Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Frontline DAO Protocol
Platform: Polygon Network
Language: Solidity
Date: August 15th, 2022

Table of contents

Introduction 4
Project Background 4
Audit Scope 4
Claimed Smart Contract Features 7
Audit Summary 11
Technical Quick Stats 12
Code Quality 13
Documentation 13
Use of Dependencies 13
AS-IS overview 14
Severity Definitions 25
Audit Findings 26
Conclusion 30
Our Methodology 31
Disclaimers 33
Appendix

- Code Flow Diagram 34
- Slither Results Log 58
- Solidity static analysis 67
- Solhint Linter 92

Email: audit@EtherAuthority.io

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by Frontline DAO to perform the Security audit of the Frontline DAO Protocol smart contracts code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on August 15th, 2022.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

Frontline DAO Protocol is a Defi Program which has functions like mint, swap, OpenTrade, burn, twap, spot, update, mock, info, transfer, set pool, claimable, zap, addLiquidity, cleanDust, etc. The Frontline DAO contract inherits the ERC20, SafeERC20, Ownable, ReentrancyGuard, Address, IUniswapV2Router02, IERC20, IERC721, Math, SafeMath, IUniswapV2Pair, Context, Initializable, TransparentUpgradeableProxy, ERC20Burnable standard smart contracts from the OpenZeppelin library. These OpenZeppelin contracts are considered community-audited and time-tested, and hence are not part of the audit scope.

Audit scope

Name	**Code Review and Security Analysis Report for Frontline DAO Protocol Smart Contracts**
Platform	**Polygon / Solidity**
File 1	Pool.sol
File 1 MD5 Hash	856107FFBD51DDC1399BEC38192D0CEB
File 2	SwapStrategyPOL.sol
File 2 MD5 Hash	9CAEB2CD15D2E1C2CECA52DDC4C5FE47
File 3	Timelock.sol

File 3 MD5 Hash	94F559046B7CB4335EE0F49341A23DA0
File 4	DaoChef.sol
File 4 MD5 Hash	E12C4E0BDCB405DD0DB61CCF7173ED06
File 5	DaoStaking.sol
File 5 MD5 Hash	9358FC7E65F92434207DD9F06F01F960
File 6	DaoZapMMSwap.sol
File 6 MD5 Hash	46D304749327ADF042F8962D64CF8EFC
File 7	NFTController.sol
File 7 MD5 Hash	7B517FFAE5E28C8D3B7020747FFA8659
File 8	NFTControllerProxy.sol
File 8 MD5 Hash	E2E8A433C71CE32907140DCF3F28480D
File 9	Fund.sol
File 9 MD5 Hash	47370A0301A3BBA40747C7FFD8A18E6B
File 10	DaoFund.sol
File 10 MD5 Hash	0DCB7E30D2EE3CBDE722E8D63D87ACF1
File 11	DevFund.sol
File 11 MD5 Hash	A0AC2988CBE65230B36071AA974D05AB
File 12	Reserve.sol
File 12 MD5 Hash	164785B5E9F9181CE4C30A503FC3B0D4
File 13	TreasuryFund.sol
File 13 MD5 Hash	EF21D2BFAFF8EB4877E0B95A99F4B8ED
File 14	MockERC20.sol
File 14 MD5 Hash	94278D4A01D92E76EBDE914556B3A6A0
File 15	MockTreasury.sol
File 15 MD5 Hash	EAB3F68107BE7B69CAFA290A0FD6FE83
File 16	MasterOracle.sol
File 16 MD5 Hash	26FFB8A6EB84AABF384A830DB4572C0A

File 17	UniswapPairOracle.sol
File 17 MD5 Hash	37801A23DE6F4571ADD278A4A062C1D5
File 18	XToken.sol
File 18 MD5 Hash	83382FC411F2E4462B30C55D6F62A2DD
File 19	YToken.sol
File 19 MD5 Hash	FFA9BDAB9AEE9D07DB46CB3A23A34696
File 20	LOVE.sol
File 20 MD5 Hash	8989CC5C1C75E964DB25F3065208D040
File 21	MMFX.sol
File 21 MD5 Hash	664C0017F4BF8498B957DB667ED68580
File 22	MMOX.sol
File 22 MD5 Hash	7B2F2773EA26033423E33F9A1D18FA02
File 23	DaoTreasury.sol
File 23 MD5 Hash	B4AEF5736647E39B4AEC058D7AF7A989
File 24	MMUSD.sol
File 24 MD5 Hash	9CE7E628B8A5C3D365CF307F3C64AD15
File 25	StratRecollateralize.sol
File 25 MD5 Hash	8825619BF2C90BD2337916B8B6CB90B2
File 26	StratReduceReserveLP.sol
File 26 MD5 Hash	EE1CAC2A02D76BCD4158A836C13CE2DB
Audit Date	August 15th,2022

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 Pool.sol • Refresh Cooldown: 1 hour • Ratio StepUp: 0.2% • Ratio StepDown: 0.2% • Price Target: 1 MMF • Price Band: 0.004 • Minimum Collateral Ratio: 9,50,000 • YToken Slippage: 20% • Redemption Fee: 0.5% • Redemption Fee Maximum: 0.9% • Minting Fee: 0.3% • Minting Fee Maximum: 0.5%	**YES, This is valid.**
File 2 SwapStrategyPOL.sol • Swap Slippage: 20%	**YES, This is valid.**
File 3 Timelock.sol • Grace Period: 14 Days • Minimum Delay: 12 Hours • Maximum Delay: 30 Days	**YES, This is valid.**
File 4 DaoChef.sol • Maximum reward: 10 Tokens Per Second • NFT Boost Rate: 100	**YES, This is valid.**
File 5 DaoStaking.sol • Rewards Duration: 1 week • Lock Duration: 4 weeks • Team Reward Percent: 20%	**YES, This is valid.**
File 6 DaoZapMMSwap.sol DaoZap is a ZapperFi's simplified version of	**YES, This is valid.**

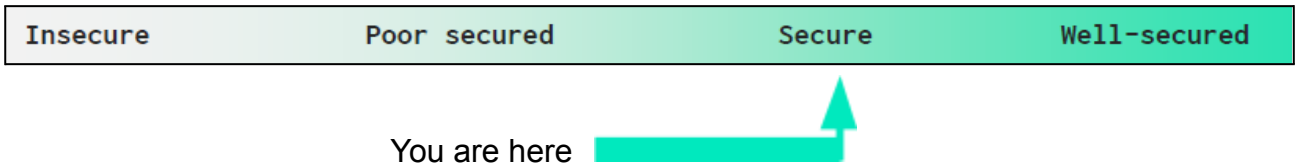
zapper contract which will: 1. use ETH to swap to target token 2. make LP between ETH and target token 3. add into DaoChef farm	
File 7 Fund.sol • Fund has functions like: allocation, transfer, etc.	**YES, This is valid.**
File 8 DaoFund.sol • Allocation: 10% • Vesting Duration: 3 Years	**YES, This is valid.**
File 9 DevFund.sol • Allocation: 10% • Vesting Duration: 2 Years	**YES, This is valid.**
File 10 Reserve.sol • Reserve has functions like: initialize, setRewarder, etc.	**YES, This is valid.**
File 11 TreasuryFund.sol • Allocation: 10% • Vesting Duration: 3 Years	**YES, This is valid.**
File 12 MockERC20.sol • MockERC20 has functions like: mint, etc.	**YES, This is valid.**
File 13 MockTreasury.sol • MockTreasury has functions like: mock, etc.	**YES, This is valid.**
File 14 MasterOracle.sol • MasterOracle has functions like: getYTokenPrice, getYTokenTWAP, etc.	**YES, This is valid.**
File 15 UniswapPairOracle.sol • Period: 60-minute Twap (Time-weighted Average Price)	**YES, This is valid.**

• Maximum Period: 48 Hours • Minimum Period: 10 Minutes • Leniency: 12 Hours	
File 16 XToken.sol • XToken has functions like: mint, setMinter, etc.	**YES, This is valid.**
File 17 YToken.sol • YToken has functions like: burn, etc.	**YES, This is valid.**
File 18 LOVE.sol • Total Supply: 30 Million	**YES, This is valid.**
File 19 MMFX.sol • Genesis Supply: 100 will be minted at genesis for liq pool seeding	**YES, This is valid.**
File 20 MMOX.sol • Genesis Supply: 100 will be minted at genesis for liq pool seeding	**YES, This is valid.**
File 21 MMUSD.sol • Genesis Supply: 100 will be minted at genesis for liq pool seeding	**YES, This is valid.**
File 22 DaoTreasury.sol • DaoTreasury has functions like: balanceOf, requestFund, etc.	**YES, This is valid.**
File 23 StratRecollateralize.sol • StratRecollateralize has functions like: recollateralize.	**YES, This is valid.**
File 24 StratReduceReserveLP.sol • StratReduceReserveLPhas functions like: reduceReserve, swap.	**YES, This is valid.**

File 25 NFTController.sol • NFTController has functions like: setBoostRate, setWhitelist, etc.	**YES, This is valid.**
File 26 NFTControllerProxy.sol • NFTControllerProxy has access to the TransparentUpgradeableProxy contract.	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. Also, these contracts do contain owner control, which does not make them fully decentralized.



We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 2 low and some very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderated
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Moderated
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Email: audit@EtherAuthority.io

Code Quality

This audit scope has 26 smart contract files. Smart contracts contain Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in the Frontline DAO Protocol are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the Frontline DAO Protocol.

The Frontline DAO team has not provided unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Some code parts are not well commented on smart contracts. We suggest using Ethereum's NatSpec style for the commenting.

Documentation

We were given a Frontline DAO Protocol smart contract code in the form of a file. The hash of that code is mentioned above in the table.

As mentioned above, code parts are not well commented. But the logic is straightforward. So it is easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Use of Dependencies

As per our observation, the libraries are used in this smart contracts infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Pool.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	nonReentrant	modifier	Passed	No Issue
8	info	external	Passed	No Issue
9	usableCollateralBalance	read	Passed	No Issue
10	calcMint	read	Passed	No Issue
11	calcRedeem	read	Passed	No Issue
12	calcExcessCollateralBalance	read	Passed	No Issue
13	refreshCollateralRatio	write	Passed	No Issue
14	receive	external	Passed	No Issue
15	mint	external	Passed	No Issue
16	redeem	external	Passed	No Issue
17	collect	external	Passed	No Issue
18	recollateralize	external	Passed	No Issue
19	checkPriceFluctuation	internal	Passed	No Issue
20	toggle	write	access only Owner	No Issue
21	setCollateralRatioOptions	write	access only Owner	No Issue
22	toggleCollateralRatio	write	access only Owner	No Issue
23	setFees	write	access only Owner	No Issue
24	setMinCollateralRatio	external	access only Owner	No Issue
25	reduceExcessCollateral	external	access only Owner	No Issue
26	setSwapStrategy	external	access only Owner	No Issue
27	setOracle	external	access only Owner	No Issue
28	setYTokenSlippage	external	access only Owner	No Issue
29	setTreasury	external	Passed	No Issue
30	transferToTreasury	internal	Passed	No Issue

SwapStrategyPOL.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue

4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	lpBalance	read	Passed	No Issue
8	execute	external	Passed	No Issue
9	calculateSwapInAmount	internal	Passed	No Issue
10	swap	internal	Passed	No Issue
11	addLiquidity	internal	Passed	No Issue
12	cleanDust	external	access only Owner	No Issue
13	changeSlippage	external	access only Owner	No Issue

Timelock.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	setDelay	write	Passed	No Issue
3	acceptAdmin	write	Passed	No Issue
4	setPendingAdmin	write	Passed	No Issue
5	queueTransaction	write	Passed	No Issue
6	cancelTransaction	write	Passed	No Issue
7	executeTransaction	write	Passed	No Issue
8	getBlockTimestamp	internal	Passed	No Issue

DaoChef.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	poolLength	read	Passed	No Issue
8	pendingReward	external	Passed	No Issue
9	updatePool	write	Passed	No Issue
10	massUpdatePools	write	Passed	No Issue
11	deposit	write	Passed	No Issue
12	withdraw	write	Passed	No Issue
13	harvest	write	Passed	No Issue
14	withdrawAndHarvest	write	Passed	No Issue
15	emergencyWithdraw	write	Passed	No Issue
16	harvestAllRewards	external	Passed	No Issue
17	checkPoolDuplicate	internal	Passed	No Issue
18	add	write	access only Owner	No Issue

19	set	write	access only Owner	No Issue
20	setRewardPerSecond	write	access only Owner	No Issue
21	setRewardMinter	external	Passed	No Issue
22	getBoost	read	Passed	No Issue
23	getSlots	read	Passed	No Issue
24	getTokenIds	read	Passed	No Issue
25	depositNFT	write	Passed	No Issue
26	withdrawNFT	write	Passed	No Issue
27	setNftController	write	access only Owner	No Issue
28	setNftBoostRate	write	access only Owner	No Issue

DaoStaking.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	addReward	write	Function input parameters lack of check	Refer Audit Findings
8	approveRewardDistributor	external	access only Owner	No Issue
9	_rewardPerToken	internal	Passed	No Issue
10	_earned	internal	Passed	No Issue
11	lastTimeRewardApplicable	read	Passed	No Issue
12	rewardPerToken	external	Passed	No Issue
13	getRewardForDuration	external	Passed	No Issue
14	claimableRewards	external	Passed	No Issue
15	totalBalance	external	Passed	No Issue
16	unlockedBalance	external	Passed	No Issue
17	earnedBalances	external	Passed	No Issue
18	lockedBalances	external	Passed	No Issue
19	withdrawableBalance	read	Passed	No Issue
20	stake	external	Passed	No Issue
21	mint	external	Function input parameters lack of check	Refer Audit Findings
22	withdraw	write	Passed	No Issue
23	getReward	write	Passed	No Issue
24	emergencyWithdraw	external	Critical operation lacks event log	Refer Audit Findings
25	withdrawExpiredLocks	external	Passed	No Issue
26	_notifyReward	internal	Passed	No Issue
27	notifyRewardAmount	external	Passed	No Issue

28	recoverERC20	external	access only Owner	No Issue
29	setTeamWalletAddress	external	Passed	No Issue
30	setTeamRewardPercent	external	Passed	No Issue
31	updateReward	modifier	Passed	No Issue
32	receive	external	Passed	No Issue

DaoZapMMSwap.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	nonReentrant	modifier	Passed	No Issue
8	zap	external	Passed	No Issue
9	swap	internal	Passed	No Issue
10	doSwapETH	internal	Passed	No Issue
11	approveToken	internal	Passed	No Issue
12	calculateSwapInAmount	internal	Passed	No Issue
13	addZap	external	access only Owner	No Issue
14	removeZap	external	access only Owner	No Issue

NFTController.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	initializer	modifier	Passed	No Issue
8	reinitializer	modifier	Passed	No Issue
9	onlyInitializing	modifier	Passed	No Issue
10	_disableInitializers	internal	Passed	No Issue
11	_setInitializedVersion	write	Passed	No Issue
12	initialize	write	access by initializer	No Issue
13	getBoostRate	external	Passed	No Issue
14	setWhitelist	external	access only Owner	No Issue
15	setDefaultBoostRate	external	access only Owner	No Issue
16	setBoostRate	external	access only Owner	No Issue

NFTControllerProxy.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	ifAdmin	modifier	Passed	No Issue
3	admin	external	access if Admin	No Issue
4	implementation	external	access if Admin	No Issue
5	changeAdmin	external	access if Admin	No Issue
6	upgradeTo	external	access if Admin	No Issue
7	upgradeToAndCall	external	access if Admin	No Issue
8	_admin	internal	Passed	No Issue
9	_beforeFallback	internal	Passed	No Issue

Fund.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	initializer	modifier	Passed	No Issue
8	reinitializer	modifier	Passed	No Issue
9	onlyInitializing	modifier	Passed	No Issue
10	_disableInitializers	internal	Passed	No Issue
11	_setInitializedVersion	write	Passed	No Issue
12	initialize	external	Passed	No Issue
13	allocation	read	Passed	No Issue
14	vestingStart	read	Passed	No Issue
15	vestingDuration	read	Passed	No Issue
16	currentBalance	read	Passed	No Issue
17	vestedBalance	read	Passed	No Issue
18	claimable	read	Passed	No Issue
19	transfer	external	access only Owner	No Issue

DaoFund.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	initialize	external	Passed	No Issue
3	allocation	read	Passed	No Issue

4	vestingStart	read	Passed	No Issue
5	vestingDuration	read	Passed	No Issue
6	currentBalance	read	Passed	No Issue
7	vestedBalance	read	Passed	No Issue
8	claimable	read	Passed	No Issue
9	transfer	external	access only Owner	No Issue
10	allocation	write	Passed	No Issue
11	vestingStart	write	Passed	No Issue
12	vestingDuration	write	Passed	No Issue

DevFund.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	initialize	external	Passed	No Issue
3	allocation	read	Passed	No Issue
4	vestingStart	read	Passed	No Issue
5	vestingDuration	read	Passed	No Issue
6	currentBalance	read	Passed	No Issue
7	vestedBalance	read	Passed	No Issue
8	claimable	read	Passed	No Issue
9	transfer	external	access only Owner	No Issue
10	allocation	write	Passed	No Issue
11	vestingStart	write	Passed	No Issue
12	vestingDuration	write	Passed	No Issue

Reserve.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	initializer	modifier	Passed	No Issue
8	reinitializer	modifier	Passed	No Issue
9	onlyInitializing	modifier	Passed	No Issue
10	_disableInitializers	internal	Passed	No Issue
11	_setInitializedVersion	write	Passed	No Issue
12	initialize	external	access by initializer	No Issue
13	setRewarder	external	Passed	No Issue
14	setPool	external	access only Owner	No Issue
15	removePool	external	access only Owner	No Issue

16	transfer	external	Passed	No Issue

TreasuryFund.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	initialize	external	Passed	No Issue
3	allocation	read	Passed	No Issue
4	vestingStart	read	Passed	No Issue
5	vestingDuration	read	Passed	No Issue
6	currentBalance	read	Passed	No Issue
7	vestedBalance	read	Passed	No Issue
8	claimable	read	Passed	No Issue
9	transfer	external	access only Owner	No Issue
10	allocation	write	Passed	No Issue
11	vestingStart	write	Passed	No Issue
12	vestingDuration	write	Passed	No Issue

MockERC20.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	name	read	Passed	No Issue
3	symbol	read	Passed	No Issue
4	decimals	read	Passed	No Issue
5	totalSupply	read	Passed	No Issue
6	balanceOf	read	Passed	No Issue
7	transfer	write	Passed	No Issue
8	allowance	read	Passed	No Issue
9	approve	write	Passed	No Issue
10	transferFrom	write	Passed	No Issue
11	increaseAllowance	write	Passed	No Issue
12	decreaseAllowance	write	Passed	No Issue
13	_transfer	internal	Passed	No Issue
14	_mint	internal	Passed	No Issue
15	_burn	internal	Passed	No Issue
16	_approve	internal	Passed	No Issue
17	_spendAllowance	internal	Passed	No Issue
18	_beforeTokenTransfer	internal	Passed	No Issue
19	_afterTokenTransfer	internal	Passed	No Issue
20	mint	write	Passed	No Issue
21	decimals	read	Passed	No Issue

MockTreasury.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	mock	write	Passed	No Issue
3	info	read	Passed	No Issue

MasterOracle.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	getXTokenPrice	write	Passed	No Issue
8	getYTokenPrice	write	Passed	No Issue
9	getXTokenTWAP	write	Passed	No Issue
10	getYTokenTWAP	write	Passed	No Issue

UniswapPairOracle.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	setPeriod	external	access only Owner	No Issue
8	update	external	Passed	No Issue
9	twap	external	Passed	No Issue
10	spot	external	Passed	No Issue
11	currentBlockTimestamp	internal	Passed	No Issue
12	currentCumulativePrices	internal	Passed	No Issue

XToken.sol

Functions

Sl.	Functions	Type	Observation	Conclusion

1	constructor	write	Passed	No Issue
2	onlyMinter	modifier	Passed	No Issue
3	setMinter	external	access only Owner	No Issue
4	removeMinter	external	access only Owner	No Issue
5	mint	external	Unlimited Minting	Refer Audit Findings
6	burn	write	Passed	No Issue
7	burnFrom	write	Passed	No Issue
8	owner	read	Passed	No Issue
9	onlyOwner	modifier	Passed	No Issue
10	renounceOwnership	write	access only Owner	No Issue
11	transferOwnership	write	access only Owner	No Issue
12	_transferOwnership	internal	Passed	No Issue

YToken.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	burn	write	Passed	No Issue
3	burnFrom	write	Passed	No Issue

LOVE.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	OpenTrade	external	Passed	No Issue
3	includeToWhitelist	write	Passed	No Issue
4	excludeFromWhitlist	write	Passed	No Issue
5	_transfer	internal	Passed	No Issue

MMFX.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyMinter	modifier	Passed	No Issue
3	setMinter	external	access only Owner	No Issue
4	removeMinter	external	access only Owner	No Issue
5	mint	external	access only Minter	No Issue
6	OpenTrade	external	Passed	No Issue
7	includeToWhitelist	write	Passed	No Issue
8	excludeFromWhitlist	write	Passed	No Issue
9	_transfer	internal	Passed	No Issue

MMOX.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyMinter	modifier	Passed	No Issue
3	setMinter	external	access only Owner	No Issue
4	removeMinter	external	access only Owner	No Issue
5	mint	external	access only Minter	No Issue
6	OpenTrade	external	Passed	No Issue
7	includeToWhitelist	write	Passed	No Issue
8	excludeFromWhitlist	write	Passed	No Issue
9	_transfer	internal	Passed	No Issue

MMUSD.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyMinter	modifier	Passed	No Issue
3	setMinter	external	access only Owner	No Issue
4	removeMinter	external	access only Owner	No Issue
5	mint	external	access only Minter	No Issue
6	OpenTrade	external	Passed	No Issue
7	includeToWhitelist	write	Passed	No Issue
8	excludeFromWhitlist	write	Passed	No Issue
9	_transfer	internal	Passed	No Issue

DaoTreasury.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	balanceOf	r	Passed	No Issue
8	requestFund	external	Passed	No Issue
9	addStrategy	external	access only Owner	No Issue
10	removeStrategy	external	access only Owner	No Issue
11	allocateFee	external	access only Owner	No Issue

StratRecollateralize.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	recollateralize	external	access only Owner	No Issue
8	receive	external	Passed	No Issue

StratReduceReserveLP.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	_transferOwnership	internal	Passed	No Issue
7	reduceReserve	external	access only Owner	No Issue
8	swap	internal	Passed	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Critical operation lacks event log:**- DaoStaking.sol**

Missing event log for: emergencyWithdraw

Resolution: Write an event log for listed events.

(2) Function input parameters lack of check: **- DaoStaking.sol**

Variable validation is not performed in the functions below:

- addReward =_rewardsToken
- mint = user

Resolution: We advise to put validation like integer type variables should be greater than 0 and address type variables should not be address(0).

Very Low / Informational / Best practices:

(1) Unlimited Minting: **- XToken.sol**

Minter can mint unlimited tokens.

Resolution: We suggest putting a minting limit.

Centralization

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- toggle: Pool owner can Turn on / off minting and redemption.
- setCollateralRatioOptions: Pool owner can configure variables related to Collateral Ratio.
- toggleCollateralRatio: Pool Owner can pause or unpause collateral ratio updates.
- setFees: Pool owners can set the protocol fees.
- setMinCollateralRatio: Pool owner can set the minimum Collateral Ratio.
- reduceExcessCollateral: Pool owner can transfer the excess balance of WETH to FeeReserve.
- setSwapStrategy: Pool owner can set the address of Swapper utils.
- setOracle: Pool owner can set new oracle address.
- setYTokenSlippage: Pool owner can set yTokenSlipage.
- setTreasury: Pool owner can set the address of the Treasury.
- cleanDust: SwapStrategyPOL owner can clean dust.
- changeSlippage: SwapStrategyPOL owner can change slippage value.
- add: DaoChef owner can add a new LP to the pool.
- set: DaoChef owner can update the given pool's reward allocation point and `IRewarder` contract.
- setRewardPerSecond: DaoChef owner can set the reward per second to be distributed.
- setRewardMinter: DaoChef owner can set the address of rewardMinter.
- setNftController: DaoChef owner can set NFT controller address.
- setNftBoostRate: DaoChef owner can set NFT Boost Rate value.
- addReward: DaoStaking owner can add a new reward token to be distributed to stakers.
- approveRewardDistributor: DaoStaking owner can modify approval for an address to call notifyRewardAmount.
- recoverERC20: DaoStaking owner can be added to support recovering LP Rewards from other systems such as BAL to be distributed to holders.

- setTeamWalletAddress: DaoStaking owner can set the address of the team wallet.
- setTeamRewardPercent: DaoStaking owner can set percent of team reward.
- addZap: DaoZapMMSwap owner can add new zap configuration.
- removeZap: DaoZapMMSwap owner can deactivate a Zap configuration.
- setWhitelist: NFTController owner can set whitelist address,
- setDefaultBoostRate: NFTController owner can set default boost rate value.
- setBoostRate: NFTController owner can set boost rate value.
- transfer: Fund owners can transfer tokens.
- setPool: Reserve owner can set pool address.
- removePool: Reserve owner can remove pool address.
- setPeriod: UniswapPairOracle owner can set maximum and minimum period.
- removeMinter: XToken minter can remove minter address.
- setMinter: XToken minter can set minter for XToken.
- mint: XToken minter can mint new XToken.
- OpenTrade: LOVE owners can trade openly.
- includeToWhitelist: LOVE owner can include address to whitelist.
- excludeFromWhitlist: LOVE owner can exclude address to whitelist.
- OpenTrade: MMFX owners can trade openly.
- includeToWhitelist: MMFX owner can include address to whitelist.
- excludeFromWhitlist: MMFX owner can exclude address to whitelist.
- OpenTrade: MMOX owners can trade openly.
- includeToWhitelist: MMOX owner can include address to whitelist.
- excludeFromWhitlist: MMOX owner can exclude address to whitelist.
- OpenTrade: MMUSD owners can trade openly.
- includeToWhitelist: MMUSD owner can include address to whitelist.
- excludeFromWhitlist: MMUSD owner can exclude address to whitelist.
- addStrategy: DaoTreasury owner can add new strategy.
- removeStrategy: DaoTreasury owner can remove current strategy.
- allocateFee: DaoTreasury owner can allocate protocol's fee to stakers.
- recollateralize: StratRecollateralize owner can recollateralize the minting pool.
- reduceReserve: StratReduceReserveLP owner can remove liquidity, buy back YToken and burn.

To make the smart contract 100% decentralized, we suggest renouncing ownership in the airdrop smart contract once its function is completed.

Conclusion

We were given a contract code in the form of files. And we have used all possible tests based on given objects as files. We have not observed any major issues in the smart contracts. **So, the smart contracts are ready for the mainnet deployment**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured"**.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

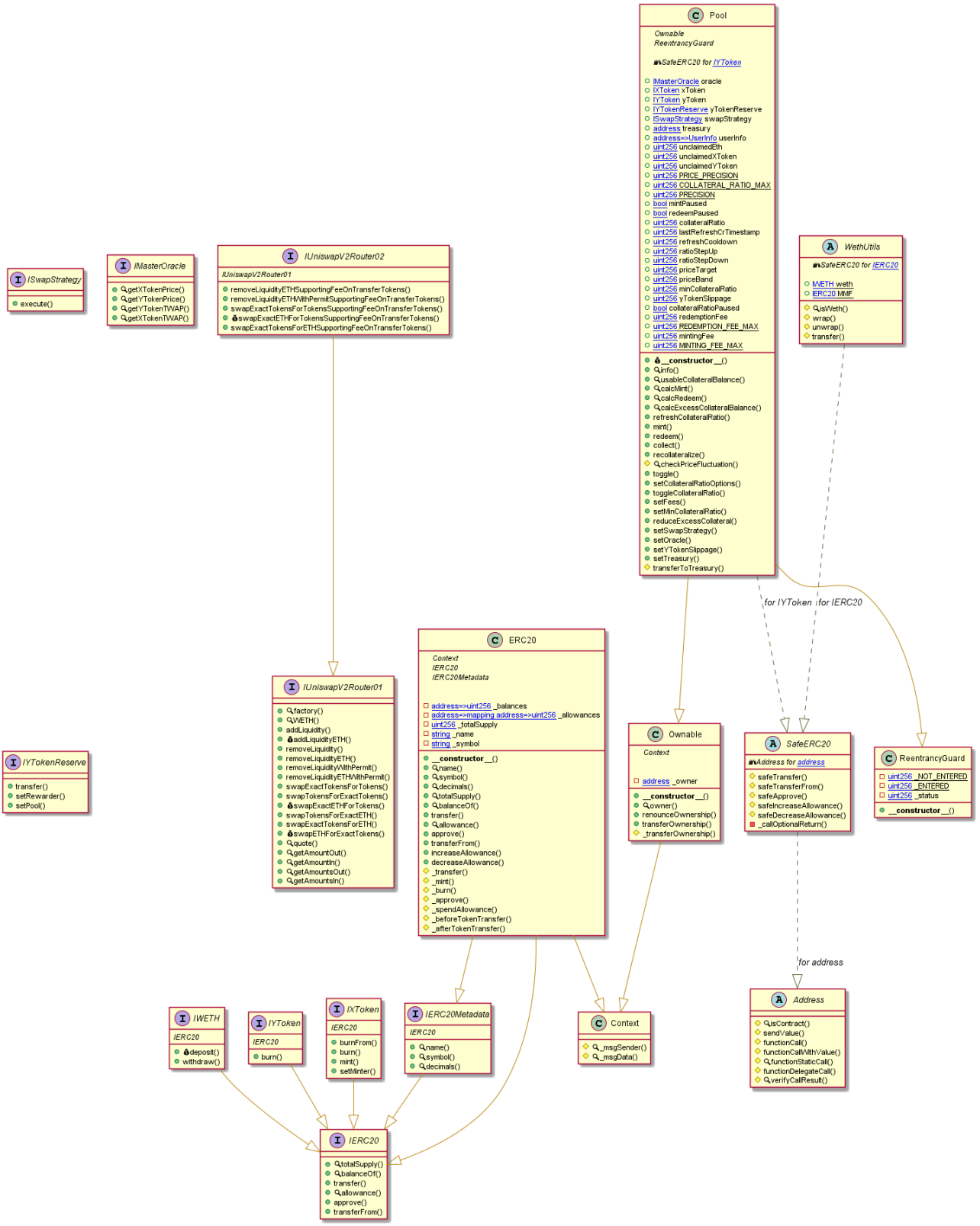
Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

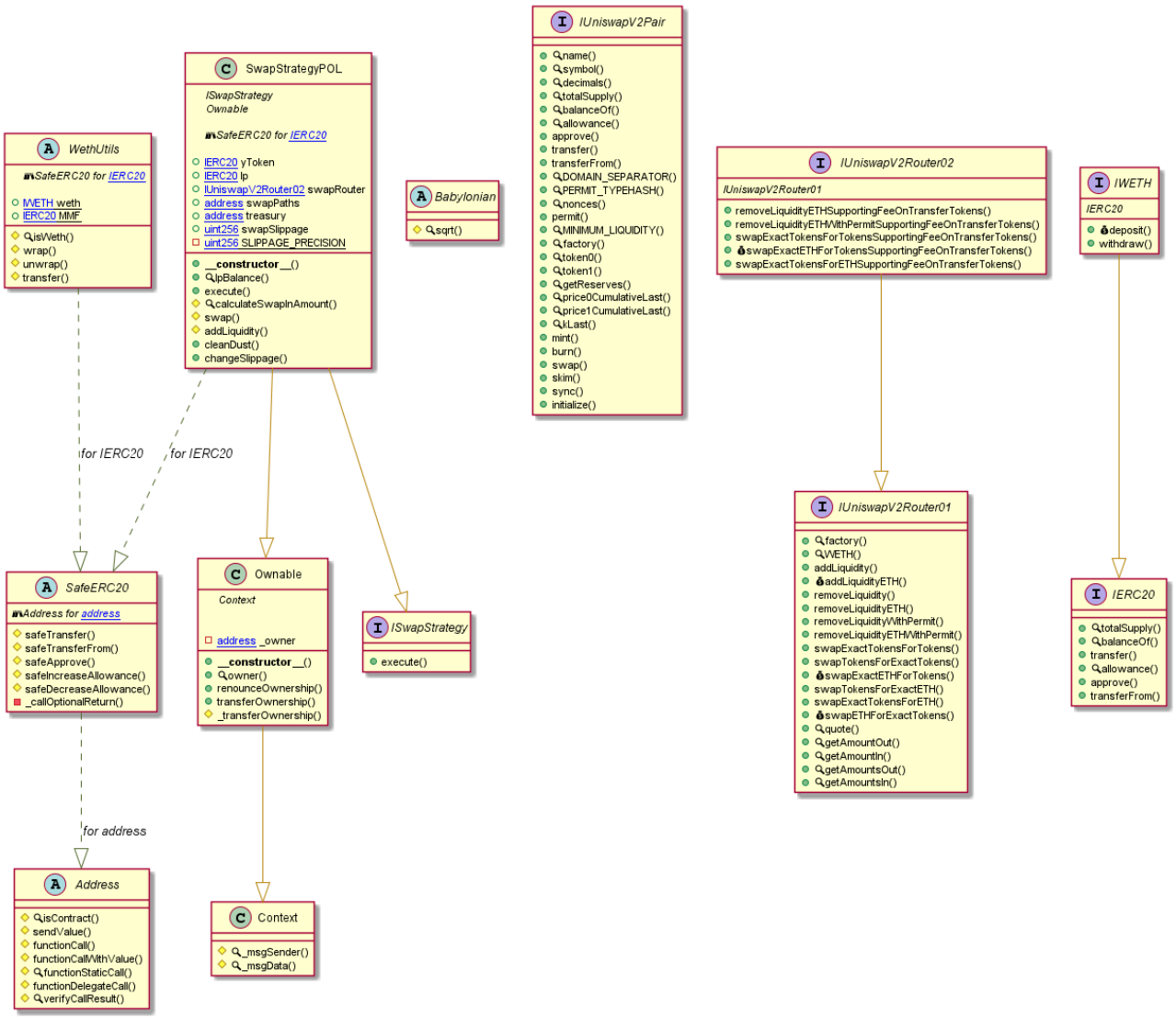
Appendix

Code Flow Diagram - Frontline DAO Protocol

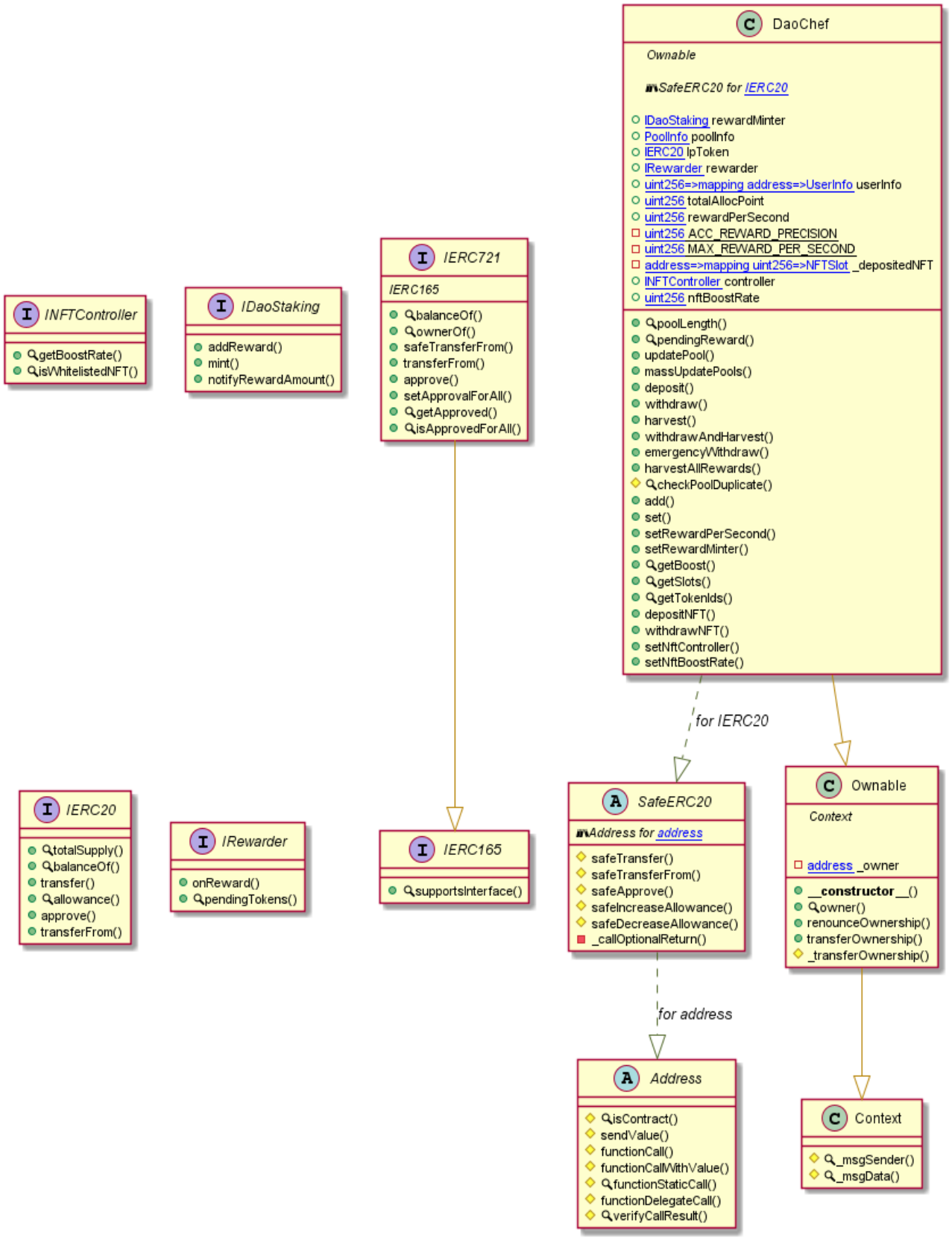
Pool Diagram



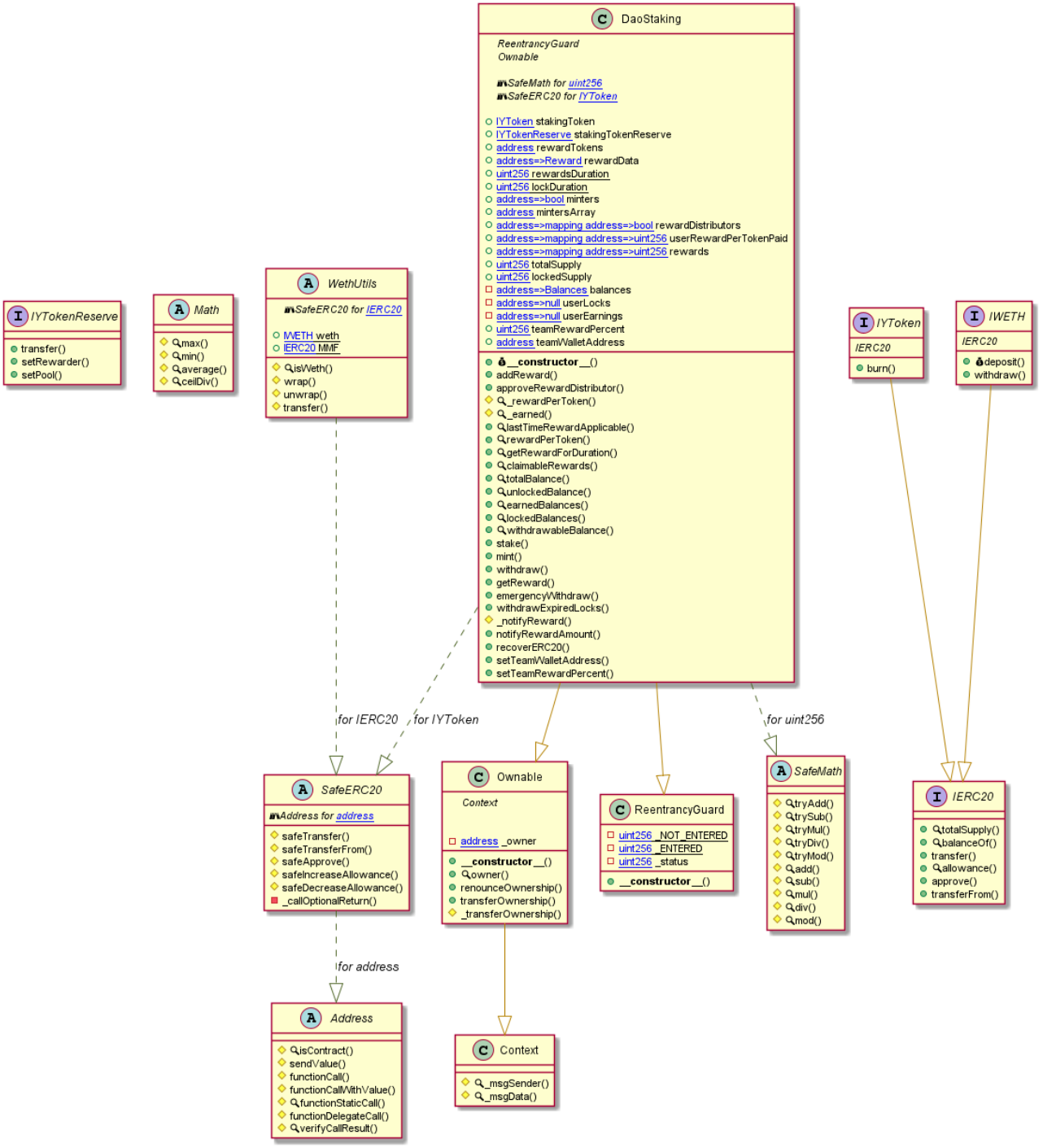
SwapStrategyPOL Diagram



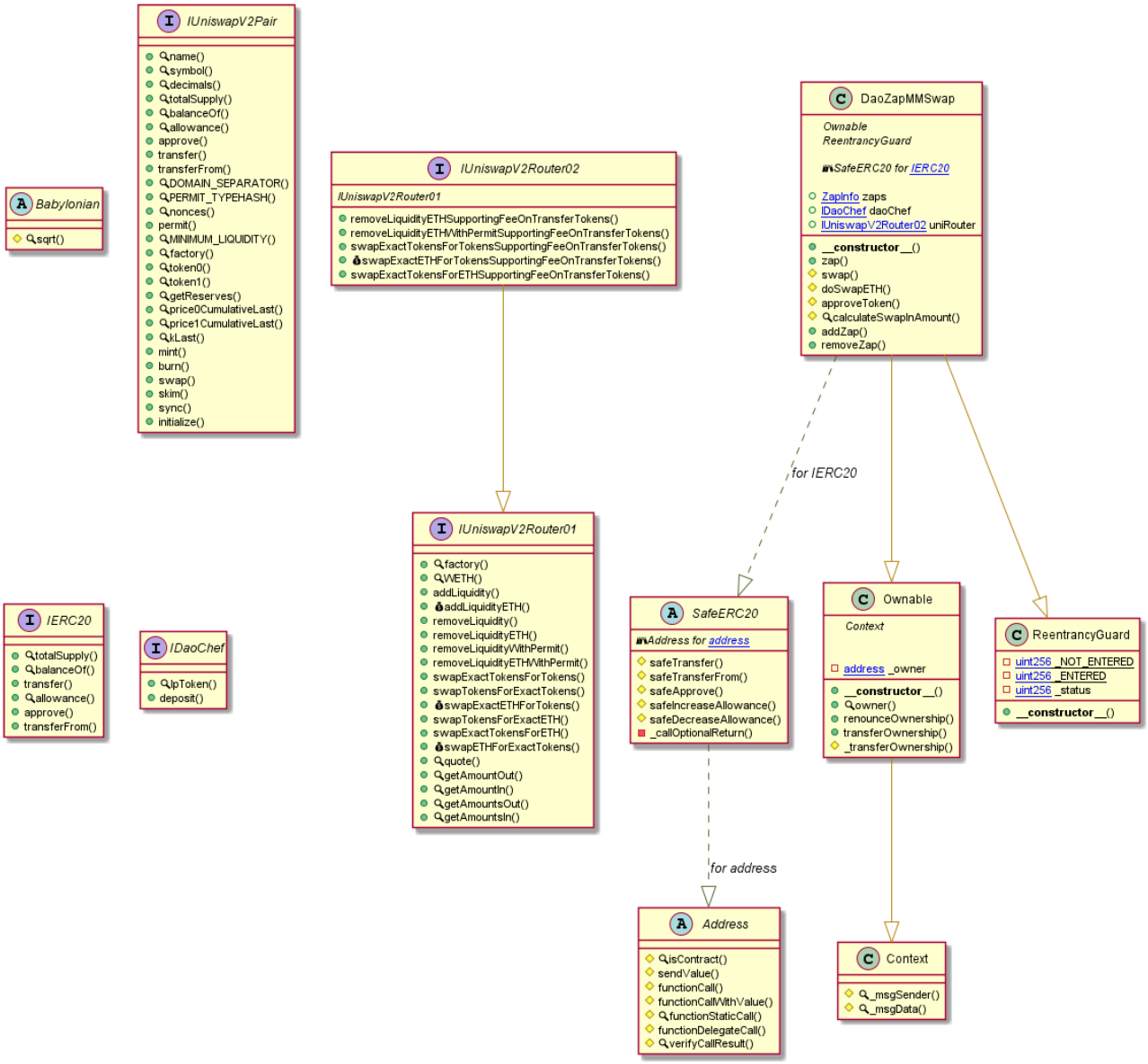
DaoChef Diagram



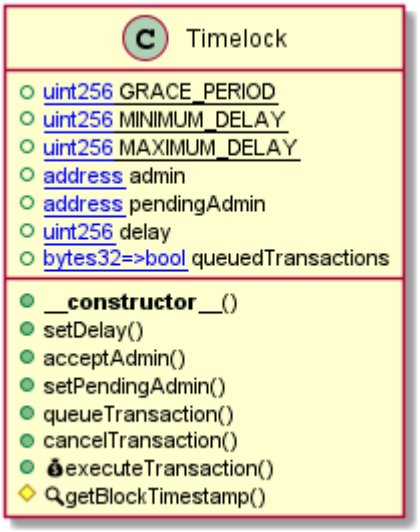
DaoStaking Diagram



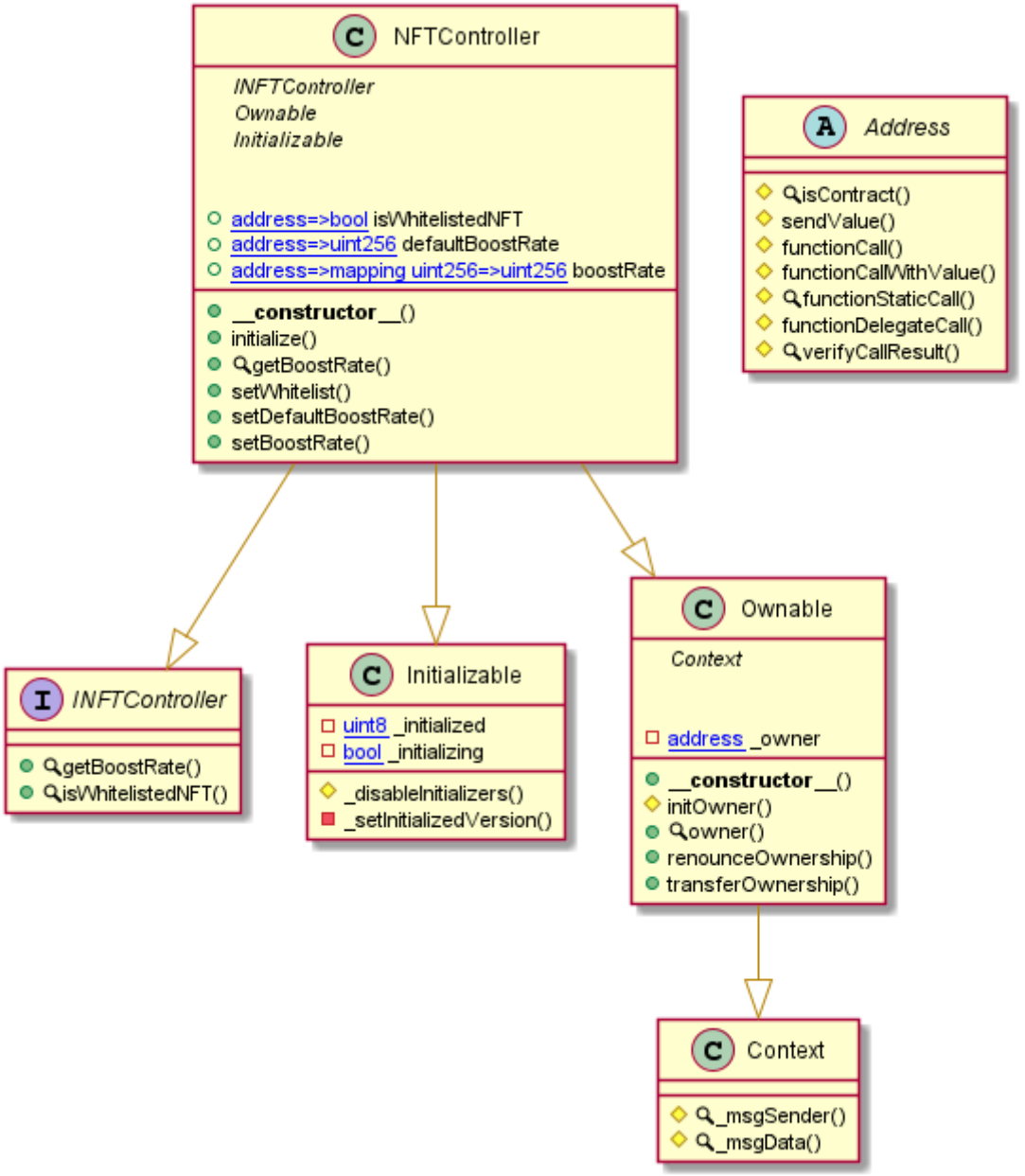
DaoZapMMSwap Diagram



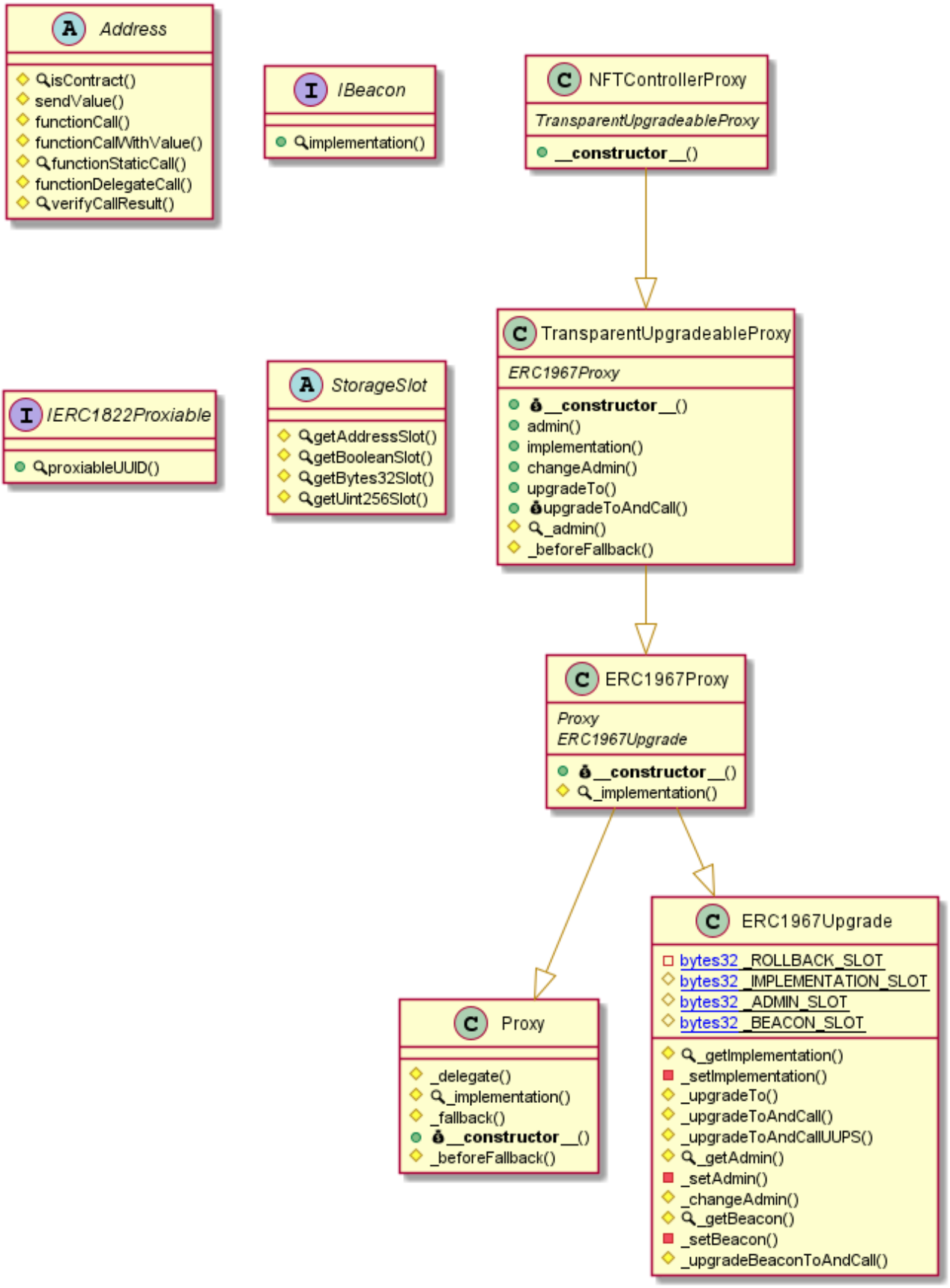
Timelock Diagram



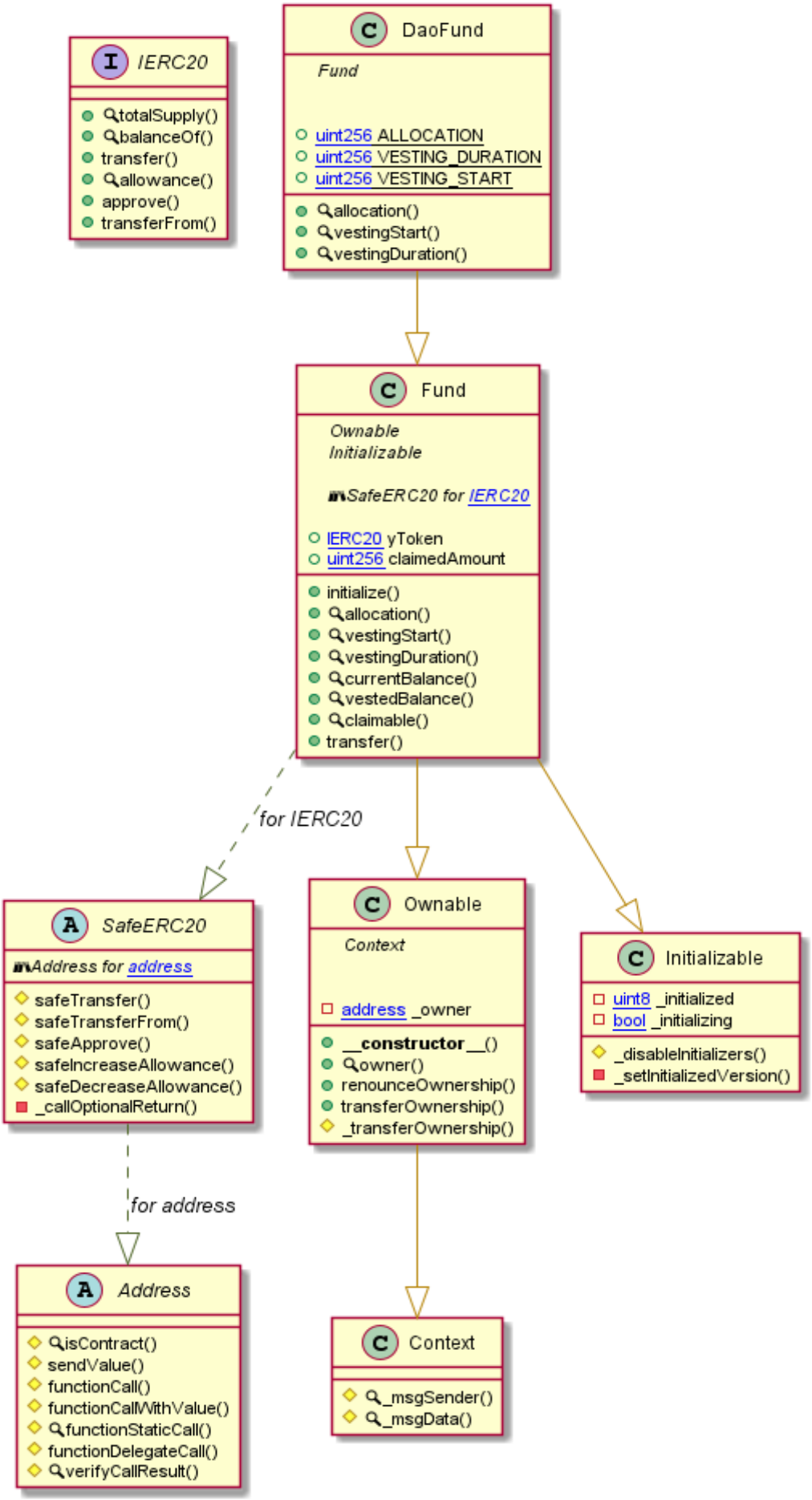
NFTController Diagram



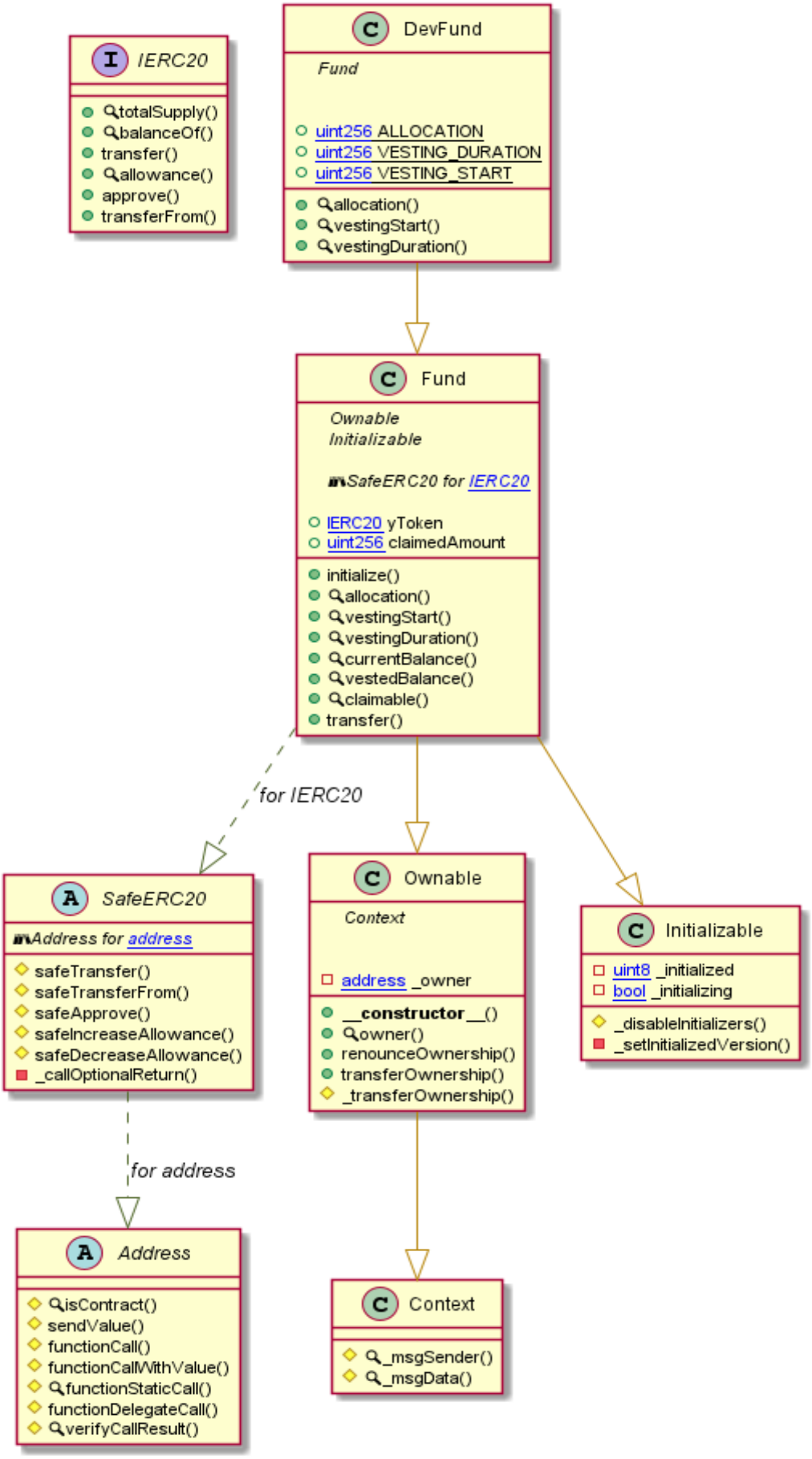
NFTControllerProxy Diagram



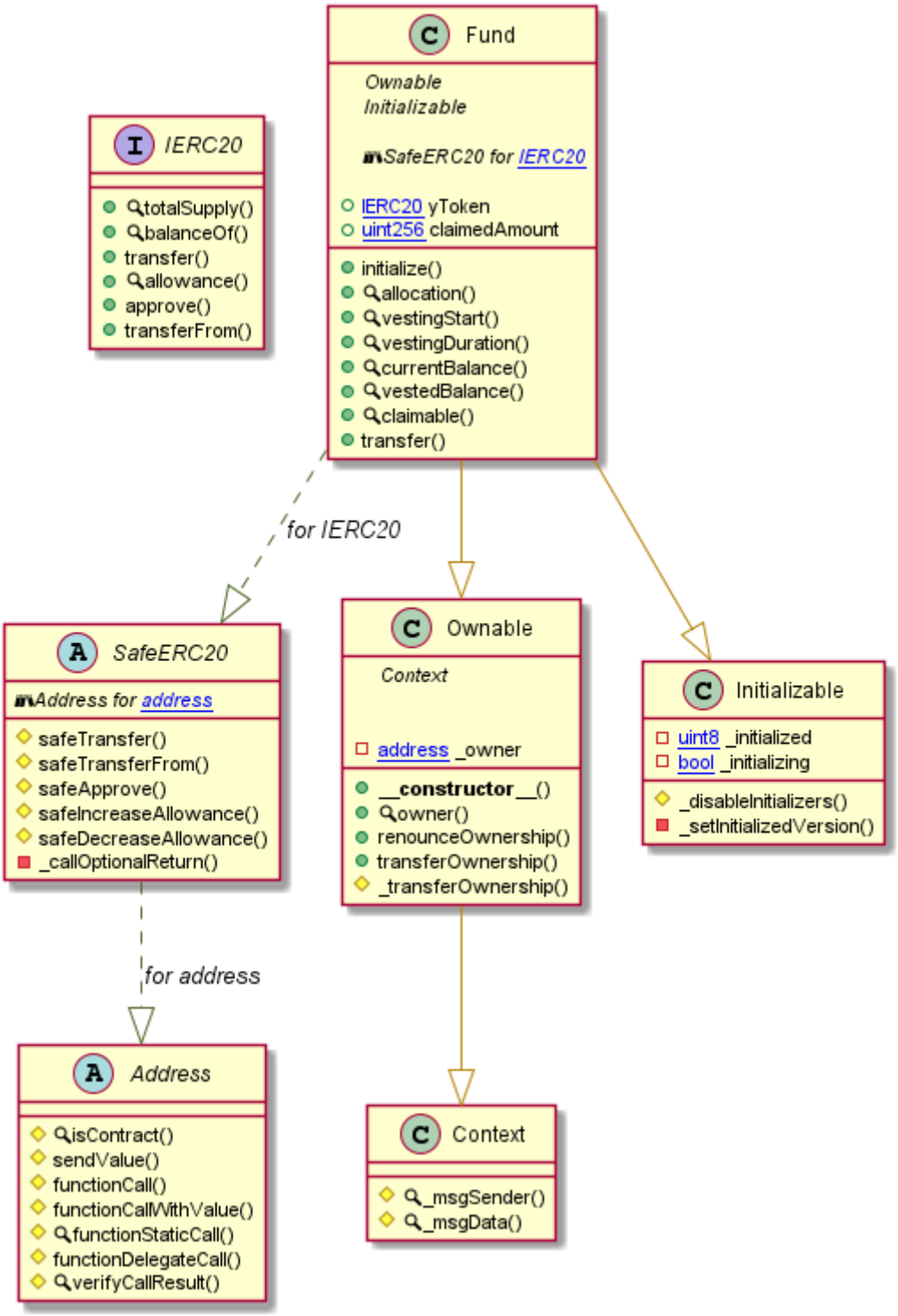
DaoFund Diagram



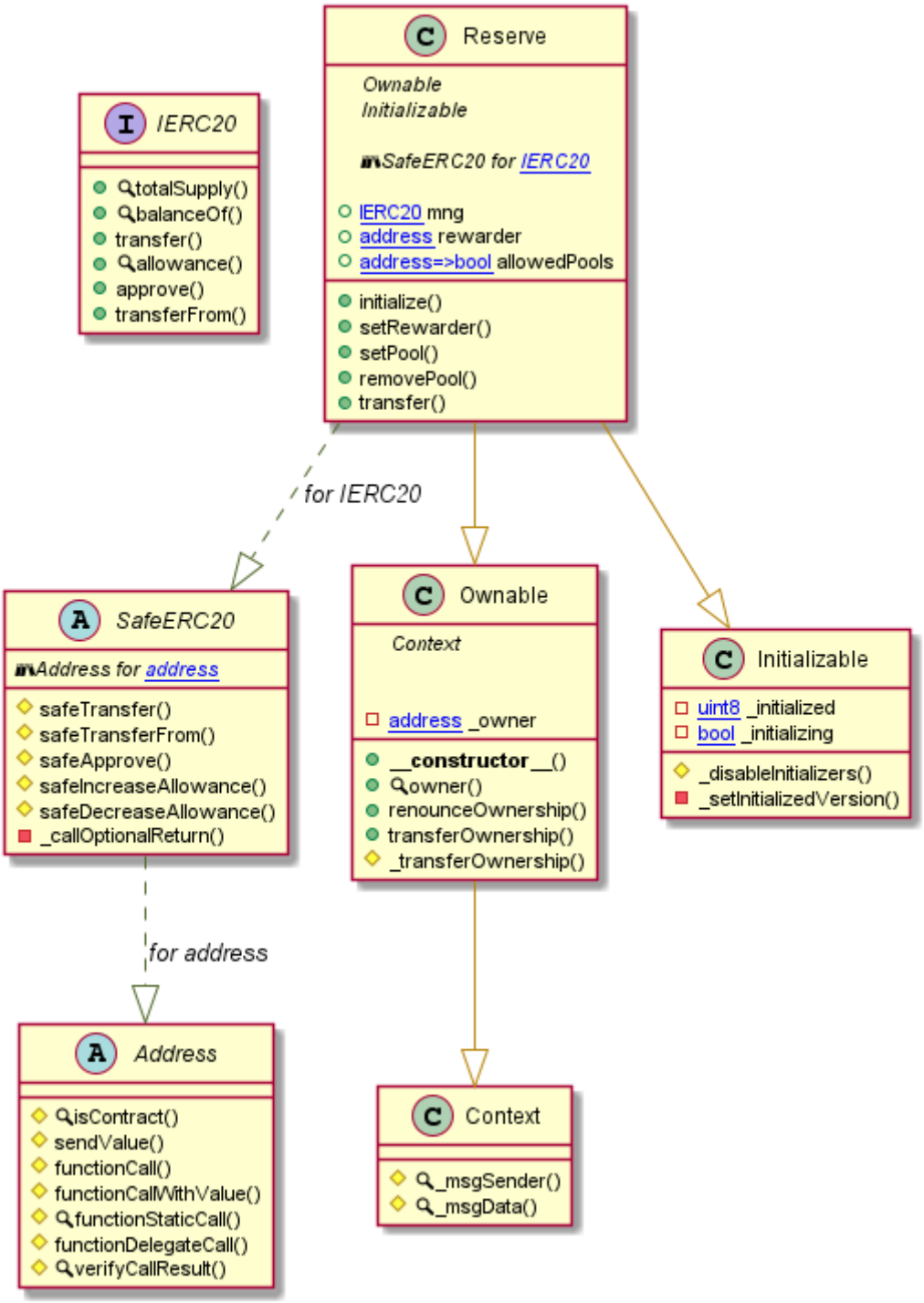
DevFund Diagram



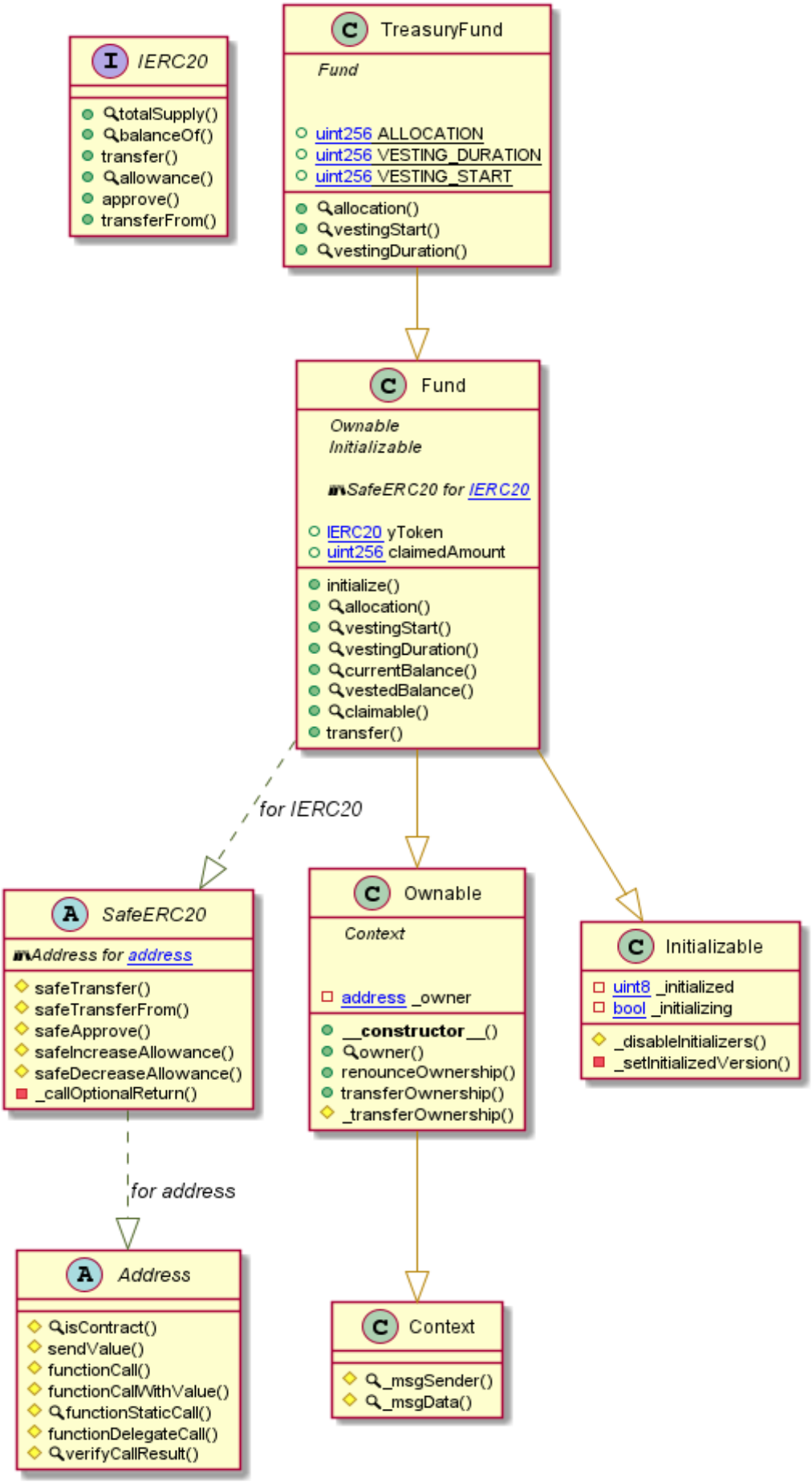
Fund Diagram



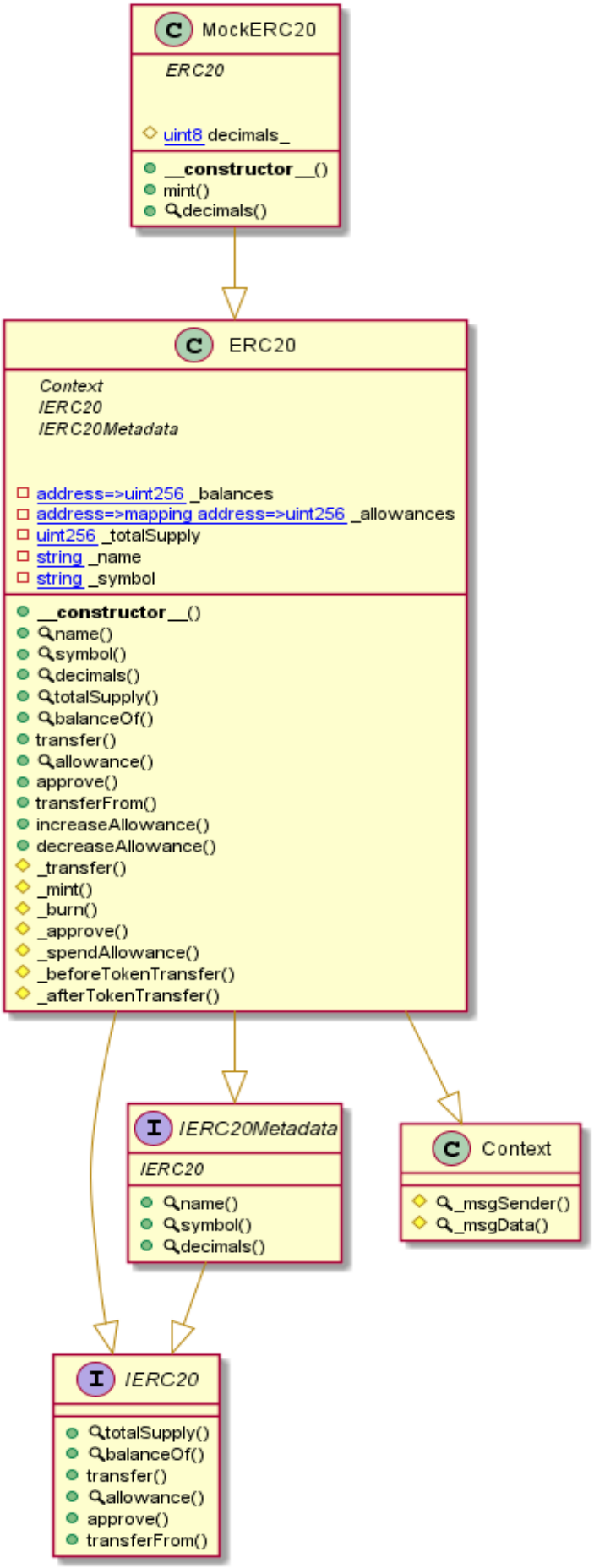
Reserve Diagram



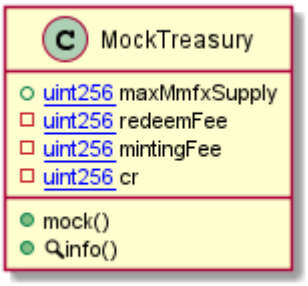
TreasuryFund Diagram



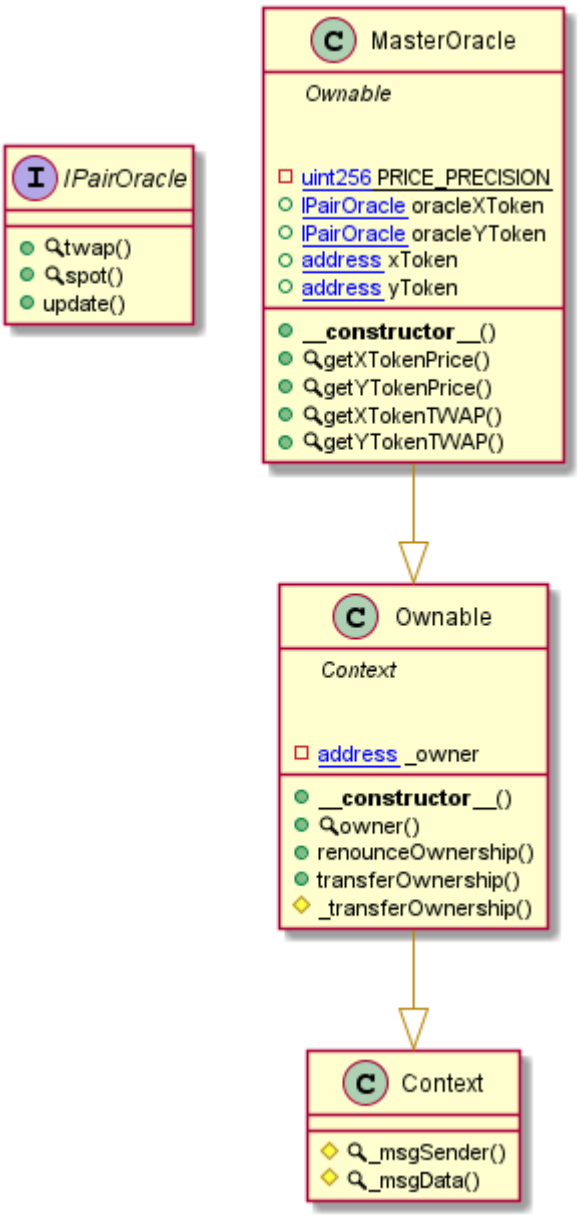
MockERC20 Diagram



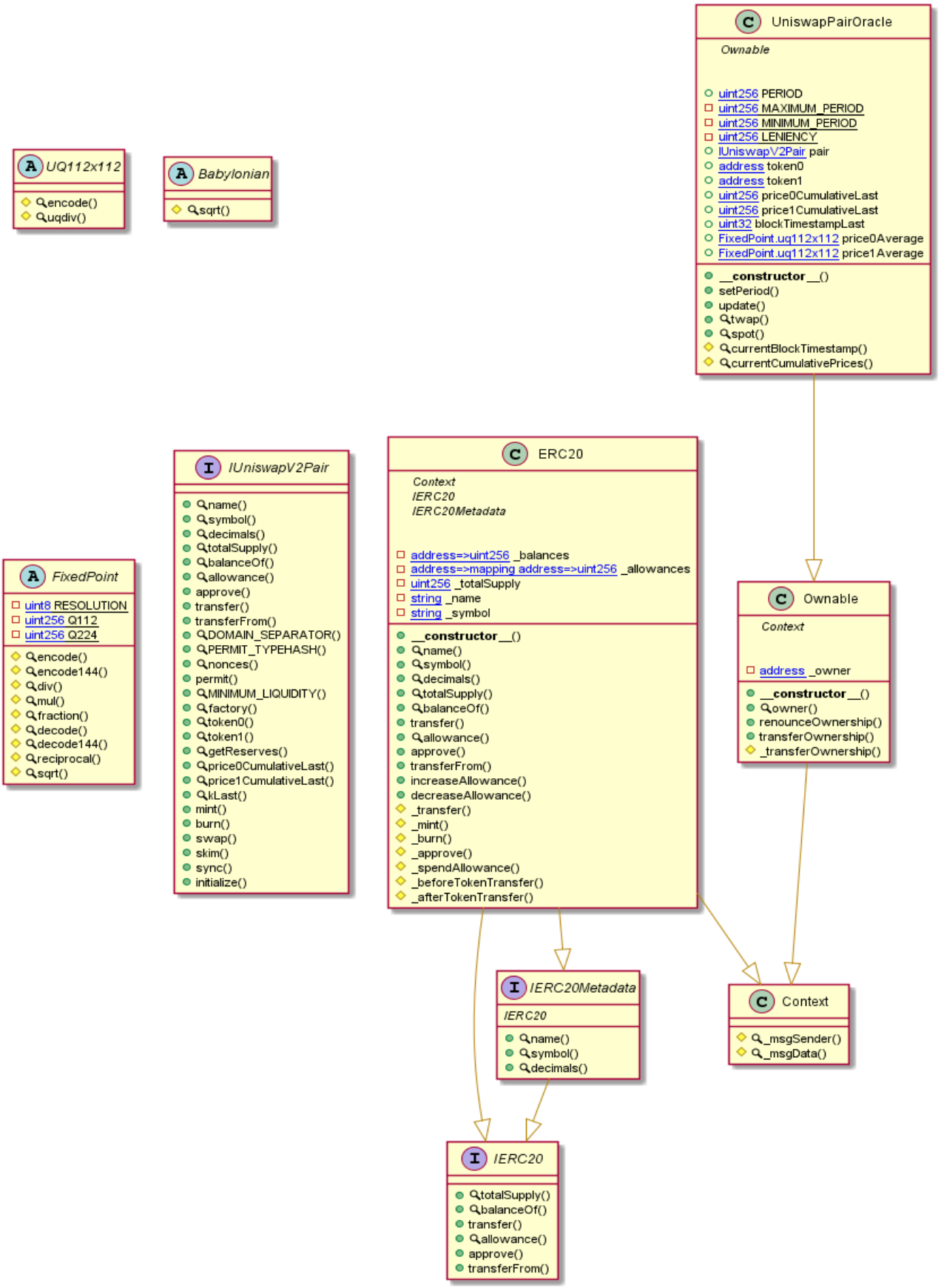
MockTreasury Diagram



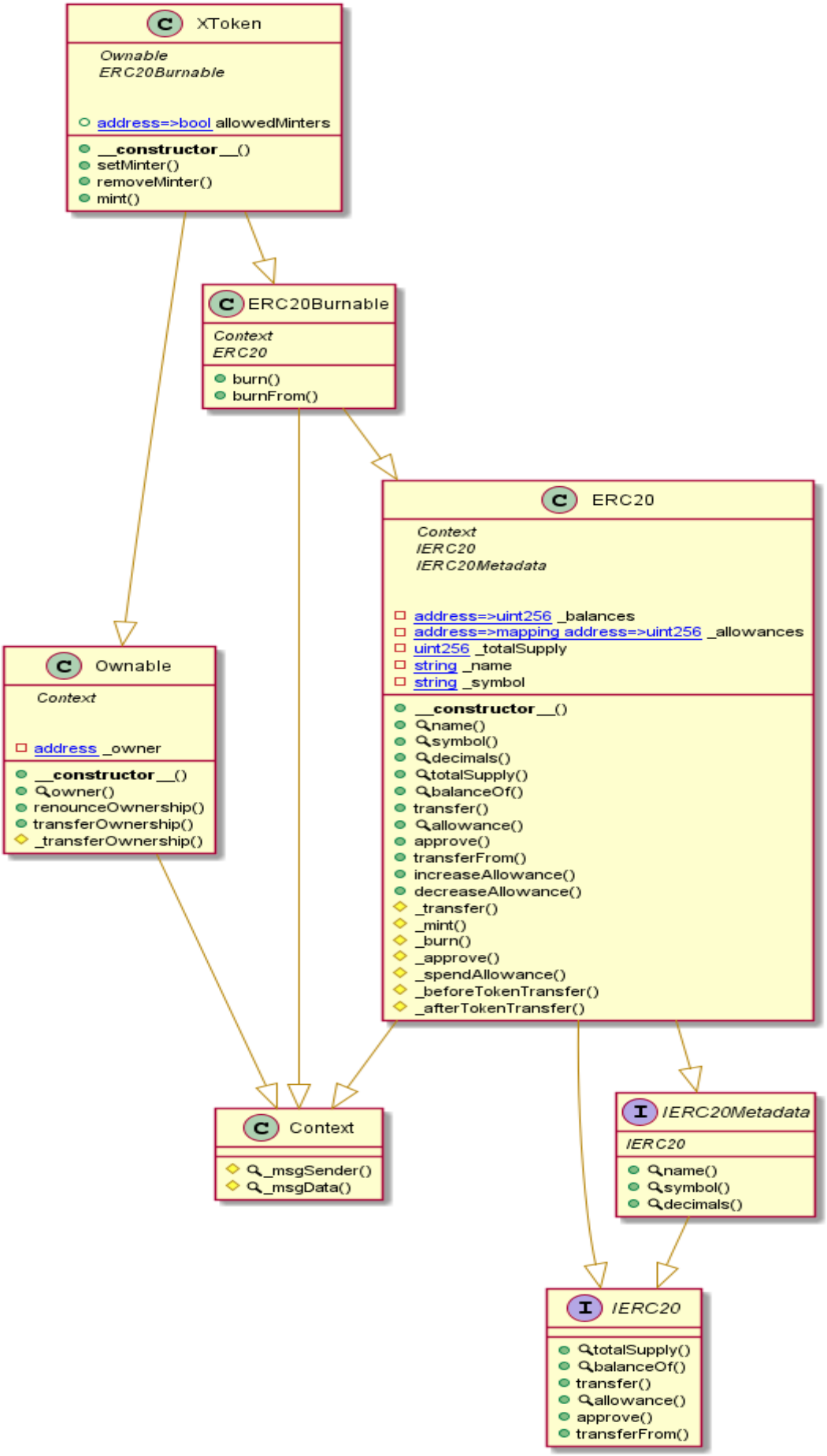
MasterOracle Diagram



UniswapPairOracle Diagram

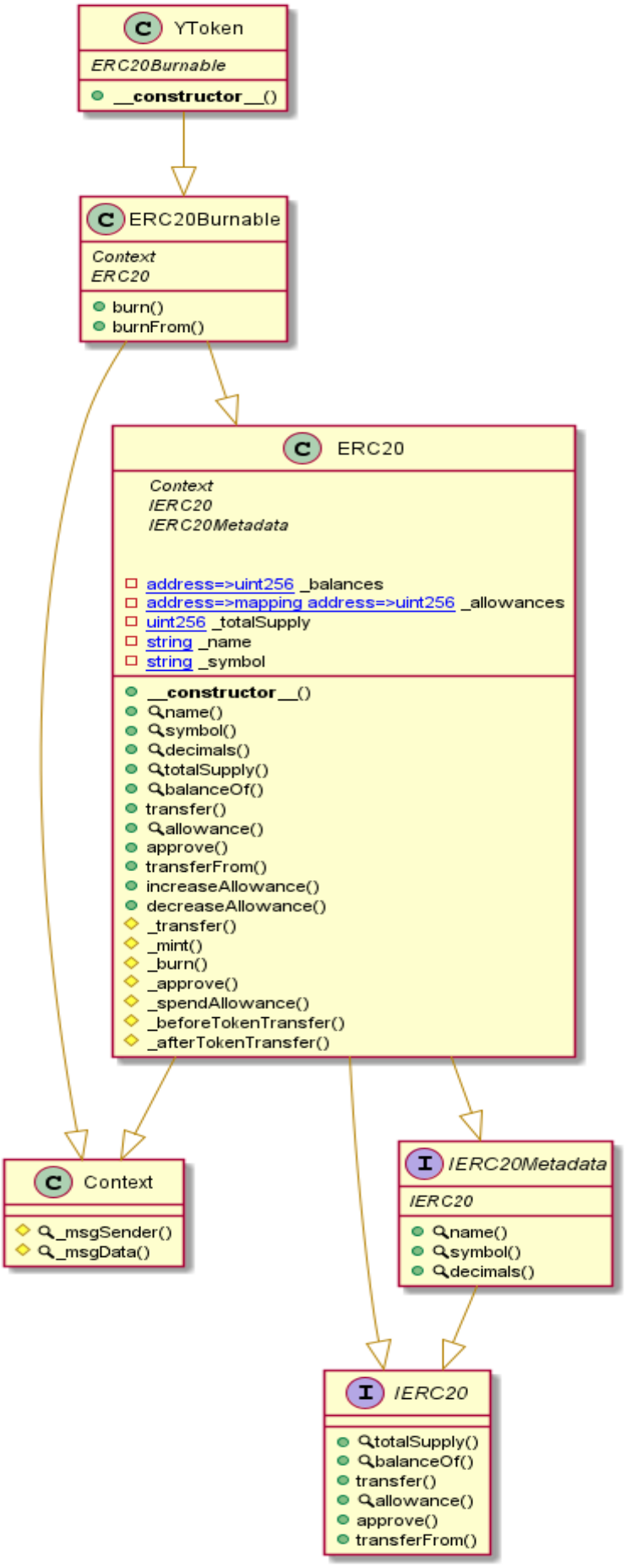


XToken Diagram

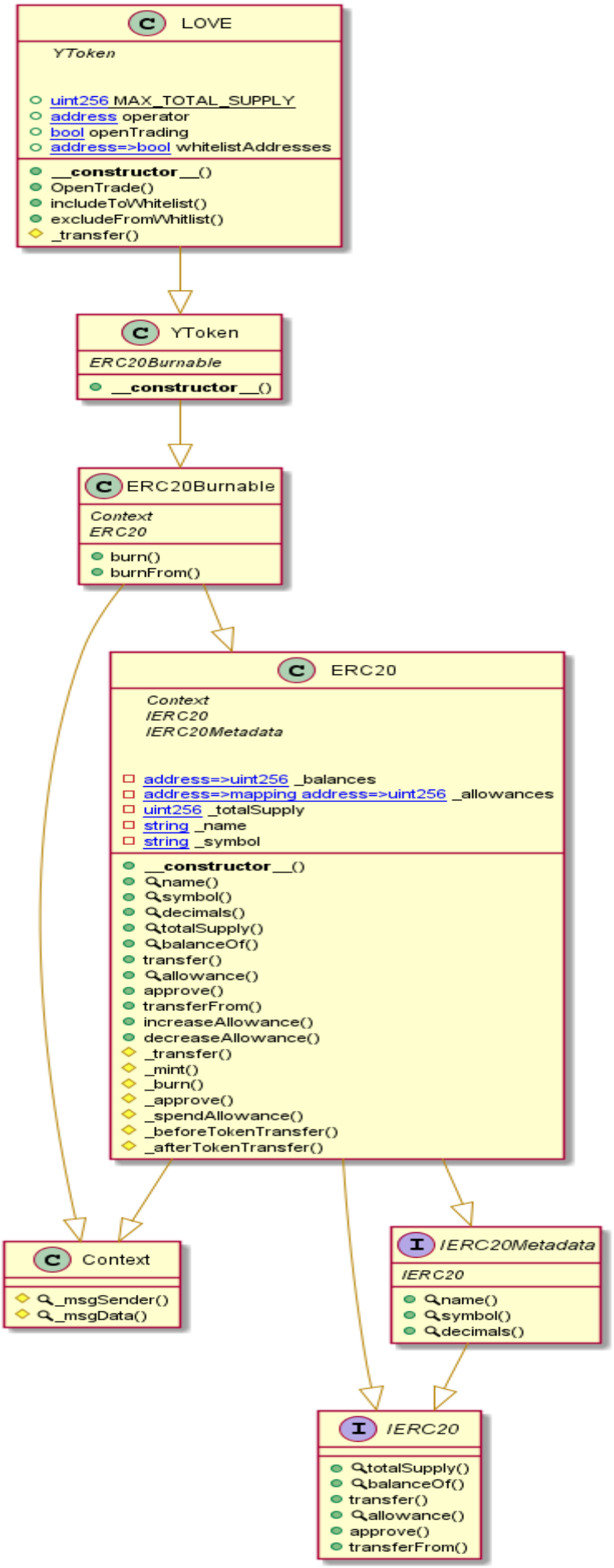


Email: audit@EtherAuthority.io

YToken Diagram

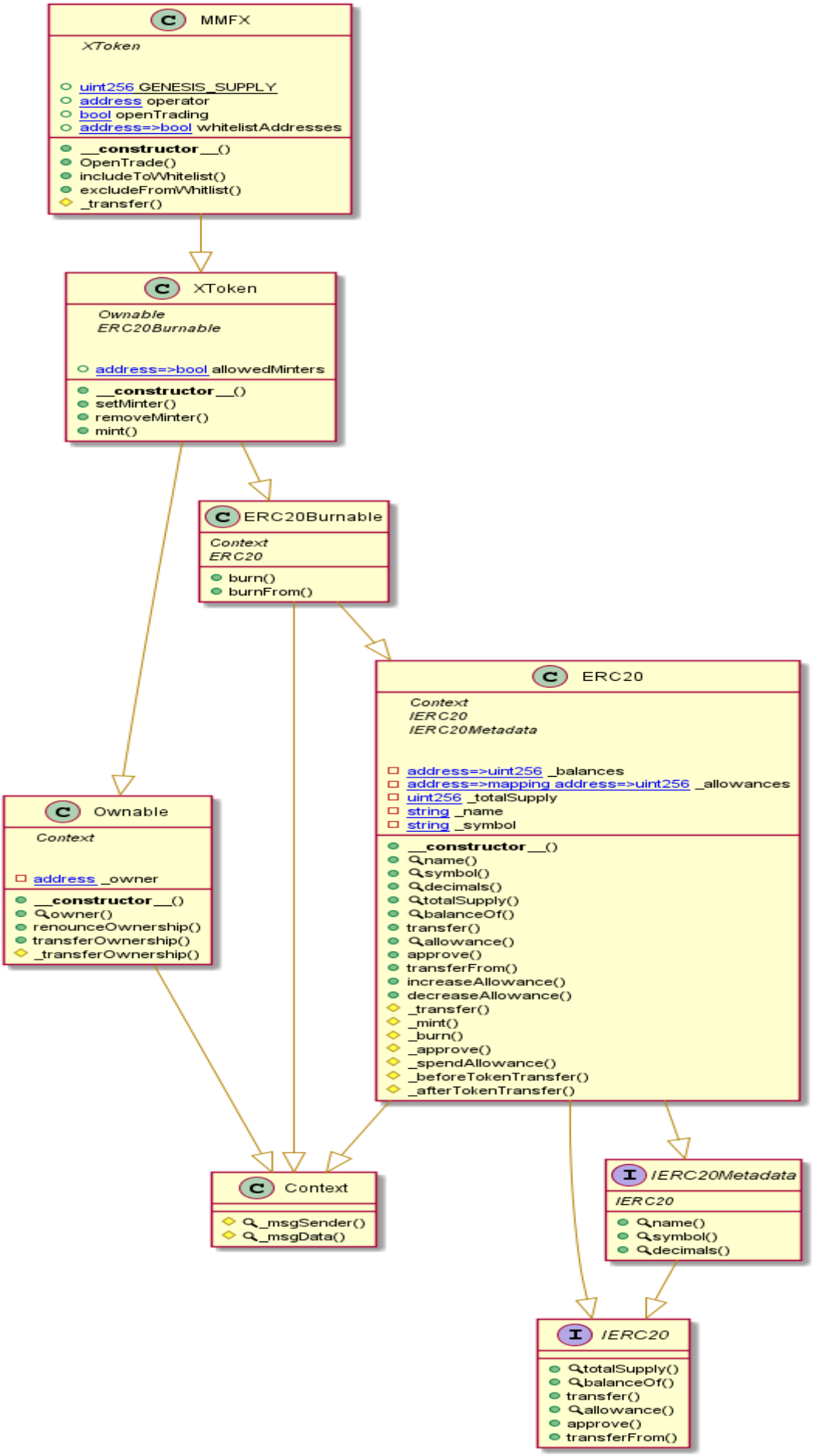


LOVE Diagram

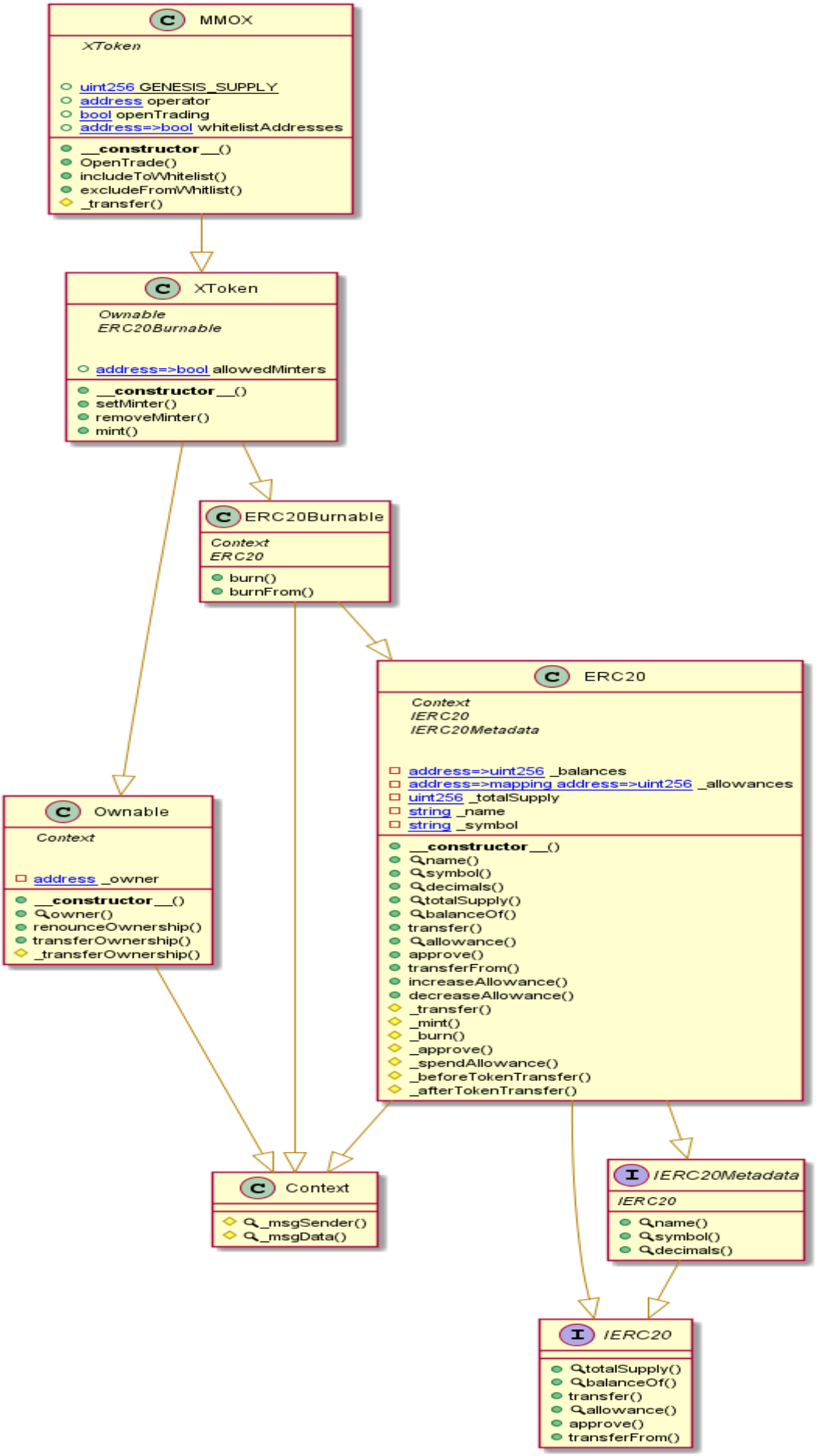


Email: audit@EtherAuthority.io

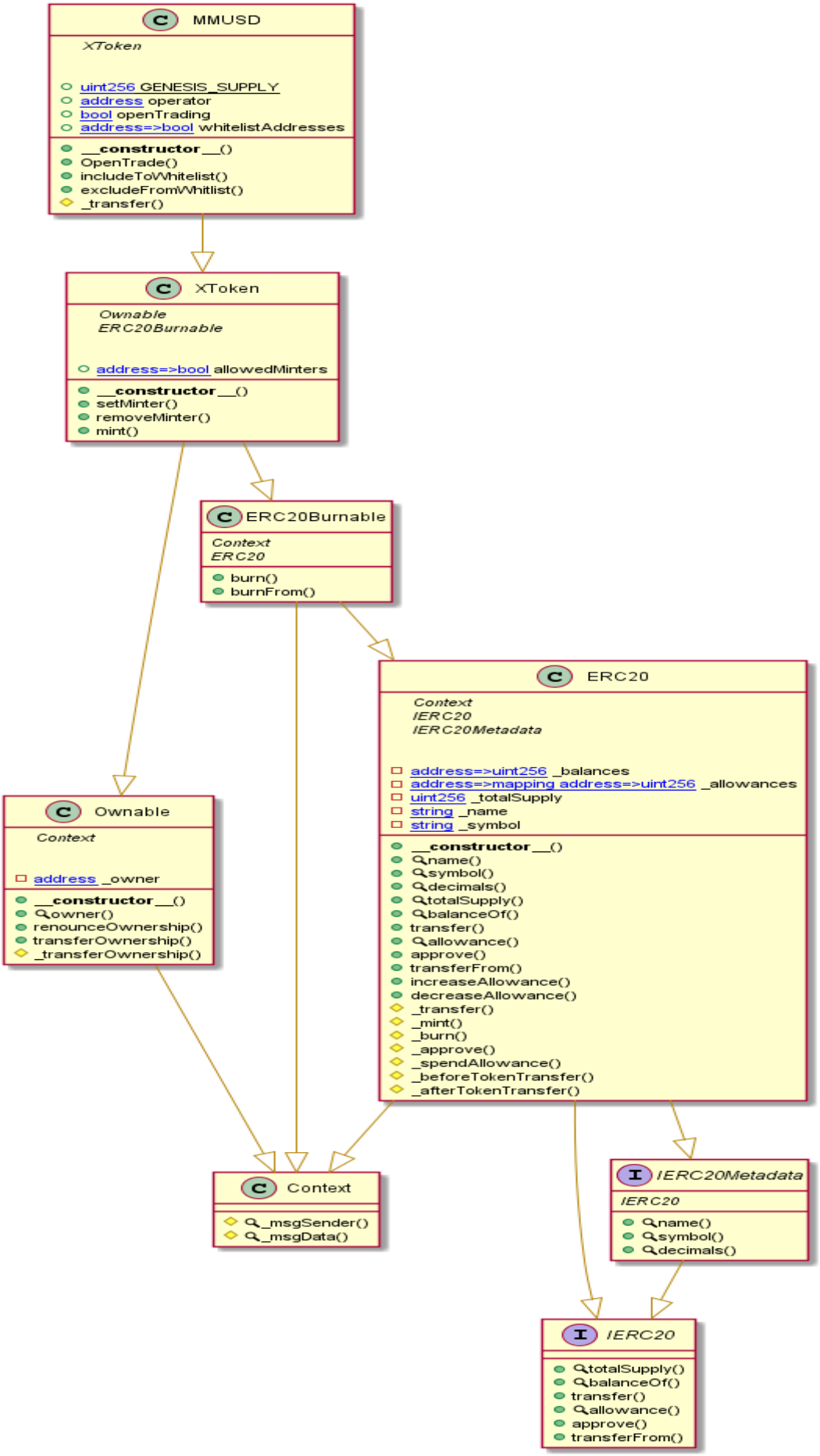
MMFX Diagram



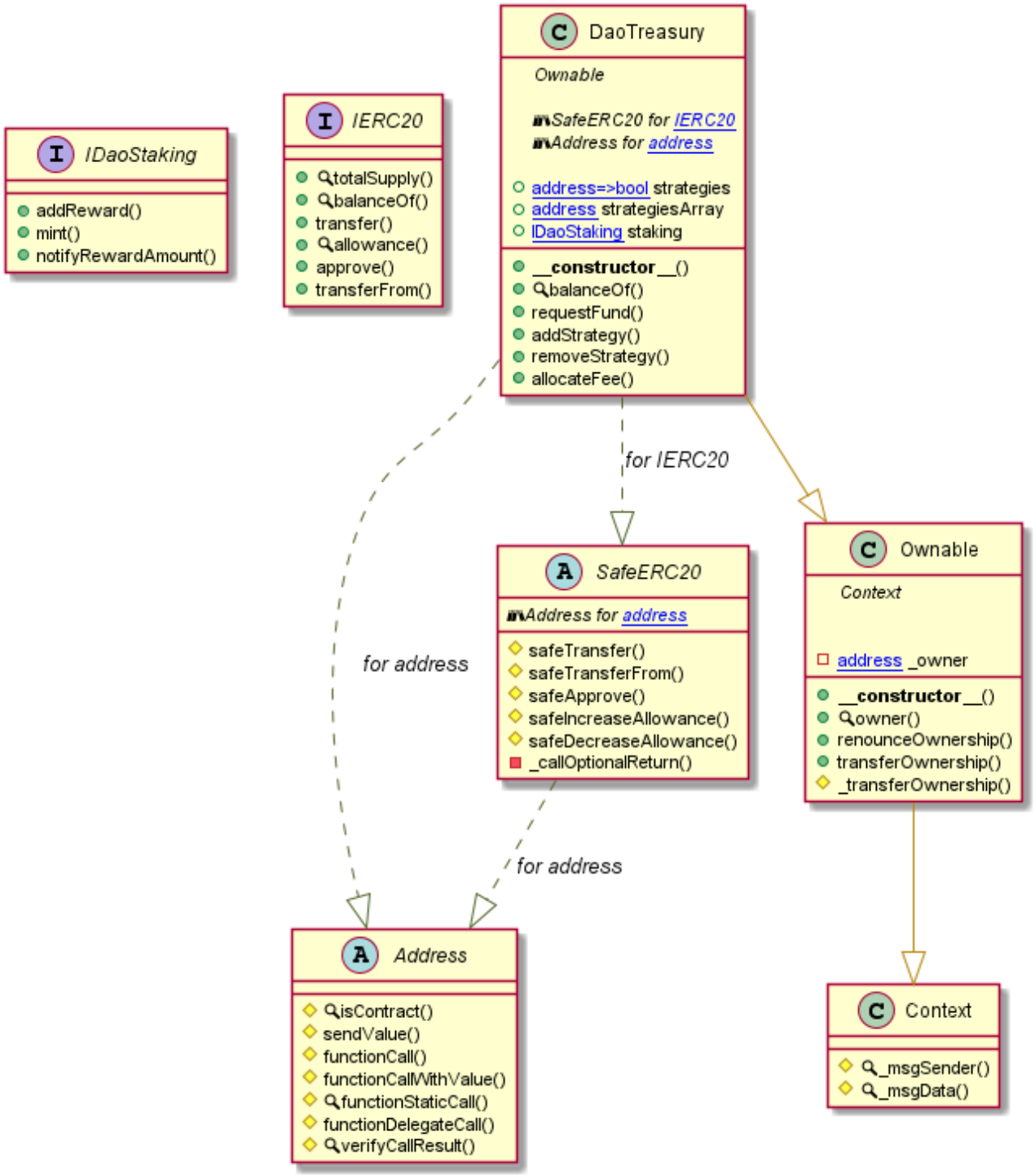
MMOX Diagram



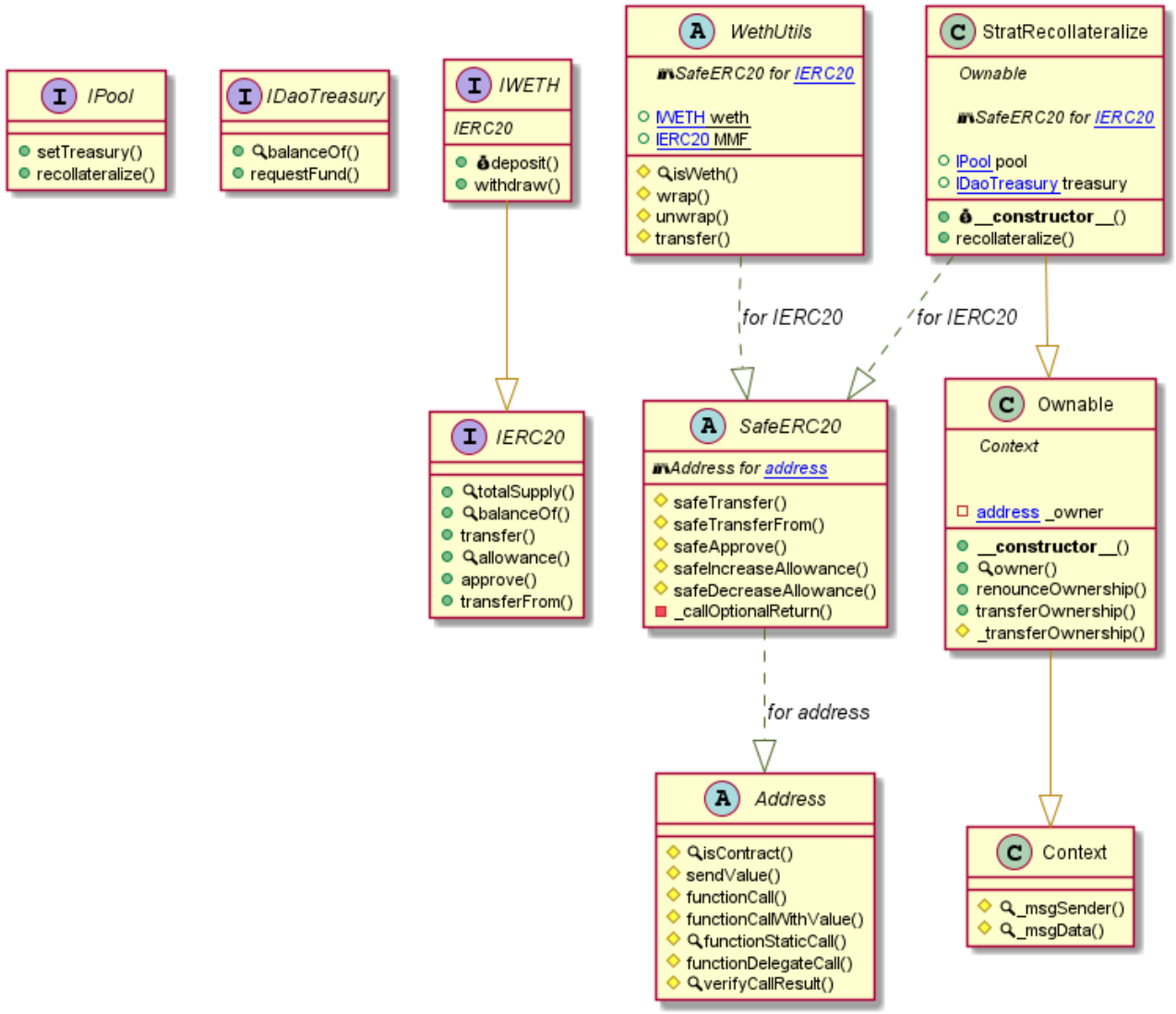
MMUSD Diagram



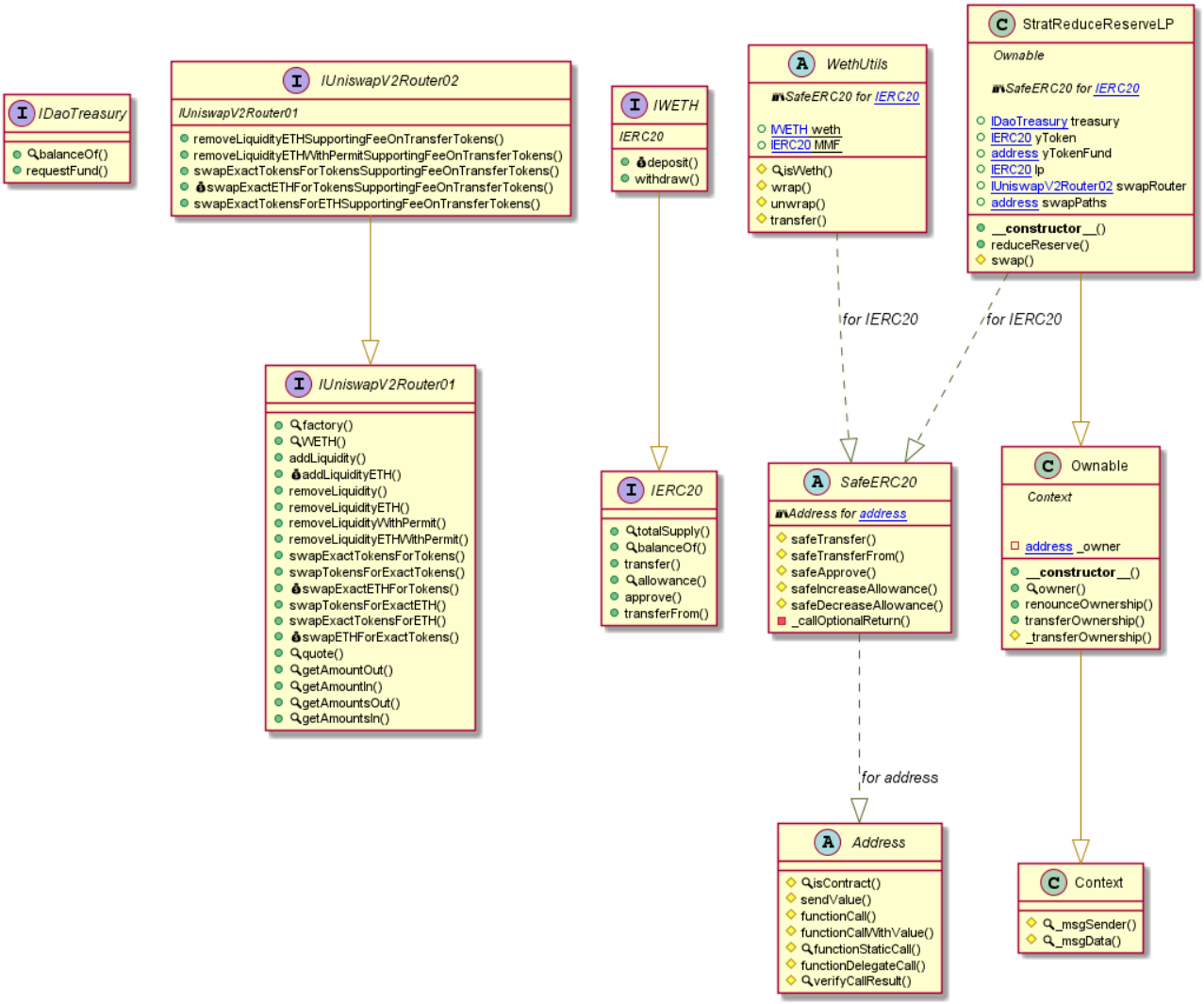
DaoTreasury Diagram



StratRecollateralize Diagram



StratReduceReserveLP Diagram



Slither Results Log

Slither log >> Pool.sol

```
INFO:Detectors:
Pool.setYTokenSlippage(uint256) (Pool.sol#1482-1486) uses literals with too many digits:
        - require(bool,string)(_slippage <= 300000,Pool::setYTokenSlippage: yTokenSlippage cannot be more than 30%) (Pool.sol#1
3)
Pool.slitherConstructorVariables() (Pool.sol#1055-1518) uses literals with too many digits:
        - yTokenSlippage = 200000 (Pool.sol#1102)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Pool.priceTarget (Pool.sol#1099) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Pool.sol#637-639)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Pool.sol#645-648)
name() should be declared external:
        - ERC20.name() (Pool.sol#688-690)
symbol() should be declared external:
        - ERC20.symbol() (Pool.sol#696-698)
decimals() should be declared external:
        - ERC20.decimals() (Pool.sol#713-715)
totalSupply() should be declared external:
        - ERC20.totalSupply() (Pool.sol#720-722)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (Pool.sol#727-729)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (Pool.sol#739-743)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (Pool.sol#762-766)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (Pool.sol#784-793)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (Pool.sol#807-811)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (Pool.sol#827-836)
refreshCollateralRatio() should be declared external:
        - Pool.refreshCollateralRatio() (Pool.sol#1232-1258)
toggle(bool,bool) should be declared external:
        - Pool.toggle(bool,bool) (Pool.sol#1403-1407)
setCollateralRatioOptions(uint256,uint256,uint256,uint256) should be declared external:
        - Pool.setCollateralRatioOptions(uint256,uint256,uint256,uint256) (Pool.sol#1414-1425)
toggleCollateralRatio(bool) should be declared external:
        - Pool.toggleCollateralRatio(bool) (Pool.sol#1429-1434)
setFees(uint256,uint256) should be declared external:
        - Pool.setFees(uint256,uint256) (Pool.sol#1439-1445)
```

```
setFees(uint256,uint256) should be declared external:
        - Pool.setFees(uint256,uint256) (Pool.sol#1439-1445)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Pool.sol analyzed (18 contracts with 75 detectors), 89 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chetan/gaza/mycontracts/OVB# slither SwapStrategyPOL.sol
```

Slither log >> SwapStrategyPOL.sol

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (SwapS
trategyPOL.sol#82) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,ui
nt256).amountBDesired (SwapStrategyPOL.sol#83)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
SwapStrategyPOL.slitherConstructorVariables() (SwapStrategyPOL.sol#686-804) uses literals with too many digits:
        - swapSlippage = 200000 (SwapStrategyPOL.sol#694)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (SwapStrategyPOL.sol#656-658)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (SwapStrategyPOL.sol#664-667)
lpBalance() should be declared external:
        - SwapStrategyPOL.lpBalance() (SwapStrategyPOL.sol#713-715)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:SwapStrategyPOL.sol analyzed (13 contracts with 75 detectors), 42 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Timelock.sol

```
INFO:Detectors:
Timelock.constructor(address,uint256).admin_ (Timelock.sol#23) lacks a zero-check on :
                - admin = admin_ (Timelock.sol#27)
Timelock.setPendingAdmin(address).pendingAdmin_ (Timelock.sol#47) lacks a zero-check on :
                - pendingAdmin = pendingAdmin_ (Timelock.sol#49)
Timelock.executeTransaction(address,uint256,string,bytes,uint256).target (Timelock.sol#87) lacks a zero-check on :
                - (success,returnData) = target.call{value: value}(callData) (Timelock.sol#111)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Timelock.executeTransaction(address,uint256,string,bytes,uint256) (Timelock.sol#86-117):
        External calls:
        - (success,returnData) = target.call{value: value}(callData) (Timelock.sol#111)
        Event emitted after the call(s):
        - ExecuteTransaction(txHash,target,value,signature,data,eta) (Timelock.sol#114)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

```
INFO:Detectors:
Low level call in Timelock.executeTransaction(address,uint256,string,bytes,uint256) (Timelock.sol#86-117):
        - (success,returnData) = target.call{value: value}(callData) (Timelock.sol#111)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
setDelay(uint256) should be declared external:
        - Timelock.setDelay(uint256) (Timelock.sol#31-38)
acceptAdmin() should be declared external:
        - Timelock.acceptAdmin() (Timelock.sol#40-45)
setPendingAdmin(address) should be declared external:
        - Timelock.setPendingAdmin(address) (Timelock.sol#47-52)
queueTransaction(address,uint256,string,bytes,uint256) should be declared external:
        - Timelock.queueTransaction(address,uint256,string,bytes,uint256) (Timelock.sol#54-69)
cancelTransaction(address,uint256,string,bytes,uint256) should be declared external:
        - Timelock.cancelTransaction(address,uint256,string,bytes,uint256) (Timelock.sol#71-84)
executeTransaction(address,uint256,string,bytes,uint256) should be declared external:
        - Timelock.executeTransaction(address,uint256,string,bytes,uint256) (Timelock.sol#86-117)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Timelock.sol analyzed (1 contracts with 75 detectors), 15 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DaoChef.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DaoChef.sol#594-596)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DaoChef.sol#602-605)
poolLength() should be declared external:
        - DaoChef.poolLength() (DaoChef.sol#662-664)
deposit(uint256,uint256,address) should be declared external:
        - DaoChef.deposit(uint256,uint256,address) (DaoChef.sol#713-734)
withdraw(uint256,uint256,address) should be declared external:
        - DaoChef.withdraw(uint256,uint256,address) (DaoChef.sol#740-761)
withdrawAndHarvest(uint256,uint256,address) should be declared external:
        - DaoChef.withdrawAndHarvest(uint256,uint256,address) (DaoChef.sol#794-824)
emergencyWithdraw(uint256,address) should be declared external:
        - DaoChef.emergencyWithdraw(uint256,address) (DaoChef.sol#829-843)
add(uint256,IERC20,IRewarder) should be declared external:
        - DaoChef.add(uint256,IERC20,IRewarder) (DaoChef.sol#868-881)
set(uint256,uint256,IRewarder,bool) should be declared external:
        - DaoChef.set(uint256,uint256,IRewarder,bool) (DaoChef.sol#888-901)
setRewardPerSecond(uint256) should be declared external:
        - DaoChef.setRewardPerSecond(uint256) (DaoChef.sol#905-910)
getSlots(address,uint256) should be declared external:
        - DaoChef.getSlots(address,uint256) (DaoChef.sol#930-941)
getTokenIds(address,uint256) should be declared external:
        - DaoChef.getTokenIds(address,uint256) (DaoChef.sol#943-954)
depositNFT(address,uint256,uint256,uint256) should be declared external:
        - DaoChef.depositNFT(address,uint256,uint256,uint256) (DaoChef.sol#956-977)
withdrawNFT(uint256,uint256) should be declared external:
        - DaoChef.withdrawNFT(uint256,uint256) (DaoChef.sol#979-991)
setNftController(address) should be declared external:
        - DaoChef.setNftController(address) (DaoChef.sol#993-996)
setNftBoostRate(uint256) should be declared external:
        - DaoChef.setNftBoostRate(uint256) (DaoChef.sol#998-1002)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DaoChef.sol analyzed (11 contracts with 75 detectors), 75 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DaoStaking.sol

```
INFO:Detectors:
DaoStaking (DaoStaking.sol#780-1272) does not implement functions:
        - Context._msgData() (DaoStaking.sol#674-676)
        - Context._msgSender() (DaoStaking.sol#670-672)
        - DaoStaking._notifyReward(address,uint256) (DaoStaking.sol#1178-1189)
        - Ownable._transferOwnership(address) (DaoStaking.sol#729-733)
        - DaoStaking.earnedBalances(address) (DaoStaking.sol#956-970)
        - DaoStaking.emergencyWithdraw() (DaoStaking.sol#1137-1151)
        - DaoStaking.getReward() (DaoStaking.sol#1118-1134)
        - DaoStaking.lockedBalances(address) (DaoStaking.sol#973-998)
        - DaoStaking.mint(address,uint256) (DaoStaking.sol#1050-1067)
        - DaoStaking.notifyRewardAmount(address,uint256) (DaoStaking.sol#1191-1203)
        - Ownable.owner() (DaoStaking.sol#693-695)
        - DaoStaking.receive() (DaoStaking.sol#1258)
        - DaoStaking.recoverERC20(address,uint256) (DaoStaking.sol#1206-1211)
        - Ownable.renounceOwnership() (DaoStaking.sol#712-714)
        - DaoStaking.setTeamRewardPercent(uint256) (DaoStaking.sol#1221-1227)
        - DaoStaking.setTeamWalletAddress(address) (DaoStaking.sol#1214-1218)
        - DaoStaking.stake(uint256,bool) (DaoStaking.sol#1025-1045)
        - DaoStaking.totalBalance(address) (DaoStaking.sol#937-939)
        - Ownable.transferOwnership(address) (DaoStaking.sol#720-723)
        - DaoStaking.unlockedBalance(address) (DaoStaking.sol#942-952)
        - DaoStaking.withdraw(uint256) (DaoStaking.sol#1072-1115)
        - DaoStaking.withdrawExpiredLocks() (DaoStaking.sol#1154-1174)
        - DaoStaking.withdrawableBalance(address) (DaoStaking.sol#1001-1019)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
```

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DaoStaking.sol#712-714)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DaoStaking.sol#720-723)
addReward(address,address) should be declared external:
        - DaoStaking.addReward(address,address) (DaoStaking.sol#868-876)
withdrawableBalance(address) should be declared external:
        - DaoStaking.withdrawableBalance(address) (DaoStaking.sol#1001-1019)
withdraw(uint256) should be declared external:
        - DaoStaking.withdraw(uint256) (DaoStaking.sol#1072-1115)
getReward() should be declared external:
        - DaoStaking.getReward() (DaoStaking.sol#1118-1134)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DaoStaking.sol analyzed (13 contracts with 75 detectors), 77 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DaoZapMMSwap.sol

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (DaoZa
pMMSwap.sol#78) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint2
56).amountBDesired (DaoZapMMSwap.sol#79)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DaoZapMMSwap.sol#632-634)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DaoZapMMSwap.sol#640-643)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DaoZapMMSwap.sol analyzed (12 contracts with 75 detectors), 46 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> NFTController.sol

```
INFO:Detectors:
Pragma version0.8.4 (NFTController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (NFTController.sol#60-65):
        - (success) = recipient.call{value: amount}() (NFTController.sol#63)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (NFTController.sol#128-139):
        - (success,returndata) = target.call{value: value}(data) (NFTController.sol#137)
Low level call in Address.functionStaticCall(address,bytes,string) (NFTController.sol#157-166):
        - (success,returndata) = target.staticcall(data) (NFTController.sol#164)
Low level call in Address.functionDelegateCall(address,bytes,string) (NFTController.sol#184-193):
        - (success,returndata) = target.delegatecall(data) (NFTController.sol#191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (NFTController.sol#343-345)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (NFTController.sol#352-355)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (NFTController.sol#357-361)
initialize(address) should be declared external:
        - NFTController.initialize(address) (NFTController.sol#372-374)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:NFTController.sol analyzed (6 contracts with 75 detectors), 24 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> NFTControllerProxy.sol

```
INFO:Detectors:
Pragma version0.8.4 (NFTControllerProxy.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/
0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (NFTControllerProxy.sol#57-62):
        - (success) = recipient.call{value: amount}() (NFTControllerProxy.sol#60)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (NFTControllerProxy.sol#125-136):
        - (success,returndata) = target.call{value: value}(data) (NFTControllerProxy.sol#134)
Low level call in Address.functionStaticCall(address,bytes,string) (NFTControllerProxy.sol#154-163):
        - (success,returndata) = target.staticcall(data) (NFTControllerProxy.sol#161)
Low level call in Address.functionDelegateCall(address,bytes,string) (NFTControllerProxy.sol#181-190):
        - (success,returndata) = target.delegatecall(data) (NFTControllerProxy.sol#188)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Slither:NFTControllerProxy.sol analyzed (9 contracts with 75 detectors), 36 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DaoFund.sol

```
INFO:Detectors:
Fund.vestedBalance() (DaoFund.sol#565-576) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= _start (DaoFund.sol#569)
        - block.timestamp > _start + _duration (DaoFund.sol#572)
Fund.transfer(address,uint256) (DaoFund.sol#583-590) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(amount <= claimable(),Fund::transfer: > vestedAmount) (DaoFund.sol#586)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
```

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (DaoFund.sol#132-137):
        - (success) = recipient.call{value: amount}() (DaoFund.sol#135)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (DaoFund.sol#200-211):
        - (success,returndata) = target.call{value: value}(data) (DaoFund.sol#209)
Low level call in Address.functionStaticCall(address,bytes,string) (DaoFund.sol#229-238):
        - (success,returndata) = target.staticcall(data) (DaoFund.sol#236)
Low level call in Address.functionDelegateCall(address,bytes,string) (DaoFund.sol#256-265):
        - (success,returndata) = target.delegatecall(data) (DaoFund.sol#263)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter Fund.initialize(address)._yToken (DaoFund.sol#548) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DaoFund.sol#422-424)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DaoFund.sol#430-433)
currentBalance() should be declared external:
        - Fund.currentBalance() (DaoFund.sol#561-563)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DaoFund.sol analyzed (8 contracts with 75 detectors), 27 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DevFund.sol

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (DevFund.sol#132-137):
        - (success) = recipient.call{value: amount}() (DevFund.sol#135)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (DevFund.sol#200-211):
        - (success,returndata) = target.call{value: value}(data) (DevFund.sol#209)
Low level call in Address.functionStaticCall(address,bytes,string) (DevFund.sol#229-238):
        - (success,returndata) = target.staticcall(data) (DevFund.sol#236)
Low level call in Address.functionDelegateCall(address,bytes,string) (DevFund.sol#256-265):
        - (success,returndata) = target.delegatecall(data) (DevFund.sol#263)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter Fund.initialize(address)._yToken (DevFund.sol#548) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DevFund.sol#422-424)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DevFund.sol#430-433)
currentBalance() should be declared external:
        - Fund.currentBalance() (DevFund.sol#561-563)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DevFund.sol analyzed (8 contracts with 75 detectors), 27 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Fund.sol

```
INFO:Detectors:
Parameter Fund.initialize(address)._yToken (Fund.sol#548) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Fund (Fund.sol#541-592) does not implement functions:
        - Fund.allocation() (Fund.sol#555)
        - Fund.vestingDuration() (Fund.sol#559)
        - Fund.vestingStart() (Fund.sol#557)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Fund.sol#422-424)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Fund.sol#430-433)
currentBalance() should be declared external:
        - Fund.currentBalance() (Fund.sol#561-563)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Fund.sol analyzed (7 contracts with 75 detectors), 28 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Reserve.sol

```
INFO:Detectors:
Reserve.setRewarder(address)._rewarder (Reserve.sol#557) lacks a zero-check on :
                - rewarder = _rewarder (Reserve.sol#559)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Address.verifyCallResult(bool,bytes,string) (Reserve.sol#272-292) uses assembly
        - INLINE ASM (Reserve.sol#284-287)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Reserve.setPool(address) (Reserve.sol#563-567) compares to a boolean constant:
        -require(bool,string)(allowedPools[_pool] == false,Reserve::setPool: ALREADY_ALLOWED) (Reserve.sol#564)
Reserve.removePool(address) (Reserve.sol#569-573) compares to a boolean constant:
        -require(bool,string)(allowedPools[_pool] == true,Reserve::removePool: NOT_ALLOWED) (Reserve.sol#570)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
```

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Reserve.sol#421-423)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Reserve.sol#429-432)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Reserve.sol analyzed (7 contracts with 75 detectors), 32 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> TreasuryFund.sol

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (TreasuryFund.sol#132-137):
        - (success) = recipient.call{value: amount}() (TreasuryFund.sol#135)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (TreasuryFund.sol#200-211):
        - (success,returndata) = target.call{value: value}(data) (TreasuryFund.sol#209)
Low level call in Address.functionStaticCall(address,bytes,string) (TreasuryFund.sol#229-238):
        - (success,returndata) = target.staticcall(data) (TreasuryFund.sol#236)
Low level call in Address.functionDelegateCall(address,bytes,string) (TreasuryFund.sol#256-265):
        - (success,returndata) = target.delegatecall(data) (TreasuryFund.sol#263)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter Fund.initialize(address)._yToken (TreasuryFund.sol#548) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (TreasuryFund.sol#422-424)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (TreasuryFund.sol#430-433)
currentBalance() should be declared external:
        - Fund.currentBalance() (TreasuryFund.sol#561-563)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:TreasuryFund.sol analyzed (8 contracts with 75 detectors), 27 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MockERC20.sol

```
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (MockERC20.sol#132-134)
symbol() should be declared external:
        - ERC20.symbol() (MockERC20.sol#140-142)
decimals() should be declared external:
        - ERC20.decimals() (MockERC20.sol#157-159)
        - MockERC20.decimals() (MockERC20.sol#470-472)
totalSupply() should be declared external:
        - ERC20.totalSupply() (MockERC20.sol#164-166)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (MockERC20.sol#171-173)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (MockERC20.sol#183-187)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MockERC20.sol#206-210)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MockERC20.sol#228-237)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MockERC20.sol#251-255)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (MockERC20.sol#271-280)
mint(uint256) should be declared external:
        - MockERC20.mint(uint256) (MockERC20.sol#466-468)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MockERC20.sol analyzed (5 contracts with 75 detectors), 18 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MockTreasury.sol

```
INFO:Detectors:
Pragma version0.8.4 (MockTreasury.sol#1) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Parameter MockTreasury.mock(uint256,uint256,uint256,uint256)._maxMmfxSupply (MockTreasury.sol#10) is not in mixedCase
Parameter MockTreasury.mock(uint256,uint256,uint256,uint256)._cr (MockTreasury.sol#11) is not in mixedCase
Parameter MockTreasury.mock(uint256,uint256,uint256,uint256)._mintingFee (MockTreasury.sol#12) is not in mixedCase
Parameter MockTreasury.mock(uint256,uint256,uint256,uint256)._redeemFee (MockTreasury.sol#13) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
mock(uint256,uint256,uint256,uint256) should be declared external:
        - MockTreasury.mock(uint256,uint256,uint256,uint256) (MockTreasury.sol#9-19)
info() should be declared external:
        - MockTreasury.info() (MockTreasury.sol#21-31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MockTreasury.sol analyzed (1 contracts with 75 detectors), 8 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MasterOracle.sol

```
INFO:Detectors:
Context._msgData() (MasterOracle.sol#17-19) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.4 (MasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Variable MasterOracle.constructor(address,address,address,address)._oracleXToken (MasterOracle.sol#92) is too similar to Master
Oracle.constructor(address,address,address,address)._oracleYToken (MasterOracle.sol#93)
Variable MasterOracle.oracleXToken (MasterOracle.sol#83) is too similar to MasterOracle.oracleYToken (MasterOracle.sol#84)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (MasterOracle.sol#56-58)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MasterOracle.sol#64-67)
getXTokenPrice() should be declared external:
        - MasterOracle.getXTokenPrice() (MasterOracle.sol#105-107)
getYTokenPrice() should be declared external:
        - MasterOracle.getYTokenPrice() (MasterOracle.sol#109-111)
getXTokenTWAP() should be declared external:
        - MasterOracle.getXTokenTWAP() (MasterOracle.sol#113-115)
getYTokenTWAP() should be declared external:
        - MasterOracle.getYTokenTWAP() (MasterOracle.sol#117-119)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MasterOracle.sol analyzed (4 contracts with 75 detectors), 11 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> UniswapPairOracle.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (UniswapPairOracle.sol#287-289)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (UniswapPairOracle.sol#295-298)
name() should be declared external:
        - ERC20.name() (UniswapPairOracle.sol#338-340)
symbol() should be declared external:
        - ERC20.symbol() (UniswapPairOracle.sol#346-348)
```

```
symbol() should be declared external:
        - ERC20.symbol() (UniswapPairOracle.sol#346-348)
decimals() should be declared external:
        - ERC20.decimals() (UniswapPairOracle.sol#363-365)
totalSupply() should be declared external:
        - ERC20.totalSupply() (UniswapPairOracle.sol#370-372)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (UniswapPairOracle.sol#377-379)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (UniswapPairOracle.sol#389-393)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (UniswapPairOracle.sol#412-416)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (UniswapPairOracle.sol#434-443)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (UniswapPairOracle.sol#457-461)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (UniswapPairOracle.sol#477-486)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:UniswapPairOracle.sol analyzed (10 contracts with 75 detectors), 48 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> XToken.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (XToken.sol#49-51)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (XToken.sol#57-60)
name() should be declared external:
        - ERC20.name() (XToken.sol#191-193)
symbol() should be declared external:
        - ERC20.symbol() (XToken.sol#199-201)
decimals() should be declared external:
        - ERC20.decimals() (XToken.sol#216-218)
totalSupply() should be declared external:
        - ERC20.totalSupply() (XToken.sol#223-225)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (XToken.sol#230-232)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (XToken.sol#242-246)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (XToken.sol#265-269)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (XToken.sol#287-296)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (XToken.sol#310-314)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (XToken.sol#330-339)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (XToken.sol#520-522)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (XToken.sol#535-538)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:XToken.sol analyzed (7 contracts with 75 detectors), 25 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> YToken.sol

```
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (YToken.sol#133-135)
symbol() should be declared external:
        - ERC20.symbol() (YToken.sol#141-143)
decimals() should be declared external:
        - ERC20.decimals() (YToken.sol#158-160)
totalSupply() should be declared external:
        - ERC20.totalSupply() (YToken.sol#165-167)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (YToken.sol#172-174)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (YToken.sol#184-188)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (YToken.sol#207-211)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (YToken.sol#229-238)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (YToken.sol#252-256)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (YToken.sol#272-281)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (YToken.sol#462-464)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (YToken.sol#477-480)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:YToken.sol analyzed (6 contracts with 75 detectors), 18 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> LOVE.sol

```
INFO:Detectors:
Pragma version0.8.4 (LOVE.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function LOVE.OpenTrade() (LOVE.sol#513-517) is not in mixedCase
Parameter LOVE.includeToWhitelist(address)._address (LOVE.sol#519) is not in mixedCase
Parameter LOVE.excludeFromWhitlist(address)._address (LOVE.sol#526) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (LOVE.sol#133-135)
symbol() should be declared external:
        - ERC20.symbol() (LOVE.sol#141-143)
decimals() should be declared external:
        - ERC20.decimals() (LOVE.sol#158-160)
totalSupply() should be declared external:
        - ERC20.totalSupply() (LOVE.sol#165-167)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (LOVE.sol#172-174)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (LOVE.sol#184-188)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (LOVE.sol#207-211)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (LOVE.sol#229-238)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (LOVE.sol#252-256)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (LOVE.sol#272-281)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (LOVE.sol#462-464)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (LOVE.sol#477-480)
includeToWhitelist(address) should be declared external:
        - LOVE.includeToWhitelist(address) (LOVE.sol#519-524)
excludeFromWhitlist(address) should be declared external:
        - LOVE.excludeFromWhitlist(address) (LOVE.sol#526-531)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:LOVE.sol analyzed (7 contracts with 75 detectors), 24 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MMFX.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (MMFX.sol#49-51)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MMFX.sol#57-60)
name() should be declared external:
        - ERC20.name() (MMFX.sol#193-195)
symbol() should be declared external:
        - ERC20.symbol() (MMFX.sol#201-203)
decimals() should be declared external:
        - ERC20.decimals() (MMFX.sol#218-220)
totalSupply() should be declared external:
        - ERC20.totalSupply() (MMFX.sol#225-227)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (MMFX.sol#232-234)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (MMFX.sol#244-248)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MMFX.sol#267-271)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MMFX.sol#289-298)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MMFX.sol#312-316)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (MMFX.sol#332-341)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (MMFX.sol#522-524)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (MMFX.sol#537-540)
includeToWhitelist(address) should be declared external:
        - MMFX.includeToWhitelist(address) (MMFX.sol#601-606)
excludeFromWhitlist(address) should be declared external:
        - MMFX.excludeFromWhitlist(address) (MMFX.sol#608-613)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MMFX.sol analyzed (8 contracts with 75 detectors), 32 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MMOX.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (MMOX.sol#49-51)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MMOX.sol#57-60)
name() should be declared external:
        - ERC20.name() (MMOX.sol#193-195)
symbol() should be declared external:
        - ERC20.symbol() (MMOX.sol#201-203)
decimals() should be declared external:
        - ERC20.decimals() (MMOX.sol#218-220)
totalSupply() should be declared external:
        - ERC20.totalSupply() (MMOX.sol#225-227)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (MMOX.sol#232-234)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (MMOX.sol#244-248)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MMOX.sol#267-271)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MMOX.sol#289-298)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MMOX.sol#312-316)
```

```
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MMOX.sol#267-271)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MMOX.sol#289-298)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MMOX.sol#312-316)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (MMOX.sol#332-341)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (MMOX.sol#522-524)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (MMOX.sol#537-540)
includeToWhitelist(address) should be declared external:
        - MMOX.includeToWhitelist(address) (MMOX.sol#601-606)
excludeFromWhitlist(address) should be declared external:
        - MMOX.excludeFromWhitlist(address) (MMOX.sol#608-613)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MMOX.sol analyzed (8 contracts with 75 detectors), 32 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> MMUSD.sol

```
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (MMUSD.sol#49-51)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (MMUSD.sol#57-60)
name() should be declared external:
        - ERC20.name() (MMUSD.sol#193-195)
symbol() should be declared external:
        - ERC20.symbol() (MMUSD.sol#201-203)
decimals() should be declared external:
        - ERC20.decimals() (MMUSD.sol#218-220)
totalSupply() should be declared external:
        - ERC20.totalSupply() (MMUSD.sol#225-227)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (MMUSD.sol#232-234)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (MMUSD.sol#244-248)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (MMUSD.sol#267-271)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (MMUSD.sol#289-298)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (MMUSD.sol#312-316)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (MMUSD.sol#332-341)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (MMUSD.sol#522-524)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (MMUSD.sol#537-540)
includeToWhitelist(address) should be declared external:
        - MMUSD.includeToWhitelist(address) (MMUSD.sol#601-606)
excludeFromWhitlist(address) should be declared external:
        - MMUSD.excludeFromWhitlist(address) (MMUSD.sol#608-613)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MMUSD.sol analyzed (8 contracts with 75 detectors), 32 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> DaoTreasury.sol

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (DaoTreasury.sol#140-145):
        - (success) = recipient.call{value: amount}() (DaoTreasury.sol#143)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (DaoTreasury.sol#208-219):
        - (success,returndata) = target.call{value: value}(data) (DaoTreasury.sol#217)
Low level call in Address.functionStaticCall(address,bytes,string) (DaoTreasury.sol#237-246):
        - (success,returndata) = target.staticcall(data) (DaoTreasury.sol#244)
Low level call in Address.functionDelegateCall(address,bytes,string) (DaoTreasury.sol#264-273):
        - (success,returndata) = target.delegatecall(data) (DaoTreasury.sol#271)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter DaoTreasury.balanceOf(address)._token (DaoTreasury.sol#477) is not in mixedCase
Parameter DaoTreasury.requestFund(address,uint256)._token (DaoTreasury.sol#486) is not in mixedCase
Parameter DaoTreasury.requestFund(address,uint256)._amount (DaoTreasury.sol#486) is not in mixedCase
Parameter DaoTreasury.addStrategy(address)._strategy (DaoTreasury.sol#495) is not in mixedCase
Parameter DaoTreasury.removeStrategy(address)._strategy (DaoTreasury.sol#505) is not in mixedCase
Parameter DaoTreasury.allocateFee(address,uint256)._token (DaoTreasury.sol#522) is not in mixedCase
Parameter DaoTreasury.allocateFee(address,uint256)._amount (DaoTreasury.sol#522) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (DaoTreasury.sol#430-432)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (DaoTreasury.sol#438-441)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:DaoTreasury.sol analyzed (7 contracts with 75 detectors), 30 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> StratRecollateralize.sol

```
INFO:Detectors:
Reentrancy in StratRecollateralize.recollateralize(uint256) (StratRecollateralize.sol#501-512):
        External calls:
        - treasury.requestFund(address(WethUtils.MMF),_amount) (StratRecollateralize.sol#505)
        - WethUtils.MMF.safeTransferFrom(address(treasury),address(this),_amount) (StratRecollateralize.sol#506)
        - WethUtils.MMF.safeIncreaseAllowance(address(pool),_amount) (StratRecollateralize.sol#508)
        - pool.recollateralize(_amount) (StratRecollateralize.sol#509)
        Event emitted after the call(s):
        - Recollateralized(_amount) (StratRecollateralize.sol#511)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (StratRecollateralize.sol#147-152):
        - (success) = recipient.call{value: amount}() (StratRecollateralize.sol#150)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (StratRecollateralize.sol#215-226):
        - (success,returndata) = target.call{value: value}(data) (StratRecollateralize.sol#224)
Low level call in Address.functionStaticCall(address,bytes,string) (StratRecollateralize.sol#244-253):
        - (success,returndata) = target.staticcall(data) (StratRecollateralize.sol#251)
Low level call in Address.functionDelegateCall(address,bytes,string) (StratRecollateralize.sol#271-280):
        - (success,returndata) = target.delegatecall(data) (StratRecollateralize.sol#278)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Constant WethUtils.weth (StratRecollateralize.sol#397) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter StratRecollateralize.recollateralize(uint256)._amount (StratRecollateralize.sol#501) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StratRecollateralize.sol#464-466)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StratRecollateralize.sol#472-475)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StratRecollateralize.sol analyzed (10 contracts with 75 detectors), 27 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> StratReduceReserveLP.sol

```
INFO:Detectors:
StratReduceReserveLP.constructor(IERC20,address,IERC20,IUniswapV2Router02,address[],IDaoTreasury)._yTokenFund (StratReduceReser
veLP.sol#632) lacks a zero-check on :
                - yTokenFund = _yTokenFund (StratReduceReserveLP.sol#640)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in StratReduceReserveLP.reduceReserve(uint256,uint256) (StratReduceReserveLP.sol#650-669):
        External calls:
        - treasury.requestFund(address(lp),_amount) (StratReduceReserveLP.sol#654)
        - lp.safeTransferFrom(address(treasury),address(this),_amount) (StratReduceReserveLP.sol#655)
        - lp.safeIncreaseAllowance(address(swapRouter),_amount) (StratReduceReserveLP.sol#658)
        - swapRouter.removeLiquidity(address(yToken),address(WethUtils.MMF),_amount,0,0,address(this),block.timestamp) (StratRe
duceReserveLP.sol#659)
        - swap(WethUtils.MMF.balanceOf(address(this)),_minYTokenAmount) (StratReduceReserveLP.sol#662)
                - WethUtils.MMF.safeIncreaseAllowance(address(swapRouter),_wethToSwap) (StratReduceReserveLP.sol#675)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StratReduceReserveLP.sol#516
)
                - swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(_wethToSwap,_minYTokenOut,swapPaths,address(
this),block.timestamp) (StratReduceReserveLP.sol#676)
```

```
INFO:Detectors:
Function IUniswapV2Router01.WETH() (StratReduceReserveLP.sol#13) is not in mixedCase
Constant WethUtils.weth (StratReduceReserveLP.sol#528) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter StratReduceReserveLP.reduceReserve(uint256,uint256)._amount (StratReduceReserveLP.sol#650) is not in mixedCase
Parameter StratReduceReserveLP.reduceReserve(uint256,uint256)._minYTokenAmount (StratReduceReserveLP.sol#650) is not in mixedCa
se
Parameter StratReduceReserveLP.swap(uint256,uint256)._wethToSwap (StratReduceReserveLP.sol#674) is not in mixedCase
Parameter StratReduceReserveLP.swap(uint256,uint256)._minYTokenOut (StratReduceReserveLP.sol#674) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Strat
ReduceReserveLP.sol#18) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,addre
ss,uint256).amountBDesired (StratReduceReserveLP.sol#19)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (StratReduceReserveLP.sol#595-597)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (StratReduceReserveLP.sol#603-606)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:StratReduceReserveLP.sol analyzed (11 contracts with 75 detectors), 33 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Solidity Static Analysis

Pool.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Pool.redeem(uint256,uint256,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1290:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 1256:33:

Gas & Economy

Gas costs:

Gas requirement of function Pool.redeem is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1290:4:

Gas costs:

Gas requirement of function Pool.collect is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1330:4:

Miscellaneous

Constant/View/Pure functions:

Pool.transferToTreasury(uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1497:4:

Similar variable names:

Pool.redeem(uint256,uint256,uint256) : Variables have very similar names "unclaimedXToken" and "unclaimedYToken". Note: Modifiers are currently not considered by this static analysis.
Pos: 1315:12:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1201:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 1175:21:

SwapStrategyPOL.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in SwapStrategyPOL.addLiquidity(uint256,uint256,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 764:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 783:16:

Gas & Economy

Gas costs:

Gas requirement of function SwapStrategyPOL.yToken is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 689:4:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 30:4:

Miscellaneous

Constant/View/Pure functions:

WethUtils.transfer(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 608:4:

Similar variable names:

SwapStrategyPOL.addLiquidity(uint256,uint256,uint256) : Variables have very similar names "_amountA" and "_amountB". Note: Modifiers are currently not considered by this static analysis.
Pos: 775:13:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 795:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 772:34:

Timelock.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Timelock.executeTransaction(address,uint256,string,bytes,uint256): Could potentially lead to re-entrancy vulnerability.

more

Pos: 86:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 120:15:

Gas & Economy

Gas costs:

Gas requirement of function Timelock.queueTransaction is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 54:4:

Miscellaneous

Similar variable names:

Timelock.(address,uint256) : Variables have very similar names "MINIMUM_DELAY" and "MAXIMUM_DELAY".

Pos: 24:26:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 112:8:

DaoChef.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in DaoChef.withdrawNFT(uint256,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 979:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 879:72:

Gas & Economy

Gas costs:

Gas requirement of function DaoChef.pendingReward is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 670:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 999:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 677:35:

DaoStaking.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in DaoStaking.getReward(): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 1118:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 911:24:

Gas & Economy

Gas costs:

Gas requirement of function DaoStaking.lockDuration is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 817:4:

Delete dynamic array:

The "delete" operation when applied to a dynamically sized array in Solidity generates code to delete each of the elements contained. If the array is large, this operation can surpass the block gas limit and raise an OOG exception. Also nested dynamically sized objects can produce the same results.

more

Pos: 1092:24:

Email: audit@EtherAuthority.io

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 1245:8:

Miscellaneous

Constant/View/Pure functions:

DaoStaking.lockedBalances(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 973:4:

Similar variable names:

DaoStaking.getReward() : Variables have very similar names "rewardTokens" and "_rewardToken". Note: Modifiers are currently not considered by this static analysis.
Pos: 1124:37:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1223:8:

Delete from dynamic array:

Using "delete" on an array leaves a gap. The length of the array remains the same. If you want to remove the empty position you need to shift items manually and update the "length" property.
more
Pos: 1161:12:

DaoZapMMSwap.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in DaoZapMMSwap.swap(address,uint256,address): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 789:4:

Gas & Economy

Gas costs:

Gas requirement of function DaoZapMMSwap.zap is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 734:4:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 26:4:

Miscellaneous

Constant/View/Pure functions:

DaoZapMMSwap.approveToken(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 833:4:

Similar variable names:

DaoZapMMSwap.addZap(address,address,uint256) : Variables have very similar names "_token" and "_token0". Note: Modifiers are currently not considered by this static analysis.
Pos: 866:16:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 876:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 852:15:

NFTController.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 128:4:

Gas & Economy

Gas costs:

Gas requirement of function NFTController.getBoostRate is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 376:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 358:8:

NFTControllerProxy.sol

Security

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.

more

Pos: 304:8:

Gas & Economy

Gas costs:

Gas requirement of function TransparentUpgradeableProxy.upgradeToAndCall is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 632:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 647:8:

DaoFund.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 575:31:

Gas & Economy

Gas costs:

Gas requirement of function DaoFund.transfer is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 583:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 586:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 575:15:

DevFund.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 575:31:

Gas & Economy

Gas costs:

Gas requirement of function DevFund.transfer is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 583:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 586:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 575:15:

Fund.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 575:31:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 586:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 575:15:

Reserve.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in SafeERC20.safeDecreaseAllowance(contract IERC20,address,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 345:4:

Gas & Economy

Gas costs:

Gas requirement of function Reserve.transfer is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 575:4:

Miscellaneous

Constant/View/Pure functions:

Reserve.transfer(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 575:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 577:8:

TreasuryFund.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 575:31:

Gas & Economy

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 586:8:

Miscellaneous

Constant/View/Pure functions:

SafeERC20._callOptionalReturn(contract IERC20,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 365:4:

No return:

Fund.vestingDuration(): Defines a return type but never explicitly returns a value.
Pos: 559:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 586:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 575:15:

MockERC20.sol

Gas & Economy

Gas costs:

Gas requirement of function MockERC20.mint is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 466:4:

Miscellaneous

MockTreasury.sol

Gas & Economy

Gas costs:

Gas requirement of function MockTreasury.mock is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 9:4:

Gas costs:

Gas requirement of function MockTreasury.info is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 21:4:

MasterOracle.sol

Gas & Economy

Gas costs:

Gas requirement of function MasterOracle.getXTokenPrice is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 105:4:

Miscellaneous

Similar variable names:

MasterOracle.getYTokenTWAP() : Variables have very similar names "xToken" and "yToken". Note: Modifiers are currently not considered by this static analysis.
Pos: 118:33:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 98:8:

UniswapPairOracle.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 764:22:

Gas & Economy

Gas costs:

Gas requirement of function UniswapPairOracle.spot is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 745:4:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 108:4:

Miscellaneous

Constant/View/Pure functions:

UniswapPairOracle.currentCumulativePrices(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 768:4:

Similar variable names:

UniswapPairOracle.update() : Variables have very similar names "price0Cumulative" and "price1Cumulative". Note: Modifiers are currently not considered by this static analysis.
Pos: 702:12:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 751:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 756:21:

XToken.sol

Gas & Economy

Gas costs:

Gas requirement of function ERC20.decreaseAllowance is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 332:4:

Miscellaneous

Constant/View/Pure functions:

ERC20._afterTokenTransfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 509:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 570:8:

YToken.sol

Gas & Economy

Gas costs:

Gas requirement of function ERC20.decreaseAllowance is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 272:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 408:12:

LOVE.sol

Gas & Economy

Gas costs:

Gas requirement of function LOVE.burnFrom is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 477:4:

Miscellaneous

Constant/View/Pure functions:

LOVE._transfer(address,address,uint256) : Potentially should be constant/view/pure but is not.
more
Pos: 533:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 538:8:

MMFX.sol

Gas & Economy

Gas costs:

Gas requirement of function MMFX.mint is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 577:4:

Miscellaneous

Constant/View/Pure functions:

MMFX._transfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 615:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 616:8:

MMOX.sol

Gas & Economy

Gas costs:

Gas requirement of function MMOX.mint is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 577:4:

Miscellaneous

Constant/View/Pure functions:

MMOX._transfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 615:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 620:8:

MMUSD.sol

Gas & Economy

Gas costs:

Gas requirement of function MMUSD.mint is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 577:4:

Miscellaneous

Constant/View/Pure functions:

MMUSD._transfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 615:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 620:8:

DaoTreasury.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in DaoTreasury.allocateFee(address,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 522:4:

Gas & Economy

Gas costs:

Gas requirement of function DaoTreasury.balanceOf is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 477:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 509:8:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 524:8:

StratRecollateralize.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in StratRecollateralize.recollateralize(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 501:4:

Gas & Economy

Gas costs:

Gas requirement of function StratRecollateralize.pool is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 491:4:

Miscellaneous

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 503:8:

StratReduceReserveLP.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in StratReduceReserveLP.reduceReserve(uint256,uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 650:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 676:127:

Gas & Economy

Gas costs:

Gas requirement of function StratReduceReserveLP.reduceReserve is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 650:4:

Miscellaneous

Constant/View/Pure functions:

SafeERC20._callOptionalReturn(contract IERC20,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 511:4:

Constant/View/Pure functions:

WethUtils.transfer(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 546:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 652:8:

Solhint Linter

Pool.sol

```
Pool.sol:523:18: Error: Parse error: missing ';' at '{'
Pool.sol:831:18: Error: Parse error: missing ';' at '{'
Pool.sol:864:18: Error: Parse error: missing ';' at '{'
Pool.sol:913:18: Error: Parse error: missing ';' at '{'
Pool.sol:964:22: Error: Parse error: missing ';' at '{'
```

SwapStrategyPOL.sol

```
SwapStrategyPOL.sol:559:18: Error: Parse error: missing ';' at '{'
```

Timelock.sol

```
Timelock.sol:3:1: Error: Compiler version 0.8.4 does not satisfy the
r semver requirement
Timelock.sol:23:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
Timelock.sol:111:51: Error: Avoid using low level calls.
Timelock.sol:120:16: Error: Avoid to make time-based decisions in
your business logic
```

DaoChef.sol

```
DaoChef.sol:523:18: Error: Parse error: missing ';' at '{'
```

DaoStaking.sol

```
DaoStaking.sol:56:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:69:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:81:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:98:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:110:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:206:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:229:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:255:18: Error: Parse error: missing ';' at '{'
DaoStaking.sol:610:18: Error: Parse error: missing ';' at '{'
```

DaoZapMMSwap.sol

```
DaoZapMMSwap.sol:561:18: Error: Parse error: missing ';' at '{'
```

Fund.sol

```
Fund.sol:351:18: Error: Parse error: missing ';' at '{'
```

NFTController.sol

```
NFTController.sol:3:1: Error: Compiler version 0.8.4 does not satisfy
the r semver requirement
NFTController.sol:63:28: Error: Avoid using low level calls.
NFTController.sol:137:51: Error: Avoid using low level calls.
NFTController.sol:191:51: Error: Avoid using low level calls.
NFTController.sol:213:17: Error: Avoid using inline assembly. It is
acceptable only in rare cases
NFTController.sol:335:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
NFTController.sol:335:20: Error: Code contains empty blocks
NFTController.sol:369:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
NFTController.sol:369:19: Error: Code contains empty blocks
```

NFTControllerProxy.sol

```
NFTControllerProxy.sol:3:1: Error: Compiler version 0.8.4 does not
satisfy the r semver requirement
NFTControllerProxy.sol:60:28: Error: Avoid using low level calls.
NFTControllerProxy.sol:134:51: Error: Avoid using low level calls.
NFTControllerProxy.sol:188:51: Error: Avoid using low level calls.
NFTControllerProxy.sol:210:17: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:262:9: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:272:9: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:282:9: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:292:9: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:304:9: Error: Avoid using inline assembly. It
is acceptable only in rare cases
NFTControllerProxy.sol:366:49: Error: Code contains empty blocks
NFTControllerProxy.sol:542:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
NFTControllerProxy.sol:559:5: Error: Explicitly mark visibility in
```

```
function (Set ignoreConstructors to true if using solidity >=0.7.0)
NFTControllerProxy.sol:655:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
NFTControllerProxy.sol:655:114: Error: Code contains empty blocks
```

DaoFund.sol

```
DaoFund.sol:351:18: Error: Parse error: missing ';' at '{'
```

DevFund.sol

```
DevFund.sol:351:18: Error: Parse error: missing ';' at '{'
```

Reserve.sol

```
Reserve.sol:350:18: Error: Parse error: missing ';' at '{'
```

MockTreasury.sol

```
MockTreasury.sol:1:1: Error: Compiler version 0.8.4 does not satisfy
the r semver requirement
```

MockERC20.sol

```
MockERC20.sol:275:18: Error: Parse error: missing ';' at '{'
MockERC20.sol:308:18: Error: Parse error: missing ';' at '{'
MockERC20.sol:357:18: Error: Parse error: missing ';' at '{'
MockERC20.sol:408:22: Error: Parse error: missing ';' at '{'
```

TreasuryFund.sol

```
TreasuryFund.sol:351:18: Error: Parse error: missing ';' at '{'
```

MasterOracle.sol

```
MasterOracle.sol:3:1: Error: Compiler version 0.8.4 does not satisfy
the r semver requirement
MasterOracle.sol:30:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
MasterOracle.sol:89:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
```

UniswapPairOracle.sol

```
UniswapPairOracle.sol:481:18: Error: Parse error: missing ';' at '{'
UniswapPairOracle.sol:514:18: Error: Parse error: missing ';' at '{'
UniswapPairOracle.sol:563:18: Error: Parse error: missing ';' at '{'
UniswapPairOracle.sol:614:22: Error: Parse error: missing ';' at '{'
UniswapPairOracle.sol:706:18: Error: Parse error: missing ';' at '{'
UniswapPairOracle.sol:785:18: Error: Parse error: missing ';' at '{'
```

XToken.sol

```
XToken.sol:336:18: Error: Parse error: missing ';' at '{'
XToken.sol:369:18: Error: Parse error: missing ';' at '{'
XToken.sol:418:18: Error: Parse error: missing ';' at '{'
XToken.sol:469:22: Error: Parse error: missing ';' at '{'
```

YToken.sol

```
YToken.sol:276:18: Error: Parse error: missing ';' at '{'
YToken.sol:309:18: Error: Parse error: missing ';' at '{'
YToken.sol:358:18: Error: Parse error: missing ';' at '{'
YToken.sol:409:22: Error: Parse error: missing ';' at '{'
```

LOVE.sol

```
LOVE.sol:276:18: Error: Parse error: missing ';' at '{'
LOVE.sol:309:18: Error: Parse error: missing ';' at '{'
LOVE.sol:358:18: Error: Parse error: missing ';' at '{'
LOVE.sol:409:22: Error: Parse error: missing ';' at '{'
```

MMFX.sol

```
MMFX.sol:336:18: Error: Parse error: missing ';' at '{'
MMFX.sol:369:18: Error: Parse error: missing ';' at '{'
MMFX.sol:418:18: Error: Parse error: missing ';' at '{'
MMFX.sol:469:22: Error: Parse error: missing ';' at '{'
```

MMOX.sol

```
MMOX.sol:336:18: Error: Parse error: missing ';' at '{'
MMOX.sol:369:18: Error: Parse error: missing ';' at '{'
MMOX.sol:418:18: Error: Parse error: missing ';' at '{'
MMOX.sol:469:22: Error: Parse error: missing ';' at '{'
```

MMUSD.sol

```
MMUSD.sol:336:18: Error: Parse error: missing ';' at '{'
MMUSD.sol:369:18: Error: Parse error: missing ';' at '{'
MMUSD.sol:418:18: Error: Parse error: missing ';' at '{'
MMUSD.sol:469:22: Error: Parse error: missing ';' at '{'
```

DaoTreasury.sol

```
DaoTreasury.sol:359:18: Error: Parse error: missing ';' at '{'
```

StratRecollateralize.sol

```
StratRecollateralize.sol:366:18: Error: Parse error: missing ';' at
'{'
```

StratReduceReserveLP.sol

```
StratReduceReserveLP.sol:497:18: Error: Parse error: missing ';' at
'{'
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io