



## **RugFreeCoins Audit**



## **UpCake Audit**

# **Smart Contract Security Audit**

**October 12, 2021**



# Contents

Audit details	1
Disclaimer	2
Background	3
About the project	4
Target market and the concept	7
Potential to grow with score points	12
Total Points	12
Contract details	13
Token distribution	14
Contract code function details	15
Contract description table	16
Security issue checking status	22
Owner privileges	24
Audit conclusion	29

# Audit details



## **Audited project**

Upcake Token



## **Contract Address**

0xae916ea422623734e3a87718524859f05ca9de6e



## **Client contact**

Upcake Team



## **Blockchain**

Binance smart chain



## **Project website**

<https://www.upcake.net/>

# 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 Upcake to perform an audit of the smart contract.

**<https://bscscan.com/token/0xae916ea422623734e3a87718524859f05ca9de6e>**

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

Upcake is a token built on the Binance Smart Chain that combines the two most successful tokenomics on the Smart Chain: Rebase and Cake Rewards. Every hour, Cake rewards are distributed automatically to all holders, the supply diminishes, and the price per token increases. Each transaction, purchase and sale incur 15% fee.

## Features

- ❖ **Rebase mechanism:** With an elastic supply, the UpCake token guarantees a forever growing chart, as the price per token is constantly increasing. Every hour, the supply is reduced and converges towards a reasonable, stable value.
- ❖ The **Cake rewards** will be distributed among every holder proportional to how many tokens each individual holds in values of **8% when buying and selling**.
- ❖ The **sustainability fee of 4% for Dev and 2% for marketing when buying and selling** is what allows UpCake to hold the aforementioned promise. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, UpCake will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.
- ❖ The additional component included under the sustainability section is a **liquidity fee of 1% from buying and selling**, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.
- ❖ Anti-bot and Anti-whale measures to keep the market stable.

# Tokenomics

## 15% fee when buying and selling

- ❖ 8% of trade goes to holders' pockets in Cake tokens.
- ❖ 1% of trade goes to the liquidity pool.
- ❖ 4% of trade goes to the marketing wallet.
- ❖ 2% of trade goes to the dev wallet.

### Grigor Barseghyan



Grigor is the leader of UpCake and he is in the last year of Business School in London, and he's been into Crypto for the past 2 years. With his team, he hopes to bring the best of the NFT world into the smart Chain.

### Suren Manrikyan



He is Suren and he has been studying Economics at Fudan University in China. He speaks 3 languages fluently: Russian, Chinese and Armenian, and will be leading communities for UpCake.

### Artash Shegunts



He is Artash and my background is in marketing. He'll be leading the marketing effort of UpCake, in partnership with BlockChainPros.

# Roadmap

## Phase 1

- ❖ Creation of Socials
- ❖ UpCake Deployment of Smart Contract with 800 Optimization
- ❖ First Audit Ordered (RugFreeCoins)
- ❖ First Marketing Efforts
- ❖ Information Campaign About Combining Rebase and Staking for Maximum Upside
- ❖ Full Doxx

## Phase 2

- ❖ First Voice AMA
- ❖ Private Sale to Give Priority to Earliest Believers
- ❖ Staking Vault Released and Operational
- ❖ Second Marketing Efforts
- ❖ Second Voice AMA

## Phase 3

- ❖ Presale on PinkSale
- ❖ Launch on PancakeSwap
- ❖ Expert Application for CG and CMC
- ❖ DexTools Trending
- ❖ Major Marketing Push on Multiple CoinSites, Facebook, Twitter and Discord Groups

## Phase 4

- ❖ First Video AMA for UpCake Community
- ❖ AMA with BankerDoge Team
- ❖ GiveAway Campaign and Community Prizes
- ❖ Second Audit Ordered (TechRate)
- ❖ AMAs with Large Investor Communities



# Target market and the concept

## Target market

- ❖ Anyone who's interested in Crypto space with long term investment plans.
- ❖ Anyone who's ready to earn a passive income in Cake by holding tokens.
- ❖ Anyone who believes in rebase mechanisms to hold tokens for a longer duration with profits.
- ❖ Anyone who's interested in owning NFTs.
- ❖ Anyone who's interested in doing daily tasks in the game and gets rewards in \$EpicHero tokens.
- ❖ Anyone who's interested in collecting NFTs or trading NFTs.
- ❖ Anyone who's interested in trading tokens.
- ❖ Anyone who's interested in staking and getting rewards.
- ❖ All Cake investors and fans out there.
- ❖ Anyone who's interested in taking part with the future plans of the UpCake token.
- ❖ Anyone who's interested in making financial transactions with any other party using Cake or UpCake as the currency.

## Core concept

### Rebase mechanism

Rebase is the newest tokenomics in BSC, which will revolutionize crypto space with an elastic supply. This guarantees a growing chart as the price per token constantly increases. There are 3 variables that influence each other to quantify the size of a token.

- ❖ Price per token
- ❖ Total supply
- ❖ Amount of BNBs in the liquidity pool

By decreasing the total supply and keeping the amount of BNBs in the LP constant, price per token mechanically increases.

### The UpCake reward system

8% of each transaction when buying and selling gets converted to Cake and is split amongst all holders. Holders will be eligible to receive tokens every five hours and rewards are proportional to how many tokens each individual holds.

## **Sustainable mechanism**

The **sustainability fee of 4% for marketing and 2% for development** is what allows UpCake to promote the token and use funds to further development of the platform. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, UpCake 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 liquidity fee of 1%**, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity

## **Anti-bot measures**

No single wallet is allowed to purchase more than 0.5% of the total supply in one transaction. This restriction, coupled with 50 seconds cool down period, which will limit bot users from fluctuating the price at launch or at any period in the life of the token.

## **Anti-whale measures**

No single private wallet will be allowed to own more than 1% of the total supply to avoid large holders having too much pricing over the market.

# UpStake, UpCake's Staking EcoSystem

As soon as our vaults become available, UpCake holders will be able to stake their UpCake tokens directly on the UpStake platform. Every time a holder stakes UpCake tokens, they have a chance to generate an UpStake NFT from one of different rarity classes. The longer and the more UpCake tokens a holder stakes, the higher the chances of them generating a NFT, and of it being of a higher rarity class.

UpStake NFTs can be collected, exchanged, and bred on the UpStake platform. Bred NFTs often will be less rare than their parents, sometimes of the same rarity as one of their parents, and in extremely few cases more rare than their parents.



UpStake NFTs can be collected, exchanged, and bred on the UpStake platform. Bred NFTs often will be less rare than their parents, sometimes of the same rarity as one of their parents, and in extremely few cases more rare than their parents.

Breeding NFTs will require a money contribution, either in BNB or in UpCake tokens. Using UpCake tokens will be at a significant discount compared to BNBs, with discounts varying depending on which phase the project is at. Paying for breeding costs in UpCake results in a:

Phase 1: 50% discount

Phase 2: 35% discount

Phase 3: 20% discount

Phase 4: 10% discount





The exchange commission for NFTs on the UpStake platform will also be subject to the same discounts.

We know that it is hard to establish the legitimacy of a new token from scratch, which is why in order to achieve our goal of having the UpCake token be used as the preferred currency in all the UpStake transactions, we decided to offer discounts steep enough to incentivize users to trade their BNBs for UpCake and use them on the platform. As we execute our plan and move from phase to phase, the token will be more established, and the discounts needed to incentivize holders will become lower.

Note that we are also allowing users to make their payments in BNB as we want to make the UpStake platform as accessible as possible for early users.





# Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	10/10
3	Information quality	10/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	10/10
7	Impact on the business	10/10
8	Preparing for the future	9/10
Total Points		<b>9.5/10</b>

# Contract details

## Token contract details for 12<sup>th</sup> October 2021

<b>Contract name</b>	UpCake
<b>Contract address</b>	0xae916ea422623734e3a87718524859f05ca9de6e
<b>Token supply</b>	1,000,000,000,000,000
<b>Token ticker</b>	UPC
<b>Decimals</b>	8
<b>Token holders</b>	4
<b>Transaction count</b>	4
<b>Auto liquidity receiver</b>	0x4dd203f368570d6855c380771cbe818f29b338b3
<b>Distributor</b>	0x287c66d9a36c6146fc77073df9a5d964e66b77a1
<b>Contract deployer address</b>	0x4Dd203f368570d6855c380771cbe818f29B338b3
<b>Contract's current owner address</b>	0x4dd203f368570d6855c380771cbe818f29b338b3

# Token distribution

**Tokens are distributed as follows:**




















- Token burn :43%
- Presale & liquidity :37%
- NFT ecosystem :15%
- Private sale :10%

# 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	informational
		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		low issue
12	Fake deposit		low issue
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		
L	Function Name	Visibility	Mutability	Modifiers
<b>SafeMath</b>	<b>Library</b>			
L	add	Internal 		
L	sub	Internal 		
L	sub	Internal 		
L	mul	Internal 		
L	div	Internal 		
L	div	Internal 		
<b>SafeMathInt</b>	<b>Library</b>			
L	mul	Internal 		
L	div	Internal 		
L	sub	Internal 		
L	add	Internal 		
L	abs	Internal 		
<b>IBEP20</b>	<b>Interface</b>			
L	totalSupply	External 		NO 
L	decimals	External 		NO 
L	symbol	External 		NO 
L	name	External 		NO 















L	getOwner	External ¶		NO¶
L	balanceOf	External ¶		NO¶
L	transfer	External ¶	⦿	NO¶
L	allowance	External ¶		NO¶
L	approve	External ¶	⦿	NO¶
L	transferFrom	External ¶	⦿	NO¶
<b>Auth</b>	<b>Implementation</b>			
L		Public ¶	⦿	NO¶
L	authorize	Public ¶	⦿	onlyOwner
L	unauthorize	Public ¶	⦿	onlyOwner
L	isOwner	Public ¶		NO¶
L	isAuthorized	Public ¶		NO¶
L	transferOwnership	Public ¶	⦿	onlyOwner
<b>IDEXFactory</b>	<b>Interface</b>			
L	createPair	External ¶	⦿	NO¶
<b>InterfaceLP</b>	<b>Interface</b>			
L	sync	External ¶	⦿	NO¶
<b>IDEXRouter</b>	<b>Interface</b>			
L	factory	External ¶		NO¶
L	WETH	External ¶		NO¶
L	addLiquidity	External ¶	⦿	NO¶

L	addLiquidityETH	External 		NO 
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External 		NO 
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External 		NO 
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External 		NO 
<b>IDividendDistributor</b>	<b>Interface</b>			
L	setDistributionCriteria	External 		NO 
L	setShare	External 		NO 
L	deposit	External 		NO 
L	process	External 		NO 
<b>DividendDistributor</b>	<b>Implementation</b>	<b>IDividendDistributor</b>		
L		Public 		NO 
L	setDistributionCriteria	External 		onlyToken
L	setShare	External 		onlyToken
L	deposit	External 		onlyToken
L	process	External 		onlyToken
L	shouldDistribute	Internal 		
L	distributeDividend	Internal 		
L	claimDividend	External 		NO 



L	getUnpaidEarnings	Public 		NO 
L	getCumulativeDividends	Internal 		
L	addShareholder	Internal 		
L	removeShareholder	Internal 		
<b>UpCake</b>	<b>Implementation</b>	<b>IBEP20, Auth</b>		
L	rebase_percentage	Public 		onlyOwner
L	rebase	Public 		onlyMaster
L		Public 		Auth
L		External 		NO 
L	totalSupply	External 		NO 
L	decimals	External 		NO 
L	symbol	External 		NO 
L	name	External 		NO 
L	getOwner	External 		NO 
L	balanceOf	Public 		NO 
L	allowance	External 		NO 
L	approve	Public 		NO 
L	approveMax	External 		NO 
L	transfer	External 		NO 
L	transferFrom	External 		NO 
L	_transferFrom	Internal 		
L	_basicTransfer	Internal 		
L	checkTxLimit	Internal 		
L	shouldTakeFee	Internal 		

L	takeFee	Internal 		
L	shouldSwapBack	Internal 		
L	clearStuckBalance	External 		authorize d
L	clearStuckBalance _sender	External 		authorize d
L	set_sell_multiplier	External 		onlyOwn er
L	tradingStatus	Public 		onlyOwn er
L	launchStatus	Public 		onlyOwn er
L	enable_hotel_Calif orniaMode	Public 		onlyOwn er
L	set_max_roomrent	Public 		onlyOwn er
L	manage_housegue sts	Public 		onlyOwn er
L	cooldownEnabled	Public 		onlyOwn er
L	swapBack	Internal 		swapping
L	setIsDividendExem pt	External 		authorize d
L	setIsFeeExempt	External 		authorize d
L	setIsTxLimitExempt	External 		authorize d
L	setIsTimelockExem pt	External 		authorize d
L	setFees	External 		authorize d
L	setFeeReceivers	External 		authorize d
L	setSwapBackSettin gs	External 		authorize d

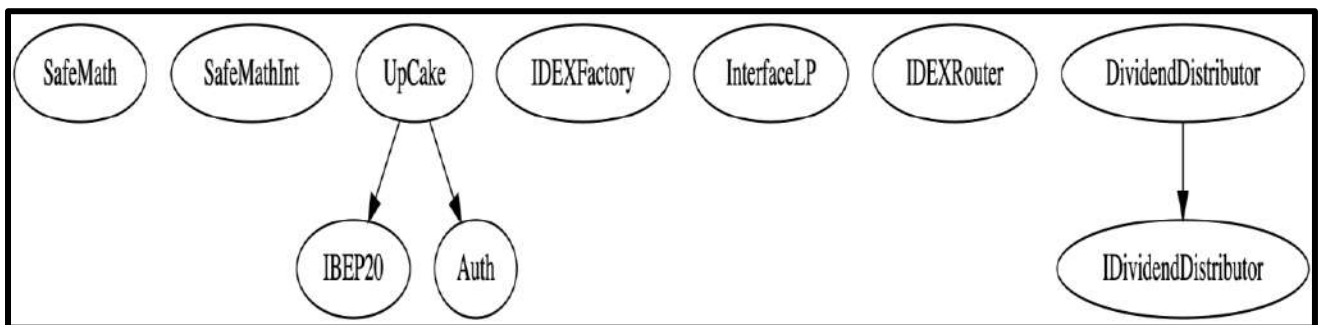
L	setTargetLiquidity	External ¶		authorized
L	manualSync	External ¶		NO¶
L	setLP	External ¶		onlyOwner
L	setMaster	External ¶		onlyOwner
L	isNotInSwap	External ¶		NO¶
L	checkSwapThreshold	External ¶		NO¶
L	setDistributionCriteria	External ¶		authorized
L	setDistributorSettings	External ¶		authorized
L	rescueToken	Public ¶		onlyOwner
L	getCirculatingSupply	Public ¶		NO¶
L	getLiquidityBacking	Public ¶		NO¶
L	isOverLiquified	Public ¶		NO¶
L	checkMaxWalletToken	External ¶		NO¶
L	checkMaxTxAmount	External ¶		NO¶
L	setMaxWalletPercent_base1000	External ¶		onlyOwner
L	setMaxTxPercent_base1000	External ¶		onlyOwner
L	multiTransfer	External ¶		onlyOwner
L	multiTransfer_fixed	External ¶		onlyOwner
L	rebase_updatebalance	External ¶		onlyOwner



### Legend

Symbol	Meaning
	Function can modify state
	Function is payable

### Inheritance Hierarchy



## Security issue checking status

- ❖ **High severity issues**  
No medium severity issues found.
- ❖ **Medium severity issues**  
No medium severity issues found.
- ❖ **Low severity issues**  
No low severity issues found.

## ❖ Informational

- The owner can enable and disable trading any time.

```
ftrace | funcSig
function tradingStatus(bool _status↑, uint256 _deadBlocks↑) public onlyOwner {
    tradingOpen = _status↑;
    if (tradingOpen && launchedAt == 0) {
        launchedAt = block.number;
        deadBlocks = _deadBlocks↑;
    }
}
```

- The owner can change the sell fee multiplier without any limit.

```
ftrace | funcSig
function set_sell_multiplier(uint256 Multiplier↑) external onlyOwner {
    sellMultiplier = Multiplier↑;
}
```

# Owner privileges

- ❖ The owner can add/ remove authorized persons.

```
ftrace | funcSig
function authorize(address adr↑) public onlyOwner {
    | | authorizations[adr↑] = true;
}

ftrace | funcSig
function unauthorize(address adr↑) public onlyOwner {
    | | authorizations[adr↑] = false;
}
```

- ❖ The owner can transfer ownership.

```
ftrace | funcSig
function transferOwnership(address payable adr↑) public onlyOwner {
    owner = adr↑;
    authorizations[adr↑] = true;
    emit OwnershipTransferred(adr↑);
}

event OwnershipTransferred(address owner);
```

- ❖ The owner can transfer the BNB balance in the contract to marketing wallet.

```
ftrace | funcSig
function clearStuckBalance(uint256 amountPercentage↑) external authorized {
    uint256 amountBNB = address(this).balance;
    payable(marketingFeeReceiver).transfer(
        (amountBNB * amountPercentage↑) / 100
    );
}
```

- ❖ The owner can transfer the contract BNB balance to owner wallet.

```
ftrace | funcSig
function clearStuckBalance_sender(uint256 amountPercentage↑)
    external
    authorized
{
    uint256 amountBNB = address(this).balance;
    payable(msg.sender).transfer((amountBNB * amountPercentage↑) / 100);
}
```

- ❖ The owner can change the sell fee multiplier.

```
ftrace | funcSig
function set_sell_multiplier(uint256 Multiplier↑) external onlyOwner {
    sellMultiplier = Multiplier↑;
}
```

- ❖ The owner can enable and disable trading.

```
ftrace | funcSig
function tradingStatus(bool _status↑, uint256 _deadBlocks↑) public onlyOwner {
    tradingOpen = _status↑;
    if (tradingOpen && launchedAt == 0) {
        launchedAt = block.number;
        deadBlocks = _deadBlocks↑;
    }
}
```

- ❖ The owner can enable buy cool down and time interval

```
ftrace | funcSig
function cooldownEnabled(bool _status↑, uint8 _interval↑) public onlyOwner {
    buyCooldownEnabled = _status↑;
    cooldownTimerInterval = _interval↑;
}
```

- ❖ The owner can enable California mode and it's settings. (anti bot feature)

```
// lobby manager
ftrace | funcSig
function enable_hotel_CaliforniaMode(bool _status↑) public onlyOwner {
    hotelCaliforniaMode = _status↑;
}

ftrace | funcSig
function set_max_roomrent(uint256 _rent_withoutdecimal↑) public onlyOwner {
    maxRoomRent = _rent_withoutdecimal↑ * 10**9;
}

ftrace | funcSig
function manage_houseguests(address[] calldata addresses↑, bool status↑)
    public
    onlyOwner
{
    for (uint256 i; i < addresses↑.length; ++i) {
        isHouseguest[addresses↑[i]] = status↑;
    }
}
```

- ❖ The owner can change swap settings and liquidity settings.

```
ftrace | funcSig
function setSwapBackSettings(bool _enabled↑, uint256 _percentage_base10000↑)
    external
    authorized
{
    swapEnabled = _enabled↑;
    swapThreshold = rSupply.div(10000).mul(_percentage_base10000↑);
}

ftrace | funcSig
function setTargetLiquidity(uint256 _target↑, uint256 _denominator↑)
    external
    authorized
{
    targetLiquidity = _target↑;
    targetLiquidityDenominator = _denominator↑;
}
```



- ❖ The owner can change the LP address.

```
ftrace | funcSig
function setLP(address _address↑) external onlyOwner {
    pairContract = InterfaceLP(_address↑);
    isFeeExempt[_address↑];
}
```

- ❖ The owner can exempt wallets from dividends, fees, transactions limit and time lock.

```
ftrace | funcSig
function setIsDividendExempt(address holder↑, bool exempt↑)
    external
    authorized
{
    require(holder↑ != address(this) && holder↑ != pair);
    isDividendExempt[holder↑] = exempt↑;
    if (exempt↑) {
        distributor.setShare(holder↑, 0);
    } else {
        distributor.setShare(holder↑, rBalance[holder↑]);
    }
}

ftrace | funcSig
function setIsFeeExempt(address holder↑, bool exempt↑) external authorized {
    isFeeExempt[holder↑] = exempt↑;
}

ftrace | funcSig
function setIsTxLimitExempt(address holder↑, bool exempt↑)
    external
    authorized
{
    isTxLimitExempt[holder↑] = exempt↑;
}

ftrace | funcSig
function setIsTimelockExempt(address holder↑, bool exempt↑)
    external
    authorized
{
    isTimelockExempt[holder↑] = exempt↑;
}
```

- ❖ The owner can change dividend distribution criteria and settings.

```
ftrace | funcSig
function setDistributionCriteria(
    uint256 _minPeriod↑,
    uint256 _minDistribution↑
) external authorized {
    distributor.setDistributionCriteria(_minPeriod↑, _minDistribution↑);
}

ftrace | funcSig
function setDistributorSettings(uint256 gas↑) external authorized {
    require(gas↑ < 900000);
    distributorGas = gas↑;
}
```

- ❖ The owner can transfer other tokens in contract to his wallet.

```
ftrace | funcSig
function rescueToken(address tokenAddress↑, uint256 tokens↑)
    public
    onlyOwner
    returns (bool success↑)
{
    return IBEP20(tokenAddress↑).transfer(msg.sender, tokens↑);
}
```

- ❖ The owner can change max wallet token amount and max transaction amount.

```
ftrace | funcSig
function setMaxWalletPercent_base1000(uint256 maxWallPercent_base1000↑)
    external
    onlyOwner
{
    _maxWalletToken = rSupply.div(1000).mul(maxWallPercent_base1000↑);
}

ftrace | funcSig
function setMaxTxPercent_base1000(uint256 maxTxPercentage_base1000↑)
    external
    onlyOwner
{
    _maxTxAmount = rSupply.div(1000).mul(maxTxPercentage_base1000↑);
}
```

# Audit conclusion

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