RugFreeCoins Audit

Palma Doge Token

Smart Contract Security Audit

September 15, 2021

Contents

Audit details

Audited project

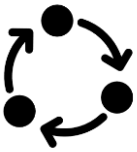
Palma Doge Token

Contract Address

0x83861df7c0cb3aacf4209430f8ab9f7c509c696b

Client contact

Palma Doge Team



Blockchain

Binance smart chain

Project website

https://palmadoge.finance/

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 Palma Doge Token to perform an audit of the smart contract.

https://bscscan.com/address/0x83861df7c0cb3aacf4209430f8ab9f7c509c696b

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

About the project

Palma Doge is a token built on the Binance Smart Chain. Each transaction, purchase incurs a fee of 12%, and sale incurs a fee of 18%. It is the new project which paves the way for a series of projects that will be carried out in the next 3 years with a clear roadmap, they will synchronize the features of the traditional and new schemes.

It will provide to experience a new chain of features from

- Stake and Farm
- NFT Game
- NFT Marketplace
- Jackpot
- Swap market
- App
- Launchpad
- Own Blockchain

Features

- Palmadoge improves on the popular buyback protocol with a new suite of innovations that will help increase returns for investors, which will be exchanging 6% of the tax when buying and 7% tax when selling for BNB buys back from the supply every minute and burn all tokens bought automatically.

- The **sustainability fee of 4% marketing when buying and selling** is what allows PalmaDoge to hold the aforementioned promise. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, PalmaDoge will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.

- The **automatic PalmaDoge rewards of 2% when buying and 7% when selling** is what PalmaDoge's core marketing strategy is based around: that tokens will be distributed among every holder proportional to how many tokens each individual holds.

Tokenomics

12% fee when buying

- 6% of trade goes to the buyback fund.
- 4% of trade goes to the development/marketing wallet.
- 2% of trade goes to holders' pockets in PalmaDoge tokens.

18% fee when selling

- 7% of trade goes to the buyback fund.
- 7% of trade goes to holders' pockets in PalmaDoge tokens.
- 4% of trade goes to the development/marketing wallet.

Roadmap


Q3
•Design & creation
•Team Building
•Initial Marketing
•5000 Telegram Members
•First Audit
•Token Tracker
Q4
•Pre-sale, launch and liquidity lock
•Listing on Coingecko
•Listing on Coin market cap
•PalmaSWAP release
•Influencer marketing campaign
•Increase social channel numbers
•Listing on Cex
Q1
•Second Audit
•Onboard Influencers from Non English natives
•Listing on More Exchanges
•Memes campaign
•Content Creation Campaign
•Key partnerships
•Palma NFT Game launch
Q2
•Website Redesign
•Merch shop
•Team Expansion
•Charity donation to Africa and
other 3rd world countries specifically
•Community meet up
•Intensify Marketing

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 in PalmaDoge tokens by holding tokens.
- Anyone who's interested in trading tokens.
- Anyone who's interested in taking part with the PalmaDoge future app functionalities.
- Anyone who's interested in playing the NFT game and winning rewards.
- Anyone who's interested in collecting NFTs or trading NFTs.
- Anyone who's interested in swap and trade tokens through the future PalmaDoge platform.
- Anyone who's interested in stake PalmaDoge tokens and earn rewards.
- Anyone who's interested in taking part in future PalmaDoge activities and ideas.
- Anyone who's interested in making financial transactions with any other party using PalmaDoge token as the currency.

Core concept

The reward system

2% of each transaction when buying and 7% when selling gets sent amongst all holders in tokens. The holders will be eligible to receive tokens, whenever a transaction occurs, and rewards are proportional to how many tokens each individual holds.

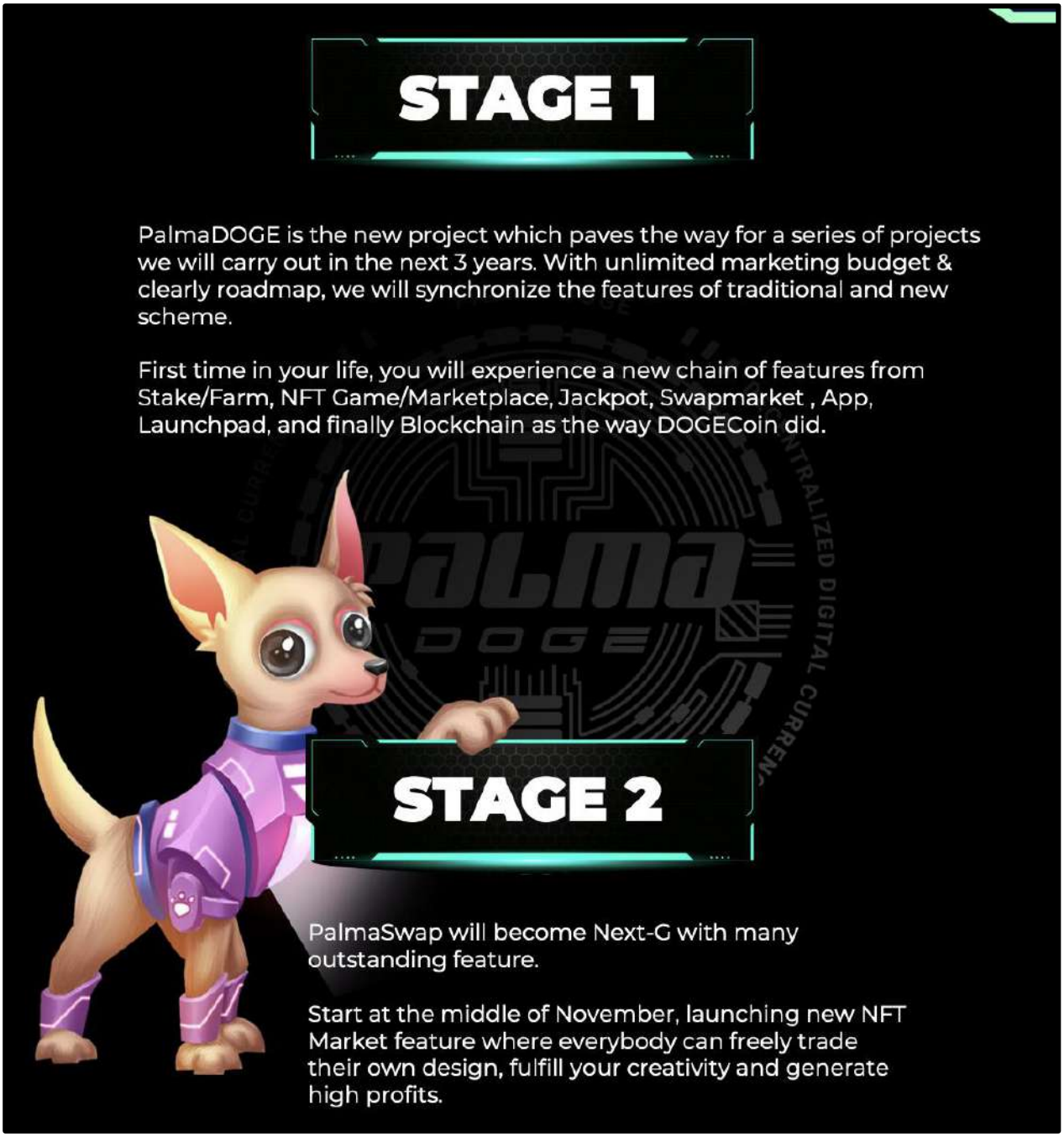
Sustainable mechanism

The **fee of 4% marketing** is what allows PalmaDoge to promote the token and use funds to further development of the platform. Tokens will be swapped into BNBs and will be sent to a marketing wallet per transaction. This way, Palmadoge 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 buyback feature: The buyback and burn mechanism collects 6% tax when buying and 7% when selling, which is stored inside the contract. Whenever a buy or sell occurs, a fraction of the buyback amount is used to automatically purchase tokens from the liquidity pool. Those tokens are immediately burned after purchase, which keeps the token price stable.

The project features

[Click here]



STAGE 3

PalmaLAND – we call it a special feature nowhere found in other projects will release on January 2022

Start with Child DOGE, you will use your NFT cards which owned to attack Zombies and get EXP & PalmaDOGE Coin

You can upgrade your DOGE and buy items to increase power

STAGE 4

PalmaAPP & Blockchain.
Start on June, 2022, we will make a detail introduction once PalmaAPP has been launched

Potential to grow with score points

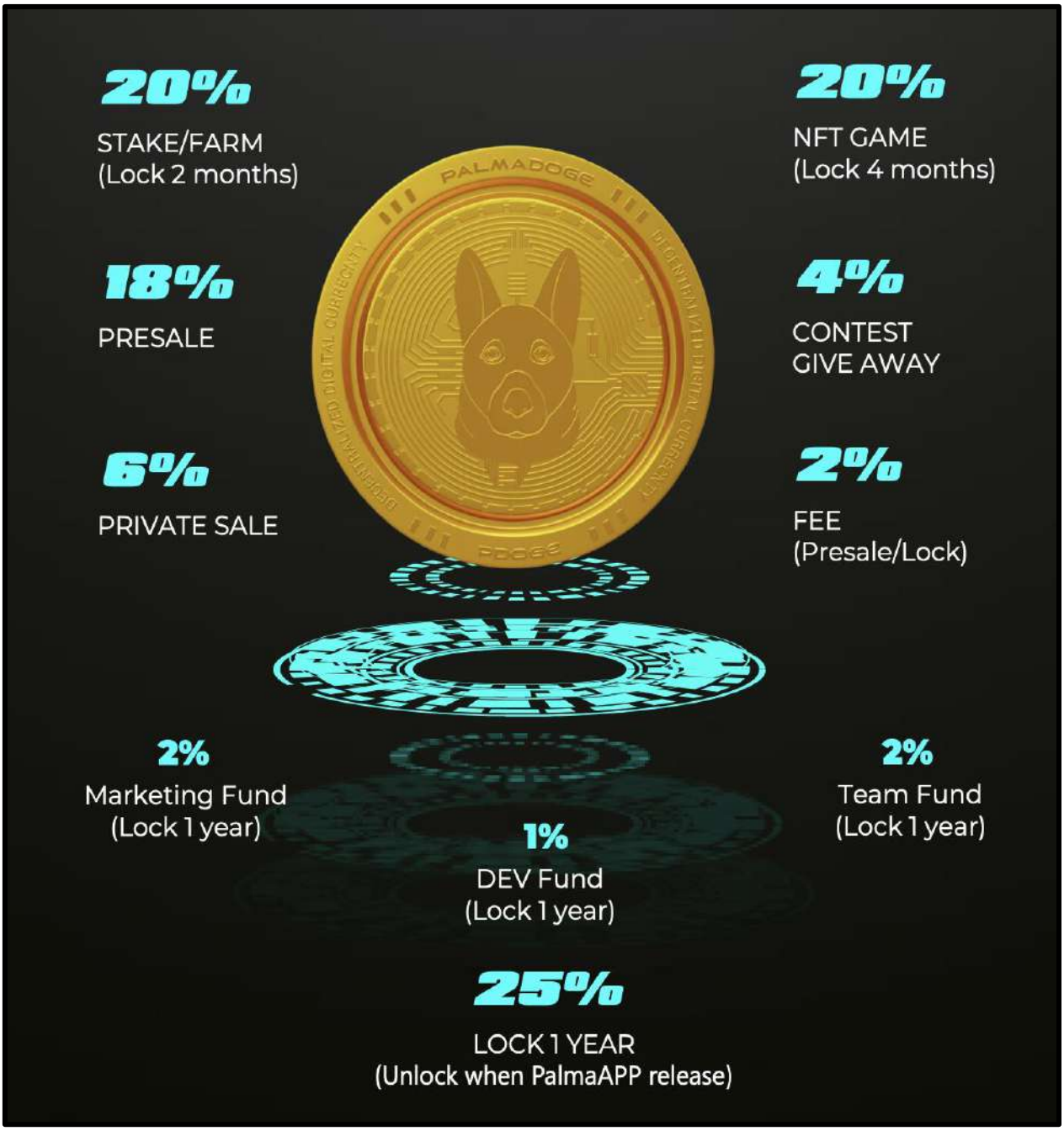
1.	Project efficiency	9/10
2.	Project uniqueness	9/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	8/10
7	Impact on the business	9/10
8	Preparing for the future	9/10
Total Points		**8.88/10**

Contract details

Token contract details for 15th September 2021

Contract name	PalmaDOGE
Contract address	0x83861df7c0cb3aacf4209430f8ab9f7c509c696b
Token supply	1,000,000,000
Token ticker	PALDOG
Decimals	9
Token holders	5
Transaction count	9
Charity address	0x818a6744563d05940609c586aceb2dcf00932498
Contract deployer address	0xe9051f9414D2765658A7c2e1f7C27A2CbE4C6355
Contract's current owner address	0xe9051f9414d2765658a7c2e1f7c27a2cbe4c6355

Token distribution



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	pass
		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

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 its visibility and mutability.

Contract	**Type**	**Bases**		
└	Function Name	Visibility	Mutability	Modifiers
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	**Interface**			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeMath	**Library**			
└	add	Internal 🔒		

└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
Address	**Library**			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	_functionCallWithValue	Private 🔐	🛑	
Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO❗

└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔘	onlyOwner
└	transferOwnership	Public ❗	🔘	onlyOwner
└	getUnlockTime	Public ❗		NO❗
└	getTime	Public ❗		NO❗
└	lock	Public ❗	🔘	onlyOwner
└	unlock	Public ❗	🔘	NO❗
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❗
└	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❗
└	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❗
PalmaDOGE	**Implementation**	**Context, IERC20, Ownable**		
└		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❗
└	isExcludedFromReward	Public ❗		NO❗

└	totalFees	Public ❗		NO❗
└	minimumTokensBeforeSwapAmount	Public ❗		NO❗
└	buyBackSellLimitAmount	Public ❗		NO❗
└	deliver	Public ❗	🔘	NO❗
└	reflectionFromToken	Public ❗		NO❗
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	Public ❗	🔘	onlyOwner
└	includeInReward	External ❗	🔘	onlyOwner
└	_approve	Private 🔐	🔘	
└	_transfer	Private 🔐	🔘	
└	swapTokens	Private 🔐	🔘	lockTheSwap
└	buyBackTokens	Private 🔐	🔘	lockTheSwap
└	swapTokensForEth	Private 🔐	🔘	
└	swapETHForTokens	Private 🔐	🔘	
└	addLiquidity	Private 🔐	🔘	
└	_tokenTransfer	Private 🔐	🔘	
└	_transferStandard	Private 🔐	🔘	
└	_transferToExcluded	Private 🔐	🔘	

└	_transferFromExcluded	Private 🔐	🛑	
└	_transferBothExcluded	Private 🔐	🛑	
└	_reflectFee	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	calculateTaxFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	isExcludedFromFee	Public ❗		NO❗
└	excludeFromFee	Public ❗	🛑	onlyOwner
└	includeInFee	Public ❗	🛑	onlyOwner
└	excludeFromSwapAndLiquify	Public ❗	🛑	onlyOwner
└	includeInSwapAndLiquify	Public ❗	🛑	onlyOwner

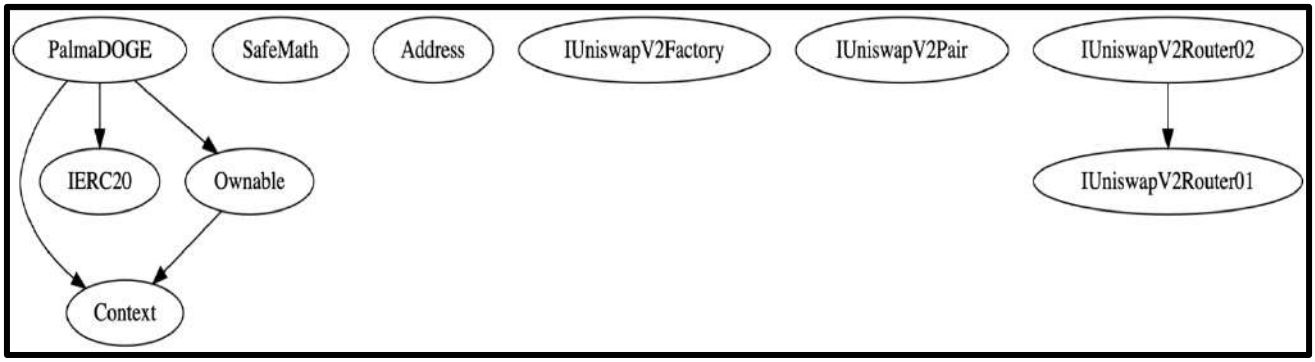
└	_getSellBnBAmount	Private 🔐		
└	_removeOldSellHistories	Private 🔐	🛑	
└	SetBuyBackMaxTimeForHistories	External ❗	🛑	onlyOwner
└	SetBuyBackDivisor	External ❗	🛑	onlyOwner
└	GetBuyBackTimeInterval	Public ❗		NO❗
└	SetBuyBackTimeInterval	External ❗	🛑	onlyOwner
└	SetBuyBackRangeRate	External ❗	🛑	onlyOwner
└	GetSwapMinutes	Public ❗		NO❗
└	SetSwapMinutes	External ❗	🛑	onlyOwner
└	setTaxFeePercent	External ❗	🛑	onlyOwner
└	setBuyFee	External ❗	🛑	onlyOwner
└	setSellFee	External ❗	🛑	onlyOwner
└	setLiquidityFeePercent	External ❗	🛑	onlyOwner
└	setBuyBackSellLimit	External ❗	🛑	onlyOwner
└	setMaxTxAmount	External ❗	🛑	onlyOwner
└	setCharityDivisor	External ❗	🛑	onlyOwner
└	setNumTokensSellToAddToBuyBack	External ❗	🛑	onlyOwner
└	setcharityAddress	External ❗	🛑	onlyOwner

L	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
L	setBuyBackEnabled	Public ❗	🛑	onlyOwner
L	setAutoBuyBackEnabled	Public ❗	🛑	onlyOwner
L	prepareForPreSale	External ❗	🛑	onlyOwner
L	afterPreSale	External ❗	🛑	onlyOwner
L	transferToAddressETH	Private 🔐	🛑	
L	changeRouterVersion	Public ❗	🛑	onlyOwner
L		External ❗	💶	NO❗
L	transferForeignToken	Public ❗	🛑	onlyOwner
L	Sweep	External ❗	🛑	onlyOwner
L	setAddressFee	External ❗	🛑	onlyOwner
L	setBuyAddressFee	External ❗	🛑	onlyOwner
L	setSellAddressFee	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**
  - **No low severity issues found**
- **Informational**
  - **The owner can change the max transaction amount without a limit.**

```
ftrace | funcSig
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner {
    _maxTxAmount = maxTxAmount;
}
```

Recommendation

Set validation to minimum transaction amount can be set.

- **The owner can change all fees without a limit.**

```
ftrace | funcSig
function setTaxFeePercent(uint256 taxFee) external onlyOwner {
    _taxFee = taxFee;
}

ftrace | funcSig
function setBuyFee(uint256 buyTaxFee, uint256 buyLiquidityFee)
    external
    onlyOwner
{
    _buyTaxFee = buyTaxFee;
    _buyLiquidityFee = buyLiquidityFee;
}

ftrace | funcSig
function setSellFee(uint256 sellTaxFee, uint256 sellLiquidityFee)
    external
    onlyOwner
{
    _sellTaxFee = sellTaxFee;
    _sellLiquidityFee = sellLiquidityFee;
}

ftrace | funcSig
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner {
    _liquidityFee = liquidityFee;
}
```

```
ftrace | funcSig
function includeInReward(address account) external onlyOwner {
    require(_isExcluded[account], "Account is not excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

Recommendation

Set validation to maximum fees can be set.

- **The owner can change charity address at any time**

  The owner can change the charity address at any time so he can put his own account to get a charity reward.

Recommendation

Remove set charity function.

Owner privileges

- The owner can include and exclude wallets from the reward.

```
ftrace | funcSig
function excludeFromReward(address account⬆) public onlyOwner {
    require(!_isExcluded[account⬆], "Account is already excluded");
    if (_rOwned[account⬆] > 0) {
        _tOwned[account⬆] = tokenFromReflection(_rOwned[account⬆]);
    }
    _isExcluded[account⬆] = true;
    _excluded.push(account⬆);
}

ftrace | funcSig
function includeInReward(address account⬆) external onlyOwner {
    require(_isExcluded[account⬆], "Account is not excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account⬆) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account⬆] = 0;
            _isExcluded[account⬆] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- The owner can include and exclude wallets from fees.

```
ftrace | funcSig
function excludeFromFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = true;
}

ftrace | funcSig
function includeInFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = false;
}
```

❖ The owner can include/exclude wallets from the swap and liquify.

```
ftrace | funcSig
function excludeFromSwapAndLiquify(address account⬆) public onlyOwner {
    _isExcludedFromSwapAndLiquify[account⬆] = true;
}

ftrace | funcSig
function includeInSwapAndLiquify(address account⬆) public onlyOwner {
    _isExcludedFromSwapAndLiquify[account⬆] = false;
}
```

❖ The owner can change buyback max time and buy back devisor.

```
ftrace | funcSig
function SetBuyBackMaxTimeForHistories(uint256 newMinutes⬆)
    external
    onlyOwner
{
    _buyBackMaxTimeForHistories = newMinutes⬆ * 1 minutes;
}

ftrace | funcSig
function SetBuyBackDivisor(uint256 newDivisor⬆) external onlyOwner {
    _buyBackDivisor = newDivisor⬆;
}
```

❖ The owner can set a buyback max range.

```
ftrace | funcSig
function SetBuyBackRangeRate(uint256 newPercent⬆) external onlyOwner {
    require(newPercent⬆ <= 100, "The value must not be larger than 100.");
    _buyBackRangeRate = newPercent⬆;
}
```

❖ The owner can set max swap minutes.

```
ftrace | funcSig
function SetSwapMinutes(uint256 newMinutes⬆) external onlyOwner {
    _intervalMinutesForSwap = newMinutes⬆ * 1 minutes;
}
```

❖ The owner can set all sell fees and buy fees.

```
ftrace | funcSig
function setTaxFeePercent(uint256 taxFee⬆) external onlyOwner {
    _taxFee = taxFee⬆;
}

ftrace | funcSig
function setBuyFee(uint256 buyTaxFee⬆, uint256 buyLiquidityFee⬆)
    external
    onlyOwner
{
    _buyTaxFee = buyTaxFee⬆;
    _buyLiquidityFee = buyLiquidityFee⬆;
}

ftrace | funcSig
function setSellFee(uint256 sellTaxFee⬆, uint256 sellLiquidityFee⬆)
    external
    onlyOwner
{
    _sellTaxFee = sellTaxFee⬆;
    _sellLiquidityFee = sellLiquidityFee⬆;
}

ftrace | funcSig
function setLiquidityFeePercent(uint256 liquidityFee⬆) external onlyOwner {
    _liquidityFee = liquidityFee⬆;
}
```

❖ The owner can change buyback sell limit.

```
ftrace | funcSig
function setBuyBackSellLimit(uint256 buyBackSellSetLimit⬆)
    external
    onlyOwner
{
    buyBackSellLimit = buyBackSellSetLimit⬆;
}
```

❖ The owner can change the max transaction amount.

```
ftrace | funcSig
function setMaxTxAmount(uint256 maxTxAmount⬆) external onlyOwner {
    _maxTxAmount = maxTxAmount⬆;
}
```

❖ The owner can change the charity divisor.

```
ftrace | funcSig
function setCharityDivisor(uint256 divisor⬆) external onlyOwner {
    CharityDivisor = divisor⬆;
}
```

❖ The owner can change charity address.

```
ftrace | funcSig
function setcharityAddress(address _charityAddress⬆) external onlyOwner {
    charityAddress = payable(_charityAddress⬆);
}
```

❖ The owner can enable and disable swap and liquify, buyback, and auto buyback.

```
ftrace | funcSig
function setSwapAndLiquifyEnabled(bool _enabled⬆) public onlyOwner {
    swapAndLiquifyEnabled = _enabled⬆;
    emit SwapAndLiquifyEnabledUpdated(_enabled⬆);
}

ftrace | funcSig
function setBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    buyBackEnabled = _enabled⬆;
    emit BuyBackEnabledUpdated(_enabled⬆);
}

ftrace | funcSig
function setAutoBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    _isAutoBuyBack = _enabled⬆;
    emit AutoBuyBackEnabledUpdated(_enabled⬆);
}
```

❖ The owner can transfer all foreign tokens to any wallet.

```
ftrace | funcSig
function transferForeignToken(address _token⬆, address _to⬆)
    public
    onlyOwner
    returns (bool _sent⬆)
{
    require(_token⬆ != address(this), "Can't let you take all native token");
    uint256 _contractBalance = IERC20(_token⬆).balanceOf(address(this));
    _sent⬆ = IERC20(_token⬆).transfer(_to⬆, _contractBalance);
}
```

❖ The owner can transfer contract BNB balance to owner's wallet.

```
ftrace | funcSig
function Sweep() external onlyOwner {
    uint256 balance = address(this).balance;
    payable(owner()).transfer(balance);
}
```

❖ The owner can add extra fees to each wallet.

```
ftrace | funcSig
function setAddressFee(
    address _address⬆,
    bool _enable⬆,
    uint256 _addressTaxFee⬆,
    uint256 _addressLiquidityFee⬆
) external onlyOwner {
    _addressFees[_address⬆].enable = _enable⬆;
    _addressFees[_address⬆]._taxFee = _addressTaxFee⬆;
    _addressFees[_address⬆]._liquidityFee = _addressLiquidityFee⬆;
}

ftrace | funcSig
function setBuyAddressFee(
    address _address⬆,
    bool _enable⬆,
    uint256 _addressTaxFee⬆,
    uint256 _addressLiquidityFee⬆
) external onlyOwner {
    _addressFees[_address⬆].enable = _enable⬆;
    _addressFees[_address⬆]._buyTaxFee = _addressTaxFee⬆;
    _addressFees[_address⬆]._buyLiquidityFee = _addressLiquidityFee⬆;
}

ftrace | funcSig
function setSellAddressFee(
    address _address⬆,
    bool _enable⬆,
    uint256 _addressTaxFee⬆,
    uint256 _addressLiquidityFee⬆
) external onlyOwner {
    _addressFees[_address⬆].enable = _enable⬆;
    _addressFees[_address⬆]._sellTaxFee = _addressTaxFee⬆;
    _addressFees[_address⬆]._sellLiquidityFee = _addressLiquidityFee⬆;
}
```

Audit conclusion

While conducting the audit of the Palma Doge smart contract, it was observed that there is nothing alarming with the code and it only contains informational concerns.