

TokenFuse: A Versatile NFT Marketplace

Ch Sree Kumar[1,2], Akshya Kumar Lenka[2], Surukati Asutosh Dora[2], Ashish Sharma[3], Ugo Fiore[4(✉)], and Diptendu Sinha Roy[1]

[1] National Institute of Technology Meghalaya, Shillong, Meghalaya, India
diptendu.sr@nitm.ac.in
[2] NIST Institute of Science and Technology (Autonomous), Berhampur, Berhampur, Odisha, India
{akshyalenka.ele.2019,surukatidora.ece.2019}@nist.edu
[3] Institute of Engineering and Technology, GLA University, Mathura, U.P., India
ashishsharma@gla.ac.in
[4] Department of Computer Science, University of Salerno, Fisciano, Italy
ufiore@unisa.it

Abstract. An NFT (non-fungible token) marketplace is a platform where people can buy and sell one-of-a-kind digital items such as art work, music, and videos. These items are represented by NFTs, which are not exchangeable for other tokens or assets and are stored on a blockchain. The growing use of blockchain technology and the desire for individuals to own and trade rare digital items have fueled the growth of NFT marketplaces. The purpose of this paper is to demonstrate the creation and working of an NFT marketplace named "TokenFuse" and discuss its features as well as the background technologies that keep it running. Our NFT marketplace uses the power of Inter Planetary File System (IPFS) Technology to provide artists with a unique and secure way to sell their original works as NFTs. In contrast to traditional marketplaces that rely on centralized servers and databases, our platform stores and distributes NFTs in a decentralized and peer-to-peer network, ensuring security and redundancy. It also provides a unique and cost-effective way for authors to monetize their works by taking advantage of blockchain technology. Our platform has extremely low fees, making it one of the most affordable options on the market.

Keywords: NFT · Ethereum · IPFS · Solidity · Blockchain

1 Introduction

NFTs (non-fungible tokens) are a mechanism to manage and control digital scarcity so that businesses based on creative work can be created and sustained. Beyond digital art, which NFTs are known for, they can support ownership of anything that is unique. This encompasses a variety of things, from event tickets to real estate and intellectual property rights. NFTs create records of ownership that are stored on a blockchain. Compared to traditional records of ownership, NFTs offer several important characteristics. Authenticity, for example, can be ensured by cryptographic techniques. Transfers of ownership are simple and direct because transactions are recorded on the blockchain. A

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14109, pp. 179–192, 2023.
https://doi.org/10.1007/978-3-031-37120-2_12

consequence is that the legitimacy of ownership can also be verified by looking at the history of the NFT on the blockchain. Smart contracts allow selective, controlled, and reliable transfers of ownership.

In an NFT marketplace, margins are considerably affected by transaction costs. In particular, since NFTs can be conceived as having a very small value, transaction costs could eat up a significant proportion of revenues, up to the point at which trading becomes uneconomical.

Building a decentralized and price-competitive marketplace requires the careful selection and integration of multiple tools organized in a coherent architecture. This paper describes the lessons learned in designing and building TokenFuse, a blockchain-based NFT marketplace that provides creators and collectors with a safe and dependable platform for trading and managing digital assets. TokenFuse is a marketplace for NFTs that aims to make it easy for creators to mint, list, and sell their digital assets and for collectors to discover, buy, and own them. TokenFuse is built on the Ethereum blockchain [1], which provides a decentralized platform for the creation and transfer of NFTs. This allows for a transparent and secure marketplace where transactions are recorded on an immutable ledger accessible to anyone. TokenFuse also utilizes smart contract [2, 3, 4] technology, which enables automatic and self-executing transactions without the need for intermediaries.

Section 2, Background, explains NFTs trading using them and their relationship to blockchain technology with use cases. Section 3 deals with building a versatile marketplace and covers TokenFuse's architecture, workings, tools used and monetization strategies. Section 4 is about the implementation, which includes screenshots and code snippets to create TokenFuse. Finally, the conclusion section summarizes the main points and emphasizes TokenFuse's potential impact on the NFT market.

2 Background

This Section provides an overview of NFT and how it connects to real blockchain applications.

2.1 NFT for Trading Using Blockchain Use Cases

Blockchain technology is used to create non-fungible tokens (NFTs), allowing for their safe and open purchase, sale, and trading [5]. NFTs are generally considered tokens that are saved on a blockchain, so it is very difficult to change the ownership record or tamper with it. Each NFT is unique and stored on the blockchain, making it simple to confirm that someone is the rightful owner of a particular object [6]. This is crucial for expensive objects that have a high potential for fraud or counterfeiting, such as art and collectibles. Blockchain technology not only offers security and verification but also makes it simple to buy, sell, and trade non-fungible tokens on online exchanges. This enables the development of a liquid market for rare items, making it simple for buyers and sellers to find and conduct business with one another. An art business can thus grow and scale, increasing the revenue generated from creative work [7].

The next section elaborates on the process of building a versatile marketplace for trading NFTs.

2.2 NFT and Trading Using Them

The process of creating and using NFTs involves several steps [8]:

1) Creation: Using blockchain, a creator or artist develops a digital asset, such as a work of art or a collectible, and gives it a special digital signature. The asset has this signature as confirmation of ownership and authenticity.
2) Minting: By uploading the digital asset to a blockchain platform like Ethereum and constructing an NFT utilizing smart contract technology, the creator or artist "mints" the digital asset. This generates a special token that identifies the owner of the digital asset and is stored on the blockchain.
3) Purchasing and selling: The NFT can be purchased and sold in a marketplace that focuses on NFTs, such as TokenFuse. An NFT is not the actual digital asset that is purchased; rather, it is the special token that signifies ownership of the asset. The asset may reside on the blockchain or elsewhere.
4) Storage: The NFT metadata is kept on the blockchain, and the purchaser can view and use the digital asset by utilising the special token.
5) Trading: The NFT can be exchanged for other NFTs on the open market; the value of each NFT depends on demand and rarity.

The next section will discuss the architectural and technological choices in the implementation of TokenFuse.

3 Building TokenFuse

TokenFuse is a blockchain platform that enables minting, browsing, and listing NFTs, as well as profile management for users. It offers a comprehensive set of features to create and tokenize unique digital assets, purchase them using cryptocurrency, and monetize them. Overall, TokenFuse is a versatile and convenient platform for the NFT community.

3.1 Architecture of TokenFuse

An NFT marketplace's architecture can change depending on the particular platform and its characteristics [9]. However, there are a few elements that are frequently found in an NFT market (Fig. 1):

1) Frontend: The user interface on the front end of an NFT marketplace is what customers use to browse and buy NFTs. ReactJs has been chosen for building the front end of TokenFuse.
2) Backend: The server-side technology that drives an NFT marketplace is referred to as the backend. It might have a database for keeping track of NFTs. IPFS is the selected backend for TokenFuse.
3) Blockchain: The technology that drives NFTs and enables safe and open transactions is the blockchain. TokenFuse runs on the Ethereum blockchain network. For writing smart contracts, TokenFuse uses the Solidity language.
4) Smart contracts: These are self-executing pieces of code that run over a blockchain. In TokenFuse, smart contracts are used for minting, buying, and selling NFTs.

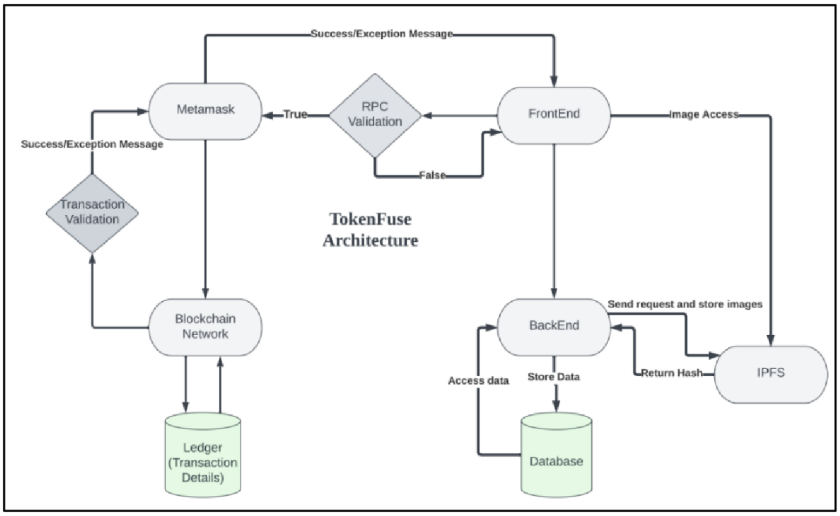

Fig. 1. Architecture of TokenFuse

3.2 Working of TokenFuse

In this section, the main operations within TokenFuse are described.

Connecting a Wallet to TokenFuse: To purchase an NFT, a user must connect their wallet to the NFT marketplace. Web3js and metamask [10] are used to connect the user's wallet to the marketplace platform.

Minting an NFT: Minting an NFT simply means creating NFT for the digital asset. It involves storing the digital asset over IPFs and then updating its metadata to the Blockchain Network. Figure 2 reports the mint function in TokenFuse.

```
function mint(address _to, uint256 _tokenId) public {
        require(!tokenExists[_tokenId]);
        tokenExists[_tokenId] = true;
        tokenOwner[_tokenId] = _to;
        emit Mint(_to, _tokenId);
    }
```

Fig. 2. Solidity code to mint NFT

Sell Function: Minting an NFT automatically triggers the sell functionality i.e., once an NFT is minted it is ready to be sold. Below is the sell function used in the marketplace (Fig. 3).

```
function sell(uint256 _tokenId, uint256 _price) public {
        require(tokenExists[_tokenId]);
        require(tokenOwner[_tokenId] == msg.sender);
        require(_price > 0);

        tokenPrice[_tokenId] = _price;

        emit Sell(msg.sender, _tokenId, _price);
    }
```

Fig. 3. Solidity code to Sell NFT

Buy Function: For buying an NFT from TokenFuse, a user needs to have sufficient eth in his/her account. He/She then chooses a particular NFT from the homepage and, after clicking the buy button, a Metamask prompt will appear to confirm the transaction. The buy function below is used in TokenFuse (Fig. 4).

```
function buy(uint256 _tokenId) public payable {
        require(tokenExists[_tokenId]);
        require(tokenOwner[_tokenId] != address(0));
        require(msg.value >= tokenPrice[_tokenId]);

        address _seller = tokenOwner[_tokenId];
        uint256 _price = tokenPrice[_tokenId];

        tokenOwner[_tokenId] = msg.sender;
        _seller.transfer(_price);

        emit Buy(msg.sender, _tokenId, _price);
    }
```

Fig. 4. Solidity code to Buy NFT

3.3 Tools Used in TokenFuse

Alchemy, IPFS, Hardhat, and OpenZeppelin are just a few of the tools and technologies that were used to build TokenFuse. Alchemy supplies the platform's infrastructure, ensuring dependable connectivity to the Ethereum network. NFT metadata is stored in a decentralized way via IPFS, simplifying retrieval and sharing. The NFT contracts on the platform can be created using Hardhat, a development environment that enables developers to build, test, and deploy smart contracts. To reduce development time and guarantee security, OpenZeppelin offers a library of reusable smart contracts that have been audited and secured. Together, these resources make up the foundation of the TokenFuse NFT market, offering a dependable and secure platform for the production, marketing, and administration of NFTs.

IPFS

IPFS (InterPlanetary File System) is a distributed file system that allows for content-addressable, peer-to-peer storage and retrieval of files using a unique cryptographic hash of their contents, with a built-in mechanism for caching, replication, and resiliency [11].

The process of storing data in IPFS is as follows:

1) Smaller pieces of data are created by subdivision. Each block is a standalone data unit.
2) Each block is used to produce a distinct cryptographic hash.
3) The hashes are then used to locate and identify the blocks, which are then stored on the IPFS network.
4) A client asks for the root block's hash when it wants to retrieve a file.
5) The root block and any other blocks that it is made up of are located by the IPFS network via a distributed hash table (DHT).
6) Blocks are then put back together and provided to the customer.
7) To increase resilience and fault tolerance, IPFS incorporates a concept known as "Content addressability," which states that data is stored and retrieved based on its content rather than its location.
8) Data replication and caching are also incorporated into IPFS, so when a block is requested numerous times, it is cached on adjacent nodes for quicker retrieval.

Hardhat

Hardhat is an open-source Ethereum development environment that is used for testing and debugging smart contracts before deploying to a live blockchain network [12]. It makes it simple for developers to test and debug their contracts on a local machine during the development process.

Developers can compile, test and deploy their contracts locally on Hardhat's built-in "development network", a small blockchain that resembles the Ethereum network. It also comes with a set of testing and debugging tools, including a JavaScript console, commands for contract deployment and maintenance, and support for other development frameworks like Truffle.

Its major objective is to increase smart contract development's efficiency and reduce error-proneness. It is especially helpful for teams creating complex decentralized applications (dApps) that require a stable development setting in order to test and debug their contracts before releasing them onto a live network.

Alchemy

Alchemy is a platform that offers a number of tools that help in creating and monitoring decentralized applications (dApps) by offering a collection of APIs, SDKs, and developer tools that can be used to interface with the blockchain and manage NFTs without having to construct everything from scratch [13]. It also offers a few extra tools to help make your dApp more effective and user-friendly, and it can reduce development time while offering a secure and reliable environment.

The Alchemy platform's user-friendly interface makes it simple to mint, manage, and trade NFTs. Developers may mint NFTs, set their metadata, and post them for sale on the marketplace thanks to the platform's set of APIs and SDKs. The Alchemy platform also has an interaction with MetaMask, which enables users to quickly link their wallet to the market and conduct transactions. Additionally, the platform offers analytics and

monitoring capabilities to assist developers in understanding user behaviour, identifying usage trends, and identifying any problems with their NFT marketplace.

OpenZeppelin
On the Ethereum blockchain, OpenZeppelin [14] is an open-source platform for creating smart contracts. It offers a set of reusable, safe, and proven smart contract components for developers to use in their applications. These components include libraries for common contract types like multi-sig wallets and access control, as well as libraries for token standards like ERC20 and ERC721.

OpenZeppelin assists programmers in creating smart contracts that are more secure, as the libraries and contract patterns have been strictly reviewed by security professionals. Developers can reduce the chance of adding security flaws to their contracts by utilizing these pre-built components.

Additionally, OpenZeppelin offers a user-friendly API that enables programmers to create typical contract features like token transfers and contract updates without having to write challenging low-level code platform.

Once the marketplace has been built, it is time to focus on how to monetize it.

3.4 Monetizing Opportunities for TokenFuse

There are several ways to monetize an NFT marketplace, including the following:

1) Commission on sales [15]: One of the most popular ways to monetize the NFT marketplace is by taking a commission on each sale.
2) Listing fees: One more approach to make money is to charge creators a listing fee to list NFTs that are open for sale.
3) Advertising: Selling ad space on TokenFuse is another option to make money off the system.

4 Implementation

This Section will showcase code snippets and screenshots of the TokenFuse application to provide a visual representation of its functionalities.

1. Creating the Smart Contract using Solidity
   i. Importing Libraries from OpenZeppelin
   ii. Defining the buy, sell and mint Functions.
   iii. Enabling a counter to keep track of NFT ID.
2. Using Alchemy as a Node
   i. Creating an Alchemy account
   ii. Creating an app and importing the API key
3. Configuring IPFS for storing the NFTs
   i. Using Pinata to upload data to IPFS
   ii. Coping the Pinata key and using it in the.env file
4. Integrating a wallet for Transaction purpose
   i. Connecting Metamask to Goerli Test Network.
   ii. Obtaining some TestEth using the faucet

iii. Pasting the Private Key of the Wallet in.env file

5. Building the Front-end
   i. Creating the Homepage and User Profile Page
   ii. Creating the Mint and Buy page.
   iii. Integrating Front-end with Back-end using EthersJS.

The above points can be elaborated by the following-

1. Creating smart contracts using Solidity involves importing useful libraries from OpenZeppelin to make the development faster and secure. We define functions like buy(), sell() and mint() to facilitate transactions. A counter is required to assign unique IDs to each NFT.
2. Alchemy is a popular node used to deploy the smart contracts on different networks like Goerli testnet. We created an Alchemy account, added an app and imported the API key to connect with Alchemy.
3. IPFS is used to store the NFT data permanently. Pinata makes it easy to upload files to IPFS. We just need to get the Pinata key and add it to the.env file.
4. A wallet is mandatory to conduct transactions on the blockchain. Connecting Metamask wallet to Goerli testnet, obtaining some test ETH using the faucet and adding the private key to.env file will enable transactions.
5. The front-end consists of homepage, user profile page, mint page and buy page. Integrating the front-end and back-end using EthersJS will provide a seamless experience to users.
6. In summary, creating NFTs on blockchain involves smart contract development, choosing a suitable node for deployment, storing NFT data on IPFS and enabling transactions through wallets. The front-end and back-end integration completes the project (Figs. 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17).

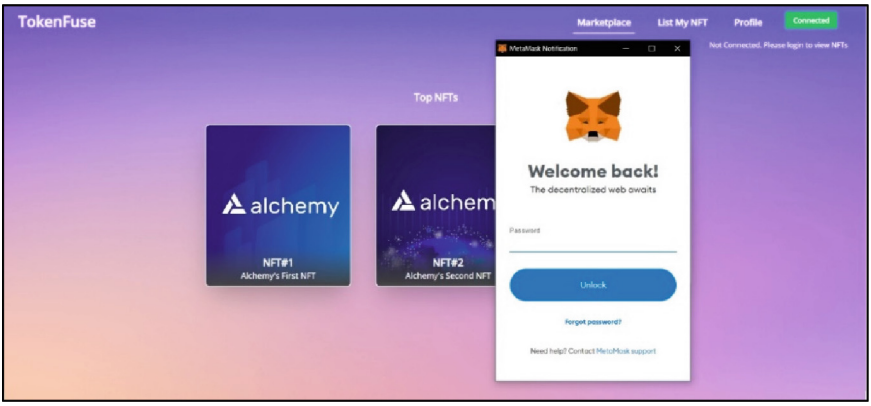

Fig. 5. Connecting Metamask wallet to TokenFuse

```
const connectwalletHandler = () => {
    if (window.ethereum) {
        provider.send("eth_requestAccounts", [])
            .then(async () => {
                await accountChangedHandler(provider.getSigner());
            })
    } else {
        setErrorMessage("Please Install MetaMask!!!");
    }
}
```

Fig. 6. Code to connect wallet to Marketplace



Fig. 7. Home page of TokenFuse

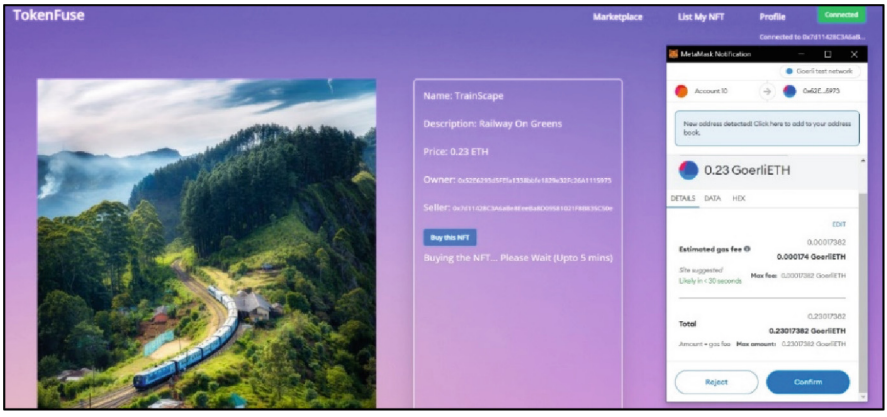

Fig. 8. Minting/ Selling an NFT in TokenFuse

```
function mintAndSellNFT(string memory tokenURI, uint256 price) public {
        _tokenIds.increment();
        uint256 newTokenId = _tokenIds.current();
        _mint(msg.sender, newTokenId);
        _setTokenURI(newTokenId, tokenURI);
        tokenIdToPrice[newTokenId] = price;
    }
```

Fig. 9. Solidity Function to Mint and Sell NFT

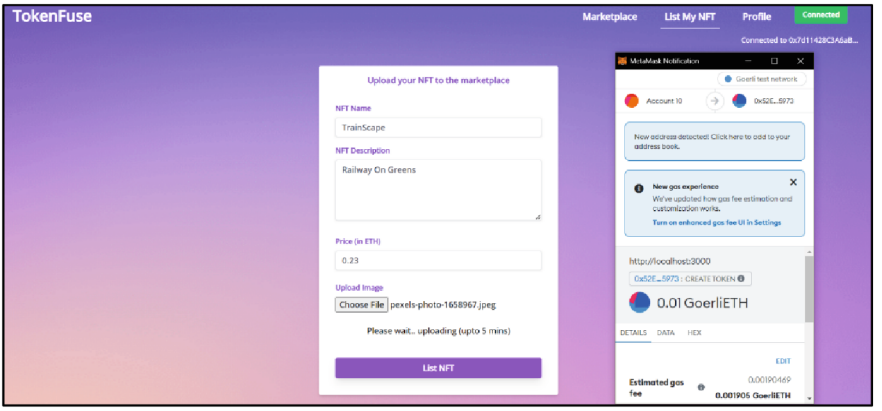

Fig. 10. Buying an NFT in TokenFuse

```
function buyNFT(uint256 tokenId) public payable {
        require(tokenIdToPrice[tokenId] > 0, "NFT is not for sale");
        require(msg.value >= tokenIdToPrice[tokenId], "Not enough ETH sent");

        address owner = ownerOf(tokenId);
        safeTransferFrom(owner, msg.sender, tokenId);

        uint256 amountToPay = tokenIdToPrice[tokenId];
        tokenIdToPrice[tokenId] = 0;
        owner.transfer(amountToPay);
    }
```

Fig. 11. Solidity Function to Buy NFT

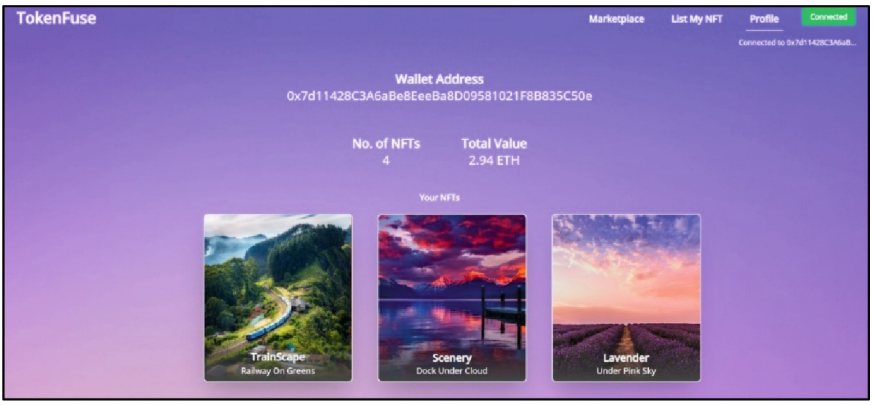

Fig. 12. Profile section of TokenFuse

```
import Navbar from "./Navbar";
import NFTTile from "./NFTTile";
import MarketplaceJSON from "../Marketplace.json";
import axios from "axios";
import { useState } from "react";

export default function Marketplace() {
const sampleData = [
    {
        "name": "NFT#1",
        "description": "Alchemy's First NFT",
        "website":"http://axieinfinity.io",
        "image":"https://gateway.pinata.cloud/ipfs/QmTsRJX7r5gyubjkdmzFrKQhHv74p5wT9LdeF
        "price":"0.03ETH",
        "currentlySelling":"True",
        "address":"0xe81Bf5A757CB4f7F82a2F23b1e59bE45c33c5b13",
    },
```

Fig. 13. Home page code snippet of TokenFuse

```
import logo from '../logo_3.png';
import fullLogo from '../full_logo.png';
import {
  BrowserRouter as Router,
  Switch,
  Route,
  Link,
  useRouteMatch,
  useParams
} from "react-router-dom";
import { useEffect, useState } from 'react';
import { useLocation } from 'react-router';

function Navbar() {

const [connected, toggleConnect] = useState(false);
const location = useLocation();
const [currAddress, updateAddress] = useState('0x');
```

Fig. 14. Navbar code snippet of TokenFuse

```
import Navbar from "./Navbar";
import axie from "../tile.jpeg";
import { useLocation, useParams } from 'react-router-dom';
import MarketplaceJSON from "../Marketplace.json";
import axios from "axios";
import { useState } from "react";

export default function NFTPage (props) {

const [data, updateData] = useState({});
const [dataFetched, updateDataFetched] = useState(false);
const [message, updateMessage] = useState("");
const [currAddress, updateCurrAddress] = useState("0x");

async function getNFTData(tokenId) {
    const ethers = require("ethers");
```

Fig. 15. Individual NFT page code snippet of TokenFuse

```
import Navbar from "./Navbar";
import { useLocation, useParams } from 'react-router-dom';
import MarketplaceJSON from "../Marketplace.json";
import axios from "axios";
import { useState } from "react";
import NFTTile from "./NFTTile";

export default function Profile () {
    const [data, updateData] = useState([]);
    const [dataFetched, updateFetched] = useState(false);
    const [address, updateAddress] = useState("0x");
    const [totalPrice, updateTotalPrice] = useState("0");

    async function getNFTData(tokenId) {
        const ethers = require("ethers");
        let sumPrice = 0;
```

Fig. 16. Profile page code snippet of TokenFuse

```
import Navbar from "./Navbar";
import { useState } from "react";
import { uploadFileToIPFS, uploadJSONToIPFS } from "../pinata";
import Marketplace from '../Marketplace.json';
import { useLocation } from "react-router";

export default function SellNFT () {
    const [formParams, updateFormParams] = useState({ name: '', description: '', price:
    const [fileURL, setFileURL] = useState(null);
    const ethers = require("ethers");
    const [message, updateMessage] = useState('');
    const location = useLocation();

    //This function uploads the NFT image to IPFS
    async function OnChangeFile(e) {
        var file = e.target.files[0];
```

Fig. 17. Sell NFT page code snippet of TokenFuse

5 Conclusion

The development and commercialization of an NFT marketplace could prove to be a viable and profitable business venture. As the use cases and interest in NFTs expand, there is a growing demand for platforms that enable the buying, selling, and trading of these digital assets. A marketplace such as TokenFuse can attract and retain customers by creating a user-friendly and secure environment, offering a diverse range of NFTs, and implementing efficient monetization mechanisms, such as charging transaction fees or taking a percentage of sales, which will ultimately result in financial success. However, it is also important to remember that it is a topic with a lot of innovation and competition,

so investing in it may be both high-risk and high-reward. Furthermore, one must always be aware of and follow rules and regulations strictly.

Future work includes ımplementing auction functionality, a facility to place bids in an existing auction, support for music, GIFs & video NFTs, implementation of transactionsusing multiple cryptocurrencies, andcustomer support.

References

1. Taş, R., Tanrıöver, Ö.Ö.: Building a decentralized application on the ethereum blockchain. In: 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). IEEE (2019)
2. Szabo, N.: Smart contracts: building blocks for digital markets. EXTROPY J. Transhum. Thought (16) **18**(2), 28 (1996)
3. Pradhan, N.R., et al.: A novel blockchain-based healthcare system design and performance benchmarking on a multi-hosted testbed. Sensors **22**(9), 3449 (2022)
4. Pradhan, N.R., Singh, A.P.: Smart contracts for automated control system in blockchain based smart cities. J. Ambient Intell. Smart Environ. **13**(3), 253–267 (2021)
5. Chohan, U.W.: Non-fungible tokens: Blockchains, scarcity, and value. Critical Blockchain Research Initiative (CBRI) Working Papers (2021)
6. Ante, L.: The non-fungible token (NFT) market and its relationship with Bitcoin and Ethereum. FinTech **1**(3), 216–224 (2022)
7. Singh, A.P., et al.: A novel patient-centric architectural framework for blockchain-enabled healthcare applications. IEEE Trans. Industr. Inf. **17**(8), 5779–5789 (2020)
8. Wang, Q., et al.: Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. arXiv preprint arXiv:2105.07447 (2021)
9. Reverse engineering open seas to build an NFT marketplace(2021) Accubits Dev Blog. https://dev.accubits.com/reverse-engineering-openseas-to-build-an-nft-market-place/
10. Lee, W.-M., Lee, W.-M.:Using the meta mask chrome extension. Beginning Ethereum Smart Contracts Programming: With Examples in Python, Solidity, and JavaScript, pp. 93–126 (2019)
11. Zheng, Q., et al.: An innovative IPFS-based storage model for blockchain. In: 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI). IEEE (2018)
12. Sarmah, M., Saxena, S., Mukherjee, S.: A decentralized crowdfunding solution on top of the ethereum blockchain. In: 2022 IEEE Silchar Subsection Conference (SILCON), Silchar, India, pp. 1–6 (2022). https://doi.org/10.1109/SILCON55242.2022.10028843
13. Chowdhury, T.R., et al.: Crypto Pay: Design of Public Blockchain Platform.AJEC (2023)
14. Pierro, G.A., Tonelli, R.: Analysis of source code duplication in ethreum smart contracts. In: 2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE (2021)
15. NonFungible: Market overview (2021). https://nonfungible.com/