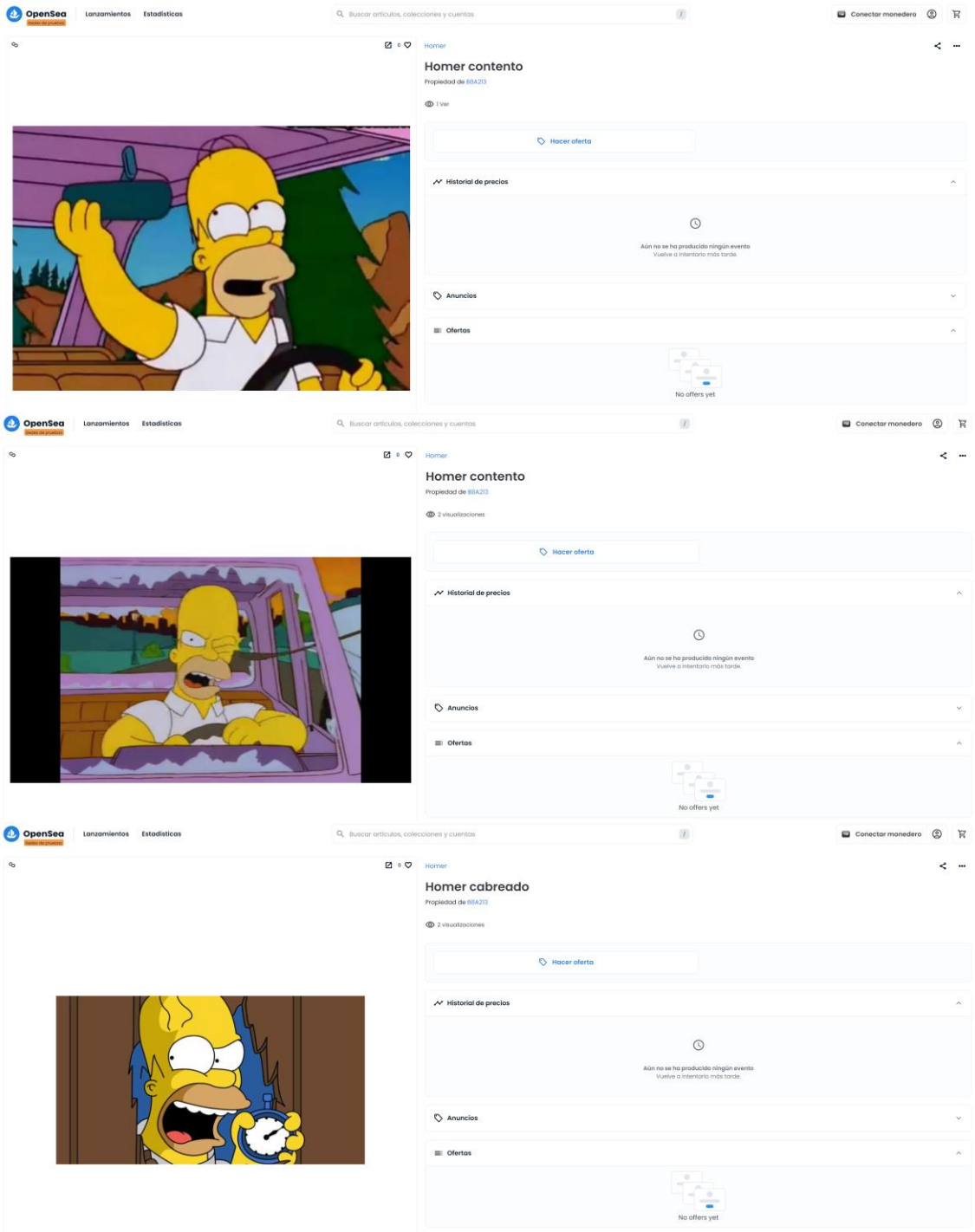
Contrato de primer minteo (ausente de automatización)

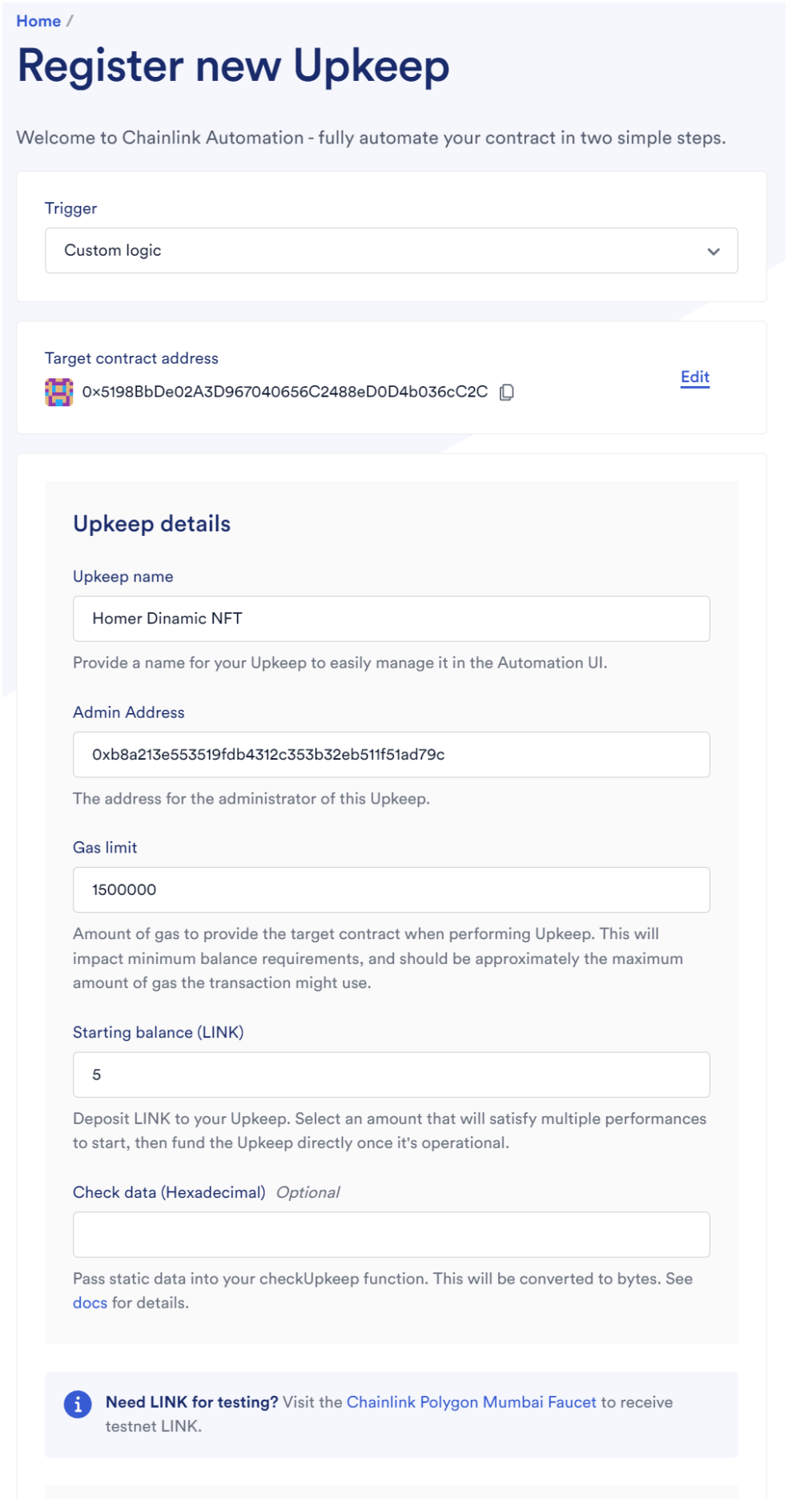
```solidity
// SPDX-License-Identifier: MIT

//Creado por Jonatan Gomez Garcia gracias a las clases de Bitlab
pragma solidity ^0.8.9;

import "@openzeppelin/contracts@4.8.2/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts@4.8.2/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts@4.8.2/access/Ownable.sol";
import "@openzeppelin/contracts@4.8.2/utils/Counters.sol";

contract Homer is ERC721, ERC721URIStorage, Ownable {
    using Counters for Counters.Counter;

    string[] IpfsUri = [
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_0.json",
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_1.json",
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_2.json"
    ];

    Counters.Counter private _tokenIdCounter;

    constructor() ERC721("Homer", "HOM") {}  infinite gas 2558200 gas

    function safeMint(address to) public {  infinite gas
        uint256 tokenId = _tokenIdCounter.current();
        _tokenIdCounter.increment();
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, IpfsUri[0]);
    }

    function pympMyRide(uint256 _tokenId) public{  infinite gas
        require(homerLevel(_tokenId) < 2, "Homer totalmente cabreado!");

        // obtiene el valor actual del auto y le suma 1
        uint256 newVal = homerLevel(_tokenId) + 1;
        // store the new URI
        string memory newUri = IpfsUri[newVal];
        //Update the URI
        _setTokenURI(_tokenId, newUri);
    }

    // helper functions
    //
   function homerLevel(uint256 _tokenId) public view returns(uint256){  infinite gas
        string memory _uri = tokenURI(_tokenId);

        uint result;
        //Guardamos el factor comparador para una mejor eficiencia en cuanto a consumo de gas
        bytes32 toFind = keccak256(abi.encodePacked(_uri));

        for (uint256 index = 0; index < IpfsUri.length; index++) {
            if(keccak256(abi.encodePacked(IpfsUri[index])) == toFind)
                result = index;
        }
        return result;
    }

    // The following functions are overrides required by Solidity.

    function _burn(uint256 tokenId) internal override(ERC721, ERC721URIStorage) {  infinite gas
        super._burn(tokenId);
    }

    function tokenURI(uint256 tokenId)  infinite gas
        public
        view
        override(ERC721, ERC721URIStorage)
        returns (string memory)
    {
        return super.tokenURI(tokenId);
    }
}

```

Como se observa en el código anterior he aplicado una pequeña mejora en términos de rendimiento de gas guardando el factor comparador en forma de variable en vez de calcularlo por cada vuelta de bucle, para este caso es casi imperceptible, ya que solo trabajamos con tres niveles o vueltas dentro del bucle. Sería interesante qué otras opciones de optimización son posibles.

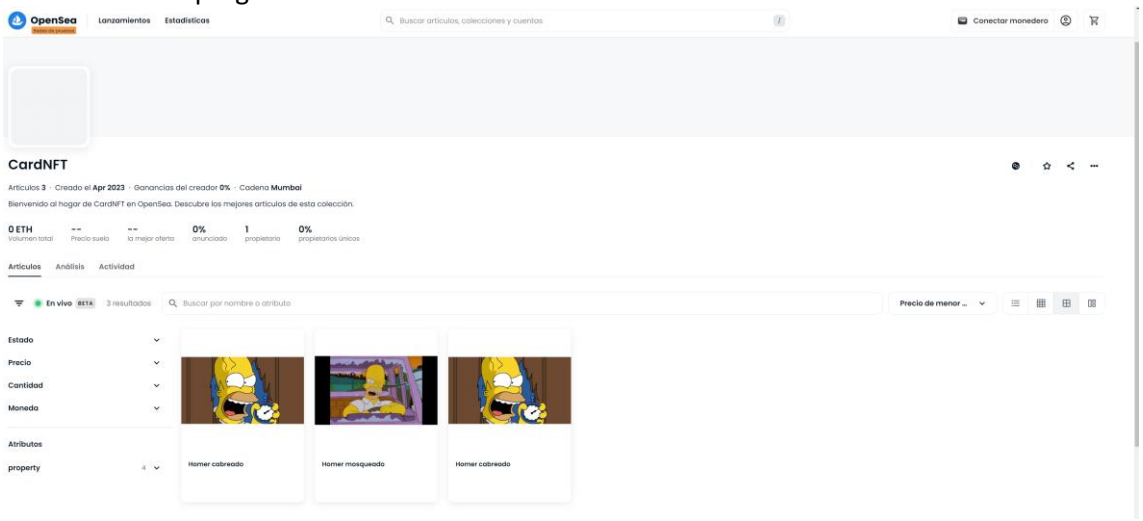


Contrato con automatización

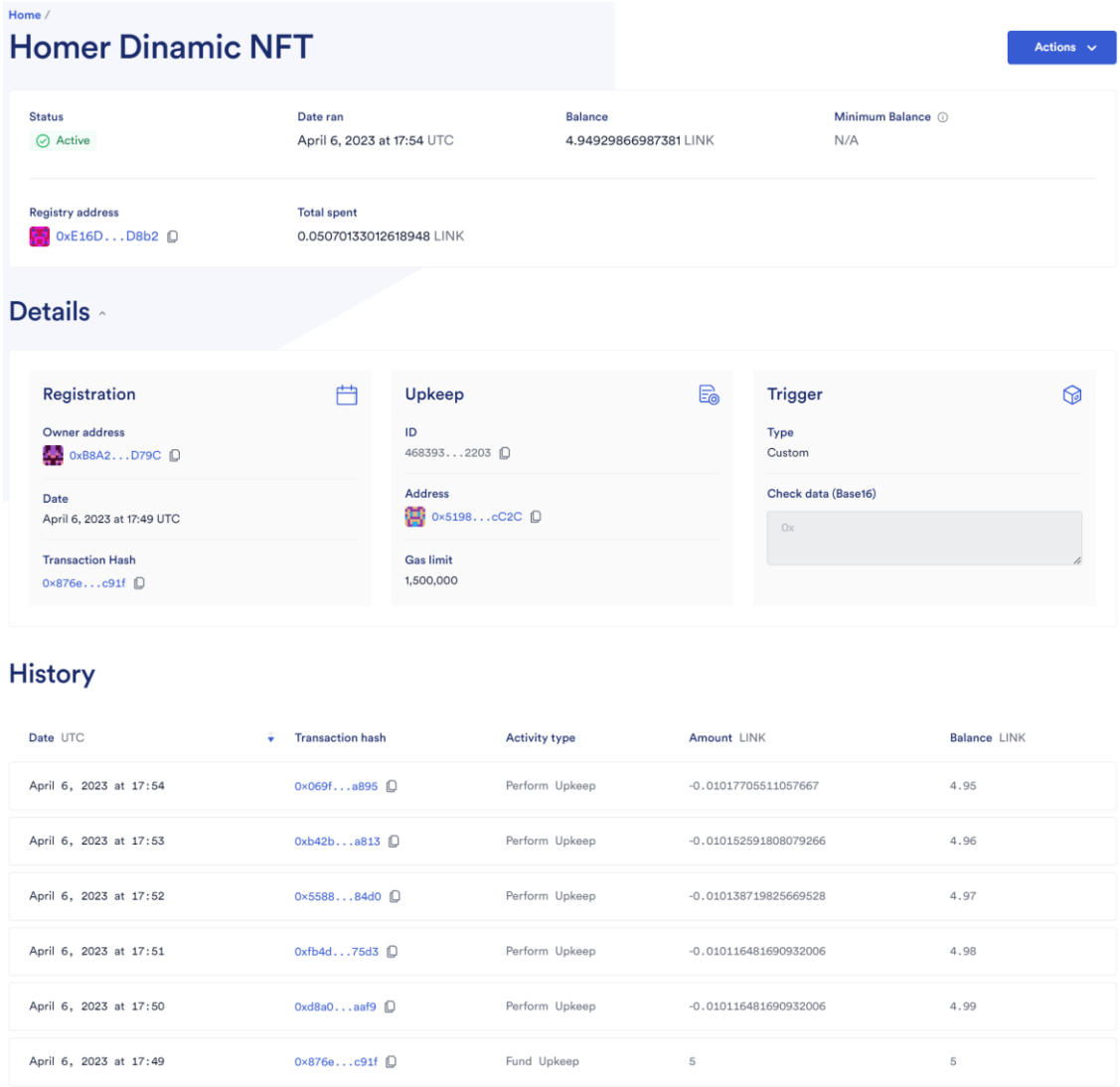
```solidity
// SPDX-License-Identifier: MIT

//Creado por Jonatan Gomez Garcia gracias a las clases de Bitlab
pragma solidity ^0.8.9;

import "@openzeppelin/contracts@4.8.2/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts@4.8.2/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts@4.8.2/access/Ownable.sol";
import "@openzeppelin/contracts@4.8.2/utils/Counters.sol";
import "@chainlink/contracts/src/v0.8/AutomationCompatible.sol";

contract Homer is ERC721, ERC721URIStorage, Ownable, AutomationCompatibleInterface {
    using Counters for Counters.Counter;

    string[] IpfsUri = [
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_0.json",
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_1.json",
        "https://gateway.pinata.cloud/ipfs/QmavFFBcRNvpUHHj9Bbv7ZqhF3zibzAsKUJNJs7J1YAZT3/homer_2.json"
    ];

    Counters.Counter private _tokenIdCounter;

    uint interval;
    uint lastTimeStamp;

    constructor(uint _interval) ERC721("CardNFT", "dNFT") {   infinite gas 2661200 gas
        interval = _interval;
        lastTimeStamp = block.timestamp;
    }


    function safeMint(address to) public {   infinite gas
        uint256 tokenId = _tokenIdCounter.current();
        _tokenIdCounter.increment();
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, IpfsUri[0]);
    }

    function pympMyRide(uint256 _tokenId) public{   infinite gas
        require(homerLevel(_tokenId) < 2, "Homer totalmente cabreado!");

        // obtiene el valor actual del auto y le suma 1
        uint256 newVal = homerLevel(_tokenId) + 1;
        // store the new URI
        string memory newUri = IpfsUri[newVal];
        //Update the URI
        _setTokenURI(_tokenId, newUri);
    }

    // helper functions
    //
   function homerLevel(uint256 _tokenId) public view returns(uint256){   infinite gas
        string memory _uri = tokenURI(_tokenId);

        uint result;
        //Guardamos el factor comparador para una mejor eficiencia en cuanto a consumo de gas
        bytes32 toFind = keccak256(abi.encodePacked(_uri));

        for (uint256 index = 0; index < IpfsUri.length; index++) {
            if(keccak256(abi.encodePacked(IpfsUri[index])) == toFind)
                result = index;
        }
        return result;
    }

    function checkUpkeep(bytes calldata /* checkData */) external view override returns (bool upkeepNeeded, bytes memory /* performData */) {   infinite gas
        upkeepNeeded = (block.timestamp - lastTimeStamp) > interval;
    }

    function performUpkeep(bytes calldata /* performData */) external override  {   infinite gas
        if ((block.timestamp - lastTimeStamp) > interval ) {
            lastTimeStamp = block.timestamp;

            uint counter = _tokenIdCounter.current();

            for (uint256 index = 0; index < counter; index++) {
                if(homerLevel(index) < 2){
                    pympMyRide(index);
                    break;
                }
            }
        }
    }

    // The following functions are overrides required by Solidity.

    function _burn(uint256 tokenId) internal override(ERC721, ERC721URIStorage) {   infinite gas
        super._burn(tokenId);
    }

    function tokenURI(uint256 tokenId)   infinite gas
        public
        view
        override(ERC721, ERC721URIStorage)
        returns (string memory)
    {
        return super.tokenURI(tokenId);
    }
}
```


Home /
Register new Upkeep
Welcome to Chainlink Automation - fully automate your contract in two simple steps.
Trigger
Custom logic
Target contract address
0x5198BbDe02A3D967040656C2488eD0D4b036cC2C
Edit
Upkeep details
Upkeep name
Homer Dinamic NFT
Provide a name for your Upkeep to easily manage it in the Automation UI.
Admin Address
0xb8a213e553519fdb4312c353b32eb511f51ad79c
The address for the administrator of this Upkeep.
Gas limit
1500000
Amount of gas to provide the target contract when performing Upkeep. This will impact minimum balance requirements, and should be approximately the maximum amount of gas the transaction might use.
Starting balance (LINK)
5
Deposit LINK to your Upkeep. Select an amount that will satisfy multiple performances to start, then fund the Upkeep directly once it's operational.
Check data (Hexadecimal) Optional
Pass static data into your checkUpkeep function. This will be converted to bytes. See docs for details.
Need LINK for testing? Visit the Chainlink Polygon Mumbai Faucet to receive testnet LINK.

Podemos ver el progreso



Finalmente, todos los NFT en su último nivel, tenemos 3 NFT, evolucionan 2 veces: resultado 6 movimientos dentro de Automation Chainlink



History

Date UTC	Transaction hash	Activity type	Amount LINK	Balance LINK
April 6, 2023 at 17:54	0x069f...a895	Perform Upkeep	-0.01017705511057667	4.95
April 6, 2023 at 17:53	0xb42b...a813	Perform Upkeep	-0.010152591808079266	4.96
April 6, 2023 at 17:52	0x5588...84d0	Perform Upkeep	-0.010138719825669528	4.97
April 6, 2023 at 17:51	0xfb4d...75d3	Perform Upkeep	-0.010116481690932006	4.98
April 6, 2023 at 17:50	0xd8a0...aaf9	Perform Upkeep	-0.010116481690932006	4.99
April 6, 2023 at 17:49	0x876e...c91f	Fund Upkeep	5	5