



**Tailor NFT**  
**NFT Collectible**  
**SMART CONTRACT AUDIT**  
**21.05.2022**

**Made in Germany by Chainsulting.de**



## Table of contents

|   |    |
|---|----|
| 1. Disclaimer.....  | 4  |
| 2. About the Project and Company .....                    | 5  |
| 2.1 Project Overview.....                                 | 6  |
| 3. Vulnerability & Risk Level .....                       | 7  |
| 4. Auditing Strategy and Techniques Applied.....          | 8  |
| 4.1 Methodology .....                                     | 8  |
| 5. Metrics .....  | 9  |
| 5.1 Tested Contract Files .....                           | 9  |
| 5.2 Used Code from other Frameworks/Smart Contracts ..... | 10 |
| 5.3 CallGrap .....  | 12 |
| 5.4 Inheritance Graph .....                               | 13 |
| 5.5 Source Lines & Risk.....                              | 14 |
| 5.6 Capabilities .....                                    | 15 |
| 5.7 Source Unites in Scope .....                          | 16 |
| 6. Scope of Work.....                                     | 18 |
| 6.1 Findings Overview .....                               | 19 |
| 6.2 Manual and Automated Vulnerability Test.....          | 20 |
| 6.2.1 Shadowed State Variable .....                       | 20 |
| 6.2.2 Floating Pragma Version Identified .....            | 22 |
| 6.2.3 Dead Code .....                                     | 22 |
| 6.2.4 Failing Unit Tests.....                             | 23 |
| 6.2.5 Storing Data Via baseURI.....                       | 26 |

|                                  |    |
|----------------------------------|----|
| 6.3 SWC Attacks .....            | 27 |
| 6.4. Verify Claims .....         | 31 |
| 7. Executive Summary.....        | 32 |
| 8. Deployed Smart Contract ..... | 32 |
| 9. About the Auditor .....       | 33 |



## 1. Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of TAILOR SAS. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

| Major Versions / Date | Description   |
|-----------------------|---|
| 0.1 (03.05.2022)      | Layout  |
| 0.4 (04.05.2022)      | Automated Security Testing<br>Manual Security Testing |
| 0.5 (05.05.2022)      | Verify Claims and Test Deployment                     |
| 0.6 (07.05.2022)      | Testing SWC Checks                                    |
| 0.9 (08.05.2022)      | Summary and Recommendation                            |
| 1.0 (09.05.2022)      | Final document  |
| 1.1 (15.05.2022)      | Re-Check  |
| 1.1 (21.05.2022)      | Added deployed contract                               |

## 2. About the Project and Company

### Company address:

TAILOR SAS  
SIREN 900570144  
14 Rue Broca  
75005 Paris, France



**Website:** <https://www.tailor-nft.com>

**Twitter:** [https://twitter.com/tailor\\_nft](https://twitter.com/tailor_nft)

**Discord:** <https://discord.gg/3gBzEuhHZZ>

**LinkedIn:** <https://www.linkedin.com/company/tailor-nft>

**Instagram:** <https://www.instagram.com/tailor.nft>

**Medium:** <https://medium.com/@tailor-nft>

## 2.1 Project Overview

Tailor builds on NFT tickets to create new experiences around live events. Your tickets will not only allow you to access to an event, it will give you access to a community, allowing early bird tickets for future events, exclusive chats and interview with the artists, exclusive tradable collectibles, immersive online experiences.

Possibilities are endless, thanks to blockchain tech that we make accessible to everyone in a simple and transparent way.

### 3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

| Level         | Value   | Vulnerability   | Risk (Required Action)  |
|---------------|---------|---|---|
| Critical      | 9 – 10  | A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.      | Immediate action to reduce risk level.                              |
| High          | 7 – 8.9 | A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way. | Implementation of corrective actions as soon as possible.           |
| Medium        | 4 – 6.9 | A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.                                     | Implementation of corrective actions in a certain period.           |
| Low           | 2 – 3.9 | A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.       | Implementation of certain corrective actions or accepting the risk. |
| Informational | 0 – 1.9 | A vulnerability that have informational character but is not effecting any of the code.   | An observation that does not determine a level of risk              |

## 4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

### 4.1 Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
  - i. Review of the specifications, sources, and instructions provided to Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
  - ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
  - iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Chainsulting describe.
2. Testing and automated analysis that includes the following:
  - i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
  - ii. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.



## 5. Metrics

The metrics section should give the reader an overview on the size, quality, flows and capabilities of the codebase, without the knowledge to understand the actual code.

### 5.1 Tested Contract Files

The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

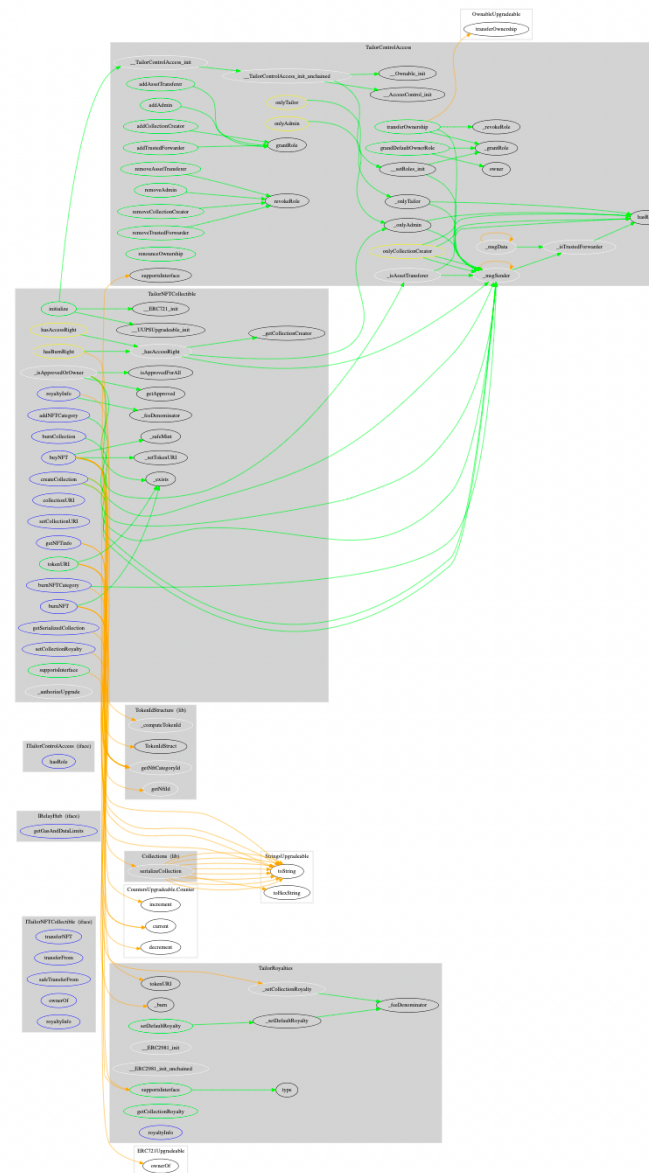
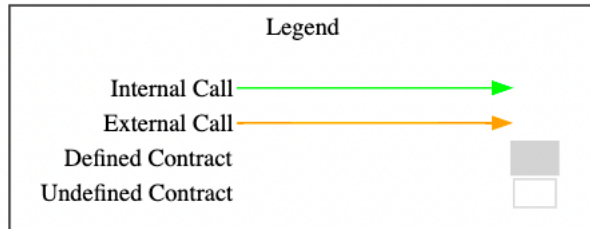
| File  | Fingerprint (MD5)                |
|---|----------------------------------|
| ./interfaces/ITailorNFTCollectible.sol            | a0c43f7913dca9e6086dd2a4018aaea5 |
| ./interfaces/ITailorControlAccess.sol             | c2e9bbfafe1c888dec34966003b2ad37 |
| ./interfaces/IRelayHub.sol                        | 7b6cda20b0f8bd5deec6b29818c12888 |
| ./contracts/access/TailorControlAccess.sol        | dbef15d7ddb908f9099c955bc430579e |
| ./contracts/token/ERC721/TailorNFTCollectible.sol | 132e2057c823bc8dac603e858f4dd47f |
| ./contracts/token/ERC2981/TailorRoyalties.sol     | 1e669c0c3138bd2d9063f506a9b06557 |
| ./contracts/token/libraries/Collections.sol       | 7c5525527fa166ec7be915f53cfcacf5 |
| ./contracts/token/libraries/TokenIdStructure.sol  | fcd95ed001f2ab07c8d1046074876e82 |

## 5.2 Used Code from other Frameworks/Smart Contracts (direct imports)

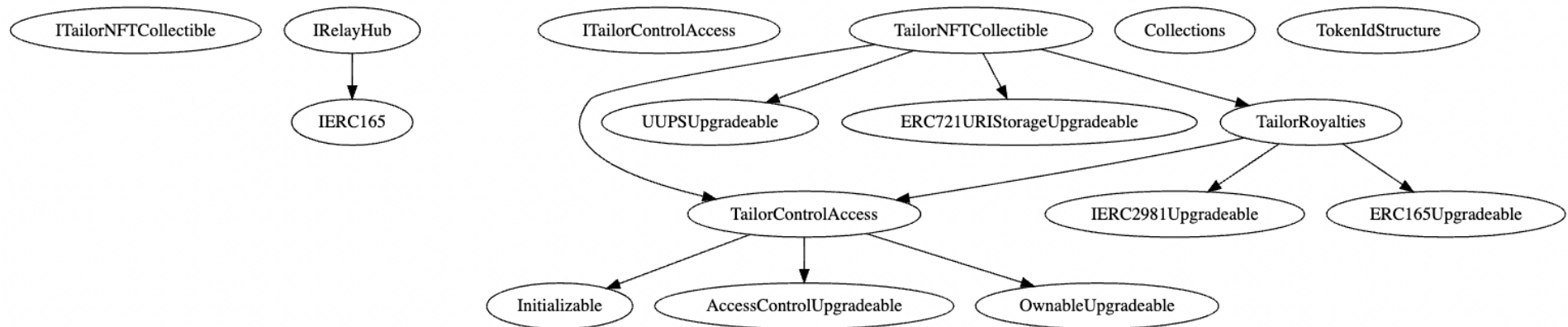
| Dependency / Import Path  | Source  |
|---|---|
| @openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol                     | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/access/AccessControlUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/access/AccessControlUpgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol                           | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/access/OwnableUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/access/OwnableUpgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/interfaces/IERC165Upgradeable.sol                       | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/interfaces/IERC165Upgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/interfaces/IERC165Upgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/interfaces/IERC2981Upgradeable.sol                      | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/interfaces/IERC2981Upgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/interfaces/IERC2981Upgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol                           | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/proxy/utils/Initializable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/proxy/utils/Initializable.sol</a>   |
| @openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol                         | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/proxy/utils/UUPSUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/proxy/utils/UUPSUpgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/token/ERC721/extensions/ERC721URIStorageUpgradeable.sol | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/token/ERC721/extensions/ERC721URIStorageUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/token/ERC721/extensions/ERC721URIStorageUpgradeable.sol</a> |
| @openzeppelin/contracts-upgradeable/utils/CountersUpgradeable.sol                           | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/CountersUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/CountersUpgradeable.sol</a>   |
| @openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol                            | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/StringsUpgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/StringsUpgradeable.sol</a>   |

| Dependency / Import Path  | Source  |
|---|---|
| @openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/introspection/ERC165Upgradeable.sol">https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/tree/v4.5.2/utils/introspection/ERC165Upgradeable.sol</a> |
| @openzeppelin/contracts/interfaces/IERC165.sol                                | <a href="https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.5.0/contracts/interfaces/IERC165.sol">https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v4.5.0/contracts/interfaces/IERC165.sol</a>   |

## 5.3 CallGrap

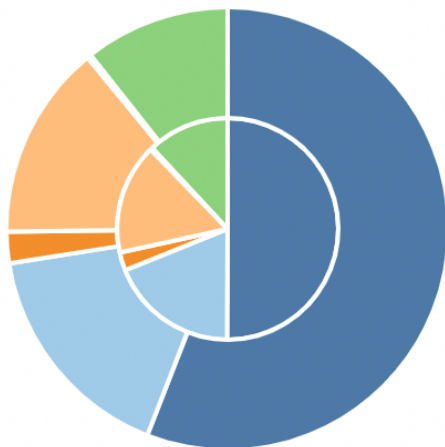


## 5.4 Inheritance Graph

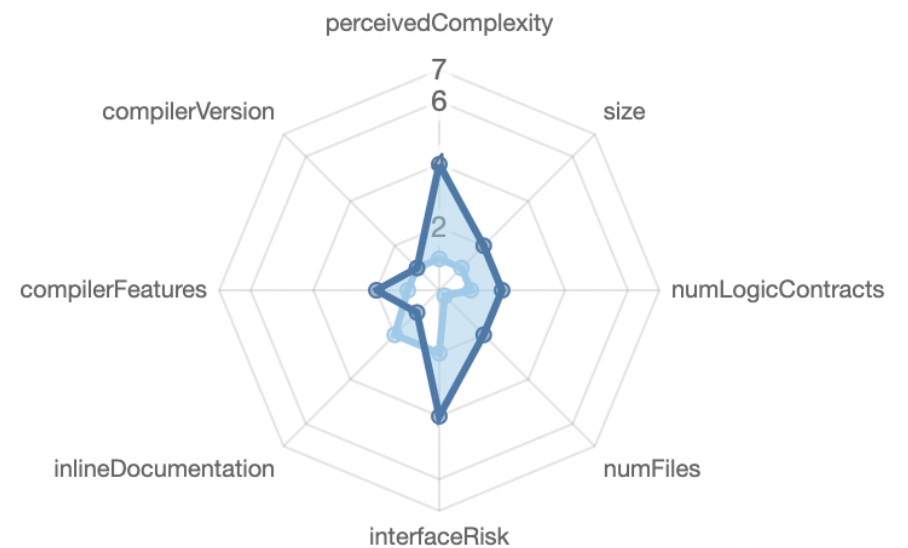


## 5.5 Source Lines & Risk

source comment single block mixed  
empty todo blockEmpty



overall average





## 5.6 Capabilities


| Solidity Versions observed  |   |  Experimental Features |  Can Receive Funds    |  Uses Assembly |  Has Destroyable Contracts |
|---|---|---|---|---|---|
| <div><div>^0.8.4</div><div>^0.8.0</div><div>^0.8.9</div></div>                                  |   |   | <div></div>   | <div>yes</div> <div>(1 asm blocks)</div>  | <div></div>   |
|  Transfers ETH |  Low-Level Calls |  DelegateCall          |  Uses Hash Functions |  ECTRecover    |  New/Create/Create2        |
| <div></div>   | <div></div>   | <div></div>   | <div>yes</div>  | <div></div>   |   |

### Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

|  |   |         |      |      |
|--|---|---------|------|------|
|  Public |  Payable |         |      |      |
| 37   | 0   |         |      |      |
| External   | Internal  | Private | Pure | View |
| 20   | 57  | 0       | 4    | 24   |










### StateVariables

| Total |  Public |
|-------|--|
| 15    | 0  |



## 5.7 Source Unites in Scope

Source: <https://github.com/Tailor-NFT/smart-contracts>

Last commit: 79d75365ebe561dc3dd1c9902b54619e483280d7

| Type  | File  | Logic Contracts | Interfaces | Lines | nLines | nSLOC | Comment Lines | Complex. Score | Capabilities  |
|---|---|-----------------|------------|-------|--------|-------|---------------|----------------|---|
|    | interfaces/ITailorNFTCollectible.sol            | _____           | 1          | 30    | 5      | 3     | 1             | 11             | _____   |
|    | interfaces/IRelayHub.sol                        | _____           | 1          | 28    | 24     | 15    | 5             | 5              | _____   |
|    | interfaces/ITailorControlAccess.sol             | _____           | 1          | 6     | 5      | 3     | 1             | 3              | _____   |
|    | contracts/access/TailorControlAccess.sol        | 1               | _____      | 194   | 174    | 107   | 46            | 99             |  |
|    | contracts/token/ERC721/TailorNFTCollectible.sol | 1               | _____      | 568   | 477    | 298   | 105           | 165            | _____   |
|  | contracts/token/ERC2981/TailorRoyalties.sol     | 1               | _____      | 152   | 122    | 70    | 31            | 38             | _____   |
|  | contracts/token/libraries/Collections.sol       | 1               | _____      | 111   | 107    | 92    | 4             | 33             | _____   |
|  | contracts/token/libraries/TokenIdStructure.sol  | 1               | _____      | 65    | 57     | 16    | 38            | 5              | _____   |



| Type  | File   | Logic Contracts | Interfaces | Lines | nLines | nSLOC | Comment Lines | Complex. Score | Capabilities  |
|---|--------|-----------------|------------|-------|--------|-------|---------------|----------------|---|
|  | Totals | 5               | 3          | 1154  | 971    | 604   | 231           | 359            |  |

Legend: [ ]

- **Lines:** total lines of the source unit
- **nLines:** normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC:** normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines:** lines containing single or block comments
- **Complexity Score:** a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

## 6. Scope of Work

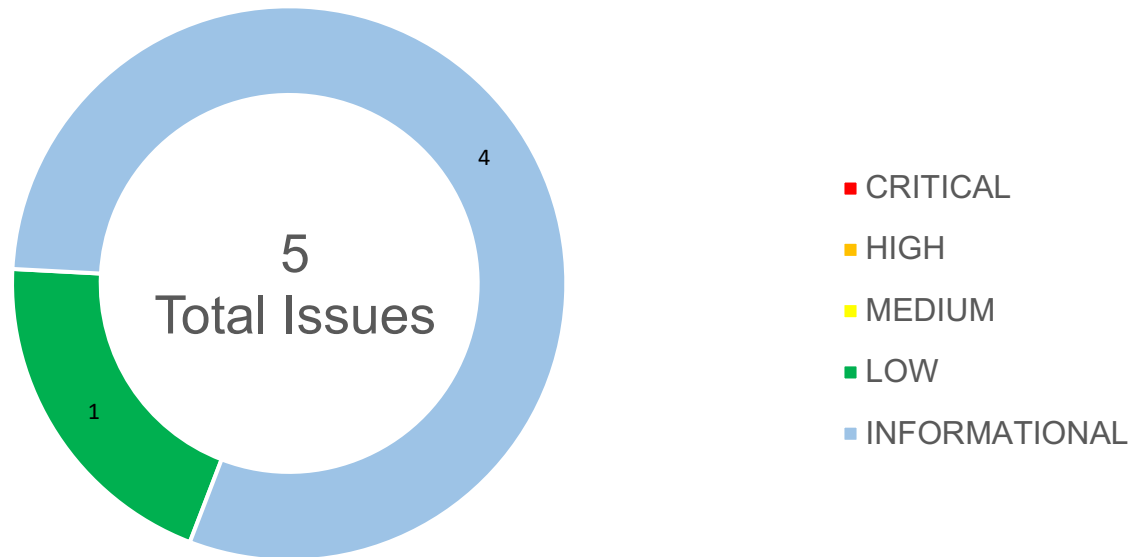
The Tailor NFT Team provided us with the files that needs to be tested. The scope of the audit is the Tailor NFT Collectible contract.

The team put forward the following assumptions regarding the security, usage of the contracts:

- The contract is using and ERC standard for NFTs
- Royalties are correctly implemented and calculated
- The smart contract is coded according to the newest standards and in a secure way.

The main goal of this audit was to verify these claims. The auditors can provide additional feedback on the code upon the client's request.

## 6.1 Findings Overview



| No    | Title                              | Severity      | Status       |
|-------|------------------------------------|---------------|--------------|
| 6.2.1 | Shadowed State Variable            | LOW           | FIXED        |
| 6.2.2 | Floating Pragma Version Identified | INFORMATIONAL | FIXED        |
| 6.2.3 | Dead Code                          | INFORMATIONAL | FIXED        |
| 6.2.4 | Failing Unit Tests                 | INFORMATIONAL | ACKNOWLEDGED |
| 6.2.5 | Storing Data Via baseURI           | INFORMATIONAL | ACKNOWLEDGED |

## 6.2 Manual and Automated Vulnerability Test

### CRITICAL ISSUES

During the audit, Chainsulting's experts found **0 Critical issues** in the code of the smart contract.

### HIGH ISSUES

During the audit, Chainsulting's experts found **0 High issues** in the code of the smart contract.

### MEDIUM ISSUES

During the audit, Chainsulting's experts found **0 Medium issues** in the code of the smart contract.

### LOW ISSUES

During the audit, Chainsulting's experts found **1 Low issue** in the code of the smart contract.

#### 6.2.1 Shadowed State Variable

Severity: LOW

Status: **FIXED**

Code: SWC-119

File(s) affected: TailorNFTCollectible.sol

Update: 31f523f67d45063cc0d65d872b753adacf2cf153

|                             |  |
|-----------------------------|--|
| <b>Attack / Description</b> | In the current implementation, state variables in the deriving contract are shadowed. Unlike functions, state variables cannot be overridden by re-declaring it in the child contract. |
| <b>Code</b>                 | Line 76 (TailorNFTCollectible.sol)   |

|                              |   |
|------------------------------|---|
|                              | <pre>__ERC721_init(name, symbol);</pre> <p>name and symbol are shadowing ERC721Upgradeable state variables.</p> <p>Line 372, 378, 382, 385 (TailorNFTCollectible.sol)</p> <pre>string calldata _baseURI,</pre> <p>baseURI shadows ERC721Upgradeable state variable.</p> <p>Line 512 (TailorNFTCollectible.sol)</p> <pre>address owner = ERC721Upgradeable.ownerOf(tokenId);</pre> <p>owner shadows OwnableUpgradeable state variable.</p> |
| <b>Result/Recommendation</b> | We highly recommend removing the shadowed state variables to avoid unintended behaviour.  |

## INFORMATIONAL ISSUES

During the audit, Chainsulting's experts found **4 Informational issues** in the code of the smart contract.

### 6.2.2 Floating Pragma Version Identified

Severity: INFORMATIONAL

Status: **FIXED**

Code: SWC-103

File(s) affected: ALL

Update: 955dbd1a60c2640898b4fe64a34d7f03a5cc2dbe

|                              |   |
|------------------------------|---|
| <b>Attack / Description</b>  | It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.   |
| <b>Code</b>                  | Line 1<br><code>^0.8.0</code>   |
| <b>Result/Recommendation</b> | It is recommended to follow the latter example, as future compiler versions may handle certain language constructions in a way the developer did not foresee. It is advised that floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version.<br><br>i.e. Pragma solidity 0.8.0 |

### 6.2.3 Dead Code

Severity: INFORMATIONAL

Status: **FIXED**

Code: CWE-561

File(s) affected: TailorNFTCollectible.sol, TailorAccessControl.sol

Update: The code is actually used as it is an override of one of the inherited contracts.

|                              |   |
|------------------------------|---|
| <b>Attack / Description</b>  | In the current implementation is some code which is never used. It extends the contract size without any benefits. Unused code is the <code>_isApprovedOrOwner</code> function in <code>TailorNFTCollectible</code> and the <code>ASSET_TRANSFERER</code> role in <code>TailorAccessControl</code> .  |
| <b>Code</b>                  | <p>Line 502 - 518 (<code>TailorNFTCollectible.sol</code>)</p> <pre>function _isApprovedOrOwner(address spender, uint256 tokenId)</pre> <p>subsequently role <code>ASSET_TRANSFERER</code> is not used</p> <p>Line 20, 77- 79, 96 – 98, 115 – 117 (<code>TailorAccessControl.sol</code>)<br/>all functions regarding <code>ASSET_TRANSFERER</code> because it is never used.</p> |
| <b>Result/Recommendation</b> | We highly recommend removing dead code to reduce contract size and thus gas costs on deployment.  |

#### 6.2.4 Failing Unit Tests

Severity: INFORMATIONAL

Status: ACKNOWLEDGED

Code: NA

File(s) affected: NA

Update: We are aware, we tested a feature which is not currently supported by the deployment lib we are using.

|                             |  |
|-----------------------------|--|
| <b>Attack / Description</b> | 4 Unit Tests are failing   |
| <b>Code</b>                 | <p><u>Log:</u></p> <p>151 passing (2m)<br/>4 failing</p> <p>1) Upgradeability<br/>#upgradeProxy - storage collisions</p> |

add state variables to base contract (tailorRoyalties):  
Error: New storage layout is incompatible

contracts/mocks/TestTailorRoyaltiesV2.sol:33: Inserted `royaltytest`  
> New variables should be placed after all existing inherited variables

contracts/mocks/TestTailorRoyaltiesV2.sol:165: Upgraded `\_\_gap` to an incompatible type  
- Bad array resize from 47 to 46  
Size cannot decrease  
at assertStorageUpgradeSafe (node\_modules/@openzeppelin/upgrades-core/src/storage/index.ts:35:11)  
at deployImpl (node\_modules/@openzeppelin/hardhat-upgrades/src/utils/deploy-impl.ts:108:29)  
at async Proxy.upgradeProxy (node\_modules/@openzeppelin/hardhat-upgrades/src/upgrade-proxy.ts:25:32)  
at async upgradeProxy (test/accessControl/Upgradability.test.ts:42:12)  
at async Context.<anonymous> (test/accessControl/Upgradability.test.ts:207:36)

2) Upgradeability  
#upgradeProxy - storage collisions  
add heritage contract with state variables to NFTCollectible:  
Error: New storage layout is incompatible

contracts/mocks/TestAppendHeritageContract.sol:21: Inserted `testHeritage`  
> New variables should be placed after all existing inherited variables

contracts/mocks/TestAppendHeritageContract.sol:22: Inserted `testMapping`  
> New variables should be placed after all existing inherited variables

contracts/mocks/TestAppendHeritageContract.sol:29: Replaced `\_\_placeholder` with `\_structHeritage` of incompatible type  
at assertStorageUpgradeSafe (node\_modules/@openzeppelin/upgrades-core/src/storage/index.ts:35:11)  
at deployImpl (node\_modules/@openzeppelin/hardhat-upgrades/src/utils/deploy-impl.ts:108:29)  
at async Proxy.upgradeProxy (node\_modules/@openzeppelin/hardhat-upgrades/src/upgrade-proxy.ts:25:32)  
at async upgradeProxy (test/accessControl/Upgradability.test.ts:42:12)  
at async Context.<anonymous> (test/accessControl/Upgradability.test.ts:250:36)

3) Upgradeability  
verify state of proxy after upgrade  
should upgrade (appendToBase) and still have the state in the proxy :  
Error: New storage layout is incompatible



|                              |   |
|------------------------------|---|
|                              | <p>contracts/mocks/TestTailorRoyaltiesV2.sol:33: Inserted `royaltytest`<br/> &gt; New variables should be placed after all existing inherited variables</p> <p>contracts/mocks/TestTailorRoyaltiesV2.sol:165: Upgraded `__gap` to an incompatible type<br/> - Bad array resize from 47 to 46<br/> Size cannot decrease<br/> at assertStorageUpgradeSafe (node_modules/@openzeppelin/upgrades-core/src/storage/index.ts:35:11)<br/> at deployImpl (node_modules/@openzeppelin/hardhat-upgrades/src/utils/deploy-impl.ts:108:29)<br/> at async Proxy.upgradeProxy (node_modules/@openzeppelin/hardhat-upgrades/src/upgrade-proxy.ts:25:32)<br/> at async upgradeProxy (test/accessControl/Upgradability.test.ts:42:12)<br/> at async Context.&lt;anonymous&gt; (test/accessControl/Upgradability.test.ts:306:36)</p> <p>4) Upgradeability<br/> verify state of proxy after upgrade<br/> should upgrade (addHeritage) and still have the state in the proxy :<br/> Error: New storage layout is incompatible</p> <p>contracts/mocks/TestAppendHeritageContract.sol:21: Inserted `testHeritage`<br/> &gt; New variables should be placed after all existing inherited variables</p> <p>contracts/mocks/TestAppendHeritageContract.sol:22: Inserted `testMapping`<br/> &gt; New variables should be placed after all existing inherited variables</p> <p>contracts/mocks/TestAppendHeritageContract.sol:29: Replaced `__placeholder` with `_structHeritage` of incompatible type<br/> at assertStorageUpgradeSafe (node_modules/@openzeppelin/upgrades-core/src/storage/index.ts:35:11)<br/> at deployImpl (node_modules/@openzeppelin/hardhat-upgrades/src/utils/deploy-impl.ts:108:29)<br/> at async Proxy.upgradeProxy (node_modules/@openzeppelin/hardhat-upgrades/src/upgrade-proxy.ts:25:32)<br/> at async upgradeProxy (test/accessControl/Upgradability.test.ts:42:12)<br/> at async Context.&lt;anonymous&gt; (test/accessControl/Upgradability.test.ts:312:36)</p> <p>error Command failed with exit code 4.</p> |
| <b>Result/Recommendation</b> | Check these test cases and fix them to ensure full test coverage  |

## 6.2.5 Storing Data Via baseURI

Severity: INFORMATIONAL

Status: ACKNOWLEDGED

Code: NA

File(s) affected: TailorNFTCollectible.sol

Update: We intend to use IPFS (or another storage service like that) later on, but not right now as we need to learn more about what / how we want to store metadata/NFT data

|                              |   |
|------------------------------|---|
| <b>Attack / Description</b>  | In the current implementation the baseURI is not hardcoded or on-chain generated, means the owner/creator is free to choose the way how the metadata file is stored.  |
| <b>Code</b>                  | <p>Line 370 - 386 (TailorNFTCollectible.sol)</p> <pre>function setCollectionURI(     uint128 collectionId,     string calldata _baseURI,     string calldata _urimetadata ) external hasAccessRight(collectionId) {     Collections.Collection storage c = collections[collectionId];     require(bytes(c.baseURI).length &gt; 0, "Collection not found");     require(         (bytes(_baseURI).length &gt; 0) &amp;&amp; (bytes(_urimetadata).length &gt; 0),         "InvalidURIError"     );      c.baseURI = _baseURI;     c.metadataURI = _urimetadata;      emit CollectionURIChanged(collectionId, _baseURI, _urimetadata); }</pre> |
| <b>Result/Recommendation</b> | We recommend using IPFS and pinning services to make the metadata behind the baseURI permanently stored or implement into the event creation service.   |

|  |   |
|--|---|
|  | <p>To ensure that data persists on IPFS, and is not deleted during garbage collection, data can be pinned to one or more IPFS nodes. Pinning gives you control over disk space and data retention. As such, you should use that control to pin any content you wish to keep on IPFS indefinitely.</p> <p>Check more information here:<br/> <a href="https://docs.ipfs.io/concepts/persistence/#persistence-versus-permanence">https://docs.ipfs.io/concepts/persistence/#persistence-versus-permanence</a></p> <p>Keep in mind even if you use an IPFS Service, the file will only exist as long if it is “pinned”. And you still may need a dedicated gateway to serve your files with a decent speed, which may lead to your metadata requests timing out in the future.</p> <p>Please investigate SVG generated on-chain visuals and on-chain stored metadata, for persistent storage.</p> |
|--|---|

## 6.3 SWC Attacks

| ID                      | Title   | Relationships  | Test Result |
|-------------------------|---|--|-------------|
| <a href="#">SWC-131</a> | Presence of unused variables                      | <a href="#">CWE-1164: Irrelevant Code</a>  | ✓           |
| <a href="#">SWC-130</a> | Right-To-Left-Override control character (U+202E) | <a href="#">CWE-451: User Interface (UI) Misrepresentation of Critical Information</a> | ✓           |
| <a href="#">SWC-129</a> | Typographical Error                               | <a href="#">CWE-480: Use of Incorrect Operator</a>                                     | ✓           |


| ID                      | Title   | Relationships   | Test Result |
|-------------------------|---|---|-------------|
| <a href="#">SWC-128</a> | DoS With Block Gas Limit                            | <a href="#">CWE-400: Uncontrolled Resource Consumption</a>                | ✓           |
| <a href="#">SWC-127</a> | Arbitrary Jump with Function Type Variable          | <a href="#">CWE-695: Use of Low-Level Functionality</a>                   | ✓           |
| <a href="#">SWC-125</a> | Incorrect Inheritance Order                         | <a href="#">CWE-696: Incorrect Behavior Order</a>                         | ✓           |
| <a href="#">SWC-124</a> | Write to Arbitrary Storage Location                 | <a href="#">CWE-123: Write-what-where Condition</a>                       | ✓           |
| <a href="#">SWC-123</a> | Requirement Violation                               | <a href="#">CWE-573: Improper Following of Specification by Caller</a>    | ✓           |
| <a href="#">SWC-122</a> | Lack of Proper Signature Verification               | <a href="#">CWE-345: Insufficient Verification of Data Authenticity</a>   | ✓           |
| <a href="#">SWC-121</a> | Missing Protection against Signature Replay Attacks | <a href="#">CWE-347: Improper Verification of Cryptographic Signature</a> | ✓           |
| <a href="#">SWC-120</a> | Weak Sources of Randomness from Chain Attributes    | <a href="#">CWE-330: Use of Insufficiently Random Values</a>              | ✓           |
| <a href="#">SWC-119</a> | Shadowing State Variables                           | <a href="#">CWE-710: Improper Adherence to Coding Standards</a>           | ✗           |

| ID                      | Title                                | Relationships  | Test Result |
|-------------------------|--------------------------------------|--|-------------|
| <a href="#">SWC-118</a> | Incorrect Constructor Name           | <a href="#">CWE-665: Improper Initialization</a>   | ✓           |
| <a href="#">SWC-117</a> | Signature Malleability               | <a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>  | ✓           |
| <a href="#">SWC-116</a> | Timestamp Dependence                 | <a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>                                    | ✓           |
| <a href="#">SWC-115</a> | Authorization through tx.origin      | <a href="#">CWE-477: Use of Obsolete Function</a>  | ✓           |
| <a href="#">SWC-114</a> | Transaction Order Dependence         | <a href="#">CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')</a> | ✓           |
| <a href="#">SWC-113</a> | DoS with Failed Call                 | <a href="#">CWE-703: Improper Check or Handling of Exceptional Conditions</a>  | ✓           |
| <a href="#">SWC-112</a> | Delegatecall to Untrusted Callee     | <a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>                                    | ✓           |
| <a href="#">SWC-111</a> | Use of Deprecated Solidity Functions | <a href="#">CWE-477: Use of Obsolete Function</a>  | ✓           |
| <a href="#">SWC-110</a> | Assert Violation                     | <a href="#">CWE-670: Always-Incorrect Control Flow Implementation</a>  | ✓           |
| <a href="#">SWC-109</a> | Uninitialized Storage Pointer        | <a href="#">CWE-824: Access of Uninitialized Pointer</a>   | ✓           |


| ID                      | Title                                | Relationships  | Test Result |
|-------------------------|--------------------------------------|--|-------------|
| <a href="#">SWC-108</a> | State Variable Default Visibility    | <a href="#">CWE-710: Improper Adherence to Coding Standards</a>              | ✓           |
| <a href="#">SWC-107</a> | Reentrancy                           | <a href="#">CWE-841: Improper Enforcement of Behavioral Workflow</a>         | ✓           |
| <a href="#">SWC-106</a> | Unprotected SELFDESTRUCT Instruction | <a href="#">CWE-284: Improper Access Control</a>                             | ✓           |
| <a href="#">SWC-105</a> | Unprotected Ether Withdrawal         | <a href="#">CWE-284: Improper Access Control</a>                             | ✓           |
| <a href="#">SWC-104</a> | Unchecked Call Return Value          | <a href="#">CWE-252: Unchecked Return Value</a>                              | ✓           |
| <a href="#">SWC-103</a> | Floating Pragma                      | <a href="#">CWE-664: Improper Control of a Resource Through its Lifetime</a> | ✗           |
| <a href="#">SWC-102</a> | Outdated Compiler Version            | <a href="#">CWE-937: Using Components with Known Vulnerabilities</a>         | ✓           |
| <a href="#">SWC-101</a> | Integer Overflow and Underflow       | <a href="#">CWE-682: Incorrect Calculation</a>                               | ✓           |
| <a href="#">SWC-100</a> | Function Default Visibility          | <a href="#">CWE-710: Improper Adherence to Coding Standards</a>              | ✓           |

## 6.4. Verify Claims


6.4.1 The contract is using and ERC standard for NFTs

**Status:** tested and verified 

6.4.2 Royalties are correctly implemented and calculated

**Status:** tested and verified 

6.4.3 The smart contract is coded according to the newest standards and in a secure way.

**Status:** tested and verified 

## 7. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase.

The main goal of the audit was to verify the claims regarding the security and functions of the smart contract. During the audit, no critical, no high, no medium, 1 low and 4 informational issues have been found, after the manual and automated security testing. No necessary need for action, as the recommendations only further enhance the code's readability, not security.

## 8. Deployed Smart Contract

VERIFIED

Proxy: <https://polygonscan.com/address/0x7D202bD4759521CEfccCd27120b073AaaeB16733#code>

Implementation: <https://polygonscan.com/address/0xb7eec13aff29c592a59af3f6b96fa6c6e9dd2b1a#code>



## 9. About the Auditor

Chainsulting is a professional software development firm, founded in 20217 and based in Germany. They show ways, opportunities, risks and offer comprehensive blockchain solutions. Some of their services include blockchain development, smart contract audits and consulting.

Chainsulting conducts code audits on market-leading blockchains such as Hyperledger, Tezos, Ethereum, Binance Smart Chain, and Solana to mitigate risk and instil trust and transparency into the vibrant crypto community. They have also reviewed and secure the smart contracts of 1Inch, POA Network, Unicrypt, Amun, Furucombo among numerous other top DeFi projects.

Chainsulting currently secures [\\$100 billion](#) in user funds locked in multiple DeFi protocols. The team behind the leading audit firm relies on their robust technical know-how in the blockchain sector to deliver top-notch smart contract audit solutions, tailored to the clients' evolving business needs.

Check our website for further information: <https://chainsulting.de>

### How We Work



**1** -----

#### PREPARATION

Supply our team with audit ready code and additional materials



**2** -----

#### COMMUNICATION

We setup a real-time communication tool of your choice or communicate via e-mails.



**3** -----

#### AUDIT

We conduct the audit, suggesting fixes to all vulnerabilities and help you to improve.



**4** -----

#### FIXES

Your development team applies fixes while consulting with our auditors on their safety.



**5** -----

#### REPORT

We check the applied fixes and deliver a full report on all steps done.