BlockSAFU

ADVANCE MANUAL
SMART CONTRACT AUDIT

Project: OpenAI BNB

Website: https://www.theweb3project.com/

PASSED

BlockSAFU Score:

82

Contract Address:

0xaD68F9c74Ee7376f819E8B1d1f17FCDa734C2389

DISCLAMER

BlockSAFU has completed this report to provide a summary of the Smart Contract functions, and any security, dependency, or cybersecurity vulnerabilities. This is often a constrained report on our discoveries based on our investigation and understanding of the current programming versions as of this report's date. To understand the full scope of our analysis, it is vital for you to at the date of this report. To understand the full scope of our analysis, you need to review the complete report. Although we have done our best in conducting our investigation and creating this report, it is vital to note that you should not depend on this report and cannot make any claim against BlockSAFU or its Subsidiaries and Team members on the premise of what has or has not been included in the report. Please remember to conduct your independent examinations before making any investment choices. We do not provide investment advice or in any way claim to determine if the project will be successful or not.

By perusing this report or any portion of it, you concur to the terms of this disclaimer. In the unlikely situation where you do not concur with the terms, you should immediately terminate reading this report, and erase and discard any duplicates of this report downloaded and/or printed by you. This report is given for data purposes as it were and on a non-reliance premise and does not constitute speculation counsel. No one should have any right to depend on the report or its substance, and BlockSAFU and its members (including holding companies, shareholders, backups, representatives, chiefs, officers, and other agents) BlockSAFU and its subsidiaries owe no obligation of care towards you or any other person, nor does BlockSAFU make any guarantee or representation to any individual on the precision or completeness of the report.

ABOUT THE AUDITOR:

BlockSAFU (BSAFU) is an Anti-Scam Token Utility that reviews Smart Contracts and Token information to Identify Rug Pull and Honey Pot scamming activity. BlockSAFUs Development Team consists of several Smart Contract creators, Auditors Developers, and Blockchain experts. BlockSAFU provides solutions, prevents, and hunts down scammers. BSAFU is a utility token with features Audit, KYC, Token Generators, and Bounty Scammers. It will enrich the crypto ecosystem.

OVERVIEW

Mint Function

- No mint functions.

Fees

- Buy 5% (owner can’t set fees up to 5%).
- Sell 5% (owner can’t set fees up to 5%).

Tx Amount

- Owner cannot set a max tx amount.

Transfer Pausable

- Owner can’t pause.

Blacklist

- Owner can’t blacklist.

Ownership

- Owner can’t take back ownership.

Proxy

- This contract has no proxy.

Anti Whale

- Owner can’t limit the number of wallet holdings.

Trading Cooldown

- Owner can’t set the selling time interval.

SMART CONTRACT REVIEW

Token Name	**OpenAI BNB**
Token Symbol	**OPENAI**
Token Decimal	18
Total Supply	1,000,000 OPENAI
Contract Address	0xf64805fD6398C00ec463b11F234730363E1104cA
Deployer Address	0xCC7D2786Bdad294730aA8eC622e623d20D51fB55
Owner Address	0xcc7d2786bdad294730aa8ec622e623d20d51fb55
Tax Fees Buy	5%
Tax Fees Sell	5%
Gas Used for Buy	Will be updated after listing on dex
Gas Used for Sell	Will be updated after listing on dex
Contract Created	Dec-24-2022 12:29:06 PM +UTC
Unlocked Date	Will be updated after listing on dex
Verified CA	Yes
Compiler	v0.8.17+commit.8df45f5f
Optimization	No with 200 runs
Sol License	SPDX-License-Identifier: MIT
Other	default evmVersion

TAX

BUY	0%	address	**SELL**	0%
chain Defelopment	1%	**0xec5a428976821bcff6777457987830c34fb33826**	**chain Defelopment**	1%
Marketing	2%	**0xcc7d2786bdad294730aa8ec622e623d20d51fb55**	**Marketing**	2%

Token Holder

Rank	Address	Quantity	Percentage	Analytics
1	0xcc7d2786bdad294730aa8ec622e623d20d51fb55	1,000,000	100.0000%	

Team Review

The OpenAI BNB team has a nice website, their website is professionally built and the Smart contract is well developed, their social media is growing with over 7,867 people in their telegram group (count in audit date).

Official Website And Social Media

Website: https://openaibnb.com

Telegram Group: https:://t.me/openaibnb_chat

Twitter: https://twitter.com/openaibsc

MANUAL CODE REVIEW

Minor-risk

1 minor-risk code issue found

Could be fixed, and will not bring problems.

1. The return value of an external transfer/transferFrom return value is checked. Recommendation: use SafeERC20, or ensure that the transfer/transferFrom return value is checked

```
function transferFrom(
      address sender,
      address recipient,
      uint256 amount
   ) external returns (bool);
```

Medium-risk

0 medium-risk code issues found

Should be fixed, could bring problems.

High-Risk

0 high-risk code issues found

Must be fixed, and will bring problem.

Critical-Risk

0 critical-risk code issues found

Must be fixed, and will bring problem.

EXTRA NOTES SMART CONTRACT

1. TWEP

```
contract OpenAIBNB is ERC20, Ownable {
    using Address for address payable;

    IUniswapV2Router02 public uniswapV2Router;
    address public  uniswapV2Pair;

    mapping (address => bool) private
_isExcludedFromFees;

    uint256 public  marketingFeeOnBuy;
    uint256 public  marketingFeeOnSell;

    uint256 public  chainDevelopmentFeeOnBuy;
    uint256 public  chainDevelopmentFeeOnSell;

    uint256 private _totalFeesOnBuy;
    uint256 private _totalFeesOnSell;

    uint256 public maxFee;

    address public  marketingWallet;
    address public  chainDevelopmentWallet;

    uint256 public  swapTokensAtAmount;
    bool    private swapping;

    event ExcludeFromFees(address indexed account, bool
isExcluded);
    event MarketingWalletChanged(address
marketingWallet);
    event ChainDevelopmentWalletChanged(address
chainDevelopmentWallet);
    event UpdateBuyFees(uint256 marketingFeeOnBuy,
```

```
uint256 chainDevelopmentFeeOnBuy);
    event UpdateSellFees(uint256 marketingFeeOnSell,
uint256 chainDevelopmentFeeOnSell);
    event SwapAndLiquify(uint256 tokensSwapped,uint256
bnbReceived,uint256 tokensIntoLiqudity);
    event SwapAndSendMarketing(uint256 tokensSwapped,
uint256 bnbSend);
    event SwapAndSendChainDevelopment(uint256
tokensSwapped, uint256 bnbSend);
    event SwapTokensAtAmountUpdated(uint256
swapTokensAtAmount);

    constructor () ERC20("OpenAI BNB", "OPENAI")
    {
        address router;
        if (block.chainid == 56) {
            router =
0x10ED43C718714eb63d5aA57B78B54704E256024E; // BSC
Pancake Mainnet Router
        } else {
            revert();
        }

        IUniswapV2Router02 _uniswapV2Router =
IUniswapV2Router02(router);
        address _uniswapV2Pair =
IUniswapV2Factory(_uniswapV2Router.factory())
            .createPair(address(this),
_uniswapV2Router.WETH());

        uniswapV2Router = _uniswapV2Router;
        uniswapV2Pair   = _uniswapV2Pair;

        _approve(address(this), address(uniswapV2Router),
type(uint256).max);
```

```
        marketingFeeOnBuy  = 2;
        marketingFeeOnSell = 2;

        chainDevelopmentFeeOnBuy  = 1;
        chainDevelopmentFeeOnSell = 1;

        maxFee             = 5;

        _totalFeesOnBuy    = marketingFeeOnBuy +
chainDevelopmentFeeOnBuy;
        _totalFeesOnSell   =  marketingFeeOnSell +
chainDevelopmentFeeOnSell;

        marketingWallet =
0xCC7D2786Bdad294730aA8eC622e623d20D51fB55;
        chainDevelopmentWallet =
0xEc5A428976821BcFf6777457987830c34FB33826;

        _isExcludedFromFees[owner()] = true;
        _isExcludedFromFees[address(0xdead)] = true;
        _isExcludedFromFees[address(this)] = true;
        _isExcludedFromFees[marketingWallet] = true;
        _isExcludedFromFees[chainDevelopmentWallet] =
true;

        _mint(owner(), 1_000_000 * (10 ** decimals()));
        swapTokensAtAmount = totalSupply() / 5_000;
    }

    receive() external payable {

     }

    function claimStuckTokens(address token) external
onlyOwner {
        require(token != address(this), "Owner cannot
```

```
claim contract's balance of its own tokens");
        if (token == address(0x0)) {

payable(msg.sender).sendValue(address(this).balance);
            return;
        }
        IERC20 ERC20token = IERC20(token);
        uint256 balance =
ERC20token.balanceOf(address(this));
        ERC20token.transfer(msg.sender, balance);
    }

    // FEE SYSTEM

    function excludeFromFees(address account, bool
excluded) external onlyOwner{
        require(_isExcludedFromFees[account] !=
excluded,"Account is already the value of 'excluded'");
        _isExcludedFromFees[account] = excluded;

        emit ExcludeFromFees(account, excluded);
    }

    function isExcludedFromFees(address account) public
view returns(bool) {
        return _isExcludedFromFees[account];
    }

    function updateBuyFees(uint256 _marketingFeeOnBuy,
uint256 _chainDevelopmentFeeOnBuy) external onlyOwner {
        marketingFeeOnBuy = _marketingFeeOnBuy;
        chainDevelopmentFeeOnBuy =
_chainDevelopmentFeeOnBuy;

        _totalFeesOnBuy   = marketingFeeOnBuy +
chainDevelopmentFeeOnBuy;
```

```
        require(_totalFeesOnBuy + _totalFeesOnSell <=
maxFee, "Total Fees cannot exceed the maximum");

        emit UpdateBuyFees(marketingFeeOnBuy,
chainDevelopmentFeeOnBuy);
    }

    function updateSellFees(uint256 _marketingFeeOnSell,
uint256 _chainDevelopmentFeeOnSell) external onlyOwner {
        marketingFeeOnSell = _marketingFeeOnSell;
        chainDevelopmentFeeOnSell =
_chainDevelopmentFeeOnSell;

        _totalFeesOnSell   = marketingFeeOnSell +
chainDevelopmentFeeOnSell;

        require(_totalFeesOnBuy + _totalFeesOnSell <=
maxFee, "Total Fees cannot exceed the maximum");

        emit UpdateSellFees(marketingFeeOnSell,
chainDevelopmentFeeOnSell);
    }

    function changeMarketingWallet(address
_marketingWallet) external onlyOwner{
        require(_marketingWallet !=
marketingWallet,"Marketing wallet is already that
address");
        require(_marketingWallet != address(0),"Marketing
wallet cannot be the zero address");
        marketingWallet = _marketingWallet;

        emit MarketingWalletChanged(marketingWallet);
    }
```

```
    function changeChainDevelopmentWallet(address
_chainDevelopmentWallet) external onlyOwner{
        require(_chainDevelopmentWallet !=
chainDevelopmentWallet,"Marketing wallet is already that
address");
        require(_chainDevelopmentWallet !=
address(0),"Marketing wallet cannot be the zero
address");
        chainDevelopmentWallet = _chainDevelopmentWallet;

        emit
ChainDevelopmentWalletChanged(chainDevelopmentWallet);
    }

    // TRANSFER SYSTEM

    function _transfer(address from,address to,uint256
amount) internal  override {
        require(from != address(0), "ERC20: transfer from
the zero address");
        require(to != address(0), "ERC20: transfer to the
zero address");

        if (amount == 0) {
            super._transfer(from, to, 0);
            return;
        }

         uint256 contractTokenBalance =
balanceOf(address(this));

        bool canSwap = contractTokenBalance >=
swapTokensAtAmount;

        if (canSwap &&
            !swapping &&
```

```
            to == uniswapV2Pair &&
            _totalFeesOnBuy + _totalFeesOnSell > 0
        ) {
            swapping = true;

            uint256 totalFee = _totalFeesOnBuy +
_totalFeesOnSell;

            uint256 marketingShare = marketingFeeOnBuy +
marketingFeeOnSell;
            uint256 chainDevelopmentShare =
chainDevelopmentFeeOnBuy + chainDevelopmentFeeOnSell;

            if (marketingShare > 0) {
                uint256 marketingTokens =
contractTokenBalance * marketingShare / totalFee;
                swapAndSendMarketing(marketingTokens);
            }

            if (chainDevelopmentShare > 0) {
                uint256 chainDevelopmentTokens =
contractTokenBalance * chainDevelopmentShare / totalFee;

swapAndSendChainDevelopment(chainDevelopmentTokens);
            }

            swapping = false;
        }

        uint256 _totalFees;
        if (_isExcludedFromFees[from] ||
_isExcludedFromFees[to] || swapping) {
            _totalFees = 0;
        } else if (from == uniswapV2Pair) {
            _totalFees = _totalFeesOnBuy;
        } else if (to == uniswapV2Pair) {
```

```
            _totalFees = _totalFeesOnSell;
        } else {
            _totalFees = 0;
        }

        if (_totalFees > 0) {
            uint256 fees = (amount * _totalFees) / 100;
            amount = amount - fees;
            super._transfer(from, address(this), fees);
        }

        super._transfer(from, to, amount);
    }

    // SWAP SYSTEM

    function setSwapTokensAtAmount(uint256 newAmount)
external onlyOwner{
        require(newAmount > totalSupply() / 1_000_000,
"SwapTokensAtAmount must be greater than 0.0001% of total
supply");
        swapTokensAtAmount = newAmount;

        emit
SwapTokensAtAmountUpdated(swapTokensAtAmount);
    }

    function swapAndSendMarketing(uint256 tokenAmount)
private {
        uint256 initialBalance = address(this).balance;

        address[] memory path = new address[](2);
        path[0] = address(this);
        path[1] = uniswapV2Router.WETH();
```

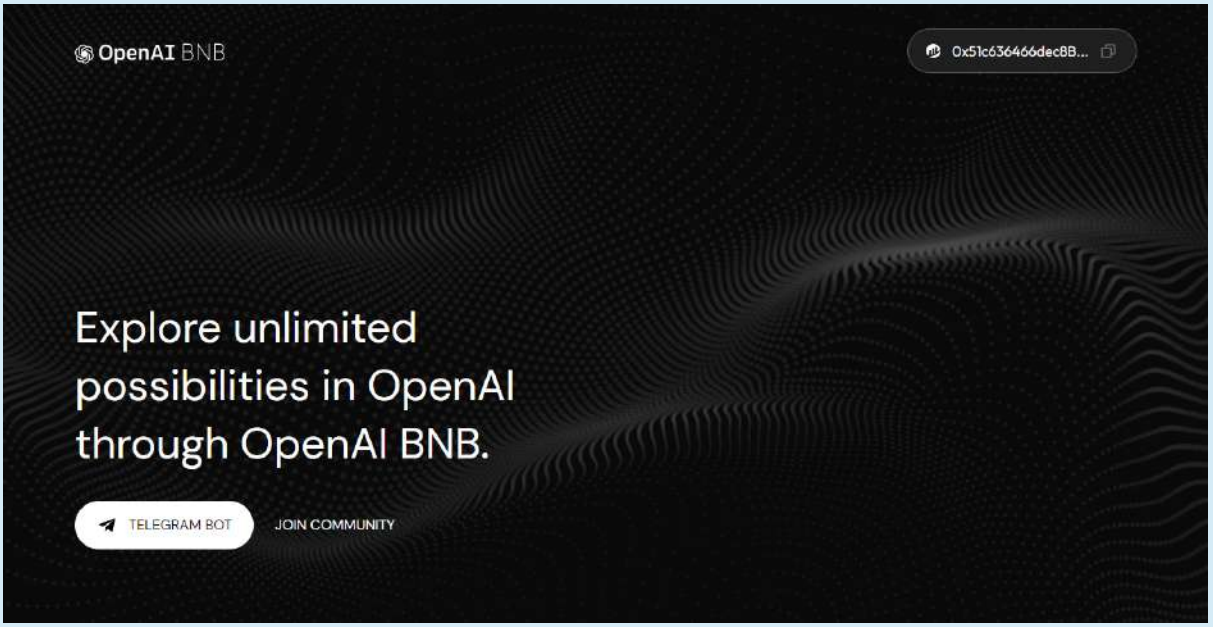
```
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0,
            path,
            address(this),
            block.timestamp);

        uint256 newBalance = address(this).balance - initialBalance;

        payable(marketingWallet).sendValue(newBalance);

        emit SwapAndSendMarketing(tokenAmount, newBalance);
    }

    function swapAndSendChainDevelopment(uint256 tokenAmount) private {
        uint256 initialBalance = address(this).balance;

        address[] memory path = new address[](2);
        path[0] = address(this);
        path[1] = uniswapV2Router.WETH();


uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0,
            path,
            address(this),
            block.timestamp);

        uint256 newBalance = address(this).balance - initialBalance;
```

```
payable(chainDevelopmentWallet).sendValue(newBalance);

        emit SwapAndSendChainDevelopment(tokenAmount,
newBalance);
    }
}
```

2. **The owner can set whitelist BSF52**

```
    function excludeFromFees(address account, bool excluded)
external onlyOwner{
        require(_isExcludedFromFees[account] != excluded,"Account
is already the value of 'excluded'");
        _isExcludedFromFees[account] = excluded;

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

WEBSITE REVIEW



- **Mobile Friendly**
- **Contains no code error**
- **SSL Secured (By Let's Encrypt SSL)**

Domain .com - Tracked by whois

First Contentful Paint:	264ms
Fully Loaded Time	879ms
Performance	90%
Accessibility	88%
Best Practices	100%
SEO	100%

RUG-PULL REVIEW

Based on the available information analyzed by us, we come to the following conclusions:

- Locked Liquidity (Locked by pinksale)

  will be updated after listing dex

- TOP 5 Holder.

  will be updated after listing dex

- The Team is not KYC

HONEYPOT REVIEW

- Ability to sell.
- The owner is not able to pause the contract.
- The owner can’t set fees

Note: Please check the disclaimer above and note that the audit makes no statements or warranties on the business model, investment attractiveness, or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the project owner.