



# **RugFreeCoins Audit**



## **Babypepe 2.0 Token Smart Contract Security Audit**

**July 9<sup>th</sup> ,2023**



# Overview

- ✓ No mint function found, the owner cannot mint tokens after initial deployment.
- ✓ The owner can't pause trading
- ✓ 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.
- ✓ The owner can't change fees by more than 20%.
- ✓ The owner can't set a max transaction limit
- ✗ The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

- **High severity issues**

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

```
function enableTrading() public onlyOwner {  
    require(!tradingEnabled, "Trading already enabled!");  
    require(hasLiqBeenAdded, "Liquidity must be added.");  
    tradingEnabled = true;  
    swapEnabled = true;  
    allowedPresaleExclusion = false;  
}
```

# 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	15
Owner privileges	17
Audit conclusion	21

# Audit details



**Audited project**  
Babypepe 2.0 Token



**Contract Address**  
0x6fA33cDf66cBb1ADbeB06C4d60cF3e6d45F13166



**Client contact**  
Babypepe 2.0 Token Team



**Blockchain**  
Binance Smart chain



**Project website**  
<http://babypepe20.baby>

# 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 Babypepe 2.0 Token Team to perform an audit of the smart contract.

<https://bscscan.com/token/0x6fa33cdf66cbb1adbeb06c4d60cf3e6d45f13166#code>

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

- 10% of trade goes to the dev 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 interested in taking part in the Babypepe 2.0 token ecosystem.
- Anyone who's interested in taking part in the future plans of Babypepe 2.0 Token.
- Anyone who's interested in making financial transactions with any other party using Babypepe 2.0 Token as the currency.



# 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	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	9/10
10	Smart contract functionality assessment	9/10
Total Points		<b>8.3/10</b>

# Contract details

## Token contract details for 9<sup>th</sup> of July 2023

<b>Contract name</b>	Babypepe 2.0
<b>Contract address</b>	0x6fA33cDf66cBb1ADbeB06C4d60cF3e6d45F13166
<b>Token supply</b>	420,000,000,000,000,000
<b>Token ticker</b>	Babypepe2.0
<b>Decimals</b>	6
<b>Token holders</b>	1
<b>Transaction count</b>	1
<b>Contract deployer address</b>	0xA889C06Bb24c382790955DC20CC423F2b03672c0
<b>Contract's current owner address</b>	0xA889C06Bb24c382790955DC20CC423F2b03672c0






















# 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	Low issue
		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
<b>SafeMath</b>	<b>Library</b>			
L	add	Internal 		
L	sub	Internal 		
L	sub	Internal 		
L	mul	Internal 		
L	div	Internal 		
L	div	Internal 		
<b>IERC20</b>	<b>Interface</b>			
L	totalSupply	External 		NO 
L	decimals	External 		NO 
L	symbol	External 		NO 
L	name	External 		NO 
L	getOwner	External 		NO 
L	balanceOf	External 		NO 
L	transfer	External 		NO 





L	allowance	External !		NO !
L	approve	External !		NO !
L	transferFrom	External !		NO !
<b>IFactoryV2</b>	<b>Interface</b>			
L	getPair	External !		NO !
L	createPair	External !		NO !
<b>IV2Pair</b>	<b>Interface</b>			
L	factory	External !		NO !
L	getReserves	External !		NO !
L	sync	External !		NO !
<b>IRouter01</b>	<b>Interface</b>			
L	factory	External !		NO !
L	WETH	External !		NO !
L	addLiquidityETH	External !		NO !
L	addLiquidity	External !		NO !
L	swapExactETHForTokens	External !		NO !
L	getAmountsOut	External !		NO !
L	getAmountsIn	External !		NO !

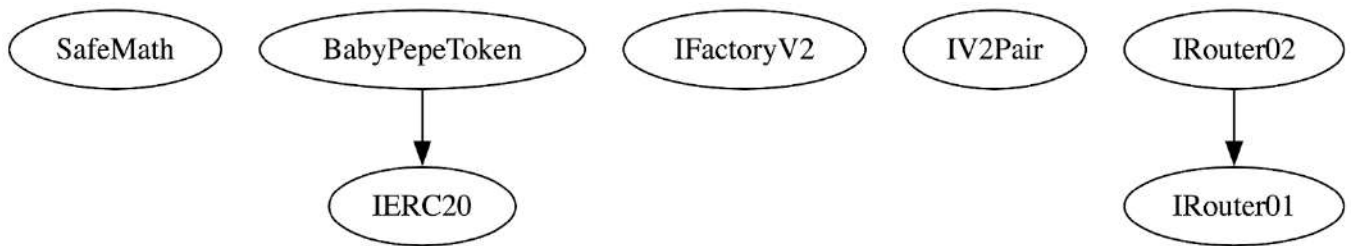
<b>IRouter02</b>	<b>Interface</b>	<b>IRouter01</b>		
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokens	External !		NO !
<b>BabyPepe Token</b>	<b>Implementation</b>	<b>IERC20</b>		
L		Public !		NO !
L		External !		NO !
L	totalSupply	External !		NO !
L	decimals	External !		NO !
L	symbol	External !		NO !
L	name	External !		NO !
L	getOwner	External !		NO !
L	allowance	Public !		NO !
L	balanceOf	Public !		NO !
L	transfer	Public !		NO !
L	approve	External !		NO !
L	_approve	Internal 		
L	approveContractContingency	External !		onlyOwner
L	transferFrom	External !		NO !

L	_hasLimits	Internal 🔒		
L	_transfer	Internal 🔒	🔴	
L	finalizeTransfer	Internal 🔒	🔴	
L	_transferAmount	Internal 🔒	🔴	
L	_checkLiquidityAdd	Internal 🔒	🔴	
L	swapTokensForEth	Private 🔒	🔴	lockThe Swap
L	sendETHToFee	Private 🔒	🔴	
L	manualswap	External !	🔴	NO !
L	manualsend	External !	🔴	NO !
L	transferOwner	External !	🔴	onlyOwner
L	renounceOwnership	External !	🔴	onlyOwner
L	excludePresaleAddresses	External !	🔴	onlyOwner
L	setDevAddress	External !	🔴	onlyOwner
L	excludeMultipleAccountsFromFees	Public !	🔴	onlyOwner
L	enableTrading	Public !	🔴	onlyOwner
L	setFee	Public !	🔴	onlyOwner
L	updateSwapEnabled	External !	🔴	onlyOwner

## Legend

Symbol	Meaning
	Function can modify state
	Function is payable

## Inheritance Hierarchy



# Security issue checking status

## ❖ High severity issues

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

```
function enableTrading() public onlyOwner {
    require(!tradingEnabled, "Trading already enabled!");
    require(hasLiqBeenAdded, "Liquidity must be added.");
    tradingEnabled = true;
    swapEnabled = true;
    allowedPresaleExclusion = false;
}
```

The owner can change the swap token amount without minimum value, the owner can stop trading by setting this to 0

## Informed & Fixed

```
function setSwapNumber(uint256 _numTokensSellToSwap) public onlyOwner {
    numTokensSellToSwap = _numTokensSellToSwap * 10 ** _decimals;
}
```

## ❖ Medium severity issues

No medium severity issues found



### ❖ Low severity issues

When the owner excludes multiple accounts from fees there is no limitation, if the owner excludes large numbers of wallets at a time this function can fail due to out of gas

```
function excludeMultipleAccountsFromFees(  
    address[] calldata _accounts,  
    bool _excluded  
) public onlyOwner {  
    for (uint256 i = 0; i < _accounts.length; i++) {  
        _isExcludedFromFee[_accounts[i]] = _excluded;  
    }  
}
```

### ❖ Centralization Risk

No Centralization issues found

# Owner privileges

- ❖ The owner can give approval to the router address to spend the max amount of tokens from the contract

```
function approveContractContingency() external onlyOwner returns (bool) {  
    _approve(address(this), address(dexRouter), type(uint256).max);  
    return true;  
}
```

- ❖ The owner can dev wallet can manually trigger the swapping

```
function manualswap() external {  
    require(msg.sender == devAddress || msg.sender == _owner);  
    uint256 contractBalance = balanceOf(address(this));  
    swapTokensForEth(contractBalance);  
}
```

- ❖ The owner and dev wallet can send contract BNB to dev address

```
function manualsend() external {  
    require(msg.sender == devAddress || msg.sender == _owner);  
    uint256 contractETHBalance = address(this).balance;  
    sendETHToFee(contractETHBalance);  
}
```

- ❖ The owner can transfer the ownership

```
function transferOwner(address _newOwner) external onlyOwner {
    require(
        _newOwner != address(0),
        "Call renounceOwnership to transfer owner to the zero address."
    );
    require(
        _newOwner != DEAD,
        "Call renounceOwnership to transfer owner to the zero address."
    );
    if (balanceOf(_owner) > 0) {
        finalizeTransfer(_owner, _newOwner, balanceOf(_owner));
    }

    address oldOwner = _owner;
    _owner = _newOwner;
    _isExcludedFromFee[_owner] = true;

    emit OwnershipTransferred(oldOwner, _newOwner);
}
```

- ❖ The owner can renounce ownership

```
function renounceOwnership() external onlyOwner {
    address oldOwner = _owner;
    _owner = address(0);
    emit OwnershipTransferred(oldOwner, address(0));
}
```

- ❖ The owner can exclude pre-sale address before adding lps

```
function excludePresaleAddresses(
    address _router,
    address _presale
) external onlyOwner {
    require(allowedPresaleExclusion);
    require(
        _router != address(this) &&
        _presale != address(this) &&
        lpPair != _router &&
        lpPair != _presale,
        "Just don't."
    );
    if (_router == _presale) {
        _liquidityHolders[_presale] = true;
    } else {
        _liquidityHolders[_router] = true;
        _liquidityHolders[_presale] = true;
    }
}
```

- ❖ The owner can change the dev wallet address

```
function setDevAddress(address _newAddress) external onlyOwner {
    require(devAddress != address(0), "address cannot be 0");
    devAddress = payable(_newAddress);
}
```

- ❖ The owner include/exclude wallets from fees

```
function excludeMultipleAccountsFromFees(
    address[] calldata _accounts,
    bool _excluded
) public onlyOwner {
    for (uint256 i = 0; i < _accounts.length; i++) {
        _isExcludedFromFee[_accounts[i]] = _excluded;
    }
}
```

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

```
function enableTrading() public onlyOwner {  
    require(!tradingEnabled, "Trading already enabled!");  
    require(hasLiqBeenAdded, "Liquidity must be added.");  
    tradingEnabled = true;  
    swapEnabled = true;  
    allowedPresaleExclusion = false;  
}
```

- ❖ The owner can change sell and buy fees for maximum upto 10% of each fee

```
function setFee(  
    uint256 _taxFeeOnBuy,  
    uint256 _taxFeeOnSell  
) public onlyOwner {  
    require(_taxFeeOnBuy <= 10, "Tax cannot be more than 1.");  
    require(_taxFeeOnSell <= 10, "Tax cannot be more than 1.");  
    taxFeeOnBuy = _taxFeeOnBuy;  
    taxFeeOnSell = _taxFeeOnSell;  
}
```

- ❖ The owner can enable/disable swapping

```
function updateSwapEnabled(bool _enabled) external onlyOwner {  
    swapEnabled = _enabled;  
}
```

- ❖ The owner can change the swap point

```
function setSwapNumber(uint256 _numTokensSellToSwap) public onlyOwner {  
    numTokensSellToSwap = _numTokensSellToSwap * 10 ** _decimals;  
}
```



# 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: **2**

Solidity code functional issue level: **PASS**

Number of owner privileges: **12**

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

Smart contract active ownership: **ACTIVE**