

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

TABLA DE CONTENIDO

ALCANCE SISTEMA DE FUMIGACIÓN	2
ROLES.....	2
TABLA 1. DEFINICIÓN ROLES	2
HISTORIAS DE USUARIO.....	2
TABLA 2. DEFINICIÓN HISTORIAS DE USUARIO	2
DEFINICIONES.....	3
TABLA 3. DEFINICIONES	3
JUSTIFICACION USO TECNOLOGIA BLOCKCHAIN PARA RESOLVER EL ALCANCE - DAPP	3
<i>Ventajas</i>	4
<i>Desventajas</i>	4
<i>Smart Contracts</i>	4
ANALISIS Y MODELO PROPUESTO – DAPP	5
DIAGRAMA DE DESPLIEGUE – DAPP	5
FIGURA 1. DIAGRAMA DE DESPLIEGUE – DAPP.....	5
DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 1	5
FIGURA 2. DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 1	5
DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 2	6
FIGURA 3. DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 2	6
DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 3	6
FIGURA 4. DIAGRAMA DE SECUENCIA – HISTORIA DE USUARIO 3	6
DIAGRAMA DE CLASES.....	7
FIGURA 5. DIAGRAMA DE CLASES FRONTEND	7
FIGURA 5. DIAGRAMA DE CLASES BACKEND	7
INSTRUCCIONES DE DESPLIEGUE – DAPP	8
TABLA 4. INSTRUCCIONES DE DESPLIEGUE DAPP	8
MANUAL DE USUARIO DAPP	17
MANUAL DE USUARIO - EXPLORADOR WEB	17
MANUAL DE USUARIO - HISTORIA DE USUARIO 1	18
MANUAL DE USUARIO - HISTORIA DE USUARIO 2	19
MANUAL DE USUARIO - HISTORIA DE USUARIO 3	20
PRUEBAS – DAPP.....	22
TABLA 5.1 PRUEBAS DAPP – OPENZEPOLIN CONTRATO ERC20 – ERC721	22
TABLA 6.1 PRUEBAS DAPP – ANÁLISIS MYTHX CONTRATO DAPP	22
TABLA 6.2 PRUEBAS DAPP – ANÁLISIS MYTHX CONTRATO ERC20	24
TABLA 6.3 PRUEBAS DAPP – ANÁLISIS MYTHX CONTRATO ERC721	25
TABLA 7. PRUEBAS DAPP - TRUFFLE.....	25
TABLA 8. PRUEBAS DAPP – SOLIDITY	25
CONCLUSIONES	29

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

ALCANCE SISTEMA DE FUMIGACIÓN

El alcance del Sistema de Fumigación esta definido en el archivo "block05trabfinal.docx" publicado por UNIR en el aula Trabajo Final de Experto (BLO).

De acuerdo al conocimiento adquirido durante el curso Experto Universitario en Desarrollo de Aplicaciones Blockchain (BLO) ofrecido por UNIR a continuación describo el alcance desagregado en roles, historias de usuario y restricciones. El Sistema de Fumigación en adelante se referencia en este documento y documentos de referencia como dapp fumigación o dapp.

Roles

Tabla 1. Definición Roles

#	Rol	Descripción
1	Customer	El rol Customer es el solicitante del servicio de fumigación en dapp fumigación, además de adicionar los datos de configuración de las parcelas
2	Provider	El rol Provider es el encargado empresa de drones de gestionar las solicitudes de servicios de fumigación, además de adicionar los datos de configuración de los drones

Historias de Usuario

Tabla 2. Definición Historias de Usuario

#	Rol	Descripción
1	Como Customer quiero registrar en dapp fumigación las características de las parcelas que soy responsable	<p>El Customer registra el nombre parcela, altura mínima y máxima de fumigación, y selecciona el pesticida para la fumigación</p> <p>El dapp fumigación genera el id parcela. Inicia en 1 con incrementos de 1 por parcela registrada</p> <p>El registro de la parcela debe realizarse con el token ERC721</p>
2	Como Customer quiero solicitar en dapp fumigación el servicio de fumigación en parcelas con drones	<p>El Customer registra la parcela, luego del registro anterior dapp fumigación asigna un dron disponible, la altura minina y máxima del dron para la fumigación y el precio del servicio de fumigación.</p> <p>Si el Customer está de acuerdo con el servicio de fumigación procede a realizar el pago</p> <p>El registro del pago debe realizarse con el token ERC20</p>
3	Como Provider quiero registrar en dapp fumigación las características de los drones	<p>El Provider registra el nombre dron, altura mínima y máxima de fumigación, selecciona el pesticida para la fumigación</p> <p>El dapp fumigación genera el precio del servicio, y registra el dron como activo para uso (true)</p> <p>El dapp fumigación genera id dron único. El id dron inicia en 1 con incrementos de 1 por dron registrado</p> <p>El registro del dron debe realizarse con el token ERC721</p>

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Definiciones

Tabla 3. Definiciones

#	Descripción
1	La seguridad de dapp fumigación se realiza sobre los contratos. La seguridad del frontend y backend de dapp fumigación no hace parte del alcance TFE. Ni es EL proposito del EXPERTO UNIVERSITARIO EN DESARROLLO DE APLICACIONES BLOCKCHAIN (BLO)
2	La lista de pesticidas no tiene formulario de registro en dapp fumigación. La lista de pesticida pertenece a las parametrizaciones dapp fumigación
3	El Rol Provider utiliza los siguientes sinónimos en el archivo index.js addressProviderDAPP: Cuenta0-Owner relacionada con el grupo de contratos de la carpeta SC_DAPP addressProviderOwnerDron: Cuenta1-Owner relacionada con el grupo de contratos de la carpeta SC_ERC721_RECORD para drones addressProviderOwnerLand: Cuenta2-Owner relacionada con el grupo de contratos de la carpeta SC_ERC721_RECORD para parcelas addressProviderOwnerERC20: Cuenta3-Owner relacionada con el grupo de contratos de la carpeta SC_ERC20_PAY para la asignación de tokens addressProviderOperatorDron: Cuenta4-Operator relacionada con el grupo de contratos de la carpeta SC_ERC721_RECORD para drones addressProviderServicePaid: Cuenta6-Operator relacionada con el grupo de contratos de la carpeta SC_ERC20_PAY para recibir el pago del servicio de drones
4	El Rol Costumer utiliza los siguientes sinónimos en el archivo index.js addressCustomerOperator: Cuenta5 relacionada con el grupo de contratos de la carpeta SC_DAPP para registrar servicios de fumigación, SC_ERC721_RECORD para registrar parcelas, y SC_ERC20_PAY para recibir tokens, y realizar pagos de servicios con tokens
5	La entrega Dapp está desplegada en Máquina Virtual Alastria Telsius
6	La Máquina Virtual Alastria Telsius se realiza despliegue mediante Remix
7	La Máquina Virtual Alastria Telsius debe tener instalado Python para trabajar en localhost.
8	Dapp cuando se inicializa o refresca realiza la ejecución automática de métodos necesarios para el funcionamiento del alcance. Describo ejecución automática; <ul style="list-style-type: none"> • Publica la versión web3 • Publica las cuentas de los contratos y cuentas de usuario provider - customer • Asigna tokens a la cuenta customer • Carga en memoria id de dron, y parcela

JUSTIFICACION USO TECNOLOGIA BLOCKCHAIN PARA RESOLVER EL ALCANCE - DAPP

los principales casos de uso y justificación de utilizar smart contracts basado en la tecnología blockchain para la solución de fumigación con drones del caso propuesto para el TFE.

Registros: Permitirá que los diferentes clientes puedan obtener sus claves con las que podrán realizar los pagos para la fumigación de sus parcelas sin intermediarios y de forma descentralizada, evitando

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

servicios de terceros donde almacenan la identidad/información en una base de datos centralizada con un coste y con posibilidades de ataques malintencionados vulnerabilizando los datos del usuario. Esta solución permite mantener los datos cifrados y seguros.

Transacciones: Permitirá que el cliente, tras haber cumplido las condiciones y requisitos para la fumigación, que los contratos inteligentes e generen transacciones para la creación/asignación del dron de la parcela y la fumigación con las condiciones indicadas por el cliente de forma automática.

Trazabilidad: Permitirá la trazabilidad de las diferentes fumigaciones almacenando de forma segura y transparente las transacciones de las direcciones de los clientes/identificadores de los drones/identificadores de las parcelas que serán únicos para cada acción de fumigar almacenándose en la cadena de bloques.

Inventario: Permitirá obtener información sobre los diferentes pesticidas que se usan para tener estadísticas reales y poder ofrecer al cliente mejoras del servicio.

Ventajas

Precisión en la ejecución de la fumigación.

Transparencia con el cliente.

Comunicación clara.

Velocidad del servicio de fumigación.

Seguridad del servicio de fumigación.

Eficiencia energética y de recursos.

Almacenamiento y respaldo del servicio de fumigación.

Ahorro en costes de infraestructura y servicios de terceros.

Confianza y satisfacción del servicio de fumigación con el cliente.

Desventajas

Confidencialidad y privacidad.

Errores en el código produciendo vulnerabilidades/fallas del servicio de fumigación.

Información errónea en la cadena tras el inicio del servicio.

Malicias frente a hackers malintencionados.

Smart Contracts

Contratos para el estándar ERC20 que crea el token de pago para el intercambio del servicio de fumigación.

Contratos para la generación de drones ERC721 token de servicio no fungible con las características propuestas en la descripción I del TFE.

Contratos para la generación de parcelas ERC721 token de servicio con las características propuestas en la descripción II del TFE.

Contrato DAPPP para las funcionalidades de fumigación con las características propuestas en la descripción III del TFE.

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

ANALISIS Y MODELO PROPUESTO – DAPP

Diagrama de Despliegue – Dapp

Figura 1. Diagrama de Despliegue – Dapp

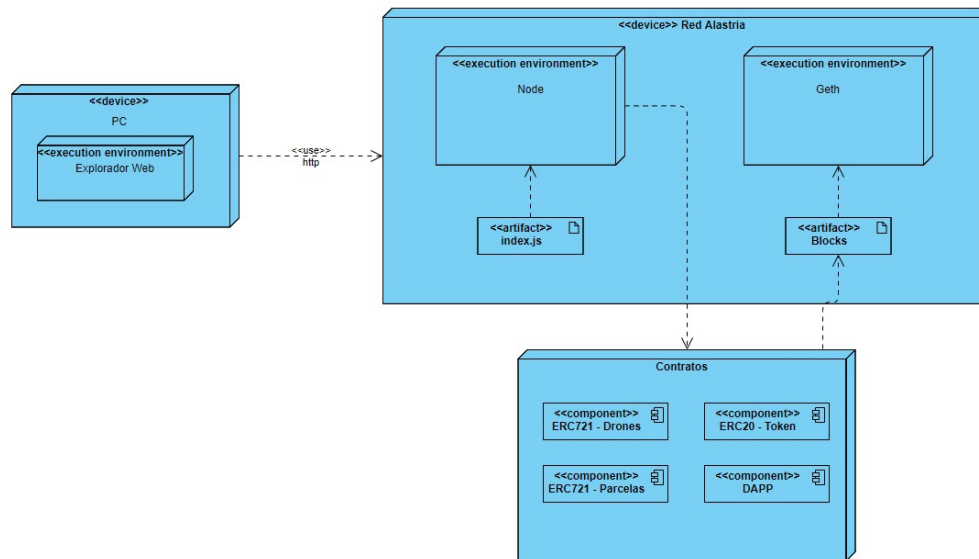
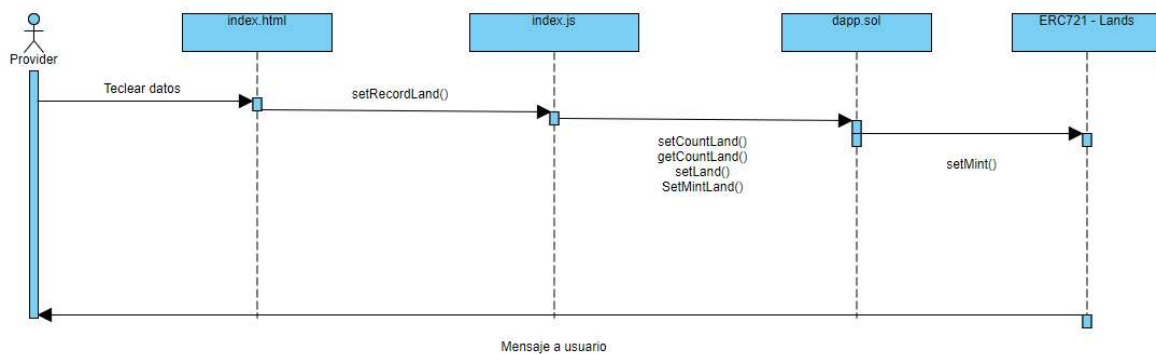


Diagrama de Secuencia – Historia de Usuario 1

Figura 2. Diagrama de Secuencia – Historia de Usuario 1



Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Diagrama de Secuencia – Historia de Usuario 2

Figura 3. Diagrama de Secuencia – Historia de Usuario 2

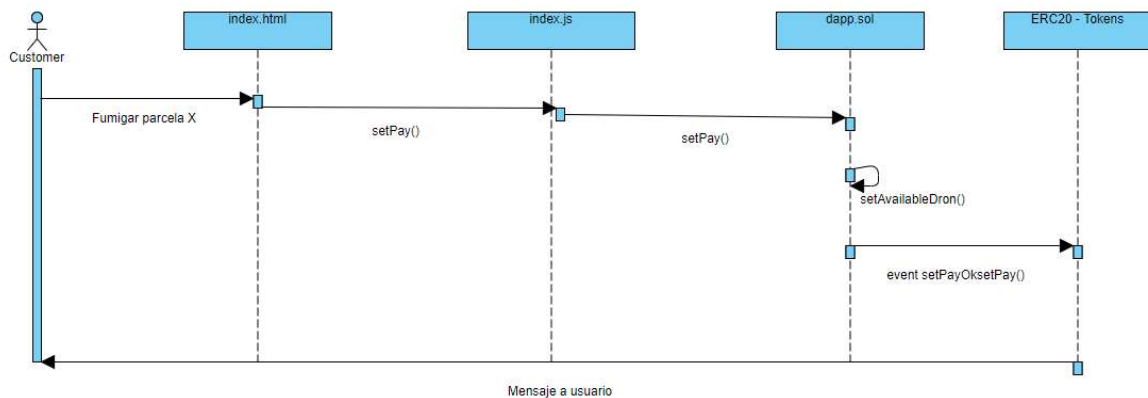
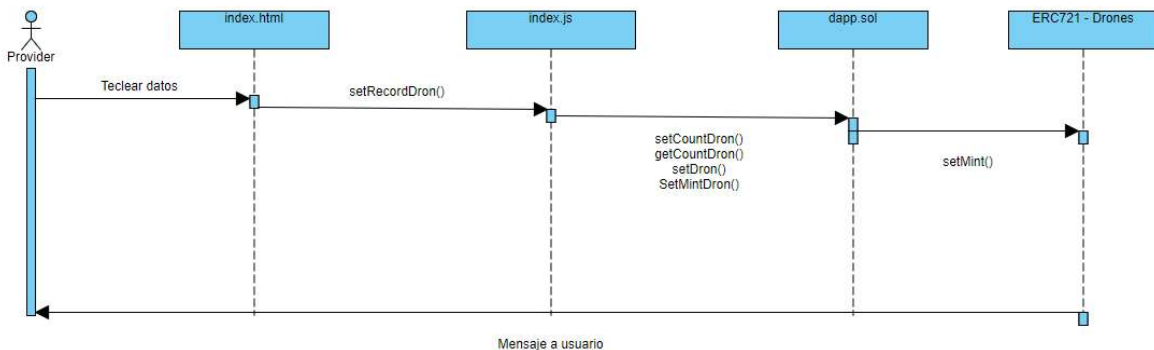


Diagrama de Secuencia – Historia de Usuario 3

Figura 4. Diagrama de Secuencia – Historia de Usuario 3



Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Diagrama de Clases

Figura 5. Diagrama de Clases Frontend

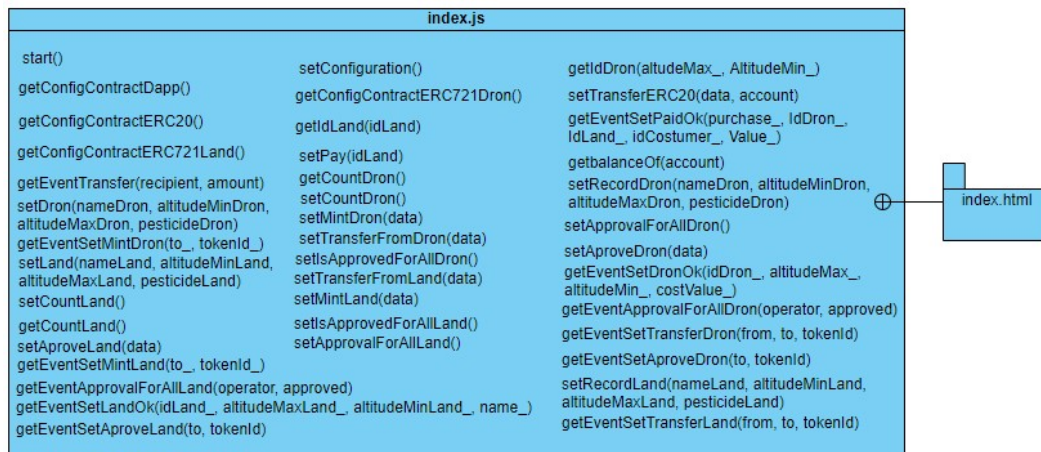
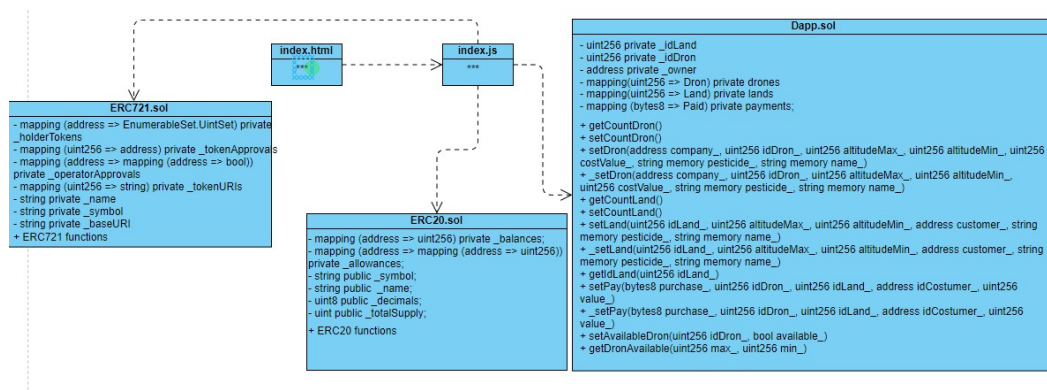


Figura 5. Diagrama de Clases Backend

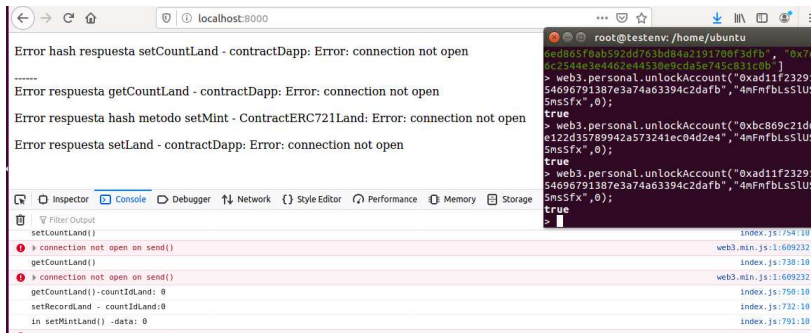


Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

INSTRUCCIONES DE DESPLIEGUE – DAPP

Tabla 4. Instrucciones de Despliegue DAPP

#	Descripción
1	Despliegue de la DAPP en la Tesnet de Alastria con Truffle y despliegue en la red oficial de Alastria Telsius en el nodo de UNIR con Remix.



Despliegue de la DAPP en la Tesnet de Alastria con Truffle

Se usa como SO el Ubuntu proporcionado por Unir de la máquina virtual de Telsius.

Descargar el repositorio test-environment de Alastria. Para ello se utilizará el comando git, desde el directorio personal.

```
git clone https://github.com/alastria/test-environment.git
```

Ejecutar los siguientes comandos de instalación.

Nota: el penúltimo comando puede tardar bastante, en algunos casos un tiempo cercano a los 10 minutos, ya que instala numerosas dependencias.

```
cd test-environment/infrastructure/testnet/
```

```
git clone https://github.com/alastria/alastria-node.git
cd alastria-node/
git checkout develop
```

```
cd ..
sudo -H $PWD/alastria-node/scripts/bootstrap.sh
rm -rf alastria-node
```


Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Comprobar la instalación de las dependencias principales, mostrando la versión de cada dependencia. Si no ocurre un error al mostrar las dependencias, estas estarán bien instaladas.

```
go version
geth version
```

```
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/
testnet# go version
go version go1.9.5 linux/amd64
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/
testnet# geth version
Geth
Version: 1.7.2-stable
Git Commit: 94e1e31eb6a97e08dff4e44a8695dab1252ca3bc
Quorum Version: 2.0.2-Alastria
Architecture: amd64
Network Id: 1
Go Version: go1.9.5
Operating System: linux
GOPATH=/root/alastria/workspace
GOROOT=/usr/local/go
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/
testnet#
```

```
constellation-node --version
```

```
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/
testnet# constellation-node --version
Constellation Node 0.3.2
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/
testnet#
```

Instalar una versión de node posterior a la 8.0. Para ello, se debe actualizar la lista de repositorios de Ubuntu y luego instalar Node. Es necesario comprobar la versión de Node ejecutando el comando `node --version`.

```
sudo apt-get remove nodejs
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash
sudo apt-get install -y nodejs
node --version
```

```
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
node --version
v8.17.0
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
```

Levantar red de Alastria, ejecutar comando desde `test-environment/infrastructure/testnet`
`sudo ./bin/start_network.sh clean 1 2`
`ps -a`

```
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
ps -a
PID TTY          TIME CMD
1884 pts/6        00:00:00 sudo
1885 pts/6        00:00:00 bash
25338 pts/6        00:00:01 geth
25346 pts/6        00:01:09 constellation-n
25591 pts/6        00:00:01 geth
25602 pts/6        00:00:18 constellation-n
25801 pts/6        00:00:00 geth
25815 pts/6        00:00:00 ps
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
```

Comandos para detener la red:

```
sudo ./bin/stop_ethstats.sh
sudo ./bin/stop_network.sh
ps -a
```

```
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
ps -a
PID TTY          TIME CMD
1884 pts/6        00:00:00 sudo
1885 pts/6        00:00:00 bash
25959 pts/6        00:00:00 ps
root@alastria-VirtualBox:/home/alastria/test-environment/infrastructure/testnet#
```

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Instalar Truffle. Verificar la instalación de Truffle con el comando `truffle version`. En este caso se ha instalado una versión superior.

```
Truffle v5.1.34-next.0 (core: 5.1.34-next.0)
Node v8.17.0
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# npm install -g truffle@next
```

Crear nuevo directorio donde ejecutaremos el `truffle init` para la preparación del proyecto:

```
mkdir TFE3
cd TFE3
truffle init
ls -lrta
```

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# ls -lrta
total 44
drwxr-xr-x 2 root root 4096 Feb 19 23:05 test
drwxr-xr-x 3 root root 4096 Feb 24 17:53 build
-rw-r--r-- 1 root root 158 Feb 24 20:42 3_migration_dapp.js
-rw-r--r-- 1 root root 3910 Feb 24 20:43 truffle-config.js
drwxr-xr-x 2 root root 4096 Feb 28 19:13 contracts
-rw-r--r-- 1 root root 198 Feb 28 19:19 2_migration_erc20.js
drwxr-xr-x 6 ubuntu ubuntu 4096 Feb 28 19:21 ..
-rw-r--r-- 1 root root 193 Feb 28 19:22 4_migration_erc721.js
-rw-r--r-- 1 ubuntu ubuntu 196 Mar 8 17:27 5_migration_erc721.js
drwxr-xr-x 6 ubuntu ubuntu 4096 Mar 8 17:27 .
drwxr-xr-x 2 root root 4096 Mar 8 17:28 migrations
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#
```

Los ficheros de migration los comentamos más adelante.

Creación del Smart Contract.

Añadir los Smart contracts del TFE en la carpeta de contracts.

Consultar y desbloquear una dirección Ethereum existente en un nodo de la red local.

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3/contracts# ls -lrta
total 72
-rw-r--r-- 1 root root 419 Feb 19 23:05 Migrations.sol
-rw-r--r-- 1 root root 8312 Feb 28 19:13 DAPP.sol
-rw-r--r-- 1 root root 9083 Feb 28 19:13 ERC20.sol
-rw-r--r-- 1 root root 33915 Feb 28 19:13 ERC721.sol
drwxr-xr-x 2 root root 4096 Feb 28 19:13 .
drwxr-xr-x 6 ubuntu ubuntu 4096 Mar 8 17:27 ..
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3/contracts#
```

Aquí deberían ya tener la red Alastria levantada si no lo está ir al directorio siguiente y ejecutar el comando siguiente:

```
cd /test-environment/infrastructure/testnet/
sudo ./bin/start_network.sh clean 1 2
```

Para conectarnos a uno de los nodos de la red local y utilizar su consola JavaScript, invocamos el siguiente comando (desde el directorio `~/test-environment/infrastructure/testnet/`).

```
sudo geth attach ipc:network/general1/geth.ipc
```

Nota: Aquí pueden cambiar el `general1` por el `general2` para desbloquear una cuenta diferente y solo se deberá tener en cuenta el puerto en el config de truffle que veremos ahora.

Se mostrará el mensaje de inicio de la consola JavaScript, seguido de su prompt, tal y como se ve en la siguiente imagen:

```
ubuntu@testenv:~/test-environment/infrastructure/testnet$ sudo geth attach ipc:network/general2/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/general2/v1.8.12-stable-f64dbb25/linux-amd64/go1.9.5
coinbase: 0xe0fd8fdc41dd71e34688d8dfbac913a694580639
at block: 474 (Mon, 08 Mar 2021 17:33:21 CET)
datadir: /home/ubuntu/test-environment/infrastructure/testnet/network/general2
modules: admin:1.0 debug:1.0 eth:1.0 istanbul:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0
>
```

Una vez en la consola, se debe ejecutar el siguiente comando para conocer las cuentas existentes en el nodo:

```
web3.eth.accounts
```

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

El output mostrado será una cuenta que debemos desbloquear y guardarla para informarla en el config de Truffle.

```
web3.personal.unlockAccount(web3.eth.accounts[0], "Passw0rd",0);
```

El comando consiste en una función JavaScript que recibe como parámetros la cuenta que se va a desbloquear, la contraseña y el número de segundos que estará desbloqueada. Si se pasa un 0 en el último parámetro, se desbloquea la cuenta de forma indefinida.

Para salir de la consola, basta con escribir Ctrl + D. Pero puede quedarse abierta sin problemas

```
> web3.eth.accounts
[ "0x0e596199ea5c6d3cbc713183e7514be022a19385" ]
>
> web3.personal.unlockAccount(web3.eth.accounts[0], "Passw0rd",0);
true
> █
```

Configurar conexión con nodo de la red local. En la carpeta *TFE3*, en el archivo *truffle-config.js* debería quedar de la siguiente forma:

Observar que el puerto es 22002 ya que usé *general2* si usan *general1* pues 22001.

```
networks: {
  development: {
    host: "127.0.0.1",      // Localhost (default: none)
    port: 22002,           // Standard Ethereum port (default: none)
    network_id: "*",       // Any network (default: none)
    gas: 0,
    gasPrice: 0,
    from: "0x0e596199ea5c6d3cbc713183e7514be022a19385"
  },
  // Another network with more advanced options...
  // advanced: {
  //   ...
  // }
```

También se debe hardcodear la entrada de la versión de compilación del siguiente modo:

```
// Configure your compilers
compilers: {
  solc: {
    version: "0.5.0",      // Fetch exact version from solc-bin (default: truffle's version)
    // docker: true,        // Use "0.5.1" you've installed locally with docker (default: false)
    settings: {             // See the solidity docs for advice about optimization and evmVersion
      optimizer: {
        enabled: false,
        runs: 200
      },
      evmVersion: "byzantium"
    }
  }
}
```

Tras varias pruebas se tuvo que bajar la versión a 0.5.0 para la compatibilidad en la red de Alastria.

Configurar el despliegue del Smart Contract.

Se debe crear un fichero de despliegue por cada contrato que se quiera desplegar y dejarlo en *TFE3* y en *TFE/migrations*.

ERC20:

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# more 2_migration_erc20.js
const ERC20 = artifacts.require("ERC20"); //Cargar Smart Contract
module.exports = function(deployer) {
  deployer.deploy(ERC20, "FMG", "FUMIGATOKEN", 1000000, 0); // Desplegar Smart Contract
};
```

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#
```

ERC721 DRON:

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# more 4_migration_erc721.js
const ERC721 = artifacts.require("ERC721"); //Cargar Smart Contract
module.exports = function(deployer) {
  deployer.deploy(ERC721, "DRON", "DRN", "www.ymi.es"); // Desplegar Smart Contract
};
```

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# █
```

ERC721 PARCELA:

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# more 5_migration_erc721.js
const ERC721 = artifacts.require("ERC721"); //Cargar Smart Contract
module.exports = function(deployer) {
  deployer.deploy(ERC721, "PARCELA", "PRC", "www.yml.es"); // Desplegar Smart Contract
};
```

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#
```

DAPP:

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# more 3_migration_dapp.js
const DAPP = artifacts.require("DAPP"); //Cargar Smart Contract
module.exports = function(deployer) {
  deployer.deploy(DAPP); // Desplegar Smart Contract
};
```

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#
```

Este fichero nos ayudará a la distribución y el orden de los despliegues de los diferentes contratos.

Para desplegar el ERC20 y los ERC721 se deben pasar las variables como muestran en las imágenes.

También se tiene que tener en cuenta que la variable del contrato tiene que llamarse igual que el nombre del contrato definido en .sol.

Finalmente ejecutar el comando truffle compile.

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# truffle compile
You can improve web3's performance when running Node.js versions older than 10.5.0 by installing the (deprecated) script package in your project.

Compiling your contracts...
=====
> Compiling ./contracts/DAPP.sol
> Compiling ./contracts/ERC20.sol
> Compiling ./contracts/ERC721.sol
> Compiling ./contracts/Migrations.sol
> Artifacts written to /home/ubuntu/Desktop/ALASTRIA/TFE3/build/contracts
> Compiled successfully using:
   - solc: 0.5.0+commit.1d4f595a.Emscripten.clang
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#
```

Para desplegar los contratos en la testnet de Alastria ejecutar comando truffle migrate

```
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# truffle migrate
You can improve web3's performance when running Node.js versions older than 10.5.0 by installing the (deprecated) script package in your project.

Compiling your contracts...
=====
> Everything is up to date, there is nothing to compile.

Starting migrations...
=====
> Network name: 'development'
> Network id: 953153391
> Block gas limit: 1153385145 (0x44b42b9)

1_initial_migration.js
=====
Deploying 'Migrations'
-----
> transaction hash: 0xbcd990a29e8e2c8b95dd73860e1302a5bdeb1957dce9111208f9d3027a271
> Blocks: 0
> contract address: 0x345464238754e5f2408c78a02a2fe085729cf11
> block number: 1481
> block timestamp: 1615222268
> account: 0xd159d199EA5c6D3CbC713183E75148e022A19385
> balance: 0
> gas used: 237773 (0x3a0cd)
> gas price: 0 gwei
> value sent: 0 ETH
> total cost: 0 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0 ETH

2_migration_erc20.js
=====
Deploying 'ERC20'
-----
> transaction hash: 0x1e44b21fd3d2fca1ae89926e073d5083ce5868fe19f08f99b4ebc3566cfe8fa5
> Blocks: 0
> contract address: 0xc09806c757a590d17bc67a16403505EEF775F9
> block number: 1486
> block timestamp: 1615222213
> account: 0x0E596199EA5c6D3CbC713183E75148e022A19385
> balance: 0
> gas used: 3533291 (0x17656b)
> gas price: 0 gwei
> value sent: 0 ETH
> total cost: 0 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0 ETH

3_migration_dapp.js
=====
Deploying 'DAPP'
-----
> transaction hash: 0xcd36240b956d1513a522b33b26b05b0d1bfa9a740c4a5741d20863ddc2d753
> Blocks: 0
> contract address: 0x8c3fC8EF2E99A1c86D7Abfa5ee7d9a4b25Cb8C00
> block number: 1490
```

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

```

> contract address: 0x8c3fC0E72E99A1c8007Abfa5ee7d9a4b25Cb8C06
> block number: 1490
> block timestamp: 1615222217
> account: 0x0E596199EA5c603CbC713183E75148e022A19385
> balance: 0
> gas used: 1330793 (0x144e69)
> gas price: 0 gwei
> value sent: 0 ETH
> total cost: 0 ETH

> Saving migration to chain.
> Saving artifacts
> Total cost: 0 ETH

4_migration_erc721.js
=====
Deploying 'ERC721'
> transaction hash: 0xe918c3a66830aa8554191b65d7d239ed573c59f415fe3d73021b1c4a2be71166
> Blocks: 0
> Contract address: 0x85c707245FEDf728c13Abf421c18b085e1031cbe
> block number: 1495
> block timestamp: 1615222222
> account: 0x0E596199EA5c603CbC713183E75148e022A19385
> balance: 0
> gas used: 1912010 (0x1d2cca)
> gas price: 0 gwei
> value sent: 0 ETH
> total cost: 0 ETH

> Saving migration to chain.
> Saving artifacts
> Saving migration to chain.
> Saving artifacts
> Total cost: 0 ETH

5_migration_erc721.js
=====
Replacing 'ERC721'
> transaction hash: 0xba4e1ace7807aa1b8e035ca3887f94cb1a63de29286320353b908bf84ee86239
> Blocks: 0
> Contract address: 0x591f49d4992138d3e33bc0362286988ee1c85289
> block number: 1499
> block timestamp: 1615222226
> account: 0x0E596199EA5c603CbC713183E75148e022A19385
> balance: 0
> gas used: 1912202 (0x1d2d8a)
> gas price: 0 gwei
> value sent: 0 ETH
> total cost: 0 ETH

> Saving migration to chain.
> Saving artifacts
> Total cost: 0 ETH

Summary
=====
> Total deployments: 5
> Final cost: 0 ETH

root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#

```

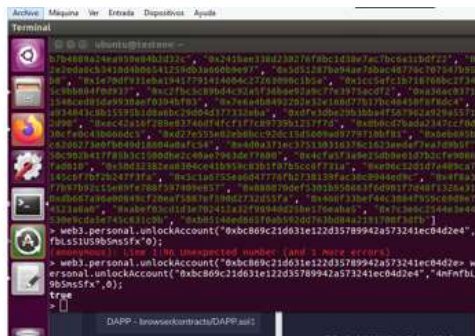

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Despliegue en la red oficial de Alastria Telsius en el nodo de UNIR con Remix.

Se debe realizar desde la VDI que nos ofrece UNIR ya que está dentro de la red y tiene acceso al nodo.

Consultar y desbloquear una cuenta en el nodo de la UNIR. Para realizar el despliegue del contrato en el nodo de la UNIR, debemos elegir una cuenta existente en dicho nodo y desbloquearla. El nodo acepta peticiones en el endpoint <http://10.141.8.11:8545>. La consulta de cuentas se realiza de la misma forma que una consulta a un nodo local: conectándonos a la consola JavaScript del nodo y pidiendo las cuentas.

```
sudo geth attach http://10.141.8.11:8545
web3.eth.accounts
```



Se mostrarán numerosas cuentas. En nuestro caso, utilizaremos 7. Procedemos a desbloquear dichas cuentas utilizando su contraseña, con la siguiente instrucción:

```
web3.personal.unlockAccount("0xbc869c21d631e122d35789942a573241ec04d2e4","4mFmf
bLsSIUS9b5msSfx",0);
```

En el nodo de UNIR las cuentas que pueden ser utilizadas por los alumnos son las siguientes:

- 0xbc869c21d631e122d35789942a573241ec04d2e4
- 0xad11f232919a54696791387e3a74a63394c2dafb
- 0x35ad6e72cb2ec714b80154b796c7835f97053d3e
- 0xa3fef7d78a13f6b6bb1cf60c20bb854c7ed2d8d17
- 0x09702705ebd2c925b3c56662e4982ebec8bce7d
- 0xc35fdb9f41a34e998f4d094922e190b4c6fd8e32
- 0x11c5395d602289b7407ceebb4fdd5707772c6ae
- 0x48d095879b4ebde16b74129c4ec9d3d78d984b80
- 0xea66394b0ecc0175b7b4889a24ea959e84b2d32c
- 0x241bae338d230276f8bc1d38e7ac7bc6a1cbdf22

Todas tienen como contraseña 4mFmfLsSIUS9b5msSfx

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yurriel	

Abrir el editor online Remix. En el navegador, debemos escribir la URL <http://remix.ethereum.org> . La página debemos abrirla con http.

Crear un nuevos Smarts Contracts.

Crear 3 nuevos .sol en la pestaña de archivos de Remix con el nombre de cada Smart Contract.

- ERC20
- ERC721
- DAPP

Escribir el código de los Smarts Contracts.

Realizar copy & paste y unificar los smarts contracts en 3 contratos ERC20, ERC721 y DAPP

Compilar los Smarts Contracts.

La versión de remix debe ajustarse a 0.5.0

Configurar conexión con el nodo de la UNIR. En el desplegable «*Environment*», elegir la opción «Web 3 Provider».

Al elegir la opción, se abrirá un cuadro de diálogo que nos preguntará si estamos seguros de que queremos conectarnos a un nodo, a lo que debemos responder «OK». Luego, nos pedirá que escribamos el endpoint del nodo de la UNIR, que es <http://10.141.8.11:8545> . Finalmente pulsamos «OK».

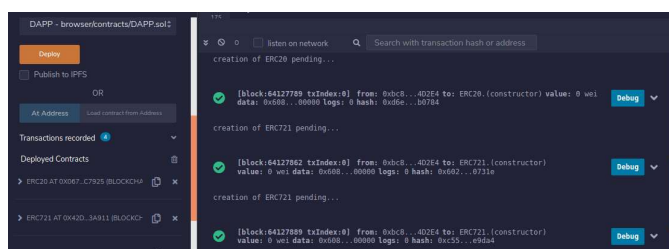
Seleccionar la cuenta para realizar el despliegue.

Se elige una de las cuentas desbloqueadas anteriormente para realizar los 4 despliegues desde Remix.

```
> web3.personal.unlockAccount("0xbc809c21d631e122d35789942a573241ec04d2e", w
ersonal.unlockAccount("0xbc809c21d631e122d35789942a573241ec04d2e4", "4nFnfbl
9b5nsSfx",0);
true
> ||
```

Desplegar los contratos.

Se despliegan los contratos desde la cuenta desbloqueada.
 ERC20 → Indicar totalSupply, nombre, símbolo y decimales.
 ERC721-DRONE → Indicar nombre, símbolo y uri.
 ERC721-PARCELA → Indicar nombre, símbolo y uri.
 DAPP



Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Levantar python para interactuar con el html.

```

root@testenv:/home/ubuntu/Desktop/TFE/OneDrive_1_8-3-20
tp.server
No command 'pyhton3' found, did you mean:
  Command 'python3' from package 'python3-minimal' (main)
python3: command not found
root@testenv:/home/ubuntu/Desktop/TFE/OneDrive_1_8-3-20
-m http.server
sudo: pyhton3: command not found
root@testenv:/home/ubuntu/Desktop/TFE/OneDrive_1_8-3-20
-m http.server
Serving HTTP on 0.0.0.0 port 8080 ...
127.0.0.1 - - [08/Mar/2021 19:00:55] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:00:56] "GET /35/index.js" 200 -
127.0.0.1 - - [08/Mar/2021 19:00:56] "GET /favicon.ico" 200 -
127.0.0.1 - - [08/Mar/2021 19:02:53] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:02:52] "GET /35/index.js" 200 -
127.0.0.1 - - [08/Mar/2021 19:06:06] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:06:33] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:07:22] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:08:56] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:30:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Mar/2021 19:31:32] "GET / HTTP/1.1" 200 -

```


Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

MANUAL DE USUARIO DAPP

Manual de Usuario - Explorador Web

A continuación describo el formulario web con sus métodos relacionados por rol.

Cuando se inicializa el formulario web deben estar configurados los contratos en Alastria con sus cuentas de usuarios.

Cuando se inicializa el formulario web la cuenta del CLIENTE recibe 60 tokens, además por cada recarga de formulario web la cuenta del CLIENTE recibe 60 tokens. En primera parte del formulario web se describen las cuentas del PROVEEDOR y el CLIENTE con sus saldos de tokens. Cada registro de servicio de fumigación exitoso, hay movimiento de tokens entre las cuentas.

La primera parte METODOS CLIENTE permite al rol customer adicionar parcelas, y solicitar servicios de fumigación.

La segunda parte METODOS EMPRESA FUMIGACIÓN CON DRONES permite al rol customer adicionar drones.

Total Token: 1000000 FDR

Saldo cuenta principal Proveedor: 999940
Saldo cuenta recaudo Proveedor: 0
Saldo cuenta cliente: 60

METODOS CLIENTE

Registra Parcelas

Propietario 0x21ba82ac484b7858DD0685C102824c6364F5F967
Nombre Parcela
Altura Mínima M
Altura Máxima M
Pesticida

Parcelas

Solicita Servicio Fumigación

Propietario 0x21ba82ac484b7858DD0685C102824c6364F5F967
ID Parcela
Pesticida Parcela
ID Dron
Altura Mínima Dron M
Altura Máxima Dron M
Total a Pagar

Drones Disponibles

METODOS EMPRESA FUMIGACIÓN CON DRONES

Registra Drones

Nombre Dron
Altura Mínima M
Altura Máxima M
Pesticida

Drones

Cuando se inicializa el formulario web se valida la correcta conexión con los contratos, y se envían 60 tokens al CLIENTE

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

```

*****
Address0
0x0000000000000000000000000000000000000000000000000000000000000000
TokenIdDron
TokenIdLand

*****
Version web3: 1.0.0-beta.35
contractERC721Dron: 0x577DA857660E38107a5DEaC8f15E1d21d50DB789
contractERC721Land: 0xFcd48b439C62332eC1Dd7361e55aF3961C3E74
contractERC20: 0x1f375e2A6Db16a1Bbdf03464E3c8CBb6ae50A5E
contractDapp: 0xb96E9529c0D7622b144180B66e45C2FD946B59

addressProviderDAPP: 0x3E313d14cA2bDB8E32c539bb9e7a5808B72E277b
addressProviderOwnerDron: 0x1e4eEd42f115320B5A87Ec61699555ca6b315c7
addressProviderOwnerLand: 0x1965F70D3F9DfEBdAE4dF8a25F74ec51242bD7f3
addressProviderOwnerERC20: 0x435714909bC63B6734bb434C705021561f44F064
addressProviderOperatorDron: 0xdA9A0d615526e7b0618A84540FE4750711e40f
addressProviderServicePaid: 0xC497C3Fe6a4E8811CA7311Bc603be111a6f068fa
addressCustomerOperator: 0x21ba82ac484b7858DD0685C102824c6364F5F967

*****
Configuracion inicial contractERC721Dron - Nombre Token: ERC721 DRON
Configuracion inicial contractERC721Dron - Simbolo Token: DRN

*****
Configuracion inicial contractERC721Land - Nombre Token: ERC721 LAND
Configuracion inicial contractERC721Land - Simbolo Token: LAND

*****
Configuracion inicial contractERC20 - Nombre Token: DAPP FUMIGACION DRONES
Configuracion inicial contractERC20 - Total Token: 1000000
Configuracion inicial contractERC20 - Simbolo Token: FDR
Configuracion inicial contractERC20 - Decimales Token: 0

*****
Respuesta hash metodo transfer - ContractERC20: 0x8307fb9176fc6a504420af10e36556d3e765c733343da78b48cb824cc94fb9e

*****
Respuesta metodo getCountDron - contractDapp: 0x64fa8462cb634aa136881775a469b543bb422b80529849a66da906d927ba0fc

*****
Respuesta metodo getCountDron - contractDapp: 1

*****
Respuesta metodo getCountLand - contractDapp: 0x300cb372286520801776244ba8cd8ef32885515c73fa87a4e3b0e921137a1b83

*****
Respuesta metodo getCountLand - contractDapp: 1

*****
Respuesta metodo balanceOf - contractERC20 - customer: 0x21ba82ac484b7858DD0685C102824c6364F5F967 - balance: 60

*****
Respuesta hash metodo approve - ContractERC20: 0x4460f7859c8247c78f5c7386d73aa3c95bfc53194532a1d28c814c6adfd429 - address: 0x21ba82ac484b7858DD0685C102824c6364F5F967

```

Manual de Usuario - Historia de Usuario 1

El cliente registra la parcela(s), y por cada registro exitoso se incrementa la tabla de la de la derecha teniendo en cuenta el filtro por cuenta de gestión cliente. Por configuración el formulario web trabaja solo con una cuenta de cliente.

MÉTODOS CLIENTE

Registra Parcelas

Propietario 0x21ba82ac484b7858DD0685C102824c6364F5F967
Nombre Parcela
Altura Mínima M
Altura Máxima M
Pesticida

****Parcelas****
Cuenta Gestion --- Cod - Nombre --- Altura Maxima - Altura Minima - Pesticida
...6364F5F967 --- 1 --- LAND1 --- 190 --- 110 --- Pesticida.A
...6364F5F967 --- 2 --- LAND2 --- 196 --- 120 --- Pesticida.A
...6364F5F967 --- 3 --- LAND3 --- 196 --- 121 --- Pesticida.B
...6364F5F967 --- 4 --- LAND3 --- 197 --- 121 --- Pesticida.C
...6364F5F967 --- 5 --- LAND5 --- 198 --- 125 --- Pesticida.C
...6364F5F967 --- 6 --- LAND6 --- 150 --- 130 --- Pesticida.A

El formulario web escribe la trazabilidad de la solicitud exitosa o fallida de la petición al contrato después de la última trazabilidad trabajada en el formulario web. A continuación describo ejemplo trazabilidad.

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

```

-----
Respuesta metodo getCountLand - contractDapp: 8

Respuesta hash metodo setMint - ContractERC721Land: 0xc0189879be0ffea065cc19ba07aec15186942c6d917ccb02d6599d5f651b3e38 - tokenId: 7

Respuesta hash metodo setLand - contractDapp: 0xa39ee4446ec7797a166efc84d48c564e8cf2481bd4b81a648a3c0a3664ccf566

Respuesta evento Transfer de setMint - ContractERC721Land - to: 0x1965F70D3F9DfEBdAE4dF8a25F74ec51242bD7f3 - tokenId: 7

Respuesta evento eventSetLandOk de setLand - contractDapp - idLand_ 7 - altitudeMin_ 50 - altitudeMax_ 128 - name_LANDs

-----
Respuesta metodo getCountLand - contractDapp: 8

Respuesta hash metodo transferFrom - ContractERC721Land: 0x74ae8be91d7a7098e6e614ea0864c52779b633601856a82bbd92d079cf2d1554

Respuesta hash metodo setApprovalForAll - ContractERC721Land: 0x43a946cc2a794c6ff3522c9b6e74d59965e2c56c7ed09b982fb43cf5889ab46a

Respuesta evento ApprovalForAll de setApprovalForAll - ContractERC721Land - owner: 0x1965F70D3F9DfEBdAE4dF8a25F74ec51242bD7f3 - approved: true

Respuesta hash metodo approve - ContractERC721Land: 0x7ae4905c4c91ba86ee8b4eaa13cb450b765de540ddb2498fdaa3028b314a965e

Respuesta hash metodo isApprovedForAll - ContractERC721Land: true

```

Manual de Usuario - Historia de Usuario 2

La condición inicial para el correcto funcionamiento es haber ingresado datos en la historia de usuario 1 e historia de usuario 3.

El cliente ingresa el id parcela de acuerdo a los id parcelas de la tabla historia de usuario 1, luego de ingresado el id parcela el formulario web valida en los contratos si hay drones disponibles, y completa los demás datos del servicio de fumigación. La tabla de la derecha teniendo en cuenta el filtro Estado lista los drones disponibles para servicio de fumigación.

Solicita Servicio Fumigación

Propietario 0x21ba82ac484b7858DD0685C102824c6364F5F967
 ID Parcela
 Pesticida Parcela Pesticida.A
 ID Dron 1
 Altura Mínima Dron M 100
 Altura Máxima Dron M 200
Total a Pagar 23

Drones Disponibles

Cod - Nombre --- Altura Máxima - Altura Mínima - Pesticida ---- Estado
 1 ---- DRON1 ----- 200 ----- 100 ----- Pesticida.A -- Disponible
 2 ---- DRON2 ----- 200 ----- 110 ----- Pesticida.A -- Disponible
 3 ---- DRON2 ----- 220 ----- 120 ----- Pesticida.B -- Disponible
 4 ---- DRON4 ----- 200 ----- 130 ----- Pesticida.C -- Disponible
 5 ---- DRON5 ----- 270 ----- 140 ----- Pesticida.C -- Disponible
 6 ---- DRON6 ----- 271 ----- 141 ----- Pesticida.D -- Disponible

El formulario web escribe la trazabilidad de la solicitud exitosa o fallida de la petición al contrato después de la última trazabilidad trabajada en el formulario web. A continuación describo ejemplo trazabilidad.

```

Respuesta metodo getIdLand - contractDapp: Pesticida.A - 190 - 110

Respuesta metodo getIdDron - contractDapp - IDdron: 1

```

Si el cliente esta de acuerdo con el dron asignado da click al botón Realiza Pago. Si el cliente tiene tokens suficientes se registra el servicio de fumigación, y se actualiza la tabla de la derecha marcando los drones disponibles a excepción del dron seleccionado para ejecutar el servicio de fumigación, y se realiza movimiento de tokens entre las cuentas CLIENTE a PROVEEDOR

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Solicita Servicio Fumigación

Propietario 0x21ba82ac484b7858DD0685C102824c6364F5F967
 ID Parcela
 Pesticida Parcela
 ID Dron
 Altura Mínima Dron M
 Altura Máxima Dron M
Total a Pagar:

Drones Disponibles

Cod - Nombre --- Altura Maxima - Altura Minima - Pesticida --- Estado
 2 ---- DRON2 ----- 200 ----- 110 ----- Pesticida.A --- Disponible
 3 ---- DRON2 ----- 220 ----- 120 ----- Pesticida.B --- Disponible
 4 ---- DRON4 ----- 200 ----- 130 ----- Pesticida.C --- Disponible
 5 ---- DRON5 ----- 270 ----- 140 ----- Pesticida.D --- Disponible
 6 ---- DRON6 ----- 271 ----- 141 ----- Pesticida.D --- Disponible
 7 ---- DRON7 ----- 190 ----- 90 ----- Pesticida.B --- Disponible

Pago Exitoso. Orden Servicio: 0x39c54ce24b3fd7bf - Valor Pagado: 23

El formulario web escribe la trazabilidad de la solicitud exitosa o fallida de la petición al contrato después de la última trazabilidad trabajada en el formulario web. A continuación describo ejemplo trazabilidad.

 Respuesta hash metodo setPay - contractDapp: 0x024cc3474a92d599c5c2ddef454bb2b277dc15067fcf4e81ffe9787a1e531745
 Respuesta evento getPastEvents de setPay - contractDapp - purchase_: 0x39c54ce24b3fd7bf - value_: 23

 Respuesta metodo getCountDron - contractDapp: 8

 Respuesta hash metodo transferFrom - ContractERC20: 0xda071c13853041c79ceb685f27fb88dcfd462bef2d9b13018e7b388971bc2b4

 Respuesta hash metodo approve - ContractERC20: 0x2883cdc86261ab49a2dfa4d52c955fa5e3313dfd5d7e5bda4cc23590f2fe6bee - address: 0xc497C3Fe6a4E8811CA7311Bc605be111a6f068fa

Manual de Usuario - Historia de Usuario 3

El proveedor registra el dron(es), y por cada registro exitoso se incrementa la tabla de la de la derecha teniendo en cuenta el filtro por cuenta de gestión cliente. Por configuración el formulario web trabaja solo con una cuenta de proveedor encargada de administrar los drones.

METODOS EMPRESA FUMIGACIÓN CON DRONES

Registra Drones

Nombre Dron
 Altura Mínima M
 Altura Máxima M
 Pesticida

Drones

Cuenta Gestion ----- Cod - Nombre --- Altura Maxima - Altura Minima - Valor Servicio -- Pesticida -- Estado
 ...750711e40f ----- 1 --- DRON1 ----- 200 ----- 100 ----- 23 ----- Pesticida.A --- true
 ...750711e40f ----- 2 --- DRON2 ----- 200 ----- 110 ----- 23 ----- Pesticida.A --- true
 ...750711e40f ----- 3 --- DRON2 ----- 220 ----- 120 ----- 23 ----- Pesticida.B --- true
 ...750711e40f ----- 4 --- DRON4 ----- 200 ----- 130 ----- 23 ----- Pesticida.C --- true
 ...750711e40f ----- 5 --- DRON5 ----- 270 ----- 140 ----- 23 ----- Pesticida.C --- true

El formulario web escribe la trazabilidad de la solicitud exitosa o fallida de la petición al contrato después de la última trazabilidad trabajada en el formulario web. A continuación describo ejemplo trazabilidad.

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

Respuesta metodo getCountDron - contractDapp: 7

Respuesta hash metodo setCountDron - contractDapp: 0xbd4c58b637a406eebe6ce60b75a670d3162c67cfae1798c065e913df50f8111a

Respuesta hash metodo setMint - ContractERC721Dron: 0x018bf4a3e7a5cbe6b5f7ec8a9755930b6180ae5ee98e8e8d3f1e6779f71c4851 - tokenId: 7

Respuesta hash metodo setDron - contractDapp: 0xdcfbefe839b6ec41d7812770991322e52524777262fe58c0b0753c8e37b006af

Respuesta evento Transfer de setMint - ContractERC721Dron - to: 0x1e4eEcD42f115320B5A87Ec61699555ca6b315c7 - tokenId: 7

Respuesta evento eventSetDronOk de setDron - contractDapp - idDron_: 7 - altitudeMin_ 90 - altitudeMax_ 190

Respuesta metodo getCountDron - contractDapp: 8

Respuesta hash metodo transferFrom - ContractERC721Dron: 0xbf046a141d3c5be3c96fcc38443b04da128b58358f036095dd642ad31bbfa360

Respuesta hash metodo setApprovalForAll - ContractERC721Dron: 0xc65d54b5fa21555faf29463aa26e6bdd4366fa42fbd605c7a1e968f692861dd

Respuesta evento ApprovalForAll de setApprovalForAll - ContractERC721Dron - owner: 0x1e4eEcD42f115320B5A87Ec61699555ca6b315c7 - approved: true

Respuesta hash metodo approve - ContractERC721Dron: 0x6bc2dd099b74cdca464350dfa95ecd61c2312418b00c7ee7cc24a0405cd48ea

Respuesta hash metodo isApprovedForAll - ContractERC721Dron: true

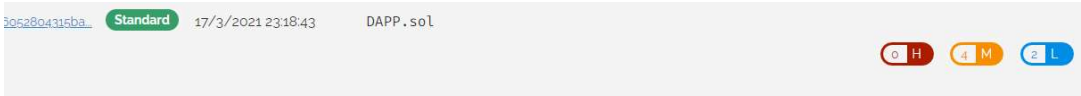
Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yurriel	

PRUEBAS – DAPP

Tabla 5.1 PRUEBAS DAPP – OpenZepelin contrato ERC20 – ERC721

#	Pruebas Proyecto OpenZepelin
1	Los contratos ERC20 y ERC721 estan basados en el proyecto OpenZepelin. Decidimos trabajar sobre este proyecto para aprovechar la seguridad de los contratos ingresadas por OpenZepelin. La referencia de los archivos de test se encuentran en el siguiente link https://github.com/OpenZeppelin/openzeppelin-contracts/tree/master/test/token

Tabla 6.1 PRUEBAS DAPP – Análisis MythX contrato DAPP

#	Análisis standard con la herramienta MythX del contrato DAPP
1	Resumen: 
2	Vulnerabilidades altas detectadas: 0
3	Vulnerabilidades medias detectadas: 4
4	Vulnerabilidades bajas detectadas: 2

Los contratos DAPP_06, ERC20 y ERC721 han sido analizados con la herramienta MythX de ConsenSys. Esta herramienta permite analizar Smart contracts de Ethereum en busca de vulnerabilidades de seguridad como: "Assertions and Property Checking", "Byte-code Safety", "Authorization Controls", "Control Flow", "ERC Standars" y "Solidity Coding Best Practices". Se pueden consultar todas las vulnerabilidades aquí: <https://mythx.io/detectors/>

Vulnerabilidades medias:

Todos con relación a funciones que pueden ser marcadas como external.

MEDIUM Function could be marked as external.
 The function definition of "setDron" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.
 SWC-000

Source file
 localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol
 Locations

```

39 // Dron set
40
41 function setDron(address company, uint256 idDron, uint256 altitudeMax, uint256 altitudeMin, uint256 costValue, string memory pesticide, string memory name) public returns
42 (bool) {
43     require(msg.sender == _owner, "Solo la empresa puede generar drones");
44     setDron(company, idDron, altitudeMax, altitudeMin, costValue, pesticide, name);
45     return true;
46 }
47
48 function setDron(address company, uint256 idDron, uint256 altitudeMax, uint256 altitudeMin, uint256 costValue, string memory pesticide, string memory name) internal returns
49 (bool) {
50     drones[idDron].company = company;
51     drones[idDron].altitudeMax = altitudeMax;
  
```

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

MEDIUM Function could be marked as external.

The function definition of "setLand" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

SWC-000

Source file

localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol

Locations

```

76 // Land setll
77
78 function setLand(uint256 idLand, uint256 altitudMax, uint256 altitudMin, address customer, string memory pesticide, string memory name) public returns (bool) {
79     require(msg.sender == owner, "Solo el propietario de la parcela puede ingresar parcelas");
80     setLand(idLand, altitudMax, altitudMin, customer, pesticide, name);
81     return true;
82 }
83
84 function _setLand(uint256 idLand, uint256 altitudMax, uint256 altitudMin, address customer, string memory pesticide, string memory name) internal returns (bool) {
85     //pone una condicion que valide - como esta cualquier address puede ingresar parcelas
86     lands[idLand].altitudMax = altitudMax;

```

MEDIUM Function could be marked as external.

The function definition of "setPay" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

SWC-000

Source file

localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol

Locations

```

104 mapping (bytes8 => Paid) private payments;
105
106 function setPay(bytes8 purchase, uint256 idDron, uint256 idLand, address idCustomer, uint256 value) public returns (bool) {
107     setPay(purchase, idDron, idLand, idCustomer, value);
108     return true;
109 }
110
111 function _setPay(bytes8 purchase, uint256 idDron, uint256 idLand, address idCustomer, uint256 value) internal returns (bool) {
112     function _setPay(bytes8 purchase, uint256 idDron, uint256 idLand, address idCustomer, uint256 value) internal returns (bool) {
113         //adiciona un require que valide el by, no exista en payments
114         payments[purchase][idDron] = idDron;

```

MEDIUM Function could be marked as external.

The function definition of "getDronAvailable" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

SWC-000

Source file

localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol

Locations

```

126 }
127
128 function getDronAvailable(uint256 max, uint256 min) public view returns (uint256 IDDron, uint256 altitudMax, uint256 altitudMin, uint256 costValue) {
129     //x= LibraryDAPP.hello(a);
130     IDDron = LibraryDAPP.getIdOfDrones(max, min);
131     altitudMax = drones[IDDron].altitudMax;
132     altitudMin = drones[IDDron].altitudMin;
133     costValue = drones[IDDron].costValue;
134 }
135
136 }

```

Vulnerabilidades bajas:

Tenemos el pragma que es mejor marcar uno especifico que marcar un mayor que versión X y una variable declarada que no se utiliza.

LOW A floating pragma is set.

The current pragma Solidity directive is "0.5.0", it is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

SWC-103

Source file

localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol

Locations

```

2 //Sistema de fumigación con drones
3
4 pragma solidity ^0.5.0;
5
6
7 import "./IDAPP_06.sol";

```


Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

LOW Unused state variable "payments".

The state variable "payments" is declared within the contract "DAPP" but its value does not seem to be used anywhere.

SWC-131

Source file

localhost/TFE/INDEX/CONTRACTS/SC_DAPP/DAPP_06.sol

Locations

```

102 //El contrato DAPP implementa un mapping (bytes32 => Paid) paído para almacenar la struct
103 //El grupo de funciones a continuación describen la operación pago del servicio de fumigación
104 mapping (bytes8 => Paid) private payments;
105
106
107 function setPay(bytes8 purchase_, uint256 idBoron_, uint256 idLand_, address idCustomer_, uint256 value_) public returns(bool) {
108     setPay(purchase_, idBoron_, idLand_, idCustomer_, value_);

```

Tabla 6.2 PRUEBAS DAPP – Análisis MythX contrato ERC20

#	Análisis standard con la herramienta MythX del contrato ERC20
1	<p>Resumen:</p> <div> <div>Source File ▾</div> <div>Scan Mode</div> <div>Submitted at ▾</div> <div>Detected Vulnerabilities ▾</div> </div> <div> <div>ERC20.sol</div> <div>Standard</div> <div>14/3/2021 12:21:01</div> <div> <div>0 H</div> <div>12 M</div> <div>1 L</div> </div> </div>
2	Vulnerabilidades altas detectadas: 0
3	Vulnerabilidades medias detectadas: 12
4	Vulnerabilidades bajas detectadas: 1

Vulnerabilidades medias:

Todos con relación a funciones que pueden ser marcadas como external.

MEDIUM Function could be marked as external.

The function definition of "name" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

SWC-000

Source file

localhost/TFE/INDEX/CONTRACTS/SC_ERC20_PAY/IERC20.sol

Locations

```

31
32 /**
33  * @dev Sets 'amount' as the allowance of 'spender' over the caller's tokens.
34  *
35  * Limits an [[Approval]] event.
36  */
37 function approve(address spender, uint256 amount) external returns (bool);
38
39 /**

```

MEDIUM Function could be marked as external.

The function definition of "totalSupply" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

SWC-000

Source file

localhost/TFE/INDEX/CONTRACTS/SC_ERC20_PAY/IERC20.sol

Locations

```

44  * Returns a boolean value indicating whether the operation succeeded.
45  *
46  * Limits a [[Transfer]] event.
47  */
48 function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);
49
50 /**

```


Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

MEDIUM Function could be marked as external.

SWC-000 The function definition of "balanceOf" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

localhost/TFE/INDEX/CONTRACTS/SC_ERC20_PAY/IERC20.sol

Locations

```

40
50 /**
51  * @dev Emitted when 'value' tokens are moved from one account ('from') to
52  * another ('to').
53  *
54  * Note that 'value' may be zero.
55  */
56 event Transfer(address indexed from, address indexed to, uint256 indexed value);

```

Vulnerabilidades bajas:

Tenemos el pragma que es mejor marcar uno específico que marcar un mayor que versión X.

LOW A floating pragma is set.

SWC-103 The current pragma Solidity directive is "0.5.0". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file

localhost/TFE/INDEX/CONTRACTS/SC_ERC20_PAY/IERC20.sol

Locations

```

1 pragma solidity ^0.5.0;
2
3 interface IERC20 {

```

Tabla 6.3 PRUEBAS DAPP – Análisis MythX contrato ERC721

#	Análisis standard con la herramienta MythX del contrato ERC721			
1	Resumen:	Scan Mode	Submitted at	Detected Vulnerabilities
	Source File			
	ERC721.sol	Standard	14/3/2021 12:24:14	<div> <div>H</div> <div>M</div> <div>L</div> </div>
2	Vulnerabilidades altas detectadas: 0			
3	Vulnerabilidades medias detectadas: 0			
4	Vulnerabilidades bajas detectadas: 0			

Tabla 7. PRUEBAS DAPP - TRUFFLE

#	Descripción
1	Revisar test.js

```

root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3# truffle test
You can improve web3's performance when running Node.js versions older than 10.5.0 by installing the (deprecated) script package in your project.
Using network 'development'.

Compiling your contracts...
> Everything is up to date, there is nothing to compile.

Contract: DAPP
  ✓ El constructor se construye con dato 1 (302ms)
  ✓ Se incrementa el contador de drones, y retorna un true (268ms)
  ✓ Se añade un dron y retorna true (489ms)

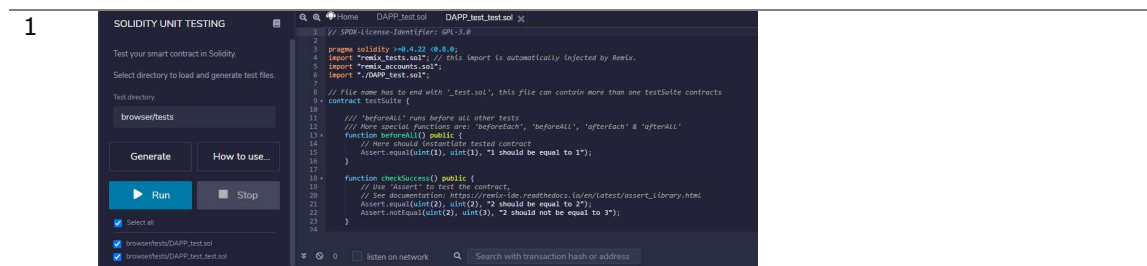
3 passing (2s)
root@testenv:/home/ubuntu/Desktop/ALASTRIA/TFE3#

```

Tabla 8. PRUEBAS DAPP – Solidity

#	Descripción
---	-------------

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	



Vulnerabilidades tras realizar test y análisis:

Gas costs:

Gas requirement of function DAPP.setDron/ DAPP.getDronAvailable/ DAPP.setLand/ DAPP.getIdLand is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 106:8: / Pos: 143:10: / Pos: 159:11: / Pos: 193:10:

Miscellaneous

Constant/View/Pure functions:

DAPP.getDronAvailable(uint256,uint256) : Is constant but potentially should not be.
more

Pos: 193:10:

Similar variable names:

LibraryDAPP.indexOf(mapping(uint256 => struct IDAPP.Dron),uint256,uint256) : Variables have very similar names "max_" and "min_".Pos: 62:22:

LibraryDAPP.indexOf(mapping(uint256 => struct IDAPP.Dron),uint256,uint256) : Variables have very similar names "max_" and "min_".Pos: 63:27:

DAPP.setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 108:28:

DAPP.setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 108:37:

DAPP.setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 108:51:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 112:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 113:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 114:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 115:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 116:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 117:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 118:16:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "_idDron" and "idDron_".Pos: 119:29:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 113:40:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 114:40:

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 119:38:

DAPP._setDron(address,uint256,uint256,uint256,uint256,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 119:52:

DAPP.setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 145:19:

DAPP.setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 145:28:

DAPP.setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 145:42:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 150:17:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 151:17:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 152:17:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 153:17:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 154:17:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "_idLand" and "idLand_".Pos: 155:31:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 150:41:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 151:41:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 155:40:

DAPP._setLand(uint256,uint256,uint256,address,string,string) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 155:54:

DAPP.getIdLand(uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 160:32:

DAPP.getIdLand(uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 161:32:

DAPP.getIdLand(uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 162:30:

DAPP.getIdLand(uint256) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 160:11:

DAPP.getIdLand(uint256) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 161:11:

DAPP.setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 172:39:

DAPP.setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idDron" and "idDron_".Pos: 172:30:

DAPP._setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 180:0:

DAPP._setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idLand" and "idLand_".Pos: 185:42:

DAPP._setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idDron" and "idDron_".Pos: 179:0:

DAPP._setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idDron" and "idDron_".Pos: 183:28:

DAPP._setPay(bytes8,uint256,uint256,address,uint256) : Variables have very similar names "_idDron" and "idDron_".Pos: 185:32:

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

DAPP.setAvailableDron(uint256,bool) : Variables have very similar names "_idDron" and "idDron_".Pos: 190:21:

DAPP.getDronAvailable(uint256,uint256) : Variables have very similar names "max_" and "min_".Pos: 195:49:

DAPP.getDronAvailable(uint256,uint256) : Variables have very similar names "max_" and "min_".Pos: 196:2:

DAPP.getDronAvailable(uint256,uint256) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 196:13:

DAPP.getDronAvailable(uint256,uint256) : Variables have very similar names "altitudeMax_" and "altitudeMin_".Pos: 197:13:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more

Pos: 107:9: / Pos: 144:11:

Asignatura	Datos del alumno	Fecha
Trabajo Final Blockchain / TFE	Apellidos: Sandoval	20-03-2021
	Nombre: Mario	
	Apellidos: Macaya	
	Nombre: Iker	
	Apellidos: Restrepo	
	Nombre: Yuriel	

CONCLUSIONES

El alcance definido por Unir, y el proposito de los contratos ERC721 y ERC20 se han interiorizado bien en el TFE, por otro lado, la definicion del contrato DAPP encargado de gestionar drones, parcelas, y las transacciones de ordenes de servicio de fumigación e interaccion con los contratos ERC721 y ERC20 reafirman los conceptos de contratos, sus versionamientos trabajados en las secciones de trabajo durante el TFE.

La interaccion entre remix y la red alastria telsus de acuerdo a los procedimientos definidos en las secciones TFE, la activacion de las cuentas en alastria telsus se han interiorizado. Existe frustracion en poner a trabajar el html y js DAPP con la red alastria, la documentacion esta basada sobre pruebas en ambiente Ganace. Trabajamos conexiones por medio de httpprovider y websocket a la red alastria telsus. En ninguno de los casos se ha logrado tener una conexion exitosa.