Hedgey NonTransferrable NFTs

Project Overview:
The NontransferrableNFTs contract purpose is to provide an easy, transparent, and trackable mechanism for people to lock tokens inside an ERC721 wrapper that has a set unlock date. This contract can be used for DAOs and web3 token teams that are selling or gifting tokens to certain individuals or companies (wallet addresses) that will own the NFTs until their unlock date has arrived. Once the current time (represented by block time) has surpassed the unlock date, the owner of the NFT would unlock the tokens represented by the NFT, and receive them to their wallet. The NFT in effect acts as a key to unlock a vault of ERC20 tokens, however this contract in particular is set such that it cannot be transferred to any other party. The purpose of avoiding the transferability is so that the locked tokens cannot be sold to another party, purposefully prohibiting secondary sales without the consent of the party who initially locked and minted the NFTs to those recipients in the first place.

Functional Requirements:
1.1 Roles:
There are three primary roles involved in the smart contract, the admin, the creators, and the owners.

a. **The Admin:** The admin's sole purpose and responsibility is to deploy the smart contract, set the new baseURI to https://nft/hedgey.finance/{network}/{contract_address}/ and then delete themselves from the contract and storage so that it is immutable
b. **The Creators:** The creators mint new NFTs, minting tokens that are locked inside NFTs, which can either be to themselves, or to other parties who now own the NFTs. Typically we would expect the creators to be token DAOs who are locking tokens and delivering these token vaults to specific individuals or organizations, whether for investment purposes or specific contribution purposes. However, the DAOs that utilize this contract do not want these locked token vaults to be transferred to anyone else, but want to keep them in the hands of these specific wallet owners.
c. **The Recipients:** The recipients of the NFTs have but one choice, they can redeem the NFTs when the unlock date has been reached. They are the organizations or individuals that have been awarded locked tokens in a vault with a specific unlock date. These recipients are expected to Only unlock, redeem and burn their NFT in exchange for their locked tokens, they cannot do any other activities.

1.2 Features:

a. The features include the ability for an organization or individual to lock tokens inside of an ERC721 NFT and mint that NFT to themselves or any other wallet address they so choose.
b. That minted NFT cannot be transferred, must be held by the recipient until they redeem the NFT.
c. That NFT can be redeemed for a set amount of ERC20 tokens that unlock after a specific time.

1.3 Tokenomics & Fees:

a. There are no fees and no tokenomics associated with the contract.

Technical Requirements

2.0 Architecture Details

a. The contract inherits the ERC721 Enumerable standards from OpenZeppelin, along with ReentrancyGuard and a library to assist with ERC20 transfers (TransferHelper)
b. The contract uses the Counters library to increment NFTIDs one at a time as they are minted
c. The contract has a baseURI which is what each token points to for its metadata. The metadata is simply a fun representation of an NFT visually, with its image and underlying details of the tokens locked by the NFT
d. The tokens actually locked by the NFT are stored in storage in a struct called Futures, which stores the locked token vault details on-chain so that there is no intervention with the metadata (aside from purely being visual). These futures structs are stored at the same ID that is associated with each individual NFT - they are mapped by the same uint256 that records the NFT token ID
e. When a user mints an NFT, it is simultaneously required to lock standard ERC20 tokens with a specific unlock date, set in UTC time, tested against block timestamp.
f. A minter can mint the NFT and lock tokens to themselves, or to any other wallet they wish, which assists with distributing locked tokens wrapped in an NFT to other wallets.
g. The owner of an NFT can only redeem their NFT, which burns the NFT, deletes the Futures struct and then delivers the ERC20 tokens that were held by the smart contract to the owner of the NFT.

2.1 Events:

Events are recorded when a new NFT is created, when an NFT is redeemed and burned, and when the baseURI is set for the first and only time. The events are used to assist with databases storing and keeping records of transactions for easy display on front end web applications.