

Tabla de contenido

| | |
|-------------------------------|----|
| Tabla de contenido..... | I |
| Índice de Ilustraciones..... | II |
| 1 Contratos inteligentes..... | 1 |
| 2 Servidor intermediario..... | 2 |

Índice de Ilustraciones

| | |
|---|----------|
| <i>Ilustración 1. Archivo de configuración de hardhat.</i> | <i>1</i> |
|---|----------|

1 Contratos inteligentes

El código fuente del contrato inteligente se encuentra en la ruta `src/smart-contracts/contracts`.

El lenguaje de programación utilizado para el desarrollo del contrato es Solidity y para la compilación y despliegue de este se ha utilizado la herramienta Hardhat.

Para compilar el contrato se utiliza el siguiente comando:

```
% yarn hardhat compile
```

Para el despliegue del contrato se utiliza el siguiente comando:

```
% yarn hardhat deploy --network localhost
```

Se le puede añadir el parámetro `--network` para indicar la red en la que se quiere desplegar, en el ejemplo sería en local. Si se ejecuta el comando omitiendo este parámetro, Hardhat creará una blockchain virtual y desplegará el contrato en ella.

Hardhat proporciona un archivo de configuración (`src/smart-contracts/hardhat-config.js`) en el que se pueden configurar, por ejemplo, diferentes redes para poder desplegar de manera sencilla en la que sea necesario. En este caso únicamente se ha configurado una red local:

```
module.exports = {
  defaultNetwork: "hardhat",
  networks: {
    localhost: {
      url: "http://localhost:8545/",
      chainId: 31337,
      blockConfirmations: 1,
    },
  },
  namedAccounts: {
    deployer: {
      default: 0,
    },
  },
  solidity: "0.8.8",
}
```

Ilustración 1. Archivo de configuración de hardhat.

Esta red local es diferente a la que se genera virtualmente al ejecutar el comando de despliegue y para poder desplegar el contrato en ella primero se debe crear un nodo mediante el comando:

```
% yarn hardhat node
```

2 Servidor intermediario

El código fuente del contrato inteligente se encuentra en la ruta `src/server`.

El servidor está desarrollando en Javascript utilizando NodeJS y ExpressJS.

El código fuente se encuentra dividido en ocho carpetas:

- La carpeta `config` contiene archivos de configuración, para la conexión a la base de datos y a la blockchain.
- La carpeta `controllers` contiene las clases controlador, encargadas de manejar las peticiones.
- La carpeta `logs` contiene los registros de las peticiones que se realizan en el servidor, así como registros de errores.
- La carpeta `middlewares` contiene utilidades personalizadas creadas para el uso del servidor, como un limitador de intentos de inicio de sesión y el verificador de cookies.
- La carpeta `models` contiene las clases modelo, encargadas de representar las entidades del modelo de datos.
- La carpeta `public` contiene las hojas de estilos (archivos `css`).
- La carpeta `routes` contiene las clases ruta donde se definen las rutas a las que llegarán las peticiones, así como el método de cada una de ellas.
- La carpeta `views` contiene las vistas del servidor intermediario, formado únicamente por una vista principal y una vista de error.

Para la base de datos se ha establecido la ruta de conexión en el archivo de entorno `.env`, para utilizar una base de datos de MongoDB diferente únicamente habría que cambiar la ruta en el archivo de configuración.

Para instalar las dependencias, es necesario hacer uso del comando:

```
% npm install
```

Para compilar el proyecto, es necesario hacer uso del comando:

```
% npm run build
```

Para iniciar el servidor intermediario, es necesario hacer uso del comando:

```
% npm start
```