expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

PEPE FLOKI CEO

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

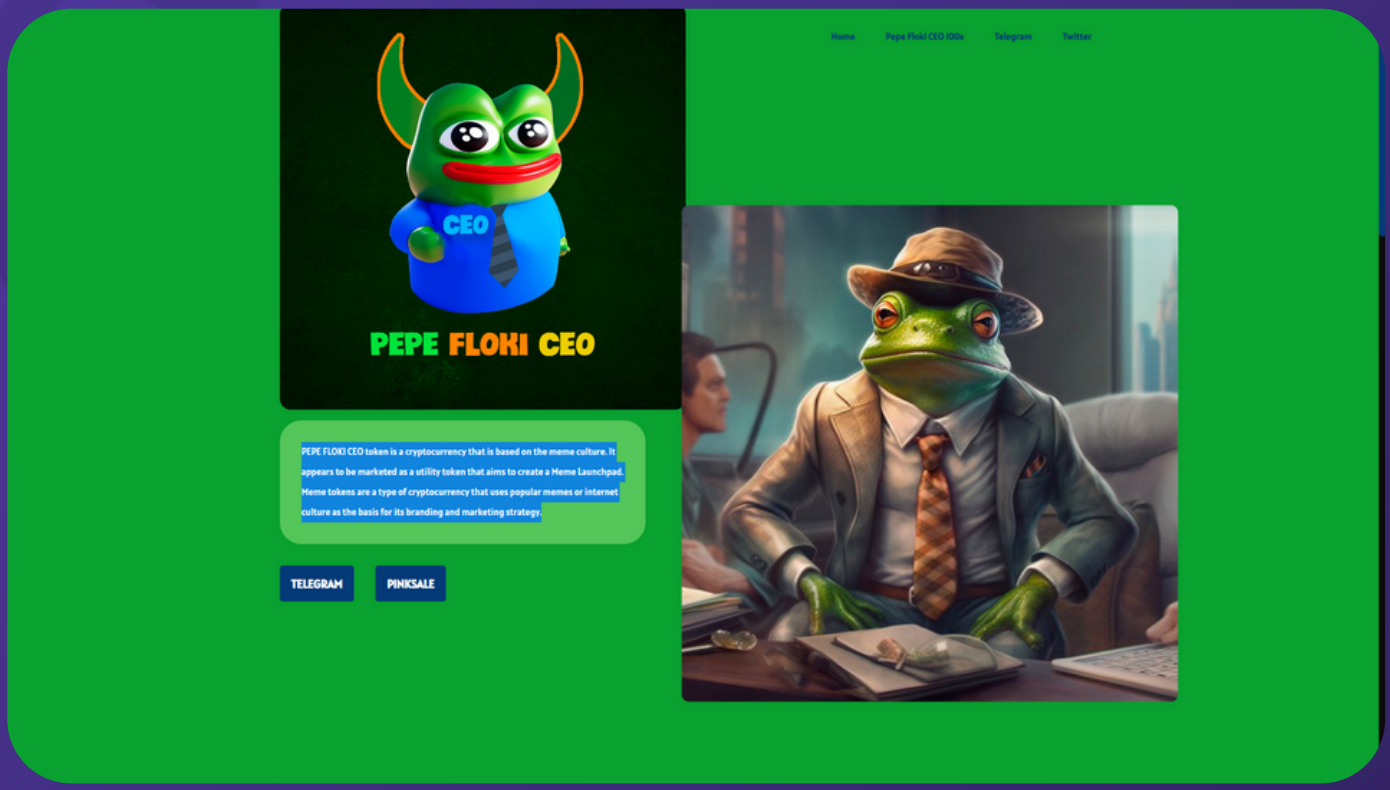
Audit Result	**Passed**
KYC Verification	**No**
Audit Date	**10 May 2023**

PROJECT DESCRIPTION

PEPE FLOKI CEO token is a cryptocurrency that is based on the meme culture. It appears to be marketed as a utility token that aims to create a Meme Launchpad. Meme tokens are a type of cryptocurrency that uses popular memes or internet culture as the basis for its branding and marketing strategy.

SOCIAL MEDIA PROFILES

PEPE FLOKI CEO



https://t.me/pepefloki_ceo

https://twitter.com/pepefloki_ceo

https://pepeflokiceo.com/

It's always good to check the social profiles of the project, before making your investment.

Team Expelee

CONTRACT DETAILS

Token Name: PEPE FLOKI CEO

Symbol: PEPE FLOKI CEO

Network: Arbitrum Chain

Language: Solidity

Contract Address:
0x6C693EE68e796C361a218b4034e317949ac6253A

Total Supply: 888,888,888,888,888

Contract SHA-256 Checksum: -
4bd01b944047a17063c40065d3cd5b3cf364f793

Owner's Wallet:
0xe2bE90542dd42F0be01F2fbD776a3AFB0c16e590

Deployer's Wallet:
0xe2bE90542dd42F0be01F2fbD776a3AFB0c16e590

OWNER PRIVILEGES

- Contract owner is able to set buy/sell/transfer tax each one up to 5%
- Contract owner is not able to blacklist an arbitrary wallet
- Contract owner is not able to set limit for buy/sell/transfer/holding amounts
- Contract owner is not able to mint new tokens
- Contract owner is able to disable trades
- **Contract owner must enable trades for holders manually**

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

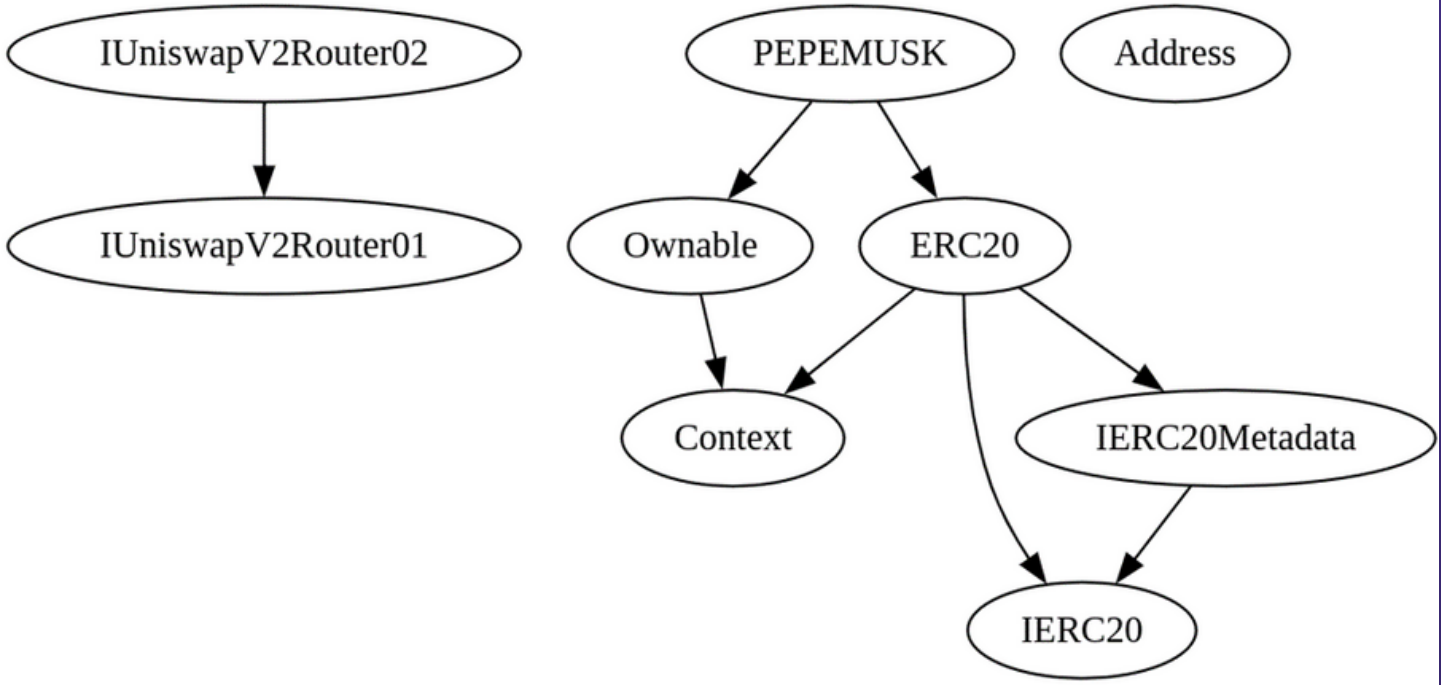
Issues on this level are minor details and warning that can remain unfixed.

Informational

Issues on this level are minor details and warning that can remain unfixed.

expelee

INHERITANCE TREES



TESTNET VERSION

Adding Liquidity ✅ Tx:
https://testnet.bscscan.com/tx/0xb2ca9e4c9c4ae7e611b363ea1a7c7454d6f2ac17f8526793f3fe1639485dcc4e

==========

Buying from a fee excluded wallet ✅ Tx (0% tax):
https://testnet.bscscan.com/tx/0xe01821791e1531de9aceeed2410a7847b2c79090daea8bf61b2e5886cb406c04

==========

Selling from a fee excluded wallet ✅ Tx (0% tax):
https://testnet.bscscan.com/tx/0x9326a81b912dd68fd1be965c4a79840616ba91444aa26a5eac2886e5b1dd1301

==========

Transferring using a fee excluded wallet ✅ Tx (0% tax):
https://testnet.bscscan.com/tx/0x46f472ed49aa12adb7e79ee912c664fbf32709672f081c4cae3e007a1aaac581

==========

TESTNET VERSION

Buying from a regular wallet ✔
Tx (up to 5% tax):
https://testnet.bscscan.com/tx/0xeebdd8930de91123de42b19755ca56fb9770aeb96cd395de6f3566cc66ad23c2

==========

Selling from a regular wallet ✔
Tx (up to 5% tax):
https://testnet.bscscan.com/tx/0xd2f7c55d91519f81c24a3e07c10f36e6b0c366e607540bae493313c51b1fd682

==========

Transferring a regular wallet ✔
Tx (up to 5%):
https://testnet.bscscan.com/tx/0x7512a6de4077496bc4b2adae2f091bb950375c44e83e343743d554fbddab9086

==========

Internal swap (BNB to fee wallets) ✔
Tx :
https://testnet.bscscan.com/tx/0xc04ca33a88a482df6b7c50c9017dda89c0155347d46dac0bc36773a3f76cf1a1

FUNCTION DETAILS

Contract Assesment

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
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 ❗
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 ❗

FUNCTION DETAILS

└	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 ❗	
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 ❗	
IERC20	Interface				
└	totalSupply	External ❗		NO ❗	

expelee

FUNCTION DETAILS

└	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 ❗

Address	Library	
└	isContract	Internal 🔒	
└	sendValue	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🔴	
└	functionDelegateCall	Internal 🔒	🔴	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		

Context	Implementation	
└	_msgSender	Internal 🔒	
└	_msgData	Internal 🔒	

Ownable	Implementation	Context
└	<Constructor>	Public ❗	🔴	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner

ERC20	Implementation	Context, IERC20, IERC20Metadata
└	<Constructor>	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 ❗

FUNCTION DETAILS

└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
PEPEMUSK	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	updateFees	External ❗	🛑	onlyOwner
└	changeMarketingWallet	External ❗	🛑	onlyOwner
└	changeStakingWallet	External ❗	🛑	onlyOwner
└	changeTreasuryWallet	External ❗	🛑	onlyOwner
└	enableTrading	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	swapAndSend	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

expelee

FINDINGS

Findings	Severity	Found
High Risk	High	2
Medium Risk	Medium	0
Low Risk	Low	0
Suggestion & discussion	Informational	0
Gas Optimizations	Gas Opt.	1

HIGH RISK FINDING

Category: Centralization
Subject: Enabling trades is not guaranteed
Overview: The owner of the contract must enable trades for public, otherwise no one would be able to buy/sell/transfer their tokens except whitelisted wallets.

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

Suggestion
To mitigate this issue there are several options:
- Enable tradings before presale
- Transfer ownership to safu dev for initial days after presale

Issue Status: Not Resolved

HIGH RISK FINDING

Category: Logical

Subject: Setting all taxes to 0 can disable sells for everyone

Overview: If all fees are set to 0, since total shares is equal to 0, internal swap would be failed due to a division by zero error.

```
function swapAndSend(uint256 tokenAmount) private {
    uint256 initialBalance = address(this).balance;

address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = uniswapV2Router.WETH();


 uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTo
kens(
tokenAmount,
        0,
path,
address(this),
block.timestamp
    );
```

HIGH RISK FINDING

```
        uint256 newBalance = address(this).balance - initialBalance;

        uint256 totalShare = totalFeesOnBuy + totalFeesOnSell;
        uint256 marketingBalance = (newBalance *
(marketingFeeOnBuy + marketingFeeOnSell)) /
 totalShare;
        uint256 stakingBalance = (newBalance *
(stakingFeeOnBuy + stakingFeeOnSell)) / totalShare;

payable(marketingWallet).sendValue(marketingBalance);
payable(stakingWallet).sendValue(stakingBalance);
payable(treasuryWallet).sendValue(
address(this).balance - initialBalance
        );

        emit SwapAndSend(tokenAmount, newBalance);
    }
```

Suggestion

To mitigate this issue, ensure that internal swap is only performed if total tax is more than 0

Issue Status: Not Resolved

SUGGESTIONS

Category: Informational
Subject: withdraw of native token is not allowed

Overview: Contract doesn't allow withdraw of the PEPEMUSK tokens from the contract.

Suggestion
its suggested to allow owner to be able to withraw PEPMUSK tokens from the contract

Issue Status: Not Resolved

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial
expelee
Expelee
expelee
expelee_official
expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.