**NFT Ticket Swap Contract**

**Description:**

NFT Ticket Swap Contract allows the exchange of NFTs with different properties or values. The cost difference between the swapped tokens is covered by the accompanying ether transfer.

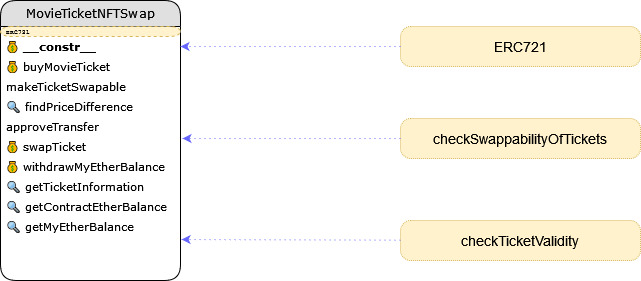
**Scenario:**

You could have a movie ticket for Star Wars, whereas I could have a movie ticket for Avengers. These tickets are not equal in properties or value, making them non-fungible and, hence, can be traded for ETH or ERC20 tokens.

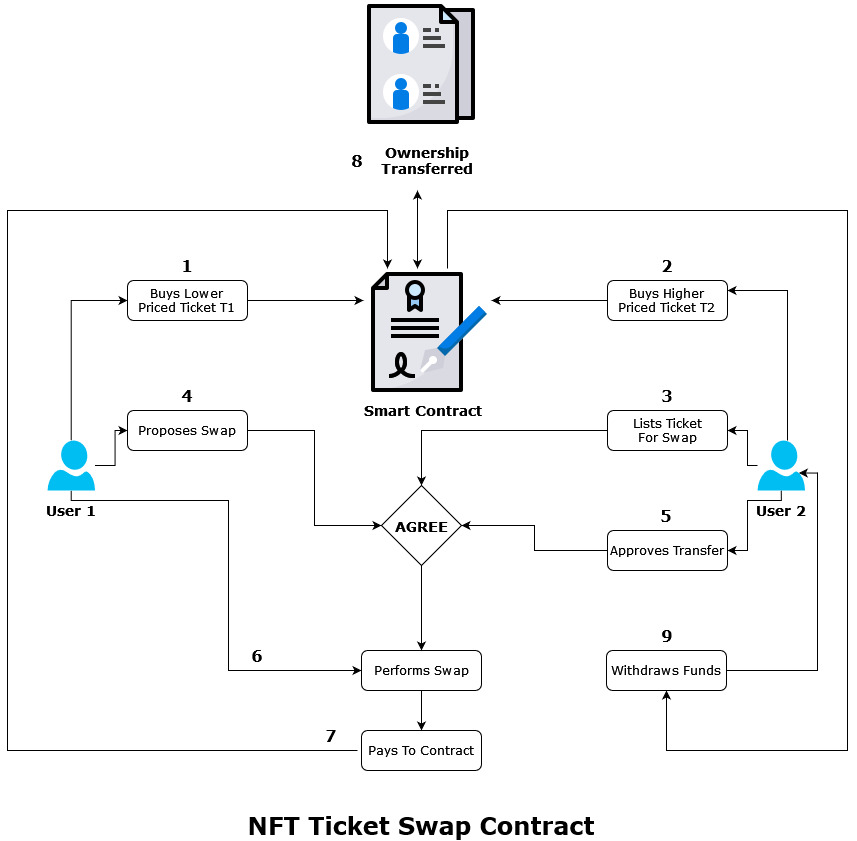
**Important Design Choice:**

An important design choice in the contract is that **only a lower-priced ticket can be exchanged for a higher-priced ticket**. Therefore, the swap transaction must be **initiated by the address that owns the lower-priced ticket**. It keeps the design simple and straightforward while avoiding complexities associated with a scenario wherein a user with the higher-priced token initiates the ticket swap. However, it has been ensured that users with the higher-priced tokens have the ability to list their tokens for exchange to give them control over their token swapping process.

**UML Diagram and Action Flow Chart**



The contract functionality is programmed in a single Contract. It does import and utilizes the ERC721 library available from Open Zepplin. In addition to the main functions, a few other additional helper functions are defined in the contract. The following diagram details the different steps required for token transfer.



**Security**

The different attacks that have been considered and prevented are as follows

* Re-entrancy attacks have been mitigated through the implementation of Checks-Effects-Interactions
* Denial of Service has been mitigated by using the pull payment method for Ether withdrawal by recipients. The swap initiator transfers Ether to contract, which is reflected in the receiver’s balance. It is not transferred immediately but only upon the receiver’s request.
* Usage of block.number instead of timestamps.
* The ERC721 OpenZepplin contract library has been used for NFT, which enhances code further
* Code has been tested for positive and negative cases to ensure that each function performs as expected when data is valid and invalid
* Usage of modifiers and state-reverting exceptions to validate conditions and handle errors is prioritized

**Efficiency**

Being an ERC721 contract that uses the ERC721 Open Zepplin library, the NFT Ticket Swap Contract avails of a few optimizations to minimize gas use.

* Optimizer has been enabled in truffle-config.js file

optimizer: {

    enabled: true,

    runs: 1

},

* A minimal number of state variables are used without impacting the functionality and security of the contract
* The pull method is used for minting tokens and withdrawing funds
* Avoiding the use of infinite loops (for, while, do-while)
* Ensuring proper visibility for functions
* Hardcoded as many variables as possible to minimize state change without compromising functionality
* Error codes replace error messages

|  |  |
| --- | --- |
| Error Code | Error Message |
| 1 | Expired or about to expire tickets can not be swapped |
| 2 | Can only swap with a higher or equal priced ticket! |
| 3 | Movie id must be valid |
| 4 | Please pay the correct amount for the ticket |
| 5 | Only ticket owner can list token for swapping! |
| 6 | The ticket should be listed for swapping |
| 7 | Please provide right payment for transfer |
| 8 | Nothing to withdraw |
| 9 | Failed to send Ether |

**Execution Instructions**

For execution instructions, please refer to the Readme.md file in the module 8 assignment folder