SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

v1.0: 13. December, 2021

Audit

Security Assessment

07. January, 2022

For

BaksDAO

Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	13. December 2021	• Layout project • Automated- /Manual-Security Testing • Summary
1.1	07. January 2022	• Reaudit

Network

Binance Smart Chain (BEP20)

Website

https://baksdao.com/

Telegram

https://t.me/BaksDAO_INT_COMMUNITY

Twitter

https://twitter.com/BaksDAO

Facebook

https://www.facebook.com/BaksDAOint/

Medium

https://medium.com/@BaksDAO

Github

https://github.com/BaksDAO/contracts-mirror/tree/master/contracts

Description

A platform to get more from your crypto assets

Project Engagement

During the 7th of December 2021, **BaksDAO Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

Logo

Contract Link

v1.1

- Github
  - https://github.com/BaksDAO/contracts-mirror/tree/master/contracts
  - Commit: c4893f8f2f72b383950bd5539a50008937c83a73

Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon aspossible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
    i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
    ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.

2. Testing and automated analysis that includes the following:
    i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

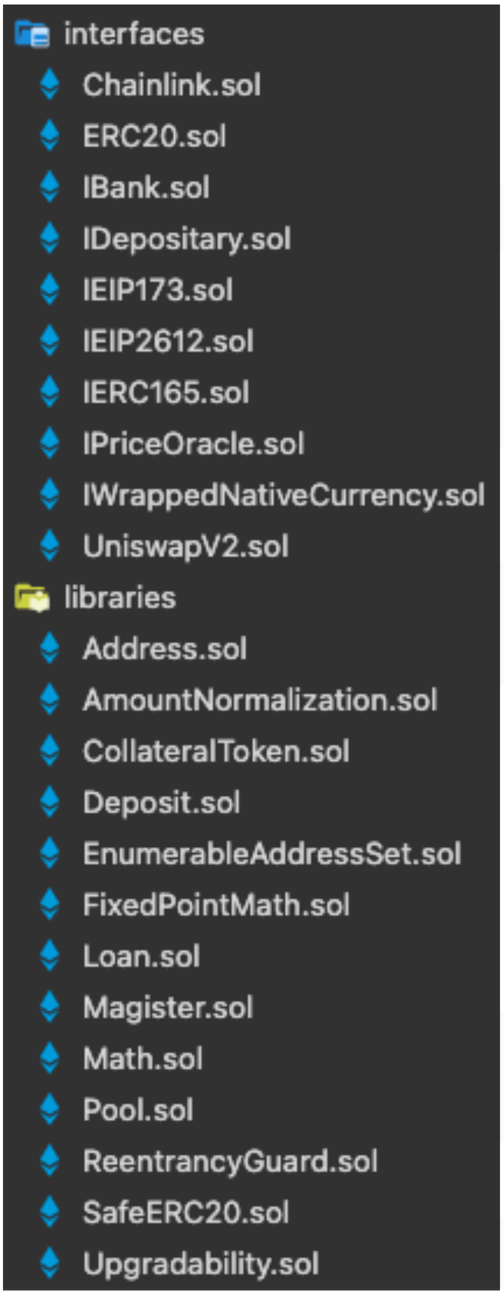
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

Used Code from other Frameworks/Smart Contracts (direct imports)

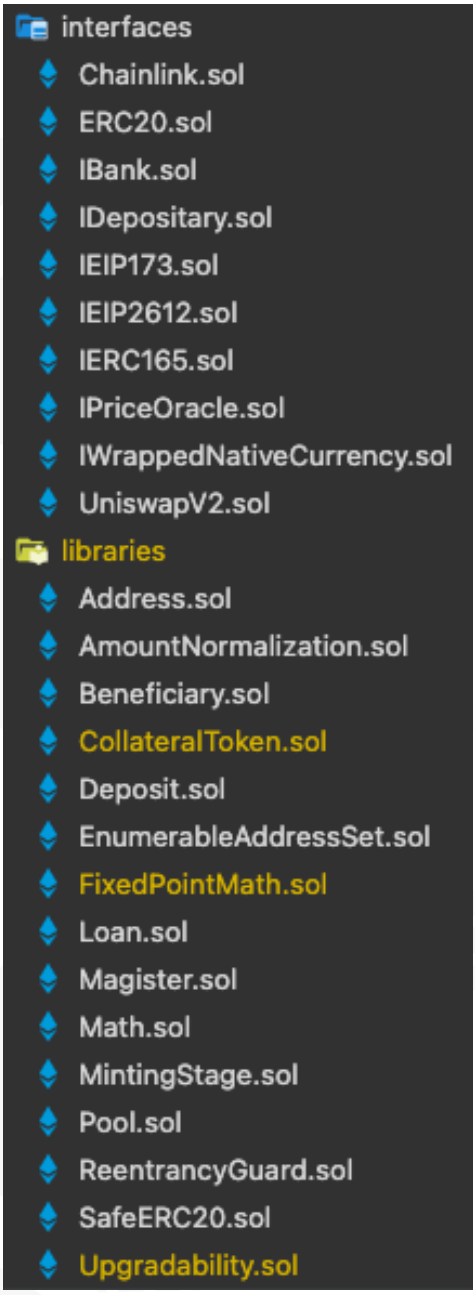
Imported packages:

v1.0



- interfaces
  - Chainlink.sol
  - ERC20.sol
  - IBank.sol
  - IDepositary.sol
  - IEIP173.sol
  - IEIP2612.sol
  - IERC165.sol
  - IPriceOracle.sol
  - IWrappedNativeCurrency.sol
  - UniswapV2.sol
- libraries
  - Address.sol
  - AmountNormalization.sol
  - CollateralToken.sol
  - Deposit.sol
  - EnumerableAddressSet.sol
  - FixedPointMath.sol
  - Loan.sol
  - Magister.sol
  - Math.sol
  - Pool.sol
  - ReentrancyGuard.sol
  - SafeERC20.sol
  - Upgradability.sol

v1.1



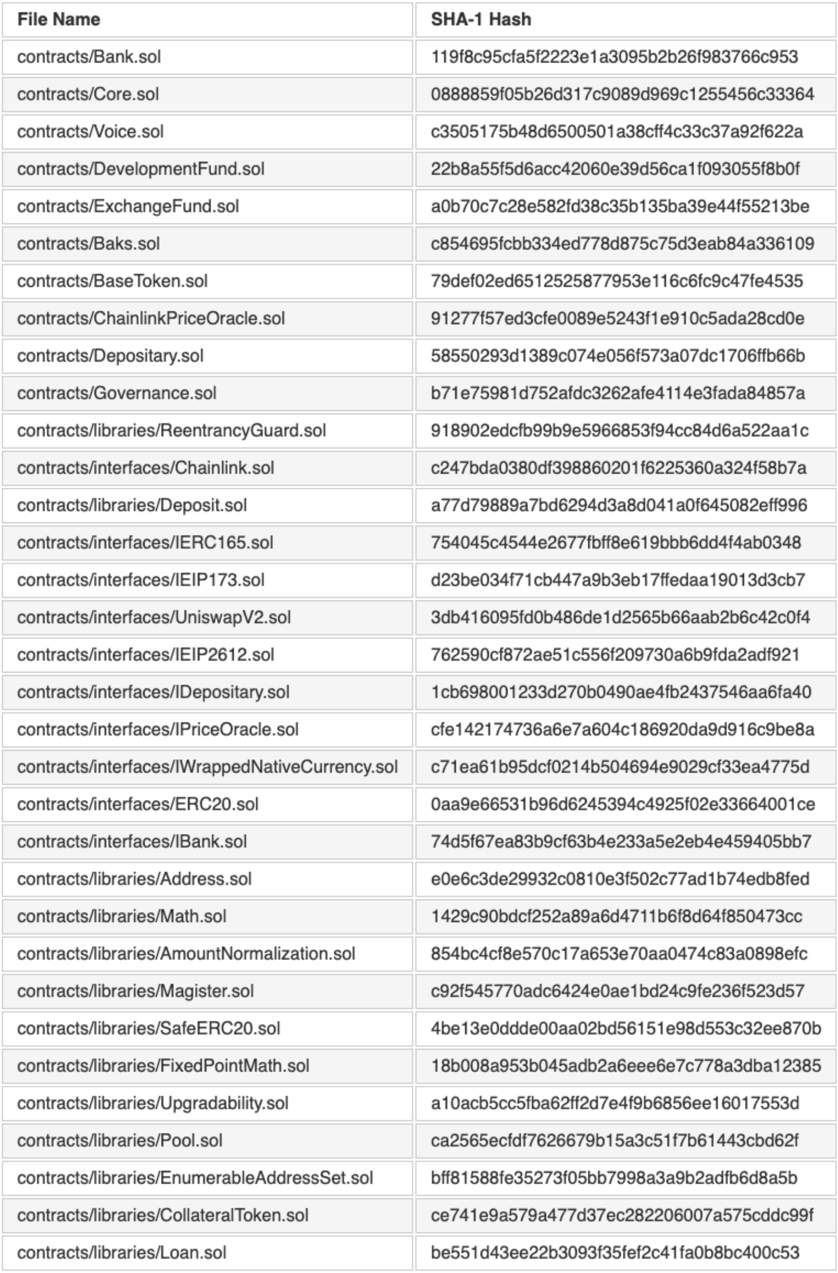
- interfaces
  - Chainlink.sol
  - ERC20.sol
  - IBank.sol
  - IDepositary.sol
  - IEIP173.sol
  - IEIP2612.sol
  - IERC165.sol
  - IPriceOracle.sol
  - IWrappedNativeCurrency.sol
  - UniswapV2.sol
- libraries
  - Address.sol
  - AmountNormalization.sol
  - Beneficiary.sol
  - CollateralToken.sol
  - Deposit.sol
  - EnumerableAddressSet.sol
  - FixedPointMath.sol
  - Loan.sol
  - Magister.sol
  - Math.sol
  - MintingStage.sol
  - Pool.sol
  - ReentrancyGuard.sol
  - SafeERC20.sol
  - Upgradability.sol

Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

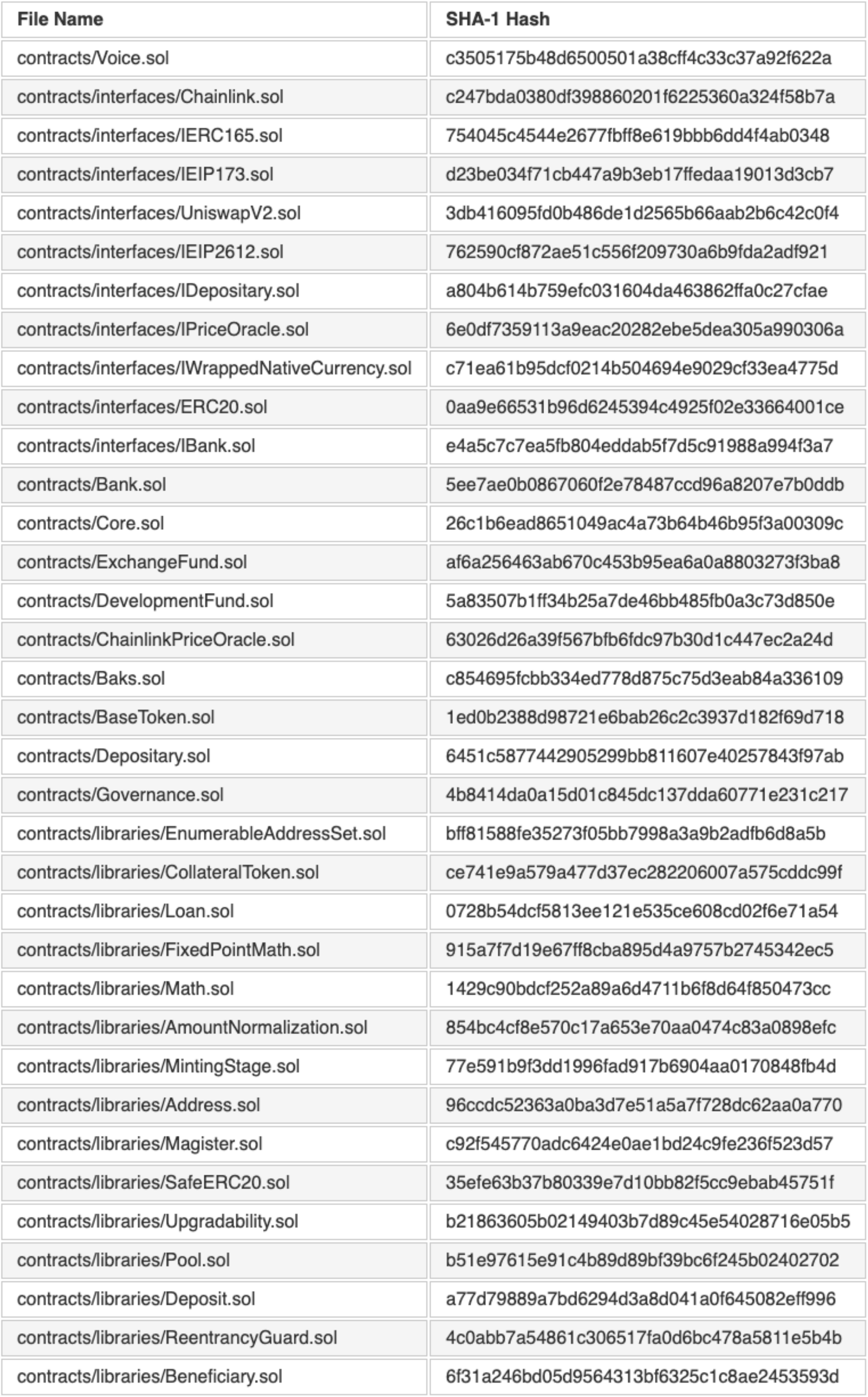
A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.

v1.0



File Name	SHA-1 Hash
contracts/Bank.sol	119f8c95cfa5f2223e1a3095b2b26f983766c953
contracts/Core.sol	0888859f05b26d317c9089d969c1255456c33364
contracts/Voice.sol	c3505175b48d6500501a38cff4c33c37a92f622a
contracts/DevelopmentFund.sol	22b8a55f5d6acc42060e39d56ca1f093055f8b0f
contracts/ExchangeFund.sol	a0b70c7c28e582fd38c35b135ba39e44f55213be
contracts/Baks.sol	c854695fcbb334ed778d875c75d3eab84a336109
contracts/BaseToken.sol	79def02ed6512525877953e116c6fc9c47fe4535
contracts/ChainlinkPriceOracle.sol	91277f57ed3cfe0089e5243f1e910c5ada28cd0e
contracts/Depositary.sol	58550293d1389c074e056f573a07dc1706ffb66b
contracts/Governance.sol	b71e75981d752afdc3262afe4114e3fada84857a
contracts/libraries/ReentrancyGuard.sol	918902edcfb99b9e5966853f94cc84d6a522aa1c
contracts/interfaces/Chainlink.sol	c247bda0380df398860201f6225360a324f58b7a
contracts/libraries/Deposit.sol	a77d79889a7bd6294d3a8d041a0f645082eff996
contracts/interfaces/IERC165.sol	754045c4544e2677fbff8e619bbb6dd4f4ab0348
contracts/interfaces/IEIP173.sol	d23be034f71cb447a9b3eb17ffedaa19013d3cb7
contracts/interfaces/UniswapV2.sol	3db416095fd0b486de1d2565b66aab2b6c42c0f4
contracts/interfaces/IEIP2612.sol	762590cf872ae51c556f209730a6b9fda2adf921
contracts/interfaces/IDepositary.sol	1cb698001233d270b0490ae4fb2437546aa6fa40
contracts/interfaces/IPriceOracle.sol	cfe142174736a6e7a604c186920da9d916c9be8a
contracts/interfaces/IWrappedNativeCurrency.sol	c71ea61b95dcf0214b504694e9029cf33ea4775d
contracts/interfaces/ERC20.sol	0aa9e66531b96d6245394c4925f02e33664001ce
contracts/interfaces/IBank.sol	74d5f67ea83b9cf63b4e233a5e2eb4e459405bb7
contracts/libraries/Address.sol	e0e6c3de29932c0810e3f502c77ad1b74edb8fed
contracts/libraries/Math.sol	1429c90bdcf252a89a6d4711b6f8d64f850473cc
contracts/libraries/AmountNormalization.sol	854bc4cf8e570c17a653e70aa0474c83a0898efc
contracts/libraries/Magister.sol	c92f545770adc6424e0ae1bd24c9fe236f523d57
contracts/libraries/SafeERC20.sol	4be13e0ddde00aa02bd56151e98d553c32ee870b
contracts/libraries/FixedPointMath.sol	18b008a953b045adb2a6eee6e7c778a3dba12385
contracts/libraries/Upgradability.sol	a10acb5cc5fba62ff2d7e4f9b6856ee16017553d
contracts/libraries/Pool.sol	ca2565ecfdf7626679b15a3c51f7b61443cbd62f
contracts/libraries/EnumerableAddressSet.sol	bff81588fe35273f05bb7998a3a9b2adfb6d8a5b
contracts/libraries/CollateralToken.sol	ce741e9a579a477d37ec282206007a575cddc99f
contracts/libraries/Loan.sol	be551d43ee22b3093f35fef2c41fa0b8bc400c53

v1.1

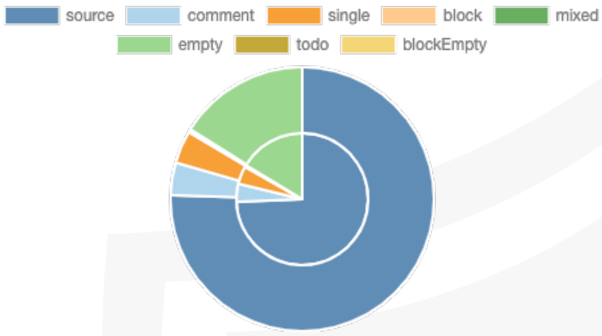


File Name	SHA-1 Hash
contracts/Voice.sol	c3505175b48d6500501a38cff4c33c37a92f622a
contracts/interfaces/Chainlink.sol	c247bda0380df398860201f6225360a324f58b7a
contracts/interfaces/IERC165.sol	754045c4544e2677fbff8e619bbb6dd4f4ab0348
contracts/interfaces/IEIP173.sol	d23be034f71cb447a9b3eb17ffedaa19013d3cb7
contracts/interfaces/UniswapV2.sol	3db416095fd0b486de1d2565b66aab2b6c42c0f4
contracts/interfaces/IEIP2612.sol	762590cf872ae51c556f209730a6b9fda2adf921
contracts/interfaces/IDepositary.sol	a804b614b759efc031604da463862ffa0c27cfae
contracts/interfaces/IPriceOracle.sol	6e0df7359113a9eac20282ebe5dea305a990306a
contracts/interfaces/IWrappedNativeCurrency.sol	c71ea61b95dcf0214b504694e9029cf33ea4775d
contracts/interfaces/ERC20.sol	0aa9e66531b96d6245394c4925f02e33664001ce
contracts/interfaces/IBank.sol	e4a5c7c7ea5fb804eddab5f7d5c91988a994f3a7
contracts/Bank.sol	5ee7ae0b0867060f2e78487ccd96a8207e7b0ddb
contracts/Core.sol	26c1b6ead8651049ac4a73b64b46b95f3a00309c
contracts/ExchangeFund.sol	af6a256463ab670c453b95ea6a0a8803273f3ba8
contracts/DevelopmentFund.sol	5a83507b1ff34b25a7de46bb485fb0a3c73d850e
contracts/ChainlinkPriceOracle.sol	63026d26a39f567bfb6fdc97b30d1c447ec2a24d
contracts/Baks.sol	c854695fcbb334ed778d875c75d3eab84a336109
contracts/BaseToken.sol	1ed0b2388d98721e6bab26c2c3937d182f69d718
contracts/Depositary.sol	6451c5877442905299bb811607e40257843f97ab
contracts/Governance.sol	4b8414da0a15d01c845dc137dda60771e231c217
contracts/libraries/EnumerableAddressSet.sol	bff81588fe35273f05bb7998a3a9b2adfb6d8a5b
contracts/libraries/CollateralToken.sol	ce741e9a579a477d37ec282206007a575cddc99f
contracts/libraries/Loan.sol	0728b54dcf5813ee121e535ce608cd02f6e71a54
contracts/libraries/FixedPointMath.sol	915a7f7d19e67ff8cba895d4a9757b2745342ec5
contracts/libraries/Math.sol	1429c90bdcf252a89a6d4711b6f8d64f850473cc
contracts/libraries/AmountNormalization.sol	854bc4cf8e570c17a653e70aa0474c83a0898efc
contracts/libraries/MintingStage.sol	77e591b9f3dd1996fad917b6904aa0170848fb4d
contracts/libraries/Address.sol	96ccdc52363a0ba3d7e51a5a7f728dc62aa0a770
contracts/libraries/Magister.sol	c92f545770adc6424e0ae1bd24c9fe236f523d57
contracts/libraries/SafeERC20.sol	35efe63b37b80339e7d10bb82f5cc9ebab45751f
contracts/libraries/Upgradability.sol	b21863605b02149403b7d89c45e54028716e05b5
contracts/libraries/Pool.sol	b51e97615e91c4b89d89bf39bc6f245b02402702
contracts/libraries/Deposit.sol	a77d79889a7bd6294d3a8d041a0f645082eff996
contracts/libraries/ReentrancyGuard.sol	4c0abb7a54861c306517fa0d6bc478a5811e5b4b
contracts/libraries/Beneficiary.sol	6f31a246bd05d9564313bf6325c1c8ae2453593d

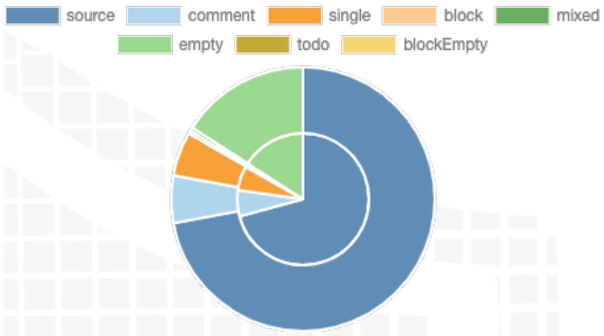
Metrics

Source Lines

v1.0

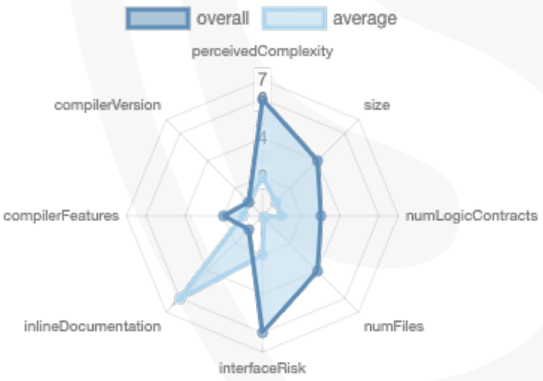


v1.1

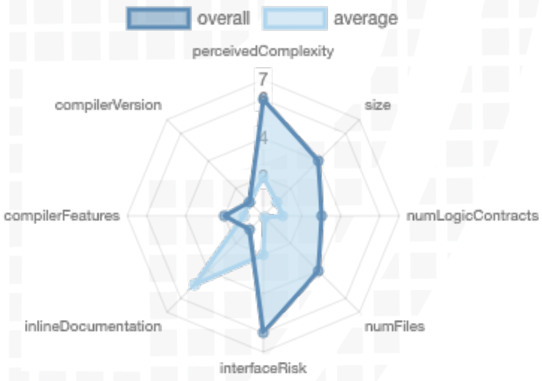


Risk Level

v1.0



v1.1



Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	10	12	14	5
1.1	10	14	14	5

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Version	Public	Payable
1.0	153	6
1.1	174	6

Version	External	Internal	Private	Pure	View
1.0	142	166	0	19	86
1.1	162	177	0	21	98

State Variables

Version	Total	Public
1.0	85	53
1.1	93	59

Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`0.8.10`		`yes`	`yes` (13 asm blocks)	
1.1	`0.8.10`		`yes`	`yes` (13 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/Create/Create2
1.0			yes	yes	yes	
1.1			yes	yes	yes	

Scope of Work

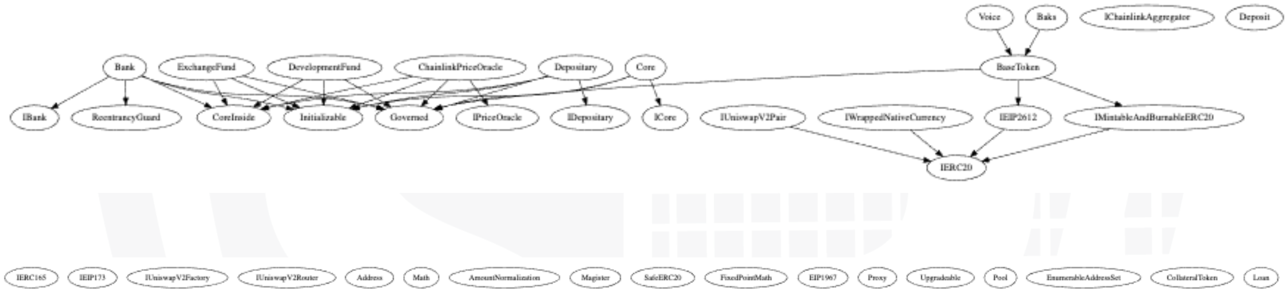
The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

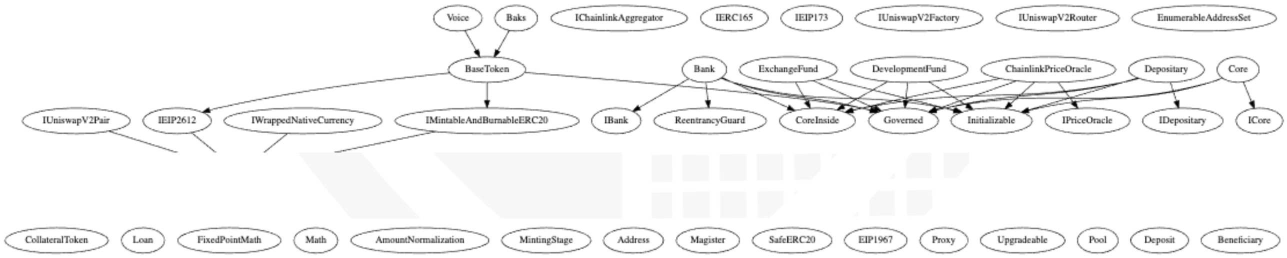
1. Correct implementation of Token standard
2. Deployer cannot mint any new tokens
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract
5. Overall checkup (Smart Contract Security)

Inheritance Graph

v1.0



v1.1



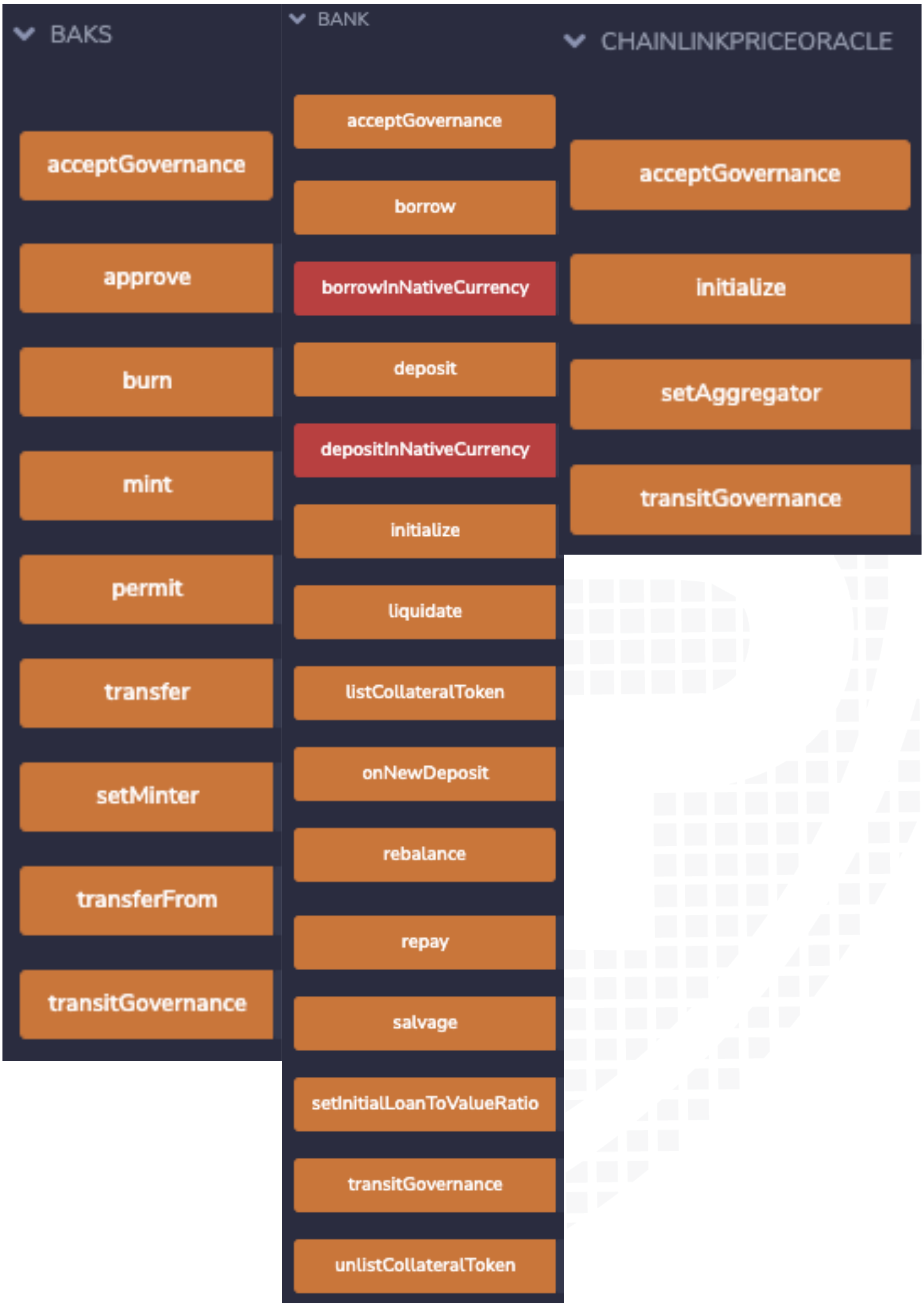
Verify Claims

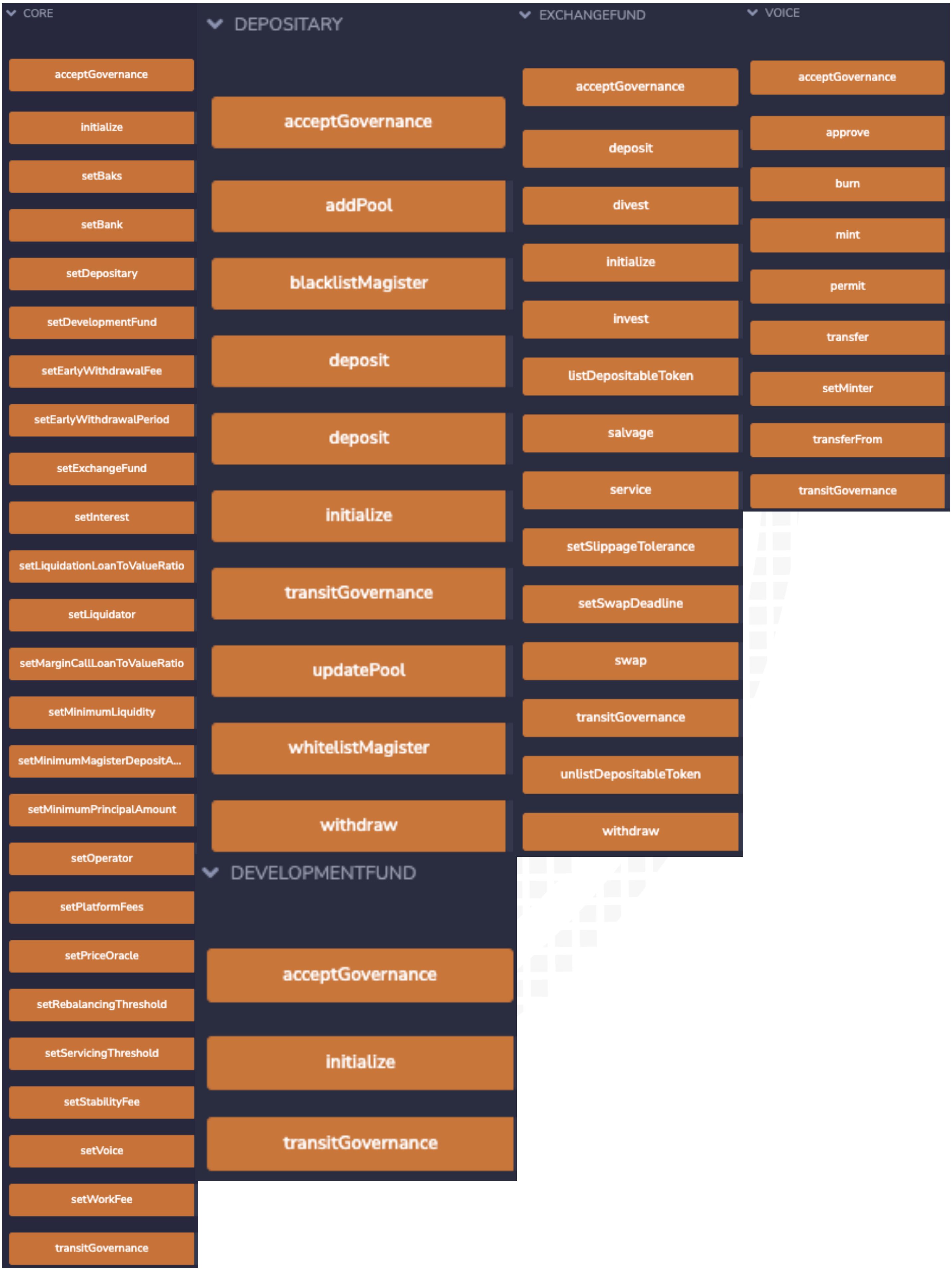
Correct implementation of Token standard

Tested	Verified
✓	✓

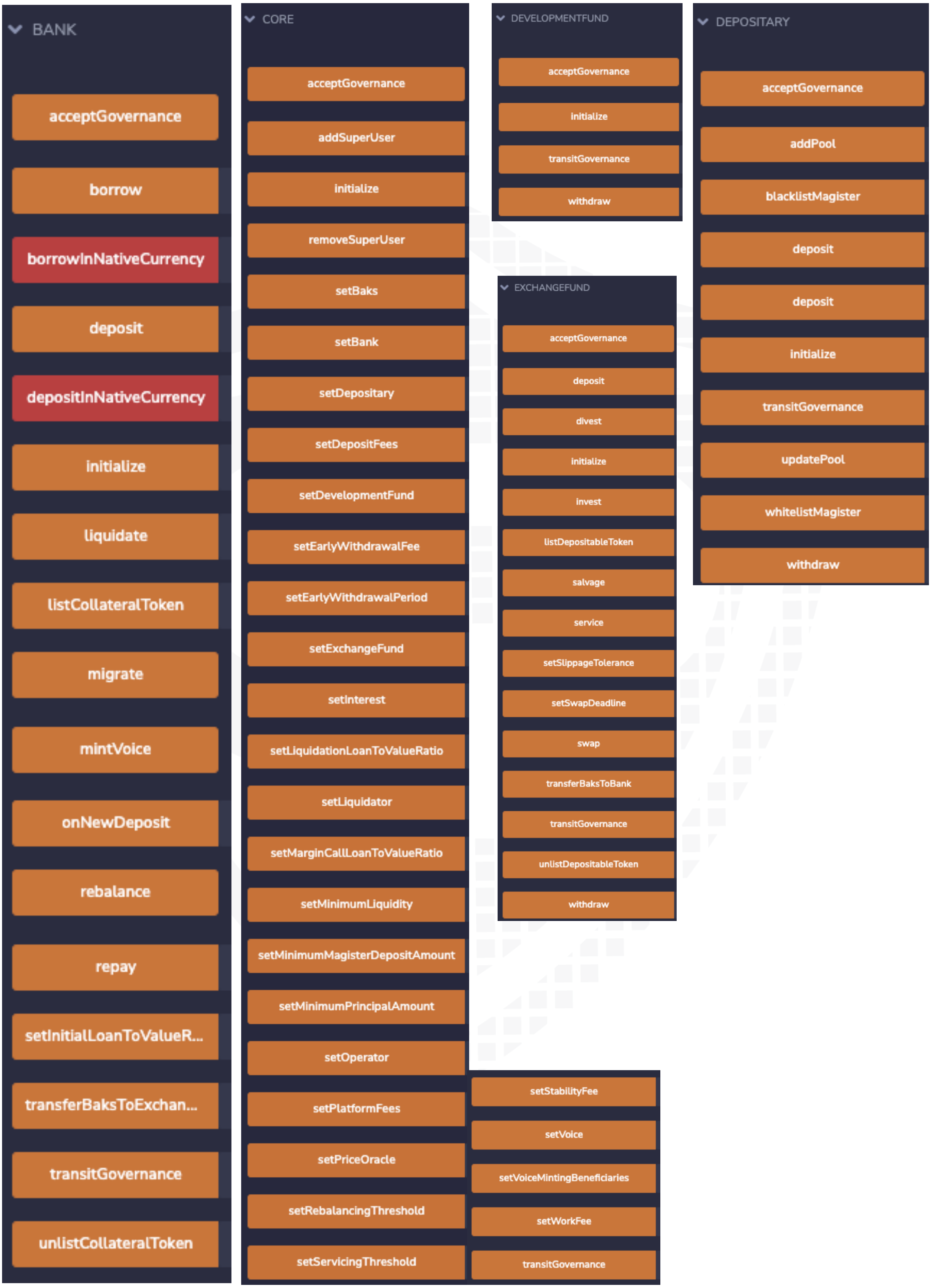
Function	Description	Exist	Tested	Verified
TotalSupply	provides information about the total token supply	✓	✓	✓
BalanceOf	provides account balance of the owner's account	✓	✓	✓
Transfer	executes transfers of a specified number of tokens to a specified address	✓	✓	✓
TransferFrom	executes transfers of a specified number of tokens from a specified address	✓	✓	✓
Approve	allow a spender to withdraw a set number of tokens from a specified account	✓	✓	✓
Allowance	returns a set number of tokens from a spender to the owner	✓	✓	✓

Write functions of contract v1.0




CORE
acceptGovernance
initialize
setBaks
setBank
setDepositary
setDevelopmentFund
setEarlyWithdrawalFee
setEarlyWithdrawalPeriod
setExchangeFund
setInterest
setLiquidationLoanToValueRatio
setLiquidator
setMarginCallLoanToValueRatio
setMinimumLiquidity
setMinimumMagisterDepositA...
setMinimumPrincipalAmount
setOperator
setPlatformFees
setPriceOracle
setRebalancingThreshold
setServicingThreshold
setStabilityFee
setVoice
setWorkFee
transitGovernance
DEPOSITARY
acceptGovernance
addPool
blacklistMagister
deposit
deposit
initialize
transitGovernance
updatePool
whitelistMagister
withdraw
DEVELOPMENTFUND
acceptGovernance
initialize
transitGovernance
EXCHANGEFUND
acceptGovernance
deposit
divest
initialize
invest
listDepositableToken
salvage
service
setSlippageTolerance
setSwapDeadline
swap
transitGovernance
unlistDepositableToken
withdraw
VOICE
acceptGovernance
approve
burn
mint
permit
transfer
setMinter
transferFrom
transitGovernance

v1.0


BANK
acceptGovernance
borrow
borrowInNativeCurrency
deposit
depositInNativeCurrency
initialize
liquidate
listCollateralToken
migrate
mintVoice
onNewDeposit
rebalance
repay
setInitialLoanToValueR...
transferBaksToExchan...
transitGovernance
unlistCollateralToken
CORE
acceptGovernance
addSuperUser
initialize
removeSuperUser
setBaks
setBank
setDepositary
setDepositFees
setDevelopmentFund
setEarlyWithdrawalFee
setEarlyWithdrawalPeriod
setExchangeFund
setInterest
setLiquidationLoanToValueRatio
setLiquidator
setMarginCallLoanToValueRatio
setMinimumLiquidity
setMinimumMagisterDepositAmount
setMinimumPrincipalAmount
setOperator
setPlatformFees
setPriceOracle
setRebalancingThreshold
setServicingThreshold
setStabilityFee
setVoice
setVoiceMintingBeneficiaries
setWorkFee
transitGovernance
DEVELOPMENTFUND
acceptGovernance
initialize
transitGovernance
withdraw
EXCHANGEFUND
acceptGovernance
deposit
divest
initialize
invest
listDepositableToken
salvage
service
setSlippageTolerance
setSwapDeadline
swap
transferBaksToBank
transitGovernance
unlistDepositableToken
withdraw
DEPOSITARY
acceptGovernance
addPool
blacklistMagister
deposit
deposit
initialize
transitGovernance
updatePool
whitelistMagister
withdraw

Deployer cannot mint any new tokens

File	Name	Exist	Tested	Verified
Baks	cannot mint	✓	✓	✗
Voice	cannot mint	✓	✓	✗

Max / Total Supply:

Comments:

v1.0

- Baks
  - onlyMinter can mint
- Voice
  - onlyMinter can mint

Deployer cannot burn or lock user funds

File	Name	Exist	Teste d	Verified
Baks	cannot lock	✓	✓	✓
	cannot burn	✓	✓	✗
Voice	cannot lock	✓	✓	✓
	cannot burn	✓	✓	✗

Deployer cannot pause the contract

File	Name	Exist	Tested	Verified
All	cannot pause	–	–	–

Overall checkup (Smart Contract Security)

Tested	Verified
✓	✓

Legend

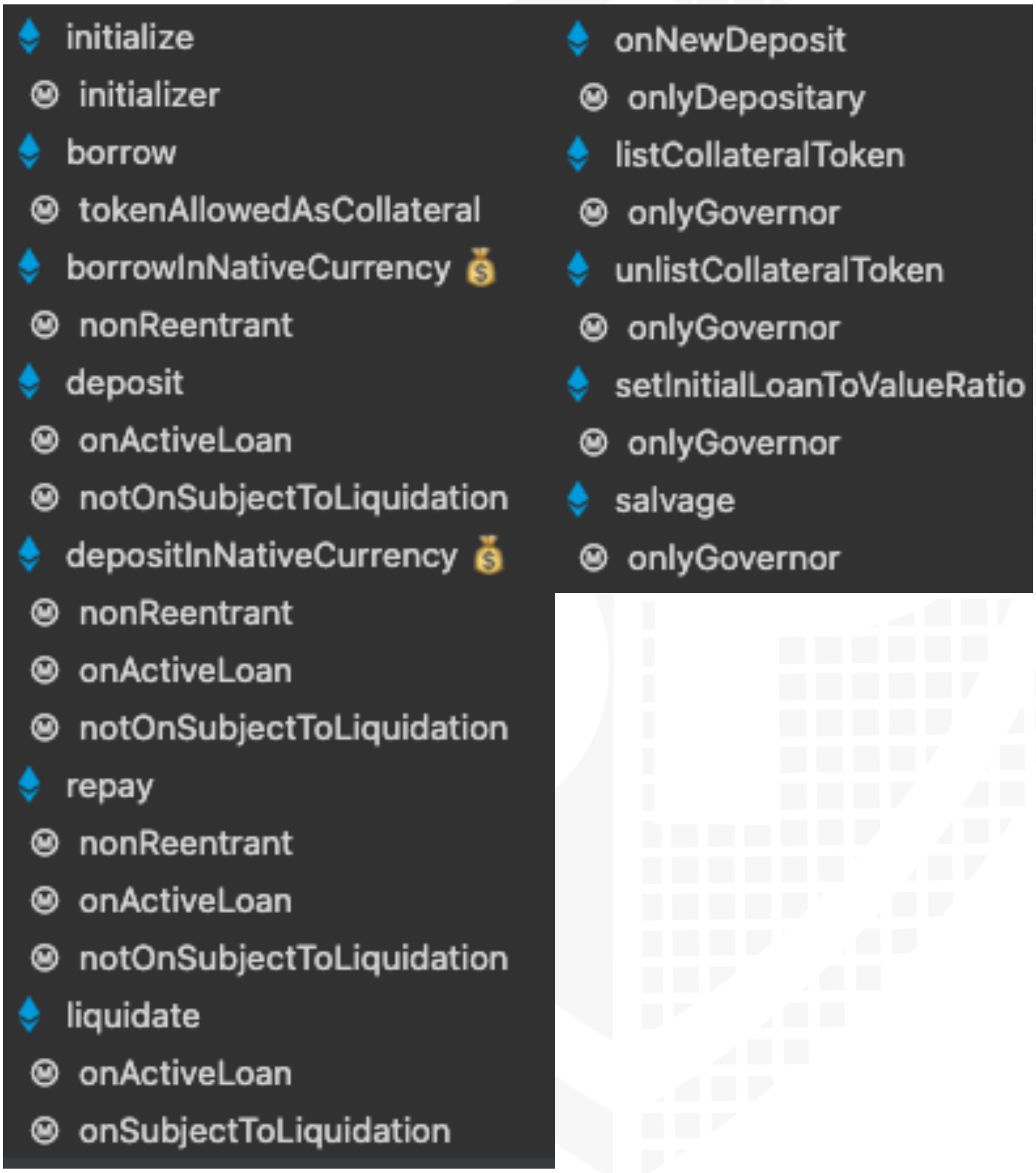
Attribute	Symbol
Verfified / Checked	✓
Partly Verified	🚩
Unverified / Not checked	✗
Not available	–

Modifiers

Baks/Voice

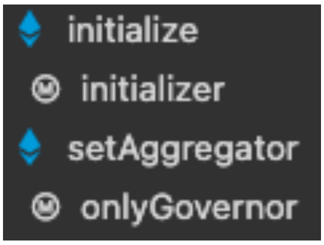
- onlyMinter
    - Mint
    - Burn
- onlyGovernor
    - setMinter

Bank



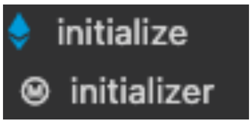
- initialize
    - initializer
- borrow
    - tokenAllowedAsCollateral
- borrowInNativeCurrency 💰
    - nonReentrant
- deposit
    - onActiveLoan
    - notOnSubjectToLiquidation
- depositInNativeCurrency 💰
    - nonReentrant
    - onActiveLoan
    - notOnSubjectToLiquidation
- repay
    - nonReentrant
    - onActiveLoan
    - notOnSubjectToLiquidation
- liquidate
    - onActiveLoan
    - onSubjectToLiquidation
- onNewDeposit
    - onlyDepositary
- listCollateralToken
    - onlyGovernor
- unlistCollateralToken
    - onlyGovernor
- setInitialLoanToValueRatio
    - onlyGovernor
- salvage
    - onlyGovernor

ChainlinkPriceOracle



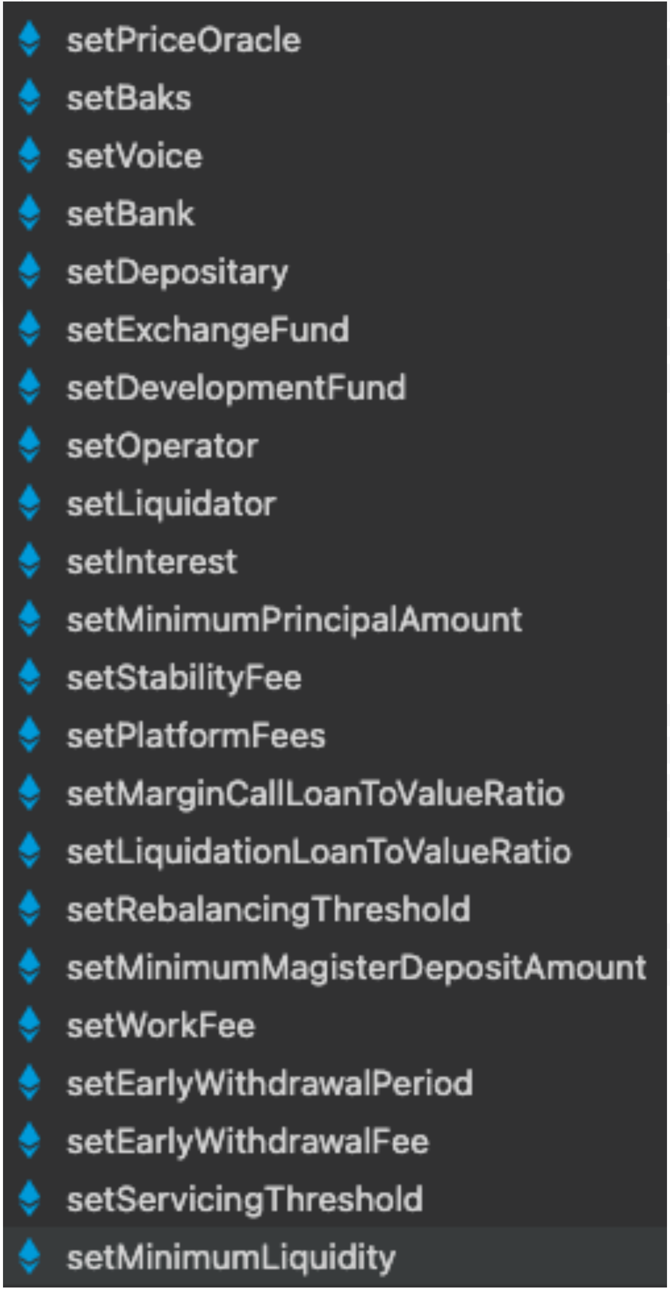
initialize
initializer
setAggregator
onlyGovernor

Core



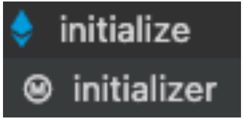
initialize
initializer

- onlyGovernor

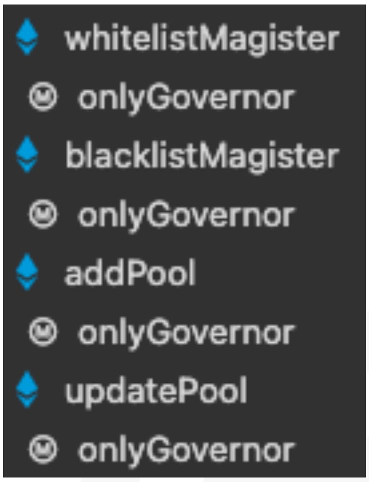


setPriceOracle
setBaks
setVoice
setBank
setDepositary
setExchangeFund
setDevelopmentFund
setOperator
setLiquidator
setInterest
setMinimumPrincipalAmount
setStabilityFee
setPlatformFees
setMarginCallLoanToValueRatio
setLiquidationLoanToValueRatio
setRebalancingThreshold
setMinimumMagisterDepositAmount
setWorkFee
setEarlyWithdrawalPeriod
setEarlyWithdrawalFee
setServicingThreshold
setMinimumLiquidity

Depositary



initialize
  initializer



whitelistMagister
  onlyGovernor
blacklistMagister
  onlyGovernor
addPool
  onlyGovernor
updatePool
  onlyGovernor

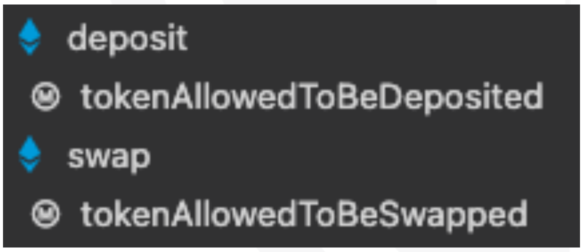
onlyGovernor is changed to onlySuperUser modifier

DevelopmentFund

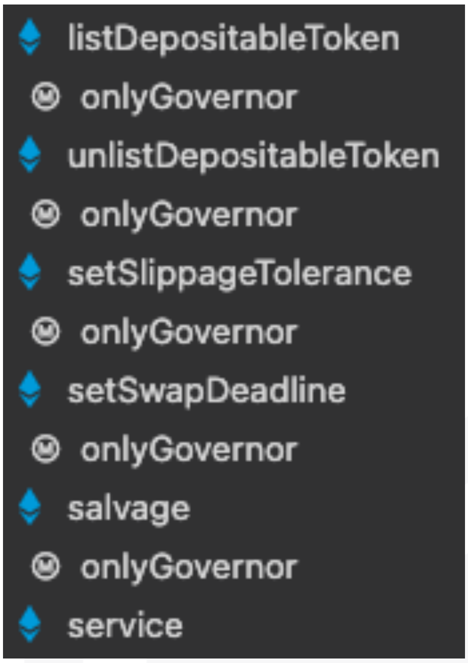
- Initialize
  - Initializer

ExchangeFund

- Initialize
  - Initializer



deposit
  tokenAllowedToBeDeposited
swap
  tokenAllowedToBeSwapped

- 

listDepositableToken
onlyGovernor
unlistDepositableToken
onlyGovernor
setSlippageTolerance
onlyGovernor
setSwapDeadline
onlyGovernor
salvage
onlyGovernor
service

- onlySuperUser modifier was added to following functions
  - Deposit
  - Invest
  - Divest
  - Withdraw
  -

Governance

- transitGovernance
  - onlyGovernor

CallGraph

V1.0

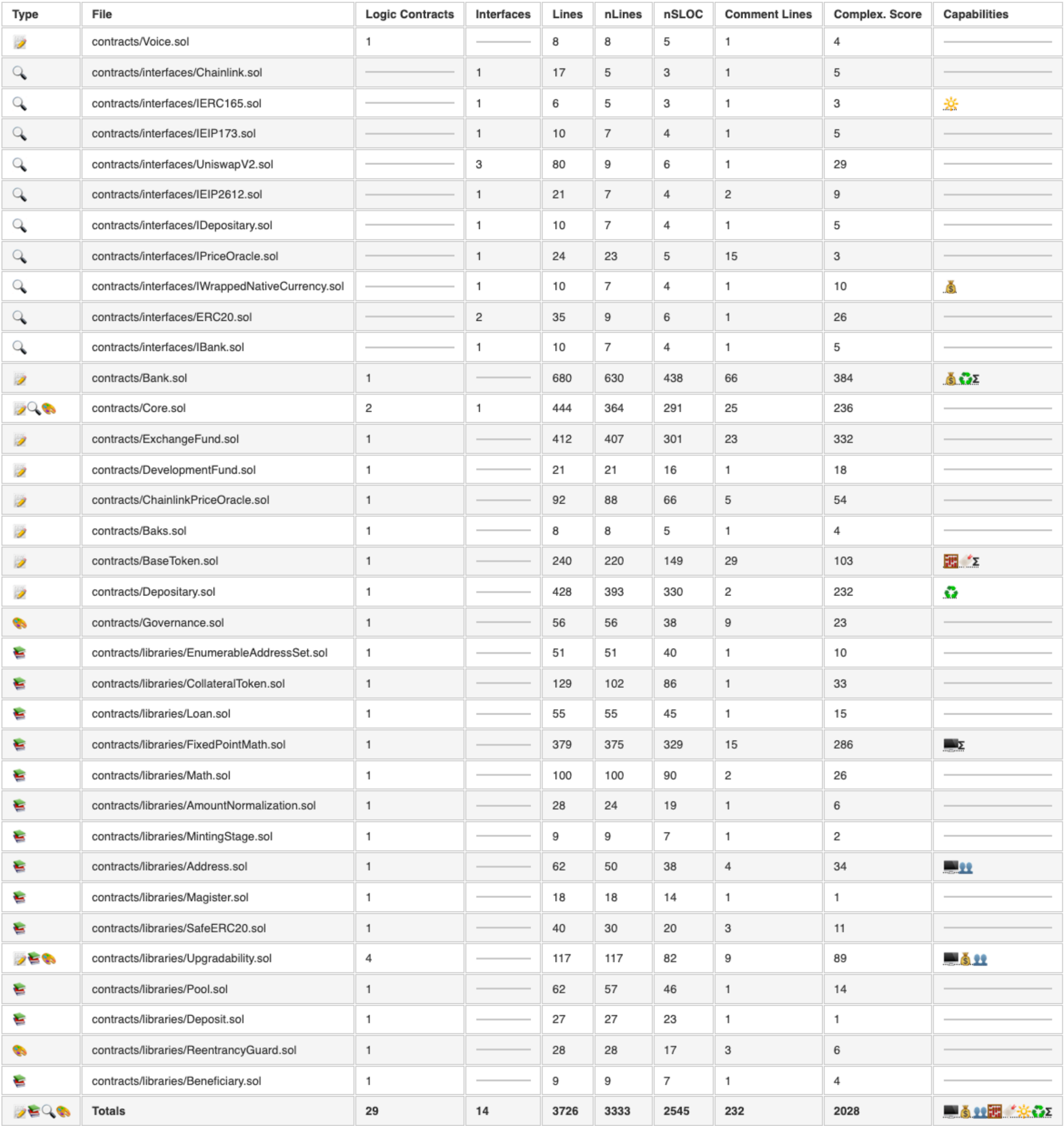
v1.1

Source Units in Scope

v1.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	contracts/Bank.sol	1	—	605	555	400	50	342	💰♻️Σ
📝🔍🎨	contracts/Core.sol	2	1	325	263	204	22	187	—
📝	contracts/Voice.sol	1	—	8	8	5	1	4	—
📝	contracts/DevelopmentFund.sol	1	—	13	13	10	1	12	—
📝	contracts/ExchangeFund.sol	1	—	402	397	294	23	319	—
📝	contracts/Baks.sol	1	—	8	8	5	1	4	—
📝	contracts/BaseToken.sol	1	—	226	206	162	2	103	🧮🌀Σ
📝	contracts/ChainlinkPriceOracle.sol	1	—	88	84	66	1	54	—
📝	contracts/Depositary.sol	1	—	422	387	323	2	225	♻️
🎨	contracts/Governance.sol	1	—	52	52	42	1	23	—
🎨	contracts/libraries/ReentrancyGuard.sol	1	—	27	27	18	1	6	—
🔍	contracts/interfaces/Chainlink.sol	—	1	17	5	3	1	5	—
📚	contracts/libraries/Deposit.sol	1	—	27	27	23	1	1	—
🔍	contracts/interfaces/IERC165.sol	—	1	6	5	3	1	3	🔆
🔍	contracts/interfaces/IEIP173.sol	—	1	10	7	4	1	5	—
🔍	contracts/interfaces/UniswapV2.sol	—	3	80	9	6	1	29	—
🔍	contracts/interfaces/IEIP2612.sol	—	1	21	7	4	2	9	—
🔍	contracts/interfaces/IDepositary.sol	—	1	6	5	3	1	3	—
🔍	contracts/interfaces/IPriceOracle.sol	—	1	24	23	8	12	3	—
🔍	contracts/interfaces/IWrappedNativeCurrency.sol	—	1	10	7	4	1	10	💰
🔍	contracts/interfaces/ERC20.sol	—	2	35	9	6	1	26	—
🔍	contracts/interfaces/IBank.sol	—	1	8	7	4	1	3	—
📚	contracts/libraries/Address.sol	1	—	60	48	39	1	34	🖥👥
📚	contracts/libraries/Math.sol	1	—	100	100	90	2	26	—
📚	contracts/libraries/AmountNormalization.sol	1	—	28	24	19	1	6	—
📚	contracts/libraries/Magister.sol	1	—	18	18	14	1	1	—
📚	contracts/libraries/SafeERC20.sol	1	—	39	29	21	1	11	—
📚	contracts/libraries/FixedPointMath.sol	1	—	374	370	333	6	286	🖥Σ
📝📚🎨	contracts/libraries/Upgradability.sol	4	—	113	113	86	1	89	🖥💰👥
📚	contracts/libraries/Pool.sol	1	—	62	58	47	1	15	—
📚	contracts/libraries/EnumerableAddressSet.sol	1	—	51	51	40	1	10	—
📚	contracts/libraries/CollateralToken.sol	1	—	129	102	86	1	33	—
📚	contracts/libraries/Loan.sol	1	—	55	55	45	1	15	—
📝📚🔍🎨	**Totals**	**27**	**14**	**3449**	**3079**	**2417**	**145**	**1902**	🖥💰👥🧮🌀🔆♻️Σ

v1.1



Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝	contracts/Voice.sol	1	—	8	8	5	1	4	—
🔍	contracts/interfaces/Chainlink.sol	—	1	17	5	3	1	5	—
🔍	contracts/interfaces/IERC165.sol	—	1	6	5	3	1	3	🔆
🔍	contracts/interfaces/IEIP173.sol	—	1	10	7	4	1	5	—
🔍	contracts/interfaces/UniswapV2.sol	—	3	80	9	6	1	29	—
🔍	contracts/interfaces/IEIP2612.sol	—	1	21	7	4	2	9	—
🔍	contracts/interfaces/IDepositary.sol	—	1	10	7	4	1	5	—
🔍	contracts/interfaces/IPriceOracle.sol	—	1	24	23	5	15	3	—
🔍	contracts/interfaces/IWrappedNativeCurrency.sol	—	1	10	7	4	1	10	💰
🔍	contracts/interfaces/ERC20.sol	—	2	35	9	6	1	26	—
🔍	contracts/interfaces/IBank.sol	—	1	10	7	4	1	5	—
📝	contracts/Bank.sol	1	—	680	630	438	66	384	💰♻️Σ
📝🔍🎨	contracts/Core.sol	2	1	444	364	291	25	236	—
📝	contracts/ExchangeFund.sol	1	—	412	407	301	23	332	—
📝	contracts/DevelopmentFund.sol	1	—	21	21	16	1	18	—
📝	contracts/ChainlinkPriceOracle.sol	1	—	92	88	66	5	54	—
📝	contracts/Baks.sol	1	—	8	8	5	1	4	—
📝	contracts/BaseToken.sol	1	—	240	220	149	29	103	🧮🌀Σ
📝	contracts/Depositary.sol	1	—	428	393	330	2	232	♻️
🎨	contracts/Governance.sol	1	—	56	56	38	9	23	—
📚	contracts/libraries/EnumerableAddressSet.sol	1	—	51	51	40	1	10	—
📚	contracts/libraries/CollateralToken.sol	1	—	129	102	86	1	33	—
📚	contracts/libraries/Loan.sol	1	—	55	55	45	1	15	—
📚	contracts/libraries/FixedPointMath.sol	1	—	379	375	329	15	286	🖥Σ
📚	contracts/libraries/Math.sol	1	—	100	100	90	2	26	—
📚	contracts/libraries/AmountNormalization.sol	1	—	28	24	19	1	6	—
📚	contracts/libraries/MintingStage.sol	1	—	9	9	7	1	2	—
📚	contracts/libraries/Address.sol	1	—	62	50	38	4	34	🖥👥
📚	contracts/libraries/Magister.sol	1	—	18	18	14	1	1	—
📚	contracts/libraries/SafeERC20.sol	1	—	40	30	20	3	11	—
📝📚🎨	contracts/libraries/Upgradability.sol	4	—	117	117	82	9	89	🖥💰👥
📚	contracts/libraries/Pool.sol	1	—	62	57	46	1	14	—
📚	contracts/libraries/Deposit.sol	1	—	27	27	23	1	1	—
🎨	contracts/libraries/ReentrancyGuard.sol	1	—	28	28	17	3	6	—
📚	contracts/libraries/Beneficiary.sol	1	—	9	9	7	1	4	—
📝📚🔍🎨	**Totals**	**29**	**14**	**3726**	**3333**	**2545**	**232**	**2028**	🖥💰👥🧮🌀🔆♻️Σ

Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
nSLOC	normalized source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

Audit Results

AUDIT PASSED

Critical issues

- no critical issues found -

High issues

- no high issues found -

Medium issues

- no medium issues found -

Low issues

Issue	File	Type	Line	Description
#1	Main	Contract doesn't import npm packages from source (like OpenZeppelin etc.)	-	We recommend to import all packages from npm directly without flatten the contract. Functions could be modified or can be susceptible to vulnerabilities
#2	Core	Missing Zero Address Validation (missing-zero-check)	185, 187, 186, 188, 248, 258, 263, 273, 268, 283, 278, 243, 253	Check that the address is not zero
#3	Baks	Local variables shadowing	7	Rename the local variables that shadow another component

Informational issues

Issue	File	Type	Line	Description
#1	ExchangeFund	Unused return values	111, 112, 114, 115, 117, 118	Ensure that all the return values of the function calls are used and handle both success and failure cases if needed by the business logic

#2	FixedPointMath	Functions that are not used	124	Remove unused functions
#3	Math	Functions that are not used	9	Remove unused functions
#4	Bank	Unused state variables	121	Remove unused state variables
#5	Core	Costly loop	400	Use a local variable for voiceTotalShares and set it after the iteration is done

Audit Comments

13. December 2021:

- Read whole report for more information

07. January 2022:

- Bank
    - New errors were added
    - MintingStage was added
    - mintVoice function was added
    - transferBaksToExchangeFund function was added
    - migrate function was added
    - getTotalValueLocked function was added
- Core
    - DepositFeesUpdated event was added
    - Voice minting schedule state variable was added
    - setDepositFees function was added
    - setVoiceMintingBeneficiaries function was added
    - New modifier (onlySuperUser) was added
- Depositary
    - getTotalValueLocked function was added
- DevelopmentFund
    - withdraw function was added
- ExchangeFund
    - transferBaksToBank function was added
- MintingStage
    - New source file was added to project

BaksDAO Test Protocol

Because the repository is private, we chose not to run a public test on the testnet and instead used a local Ganache installation for the subsequent tests.

Compiling successful

```
Compiling 36 files with 0.8.10
Generating typings for: 44 artifacts in dir: ./src/types/ for target: ethers-v5
Successfully generated 65 typings!
Compilation finished successfully
```

Note: Solidity 0.8.10 is not fully supported by Hardhat yet. Hardhat can still be used, but some features, like stack traces, might not work correctly.

Deployment

All contracts were deployed on local running blockchain node. No excessive gas usages were detected.

Conclusion

The core functions such as deposit and withdraw also work manually on the blockchain as indicated in the unit tests. Corresponding events are fired. Custom claims were not specified, the basic functions that can be constructed from the whitepaper and the unit tests were tested.

The contract is safe to deploy and basic logic errors were not detected. The code is written to the best standard and sufficiently commented. Known risks were checked by the auditor and the code was scanned for other vulnerabilities as well.

All unit tests make sense and have also been checked. Logic errors could not be found here either.

Disclaimer

We have checked and verified the code to the best of our knowledge. However, deeper logic errors cannot be excluded and Solidproof.io cannot be held liable for any damage that may occur.

All Unit Tests successful

42 passing (5s)

```
Bank
  governed functions
    ✓ lists and unlists token as collateral (58ms)
    initial loan-to-value ratio
      ✓ reverts when token is not allowed as collateral (44ms)
      ✓ reverts when new initial loan-to-value ratio equal or higher than margin call loan-to-value ratio
      ✓ updates initial loan-to-value ratio
    salvation
      ✓ reverts when trying to salvage one of collateral tokens
      ✓ reverts when trying to salvage stablecoin
  borrow
    ✓ reverts when token is not allowed as collateral
    ✓ reverts when `amount` is zero (47ms)
    ✓ reverts when user don't approve a sufficient amount of collateral tokens (40ms)
    ✓ mints correct amount of stablecoin and creates a loan (171ms)
  loan interactions
    deposit
      ✓ reverts on inactive loan
      ✓ reverts when `amount` is zero
      ✓ reverts when user don't approve a sufficient amount of collateral tokens
      ✓ successfully deposits
    repay
      ✓ reverts when loan is subject to liquidation
      ✓ reverts when `amount` is zero
    liquidation
      ✓ should revert when loan is not subject to liquidation
      ✓ should liquidate loan

Base Token
  ✓ have correct minter address
  metadata
    ✓ returns the correct name
    ✓ returns the correct symbol
    ✓ returns the correct number of decimals
  total supply
    ✓ returns the total supply of tokens

Core
  bank
    ✓ updates stability fee
    ✓ updates margin call loan-to-value ratio
    ✓ updates liquidation loan-to-value ratio
    platform fees
      ✓ reverts when platform fees sum up to less than one
      ✓ reverts when platform fees sum up to more than one
      ✓ updates platform fees
  stabilization fund
    ✓ updates rebalancing threshold
  exchange fund
    ✓ updates minimum liquidity
    ✓ updates servicing threshold

Exchange Fund
  governed functions
    ✓ lists and unlists depositable token (47ms)
    ✓ updates slippage tolerance
    ✓ updates swap deadline
    ✓ service (stableCoin=>token) and need addLiquidity (67ms)
    ✓ service (stableCoin=>token) and not need addLiquidity (49ms)
    ✓ service (token=>stableCoin) and need addLiquidity (83ms)
    ✓ service (token=>stableCoin) and not need addLiquidity (68ms)

Math
  ✓ sqrt
  ✓ fpsqrt
  ✓ abs
```

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	PASSED
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	PASSED
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	PASSED
SWC-133	Hash Collisions With Multiple Variable Length	CWE-294: Authentication Bypass by Capture-replay	PASSED
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	PASSED
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	NOT PASSED
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	PASSED
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	PASSED
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	PASSED

SWC-127	Arbitrary Jump with Function Type	CWE-695: Use of Low-Level Functionality	PASSED
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	PASSED
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	PASSED
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	PASSED
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	PASSED
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	PASSED
SWC-120	Weak Sources of Randomness from Chain	CWE-330: Use of Insufficiently Random Values	PASSED
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	NOT PASSED
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	PASSED
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	PASSED

SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	**PASSED**
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	**PASSED**
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	**PASSED**
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	**PASSED**
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	**PASSED**
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	**PASSED**
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	**PASSED**
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	**PASSED**
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	**PASSED**
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	**PASSED**
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	**PASSED**

SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	PASSED
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	NOT PASSED
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	PASSED
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	PASSED
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	PASSED
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

Solid Proofed

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY