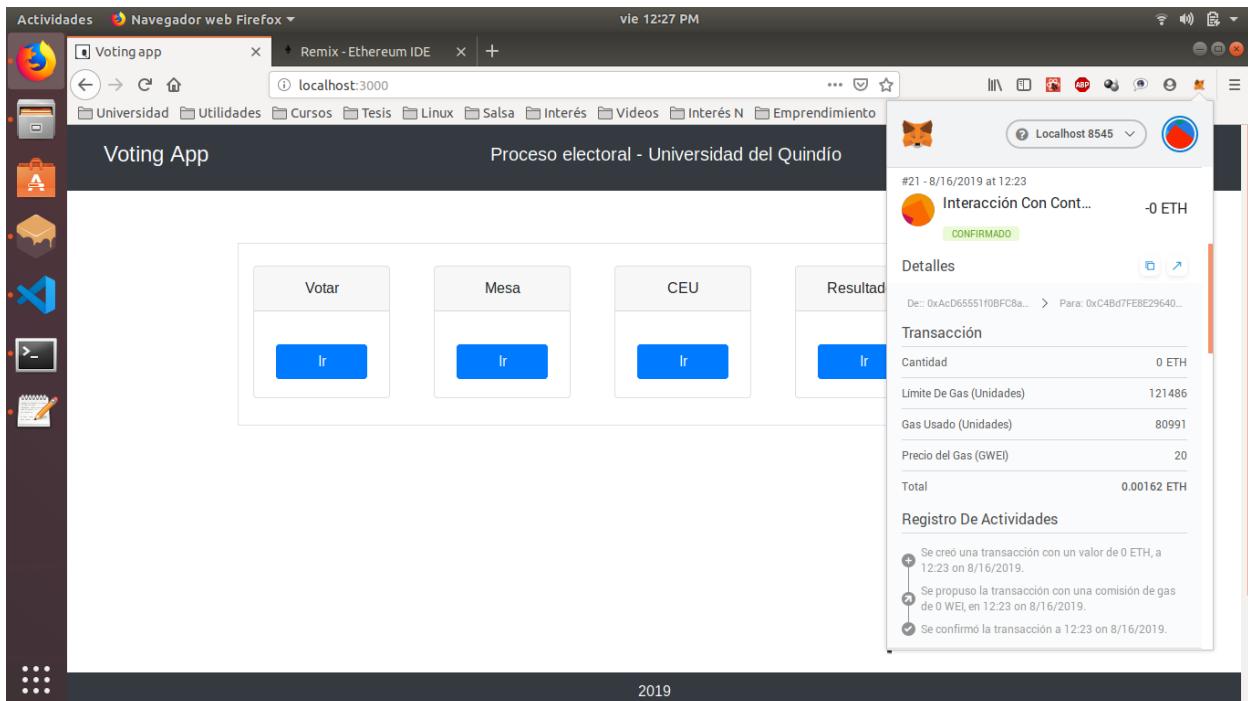


Figura 73: Detalles una transacción aleatoria generada desde la página web. Fuente propia.

Al igual que se pueden visualizar los detalles de las transacciones, también es posible ver los detalles de un bloque cualquiera, en este caso el último bloque generado. Estos detalles se pueden visualizar en la

The screenshot shows the Ganache application interface. At the top, there's a header bar with the title 'Ganache' and a timestamp 'vie 12:28 PM'. Below the header are several navigation tabs: 'ACCOUNTS', 'BLOCKS', 'TRANSACTIONS', 'CONTRACTS', 'EVENTS', and 'LOGS'. A search bar is located at the top right. The main content area is titled 'BLOCK 22'. It displays the following information:

GAS USED	GAS LIMIT	MINED ON	BLOCK HASH
80991	6721975	2019-08-16 12:23:53	0x77ba1fc7f6f75ecc36e0fd4a47439bdb2f31a612ee37057b3265ff38528d051e

Below this, there's a section for a transaction with the TX HASH: 0x800bf4d6d8c792d6cfe718129ef00c40659f7932ff8b2ae6b7f51f314fe2feba. It shows the 'FROM ADDRESS' (0xAcD65551f0BFC8a9D1ee77a42d15C2eA0e562e12) and the 'TO CONTRACT ADDRESS' (0xC4Bd7FE8E296405E86d0a56CDE74502169F134f9). The 'GAS USED' is 80991 and the 'VALUE' is 0.

Figura 74: Detalles de un bloque en Ganache. Fuente propia.