auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

PEPE KING ETH

DATED : 12 May 23'

AUDIT SUMMARY

Project name – PEPE KING ETH

Date: 12 May, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	2	0	2
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/address/0x2982ED75e608eB278d44176259d891675301f535#code

Token Information

Token Name : PEPE KING ETH

Token Symbol: PEPEK

Decimals: 9

Token Supply: 420,690,000,000,000

Token Address:
0x459a5F4516AE9c7d4aaEA8E67e7b7B00b39a0A69

Checksum:
f9b6e68f6fa590349d1f16400b3b4d6fcafcac63

Owner:
0xEE433d098980d0Fd8590448D072A142d9632Ec1A
(at time of writing the audit)

Deployer:
0xEE433d098980d0Fd8590448D072A142d9632Ec1A

TOKEN OVERVIEW

Fees:

Buy Fees: 0%

Sell Fees: 8%

Transfer Fees: 0%

Fees Privilege: None

Ownership: Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: including and excluding form fee - changing distribution settings - changing internal swap settings

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- Return values of low-level calls
- Private modifier
- Multiple Sends
- Using Suicide
- Gas Limitand Loops
- Address hardcoded
- Exception Disorder
- Using inline assembly
- Divide before multiply
- Missing Zero Address Validation
- Compiler version not fixed
- **Gasless Send**
- Using block.timestamp
- Re-entrancy
- Tautology or contradiction
- Timestamp Dependence
- Revert/require functions
- Use of tx.origin
- Integer overflow/underflow
- Dangerous strict equalities
- Using SHA3
- Using throw

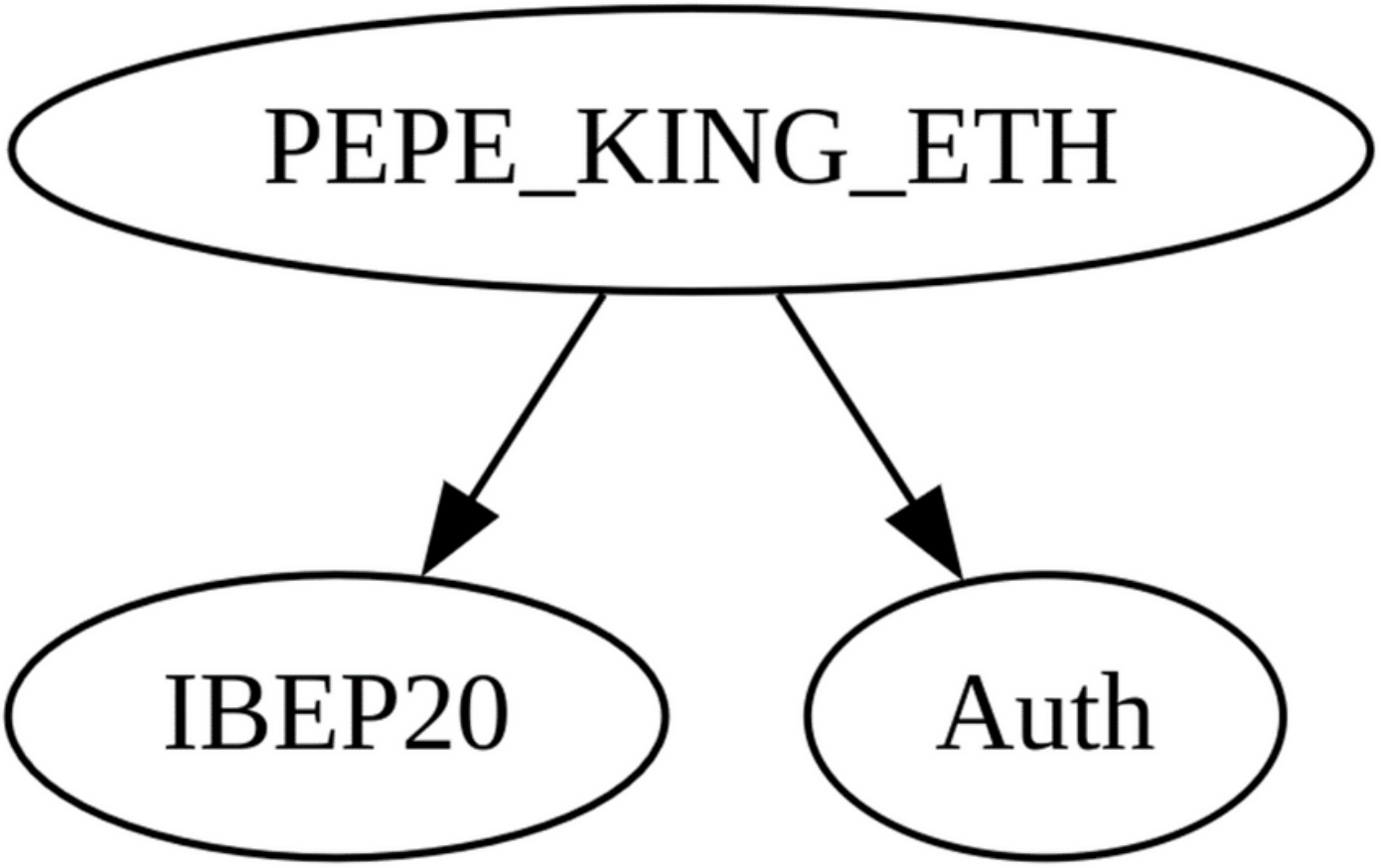
CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	0
Medium-Risk	2
Low-Risk	0
Gas Optimization / Suggestions	2

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change fees (0% buy | 8% sell | 0% transfer)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC165	Interface			
└	supportsInterface	External ❗		NO ❗
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
IBEP20	Interface			
└	totalSupply	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	name	External ❗		NO ❗
└	getOwner	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🔘	NO ❗
└	burn	External ❗	🔘	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🔘	NO ❗
└	transferFrom	External ❗	🔘	NO ❗
Auth	Implementation			
└	<Constructor>	Public ❗	🔘	NO ❗
└	isOwner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🔘	onlyOwner
└	transferOwnership	Public ❗	🔘	onlyOwner
IDEXFactory	Interface			

CONTRACT ASSESMENT

└	createPair	External ❗	🔴	NO ❗
IDEXRouter	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidity	External ❗	🔴	NO ❗
└	addLiquidityETH	External ❗	💸	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💸	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
IDividendDistributor	Interface			
└	setDistributionCriteria	External ❗	🔴	NO ❗
└	setShare	External ❗	🔴	NO ❗
└	deposit	External ❗	💸	NO ❗
└	process	External ❗	🔴	NO ❗
DividendDistributor	Implementation	IDividendDistributor		
└	<Constructor>	Public ❗	🔴	NO ❗
└	setDistributionCriteria	External ❗	🔴	onlyToken
└	setShare	External ❗	🔴	onlyToken
└	deposit	External ❗	💸	onlyToken
└	process	External ❗	🔴	onlyToken
└	shouldDistribute	Internal 🔒		
└	distributeDividend	Internal 🔒	🔴	
└	claimDividend	External ❗	🔴	onlyToken
└	getUnpaidEarnings	Public ❗		NO ❗
└	getCumulativeDividends	Internal 🔒		
└	addShareholder	Internal 🔒	🔴	
└	removeShareholder	Internal 🔒	🔴	
└	setDividendTokenAddress	External ❗	🔴	onlyToken
PEPE_KING_ETH	Implementation	IBEP20, Auth		
└	<Constructor>	Public ❗	🔴	Auth
└	<Receive Ether>	External ❗	💸	NO ❗
└	totalSupply	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	name	External ❗		NO ❗
└	getOwner	External ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	allowance	External ❗		NO ❗

CONTRACT ASSESMENT

└	approve	Public ❗	🛑	NO ❗
└	approveMax	External ❗	🛑	NO ❗
└	setIsFeeExempt	External ❗	🛑	onlyOwner
└	burn	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	_transferFrom	Internal 🔒	🛑	
└	_basicTransfer	Internal 🔒	🛑	
└	shouldTakeFee	Internal 🔒		
└	shouldTakeFeer	Internal 🔒		
└	getTotalFee	Public ❗		NO ❗
└	getMultipliedFee	Public ❗		NO ❗
└	takeFee	Internal 🔒	🛑	
└	shouldSwapBack	Internal 🔒		
└	swapBack	Internal 🔒	🛑	swapping
└	buyTokens	Internal 🔒	🛑	swapping
└	launched	Internal 🔒		
└	setIsDividendExempt	External ❗	🛑	onlyOwner
└	setFeeReceivers	External ❗	🛑	onlyOwner
└	setSwapBackSettings	External ❗	🛑	onlyOwner
└	setTargetLiquidity	External ❗	🛑	onlyOwner
└	manualSend	External ❗	🛑	NO ❗
└	setDistributionCriteria	External ❗	🛑	onlyOwner
└	claimDividend	External ❗	🛑	NO ❗
└	getUnpaidEarnings	Public ❗		NO ❗
└	setDistributorSettings	External ❗	🛑	onlyOwner
└	getCirculatingSupply	Public ❗		NO ❗
└	getLiquidityBacking	Public ❗		NO ❗
└	isOverLiquified	Public ❗		NO ❗

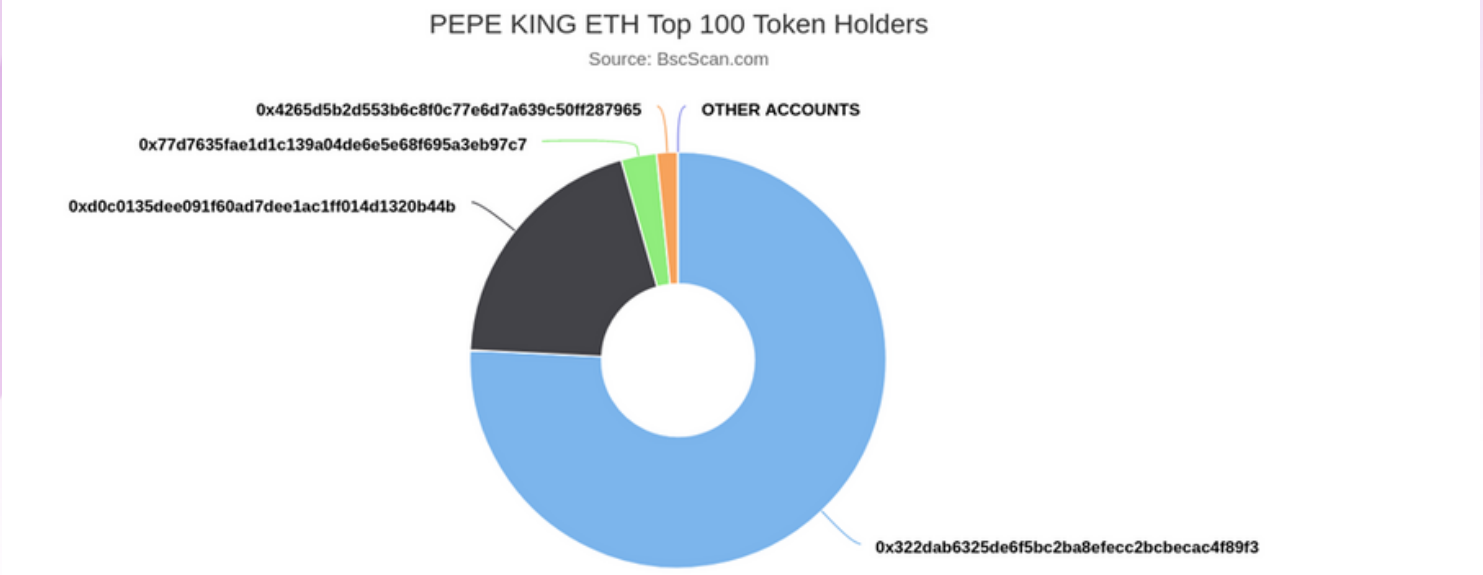
Legend

Symbol	Meaning
🛑	Function can modify state
💲	Function is payable

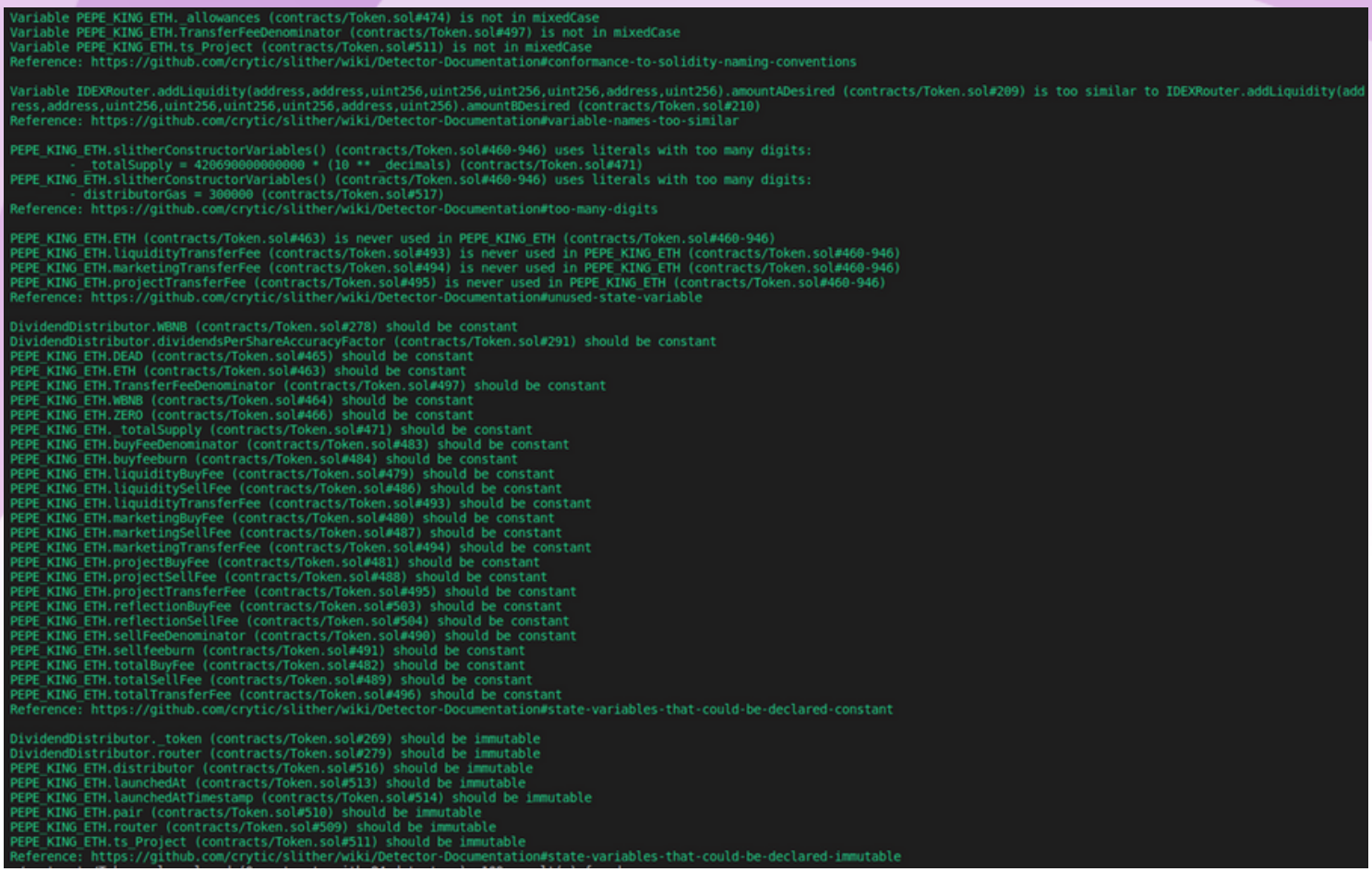
TOKEN DISTRIBUTION AT TIME OF AUDIT

The top 100 holders collectively own 100.00% (420,690,000,000,000.00 Tokens) of PEPE KING ETH

Token Total Supply: 420,690,000,000,000.00 Token | Total Token Holders: 4



STATIC ANALYSIS



```
Variable PEPE_KING_ETH._allowances (contracts/Token.sol#474) is not in mixedCase
Variable PEPE_KING_ETH.TransferFeeDenominator (contracts/Token.sol#497) is not in mixedCase
Variable PEPE_KING_ETH.ts_Project (contracts/Token.sol#511) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IDEXRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#209) is too similar to IDEXRouter.addLiquidity(add
ress,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

PEPE_KING_ETH.slitherConstructorVariables() (contracts/Token.sol#460-946) uses literals with too many digits:
        - _totalSupply = 420690000000000 * (10 ** _decimals) (contracts/Token.sol#471)
PEPE_KING_ETH.slitherConstructorVariables() (contracts/Token.sol#460-946) uses literals with too many digits:
        - distributorGas = 300000 (contracts/Token.sol#517)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

PEPE_KING_ETH.ETH (contracts/Token.sol#463) is never used in PEPE_KING_ETH (contracts/Token.sol#460-946)
PEPE_KING_ETH.liquidityTransferFee (contracts/Token.sol#493) is never used in PEPE_KING_ETH (contracts/Token.sol#460-946)
PEPE_KING_ETH.marketingTransferFee (contracts/Token.sol#494) is never used in PEPE_KING_ETH (contracts/Token.sol#460-946)
PEPE_KING_ETH.projectTransferFee (contracts/Token.sol#495) is never used in PEPE_KING_ETH (contracts/Token.sol#460-946)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DividendDistributor.WBNB (contracts/Token.sol#278) should be constant
DividendDistributor.dividendsPerShareAccuracyFactor (contracts/Token.sol#291) should be constant
PEPE_KING_ETH.DEAD (contracts/Token.sol#465) should be constant
PEPE_KING_ETH.ETH (contracts/Token.sol#463) should be constant
PEPE_KING_ETH.TransferFeeDenominator (contracts/Token.sol#497) should be constant
PEPE_KING_ETH.WBNB (contracts/Token.sol#464) should be constant
PEPE_KING_ETH.ZERO (contracts/Token.sol#466) should be constant
PEPE_KING_ETH._totalSupply (contracts/Token.sol#471) should be constant
PEPE_KING_ETH.buyFeeDenominator (contracts/Token.sol#483) should be constant
PEPE_KING_ETH.buyfeeburn (contracts/Token.sol#484) should be constant
PEPE_KING_ETH.liquidityBuyFee (contracts/Token.sol#479) should be constant
PEPE_KING_ETH.liquiditySellFee (contracts/Token.sol#486) should be constant
PEPE_KING_ETH.liquidityTransferFee (contracts/Token.sol#493) should be constant
PEPE_KING_ETH.marketingBuyFee (contracts/Token.sol#480) should be constant
PEPE_KING_ETH.marketingSellFee (contracts/Token.sol#487) should be constant
PEPE_KING_ETH.marketingTransferFee (contracts/Token.sol#494) should be constant
PEPE_KING_ETH.projectBuyFee (contracts/Token.sol#481) should be constant
PEPE_KING_ETH.projectSellFee (contracts/Token.sol#488) should be constant
PEPE_KING_ETH.projectTransferFee (contracts/Token.sol#495) should be constant
PEPE_KING_ETH.reflectionBuyFee (contracts/Token.sol#503) should be constant
PEPE_KING_ETH.reflectionSellFee (contracts/Token.sol#504) should be constant
PEPE_KING_ETH.sellFeeDenominator (contracts/Token.sol#490) should be constant
PEPE_KING_ETH.sellfeeburn (contracts/Token.sol#491) should be constant
PEPE_KING_ETH.totalBuyFee (contracts/Token.sol#482) should be constant
PEPE_KING_ETH.totalSellFee (contracts/Token.sol#489) should be constant
PEPE_KING_ETH.totalTransferFee (contracts/Token.sol#496) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DividendDistributor._token (contracts/Token.sol#269) should be immutable
DividendDistributor.router (contracts/Token.sol#279) should be immutable
PEPE_KING_ETH.distributor (contracts/Token.sol#516) should be immutable
PEPE_KING_ETH.launchedAt (contracts/Token.sol#513) should be immutable
PEPE_KING_ETH.launchedAtTimestamp (contracts/Token.sol#514) should be immutable
PEPE_KING_ETH.pair (contracts/Token.sol#510) should be immutable
PEPE_KING_ETH.router (contracts/Token.sol#509) should be immutable
PEPE_KING_ETH.ts_Project (contracts/Token.sol#511) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0xf421797626869aedae159b5f4d323d6853dc86e1051f45f137034d48f321c6d4

2- Buying when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x467d7f44fb9e9b3bd402e1567bbe202ff2d6ae198aff38a7ad8eeb7db2f5acfc

3- Selling when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x66c71b9407ce7e45ba2b6d53c536c8a8b0e3ff23080ce1b6d76ad288e4646cf7

4- Transferring when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0xd02e4b75699b95e6c7365b18a1cfaf9fd6808d2f5394bbdba8e6cdf0f3fca895

5- Buying when not excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x2458a523143af06e5ce4228c190168982822bdd0081cd084e9a15d68b21b61aa

6- Selling when not excluded from fees (8% tax) (passed):

https://testnet.bscscan.com/tx/0xa11fb10ae16076c856c98766e8d0660f3811413157c85efe3d460a98d33246da

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xb7f9b79d43f95ed95f9b1de90e9eb64086c76608a2c4b38a5889f9decb1421b7

7- Rewards, Marketing fee (passed):
https://testnet.bscscan.com/tx/0xde620afcd2c7e9f2cc4fc81c1226b09ade00c7123e57e37d6c31471dae43083a

MANUAL TESTING

Logical – 0 swap threshold may disable sell/transfers

Severity: Medium

function: swapBack

Status: Not Resolved

Overview:

if swapThreshold is set to 0, internal swap would be failed because contract tries to swap 0 tokens for BNB

```
function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

Suggestion

To mitigate this Logical issue, make sure that swapTokensAtAmount is always greater than 0

```
function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
    require(swapThreshold > totalSupply / 1000000);
}
```

MANUAL TESTING

Centralization – LP tokens going to an EOA

Severity: Medium

function: swapBack

Status: Not Resolved

Overview:

LP tokens generated from auto-liqudity are sent to an EOA account, this accumulated tokens may be used by a malicious owner to remove a portion of tokens from the pool

```
if (amountToLiquify > 0) {
    router.addLiquidityETH{value: amountBNBLiquidity}(
        address(this),
        amountToLiquify,
        0,
        0,
        autoLiquidityReceiver,
        block.timestamp
    );
    emit AutoLiquify(amountBNBLiquidity, amountToLiquify);
}
```

Suggestion

To mitigate this Centralization issue:

- Burn the new LP tokens

or

- Lock the new LP tokens

MANUAL TESTING

Informational – Use of external token

Status: Not Resolved

Overview:

Contract is using 0x2170Ed0880ac9A755fd29B2688956BD959F933F8 (ETH) as the reward token. This contract is beyond scope of this auidt and we assume that this token has enough liquidity and doesn't have major issues that could disable the trades and affect reward system of **PEPEK** token.

MANUAL TESTING

Informational – Redundant code

Status: Not Resolved

Overview:

Since fees are immutable, some parts of the contract are redundant and are only increasing gas usaage. Its suggested to remove this redundant parts from the contract:

- functions related to auto-liquidity
- functions and codes related to burn

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 guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace 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 Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always 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 Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace