RugFreeCoins Audit

AlienBet Token Audit

Smart Contract Security Audit

August 24, 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	10
Total Points	10
Contract details	11
Token distribution	12
Contract code function details	13
Contract description table	14
Security issue checking status	22
Owner privileges	22
Audit conclusion	26

Audit details

Audited project

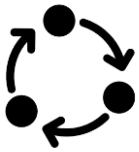
AlienBet Token

Contract Address

0x805e9c7F7BB746d71357c4Ff63C8424F80cCBE65

Client contact

AlienBet Token Team



Blockchain

Binance smart chain

Project website

https://alienbet.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 AlienBet to perform an audit of the smart contract.

https://bscscan.com/token/0x805e9c7F7BB746d71357c4Ff63C8424F80cCBE65

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

AlientBet is a token built on the Binance Smart Chain. Each transaction, purchase incur a 15% fee, and sales incur a 25% fee for 3 hours after the launch and 20% after 3 hours.

AlienBet has a Gambling Dapp on Binance Smart Chain. It is a collection of decentralized, trustless, and immutable blockchain games running directly on the Binance Smart Chain network with a wide variety of games including:

- Coin Flip
- Barbell Roll
- Roulette
- Dice It
- Treasure Chest

AlienBet's vision is to grow and reach as many players as they can with their easy to use dapp, and an incredible opportunity for passive income with $ALIEN, it will be easy to achieve all of the goals.

Features

- **8%** of each transaction when buying and equally increased value when selling gets converted to BNB and is split amongst all holders. The holders will be eligible to receive tokens everyone hour and rewards are proportional to how many tokens each individual holds.

- 2% of liquidity fee, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity. This is a key element for decentralized exchanges like Pancakeswap.

- 0 - 5% fee from buying and equally increased value when selling is allocated to buyback and burn tokens to save from massive dips in order to keep the token market price stable.

Tokenomics

15% tax when buying

- 8% of trade goes to holders pockets in BNB.
- 2% of trade goes to the liquidity pool.
- 0-5% of trade goes to buyback & burn.

During the first 3 hours after launch there will be an additional 10% selling tax. It will remain at 5% afterwards.

Roadmap


ROAD MAP
ALIENBET
PHASE 1
ALIEN Token Development
AlienBet Games Development
$ALIEN Token Deployment
AlienBet Games Deployment
Launch Alien.Bet Website & Socials
A-Ads.com Advertisments
Influencers Marketing
Promo On Coin Vote Sites
$ALIEN Token Audit
NFT Collection Launch (08/24)
Private Sale (08/25)
DxSale Presale (08/26)
Public Launch
PHASE 2
Drop NFT Collection On Airnfts
Launch Jackpot Feature
CoinMarketCap Listing
CoinGecko Listing
Blockfolio Listing
First CEX listing
BTOK Promotion
Online Banner Ads
Influencer Marketing
PHASE 3
15,000 Holders
First Video AMA & Team Doxxing
Legal Entity Formation
More Gambling Games!
AlienBet Unique NFT Marketplace
AlienBet Swap & Cross-Chain Bridge

Target market and the concept

Target market

- Anyone who's interested in playing games and earning.
- Anyone who's interested in taking part with Alienbet NFTs.
- Anyone who's interested in Crypto space with long term investment plans.
- Anyone who's ready to earn a passive income in BNB by holding tokens.
- Anyone who's interested in trading tokens.
- All BNB investors and fans out there.
- Anyone who's interested in making financial transactions with any other party using AlienBet token as the currency.

Core concept

The BNB reward system

8% of each transaction when buying and an equally increased value when selling gets converted to BNB and is split amongst all holders. The rewards are sent to holders that have at least 1000 AlienBet tokens, holders will be eligible to receive tokens everyone hour and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The buyback and burn mechanism collect 0-5% tax when buying and an equally increased value when selling, which is stored inside the contract. Whenever a buy or sell occurs, a fraction of the buyback amount is used to automatically purchase tokens from the liquidity pool. Those tokens are immediately burned after purchase, which keeps the token price stable.

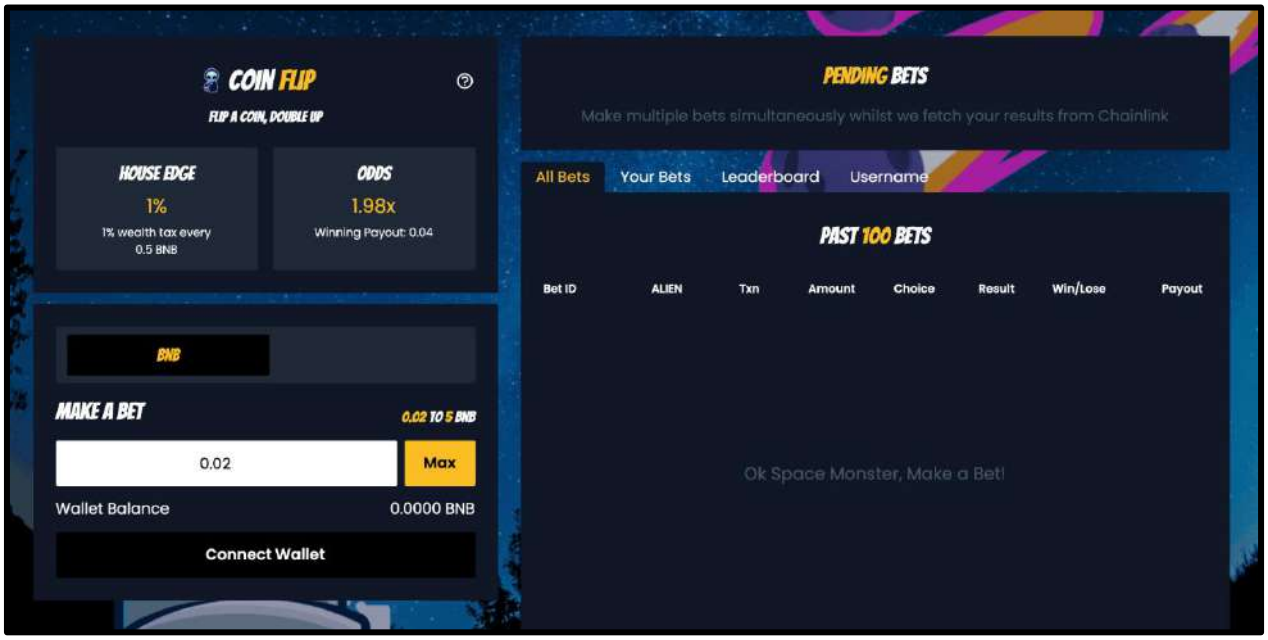
The liquidity fee of 2%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

Use case

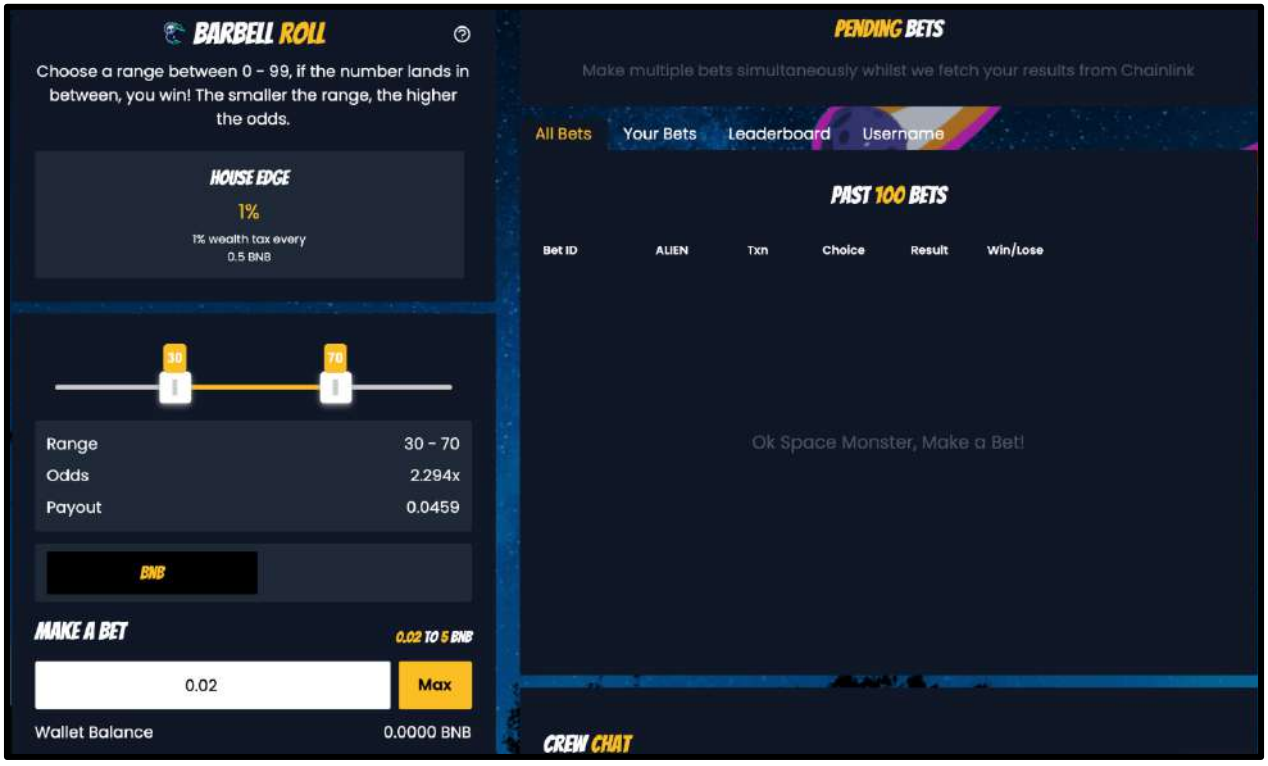
NFT collection

Dapp gambling games

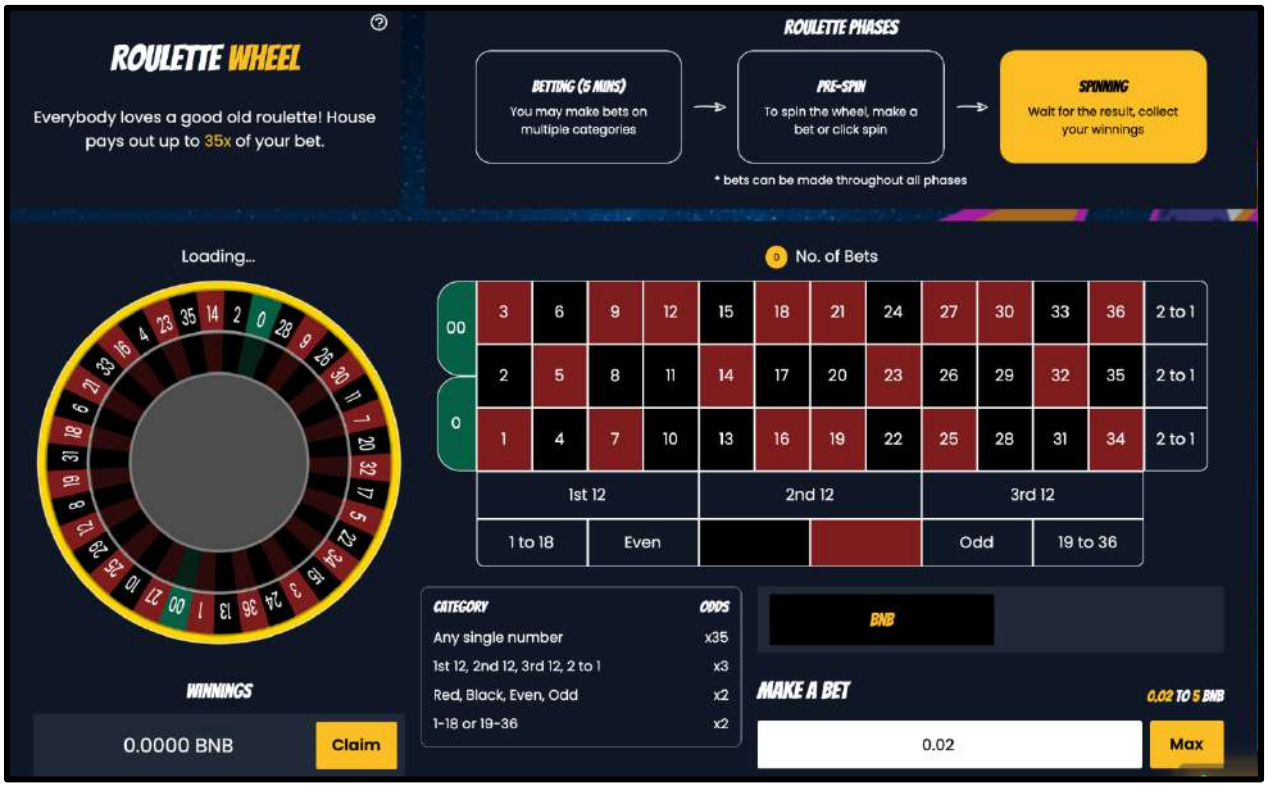
Coin flip



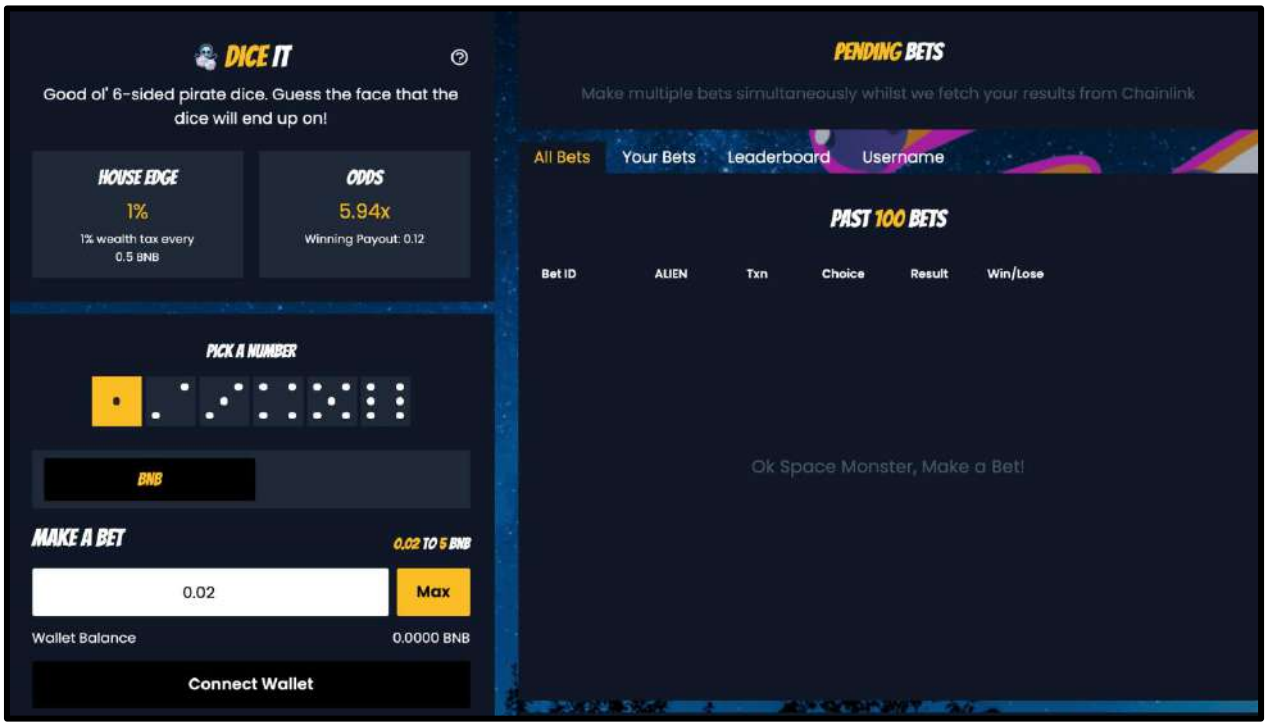
Barbell roll



Roulette



Dice IT



Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	10/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	9/10
7	Impact on the business	10/10
8	Preparing for the future	10/10
Total Points		**9.38/10**

Contract details

Token contract details for 24th August 2021

Contract name	AlienBet
Contract address	0x805e9c7F7BB746d71357c4Ff63C8424F80cCBE65
Token supply	1,000,000
Token ticker	ALIEN
Decimals	18
Token holders	1
Transaction count	1
Liquidity wallet	0xf7f32daea3162fd7cdb9bf02014288d4f2d0e6ff
Dividend Tracker	0xac919d188acecd08f8b32b67718b8ac280f6a9fc
Contract deployer address	0xf7f32dAEa3162Fd7cDB9Bf02014288D4F2d0e6fF
Contract's current owner address	0xf7f32daea3162fd7cdb9bf02014288d4f2d0e6ff

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		
L	Function Name	Visibility	Mutability	Modifiers
ALIEN	**Implementation**	**ERC20, Ownable**		
L		Public		ERC20
L		External		NO
L	updateDividendTracker	Public		onlyOwner
L	updateUniswapV2Router	Public		onlyOwner
L	excludeFromFees	Public		onlyOwner
L	excludePresaleContractFromFees	Public		onlyOwner
L	excludeMultipleAccountsFromFees	Public		onlyOwner
L	setAutomatedMarketMakerPair	Public		onlyOwner
L	blacklistAddress	External		onlyOwner
L	_setAutomatedMarketMakerPair	Private		
L	excludeFromDividends	External		onlyOwner
L	isExcludedFromDividends	External		NO
L	updateLiquidityWallet	Public		onlyOwner
L	updateGasForProcessing	Public		onlyOwner
L	updateClaimWait	External		onlyOwner
L	getClaimWait	External		NO
L	getTotalDividendsDistributed	External		NO
L	isExcludedFromFees	Public		NO
L	isExcludedPresaleContractFromFees	Public		NO
L	withdrawableDividendOf	Public		NO

L	dividendTokenBalanceOf	Public ❗		NO❗
L	getAccountDividendsInfo	External ❗		NO❗
L	getAccountDividendsInfoAtIndex	External ❗		NO❗
L	processDividendTracker	External ❗	🛑	NO❗
L	claim	External ❗	🛑	NO❗
L	setTradingEnabled	External ❗	🛑	onlyOwner
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfDividendTokenHolders	External ❗		NO❗
L	setBuyBackFee	External ❗	🛑	onlyOwner
L	setBNBRewardsFee	External ❗	🛑	onlyOwner
L	setRewardsPool	External ❗	🛑	onlyOwner
L	setLiquidityFee	External ❗	🛑	onlyOwner
L	setBuy2RewThreshold	External ❗	🛑	onlyOwner
L	setRewardsPoolFee	External ❗	🛑	onlyOwner
L	setMaxSellTxAmount	External ❗	🛑	onlyOwner
L	setSwapEnabled	External ❗	🛑	onlyOwner
L	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
L	_transfer	Internal 🔒	🛑	
L	swapBuyBackTokens	Private 🔐	🛑	lockTheSwap
L	buyBackTokens	Private 🔐	🛑	lockTheSwap
L	swapETHForTokens	Private 🔐	🛑	
L	swapAndLiquify	Private 🔐	🛑	lockTheSwap
L	swapTokensForEth	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	setBuyBackEnabled	Public ❗	🛑	onlyOwner
L	setBuybackUpperLimit	External ❗	🛑	onlyOwner
L	swapAndSendToReward	Private 🔐	🛑	lockTheSwap

L	manualSendFromBuybackToRewardsPool	External ❗	🛑	onlyOwner
ALIENDividendTracker	**Implementation**	**DividendPayingToken, Ownable**		
L		Public ❗	🛑	DividendPayingToken
L	_transfer	Internal 🔒	🛑	
L	withdrawDividend	Public ❗	🛑	NO❗
L	excludeFromDividends	External ❗	🛑	onlyOwner
L	updateClaimWait	External ❗	🛑	onlyOwner
L	isExcludedFromDividends	Public ❗		NO❗
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfTokenHolders	External ❗		NO❗
L	getAccount	Public ❗		NO❗
L	getAccountAtIndex	Public ❗		NO❗
L	canAutoClaim	Private 🔐		
L	setBalance	External ❗	🛑	onlyOwner
L	process	Public ❗	🛑	NO❗
L	processAccount	Public ❗	🛑	onlyOwner
Context	**Implementation**			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
DividendPayingToken	**Implementation**	**ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface**		
L		Public ❗	🛑	ERC20
L		External ❗	💸	NO❗
L	distributeDividends	Public ❗	💸	NO❗

└	withdrawDividend	Public ❗	🛑	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	
DividendPayingTokenInterface	**Interface**			
└	dividendOf	External ❗		NO❗
└	distributeDividends	External ❗	💸	NO❗
└	withdrawDividend	External ❗	🛑	NO❗
DividendPayingTokenOptionalInterface	**Interface**			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
ERC20	**Implementation**	**Context, IERC20, IERC20Metadata**		
└		Public ❗	🛑	NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	decimals	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗

└	transfer	Public	●	NO
└	allowance	Public		NO
└	approve	Public	●	NO
└	transferFrom	Public	●	NO
└	increaseAllowance	Public	●	NO
└	decreaseAllowance	Public	●	NO
└	_transfer	Internal	●	
└	_mint	Internal	●	
└	_burn	Internal	●	
└	_approve	Internal	●	
└	_beforeTokenTransfer	Internal	●	
IERC20	**Interface**			
└	totalSupply	External		NO
└	balanceOf	External		NO
└	transfer	External	●	NO
└	allowance	External		NO
└	approve	External	●	NO
└	transferFrom	External	●	NO
IERC20Metadata	**Interface**	**IERC20**		
└	name	External		NO
└	symbol	External		NO
└	decimals	External		NO
IterableMapping	**Library**			
└	get	Public		NO
└	getIndexOfKey	Public		NO
└	getKeyAtIndex	Public		NO

└	size	Public ❗		NO❗
└	set	Public ❗	🔘	NO❗
└	remove	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❗
IUniswapV2Router01	**Interface**			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🔘	NO❗
└	addLiquidityETH	External ❗	💳	NO❗
└	removeLiquidity	External ❗	🔘	NO❗
└	removeLiquidityETH	External ❗	🔘	NO❗
└	removeLiquidityWithPermit	External ❗	🔘	NO❗
└	removeLiquidityETHWithPermit	External ❗	🔘	NO❗
└	swapExactTokensForTokens	External ❗	🔘	NO❗
└	swapTokensForExactTokens	External ❗	🔘	NO❗
└	swapExactETHForTokens	External ❗	💳	NO❗
└	swapTokensForExactETH	External ❗	🔘	NO❗
└	swapExactTokensForETH	External ❗	🔘	NO❗

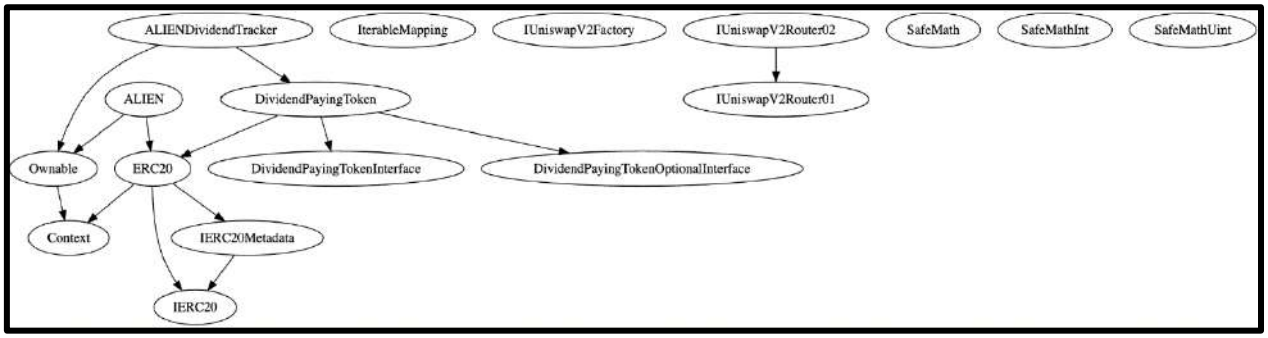
└	swapETHForExact Tokens	External ❗	💸	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	**Interface**	**IUniswapV2Router01**		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💸	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
SafeMath	**Library**			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		

└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
SafeMathInt	**Library**			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	**Library**			
└	toInt256Safe	Internal 🔒		

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 update the dividend tracker.

```
ftrace | funcSig
function updateDividendTracker(address newAddress⬆) public onlyOwner {
    require(newAddress⬆ != address(dividendTracker), "ALIEN: The dividend tracker already has that address");

    ALIENDividendTracker newDividendTracker = ALIENDividendTracker(payable(newAddress⬆));

    require(newDividendTracker.owner() == address(this), "ALIEN: The new dividend tracker must be owned by the ALIEN token contract");

    newDividendTracker.excludeFromDividends(address(newDividendTracker));
    newDividendTracker.excludeFromDividends(address(this));
    newDividendTracker.excludeFromDividends(owner());
    newDividendTracker.excludeFromDividends(address(uniswapV2Router));

    emit UpdateDividendTracker(newAddress⬆, address(dividendTracker));

    dividendTracker = newDividendTracker;
}
```

- The owner can update the v2 router address.

```
ftrace | funcSig
function updateUniswapV2Router(address newAddress⬆) public onlyOwner {
    require(newAddress⬆ != address(uniswapV2Router), "ALIEN: The router already has that address");
    emit UpdateUniswapV2Router(newAddress⬆, address(uniswapV2Router));
    uniswapV2Router = IUniswapV2Router02(newAddress⬆);
}
```

❖ The owner can exclude wallets and pre-sale contracts from fees.

```
ftrace | funcSig
function excludeFromFees(address account⬆, bool excluded⬆) public onlyOwner {
    require(_isExcludedFromFees[account⬆] != excluded⬆, "ALIEN: Account is already the value of 'excluded'");
    _isExcludedFromFees[account⬆] = excluded⬆;

    emit ExcludeFromFees(account⬆, excluded⬆);
}

ftrace | funcSig
function excludePresaleContractFromFees(address account⬆, bool excluded⬆) public onlyOwner {
    require(_isExcludedPresaleContract[account⬆] != excluded⬆, "ALIEN: Account is already the value of 'excluded'");
    _isExcludedPresaleContract[account⬆] = excluded⬆;

    emit ExcludeFromFees(account⬆, excluded⬆);
}
```

❖ The owner can exclude multiple accounts from fees.

```
ftrace | funcSig
function excludeMultipleAccountsFromFees(address[] calldata accounts⬆, bool excluded⬆) public onlyOwner {
    for(uint256 i = 0; i < accounts⬆.length; i++) {
        _isExcludedFromFees[accounts⬆[i]] = excluded⬆;
    }

    emit ExcludeMultipleAccountsFromFees(accounts⬆, excluded⬆);
}
```

❖ The owner can blacklist wallets.

```
ftrace | funcSig
function blacklistAddress(address account⬆, bool value⬆) external onlyOwner{
    _isBlacklisted[account⬆] = value⬆;
}
```

❖ The owner can exclude wallets from dividend.

```
ftrace | funcSig
function excludeFromDividends(address account⬆) external onlyOwner{
    dividendTracker.excludeFromDividends(account⬆);
}
```

❖ The owner can update the liquidity wallet.

```
ftrace | funcSig
function updateLiquidityWallet(address newLiquidityWallet⬆) public onlyOwner {
    require(newLiquidityWallet⬆ != liquidityWallet, "ALIEN: The liquidity wallet is already this address");
    excludeFromFees(newLiquidityWallet⬆, true);
    emit LiquidityWalletUpdated(newLiquidityWallet⬆, liquidityWallet);
    liquidityWallet = newLiquidityWallet⬆;
}
```

❖ The owner can update the gas fee for the processing.

```
ftrace | funcSig
function updateGasForProcessing(uint256 newValue⬆) public onlyOwner {
    require(newValue⬆ >= 200000 && newValue⬆ <= 500000, "ALIEN: gasForProcessing must be between 200,000 and 500,000");
    require(newValue⬆ != gasForProcessing, "ALIEN: Cannot update gasForProcessing to same value");
    emit GasForProcessingUpdated(newValue⬆, gasForProcessing);
    gasForProcessing = newValue⬆;
}
```

❖ The owner can change the claim wait.

```
ftrace | funcSig
function updateClaimWait(uint256 claimWait⬆) external onlyOwner {
    dividendTracker.updateClaimWait(claimWait⬆);
}
```

❖ The owner can change buyback,reward and liquidity fee.

```
ftrace | funcSig
function setBuyBackFee(uint256 value⬆) external onlyOwner {
    buyBackFee = value⬆;
    totalFees = BNBRewardsFee.add(liquidityFee).add(buyBackFee).add(
        rewardsPoolFee
    );
}

ftrace | funcSig
function setBNBRewardsFee(uint256 value⬆) external onlyOwner {
    BNBRewardsFee = value⬆;
    totalFees = BNBRewardsFee.add(liquidityFee).add(buyBackFee).add(
        rewardsPoolFee
    );
}

ftrace | funcSig
function setLiquidityFee(uint256 value⬆) external onlyOwner {
    liquidityFee = value⬆;
    totalFees = BNBRewardsFee.add(liquidityFee).add(buyBackFee).add(
        rewardsPoolFee
    );
}
```

❖ The owner can change the reward pool address.

```
ftrace | funcSig
function setRewardsPool(address payable rewardsPoolAddress⬆)
    external
    onlyOwner
{
    rewardsPool = rewardsPoolAddress⬆;
}
```

❖ The owner can change the reward pool fee.

```
ftrace | funcSig
function setRewardsPoolFee(uint256 value⬆) external onlyOwner {
    rewardsPoolFee = value⬆;
    totalFees = BNBRewardsFee.add(liquidityFee).add(buyBackFee).add(
        rewardsPoolFee
    );
}
```

❖ The owner can change the max sell transaction amount.

```
ftrace | funcSig
function setMaxSellTxAmount(uint256 amount⬆) external onlyOwner {
    maxSellTransactionAmount = amount⬆ * 10**18;
}
```

❖ The owner can enable/disable swap.

```
ftrace | funcSig
function setSwapEnabled(bool value⬆) external onlyOwner {
    swapEnabled = value⬆;
}
```

❖ The owner can change token swap point.

```
ftrace | funcSig
function setSwapTokensAtAmount(uint256 amount⬆) external onlyOwner {
    swapTokensAtAmount = amount⬆ * 10**18;
}
```

❖ The owner can enable/disable buy back.

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

❖ The owner can change the buyback upper limit.

Audit conclusion

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