

Smart Contract Security Audit V1

Liquidity Manager Smart Contract Audit

Jun 27, 2025



<https://saferico.com/>

business@saferico.com

https://t.me/SFI_ANN

—

Table of Contents

Table of Contents

Background

Project Information

Smart Contract Information

Executive Summary

File and Function Level Report

File in Scope:

Issues Checking Status

SWC Attack Analysis

Severity Definitions

Audit Findings

Automatic testing

Testing proves

Inheritance graph

Call graph

Source lines

Risk level

Source units in scope

Capabilities

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Platform:** Binance Smart Chain
- **Name:** LiquidityManager
- **Language :** Solidity
- **Contract Address:** 0xc04b66d590218252b3cd5a42183cf05393cdb6e9, 0x5a36f36d7387acd2d8c7e8a35372f20cb6910d12
- **Code Source:** <https://github.com/TerraDharitri/drt-pREWA/tree/main/contracts>

LIQUIDITYMANAGER:

pREWA LIQUIDITY ON DEX

A **secure**, upgradable contract for managing pREWA token pair liquidity on a DEX (e.g. PancakeSwap), integrates with AccessControl, EmergencyController, Oracles, PriceGuard, PriceProtection.

Author Rewa

License MIT

Solidity version 0.8,28



DATA STORAGE

PairInfo struct

pairAddress	active
tokenAddress	active.

Mappings and variables

`_pairs` mapping bytes2 → Lpinfo
`_isRegisteredAndActiveLpToken`
`_pairRegistrationInProgress` bool
`_pREWATokenAddress`
`_routerAddress`
`_factoryAddress`
`_slippageTolerance`
`pendingBNBRefunds` (address → uint256)
`maxDeadlineOffset` (uint256)

KEY OPERATIONS

For tracking liquidity actions updates.



addLiquidity

addLine

EVENT EMISSIONS

- for tracking liquidity actions
- Price protection Price guard functionality
- Price protection PriceGuard
- BNB refund handling Error handling

KEY OPERATIONS

For tracking liquidity actions.

addLiquidity (otherToken, pREWAAmountDesired, otherAmountDesired, pREWAMin, deadline)



removeLiquidity, removeLiquidity, illiquidity



HOW IT WORKS

- Admin initializes contract with `initializable`, `router`, `accessControl`, `emergency`

Executive Summary

According to our assessment, the customer's solidity smart contract is **Well-Secured**.

Well Secured	✓
Secured	
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 1 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

LiquidityManager.sol

Audit Score:

100% secure



File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
LiquidityManager.sol	1fbd84a1081ddfd42a4cecf800c9bc2d990bd753	0xc04b66d590218252b3cd5a42183cf05393cdb6e9, 0x5a36f36d7387acd2d8c7e8a35372f20cb6910d12

- Contract: LiquidityManager
- Inherit: Initializable, ReentrancyGuardUpgradeable, LiquidityManagerStorage, ILiquidityManager, EmergencyAwareBase
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
accessControl	✓	Read / public	Passed
getLPTokenAddress	✓	Read / public	Passed
checkEmergencyState	✓	Read / public	Passed
emergencyController	✓	Read / public	Passed
getEmergencyController	✓	Read / public	Passed
isEmergencyPaused	✓	Read / public	Passed
getPairInfo	✓	Read / public	Passed
maxDeadlineOffset	✓	Read / public	Passed
oracleIntegration	✓	Read / public	Passed
pendingBNBRefunds	✓	Read / public	Passed
priceGuard	✓	Read / public	Passed
addLiquidityBNB	✓	Write / payable	Passed
renounceOwnership	✓	Write / public	Passed
transferOwnership	✓	Write / public	Passed

addLiquidity	✓	Write / public	Passed
initialize	✓	Write / public	Passed
approve	✓	Write / public	Passed
recoverFailedBNBRefund	✓	Write / public	Passed
emergencyShutdown	✓	Write / public	Passed
recoverTokens	✓	Write / public	Passed
registerPair	✓	Write / public	Passed
removeLiquidityBNB	✓	Write / public	Passed
removeLiquidity	✓	Write / public	Passed
setMaxDeadlineOffset	✓	Write / public	Passed
setPairStatus	✓	Write / public	Passed
setRouterAddress	✓	Write / public	Passed
setSlippageTolerance	✓	Write / public	Passed
updateOracleIntegration	✓	Write / public	Passed
updatePriceGuard	✓	Write / public	Passed
setEmergencyController	✓	Write / public	Passed

Issues Checking Status

SWC Attack Analysis

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) for more info check

<https://swcregistry.io/>

No.	Issue Description	Checking Status
136	Unencrypted Private Data On-Chain	Passed
135	Code With No Effects	Passed
134	Message call with hardcoded gas amount	Passed
133	Hash Collisions With Multiple Variable Length Arguments	Passed
132	Unexpected Ether balance	Passed
131	Presence of unused variables	Passed
130	Right-To-Left-Override control character (U+202E)	Passed
129	Typographical Error	Passed
128	DoS with block gas limit.	Passed
127	Arbitrary Jump with Function Type Variable	Passed
126	Insufficient Gas Griefing	Passed
125	Incorrect Inheritance Order	Passed
124	Write to Arbitrary Storage Location	Passed
123	Requirement Violation	Passed
122	Lack of Proper Signature Verification	Passed
121	Missing Protection against Signature Replay Attacks	Passed
120	Weak Sources of Randomness from Chain Attributes	Passed
119	Shadowing State Variables	Passed

118	Incorrect Constructor Name	Passed
117	Signature Malleability	Passed
116	Block values as a proxy for time	Not Passed
115	Authorization through tx.origin	Passed
114	Transaction Order Dependence	Passed
113	DoS with Failed Call	Passed
112	Delegatecall to Untrusted Callee	Passed
111	Use of Deprecated Solidity Functions	Passed
110	Assert Violation	Passed
109	Uninitialized Storage Pointer	Passed
108	State Variable Default Visibility	Passed
107	Reentrancy	Passed
106	Unprotected SELFDESTRUCT Instruction	Passed
105	Unprotected Ether Withdrawal	Passed
104	Unchecked Call Return Value	Passed
103	Floating Pragma	Passed
102	Outdated Compiler Version	Passed
101	Integer Overflow and Underflow	Passed
100	Function Default Visibility	Passed

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens 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 functions
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
Note	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Audit Findings

Critical:

No Critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found.

Medium:

No Medium severity vulnerabilities were found.

Low:

Zero Address Checks in Modifiers

Description

The `onlyAdminRole` and `onlyParameterRole` modifiers check `if (address(accessControl) == address(0)) revert LM_AccessControlZero();`. This check is redundant after `initialize` is called and `accessControl` is set to a non-zero address.

Recommendation

Remove these redundant checks from the modifiers. The `initialize` function already ensures these addresses are valid contracts. If the `accessControl` reference ever becomes `address(0)` after initialization (e.g., due to an upgrade bug or self-destruct of the `AccessControl` contract), it indicates a deeper, system-level issue that would likely need a full system re-deployment or emergency upgrade.

Status: **Acknowledged**.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

Use of `block.timestamp` for comparisons

The value of `block.timestamp` can be manipulated by the miner. And conditions with strict equality is difficult to achieve - `block.timestamp`.

```
if (tokenAddress_ == address(0)) revert Vesting_TokenZero();
    if (beneficiaryAddress_ == address(0)) revert
Vesting_BeneficiaryZero();
    if (initialOwnerAddress_ == address(0)) revert
Vesting_OwnerZeroV();
```

```
        if (durationValue_ == 0) revert Vesting_DurationZero();
        if (totalVestingAmount_ == 0) revert Vesting_AmountZeroV();
        if (cliffDurationValue_ > durationValue_) revert
Vesting_CliffLongerThanDuration();
        if (startTimeValue_ < block.timestamp && startTimeValue_ !=
0) revert Vesting_StartTimeInvalid();
```

Recommendation

Avoid use of `block.timestamp`.

Automatic Testing

1- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

☒ Select all ☒ Autorun Run

Security

☒ Select Security

- ☒ **Transaction origin:**
'tx.origin' used
- ☒ **Check-effects-interaction:**
Potential reentrancy bugs
- ☒ **Inline assembly:**
Inline assembly used
- ☒ **Block timestamp:**
Can be influenced by miners
- ☒ **Low level calls:**
Should only be used by experienced devs
- ☒ **Block hash:**
Can be influenced by miners
- ☒ **Selfdestruct:**
Contracts using destructured contract can be broken

Gas & Economy

☒ Select Gas & Economy

- ☒ **Gas costs:**
Too high gas requirement of functions
- ☒ **This on local calls:**
Invocation of local functions via 'this'
- ☒ **Delete dynamic array:**
Use require/assert to ensure complete deletion
- ☒ **For loop over dynamic array:**
Iterations depend on dynamic array's size
- ☒ **Ether transfer in loop:**
Transferring Ether in a for/while/do-while loop

SOLIDITY STATIC ANALYSIS

ERC

☒ Select ERC

- ☒ **ERC20:**
'decimals' should be 'uint8'

Miscellaneous

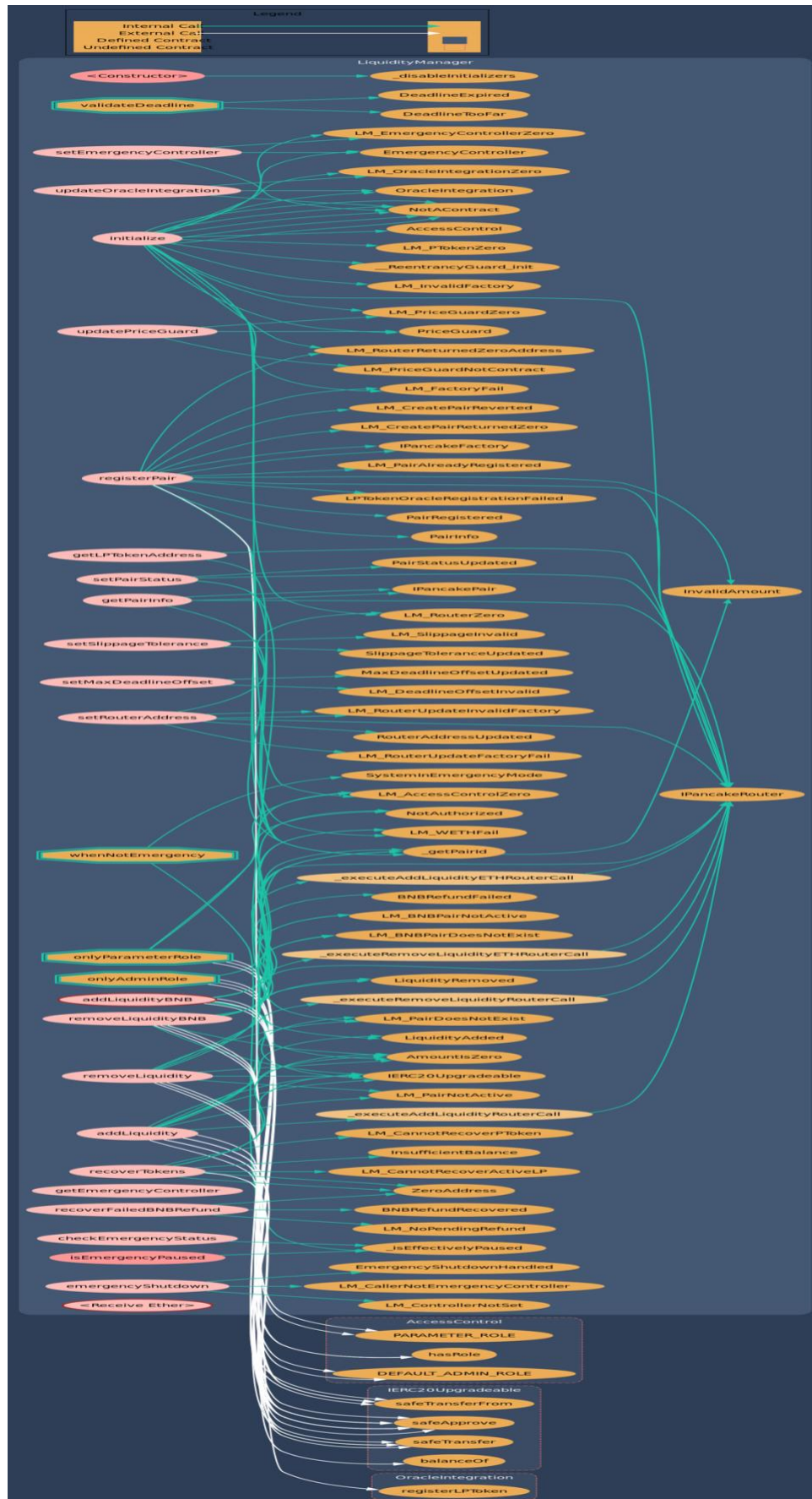
☒ Select Miscellaneous

- ☒ **Constant/View/Pure functions:**
Potentially constant/view/pure functions
- ☒ **Similar variable names:**
Variable names are too similar
- ☒ **No return:**
Function with 'returns' not returning
- ☒ **Guard conditions:**
Ensure appropriate use of require/assert
- ☒ **Result not used:**
The result of an operation not used
- ☒ **String length:**
Bytes length != String length
- ☒ **Delete from dynamic array:**
'delete' leaves a gap in array
- ☒ **Data truncated:**
Division on int/uint values truncates the result

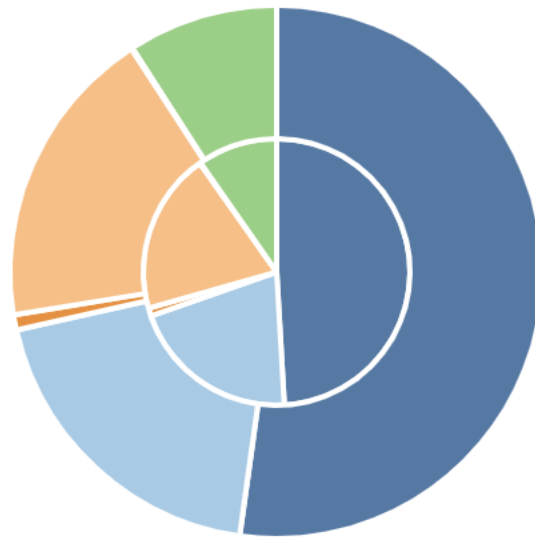
2- Inheritance graph



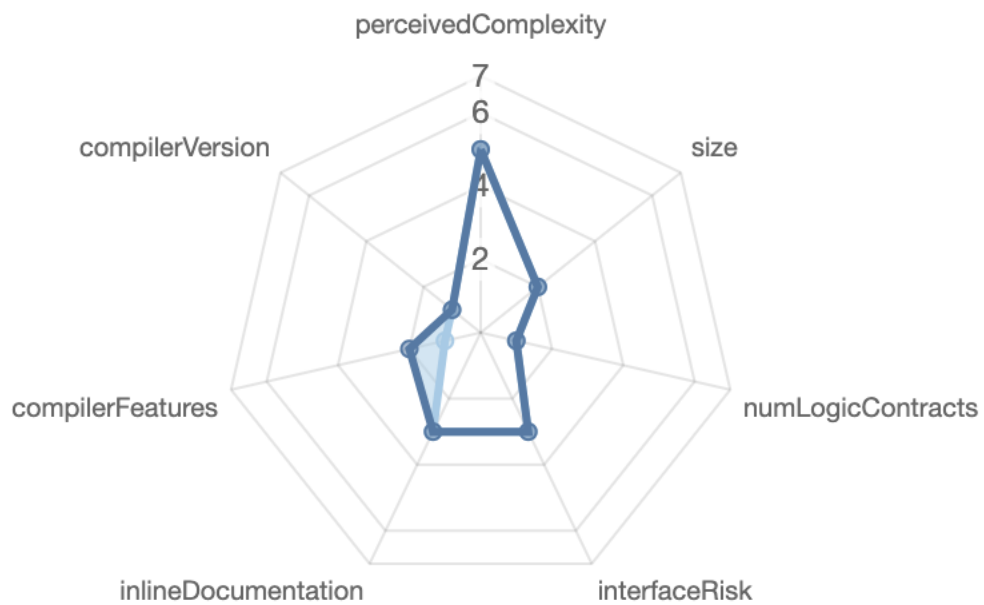
3- Call graph



Source lines



Risk level



Source units in scope

Source Units in Scope

Source Units Analyzed: 1
Source Units in Scope: 1 (100%)

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/liquidity/LiquidityManager.sol	1	_____	797	736	456	191	471	
	Totals	1	_____	797	736	456	191	471	

Legend: [-]

- **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, ...)

Capabilities

Components

 Contracts	 Libraries	 Interfaces	 Abstract
1	0	0	0

Exposed Functions

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

 Public	 Payable
22	2

External	Internal	Private	Pure	View
21	24	1	0	6

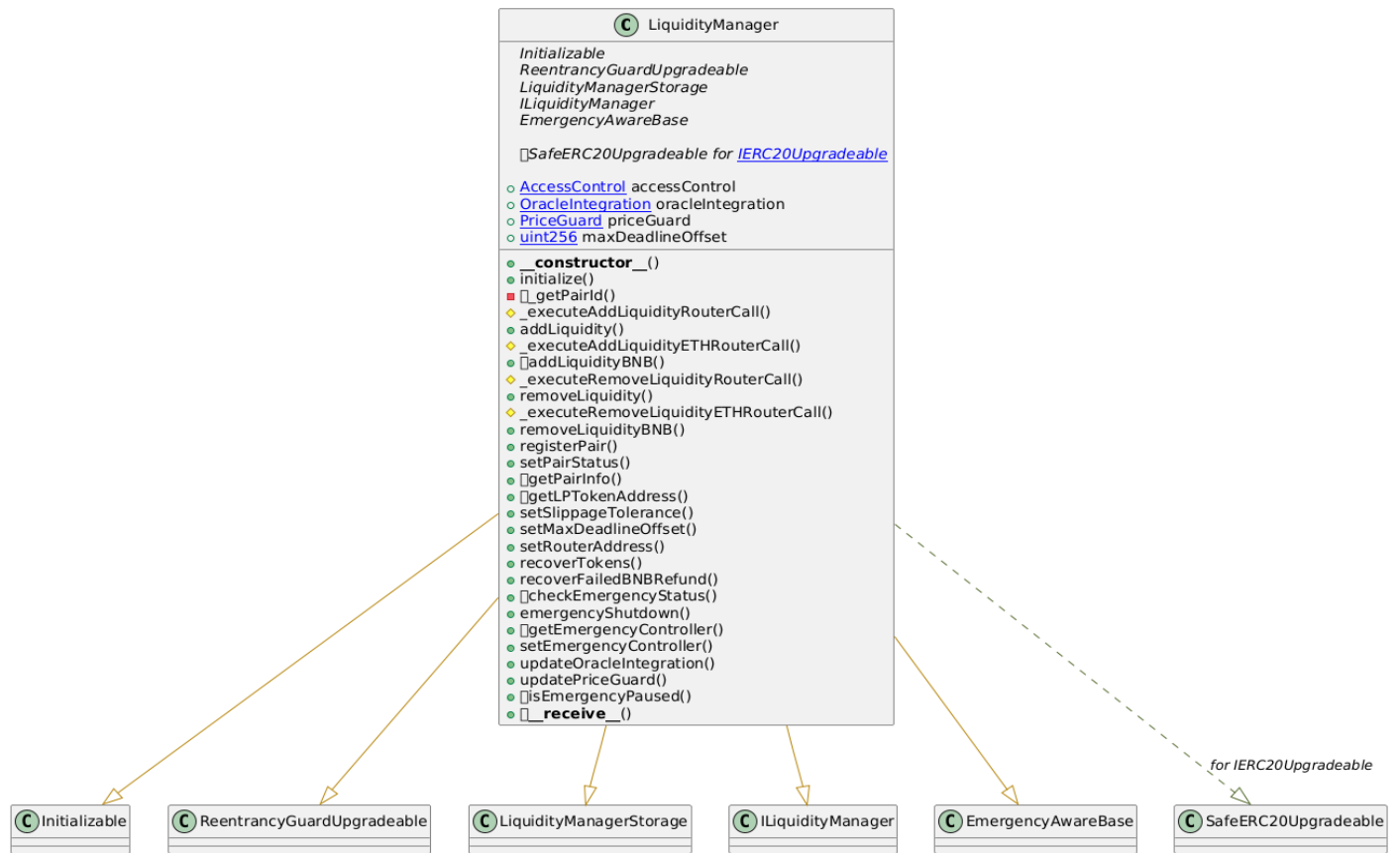
StateVariables

Total	 Public
4	4

Capabilities

Solidity Versions observed	 Experimental Features	 Can Receive Funds	 Uses Assembly	 Has Destroyable Contracts	
0.8.28		yes	yes (9 asm blocks)		
 Transfers ETH	 Low-Level Calls	 DelegateCall	 Uses Hash Functions	 ECRecover	 New/Create/Create2
			yes		

Unified Modeling Language (UML)



Functions signature

Function Name	Sighash	Function Signature
-----	-----	-----
initialize	cc2a9a5b	initialize(address,address,address,address,address,address)
addLiquidity	2563e426	addLiquidity(address,uint256,uint256,uint256,uint256)
addLiquidityBNB	b1e12e58	addLiquidityBNB(uint256,uint256,uint256,uint256)
removeLiquidity	13928082	removeLiquidity(address,uint256,uint256,uint256,uint256)
removeLiquidityBNB	48790d0d	removeLiquidityBNB(uint256,uint256,uint256,uint256)
registerPair	dc314b44	registerPair(address)
setPairStatus	5a0d1229	setPairStatus(address,bool)
getPairInfo	d4ff73f7	getPairInfo(address)
getLPTokenAddress	4939ea4d	getLPTokenAddress(address)
setSlippageTolerance	117dalee	setSlippageTolerance(uint256)
setMaxDeadlineOffset	9ba5b5a1	setMaxDeadlineOffset(uint256)
setRouterAddress	41cb87fc	setRouterAddress(address)
recoverTokens	61b0a56e	recoverTokens(address,uint256,address)
recoverFailedBNBRefund	c053ed44	recoverFailedBNBRefund(address)
checkEmergencyStatus	8aed668a	checkEmergencyStatus(bytes4)
emergencyShutdown	eb1676c1	emergencyShutdown(uint8)
getEmergencyController	c993cc9d	getEmergencyController()
setEmergencyController	6ca7dc89	setEmergencyController(address)
updateOracleIntegration	4b9a01b5	updateOracleIntegration(address)
updatePriceGuard	c299e0f7	updatePriceGuard(address)
isEmergencyPaused	290d10c4	isEmergencyPaused()

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/LiquidityManager.sol	1fbd84a1081ddfd42a4cecf800c9bc2d990bd753
/Users/macbook/Desktop/drt-pREWA/contracts/core/ContractRegistry.sol	6557f2c40d50caa85b0321c0811ecff0147d5502
/Users/macbook/Desktop/drt-pREWA/contracts/access/AccessControl.sol	40163fa249f10365c03cf1fa9ddd26e2f6eb1005
/Users/macbook/Desktop/drt-pREWA/contracts/access/storage/AccessControlStorage.sol	2a1f4c3d6956a89011b090a62b15254b1d720d65
/Users/macbook/Desktop/drt-pREWA/contracts/access/interfaces/IAccessControl.sol	540ec54eaadelc05a650af086ebf959654ec6322
/Users/macbook/Desktop/drt-pREWA/contracts/libraries/Errors.sol	0974f6f49e3b655fa93a2792154f1b506ec03c74
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/storage/LiquidityManagerStorage.sol	968913fd820da108d5f211c9f0cb230147af0376
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/interfaces/ILiquidityManager.sol	2385fc3435b6af45c585ef9025ddad70d12d2a65
/Users/macbook/Desktop/drt-pREWA/contracts/interfaces/IEmergencyAware.sol	be6e0a371b0a5f62d5c7dc06e5c0c4121dcc7ca1
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/interfaces/IPancakeRouter.sol	1288ced7b11112ee24339d5e8a0b80d1a84489f4
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/interfaces/IPancakeFactory.sol	54ad4aefc2d331f1055d7c917dd66f70ab9db126
/Users/macbook/Desktop/drt-pREWA/contracts/liquidity/interfaces/IPancakePair.sol	9a5b08f2b661520cc40a570df0ac173b4a9a6c28
/Users/macbook/Desktop/drt-pREWA/contracts/controllers/EmergencyController.sol	8e73765eaa98f95db8f5eb7a40cd13b0dc845266
/Users/macbook/Desktop/drt-pREWA/contracts/interfaces/IEmergencyController.sol	40bc149ec8a1da85c0e4c6170b06f3a0f3b77f3b
/Users/macbook/Desktop/drt-pREWA/contracts/controllers/EmergencyTimelockController.sol	94735e88ad7f750ead488b7ad618b1486cbdb483
/Users/macbook/Desktop/drt-pREWA/contracts/libraries/Constants.sol	62835a8af9d7ab21c0cad9ca03abff6cf4e8e68f

```

| /Users/macbook/Desktop/drt-pREWA/contracts/oracle/OracleIntegration.sol
| 56835e91c3d168cbc4e565e742c575c2117bb3bb |
| /Users/macbook/Desktop/drt-
pREWA/contracts/interfaces/AggregatorV3Interface.sol |
008fa0a63980b32bdecef492c7d1ae5b441a5c3 |
| /Users/macbook/Desktop/drt-pREWA/contracts/libraries/DecimalMath.sol |
1f5b0a2a73498dc5cb877c08aea3c971646fb4a5 |
| /Users/macbook/Desktop/drt-pREWA/contracts/security/PriceGuard.sol |
28b3b3389997bc046b915bbb7a579b615ca0597b |
| /Users/macbook/Desktop/drt-pREWA/contracts/utils/EmergencyAwareBase.sol
| 528be04847c5d3cf03edd961c878359eb3cafc6a |

```

Contracts Description Table

Contract	Type	Bases	
:-----: :-----: :-----: :-----:			
: :-----:			
L	**Function Name**	**Visibility**	**Mutability**
Modifiers			
LiquidityManager Implementation Initializable, ReentrancyGuardUpgradeable, LiquidityManagerStorage, ILiquidityManager, EmergencyAwareBase			
L	<Constructor>	Public !	NO !
L	initialize	External !	initializer
L	_getPairId	Private 🔒	
L	_executeAddLiquidityRouterCall	Internal 🔒	
L	addLiquidity	External !	nonReentrant whenNotEmergency validateDeadline
L	_executeAddLiquidityETHRouterCall	Internal 🔒	
L	addLiquidityBNB	External !	nonReentrant whenNotEmergency validateDeadline
L	_executeRemoveLiquidityRouterCall	Internal 🔒	
L	removeLiquidity	External !	nonReentrant whenNotEmergency validateDeadline
L	_executeRemoveLiquidityETHRouterCall	Internal 🔒	
L	removeLiquidityBNB	External !	nonReentrant whenNotEmergency validateDeadline
L	registerPair	External !	onlyParameterRole
L	setPairStatus	External !	onlyParameterRole nonReentrant
L	getPairInfo	External !	NO !
L	getLPTokenAddress	External !	NO !
L	setSlippageTolerance	External !	onlyParameterRole nonReentrant
L	setMaxDeadlineOffset	External !	onlyParameterRole nonReentrant
L	setRouterAddress	External !	onlyAdminRole nonReentrant

```

| L | recoverTokens | External ! |  | onlyAdminRole nonReentrant |
| L | recoverFailedBNBRefund | External ! |  | onlyAdminRole
nonReentrant |
| L | checkEmergencyStatus | External ! | |NO! |
| L | emergencyShutdown | External ! |  |NO! |
| L | getEmergencyController | External ! | |NO! |
| L | setEmergencyController | External ! |  | onlyAdminRole |
| L | updateOracleIntegration | External ! |  | onlyAdminRole |
| L | updatePriceGuard | External ! |  | onlyAdminRole |
| L | isEmergencyPaused | Public ! | |NO! |
| L | <Receive Ether> | External ! |  |NO! |
|||||
| **ContractRegistry** | Implementation | Initializable,
OwnableUpgradeable, ReentrancyGuardUpgradeable |||
| L | <Constructor> | Public ! |  |NO! |
| L | initialize | External ! |  | initializer |
| L | registerContract | External ! |  | onlyParameterRole
nonReentrant |
| L | updateContract | External ! |  | onlyParameterRole nonReentrant
|
| L | removeContract | External ! |  | onlyParameterRole nonReentrant
|
| L | setContractActive | External ! |  | onlyParameterRole
nonReentrant |
| L | getContractAddress | External ! | |NO! |
| L | getContractName | External ! | |NO! |
| L | getContractInfo | External ! | |NO! |
| L | listContracts | External ! | |NO! |
| L | getContractsByType | External ! | |NO! |
| L | contractExists | External ! | |NO! |
| L | getContractCount | External ! | |NO! |
|||||
| **AccessControl** | Implementation | Initializable,
AccessControlStorage, IAccessControl |||
| L | <Constructor> | Public ! |  |NO! |
| L | initialize | External ! |  | initializer |
| L | hasRole | External ! | |NO! |
| L | getRoleAdmin | External ! | |NO! |
| L | grantRole | External ! |  |NO! |
| L | revokeRole | External ! |  |NO! |
| L | renounceRole | External ! |  |NO! |
| L | setRoleAdmin | External ! |  |NO! |
| L | getRoleMember | External ! | |NO! |
| L | getRoleMemberCount | External ! | |NO! |
| L | getRoleMembersPaginated | External ! | |NO! |
| L | _grantRole | Internal  |  | |
| L | _revokeRole | Internal  |  | |
| L | _setRoleAdmin | Internal  |  | |
|||||
| **AccessControlStorage** | Implementation | |||
|||||

```

```

|  **IAccessControl** | Interface |   |||
|  L | hasRole | External ! |   |NO! |
|  L | getRoleAdmin | External ! |   |NO! |
|  L | grantRole | External ! |   |NO! |
|  L | revokeRole | External ! |   |NO! |
|  L | renounceRole | External ! |   |NO! |
|  L | getRoleMember | External ! |   |NO! |
|  L | getRoleMemberCount | External ! |   |NO! |
|  L | getRoleMembersPaginated | External ! |   |NO! |
|  |||||
|  **LiquidityManagerStorage** | Implementation |   |||
|  |||||
|  **ILiquidityManager** | Interface | IEmergencyAware |||
|  L | addLiquidity | External ! |   |NO! |
|  L | addLiquidityBNB | External ! |   |NO! |
|  L | removeLiquidity | External ! |   |NO! |
|  L | removeLiquidityBNB | External ! |   |NO! |
|  L | registerPair | External ! |   |NO! |
|  L | setPairStatus | External ! |   |NO! |
|  L | getPairInfo | External ! |   |NO! |
|  L | getLPTokenAddress | External ! |   |NO! |
|  L | setSlippageTolerance | External ! |   |NO! |
|  L | setMaxDeadlineOffset | External ! |   |NO! |
|  L | setRouterAddress | External ! |   |NO! |
|  L | recoverTokens | External ! |   |NO! |
|  L | recoverFailedBNBRefund | External ! |   |NO! |
|  |||||
|  **IEmergencyAware** | Interface |   |||
|  L | checkEmergencyStatus | External ! |   |NO! |
|  L | emergencyShutdown | External ! |   |NO! |
|  L | getEmergencyController | External ! |   |NO! |
|  L | setEmergencyController | External ! |   |NO! |
|  L | isEmergencyPaused | External ! |   |NO! |
|  |||||
|  **IPancakeRouter** | Interface |   |||
|  L | factory | External ! |   |NO! |
|  L | weth | External ! |   |NO! |
|  L | addLiquidity | External ! |   |NO! |
|  L | addLiquidityETH | External ! |   |NO! |
|  L | removeLiquidity | External ! |   |NO! |
|  L | removeLiquidityETH | External ! |   |NO! |
|  L | swapExactTokensForTokens | External ! |   |NO! |
|  L | swapTokensForExactTokens | External ! |   |NO! |
|  L | swapExactETHForTokens | External ! |   |NO! |
|  L | swapTokensForExactETH | External ! |   |NO! |
|  L | swapExactTokensForETH | External ! |   |NO! |
|  L | swapETHForExactTokens | External ! |   |NO! |
|  L | getAmountsOut | External ! |   |NO! |
|  L | getAmountsIn | External ! |   |NO! |
|  |||||
|  **IPancakeFactory** | Interface |   |||

```

```

| L | getPair | External ! | | NO! |
| L | createPair | External ! | ● NO! |
| L | setFeeTo | External ! | ● NO! |
| L | setFeeToSetter | External ! | ● NO! |
| L | feeTo | External ! | | NO! |
| L | feeToSetter | External ! | | NO! |
| L | allPairsLength | External ! | | NO! |
| L | allPairs | External ! | | NO! |
|||||
| **IPancakePair** | Interface | |||
| L | token0 | External ! | | NO! |
| L | token1 | External ! | | NO! |
| L | getReserves | External ! | | NO! |
| L | price0CumulativeLast | External ! | | NO! |
| L | price1CumulativeLast | External ! | | NO! |
| L | totalSupply | External ! | | NO! |
| L | balanceOf | External ! | | NO! |
| L | allowance | External ! | | NO! |
| L | approve | External ! | ● NO! |
| L | transfer | External ! | ● NO! |
| L | transferFrom | External ! | ● NO! |
| L | swap | External ! | ● NO! |
| L | mint | External ! | ● NO! |
| L | burn | External ! | ● NO! |
|||||
| **EmergencyController** | Implementation | Initializable,
ReentrancyGuardUpgradeable, IEmergencyController |||
| L | <Constructor> | Public ! | ● NO! |
| L | initialize | External ! | ● initializer |
| L | setRequiredApprovals | External ! | ● onlyAdminRole |
| L | setLevel3TimelockDuration | External ! | ● onlyAdminRole |
| L | setRecoveryAdminAddress | External ! | ● onlyAdminRole |
| L | approveLevel3Emergency | External ! | ● onlyEmergencyRole
nonReentrant |
| L | cancelLevel3Emergency | External ! | ● onlyEmergencyRole
nonReentrant |
| L | executeLevel3Emergency | External ! | ● onlyEmergencyRole
nonReentrant |
| L | setEmergencyLevel | External ! | ● onlyEmergencyRole
nonReentrant |
| L | enableEmergencyWithdrawal | External ! | ● onlyEmergencyRole
nonReentrant |
| L | pauseSystem | External ! | ● onlyPauserRole nonReentrant |
| L | unpauseSystem | External ! | ● onlyPauserRole nonReentrant |
| L | recoverTokens | External ! | ● onlyEmergencyRole nonReentrant |
|
| L | registerEmergencyAwareContract | External ! | ●
onlyEmergencyRole |
| L | removeEmergencyAwareContract | External ! | ●
onlyEmergencyRole |
| L | processEmergencyForContract | External ! | ● nonReentrant |

```

```

| L | updateFunctionRestriction | External ! | 🔒 | onlyEmergencyRole |
| L | getEmergencyLevel | External ! | |NO! |
| L | getEmergencyWithdrawalSettings | External ! | |NO! |
| L | isSystemPaused | External ! | |NO! |
| L | getEmergencyAwareContractsPaginated | External ! | |NO! |
| L | getApprovalStatus | External ! | |NO! |
| L | isFunctionRestricted | External ! | |NO! |
| L | _resetApprovals | Internal 🔓 | 🔒 | |
| | | | |
| **IEmergencyController** | Interface | | |
| L | setEmergencyLevel | External ! | 🔒 |NO! |
| L | enableEmergencyWithdrawal | External ! | 🔒 |NO! |
| L | pauseSystem | External ! | 🔒 |NO! |
| L | unpauseSystem | External ! | 🔒 |NO! |
| L | recoverTokens | External ! | 🔒 |NO! |
| L | getEmergencyLevel | External ! | |NO! |
| L | getEmergencyWithdrawalSettings | External ! | |NO! |
| L | isSystemPaused | External ! | |NO! |
| L | getEmergencyAwareContractsPaginated | External ! | |NO! |
| | | | |
| **EmergencyTimelockController** | Implementation | Initializable,
ReentrancyGuardUpgradeable | | |
| L | <Constructor> | Public ! | 🔒 |NO! |
| L | initialize | External ! | 🔒 | initializer |
| L | setAllowedTarget | External ! | 🔒 | onlyEmergencyRole |
| L | setAllowedFunctionSelector | External ! | 🔒 | onlyEmergencyRole |
|
| L | proposeEmergencyAction | External ! | 🔒 | onlyEmergencyRole
nonReentrant |
| L | executeEmergencyAction | External ! | 🔒 | onlyEmergencyRole
nonReentrant |
| L | cancelEmergencyAction | External ! | 🔒 | onlyEmergencyRole
nonReentrant |
| L | updateTimelockDuration | External ! | 🔒 | onlyEmergencyRole |
| L | getActionStatus | External ! | |NO! |
| L | getAllActionIds | External ! | |NO! |
| L | getActionDetails | External ! | |NO! |
| L | isFunctionSelectorAllowed | External ! | |NO! |
| L | isTargetAllowed | External ! | |NO! |
| | | | |
| **Constants** | Library | | |
| | | | |
| **OracleIntegration** | Implementation | Initializable,
OwnableUpgradeable, ReentrancyGuardUpgradeable | | |
| L | <Constructor> | Public ! | 🔒 |NO! |
| L | initialize | External ! | 🔒 | initializer |
| L | setLiquidityManagerAddress | External ! | 🔒 | onlyOwner |
| L | setPriceFeed | External ! | 🔒 | onlyOwner nonReentrant |
| L | registerLPToken | External ! | 🔒 | onlyLiquidityManagerOrOwner
nonReentrant |
| L | setFallbackPrice | External ! | 🔒 | onlyOwner nonReentrant |

```

```

| L | setStalenessThreshold | External ! | 🔒 | onlyOwner nonReentrant
|
| L | setMinAcceptablePrice | External ! | 🔒 | onlyOwner nonReentrant
|
| L | getTokenPrice | Public ! | |NO! |
| L | fetchAndReportTokenPrice | External ! | 🔒 | nonReentrant |
| L | getLPTokenValue | External ! | |NO! |
| L | getLPTokenValueAlternative | External ! | |NO! |
| L | validatePriceAgainstOracle | External ! | |NO! |
| L | _getPriceFromOracle | Private 🔒 | | |
| L | _fetchAndReportPriceFromOracle | Private 🔒 | 🔒 | |
| L | _verifyPriceFeed | Private 🔒 | | |
| L | _getPriceFeedDecimals | Private 🔒 | | |
| L | convertToStandardPrecision | Public ! | |NO! |
| L | _getTokenDecimals | Private 🔒 | | |
| L | getPriceFeedInfo | External ! | |NO! |
| L | getLPTokenInfo | External ! | |NO! |
| L | getStalenessThreshold | External ! | |NO! |
| L | getMinAcceptablePrice | External ! | |NO! |
| |||||
| **AggregatorV3Interface** | Interface | | |
| L | decimals | External ! | |NO! |
| L | description | External ! | |NO! |
| L | version | External ! | |NO! |
| L | latestRoundData | External ! | |NO! |
| L | getRoundData | External ! | |NO! |
| |||||
| **DecimalMath** | Library | | |
| L | mulScaled | Internal 🔒 | | |
| L | divScaled | Internal 🔒 | | |
| L | mulDiv | Internal 🔒 | | |
| L | calculatePercentage | Internal 🔒 | | |
| L | applySlippage | Internal 🔒 | | |
| L | scaleUp | Internal 🔒 | | |
| L | scaleDown | Internal 🔒 | | |
| L | weightedAverage | Internal 🔒 | | |
| L | compound | Internal 🔒 | | |
| L | divRounded | Internal 🔒 | | |
| |||||
| **PriceGuard** | Implementation | Initializable, OwnableUpgradeable,
ReentrancyGuardUpgradeable, IEmergencyAware | |
| L | <Constructor> | Public ! | 🔒 |NO! |
| L | initialize | External ! | 🔒 | initializer |
| L | checkPriceImpact | External ! | 🔒 | nonReentrant
whenNotEffectivelyPaused |
| L | getExpectedPrice | External ! | | whenNotEffectivelyPaused |
| L | validateSlippage | External ! | |NO! |
| L | calculateMinimumOutput | External ! | |NO! |
| L | registerCommitment | External ! | 🔒 | nonReentrant
whenNotEffectivelyPaused |

```

```

| L | verifyCommitment | External ! | 🔒 | nonReentrant
whenNotEffectivelyPaused |
| L | setPriceGuardParameters | External ! | 🔒 | onlyOwner |
| L | setCommitRevealParameters | External ! | 🔒 | onlyOwner |
| L | setMinAcceptablePrice | External ! | 🔒 | onlyOwner |
| L | setOracleIntegration | External ! | 🔒 | onlyOwner |
| L | checkEmergencyStatus | External ! | | NO! |
| L | emergencyShutdown | External ! | 🔒 | NO! |
| L | getEmergencyController | External ! | | NO! |
| L | setEmergencyController | External ! | 🔒 | onlyOwner |
| L | isEmergencyPaused | Public ! | | NO! |
| L | preventFrontRunning | External ! | | NO! |
|||||
| **EmergencyAwareBase** | Implementation | IEmergencyAware |||
| L | _isEffectivelyPaused | Internal 🔒 | | |

```

Legend

Symbol	Meaning
:-----:	-----
🔒	Function can modify state
💰	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract 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.

Security state of the reviewed contract is “Well Secured”.

- ✓ No volatile code.
- ✓ No high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.