RugFreeCoins Audit

Viking Floki Audit

Smart Contract Security Audit

November 14, 2021

Contents

Audit details

Audited project

Viking Floki Token

Contract Address

0x42FcD264D6C61bAe8BBf52CCa0776dD914E497E9

Client contact

Viking Floki Team

Blockchain

Binance smart chain

Project website

https://vikingfloki.org/

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at 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 us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: 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 Rugfreecoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rugfreecoins) owe no duty of care towards you or any other person, nor does Rugfreecoins 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 Rugfreecoins 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, Rugfreecoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rugfreecoins, 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.

Background

Rugfreecoins was commissioned by Viking Floki to perform an audit of the smart contract.

https://bscscan.com/token/0x42FcD264D6C61bAe8BBf52CCa0776dD914E497E9

The focus of this audit is to verify that the smart contract is secure, resilient and working according to the specifications.

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

About the project

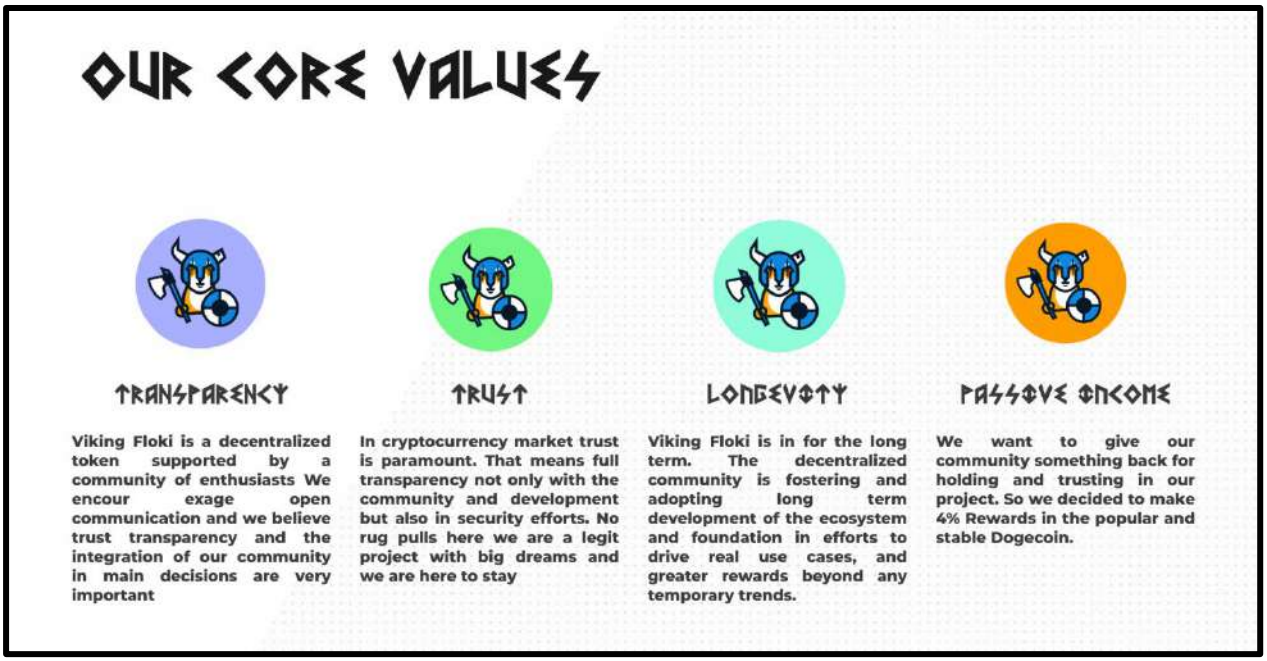
Viking Floki is a token built on the Binance Smart Chain. Every hour, Doge rewards are distributed automatically to all holders. Each transaction, purchase and sale incur 12% fee.

Features

- The **Doge rewards** will be distributed among every holder proportional to how many tokens each individual holds in values of **4% when buying and selling**.

- The **sustainability fee of 6% for marketing and manual buyback when buying and selling** is what allows Viking Floki to hold the aforementioned promise. Tokens will be swapped into Doge and will be sent to a marketing wallet per transaction. This way, Viking Floki will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.

- The additional component included under the sustainability section is a **liquidity fee of 2% from buying and selling,** which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

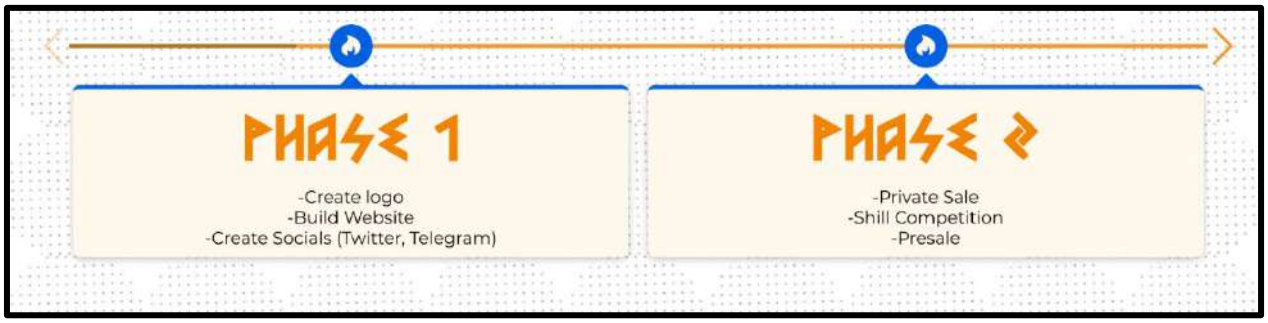


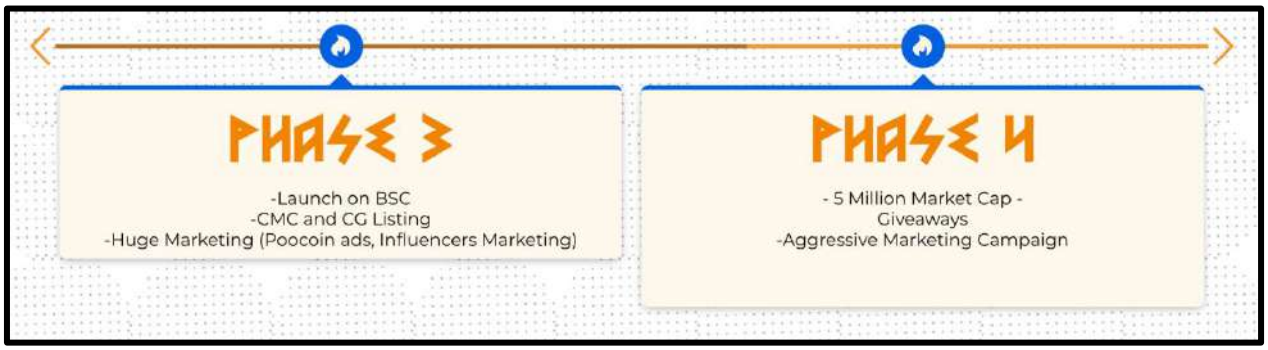
Tokenomics

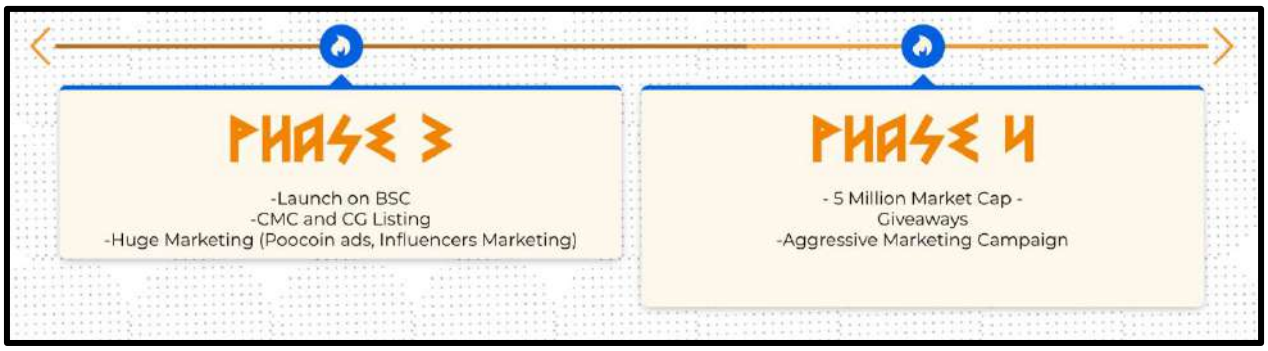
12% fee when buying and selling

- 4% of trade goes to holders' pockets in Doge tokens.
- 6% of trade goes to the marketing and manual buyback wallet.
- 2% of trade goes to the liquidity pool.

Roadmap


PHASE 1
-Create logo
-Build Website
-Create Socials (Twitter, Telegram)
PHASE 2
-Private Sale
-Shill Competition
-Presale


PHASE 3
-Launch on BSC
-CMC and CG Listing
-Huge Marketing (Poocoin ads, Influencers Marketing)
PHASE 4
- 5 Million Market Cap -
Giveaways
-Aggressive Marketing Campaign


PHASE 3
-Launch on BSC
-CMC and CG Listing
-Huge Marketing (Poocoin ads, Influencers Marketing)
PHASE 4
- 5 Million Market Cap -
Giveaways
-Aggressive Marketing Campaign

Target market and the concept

Target market

- Anyone who's interested in Crypto space with long term investment plans.
- Anyone who's ready to earn a passive income in Doge by holding tokens.
- Anyone who's interested in trading tokens.
- All Doge investors and fans out there.
- Anyone who's interested in taking part with the future plans of the Viking Floki token.
- Anyone who's interested in making financial transactions with any other party using Doge or Viking Floki as the currency.

Core concept

The Viking Floki reward system

4% of each transaction when buying and selling gets converted to Doge, and is split amongst all holders. Holders will be eligible to receive tokens every hour hours and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The **sustainability fee of 3% for marketing** is what allows Viking Floki to promote the token and use funds to further development of the platform. Tokens will be swapped into Doge and will be sent to a marketing wallet per transaction. This way, Viking Floki will have access to the funds without selling tokens as the traditional way, which will enable them to consume funds without hurting the project.

The controlled buyback collects 3% tax on each transaction, which is stored inside a private wallet. It is to save from massive dips in order to keep the token market price stable.

The liquidity fee of 2%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

Potential to grow with score points

1.	Project efficiency	7/10
2.	Project uniqueness	7/10
3	Information quality	7/10
4	Service quality	7/10
5	System quality	7/10
6	Impact on the community	8/10
7	Impact on the business	8/10
8	Preparing for the future	5/10
Total Points		**7/10**

Contract details

Token contract details for 14th November 2021

Contract name	Viking Floki
Contract address	0x42FcD264D6C61bAe8BBf52CCa0776dD914E497E9
Token supply	100,000,000,000
Token ticker	VF
Decimals	18
Token holders	2
Transaction count	2
Marketing address	0x3a7c1348668a76a1c2b3bfe7c6ea51b0fa567fcd
Dividend tracker	0x0310bfa7fc17cf66cb9a7af8d1827aba8218cb1a
Reward Token	0xba2ae424d960c26247dd6c32edc70b295c744c43
Contract deployer address	0x553ab66e1169875666571148490970 9c3be20f90
Contract's current owner address	0x069fD156c0d22E5D5F68e92f3237624B8eB6Ae9C

Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	pass
		SafeMath features	pass
		Fallback usage	pass
		tx.origin usage	pass
		deprecated items	pass
		Redundant code	pass
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Selfdestruct function security	pass
3	Business security	Access control of owners	low issue
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		pass
5	Reentrancy		pass
6	Exceptional reachable state		pass
7	Transaction ordering dependence		pass
8	Block properties dependence		pass
9	Pseudo random number generator (PRNG)		pass
10	DoS (Denial of Service)		pass
11	Token vesting implementation		pass
12	Fake deposit		pass
13	Event security		pass

Contract description table

Below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions and implementations with its visibility and mutability.

Contract	**Type**	**Bases**		
└	Function Name	Visibility	Mutability	Modifiers
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	**Interface**			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeMath	**Library**			
└	tryAdd	Internal 🔒		

└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
ERC20	**Implementation**	**Context, IERC20**		
└		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 🔒	🛑	
└	_setupDecimals	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
AddressUpgradeable	**Library**			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	

└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	_verifyCallResult	Private 🔐		
Initializable	**Implementation**			
└	_isConstructor	Private 🔐		
ContextUpgradeable	**Implementation**	**Initializable**		
└	__Context_init	Internal 🔒	🛑	initializer
└	__Context_init_unchained	Internal 🔒	🛑	initializer
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20Upgradeable	**Interface**			
└	totalSupply	External ❗		NO❗

└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🔘	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🔘	NO❗
└	transferFrom	External ❗	🔘	NO❗
SafeMathUpgradeable	**Library**			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		

└	div	Internal 🔒		
└	mod	Internal 🔒		
ERC20Upgradeable	**Implementation**	**Initializable, ContextUpgradeable, IERC20Upgradable**		
└	__ERC20_init	Internal 🔒	🛑	initializer
└	__ERC20_init_unchained	Internal 🔒	🛑	initializer
└	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 🔒	🔘	
└	_setupDecimals	Internal 🔒	🔘	
└	_beforeTokenTransfer	Internal 🔒	🔘	
Ownable	**Implementation**	**Context**		
└		Public ❗	🔘	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔘	onlyOwner
└	transferOwnership	Public ❗	🔘	onlyOwner
OwnableUpgradeable	**Implementation**	**Initializable, ContextUpgradeable**		
└	__Ownable_init	Internal 🔒	🔘	initializer
└	__Ownable_init_unchained	Internal 🔒	🔘	initializer
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔘	onlyOwner

└	transferOwnership	Public ❗	🔘	onlyOwner
Clones	**Library**			
└	clone	Internal 🔒	🔘	
└	cloneDeterministic	Internal 🔒	🔘	
└	predictDeterministicAddress	Internal 🔒		
└	predictDeterministicAddress	Internal 🔒		
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❗
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❗
IUniswapV2Pair	**Interface**			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗

L	allowance	External ❗		NO❗
L	approve	External ❗	🔘	NO❗
L	transfer	External ❗	🔘	NO❗
L	transferFrom	External ❗	🔘	NO❗
L	DOMAIN_SEPARATOR	External ❗		NO❗
L	PERMIT_TYPEHASH	External ❗		NO❗
L	nonces	External ❗		NO❗
L	permit	External ❗	🔘	NO❗
L	MINIMUM_LIQUIDITY	External ❗		NO❗
L	factory	External ❗		NO❗
L	token0	External ❗		NO❗
L	token1	External ❗		NO❗
L	getReserves	External ❗		NO❗
L	price0CumulativeLast	External ❗		NO❗
L	price1CumulativeLast	External ❗		NO❗
L	kLast	External ❗		NO❗
L	mint	External ❗	🔘	NO❗
L	burn	External ❗	🔘	NO❗

└	swap	External ❗	🔘	NO❗
└	skim	External ❗	🔘	NO❗
└	sync	External ❗	🔘	NO❗
└	initialize	External ❗	🔘	NO❗
SafeMathInt	**Library**			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	**Library**			
└	toInt256Safe	Internal 🔒		
IterableMapping	**Library**			
└	get	Public ❗		NO❗
└	getIndexOfKey	Public ❗		NO❗

└	getKeyAtIndex	Public ❗		NO❗
└	size	Public ❗		NO❗
└	set	Public ❗	🔘	NO❗
└	remove	Public ❗	🔘	NO❗
IBabyToken	**Interface**			
└	initialize	External ❗	🔘	NO❗
DividendPayingToken Interface	**Interface**			
└	dividendOf	External ❗		NO❗
└	withdrawDividend	External ❗	🔘	NO❗
DividendPayingToken OptionalInterface	**Interface**			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
DividendPayingToken	**Implementation**	**ERC20Upgradeable, OwnableUpgradeable, DividendPayingTokenInterface,**		

		DividendPayingTokenOptionalInterface		
└	__DividendPayingToken_init	Internal 🔒	🔘	initializer
└	distributeCAKEDividends	Public ❗	🔘	onlyOwner
└	withdrawDividend	Public ❗	🔘	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🔘	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🔘	
└	_mint	Internal 🔒	🔘	
└	_burn	Internal 🔒	🔘	
└	_setBalance	Internal 🔒	🔘	
BABYTOKEN	**Implementation**	**ERC20Upgradeable, OwnableUpgradeable, IBabyToken**		
└		Public ❗	🔘	NO❗

L	initialize	External ❗	🔘	initializer
L		External ❗	💵	NO❗
L	setSwapTokensAtAmount	External ❗	🔘	onlyOwner
L	updateDividendTracker	Public ❗	🔘	onlyOwner
L	updateUniswapV2Router	Public ❗	🔘	onlyOwner
L	excludeFromFees	Public ❗	🔘	onlyOwner
L	excludeMultipleAccountsFromFees	Public ❗	🔘	onlyOwner
L	setMarketingWallet	External ❗	🔘	onlyOwner
L	setTokenRewardsFee	External ❗	🔘	onlyOwner
L	setLiquiditFee	External ❗	🔘	onlyOwner
L	setMarketingFee	External ❗	🔘	onlyOwner
L	setAutomatedMarketMakerPair	Public ❗	🔘	onlyOwner
L	blacklistAddress	External ❗	🔘	onlyOwner
L	_setAutomatedMarketMakerPair	Private 🔐	🔘	
L	updateGasForProcessing	Public ❗	🔘	onlyOwner
L	updateClaimWait	External ❗	🔘	onlyOwner
L	getClaimWait	External ❗		NO❗
L	getTotalDividendsDistributed	External ❗		NO❗

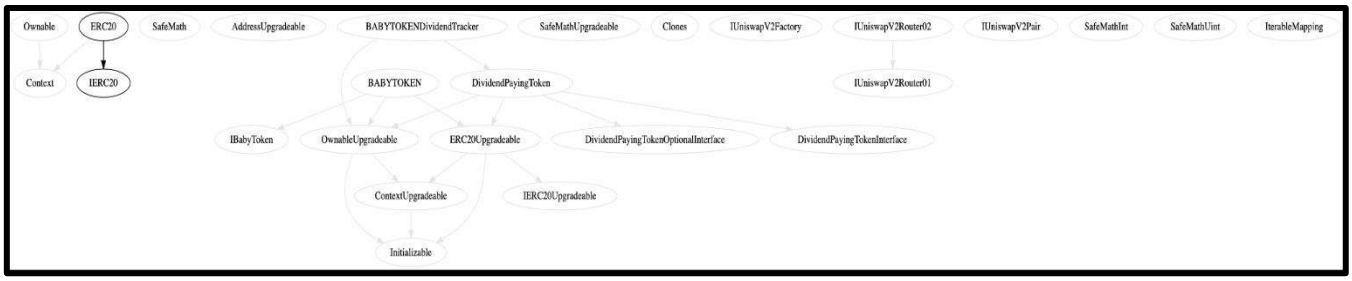
L	isExcludedFromFees	Public ❗		NO❗
L	withdrawableDividendOf	Public ❗		NO❗
L	dividendTokenBalanceOf	Public ❗		NO❗
L	excludeFromDividends	External ❗	🛑	onlyOwner
L	getAccountDividendsInfo	External ❗		NO❗
L	getAccountDividendsInfoAtIndex	External ❗		NO❗
L	processDividendTracker	External ❗	🛑	NO❗
L	claim	External ❗	🛑	NO❗
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfDividendTokenHolders	External ❗		NO❗
L	_transfer	Internal 🔒	🛑	
L	swapAndSendToFee	Private 🔐	🛑	
L	swapAndLiquify	Private 🔐	🛑	
L	swapTokensForEth	Private 🔐	🛑	
L	swapTokensForCake	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	swapAndSendDividends	Private 🔐	🛑	

BABYTOKENDividend Tracker	**Implementation**	**OwnableUpgradeable, DividendPayingToken**		
└	initialize	External ❗	🔘	initializer
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	excludeFromDividends	External ❗	🔘	onlyOwner
└	updateClaimWait	External ❗	🔘	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🔘	onlyOwner
└	process	Public ❗	🔘	NO❗
└	processAccount	Public ❗	🔘	onlyOwner

Legend

Symbol	Meaning
	**Function can modify state**
	Function is payable

Inheritance Hierarchy



Security issue checking status

- **High severity issues**
  No medium severity issues found.

- **Medium severity issues**
  No medium severity issues found.

- **Low severity issues**

  The addLiquidity function calls the uniswapV2Router.addLiquidityETH function with the address specified as owner() for acquiring the generated LP tokens from the Viking Floki (VF)-BNB pool. As a result,over time the _owner address will accumulate a significant portion of LP tokens. If the _owner is an EOA(Externally Owned Account), mishandling of its private key can have devastating consequences to the project as a whole.

  Recommendation
  We advise the address of the uniswapV2Router.addLiquidityETH function call to be replaced by the contract itself.

```
ftrace | funcSig
function addLiquidity(uint256 tokenAmount⬆, uint256 ethAmount⬆) private {
    // approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount⬆);

    // add the liquidity
    uniswapV2Router.addLiquidityETH{value: ethAmount⬆}(
        address(this),
        tokenAmount⬆,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        address(0),
        block.timestamp
    );
}
```

Owner privileges

- The owner can change token swap point.

```
ftrace | funcSig
function setSwapTokensAtAmount(uint256 amount) external onlyOwner {
    swapTokensAtAmount = amount;
}
```

- The owner can Update dividend tracker.

```
ftrace | funcSig
function updateDividendTracker(address newAddress) public onlyOwner {
    require(
        newAddress != address(dividendTracker),
        "BABYTOKEN: The dividend tracker already has that address"
    );

    BABYTOKENDividendTracker newDividendTracker = BABYTOKENDividendTracker(
        payable(newAddress)
    );

    require(
        newDividendTracker.owner() == address(this),
        "BABYTOKEN: The new dividend tracker must be owned by the BABYTOKEN token contract"
    );

    newDividendTracker.excludeFromDividends(address(newDividendTracker));
    newDividendTracker.excludeFromDividends(address(this));
    newDividendTracker.excludeFromDividends(owner());
    newDividendTracker.excludeFromDividends(address(uniswapV2Router));

    emit UpdateDividendTracker(newAddress, address(dividendTracker));

    dividendTracker = newDividendTracker;
}
```

❖ The owner can update router address.

```
ftrace | funcSig
function updateUniswapV2Router(address newAddress⬆) public onlyOwner {
    require(
        newAddress⬆ != address(uniswapV2Router),
        "BABYTOKEN: The router already has that address"
    );
    emit UpdateUniswapV2Router(newAddress⬆, address(uniswapV2Router));
    uniswapV2Router = IUniswapV2Router02(newAddress⬆);
    address _uniswapV2Pair = IUniswapV2Factory(uniswapV2Router.factory())
        .createPair(address(this), uniswapV2Router.WETH());
    uniswapV2Pair = _uniswapV2Pair;
}
```

❖ The owner can exclude wallets from fee.

```
ftrace | funcSig
function excludeFromFees(address account⬆, bool excluded⬆) public onlyOwner {
    require(
        _isExcludedFromFees[account⬆] != excluded⬆,
        "BABYTOKEN: Account is already the value of 'excluded'"
    );
    _isExcludedFromFees[account⬆] = excluded⬆;

    emit ExcludeFromFees(account⬆, excluded⬆);
}

ftrace | funcSig
function excludeMultipleAccountsFromFees(
    address[] calldata accounts⬆,
    bool excluded⬆
) public onlyOwner {
    for (uint256 i = 0; i < accounts⬆.length; i++) {
        _isExcludedFromFees[accounts⬆[i]] = excluded⬆;
    }

    emit ExcludeMultipleAccountsFromFees(accounts⬆, excluded⬆);
}
```

❖ The owner can change marketing wallet address.

```
ftrace | funcSig
function setMarketingWallet(address payable wallet⬆) external onlyOwner {
    _marketingWalletAddress = wallet⬆;
}
```

❖ The owner can change all fees.

```
ftrace | funcSig
function setTokenRewardsFee(uint256 value⬆) external onlyOwner {
    tokenRewardsFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
}

ftrace | funcSig
function setLiquiditFee(uint256 value⬆) external onlyOwner {
    liquidityFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
}

ftrace | funcSig
function setMarketingFee(uint256 value⬆) external onlyOwner {
    marketingFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
}
```

❖ The owner can change pair address.

```
ftrace | funcSig
function setAutomatedMarketMakerPair(address pair⬆, bool value⬆)
    public
    onlyOwner
{
    require(
        pair⬆ != uniswapV2Pair,
        "BABYTOKEN: The PancakeSwap pair cannot be removed from automatedMarketMakerPairs"
    );

    _setAutomatedMarketMakerPair(pair⬆, value⬆);
}
```

❖ The owner can blacklist wallets.

```
ftrace | funcSig
function blacklistAddress(address account⬆, bool value⬆) external onlyOwner {
    _isBlacklisted[account⬆] = value⬆;
}
```

❖ The owner can update Gas fee when sending dividends.

```
ftrace | funcSig
function updateGasForProcessing(uint256 newValue⬆) public onlyOwner {
    require(
        newValue⬆ >= 200000 && newValue⬆ <= 500000,
        "BABYTOKEN: gasForProcessing must be between 200,000 and 500,000"
    );
    require(
        newValue⬆ != gasForProcessing,
        "BABYTOKEN: Cannot update gasForProcessing to same value"
    );
    emit GasForProcessingUpdated(newValue⬆, gasForProcessing);
    gasForProcessing = newValue⬆;
}
```

❖ The owner can exclude wallets from dividend.

```
ftrace | funcSig
function excludeFromDividends(address account⬆) external onlyOwner {
    dividendTracker.excludeFromDividends(account⬆);
}
```

Audit conclusion

While conducting the audit of the Viking Floki smart contract, it was observed that there is nothing alarming with the code, and it only contains low severity issues.