Submitted audit for **GbtLiquidity** on 23 November 2023
Audit result: **Passed**

Token Address: - 0xfD609D76DDfa6DAb2944C650981675148E899eB0
Name: GoldenBambooToken
Symbol: GBT
Decimals: 18
Network: Binance smart chain
Token Type: ERC20
Owner: - 0x0000000000000000000000000000000000000000
Deployer: - 0x0000000000000000000000000000000000000000
Token Supply: 100000000000000000000
Checksum: 2f36b9f293fd8f6ff54eccdccdf888fd
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/0x335928fce2357f431f18c5e860283d6743243771#code

Tools:

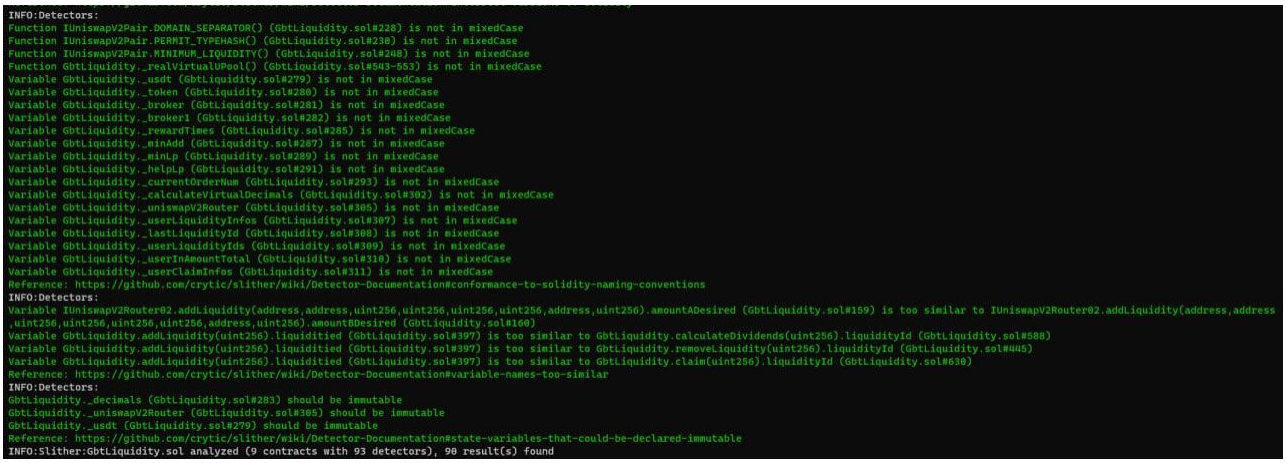
1. Manual Review: The code has undergone a line-by-line review by the **Ace** team.
2. BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.
3. Slither: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

Static Analysis

A static analysis of the code was performed using Slither.

```
INFO:Detectors:
GbtLiquidity._claim(address,GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#640-660) uses a dangerous strict equality:
        - info.status == 0 && info.claim >= info.amount.mul(_rewardTimes) (GbtLiquidity.sol#648)
GbtLiquidity.calculateDividends(uint256) (GbtLiquidity.sol#588-607) uses a dangerous strict equality:
        - info.status == 1 || info.status == 2 || info.account == address(0) (GbtLiquidity.sol#590)
GbtLiquidity.calculateVirtualLP(uint256) (GbtLiquidity.sol#578-586) uses a dangerous strict equality:
        - virtualLPPool == 0 (GbtLiquidity.sol#579)
GbtLiquidity.canAddLiquidity() (GbtLiquidity.sol#386-388) uses a dangerous strict equality:
        - _minLp == 0 || _minLp > _helpLp (GbtLiquidity.sol#387)
GbtLiquidity.getRemoveIds(uint256,uint256) (GbtLiquidity.sol#500-522) uses a dangerous strict equality:
        - info.status == 0 && info.amount > 0 (GbtLiquidity.sol#514)
GbtLiquidity.userInAmountSum(address) (GbtLiquidity.sol#731-738) uses a dangerous strict equality:
        - info.status == 0 (GbtLiquidity.sol#734)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in GbtLiquidity._claim(address,GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#640-660):
        External calls:
        - _transfer(_usdt,account,amount) (GbtLiquidity.sol#643)
                - IERC20(token).transfer(account,amount) (GbtLiquidity.sol#561)
        State variables written after the call(s):
        - totalDividends += amount (GbtLiquidity.sol#646)
        GbtLiquidity.totalDividends (GbtLiquidity.sol#300) can be used in cross function reentrancies:
        - GbtLiquidity._claim(address,GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#640-660)
        - GbtLiquidity._realVirtualUPool() (GbtLiquidity.sol#543-553)
        - GbtLiquidity._removeVirtuals(GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#481-497)
        - GbtLiquidity.totalDividends (GbtLiquidity.sol#300)
        - _removeVirtuals(info) (GbtLiquidity.sol#652)
                - totalDividends -= info.claim (GbtLiquidity.sol#495)
        GbtLiquidity.totalDividends (GbtLiquidity.sol#300) can be used in cross function reentrancies:
        - GbtLiquidity._claim(address,GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#640-660)
        - GbtLiquidity._realVirtualUPool() (GbtLiquidity.sol#543-553)
        - GbtLiquidity._removeVirtuals(GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#481-497)
        - GbtLiquidity.totalDividends (GbtLiquidity.sol#300)
        - _removeVirtuals(info) (GbtLiquidity.sol#652)
                - virtualLPPool -= info.virtualLP (GbtLiquidity.sol#490)
                - virtualLPPool = 0 (GbtLiquidity.sol#492)
        GbtLiquidity.virtualLPPool (GbtLiquidity.sol#298) can be used in cross function reentrancies:
        - GbtLiquidity._removeVirtuals(GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#481-497)
        - GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442)
        - GbtLiquidity.calculateDividends(uint256) (GbtLiquidity.sol#588-607)
        - GbtLiquidity.calculateVirtualLP(uint256) (GbtLiquidity.sol#578-586)
        - GbtLiquidity.virtualLPPool (GbtLiquidity.sol#298)
        - _removeVirtuals(info) (GbtLiquidity.sol#652)
                - virtualUPool -= amount (GbtLiquidity.sol#484)
                - virtualUPool = 0 (GbtLiquidity.sol#486)
        GbtLiquidity.virtualUPool (GbtLiquidity.sol#296) can be used in cross function reentrancies:
        - GbtLiquidity._realVirtualUPool() (GbtLiquidity.sol#543-553)
        - GbtLiquidity._removeVirtuals(GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#481-497)
        - GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442)
        - GbtLiquidity.virtualUPool (GbtLiquidity.sol#296)
Reentrancy in GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442):
```

```
INFO:Detectors:
GbtLiquidity.getRemoveIds(uint256,uint256).success (GbtLiquidity.sol#510) is a local variable never initialized
GbtLiquidity.helpRemove(uint256[]).successed (GbtLiquidity.sol#527) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
GbtLiquidity._removeLiquidity(uint256,address) (GbtLiquidity.sol#566-575) ignores return value by IERC20(_lpToken()).approve(address(_uniswapV2Router),lp) (GbtLiquidity.sol#572)
GbtLiquidity._removeLiquidity(uint256,address) (GbtLiquidity.sol#566-575) ignores return value by _uniswapV2Router.removeLiquidity(_usdt,_token,lp,0,0,account,block.timestamp) (GbtLiquidity.sol#573)
GbtLiquidity._lpValue(uint256,bool) (GbtLiquidity.sol#610-623) ignores return value by (reserve1,reserve2) = IUniswapV2Pair(pair).getReserves() (GbtLiquidity.sol#612)
GbtLiquidity._swapTokensForTokensTo(uint256,address) (GbtLiquidity.sol#694-711) ignores return value by IERC20(_usdt).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#700)
GbtLiquidity._addLiquidityTokens(uint256,uint256) (GbtLiquidity.sol#713-728) ignores return value by IERC20(_token).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#715)
GbtLiquidity._addLiquidityTokens(uint256,uint256) (GbtLiquidity.sol#713-728) ignores return value by IERC20(_usdt).approve(address(_uniswapV2Router),uAmount) (GbtLiquidity.sol#716)
GbtLiquidity._addLiquidityTokens(uint256,uint256) (GbtLiquidity.sol#713-728) ignores return value by (None,None,liquidityNum) = _uniswapV2Router.addLiquidity(_usdt,_token,uAmount,tokenAmount,0,0,address(this),
block.timestamp) (GbtLiquidity.sol#718-727)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
GbtLiquidity.setUint(uint256,uint256) (GbtLiquidity.sol#374-384) should emit an event for:
        - _rewardTimes = param (GbtLiquidity.sol#376)
        - _minAdd = param (GbtLiquidity.sol#378)
        - _helpLp = param (GbtLiquidity.sol#380)
        - _calculateVirtualDecimals = param (GbtLiquidity.sol#382)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Ownable.changeOwner(address).newOwner (GbtLiquidity.sol#77) lacks a zero-check on :
                - _owner = newOwner (GbtLiquidity.sol#79)
GbtLiquidity.setAddress(address,uint256).param (GbtLiquidity.sol#364) lacks a zero-check on :
                - _token = param (GbtLiquidity.sol#366)
                - _broker = param (GbtLiquidity.sol#368)
                - _broker1 = param (GbtLiquidity.sol#370)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
GbtLiquidity._realVirtualUPool() (GbtLiquidity.sol#543-553) has external calls inside a loop: balance = IERC20(_usdt).balanceOf(address(this)) (GbtLiquidity.sol#544)
GbtLiquidity._transfer(address,address,uint256) (GbtLiquidity.sol#556-563) has external calls inside a loop: IERC20(token).balanceOf(address(this)) < amount (GbtLiquidity.sol#557)
GbtLiquidity._transfer(address,address,uint256) (GbtLiquidity.sol#556-563) has external calls inside a loop: amount = IERC20(token).balanceOf(address(this)) (GbtLiquidity.sol#558)
GbtLiquidity._transfer(address,address,uint256) (GbtLiquidity.sol#556-563) has external calls inside a loop: IERC20(token).transfer(account,amount) (GbtLiquidity.sol#561)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Reentrancy in GbtLiquidity._claim(address,GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#640-660):
        External calls:
        - _transfer(_usdt,account,amount) (GbtLiquidity.sol#643)
                - IERC20(token).transfer(account,amount) (GbtLiquidity.sol#561)
        State variables written after the call(s):
        - _removeVirtuals(info) (GbtLiquidity.sol#652)
                - _currentOrderNum = _currentOrderNum - 1 (GbtLiquidity.sol#496)
                - _currentOrderNum = 0 (GbtLiquidity.sol#496)
        - _minLp = 0 (GbtLiquidity.sol#657)
        - _userClaimInfos[info.id].push(ClaimInfo(block.timestamp,amount)) (GbtLiquidity.sol#654)
Reentrancy in GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442):
        External calls:
        - IERC20(_usdt).transferFrom(account,address(this),amount) (GbtLiquidity.sol#403)
        - addLiquidityNum = _swapAndLiquify(amount) (GbtLiquidity.sol#405)
INFO:Detectors:
Reentrancy in GbtLiquidity._swapAndLiquify(uint256) (GbtLiquidity.sol#679-692):
        External calls:
        - _swapTokensForTokensTo(half,address(this)) (GbtLiquidity.sol#685)
                - IERC20(_usdt).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#700)
                - _uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,to,block.timestamp) (GbtLiquidity.sol#703-709)
        Event emitted after the call(s):
        - SwapAndLiquify(half,newBalance,otherHalf) (GbtLiquidity.sol#690)
Reentrancy in GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442):
        External calls:
        - IERC20(_usdt).transferFrom(account,address(this),amount) (GbtLiquidity.sol#403)
        - addLiquidityNum = _swapAndLiquify(amount) (GbtLiquidity.sol#405)
                - IERC20(_token).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#715)
                - IERC20(_usdt).approve(address(_uniswapV2Router),uAmount) (GbtLiquidity.sol#716)
                - (None,None,liquidityNum) = _uniswapV2Router.addLiquidity(_usdt,_token,uAmount,tokenAmount,0,0,address(this),block.timestamp) (GbtLiquidity.sol#718-727)
                - IERC20(_usdt).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#700)
                - _uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,to,block.timestamp) (GbtLiquidity.sol#703-709)
        Event emitted after the call(s):
        - SwapAndLiquify(half,newBalance,otherHalf) (GbtLiquidity.sol#690)
                - addLiquidityNum = _swapAndLiquify(amount) (GbtLiquidity.sol#405)
Reentrancy in GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442):
        External calls:
        - IERC20(_usdt).transferFrom(account,address(this),amount) (GbtLiquidity.sol#403)
        - addLiquidityNum = _swapAndLiquify(amount) (GbtLiquidity.sol#405)
                - IERC20(_token).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#715)
                - IERC20(_usdt).approve(address(_uniswapV2Router),uAmount) (GbtLiquidity.sol#716)
                - (None,None,liquidityNum) = _uniswapV2Router.addLiquidity(_usdt,_token,uAmount,tokenAmount,0,0,address(this),block.timestamp) (GbtLiquidity.sol#718-727)
                - IERC20(_usdt).approve(address(_uniswapV2Router),tokenAmount) (GbtLiquidity.sol#700)
                - _uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,to,block.timestamp) (GbtLiquidity.sol#703-709)
        - IERC20(_token).transfer(_token,balance) (GbtLiquidity.sol#426)
        Event emitted after the call(s):
        - AddLiquidity(account,amount) (GbtLiquidity.sol#441)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
GbtLiquidity.canAddLiquidity() (GbtLiquidity.sol#386-388) uses timestamp for comparisons
        Dangerous comparisons:
        - _minLp == 0 || _minLp > _helpLp (GbtLiquidity.sol#387)
GbtLiquidity.addLiquidity(uint256) (GbtLiquidity.sol#391-442) uses timestamp for comparisons
        Dangerous comparisons:
        - _minLp == 0 || virtualLP < _minLp (GbtLiquidity.sol#430)
GbtLiquidity.removeLiquidity(uint256) (GbtLiquidity.sol#445-478) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(info.status != 2,the order is over) (GbtLiquidity.sol#450)
GbtLiquidity._removeVirtuals(GbtLiquidity.LiquidityInfo) (GbtLiquidity.sol#481-497) uses timestamp for comparisons
        Dangerous comparisons:
        - virtualLPPool >= info.virtualLP (GbtLiquidity.sol#489)
GbtLiquidity.getRemoveIds(uint256,uint256) (GbtLiquidity.sol#500-522) uses timestamp for comparisons
        Dangerous comparisons:
```



Summary:
- Owner can renounce the ownership.
- Owner can transfer the ownership.
- Owner can set the address.
- Owner can set the uint.

Findings:
Critical: 0
High: 0
Medium: 0
Low: 2
Suggestions & Optimizations: 3

Centralization – Missing Zero Address

Severity: Low
function: setAddress and setUint
Status: Open
Overview:

functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function setAddress(address param, uint status) external onlyOwner {
        if (status == 0) {
            _token = param;
        } else if (status == 1) {
            _broker = param;
        } else if (status == 2) {
            _broker1 = param;
        }
    }
function setUint(uint param, uint status) external onlyOwner {
        if (status == 0) {
            _rewardTimes = param;
        } else if (status == 1) {
            _minAdd = param;
        } else if (status == 2) {
            _helpLp = param;
        } else if (status == 3) {
            _calculateVirtualDecimals = param;
        }
    }
```

Suggestion:

It is suggested that the address should not be zero or dead.

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 setAddress(address param, uint status) external onlyOwner {
        if (status == 0) {
            _token = param;
        } else if (status == 1) {
            _broker = param;
        } else if (status == 2) {
            _broker1 = param;
        }
    }
function setUint(uint param, uint status) external onlyOwner {
        if (status == 0) {
            _rewardTimes = param;
        } else if (status == 1) {
            _minAdd = param;
        } else if (status == 2) {
            _helpLp = param;
        } else if (status == 3) {
            _calculateVirtualDecimals = param;
        }
    }
```

Optimization

Severity: Informational
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: Informational
subject: uint256
Status: Open
Overview:

Use uit256 instead of uint. uint is an alias for uint256 and is not recommended for use. The variable size should be clarified, as this can cause issues when encoding data with selectors if the alias is mistakenly used within the signature string.

```
function addLiquidity(uint amount) external {
        require(amount >= _minAdd, "Liquidity:The admission amount is too low");
        uint realAmount = amount.div(1 * 10 ** _decimals);
        require(realAmount.mul(1 * 10 ** _decimals) == amount, "Liquidity:Add only
integers");
        address account = _msgSender();
        uint invest = Broker(_broker).userInvestTotal(account) +
Broker(_broker1).userInvestTotal(account);
        uint liquiditied = _userInAmountTotal[account];
```

Optimization

Severity: Informational
subject: Remove Safe Math
Status: Open
Line: 93 - 151
Overview:

compiler version above 0.8.0 has the ability to control arithmetic overflow/underflow, It is recommended to remove the unwanted code in order to avoid high gas fees.