RugFreeCoins Audit

Infinity Wealth Audit

Smart Contract Security Audit

November 03, 2021

Contents

Audit details	1
Disclaimer	2
Background	3
About the project	4
Target market and the concept	6
Potential to grow with score points	7
Total Points	7
Contract details	8
Token distribution	9
Contract code function details	10
Contract description table	11
Security issue checking status	21
Owner privileges	23
Audit conclusion	29

Audit details

Audited project

Infinity Wealth Token

Contract Address

0xd30de3a9c917642a43e3df4f0d2627a26b6e4d6a

Client contact

Infinity Wealth Team

Blockchain

Binance smart chain

Project website

https://infinitywealth.io/

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/0xd30de3a9c917642a43e3df4f0d2627a26b6e4d6a

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

Infinity Wealth is a token built on the Binance Smart Chain. Each transaction, purchase incurs 11% and sale incur 14% fee.

Features

- The **Infinity Wealth Token** rewards will be distributed among every holder proportional to how many tokens each individual holds in values of **2% when buying and selling**.

- The **sustainability fee of 5% when buying and 6% when selling for marketing and 1% when buying and selling** is what allows Infinity Wealth to hold the aforementioned promise. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, Infinity Wealth 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.

- **The buyback and burn fee of 2% from buying and 4% from selling** is used to save from massive dips to keep the token market price stable.

Tokenomics

11% fee when buying

- 2% of trade goes to holders' pockets in Infinity Wealth tokens.
- 1% of trade goes to the liquidity pool.
- 5% of trade goes to the marketing wallet.
- 1% of trade goes to the dev wallet.
- 2% of trade goes to buyback & burn.

14% fee when selling

- 2% of trade goes to holders' pockets in Infinity Wealth tokens.
- 1% of trade goes to the liquidity pool.
- 6% of trade goes to the marketing wallet.
- 1% of trade goes to the dev wallet.
- 4% of trade goes to buyback & burn.

Roadmap

Project Conception & Launch – Q4,2021

Focusing on smooth launch during Q4.

- Contract
- Private and pre-sale
- PCS launch
- CMC & CG listing
- ETH Blockchain implementation

Moon landing – Q1,2022

We will be looking at getting the vaults developed. Our multi blockchain capability should now be available. We will expand on our operations and start planning the future platform. During this time, we will continue driving mass marketing and doing several partnership outreach program initiatives.

- iOS/ Android App (Beta)
- Legal entity setup
- Platform planning
- Team expansion
- Website V2 launch

Stellar Slightshot – Q2,2022

Strategic partnership and joint ventures will allow us to expand further into other opportunities. We will also be launching updates to our mobile apps and invite users to our beta platform.

- iOS/Android app updates
- Platform beta release invite
- Sponsored crypto events
- Major partnership announcements
- Community meetups

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 Infinity Wealth Tokens by holding tokens.
- Anyone who's interested in trading tokens.
- Anyone who's interested in taking part with the future plans of the Infinity Wealth token.
- Anyone who's interested in making financial transactions with any other party using Infinity Wealth as the currency.

Core concept

The Infinity Wealth reward system

2% of each transaction when buying and selling gets converted to tokens 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 5% when buying and 6% when selling for marketing and 1% for development** is what allows Infinity Wealth 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, Infinity Wealth 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.

The controlled buyback collects 2% tax when buying and 4% when selling on each transaction, which is stored inside a private wallet in BNB. It is to save from massive dips in order to keep the token market price stable.

Potential to grow with score points

1.	Project efficiency	8/10
2.	Project uniqueness	8/10
3	Information quality	8/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	8/10
7	Impact on the business	8/10
8	Preparing for the future	7/10
Total Points		**8.125/10**

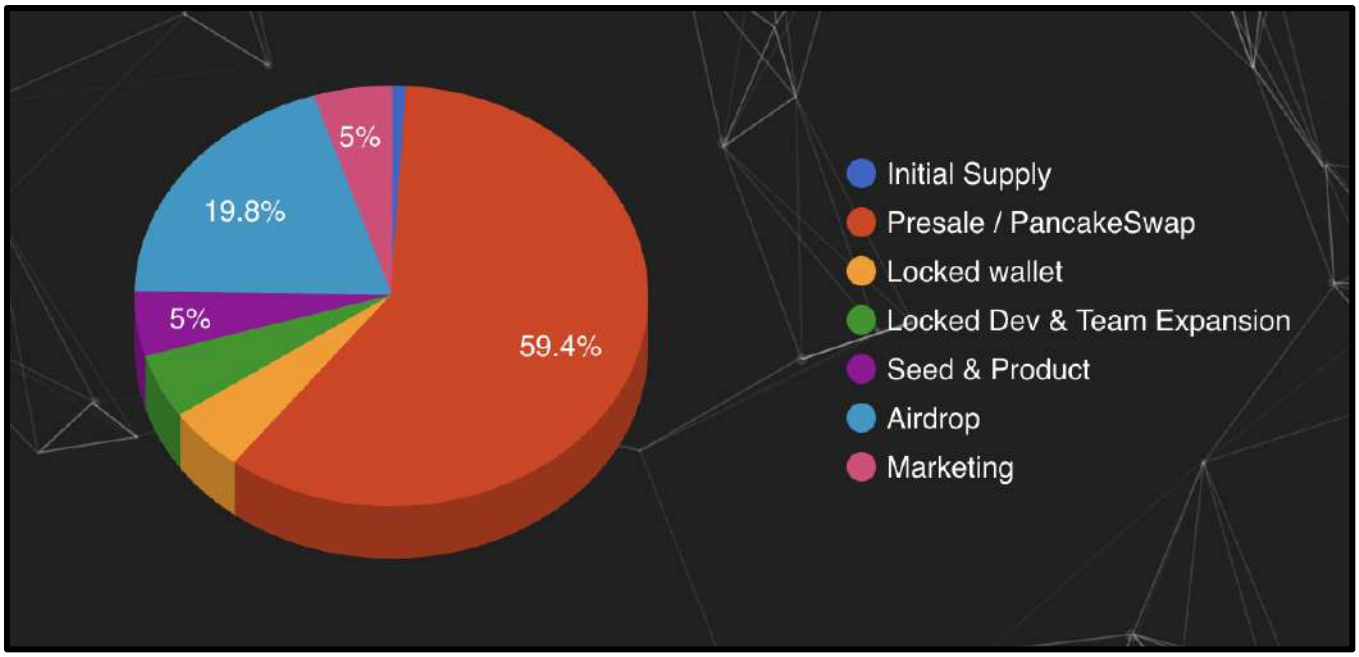
Contract details

Token contract details for 03rd November 2021

Contract name	Infinity Wealth
Contract address	0xd30de3a9c917642a43e3df4f0d2627a26b6e4d6a
Token supply	1,000,000,000,000
Token ticker	INFINITY
Decimals	18
Token holders	1
Transaction count	1
Buyback wallet	0x9dddb0a127dc3692d029e2034a6748d44b6c2a9a
Dev wallet	0x18fca8704312760a0d4ef10608c5cf85dd01f6ba
Marketing wallet	0xbe4557aba4dfa3017a9a1c4ae051a3fd8fee8a90
Contract deployer address	0x18fcA8704312760A0D4ef10608C5Cf85Dd01F6Ba
Contract's current owner address	0x578d6b37f5afdc1bf2a4db31fe33448a7f55e9b7

Token distribution

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	medium/low issue
		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	informational
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
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 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		

└	sub	Internal 🔒		
└	div	Internal 🔒		
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Address	**Library**			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	_verifyCallResult	Private 🔐		

Ownable	**Implementation**	**Context**		
└		Public ❗	🔘	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔘	onlyOwner
└	transferOwnership	Public ❗	🔘	onlyOwner
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❗

L	swapTokensForExactTokens	External ❗	🔘	NO❗
L	swapExactETHForTokens	External ❗	💷	NO❗
L	swapTokensForExactETH	External ❗	🔘	NO❗
L	swapExactTokensForETH	External ❗	🔘	NO❗
L	swapETHForExactTokens	External ❗	💷	NO❗
L	quote	External ❗		NO❗
L	getAmountOut	External ❗		NO❗
L	getAmountIn	External ❗		NO❗
L	getAmountsOut	External ❗		NO❗
L	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	**Interface**	**IUniswapV2 Router01**		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔘	NO❗
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔘	NO❗
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔘	NO❗
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💷	NO❗

└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔘	NO❗
InfinityWealth	**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❗

└	Presale_BEGIN	External ❗	🔘	onlyOwner
└	Presale_END	External ❗	🔘	onlyOwner
└	deliver	Public ❗	🔘	NO❗
└	reflectionFromToken	Public ❗		NO❗
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	Public ❗	🔘	onlyOwner
└	includeInReward	External ❗	🔘	onlyOwner
└	set_Transfers_Without_Fees	External ❗	🔘	onlyOwner
└	excludeFromFee	Public ❗	🔘	onlyOwner
└	includeInFee	Public ❗	🔘	onlyOwner
└	set_as_Pair	External ❗	🔘	onlyOwner
└	_set_Fees	External ❗	🔘	onlyOwner
└	Wallet_Update_Marketing	Public ❗	🔘	onlyOwner
└	Wallet_Update_Buyback	Public ❗	🔘	onlyOwner
└	set_Swap_And_Liquify_Enabled	Public ❗	🔘	onlyOwner
└	set_Number_Of_Transactions_Before_Liquify_Trigger	Public ❗	🔘	onlyOwner
└		External ❗	💵	NO❗

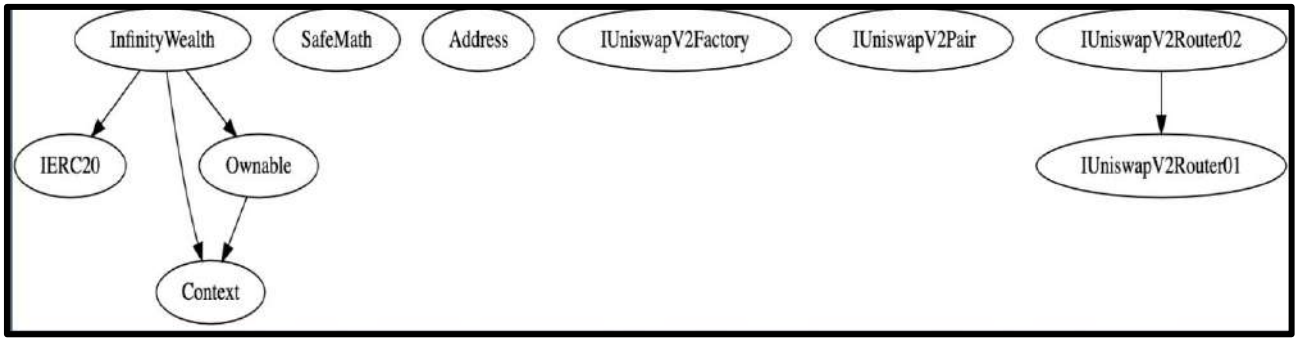
└	blacklist_Add_Wallets	External ❗	🛑	onlyOwner
└	blacklist_Remove_Wallets	External ❗	🛑	onlyOwner
└	blacklist_Switch	Public ❗	🛑	onlyOwner
└	set_Max_Transaction_Percent_x100	External ❗	🛑	onlyOwner
└	set_Max_Wallet_Holding_Percent_x100	External ❗	🛑	onlyOwner
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	_reflectFee	Private 🔐	🛑	
└	calculateFeeReflection	Private 🔐		
└	calculateLiquidityAndPromoFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	removeWalletLimits	Private 🔐	🛑	

L	restoreWalletLimits	Private 🔐	🛑	
L	_approve	Private 🔐	🛑	
L	_transfer	Private 🔐	🛑	
L	sendToWallet	Private 🔐	🛑	
L	precDiv	Internal 🔒		
L	swapAndLiquify	Private 🔐	🛑	lockTheSwap
L	swapTokensForEth	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	remove_Random_Tokens	Public ❗	🛑	onlyOwner
L	set_New_Router_and_Make_Pair	Public ❗	🛑	onlyOwner
L	set_New_Router_Address	Public ❗	🛑	onlyOwner
L	set_New_Pair_Address	Public ❗	🛑	onlyOwner
L	process_SwapAndLiquify_Now	Public ❗	🛑	onlyOwner
L	_tokenTransfer	Private 🔐	🛑	
L	_transferStandard	Private 🔐	🛑	
L	_transferToExcluded	Private 🔐	🛑	
L	_transferFromExcluded	Private 🔐	🛑	
L	_transferBothExcluded	Private 🔐	🛑	

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**
  - Immutable variable.

```
address payable public Wallet_Burn =
    payable(0x000000000000000000000000000000000000dEaD);
```

Wallet_burn is only used for checking some conditions and not transferring any payments to burn wallet inside contract so by removing payable and marking it as immutable can reduce some gas fees

❖ **Low severity issues**

- In the includeInReward function, if they use a long wallet list there can be an OUT_OF_GAS issue, better to use a small array list at once.

```
ftrace | funcSig
function includeInReward(address account⬆) external onlyOwner {
    require(_isExcluded[account⬆], "Account is already included");
    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;
        }
    }
}
```

❖ **Informational**

- Token buy back feature is not functioning through the contract, buyback BNB amount goes to a private wallet and wallet owner should manually do the buyback**.**

```
    totalBNB = address(this).balance;
    splitM = precDiv(_FeeMarketing, _promoFee, 2);
    uint256 marketingBNB = (totalBNB * splitM) / 100;
    splitBB = precDiv(_FeeBuyBack, _promoFee, 2);
    uint256 buybackBNB = (totalBNB * splitBB) / 100;
    sendToWallet(Wallet_Marketing, marketingBNB);
    sendToWallet(Wallet_BuyBack, buybackBNB);
    sendToWallet(Wallet_Dev, (totalBNB - marketingBNB - buybackBNB));
} else if (_promoFee != 0 && _FeeLiquidity == 0) {
    swapTokensForEth(contractTokenBalance⬆);
    totalBNB = address(this).balance;
    splitM = precDiv(_FeeMarketing, _promoFee, 2);
    uint256 marketingBNB = (totalBNB * splitM) / 100;
    splitBB = precDiv(_FeeBuyBack, _promoFee, 2);
    uint256 buybackBNB = (totalBNB * splitBB) / 100;
    sendToWallet(Wallet_Marketing, marketingBNB);
    sendToWallet(Wallet_BuyBack, buybackBNB);
    sendToWallet(Wallet_Dev, (totalBNB - marketingBNB - buybackBNB));
}
```

Owner privileges

- The owner can start and end pre sale.

```
// Get ready for presale!
ftrace | funcSig
function Presale_BEGIN() external onlyOwner {
    set_Swap_And_Liquify_Enabled(false);
    removeAllFee();
    removeWalletLimits();
}

// Presale done! Set all fees
ftrace | funcSig
function Presale_END() external onlyOwner {
    set_Swap_And_Liquify_Enabled(true);
    restoreAllFee();
    restoreWalletLimits();
}
```

- The owner can include and 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 included");
    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 enable and disable set transfer without fees. (If this is true, fees will deduct when transferring tokens to wallet to wallet).

```
ftrace | funcSig
function set_Transfers_Without_Fees(bool true_or_false⬆) external onlyOwner {
    noFeeToTransfer = true_or_false⬆;
}
```

❖ The owner can include and exclude wallets from fees.

```
// Set a wallet address so that it does not have to pay transaction fees
ftrace | funcSig
function excludeFromFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = true;
}

// Set a wallet address so that it has to pay transaction fees
ftrace | funcSig
function includeInFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = false;
}
```

❖ The owner can change marketing and buyBack wallet.

```
//Update the marketing wallet
ftrace | funcSig
function Wallet_Update_Marketing(address payable wallet⬆) public onlyOwner {
    Wallet_Marketing = wallet⬆;
    _isExcludedFromFee[Wallet_Marketing] = true;
}

//Update the Buyback wallet
ftrace | funcSig
function Wallet_Update_Buyback(address payable wallet⬆) public onlyOwner {
    Wallet_BuyBack = wallet⬆;
    _isExcludedFromFee[Wallet_BuyBack] = true;
}
```

❖ The owner can enable/disable swap and change swap trigger point.

```
// Toggle on and off to activate auto liquidity and the promo wallet
ftrace | funcSig
function set_Swap_And_Liquify_Enabled(bool true_or_false⬆) public onlyOwner {
    swapAndLiquifyEnabled = true_or_false⬆;
    emit SwapAndLiquifyEnabledUpdated(true_or_false⬆);
}

// This will set the number of transactions required before the 'swapAndLiquify' function triggers
ftrace | funcSig
function set_Number_Of_Transactions_Before_Liquify_Trigger(
    uint8 number_of_transactions⬆
) public onlyOwner {
    swapTrigger = number_of_transactions⬆;
}
```

❖ The owner can turn on and off blacklist mode.

```
//Blacklist Switch - Turn on/off blacklisted wallet restrictions
ftrace | funcSig
function blacklist_Switch(bool true_or_false⬆) public onlyOwner {
    noBlackList = true_or_false⬆;
}
```

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

```
// Set the Max transaction amount (percent of total supply)
ftrace | funcSig
function set_Max_Transaction_Percent_x100(
    uint256 max_Transaction_Percent_X100⬆
) external onlyOwner {
    _maxTxAmount = (_tTotal * max_Transaction_Percent_X100⬆) / 10000;
}

// Set the maximum permitted wallet holding (percent of total supply)
ftrace | funcSig
function set_Max_Wallet_Holding_Percent_x100(
    uint256 max_Transaction_Percent_X100⬆
) external onlyOwner {
    _maxWalletToken = (_tTotal * max_Transaction_Percent_X100⬆) / 10000;
}
```

❖ The owner can change the router address.

```
// Set new router
ftrace | funcSig
function set_New_Router_Address(address newRouter⬆) public onlyOwner {
    IUniswapV2Router02 _newPCSRouter = IUniswapV2Router02(newRouter⬆);
    uniswapV2Router = _newPCSRouter;
}

// Set new address - This will be the 'Cake LP' address for the token pairing
ftrace | funcSig
function set_New_Pair_Address(address newPair⬆) public onlyOwner {
    uniswapV2Pair = newPair⬆;
    _isPair[uniswapV2Pair] = true;
}
```

❖ The owner can process swap and liquidity add manually.

```
// Manual 'swapAndLiquify' Trigger (Enter the percent of the tokens that you'd like to send to swap and liquify)
ftrace | funcSig
function process_SwapAndLiquify_Now(uint256 percent_Of_Tokens_To_Liquify⬆)
    public
    onlyOwner
{
    // Do not trigger if already in swap
    require(
        !inSwapAndLiquify,
        "Currently processing liquidity, try later."
    );
    if (percent_Of_Tokens_To_Liquify⬆ > 100) {
        percent_Of_Tokens_To_Liquify⬆ == 100;
    }
    uint256 tokensOnContract = balanceOf(address(this));
    uint256 sendTokens = (tokensOnContract * percent_Of_Tokens_To_Liquify⬆) /
        100;
    swapAndLiquify(sendTokens);
}

/*
```

❖ The owner can change all fees.

```
ftrace | funcSig
function _set_Fees(
    uint256 Liquidity,
    uint256 Reflection,
    uint256 Marketing_BUY,
    uint256 Marketing_SELL,
    uint256 BuyBack_BUY,
    uint256 BuyBack_SELL
) external onlyOwner {
    // Set the fees

    _FeeLiquidity = Liquidity;
    _FeeReflection = Reflection;
    _FeeMarketing_BUY = Marketing_BUY;
    _FeeMarketing_SELL = Marketing_SELL;
    _FeeBuyBack_BUY = BuyBack_BUY;
    _FeeBuyBack_SELL = BuyBack_SELL;

    // For calculations and processing

    _FeeTotal_BUY =
        _FeeMarketing_BUY +
        _FeeBuyBack_BUY +
        _FeeDev +
        _FeeLiquidity +
        _FeeReflection;
    _FeeTotal_SELL =
        _FeeMarketing_SELL +
        _FeeBuyBack_SELL +
        _FeeDev +
        _FeeLiquidity +
        _FeeReflection;
}
```

❖ The owner can add and remove wallets from blacklist.

```
ftrace | funcSig
function blacklist_Add_Wallets(address[] calldata addresses⬆)
    external
    onlyOwner
{
    uint256 startGas;
    uint256 gasUsed;

    for (uint256 i; i < addresses⬆.length; ++i) {
        if (gasUsed < gasleft()) {
            startGas = gasleft();
            if (!_isBlacklisted[addresses⬆[i]]) {
                _isBlacklisted[addresses⬆[i]] = true;
            }
            gasUsed = startGas - gasleft();
        }
    }
}

// Blacklist - block wallets (REMOVE - COMMA SEPARATE MULTIPLE WALLETS)
ftrace | funcSig
function blacklist_Remove_Wallets(address[] calldata addresses⬆)
    external
    onlyOwner
{
    uint256 startGas;
    uint256 gasUsed;

    for (uint256 i; i < addresses⬆.length; ++i) {
        if (gasUsed < gasleft()) {
            startGas = gasleft();
            if (_isBlacklisted[addresses⬆[i]]) {
                _isBlacklisted[addresses⬆[i]] = false;
            }
            gasUsed = startGas - gasleft();
        }
    }
}
```

Audit conclusion

While conducting the audit of the Infinity Wealth smart contract, it was observed that there is nothing alarming with the code and it only contains a medium severity issue, low severity issue and an informational concern.