RugFreeCoins Audit

Fuck Them Heggies Token Audit

Smart Contract Security Audit

August 04, 2021

Contents

Audit details

Audited project

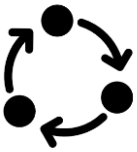
Fuck Them Heggies Token

Contract Address

0xd51D0e1A1cea1d1fa144B7327730fa001F984874

Client contact

Fuck Them Heggies Team



Blockchain

Binance smart chain

Project website

http://fthemheggies.com/

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 Fuck Them Heggies to perform an audit of the smart contract.

https://bscscan.com/token/0xd51D0e1A1cea1d1fa144B7327730fa001F984874

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

Fuck Them Heggies is a token built on the Binance Smart Chain, and the contract has been upgraded from version 1 to version 2. This is version 2, and each transaction, purchase incur a 12% fee, and sales incur a 14% fee.

Features

- The **automatic FTHT rewards of 4% when buying and 5% when selling** is what FTHT's entire marketing strategy is based around: that tokens will be distributed among every holder proportional to how many tokens each individual holds.

- The liquidity fee of 3%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity. This is a key element for decentralized exchanges like Pancakeswap.

- The **sustainability fee of 2% marketing** is what allows FTHT to hold the aforementioned promise. Tokens will be swapped into FTHT and will be sent to a marketing wallet per transaction. This way, FTHT will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.

- **3% Charity fee when buying and 4% charity fee when selling** per transaction will be swapped to BNB and sent to a private wallet. This will empower the FTHT community in the long run and motivate more people to join in!

- **Buyback feature:** If the token selling limit exceeds the token amount of 1,000,000,000 from the last buyback time within the last 24 hours, if the buyback feature is enabled, the contract will calculate the sell history average and buybacks 10% tokens from the total average.

Fuck Them Heggies Token was born out of the "Hodl" and "Diamond Hands" movement. The developer, who is a holder of AMC stock, grew frustrated that hedge funds were allowed to run wild with little to no oversight, while they grew big and rich. They manipulated the markets and left retail investors in their wake. FTHT believes that retail investors are more powerful than they know. They want to empower them with education.

The story

Fuck Them Heggies Token's story begins with a conversation between two brothers. These brothers, who were avid in the cryptomarket, especially Dogecoin were having a conversation one day and the developer said "Fuck Them Heggies." This sparked an idea of a token that will someday be used to empower retail investors. Also being avid crypto investors, and having invested in projects on the Ethereum Chain, they realized that the gas fees are too high and that the Binance Smart Chain was a better avenue.

The mission

FTHT’s mission is a movement in the making. They're striving to provide "Empowerment Through Education."

The goal

FTHT's goal is to have an online learning platform that will help retail investors make more informed decisions about how to use their money. "FOMO" and "Yolo" are not investment strategies that are sustainable. The team wants to empower people with the knowledge that they can play the same game as the "hedge funds." There are more of us than there are of them.

Tokenomics

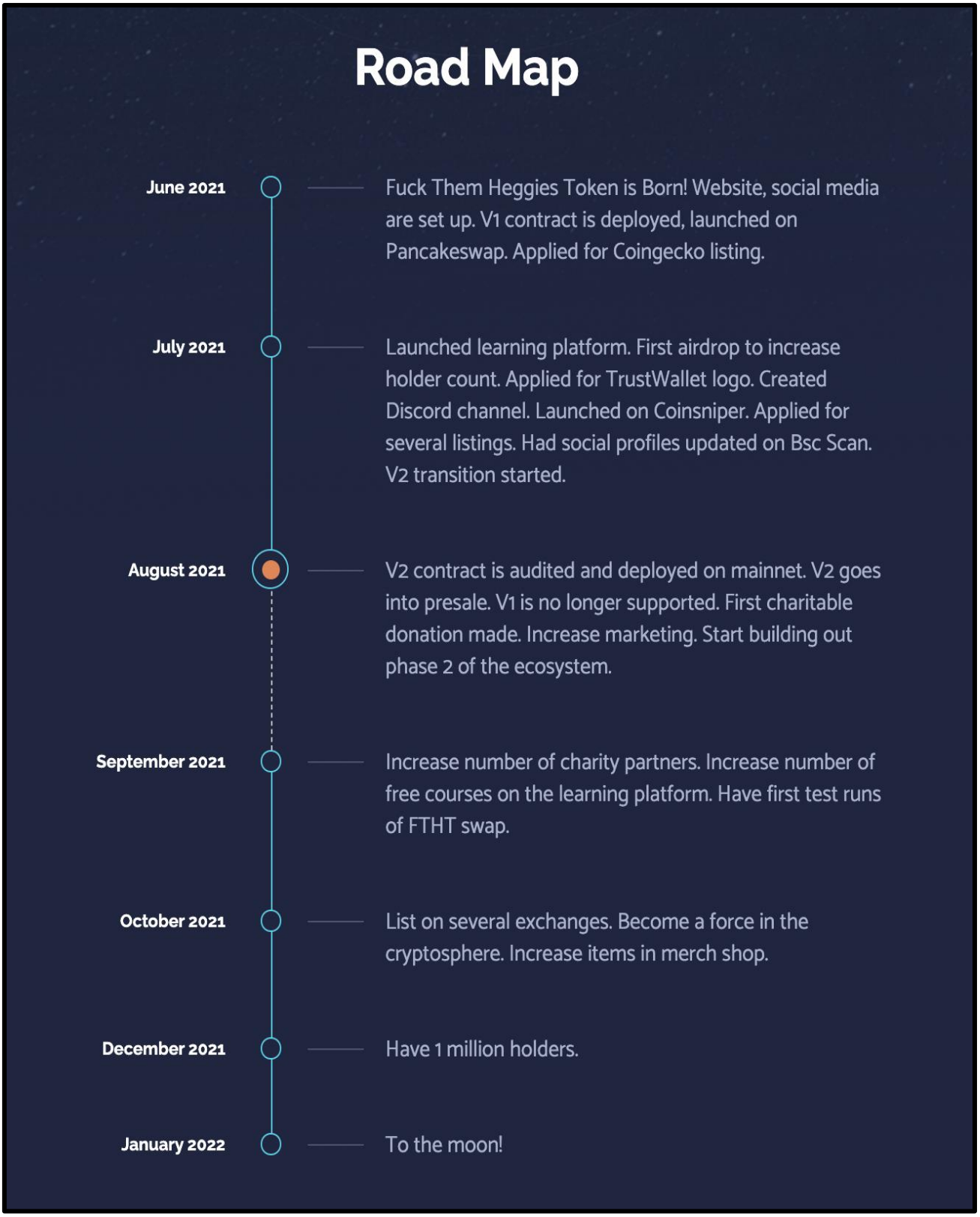
12% fee when buying

- 4% of every trade goes to holders pockets in tokens.
- 2% of every trade goes to the marketing wallet.
- 3% of every trade goes to a charity wallet.
- 3% of every trade goes to the liquidity pool.

14% fee when selling (Additional 2%)

- 5% of every trade goes to holders pockets in tokens.
- 4% of every trade goes to a charity wallet.
- 2% of every trade goes to the marketing wallet.
- 3% of every trade goes to the liquidity pool.

Roadmap


Road Map
June 2021
Fuck Them Heggies Token is Born! Website, social media are set up. V1 contract is deployed, launched on Pancakeswap. Applied for Coingecko listing.
July 2021
Launched learning platform. First airdrop to increase holder count. Applied for TrustWallet logo. Created Discord channel. Launched on Coinsniper. Applied for several listings. Had social profiles updated on Bsc Scan. V2 transition started.
August 2021
V2 contract is audited and deployed on mainnet. V2 goes into presale. V1 is no longer supported. First charitable donation made. Increase marketing. Start building out phase 2 of the ecosystem.
September 2021
Increase number of charity partners. Increase number of free courses on the learning platform. Have first test runs of FTHT swap.
October 2021
List on several exchanges. Become a force in the cryptosphere. Increase items in merch shop.
December 2021
Have 1 million holders.
January 2022
To the moon!

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 by holding tokens.
- Anyone who's interested in trading tokens.
- Anyone who's interested in supporting a good cause. (Donation feature)
- Anyone who's interested in playing fun games and getting rewarded.
- Anyone who's interested in taking part with the e-learning platform.
- Anyone who's interested in stake and farm FTHT and earn rewards.
- Anyone who's interested in making financial transactions with any other party using FTHT as the currency.

Core concept

The reward system

4% of each transaction when buying and 5% of each transaction when selling gets sent amongst all holders in tokens. The holders will be eligible to receive tokens, whenever a transaction occurs, and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The liquidity fee of 3%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

The **fee of 2% marketing** is what allows FTHT to promote the token and use funds to further development of the platform. Tokens will be swapped into BNBs and will be sent to a marketing wallet per transaction. This way, FTHT 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 buyback feature: If the token selling limit exceeds the token amount of 1,000,000,000 from the last buyback time within the last 24 hours, if the buyback feature is enabled, the contract will calculate the sell history average and buybacks 10% tokens from the total average. Those tokens are immediately burned after purchase, which keeps the token market price stable.

Good cause

3% Charity fee when buying and 4% charity fee when selling per transaction will be swapped to BNB and sent to a private wallet which is allocated for charity causes.

The FTHT team will partner with different charities that help support their mission to empower people through education.

The future plan

Fuck Them Heggies – The Game

A video game will be released in the coming weeks, where the holders can play and get rewards in tokens.

FTHT learning platform

FTHT University. A learning platform made by the FTHT community for the FTHT community. This learning platform will be open to anyone who wants to create and sell their own courses. The backbone of the payment system will be $FTHT.

Exchange platform

FTHT is going to provide the most advanced exchange with farming, staking and IFO capabilities.

FTHT incubator and crowdsource

FTHT will provide support for creative projects or people that will strive to change the cryptoworld with good concepts and use cases.

Charities

The team will partner with different charities that help support their mission to empower people through education.

Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	8/10
3	Information quality	7/10
4	Service quality	7/10
5	System quality	7/10
6	Impact on the community	10/10
7	Impact on the business	8/10
8	Preparing for the future	9/10
Total Points		**8.13/10**

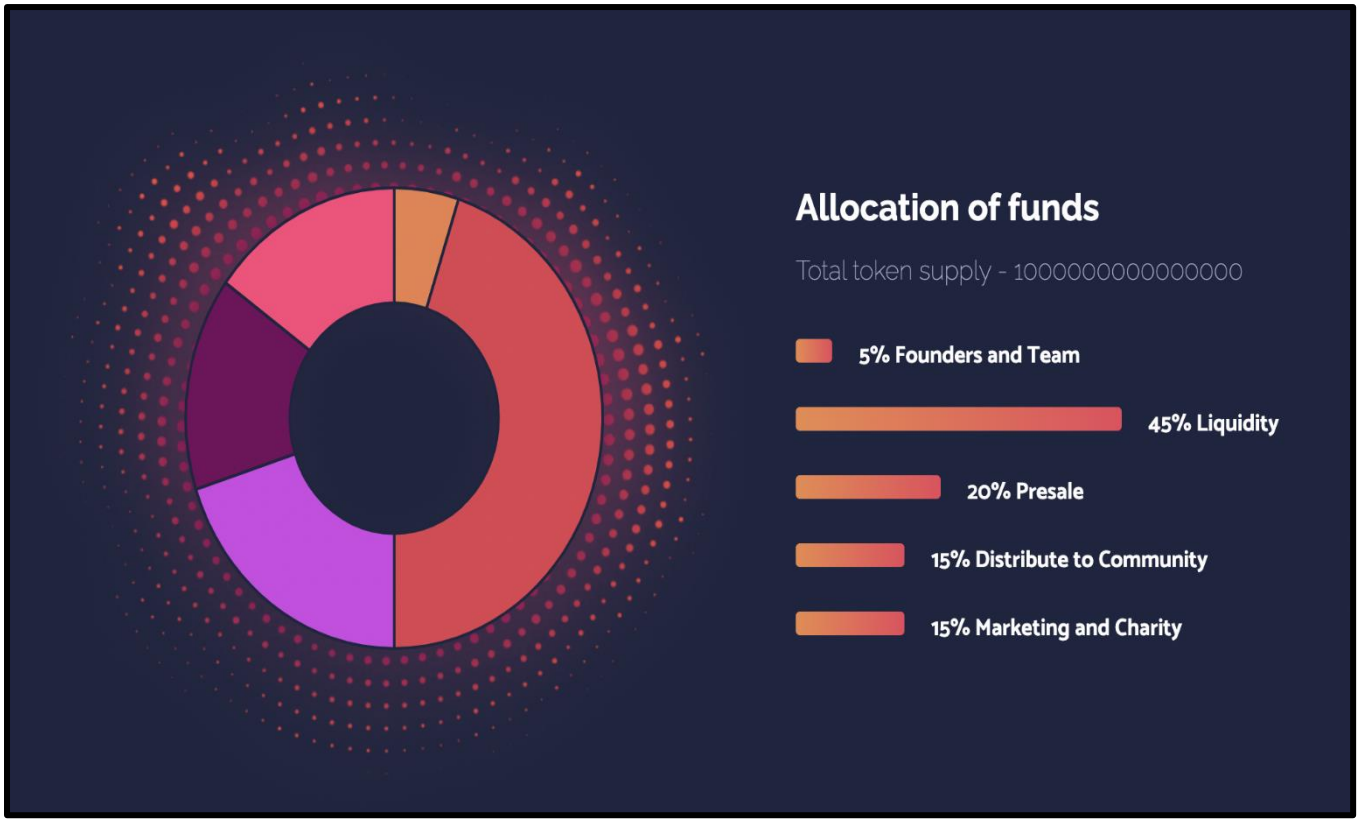
Contract details

Token contract details for 04th August 2021

Contract name	Fuck Them Heggies Token
Contract address	0xd51D0e1A1cea1d1fa144B7327730fa001F984874
Token supply	1,000,000,000,000,000
Token ticker	FTH
Decimals	9
Token holders	1
Transaction count	1
Charity address	0x97845a5c0660508b6ac5b0b5d10644281d253a87
Marketing address	0x0c2b1630ae15a8eed66e888b4b7690fa0011aa5b
Contract deployer address	0x1Df925D52a7dade6411Cf669fa796F1e62bBc509
Contract's current owner address	0x072fdf308e3bf58873e29dbccce144e0b180487b

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	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
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
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 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		

└	mod	Internal 🔒		
Address	**Library**			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	_functionCallWithValue	Private 🔐	🛑	
Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	getUnlockTime	Public ❗		NO❗
└	getTime	Public ❗		NO❗
└	lock	Public ❗	🛑	onlyOwner
└	unlock	Public ❗	🛑	NO❗
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❗
└	swapTokensForExactTokens	External ❗	🔘	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🔘	NO❗

L	swapExactTokens ForETH	External ❗	🔘	NO❗
L	swapETHForExact Tokens	External ❗	💵	NO❗
L	quote	External ❗		NO❗
L	getAmountOut	External ❗		NO❗
L	getAmountIn	External ❗		NO❗
L	getAmountsOut	External ❗		NO❗
L	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	**Interface**	**IUniswapV2Router0 1**		
L	removeLiquidityET HSupportingFeeO nTransferTokens	External ❗	🔘	NO❗
L	removeLiquidityET HWithPermitSupp ortingFeeOnTransf erTokens	External ❗	🔘	NO❗
L	swapExactTokens ForTokensSupport ingFeeOnTransfer Tokens	External ❗	🔘	NO❗
L	swapExactETHFor TokensSupporting FeeOnTransferTo kens	External ❗	💵	NO❗
L	swapExactTokens ForETHSupporting FeeOnTransferTo kens	External ❗	🔘	NO❗
FTHV2	**Implementation**	**Context, IERC20, Ownable**		
L		Public ❗	🔘	NO❗
L	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❗
└	minimumTokensBeforeSwapAmount	Public ❗		NO❗
└	buyBackSellLimitAmount	Public ❗		NO❗
└	deliver	Public ❗	🔘	NO❗
└	reflectionFromToken	Public ❗		NO❗
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	Public ❗	🔘	onlyOwner
└	includeInReward	External ❗	🔘	onlyOwner
└	_approve	Private 🔐	🔘	
└	_transfer	Private 🔐	🔘	
└	swapTokens	Private 🔐	🔘	lockTheSwap

└	buyBackTokens	Private 🔐	🛑	lockTheSwap
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForEth	Private 🔐	🛑	
└	swapETHForTokens	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	
└	_transferToExcluded	Private 🔐	🛑	
└	_transferFromExcluded	Private 🔐	🛑	
└	_transferBothExcluded	Private 🔐	🛑	
└	_reflectFee	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	calculateTaxFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	isExcludedFromFee	Public ❗		NO❗

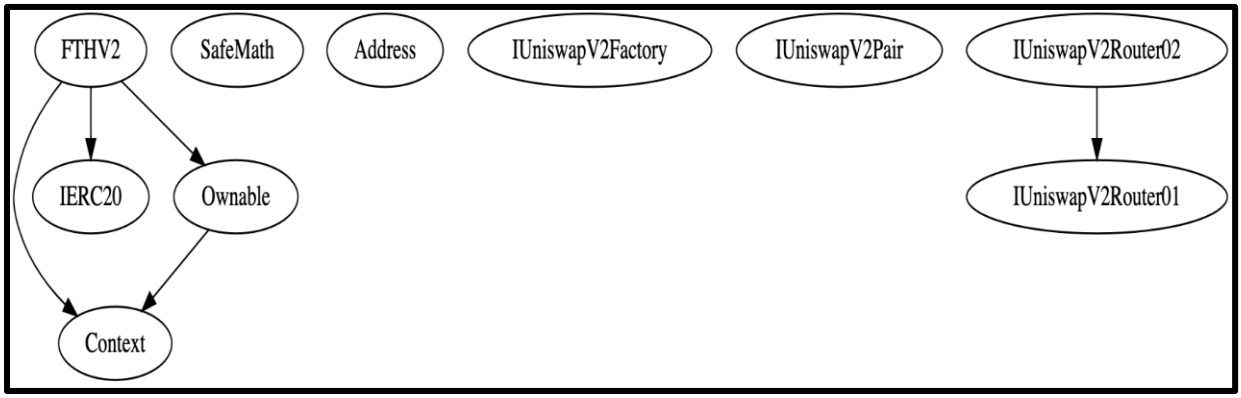
L	excludeFromFee	Public		onlyOwner
L	includeInFee	Public		onlyOwner
L	_getSellBnBAmount	Private		
L	_removeOldSellHistories	Private		
L	SetBuyBackMaxTimeForHistories	External		onlyOwner
L	SetBuyBackDivisor	External		onlyOwner
L	GetBuyBackTimeInterval	Public		NO
L	SetBuyBackTimeInterval	External		onlyOwner
L	SetBuyBackRangeRate	External		onlyOwner
L	GetSwapMinutes	Public		NO
L	SetSwapMinutes	External		onlyOwner
L	setTaxFeePercent	External		onlyOwner
L	setBuyFee	External		onlyOwner
L	setSellFee	External		onlyOwner
L	setLiquidityFeePercent	External		onlyOwner
L	setBuyBackSellLimit	External		onlyOwner
L	setMaxTxAmount	External		onlyOwner
L	setMarketingDivisor	External		onlyOwner
L	setCharityDivisor	External		onlyOwner
L	setNumTokensSellToAddToBuyBack	External		onlyOwner
L	setMarketingAddress	External		onlyOwner

L	setCharityAddress	External ❗	🛑	onlyOwner
L	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
L	setBuyBackEnabled	Public ❗	🛑	onlyOwner
L	setAutoBuyBackEnabled	Public ❗	🛑	onlyOwner
L	prepareForPreSale	External ❗	🛑	onlyOwner
L	afterPreSale	External ❗	🛑	onlyOwner
L	changeMaxTransactionAmount	Public ❗	🛑	onlyOwner
L	transferToAddressETH	Private 🔐	🛑	
L	changeRouterVersion	Public ❗	🛑	onlyOwner
L		External ❗	💵	NO❗
L	transferForeignToken	Public ❗	🛑	onlyOwner
L	Sweep	External ❗	🛑	onlyOwner
L	setAddressFee	External ❗	🛑	onlyOwner
L	setBuyAddressFee	External ❗	🛑	onlyOwner
L	setSellAddressFee	External ❗	🛑	onlyOwner

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.

Owner privileges

- The owner can renounce and transfer the ownership.

```
ftrace | funcSig
function renounceOwnership() public virtual onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = address(0);
}

ftrace | funcSig
function transferOwnership(address newOwner⬆) public virtual onlyOwner {
    require(
        newOwner⬆ != address(0),
        "Ownable: new owner is the zero address"
    );
    emit OwnershipTransferred(_owner, newOwner⬆);
    _owner = newOwner⬆;
}
```

- 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 not 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 include/exclude wallet form fee.

```
ftrace | funcSig
function excludeFromFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = true;
}

ftrace | funcSig
function includeInFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = false;
}
```

❖ The owner can change buyback max time.

```
ftrace | funcSig
function SetBuyBackMaxTimeForHistories(uint256 newMinutes⬆)
    external
    onlyOwner
{
    _buyBackMaxTimeForHistories = newMinutes⬆ * 1 minutes;
}
```

❖ The owner can change the buy back divisor.

```
ftrace | funcSig
function SetBuyBackDivisor(uint256 newDivisor⬆) external onlyOwner {
    _buyBackDivisor = newDivisor⬆;
}
```

❖ The owner can change the buy back interval.

```
ftrace | funcSig
function SetBuyBackTimeInterval(uint256 newMinutes⬆) external onlyOwner {
    _buyBackTimeInterval = newMinutes⬆ * 1 minutes;
}
```

❖ The owner can change the buy back swap rate.

```
ftrace | funcSig
function SetBuyBackRangeRate(uint256 newPercent⬆) external onlyOwner {
    require(newPercent⬆ <= 100, "The value must not be larger than 100.");
    _buyBackRangeRate = newPercent⬆;
}
```

❖ The owner can change the swap interval.

```
ftrace | funcSig
function SetSwapMinutes(uint256 newMinutes⬆) external onlyOwner {
    _intervalMinutesForSwap = newMinutes⬆ * 1 minutes;
}
```

❖ The owner can change the tax fee.

```
ftrace | funcSig
function setTaxFeePercent(uint256 taxFee⬆) external onlyOwner {
    _taxFee = taxFee⬆;
}
```

❖ The owner can change buying fees and selling fees.

```
ftrace | funcSig
function setBuyFee(uint256 buyTaxFee⬆, uint256 buyLiquidityFee⬆)
    external
    onlyOwner
{
    _buyTaxFee = buyTaxFee⬆;
    _buyLiquidityFee = buyLiquidityFee⬆;
}

ftrace | funcSig
function setSellFee(uint256 sellTaxFee⬆, uint256 sellLiquidityFee⬆)
    external
    onlyOwner
{
    _sellTaxFee = sellTaxFee⬆;
    _sellLiquidityFee = sellLiquidityFee⬆;
}
```

❖ The owner can change the liquidity fee.

```
ftrace | funcSig
function setLiquidityFeePercent(uint256 liquidityFee⬆) external onlyOwner {
    _liquidityFee = liquidityFee⬆;
}
```

❖ The owner can change the buy back sell limit.

```
ftrace | funcSig
function setBuyBackSellLimit(uint256 buyBackSellSetLimit⬆)
    external
    onlyOwner
{
    buyBackSellLimit = buyBackSellSetLimit⬆;
}
```

❖ The owner can change the max transaction amount.

```
ftrace | funcSig
function setMaxTxAmount(uint256 maxTxAmount⬆) external onlyOwner {
    _maxTxAmount = maxTxAmount⬆;
}
```

❖ The owner can change the marketing divisor.

```
ftrace | funcSig
function setMarketingDivisor(uint256 divisor⬆) external onlyOwner {
    marketingDivisor = divisor⬆;
}
```

❖ The owner can change marketing and charity addresses.

```
ftrace | funcSig
function setMarketingAddress(address _marketingAddress⬆) external onlyOwner {
    marketingAddress = payable(_marketingAddress⬆);
}

ftrace | funcSig
function setCharityAddress(address _charityAddress⬆) external onlyOwner {
    charityAddress = payable(_charityAddress⬆);
}
```

❖ The owner can enable/disable liquidity swap, buy back and auto buy back.

```
ftrace | funcSig
function setSwapAndLiquifyEnabled(bool _enabled⬆) public onlyOwner {
    swapAndLiquifyEnabled = _enabled⬆;
    emit SwapAndLiquifyEnabledUpdated(_enabled⬆);
}

ftrace | funcSig
function setBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    buyBackEnabled = _enabled⬆;
    emit BuyBackEnabledUpdated(_enabled⬆);
}

ftrace | funcSig
function setAutoBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    _isAutoBuyBack = _enabled⬆;
    emit AutoBuyBackEnabledUpdated(_enabled⬆);
}
```

❖ The owner can change the router address.

```
ftrace | funcSig
function changeRouterVersion(address _router⬆)
    public
    onlyOwner
    returns (address _pair⬆)
{
    IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(_router⬆);

    _pair⬆ = IUniswapV2Factory(_uniswapV2Router.factory()).getPair(
        address(this),
        _uniswapV2Router.WETH()
    );
    if (_pair⬆ == address(0)) {
        // Pair doesn't exist
        _pair⬆ = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(
            address(this),
            _uniswapV2Router.WETH()
        );
    }
    uniswapV2Pair = _pair⬆;

    // Set the router of the contract variables
    uniswapV2Router = _uniswapV2Router;
}
```

❖ The owner can transfer all foriegn token to any wallet.

```
ftrace | funcSig
function transferForeignToken(address _token⬆, address _to⬆)
    public
    onlyOwner
    returns (bool _sent⬆)
{
    require(_token⬆ != address(this), "Can't let you take all native token");
    uint256 _contractBalance = IERC20(_token⬆).balanceOf(address(this));
    _sent⬆ = IERC20(_token⬆).transfer(_to⬆, _contractBalance);
}
```

❖ The owner can get full contract balance to the owner's account.

```
ftrace | funcSig
function Sweep() external onlyOwner {
    uint256 balance = address(this).balance;
    payable(owner()).transfer(balance);
}
```

Audit conclusion

While conducting the audit of the Fuck Them Heggies Token smart contract, it was observed that there is nothing alarming with the code.