0.7

Pancake Bunny Process Quality Review

Score: 32%

Overview

This is a Process Quality Review of PancakeBunny completed on May 4, 2021. It was performed using the Process Review process (version 0.7) and is documented here. The review was performed by Lucas of DeFiSafety. Check out our Telegram.

The final score of the review is 32%, a Clear Fail. The breakdown of the scoring is in Scoring Appendix. For our purposes, a pass is 70%.

Summary of the Process

Very simply, the review looks for the following declarations from the developer's site. With these declarations, it is reasonable to trust the smart contracts.

- **Here are my smart contracts on the blockchain**
- **Here is the documentation that explains what my smart contracts do**
- **Here are the tests I ran to verify my smart contract**
- **Here are the audit(s) performed on my code by third party experts**
- **Here are the admin controls and strategies**

Disclaimer

This report is for informational purposes only and does not constitute investment advice of any kind, nor does it constitute an offer to provide investment advisory or other services. Nothing in this report shall be considered a solicitation or offer to buy or sell any security, token, future, option or other financial instrument or to offer or provide any investment advice or service to any person in any jurisdiction. Nothing contained in this report constitutes investment advice or offers any opinion with respect to the suitability of any security, and the views expressed in this report should not be taken as advice to buy, sell or hold any security. The information in this report should not be relied upon for the purpose of investing. In preparing the information contained in this report, we have not taken into account the investment needs, objectives and financial circumstances of any particular investor. This information has no regard to the specific investment objectives, financial situation and particular needs of any specific recipient of this information and investments discussed may not be suitable for all investors.

Any views expressed in this report by us were prepared based upon the information available to us at the time such views were written. The views expressed within this report are limited to DeFiSafety and the author and do not reflect those of any additional or third party and are strictly based upon DeFiSafety, its authors, interpretations and evaluation of relevant data. Changed or additional information could cause such

Chain

This section indicates the blockchain used by this protocol.

✓ **Chain: Binance**

Guidance:
Ethereum
Binance

Code and Team

This section looks at the code deployed on the Mainnet that gets reviewed and its corresponding software repository. The document explaining these questions is here. This review will answer the questions;

1) Are the executing code addresses readily available? (%)
2) Is the code actively being used? (%)
3) Is there a public software repository? (Y/N)
4) Is there a development history visible? (%)
5) Is the team public (not anonymous)? (Y/N)

1) Are the executing code addresses readily available? (%)

✓ Answer: 100%

They are available at website https://pancakebunny-finance.readthedocs.io/en/main/contracts.html as indicated in the Appendix.

Guidance:

100%	Clearly labelled and on website, docs or repo, quick to find
70%	Clearly labelled and on website, docs or repo but takes a bit of looking
40%	Addresses in mainnet.json, in discord or sub graph, etc
20%	Address found but labelling not clear or easy to find
0%	Executing addresses could not be found

How to improve this score

Make the Ethereum addresses of the smart contract utilized by your application available on either your website or your GitHub (in the README for instance). Ensure the addresses is up to date. This is a very important question wrt to the final score.

2) Is the code actively being used? (%)

✓ Answer: 100%

Activity is 1617 transactions a day on contract *AdminUpgradabilityProxy.sol*, as indicated in the Appendix.

Percentage Score Guidance

100%	More than 10 transactions a day
70%	More than 10 transactions a week
40%	More than 10 transactions a month
10%	Less than 10 transactions a month
0%	No activity

3) Is there a public software repository? (Y/N)

✓ Answer: Yes

GitHub: https://github.com/PancakeBunny-finance/Bunny

Is there a public software repository with the code at a minimum, but normally test and scripts also (Y/N). Even if the repo was created just to hold the files and has just 1 transaction, it gets a Yes. For teams with private repos, this answer is No.

4) Is there a development history visible? (%)

✓ Answer: 100%

With 118 commits and 2 branches, this is clearly a well-researched document.

This checks if the software repository demonstrates a strong steady history. This is normally demonstrated by commits, branches and releases in a software repository. A healthy history demonstrates a history of more than a month (at a minimum).

Guidance:

100%	Any one of 100+ commits, 10+branches
70%	Any one of 70+ commits, 7+branches
50%	Any one of 50+ commits, 5+branches

30%	Any one of 30+ commits, 3+branches
0%	Less than 2 branches or less than 10 commits

5) Is the team public (not anonymous)? (Y/N)

> ⚠ Answer: No

The team behind PancakeBunny is Anonymous.

For a yes in this question the real names of some team members must be public on the website or other documentation. If the team is anonymous and then this question is a No.

Documentation

This section looks at the software documentation. The document explaining these questions is here.

Required questions are;

6) Is there a whitepaper? (Y/N)
7) Are the basic software functions documented? (Y/N)
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)
10) Is it possible to trace from software documentation to the implementation in code (%)

6) Is there a whitepaper? (Y/N)

> ✓ Answer: Yes

Location: https://pancakebunny-finance.readthedocs.io/en/main/bunnytoken.html

7) Are the basic software functions documented? (Y/N)

> ⚠ Answer: No

There does not seem to be any evident software function documentation.

How to improve this score

Write the document based on the deployed code. For guidance, refer to the SecurEth System Description Document.

8) Does the software function documentation fully (100%) cover the deployed contracts? (%)

> ⚠ Answer: 0%

With no evident software function documentation, it does not fully cover the deployed contracts.

Guidance:

100%	All contracts and functions documented
80%	Only the major functions documented
79-1%	Estimate of the level of software documentation
0%	No software documentation

How to improve this score

This score can improve by adding content to the requirements document such that it comprehensively covers the requirements. For guidance, refer to the SecurEth System Description Document . Using tools that aid traceability detection will help.

9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)

> ⚠ Answer: 0%

Code examples are in the Appendix. As per the SLOC, there is 19% commenting to code (CtC).

The Comments to Code (CtC) ratio is the primary metric for this score.

Guidance:

100%	CtC > 100 Useful comments consistently on all code
90-70%	CtC > 70 Useful comment on most code
60-20%	CtC > 20 Some useful commenting
0%	CtC < 20 No useful commenting

How to improve this score

This score can improve by adding comments to the deployed code such that it comprehensively covers the code. For guidance, refer to the SecurEth Software Requirements.

10) Is it possible to trace from software documentation to the implementation in code (%)

⚠ Answer: 0%

There is no evident software function documentation.

Guidance:

100%	Clear explicit traceability between code and documentation at a requirement level for all code
60%	Clear association between code and documents via non explicit traceability
40%	Documentation lists all the functions and describes their functions
0%	No connection between documentation and code

How to improve this score

This score can improve by adding traceability from requirements to code such that it is clear where each requirement is coded. For reference, check the SecurEth guidelines on traceability.

Testing

This section looks at the software testing available. It is explained in this document. This section answers the following questions;

11) Full test suite (Covers all the deployed code) (%)
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)
13) Scripts and instructions to run the tests (Y/N)
14) Report of the results (%)
15) Formal Verification test done (%)
16) Stress Testing environment (%)

11) Is there a Full test suite? (%)

⚠ Answer: 12%

With a TtC of 12%, there is a severely limited number of tests availbile.

This score is guided by the Test to Code ratio (TtC). Generally a good test to code ratio is over 100%. However the reviewers best judgement is the final deciding factor.

Guidance:

100%	TtC > 120% Both unit and system test visible
80%	TtC > 80% Both unit and system test visible
40%	TtC < 80% Some tests visible
0%	No tests obvious

How to improve this score

This score can improve by adding tests to fully cover the code. Document what is covered by traceability or test results in the software repository.

12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)

> ⓘ Answer: 30%

Some tests are evident, but not clearly not complete.

Guidance:

100%	Documented full coverage
99-51%	Value of test coverage from documented results
50%	No indication of code coverage but clearly there is a reasonably complete set of tests
30%	Some tests evident but not complete
0%	No test for coverage seen

How to improve this score

This score can improve by adding tests achieving full code coverage. A clear report and scripts in the software repository will guarantee a high score.

13) Scripts and instructions to run the tests (Y/N)

> ⚠ Answer: No

There is no evidence of scripts and instructions to run the tests.

How to improve this score

Add the scripts to the repository and ensure they work. Ask an outsider to create the environment and run the tests. Improve the scripts and docs based on their feedback.

14) Report of the results (%)

> ⚠ Answer: 0%

Guidance:

100%	Detailed test report as described below
70%	GitHub Code coverage report visible
0%	No test report evident

How to improve this score

Add a report with the results. The test scripts should generate the report or elements of it.

15) Formal Verification test done (%)

⚠ Answer: 0%

16) Stress Testing environment (%)

⚠ Answer: 0%

Security

This section looks at the 3rd party software audits done. It is explained in this document. This section answers the following questions;

17) Did 3rd Party audits take place? (%)
18) Is the bounty value acceptably high?

17) Did 3rd Party audits take place? (%)

ⓘ Answer: 40%

Haechi preformed an audit on PancakeBunny. The audit meets the reqs for a 70%. But, on reading the audit it is very thin and mentions a big weakness on code not within the scope of the audit. Audit score reduced from 70% to 40%.

Guidance:

100%	Multiple Audits performed before deployment and results public and implemented or not required
90%	Single audit performed before deployment and results public and implemented or not required
70%	Audit(s) performed after deployment and no changes required. Audit report is public
20%	No audit performed

0% Audit Performed after deployment, existence is public, report is not public and no improvements deployed OR smart contract address' not found, question

18) Is the bounty value acceptably high (%)

⚠ Answer: 0%

No bug bounty program offered.

Bug Bounty Location:

Guidance:

100%	Bounty is 10% TVL or at least 1M
90%	Bounty is 5% TVL or at least 500k
70%	Bounty is 100k or over
40%	Bounty is 50k or over
20%	Bug bounty program bounty is less than 50k
0%	No bug bounty program offered

Access Controls

This section covers the documentation of special access controls for a DeFi protocol. The admin access controls are the contracts that allow updating contracts or coefficients in the protocol. Since these contracts can allow the protocol admins to "change the rules", complete disclosure of capabilities is vital for user's transparency. It is explained in this document. The questions this section asks are as follow;

20) Can a user clearly and quickly find the status of the admin controls?
21) Is the information clear and complete?
22) Is the information in non-technical terms that pertain to the investments?
23) Is there Pause Control documentation including records of tests?

19) Can a user clearly and quickly find the status of the admin controls (%)

✓ Answer: 100%

Governance section is clearly labeled in the wiki.

Location: https://pancakebunny-finance.readthedocs.io/en/main/governance.html

Guidance:

100%	Clearly labelled and on website, docs or repo, quick to find
70%	Clearly labelled and on website, docs or repo but takes a bit of looking
40%	Access control docs in multiple places and not well labelled

20%	Access control docs in multiple places and not labelled
0%	Admin Control information could not be found

NOTE: DefiSafety does not consider any information found on third party audits as admissible for scoring.

20) Is the information clear and complete (%)

⚠ Answer: 10%

No indication of which contracts are upgradeable. Roles are only mentioned in the audit, not in protocol docs. The degree of change an owner can enact is not defined.

All contracts are clearly labelled as upgradeable (or not) -- 0%
The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 10%
The capabilities for change in the contracts are described -- 0%

Guidance:
All the contracts are immutable -- 100% OR

All contracts are clearly labelled as upgradeable (or not) -- 30% AND
The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 30% AND
The capabilities for change in the contracts are described -- 30%

How to improve this score

Create a document that covers the items described above. An example is enclosed.

21) Is the access control information in non-technical terms that pertain to the investments (%)

⚠ Answer: 0%

The degree of change an owner can enact is not defined. The impact these changes for investors are not defined.

Guidance:

100%	All the contracts are immutable
90%	Description relates to investments safety and updates in clear, complete non-software l language
30%	Description all in software specific language
0%	No admin control information could not be found

How to improve this score

Create a document that covers the items described above in plain language that investors can understand. An example is enclosed.

22) Is there Pause Control documentation including records of tests (%)

Answer: 0%

Guidance:

100%	All the contracts are immutable or no pause control needed and this is explained OR
100%	Pause control(s) are clearly documented and there is records of at least one test within 3 months
80%	Pause control(s) explained clearly but no evidence of regular tests
40%	Pause controls mentioned with no detail on capability or tests
0%	Pause control not documented or explained

How to improve this score

Create a document that covers the items described above in plain language that investors can understand. An example is enclosed.

Appendices

Author Details

The author of this review is Rex of DeFi Safety.

Email : rex@defisafety.com Twitter : @defisafety

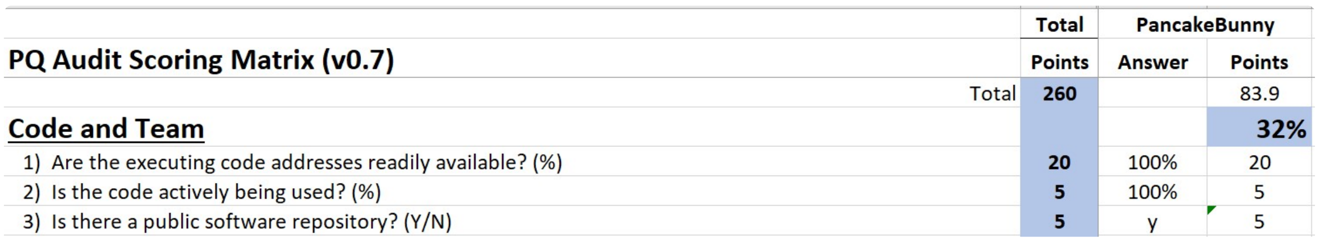
I started with Ethereum just before the DAO and that was a wonderful education. It showed the importance of code quality. The second Parity hack also showed the importance of good process. Here my aviation background offers some value. Aerospace knows how to make reliable code using quality processes.

I was coaxed to go to EthDenver 2018 and there I started SecuEth.org with Bryant and Roman. We created guidelines on good processes for blockchain code development. We got EthFoundation funding to assist in their development.

Process Quality Reviews are an extension of the SecurEth guidelines that will further increase the quality processes in Solidity and Vyper development.

DeFiSafety is my full time gig and we are working on funding vehicles for a permanent staff.

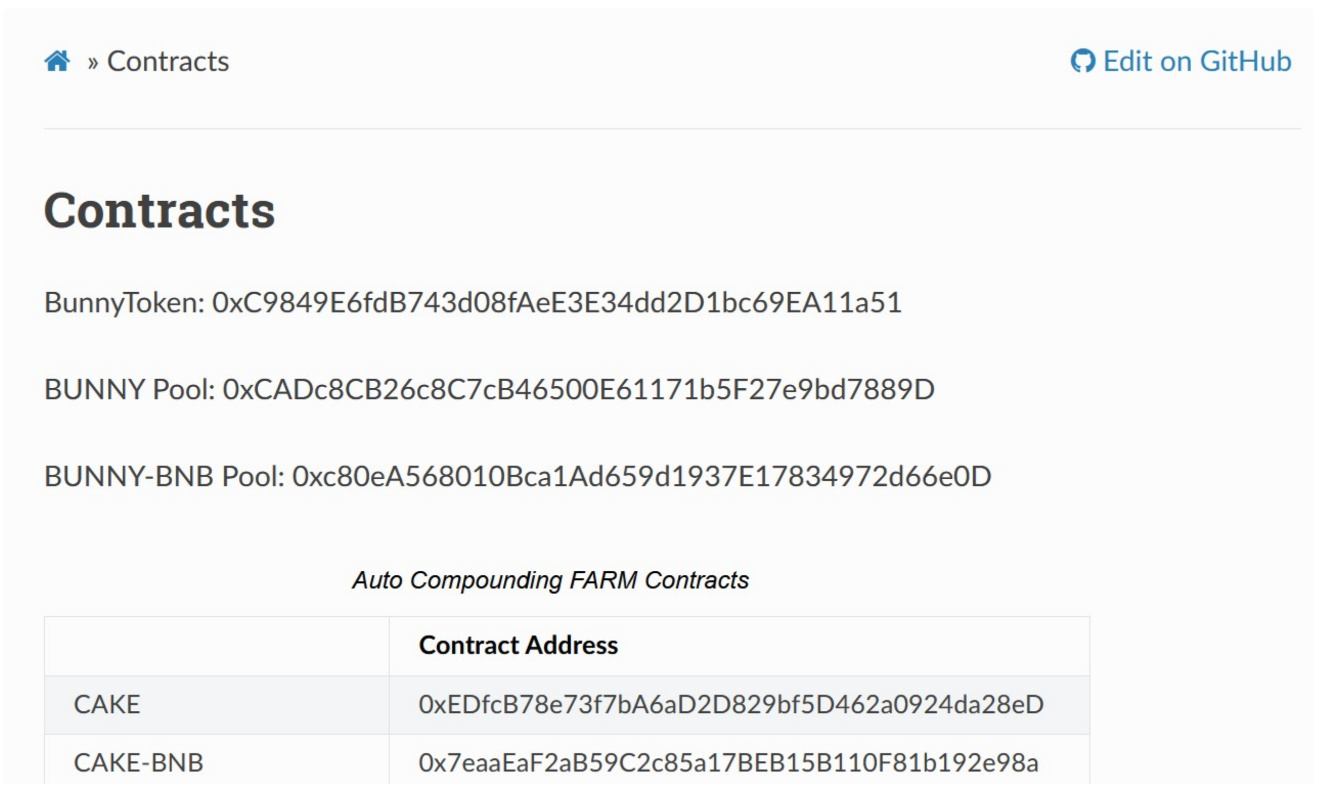
Scoring Appendix



	Total	PancakeBunny	
PQ Audit Scoring Matrix (v0.7)	**Points**	**Answer**	**Points**
Total	**260**		83.9
Code and Team			**32%**
1) Are the executing code addresses readily available? (%)	**20**	100%	20
2) Is the code actively being used? (%)	**5**	100%	5
3) Is there a public software repository? (Y/N)	**5**	y	5

4) Is there a development history visible? (%)	**5**	100%	5
5) Is the team public (not anonymous)? (Y/N)	**15**	N	0
Code Documentation			
6) Is there a whitepaper? (Y/N)	**5**	y	5
7) Are the basic software functions documented? (Y/N)	**10**	N	0
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)	**15**	0%	0
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)	**5**	0%	0
10) Is it possible to trace from software documentation to the implementation in code (%)	**10**	0%	0
Testing			
11) Full test suite (Covers all the deployed code) (%)	**20**	12%	2.4
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)	**5**	30%	1.5
13) Scripts and instructions to run the tests? (Y/N)	**5**	N	0
14) Report of the results (%)	**10**	0%	0
15) Formal Verification test done (%)	**5**	0%	0
16) Stress Testing environment (%)	**5**	0%	0
Security			
17) Did 3rd Party audits take place? (%)	**70**	40%	28
18) Is the bug bounty acceptable high? (%)	**10**	0%	0
Access Controls			
19) Can a user clearly and quickly find the status of the admin controls	**5**	0%	0
20) Is the information clear and complete	**10**	30%	3
21) Is the information in non-technical terms	**10**	90%	9
22) Is there Pause Control documentation including records of tests	**10**	0%	0
Section Scoring			
Code and Team	50	70%	
Documentation	45	11%	
Testing	50	8%	
Security	80	35%	
Access Controls	35	34%	

Executing Code Appendix



» Contracts Edit on GitHub

Contracts

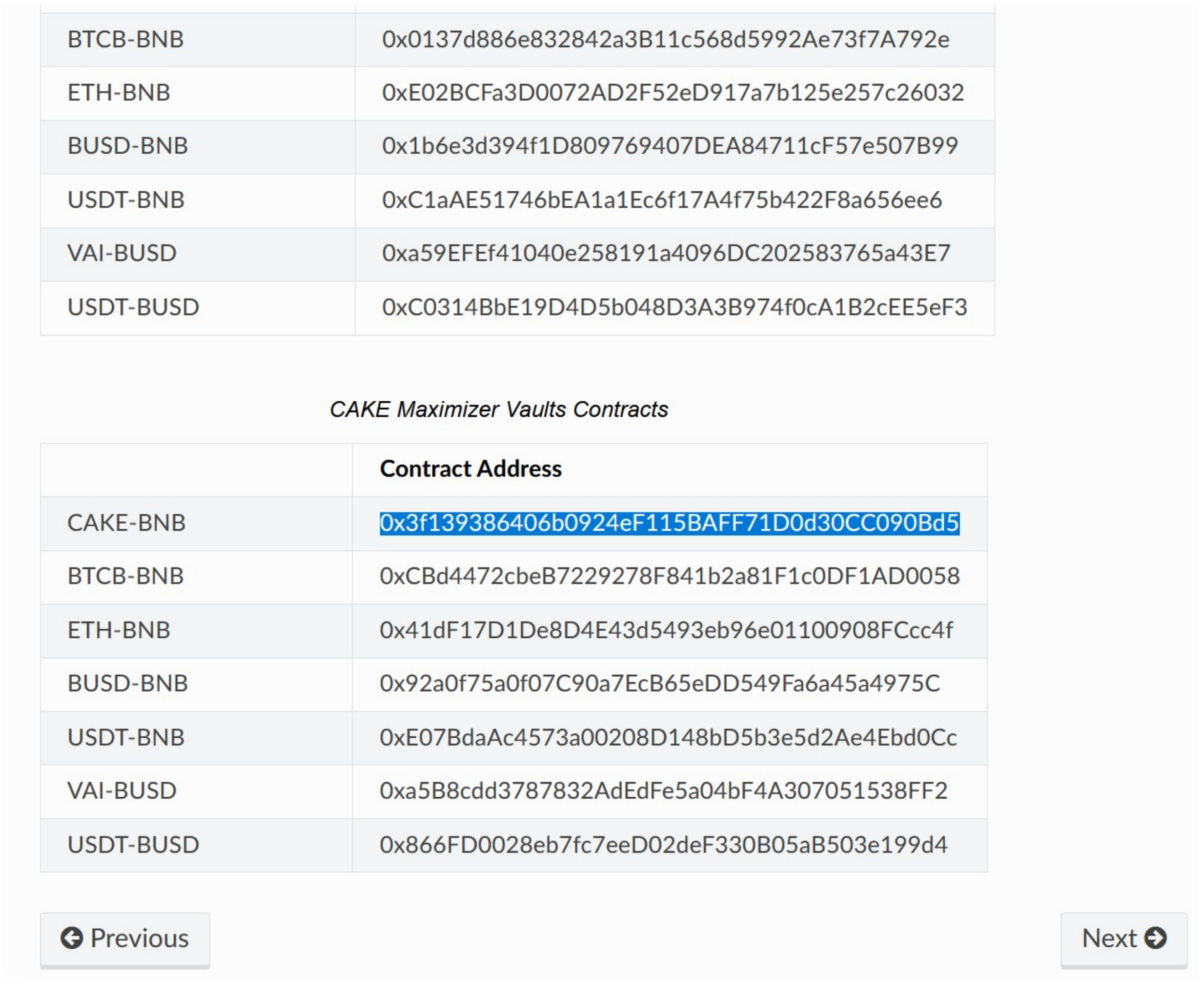
BunnyToken: 0xC9849E6fdB743d08fAeE3E34dd2D1bc69EA11a51

BUNNY Pool: 0xCADc8CB26c8C7cB46500E61171b5F27e9bd7889D

BUNNY-BNB Pool: 0xc80eA568010Bca1Ad659d1937E17834972d66e0D

Auto Compounding FARM Contracts

	Contract Address
CAKE	0xEDfcB78e73f7bA6aD2D829bf5D462a0924da28eD
CAKE-BNB	0x7eaaEaF2aB59C2c85a17BEB15B110F81b192e98a

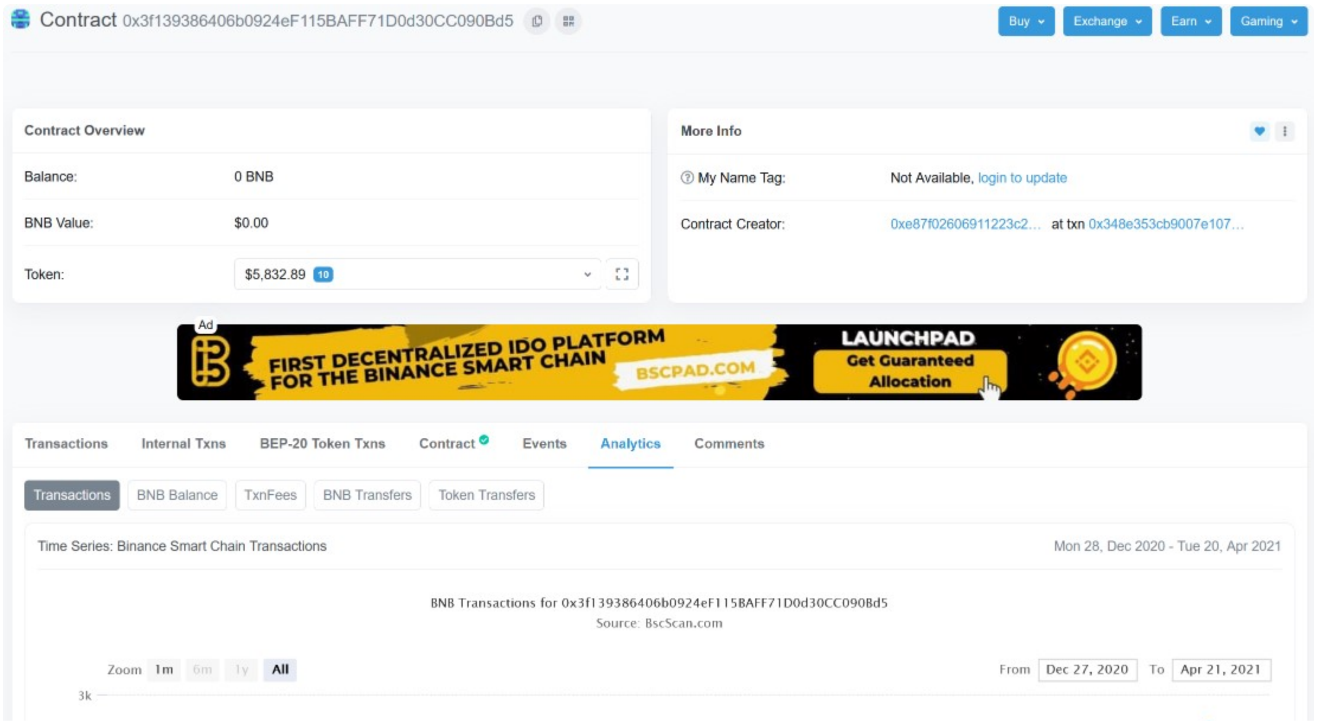


BTCB-BNB	0x0137d886e832842a3B11c568d5992Ae73f7A792e
ETH-BNB	0xE02BCFa3D0072AD2F52eD917a7b125e257c26032
BUSD-BNB	0x1b6e3d394f1D809769407DEA84711cF57e507B99
USDT-BNB	0xC1aAE51746bEA1a1Ec6f17A4f75b422F8a656ee6
VAI-BUSD	0xa59EFEf41040e258191a4096DC202583765a43E7
USDT-BUSD	0xC0314BbE19D4D5b048D3A3B974f0cA1B2cEE5eF3

CAKE Maximizer Vaults Contracts

	Contract Address
CAKE-BNB	0x3f139386406b0924eF115BAFF71D0d30CC090Bd5
BTCB-BNB	0xCBd4472cbeB7229278F841b2a81F1c0DF1AD0058
ETH-BNB	0x41dF17D1De8D4E43d5493eb96e01100908FCcc4f
BUSD-BNB	0x92a0f75a0f07C90a7EcB65eDD549Fa6a45a4975C
USDT-BNB	0xE07BdaAc4573a00208D148bD5b3e5d2Ae4Ebd0Cc
VAI-BUSD	0xa5B8cdd3787832AdEdFe5a04bF4A307051538FF2
USDT-BUSD	0x866FD0028eb7fc7eeD02deF330B05aB503e199d4

Code Used Appendix



Contract 0x3f139386406b0924eF115BAFF71D0d30CC090Bd5

Contract Overview

Balance:	0 BNB
BNB Value:	$0.00
Token:	$5,832.89 10

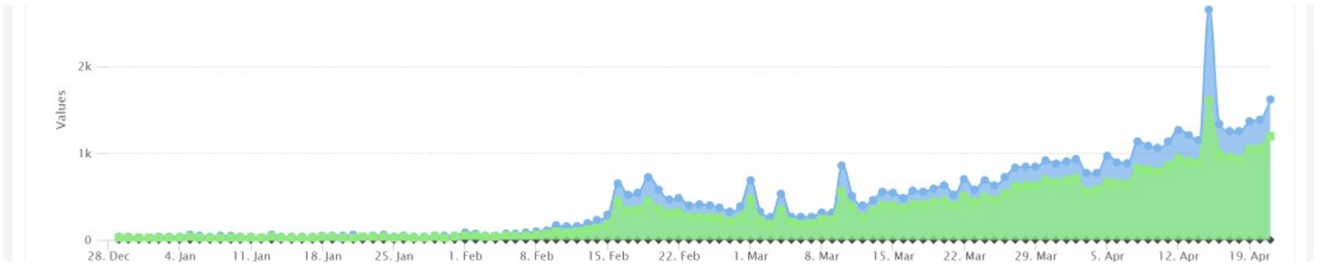
More Info

My Name Tag:	Not Available, login to update
Contract Creator:	0xe87f02606911223c2… at txn 0x348e353cb9007e107…

Ad
FIRST DECENTRALIZED IDO PLATFORM FOR THE BINANCE SMART CHAIN
BSCPAD.COM
LAUNCHPAD
Get Guaranteed Allocation

Time Series: Binance Smart Chain Transactions — Mon 28, Dec 2020 - Tue 20, Apr 2021

BNB Transactions for 0x3f139386406b0924eF115BAFF71D0d30CC090Bd5
Source: BscScan.com



Example Code Appendix

```
/ SPDX-License-Identifier: MIT
pragma solidity 0.6.12;
pragma experimental ABIEncoderV2;

/*
  ___                      _   _
 | _ )_  _ _ _  _ _ _  _  | | | |
 | _ \ || | ' \| ' \ || | |_| |_|
 |___/\_,_|_||_|_||_\_, | (_) (_)
                    |__/
*
* MIT License
* ===========
*
* Copyright (c) 2020 BunnyFinance
*
* Permission is hereby granted, free of charge, to any person obtaining a copy
* of this software and associated documentation files (the "Software"), to deal
* in the Software without restriction, including without limitation the rights
* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
* copies of the Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be included in all
* copies or substantial portions of the Software.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
*/

import '@pancakeswap/pancake-swap-lib/contracts/math/SafeMath.sol';
import '@pancakeswap/pancake-swap-lib/contracts/token/BEP20/IBEP20.sol';
import '@pancakeswap/pancake-swap-lib/contracts/token/BEP20/SafeBEP20.sol';
import "@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol";

import "../interfaces/IBunnyMinterV2.sol";
import "../interfaces/IBunnyChef.sol";

import "../interfaces/IStrategy.sol";
import "./BunnyToken.sol";


contract BunnyChef is IBunnyChef, OwnableUpgradeable {
    using SafeMath for uint;
    using SafeBEP20 for IBEP20;

    /* ========== CONSTANTS ============= */

    BunnyToken public constant BUNNY = BunnyToken(0xC9849E6fdB743d08fAeE3E34dd2D1bc69EA11a

    /* ========== STATE VARIABLES ========== */

    address[] private _vaultList;
    mapping(address => VaultInfo) vaults;
    mapping(address => mapping(address => UserInfo)) vaultUsers;

    IBunnyMinterV2 public minter;

    uint public startBlock;
    uint public override bunnyPerBlock;
    uint public override totalAllocPoint;

    /* ========== MODIFIERS ========== */

    modifier onlyVaults {
        require(vaults[msg.sender].token != address(0), "BunnyChef: caller is not on the v
        _;
    }

    modifier updateRewards(address vault) {
        VaultInfo storage vaultInfo = vaults[vault];
        if (block.number > vaultInfo.lastRewardBlock) {
            uint tokenSupply = tokenSupplyOf(vault);
            if (tokenSupply > 0) {
                uint multiplier = timeMultiplier(vaultInfo.lastRewardBlock, block.number);
                uint rewards = multiplier.mul(bunnyPerBlock).mul(vaultInfo.allocPoint).div
                vaultInfo.accBunnyPerShare = vaultInfo.accBunnyPerShare.add(rewards.mul(1e
            }
            vaultInfo.lastRewardBlock = block.number;
        }
        _;
    }

    /* ========== EVENTS ========== */

    event NotifyDeposited(address indexed user, address indexed vault, uint amount);
    event NotifyWithdrawn(address indexed user, address indexed vault, uint amount);
    event BunnyRewardPaid(address indexed user, address indexed vault, uint amount);

    /* ========== INITIALIZER ========== */

    function initialize(uint _startBlock, uint _bunnyPerBlock) external initializer {
        __Ownable_init();

        startBlock = _startBlock;
        bunnyPerBlock = _bunnyPerBlock;
    }

    /* ========== VIEWS ========== */

    function timeMultiplier(uint from, uint to) public pure returns (uint) {
        return to.sub(from);
    }

    function tokenSupplyOf(address vault) public view returns (uint) {
        return IStrategy(vault).totalSupply();
    }

    function vaultInfoOf(address vault) external view override returns (VaultInfo memory) 
        return vaults[vault];
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	36	8240	1586	1102	5552	1028

Comments to Code 1102/5552 = 19%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	10	1098	231	147	720	76

Tests to Code 720/5552 = 12%