Music Streaming Decentralized Application JJAX

Jake Grieves, Andrew McWilliam, Xinrui He, Jiayi Zhong
Financial Technology with Data Science
University of Bristol
Bristol, U.K.

I. Introduction

The advent of music streaming platforms has completely transformed the way we purchase and consume music, but it has also given rise to several issues. While it is extremely convenient for music enthusiasts to use these platforms and enjoy music with just a click, high fees and restricted content access prevent some users from availing the service. On the other hand, for music creators, especially for those who are not mainstream, these platforms provide an opportunity to showcase their work to a wider audience. However, uploading music to traditional streaming apps can be a costly and challenging task. Music creators need to use distributing services to upload their music and face issues such as high fees, low income, and copyright concerns.

Our decentralized streaming music platform JJAX attempts to solve these problems with blockchain technology. Our platform is built on Ethereum, utilizing smart contracts, tokenization, and NFTs to provide a more open and transparent environment for all users.

For creators, JJAX offers a fairer and more transparent revenue sharing system. We provide free music file uploading services, enabling original creators to upload their music to our platform, set their own prices, and receive the majority of their earnings in cryptocurrency when other users purchase their music, all without worrying about piracy. Secondary creators can purchase music at a lower price on our platform, remix it, and then upload their creations for listening or resale, with original creators receiving a share of the earnings. Users can easily browse and search for new music on our platform and purchase their favorite tracks at a lower price.

This report will first discuss the background and motivation behind creating JJAX. Then, we'll describe the overall system architecture and user interface design. We'll also delve into the technical details of how JJAX operates, including the use of smart contracts, tokenization, and NFTs. We'll evaluate the effectiveness of JJAX and discuss future directions for improvement. Finally, we'll summarize the development process and the challenges we faced along the way.

In summary, JJAX represents an innovative solution to the problems plaguing traditional music streaming platforms, and we believe it has the potential to disrupt the industry and create a fairer and more sustainable music consumption model.

II. BACKGROUND

With the growth of the internet and the popularity of music, composers and musicians are beginning to have more and more markets to distribute their music, but at the same time the income generated by their works is becoming less and less due to the excessive number of third-party music markets on the internet, and can also be affected by copyright issues for the music.

Listeners dominate the market as there are several different platforms for music to choose from. However, most of these music platforms are centralised, so there are multiple formalities between creator and listener transactions, and significant fees are charged, i.e. the price of the creator's own work plus intermediate fees is the actual price paid by the consumer. In particular, the monopoly of some music platforms on well-known music works has led to high prices, with most of the benefits of this monopoly going to the intermediary platforms, while the profits for creators remain limited, giving rise to a lot of pirated music resources, which further undermines the interests of creators. Moreover, these central organisations collect data on users' behaviour and analyse their habits, which in turn violates consumers' privacy to a certain extent.

As a result, decentralised trading software has emerged. For the music market, decentralised applications allow for direct trading between music creators and users, making it easier for new creators to promote their work, and for users to sell their music at a lower price due to fewer fees. In addition, the decentralised music platform ensures that the music purchased by users is copyrighted, and for fans of specific creators or music lovers, it ensures that the music they purchase is authoritative. Based on current blockchain technology and smart contracts, it is possible to implement this decentralised music platform where the blockchain and smart contracts act as a third-party organisation where users can directly purchase copyrighted music.

The best example of the current development in Dapp is Audius, a platform for decentralised music sharing and streaming protocols, a format that dramatically increases the ability of creators to express themselves freely and to communicate directly with their fans, while ensuring that music creators receive the majority of the revenue from the benefits created by their work. 90% of the platform's revenue will go directly to the artist. Audius is designed to give creators more control over their music, as opposed to other third-party platforms, where only 12% of revenue goes to the artist. By uploading their music to Audius, creators can leave a permanent record of their work, which will be protected by a decentralised network of nodes.

A key difference between Audius and other streaming music

platforms such as Spotify is that it operates in a decentralised way using blockchain technology, such as NFT gating, which allows artists to distribute exclusive content to NFT holders. 2 can ”gate” their content here, so only token holders of the corresponding collection can access it. Audius effectively solves the copyright challenges faced by creators in the music industry, including music rights, royalties and ownership, giving every user the freedom to publish or listen to music, while ensuring the rights of both creators and consumers.
The profitability of Dapps is also limited due to the decentralised model of Dapps which simplifies the transaction model and eliminates many fees. Many Dapps will be profitable on a drawback basis, such as simulated stock market or gambling Dapps where users are required to buy or hold certain assets or tokens to participate. When the price of these assets or tokens changes, the Dapp re-calculates the value of the user's assets and pays the holder a corresponding bonus, from which the Dapp can take a percentage of the bonus as a source of profit. Or it can take a direct commission when users perform trading activities and top-ups. Alternatively, the most straightforward way to monetise an app is to include advertising in the app and receive a fee for doing so. This model is also applicable to music Dapps.
In the music Dapp, users can buy individual tracks or entire albums to access music services and the Dapp can earn revenue from the sale of the music, for example Audius earns a 10fee on each sale. In addition to this Dapp can also combine music related products and services with its music services, for example selling music tickets, selling album or songwriter

III. DESIGN DOCUMENTATION

A. Initial Idea

The initial idea for JJAX, outlined a blockchain Dapp similar to the online record store Bandcamp [3]. In the idea, users would be able to upload and buy digital music, with features existing for the streaming of said music. Adding to this point the implementation of auction-like sales was explored, in which users can bid to buy unique pieces of music.
The idea while sound in nature runs into issues when considering its use within the Ethereum blockchain. Storing files on the blockchain is incredibly expensive and therefore alternate solutions were required, this led to the research of the IPFS system, however, this in turn led to further issues. Without a dedicated node or server, the IFPS system can be incredibly slow which is detrimental to the ”Streaming” ethos. Considering the auctions section of the idea, the project runs into the ”free rider problem”. While the user who won the auction will get a digital file of the ”unique piece of music”, should a third party get access to the file, there is no way to prove ownership and in effect, the first and third-party users have exactly the same product. Therefore users are likely to feel auctions are unjustified and are unlikely to use the product. With these two problems in mind, the group came to the consensus that the project fails to justify its own creation, as we would simply be creating a lesser version of a product that already exists.

B. The Solution

NFTs (Non-Fungible-Tokens) are well explored in the creation and selling of digital art and act as proof of ownership for digital files. While typically the technology exists as a way for artists to ”mint” and sell digital image files, the technology is incredibly general and can be used in conjunction with the initial idea. By combining the original idea with the NFT technology, we pivot into creating a more proof-of-ownership-orientated, music distribution platform.

C. Evaluated Idea

Building from the evaluation of the initial idea, a project idea was drawn up to better combine the research and ideas the group had. This idea outlines a decentralized music distribution app, heavily relying on NFT technology to validate and prove ownership of music.
The main functionality of the app will allow an artist to upload a unique piece of music to the IPFS system, in which a token will be created representing the uploaded content. This token can be exchanged between accounts and auctioned off for a certain amount of Ethereum and will follow the NFT principles. This token will also be associated with only one address and therefore the ownership of the token can be validated through the blockchain.

The main user base of the project is intended to be that of artists and dedicated music fans. While there exist numerous existing music platforms for artists to share their work. The project introduces the novel idea of proof of ownership, therefore allowing the project to operate in an untapped niche. To create music, musicians often iterate upon musical ideas in order to create the end product, however, the initial ideas or demos are often unreleased and therefore aren't monetized. While some artists see some moderate success in releasing these demos alongside album releases, this often leads to critique from the more casual consumers, as the music is often unpolished and or unfinished. Through the ideas proposed, an artist would be able to tokenize and mint a musical demo before being able to sell the product in an auction-like sale. The dedicated music fans would be able to own a unique piece of music from the artist, and the artist would be able to monetize their demos without the risk of tarnishing their legacy by releasing unpolished/unfinished music.

D. NFT Standards

NFTs are a widely used technology within the blockchain and because of this, there exist numerous standards that projects can make use of. Therefore as an important first step, a viable standard should be identified and built upon in order to create the project. As there are so many options a few well-used standards are listed and described below:

- **ERC 20** ”The Token Standard”. The original standard for fungible tokens, an ERC-20 Token acts just like ETH, and

because of this, each token is made equal. While not an NFT standard this standard may be useful to implement voting and or governance tokens within the project. [4]
- **ERC 721** A "Non-Fungible Token Standard", in this standard tokens are unique and distinguishable from one another and each ERC721 token has a distinct identifier (token ID). [5]
- **ERC 1155** The "Multi-Token Standard". This standard is similar to the ERC 721 standard with the caveat that multiples of each token can exist. For example, a user could own x amount of a given NFT and is given the ability to send a specified number of them to another user. [6]
- **ERC 998** "Composable NFTs". Extending upon the ERC 721 standard users are able to own ERC 721 and ERC 20 tokens but users are now given the ability to "merge several NFTs into one NFT". [7]

While a plethora of standards exist within the NFT space, the goal of the project is to create a viable product, and therefore a single standard should be picked and stuck to in order to reduce complexity (and therefore wasted resources) within the project. Out of the ERC standards listed, ERC 721 and ERC 1155 are the obvious picks, offering the functionality outlined in the "evaluated idea" section. While ERC 1155 allows for proof of ownership and in the example of the project, would allow musicians to mint and sell multiple versions of a demo. This somewhat goes against the project idea. The main brief outlines the uniqueness of the music being sold, and therefore by allowing demos to be minted multiple times, this uniqueness is diluted. Along with this, consumers of music will find no added benefits of owning multiples of the same demo. Owning the song 100 times will have the same user experience as owning the song once, and therefore this is likely to fuel a secondary market of song resellers, which goes against the principles of the project. The main goal again is to facilitate the selling of unique music from musicians to dedicated fans. Because of the reasons listed, the ERC 721 "Non-Fungible Token Standard" will be used as the backbone for the project.

E. The ERC 721 Standard

The ERC 721 Standard defines a standard interface for creating and managing unique tokens and will be used as the basis for the project. Every ERC-721 compliant contract must implement the ERC 721 interface, defined by the following functions, as listed on the OpenZeppelin website [8].

- **event Transfer(from,to,tokenId):** Emits when ownership of a NFT changes.
- **event Approval(owner, approved, tokenId)** emits when the approved address for a NFT is changed.
- **event ApprovalForAll(owner, operator, approved)** emits when an operator is enabled or disabled for a owner
- **function balanceOf(owner)** returns a count of all NFTs assigned to a owner
- **function ownerOf(tokenId)** returns the address associated to a given token.
- **function safeTransferFrom(from, to, tokenId)** Transfers ownership of an NFT from one address to another.
- **function safeTransferFrom(from, to, tokenId, data)** Same as the other safe transfer function, but with the added ability to send data to the receiver.
- **function TransferFrom(from, to , tokenId)** Same as safetransfer however if the receiver address is incapable of receiving a token, the token may be permanently lost.
- **function approve(to, tokenId)** can change the approved address for an NFT, i.e. allows the approved to control the specified NFT
- **function setApprovalForAll(operator, _approved)** allows a third party to maange all of the sender's assets.
- **function getApproved(tokenId)** Gets the approved address for a single NFT
- **function isApprovedForAll(owner, operator)** Query if an address is an authorized operator for another address

These functions act as the basis for the ERC 721 standard and therefore the basis of the project. While manual implementation is possible, there exists the open-source library OpenZeppelin which contains implementation for industry standards such as ERC 20, ERC 777 and in the case of the project ERC 721. Therefore through the use of contract inheritance, the project will make use of the pre-defined open zeppelin contracts. The project makes use of 5 open-souce contracts, which are as defined:

- **ERC721.sol:** This smart contract contains basic implementation of the functions listed in the ERC 721 standard, and therefore allows for the creation and distribution of NFTs. The implementation however lacks a lot of the functionality listed in the brief. Within the contract, tokens are non-fungible, however, the ID must be defined by the user, and the tokens have no way of linking to real-world data.
- **Ownable.sol** This contract acts as an extension of the ERC 721 contract and adds the safemint function. This function when called allows approved users to create a new token. The functionality within the contract is incredibly basic and therefore adjustments will need to be made to edit the functionality in line with the end goal.
- **Counters.sol** Again acting as an extension of the ERC 721 smart contract, the smart contract adds enumeration features, which allow for newly minted tokens to be given a unique id.
- **Enumerable.sol** This extension allows for on-chain enumeration of all tokens.
- **URIStorage.sol** This extension edits the safemint function, and the tokens as a whole by now permitting the passing of a IPFS URI. This extension allows the tokens to be linked to a object within the IPFS system and therefore adds the main functionality of the NFT system.

With the inheritance in place, the backbone of the project

```
{ "Attributes": [
{ "trait_type": "Artist",
"value": [ARTIST] },
{ "trait_type": "Genre",
"value": [GENRE] } ],
"description": [DESCRIPTION],
"image": [IMAGE CID],
"mp3": [MP3 CID],
"name": [TITLE] }
```

Listing 1: JSON example

is complete, and work can now begin on the project.

F. IPFS Implementation

The solidity blockchain is both slow and expensive and therefore the storage of the NFT files on chain is out of the question, therefore a workaround is needed. IPFS is a ”modular suite of protocols” that facilitates the organizing and transferring of data, in a decentralized fashion [9], and will be the protocol the project uses for file storage and sharing. The protocol is a decentralized and distributed file-sharing system, that creates a permanent and censor-resistant method for file sharing/storage. The interesting part of the protocol is its use of a content-addressable system in which files are identified and accessed based on content rather than location (such as on a server). IPFS will act as the file storage section of the project and will host the metadata and content of the NFTs.

There exist three components for a song, the mp3 (or file), the album art, and the metadata. In this section, I will describe the process by which the songs will be uploaded and minted as an NFT.

1) Through the use of a third-party API, the song’s mp3 data and album art are uploaded to the IPFS system. The CID for both are returned and stored as variables.
2) A JSON object is created, containing the CID for both the mp3 and the album art, along with any relevant metadata. e.g. Title of the song, genre, ect.
3) The JSON object is uploaded to the IPFS system and the CID is returned and stored as a variable.
4) The SafeMint function is called and the CID of the JSON function is passed as a parameter.
5) The NFT is now created and can be shared and sold at will.

Every JSON file used within the project will follow the predefined structure, see Listing: 1. Which allows a given NFT to contain the files necessary for the end product.

IV. Project Execution

V. Conclusion

References

[1]
[2]
[3] https://bandcamp.com/about
[4] https://ethereum.org/en/developers/docs/standards/tokens/erc-20/
[5] https://eips.ethereum.org/EIPS/eip-721
[6] https://ethereum.org/en/developers/docs/standards/tokens/erc-1155/
[7] https://eips.ethereum.org/EIPS/eip-998
[8] https://docs.openzeppelin.com/contracts/2.x/api/token/erc721
[9] https://docs.ipfs.tech/concepts/what-is-ipfs/defining-ipfs