



# **RugFreeCoins Audit**



## **Blue Pepe Token Smart Contract Security Audit**

**July 27<sup>th</sup> ,2023**



# 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 once it's enabled
- ✗ The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.
- ✓ The owner can't change fees.
- ✓ 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.

- **High severity issues**

The owner has the ability to change the reward token address, and they also have the power to stop selling by changing it to an un-swappable token.

```
function updateRewardToken(address _newContract) external onlyOwner {  
    rewardToken = _newContract;  
    dividendTracker.setRewardTokenAddress(_newContract);  
}
```

The owner has the ability to set the swap point to a very high limit. However, it's important to note that if the owner changes this limit to an excessively high amount, there is a risk of the swap failing when the contract attempts to sell a very large amount at once.

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {  
    require(  
        newAmount > totalSupply() / 100_000,  
        "SwapTokensAtAmount must be greater than 0.001% of total supply"  
    );  
    swapTokensAtAmount = newAmount;  
}
```

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading already enabled.");  
    tradingEnabled = true;  
    swapEnabled = true;  
}
```

# Contents

Overview	ii
Audit details	1
Disclaimer	2
Background	3
Target market and the concept	5
Potential to grow with score points	6
Total Points	6
Contract details	7
Contract code function details	8
Contract description table	10
Security issue checking status	22
Owner privileges	24
Audit conclusion	28

# Audit details



**Audited project**  
Blue Pepe Token



**Contract Address**  
0x453259276593d4fD1c8811046f8ab5a6F1833b2b



**Client contact**  
Blue Pepe Token Team



**Blockchain**  
Binance Smart chain



**Project website**  
<https://blue-pepe.vip>

# 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 Blue Pepe Token Team to perform an audit of the smart contract.

<https://bscscan.com/address/0x453259276593d4fD1c8811046f8ab5a6F1833b2b>

This audit focuses on verifying 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.

# Tokenomics

## 10% tax when buying & selling

- 6% of trade goes to the treasury wallet in BNB
- 3% trade distributes among holders as rewards in XRP.
- 1% 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 taking part in the Blue Pepe token ecosystem.
- Anyone who's interested in taking part in the future plans of Blue Pepe Token.
- Anyone who's interested in making financial transactions with any other party using Blue Pepe Token as the currency.

# Potential to grow with score points

1.	Project efficiency	8/10
2.	Project uniqueness	7/10
3	Information quality	8/10
4	Service quality	8/10
5	System quality	8/10
6	Impact on the community	8/10
7	Impact on the business	9/10
8	Preparing for the future	8/10
9	Smart contract security	7/10
10	Smart contract functionality assessment	9/10
Total Points		<b>8.0/10</b>

# Contract details

## Token contract details for 27<sup>th</sup> of July 2023

<b>Contract name</b>	Blue Pepe
<b>Contract address</b>	0x453259276593d4fD1c8811046f8ab5a6F1833b2b
<b>Token supply</b>	1,000,000,000,000
<b>Token ticker</b>	BPP
<b>Decimals</b>	9
<b>Token holders</b>	1
<b>Transaction count</b>	1
<b>Contract deployer address</b>	0x67377ccddd42662e2367dc0476fea581742f6539
<b>Contract 's current owner address</b>	0x67377cCDdD42662E2367dc0476FeA581742F6539
<b>Reward Token</b>	0x1d2f0da169ceb9fc7b3144628db156f3f6c60dbe
<b>Treasury wallet</b>	0x67377ccddd42662e2367dc0476fea581742f6539
<b>Dividend Tracker</b>	0x7943d265b961f26c5834f9f9d4c8741388abbe85

# 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 & centralization	Access control of owners	High 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








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 		
Ownable	Implementation	Context		
L		Public !		NO !
L	owner	Public !		NO !
L	renounceOwnership	Public !		onlyOwner
L	transferOwnership	Public !		onlyOwner
L	_transferOwnership	Internal 		
SafeMath	Library			
L	add	Internal 		
L	sub	Internal 		
L	sub	Internal 		
L	mul	Internal 		










L	div	Internal 🔒		
L	div	Internal 🔒		
L	mod	Internal 🔒		
L	mod	Internal 🔒		
<b>SafeMathInt</b>	<b>Library</b>			
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	sub	Internal 🔒		
L	add	Internal 🔒		
L	abs	Internal 🔒		
L	toUint256Safe	Internal 🔒		
<b>SafeMathUint</b>	<b>Library</b>			
L	toInt256Safe	Internal 🔒		
<b>Iterable Mapping</b>	<b>Library</b>			
L	get	Public !		NO !
L	getIndexOfKey	Public !		NO !
L	getKeyAtIndex	Public !		NO !
L	size	Public !		NO !
L	set	Public !	🔴	NO !

















L	remove	Public !		NO !
<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>IUniswapV2 Pair</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	mint	External !		NO !
L	burn	External !		NO !
L	swap	External !		NO !
L	skim	External !		NO !
L	sync	External !		NO !
L	initialize	External !		NO !
IUniswapV2 Router01	Interface			

L	factory	External !		NO !
L	WETH	External !		NO !
L	addLiquidity	External !		NO !
L	addLiquidityETH	External !		NO !
L	removeLiquidity	External !		NO !
L	removeLiquidityETH	External !		NO !
L	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 !
IUniswapV2Router02	Interface	IUniswapV2Router01		
























L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
<b>IERC20</b>	<b>Interface</b>			
L	totalSupply	External !		NO !
L	balanceOf	External !		NO !
L	allowance	External !		NO !
L	transfer	External !		NO !
L	approve	External !		NO !
L	transferFrom	External !		NO !
<b>IERC20 Metadata</b>	<b>Interface</b>	<b>IERC20</b>		
L	name	External !		NO !
L	symbol	External !		NO !
L	decimals	External !		NO !
<b>ERC20</b>	<b>Implementation</b>	<b>Context, IERC20, IERC20 Metadata</b>		
L		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	_transfer	Internal 		
L	_mint	Internal 		
L	_burn	Internal 		
L	_approve	Internal 		
L	_beforeTokenTransfer	Internal 		
<b>Dividend PayingToken Interface</b>	<b>Interface</b>			
L	dividendOf	External !		NO !
L	withdrawDividend	External !		NO !

<b>Dividend Paying TokenOptional Interface</b>	<b>Interface</b>			
L	withdrawableDividendOf	External !		NO !
L	withdrawnDividendOf	External !		NO !
L	accumulativeDividendOf	External !		NO !
<b>Dividend PayingToken</b>	<b>Implementation</b>	<b>ERC20, Ownable, Dividend Paying Token Interface, Dividend Paying Token Optional Interface</b>		
L		Public !	●	ERC20
L	distributeDividends	Public !	●	onlyOwner
L	setRewardTokenAddress	External !	●	onlyOwner
L	withdrawDividend	Public !	●	NO !
L	_withdrawDividendOfUser	Internal 🔒	●	
L	dividendOf	Public !		NO !
L	withdrawableDividendOf	Public !		NO !
L	withdrawnDividendOf	Public !		NO !
L	accumulativeDividendOf	Public !		NO !
L	_transfer	Internal 🔒	●	
L	_mint	Internal 🔒	●	

L	_burn	Internal 🔒	🔴	
L	_setBalance	Internal 🔒	🔴	
<b>Dividend Tracker</b>	<b>Implementation</b>	<b>Ownable, Dividend Paying Token</b>		
L		Public !	🔴	Dividend Paying Token
L	_transfer	Internal 🔒		
L	withdrawDividend	Public !		NO !
L	setRewardTokenAddress	External !	🔴	onlyOwner
L	updateMinimumTokenBalanceForDividends	External !	🔴	onlyOwner
L	excludeFromDividends	External !	🔴	onlyOwner
L	updateClaimWait	External !	🔴	onlyOwner
L	setLastProcessedIndex	External !	🔴	onlyOwner
L	getLastProcessedIndex	External !		NO !
L	getNumberOfTokenHolders	External !		NO !
L	getAccount	Public !		NO !
L	getAccountAtIndex	Public !		NO !
L	canAutoClaim	Private 🗝️		
L	setBalance	External !	🔴	onlyOwner
L	process	Public !	🔴	NO !
L	processAccount	Public !	🔴	onlyOwner

bluepepe	Implementation	ERC20, Ownable		
L		Public !		ERC20
L		External !		NO !
L	claimStuckTokens	External !		NO !
L	isContract	Internal 		
L	sendBNB	Internal 		
L	_setAutomatedMarketMakerPair	Private 		
L	excludeFromFees	External !		onlyOwner
L	isExcludedFromFees	Public !		NO !
L	changeTreasuryWallet	External !		onlyOwner
L	updateRewardToken	External !		onlyOwner
L	enableTrading	External !		onlyOwner
L	_transfer	Internal 		
L	swapAndLiquify	Private 		
L	swapAndSendDividends	Private 		
L	setSwapTokensAtAmount	External !		onlyOwner
L	setSwapEnabled	External !		onlyOwner
L	updateGasForProcessing	Public !		onlyOwner
L	updateMinimumBalanceForDividends	External !		onlyOwner
L	updateClaimWait	External !		onlyOwner

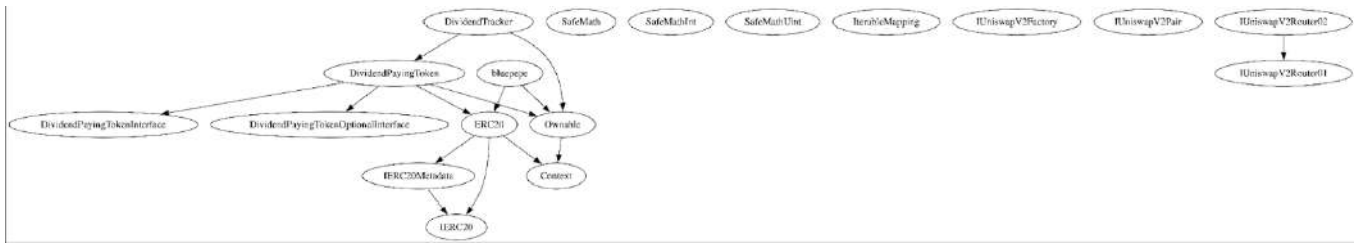


L	getClaimWait	External !		NO !
L	getTotalDividendsDistributed	External !		NO !
L	withdrawableDividendOf	Public !		NO !
L	dividendTokenBalanceOf	Public !		NO !
L	totalRewardsEarned	Public !		NO !
L	excludeFromDividends	External !	🔴	onlyOwner
L	getAccountDividendsInfo	External !		NO !
L	getAccountDividendsInfoAtIndex	External !		NO !
L	processDividendTracker	External !	🔴	NO !
L	claim	External !	🔴	NO !
L	claimAddress	External !	🔴	onlyOwner
L	getLastProcessedIndex	External !		NO !
L	setLastProcessedIndex	External !	🔴	onlyOwner
L	getNumberOfDividendTokenHolders	External !		NO !

### Legend

Symbol	Meaning
🔴	Function can modify state
💰	Function is payable

## Inheritance Hierarchy



# Security issue checking status

## ❖ High severity issues

The owner has the ability to change the reward token address, and they also have the power to stop selling by changing it to an un-swappable token.

```
function updateRewardToken(address _newContract) external onlyOwner {  
    rewardToken = _newContract;  
    dividendTracker.setRewardTokenAddress(_newContract);  
}
```

The owner has the ability to set the swap point to a very high limit. However, it's important to note that if the owner changes this limit to an excessively high amount, there is a risk of the swap failing when the contract attempts to sell a very large amount at once.

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {  
    require(  
        newAmount > totalSupply() / 100_000,  
        "SwapTokensAtAmount must be greater than 0.001% of total supply"  
    );  
    swapTokensAtAmount = newAmount;  
}
```

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading already enabled.");  
    tradingEnabled = true;  
    swapEnabled = true;  
}
```

### ❖ Medium severity issues

The owner can modify the minimum token balance required to receive rewards, and there is no maximum limit on this. Furthermore, the owner can prevent individuals from receiving rewards by setting this limit to an exceptionally high value."

```
function updateMinimumBalanceForDividends(  
    uint256 newMinimumBalance  
) external onlyOwner {  
    dividendTracker.updateMinimumTokenBalanceForDividends(  
        newMinimumBalance  
    );  
}
```

### ❖ Low severity issues

No low-severity issues found.

# Owner privileges

- ❖ Anyone can call the claimStuckTokens function, when someone calls this function with BEP20 token address those tokens will go to the treasury wallet.

```
function claimStuckTokens(address token) external {
    require(token != address(this), "Cannot claim native tokens");
    if (token == address(0x0)) {
        sendBNB(payable(treasuryWallet), address(this).balance);
        return;
    }
    IERC20 ERC20token = IERC20(token);
    uint256 balance = ERC20token.balanceOf(address(this));
    ERC20token.transfer(treasuryWallet, balance);
}
```

- ❖ The owner can include/exclude wallets from fees

```
function excludeFromFees(
    address account,
    bool excluded
) external onlyOwner {
    require(
        _isExcludedFromFees[account] != excluded,
        "Account is already set to that state"
    );
    _isExcludedFromFees[account] = excluded;

    emit ExcludeFromFees(account, excluded);
}
```



- ❖ The owner can change the Treasury wallet address

```
function changeTreasuryWallet(address _treasuryWallet) external onlyOwner {
    require(
        _treasuryWallet != treasuryWallet,
        "Marketing wallet is already that address"
    );
    require(
        !isContract(_treasuryWallet),
        "Marketing wallet cannot be a contract"
    );
    require(
        _treasuryWallet != DEAD,
        "Marketing wallet cannot be the zero address"
    );
    treasuryWallet = _treasuryWallet;
    emit TreasuryWalletChanged(treasuryWallet);
}
```

- ❖ The owner can enable trading, once enabled can not disable it again

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled.");
    tradingEnabled = true;
    swapEnabled = true;
}
```

- ❖ The owner can change the swap token amount

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {
    require(
        newAmount > totalSupply() / 100_000,
        "SwapTokensAtAmount must be greater than 0.001% of total supply"
    );
    swapTokensAtAmount = newAmount;
}
```

- ❖ The owner can enable/disable swapping

```
function setSwapEnabled(bool _enabled) external onlyOwner {  
    require(swapEnabled != _enabled, "swapEnabled already at this state.");  
    swapEnabled = _enabled;  
}
```

- ❖ The owner can change maximum processing gas limit for the reward tracker between 200000-500000

```
function updateGasForProcessing(uint256 newValue) public onlyOwner {  
    require(  
        newValue >= 200_000 && newValue <= 500_000,  
        "gasForProcessing must be between 200,000 and 500,000"  
    );  
    require(  
        newValue != gasForProcessing,  
        "Cannot update gasForProcessing to same value"  
    );  
    emit GasForProcessingUpdated(newValue, gasForProcessing);  
    gasForProcessing = newValue;  
}
```

- ❖ The owner can change the minimum token balance to hold receive rewards

```
function updateMinimumBalanceForDividends(  
    uint256 newMinimumBalance  
) external onlyOwner {  
    dividendTracker.updateMinimumTokenBalanceForDividends(  
        newMinimumBalance  
    );  
}
```

- ❖ The owner can change the minimum time period to receive rewards between 1 hour and 24 hours

```
function updateClaimWait(uint256 newClaimWait) external onlyOwner {  
    require(  
        newClaimWait >= 3_600 && newClaimWait <= 86_400,  
        "claimWait must be updated to between 1 and 24 hours"  
    );  
    dividendTracker.updateClaimWait(newClaimWait);  
}
```

# 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: **3**

Solidity code functional issue level: **PASS**

Number of owner privileges: **9**

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

Smart contract active ownership: **ACTIVE**