auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Porn AI

DATED : 15 MAR 23'

AUDIT SUMMARY

Project name – Porn AI

Date: 15 March, 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	0	1	0
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 Testnet network:
all tests were done on Bsc Testnet network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0xcccFA10e1b27f8775e9A4f15222b0E31aC6da726

Token Information

Token Name : Porn AI

Token Symbol: PAI

Decimals: 18

Token Supply:
1,000,000,000,000,000,000,000,000,000

Token Address:
0xB9Eb734BDA3cfBFdF944d31022819194Bc96D97E

Checksum:
1f2fc18bc0c9d665cb1505a567a63130ac1fb09f

Owner:
0x502a37F708722f0D0496Ef46f3BaCeFd1168D69c
(AT THE TIME OF WRITING AUDIT)

TOKEN OVERVIEW

Fees:

Buy Fees: 10% currently

Sell Fees: 10% currently

Transfer Fees: 10% currently

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding form fee - changing distirbution settings (min tokens to be eligible, cooldown between claims etc)

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

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

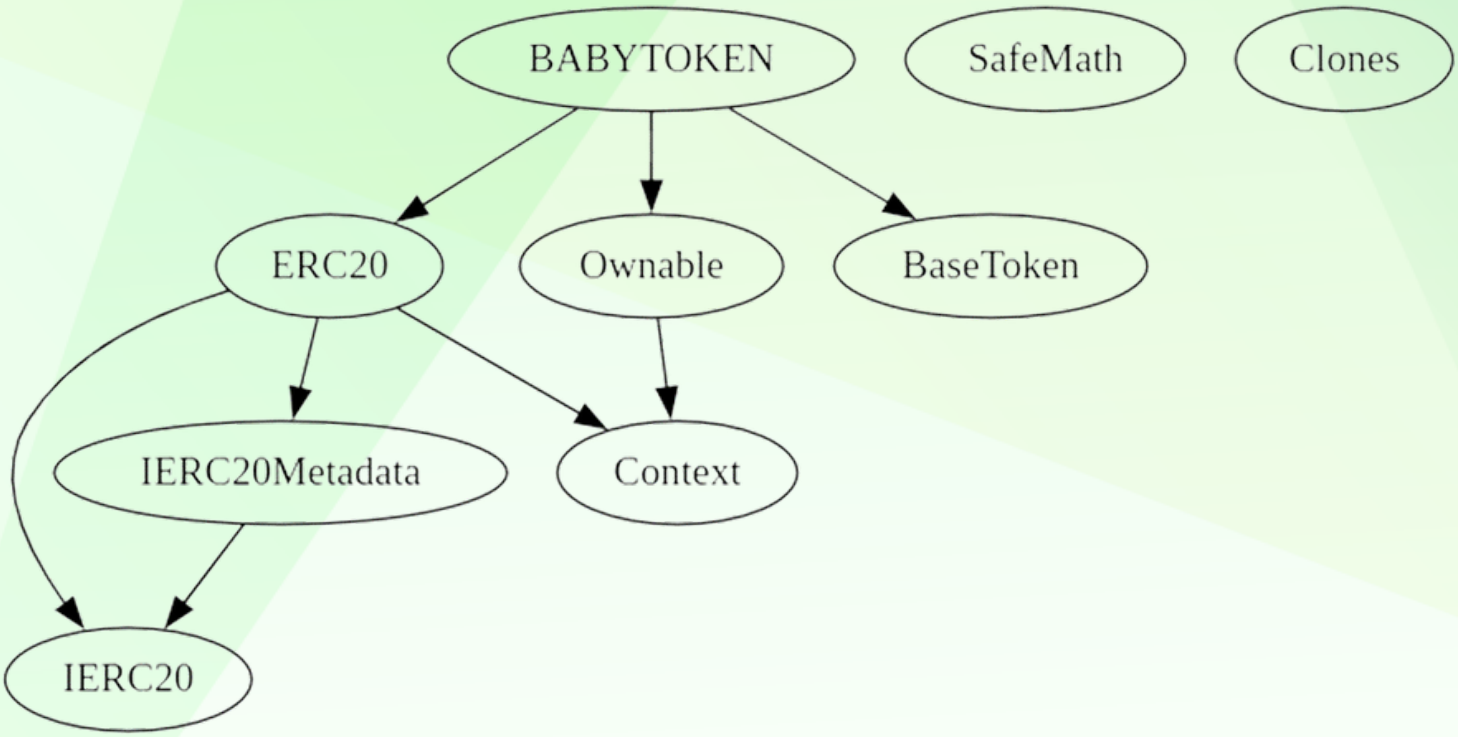
CLASSIFICATION OF RISK

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	0
Low-Risk	1
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is able to change buy/sell/transfer fees but sum of fees can not exceed 25%
- Contract is recognized as proxy in bscscan, however this is because dividend distributor is minimal proxy contract which is not upgradeable currently
- 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

OVERVIEW

This token has three different types of fees, which are used for marketing, maintaining liquidity, and providing rewards to holders. The fees collected are converted to a stablecoin called BUSD and sent to a marketing wallet. Another part of the fee is used to maintain liquidity and is sent to a dividend tracker to provide BUSD rewards to holders. Rewards are distributed to holders every time a buy, sell or transfer occurs, and can also be manually claimed from the BscScan website.

TOKEN DISTRIBUTION

It should be noted that the owner currently holds 100% of the total supply. However, information about the distribution of these tokens is not available, and it is recommended that investors exercise caution when considering this aspect.

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	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❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
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❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
Ownable	Implementation	Context		

CONTRACT ASSESMENT

└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
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 🔒		
Clones	Library			
└	clone	Internal 🔒	🛑	
└	cloneDeterministic	Internal 🔒	🛑	
└	predictDeterministicAddress	Internal 🔒		
└	predictDeterministicAddress	Internal 🔒		
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResult	Internal 🔒		
IUniswapV2Factory	Interface			
└	feeTo	External ❗		NO❗

CONTRACT ASSESMENT

└	feeToSetter	External ❗			NO❗
└	getPair	External ❗			NO❗
└	allPairs	External ❗			NO❗
└	allPairsLength	External ❗			NO❗
└	createPair	External ❗	🛑		NO❗
└	setFeeTo	External ❗	🛑		NO❗
└	setFeeToSetter	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❗
IERC20Upgradeable	Interface				
└	totalSupply	External ❗			NO❗
└	balanceOf	External ❗			NO❗
└	transfer	External ❗	🛑		NO❗
└	allowance	External ❗			NO❗

CONTRACT ASSESMENT

└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20MetadataUpgradeable	Interface	IERC20Upgradeable		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
Initializable	Implementation			
ContextUpgradeable	Implementation	Initializable		
└	__Context_init	Internal 🔒	🛑	initializer
└	__Context_init_unchained	Internal 🔒	🛑	initializer
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
ERC20Upgradeable	Implementation	Initializable, ContextUpgradeable, IERC20Upgradeable, IERC20MetadataUpgradeable		
└	__ERC20_init	Internal 🔒	🛑	initializer
└	__ERC20_init_unchained	Internal 🔒	🛑	initializer
└	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❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
OwnableUpgradeable	Implementation	Initializable, ContextUpgradeable		
└	__Ownable_init	Internal 🔒	🛑	initializer
└	__Ownable_init_unchained	Internal 🔒	🛑	initializer
└	owner	Public ❗		NO❗

CONTRACT ASSESMENT

└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
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❗
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	swap	External ❗	🛑	NO❗
└	skim	External ❗	🛑	NO❗
└	sync	External ❗	🛑	NO❗
└	initialize	External ❗	🛑	NO❗
SafeMathInt	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		

CONTRACT ASSESMENT

SafeMathUint	Library			
└	toInt256Safe	Internal 🔒		
IterableMapping	Library			
└	get	Internal 🔒		
└	getIndexOfKey	Internal 🔒		
└	getKeyAtIndex	Internal 🔒		
└	size	Internal 🔒		
└	set	Internal 🔒	🛑	
└	remove	Internal 🔒	🛑	
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗		NO❗
└	withdrawDividend	External ❗	🛑	NO❗
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
DividendPayingToken	Implementation	ERC20Upgradeable, OwnableUpgradeable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
└	__DividendPayingToken_init	Internal 🔒	🛑	initializer
└	distributeCAKEDividends	Public ❗	🛑	onlyOwner
└	withdrawDividend	Public ❗	🛑	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	
BABYTOKENDividendTracker	Implementation	OwnableUpgradeable, DividendPayingToken		
└	initialize	External ❗	🛑	initializer
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	isExcludedFromDividends	Public ❗		NO❗
└	updateClaimWait	External ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	updateMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO❗
└	processAccount	Public ❗	🛑	onlyOwner
BaseToken	Implementation		
BABYTOKEN	Implementation	ERC20, Ownable, BaseToken	
└	<Constructor>	Public ❗	💵	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	External ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setTokenRewardsFee	External ❗	🛑	onlyOwner
└	setLiquiditFee	External ❗	🛑	onlyOwner
└	setMarketingFee	External ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	updateMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner
└	getMinimumTokenBalanceForDividends	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	isExcludedFromFees	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	isExcludedFromDividends	Public ❗		NO❗
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfDividendTokenHolders	External ❗		NO❗
└	_transfer	Internal 🔒	🛑	

CONTRACT ASSESMENT

└	swapAndSendToFee	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	swapTokensForCake	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapAndSendDividends	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Function OwnableUpgradeable.__Ownable_init() (contracts/Token.sol#2087-2090) is not in mixedCase
Function OwnableUpgradeable.__Ownable_init_unchained() (contracts/Token.sol#2092-2094) is not in mixedCase
Variable OwnableUpgradeable.__gap (contracts/Token.sol#2140) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#2176) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#2178) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#2209) is not in mixedCase
Function DividendPayingToken.__DividendPayingToken_init(address,string,string) (contracts/Token.sol#2507-2515) is not in mixedCase
Parameter DividendPayingToken.__DividendPayingToken_init(address,string,string)._rewardToken (contracts/Token.sol#2508) is not in mixedCase
Parameter DividendPayingToken.__DividendPayingToken_init(address,string,string)._name (contracts/Token.sol#2509) is not in mixedCase
Parameter DividendPayingToken.__DividendPayingToken_init(address,string,string)._symbol (contracts/Token.sol#2510) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/Token.sol#2568) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/Token.sol#2576) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/Token.sol#2585) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/Token.sol#2596) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#2487) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter BABYTOKENDividendTracker.getAccount(address)._account (contracts/Token.sol#2757) is not in mixedCase
Variable BABYTOKEN._marketingWalletAddress (contracts/Token.sol#2982) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#1261) is too similar to IUniswapV2Router
01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#1262)
Variable DividendPayingToken.__DividendPayingToken_init(address,string,string)._rewardToken (contracts/Token.sol#2508) is too similar to BABYTOKENDividendTracker.initialize(address,u
int256).rewardToken_ (contracts/Token.sol#2690)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/Token.sol#2541) is too similar to BABYTOKENDividendTracker.getAccount(address).withdraw
ableDividends (contracts/Token.sol#2765)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Clones.clone(address) (contracts/Token.sol#880-895) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d730000000000000000000000000000) (contracts/Token.sol#883-886)
Clones.clone(address) (contracts/Token.sol#880-895) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (contracts/Token.sol#888-891)
Clones.cloneDeterministic(address,bytes32) (contracts/Token.sol#904-922) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d730000000000000000000000000000) (contracts/Token.sol#910-913)
Clones.cloneDeterministic(address,bytes32) (contracts/Token.sol#904-922) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (contracts/Token.sol#915-918)
Clones.predictDeterministicAddress(address,bytes32,address) (contracts/Token.sol#927-948) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d730000000000000000000000000000) (contracts/Token.sol#934-937)
Clones.predictDeterministicAddress(address,bytes32,address) (contracts/Token.sol#927-948) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3ff0000000000000000000000000000000000000000) (contracts/Token.sol#939-942)
BABYTOKEN.constructor(string,string,uint256,address[4],uint256[3],uint256) (contracts/Token.sol#3020-3083) uses literals with too many digits:
        - gasForProcessing = 300000 (contracts/Token.sol#3047)
BABYTOKEN.updateGasForProcessing(uint256) (contracts/Token.sol#3156-3167) uses literals with too many digits:
        - require(bool,string)(newValue >= 200000 && newValue <= 500000,BABYTOKEN: gasForProcessing must be between 200,000 and 500,000) (contracts/Token.sol#3157-3160)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#2256) is never used in SafeMathInt (contracts/Token.sol#2254-2311)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

BABYTOKEN.dividendTracker (contracts/Token.sol#2971) should be immutable
BABYTOKEN.rewardToken (contracts/Token.sol#2973) should be immutable
BABYTOKEN.uniswapV2Pair (contracts/Token.sol#2967) should be immutable
BABYTOKEN.uniswapV2Router (contracts/Token.sol#2966) 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 issues found

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):
liquidity added on Pancakeswap V2:

https://testnet.bscscan.com/tx/0x87355110d7eaabe8bc822f3082430e4163ba8d5dba6a15836237ea158e904400

2- Buying when excluded (0%)(Passed):

https://testnet.bscscan.com/tx/0xaa8e8c19e4fd549db0c3c2cdb40f092758f7a3d31f2f96f70d7ce45274ae95c2

3- Selling when excluded (0%)(Passed):

htttps://testnet.bscscan.com/tx/0x0b5a0be91790fea950702b86ec947e9c7355fd6c5f833635157918df045bb462

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

https://testnet.bscscan.com/tx/0xe6e0225e52d2cbba36a1ab97f8311f66322ae2834a6656ace7c6fd3f693dc059

5- Buying when not excluded (10% tax) (passed):

https://testnet.bscscan.com/tx/0x2cf9f3ba375835e0d6aed31d03b86d3da503d20c6fdb674e54383e3408f164d7

FUNCTIONAL TESTING

6- Selling when not excluded (10% tax) (passed):

https://testnet.bscscan.com/tx/0x1b560dd4cbc15ae362abab9be082df436f8df0476faf5acc291ebd46b0f9e9cd

7- Transferring when not excluded(10% tax) (passed):

https://testnet.bscscan.com/tx/0x0d0fdd2349e0c7b0330480361f7c69fab9d7d247c8a919facb3d11840d78619b

8- Internal swap (passed):

marketing wallet received BUSD
https://testnet.bscscan.com/address/0xa2da001d772453f7a1d520148663462ebcbd79b4#tokentxns

9- Reflections (passed):
as seen in this transaction:
https://testnet.bscscan.com/tx/0x1b560dd4cbc15ae362abab9be082df436f8df0476faf5acc291ebd46b0f9e9cd
distributor is sending BUSD to holders (auto distribution)

10- Auto Liquidity (passed):
https://testnet.bscscan.com/token/0xd72699c11fb075dc8ddf9652fcbd5ae19dc7e9bf?a=0x000000000000000000000000000000000000dead

MANUAL TESTING

Issue: some ERC20 tokens may not return a boolean after transfer success

Type : Logical

Function: **_withdrawDividendOfUser**

Line: 2370 - 2380

Severity: Low

Overview:

Some ERC20 contracts may not support returning a boolean (true) if the transfer was successful

```
function _withdrawDividendOfUser(
    address payable user
) internal returns (uint256) {
    uint256 _withdrawableDividend = withdrawableDividendOf(user);
    if ( _withdrawableDividend > 0) {
        withdrawnDividends[user] = withdrawnDividends[user].add(
            _withdrawableDividend
        );
        emit DividendWithdrawn(user, _withdrawableDividend);
        bool success = IERC20(rewardToken).transfer( //@audit use safeTransfer to transfer tokens
            user,
            _withdrawableDividend
        );

        if (!success) {
            withdrawnDividends[user] = withdrawnDividends[user].sub(
                _withdrawableDividend
            );
            return 0;
        }
```

Recommendation:

• use SafeERC20 to handle token transfers

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