UNBLOCKED

Smart Contract Audit

UNBLOCKED x SPOTIZ

Disclamer

Disclaimer 2 UNBLOCKED reports are not, or should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. UNBLOCKED do not cover testing or auditing the integration with external contract or services (such as OpenZeppelin, Uniswap, PancakeSwap etc’...)

UNBLOCKED Audits do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. UNBLOCKED Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort. UNBLOCKED Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. UNBLOCKED’s position is that each company and individual are responsible for their own due diligence and continuous security. UNBLOCKED in no way claims any guarantee of security or functionality of the technology we agree to analyze

Introduction

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, documenting any issues as there were discovered.

Scope

To ensure the correct version of the audited code, each file was hashed using the SHA256 hash function. The different hashes of the smart contracts audited are the following:

- **SpotizCollection.sol**
a8ad64983fd19e9adc290f291a91843ab426a1a32b619c948aab4a0092a320f6

- **SaleConfigurationStandard.sol**
854e9d3c6615bb95386988d42ff2e71c7bdfa89033cc71cc064183 18e631b6d3

- **HasRoyalties.sol**
5d6095fc1ac90710fe9338360d224d56cc16f64a351fc92a11903ca f9ef541b2

- **WhitelistingStandard.sol**
fb256d2dc632197a7ea0f41b9baa00c0e1942c6649cc0fa08cd44 d5f51ab281e

- **deployMainnet.ts**
83b0fe326a656fab35f0ead6392b9afd6cd873b359fdc669739e1 f968b96a518

Some libraries of opensea, chainlink and openZeppelin have been used. They are already audited, so this is out of the scope.

The auditing process follows a routine series of steps :

- Compilation
- Manuel Review
- MythX Static Analysis
- Code Style and best practices review
- Tests
- Initialization script when deploying the smar contract
- **TO BE VALIDATED BEFORE LAUNCH**

The specifications of smart contracts were based on the following details :

ICO dates : 16/01/2023 12:00 au 20/01/2023 12:00 NFT Total Supply : 7000 NFT

- Private 1200 NFT (250 hf)
- Public 5600 NFT (390chf)
- Team 100 NFT
- Advisor 50 NFT
- Partners 50 NFT

2 Wallets

- Spotiz Cost + Technical dev : Spotiz Holding Wallet 1
- Treso DAO : Wallet 2

Conditions of passage from private sale to public sale

- Depending on Spotiz marketing strategy. As owner of the contract, the Spotiz team can open or close the private sale at any time.

Conditions to end the public sale

- Depending on Spotiz marketing strategy. As owner of the contract, the Spotiz team can open or close the public sale at any time.

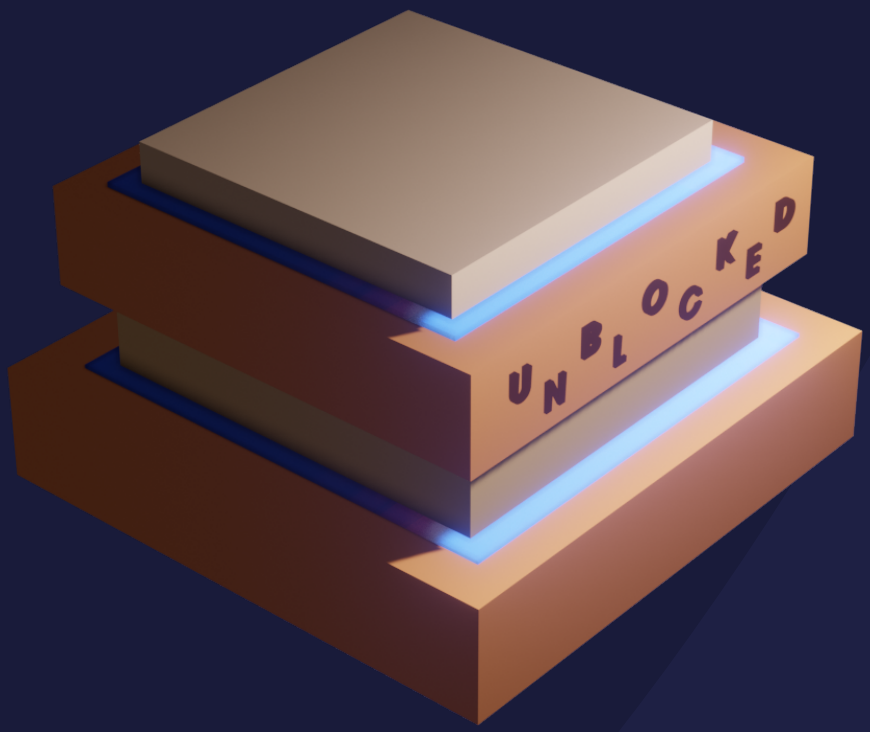
Compilation

```
[redacted]:~/projects/collectio-spotiz$ npm run compile

> collectio-spotiz@1.0.0 compile [redacted]/projects/collectio-spotiz
> npx hardhat compile

Nothing to compile
No need to generate any newer typings.
```

No warnings neither errors was founded.



Manual Review

Each issue has an assigned severity :

- Minor issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
- Medium issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
- Major issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- Critical issues are directly exploitable security vulnerabilities that need to be fixed.

SpotizCollection.sol

Medium Minor

No major/critical issues to mention but some recommendations and informations based on client specifications.

Recommandations

- **Function releaseFundsForAddress**

  Add an onlyOwner modifier to avoid DOS attacks from bots ! Dev team answer This method allows sale's stakeholders, which may not be only the contract's owner, to have a share of primary sale revenues. This is the reason why there is any modifier or restriction on who can call this entry point. Moreover, there are any storage manipulations extending mappings or arrays reachable from this entry point. Revenues are sent to an externally owned account so there is no way to make a DOS attack.

Manual Review

- **Function balanceOfUser** ✓

  This function returns an array of token Ids and no the exact number of tokens owned by user. Review the naming of the function or add (.length) to get the exact number of tokens.

  Dev team answer

  This method is implemented for integration's purposes. As a method's return, we provide an array of token IDs held by an user. In other words, it is his item's balance displayed as an array. For the user's balance, in terms of the total amount held, please refer to the ERC-721 standard method : balanceOf(address user).

- **Variable publicSaleHardcap** ✓

  Initialized by default to unlimited. Need to execute a specific function (`setPublicSaleHardcap`) to set this variable to the correct number. We recommend to initialize this variable by default to avoid transactions fees.

  Dev team answer

  This variable is set to unlimited since there are no specifications on the hard cap per user during the sales. However we offer the opportunity to review this hard cap depending on the Spotiz marketing strategy which could evolve overtime.

Manual Review

- **Variable privateSaleHardcap**

  Not initialized by default. Need to execute a specific function (setPrivateSaleHardcap) to set this variable. We recommend to initialize this variable by default to avoid transactions fees.

- **SaleConfigurationStandard.sol**

  No critical issues to mention but some recommendations and informations based on client specifications.

Questions to be answered

- **Variable maxPublicSaleMintPerTx** ✔

  This is not a client specification. This variable is initialized to 20, is it validated by the client ?

Dev team answer

This variable is not defined by client specification. We set it at 20 because of technical limitations due to the gas limit per transaction which allows us to mint only 20 items before reverting “out of gas” error.

Manual Review

- **Variable maxPrivateSaleMintPerT** ✔

This is not a client specification. This variable is initialized to 20, is it validated by the client ?

- **Variable totalReservedForAirdrop** ✔

This variable is initialized to 5800, is it validated by the client ?

I don't think ! Her are the specifications :

- Team 100 NFT
- Advisor 50 NFT
- Partners 50 NFT

=> **We recommend to initialize these variables by default to avoid transactions fees**

Dev team answer

This variable is set to 5800 in order to cap the allowed amount for private sale. We can apply two strategies to mimic a private sale with only public sale methods : - we can set the total maximum supply dynamically over the two sales. - we can reserve a portion of the supply with total supply reserved for the airdrop decremented dynamically over the two sales. Finally we decided to apply the first strategy to avoid computation errors for the Spotiz team when they will set configurations during sales. At deployment the totalReservedForAirdrop will be set to zero and the Spotiz team will define its value before the airdrop planned regarding the specifications

- **Function setPrivateSaleStatus** ✓
- **Function setPublicSaleStatus** ✓

HasRoyalties.sol

OK

WhitelistingStandard.sol ✓

No critical/major issues to mention but some recommendations and informations based on client specifications.

Questions to be answered

- How the workflow of whitelisting is working ?
- When you are going to use the whitelisting ?

Dev team answer

Private sale : All variables, methods and whitelisting processes referring to the private sale could not be used by the Spotiz team due to time constraints to integrate the whitelisting process in the application's back-end. However, if Spotiz delays its ICO, the developer team could provide the implementation allowing them to make use of private sale logic in dedicated variables and methods. For now, we deliver our product without taking into account private sale features. Spotiz team can mimic a private sale by using only public sales methods without a whitelisting process and managing the maximum total supply dynamically.

Code Style

Gas Optimization - variables initializations

To avoid waste of the gas fees by calling functions to configure the smart contract after deployment, initialize all the variables by default especially those (based on client specifications) :

NFT Total Supply

- 7000 NFT

HardCaps

- Private 1200 NFT
- Public 5600 NFT

Airdrop

- Team 100 NFT
- Advisor 50 NFT
- Partners 50 NFT

MythX Static analysis

0 ISSUE For SpotizCollection.sol

Source File	Scan Mode	Submitted at	Detected Vulnerabilities	Status	Actions
SpotizCollection.sol	Deep	10/01/2023 22:05:19	0 H 0 M 0 L	Finished	PDF report

Tests

```
Running 1 test for src/test/CollectionStandard.IntefaceId.state.t.sol:CollectionStandardInterfaceIdStateTest
[PASS] testFuzz_supportsInterface_func() (gas: 12450)
Test result: ok. 1 passed; 0 failed; finished in 2.95ms

Running 3 tests for src/test/CollectionStandard_checkTokenURI.t.sol:CollectionStandardCheckTokenURI
[PASS] testFuzz_setSeedManually_func(uint256) (runs: 2000, μ: 54530, ~: 55277)
[PASS] testFuzz_setSeedManually_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12402, ~: 12402)
[PASS] test_mintPublicSale_func_RevealWithoutMixing() (gas: 1284606)
Test result: ok. 3 passed; 0 failed; finished in 1.10s

Running 1 test for src/test/CollectionStandard.func.t.sol:CollectionStandardFuncTest
[PASS] test_soldOutDuringPublicSale_scenario() (gas: 1186368926)
Test result: ok. 1 passed; 0 failed; finished in 7.41s
Running 110 tests for src/test/CollectionStandard.t.sol:CollectionStandardTest
[PASS] testFuzz_airdrop_func(address[]) (runs: 2000, μ: 12043061, ~: 12143951)
[PASS] testFuzz_airdrop_func_withRevert_callerNotAllowedToAirdrop(address,address[]) (runs: 2000, μ: 44438, ~: 43612)
[PASS] testFuzz_airdrop_func_withRevert_exceedsReservedAllocation(address[]) (runs: 2000, μ: 170939, ~: 171049)
[PASS] testFuzz_airdrop_func_withRevert_exceedsReservedAllocationWithFullAirdrop(address[]) (runs: 2000, μ: 23790620, ~: 23791628)
[PASS] testFuzz_airdrop_func_withRevert_mintableHardcapReached(address[]) (runs: 2000, μ: 327832, ~: 327213)
[PASS] testFuzz_deleteDefaultRoyalty_func_withRevert_callerNotOwner(address) (runs: 2000, μ: 12484, ~: 12484)
[PASS] testFuzz_emergencyWithdraw_func_withRevert_callerNotOwner(address) (runs: 2000, μ: 12481, ~: 12481)
[PASS] testFuzz_freeze_func(uint16,uint8) (runs: 2000, μ: 944792, ~: 610135)
[PASS] testFuzz_freeze_func_withRevert_callerNotOwner(address,uint256[]) (runs: 2000, μ: 35303, ~: 35164)
[PASS] testFuzz_fulfillRandomness_internalFunc_success(uint256) (runs: 2000, μ: 149016, ~: 149016)
[PASS] testFuzz_getMetadata_afterReveal(uint16,uint8) (runs: 2000, μ: 870162, ~: 597521)
[PASS] testFuzz_getMetadata_beforeReveal(string,uint16,uint8) (runs: 2000, μ: 585955, ~: 393367)
[PASS] testFuzz_getMetadata_withRevert_nonExistentToken(uint256) (runs: 2000, μ: 11614, ~: 11614)
[PASS] testFuzz_mintPrivateSale_func(uint16,uint8) (runs: 2000, μ: 356504, ~: 289831)
[PASS] testFuzz_mintPrivateSale_func_withRevert_callerIsContract(uint256) (runs: 2000, μ: 11853, ~: 11853)
[PASS] testFuzz_mintPrivateSale_func_withRevert_callerNotWhitelisted(uint16,uint16,uint256) (runs: 2000, μ: 83222, ~: 83358)
[PASS] testFuzz_mintPrivateSale_func_withRevert_incorrectEthAmountSent(uint16,uint8) (runs: 2000, μ: 98012, ~: 98706)
[PASS] testFuzz_mintPrivateSale_func_withRevert_mintableHardcapReached(uint16,uint8) (runs: 2000, μ: 215337, ~: 216976)
[PASS] testFuzz_mintPrivateSale_func_withRevert_mintingQuantityTooHigh(uint16,uint256) (runs: 2000, μ: 48952, ~: 48952)
[PASS] testFuzz_mintPrivateSale_func_withRevert_privateSaleNotActivated(uint16,uint256) (runs: 2000, μ: 19946, ~: 19946)
[PASS] testFuzz_mintPublicSale_func(uint16,uint8) (runs: 2000, μ: 485214, ~: 309754)
[PASS] testFuzz_mintPublicSale_func_withRevert_callerIsContract(uint256) (runs: 2000, μ: 8780, ~: 8780)
[PASS] testFuzz_mintPublicSale_func_withRevert_incorrectEthAmountSent(uint16,uint8) (runs: 2000, μ: 61242, ~: 61939)
[PASS] testFuzz_mintPublicSale_func_withRevert_mintableHardcapReached(uint16,uint8) (runs: 2000, μ: 187309, ~: 188709)
[PASS] testFuzz_mintPublicSale_func_withRevert_mintingQuantityTooHigh(uint16,uint256) (runs: 2000, μ: 44779, ~: 44779)
[PASS] testFuzz_mintPublicSale_func_withRevert_publicSaleNotActivated(uint16,uint256) (runs: 2000, μ: 17171, ~: 17171)
[PASS] testFuzz_release_func_withRevert_accountHasNoShares(address) (runs: 2000, μ: 16163, ~: 16163)
[PASS] testFuzz_requestChainlinkVRF_func_withRevert_callerNotOwner(address) (runs: 2000, μ: 12575, ~: 12575)
[PASS] testFuzz_requestChainlinkVRF_func_withRevert_notEnoughLINKInsideContractBalance(uint256) (runs: 2000, μ: 42042, ~: 42042)
[PASS] testFuzz_resetRequestChainlinkVRF_func_withRevert_callerNotOwner(address) (runs: 2000, μ: 12485, ~: 12485)
[PASS] testFuzz_resetTokenRoyalty_func(uint256) (runs: 2000, μ: 10047, ~: 10047)
[PASS] testFuzz_resetTokenRoyalty_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12551, ~: 12551)
[PASS] testFuzz_setAirdropRole_func(address,bool) (runs: 2000, μ: 19039, ~: 13179)
[PASS] testFuzz_setAirdropRole_func_withRevert_callerNotOwner(address,address,bool) (runs: 2000, μ: 12986, ~: 12986)
[PASS] testFuzz_setBaseURI_func(string) (runs: 2000, μ: 71667, ~: 79704)
[PASS] testFuzz_setBaseURI_func_withRevert_callerNotOwner(address,string) (runs: 2000, μ: 13752, ~: 13763)
[PASS] testFuzz_setDefaultRoyalty_func(address,uint96) (runs: 2000, μ: 33891, ~: 33893)
[PASS] testFuzz_setDefaultRoyalty_func_withRevert_callerNotOwner(address,address,uint96) (runs: 2000, μ: 12983, ~: 12983)
[PASS] testFuzz_setLinkTokenPriceForVRF_func(uint256) (runs: 2000, μ: 30223, ~: 31049)
[PASS] testFuzz_setLinkTokenPriceForVRF_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12573, ~: 12573)
[PASS] testFuzz_setMaxMintPerTx_func(uint256,uint256) (runs: 2000, μ: 19655, ~: 20013)
[PASS] testFuzz_setMaxMintPerTx_func_withRevert_callerNotOwner(address,uint256,uint256) (runs: 2000, μ: 12635, ~: 12635)
[PASS] testFuzz_setMaxTotalSupply_func(uint256) (runs: 2000, μ: 13690, ~: 13885)
[PASS] testFuzz_setMaxTotalSupply_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12620, ~: 12620)
[PASS] testFuzz_setPreRevealURI_func(string) (runs: 2000, μ: 70927, ~: 79663)
[PASS] testFuzz_setPreRevealURI_func_withRevert_callerNotOwner(address,string) (runs: 2000, μ: 13705, ~: 13718)
[PASS] testFuzz_setPrivateSaleHardcap_func(uint256) (runs: 2000, μ: 13721, ~: 13928)
[PASS] testFuzz_setPrivateSaleHardcap_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12617, ~: 12617)
[PASS] testFuzz_setPrivateSalePrice_func(uint256) (runs: 2000, μ: 13780, ~: 13940)
[PASS] testFuzz_setPrivateSalePrice_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12662, ~: 12662)
```

Tests

```
testFuzz_setPublicSaleHardcap_func(uint256) (runs: 2000, μ: 13536, ~: 13873)
testFuzz_setPublicSaleHardcap_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12595, ~: 12595)
testFuzz_setPublicSalePrice_func(uint256) (runs: 2000, μ: 13704, ~: 13896)
testFuzz_setPublicSalePrice_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12553, ~: 12553)
testFuzz_setPublicSaleStatus_func(bool) (runs: 2000, μ: 17781, ~: 11354)
testFuzz_setPublicSaleStatus_func_withRevert_callerNotOwner(address,bool) (runs: 2000, μ: 12750, ~: 12750)
testFuzz_setSeedManually_func(uint256) (runs: 2000, μ: 54545, ~: 55431)
testFuzz_setSeedManually_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12705, ~: 12705)
testFuzz_setTokenRoyalty_func(uint256,address,uint96) (runs: 2000, μ: 34087, ~: 34089)
testFuzz_setTokenRoyalty_func_withRevert_callerNotOwner(address,uint256,address,uint96) (runs: 2000, μ: 13068, ~: 13068)
testFuzz_setTotalReservedForAirdrop_func(uint256) (runs: 2000, μ: 13605, ~: 13819)
testFuzz_setTotalReservedForAirdrop_func_withRevert_callerNotOwner(address,uint256) (runs: 2000, μ: 12596, ~: 12596)
testFuzz_setWhitelistSigner_func(address) (runs: 2000, μ: 31353, ~: 31353)
testFuzz_setWhitelistSigner_func_withRevert_callerNotOwner(address,address) (runs: 2000, μ: 12816, ~: 12816)
testFuzz_tokenURI_afterReveal(uint16,uint8) (runs: 2000, μ: 865260, ~: 597487)
testFuzz_tokenURI_beforeReveal(string,uint16,uint8) (runs: 2000, μ: 571981, ~: 348831)
testFuzz_tokenURI_withRevert_nonExistentToken(uint256) (runs: 2000, μ: 11537, ~: 11537)
testFuzz_verifyWhitelist_func(address) (runs: 2000, μ: 46721, ~: 46721)
testFuzz_verifyWhitelist_func_withRevert_invalidSignature_notSignedByWhitelistSigner(uint16) (runs: 2000, μ: 44480, ~: 44
testFuzz_verifyWhitelist_func_withRevert_invalidSignature_randomVRS(uint8,bytes32,bytes32) (runs: 2000, μ: 41874, ~: 4187
testFuzz_verifyWhitelist_func_withRevert_invalidSignature_signatureUsedByWrongCaller(address) (runs: 2000, μ: 46953, ~: 4
test_airdrop_func_fullyRedeemed() (gas: 23750953)
test_airdrop_func_partiallyRedeemed() (gas: 23630247)
test_deleteDefaultRoyalty_func() (gas: 9847)
test_emergencyWithdraw_func() (gas: 2394869)
test_freeze_func() (gas: 3621595)
test_fulfillRandomness_internalFunc_fail() (gas: 147965)
test_fulfillRandomness_internalFunc_success() (gas: 148533)
test_getMetadata_afterReveal() (gas: 3035394)
test_isRevealed_func_afterReveal() (gas: 151777)
test_isRevealed_func_beforeReveal() (gas: 8961)
] test_mintPrivateSale_func_mintOneNFT() (gas: 228547)
] test_mintPrivateSale_func_withRevert_individualHardcapReached(uint16) (runs: 2000, μ: 1292029, ~: 12920
] test_mintPrivateSale_func_withRevert_mintableHardcapReached() (gas: 215214)
] test_mintPrivateSale_func_zeroAmountToMint() (gas: 105191)
] test_mintPublicSale_func_mintOneNFT() (gas: 191398)
] test_mintPublicSale_func_withRevert_individualHardcapReached(uint16,uint8) (runs: 2000, μ: 72166, ~: 73
] test_mintPublicSale_func_withRevert_mintableHardcapReached() (gas: 186640)
] test_mintPublicSale_func_zeroAmountToMint() (gas: 65455)
] test_release_func() (gas: 2549879)
] test_release_func_withRevert_noPaymentDue() (gas: 39222)
] test_requestChainlinkVRF_func() (gas: 117327)
] test_requestChainlinkVRF_func_withRevert_cannotRequestVRFTwice() (gas: 118950)
] test_resetRequestChainlinkVRF_func() (gas: 11046)
] test_setAirdropRole_func() (gas: 32836)
] test_setBaseURI_func() (gas: 34232)
] test_setLinkTokenPriceForVRF_func() (gas: 30965)
] test_setMaxMintPerTx_func() (gas: 19959)
] test_setMaxTotalSupply_func() (gas: 13823)
] test_setPreRevealURI_func() (gas: 34254)
] test_setPrivateSaleHardcap_func() (gas: 13889)
] test_setPrivateSalePrice_func() (gas: 13852)
] test_setPrivateSaleStatus_func() (gas: 31152)
] test_setPublicSalePrice_func() (gas: 13788)
] test_setPublicSaleStatus_func() (gas: 31100)
] test_setSeedManually_func() (gas: 55324)
] test_setTotalReservedForAirdrop_func() (gas: 13731)
] test_setWhitelistSigner_func() (gas: 31188)
result: ok. 110 passed; 0 failed; finished in 90.35s
```

Initialization Script

deployMainnet.ts

Dev team specification for deployment process

Initialization respect specifications and are visible in the deployMainnet.ts script. Primary market revenues and secondary market fees will be splitted manually by the Spotiz team.

Warning :

- Use the correct addresses for chainlink settings.
- After deploying the smart contract, please check the transfer of the ownership to the spotiz team

```
/** @dev DEPLOY VALUES **/
const initialPrivateSalePrice = ethers.utils.parseEther("0.2");
const initialPublicSalePrice = 0;
const royaltyFeesReceiver = "0x9956606B6351c0DCfb07A6C6db1cf5e294e7DF12";
const royaltyFeesNumerator = 700;
const name = "Spotiz Collection";
const symbol = "SPOT";
const newMaxTotalSupply = 1200;
/** @dev Rinkeby Settings **/
const chainlink = {
  vrfCoordinatorAddress: "0x271682DEB8C4E0901D1a1550aD2e64D568E69909",
  linkTokenAddress: "0x514910771AF9Ca656af840dff83E8264EcF986CA",
  keyHash: "0xff8dedfbfa60af186cf3c830acbc32c05aae823045ae5ea7da1e45fbfaba4f92", // 500 gwei
  linkTokenPriceForVRF: ethers.utils.parseEther("0.25"),
};
/** @dev Revenue sharing for sales, need at least one payee. **/
/* If there is more than one payee you must ensure the sum of shares (expressed as percentage) is equal to 100 */
const revenueShare = { payees: ["0x9956606B6351c0DCfb07A6C6db1cf5e294e7DF12"], shares: [100] };
```

TO BE VALIDATED BEFORE LAUNCH

Before Launch we need a validation of all these points from the Spotiz team to validate the audit.

1. Conditions of passage from private sale to public sale ✔
2. Conditions to end the public sale ✔
3. ICO Dates ✔
4. Function setPrivateSaleStatus ✔
5. Function setPublicSaleStatus ✔
6. Wallets primary market ✔

Based on the deployMainnet.ts script the wallet which will receive 100% of the funds is 0x9956606B6351c0DCfb07A6C6db1cf5e294e7DF12

The split will be done manually by the Spotiz team

7. Secondary market fees ✔

Based on the deployMainnet.ts script the wallet which will receive 7% of the secondary fees is 0x9956606B6351c0DCfb07A6C6db1cf5e294e7DF12

TO BE VALIDATED BEFORE LAUNCH

Before Launch we need a validation of all these points from the Spotiz team to validate the audit.

8. Collection details ✔

Based on the deployMainnet.ts script here are the details to be validated :

name = "Spotiz Collection"
symbol = "SPOT";
newMaxTotalSupply = 1200;

Disclosure

The report and the analysis described therein are created solely for the client and published with their consent. The scope of our review is limited to a review of the Solidity code and only the Solidity code that we indicate as part of the scope of our review in this report. The Solidity language itself remains under development and is subject to unknown risks and defects. The review does not extend to the compiler layer, nor to other areas beyond Solidity that may present security risks. Cryptographic tokens are emerging technologies and carry high levels of technical risk and uncertainty.

UNBLOCKED