


# Advanced Manual Smart Contract Audit

October 14, 2022

 [CoinsultAudits](https://twitter.com/CoinsultAudits)

 [info@coinsult.net](mailto:info@coinsult.net)

 [coinsult.net](https://coinsult.net)

Audit requested by



**Sport Fantasy**

0x4b317c9a320f242b29a707f8507a96b54a28cc79

# Table of Contents

## 1. Audit Summary

- 1.1 Audit scope
- 1.2 Tokenomics
- 1.3 Source Code

## 2. Disclaimer

## 3. Global Overview

- 3.1 Informational issues
- 3.2 Low-risk issues
- 3.3 Medium-risk issues
- 3.4 High-risk issues

## 4. Vulnerabilities Findings

## 5. Contract Privileges

- 5.1 Maximum Fee Limit Check
- 5.2 Contract Pausability Check
- 5.3 Max Transaction Amount Check
- 5.4 Exclude From Fees Check
- 5.5 Ability to Mint Check
- 5.6 Ability to Blacklist Check
- 5.7 Owner Privileges Check

## 6. Notes

- 6.1 Notes by Coinsult
- 6.2 Notes by Sport Fantasy

## 7. Contract Snapshot

## 8. Website Review

## 9. Certificate of Proof

# Audit Summary

Project Name	Sport Fantasy
Website	<a href="https://sportfantasy.finance/">https://sportfantasy.finance/</a>
Blockchain	Binance Smart Chain
Smart Contract Language	Solidity
Contract Address	0x4b317c9a320f242b29a707f8507a96b54a28cc79
Audit Method	Static Analysis, Manual Review
Date of Audit	14 October 2022

This audit report has been prepared by Coinsult's experts at the request of the client. In this audit, the results of the static analysis and the manual code review will be presented. The purpose of the audit is to see if the functions work as intended, and to identify potential security issues within the smart contract.

The information in this report should be used to understand the risks associated with the smart contract. This report can be used as a guide for the development team on how the contract could possibly be improved by remediating the issues that were identified.

# Audit Scope

## Source Code

Coinsult was commissioned by Sport Fantasy to perform an audit based on the following code:

<https://bscscan.com/address/0x4b317c9a320f242b29a707f8507a96b54a28cc79#code>

Note that we only audited the code available to us on this URL at the time of the audit. If the URL is not from any block explorer (main net), it may be subject to change. Always check the contract address on this audit report and compare it to the token you are doing research for.

## SAFU by ContractChecker

## Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0x0e9514d7cf4a0675b90256b7214686439a6fdf55	1,000,000	100.0000%

# Audit Method

Coinsult's manual smart contract audit is an extensive methodical examination and analysis of the smart contract's code that is used to interact with the blockchain. This process is conducted to discover errors, issues and security vulnerabilities in the code in order to suggest improvements and ways to fix them.

## ➔ Automated Vulnerability Check

Coinsult uses software that checks for common vulnerability issues within smart contracts. We use automated tools that scan the contract for security vulnerabilities such as integer-overflow, integer-underflow, out-of-gas-situations, unchecked transfers, etc.

## ➔ Manual Code Review

Coinsult's manual code review involves a human looking at source code, line by line, to find vulnerabilities. Manual code review helps to clarify the context of coding decisions. Automated tools are faster but they cannot take the developer's intentions and general business logic into consideration.

## ➔ Used Tools

- Slither: Solidity static analysis framework
- Remix: IDE Developer Tool
- CWE: Common Weakness Enumeration
- SWC: Smart Contract Weakness Classification and Test Cases
- DEX: Testnet Blockchains

# Risk Classification

Coinsult uses certain vulnerability levels, these indicate how bad a certain issue is. The higher the risk, the more strictly it is recommended to correct the error before using the contract.

Vulnerability Level	Description
● Informational	Does not compromise the functionality of the contract in any way
● Low-Risk	Won't cause any problems, but can be adjusted for improvement
● Medium-Risk	Will likely cause problems and it is recommended to adjust
● High-Risk	Will definitely cause problems, this needs to be adjusted

Coinsult has four statuses that are used for each risk level. Below we explain them briefly.

Risk Status	Description
Total	Total amount of issues within this category
Pending	Risks that have yet to be addressed by the team
Acknowledged	The team is aware of the risks but does not resolve them
Resolved	The team has resolved and remedied the risk

# Disclaimer

This audit report has been prepared by Coinsult's experts at the request of the client. In this audit, the results of the static analysis and the manual code review will be presented. The purpose of the audit is to see if the functions work as intended, and to identify potential security issues within the smart contract.

The information in this report should be used to understand the risks associated with the smart contract. This report can be used as a guide for the development team on how the contract could possibly be improved by remediating the issues that were identified.

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

# Global Overview

## Manual Code Review

In this audit report we will highlight the following issues:

Vulnerability Level	Total	Pending	Acknowledged	Resolved
<span style="color: #00A0C0;">●</span> Informational	0	0	0	0
<span style="color: #00C000;">●</span> Low-Risk	2	0	2	0
<span style="color: #C0C000;">●</span> Medium-Risk	0	0	0	0
<span style="color: #C00000;">●</span> High-Risk	0	0	0	0

## Centralization Risks

Coinsult checked the following privileges:

Contract Privilege	Description
Owner can mint?	<span style="color: #00C000;">●</span> Owner cannot mint new tokens
Owner can blacklist?	<span style="color: #00C000;">●</span> Owner cannot blacklist addresses
Owner can set fees > 25%?	<span style="color: #00C000;">●</span> Owner cannot set the sell fee to 25% or higher
Owner can exclude from fees?	<span style="color: #00A0C0;">●</span> Owner can exclude from fees
Owner can pause trading?	<span style="color: #00C000;">●</span> Owner cannot pause the contract
Owner can set Max TX amount?	<span style="color: #00C000;">●</span> Owner cannot set max transaction amount

More owner privileges are listed later in the report.



Error Code	Description
SLT: 078	Conformance to numeric notation best practices

● **Low-Risk:** Could be fixed, will not bring problems.

### Too many digits

Literals with many digits are difficult to read and review.

```
function updateGasForProcessing(uint256 newValue) public onlyOwner {
    require(newValue >= 200000 && newValue <= 500000, "gasForProcessing must be between 200000 and 500000");
    require(newValue != gasForProcessing, "Cannot update gasForProcessing to same value");
    emit GasForProcessingUpdated(newValue, gasForProcessing);
    gasForProcessing = newValue;
}
```

### Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

### Exploit scenario

```
contract MyContract{
    uint 1_ether = 1000000000000000000;
}
```

While `1_ether` looks like 1 ether, it is 10 ether. As a result, it's likely to be used incorrectly.

Error Code	Description
SWC-104	CWE-252: Unchecked Return Value

**Low-Risk:** Could be fixed, will not bring problems.

## Unchecked transfer

The return value of an external transfer/transferFrom call is not checked.

```
function claimStuckTokens(address token) external onlyOwner {
    require(token != address(this), "Owner cannot claim native tokens");
    if (token == address(0x0)) {
        payable(msg.sender).transfer(address(this).balance);
        return;
    }
    IERC20 ERC20token = IERC20(token);
    uint256 balance = ERC20token.balanceOf(address(this));
    ERC20token.transfer(msg.sender, balance);
}
```

## Recommendation

Use SafeERC20, or ensure that the transfer/transferFrom return value is checked.

## Exploit scenario

```
contract Token {
    function transferFrom(address _from, address _to, uint256 _value) public returns (bool success);
}
contract MyBank{
    mapping(address => uint) balances;
    Token token;
    function deposit(uint amount) public{
        token.transferFrom(msg.sender, address(this), amount);
        balances[msg.sender] += amount;
    }
}
```

Several tokens do not revert in case of failure and return false. If one of these tokens is used in MyBank, deposit will not revert if the transfer fails, and an attacker can call deposit for free..

# Contract Privileges

## Maximum Fee Limit Check

Error Code	Description
CEN-01	Centralization: Operator Fee Manipulation

Coinsult tests if the owner of the smart contract can set the transfer, buy or sell fee to 25% or more. It is bad practice to set the fees to 25% or more, because owners can prevent healthy trading or even stop trading when the fees are set too high.

Type of fee	Description
Transfer fee	● Owner cannot set the transfer fee to 25% or higher
Buy fee	● Owner cannot set the buy fee to 25% or higher
Sell fee	● Owner cannot set the sell fee to 25% or higher

Type of fee	Description
Max transfer fee	0%
Max buy fee	8%
Max sell fee	16%

## Function

```
function updateBuyFees(uint256 _rewardFeeOnBuy) external onlyOwner {
    require(_rewardFeeOnBuy <= 8, "Reward Fee on buy cannot be more than 8%");
    rewardFeeOnBuy = _rewardFeeOnBuy;
    emit UpdateBuyFees(poolGameFeeOnBuy, rewardFeeOnBuy);
}

function updateSellFees(uint256 _poolGameFeeOnSell, uint256 _rewardFeeOnSell) external onlyOwner {
    require(_poolGameFeeOnSell <= 8, "PoolGame Fee on sell cannot be more than 8%");
    require(_rewardFeeOnSell <= 8, "Reward Fee on sell cannot be more than 8%");
    rewardFeeOnSell = _rewardFeeOnSell;
    poolGameFeeOnSell = _poolGameFeeOnSell;
    emit UpdateSellFees(_poolGameFeeOnSell, _rewardFeeOnSell);
}
```

## Contract Pausability Check

Error Code	Description
CEN-02	Centralization: Operator Pausability

Coinsult tests if the owner of the smart contract has the ability to pause the contract. If this is the case, users can no longer interact with the smart contract; users can no longer trade the token.

Privilege Check	Description
Can owner pause the contract?	<span style="color: green;">●</span> Owner cannot pause the contract

## Max Transaction Amount Check

Error Code	Description
CEN-03	Centralization: Operator Transaction Manipulation

Coinsult tests if the owner of the smart contract can set the maximum amount of a transaction. If the transaction exceeds this limit, the transaction will revert. Owners could prevent normal transactions to take place if they abuse this function.

Privilege Check	Description
Can owner set max tx amount?	<span style="color: green;">●</span> Owner cannot set max transaction amount

## Exclude From Fees Check

Error Code	Description
CEN-04	Centralization: Operator Exclusion

Coinsult tests if the owner of the smart contract can exclude addresses from paying tax fees. If the owner of the smart contract can exclude from fees, they could set high tax fees and exclude themselves from fees and benefit from 0% trading fees. However, some smart contracts require this function to exclude routers, dex, cex or other contracts / wallets from fees.

Privilege Check	Description
Can owner exclude from fees?	<input checked="" type="radio"/> Owner can exclude from fees

## Function

```
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);
}
```

## Ability To Mint Check

Error Code	Description
CEN-05	Centralization: Operator Increase Supply

Coinsult tests if the owner of the smart contract can mint new tokens. If the contract contains a mint function, we refer to the token's total supply as non-fixed, allowing the token owner to "mint" more tokens whenever they want.

A mint function in the smart contract allows minting tokens at a later stage. A method to disable minting can also be added to stop the minting process irreversibly.

Minting tokens is done by sending a transaction that creates new tokens inside of the token smart contract. With the help of the smart contract function, an unlimited number of tokens can be created without spending additional energy or money.

Privilege Check	Description
Can owner mint?	<span style="color: green;">●</span> Owner cannot mint new tokens

## Ability To Blacklist Check

Error Code	Description
CEN-06	Centralization: Operator Dissalows Wallets

Coinsult tests if the owner of the smart contract can blacklist accounts from interacting with the smart contract. Blacklisting methods allow the contract owner to enter wallet addresses which are not allowed to interact with the smart contract.

This method can be abused by token owners to prevent certain / all holders from trading the token. However, blacklists might be good for tokens that want to rule out certain addresses from interacting with a smart contract.

Privilege Check	Description
Can owner blacklist?	● Owner cannot blacklist addresses



## Other Owner Privileges Check

Error Code	Description
CEN-100	Centralization: Operator Privileges

Coinsult lists all important contract methods which the owner can interact with.

- ⚠ Owner can exclude from dividends
- ⚠ Owner can update claimwait without constraints
- ⚠ Owner can update minimum token balance for dividends without constraints

# Notes

## Notes by Sport Fantasy

No notes provided by the team.

## Notes by Coinsult

No notes provided by Coinsult

# Contract Snapshot

This is how the constructor of the contract looked at the time of auditing the smart contract.

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

    uint256 public rewardFeeOnBuy = 2;
    uint256 public rewardFeeOnSell = 2;

    uint256 public poolGameFeeOnBuy = 0;
    uint256 public poolGameFeeOnSell = 2;

    address public poolGameWallet = 0x2156A86745520aE3F81E2168779fAe7f4aB108f8;
    address public DevAndMarketingWallet = 0xc4EA293b0C80710b3b1e31F47435305Ff671CbcE;

    IUniswapV2Router02 public uniswapV2Router;
    address public uniswapV2Pair;

    address private DEAD = 0x0000000000000000000000000000000000000000000000000000000000000000dEaD;

    bool private swapping;
    uint256 public swapTokensAtAmount;
    bool public swapEnabled;

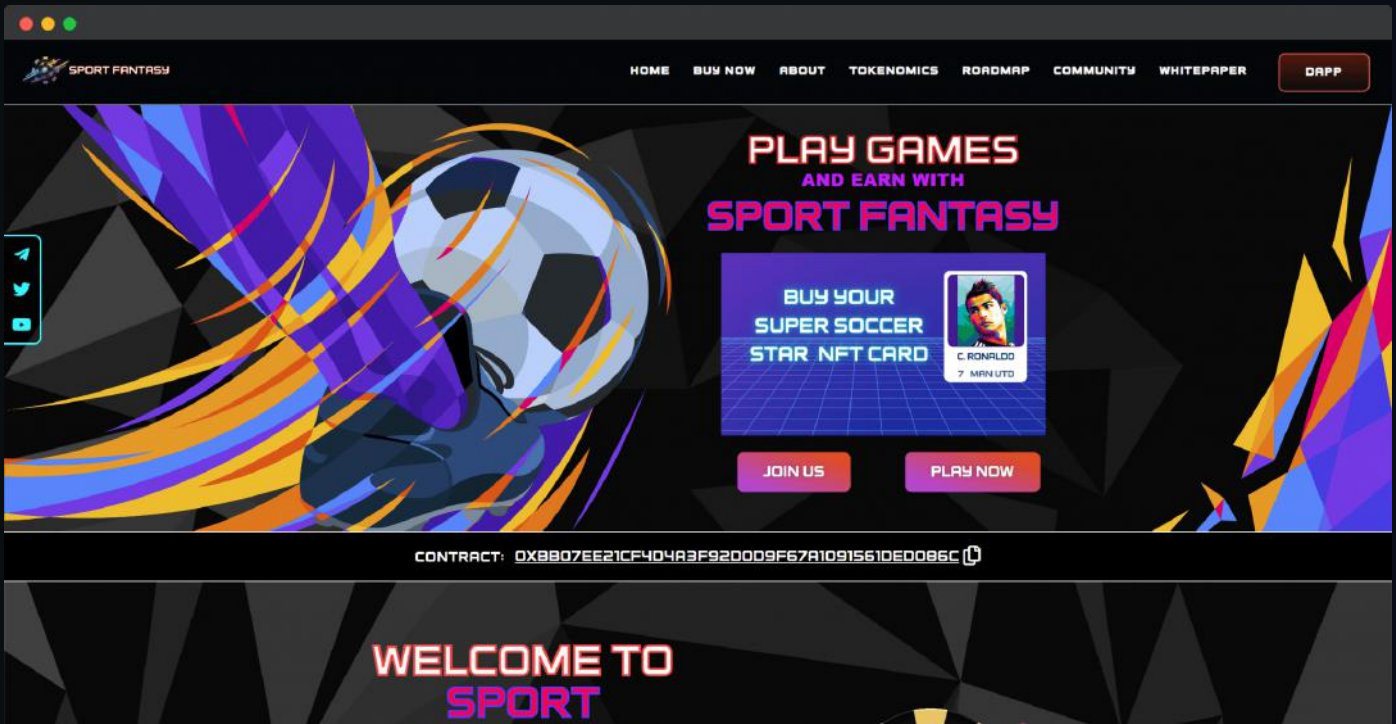
    mapping (address => bool) private _isExcludedFromFees;
    mapping (address => bool) public automatedMarketMakerPairs;

    DividendTracker public dividendTracker;
    address public rewardToken = 0xe9e7CEA3DedcA5984780Bafc599bD69ADd087D56;
    uint256 public gasForProcessing = 300000;

    event ExcludeFromFees(address indexed account, bool isExcluded);
    event UpdateBuyFees( uint256 poolGameFeeOnBuy, uint256 rewardFeeOnBuy);
    event UpdateSellFees( uint256 poolGameFeeOnSell, uint256 rewardFeeOnSell);
    event PoolGameWalletChanged(address poolGameWallet);
    event UpdateDividendTracker(address indexed newAddress, address indexed oldAddress);
    event GasForProcessingUpdated(uint256 indexed newValue, uint256 indexed oldValue);
    event SwapTokensAtAmountUpdated(uint256 swapTokensAtAmount);
    event SendDividends(uint256 amount);
```



# Website Review

Coinsult checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.



Type of check	Description
Mobile friendly?	● The website is mobile friendly
Contains jQuery errors?	● The website does not contain jQuery errors
Is SSL secured?	● The website is SSL secured
Contains spelling errors?	● The website does not contain spelling errors

# Certificate of Proof

-  KYC verified by Coinsult partner
-  Not KYC verified by Coinsult

## Sport Fantasy

Completed KYC Verification at a Coinsult partner



✓ Project Owner Identified

✓ Contract: 0x4b317c9a320f242b29a707f8507a96b54a28cc79

## Sport Fantasy

Audited by Coinsult.net



Date: 14 October 2022

✓ Advanced Manual Smart Contract Audit



Coinsult

coinsult.net

# End of report

## Smart Contract Audit



CoinsultAudits



info@coinsult.net



coinsult.net

Request your smart contract audit / KYC

[t.me/coinsult\\_tg](https://t.me/coinsult_tg)