Department of Information Engineering and
Computer Science
(DISI)
University of Trento, Italy

Blockchain - AY 2023/2024

Project Report

Emanuele Civini [247198] Federico Nardelli [248334] Jacopo Manenti [247279]

Contents

1 Mission

Our mission is to empower small record companies by providing a decentralized platform that enables them to quickly monetize their emerging artists' music rights by selling them to fans and investors. We aim to create a fair and transparent marketplace leveraging blockchain technology to ensure record companies gain liquidity while offering fans and investors unique and valuable NFTs (album rights) linked to their favorite artist.

2 Market Overview

2.1 The role of Record Labels

Record labels play a crucial role in the music industry. They are responsible for discovering, signing, and promoting artists, as well as managing the production, distribution, and marketing of their music. In exchange for these services, record labels take a percentage of the revenue generated by the artists they represent.

Record labels also provide artists with access to resources and connections that can help propel their careers to new heights. Additionally, they often assist with negotiating contracts and licensing deals, ensuring that artists are fairly compensated for their work.

The music industry has undergone significant changes over the years. In the past, record labels relied heavily on physical album sales, but with the advent of digital technology, the landscape has shifted. Online platforms and streaming services have become the primary means of consuming music, presenting both challenges and opportunities for record labels.

2.2 Revenues Channels

With the rise of these channels, record labels now have new avenues to generate revenue. Below is a list of potential revenue channels (not exhaustive):

- **Album Sales and Digital Downloads**: Physical album sales have declined but still contribute to revenue, along with digital downloads.
- **Streaming Services and Royalties**: Streaming services like Spotify, Apple Music, and Tidal dominate music consumption, with record labels earning revenue through royalties. However, per-stream payouts are much lower than album sales, creating challenges in generating substantial income.
- **Merchandising and Licensing**: Record labels maximize revenue through merchandising and licensing deals, selling artist merchandise and licensing music for commercials, movies, and TV shows. These strategies However, to effectively implement this strategy, the artist needs to already be popular, making these operations more profitable.

2.3 The struggle

2.3.1 For labels

Despite the many available solutions, today's record labels are grappling with significant financial challenges, particularly the struggle to generate stable revenue. They face rising production and promotion costs while contending with shrinking margins from music sales.

This pressure is further exacerbated by the dominance of a few industry giants, making it difficult for smaller labels to penetrate the market and retain key artists. These major players often lure away talent with their superior resources. While there are occasional stories of artists being unexpectedly "discovered" by a small label and skyrocketing to fame, such cases are rare exceptions.

2.3.2 For artists

For an artist today, connecting with a major record company is often crucial for reaching a broad audience. However, the contractual terms offered by these big labels are frequently unfavorable, placing artists in difficult situations. In contrast, smaller labels typically offer better and fairer conditions, although they lack the extensive reach and resources of the major players. This dynamic creates an unfair barrier to entry for genuine artists, as the monopolistic practices of the larger labels undermine a free and fair market, making the already challenging path to success even more difficult.

2.4 The opportunity

In a landscape dominated by a few major labels, where small record companies struggle to compete and artists often face unfavorable conditions, we have identified a niche opportunity to introduce a new form of financing.

The concept is straightforward: a small record company can sell a portion of its royalty package linked to an album (for example, 10% of Album X by Artist Y), which includes an assigned percentage of future earnings from an emerging artist they choose to invest in. This approach allows the label to convert a non-liquid asset (emerging artist's royalties) into immediate financial resources, which can be reinvested into other projects or marketing efforts. On the other side, fans and investors interested in speculative opportunities can purchase these royalties, giving them the potential to benefit from the artist's future success. This creates a win-win situation, providing financial flexibility for the label and a unique investment opportunity for supporters and speculators.

To provide this service, we developed an NFT marketplace where album royalty rights are tokenized. By tokenizing just the royalty rights, the platform is able to streamline and manage the distribution of royalty streams accurately without employing a third party that could take a significant percentage of this revenue stream. The token holders receive regular payments in proportion to their share of the royalty rights.

Our platform is designed to serve several key audiences:

- Small record companies seeking to monetize their music rights.
- Fans and collectors looking for unique and potentially valuable music NFTs.
- Investors aiming to speculate on the future success of artists.

2.4.1 Royalty digression

Traditionally, the royalty rates negotiated in artist contracts directly impact how much revenue record labels make. These rates can vary depending on the artist's popularity, the label's bargaining power, and market conditions. Furthermore, royalty rates can be structured in various ways, such as a percentage of gross revenue, net revenue, or a combination of both. Record labels aim to strike a balance between providing artists with fair compensation and ensuring their own profitability.

This explanation outlines how the music industry relies on royalties as a primary form of payment for musicians. Here's a breakdown of the key concepts:

1. **Royalties**: These are payments made to the owner of an asset (like an album) for the right to use/own that asset. In the music industry, royalties are generated from the licensing of copyrighted songs/albums and recordings. Royalties are paid out before other financial obligations, such as stockholder dividends, and are typically distributed at regular intervals (e.g., monthly or quarterly).
2. **Types of Royalties**: There are different kinds of royalties depending on how the music is used:
    - **Reproduction Royalties**: Earned from sales or streaming of the music.
    - **Performance Royalties**: Earned whenever the music is played publicly (e.g., on the radio, in bars, or during live performances).
    - **Synchronization (Synch) Royalties**: Earned when music is licensed for use in TV shows, films, commercials, or video games. This is typically a one-time payment.
3. **Stakeholders in Royalties**: Several parties may have a claim to royalties:
    - **Sound Recording**: Bands typically sign contracts with labels that own the Sound Recording Copyright and distribute royalties to band members, producers, and session musicians.
    - **Composition**: Songwriters often sign deals with publishers who manage the copyright and collect royalties. These royalties are usually split 50/50 between the songwriter and the publisher, with multiple songwriters potentially involved, each receiving a share.

Typically, both songwrites and artists assign their rights to a third party for management, instead of attempting to track a song's use and seek payment independently. Song copyrights are typically assigned to a record label.

3 Technology

Our platform is built on the Polygon network to ensure low transaction fees and fast processing times. Key technologies include:

- **Blockchain and Smart Contracts:** We employ three distinct smart contracts:
    - **Factory Contract:** Oversees the deployment of NFTs that represent record companies and manages the associated addresses (e.g., record company managers) that are authorized to establish and operate these entities.
    - **NFT Contract:** Automates the minting process for each artist and album, as well as the distribution of royalties to the rightful holders.

  - **Marketplace Contract:** Facilitates the ownership management of NFTs, including buying, selling, and bidding transactions on the platform.

- **Web3 Integration:** MetaMask is integrated to enable seamless user interaction with the blockchain, ensuring secure and straightforward access to the platform's features.

- **Frameworks:** Our development stack includes:

  - **Foundry:** A comprehensive smart contract development toolchain that handles dependencies, compiles projects, runs tests, deploys contracts, and allows command-line and Solidity script-based interaction with the blockchain.
  - **Next.js, TypeScript, Shadcn:** Utilized for building the user interface, ensuring a responsive, scalable, and modern frontend experience.
  - **Ethers.js:** A JavaScript library that facilitates interaction with the Ethereum blockchain and its ecosystem, making it easier to build and deploy decentralized applications.

- **Pinata Integration:** We use Pinata for storing and managing files on the InterPlanetary File System (IPFS), a decentralized file storage protocol. NFT metadata and images are stored on Pinata, with the resulting CID (Content Identifier) retrieved and stored on the blockchain using the following URL format: `"https://gateway.pinata.cloud/ipfs/${cid}"`. IPFS ensures that files are distributed across a decentralized network of nodes, providing robust security and ensuring that data stored in the network cannot be altered, even in the event of a hacking attempt. This guarantees the immutability of NFT data.

4 Why, When, How

Why: Small record companies often struggle to gain liquidity and fair compensation. Our platform provides a solution by offering a decentralized marketplace that rewards both record companies and fans/investors. The latter will be able to access to royalty streams that were previously available only to industry insiders, private equity, or institutional funds.

When: The project will be rolled out in phases, starting with the beta launch of the platform to onboard initial users and record companies. Full launch is planned after thorough testing and feedback incorporation.

How:

- Record companies will pay a small fee to publish their music rights on the platform (i.e., to be whitelisted).

- Upon publishing, the record company will provide their wallet address. The admin will whitelist this wallet to grant access to the necessary functionalities.

- After an album is minted, the specified number of music rights will be tokenized (e.g., 1000 tokens).

- When an NFT is sold, 5% of the sale price will be paid as royalties to the record company, and 1.5% will go to the platform to cover operational costs.

- NFT holders will benefit from potential appreciation in value if the artist gains popularity. Additionally, NFTs may offer other privileges such as access to exclusive content, presales, discounted concert tickets, and special merchandise.

5 Operation

The platform operates as follows:

- A small record company submits a request to join the platform and is whitelisted by the admin through a dedicated section of the web interface accessible only by the admin ("whitelist" section).
- Upon approval, the record company's wallet is registered in the system, unlocking a section of the web interface where new artists can be added and their albums minted ("mint" section).
- The platform supports login via MetaMask, utilizing the Polygon network (Ethereum Layer 2) for transactions.
- If the wallet belongs to a registered record company, the relevant functionalities for managing collections and minting albums become accessible.
- If the wallet belongs to a regular user, they can browse various collections, apply filters, view album details, and purchase or bid on albums as desired.
- Both record companies and regular users can buy or auction owned or created albums.

6 Value Proposition Canvas

By using the value proposition canvas, we go on to analyze the value proposition, the customer (small record) and the client profile (Retail investor).

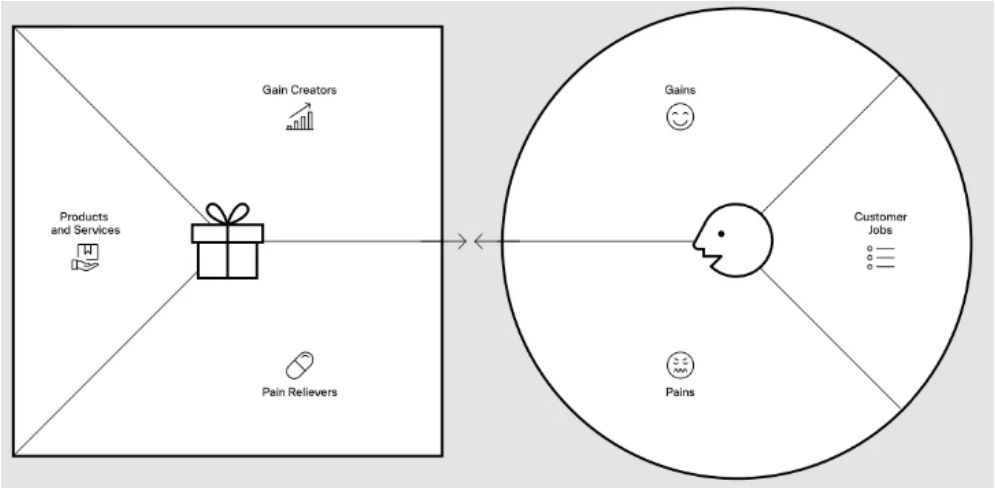

Figure 6.1: Value proposition canvas

6.1 Customer Profile

6.1.1 Customer Jobs

1. **Small records:**

   - **Artist Discovery and Development:** Finding and signing emerging artists with potential, and providing guidance and resources to help develop their talent and marketability.
   - **Music Production and Distribution:** Overseeing the recording process and managing the distribution of music across physical and digital platforms.
   - **Marketing and Promotion:** Creating and executing campaigns to build the artist's brand, increase visibility, and promote their music through various channels.
   - **Monetization of Music Rights:** Managing and monetizing the music rights of artists by securing royalties and exploring additional revenue streams like licensing and merchandising.
   - **Financial Management:** Handling the financial operations of the label, including budgeting, accounting, and ensuring timely payment of royalties and other obligations.
   - **Relationship Management:** Building and maintaining relationships with industry stakeholders, artists, and fans to foster a supportive network around the label.

2. **Retail investor:**

   - **Buying Stock in Record Companies or Publishers:** Invest in publicly traded record companies or music publishers that discover and promote artists.
   - **Investing in Royalty Funds:** Purchase shares in royalty funds that own the rights to music catalogs and distribute royalties to investors.

6.1.2 Gains

1. **Small records:**

   - **Revenue Generation through Music Sales:** Effective production and distribution enable the label to generate revenue from both physical and digital sales, reaching a wide audience.
   - **Brand Visibility and Artist Promotion:** Through strategic marketing and promotion, the label increases the visibility of its artists, leading to higher sales, streaming numbers, and fan engagement.

2. **Retail investor:**

   - **Earning Passive Income from Royalties:** Investing in royalty funds or purchasing music rights directly allows investors to earn passive income through royalty payments, which can provide a steady stream of revenue over time.
   - **Potential for High Returns:** Although risky, investing in the right artist or music rights could lead to significant returns, especially if the artist gains popularity or the value of the music catalog appreciates over time.

6.1.3 Pains

1. **Small record:**

   - **Limited Financial Resources:** Small music labels often struggle with limited financial resources, making it difficult to invest in talent development, marketing, and production at the same level as larger competitors.

- **High Operational Costs:** The costs associated with producing, promoting, and distributing music can be high, putting pressure on the label's financial stability.
- **Challenges in Artist Retention:** Without substantial financial backing, small labels may find it difficult to retain successful artists who are often lured away by larger labels offering better contracts.
- **Limited Market Reach:** Small labels may have limited access to broad distribution networks and marketing channels, hindering their ability to reach larger audiences and grow their brand.

2. **Retail investor:**

- **Limited Direct Investment Options:** Buying stock in record companies or publishers provides only indirect exposure to specific artists, which may not satisfy investors looking for a more direct connection to the music they love.
- **Access Restrictions for Royalty Funds:** Many royalty funds are private and only available to accredited investors, limiting access for retail investors who may not meet the criteria to participate in these funds.
- **High Costs and Barriers to Entry:** Purchasing rights directly to songs, albums, or catalogs can be prohibitively expensive for most retail investors, especially when considering high-profile catalogs that sell for large sums.

6.2 Value Proposition

6.2.1 Gain Creators

1. **Small record:**

- **Liquidity:** swiftly monetize the investments in emerging artists by selling or auctioning music rights, providing immediate cash flow.
- **Alternative Financing Options:** quick access to alternative financing, contributing to the stabilization of revenue streams and enabling reinvestment into new projects.
- **Increased Autonomy through Decentralization:** Eliminates the need for intermediaries, ensuring that record companies retain more control and receive fair compensation for their artists' work.
- **Enhanced Market Reach:** Allows record companies to reach a global audience, expanding their potential market and increasing the visibility of their artists.

2. **Retail investor:**

- **Access to Diverse Investment Opportunities:** Providing a range of investment options, such as different genres, artists, and types of rights, allows retail investors to diversify their portfolios according to their preferences and risk tolerance.
- **Transparency and Trust:** Utilizing blockchain technology to ensure transparency in transactions can build trust with retail investors.
- **Portfolio Diversification:** Providing diverse investment options in the music industry enables retail investors to diversify their portfolios with assets that are uncorrelated with traditional markets, potentially reducing overall investment risk.
- **Passive Income Generation:** Structuring investments to provide regular royalty payments allows retail investors to earn passive income from their holdings, enhancing their overall financial returns.

6.2.2 Pain Relievers

1. **Small record:**

   - **Risk Mitigation:** Diversifying revenue streams through the sale or licensing of music rights can reduce financial risk and provide more stable income for record companies.
   - **Access to Alternative and Cheaper Financing:** Selling or auctioning music rights can alleviate financial pressure and reduce dependence on traditional loans or credit lines.
   - **Simplified Rights Management:** Utilizing decentralized platforms and blockchain technology allows record companies to lower transaction fees, simplify the management and distribution of music rights and royalties, reduce administrative burdens, and eliminate the need for intermediaries, thereby retaining more revenue from their music rights.
   - **Enhanced Artist Retention:** Providing flexible monetization strategies and fair compensation can help record companies retain their artists by offering competitive deals that meet the needs of both parties.
   - **Increased Market Reach:** Leveraging platforms that expand the global distribution and promotion of music can help small record companies reach wider audiences and grow their market presence.

2. **Retail investor:**

   - **Lower Barriers to Entry:** Providing access to fractional ownership of music rights or smaller-scale investments allows retail investors to participate in the music industry without needing large sums of capital.
   - **Simplified Investment Process:** Offering a user-friendly platform with clear information and streamlined processes makes it easier for retail investors to navigate the complexities of investing in music rights.

6.2.3 Products and Services

- **Tokenized Music Royalties:** Unique digital assets that represent fractional ownership of music rights, allowing both retail investors and record companies to easily buy, sell, and trade music royalties on the blockchain.
- **Record Company Profiles:** Customizable and interactive profiles that enable record companies to showcase their artists, music catalogs, and upcoming releases, while directly connecting with fans and potential investors.
- **Decentralized Marketplace Platform:** A user-friendly, decentralized platform built on the Polygon network, designed for the secure buying, selling, and trading of music NFTs and royalty rights, with low transaction fees and high transparency.
- **Automated Royalty Management:** Smart contract-powered automation of royalty calculation, distributions, buy and sell, ensuring accurate and timely payments to rights holders, without the need for intermediaries.
- **Investment Analytics and Insights:** Tools and resources that provide investors with detailed analytics, market trends, and insights into the potential performance of their music rights investments, helping them make informed decisions.

7 Our Solution

7.1 Project Environment Setup

To set up the project environment, you can clone the GitHub repository: `https://github.com/jacopomanenti01/Blockchain`.

Follow these steps to set up the environment:

1. Clone the repository:

```
git clone https://github.com/jacopomanenti01/Blockchain
```

2. Navigate to the project directory:

```
cd Blockchain
```

3. Change the script to executable, if is not, before launching:

```
chmod +x setup.sh
./setup.sh
```

This will install all the necessary node modules and other dependencies. Once finished, the environment is set up.

To run the server, use the following command:

```
npm run dev
```

It will start locally on: `http://localhost:3000`

7.2 Platform Features

After setting up and running the server, you can see the front-end landing page. Users can log in with MetaMask on the Polygon Amoy testnet. If your address is already whitelisted as an admin, you can access the Whitelist section at the top of the page. Here, you can deploy NFTs for record label companies with the following details:

- **Record Name**: The name of the record label company.
- **Record Address**: The address associated to interact by adding artists and albums.
- **Record Treasury**: The address of the treasury wallet (likely a cold wallet) of the record label company.

Only one NFT can be deployed for each record address to represent the record label company. The JSON file is uploaded using the IPFS system with Pinata to store and retrieve the metadata. If the transaction is successful, it is saved; otherwise, the file is deleted.

Once an NFT for a record company is deployed, they have granted access to minting artists and albums. Before minting albums, it is mandatory to add the artist owned by the record label company. The artist has:

- **Artist's Name**
- **Description**
- **Genre**

The JSON file is uploaded to Pinata, and the transaction is reverted if unsuccessful.
To mint an album, the following parameters must be entered in the form:

- **Title**
- **Year**
- **Songs**
- **Singer Associated**
- **Royalties**

The JSON file is uploaded to Pinata, and the transaction is reverted if unsuccessful. All transactions can be seen on the block explorer of the Amoy Polygon testnet: `https://amoy.polygonscan.com/`

7.3 Smart Contracts

Our platform utilizes smart contracts to manage the creation, sale, and royalties of music NFTs. Below is an overview of the three main smart contracts used in our system: NFTFactory, NFT, and Marketplace.

7.3.1 NFTFactory.sol

The NFTFactory contract is responsible for deploying new NFT contracts for record companies and managing the association between admins and their respective NFT contracts.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "openzeppelin-contracts/contracts/access/AccessControl.sol";
import "./interfaces/INFTFactory.sol";
import "./NFT.sol";

contract NFTFactory is AccessControl, INFTFactory {

    mapping (address => bool) public isFactoryDeployed;
    mapping (address => address) public associatedNFT;

    constructor() {
        _grantRole(DEFAULT_ADMIN_ROLE, msg.sender);
    }

    function deployNFT(string memory _name, address _recordCompanyAdmin, address _treasury,
        NFT nft = new NFT(_name, _recordCompanyAdmin, _treasury, _initialFee);
        isFactoryDeployed[address(nft)] = true;
        associatedNFT[_recordCompanyAdmin] = address(nft);
        emit NewNFT(address(nft), _recordCompanyAdmin);
```

```
    }

    function setAssociatedNFT(address _admin, address _nft) external {
        require(isFactoryDeployed[msg.sender], "NFT is not factory deployed");
        require(_nft == msg.sender || _nft == address(0), "Invalid NFT");

        if (_nft != address(0)) {
            require(associatedNFT[_admin] == address(0), "Admin already has associated NFT");
        }
        associatedNFT[_admin] = _nft;
    }
}
```

7.3.2 NFT.sol

The NFT contract represents the NFTs managed by a record company, including functionality for creating singers and albums, and updating record company fees.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "openzeppelin-contracts/contracts/access/AccessControl.sol";
import "openzeppelin-contracts/contracts/token/ERC1155/extensions/ERC1155Supply.sol";
import "./interfaces/INFT.sol";
import "./interfaces/INFTFactory.sol";

contract NFT is ERC1155Supply, AccessControl, INFT {
    struct Singer {
        string stageName;
        string description;
        string genre;
        string imageUrl;
        bool exists;
    }

    struct Album {
        uint256 shareCount;
        uint256 singerId;
        string metadataUrl;
    }

    bytes32 public constant RECORD_COMPANY_ROLE = keccak256("RECORD_COMPANY_ROLE");

    mapping(uint256 => Singer) public singers;
    mapping(uint256 => Album) public albums;
    mapping(string => bool) private singerExists;

    uint256 public singerIdCounter;
    uint256 public albumIdCounter;
    uint256 public recordCompanyFee;
    string public name;
    address public treasury;
    INFTFactory public factory;
```

```
constructor(string memory _name, address _recordCompanyAdmin, address _treasury, uint _
    _grantRole(DEFAULT_ADMIN_ROLE, tx.origin); // tx.origin since factory is deploying
    _grantRole(RECORD_COMPANY_ROLE, _recordCompanyAdmin);
    _setRoleAdmin(RECORD_COMPANY_ROLE, RECORD_COMPANY_ROLE); // Only record company can
    treasury = _treasury;
    name = _name;
    recordCompanyFee = _initialFee;
    factory = INFTFactory(msg.sender);
}

function createSinger(string memory _stageName, string memory _description, string memo
    require(!singerExists[_stageName], "Singer already exists");
    Singer storage singer = singers[singerIdCounter];
    singer.stageName = _stageName;
    singer.description = _description;
    singer.genre = _genre;
    singer.imageUrl = _imageUrl;
    singer.exists = true;
    singerIdCounter++;
}

function createAlbum(uint256 _shareCount, uint256 _singerId, string memory _metadataUrl
    require(singers[_singerId].exists, "Singer does not exist");
    Album storage album = albums[albumIdCounter];
    album.metadataUrl = _metadataUrl;
    album.singerId = _singerId;
    album.shareCount = _shareCount;
    _mint(msg.sender, albumIdCounter, _shareCount, "");
    albumIdCounter++;
}

function updateRecordCompanyFee(uint256 _newFee) external onlyRole(DEFAULT_ADMIN_ROLE)
    recordCompanyFee = _newFee;
}

function updateRecordTreasury(address _treasury) external onlyRole(RECORD_COMPANY_ROLE)
    treasury = _treasury;
}

function uri(uint256 _id) public view override returns (string memory) {
    require(exists(_id), "Nonexistent token");
    return albums[_id].metadataUrl;
}

function supportsInterface(bytes4 _interfaceId) public view virtual override(AccessCont
   return super.supportsInterface(_interfaceId);
}

function grantRole(bytes32 _role, address _account) public override onlyRole(getRoleAdm
    _grantRole(_role, _account);
```

```
        if (_role == RECORD_COMPANY_ROLE) {
            factory.setAssociatedNFT(_account, address(this));
        }
    }

    function revokeRole(bytes32 _role, address _account) public override onlyRole(getRoleAdmin
        _revokeRole(_role, _account);
        if (_role == RECORD_COMPANY_ROLE) {
            factory.setAssociatedNFT(_account, address(0));
        }
    }
}
```

7.3.3 Marketplace.sol

The Marketplace contract facilitates the buying and selling of NFTs, including auctions and fixed-price sales. It handles payments, collects fees, and transfers NFTs.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "openzeppelin-contracts/contracts/token/ERC1155/IERC1155.sol";
import "openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol";
import "openzeppelin-contracts/contracts/access/AccessControl.sol";
import "openzeppelin-contracts/contracts/utils/ReentrancyGuard.sol";
import "./interfaces/IMarketplace.sol";
import "./interfaces/INFTFactory.sol";
import "./interfaces/INFT.sol";

contract Marketplace is AccessControl, ReentrancyGuard, IMarketplace {
    using SafeERC20 for IERC20;
    uint public constant PERCENT_DIVIDER = 1000000; // percentage divider, 6 decimals
    address public mpFeesCollector;
    uint public mpFeesPercentage;
    uint public orderCounter;
    uint public auctionCounter;
    INFTFactory public nftFactory;
    mapping(uint => Order) public orders;
    mapping(uint => Auction) public auctions;
    mapping(uint => bool) public isAuction;

    struct Order {
        address paymentToken; // if token is address(0), it means native coin
        uint price; // sell price for single token
        uint amount;
        uint left; // amount of NFTs in the order not yet sold
        uint tokenId;
        address owner; // address that creates the listing
        address collection;  // NFT address
    }

    struct Auction {
        uint auctionId;     // auction ID, starting from 1
```

```
    address paymentToken; // if token is address(0), it means native coin
    uint basePrice;
    uint minIncrement;
    uint deadline;
    uint highestBid;
    uint amount;
    address owner; // address that creates the listing
    address collection;  // NFT address
    address highestBidder;
}

constructor(address _nftFactory, address _mpFeesCollector, uint _mpFeesPercentage) {
    nftFactory = INFTFactory(_nftFactory);
    mpFeesCollector = _mpFeesCollector;
    mpFeesPercentage = _mpFeesPercentage;
    _grantRole(DEFAULT_ADMIN_ROLE, _msgSender());
}

function setNFTFactory(address _newFactory) external onlyRole(DEFAULT_ADMIN_ROLE) {
    nftFactory = INFTFactory(_newFactory);
}

function setNewTreasury(address _newMPFeesCollector) external onlyRole(DEFAULT_ADMIN_RO
    mpFeesCollector = _newMPFeesCollector;
}

function setMarketPlaceFee(uint _newMPFeesPercentage) external onlyRole(DEFAULT_ADMIN_R
    require(_newMPFeesPercentage <= PERCENT_DIVIDER, "Fee over 100%");
    mpFeesPercentage = _newMPFeesPercentage;
    emit NewMPFees(mpFeesPercentage);
}

function createOrder(address _collection, uint _tokenId, uint _amount, uint _price, add
    require(_amount > 0, "Amount must be greater than 0");
    require(nftFactory.isFactoryDeployed(_collection), "Invalid collection");
    IERC1155(_collection).safeTransferFrom(msg.sender, address(this), _tokenId, _amount
    Order storage order = orders[orderCounter];
    order.paymentToken = _paymentToken;
    order.price = _price;
    order.amount = _amount;
    order.tokenId = _tokenId;
    order.owner = msg.sender;
    order.collection = _collection;
    order.left = _amount;
    emit NewOrder(orderCounter, _collection, _tokenId, _amount, _price, msg.sender);
    orderCounter++;
}

function createAuction(address _collection, uint _tokenId, uint _amount, uint _basePric
    require(_amount > 0, "Amount must be greater than 0");
    require(_deadline > block.timestamp, "Deadline must be in the future");
```

```
        require(nftFactory.isFactoryDeployed(_collection), "Invalid collection");
        IERC1155(_collection).safeTransferFrom(msg.sender, address(this), _tokenId, _amount, "
        Auction storage auction = auctions[auctionCounter];
        auction.auctionId = auctionCounter;
        auction.paymentToken = _paymentToken;
        auction.basePrice = _basePrice;
        auction.minIncrement = _minIncrement;
        auction.deadline = _deadline;
        auction.highestBid = 0;
        auction.amount = _amount;
        auction.owner = msg.sender;
        auction.collection = _collection;
        auction.highestBidder = address(0);
        isAuction[auctionCounter] = true;
        emit NewAuction(auctionCounter, _basePrice, _minIncrement, _deadline);
        auctionCounter++;
    }

    function bid(uint _auctionId, uint _amount) external payable nonReentrant {
        require(isAuction[_auctionId], "Order is not an auction");
        Auction storage auction = auctions[_auctionId];
        require(block.timestamp < auction.deadline, "Auction has ended");
        if (auction.paymentToken == address(0)) {
            _amount = msg.value;
        }
        require(_amount >= auction.basePrice, "Bid must be at least base price");
        require(_amount >= auction.highestBid + auction.minIncrement, "Bid increment is too lo
        if (auction.highestBidder != address(0)) {
            if (auction.paymentToken == address(0)) {
                processPaymentETH(auction.highestBidder, auction.highestBid, "Unable to repay
            } else {
                IERC20(auction.paymentToken).safeTransfer(auction.highestBidder, auction.highe
            }
        }
        if (auction.paymentToken != address(0)) {
            IERC20(auction.paymentToken).safeTransferFrom(msg.sender, address(this), _amount);
        }
        auction.highestBid = _amount;
        auction.highestBidder = msg.sender;
        emit NewBid(_auctionId, msg.sender, _amount);
    }

    function endAuction(uint _auctionId) external nonReentrant {
        require(isAuction[_auctionId], "Order is not an auction");
        Auction storage auction = auctions[_auctionId];
        require(block.timestamp >= auction.deadline, "Auction is still ongoing");
        if (auction.highestBid >= auction.basePrice) {
            INFT nft = INFT(auction.collection);
            uint platformFee = (auction.highestBid * mpFeesPercentage) / PERCENT_DIVIDER;
            uint recordCompanyFee = (auction.highestBid * nft.recordCompanyFee()) / PERCENT_DI
            uint sellerAmount = auction.highestBid - platformFee - recordCompanyFee;
```

```
        if (auction.paymentToken == address(0)) {
            processPaymentETH(mpFeesCollector, platformFee, "Unable to pay fees collect
            processPaymentETH(nft.treasury(), recordCompanyFee, "Unable to pay record c
            processPaymentETH(auction.owner, sellerAmount, "Unable to pay seller");
        } else {
            IERC20(auction.paymentToken).safeTransfer(mpFeesCollector, platformFee);
            IERC20(auction.paymentToken).safeTransfer(nft.treasury(), recordCompanyFee)
            IERC20(auction.paymentToken).safeTransfer(auction.owner, sellerAmount);
        }
        IERC1155(auction.collection).safeTransferFrom(address(this), auction.highestBid
    } else {
        IERC1155(auction.collection).safeTransferFrom(address(this), auction.owner, auc
    }
    emit AuctionEnded(_auctionId, auction.highestBidder, auction.highestBid);
}

function buy(uint _orderId, uint _buyAmount) external payable nonReentrant {
    Order storage order = orders[_orderId];
    require(_buyAmount > 0, "Invalid amount");
    require(_buyAmount <= order.left, "Not enough tokens to buy");
    INFT nft = INFT(order.collection);
    uint totalPrice = order.price * _buyAmount;
    uint platformFee = (totalPrice * mpFeesPercentage) / PERCENT_DIVIDER;
    uint recordCompanyFee = (totalPrice * nft.recordCompanyFee()) / PERCENT_DIVIDER;
    uint sellerAmount = totalPrice - platformFee - recordCompanyFee;
    if (order.paymentToken == address(0)) {
        require(msg.value >= totalPrice, "Insufficient funds");
        processPaymentETH(mpFeesCollector, platformFee, "Unable to pay fees collector")
        processPaymentETH(nft.treasury(), recordCompanyFee, "Unable to pay record compa
        processPaymentETH(order.owner, sellerAmount, "Unable to pay seller");
    } else {
        IERC20(order.paymentToken).safeTransferFrom(msg.sender, address(this), totalPri
        IERC20(order.paymentToken).safeTransfer(mpFeesCollector, platformFee);
        IERC20(order.paymentToken).safeTransfer(nft.treasury(), recordCompanyFee);
        IERC20(order.paymentToken).safeTransfer(order.owner, sellerAmount);
    }
    IERC1155(order.collection).safeTransferFrom(address(this), msg.sender, order.tokenI
    order.left -= _buyAmount;
    emit OrderFilled(_orderId, msg.sender, _buyAmount);
}

function cancel(uint _id) external {
    require(_id < orderCounter, "Invalid order id");
    Order storage order = orders[_id];
    require(order.owner == msg.sender, "Not token owner");
    IERC1155(order.collection).safeTransferFrom(address(this), msg.sender, order.tokenI
    order.left = 0;
    emit OrderCancelled(_id);
}

function processPaymentETH(address _to, uint _amount, string memory _error) internal {
```

```
        (bool success, ) = address(_to).call{value: _amount}("");
        require(success, _error);
    }

    function onERC1155Received(address, address, uint, uint, bytes calldata) external pure ret
        return bytes4(keccak256("onERC1155Received(address,address,uint256,uint256,bytes)"));
    }
}
```

Our platform will deploy the NFTFactory contract, granting admin roles to whitelist new record label companies. These companies can interact and participate on our marketplace by minting new NFTs of their artists' albums.

7.4 Compilation and Testing

To compile and build the ABIs of the contracts, navigate to the 'contracts' folder and execute:

```
cd contracts/
forge build
```

This will generate an 'out' directory containing the ABIs for the contracts.

Several tests are written to test the coverage of the smart contracts. To run the tests, navigate to the contracts directory and execute:

```
forge test
```

8 Roadmap

Our development roadmap outlines the key milestones and future implementations planned for our platform:

Phase 1: Beta Launch

- Develop and test the smart contract for NFT creation and royalty distribution.
- Launch a beta version of the platform to onboard initial record companies and users.
- Collect feedback to improve user experience and functionality.

Phase 2: Full Launch

- Implement additional features such as record company profiles and fan/investor interaction tools.
- Expand marketing efforts to attract more record companies and users.
- Establish partnerships with music blogs and independent radio stations.

Phase 3: Future Implementations

- Introduce new types of NFTs, such as limited edition releases and exclusive content.
- Develop a mobile application for easier access and engagement.

- Explore integration with other blockchain networks to enhance scalability and reach.
- Offer presale and discounted concert tickets to NFT holders.
- Provide special signed merchandise and other exclusive items for fans and investors.

Bibliography