0.7

88mph V3.0 (0.7) Process Quality Review

Score: 74%

Overview

This is a Process Quality Review of 88mph completed on 15/09/2021. A 88mph V1 review was published in the past and can be found here. It was performed using the Process Review process (version 0.7.3) and is documented here. The review was performed by Nick of DeFiSafety. Check out our Telegram.

The final score of the review is **74%**, 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.

Chain

This section indicates the blockchain used by this protocol.

> **Chain:** Ethereum

Guidance:

Ethereum
Binance Smart Chain
Polygon
Avalanche
Terra

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:** 70%

They are available at website https://88mph.app/docs/addresses, which feature the v2 addresses. There is also https://docs.88mph.app/developer-docs/addresses, featuring the V3 addresses as indicated in the Appendix. The docs are not centralized in the docs linked from the app's homepage (https://docs.88mph.app/) and to find them requires additional research. The developer docs are listed in the homepage, with traceability to the contract addresses, but presentation in JSON format requires additional work to view the contract addressses themselves.

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

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 towards the final score.

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

> **Answer:** 100%

Activity is 18 transactions a day on contract 0x8888801aF4d980682e47f1A9036e589479e835C5, 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:** Yes

GitHub: https://github.com/88mphapp

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%

88mph has a strong 367 commits and 23 branches, making its development history well documented.

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

The team consists of Guillame Palayer and Zefram Lou.

Location: https://github.com/trailofbits/publications/blob/master/reviews/88mph.pdf

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.88mph.app/

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

> **Answer:** Yes

Core software functions such as DInterest, Withdrawal & Deposit are clearly documented.
https://docs.88mph.app/developer-docs/smart-contract-architecture

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

> **Answer:** 80%

Smart contracts, governance, and other deployed functions are clearly covered in the documentation. Not all functions are documented.

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:** 23%

Code examples are in the Appendix. As per the SLOC, there is 23% 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%

88mph has clear code explanations for all major functions in the documents, though traceability is nonexplicit.

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:** 80%

Code examples are in the Appendix. As per the SLOC, there is 117% 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

How to improve this score:

This score can improved 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:** 50%

Neither auditors nor the protocol itself successfully generated a code coverage score. Trail of bits did a "coverage" part of their audit, but did not include percentage values of code coverage, and this can therefore not be counted into the scoring.

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

Scrips/Instructions location: https://github.com/88mphapp/88mph-contracts

14) Report of the results (%)

> ⚠ **Answer:** 0

There is no report of the test's results.

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 formal verification of this protocol.

16) Stress Testing environment (%)

> ✓ **Answer:** 100%

88mph has been deployed on the Rinkeby test network in full.

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%

Three audits have been released for V3: Trail of Bits, Code423n4, and PeckShield. Two other audits have been made for previous versions, and all audits were completed before the code was implemented.

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:** 70%

88mph uses Immunefi's bug bounty program at the same time as offering a $100,420 maximum bounty.

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%

Governance has controls to the access controls, which is clearly labelled under the governance section.

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: 15%

There is no information on access controls beyond a mention in a medium article, which has since been changed.

There is a mention of change capabilities under governance in the documents.

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: 0%

There is no mention of admin controls.

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:** 40%

There is no mention of a pause control function in the documents. There is one documented use in a medium article, in which the pause control was used to prevent user fund theft on the 7th of June, 2021.

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.

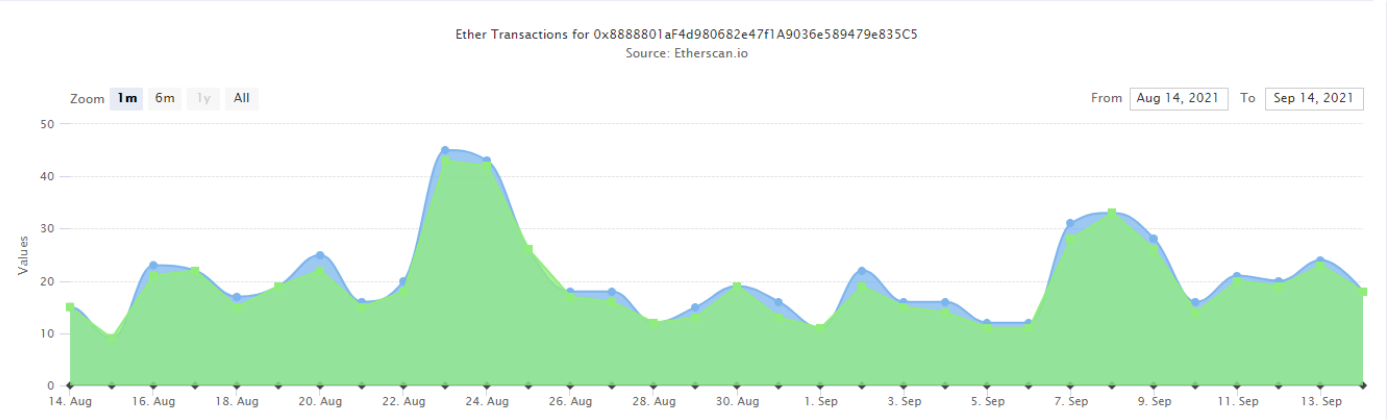
Scoring Appendix

	Total	88mph	
PQ Audit Scoring Matrix (v0.7)	**Points**	**Answer**	**Points**
Total	**260**		194.15
Code and Team			**75%**
1) Are the executing code addresses readily available? (%)	**20**	70%	14
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**	23%	1.15
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**	80%	16
12) Code coverage (Covers all the deployed lines of code, or ex	**5**	50%	2.5
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**	70%	7
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**	15%	1.5
21) Is the information in non-technical terms	**10**	0%	0
22) Is there Pause Control documentation including records of	**10**	40%	4
Section Scoring			
Code and Team	50	88%	
Documentation	45	76%	
Testing	50	57%	
Security	80	96%	
Access Controls	35	30%	

Executing Code Appendix

solcInputs	Fix MPHMinterLegacy	7 hours ago
.chainId	Add mainnet DAI via Compound deployment	28 days ago
88mph BAL via Aave Deposit.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave Deposit_Implementation.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave Deposit_Proxy.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave Yield Token.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave Yield Token_Implementation.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave Yield Token_Proxy.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave--AaveMarket.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave--AaveMarket_Implementation.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave--AaveMarket_Proxy.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave--EMAOracle.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave_Implementation.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAL via Aave_Proxy.json	Deploy 13 Aave pools to mainnet	5 days ago
88mph BAT via Compound Deposit.json	Deploy 7 compound pools to mainnet	2 days ago
88mph BAT via Compound Deposit_Implementation.json	Deploy 7 compound pools to mainnet	2 days ago
88mph BAT via Compound Deposit_Proxy.json	Deploy 7 compound pools to mainnet	2 days ago
88mph BAT via Compound Yield Token.json	Deploy 7 compound pools to mainnet	2 days ago

Code Used Appendix



Example Code Appendix

```
pragma solidity 0.8.4;

import {ERC20} from "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import {SafeERC20} from "./libs/SafeERC20.sol";
import {
    ReentrancyGuardUpgradeable
} from "@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol";
import {
    AddressUpgradeable
} from "@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol";
import {BoringOwnable} from "./libs/BoringOwnable.sol";

```

```
import {
    MulticallUpgradeable
} from "@openzeppelin/contracts-upgradeable/utils/MulticallUpgradeable.sol";
import {MoneyMarket} from "./moneymarkets/MoneyMarket.sol";
import {IFeeModel} from "./models/fee/IFeeModel.sol";
import {IInterestModel} from "./models/interest/IInterestModel.sol";
import {NFT} from "./tokens/NFT.sol";
import {FundingMultitoken} from "./tokens/FundingMultitoken.sol";
import {MPHMinter} from "./rewards/MPHMinter.sol";
import {IInterestOracle} from "./models/interest-oracle/IInterestOracle.sol";
import {PRBMathUD60x18} from "prb-math/contracts/PRBMathUD60x18.sol";
import {Rescuable} from "./libs/Rescuable.sol";
import {console} from "hardhat/console.sol";

/**
    @title DeLorean Interest -- It's coming back from the future!
    @author Zefram Lou
    @notice The main pool contract for fixed-rate deposits
    @dev The contract to interact with for most actions
 */
contract DInterest is
    ReentrancyGuardUpgradeable,
    BoringOwnable,
    Rescuable,
    MulticallUpgradeable
{
    using SafeERC20 for ERC20;
    using AddressUpgradeable for address;
    using PRBMathUD60x18 for uint256;

    // Constants
    uint256 internal constant PRECISION = 10**18;
    /**
        @dev used for sumOfRecordedFundedPrincipalAmountDivRecordedIncomeIndex
     */
    uint256 internal constant EXTRA_PRECISION = 10**27;
    /**
        @dev used for funding.principalPerToken
     */
    uint256 internal constant ULTRA_PRECISION = 2**128;
    /**
        @dev Specifies the threshold for paying out funder interests
     */
    uint256 internal constant FUNDER_PAYOUT_THRESHOLD_DIVISOR = 10**10;

    // User deposit data
    // Each deposit has an ID used in the depositNFT, which is equal to its index in `depos
    struct Deposit {
        uint256 virtualTokenTotalSupply; // depositAmount + interestAmount, behaves like a
        uint256 interestRate; // interestAmount = interestRate * depositAmount
        uint256 feeRate; // feeAmount = feeRate * depositAmount
        uint256 averageRecordedIncomeIndex; // Average income index at time of deposit, use
        uint64 maturationTimestamp; // Unix timestamp after which the deposit may be withd
```

```
65          uint64 fundingID; // The ID of the associated Funding struct. 0 if not funded.
66      }
67      Deposit[] internal deposits;
68
69      // Funding data
70      // Each funding has an ID used in the fundingMultitoken, which is equal to its index i
71      struct Funding {
72          uint64 depositID; // The ID of the associated Deposit struct.
73          uint64 lastInterestPayoutTimestamp; // Unix timestamp of the most recent interest
74          uint256 recordedMoneyMarketIncomeIndex; // the income index at the last update (cr
75          uint256 principalPerToken; // The amount of stablecoins that's earning interest fo
76      }
77      Funding[] internal fundingList;
78      // the sum of (recordedFundedPrincipalAmount / recordedMoneyMarketIncomeIndex) of all
79      uint256 public sumOfRecordedFundedPrincipalAmountDivRecordedIncomeIndex;
80
81      // Params
82      /**
83          @dev Maximum deposit period, in seconds
84       */
85      uint64 public MaxDepositPeriod;
86      /**
87          @dev Minimum deposit amount, in stablecoins
88       */
89      uint256 public MinDepositAmount;
90
91      // Global variables
92      uint256 public totalDeposit;
93      uint256 public totalInterestOwed;
94      uint256 public totalFeeOwed;
95      uint256 public totalFundedPrincipalAmount;
96
97      // External smart contracts
98      IFeeModel public feeModel;
99      IInterestModel public interestModel;
100     IInterestOracle public interestOracle;
101     NFT public depositNFT;
102     FundingMultitoken public fundingMultitoken;
103     MPHMinter public mphMinter;
104
105     // Extra params
106     /**
107         @dev The maximum amount of deposit in the pool. Set to 0 to disable the cap.
108      */
109     uint256 public GlobalDepositCap;
110
111     // Events
112     event EDeposit(
113         address indexed sender,
114         uint256 indexed depositID,
115         uint256 depositAmount,
116         uint256 interestAmount,
117         uint256 feeAmount,
```

```
        uint64 maturationTimestamp
    );
    event ETopupDeposit(
        address indexed sender,
        uint64 indexed depositID,
        uint256 depositAmount,
        uint256 interestAmount,
        uint256 feeAmount
    );
    event ERolloverDeposit(
        address indexed sender,
        uint64 indexed depositID,
        uint64 indexed newDepositID
    );
    event EWithdraw(
        address indexed sender,
        uint256 indexed depositID,
        bool indexed early,
        uint256 virtualTokenAmount,
        uint256 feeAmount
    );
    event EFund(
        address indexed sender,
        uint64 indexed fundingID,
        uint256 fundAmount,
        uint256 tokenAmount
    );
    event EPayFundingInterest(
        uint256 indexed fundingID,
        uint256 interestAmount,
        uint256 refundAmount
    );
    event ESetParamAddress(
        address indexed sender,
        string indexed paramName,
        address newValue
    );
    event ESetParamUint(
        address indexed sender,
        string indexed paramName,
        uint256 newValue
    );

    function __DInterest_init(
        uint64 _MaxDepositPeriod,
        uint256 _MinDepositAmount,
        address _feeModel,
        address _interestModel,
        address _interestOracle,
        address _depositNFT,
        address _fundingMultitoken,
        address _mphMinter
    ) internal initializer {
```

```
171         __ReentrancyGuard_init();
172         __Ownable_init();
173 
174         feeModel = IFeeModel(_feeModel);
175         interestModel = IInterestModel(_interestModel);
176         interestOracle = IInterestOracle(_interestOracle);
177         depositNFT = NFT(_depositNFT);
178         fundingMultitoken = FundingMultitoken(_fundingMultitoken);
179         mphMinter = MPHMinter(_mphMinter);
180         MaxDepositPeriod = _MaxDepositPeriod;
181         MinDepositAmount = _MinDepositAmount;
182     }
183 
184     /**
185         @param _MaxDepositPeriod The maximum deposit period, in seconds
186         @param _MinDepositAmount The minimum deposit amount, in stablecoins
187         @param _feeModel Address of the FeeModel contract that determines how fees are cha
188         @param _interestModel Address of the InterestModel contract that determines how muc
189         @param _interestOracle Address of the InterestOracle contract that provides the ave
190         @param _depositNFT Address of the NFT representing ownership of deposits (owner mus
191         @param _fundingMultitoken Address of the ERC1155 multitoken representing ownership
192         @param _mphMinter Address of the contract for handling minting MPH to users
193      */
194     function initialize(
195         uint64 _MaxDepositPeriod,
196         uint256 _MinDepositAmount,
197         address _feeModel,
198         address _interestModel,
199         address _interestOracle,
200         address _depositNFT,
201         address _fundingMultitoken,
202         address _mphMinter
203     ) external virtual initializer {
204         __DInterest_init(
205             _MaxDepositPeriod,
206             _MinDepositAmount,
207             _feeModel,
208             _interestModel,
209             _interestOracle,
210             _depositNFT,
211             _fundingMultitoken,
212             _mphMinter
213         );
214     }
215 
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	24	4835	500	803	3532	212

Comments to Code 803/3532 = 23%

Javascript / Typescript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	6	4854	471	345	4038	26
TypeScript	1	112	12	0	100	0
Total	7	4966	483	345	4138	26

Tests to Code 4138/3532 = 117%