Submitted audit for **SolanaInu** token on 13 November 2023
Audit result: **Passed**

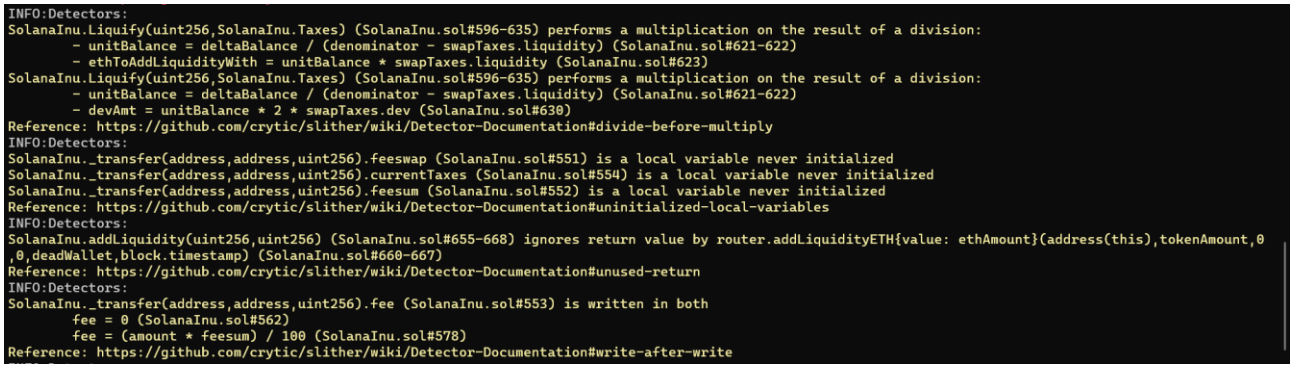
Token Address: 0x9B699293561f7738eA9f8D1b95412E811d530547
Name: Solana Inu
Symbol: Solana
Decimals: 18
Network: Binance smart chain
Token Type: ERC20
Owner: 0xdd157AbfF1F2688f6020ED4cb83bee76F9911c6658
Deployer: 0xdd157AbfF1F2688f6020ED4cb83bee76F9911c66
Token Supply: 100000
Checksum: 30b62c72cb68e6e74fc455033097b98b
Testnet version:
The tests were performed using the contract deployed on the Binance smart chain Testnet, which can be found at the following address:
https://testnet.bscscan.com/address/0x9197274ae3c74794fbdec24b326c68dd2c9820ed#code

Tools:

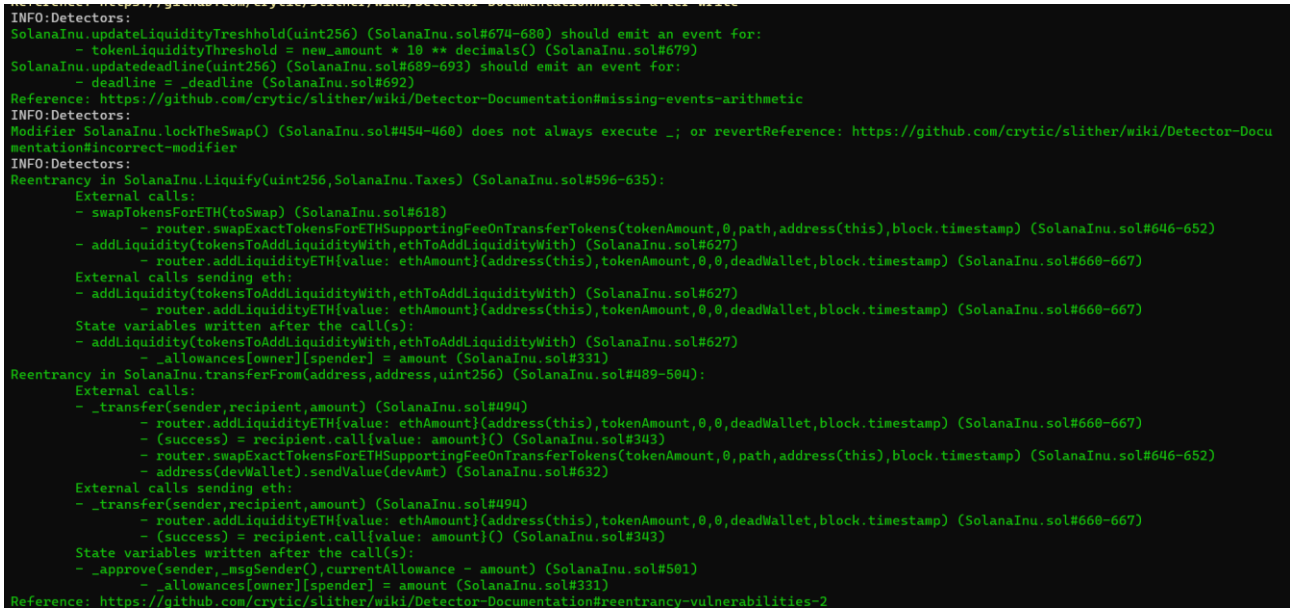
1. Manual Review: The code has undergone a line-by-line review by the **Ace** team.
2. BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.
3. Slither: The code has undergone static analysis using Slither.

Static Analysis

A static analysis of the code was performed using Slither.



```
INFO:Detectors:
SolanaInu.Liquify(uint256,SolanaInu.Taxes) (SolanaInu.sol#596-635) performs a multiplication on the result of a division:
        - unitBalance = deltaBalance / (denominator - swapTaxes.liquidity) (SolanaInu.sol#621-622)
        - ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity (SolanaInu.sol#623)
SolanaInu.Liquify(uint256,SolanaInu.Taxes) (SolanaInu.sol#596-635) performs a multiplication on the result of a division:
        - unitBalance = deltaBalance / (denominator - swapTaxes.liquidity) (SolanaInu.sol#621-622)
        - devAmt = unitBalance * 2 * swapTaxes.dev (SolanaInu.sol#630)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
SolanaInu._transfer(address,address,uint256).feeswap (SolanaInu.sol#551) is a local variable never initialized
SolanaInu._transfer(address,address,uint256).currentTaxes (SolanaInu.sol#554) is a local variable never initialized
SolanaInu._transfer(address,address,uint256).feesum (SolanaInu.sol#552) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
SolanaInu.addLiquidity(uint256,uint256) (SolanaInu.sol#655-668) ignores return value by router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0
,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
SolanaInu._transfer(address,address,uint256).fee (SolanaInu.sol#553) is written in both
        fee = 0 (SolanaInu.sol#562)
        fee = (amount * feesum) / 100 (SolanaInu.sol#578)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#write-after-write
```



```
INFO:Detectors:
SolanaInu.updateLiquidityTreshhold(uint256) (SolanaInu.sol#674-680) should emit an event for:
        - tokenLiquidityThreshold = new_amount * 10 ** decimals() (SolanaInu.sol#679)
SolanaInu.updatedeadline(uint256) (SolanaInu.sol#689-693) should emit an event for:
        - deadline = _deadline (SolanaInu.sol#692)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Modifier SolanaInu.lockTheSwap() (SolanaInu.sol#454-460) does not always execute _; or revertReference: https://github.com/crytic/slither/wiki/Detector-Docu
mentation#incorrect-modifier
INFO:Detectors:
Reentrancy in SolanaInu.Liquify(uint256,SolanaInu.Taxes) (SolanaInu.sol#596-635):
        External calls:
        - swapTokensForETH(toSwap) (SolanaInu.sol#618)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SolanaInu.sol#646-652)
        - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
        External calls sending eth:
        - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
        State variables written after the call(s):
        - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
                - _allowances[owner][spender] = amount (SolanaInu.sol#331)
Reentrancy in SolanaInu.transferFrom(address,address,uint256) (SolanaInu.sol#489-504):
        External calls:
        - _transfer(sender,recipient,amount) (SolanaInu.sol#494)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
                - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SolanaInu.sol#646-652)
                - address(devWallet).sendValue(devAmt) (SolanaInu.sol#632)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (SolanaInu.sol#494)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
                - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),currentAllowance - amount) (SolanaInu.sol#501)
                - _allowances[owner][spender] = amount (SolanaInu.sol#331)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
```

```
INFO:Detectors:
Reentrancy in SolanaInu.Liquify(uint256,SolanaInu.Taxes) (SolanaInu.sol#596-635):
        External calls:
        - swapTokensForETH(toSwap) (SolanaInu.sol#618)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SolanaInu.sol#646-652)
        - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
        External calls sending eth:
        - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (SolanaInu.sol#332)
                - addLiquidity(tokensToAddLiquidityWith,ethToAddLiquidityWith) (SolanaInu.sol#627)
Reentrancy in SolanaInu._transfer(address,address,uint256) (SolanaInu.sol#540-594):
        External calls:
        - Liquify(feeswap,currentTaxes) (SolanaInu.sol#583)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
                - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SolanaInu.sol#646-652)
                - address(devWallet).sendValue(devAmt) (SolanaInu.sol#632)
        External calls sending eth:
        - Liquify(feeswap,currentTaxes) (SolanaInu.sol#583)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
                - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (SolanaInu.sol#293)
                - super._transfer(sender,address(this),feeAmount) (SolanaInu.sol#591)
        - Transfer(sender,recipient,amount) (SolanaInu.sol#293)
                - super._transfer(sender,recipient,amount - fee) (SolanaInu.sol#586)
Reentrancy in SolanaInu.transferFrom(address,address,uint256) (SolanaInu.sol#489-504):
        External calls:
        - _transfer(sender,recipient,amount) (SolanaInu.sol#494)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (SolanaInu.sol#660-667)
                - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SolanaInu.sol#646-652)
                - address(devWallet).sendValue(devAmt) (SolanaInu.sol#632)
        External calls sending eth:
```

```
INFO:Detectors:
Context._msgData() (SolanaInu.sol#13-16) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.19 (SolanaInu.sol#6) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
solc-0.8.22 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (SolanaInu.sol#337-348):
        - (success) = recipient.call{value: amount}() (SolanaInu.sol#343)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IRouter.WETH() (SolanaInu.sol#401) is not in mixedCase
Function SolanaInu.Liquify(uint256,SolanaInu.Taxes) (SolanaInu.sol#596-635) is not in mixedCase
Parameter SolanaInu.updateLiquidityTreshhold(uint256).new_amount (SolanaInu.sol#674) is not in mixedCase
Function SolanaInu.EnableTrading() (SolanaInu.sol#682-687) is not in mixedCase
Parameter SolanaInu.updatedeadline(uint256)._deadline (SolanaInu.sol#689) is not in mixedCase
Parameter SolanaInu.updateExemptFee(address,bool)._address (SolanaInu.sol#718) is not in mixedCase
Variable SolanaInu.genesis_block (SolanaInu.sol#436) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (SolanaInu.sol#14)" inContext (SolanaInu.sol#8-17)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
SolanaInu.launchtax (SolanaInu.sol#438) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
SolanaInu.pair (SolanaInu.sol#428) should be immutable
SolanaInu.router (SolanaInu.sol#427) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:SolanaInu.sol analyzed (9 contracts with 93 detectors), 32 result(s) found
```

Functional Tests
Router (PCS V2):

1- Approve (passed):
https://testnet.bscscan.com/tx/0x5bede0b31b5d0f63d9d5d7f6cf7f17e1a356122a9693eac4acc1cf84553cd961

2- Enable Trading (passed):
https://testnet.bscscan.com/tx/0x6deeab9fd8adcaeacb3b3d275182611e4a25fbb5e19441fc341b69f24b9894cb

3- Bulk Exempt Fee (passed):
https://testnet.bscscan.com/tx/0x5c61433e8d189826d0cf1b09ead70ac638757c316fb8f0c712687351dcc304e1

4- **Increase Allowance** (passed):
https://testnet.bscscan.com/tx/0xa3296f53a648751d05c178e6f9b6a2b8c4e6ff3e97e9afa2cc8fc04aaad00bdb

5- Decrease Allowance passed):
https://testnet.bscscan.com/tx/0xdcbde039b4082bf1351d3b32b00d392393839923f7e3a8f481506a4bf4f486bf

6- Transfer (passed):
https://testnet.bscscan.com/tx/0x46fe76a0435c619ef3b312d77154500ceb45d758834276325c07c787c1cf1539

7- Transfer Ownership (passed):
https://testnet.bscscan.com/tx/0xe309c1d8bb41844f8146ae8214421490e85f00b925eff5f9d91bb50f51df1f40

Summary:
- Owner can renounce ownership.
- Owner can transfer ownership.
- Owner can update liquidityprovide.
- Owner can update liquiditytreshhold.
- Owner can enable trading.
- Owner can update deadline.
- Owner can update wallets.
- Owner update Exempt fees.

Findings:
Critical: 0
High: 0
Medium: 0
Low: 2
Suggestions & Optimizations: 1

Optimization

Severity: Low
subject: floating Pragma Solidity version
Status: Open
Overview:
It is considered best practice to pick one compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

```
pragma solidity ^0.8.19;
```

Suggestion

Adding the latest constant version of solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

Optimization

Severity: Low
subject: Missing Events
Status: Open
Overview:
They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function EnableTrading() external onlyOwner {
        require(!tradingEnabled, "Cannot re-enable trading");
        tradingEnabled = true;
        providingLiquidity = true;
        genesis_block = block.number;
    }

    function updatedeadline(uint256 _deadline) external onlyOwner {
        require(!tradingEnabled, "Dev Can't change when trading has started");
        require(_deadline < 5, "Deadline should be less than 5 Blocks");
        deadline = _deadline;
    }
```

```
    function updateDevWallet(address newWallet) external onlyOwner {
        require(newWallet != address(0), "Fee Address cannot be 0 address");
        devWallet = newWallet;
    }
```

```
function transferOwnership(address newOwner) public virtual onlyOwner {
        require(
            newOwner != address(0),
            "Ownable: new owner is the zero address"
        );
        _setOwner(newOwner);
    }
```

Suggestion:
Events are important and should be emitted for tracking this off-chain for all important functions.

Optimization

Severity: Suggestion/Informational
subject: Wrong Naming Convention
Status: Open
Overview:
Wrong naming convention. Private Functions' name should start with '_'

```
function Liquify(
        uint256 feeswap,
        Taxes memory swapTaxesS
    ) private lockTheSwap {
        if (feeswap == 0) {
            return;
        }
```

Suggestion:
It is recommended that Clear and consistent naming conventions are essential for writing clean code. They improve code readability and help developers understand the purpose and functionality of variables, functions, and contracts.