RugFreeCoins Audit

Baby Elephant Token

Smart Contract Security Audit

February 18, 2022

Contents

Audit details

Audited project
Baby Elephant Token

Contract Address

0xe79bdf3b0650Ef21a5BAc9625f7ea2732918E66c

Client contact

Baby Elephant Token Team

Blockchain

Binance smart chain

Project website

www.babyelephant.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 Baby Elephant Token Team to perform an audit of the smart contract.

https://bscscan.com/token/0xe79bdf3b0650Ef21a5BAc9625f7ea2732918E66c

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 improving the security posture of the smart contract by remediating the issues that were identified.

About the project

Baby Elephant is a token built on the Binance Smart Chain that is with an innovative investment use case the main purpose of which is to seek out constant revenue sources of betting, trading bots, swapping, P2E games, NFTs and staking. Each transaction, purchase incurs 0% fee, and sale incurs a 11% fee.

Features

The **sustainability fee of 5% when selling for project funds and 3% for utilities wallet** is what allows Baby Elephant to hold the aforementioned promise. Tokens will be swapped into BUSD and will be sent to project fund and utility wallets. This way, Baby Elephant will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.

Baby Elephant has the burn strategy that a **1% fee in each transaction when selling** is getting charged that benefits and rewards those who invest long-term. This feature slowly reduces supply making each Baby Elephant more and more valuable.

The **Baby Elephant Token rewards** will be distributed among every holder proportional to how many tokens each individual holds in values of **2% when selling.**

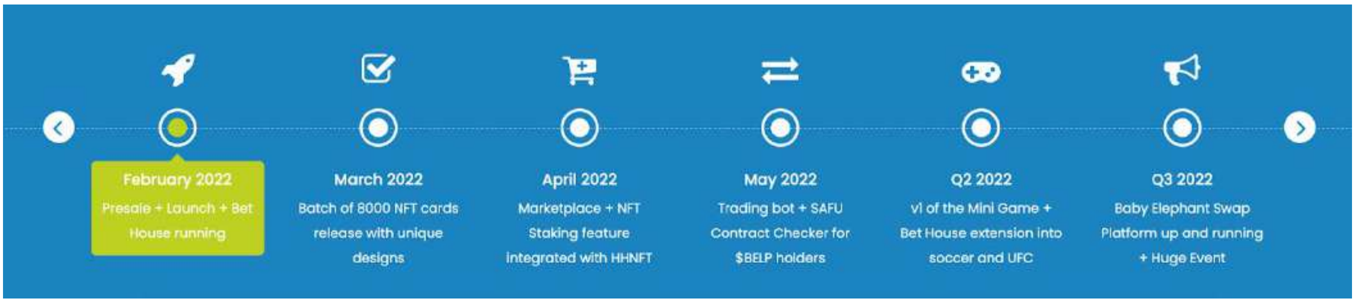
Tokenomics

0% fee when buying

11% fee when selling

- 2% of trade goes to holders pockets in tokens.
- 1% of trade goes to the burn wallet.
- 5% of trade goes to the project wallet in BUSD.
- 3% of trade goes to the utilities wallet in BUSD.

Roadmap



February 2022
Presale + Launch + Bet House running

March 2022
Batch of 8000 NFT cards release with unique designs

April 2022
Marketplace + NFT Staking feature integrated with HHNFT

May 2022
Trading bot + SAFU Contract Checker for $BELP holders

Q2 2022
v1 of the Mini Game + Bet House extension into soccer and UFC

Q3 2022
Baby Elephant Swap Platform up and running + Huge Event

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 tokens by holding tokens.
- Anyone who’s interested in trading tokens.
- Anyone who’s interested in collecting NFTs or trading NFTs.
- Anyone who’s interested in taking part with the Bet house feature of the project.
- Anyone who’s interested in using the trading bot for trading.
- Anyone who’s interested in taking part with the future plans of the Baby Elephant token.
- Anyone who’s interested in taking part with Baby Elephant play to earn game and win rewards.
- Anyone who’s interested in making financial transactions with any other party using Baby Elephant as the currency.

Core concept

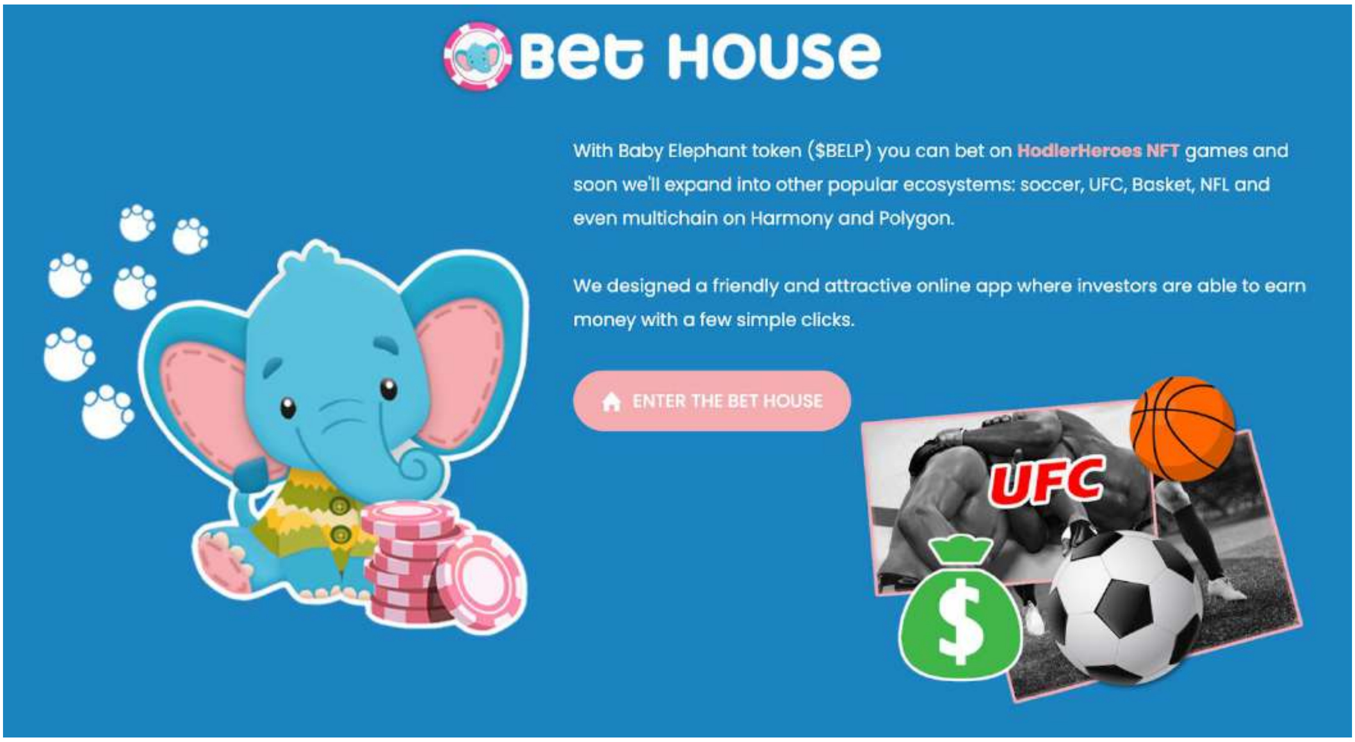
The Baby Elephant reward system

2% of each transaction when selling is split amongst all holders in Baby Elephant tokens. Holders will be eligible to receive tokens in every tx and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The **sustainability fee of 5% when selling for project fund and 3% fee for utility wallet** is what allows Baby Elephant to promote the token and use funds to further the development of the platform. Tokens will be swapped into BUSD and will be sent to wallets.. This way, Baby Elephant will have access to the funds without selling tokens as the traditional way, which will enable them to consume funds without hurting the project.

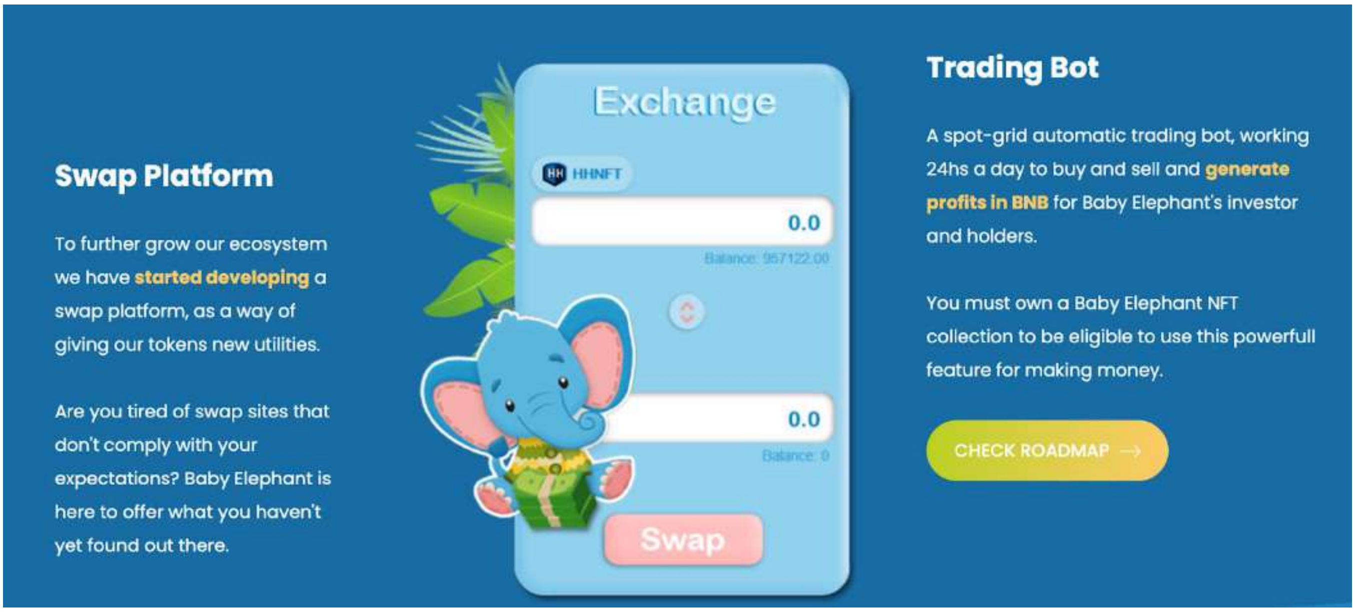
Baby Elephant has the burn strategy that **1% fee in each transaction when selling** is getting charged that benefits and rewards those who invest long-term. This feature slowly reduces supply making each Baby Elephant more and more valuable.



Bet House

With Baby Elephant token ($BELP) you can bet on **HodlerHeroes NFT** games and soon we'll expand into other popular ecosystems: soccer, UFC, Basket, NFL and even multichain on Harmony and Polygon.

We designed a friendly and attractive online app where investors are able to earn money with a few simple clicks.



Swap Platform

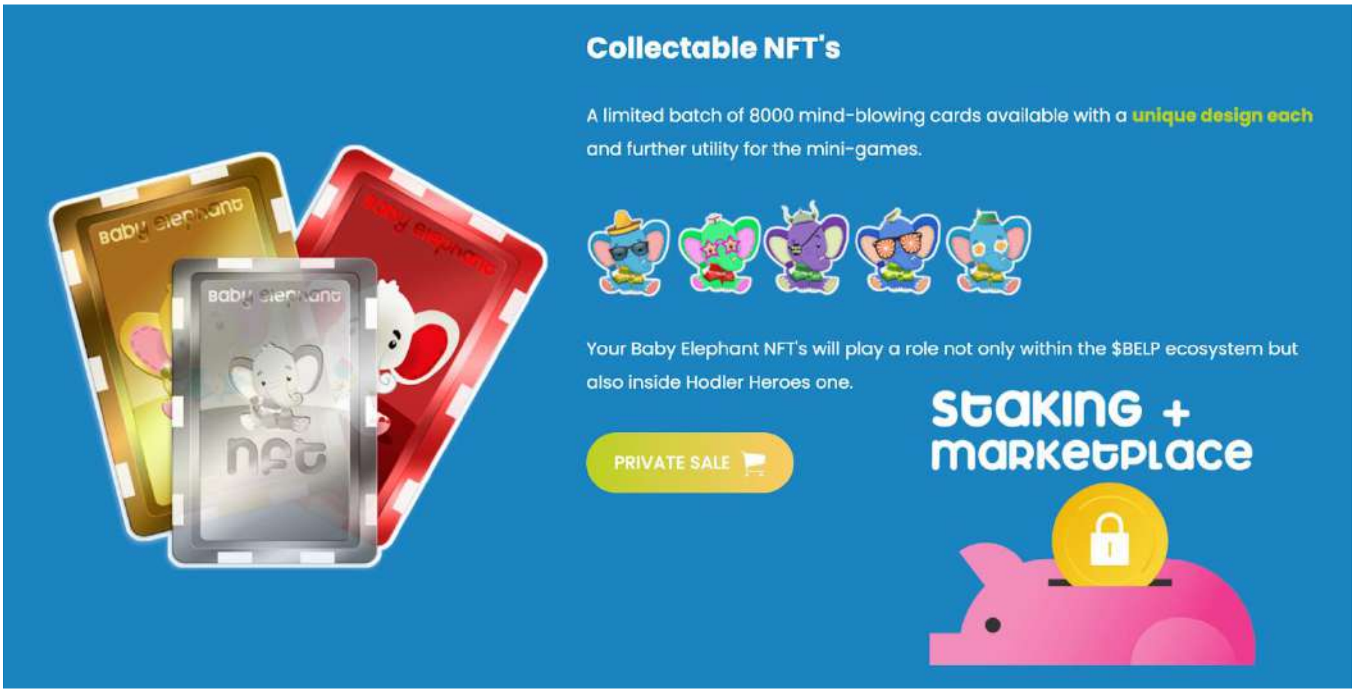
To further grow our ecosystem we have **started developing** a swap platform, as a way of giving our tokens new utilities.

Are you tired of swap sites that don't comply with your expectations? Baby Elephant is here to offer what you haven't yet found out there.

Trading Bot

A spot-grid automatic trading bot, working 24hs a day to buy and sell and **generate profits in BNB** for Baby Elephant's investor and holders.

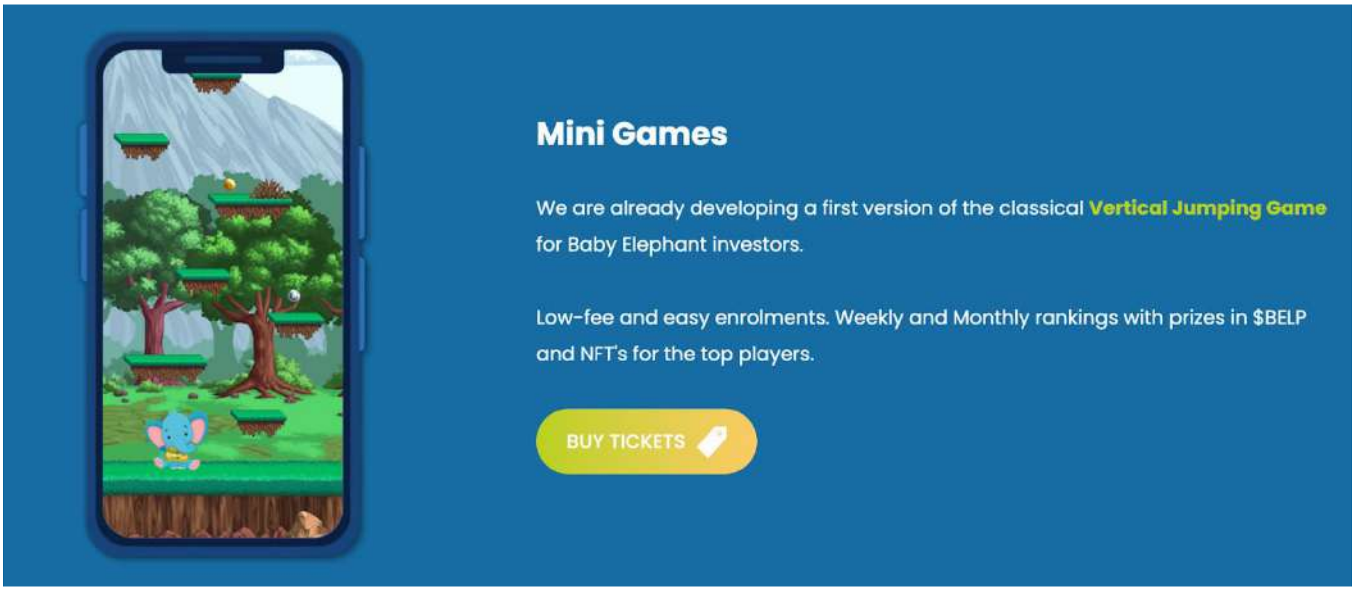
You must own a Baby Elephant NFT collection to be eligible to use this powerfull feature for making money.



Collectable NFT's

A limited batch of 8000 mind-blowing cards available with a **unique design each** and further utility for the mini-games.

Your Baby Elephant NFT's will play a role not only within the $BELP ecosystem but also inside Hodler Heroes one.



Mini Games

We are already developing a first version of the classical **Vertical Jumping Game** for Baby Elephant investors.

Low-fee and easy enrolments. Weekly and Monthly rankings with prizes in $BELP and NFT's for the top players.

Potential to grow with score points

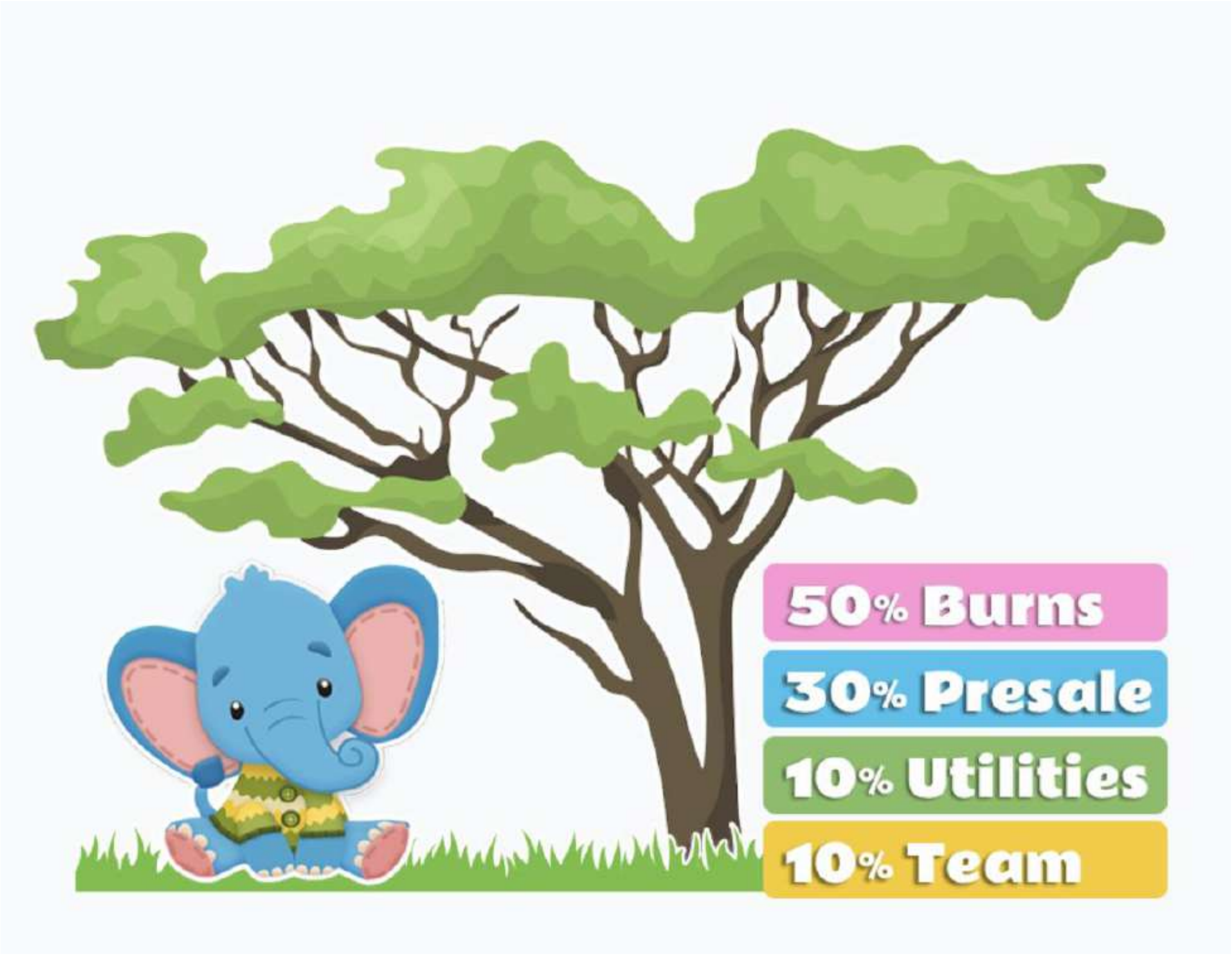
1.	Project efficiency	10/10
2.	Project uniqueness	10/10
3	Information quality	10/10
4	Service quality	10/10
5	System quality	10/10
6	Impact on the community	10/10
7	Impact on the business	10/10
8	Preparing for the future	10/10
Total Points		**10/10**

Contract details

Token contract details for 18th February 2022

Contract name	Baby Elephant
Contract address	0xe79bdf3b0650Ef21a5BAc9625f7ea2732918E66c
Token supply	14,000,000
Token ticker	BELP
Decimals	9
Token holders	2
Transaction count	18
Marketing address	0xb6bddf12fa5bde762a4dca43ef9e9794cbfb6f86
Team wallet address	0xcd0c4b921b4a17896511e4720d6ce8f63ee8813d
Contract deployer address	0x97361F6979756A647458DC3EAAB802Db416997a3
Contract’s current owner address	0xc284a5b5a6b53e2d93d74ed1abe874d3cdb5513a

3. Token distribution for
Tokens are distributed as follows:



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
Babyelephant	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❗

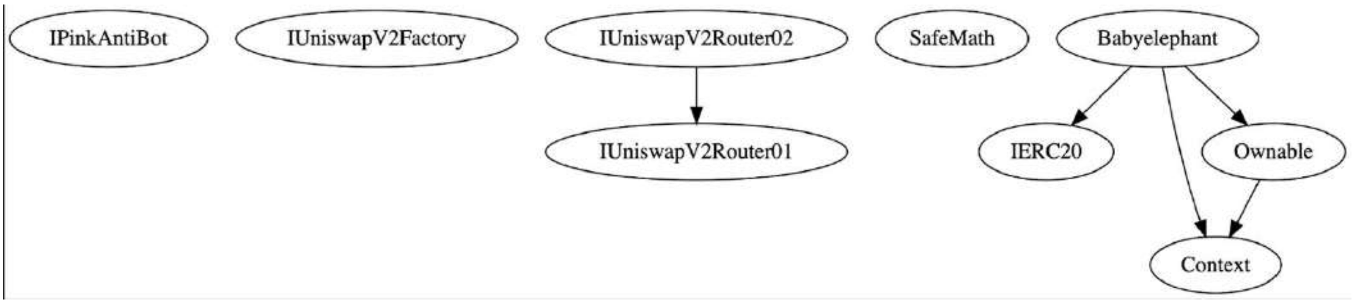
└	deliver	Public ❗	🛑	NO❗
└	reflectionFromToken	Public ❗		NO❗
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	Public ❗	🛑	onlyOwner
└	includeInReward	External ❗	🛑	onlyOwner
└	setUtilitiesWallet	External ❗	🛑	onlyOwner
└	setProjectWallet	External ❗	🛑	onlyOwner
└	setEnableAntiBot	External ❗	🛑	onlyOwner
└	changeSwapAmount	External ❗	🛑	onlyOwner
└	setExcludedFromFee	External ❗	🛑	onlyOwner
└	tradingEnable	External ❗	🛑	onlyOwner
└	updateBuyFees	External ❗	🛑	onlyOwner
└	updateSellFees	External ❗	🛑	onlyOwner
└	updateSwapPercentage	External ❗	🛑	onlyOwner
└	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
└		External ❗	💵	NO❗
└	setUniswapRouter	External ❗	🛑	onlyOwner
└	setUniswapPair	External ❗	🛑	onlyOwner
└	setAuthorizedWallets	External ❗	🛑	onlyOwner
└	setExcludedFromAutoLiquidity	External ❗	🛑	onlyOwner

└	_reflectFee	Private 🔐	🛑	
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	takeTokenFees	Private 🔐	🛑	
└	takeTransactionFee	Private 🔐	🛑	
└	calculateFee	Private 🔐		
└	isExcludedFromFee	Public ❗		NO❗
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapAndSendBusd	Private 🔐	🛑	lockTheSwap
└	swapTokensForTokens	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	
└	_transferBothExcluded	Private 🔐	🛑	
└	_transferToExcluded	Private 🔐	🛑	
└	_transferFromExcluded	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	**Function can modify state**
💵	**Function is payable**

Inheritance Hierarchy



Security issue checking status

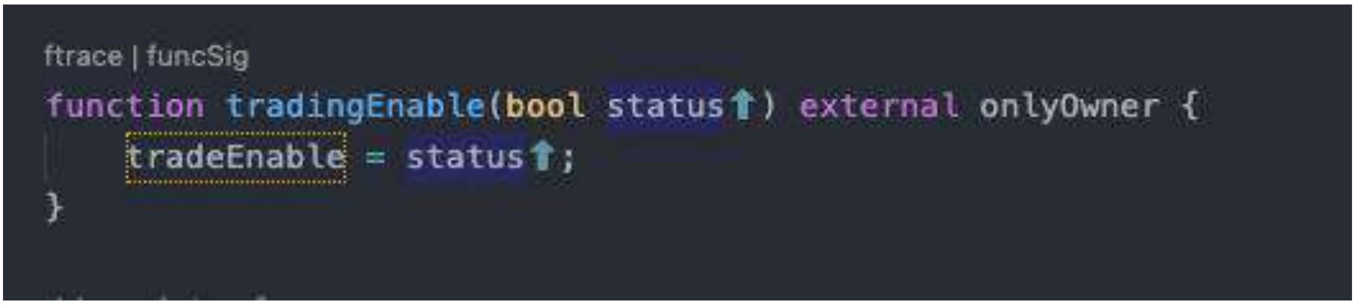
- ❖ **High severity issues**
  No high severity issues found.

- ❖ **Medium severity issues**
  No medium severity issues found.

- ❖ **Low severity issues**
  No low severity issues found.

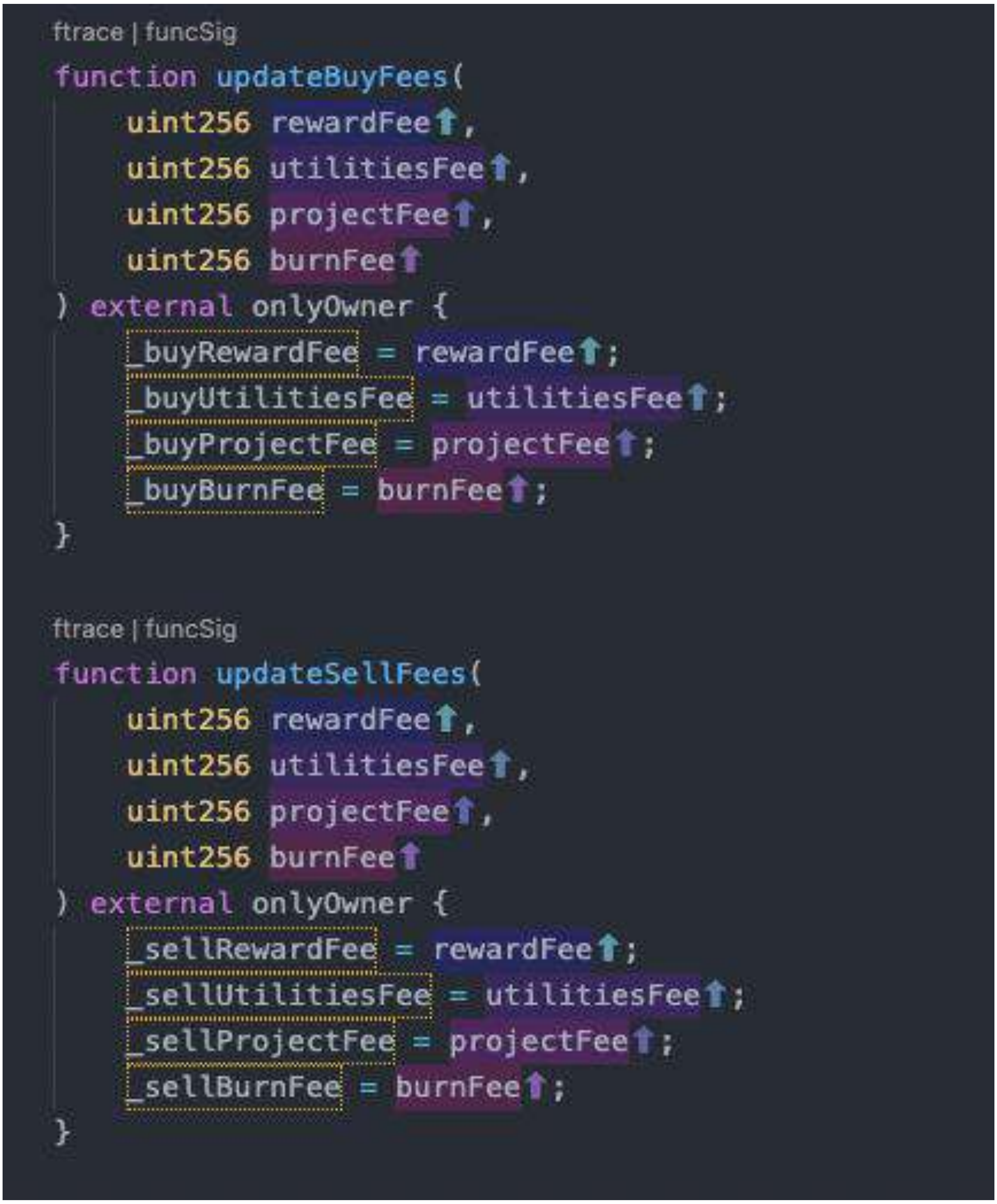
- ❖ **Informational**

- ❖ Owner can enable/disable trading anytime.



```
ftrace | funcSig
function tradingEnable(bool status) external onlyOwner {
    tradeEnable = status;
}
```

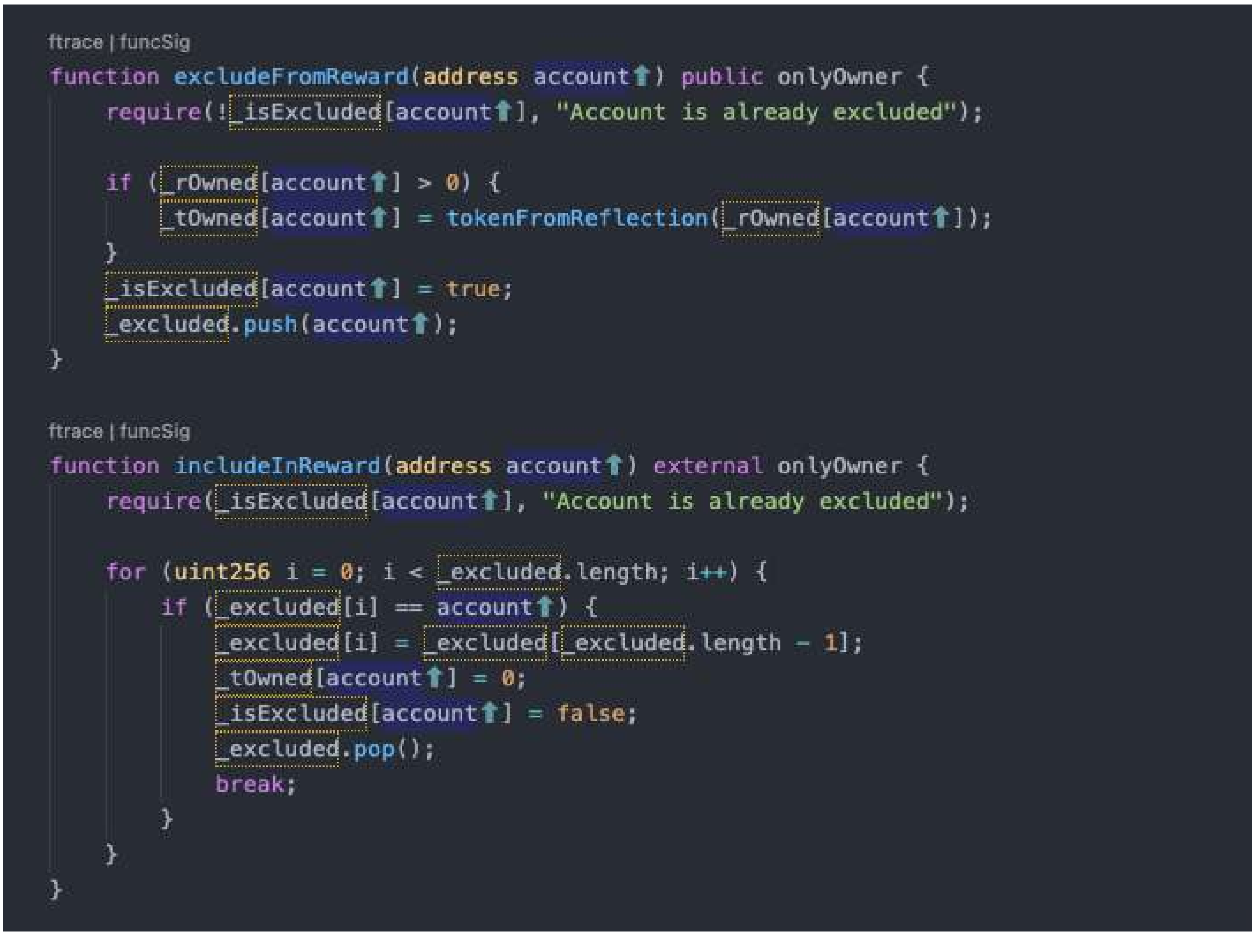
❖ Owner can change max transaction amount without any minimum limit.



```
ftrace | funcSig
function updateBuyFees(
    uint256 rewardFee,
    uint256 utilitiesFee,
    uint256 projectFee,
    uint256 burnFee
) external onlyOwner {
    _buyRewardFee = rewardFee;
    _buyUtilitiesFee = utilitiesFee;
    _buyProjectFee = projectFee;
    _buyBurnFee = burnFee;
}

ftrace | funcSig
function updateSellFees(
    uint256 rewardFee,
    uint256 utilitiesFee,
    uint256 projectFee,
    uint256 burnFee
) external onlyOwner {
    _sellRewardFee = rewardFee;
    _sellUtilitiesFee = utilitiesFee;
    _sellProjectFee = projectFee;
    _sellBurnFee = burnFee;
}
```

Owner privileges

- The owner can include/exclude wallets from rewards



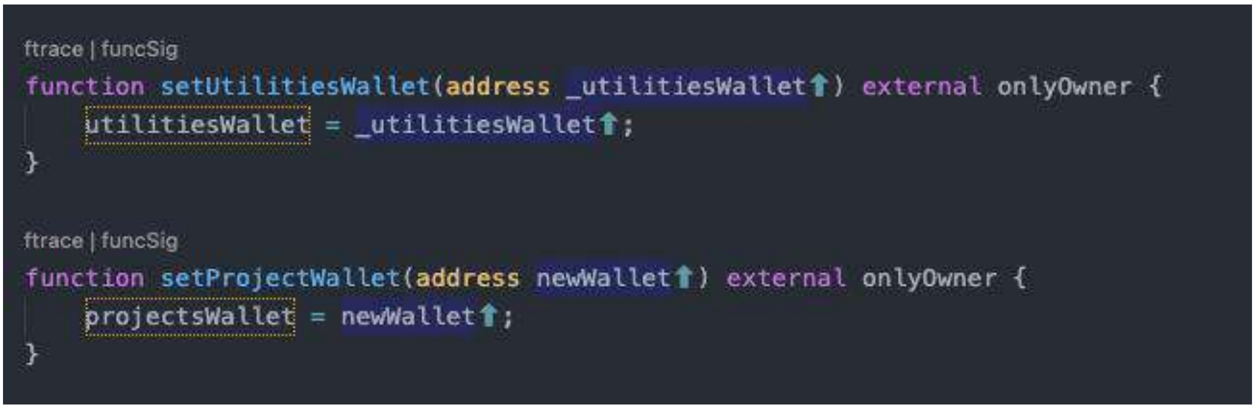
ftrace | funcSig

```
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 already 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 change utilities and project wallet



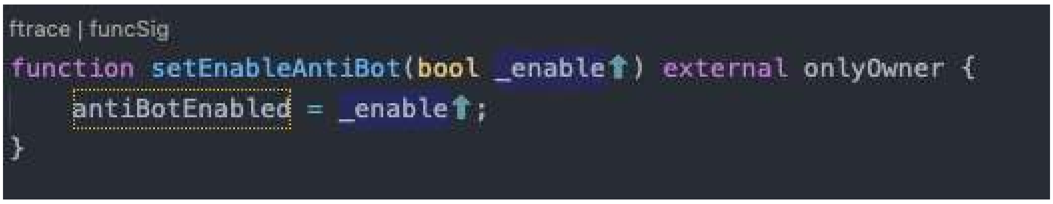
ftrace | funcSig

```
function setUtilitiesWallet(address _utilitiesWallet) external onlyOwner {
    utilitiesWallet = _utilitiesWallet;
}
```

ftrace | funcSig

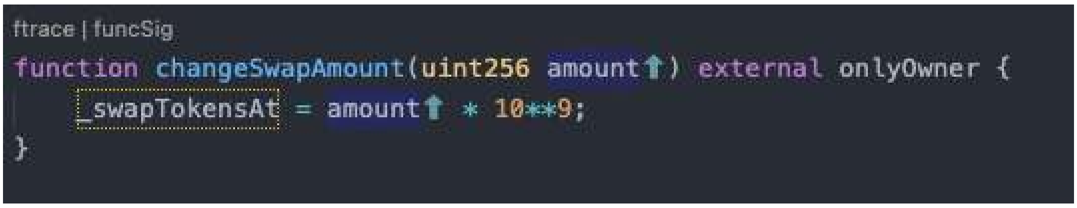
```
function setProjectWallet(address newWallet) external onlyOwner {
    projectsWallet = newWallet;
}
```

- The owner can enable/disable pink antibot.



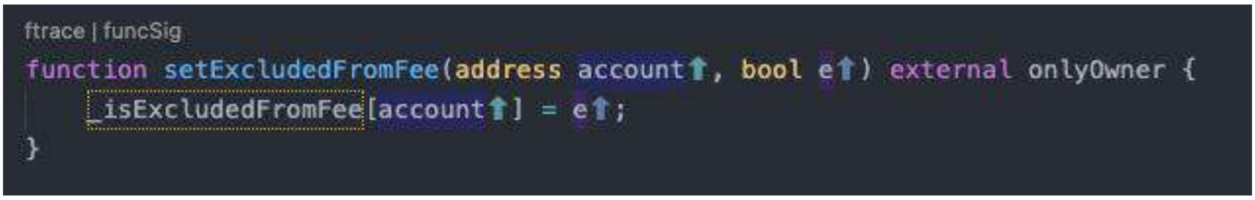
```
ftrace | funcSig
function setEnableAntiBot(bool _enable) external onlyOwner {
    antiBotEnabled = _enable;
}
```

- The owner can change the swap point.



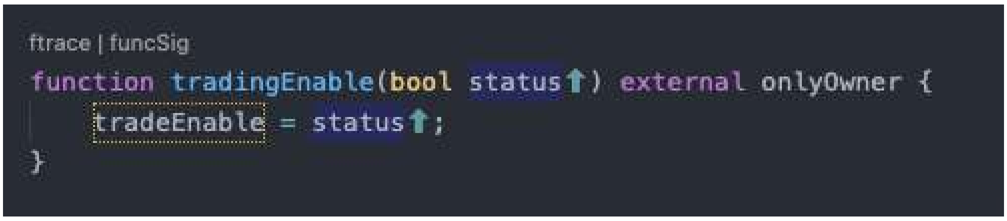
```
ftrace | funcSig
function changeSwapAmount(uint256 amount) external onlyOwner {
    _swapTokensAt = amount * 10**9;
}
```

- The owner can exclude wallets from fee.



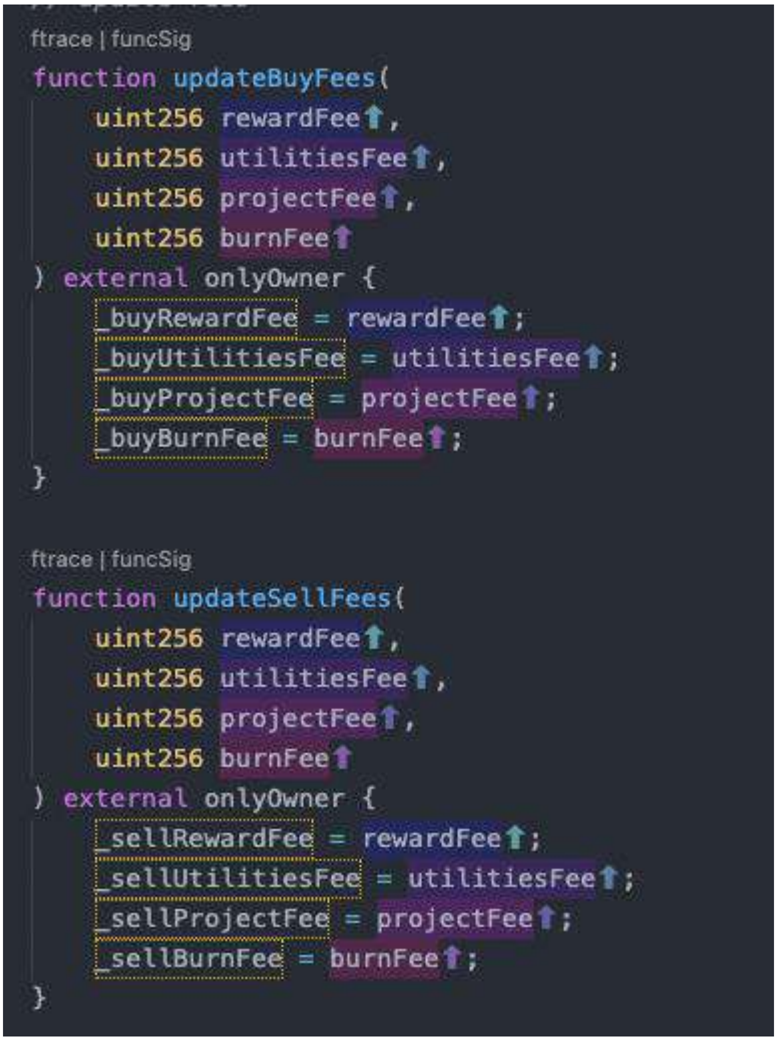
```
ftrace | funcSig
function setExcludedFromFee(address account, bool e) external onlyOwner {
    _isExcludedFromFee[account] = e;
}
```

- The owner can enable/disable trading.



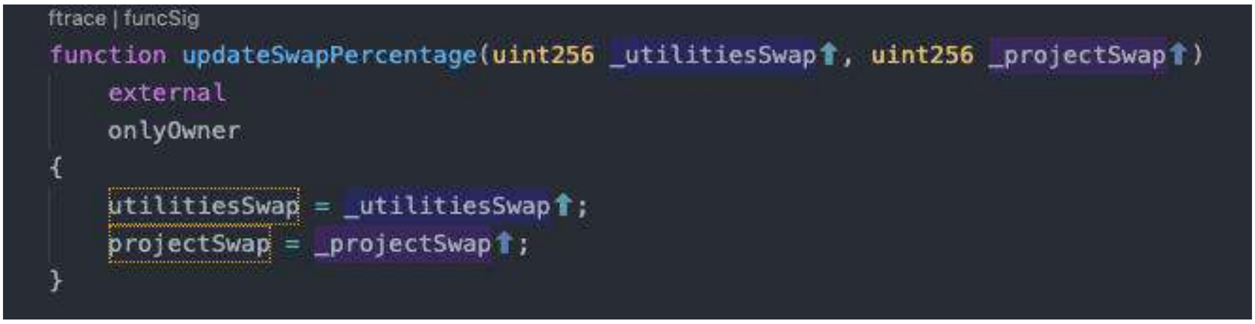
```
ftrace | funcSig
function tradingEnable(bool status) external onlyOwner {
    tradeEnable = status;
}
```

- ❖ The owner can update all buy and sell fees



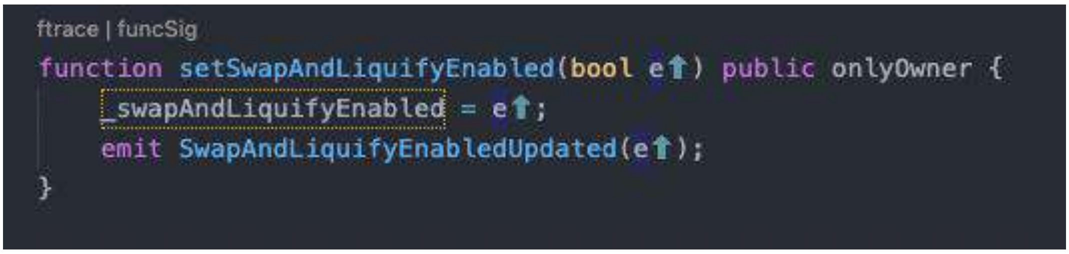
```
ftrace | funcSig
function updateBuyFees(
    uint256 rewardFee,
    uint256 utilitiesFee,
    uint256 projectFee,
    uint256 burnFee
) external onlyOwner {
    _buyRewardFee = rewardFee;
    _buyUtilitiesFee = utilitiesFee;
    _buyProjectFee = projectFee;
    _buyBurnFee = burnFee;
}

ftrace | funcSig
function updateSellFees(
    uint256 rewardFee,
    uint256 utilitiesFee,
    uint256 projectFee,
    uint256 burnFee
) external onlyOwner {
    _sellRewardFee = rewardFee;
    _sellUtilitiesFee = utilitiesFee;
    _sellProjectFee = projectFee;
    _sellBurnFee = burnFee;
}
```

- ❖ The owner can update swap percentages



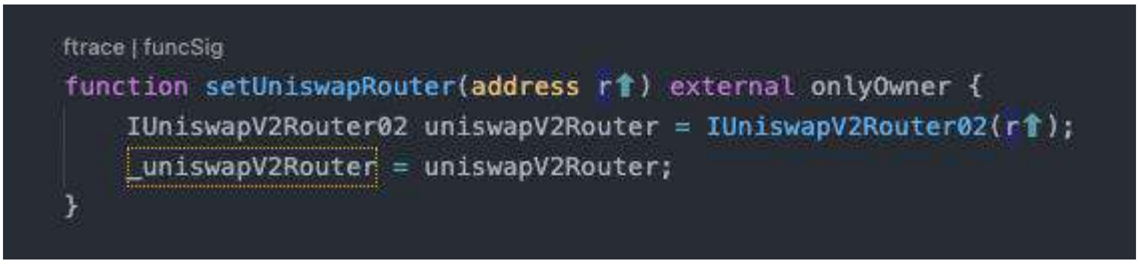
```
ftrace | funcSig
function updateSwapPercentage(uint256 _utilitiesSwap, uint256 _projectSwap)
    external
    onlyOwner
{
    utilitiesSwap = _utilitiesSwap;
    projectSwap = _projectSwap;
}
```

- The owner can enable/disable swapping



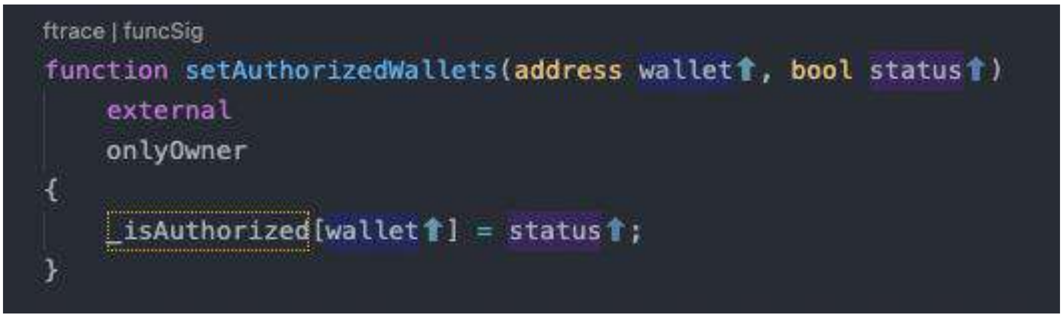
```
ftrace | funcSig
function setSwapAndLiquifyEnabled(bool e) public onlyOwner {
    _swapAndLiquifyEnabled = e;
    emit SwapAndLiquifyEnabledUpdated(e);
}
```

- The owner can update uniswap router address



```
ftrace | funcSig
function setUniswapRouter(address r) external onlyOwner {
    IUniswapV2Router02 uniswapV2Router = IUniswapV2Router02(r);
    _uniswapV2Router = uniswapV2Router;
}
```

- The owner can add/remove authorized wallets



```
ftrace | funcSig
function setAuthorizedWallets(address wallet, bool status)
    external
    onlyOwner
{
    _isAuthorized[wallet] = status;
}
```

Audit conclusion

While conducting the audit of the Baby Elephant smart contract, it was observed that there is nothing alarming with the code.