BlockSAFU

ADVANCE MANUAL
SMART CONTRACT AUDIT

Project: Poodlefootball

Website: http://poodlefootball.io/

BlockSAFU Score:

82

Contract Address:

0xD79FfE4543cCB9807AB19Ad85d6480C948807Cfb

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

Tx Amount

- Owner cannot set a max tx amount.

Transfer Pausable

- Owner cannot pause.

Blacklist

- Owner cannot set a blacklist.

Ownership

- Owner cannot take back ownership.

Proxy

- This contract has no proxy.

Anti Whale

- Owner cannot limit the number of wallet holdings.

Trading Cooldown

- Owner cannot set the selling time interval.

SMART CONTRACT REVIEW

Token Name	**Poodle Football Inu**
Token Symbol	**PFI**
Token Decimal	18
Total Supply	100,000,000,000,000 **PFI**
Contract Address	0xD79FfE4543cCB9807AB19Ad85d6480C948807Cfb
Deployer Address	0x82d2806a687da372a71f93859552F82E346121B0
Owner Address	0x82d2806a687da372a71f93859552f82e346121b0
Tax Fees Buy	0%
Tax Fees Sell	0%
Gas Used for Buy	*will be updated after the DEX listing*
Gas Used for Sell	*will be updated after the DEX listing*
Contract Created	Aug-29-2022 01:32:25 PM +UTC
Initial Liquidity	*will be updated after the DEX listing*
Liquidity Status	Locked
Unlocked Date	*will be updated after the DEX listing*
Verified CA	Yes
Compiler	v0.6.12+commit.27d51765
Optimization	Yes with 200 runs
Sol License	MIT License
Top 5 Holders	*will be updated after the DEX listing*
Other	default evmVersion

TAX

BUY	5%	Address	**SELL**	5%
Tax fee	1%	Reflection	Tax Fee	1%
Advertisement Fee	2%	0x82d2806a687da372a71f93859552F82E346121B0	Advertisement Fee	2%
Burn Fee	2%	Burn	Burn Fee	2%

Token Holder

Rank	Address	Quantity	Percentage	Analytics
1	0x82d2606a687da372a71f93859552f82e346121b0	100,000,000,000,000	100.0000%	

[Download CSV Export]

Team Review

The PFI team has a nice website, their website is professionally built and the Smart contract is well developed, their social media is growing with over 10 people in their telegram group (count in audit date).

Official Website And Social Media

Website: http://poodlefootball.io/

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

Twitter: https://twitter.com/poodle_football

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. IERC20

```
interface IERC20 {

    function totalSupply() external view returns (uint256);

    /**
     * @dev Returns the amount of tokens owned by `account`.
     */
    function balanceOf(address account) external view returns
(uint256);

    /**
     * @dev Moves `amount` tokens from the caller's account to
`recipient`.
     *
     * Returns a boolean value indicating whether the operation
succeeded.
     *
     * Emits a {Transfer} event.
     */
    function transfer(address recipient, uint256 amount) external
returns (bool);

    /**
     * @dev Returns the remaining number of tokens that `spender`
will be
     * allowed to spend on behalf of `owner` through
{transferFrom}. This is
     * zero by default.
     *
     * This value changes when {approve} or {transferFrom} are
called.
     */
    function allowance(address owner, address spender) external
view returns (uint256);

    /**
     * @dev Sets `amount` as the allowance of `spender` over the
caller's tokens.
     *
```

```
     * Returns a boolean value indicating whether the operation
succeeded.
     *
     * IMPORTANT: Beware that changing an allowance with this
method brings the risk
     * that someone may use both the old and the new allowance by
unfortunate
     * transaction ordering. One possible solution to mitigate this
race
     * condition is to first reduce the spender's allowance to 0
and set the
     * desired value afterwards:
     *
https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
     *
     * Emits an {Approval} event.
     */
    function approve(address spender, uint256 amount) external
returns (bool);

    /**
     * @dev Moves `amount` tokens from `sender` to `recipient`
using the
     * allowance mechanism. `amount` is then deducted from the
caller's
     * allowance.
     *
     * Returns a boolean value indicating whether the operation
succeeded.
     *
     * Emits a {Transfer} event.
     */
    function transferFrom(address sender, address recipient,
uint256 amount) external returns (bool);

    /**
     * @dev Emitted when `value` tokens are moved from one account
(`from`) to
     * another (`to`).
     *
     * Note that `value` may be zero.
     */
```

```
    event Transfer(address indexed from, address indexed to,
uint256 value);

    /**
     * @dev Emitted when the allowance of a `spender` for an
`owner` is set by
     * a call to {approve}. `value` is the new allowance.
     */
    event Approval(address indexed owner, address indexed spender,
uint256 value);
}
```

IERC20 Normal Base Template

2. Safemath

```
library SafeMath {
    /**
     * @dev Returns the addition of two unsigned integers,
reverting on
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
     * Requirements:
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure returns
(uint256) {
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow");

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned integers,
reverting on
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure returns
(uint256) {
        return sub(a, b, "SafeMath: subtraction overflow");
    }

    /**
     * @dev Returns the subtraction of two unsigned integers,
reverting with custom message on
     * overflow (when the result is negative).
     *
```

```
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b, string memory errorMessage)
internal pure returns (uint256) {
        require(b <= a, errorMessage);
        uint256 c = a - b;

        return c;
    }

    /**
     * @dev Returns the multiplication of two unsigned integers,
reverting on
     * overflow.
     *
     * Counterpart to Solidity's `*` operator.
     *
     * Requirements:
     *
     * - Multiplication cannot overflow.
     */
    function mul(uint256 a, uint256 b) internal pure returns
(uint256) {
        // Gas optimization: this is cheaper than requiring 'a'
not being zero, but the
        // benefit is lost if 'b' is also tested.
        // See:
https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplication overflow");

        return c;
    }
```

```
    /**
     * @dev Returns the integer division of two unsigned integers.
Reverts on
     * division by zero. The result is rounded towards zero.
     *
     * Counterpart to Solidity's `/` operator. Note: this function
uses a
     * `revert` opcode (which leaves remaining gas untouched)
while Solidity
     * uses an invalid opcode to revert (consuming all remaining
gas).
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b) internal pure returns
(uint256) {
        return div(a, b, "SafeMath: division by zero");
    }

    /**
     * @dev Returns the integer division of two unsigned integers.
Reverts with custom message on
     * division by zero. The result is rounded towards zero.
     *
     * Counterpart to Solidity's `/` operator. Note: this function
uses a
     * `revert` opcode (which leaves remaining gas untouched)
while Solidity
     * uses an invalid opcode to revert (consuming all remaining
gas).
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b, string memory errorMessage)
internal pure returns (uint256) {
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in
```

```
which this doesn't hold

        return c;
    }

    function mod(uint256 a, uint256 b) internal pure returns
(uint256) {
        return mod(a, b, "SafeMath: modulo by zero");
    }

    /**
     * @dev Returns the remainder of dividing two unsigned
integers. (unsigned integer modulo),
     * Reverts with custom message when dividing by zero.
     *
     * Counterpart to Solidity's `%` operator. This function uses
a `revert`
     * opcode (which leaves remaining gas untouched) while
Solidity uses an
     * invalid opcode to revert (consuming all remaining gas).
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b, string memory errorMessage)
internal pure returns (uint256) {
        require(b != 0, errorMessage);
        return a % b;
    }
}
```

3. PFI Contract

```
contract PoodleFootballInu is Context, IERC20, Ownable {
    using SafeMath for uint256;
    using Address for address;

    uint256 private constant MAX = ~uint256(0);
    uint256 private _tTotal = 100*1e12* 10**18;
    uint256 private _rTotal = (MAX - (MAX % _tTotal));
    uint256 private _tFeeTotal;

    uint256 public _buyTaxFee = 1;
    uint256 public _buyAdvestisementFee = 2;
    uint256 public _buyBurnFee = 2;

    uint256 public _sellTaxFee = 1;
    uint256 public _sellAdvestisementFee = 2;
    uint256 public _sellBurnFee = 2;

    uint256 private _taxFee = _buyTaxFee;
    uint256 private _previousTaxFee = _taxFee;

    uint256 private _burnFee = _buyBurnFee;
    uint256 private _previousBurn= _burnFee;

    uint256 public _advestisementFee = _buyAdvestisementFee;
    uint256 private _previousAdvestisementFee = _advestisementFee;

    address public constant DEAD_ADDRESS =
0x000000000000000000000000000000000000dEaD;
    address public  advertisementWallet =
0x82d2806a687da372a71f93859552F82E346121B0;

    string private _name = "Poodle Football Inu";
    string private _symbol = "PFI";
    uint8 private _decimals = 18;

    mapping (address => uint256) private _rOwned;
    mapping (address => uint256) private _tOwned;
    mapping (address => mapping (address => uint256)) private
_allowances;

    mapping (address => bool) private _isExcludedFromFee;
```

```
    mapping (address => bool) public ammPairs;

    mapping (address => bool) private _isExcluded;
    address[] private _excluded;

    constructor () public {
        _rOwned[_msgSender()] = _rTotal;

        IUniswapV2Router01 _uniswapV2Router =
IUniswapV2Router01(0x10ED43C718714eb63d5aA57B78B54704E256024E);

         // Create a uniswap pair for this new token
        address _uniswapV2Pair =
IUniswapV2Factory(_uniswapV2Router.factory())
            .createPair(address(this), _uniswapV2Router.WETH());

        ammPairs[_uniswapV2Pair] = true;

        _isExcludedFromFee[owner()] = true;
        _isExcludedFromFee[address(this)] = true;

        excludeFromReward(DEAD_ADDRESS);

        emit Transfer(address(0), _msgSender(), _tTotal);
    }

    function name() public view returns (string memory) {
        return _name;
    }

    function symbol() public view returns (string memory) {
        return _symbol;
    }

    function decimals() public view returns (uint8) {
        return _decimals;
    }

    function totalSupply() public view override returns (uint256)
{
        return _tTotal;
    }
```

```
    function balanceOf(address account) public view override
returns (uint256) {
        if (_isExcluded[account]) return _tOwned[account];
        return tokenFromReflection(_rOwned[account]);
    }

    function transfer(address recipient, uint256 amount) public
override returns (bool) {
        _transfer(_msgSender(), recipient, amount);
        return true;
    }

    function allowance(address owner, address spender) public view
override returns (uint256) {
        return _allowances[owner][spender];
    }

    function approve(address spender, uint256 amount) public
override returns (bool) {
        _approve(_msgSender(), spender, amount);
        return true;
    }

    function transferFrom(address sender, address recipient,
uint256 amount) public override returns (bool) {
        _transfer(sender, recipient, amount);
        _approve(sender, _msgSender(),
_allowances[sender][_msgSender()].sub(amount, "ERC20: transfer
amount exceeds allowance"));
        return true;
    }

    function increaseAllowance(address spender, uint256
addedValue) public virtual returns (bool) {
        _approve(_msgSender(), spender,
_allowances[_msgSender()][spender].add(addedValue));
        return true;
    }

    function decreaseAllowance(address spender, uint256
subtractedValue) public virtual returns (bool) {
```

```
        _approve(_msgSender(), spender,
_allowances[_msgSender()][spender].sub(subtractedValue, "ERC20:
decreased allowance below zero"));
        return true;
    }

    function isExcludedFromReward(address account) public view
returns (bool) {
        return _isExcluded[account];
    }

    function totalFees() public view returns (uint256) {
        return _tFeeTotal;
    }

    function reflectionFromToken(uint256 tAmount, bool
deductTransferFee) public view returns(uint256) {
        require(tAmount <= _tTotal, "Amount must be less than
supply");
        if (!deductTransferFee) {
            (uint256 rAmount,,,,,,) = _getValues(tAmount);
            return rAmount;
        } else {
            (,uint256 rTransferAmount,,,,,) = _getValues(tAmount);
            return rTransferAmount;
        }
    }

    function tokenFromReflection(uint256 rAmount) public view
returns(uint256) {
        require(rAmount <= _rTotal, "Amount must be less than
total reflections");
        uint256 currentRate =  _getRate();
        return rAmount.div(currentRate);
    }

    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);
    }

    function includeInReward(address account) external onlyOwner()
{
        require(_isExcluded[account], "Account is already
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;
            }
        }
    }

    function _transferBothExcluded(address sender, address
recipient, uint256 tAmount) private {
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee,
uint256 tTransferAmount, uint256 tFee, uint256 tAdvertisement,
uint256 tBurn) = _getValues(tAmount);
        _tOwned[sender] = _tOwned[sender].sub(tAmount);
        _rOwned[sender] = _rOwned[sender].sub(rAmount);
        _tOwned[recipient] =
_tOwned[recipient].add(tTransferAmount);
        _rOwned[recipient] =
_rOwned[recipient].add(rTransferAmount);
        _takeAdvertisement(tAdvertisement);
        _reflectFee(rFee, tFee);
        _takeBurn(tBurn);
        emit Transfer(sender, recipient, tTransferAmount);
    }

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

```
    function changeAdvestisementWallets(address wallet) public
onlyOwner{
        advertisementWallet = wallet;
    }

    function includeInFee(address account) public onlyOwner {
        _isExcludedFromFee[account] = false;
    }

    function _reflectFee(uint256 rFee, uint256 tFee) private {
        _rTotal = _rTotal.sub(rFee);
        _tFeeTotal = _tFeeTotal.add(tFee);
    }

    function _getValues(uint256 tAmount) private view returns
(uint256, uint256, uint256, uint256, uint256, uint256,uint256) {
        (uint256 tTransferAmount, uint256 tFee, uint256
tAdvertisement, uint256 tBurn) = _getTValues(tAmount);
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee) =
_getRValues(tAmount, tFee, tAdvertisement,tBurn, _getRate());
        return (rAmount, rTransferAmount, rFee, tTransferAmount,
tFee, tAdvertisement,tBurn);
    }

    function _getTValues(uint256 tAmount) private view returns
(uint256, uint256,uint256, uint256) {
        uint256 tFee = calculateTaxFee(tAmount);
        uint256 tAdvertisement =
calculateAdvestisementFee(tAmount);
        uint256 tBurn = calculateBurnFee(tAmount);

        uint256 tTransferAmount =
tAmount.sub(tFee).sub(tAdvertisement).sub(tBurn);

        return (tTransferAmount, tFee, tAdvertisement,tBurn);
    }

    function _getRValues(uint256 tAmount, uint256 tFee, uint256
tAdvertisement,uint256 tBurn, uint256 currentRate) private pure
returns (uint256, uint256, uint256) {
        uint256 rAmount = tAmount.mul(currentRate);
        uint256 rFee = tFee.mul(currentRate);
```

```
        uint256 rAdvertisement = tAdvertisement.mul(currentRate);
        uint256 rBurn = tBurn.mul(currentRate);
        uint256 rTransferAmount =
rAmount.sub(rFee).sub(rAdvertisement).sub(rBurn);
        return (rAmount, rTransferAmount, rFee);
    }

    function _getRate() private view returns(uint256) {
        (uint256 rSupply, uint256 tSupply) = _getCurrentSupply();
        return rSupply.div(tSupply);
    }

    function _getCurrentSupply() private view returns(uint256,
uint256) {
        uint256 rSupply = _rTotal;
        uint256 tSupply = _tTotal;
        for (uint256 i = 0; i < _excluded.length; i++) {
            if (_rOwned[_excluded[i]] > rSupply ||
_tOwned[_excluded[i]] > tSupply) return (_rTotal, _tTotal);
            rSupply = rSupply.sub(_rOwned[_excluded[i]]);
            tSupply = tSupply.sub(_tOwned[_excluded[i]]);
        }
        if (rSupply < _rTotal.div(_tTotal)) return (_rTotal,
_tTotal);
        return (rSupply, tSupply);
    }

    function _takeAdvertisement(uint256 tAdvertisement) private {
        uint256 currentRate =  _getRate();
        uint256 rAdvertisement = tAdvertisement.mul(currentRate);
        _rOwned[advertisementWallet] =
_rOwned[advertisementWallet].add(rAdvertisement);
        if(_isExcluded[advertisementWallet])
            _tOwned[advertisementWallet] =
_tOwned[advertisementWallet].add(tAdvertisement);
    }

      function _takeBurn(uint256 tBurn) private {
        uint256 currentRate =  _getRate();
        uint256 rBurn = tBurn.mul(currentRate);
        _rOwned[DEAD_ADDRESS] = _rOwned[DEAD_ADDRESS].add(rBurn);
        if(_isExcluded[DEAD_ADDRESS])
```

```
            _tOwned[DEAD_ADDRESS] =
_tOwned[DEAD_ADDRESS].add(tBurn);
    }

    function calculateTaxFee(uint256 _amount) private view returns
(uint256) {
        return _amount.mul(_taxFee).div(
            10**2
        );
    }

    function calculateAdvestisementFee(uint256 _amount) private
view returns (uint256) {
        return _amount.mul(_advestisementFee).div(
            10**2
        );
    }

      function calculateBurnFee(uint256 _amount) private view
returns (uint256) {
        return _amount.mul(_burnFee).div(
            10**2
        );
    }

    function removeAllFee() private {
        if(_taxFee == 0 && _advestisementFee == 0 && _burnFee ==0
) return;

        _previousTaxFee = _taxFee;
        _previousAdvestisementFee = _advestisementFee;
        _previousBurn = _burnFee;
        _taxFee = 0;
        _advestisementFee = 0;
        _burnFee = 0;
    }

    function restoreAllFee() private {
        _taxFee = _previousTaxFee;
        _advestisementFee = _previousAdvestisementFee;
        _burnFee = _previousBurn;
```

```
    }

    function _approve(address owner, address spender, uint256
amount) private {
        require(owner != address(0), "ERC20: approve from the zero
address");
        require(spender != address(0), "ERC20: approve to the zero
address");

        _allowances[owner][spender] = amount;
        emit Approval(owner, spender, amount);
    }

    function _transfer(
        address from,
        address to,
        uint256 amount
    ) private {
        require(from != address(0), "ERC20: transfer from the zero
address");
        require(to != address(0), "ERC20: transfer to the zero
address");
        require(amount > 0, "Transfer amount must be greater than
zero");

        bool takeFee = true;

        //if any account belongs to _isExcludedFromFee account
then remove the fee
        if(_isExcludedFromFee[from] || _isExcludedFromFee[to]){
            takeFee = false;
        }

        //transfer amount, it will take tax, burn, advertisement
fee
        _tokenTransfer(from,to,amount,takeFee);
    }
    function _tokenTransfer(address sender, address recipient,
uint256 amount,bool takeFee) private {
        if(!takeFee){
            removeAllFee();
        }else{
```

```
            bool isBuy = ammPairs[sender];
            bool isSell = ammPairs[recipient];
            if(isBuy){
                _taxFee = _buyTaxFee;
                _advestisementFee = _buyAdvestisementFee;
                _burnFee = _buyBurnFee;
            }else if(isSell){
                _taxFee = _sellTaxFee;
                _advestisementFee = _sellAdvestisementFee;
                _burnFee = _sellBurnFee;
            }
            takeFee = isBuy || isSell;

            if(!takeFee){
                removeAllFee();
            }
        }

        if (_isExcluded[sender] && !_isExcluded[recipient]) {
            _transferFromExcluded(sender, recipient, amount);
        } else if (!_isExcluded[sender] && _isExcluded[recipient])
{
            _transferToExcluded(sender, recipient, amount);
        } else if (!_isExcluded[sender] &&
!_isExcluded[recipient]) {
            _transferStandard(sender, recipient, amount);
        } else if (_isExcluded[sender] && _isExcluded[recipient])
{
            _transferBothExcluded(sender, recipient, amount);
        } else {
            _transferStandard(sender, recipient, amount);
        }

        if(!takeFee)
            restoreAllFee();
    }
    function _transferStandard(address sender, address recipient,
uint256 tAmount) private {
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee,
uint256 tTransferAmount, uint256 tFee, uint256 tAdvertisement,
uint256 tBurn) = _getValues(tAmount);
        _rOwned[sender] = _rOwned[sender].sub(rAmount);
```

```
        _rOwned[recipient] =
_rOwned[recipient].add(rTransferAmount);
        _takeAdvertisement(tAdvertisement);
        _reflectFee(rFee, tFee);
        _takeBurn(tBurn);

        emit Transfer(sender, recipient, tTransferAmount);
    }
     function _transferToExcluded(address sender, address
recipient, uint256 tAmount) private {
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee,
uint256 tTransferAmount, uint256 tFee, uint256 tAdvertisement,
uint256 tBurn) = _getValues(tAmount);
        _rOwned[sender] = _rOwned[sender].sub(rAmount);
        _tOwned[recipient] =
_tOwned[recipient].add(tTransferAmount);
        _rOwned[recipient] =
_rOwned[recipient].add(rTransferAmount);
        _takeAdvertisement(tAdvertisement);
        _takeBurn(tBurn);

        _reflectFee(rFee, tFee);
        emit Transfer(sender, recipient, tTransferAmount);
    }
    function _transferFromExcluded(address sender, address
recipient, uint256 tAmount) private {
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee,
uint256 tTransferAmount, uint256 tFee, uint256 tAdvertisement,
uint256 tBurn) = _getValues(tAmount);
        _tOwned[sender] = _tOwned[sender].sub(tAmount);
        _rOwned[sender] = _rOwned[sender].sub(rAmount);
        _rOwned[recipient] =
_rOwned[recipient].add(rTransferAmount);
        _takeAdvertisement(tAdvertisement);
        _reflectFee(rFee, tFee);
        _takeBurn(tBurn);

        emit Transfer(sender, recipient, tTransferAmount);
    }
}
```

READ CONTRACT (ONLY NEED TO KNOW)

1. DEAD
0x000000000000000000000000000000000000dead address
(Shows dead address)

2. _advertisementFee
2 uint256
(Shows advertisement fee)

2. _buyAdvertisementFee
2 uint256
(Shows buy advertisement fee)

5. getOwner
0x82d2806a687da372a71f93859552f82e346121b0 address
(Function for read owner)

6. name
Poodle Football Inu string
(Function for read Token name))

WRITE CONTRACT

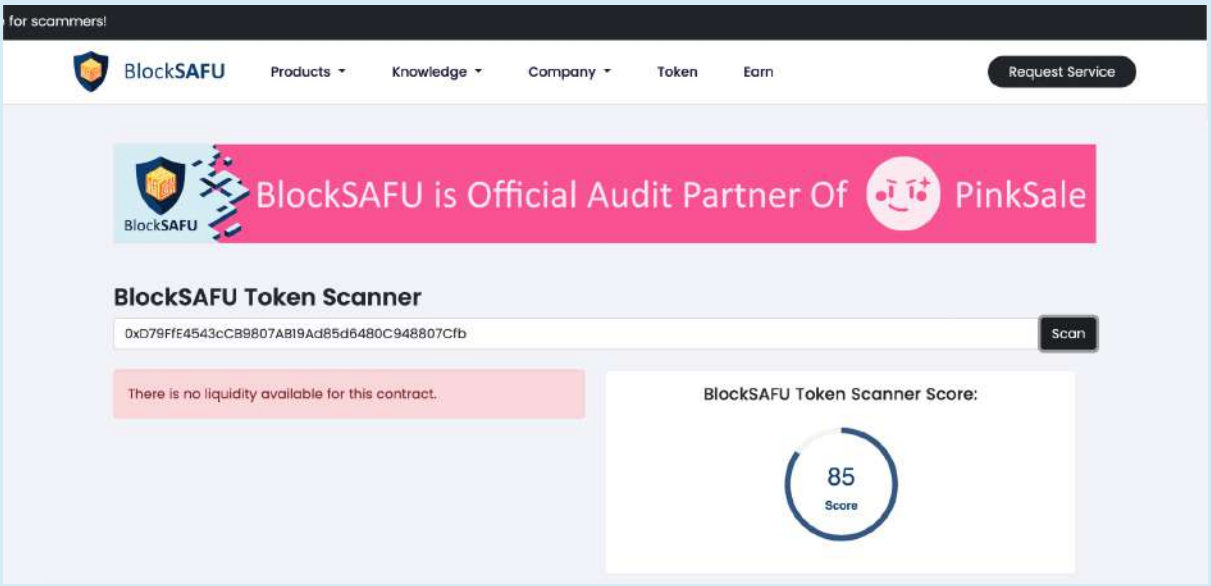
1. renounceOwnership
(Renouncing ownership will leave the contract without an owner, thereby removing any functionality that is only available to the owner)

2. transferOwnership
newOwner (address)
(Its function is to change the owner)

3. changeAdvertisementWallet
wallet (address)
(The form is filled with address, for set advertisement wallet)

BlockSAFU Scanner

https://blocksafu.com/token-scanner



Token Information

Indicator	Value
Token Name	Poodle Football Inu
Token Symbol	PFI
Total Supply	100,000,000,000,000
Already Listed On Dex	Already Listed
Dex Listed	PancakeV2
Open Source	Open Source
Price	$0.00000000
Volume 24H	$0.00
Liquidity	$0 (0.00 BNB)
Tx Count 24H	0
Marketcap	$0

Security Information

Indicator	Value
Honeypot	Liquidity Not Available
Buy Fees	0%
Sell Fees	0%
Buy Gas	0 **Gwei (0.000000 BNB / $0.00)**
Sell Gas	0 **Gwei (0.000000 BNB / $0.00)**
Holder Count	1 **Holders**

Honeypot Safety

Indicator	Value
Can Take Back Ownership	Not detected
Owner Change Balance	Not detected
Blacklist	Not detected
Modify Fees	Not detected
Proxy	Not detected
Whitelisted	Not detected
Anti Whale	Not detected
Trading Cooldown	Not detected
Transfer Pausable	Not detected
Cannot Sell All	Not detected

Rug Pull Safety

Indicator	Value
Hidden Owner	Not detected
Creator Address	0x82d2806a...1b0
Creator Balance	100,000,000,000,000 PFI
Creator Percent	100%
Owner Address	0x82d2806a...1b0
Owner Balance	100,000,000,000,000 PFI
Owner Percent	100%
Lp Holder Count	0
Lp Total Supply	NaN
Mint	Not detected

WEBSITE REVIEW

🟢 **Mobile Friendly**

🟢 **Contains no code error**

🔴 **SSL Not Secure**

Web-Tech stack: Font Awesome, Particle js

Domain .io (Namecheap) - Tracked by whois

First Contentful Paint:	573ms
Fully Loaded Time	1.5s
Performance	72%
Accessibility	94%
Best Practices	83%
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 DEX listing*)

- TOP 5 Holder.

  (*Will be updated after DEX listing*)

- The Team KYC on PinkSale

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. Note, 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.