RugFreeCoins Audit

Sprint’s Token

Smart Contract Security Audit

May 11, 2022

Contents

Audit details 1

Disclaimer 2

Background 3

About the project 4

Target market and the concept 7

Potential to grow with score points 8

Total Points 8

Contract details 9

Contract code function details 11

Contract description table 13

Security issue checking status 28

Owner privileges 29

Audit conclusion 33

Audit details

Audited project
Sprint’s Token

Contract Address

0x33E4a44E30293717261B4c8c05d85F8DA4ee2319

Client contact

Sprint’s Token Team

Blockchain

Binance smart chain

Project website

https://www.sprintn.fit/

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 the Sprint's Token Team to perform an audit of the smart contract.

https://bscscan.com/token/0x33E4a44E30293717261B4c8c05d85F8DA4ee2319

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, and long-term sustainability, and as a guide to improving the security posture of the smart contract by remediating the issues that were identified.

.

About the project

Sprint’s Finance is a token built on the Binance Smart Chain that is with an innovative investment use case the main purpose of which is to seek out constant revenue sources, it is a method of health training, and based on that, users can earn corresponding profits. That is the contemporary trend **Move-To-Earn**. Users can own their favorite NFTs to move outdoors to make profits with Sprint’s tokens. Each transaction, purchase, and sale incur a 10% fee.

Features

- The **Green Metaverse Token rewards** will be distributed in Sprint’s tokens among every holder proportional to how many tokens each individual holds in values of **3% when buying and selling.**

- The **sustainability fee of 5% when buying and selling for marketing** is what allows Sprint’s Token to hold the aforementioned promise. Tokens will be swapped into BNB and will be sent to a marketing wallet. This way, Sprint’s Token will have enough funds to promote the coin and spend for future development and marketing without selling tokens as the traditional way.
- The additional component included under the sustainability section is a **liquidity fee of 2% when buying and selling**, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

Roadmap

- Brainstorming - Idea Creation
- Marketing Research & Development
- Website Building
- Social Media Platform
- Smart Contract Deployed
- Contract Audit
- Fairlaunch Creation on Pinksale
- Marketing Campaign
- Listing on CoinGecko
- Listing on CoinMarketCap
- Website V.2
- Ecosystem Mechanism Design
- Application UI Design
- Application Development
- NFT Marketplace Development
- Application Official Release

Tokenomics

10% fee when buying and selling

- 3% of trade goes to holders pockets in Sprint’s token rewards.
- 5% of trade goes to the marketing wallet in BNB.
- 2% of trade goes to the liquidity pool

Target market and the concept

Target market

- Anyone who’s interested in the Crypto space with long-term investment plans.
- Anyone who’s ready to earn a passive income by holding tokens.
- Anyone who’s interested in trading tokens.
- Anyone who’s interested in collecting NFTs or trading NFTs.
- Anyone who’s interested in taking part with staking.
- Anyone who’s interested in exercise and gets rewards through moving activities.
- Anyone who’s interested in taking part in the future plans of Sprint’s Token Ecosystem.
- Anyone who’s interested in making financial transactions with any other party using Sprint’s as the currency.

Core concept

The Green Metaverse Token rewards system

3% of each transaction when buying and selling gets converted to **Green Metaverse** tokens and is split amongst all holders. Holders will be eligible to receive tokens in each transaction and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The **sustainability fee of 5% when buying and selling for dev and marketing** is what allows Sprint’s Token to promote the token and use funds to further the development of the platform. Tokens will be swapped into BNB and will be sent to a marketing wallet. This way, Sprint’s Token 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 liquidity fee of 2% when buying and selling, is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

Potential to grow with score points

1.	Project efficiency	8/10
2.	Project uniqueness	8/10
3	Information quality	8/10
4	Service quality	8/10
5	System quality	9/10
6	Impact on the community	9/10
7	Impact on the business	9/10
8	Preparing for the future	7/10
9	Smart contract security	10/10
10	Smart contract functionality assessment	10/10
Total Points		**8.6/10**

Contract details

Token contract details for 11th May 2022

Contract name	SPRINT
Contract address	0x33E4a44E30293717261B4c8c05d85F8DA4ee2319
Token supply	1,000,000,000,000,000
Token ticker	SPRINT
Decimals	18
Token holders	4
Transaction count	6
Dividend tracker	0x1910f32a5947a80a88d83c559f28e0441b2ebef4
Marketing wallet	0x655af86a36b7f5607dcf64606900c1e487165027
Reward Token	0x3019bf2a2ef8040c242c9a4c5c4bd4c81678b2a1
Contract deployer address	0x02b373bb3513C9Ff881db2c1fd789F4355aD9254
Contract’s current owner address	0x02b373bb3513c9ff881db2c1fd789f4355ad9254

Tokens are distributed as follows:

- Presale - 30%
- Liquidity Pool - 15%
- Staking Rewards - 30%
- Marketing and Partnership - 15%
- Initial Burn - 5%
- Airdrop - 5%

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	High
		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

The 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 their visibility and mutability.

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, IERC20 Metadata		
└		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 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	

Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
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 🔒		

Clones	**Library**			
L	clone	Internal 🔒	🛑	
L	cloneDeterministic	Internal 🔒	🛑	
L	predictDeterministicAddress	Internal 🔒		
L	predictDeterministicAddress	Internal 🔒		
IUniswapV2 Factory	**Interface**			
L	feeTo	External ❗		NO❗
L	feeToSetter	External ❗		NO❗
L	getPair	External ❗		NO❗
L	allPairs	External ❗		NO❗
L	allPairsLength	External ❗		NO❗
L	createPair	External ❗	🛑	NO❗
L	setFeeTo	External ❗	🛑	NO❗
L	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2 Router01	**Interface**			
L	factory	External ❗		NO❗
L	WETH	External ❗		NO❗
L	addLiquidity	External ❗	🛑	NO❗
L	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❗
IUniswapV2 Router02	**Interface**	**IUniswapV2 Router01**		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗

└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
IERC20 Upgradeable	**Interface**			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20Metadata Upgradeable	**Interface**	**IERC20 Upgradeable**		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
Initializable	**Implementation**			
Context Upgradeable	**Implementation**	**Initializable**		
└	__Context_init	Internal 🔒	🛑	initializer
└	__Context_init_unchained	Internal 🔒	🛑	initializer
└	_msgSender	Internal 🔒		

└	_msgData	Internal 🔒		
ERC20 Upgradeable	**Implementation**	**Initializable, Context Upgradeable, IERC20 Upgradeable, IERC20 Metadata Upgradeable**		
└	__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 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
OwnableUpgradeable	**Implementation**	**Initializable, ContextUpgradeable**		
└	__Ownable_init	Internal 🔒	🛑	initializer
└	__Ownable_init_unchained	Internal 🔒	🛑	initializer
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
IUniswapV2Pair	**Interface**			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗

└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	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❗
DividendPaying TokenInterface	**Interface**			
└	dividendOf	External ❗		NO❗
└	withdrawDividend	External ❗	🛑	NO❗

DividendPayingTokenOptionalInterface	**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 DividendTracker	**Implementation**	**Ownable Upgradeable, Dividend PayingToken**		
└	initialize	External ❗	🛑	initializer
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	isExcludedFromDividends	Public ❗		NO❗
└	updateClaimWait	External ❗	🛑	onlyOwner
└	updateMinimumTokenBalanceForDividends	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

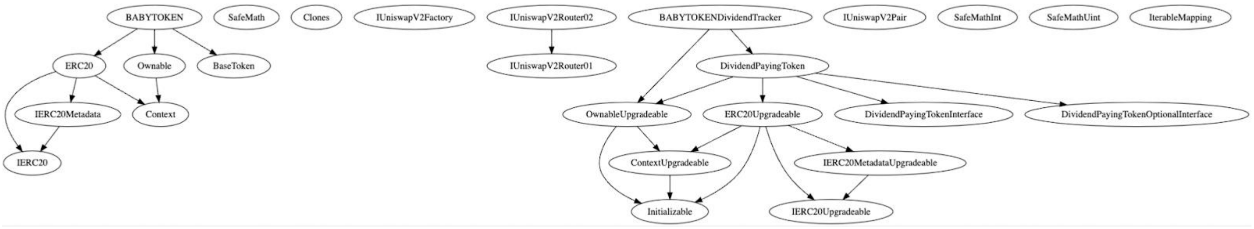
BaseToken	**Implementation**			
BABYTOKEN	**Implementation**	**ERC20, Ownable, BaseToken**		
└		Public ❗	💵	ERC20
└		External ❗	💵	NO❗
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateDividendTracker	Public ❗	🛑	onlyOwner
└	updateUniswapV2Router	Public ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setTokenRewardsFee	External ❗	🛑	onlyOwner
└	setLiquiditFee	External ❗	🛑	onlyOwner
└	setMarketingFee	External ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	updateMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner

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

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy



Security issue checking status

- **High severity issues**

  No High severity issues found.

- **Medium severity issues**

  No medium severity issues found

- **Low severity issues**

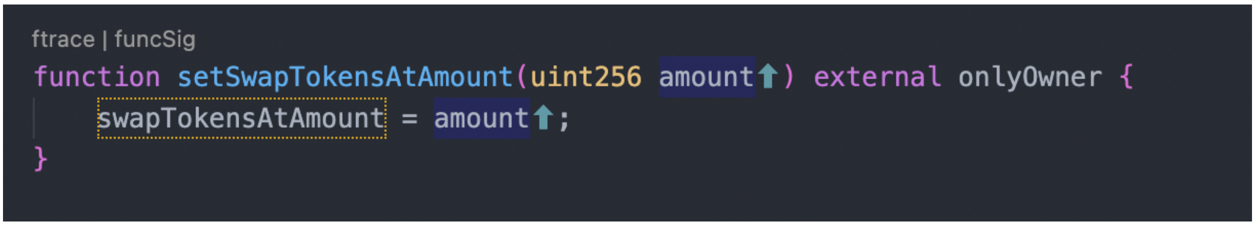
  No low severity issues found

- **Centralization risk**

  No Centralization issues found

Owner privileges

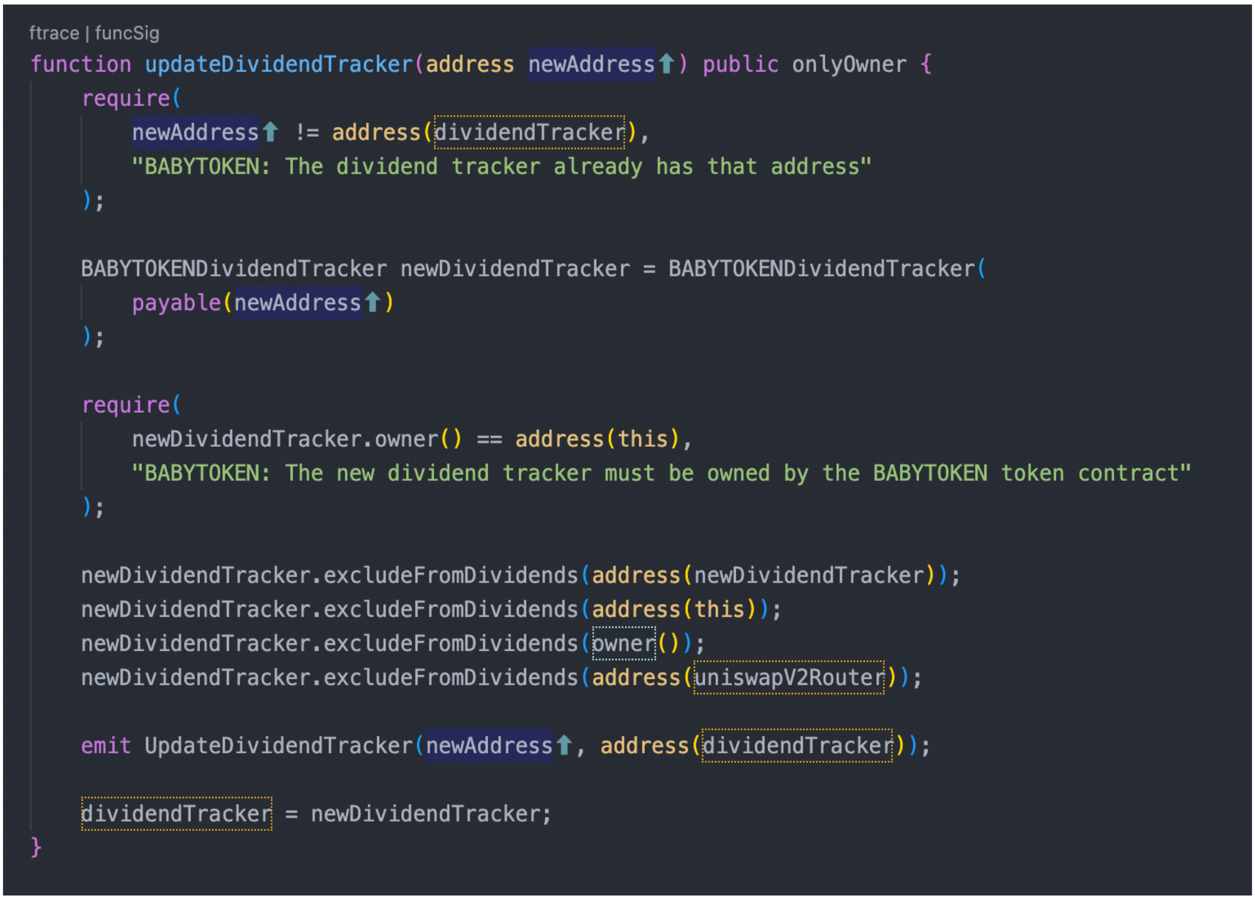
- ❖ The owner can change token swap point



ftrace | funcSig

```
function setSwapTokensAtAmount(uint256 amount⬆) external onlyOwner {
    swapTokensAtAmount = amount⬆;
}
```

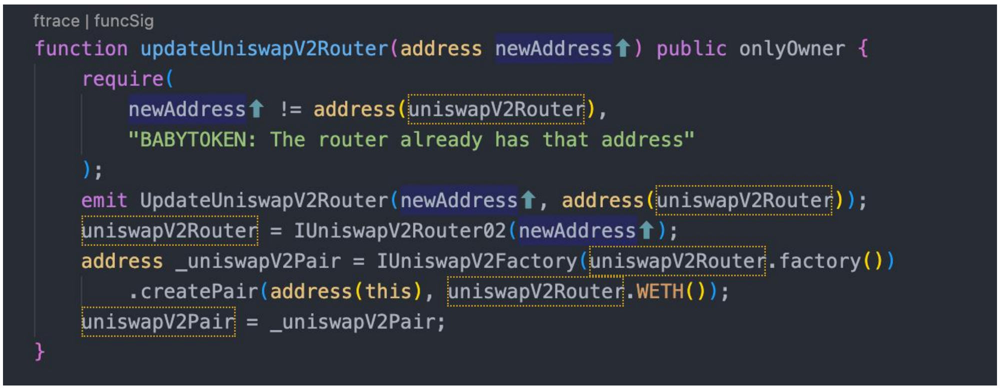
- ❖ The owner can update the dividend tracker



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;
}
```

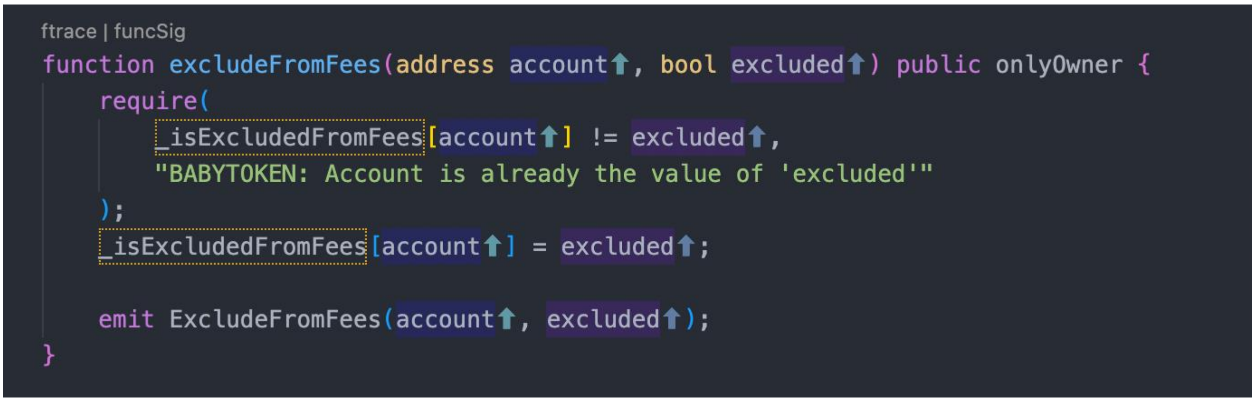
- The owner can update the router address



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;
}
```

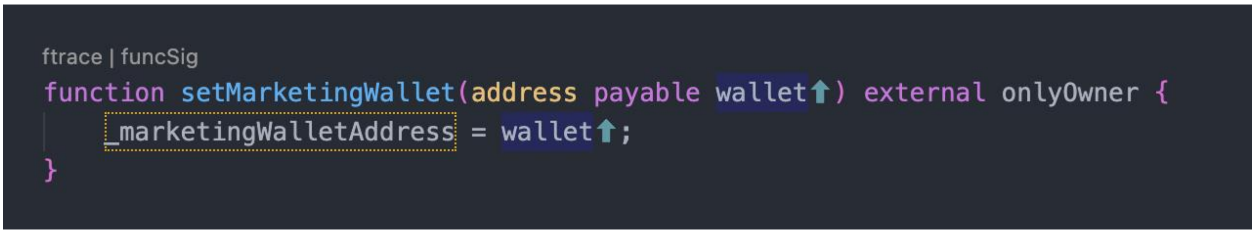
- The owner can include/exclude wallets from fees



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);
}
```

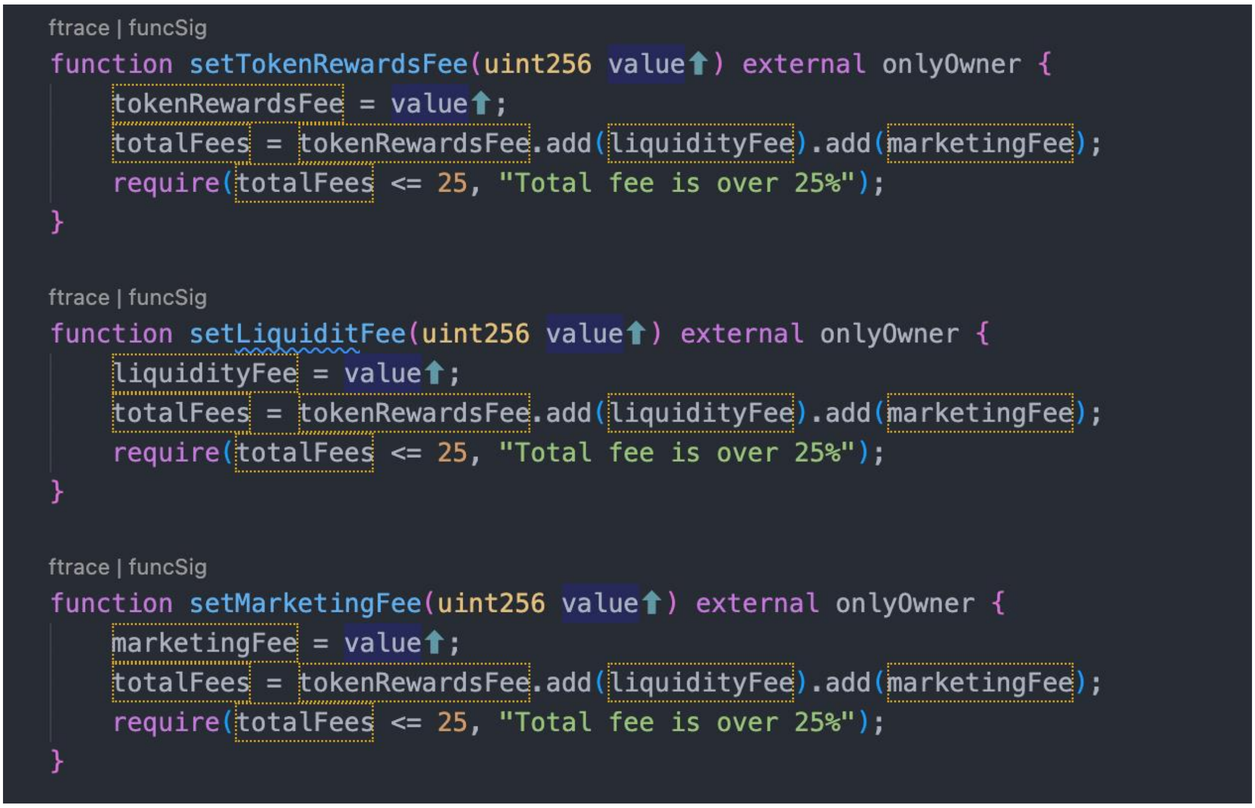
- The owner can change the marketing wallet



ftrace | funcSig

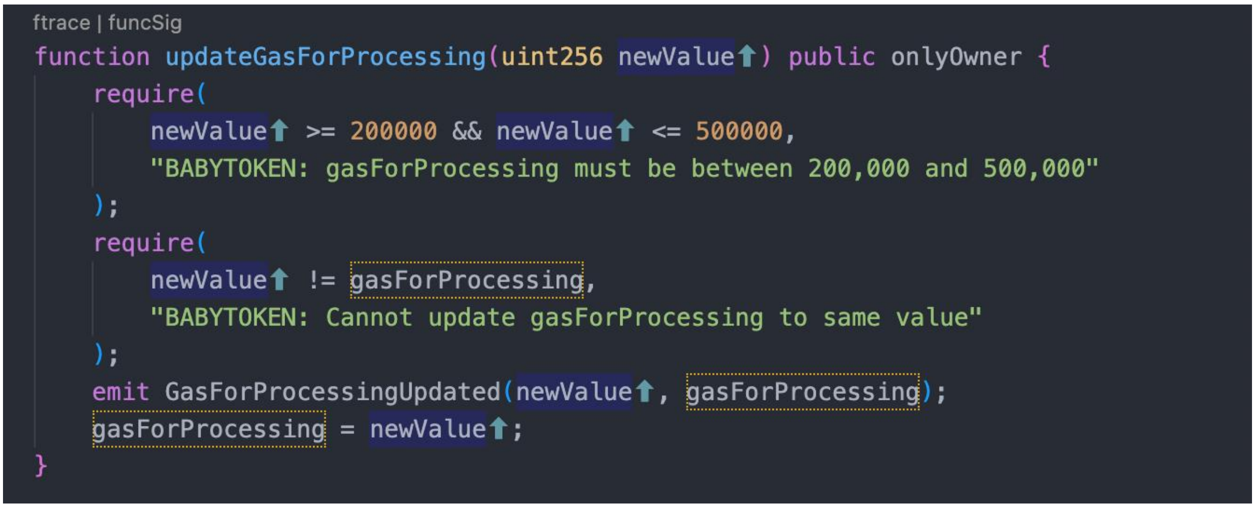
```
function setMarketingWallet(address payable wallet) external onlyOwner {
    _marketingWalletAddress = wallet;
}
```

- The owner can change all fees, total fees maximum up to 25%



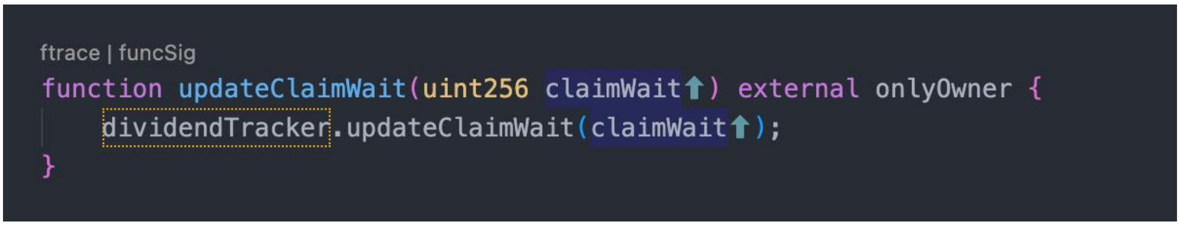
```
ftrace | funcSig
function setTokenRewardsFee(uint256 value⬆) external onlyOwner {
    tokenRewardsFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
    require(totalFees <= 25, "Total fee is over 25%");
}

ftrace | funcSig
function setLiquiditFee(uint256 value⬆) external onlyOwner {
    liquidityFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
    require(totalFees <= 25, "Total fee is over 25%");
}

ftrace | funcSig
function setMarketingFee(uint256 value⬆) external onlyOwner {
    marketingFee = value⬆;
    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
    require(totalFees <= 25, "Total fee is over 25%");
}
```

- The owner can change maximum gas for process dividend tracker



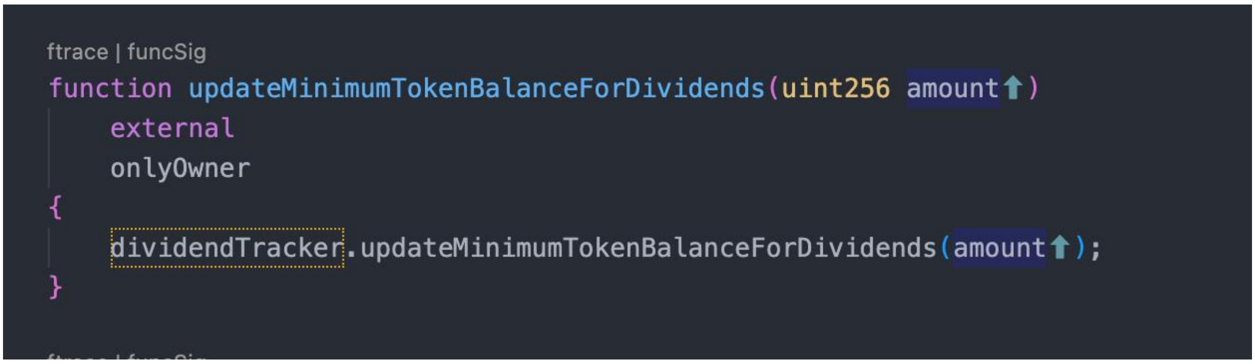
```
ftrace | funcSig
function updateGasForProcessing(uint256 newValue⬆) public onlyOwner {
    require(
        newValue⬆ >= 200000 && newValue⬆ <= 500000,
        "BABYTOKEN: gasForProcessing must be between 200,000 and 500,000"
    );
    require(
        newValue⬆ != gasForProcessing,
        "BABYTOKEN: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

- The owner can change dividend receive claim wait



```
ftrace | funcSig
function updateClaimWait(uint256 claimWait⬆) external onlyOwner {
    dividendTracker.updateClaimWait(claimWait⬆);
}
```

- The owner can change minimum token amount to have receive dividends



```
ftrace | funcSig
function updateMinimumTokenBalanceForDividends(uint256 amount)
    external
    onlyOwner
{
    dividendTracker.updateMinimumTokenBalanceForDividends(amount);
}
```

Audit conclusion

RugFreeCoins team has performed in-depth testings, line by line manual code review, and automated audit of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, manipulations, and hacks. According to the smart contract audit.

Smart contract functional Status: **PASSED**

Number of risk issues: **NONE**

Solidity code functional issue level: **PASSED**

Number of owner privileges: **9**

Centralization risk correlated to the active owner: **LOW**

Smart contract active ownership: **YES**