

# **unir**

LA UNIVERSIDAD  
EN INTERNET

Trabajo Final de Experto (BLO) - PER6061 2022-2023

Hecho por: Rodrigo Garvía Gamboa y Antonio  
Gonzalez Ruiz

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

# ÍNDICE

**1.JUSTIFICACIÓN USO TECNOLOGÍA BLOCKCHAIN PARA RESOLVER EL PROBLEMA RESUELTO.**

**2. ANÁLISIS Y MODELO DEL SISTEMA PROPUESTO**

**2.1. DIAGRAMA DE DESPLIEGUE(CLIENTE)**

**2.2 . CREACIÓN DE PARCELA(CLIENTE)**

**2.3 . CREACIÓN DE DRON (EMPRESA)**

**2.4. DIAGRAMA DE CLASES**

**3. CONTRATOS Y FUNCIONES PRINCIPALES**

**4.DESCRIPCIÓN DEL ENTORNO DE DESARROLLO UTILIZADO**

**3.1. VISUAL STUDIO CODE**

**3.2. REMIX IDE**

**3.3. SOLIDITY**

**3.4. MINIWEB**

**5.INSTRUCCIONES DE DESPLIEGUE**

**6.TESTING**

**6.1.MANUAL DE USO**

Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

## 1. JUSTIFICACION USO TECNOLOGIA BLOCKCHAIN PARA RESOLVER EL PROBLEMA RESUELTO.

Estos son los casos de uso del empleo de smart contracts basado en la tecnología blockchain para la solución del trabajo final de UNIR.

Principalmente hay dos opciones de registro:

Registro de parcelas: los clientes tienen acceso a registrar sus parcelas con sus correspondientes datos.

Registro de drones: **SOLO LA EMPRESA** tiene acceso a la creación de drones donde en ello se incluyen sus correspondientes datos y además el precio por fumigación.

Todo esto se realiza de una forma descentralizada y los clientes tienen la comodidad de tener una interfaz sencilla e intuitiva, además de tener una contratación instantánea de los drones para sus parcelas. Gracias a las funciones de los SMART CONTRACTS, no se necesita tener a una persona que tenga que atender a los clientes ya que está todo el proceso automatizado.

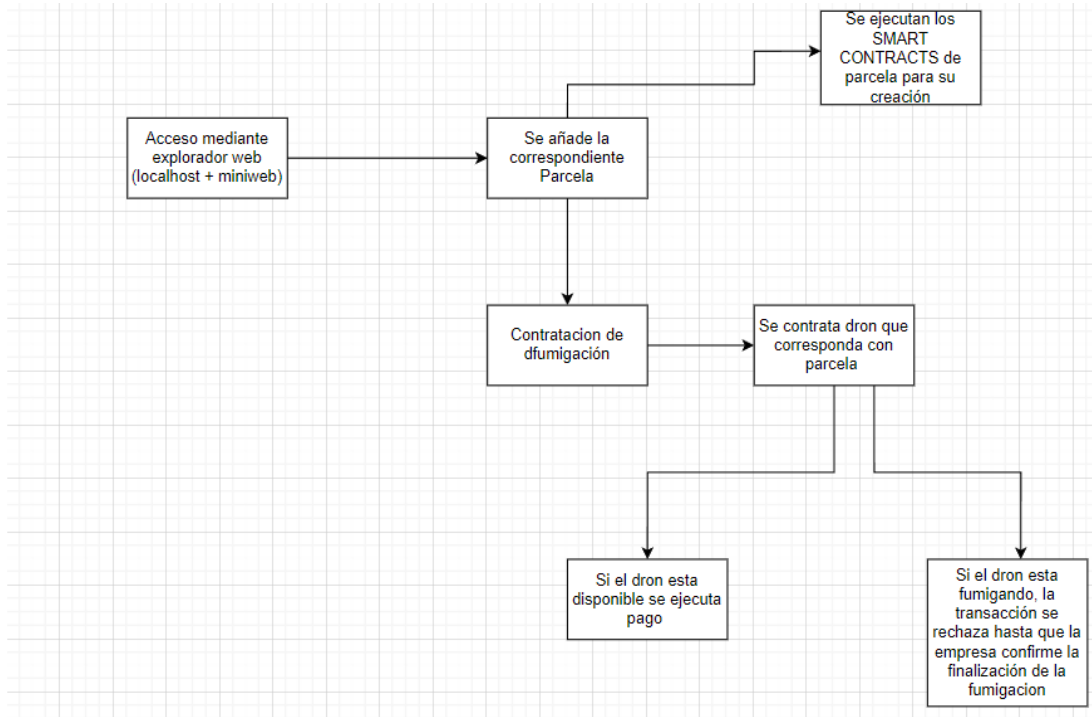
El modelo empleado es un P2P lo que implica la conexión y transferencia directa e inmediata entre empresa y cliente.

Por otra parte el código puede tener alguna vulnerabilidad por lo que pueden entrar personas a realizar actos con malicia.

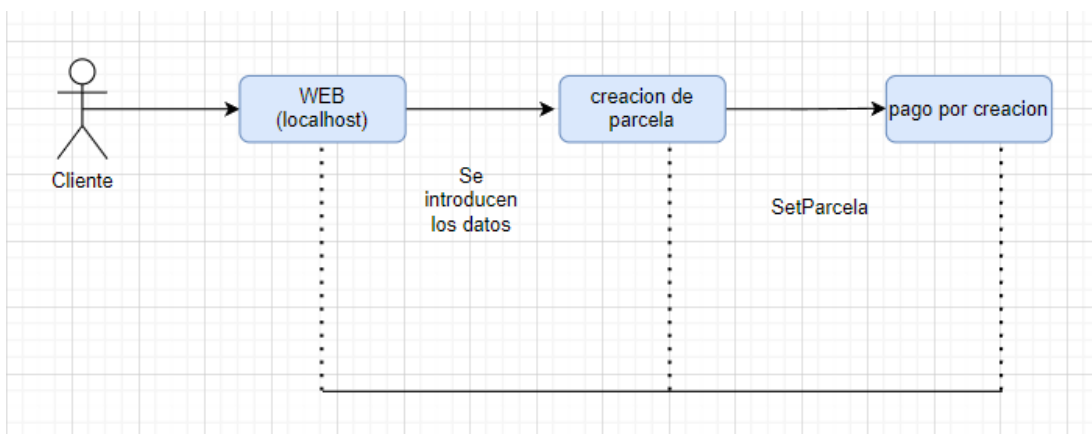
## 2. . ANÁLISIS Y MODELO DEL SISTEMA PROPUESTO

### 2.1 FIGURA 1. DIAGRAMA DE DESPLIEGUE(CLIENTE)

Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



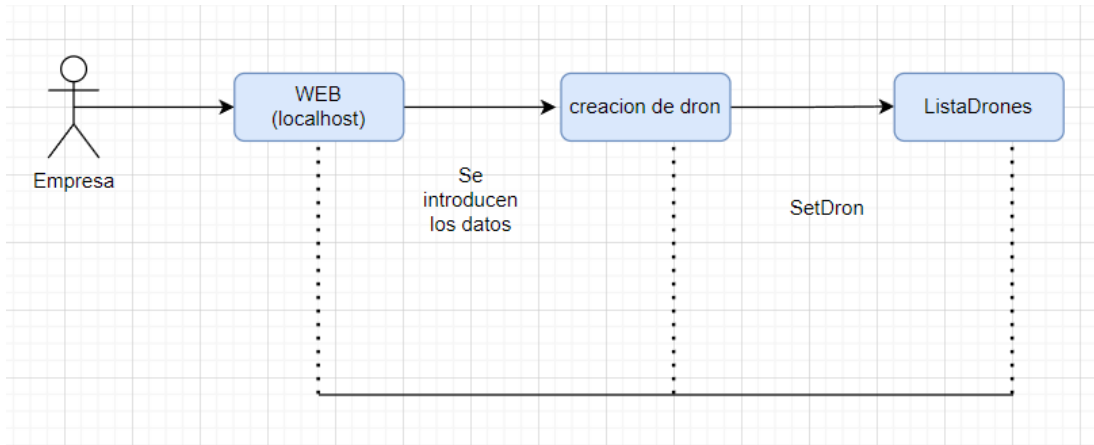
## 2.2 FIGURA 2 CREACIÓN DE PARCELA(CLIENTE)



El pago de la parcela se realiza con gas para crear el token Parcela ERC721

## 2.3 FIGURA 3 CREACIÓN DE DRON (EMPRESA)

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



El pago del dron se realiza con gas para crear el token Dron ERC721

### 3. CONTRATOS Y FUNCIONES PRINCIPALES.

#### Función registrar\_dron y registrar\_parcela.

Esta función es la que crea los tokens, y guarda sus variables de estado.

```

//REGISTRAR DRON PLATAFORMA
@trace | funcSig
function registrar_dron(address tol, uint altura_maximal, uint altura_minimal, uint256 precio!, uint256 cargasBaterial, uint256 M2fumigado!, uint256[] memory pesticida!) public returns(uint256) {
    require(altura_maximal > altura_minimal, "Valor altura incorrecta");
    TokenDron _token_dron = TokenDron(contract TokenDron);
    uint256 tokenId = _token_dron.mint(tol, altura_maximal, altura_minimal, precio!, cargasBaterial, M2fumigado!, pesticida!);
    return tokenId;
}
  
```

#### Función AltaFumigacion.

Contrata la fumigación después de comprobar que el dron está disponible y actualiza su estado.

```

//ALTA FUMIGACION
@trace | funcSig
function AltaFumigacion (uint256 idDron!, uint256 idParcela!, uint256 idPesticida!) external returns (bool result!) {
    require (DisponibilidadDron(idDron!), "El Dron no esta disponible");
    require ([compatibilidad(idDron!,idParcela!), "El Dron no es compatible"]);
    listaFumigacion[idDron!].idParcela=idParcela!;
    listaFumigacion[idDron!].estado = estadoAlta.Fumigando;
    listaFumigacion[idDron!].idPesticida = idPesticida!;
    emit FumigacionContratada(idDron!, idParcela!, idPesticida!, msg.sender, true);
    return true;
}
  
```

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

### Función Fumigacion.

Confirma la fumigación y deja disponible de nuevo el Dron.

```
//EJECUCION DE FUMIGACION
event FumigacionCompletada(uint256 idDron, uint256 idParcela, estadoAlta estado);

ftrace | funcSig
function Fumigacion (uint256 DronFumigador!, uint256 ParcelaFumigar!) external onlyOwner returns (bool result!) {
    require(listaFumigacion[DronFumigador!].idParcela == ParcelaFumigar!, "El Dron no tiene asignada parcela");
    require(listaFumigacion[DronFumigador!].estado == estadoAlta.Fumigando, "Dron no activo en parcela");
    listaFumigacion[DronFumigador!].estado= estadoAlta.Completada;
    emit FumigacionCompletada(DronFumigador!,ParcelaFumigar!,listaFumigacion[DronFumigador!].estado);
    return true;
}
```

### Función compatibilidad.

Aquí se define los requisitos que debe cumplir el Dron para fumigar la parcela, que vienen dados por la altura mínima, máxima y los pesticidas, debe disponer de los pesticidas que necesita la parcela y los valores de altura deben cumplir un rango.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

```
//COMPATIBILIDAD DRON CON PARCELA
fttrace|funcSig
function compatibilidad (uint256 idDron!, uint256 idParcela!) public view returns (bool result!) {

    TokenDron _token_dron = TokenDron(contract TokenDron);
    TokenParcela _token_parcela = TokenParcela(contract TokenParcela);

    if (_token_dron.getAltura_minima(idDron!) > _token_parcela.getAltura_maxima(idParcela!))
        return false;

    if (_token_parcela.getAltura_minima(idParcela!) > _token_dron.getAltura_maxima(idDron!))
        return false;

    uint256[] memory PesticidasDron = _token_dron.listaPesticidas(idDron!);
    uint256[] memory PesticidasParcela = _token_parcela.listaPesticidas(idParcela!);

    bool PesticidalLeno;

    if (PesticidasParcela.length > PesticidasDron.length){
        return false;
    }

    if (PesticidasParcela.length == 0) {
        return true;
    }
    for (uint256 i = 0; i < PesticidasParcela.length ; i++) {
        PesticidalLeno=false;
        for (uint256 j = 0; j < PesticidasDron.length ; j++) {
            if (PesticidasParcela[i] == PesticidasDron[j] ) {
                PesticidalLeno=true;
            }
        }
        if (PesticidalLeno == false)
        {
            return false;
        }
    }

    return true;
}
```

### Función CosteOperacion.

Esta función calcula el precio de la operación con los datos de las veces que recarga la batería el dron en función de la duración de la batería del dron y la superficie de la parcela más el precio del propio dron.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

```

//COSTE FUMIGACION
ftrace|funcSig
function costeOperacion (uint256 idDron!, uint256 idParcela!) external returns (uint256 result!) {
    uint256 vueloDron;
    TokenDron _token_dron = TokenDron(contract TokenDron);
    TokenParcela _token_parcela = TokenParcela(contract TokenParcela);
    vueloDron = (_token_parcela.SuperficieParcela(idParcela!) / _token_dron.VelocidadDron(idDron!));

    emit costeFumigacion (_token_dron.PrecioDron(idDron!), 1 ,_token_dron.BateriaDron(idDron!), vueloDron);
    if (vueloDron <= _token_dron.BateriaDron(idDron!))
        return (_token_dron.PrecioDron(idDron!));

    uint256 recarga = (vueloDron / _token_dron.BateriaDron(idDron!));
    emit costeFumigacion(_token_dron.PrecioDron(idDron!), 1 ,_token_dron.BateriaDron(idDron!), vueloDron);
    return (_token_dron.PrecioDron(idDron!) * recarga);
}

```

### Contrato TokenDron / TokenParcela.

Incluye los datos de cada token, su estructura de datos.

### Contrato FumiDronToken.

Es el token ERC721, el NFT y que tiene una estructura de datos común que heredan los tokens de Drones y Parcelas.

### Contrato FumiDronERC20.

Es el token ERC20 como tal, que se utiliza para los pagos, el cuál se transfiere a la cuenta de la empresa, osea el owner.

### DIAGRAMA DE CLASES:





Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

Remix es una aplicación de código abierto que permite la creación y a su vez ejecución de contratos inteligentes, además permite al usuario ver errores y soluciones rápidas. Este se basa en un editor de Ethereum el cual es perfecto para la programación en Solidity.

### **3. Solidity**

Solidity es un lenguaje de programación el cual se emplea para crear smart contracts basados en la red de ethereum.

### **4. Miniweb**

MiniWeb es un servidor HTTP el cual se usa para poder acceder a los contratos mediante un servidor web localhost.

### **5. Ganache**

Usamos ganache para crear un nodo local y tener una lista de cuentas donde poder realizar todas las pruebas antes del despliegue final en la testnet.

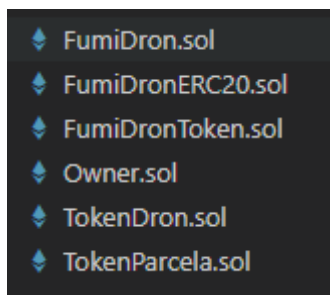
### **6. Metamask**

Billetera para interactuar con la plataforma y aceptar las transacciones.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

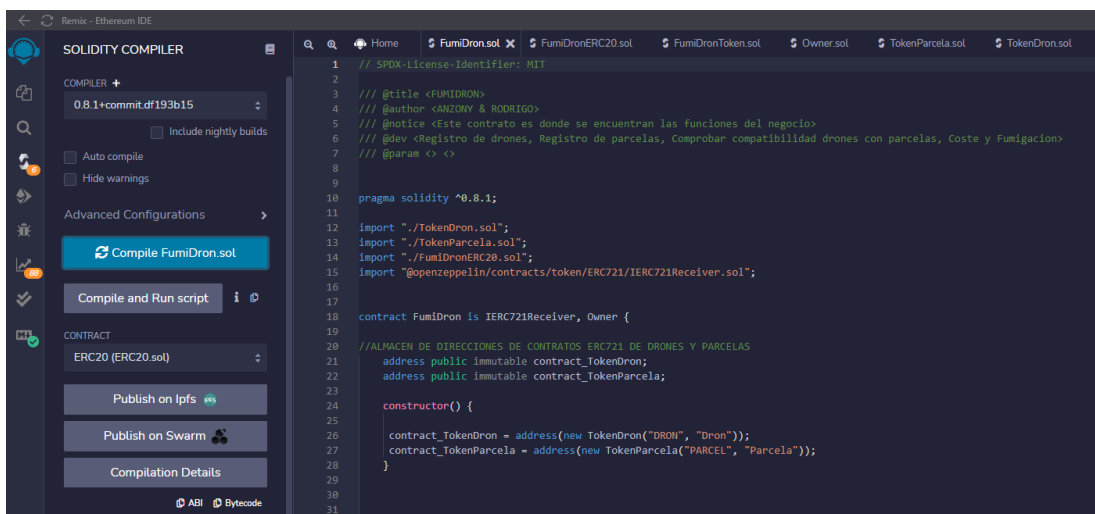
## 5. INSTRUCCIONES DE DESPLIEGUE.

1. Los contratos son 6:



Para el despliegue en la red de los contratos usamos **Remix**, con el plugin **Remixd** conectados al localhost de nuestro área de trabajo donde tenemos los contratos almacenados que hemos estado construyendo con **VS CODE**.

Los compilamos y comprobamos que no hay ningún error, ya antes de realizar todo esto hemos realizado más pruebas localmente usando **Ganache**.

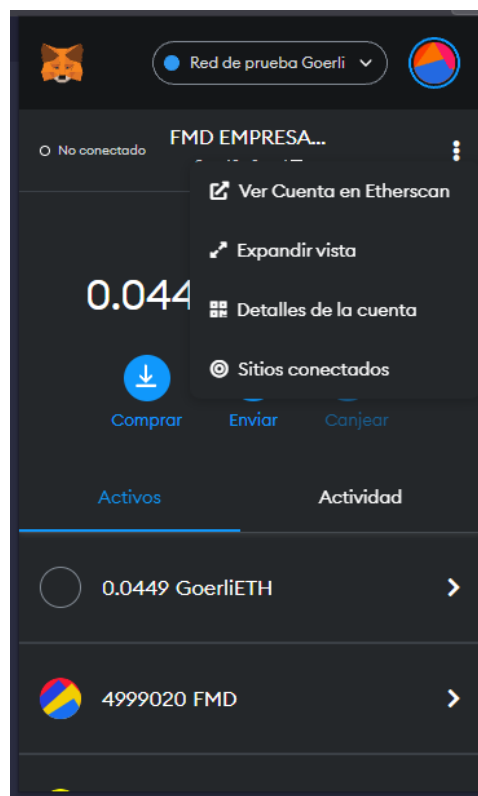


Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

Realizamos el despliegue en una dirección que va a ser la empresa y el dueño del proyecto, quién creará los drones y dará la orden de fumigación y recibirá los tokens.

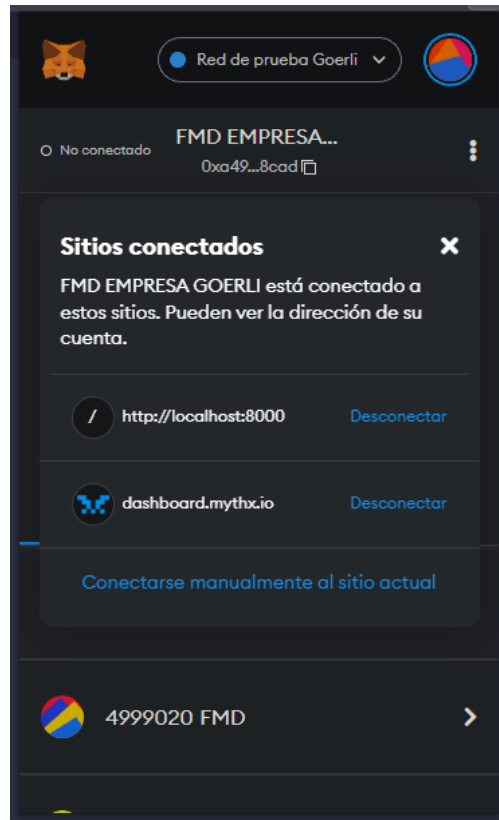
Para hacer el despliegue en la red de pruebas de **Goerli** debemos conectar nuestro Metamask con dicha cuenta a Remix.

Estando en la página de Remix accedemos a metamask y podremos comprobar que aparecemos como “No conectado”, pulsamos en los 3 puntos y entramos en “**Sitios conectados**”.



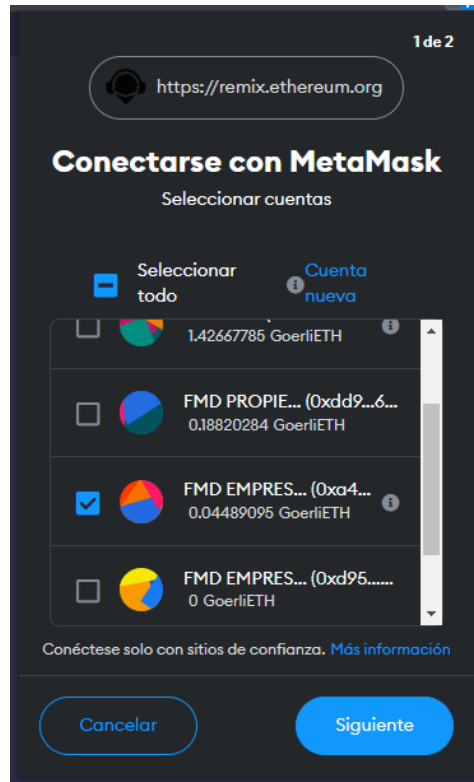
Accedemos a “**Conectarse manualmente al sitio actual**”.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

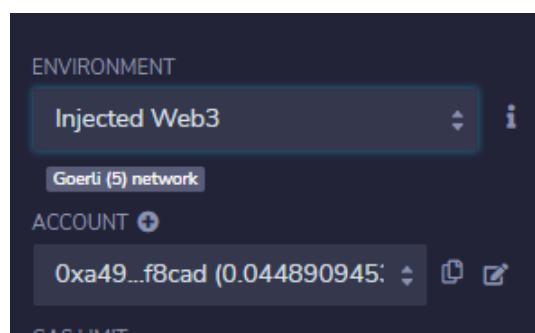


Seleccionamos la cuenta y siguiente.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



Con esto ya tenemos Metamask conectado a Remix y ahora podremos seleccionar la opción para desplegar en Goerli usando la opción **“injected Web3”**.



Ahora ya podemos hacer el despliegue y nos deberá cobrar el gas correspondiente para la creación del contrato.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is active, showing the 'ENVIRONMENT' set to 'Injected Web3', 'ACCOUNT' as '0xE2D...64EB8', 'GAS LIMIT' as '3000000', and 'VALUE' as '0 Wei'. The 'CONTRACT' dropdown shows 'FumiDron - contracts/FumiDron.sol'. The 'Deploy' button is highlighted. Below it, there are options for 'Publish to IPFS' and 'At Address'. The main editor displays the Solidity code for the 'FumiDron' contract, which includes imports for 'TokenDron', 'TokenParcela', 'FumiDronERC20', and 'IERC721Receiver'. The code defines a 'contract FumiDron' with a constructor that initializes 'contract\_TokenDron' and 'contract\_TokenParcela'. It also includes a 'struct fumigacion' and an 'enum estadoAlta'.

The screenshot shows a MetaMask notification window titled 'MetaMask Notification'. It displays a transaction for 'FMD EMP...' on the 'Red de prueba Goerli' network. The transaction is labeled 'Contrato nuevo' and 'IMPLEMENTACIÓN DE CONTRATO'. The URL 'https://remix.ethereum.org' is shown. The notification includes a 'DETALLES' tab and a 'DATOS' tab. The 'DATOS' tab shows the following information:

- Gas (estimada):** 0.02397942
- Gas:** 0.02397942 GoerliETH
- Muy probable en < 15 segundos**
- Tarifa máxima:** 0.02397942 GoerliETH
- Total:** 0.02397942
- 0.02397942 GoerliETH**
- Cantidad + tarifa de gas:** Cantidad máxima: 0.02397942 GoerliETH

At the bottom, there are two buttons: 'Rechazar' and 'Confirmar'.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

```

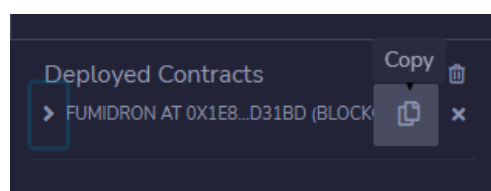
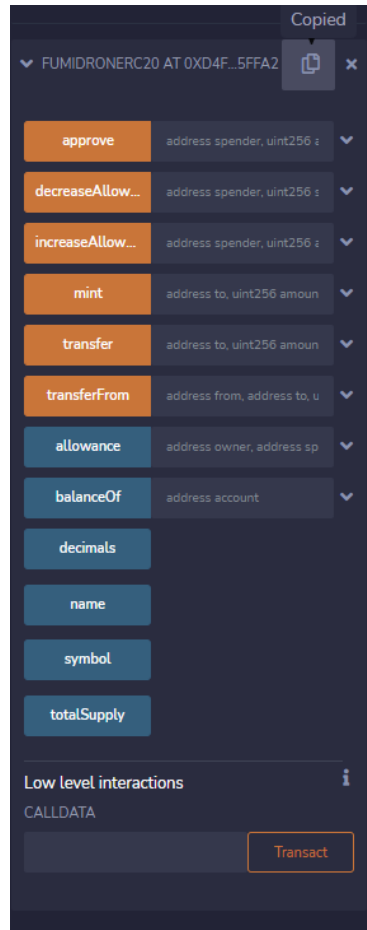
[block:7157239 txIndex:2] from: 0xE2D...64E88 to: FumiDron.(constructor) value: 0 wei data: 0x60c...10033 logs: 0 hash: 0xda6...8a543
status true Transaction mined and execution succeed
transaction hash 0x1ac68c43af6e0f87ea5ead8dbfcd49b9380a6732b191daef05d75483f26caa79
from 0xE2D19164d0DF8888dC01ABC0d4aC849403664E88
to FumiDron.(constructor)
gas 9591767 gas
transaction cost 9591767 gas
input 0x60c...10033
decoded input {}
decoded output -
logs []
val 0 wei

```

Una vez tenemos desplegados todos los contratos tenemos que copiar la dirección del contrato *FumiDron* y *FumiDronERC20* que nos ha proporcionado Remix para añadirla en el código JS junto con el **ABI**.

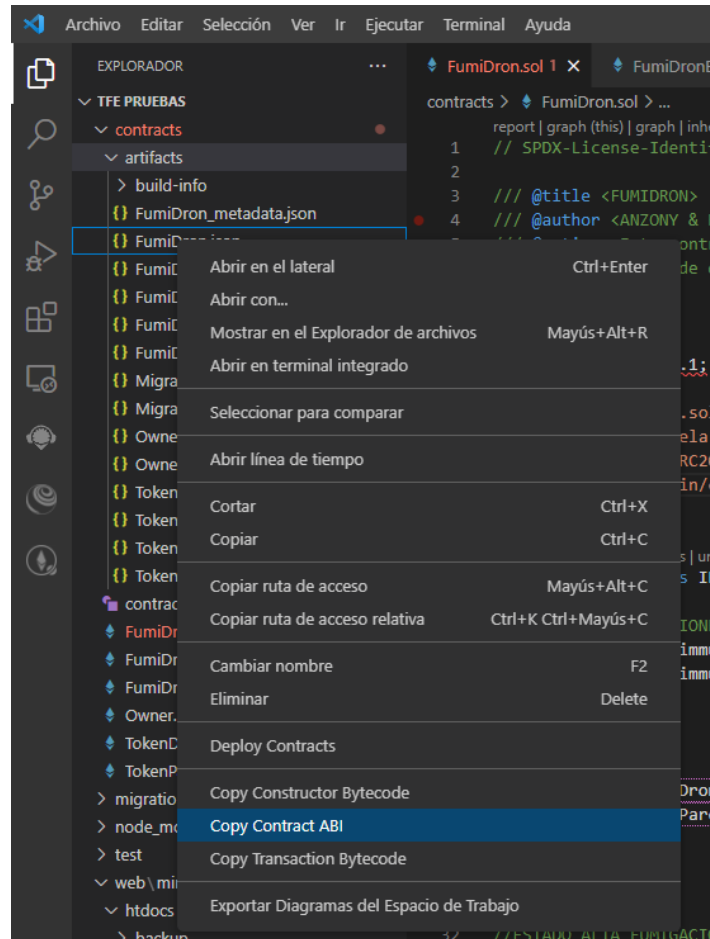


Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



Para copiar el ABI en una sola línea podemos usar **VS CODE**, hay que tener instalado el plugin de **Truffle** para poder usar esta opción:

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



Tenemos que añadir en el código JS de la página web la dirección de Metamask con la que hemos desplegado los contratos y la que va a ser la empresa, las direcciones de los contratos *FumiDron* y *FumiDronERC20* y sus respectivos **ABI**.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

```

<script>

const ABI_FUMIDRONERC20 = [{"inputs":[],"stateMutability":"nonpayable","type":"constructor"},{"anonymous":false,"inputs":[{"indexed":false,"name":"to","type":"address"}],"outputs":[{"indexed":false,"name":"value","type":"uint256"}],"stateMutability":"payable","type":"function"}]
const ABI_FUMIDRON = [{"inputs":[],"stateMutability":"nonpayable","type":"constructor"},{"anonymous":false,"inputs":[{"indexed":false,"name":"to","type":"address"}],"outputs":[{"indexed":false,"name":"value","type":"uint256"}],"stateMutability":"payable","type":"function"}]

var instanciaEmpresa;
var instanciaTokenERC20;
var web3;
var cuentaUsuario;
var cuentaEmpresa;

async function start() {

  window.addEventListener('load', async () => {
    if (window.ethereum) {
      console.log("Metamask detected!!!");
      web3 = new Web3(window.ethereum);
      await window.ethereum.enable();
      cuentaEmpresa="0xE2D19164d0Df8088dC01ABc0d4aCB49403664EB8";
      instanciaTokenERC20 = await new web3.eth.Contract(ABI_FUMIDRONERC20, "0xD4F156bb5DF85248c5930f88F03350E9f085ffa2");
      instanciaEmpresa= await new web3.eth.Contract(ABI_FUMIDRON, "0x8C8c389247092919e5985038eBB7E4D064E69146");
      var accounts = await web3.eth.getAccounts();
      var accountInterval = setInterval( async function() {
        var accounts = await web3.eth.getAccounts();
        var cuenta0 = accounts[0];

        if (cuenta0 !== cuentaUsuario) {
          cuentaUsuario = cuenta0;
          TotalDrones().then(displayDrones);
          pintarBalance(cuentaUsuario);
          TotalParcelas().then(displayParcelas);
        }
      }, 100);
    }
    else {console.error("Metamask not detected");}
  });
}

start();

```

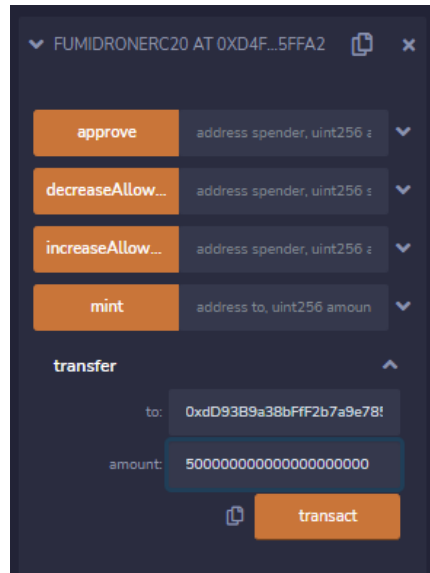
En Metamask tanto la empresa como los clientes para poder visualizar en sus billeteras el token deberán añadirlo usando la dirección del token ERC20 desplegado.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

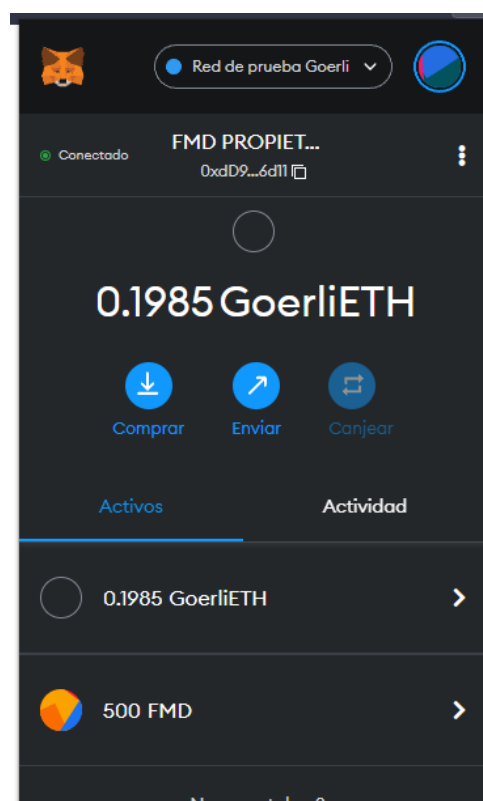
The screenshot shows a mobile application interface for adding a custom token. At the top, there is a header with a fox icon, a dropdown menu set to 'Red de prueba Goerli', and a circular icon. Below the header, the title 'AGREGAR TOKENS' is displayed with a close button. A warning box states 'Token personalizado' and 'Aprenda más sobre estafas y riesgos en seguridad.' The form includes a 'Dirección de contrato de token' field with the value '0xD4F156bb5DF8524Bc5930f88F033', a 'Símbolo del token' field with the value 'FMD' and an 'Editor' link, and a 'Decimales del token' field with the value '18'. A blue button at the bottom is labeled 'Añadir token personalizado'.

La wallet de la empresa al ser quien ha desplegado el token dispone del supply de este pero los clientes deberían disponer de ellos para realizar los pagos, con lo cuál realizaremos una transferencia desde el contrato mismo del token a la dirección de la wallet que queramos realice operaciones en la plataforma.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	



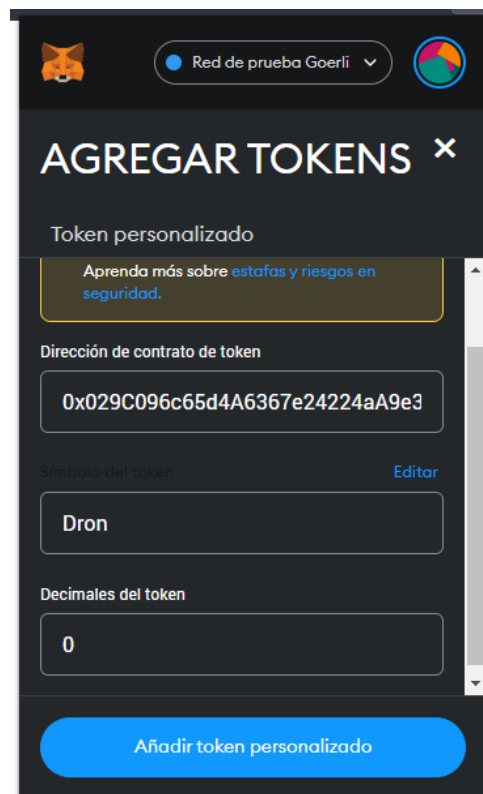
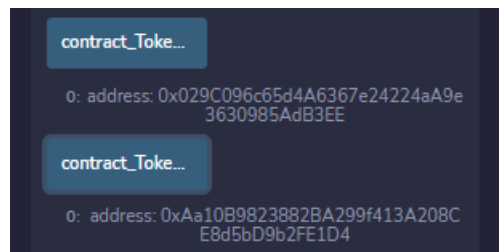
Ahora el cliente dispone de saldo en su billetera.



Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

Si lo deseamos podemos añadir a Metamask aunque no es necesario los tokens **ERC721** que se van creando, con Remix accediendo al contrato *FumiDron* podremos copiar la dirección tanto del ERC721 de los Drones como de las Parcelas.

Hay que tener en cuenta que en este caso no añade automáticamente los decimales como en el caso del token ERC20, hay que añadir un 0.



Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

Token personalizado

Aprenda más sobre [estados y riesgos en seguridad](#).

Dirección de contrato de token

0xAa10B9823882BA299f413A208CE...

Símbolo del token Edit

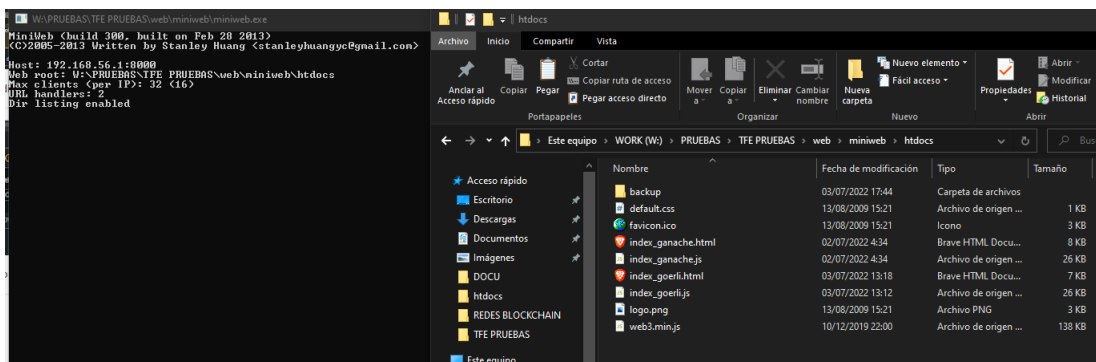
Parcela

Decimales del token

0

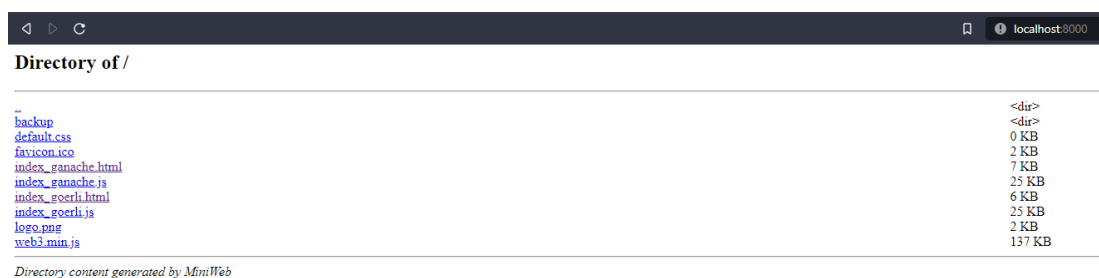
Añadir token personalizado

2. Arrancar el servidor HTTP usando **Miniweb**, colocamos el html y js, en la carpeta del server “**htdocs**” y ejecutamos el exe “**miniweb.exe**”.



Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

Ahora en el navegador entramos en localhost escribiendo “**localhost:8000**”, nos aparecerá el contenido que hemos introducido en la carpeta del server y accedemos abriendo el index “**index\_goerli.html**”.



Y ya tenemos acceso a la plataforma de *FumiDron*, solo faltará conectar nuestro Metamask cuando nos aparezca el aviso.

## 6. TESTING

### 6.1 MANUAL DE USO

Aquí explicamos cada apartado de la web:



Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

FumiDron S.A. --Empresa de fumigacion con drones--

**FUMIDRON TOKEN (FMD) SALDO: 500** ————— Cantidad de tokens disponibles de la wallet conectada a la plataforma.

**LISTA DE DRONES**

Id	Altura máxima	Altura mínima	Duración batería	M2 fumigación	Precio	Pesticidas	Estado del dron
1	40	5	20	50	20	1	Parcela :2 / Estado :fumigando / Pesticida :A
2	40	2	20	50	20	1,2,3,4,5	Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.
3	40	5	10	50	20	1,2,3,4,5	Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.
4	40	5	15	50	10	1,2	Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.

**LISTA DE PARCELAS**

Id	Altura máxima	Altura mínima	Superficie	Pesticidas
1	40	5	50	1
2	40	5	50	1

Lista de parcelas creadas con todas sus propiedades.

**GESTIÓN DE PARCELAS**

**Datos de las parcelas**

Altura Máxima  
 Altura Mínima  
 Superficie(m2)  
Pesticidas que necesita: ☒ A ☒ B ☒ C ☐ D ☐ E

**Solicitar Fumigación**

Id Dron  
 Id Parcela  
 Id Pesticida

Operativa por parte del cliente y pago de la fumigación que solicita con los requisitos que necesita su parcela.

**OPERATIVA DE DRONES**

**Datos de los drones**

Altura Máxima  
 Altura Mínima  
 Precio  
 Duración batería  
 Superficie(m2) fumigación  
Pesticidas disponibles: ☒ A ☒ B ☐ C ☐ D ☐ E

**Consulta**

Id Dron  
 Id Parcela  
 Id Pesticida  
   
Resultado de operaciones |true|

Operativa por parte de la Empresa para la creación de drones, comprobar su estado y dar órden de fumigar y que vuelva a estar disponible para otra fumigación.

En la web hay 2 apartados, el de uso por los clientes quienes crean las Parcelas, y la empresa que crea los Drones y consulta su estado.

Los clientes una vez crean las parcelas con las características necesarias pagando la transacción correspondiente de gas pueden solicitar la fumigación y para ello deben pagar con el token de la plataforma FumiDron (FMD).

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

## GESTIÓN DE PARCELAS

### Datos de las parcelas

Altura Máxima  
 Altura Mínima  
 Superficie(m2)  
 Pesticidas que necesita: ☐ A ☐ B ☐ C ☐ D ☐ E

### Solicitar Fumigación

Id Dron  
 Id Parcela  
 Id Pesticida



Una vez realizado el pago y la fumigación, el Dron queda disponible para otra fumigación.

También puede consultar el estado y compatibilidad de los drones con las parcelas, el resultado aparecerá en “Resultado de operaciones”, indicando un “true” o “false”.

Asignatura	Datos del alumno	Fecha
Trabajo Final	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

## OPERATIVA DE DRONES


### Datos de los drones

<input type="text"/>	Altura Máxima
<input type="text"/>	Altura Mínima
<input type="text"/>	Precio
<input type="text"/>	Duración batería
<input type="text"/>	Superficie(m2) fumigación

Pesticidas disponibles: ☐ A ☐ B ☐ C ☐ D ☐ E

### Consulta y Despliegue

<input type="text" value="1"/>	Id Dron
<input type="text" value="2"/>	Id Parcela
<input type="text" value="1"/>	Id Pesticida

Resultado de operaciones  

En la parte superior de la web junto a la lista de Drones podemos ver los estados de los mismos:

```

;
Estado del dron
Parcela :2 / Estado :fumigando / Pesticida :A
Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.
Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.
Parcela :pendiente. / Estado :sin asignar / Pesticida :pendiente.

```

## 6.2 CONCLUSIONES

Con este trabajo hemos aprendido a aplicar los conceptos de todo lo aprendido sobre tecnologías BLOCKCHAIN aplicado en un caso práctico. Además hemos sabido utilizar herramientas las cuales algunas tienen un proceso de utilización complejo y

Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

además el cómo hacer que varios programas o herramientas que no son compatibles según versión lo sean para este proyecto.

## 7. TESTING

Para el testing hemos empleado el IDE REMIX porque las herramientas de Truffle y Mocha debido a la importación de las librerías.

## 8. ENLACES CONTRATOS DESPLEGADOS GOERLI

**FUMIDRON - 0x1e8C398196C106A72D8dF48E008Ff82e44fD31bd**

**FUMIDRONERC20 - 0x9676edC971e0DAF2E382731982dccD764B8311E6**

Asignatura	Datos del alumno	Fecha
<b>Trabajo Final</b>	Apellidos: Gonzalez, Garvía	
	Nombre: Antonio, Rodrigo	

**Nota:** Por problemas de versiones de compiladores y otros problemas que por falta de tiempo no se ha podido profundizar más en su resolución, puesto que hemos preferido centrarnos primeramente en la funcionalidad de proyecto en sí, hemos descartado la idea de desplegarlo en Alastria y hemos recurrido a hacerlo en la testnet de Ethereum Goerli.

EL CODIGO DE FRONTEND Y BACKEND SE  
PUEDE ENCONTRAR EN GITHUB A TRAVES  
DE ESTE LINK

[https://github.com/Byrevenge3r/TFE\\_UNIR\\_BLOCKCHAIN](https://github.com/Byrevenge3r/TFE_UNIR_BLOCKCHAIN)