Niftify Smart Contracts Final Audit Report

Project Synopsis

Project Name	**Niftify**
Platform	Ethereum, Solidity
Github Repo	https://github.com/Niftify-io/token-smart-contracts
Deployed Contract	Not Deployed
Total Duration	6 Days
Timeline of Audit	**25th August 2021 to 2nd September 2021**

Contract Details

Total Contract(s)	4
Name of Contract(s)	*NiftifyERC1155.sol, NiftifyERC721.sol, TokenStorage.sol, NiftifyNFTVoucher.sol*
Language	Solidity
Commit Hash	`19d63f7369e3b3c61a860ea147ac7bfb8c689ad7`

Contract Vulnerabilities Synopsis

Issues	Open Issues	Closed Issues
Critical Severity	0	0
Medium Severity	0	1
Low Severity	0	2
Information	3	3
Total Found	3	5

Detailed Results

The contract has gone through several stages of the audit procedure that includes structural analysis, automated testing, manual code review, etc.

All the issues have been explained and discussed in detail below. Along with the explanation of the issue found during the audit, the recommended way to overcome the issue or improve the code quality has also been mentioned.

A. Contract Name: TokenStorage

High Severity Issues
None Found

Medium Severity Issues
A.1 Operator Role has not been set up in the contract
Line no - 8
Status: CLOSED
Explanation:
The **OPERATOR_ROLE** in the TokenStorage contract has not been assigned to any particular address within the contract.
Is this Intended?
While the **Minter** and **Redeemer** role can be handled later via admin, the operator role might need an initial setup as functions like **_setBlocked_** uses Operator_Roles as a requirement, in the contract.

```
66
67     function setBlocked(uint256 _tokenId, bool _value) external onlyOperator {
68       _blocked[_tokenId] = _value;
69     }
70
```

Recommendation:
If the above-mentioned scenario is not intended, then the Operator Role should be assigned to a particular address while deployment.

Low Severity Issues

A.2 Absence of Error messages in Require Statements
Line no - 47
Status: CLOSED
Description:
The **TokenStorage** contract includes a **require statement** in the **_setRoyalty** function(*at the above-mentioned line*) that doesn't contain any error message within itself.
While this makes it troublesome to detect the reason behind a particular function revert, it also reduces the readability of the code.

Recommendation:

Error Messages must be included in every require statement in the contract.

A.3 External Visibility should be preferred
Explanation:

Those functions that are never called throughout the contract should be marked as **external** instead of **public** visibility.
This will effectively result in Gas Optimization as well.

The following function in the TokenStorage contract has been assigned **public visibility** but never called form within the contract:
- **metadataHash() #Line 22**

Recommendation:
If the **PUBLIC** visibility of the above-mentioned functions is not intended, then the **EXTERNAL** visibility keyword should be preferred.

Informational

A.4 Coding Style Issues in the Contract
Status: Not Considered

Explanation:
Code readability of a Smart Contract is largely influenced by the Coding Style issues and in some specific scenarios may lead to bugs in the future.

During the automated testing, it was found that the **TokenStorage** contract had quite a few code-style issues.

```
INFO:Detectors:
Parameter TokenStorage.metadataHash(uint256)._tokenId (myFlats/flatTokenStorage.sol#471) is not in mixedCase
Parameter TokenStorage.creator(uint256)._tokenId (myFlats/flatTokenStorage.sol#483) is not in mixedCase
Parameter TokenStorage.royalty(uint256)._tokenId (myFlats/flatTokenStorage.sol#491) is not in mixedCase
Parameter TokenStorage.royaltyInfo(uint256,uint256)._tokenId (myFlats/flatTokenStorage.sol#500) is not in mixedCase
Parameter TokenStorage.royaltyInfo(uint256,uint256)._salePrice (myFlats/flatTokenStorage.sol#500) is not in mixedCase
Parameter TokenStorage.blocked(uint256)._tokenId (myFlats/flatTokenStorage.sol#512) is not in mixedCase
Parameter TokenStorage.setBlocked(uint256,bool)._tokenId (myFlats/flatTokenStorage.sol#516) is not in mixedCase
Parameter TokenStorage.setBlocked(uint256,bool)._value (myFlats/flatTokenStorage.sol#516) is not in mixedCase
Parameter TokenStorage.supportsInterface(bytes4)._interfaceId (myFlats/flatTokenStorage.sol#520) is not in mixedCase
```

Recommendation:
Therefore, it is recommended to fix the issues like naming convention, indentation, and code layout issues in a smart contract.

A.5 Multiplication is being performed on the result of Division
Line no - 83-94

Status: CLOSED

Explanation:

During the automated testing of the **TokenStorage** contract, it was found that the contract includes a function that performs multiplication on the result of a Division.

Integer Divisions in Solidity might truncate. Moreover, this performing division before multiplication might lead to a loss of precision.

The following function involves division before multiplication in the mentioned lines:
- **mulScale** at 83-94

    Automated Tets result has been attached below:

```
TokenStorage.mulScale(uint256,uint256,uint32) (myFlats/flatTokenStorage.sol#532-543) performs a multiplication on the result of a division:
        -c = y / scale (myFlats/flatTokenStorage.sol#539)
        -a * c * scale + a * d + b * c + (b * d) / scale (myFlats/flatTokenStorage.sol#542)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
```

Recommendation:

Solidity doesn't encourage arithmetic operations that involve division before multiplication.
Therefore it is highly recommended to write extensive test cases to ensure the function behavior is as per the expectation. The above-mentioned function should be redesigned only if they do not lead to expected results.

A.6 NatSpec Annotations must be included
Status: CLOSED

Description:
The smart contracts do not include the NatSpec annotations adequately.

Recommendation:
Cover by NatSpec all Contract methods.

B. Contract Name: NiftifyERC721

Informational

B.1 Coding Style Issues in the Contract

Status: Not Considered

Explanation:
Code readability of a Smart Contract is largely influenced by the Coding Style.

During the automated testing, it was found that the **NiftifyERC721** contract had quite a few code-style issues.

```
Parameter NiftifyERC721.redeem(NiftifyNFTVouchers.ERC721Voucher)._voucher (myFlats/flaterc721.sol#2027) is not in mixedCase
Parameter NiftifyERC721.mint(NiftifyNFTVouchers.ERC721Voucher)._voucher (myFlats/flaterc721.sol#2044) is not in mixedCase
Parameter NiftifyERC721.mintAndTransfer(NiftifyNFTVouchers.ERC721Voucher,address)._voucher (myFlats/flaterc721.sol#2058) is not in mixedCase
Parameter NiftifyERC721.mintAndTransfer(NiftifyNFTVouchers.ERC721Voucher,address)._receiver (myFlats/flaterc721.sol#2058) is not in mixedCase
Parameter NiftifyERC721.updateMetadataHash(NiftifyNFTVouchers.UpdateMetadataHashVoucher)._voucher (myFlats/flaterc721.sol#2066) is not in mixedCase
Parameter NiftifyERC721.supportsInterface(bytes4)._interfaceId (myFlats/flaterc721.sol#2098) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Recommendation:
Therefore, it is quite effective to fix the issues like naming convention, indentation, and code layout issues in a smart contract.

B.2 NatSpec Annotations must be included

Status: CLOSED

Description:
The smart contracts do not include the NatSpec annotations adequately.

Recommendation:
Cover by NatSpec all Contract methods.

C. Contract Name: NiftifyERC1155

Informational

C.1 Coding Style Issues in the Contract

Status: Not Considered

Explanation:
Code readability of a Smart Contract is largely influenced by the Coding Style.

During the automated testing, it was found that the **NiftifyERC1155** contract had quite a few code-style issues.

```
Parameter NiftifyERC1155.redeem(NiftifyNFTVouchers.ERC1155Voucher)._voucher (myFlats/flaterc1155.sol#2131) is not in mixedCase
Parameter NiftifyERC1155.mint(NiftifyNFTVouchers.ERC1155Voucher)._voucher (myFlats/flaterc1155.sol#2157) is not in mixedCase
Parameter NiftifyERC1155.mintAndTransfer(NiftifyNFTVouchers.ERC1155Voucher,address)._voucher (myFlats/flaterc1155.sol#2174) is not in mixedCase
Parameter NiftifyERC1155.mintAndTransfer(NiftifyNFTVouchers.ERC1155Voucher,address)._receiver (myFlats/flaterc1155.sol#2174) is not in mixedCase
Parameter NiftifyERC1155.updateMetadataHash(NiftifyNFTVouchers.UpdateMetadataHashVoucher)._voucher (myFlats/flaterc1155.sol#2188) is not in mixedCase
Parameter NiftifyERC1155.supportsInterface(bytes4)._interfaceId (myFlats/flaterc1155.sol#2225) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Recommendation:

Therefore, it is recommended to fix the issues like naming convention, indentation, and code layout issues in a smart contract.

C.2 NatSpec Annotations must be included
Status: CLOSED

Description:
The smart contracts do not include the NatSpec annotations adequately.

Recommendation:
Cover by NatSpec all Contract methods.

Contract Name: NiftifyNFTVouchers
No issues Found