A Project Report on

NFT Marketplace

Course: SIL763

Submitted by

Name	Entry Number
Arindam Sal	2024JCS2041
Ayan Shil	2024JCS2048

Project Overview:

The project implements a decentralized **NFT (Non-Fungible Token) Marketplace**, leveraging blockchain technology to facilitate secure minting, buying, and selling of digital assets. This decentralized platform enhances ownership verification and transactional trust through the use of Ethereum-based **smart contracts** and **ERC-721 compliant NFTs**.

Key Features:

1. **ERC-721 NFT Implementation:**
    - Supports minting, transferring, and burning NFTs.
    - Metadata is stored on IPFS for decentralized storage.
2. **Marketplace Functionalities:**
    - Listing and unlisting NFTs for sale.
    - Executing NFT sales with the transfer of ownership.
    - Burning NFTs by owner.
3. **Frontend Integration:**
    - React.js-based UI for user interaction.
    - Wallet connection using Metamask for seamless blockchain interaction.

Technologies Used:

- **Hardhat:** A development environment for Ethereum smart contracts, enabling deployment and testing.
- **Alchemy:** Blockchain API provider used to deploy and interact with the Ethereum network.
- **Metamask:** Browser-based wallet for interacting with blockchain applications.
- **Pinata:** For uploading and accessing NFT metadata and images on IPFS.

Acknowledgment: The frontend design and components of the NFT Marketplace were adapted from an existing GitHub source. Modifications aligned it with the project's backend and smart contract functionality.

Design Decisions

Smart Contract Structure

- NFTMarketplace Contract:
  - Extends ERC721URIStorage to implement minting and burning of NFTs.
  - Maintains a mapping (`idToListedToken`) for active NFT listings, ensuring efficient query and retrieval.
  - Implements sale execution logic, including ownership transfer and payment distribution.
- Burn Functionality:
  - Allows NFT owners to remove their assets from the marketplace permanently.

Metadata Storage

- Metadata is hosted on IPFS using Pinata, ensuring decentralized and tamper-proof data storage.
- `tokenURI` links point to metadata JSON containing NFT details such as name, description, price, and image.

Decentralized Interactions

- Metamask Integration:
  - Provides secure wallet access for users to connect and interact with the DApp.
- Payment Processing:
  - Uses payable functions to handle Ether transactions securely during minting and sales.

Frontend Design

- React.js was used to create an intuitive user interface.
- Features include:
  - Marketplace display of all NFTs.
  - Profile section to view owned NFTs and sales history.
  - NFT minting form for creating new listings.

Gas Optimization

- Smart contracts include mechanisms for efficient handling of listings and storage.
- Hardhat's optimizer was configured to reduce deployment and transaction costs.

Application Flow

Minting NFTs

- Users mint NFTs by uploading metadata to IPFS through the frontend.
- The contract:
    - Assigns a new token ID.
    - Links the `tokenURI` to the metadata.
    - Transfers ownership to the user.

Listing NFTs for Sale

- The `createToken` function lists newly minted NFTs by:
    - Transferring the NFT to the contract's custody.
    - Setting the initial sale price.
    - Updating the `idToListedToken` mapping.

Executing NFT Sales

- Buyers interact with the `executeSale` function:
    - Ether is transferred to the seller and marketplace owner (list fee).
    - Ownership is transferred to the buyer.
    - The token is removed from the active listings.

User Interface Flow

- **Home Page:** Displays all NFTs available for purchase with metadata and prices.
- **Profile Page:** Allows users to view their owned and listed NFTs.
- **Minting Page:** Provides a form to upload NFT data and create listings.

Challenges and Optimizations

- Challenge: Ensuring seamless integration between frontend and blockchain.
    - Solution: Implemented Metamask and Ether.js for efficient contract interaction.
- Challenge: Managing decentralized storage for metadata.
    - Solution: Integrated Pinata for IPFS-based data hosting.