Smart Contract Security Audit V1

Chaufr Smart Contract Audit

Jun 13, 2025

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

Table of Contents

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**: Chaufr
- **Language** : Solidity
- **Contract Address**: 0x1dB4f9623605091d7104f3a8fb6A9B034817A5B0
- **Code Source**: https://testnet.bscscan.com/address/0x1dB4f9623605091d7104f3a8fb6A9B034817A5B0#code

ChaufrCoin (CHUFR)

Smart Contract Overview

ChaufrCoin: A secure, fee-based ERC20 token with burn and treasury features - Built on Ethereum V2 integration

Audit

Safence

Audit by SaferICO | 95 Audited by SaferICO

Key Information

Token Name:
ChaufrCoin (CHUFR)

Total Supply
1,000,000,000 CHUFR ((e) +118)

Admin
Deployer-set admin address

Treasury Wallet
Configurable address for fee collection

Router
Uniswap V2 Router for liquidity

How It Works

User initates
transfer (buy/sell/regular)

Contract checks
for fee eligibility and transfer limits

Applies buy/sell fees (if AMM pair involved) and burn (if enabled

Security
ReentrancyGuard for protection
Ownable for admin control
Safe taken/ETH withorawal functions

Core Features

Fee System

- Buy Fee: 1% (configurable, max 5%)
- Sell Fee: 1% (configurable, max 5%)
- Burn Fee: 1% (configurable. max 100% optional)
- Fees sent to Treasury Wallet

Burn Mechanism

- Optional token burning (enabled/disabled by owner)
- Burns takens during buy/sell renabled

Transfer Limits

- 1 transfer per block per EOA (configurable)
- Can be disabled by owner

Events & Admin Controls

Admin
FeesSet
BurnFeesSet
LimitsRemoved
TreasuryWallet
AMMSet

Events
FeesSet
BurnFeesSet
LimitsRemoved
TreasuryWaletSet
AMMSet

Securcy & Integrations
Creates liquidity pair with WETH
Supports AMM trading

Powered by Solidity 0.5.28 | Deployed on

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, 2 low, 0 very low-level issues and 2 note in all solidity files of the contract

The files:

Chaufr.sol

Audit Score:

99% secure

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
Chaufr.sol	3bddc20dec18f65ff00ad0d4bef6c663745d93f3	0x1dB4f9623605091d7104f3a8fb6A9B034817A5B0

- Contract: Chaufr
- Inherit: Erc20, Ownable, ReentrancyGuard
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
admin	✓	Read / public	**Passed**
allowance	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
blockTransferCount	✓	Read / public	**Passed**
burnFee	✓	Read / public	**Passed**
buyfees	✓	Read / public	**Passed**
owner	✓	Read / public	**Passed**
decimals	✓	Read / public	**Passed**
isAMM	✓	Read / public	**Passed**
feeDenominator	✓	Read / public	**Passed**
isBurnallowed	✓	Read / public	**Passed**
isExcludedFromTransferLimits	✓	Read / public	**Passed**
isExcludedFromFee	✓	Read / public	**Passed**
liquityPair	✓	Read / public	**Passed**

maxBuyFee	✓	Read / public	**Passed**
maxBurn	✓	Read / public	**Passed**
maxSellFee	✓	Read / public	**Passed**
name	✓	Read / public	**Passed**
router	✓	Read / public	**Passed**
sellFees	✓	Read / public	**Passed**
symbol	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
transferLimitEnbaled	✓	Read / public	**Passed**
treasuryTokens	✓	Read / public	**Passed**
treasuryWallet	✓	Read / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
decreaseAllowance	✓	Write / public	**Passed**
increaseAllowance	✓	Write / public	**Passed**
approve	✓	Write / public	**Passed**
burn	✓	Write / public	**Passed**
removeLimits	✓	Write / public	**Passed**
setAMM	✓	Write / public	**Passed**
setAddressExcludedFromTransferLimits	✓	Write / public	**Passed**
setBurnFeature	✓	Write / public	**Passed**
setFees	✓	Write / public	**Passed**
setBurnFees	✓	Write / public	**Passed**
transfer	✓	Write / public	**Passed**
setTreasuryWallet	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
setWalletExcludedFromFee	✓	Write / public	**Passed**
withdrawStuckETH	✓	Write / public	**Passed**
withdrawStuckTokens	✓	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	**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:

#admin vs owner() Discrepancy

Description

The constructor parameter `_admin` is used to mint the `totalSupply` (`_mint(admin, totalSupply);`). However, the `Ownable` contract (which `Chaufr` inherits from) sets `owner()` to `msg.sender` of the constructor. This means that by default, the `admin` address will receive the initial supply, but only `msg.sender` (the deployer) will have `onlyOwner` privileges. If the intention was for `_admin` to be the initial owner, `transferOwnership(_admin)` should be called in the constructor.

Recommendation

Clarify the roles.

- Option A (Recommended): Remove the `admin` state variable and make `_initialSupplyRecipient` a constructor argument. Mint the total supply to `_initialSupplyRecipient` (which could be the `owner()` or another designated address). The `owner()` will remain `msg.sender` by default.
- Option B: If `_admin` *is* meant to be the owner, call `transferOwnership(_admin);` in the constructor.

Status: Acknowledged.

#Owner privileges (In the period when the owner isn't renounced)

Description

The owner can change the Fees.
The owner can exclude any address from the fees and limits.

```
function setFees(uint8 buyFee, uint8 sellFee) external onlyOwner {
        require(buyFee <= maxBuyFee, "Buy fee exceeds 5%");
        require(sellFee <= maxSellFee, "Sell fee exceeds 5%");

        buyFees = buyFee;
        sellFees = sellFee;

        emit FeesSet(buyFee, sellFee);
    }

    /// @notice Sets burn fee
    /// @param newBurnFee Burn fee (in basis points, max 5%)
    function setBurnFees(uint8 newBurnFee) external onlyOwner {
        require(newBurnFee <= maxBurnFee, "Burn fee exceeds 5%");

        burnFee = newBurnFee;

        emit BurnFeesSet(newBurnFee);
    }
function setWalletExcludedFromFees(address wallet, bool isExcluded) external
onlyOwner {
        isExcludedFromFee[wallet] = isExcluded;
        emit WalletExcludedFromFees(wallet, isExcluded);
    }

    /// @notice Excludes or includes a wallet from transfer limits
    /// @param wallet Wallet address
    /// @param isExcluded Whether to exclude from transfer limits
    function setAddressExcludedFromTransferLimits(address wallet, bool isExcluded)
external onlyOwner {
        require(wallet != address(0), "Zero address not allowed");
        isExcludedFromTransferLimits[wallet] = isExcluded;
        emit WalletExcludedFromLimits(wallet, isExcluded);
    }
```

Remediation
Make these functions internal in next version or the team should announce the investors before doing anything to give them time if they want to do anything.

P.S: This issue is common to the majority of those smart contracts.
Status: Acknowledged.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Fixed 0xdead Address

Description

`isExcludedFromFee[address(0xdead)] = true;` This hardcoded address is sometimes used as a dummy address or a "burn" address. While common, it ties the contract to a specific convention. If the burn address is configurable, it should be a state variable.

Recommendation

Keep it as is if it's meant to be a fixed exclusion for the common null address. Or, make it configurable by the owner if this is a desired feature.

#No emergency pause function

Description

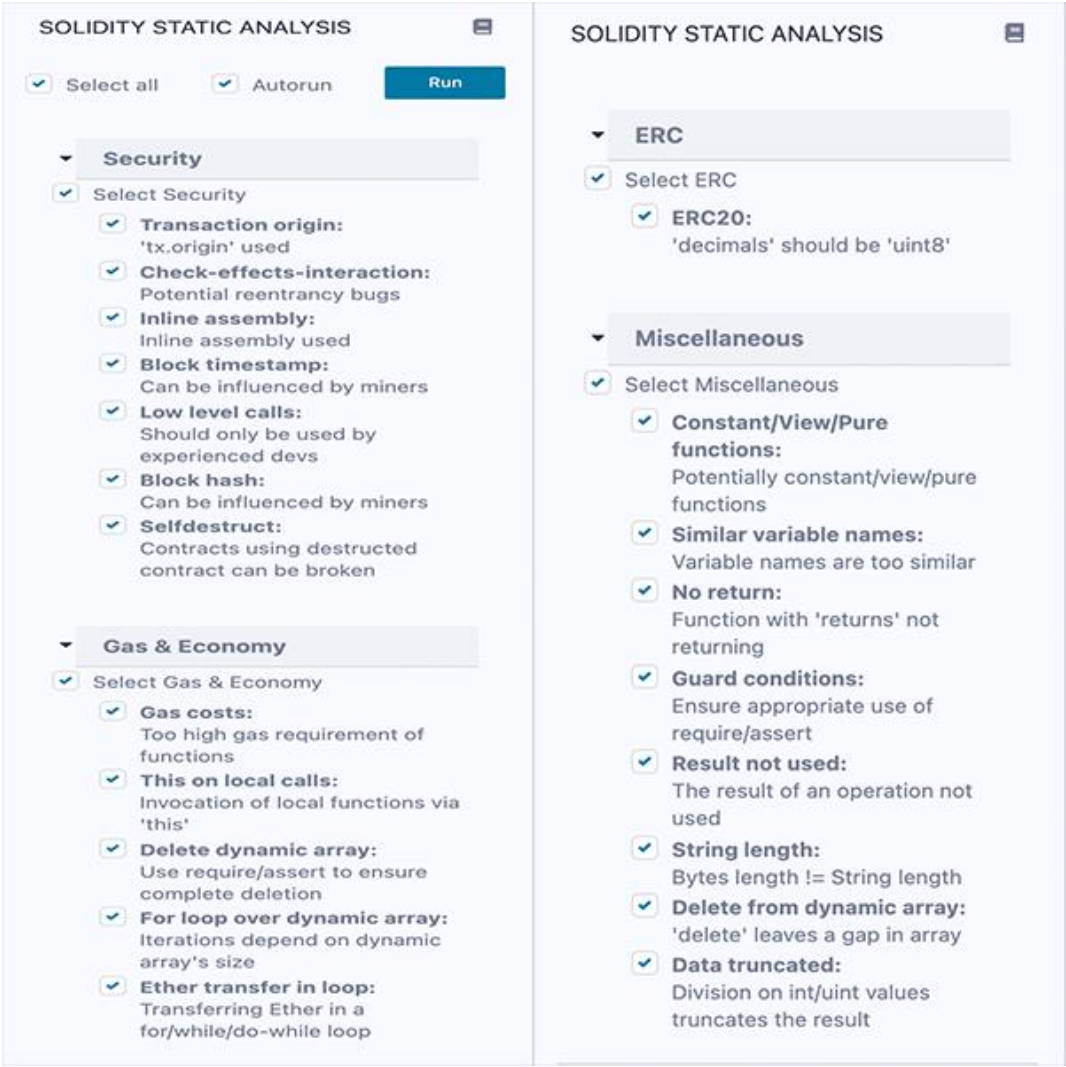
No mechanism exists to pause trading or transfers during an exploit or bug.

Recommendation

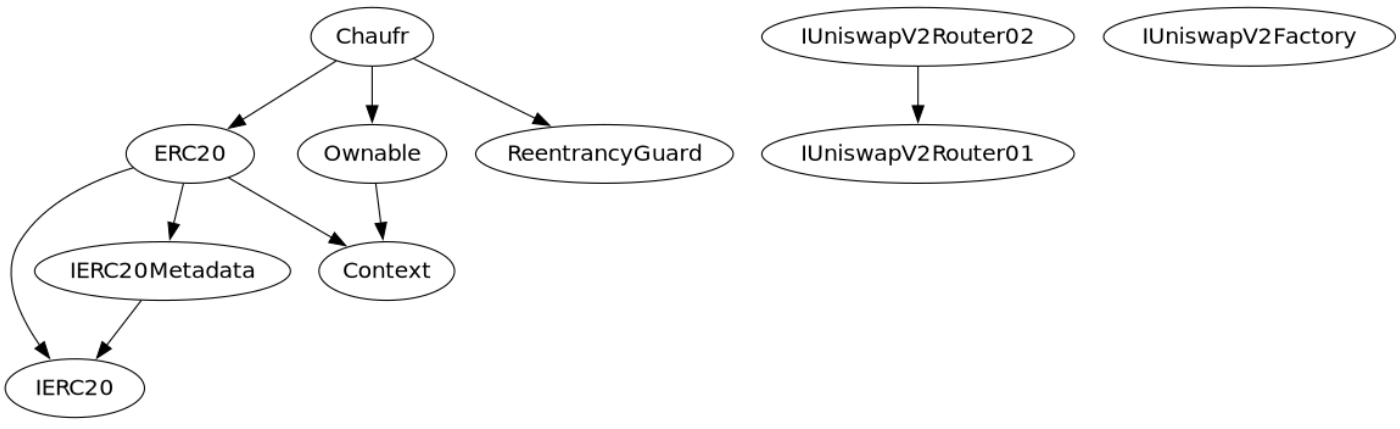
Implement Pausable or a custom pause modifier.

Automatic Testing

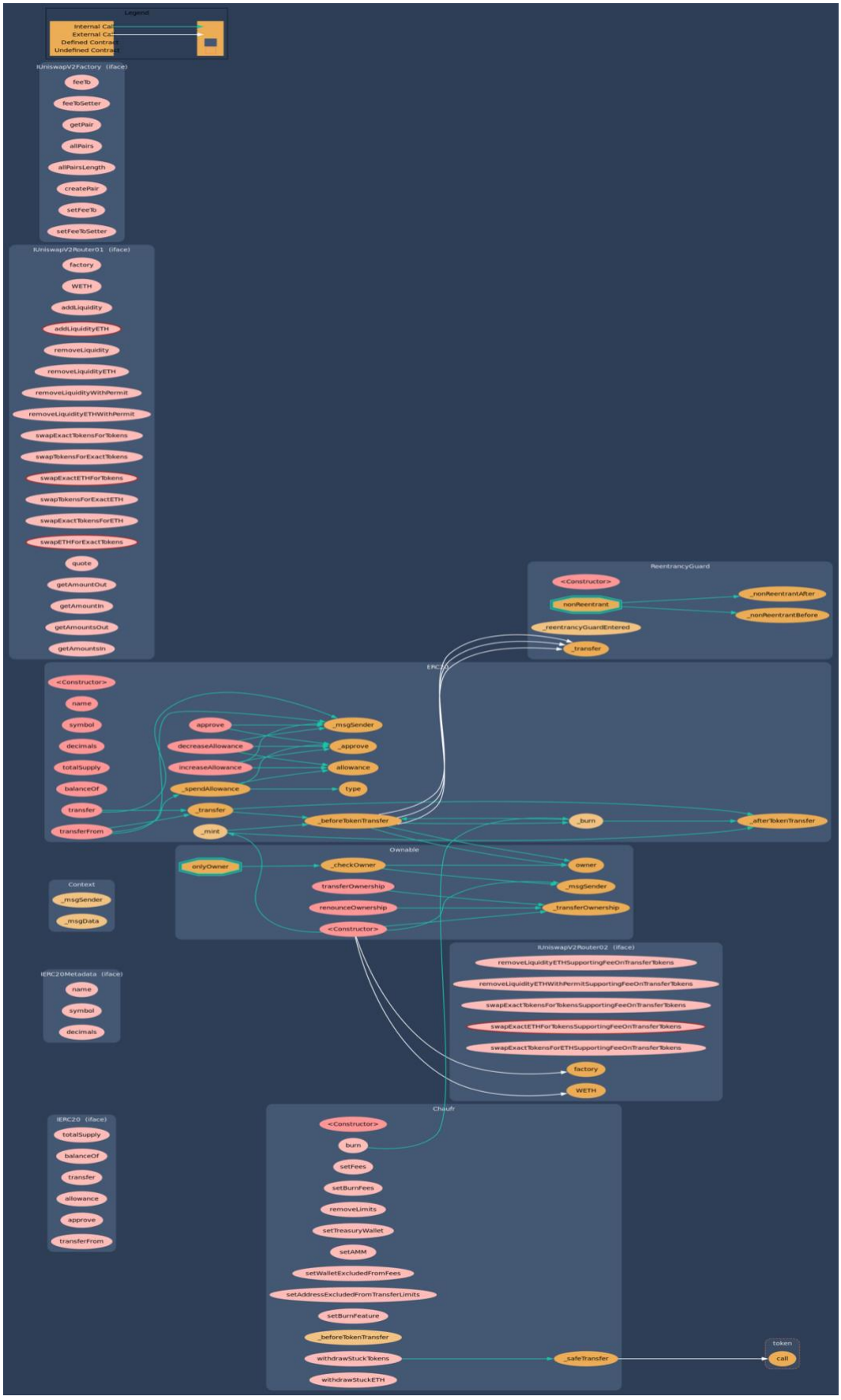
1- SOLIDITY STATIC ANALYSIS



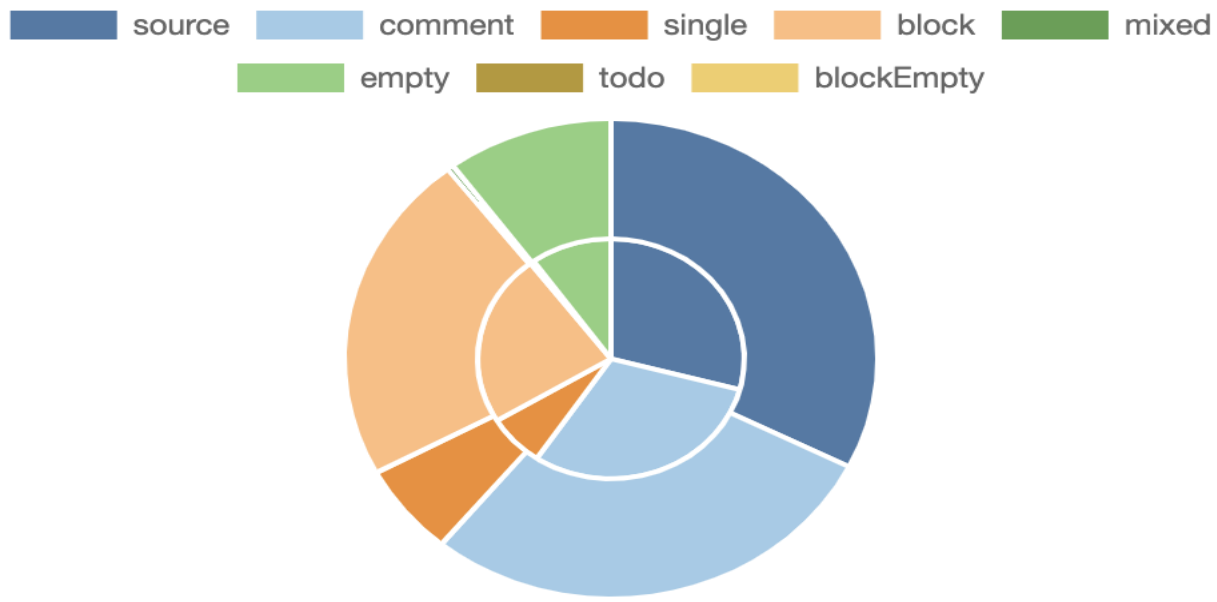
2- Inheritance graph



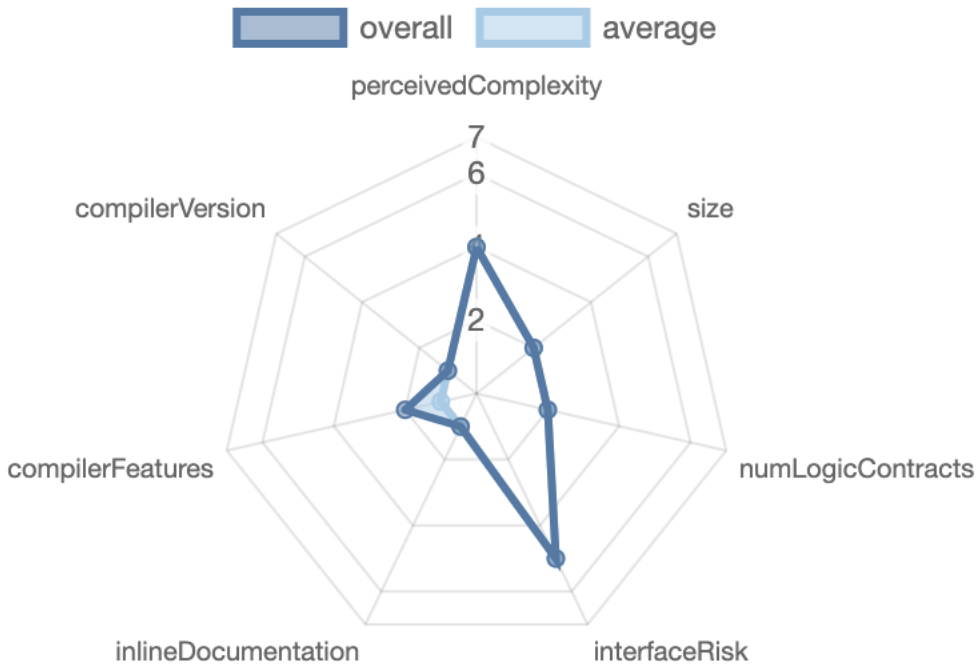
3- Call graph


Legend
IUniswapV2Factory (iface)
feeTo
feeToSetter
getPair
allPairs
allPairsLength
createPair
setFeeTo
setFeeToSetter
IUniswapV2Router01 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityWithPermit
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapTokensForExactTokens
swapExactETHForTokens
swapTokensForExactETH
swapExactTokensForETH
swapETHForExactTokens
quote
getAmountOut
getAmountIn
getAmountsOut
getAmountsIn
ReentrancyGuard
<Constructor>
nonReentrant
_reentrancyGuardEntered
_nonReentrantAfter
_nonReentrantBefore
_transfer
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transfer
transferFrom
approve
decreaseAllowance
increaseAllowance
_spendAllowance
_msgSender
_approve
allowance
type
_transfer
_mint
_beforeTokenTransfer
_burn
_afterTokenTransfer
Ownable
onlyOwner
_checkOwner
transferOwnership
renounceOwnership
<Constructor>
owner
_msgSender
_transferOwnership
Context
_msgSender
_msgData
IERC20Metadata (iface)
name
symbol
decimals
IUniswapV2Router02 (iface)
removeLiquidityETHSupportingFeeOnTransferTokens
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactETHForTokensSupportingFeeOnTransferTokens
swapExactTokensForETHSupportingFeeOnTransferTokens
factory
WETH
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Chaufr
<Constructor>
burn
setFees
setBurnFees
removeLimits
setTreasuryWallet
setAMM
setWalletExcludedFromFees
setAddressExcludedFromTransferLimits
setBurnFeature
_beforeTokenTransfer
withdrawStuckTokens
withdrawStuckETH
_safeTransfer
token
call

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
📝🔍💊	Chaufr.sol	5	5	1112	859	364	449	360	💰🧱🔆Σ
📝🔍💊	**Totals**	**5**	**5**	**1112**	**859**	**364**	**449**	**360**	💰🧱🔆Σ

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
2	0	5	3

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
66	4

External	Internal	Private	Pure	View
52	66	3	5	24

StateVariables

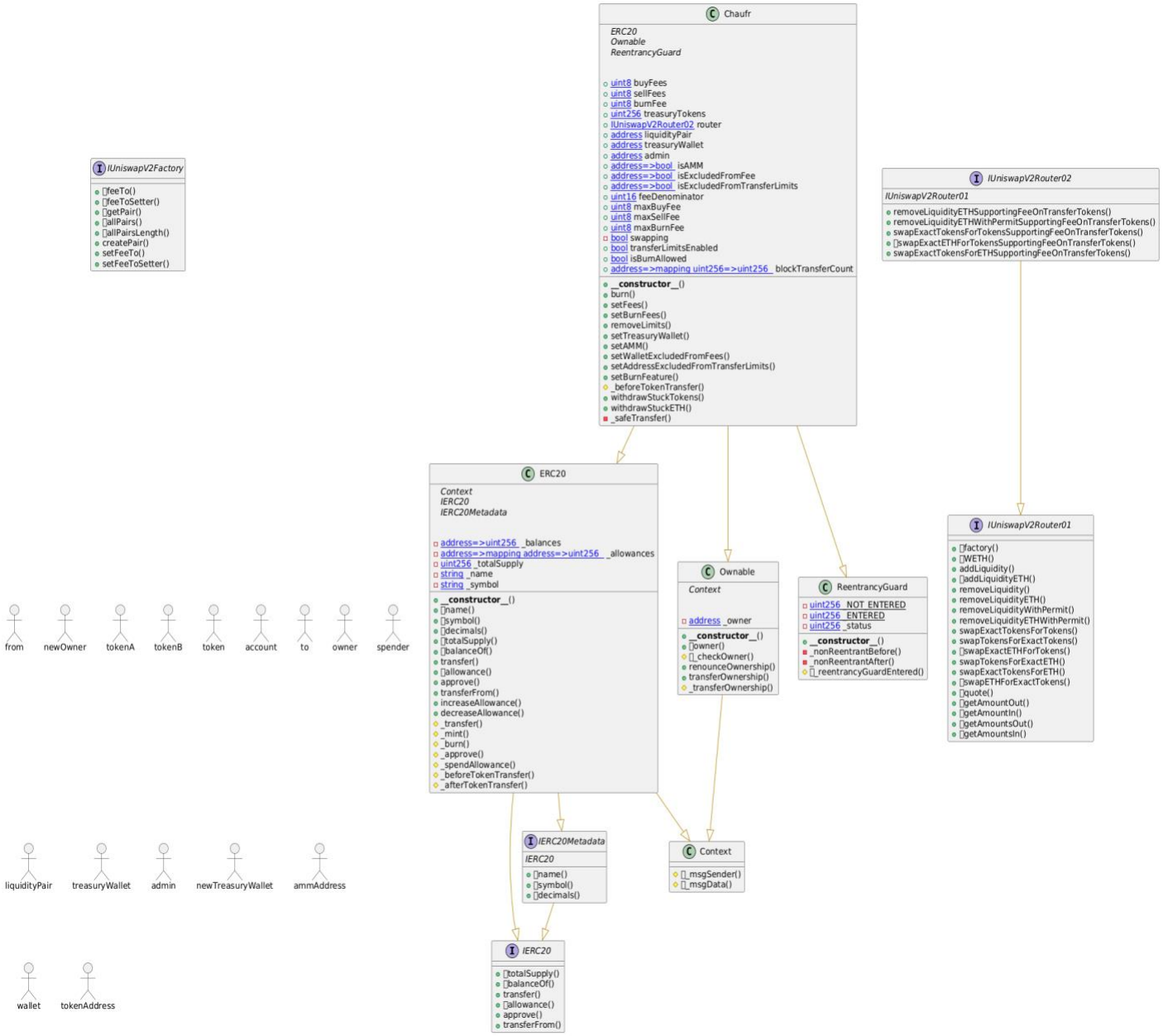
Total	🌐Public
28	18

Capabilities

Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥️ Uses Assembly	💣 Has Destroyable Contracts
`^0.8.0` `>=0.6.2` `>=0.5.0` `0.8.28`		`yes`		

📩 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
totalSupply	18160ddd	totalSupply()
balanceOf	70a08231	balanceOf(address)
transfer	a9059cbb	transfer(address,uint256)
allowance	dd62ed3e	allowance(address,address)
approve	095ea7b3	approve(address,uint256)
transferFrom	23b872dd	transferFrom(address,address,uint256)
name	06fdde03	name()
symbol	95d89b41	symbol()
decimals	313ce567	decimals()
name	06fdde03	name()
symbol	95d89b41	symbol()
decimals	313ce567	decimals()
totalSupply	18160ddd	totalSupply()
balanceOf	70a08231	balanceOf(address)
transfer	a9059cbb	transfer(address,uint256)
allowance	dd62ed3e	allowance(address,address)
approve	095ea7b3	approve(address,uint256)
transferFrom	23b872dd	transferFrom(address,address,uint256)
increaseAllowance	39509351	increaseAllowance(address,uint256)
decreaseAllowance	a457c2d7	decreaseAllowance(address,uint256)
owner	8da5cb5b	owner()
renounceOwnership	715018a6	renounceOwnership()
transferOwnership	f2fde38b	transferOwnership(address)
factory	c45a0155	factory()
WETH	ad5c4648	WETH()
addLiquidity	e8e33700	addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)
addLiquidityETH	f305d719	addLiquidityETH(address,uint256,uint256,uint256,address,uint256)
removeLiquidity	baa2abde	removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
removeLiquidityETH	02751cec	removeLiquidityETH(address,uint256,uint256,uint256,address,uint256)
removeLiquidityWithPermit	2195995c	removeLiquidityWithPermit(address,address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
removeLiquidityETHWithPermit	ded9382a	removeLiquidityETHWithPermit(address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
swapExactTokensForTokens	38ed1739	swapExactTokensForTokens(uint256,uint256,address[],address,uint256)
swapTokensForExactTokens	8803dbee	swapTokensForExactTokens(uint256,uint256,address[],address,uint256)
swapExactETHForTokens	7ff36ab5	swapExactETHForTokens(uint256,address[],address,uint256)

| swapTokensForExactETH | 4a25d94a |
swapTokensForExactETH(uint256,uint256,address[],address,uint256) |
| swapExactTokensForETH | 18cbafe5 |
swapExactTokensForETH(uint256,uint256,address[],address,uint256) |
| swapETHForExactTokens | fb3bdb41 |
swapETHForExactTokens(uint256,address[],address,uint256) |
quote	ad615dec	quote(uint256,uint256,uint256)
getAmountOut	054d50d4	getAmountOut(uint256,uint256,uint256)
getAmountIn	85f8c259	getAmountIn(uint256,uint256,uint256)
getAmountsOut	d06ca61f	getAmountsOut(uint256,address[])
getAmountsIn	1f00ca74	getAmountsIn(uint256,address[])
removeLiquidityETHSupportingFeeOnTransferTokens	af2979eb
removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,uint256,u
int256,address,uint256)
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	5b0d5984
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens(address,uint256
,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
swapExactTokensForTokensSupportingFeeOnTransferTokens	5c11d795
swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,add
ress[],address,uint256)
swapExactETHForTokensSupportingFeeOnTransferTokens	b6f9de95
swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],addr
ess,uint256)
swapExactTokensForETHSupportingFeeOnTransferTokens	791ac947
swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,addres
s[],address,uint256)
feeTo	017e7e58	feeTo()
feeToSetter	094b7415	feeToSetter()
getPair	e6a43905	getPair(address,address)
allPairs	1e3dd18b	allPairs(uint256)
allPairsLength	574f2ba3	allPairsLength()
createPair	c9c65396	createPair(address,address)
setFeeTo	f46901ed	setFeeTo(address)
setFeeToSetter	a2e74af6	setFeeToSetter(address)
burn	42966c68	burn(uint256)
setFees	4fcd2446	setFees(uint8,uint8)
setBurnFees	a881eb85	setBurnFees(uint8)
removeLimits	751039fc	removeLimits()
setTreasuryWallet	a8602fea	setTreasuryWallet(address)
setAMM	a9d3cd8a	setAMM(address,bool)
setWalletExcludedFromFees	8e89cf4d
setWalletExcludedFromFees(address,bool)
setAddressExcludedFromTransferLimits	1ef3e939
setAddressExcludedFromTransferLimits(address,bool)
setBurnFeature	e3d35e1c	setBurnFeature(bool)
withdrawStuckTokens	bd61f0a6	withdrawStuckTokens(address,uint256)
withdrawStuckETH	f5648a4f	withdrawStuckETH()

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/Chaufr.sol	3bddc20dec18f65ff00ad0d4bef6c663745d93f3

Contracts Description Table

Contract	Type	Bases
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	decimals	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	Public ❗	🛑	NO❗
└	allowance	Public ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	


└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
ReentrancyGuard	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	_nonReentrantBefore	Private 🔐	🛑	
└	_nonReentrantAfter	Private 🔐	🛑	
└	_reentrancyGuardEntered	Internal 🔒		
IUniswapV2Router01	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🛑	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🛑	NO❗
└	removeLiquidityETH	External ❗	🛑	NO❗
└	removeLiquidityWithPermit	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
└	swapExactTokensForTokens	External ❗	🛑	NO❗
└	swapTokensForExactTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🛑	NO❗
└	swapExactTokensForETH	External ❗	🛑	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗

🔘 |NO❗ |
IUniswapV2Factory	Interface			
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗
└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	🔘	NO❗
└	setFeeTo	External ❗	🔘	NO❗
└	setFeeToSetter	External ❗	🔘	NO❗
Chaufr	Implementation	ERC20, Ownable, ReentrancyGuard		
└	<Constructor>	Public ❗	🔘	ERC20 Ownable
└	burn	External ❗	🔘	onlyOwner
└	setFees	External ❗	🔘	onlyOwner
└	setBurnFees	External ❗	🔘	onlyOwner
└	removeLimits	External ❗	🔘	onlyOwner
└	setTreasuryWallet	External ❗	🔘	onlyOwner
└	setAMM	External ❗	🔘	onlyOwner
└	setWalletExcludedFromFees	External ❗	🔘	onlyOwner
└	setAddressExcludedFromTransferLimits	External ❗	🔘
onlyOwner
└	setBurnFeature	External ❗	🔘	onlyOwner
└	_beforeTokenTransfer	Internal 🔒	🔘	
└	withdrawStuckTokens	External ❗	🔘	onlyOwner nonReentrant
└	withdrawStuckETH	External ❗	🔘	onlyOwner nonReentrant
└	_safeTransfer	Private 🔐	🔘	nonReentrant

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.