算法生成艺术铭文NFT

Generative Art NFT Based on Bitcoin Chain

https://chain.link/hackathon

Technical Features

A：Generative Art Algorithm to create NFT

B：Using HTML as NFT file(High Resolution, Low Storage Space)

C：Storing Algorithm Files and NFT Files on the Bitcoin Chain
(Instead of IPFS)

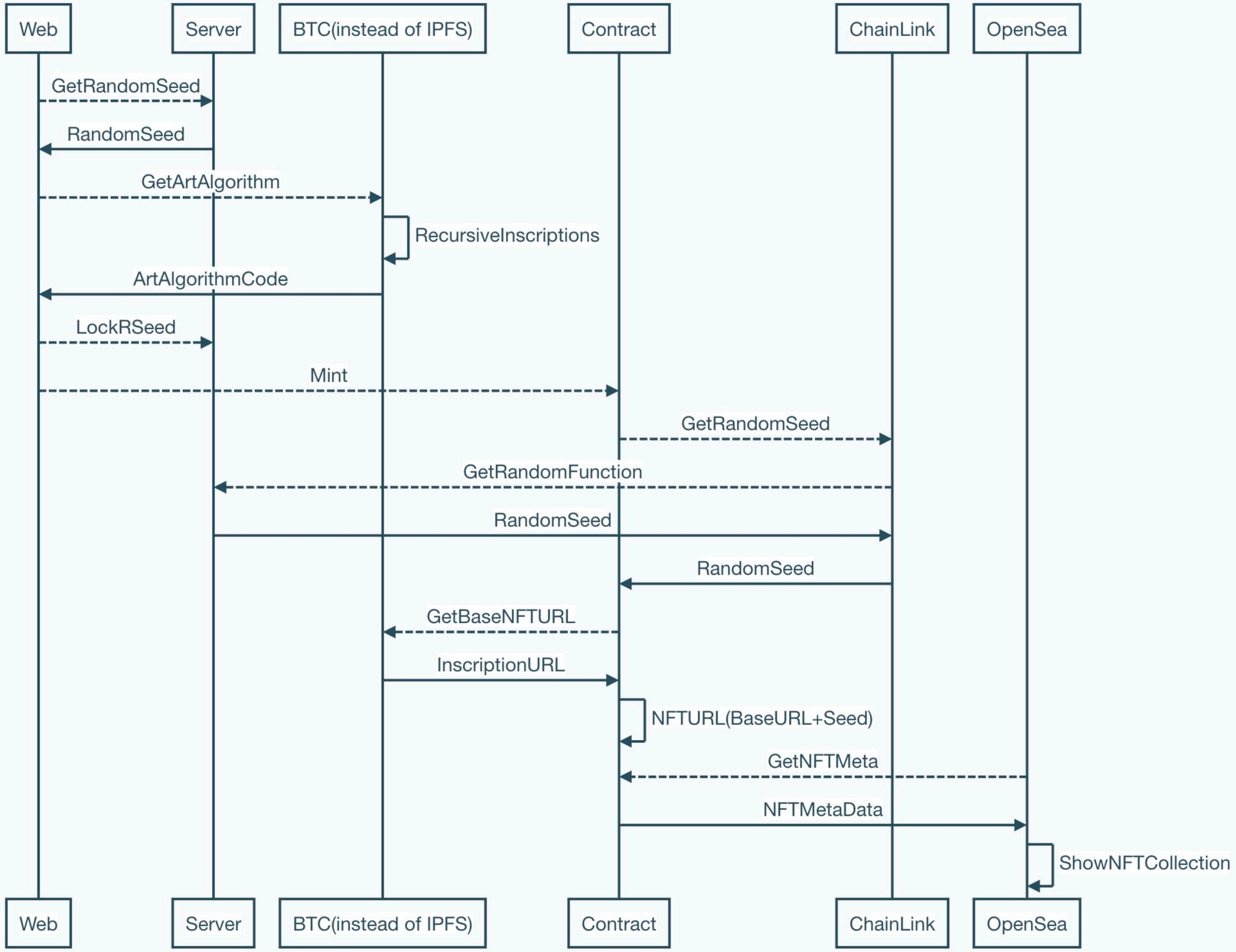
D：Recursive Inscription Handling Scheme

E：Using Chainlink Functions to Obtain a Random Seed

F: "Generating art" has more market potential relative to "AI prompts".

G：ERC721 contracts deployed on the Avalanche chain.

GenerativeArtNFT Project Architecture Sequence



NFT Html File Using Recursive Inscription Files On BTC Chain

```
<html lang="en">
<head>
    <meta ticker="ColorMask" opt="mint" hashseed="qbZBPGd0omxGVvNg">
    <script type="module">

        import * as fcompress from '/content/
            86713dc4a459cedb94cdc2ad64b29819d146f1146dfa45ea24948a89c86cf8c4i0';

        async function getBaseFile()
        {
            const baser = await fetch('/content/
                c116f440066792e41ff2832f556ec54e3558f521953a3f3db9dbd6196ec3c52fi0');
            const bases = await baser.text();
            const base = fcompress.strFromU8(fcompress.gunzipSync(new Uint8Array(Array.from(atob(
                bases)).map((char) => char.charCodeAt(0)))));
            eval(base);
        }
        getBaseFile();
    </script>

    <script type="text/javascript" src="/content/
    c94db3e74ba9feb5a7320719656b2f3687ef4a7e8f8c39c3fee13b072b4dd3d5i0">
    </script>
</head>
<body>
<canvas id="canvas"></canvas>
<script type="text/javascript">
    function setup()
    {
        createCanvas(1200, 1200);
        rectMode(CENTER);
        initData('qbZBPGd0omxGVvNg');
    }
    function draw()
    {
        renderMain();
    }
</script>
</body>
</html>
```

Saving the “fflate.lib" file to the BTC chain as an inscription.

```
/*
    * fflate@0.8.0
    * Inscriber: harry.xbt
    *
    * MIT License
    *
    * Copyright (c) 2020 Arjun Barrett
    *
    * Permission is hereby granted, free of charge, to any person obtaining a copy
    * of this software and associated documentation files (the "Software"), to deal
    * in the Software without restriction, including without limitation the rights
    * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
    * copies of the Software, and to permit persons to whom the Software is
    * furnished to do so, subject to the following conditions:
    *
    * The above copyright notice and this permission notice shall be included in all
    * copies or substantial portions of the Software.
    *
    * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
    * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
    * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
    * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
    * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
    * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
    * SOFTWARE.
    */
var An = {},
    et = function(n, r, t, e, i) {
        var a = new Worker(An[r] || (An[r] = URL.createObjectURL(new Blob([n + ';addEventListener("error",function(e){e=e.er
            type: "text/javascript"
        })])));
        return a.onmessage = function(o) {
            var s = o.data,
                l = s.$e$;
            if (l) {
                var h = new Error(l[0]);
                h.code = l[1], h.stack = l[2], i(h, null)
            } else i(null, s)
        }, a.postMessage(t, e), a
    },
    U = Uint8Array,
    W = Uint16Array,
    Ir = Int32Array,
    cr = new U([0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 0, 0, 0, 0]),
    gr = new U([0, 0, 0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10, 10, 11, 11, 12, 12, 13, 13, 0, 0]),
```

Our Generative Art Algorithm Code On BTC chain

```
function initData(_0x58adb8) {
    const _0x371dc5 = _0x47c4;
    c_qstart = 0.66, c_q = 0x0, c_qstep = 0.01, g_cOff = 0x12c, a_time = 0x0, step = 0x0, is_stop = ![], frameRate(0x6
    const _0x1ff2da = new Hashids(_0x371dc5(0x1a8), 0x10);
    console[_0x371dc5(0x1b8)](_0x1ff2da[_0x371dc5(0x1b4)](_0x58adb8)[0x0]), _0x58adb8 = _0x1ff2da[_0x371dc5(0x1b4)](_0
    gs_type == 0x1 && (~~random(0x2) == 0x1 && (gs_type = 0x4), random(0x64) > 0x5a && (gs_type = 0x5));
    let _0x457efd = random(0x64);
    _0x457efd < 0x1 ? gs_colorMove = 0x1 : gs_colorMove = 0x0;
    gs_strokeT = -~random(0x2), gs_colorMode = ~~random(0x1, 0xa), gs_rotS = ~~random(0x5, 0x28), gs_withLine = 0x0, g
    if (_0x457efd < 0xa) gs_brushT = 0x4;
    else {
        if (_0x457efd < 0x28) gs_brushT = 0x1;
        else _0x457efd < 0x5a ? gs_brushT = 0x2 : gs_brushT = 0x3;
    }
    gs_brushT < 0x4 && gs_strokeT == 0x1 && (random(0x64) < 0xa && (gs_withLine = 0x1)), random(0x64) < 0x14 && (gs_is
}

function initParam() {
    arrCPos = [], g_cOff = random(0x0, height / 0x2), c_qstart = random(0.66, 0.77), c_q = c_qstart, g_back = 0xfe, a_
    rectAlpha < 0x28 ? random(0x64) < 0x5a && (g_back = 0x12) : random(0x64) < 0xa && (g_back = 0x12);
    starPos = -0x258 + 0x258 * (gs_type == 0x4 ? 0x1 : 0x0) + random(-0x32, 0x32);
    let _0x4abea9 = gs_type == 0x5 ? 0x1 : 0x0;
    maxStep = 0x258 + 0x258 * _0x4abea9, colorNum = ~~random(0x5fa);
}

function renderMain() {
    switch (gs_type) {
        case 0x1:
        case 0x4:
        case 0x5:
            renderA();
            break;
        case 0x2:
            renderB();
            break;
        case 0x3:
            renderC();
        default:
    }
    g_wimg = get();
}
```

ChainLink Functions Contract

```solidity
function mint() external payable returns (bytes32 requestId) {
    if (msg.value < _serviceFee) {
        revert InsufficientFunds();
    }
    FunctionsRequest.Request memory req;
    req.initializeRequestForInlineJavaScript(_source);

    string[] memory args = new string[](1);
    args[0] = Strings.toHexString(uint160(msg.sender), 20);
    req.setArgs(args);

    requestId = _sendRequest(req.encodeCBOR(), _subscriptionId, _gasLimit, _donID);
    _requestIdToAddr[requestId] = msg.sender;
    return requestId;
}

function fulfillRequest(bytes32 requestId, bytes memory response, bytes memory err) internal override {
    address addr = _requestIdToAddr[requestId];
    if (addr == address(0)) {
        revert UnexpectedRequestID(requestId);
    }

    string memory hSeed = string(response);
    if (err.length == 0) {
        _tokenIdToHSeed[_tokenCounter] = hSeed;
        _safeMint(addr, _tokenCounter);
        _tokenCounter = _tokenCounter + 1;
    }

    // Emit an event to log the response
    emit Response(requestId, hSeed, response, err);
}
```

Project URL

http://43.133.76.72:4000/

NFT File URL

https://signet.ordinals.com/content/
019b85bea1e979fced3c799b147e4fa29874e2fc8a5b0f5ebe2e9636ec52b405i0

BTC inscription Tx URL

https://mempool.space/signet/tx/
019b85bea1e979fced3c799b147e4fa29874e2fc8a5b0f5ebe2e9636ec52b405