0.7

PieDAO Process Quality Review

Score: 82%

Overview

This is a PieDAO Process Quality Review completed on October 1st 2021. It was performed using the Process Review process (version 0.7.3) and is documented here. The review was performed by Nic of DeFiSafety. Check out our Telegram.

The final score of the review is **82%**, a **PASS**. 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

views to change. All information is subject to possible correction. Information may quickly become unreliable for various reasons, including changes in market conditions or economic circumstances.

This completed report is copyright (c) DeFiSafety 2021. Permission is given to copy in whole, retaining this copyright label.

Chain

This section indicates the blockchain used by this protocol.

> ✓ **Chain:** Ethereum, Binance Smart Chain

Guidance:

Ethereum
Binance Smart Chain
Polygon
Avalanche
Terra
Celo
Arbitrum
Solana

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 following 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://docs.piedao.org/technical/deployed-smart-contracts, 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 labeling not clear or easy to find
0% Executing addresses could not be found

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

> **Answer:** 100%

Activity is over 10 internal transactions a day on contract *PV2SmartPool*, as indicated in the Appendix.

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:** 100%

GitHub: https://github.com/pie-dao/pie-smart-pools.

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

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

> **Answer:** 100%

With 181 commits and 17 branches, PieDAO's "Smart Pools" GitHub repository has a robust development history.

This metric 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 30 commits

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

> **Answer:** Yes

Location: https://medium.com/piedao/about.

For a **"Yes"** in this question, the real names of some team members must be public on the website or other documentation (LinkedIn, etc). If the team is anonymous, 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://docs.piedao.org/.

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

> **Answer:** Yes

PieDAO's basic software functions are all documented under the "Dev Docs" section of their documentation at https://docs.piedao.org/technical/architecture.

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

> **Answer:** 80%

PieDAO's software documentation covers most of the deployed contracts at https://docs.piedao.org/technical/architecture, as well as in the README.md of their corresponding GitHub repositories. The only contract that does not have its functions documented is the "Governance" contract.

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

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

> **Answer:** 55%

Code examples are in the Appendix. As per the SLOC, there is 55% 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:** 60%

There is a strong presence of non explicit traceability between the software functions' documentation and their implementation in the PieDao source code.

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 documentation to code such that it is clear where each outlined function is coded in the source code. 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:** 100%

Code examples are in the Appendix. As per the SLOC, there is 281% testing to code (TtC).

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

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

> **Answer:** 73%

PieDAO has an average code coverage percentage of 72.5 when adding up and averaging all coverage documented in their Quantstamp v1 and v2 audits.

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 improved by adding tests that achieve 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:** Yes

Scripts/Instructions location: https://github.com/pie-dao/pie-smart-pools/blob/development/README.MD

14) Report of the results (%)

> ⚠ **Answer:** 0%

There is no evidence of a PieDao test result report in any of their GitHub repositories or documentation.

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%

There is no evidence of a PieDAO Formal Verification test in their documentation.

16) Stress Testing environment (%)

> **Answer:** 100%

There is evidence of PieDAO's testnet smart contract usage at https://docs.piedao.org/technical/deployed-smart-contracts.

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:** 100%

PieDAO's has had 4 audits performed and published as seen from this GitHub repository. All of the reports were published before the specific contracts' mainnet launches.

Most fix recommendations were implemented by the PieDAO team.

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

50% Audit(s) performed after deployment and changes needed but not implemented
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, (where question 1 is 0%)

Deduct 25% if code is in a private repo and no note from auditors that audit is applicable to deployed code

18) Is the bounty value acceptably high (%)

> ⚠ **Answer:** 0%

There is no evidence of a PieDAO Bug Bounty program.

Guidance:

100% Bounty is 10% TVL or at least $1M AND active program (see below)
90% Bounty is 5% TVL or at least 500k AND active program
80% Bounty is 5% TVL or at least 500k
70% Bounty is 100k or over AND active program
60% Bounty is 100k or over
50% Bounty is 50k or over AND active program
40% Bounty is 50k or over
20% Bug bounty program bounty is less than 50k
0% No bug bounty program offered

An active program means that a third party (such as Immunefi) is actively driving hackers to the site. An inactive program would be static mentions on the docs.

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;

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

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

> ✓ **Answer:** 100%

PieDAO's governance section can clearly be found at https://docs.piedao.org/governance/dough.

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

20) Is the information clear and complete (%)

> **Answer:** 90%

a) All smart pool contracts are clearly labelled as upgradeable at https://docs.piedao.org/technical/updating-a-smart-pool-implementation

b) There is a description of the PieDAO MultiSig at https://docs.piedao.org/technical/multisigs.

c) Capabilities for change in contracts are described in software specific language; https://docs.piedao.org/technical/updating-a-smart-pool-implementation

Guidance:

All the contracts are immutable -- 100% OR

a) All contracts are clearly labelled as upgradeable (or not) -- 30% AND
b) The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 30% AND
c) 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 information in non-technical terms that pertain to the investments (%)

> **Answer:** 30%

PieDAO's provides this information in software specific language; https://docs.piedao.org/technical/updating-a-smart-pool-implementation

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:** 80%

Pause Controls is clearly explained at https://docs.piedao.org/technical/untitled#features-and-use-cases, but there is no evidence of any regular tests.

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	PieDAO Update	
PQ Audit Scoring Matrix (v0.7)	Points	Answer	Points

Total	260		214.4
Code and Team			**82%**
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	y	15
Code Documentation			
6) Is there a whitepaper? (Y/N)	5	Y	5
7) Are the basic software functions documented? (Y/N)	10	y	10
8) Does the software function documentation fully (100%) cov	15	80%	12
9) Are there sufficiently detailed comments for all functions w	5	55%	2.75
10) Is it possible to trace from software documentation to the	10	60%	6
Testing			
11) Full test suite (Covers all the deployed code) (%)	20	100%	20
12) Code coverage (Covers all the deployed lines of code, or ex	5	73%	3.65
13) Scripts and instructions to run the tests? (Y/N)	5	y	5
14) Report of the results (%)	10	0%	0
15) Formal Verification test done (%)	5	0%	0
16) Stress Testing environment (%)	5	100%	5
Security			
17) Did 3rd Party audits take place? (%)	70	100%	70
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	5	100%	5
20) Is the information clear and complete	10	90%	9
21) Is the information in non-technical terms	10	30%	3
22) Is there Pause Control documentation including records of	10	80%	8
Section Scoring			
Code and Team	50	100%	
Documentation	45	79%	
Testing	50	67%	
Security	80	88%	
Access Controls	35	71%	

Executing Code Appendix

Ethereum Mainnet | Binance SmartChain | Görli | Rinkeby

Pies

Name	Address
BTC++	0x0327112423f3a68efdf1fcf402f6c5cb9f7c33fd
DEFI+S	0xad6a626ae2b43dcb1b39430ce496d2fa0365ba9c
DEFI++	0x8d1ce361eb68e9e05573443c407d4a3bed23b033

BCP	0xe4f726adc8e89c6a6017f01eada77865db22da14
PProxiedFactory	0x28a1474ccdb0353b9c0570b54e669fc4d6ed55a6
PCappedSmartPool IMPL v1	0x6ba679CFfceEe63f1505F975BAdEf5ABfA1ae9EA
PV2SmartPool v2	0x706f00ea85a71eb5d7c2ce2ad61dbbe62b616435

Code Used Appendix

Parent Txn Hash	Block	Age	From		To	Value	
0xef1ed8893106f528ed5...	13332328	11 hrs 44 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0xef1ed8893106f528ed5...	13332328	11 hrs 44 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0xef1ed8893106f528ed5...	13332328	11 hrs 44 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0xef1ed8893106f528ed5...	13332328	11 hrs 44 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0xef1ed8893106f528ed5...	13332328	11 hrs 44 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x3bb39c3e343f85a6e7...	13332318	11 hrs 46 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x7493334e38cf97e877...	13332279	11 hrs 54 mins ago	0xad6a626ae2b43dcb1b...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x71ee1a3494dca89915...	13330737	17 hrs 44 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x71ee1a3494dca89915...	13330737	17 hrs 44 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x09993913e01c1402c2...	13330724	17 hrs 47 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x9b18f32c10b51e6c68...	13330231	19 hrs 32 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x563b1b93637807e222...	13329874	20 hrs 59 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	
0x563b1b93637807e222...	13329874	20 hrs 59 mins ago	0xe4f726adc8e89c6a60...	→	0x706f00ea85a71eb5d7...	0 Ether	

This website uses cookies to improve your experience and has an updated Privacy Policy. Got It

Example Code Appendix

```
contract PV2SmartPool is IPV2SmartPool, PCToken, ReentryProtection {
  using LibSafeApprove for IERC20;

  event TokensApproved();
  event ControllerChanged(address indexed previousController, address indexed newControlle
  event PublicSwapSetterChanged(address indexed previousSetter, address indexed newSetter)
  event TokenBinderChanged(address indexed previousTokenBinder, address indexed newTokenBi
  event PublicSwapSet(address indexed setter, bool indexed value);
  event SwapFeeSet(address indexed setter, uint256 newFee);
  event CapChanged(address indexed setter, uint256 oldCap, uint256 newCap);
  event CircuitBreakerTripped();
  event JoinExitEnabledChanged(address indexed setter, bool oldValue, bool newValue);
  event CircuitBreakerChanged(
    address indexed _oldCircuitBreaker,
    address indexed _newCircuitBreaker
  );

  modifier ready() {
```

```
    require(address(PBStorage.load().bPool) != address(0), "PV2SmartPool.ready: not ready");
    _;
  }

  modifier onlyController() {
    require(
      msg.sender == PBStorage.load().controller,
      "PV2SmartPool.onlyController: not controller"
    );
    _;
  }

  modifier onlyPublicSwapSetter() {
    require(
      msg.sender == PBStorage.load().publicSwapSetter,
      "PV2SmartPool.onlyPublicSwapSetter: not public swap setter"
    );
    _;
  }

  modifier onlyTokenBinder() {
    require(
      msg.sender == PBStorage.load().tokenBinder,
      "PV2SmartPool.onlyTokenBinder: not token binder"
    );
    _;
  }

  modifier onlyPublicSwap() {
    require(
      PBStorage.load().bPool.isPublicSwap(),
      "PV2SmartPool.onlyPublicSwap: swapping not enabled"
    );
    _;
  }

  modifier onlyCircuitBreaker() {
    require(
      msg.sender == P2Storage.load().circuitBreaker,
      "PV2SmartPool.onlyCircuitBreaker: not circuit breaker"
    );
    _;
  }

  modifier onlyJoinExitEnabled() {
    require(
      P2Storage.load().joinExitEnabled,
      "PV2SmartPool.onlyJoinExitEnabled: join and exit not enabled"
    );
    _;
  }

  modifier withinCap() {
```

```
    _;
    require(totalSupply() < PCSStorage.load().cap, "PV2SmartPool.withinCap: Cap limit reacl
  }

  /**
        @notice Initialises the contract
        @param _bPool Address of the underlying balancer pool
        @param _name Name for the smart pool token
        @param _symbol Symbol for the smart pool token
        @param _initialSupply Initial token supply to mint
    */
  function init(
    address _bPool,
    string calldata _name,
    string calldata _symbol,
    uint256 _initialSupply
  ) external override {
    PBStorage.StorageStruct storage s = PBStorage.load();
    require(address(s.bPool) == address(0), "PV2SmartPool.init: already initialised");
    require(_bPool != address(0), "PV2SmartPool.init: _bPool cannot be 0x00....000");
    require(_initialSupply != 0, "PV2SmartPool.init: _initialSupply can not zero");
    s.bPool = IBPool(_bPool);
    s.controller = msg.sender;
    s.publicSwapSetter = msg.sender;
    s.tokenBinder = msg.sender;
    PCStorage.load().name = _name;
    PCStorage.load().symbol = _symbol;

    LibPoolToken._mint(msg.sender, _initialSupply);
  }

  /**
    @notice Sets approval to all tokens to the underlying balancer pool
    @dev It uses this function to save on gas in joinPool
  */
  function approveTokens() public override noReentry {
    IBPool bPool = PBStorage.load().bPool;
    address[] memory tokens = bPool.getCurrentTokens();
    for (uint256 i = 0; i < tokens.length; i++) {
      IERC20(tokens[i]).safeApprove(address(bPool), uint256(-1));
    }
    emit TokensApproved();
  }

  // POOL EXIT -----------------------------------------------

  /**
        @notice Burns pool shares and sends back the underlying assets leaving some in the
        @param _amount Amount of pool tokens to burn
        @param _lossTokens Tokens skipped on redemption
    */
  function exitPoolTakingloss(uint256 _amount, address[] calldata _lossTokens)
    external
```

```
    override
    ready
    noReentry
    onlyJoinExitEnabled
  {
    LibPoolEntryExit.exitPoolTakingloss(_amount, _lossTokens);
  }

  /**
        @notice Burns pool shares and sends back the underlying assets
        @param _amount Amount of pool tokens to burn
    */
  function exitPool(uint256 _amount) external override ready noReentry onlyJoinExitEnabled
    LibPoolEntryExit.exitPool(_amount);
  }

  /**
    @notice Burn pool tokens and redeem underlying assets. With front running protection
    @param _amount Amount of pool tokens to burn
    @param _minAmountsOut Minimum amounts of underlying assets
  */
  function exitPool(uint256 _amount, uint256[] calldata _minAmountsOut)
    external
    override
    ready
    noReentry
    onlyJoinExitEnabled
  {
    LibPoolEntryExit.exitPool(_amount, _minAmountsOut);
  }

  /**
        @notice Exitswap single asset pool exit given pool amount in
        @param _token Address of exit token
        @param _poolAmountIn Amount of pool tokens sending to the pool
        @return tokenAmountOut amount of exit tokens being withdrawn
    */
  function exitswapPoolAmountIn(
    address _token,
    uint256 _poolAmountIn,
    uint256 _minAmountOut
  )
    external
    override
    ready
    noReentry
    onlyPublicSwap
    onlyJoinExitEnabled
    returns (uint256 tokenAmountOut)
  {
    return LibPoolEntryExit.exitswapPoolAmountIn(_token, _poolAmountIn, _minAmountOut);
  }

```

```
178   /**
179         @notice Exitswap single asset pool entry given token amount out
180         @param _token Address of exit token
181         @param _tokenAmountOut Amount of exit tokens
182         @return poolAmountIn amount of pool tokens being deposited
183     */
184   function exitswapExternAmountOut(
185     address _token,
186     uint256 _tokenAmountOut,
187     uint256 _maxPoolAmountIn
188   )
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	11	1848	238	573	1037	65

Comments to Code 573/1037 = 55%

Javascript/TypeScript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	6	114	29	6	79	1
TypeScript	12	3449	518	95	2836	56

Tests to Code 2915/1037 = 281%