To structure a smart contract on the Cardano platform, following the defined parameters, we can conceptualize the process as follows:

1. Definition of the Contract and Main Variables

• Applicant address: The person requesting the creation of the NFT.

• Receiver Address: The person who authorizes the payment and receives the funds.

• Payment amount: The amount of ADA to be transferred for the creation of the NFT.

• Title and notes: Information that will be stored in the NFT.

• Transaction hash: The hash that represents the transaction in the blockchain.

2. Main Components of the Contract

to. NFT Creation Request

• The applicant initiates an application by providing the title, notes, payment amount and recipient's address.

• The contract verifies that the payment amount is correct and stores the NFT data in a temporary state.
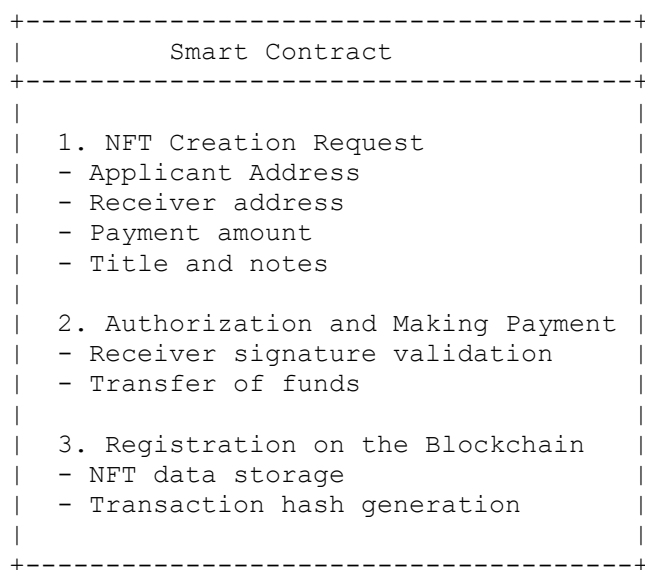
b. Authorization and Making Payment

• The payment recipient reviews the request and digitally signs the transaction to authorize the payment.

• The contract validates the recipient's signature to guarantee authorization.

c. Registration on the Blockchain

• Once payment is authorized, the contract transfers the funds to the recipient.

• The NFT data (title, notes, recipient address) is recorded on the blockchain.

• A hash of the transaction is generated, which is stored along with the NFT data.

3. Conceptual Diagram

```
+-----------------------------------+
|            Smart Contract         |
+-----------------------------------+
|                                   |
|                                   |
|   1. NFT Creation Request         |
|   - Applicant Address             |
|   - Receiver address              |
|   - Payment amount                |
|   - Title and notes               |
|                                   |
|   2. Authorization and Making Payment |
|   - Receiver signature validation |
|   - Transfer of funds             |
|                                   |
|   3. Registration on the Blockchain |
|   - NFT data storage              |
|   - Transaction hash generation   |
|                                   |
+-----------------------------------+
```

4. Smart Contract Steps

- Creation Request

• The requester calls the smart contract and provides the necessary details (title, notes, payment amount, recipient address).

• The contract verifies that the payment amount is correct.

- Payment Authorization

• The recipient reviews the request and digitally signs to authorize payment.

• The contract validates the recipient's signature to ensure that the authorization is valid.

- Payment and Registration

• Once authorized, the contract transfers the funds to the recipient.

• The NFT data (title, notes, recipient address) is recorded on the blockchain.

• A hash of the transaction is generated and stored.

5. Implementation (Conceptual)

```typescript
// console.log(api);
// console.log(lucid);
// console.log(address);
export const toyHastala3 = async () => {
    let conectado = await conectar();
    let texto: string = await address();
    return console.log(texto);
};
// export cost crearTitulos = async(params) => {
//    if (lucid) {
//        crearJSONcip25();
//    }
// }
export const demoTx = async () => {
    if (lucid) {
        const tx = await lucid
            .newTx()
            // .payToAddress(
            //    'addr_test1qrwpq1t04uIue8ow3d6ydcms8uczqrr6ufm8uctu3psg4yxqa7j5l6vncredu53kk3k74f9uxjekclpc3af7k5wf3pdqtk6tn9',
            //    { lovelace: 5000000n },
            // )
            .payToAddress(
                'addr_test1qpc6mrwu9cucrq4w6y69qchflvypq76a47ylvjvm2wph4szeq579yu2z8s4m4tn0a9g4gfce58p25afc24knsf6pj96sz35wnt',
                { lovelace: 5000000n },
            )
            .complete();
        const signedTx = await tx.sign|.complete();
        const txHash = await signedTx.submit();
    }
    // console.log(demoTx());
    //console.log('ya me pase demotx');
};
// if (window.cardano) {
//    conectar();demoTx()
// }
if ((await cardanoIsEnabled()) && showBalance && wallet) {
    getBalance();
    setWallet();
}
export const crearTitulos = async (): Promise<Resultado<any>> => {
    //let resultado: Resultado<any> = { type: "error", error: new Error("No se pudo crear el titulo") };
    try {
        if (lucid) {
            var cedula = document.getElementById("cedula");
            var nombre = document.getElementById("nombre");
            var apellido = document.getElementById("apellido");
            var titulo = document.getElementById("titulo_cargado");
```

```typescript
export const crearTitulos = async (): Promise<Resultado<any>> => {
            var notas = document.getElementById("nota_cargado");
            const datos = {
                cedula: cedula.value,
                nombre: nombre.value,
                apellido: apellido.value,
                titulo: titulo.value,
                notas: notas.value
            };
            //console.log(datos["cedula"]);
            const dadaPM_TitulosyNotas: MintingPolicy = {
                type: "PlutusV2",
                script: contratos.scripts.pm_titulosNotas
            }
            const dadaVal_Validacion: SpendingValidator = {
                type: "PlutusV2",
                script: contratos.scripts.val_verificacion
            }
            const dadaValDireccion: string = lucid.utils.validatorToAddress(dadaVal_Validacion);
            const direccionEstudiante = await lucid.wallet.address();
            const tokenName_Titulos = "ADA"+titulo+"+datos['nombre'];
            const tokenName_Notas = "ADANotas"+datos['nombre'];
            const titulos_tokenName = fromText(tokenName_Titulos);
            const notas_tokenName = fromText(tokenName_Notas);
            const titulosynotas_pid: PolicyId = lucid.utils.mintingPolicyToId(dadaPM_TitulosyNotas);
            const titulos_dada: Unit = toUnit(titulosynotas_pid, titulos_tokenName);
            const notas_dada: Unit = toUnit(titulosynotas_pid, notas_tokenName);
            const titulosNotasRedeemer = BigInt(1);
            const mintRedeemer = Data.to(titulosNotasRedeemer);
            // console.log(dadaPM_TitulosyNotas);
            // console.log(datos);
            // console.log("address -> " + direccionEstudiante);
            // console.log("Nombre Token Titulos: " + tokenName_Titulos + " -> " + titulos_tokenName);
            // console.log("Nombre Token Notas: " + tokenName_Notas + " -> " + notas_tokenName);
            // console.log("pid         -> " + titulosynotas_pid)
            // console.log("NFT Titulo:   -> " + titulos_dada);
            // console.log("NFT Notas     -> " + notas_dada);
            // console.log(fromUnit(titulos_dada));
            // console.log(fromUnit(notas_dada));
            // console.log("redeemer      -> " + titulosyNotasRedeemer);
            // console.log(mintRedeemer);
            const jsonData: MD_Titulos = {
                [titulosynotas_pid]: {
                    [tokenName_Titulos]: {
                        id: datos["titulo"],
                        name: datos["titulo"],
                        //image: "https://shorturl.at/NRvii"
```

```typescript
export const crearTitulos = async (): Promise<Resultado<any>> => {
            const jsonData: MD_Titulos = {
                [titulosynotas_pid]: {
                    [tokenName_Titulos]: {
                        image: "https://i.blogs.es/ceda9c/dalle/1366_2000.jpg",
                        description: "Diplomada titulo Test"
                    },
                    [tokenName_Notas]: {
                        id: datos['notas'],
                        name: datos['notas'],
                        //image: "https://shorturl.at/NRvii",
                        image: "https://acortar.link/u9ddTP",
                        description: "Diplomada Notas Test"
                    },
                    datos_estudiante: {
                        hash: "fac7b8513f4b985174e85a02ee8165fc",
                        //cedula: datos['cedula'],
                        nombres: datos['nombre'],
                        apellidos: datos['apellido'],
                        cedula: datos['cedula']
                    },
                },
            };
            const tx: any = await lucid
                .newTx()
                .mintAssets({ [titulos_dada]: BigInt(2), [notas_dada]: BigInt(2) }, mintRedeemer)
                .payToAddress(direccionEstudiante, { [titulos_dada]: BigInt(1), [notas_dada]: BigInt(1), lovelace: BigInt(14000000)})
                .payToContract(dadaValDireccion, Data.to(BigInt(1)), { [titulos_dada]: BigInt(1), [notas_dada]: BigInt(1), lovelace: BigInt(14000000)})
                .attachMintingPolicy(dadaPM_TitulosyNotas)
                .attachMetadata("721", jsonData)
                .complete();
            const signedTx = await tx.sign().complete();
            const txHash = await signedTx.submit();
            // const success = await lucid!.awaitTx(txHash)
            // console.log(tx);
            // console.log(success);
            //return txHash;
            console.log(txHash);
            const mensaje = "por aqui si es";
            //console.log(mensaje);
            return { type: "ok", data: "tx"};
            //return { type: "ok", data: txHash };
        }
    } catch (error) {
        if (error instanceof Error) return { type: "error", error: error };
        return { type: "error", error: new Error(error as string) };
    }
```

documents.blade.php   sidebar-menu.blade.php   demostra.ts •   wallet.ts   web.php   aspiranteController.php

resources > js > demostra.ts > ...

```
229    }
230    const hastala = document.getElementById("hastala");
231    if (hastala) {
232        hastala.addEventListener("click", async function () {
233            const demo = await crearTitulos();
234        });
235    }
```