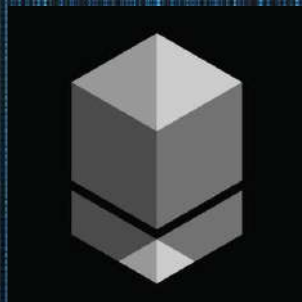




# **RugFreeCoins Audit**



## **Black Token Smart Contract Security Audit**

**November 21<sup>st</sup>, 2022**



# Contents

Audit details	1
Disclaimer	2
Overview	3
Background	4
Roadmap	5
Target market and the concept	8
Potential to grow with score points	9
Total Points	9
Contract details	10
Contract code function details	11
Contract description table	13
Security issue checking status	21
Owner privileges	23
Audit conclusion	25

# Audit details



**Audited project**  
Black Token



**Contract Address**  
0xa2F1a99a74d4cc072B810b1696239e4Dd76221C4



**Client contact**  
Black Token Team



**Blockchain**  
Binance Smart chain



**Project website**  
<https://blackproject.tech>

# 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.

# Overview

- ✓ No mint function found; the owner cannot mint tokens after initial deployment.
- ✓ The owner can't set a max transaction limit
- ✓ The owner can't pause trading.
- ✓ The owner can't set fees over 25%.
- ✓ The owner can't blacklist wallets.
- ✓ The owner can't set a max wallet limit
- ✓ The owner can't claim the contract's balance of its own token.

# Background

Rugfreecoins was commissioned by the Black Token Team to perform an audit of the smart contract.

<https://bscscan.com/address/0xa2f1a99a74d4cc072b810b1696239e4dd76221c4>

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 smart contract's security posture by remediating the identified issues.

.

# Roadmap

## Phase 1

- Value Proposition
- White Paper v1
- Proof of Concept
- Contract Development
- Development Black Wallet
- Financial Partnerships
- Press Releases
- Card Prototype
- Public AMAs
- Initial Impact Marketing

## Phase 2

- Conducting an IDO in Presale format at the Pinksale Launchpad
- Listing on Pancakeswap
- Launching crowdsourcing companies on the special platforms
- Team expansion
- Creating a dedicated support service
- Integration with Ledger Wallet hardware
- Development and implementation of a browser extension for web3 integrations in the application

## Phase 3

- Development and implementation into the application of the NFT explorer
- Development and implementation of DeFi explorer in the application
- Seeking and entering into partnership agreements with cybersecurity organizations
- Expanding the application's security settings and introducing new features
- Conclusion of a partnership agreement with the Advanced Cash payment system
- Integration of existing AML solutions into "Black Wallet" application or development of own technology.
- Conclusion of the partnership agreement with a large CEX crypto-exchange to create a cryptocurrency payment gateway

## **Phase 4**

- Issue UnionPay debit cards for Russian and CIS residents\* and prepare a module to work with them
- Issuance of Visa private cards for clients all over the world\* and preparation of the module for working with them
- Implementation of P2P exchanger in Black Wallet application
- Development and implementation of third-party services in the Marketplace application
- Searching for and taking part in a gas pedal or startup incubator
- Presenting the project at the Web Summit
- Presenting the project at the Slush (Startup Event)
- Project presentation at hub.berlin Conference



# Tokenomics

**0% when buying**

**6% when selling**

- 6% of trade goes to the marketing wallet in BNB

# 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 ready to staking and receive rewards.
- Anyone who's interested in taking part in the Black token ecosystem.
- Anyone who's interested in taking part in the future plans of Black Token.
- Anyone who's interested in making financial transactions with any other party using Black Token as the currency.

# Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	9/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	9/10
7	Impact on the business	9/10
8	Preparing for the future	9/10
9	Smart contract security	9/10
10	Smart contract functionality assessment	10/10
Total Points		<b>9.1/10</b>

# Contract details

## Token contract details for 21<sup>st</sup> of November 2022

Contract name	Black Token
Contract address	0xa2F1a99a74d4cc072B810b1696239e4Dd76221C4
Token supply	1,000,000,000
Token ticker	BLACK
Decimals	18
Token holders	1
Transaction count	2
Marketing address	0x3129f0db1be77582f6ebf8e24f40180937bc2625
Contract deployer address	0x5f85b9A37C7fAcfEF40F0866b212586E3bD66afe
Contract's current owner address	0x5f85b9a37c7facfef40f0866b212586e3bd66afe

# 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
		Self-destruct function security	pass
3	Business security	Access control of owners	pass
		Business logics	Low severity
		Business implementations	Low severity
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






































<b>12</b>	<b>Fake deposit</b>		<b>pass</b>
<b>13</b>	<b>Event security</b>		<b>pass</b>










# 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		
L	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		
IERC20	Interface			
L	totalSupply	External 		NO 
L	balanceOf	External 		NO 
L	transfer	External 		NO 
L	allowance	External 		NO 
L	approve	External 		NO 
L	transferFrom	External 		NO 
Address	Library			
L	isContract	Internal 		
L	sendValue	Internal 		
L	functionCall	Internal 		









L	functionCall	Internal 		
L	functionCallWithValue	Internal 		
L	functionCallWithValue	Internal 		
L	functionStaticCall	Internal 		
L	functionStaticCall	Internal 		
L	functionDelegateCall	Internal 		
L	functionDelegateCall	Internal 		
L	verifyCallResultFromTarget	Internal 		
L	verifyCallResult	Internal 		
L	_revert	Private 		
<b>Ownable</b>	<b>Implementation</b>	<b>Context</b>		
L		Public 		NO 
L	owner	Public 		NO 
L	_checkOwner	Internal 		
L	renounceOwnership	Public 		onlyOwner
L	transferOwnership	Public 		onlyOwner
L	_transferOwnership	Internal 		
<b>IUniswapV2 Factory</b>	<b>Interface</b>			
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 !
<b>IUniswapV2Pair</b>	<b>Interface</b>			
L	name	External !		NO !
L	symbol	External !		NO !
L	decimals	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	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	burn	External !		NO !
L	swap	External !		NO !
L	skim	External !		NO !
L	sync	External !		NO !
L	initialize	External !		NO !
<b>IUniswapV2 Router01</b>	<b>Interface</b>			
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	removeLiquidityWithPermit	External !		NO !
L	removeLiquidityETHWithPermit	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	quote	External !		NO !
L	getAmountOut	External !		NO !
L	getAmountIn	External !		NO !
L	getAmountsOut	External !		NO !
L	getAmountsIn	External !		NO !
<b>IUniswapV2Router02</b>	<b>Interface</b>	<b>IUniswapV2Router01</b>		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !

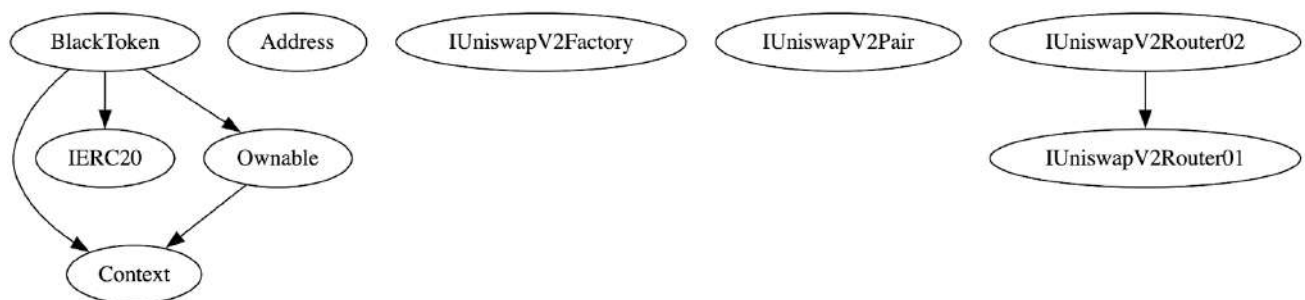
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
<b>BlackToken</b>	<b>Implementation</b>	<b>Context, IERC20, Ownable</b>		
L		Public !		NO !
L	contractVersion	Public !		NO !
L	contractDev	Public !		NO !
L	contractEdition	Public !		NO !
L	name	Public !		NO !
L	symbol	Public !		NO !
L	decimals	Public !		NO !
L	totalSupply	Public !		NO !
L	balanceOf	Public !		NO !
L	transfer	Public !		NO !
L	allowance	Public !		NO !
L	approve	Public !		NO !
L	transferFrom	Public !		NO !
L	increaseAllowance	Public !		NO !
L	decreaseAllowance	Public !		NO !
L	minimumTokensBeforeSwapAmount	Public !		NO !
L	_approve	Private 🐶		

L	_transfer	Private 🗝️	🛑	
L	swapAndLiquify	Public !	🛑	lockThe Swap
L	swapTokensForEth	Private 🗝️	🛑	
L	addLiquidity	Private 🗝️	🛑	
L	_tokenTransfer	Private 🗝️	🛑	
L	countUpFeeShare	Private 🗝️	🛑	
L	_transferBothExcluded	Private 🗝️	🛑	
L	_getTVValues	Private 🗝️		
L	_takeLiquidity	Private 🗝️	🛑	
L	isExcludedFromFee	External !		NO !
L	excludeFromFee	External !	🛑	onlyOwner
L	includeInFee	External !	🛑	onlyOwner
L	setMarketingWalletAddress	External !	🛑	onlyOwner
L	transferToAddressETH	Private 🗝️	🛑	
L		External !	📄	NO !
L	swapETHForTokens	Private 🗝️	🛑	
L	setZeroMarketingFee	External !	🛑	onlyOwner
L	recoverETHfromContract	External !	🛑	onlyOwner
L	recoverTokensFromContract	External !	🛑	onlyOwner

## 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

**Sending wrong swap amount:** When sending BNB it's sending initial BNB balance, not swapped BNB amount, it should send swapped BNB amount instead of initial BNB amount

```
function swapAndLiquify() public lockTheSwap {
    uint256 initialBalance = address(this).balance;
    initialBalance = address(this).balance;
    uint256 totalTokens = balanceOf(address(this));
    swapTokensForEth(totalTokens);

    uint256 walletsTotal = marketingTokensCollected;

    uint256 ethForMarketing = (initialBalance * marketingTokensCollected) /
        walletsTotal;

    transferToAddressETH(marketingWalletAddress, ethForMarketing);

    marketingTokensCollected = 0;
}
```

**Wrong calculations:** marketingTokensCollected value assigning to walletsTotal variable and when calculating ethForMarketing, initialBalance multiply by marketingTokensCollected and divided by walletsTotal, but in this case both value will be the same, so no need to do that calculation.

```
uint256 walletsTotal = marketingTokensCollected;

uint256 ethForMarketing = (initialBalance * marketingTokensCollected) /
    walletsTotal;
```



**Unused functions:** remove unused functions and variable to save gas

All liquidity functions are there, but in the contract it's not used.

```
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
        deadWallet,
        block.timestamp
    );
}
```

Swap ETH function is there, but it's not used in the contract

```
/////---dev---/////
event SwapETHForTokens(uint256 amountIn, address[] path);

ftrace | funcSig
function swapETHForTokens(uint256 amount↑) private {
    // generate the uniswap pair path of token -> weth
    address[] memory path = new address[](2);
    path[0] = uniswapV2Router.WETH();
    path[1] = address(this);
    // make the swap
    uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{
        value: amount↑
    }(
        swapOutput, // accept any amount of Tokens
        path,
        deadWallet, // Burn address
        block.timestamp + 300
    );
    emit SwapETHForTokens(amount↑, path);
}
```

#### ❖ Centralization Risk

No Centralization risk found

# Owner privileges

- ❖ The owner can include/exclude wallets from the fee

```
ftrace | funcSig
function excludeFromFee(address account↑) external onlyOwner {
    _isExcludedFromFee[account↑] = true;
    emit AuditLog(
        "We have excluded the following wallet in fees:",
        account↑
    );
}

//include wallet back in fees.
ftrace | funcSig
function includeInFee(address account↑) external onlyOwner {
    _isExcludedFromFee[account↑] = false;
    emit AuditLog(
        "We have including the following wallet in fees:",
        account↑
    );
}

//set a new marketing wallet.
```

- ❖ The owner can change the marketing wallet

```
//set a new marketing wallet.
ftrace | funcSig
function setMarketingWalletAddress(address _marketingWallet↑)
    external
    onlyOwner
{
    require(
        _marketingWallet↑ != address(0),
        "setMarketingWalletAddress: ZERO"
    );
    marketingWalletAddress = payable(_marketingWallet↑);
    emit AuditLog(
        "We have Updated the MarketingWallet:",
        marketingWalletAddress
    );
}
```

- ❖ The owner can set all fees 0, once them set to 0 cannot take fees again

```
ftrace | funcSig
function setZeroMarketingFee() external onlyOwner {
    require(
        _saleMarketingFee != 0,
        "Your MarketingFee is currently set to 0."
    );
    _saleMarketingFee = 0;
    totalSwapableSaleFee = 0;
}
```

- ❖ The owner can take BNB from the contract to marketing wallet

```
// Withdraw ETH that's potentially stuck in the Contract
ftrace | funcSig
function recoverETHfromContract() external onlyOwner {
    payable(marketingWalletAddress).transfer(address(this).balance);
    emit AuditLog(
        "We have recover the stock eth from contract.",
        marketingWalletAddress
    );
}
```

- ❖ The owner can get any BEP20 tokens from contract to marketing wallet (cannot take native tokens)

```
// Withdraw ERC20 tokens that are potentially stuck in Contract
ftrace | funcSig
function recoverTokensFromContract(address _tokenAddress↑, uint256 _amount↑)
    external
    onlyOwner
{
    require(
        _tokenAddress↑ != address(this),
        "Owner can't claim contract's balance of its own tokens"
    );
    IERC20(_tokenAddress↑).transfer(marketingWalletAddress, _amount↑);
    emit Log("We have recovered tokens from contract:", _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: **PASS**

Number of risk issues: **4**

Solidity code functional issue level: **PASS**

Number of owner privileges: **5**

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

Smart contract active ownership: **ACTIVE**