DOI 10.34185/1562-9945-1-156-2025-12
UDC 004.415.2

I.V. Ponomarev

DEVELOPMENT OF A WEB APPLICATION FOR PROVIDING SERVICES THROUGH NFT OWNERSHIP

Abstract. A solution for creating a web application for providing subscription-based services using NFT technology is presented. The ERC721 standard for smart contracts of the Ethereum blockchain is used to create NFT. A three-tier architecture of the application is used, which includes a client interface, a server, and a blockchain layer. The features of smart contract integration are considered, which ensures uniqueness, transparency, and security of operation. The main functions include private and public token sales, subscription renewal, and user access to services.
Keywords: NFT, ERC721, token, smart contract, blockchain, Ethereum, Web3.js, OpenZeppelin, MetaMask.

Formulation of the problem. In the modern world, non-fungible tokens (NFTs) have gained popularity due to the ability to digitally prove ownership of unique objects. These tokens are used in various industries: art, games, digital collections and even to prove ownership of real-world objects. At the same time, the emergence of NFTs has opened up new opportunities for creating interactive web applications based on the use of tokens as access to services or privileges. However, the integration of NFTs into web applications requires addressing issues of security, transaction transparency, ensuring compliance with blockchain standards and effective interaction between the user and smart contracts.

Purpose of the research. It is necessary to consider the possibilities of creating a web application using NFTs to provide services in the form of a monthly subscription. The application should have the ability to initially sell tokens through public and private stages, extend the monthly subscription tied to the NFT, and gain access to services.

Such services can be: educational courses, video/audio collections, access to software, etc.

Main part. A non-fungible token is a unique digital asset that proves ownership of a specific object or content in the digital world. Unlike cryptocurrencies such as Bitcoin or Ether, which are fungible (one Bitcoin equals another Bitcoin), each NFT has unique properties that make it unique.

These tokens are created and stored on a blockchain, most often on the Ethereum platform, although other blockchains are also possible.

Each NFT has a unique identifier and metadata that defines its uniqueness. This data guarantees the authenticity and ownership history of the token. For example, when an artist

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

creates digital art and converts it into an NFT, this token confirms that a specific digital copy is original and belongs to a specific person.

Using NFTs in applications has several important advantages.

First, NFTs provide uniqueness of digital assets, which is critical for proving ownership and authenticity. This is especially true for digital works of art, collectibles, and other unique assets. Second, NFTs allow users to have full control over their digital assets, which makes it easier to transfer, sell, or exchange them without the need for intermediaries. This opens up new opportunities for monetizing digital services and content.

In addition, NFTs can be integrated with smart contracts to automate various processes, such as paying royalties to authors when their works are resold. Transparency and trust in the system are ensured by the fact that all transactions with NFTs are stored on the blockchain [1].

One of the key features of the Ethereum blockchain is the ability to create and manage both fungible and non-fungible tokens. Several standards have been developed to ensure the compatibility and interoperability of these tokens, including ERC20 and ERC721. ERC20 is the standard for creating fungible tokens, while ERC721 is the standard for creating non-fungible tokens.

The main functions that ERC721 adds to a smart contract are:

- balanceOf - returns the number of tokens owned by a given recipient;
- ownerOf - returns the owner of a given token;
- transferFrom - allows you to transfer a token from one owner to another;
- safeTransferFrom - transfers a given NFT from one account to another;
- approve - allows the token owner to grant another recipient permission to manage the token;
- getApproved - returns the address that has been granted permission to manage a given token;
- setApprovalForAll - allows or revokes permission for another recipient to manage all of the owner's tokens;
- isApprovedForAll - checks whether the recipient has been given permission to manage all of the owner's tokens [2].

The web application being created has a three-tier architecture: client layer, server layer, and blockchain layer.

At the blockchain level, the developed smart contract on the Ethereum platform uses the ERC721 standard to create an NFT with the name NewsToken and the symbol NTK. The necessary modules are imported from the OpenZeppelin library: ERC721 - the NFT standard, ERC721Burnable - provides the ability to modify the number of NFTs, and Ownable - is responsible for the ability to own and own rights to NFTs [3].

Variables are defined that are responsible for the settings and current state of the contract: the current and maximum number of NFTs, the purchase and subscription price, variables for starting the sale, blocking subscriptions, and transferring tokens, as well as hash tables with wallet addresses for private token sales whitelistedAddresses and the validity periods of the current subscription for each token.

The constructor calls the ERC721 constructor, where the token name and symbol are specified, and the smart contract owner is passed to the Ownable constructor.

The noContractExec modifier is defined to prevent massive initial sale to speculators through other smart contracts.

Taking into account the specifics of the initial sale in the form of a private and public round, the corresponding minting functions whitelistMint() and publicMint() have been developed. The public stage gives access to purchase NFT to anyone, and during the private sale, only users whose addresses are listed in the whitelistedAddresses hash table have the right to purchase. Checks are provided for the correct amount to be paid, the presence of the buyer in the private sale list, etc. The subscription renewal function renewToken(uint tokenId) accepts the NFT token identifier as a parameter. Provided that this token is owned, the subscription is extended for 30 calendar days, which generates a tokenRenew event.

```
    function renewToken(uint tokenId) public payable noContractExec
{
      require(msg.value == renewalPrice, "Incorrect amount of ether
sent.");
      require(_exists(tokenId), "Token does not exist.");
      require(renewalsEnabled, "Renewals are currently disabled");

      uint256 _currentexpiryTime = expiryTime[tokenId];

      if (block.timestamp > _currentexpiryTime) {
            expiryTime[tokenId] = block.timestamp + 30 days;
        } else {
            expiryTime[tokenId] += 30 days;
        }
      emit tokenRenew(tokenId, expiryTime[tokenId]);
    }
```

The smart contract also defines functions for solving various administrative tasks: minting and subscribing a token to another user at the administrator's expense, changing the price for the token and subscription, adding and removing an address from the list for private sale, setters of control variables and outputting the balance from the smart contract. We have functions for viewing variables (getters) and transferring tokens to another user.

The site consists of the main page and pages for authorization, NFT sale and description. Integration with the Ethereum blockchain smart contract is performed through the web3.js library. It provides developers with the ability to easily connect to the network, send transactions, interact with smart contracts and receive information from the blockchain.

The authorization page checks for the Metamask extension, prompts you to sign a request to own this wallet, and if everything is successful, the main page of the website opens.

If the user has an NFT, the main page displays the user's address, the token identifier he owns, the date and time of the expiration of the last purchased subscription. The user has the

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

opportunity to receive services in the form of information files: the subscription is checked for relevance, all available files are collected and downloaded to the user.

To renew a subscription to services, the renewSub() function is executed, which receives the renewal price and a token for a request to the smart contract, namely to the renewToken(uint tokenId) function. If the subscription is successfully renewed, 30 days are added to the expiration date. The NFT sale page displays two stages of purchase: private and public, the status of these stages (active or inactive), the user's access to the active stage, the maximum and current number of NFTs purchased. When the user chooses to purchase an NFT, the mintNFT() function is executed, which receives the purchase price, checks whether there are still free tokens for sale, rechecks the relevance of the stages, selects the active one and accordingly sends a request to the whitelistMint() or publicMint() functions of the smart contract.

The server part based on Node.js allows you to process many requests simultaneously, making the system more scalable and productive.

Conclusions. The proposed web application demonstrates the practical implementation of using NFTs to manage subscriptions to information services. Such a system provides users with significant benefits. NFTs provide ownership, and the absence of a password database eliminates the risks of hacking the server. Payment with cryptocurrency allows you to be financially independent, and the transparency of the blockchain eliminates the possibility of data forgery. The anonymity of wallets guarantees the protection of personal information, ensuring confidentiality and security.

REFERENCES

1. Imran Bashir. Mastering Blockchain: Inner workings of blockchain, from cryptography and decentralized identities, to DeFi, NFTs and Web3, Fourth Edition. – Packt Publishing, 2023. – 818 p.

2. ERC-721: Non-Fungible Token Standard. [Electronic resource]. Access mode https://eips.ethereum.org/EIPS/eip-721.

3. ERC721. [Electronic resource].
Access mode https://docs.openzeppelin.com/contracts/3.x/erc721.

Received 04.02.2025.
Accepted 06.02.2025.

Розробка веб-додатку з надання послуг через володіння NFT

Невзаємозамінні токени (NFT) набули популярності завдяки можливості цифрового підтвердження права власності на унікальні об'єкти. Поява NFT відкрила нові можливості для створення інтерактивних веб-додатків, що базуються на використанні токенів як доступу до різноманітних послуг чи привілеїв. Інтеграція NFT у веб-додатки вимагає вирішення питань безпеки, прозорості транзакцій, забезпечення відповідності стандартам блокчейна та ефективної взаємодії між користувачем і смарт-контрактами.

Представлено рішення для створення веб-додатку з використанням NFT для надання послуг в вигляді щомісячної підписки. Такими послугами можуть бути: навчальні курси, відео/аудіо збірки, доступ до програмного забезпечення тощо.

Створюваний веб-додаток має трирівневу архітектуру: клієнтський та серверний рівні та рівень блокчейну.

На рівні блокчейну розроблений смарт-контракт на платформі Ethereum використовує стандарт ERC721 для створення і управління NFT. Передбачено функції для карбування токенів, продовження підписок, управління цінами та статусом продажу.

Інтеграція сайту з смарт-контрактом виконується через бібліотеку web3.js. Вона надає розробникам можливість легко підключатися до мережі, відправляти транзакції, взаємодіяти зі смарт-контрактами та отримувати інформацію з блокчейну.

Реалізовано сторінки авторизації, продажу NFT, управління підписками та доступу до послуг. Користувачі мають можливість підписуватися на послуги, переглядати активні підписки, а також здійснювати купівлю та передачу NFT.

Серверна частина веб-додатку на базі Node.js дозволяє обробляти багато запитів одночасно, роблячи систему більш масштабованою та продуктивною.

Основні переваги системи включають унікальність цифрових активів, фінансову незалежність завдяки використанню криптовалют, а також захист персональних даних завдяки анонімності гаманців. Використання технологій блокчейну забезпечує високий рівень безпеки, виключаючи ризики підробок даних та забезпечуючи безпечний доступ до інформації.

Пономарьов Ігор Володимирович - кандидат технічних наук, доцент кафед-ри електронних обчислювальних машин факультету фізики, електроніки та комп'ютерних систем Дніпровського національного університету ім.О.Гончара.

Ponomarev Igor Volodimirovich - candidate of technical sciences, associate professor of the department of electronic computers of the faculty of physics electronics and computer systems ofthe Oles Honchar Dnipro National University.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)