Smart Contract Security Audit

Audit details:

Audited project:	DragonBall
Deployer address	0x44dffb00c0af919f8637120bcf85109ce517c1ae
Blockchain:	Binance Smart Chain
Project website:	https://dragonballfinance.org

April, 2021
TechRate

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and TechRate hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

TechRate was commissioned by DragonBall to perform an audit of smart contracts:

- *https://bscscan.com/address/0xceB2f5e9C7F2D3BCd12A7560D73c56f3396af3F9#code*
- *https://bscscan.com/address/0xdbdb324fbf3cf6456358bd862a2aa81147a55e9a#code*
- *https://bscscan.com/address/0x6679d8e3cda6441d5c5c12691c3de9e47d80663d#code*
- *https://bscscan.com/address/0xa589a74d31e503073d7c7932a67abd3f20f9eeb4#code*

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts details

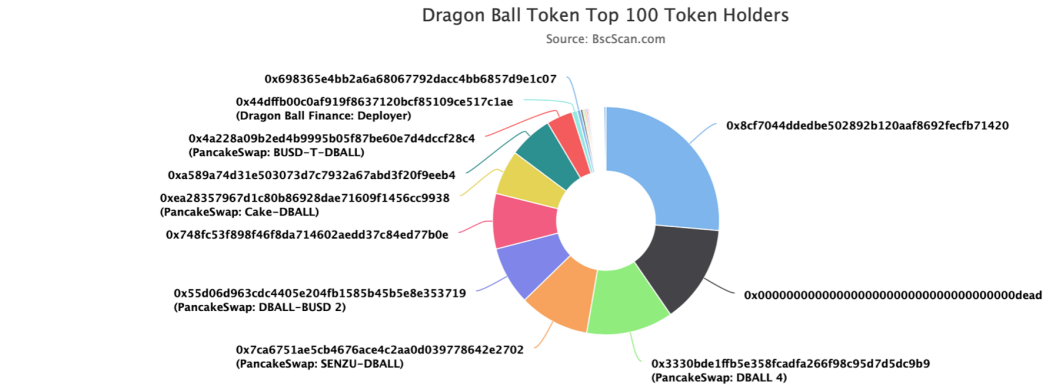
Token contract details for 29.04.2021.

Contract name:	DragonBall
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0xceB2f5e9C7F2D3BCd12A7560D73c56f3396af3F9
Total supply:	1_646_047_317_075_660_936_084_057
Token ticker:	DBALL
Decimals:	18
Token holders:	525
Transactions count:	132942
Top 100 holders dominance:	99.71 %
Contract deployer address:	0x44dffb00c0af919f8637120bcf85109ce517c1ae
Contract’s current owner address:	0x8cf7044ddedbe502892b120aaf8692fecfb71420

DragonBall top 100 token distribution

The top 100 holders collectively own 99.71% (1,641,319.70 Tokens) of Dragon Ball Token

Token Total Supply: 1,646,047.32 Token | Total Token Holders: 525

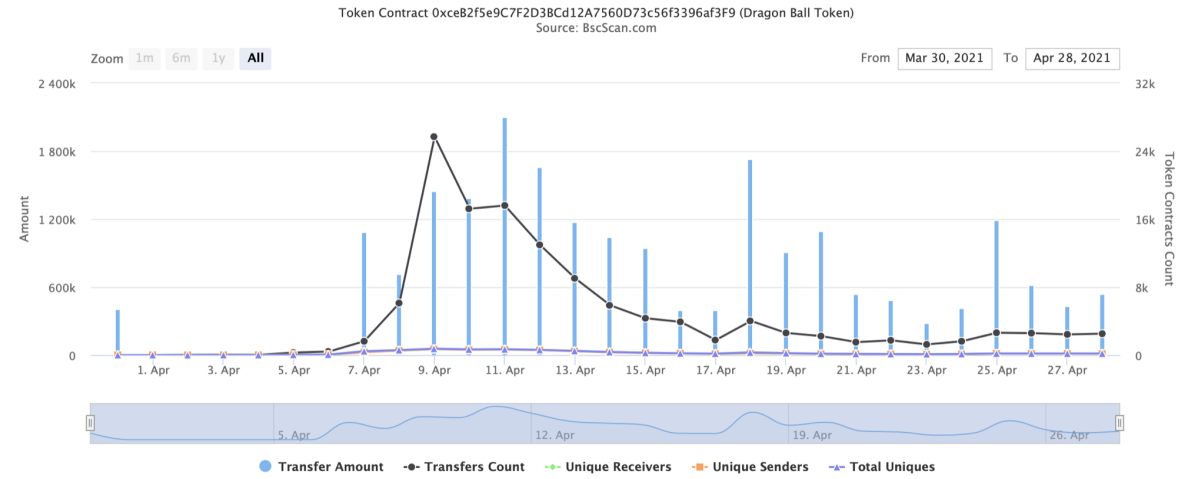


(A total of 1,641,319.70 tokens held by the top 100 accounts from the total supply of 1,646,047.32 token)

DragonBall contract interaction details

Time Series: Token Contract Overview

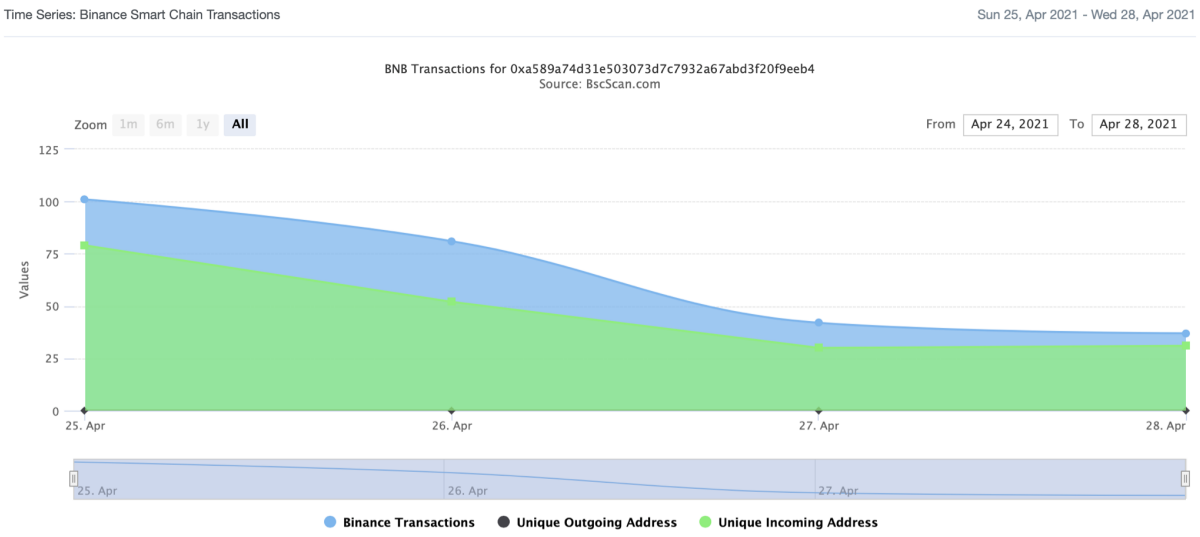
Wed 31, Mar 2021 - Wed 28, Apr 2021



DragonBall top 10 token holders

Rank	Address	Quantity (Token)	Percentage
1	0x8cf7044ddedbe502892b120aaf8692fecfb71420	434,401.720761089055754005	26.3906%
2	0x000000000000000000000000000000000000dead	230,223.03906965550551858	13.9864%
3	PancakeSwap: DBALL 4	203,600.08287130670821292	12.3690%
4	PancakeSwap: SENZU-DBALL	164,810.334900901799802778	10.0125%
5	PancakeSwap: DBALL-BUSD 2	135,859.14554908756855389	8.2537%
6	0x748fc53f898f46f8da714602aedd37c84ed77b0e	129,403.476636915575669071	7.8615%
7	PancakeSwap: Cake-DBALL	103,836.028880860055834844	6.3082%
8	0xa589a74d31e503073d7c7932a67abd3f20f9eeb4	102,553.98947857180756303	6.2303%
9	PancakeSwap: BUSD-T-DBALL	61,272.81253013172314186	3.7224%
10	Dragon Ball Finance: Deployer	12,471.553301582617437713	0.7577%

SmartChef transactions



SmartChef contract details for 29.04.2021.

Contract name:	SmartChef
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0xa589a74d31e503073d7c7932a67abd3f20f9eeb4
Reward token:	0xbb4cdb9cbd36b01bd1cbaebf2de08d9173bc095c
Syrup	0xceb2f5e9c7f2d3bcd12a7560d73c56f3396af3f9
Reward per block:	36_517_361_110_000
Start block:	6869346
Burn multiplier:	30
Bonus end block:	7009176
Owner:	0x44dffb00c0af919f8637120bcf85109ce517c1ae

SmartChef contract Pools info:

Pool with id 0:

lpToken *address* : 0xceB2f5e9C7F2D3BCd12A7560D73c56f3396af3F9
allocPoint *uint256* : 1000
lastRewardBlock *uint256* : 6982704
accRewardPerShare *uint256* : 130975567983215

DragonBallFactory functions outline

+ [Int] IDragonBallFactory
  - [Ext] feeTo
  - [Ext] feeToSetter
  - [Ext] getPair
  - [Ext] allPairs
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #
  - [Ext] setFeeToSetter #

+ [Int] IDragonBallPair
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] DOMAIN_SEPARATOR
  - [Ext] PERMIT_TYPEHASH
  - [Ext] nonces
  - [Ext] permit #
  - [Ext] MINIMUM_LIQUIDITY
  - [Ext] factory
  - [Ext] token0
  - [Ext] token1
  - [Ext] getReserves
  - [Ext] price0CumulativeLast
  - [Ext] price1CumulativeLast
  - [Ext] kLast
  - [Ext] mint #
  - [Ext] burn #
  - [Ext] swap #
  - [Ext] skim #
  - [Ext] sync #
  - [Ext] initialize #

+ [Int] IDragonBallBEP20
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply

  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] DOMAIN_SEPARATOR
  - [Ext] PERMIT_TYPEHASH
  - [Ext] nonces
  - [Ext] permit #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] mul

+ DragonBallBEP20 (IDragonBallBEP20)
  - [Pub] <Constructor> #
  - [Int] _mint #
  - [Int] _burn #
  - [Prv] _approve #
  - [Prv] _transfer #
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] permit #

+ [Lib] Math
  - [Int] min
  - [Int] sqrt

+ [Lib] UQ112x112
  - [Int] encode
  - [Int] uqdiv

+ [Int] IBEP20
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #

+ [Int] IDragonBallCallee
  - [Ext] DragonBallCall #

+ DragonBallPair (IDragonBallPair, DragonBallBEP20)
  - [Pub] getReserves
  - [Prv] _safeTransfer #
  - [Pub] <Constructor> #
  - [Ext] initialize #
  - [Prv] _update #
  - [Prv] _mintFee #
  - [Ext] mint #
    - modifiers: lock
  - [Ext] burn #
    - modifiers: lock
  - [Ext] swap #
    - modifiers: lock
  - [Ext] skim #
    - modifiers: lock
  - [Ext] sync #
    - modifiers: lock

+ DragonBallFactory (IDragonBallFactory)
  - [Pub] <Constructor> #
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #
  - [Ext] setFeeToSetter #

($) = payable function
= non-constant function

DragonBallRouter functions outline

+ [Int] IDragonBallFactory
  - [Ext] feeTo
  - [Ext] feeToSetter
  - [Ext] getPair
  - [Ext] allPairs
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #
  - [Ext] setFeeToSetter #

+ [Lib] TransferHelper
  - [Int] safeApprove #
  - [Int] safeTransfer #
  - [Int] safeTransferFrom #
  - [Int] safeTransferBNB #

+ [Int] IDragonBallRouter01
  - [Ext] factory
  - [Ext] WBNB
  - [Ext] addLiquidity #
  - [Ext] addLiquidityBNB ($)
  - [Ext] removeLiquidity #
  - [Ext] removeLiquidityBNB #
  - [Ext] removeLiquidityWithPermit #
  - [Ext] removeLiquidityBNBWithPermit #
  - [Ext] swapExactTokensForTokens #
  - [Ext] swapTokensForExactTokens #
  - [Ext] swapExactBNBForTokens ($)
  - [Ext] swapTokensForExactBNB #
  - [Ext] swapExactTokensForBNB #
  - [Ext] swapBNBForExactTokens ($)
  - [Ext] quote
  - [Ext] getAmountOut
  - [Ext] getAmountIn
  - [Ext] getAmountsOut
  - [Ext] getAmountsIn

+ [Int] IDragonBallRouter02 (IDragonBallRouter01)
  - [Ext] removeLiquidityBNBSupportingFeeOnTransferTokens #
  - [Ext] removeLiquidityBNBWithPermitSupportingFeeOnTransferTokens #
  - [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
  - [Ext] swapExactBNBForTokensSupportingFeeOnTransferTokens ($)
  - [Ext] swapExactTokensForBNBSupportingFeeOnTransferTokens #

+ [Int] IDragonBallPair
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] DOMAIN_SEPARATOR
  - [Ext] PERMIT_TYPEHASH
  - [Ext] nonces
  - [Ext] permit #
  - [Ext] MINIMUM_LIQUIDITY
  - [Ext] factory
  - [Ext] token0
  - [Ext] token1
  - [Ext] getReserves
  - [Ext] price0CumulativeLast
  - [Ext] price1CumulativeLast
  - [Ext] kLast
  - [Ext] mint #
  - [Ext] burn #
  - [Ext] swap #
  - [Ext] skim #
  - [Ext] sync #
  - [Ext] initialize #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] mul

+ [Lib] DragonBallLibrary
  - [Int] sortTokens
  - [Int] pairFor
  - [Int] getReserves
  - [Int] quote
  - [Int] getAmountOut
  - [Int] getAmountIn
  - [Int] getAmountsOut
  - [Int] getAmountsIn

+ [Int] IBEP20
  - [Ext] name
  - [Ext] symbol

- [Ext] decimals
- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] allowance
- [Ext] approve #
- [Ext] transfer #
- [Ext] transferFrom #

+ [Int] IWBNB
- [Ext] deposit ($)
- [Ext] transfer #
- [Ext] withdraw #

+ DragonBallRouter (IDragonBallRouter02)
- [Pub] <Constructor> #
- [Ext] <Fallback> ($)
- [Int] _addLiquidity #
- [Ext] addLiquidity #
  - modifiers: ensure
- [Ext] addLiquidityBNB ($)
  - modifiers: ensure
- [Pub] removeLiquidity #
  - modifiers: ensure
- [Pub] removeLiquidityBNB #
  - modifiers: ensure
- [Ext] removeLiquidityWithPermit #
- [Ext] removeLiquidityBNBWithPermit #
- [Pub] removeLiquidityBNBSupportingFeeOnTransferTokens #
  - modifiers: ensure
- [Ext] removeLiquidityBNBWithPermitSupportingFeeOnTransferTokens #
- [Int] _swap #
- [Ext] swapExactTokensForTokens #
  - modifiers: ensure
- [Ext] swapTokensForExactTokens #
  - modifiers: ensure
- [Ext] swapExactBNBForTokens ($)
  - modifiers: ensure
- [Ext] swapTokensForExactBNB #
  - modifiers: ensure
- [Ext] swapExactTokensForBNB #
  - modifiers: ensure
- [Ext] swapBNBForExactTokens ($)
  - modifiers: ensure
- [Int] _swapSupportingFeeOnTransferTokens #
- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
  - modifiers: ensure
- [Ext] swapExactBNBForTokensSupportingFeeOnTransferTokens ($)

    - modifiers: ensure
  - [Ext] swapExactTokensForBNBSupportingFeeOnTransferTokens #
    - modifiers: ensure
  - [Pub] quote
  - [Pub] getAmountOut
  - [Pub] getAmountIn
  - [Pub] getAmountsOut
  - [Pub] getAmountsIn

($) = payable function
= non-constant function

DragonBallRouter functions outline

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

+ [Int] IBEP20
  - [Ext] totalSupply
  - [Ext] decimals
  - [Ext] symbol
  - [Ext] name
  - [Ext] getOwner
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Int] functionStaticCall
  - [Int] functionStaticCall
  - [Prv] _verifyCallResult

+ [Lib] SafeBEP20
  - [Int] safeTransfer #
  - [Int] safeTransferFrom #
  - [Int] safeApprove #
  - [Int] safeIncreaseAllowance #
  - [Int] safeDecreaseAllowance #
  - [Prv] _callOptionalReturn #

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ Ownable (Context)
  - [Int] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner

+ SmartChef (Ownable)
  - [Pub] <Constructor> #
  - [Pub] stopReward #
    - modifiers: onlyOwner
  - [Pub] adjustBlockEnd #
    - modifiers: onlyOwner
  - [Pub] getMultiplier
  - [Ext] pendingReward
  - [Pub] updatePool #
  - [Pub] deposit #
  - [Pub] withdraw #
  - [Pub] emergencyWithdraw #

($) = payable function
= non-constant function

Issues Checking Status

№	Issue description.	Checking status
1	Compiler errors.	Passed
2	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Economy model of the contract.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Open Zeppelin contracts implementation and usage.	Passed
21	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Integer underflow possibility

Issue:

Possibility of integer underflow if the burn multiplier variable is more than 1000.

```
function deposit(uint256 _amount⬆) public {
    PoolInfo storage pool = poolInfo[0];
    UserInfo storage user = userInfo[msg.sender];
    require(pool.lpToken.balanceOf(address(this)) <= maxDeposit,"Deposit limit reached!!");
    updatePool(0);
    if (user.amount > 0) {
        uint256 pending = user.amount.mul(pool.accRewardPerShare).div(1e18).sub(user.rewardDebt);
        if(pending > 0) {
            user.rewardDebt = user.amount.mul(pool.accRewardPerShare).div(1e18);
            rewardToken.safeTransfer(address(msg.sender), pending);
        }
    }
    if(_amount⬆ > 0) {
        uint256 burnAmount = _amount⬆.mul(burnMultiplier).div(1000);
        pool.lpToken.safeTransferFrom(address(msg.sender), address(this), _amount⬆ - burnAmount);
        if (burnAmount > 0) {
            pool.lpToken.safeTransferFrom(address(msg.sender), address(0x00dead), burnAmount);
        }
        user.amount = user.amount.add(_amount⬆ - burnAmount);
    }
    user.rewardDebt = user.amount.mul(pool.accRewardPerShare).div(1e18);

    emit Deposit(msg.sender, _amount⬆);
}
```

Recommendation:

Use the SafeMath library. In this case the bur multiplier is 30 and there will not be any integer underflow, but if you will redeploy it with another data, there could occur this issue.

Owner privileges

- ❏ Owner can stop the reward in the SmartChef contract.
- ❏ Owner can adjust the block end.

Conclusion

Smart contracts do not contain any high severity issues!

Techrate note:
Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.