

# expelee

---

A Secure Place For Web3

---

## SMART CONTRACT AUDIT OF Decatus Fair Launch



Contract Address

**0xDaCccbb4f2dE7ccC0916BFe5adb18aCdeF318CB8**

# Audit Summary

Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

**Audit Result:** **PASSED (Medium Risk Severity)**

**Ownership:** **NOT RENOUNCED**

**KYC Verification:** Not done till date of audit

**Audit Date:** 27/06/2022

**Audit Team:** **EXPELEE**

Be aware that smart contracts deployed on the blockchain aren't resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

# DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always Do your own research and protect yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.

# Contract Review

Contract Name	decatus
Compiler Version	v0.8.15+commit.e14f2714
Optimization	No with 200 runs
License	None license
Explorer	<a href="https://bscscan.com/address/0xDaCccbb4f2dE7ccC0916BFfe5adb18aCdeF318CB8#code">https://bscscan.com/address/0xDaCccbb4f2dE7ccC0916BFfe5adb18aCdeF318CB8#code</a>
Symbol	DECA
Decimals	18
Total Supply	550,000,000
Domain	<a href="https://decatus.io/">https://decatus.io/</a>

# Project Review

Token Name: DECA

Web Site: <https://decatus.io/>

Twitter: <https://twitter.com/DecatusGames>

Telegram: <https://t.me/decatusgames2>

Contract Address:

0xDaCccbb4f2dE7ccC0916BFe5adb18aCdeF318CB8

Platform: Binance Smart Chain

Token Type: BEP 20

Language: SOLIDITY

# Audit Methodology

The scope of this report is to audit the smart contract source code. We have scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

## Category

---

### Smart Contract Vulnerabilities

- Unhandled Exceptions
- Transaction Order Dependency
- Integer Overflow
- Unrestricted Action
- Incorrect Inheritance Order
- Typographical Errors
- Requirement Violation

### Source Code Review

- Gas Limit and Loops
- Deployment Consistency
- Repository Consistency
- Data Consistency
- Token Supply Manipulation

### Functional Assessment

- Operations Trail & Event Generation
- Assets Manipulation
- Liquidity Access

# Vulnerability Checklist

Nº	Description.	Result
1	Compiler warnings.	Passed
2	Race conditions and Re-entrancy. Cross-function raceconditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Economy model.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Zeppelin module.	Passed
21	Fallback function security.	Passed

# Manual Audit

 Low-Risk

 2 low-risk code issues found



 Low-Risk

## 1) Missing zero address validation

```
function bnbWithdraw(address payable wallet) public virtual onlyOwner {  
    wallet.transfer(address(this).balance);  
}
```

### Recommendation

Check that the address is not zero.

## 2)Unchecked transfer

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

```
function dustTransfer(IERC20 _contract ,address wallet) public virtual onlyOwner {  
    _contract.transfer(wallet,_contract.balanceOf(address(this)));  
}
```

## Recommendation

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

# Audit Summary

Compiled with solc

Number of lines: 413 (+ 0 in dependencies, + 0 in tests)

Number of assembly lines: 0

Number of contracts: 6 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 18

Number of informational issues: 19

Number of low issues: 2

ERCs: ERC20

Name	# functions	ERCs	ERC20 info	Complex code	Features
SafeMath	13			No	
decatus	44	ERC20	Pausable No Minting Approve Race Cond.	No	Send ETH Tokens interaction

# Manual Audit (Contract Function)

```
contract decatus is IERC20,Context,Pausable,Ownable{
    mapping(address => uint256) private _balances;

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

    uint256 private _totalSupply;
    string private _name;
    string private _symbol;

    constructor(string memory name_, string memory symbol_) {
        _name = name_;
        _symbol = symbol_;
        _mint( owner(),5500000000000000000000000000);
    }

    function _mint(address account, uint256 amount) internal virtual {
        require(account != address(0), "ERC20: mint to the zero address");

        _beforeTokenTransfer();

        _totalSupply += amount;
        _balances[account] += amount;
        emit Transfer(address(0), account, amount);

        _afterTokenTransfer(address(0), account, amount);
    }

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

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

    function decimals() public view virtual override returns (uint8) {
        return 18;
    }

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

## Important Points To Consider

✓ Source does not contain blacklist capability

✗ Source does not contain a pausable contract

✓ Verified contract source

✗ Token is sellable (not a honeypot) at this time

✗ Ownership renounced or source does not contain an owner contract

✗ Owner/creator wallet contains less than 10% of circulating token supply (61%)

# About Expelee


Expelee is a community driven organisation dedicated to fostering an anti-rug movement. We're here to keep investment safe from fraudsters. We've encountered several rug pulls and know how it feels to be duped, which is why we don't want anybody else to go through the same experience. We are here to raise awareness through our services so that the future of cryptocurrency can be rug-free.

The auditing process focuses to the following considerations with collaboration of an expert team:

- Functionality test of the Smart Contract to determine if proper logic has been followed throughout the whole process.
- Manually detailed examination of the code line by line by experts.
- Live test by multiple clients using Test net.
- Analysing failure preparations to check how the Smart
- Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analysing the security of the on-chain data.

## Social Media

 /Expelee

 /ExpeleeOfficial

 /expelee-co