



# SMART CONTRACT FREE AI-BASED AUDIT

FENIX COIN

Scan and check this report  
was posted at Soken Github



July, 2024

Website: [soken.io](https://soken.io)

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>Disclaimer</b>	<b>3</b>
<b>Procedure</b>	<b>4</b>
<b>Terminology</b>	<b>5</b>
<b>Limitations</b>	<b>5</b>
<b>Basic Security Recommendation</b>	<b>5</b>
<b>Token Contract Details for 03.07.2024</b>	<b>6</b>
<b>Audit Details</b>	<b>6</b>
<b>Social Profiles</b>	<b>7</b>
<b>Project Website Overview</b>	<b>7</b>
<b>Recommendations for the website</b>	<b>8</b>
<b>Token Analytics</b>	<b>9</b>
<b>FXC Token Distribution</b>	<b>9</b>
<b>Vulnerabilities checking</b>	<b>11</b>
<b>Security Issues</b>	<b>12</b>
<b>Conclusion for project owner</b>	<b>16</b>
<b>Soken Contact Info</b>	<b>17</b>

# Disclaimer

This is a comprehensive report based on our automated and manual examination of cybersecurity vulnerabilities and framework flaws of the project's smart contract.

Reading the full analysis report is essential to build your understanding of project's security level. It is crucial to take note, though we have done our best to perform this analysis and report, that you should not rely on the our research and cannot claim what it states or how we created it.

Before making any judgments, you have to conduct your own independent research.

We will discuss this in more depth in the following disclaimer - please read it fully.

**DISCLAIMER:** You agree to the terms of this disclaimer by reading this report or any portion thereof. Please stop reading this report and remove and delete any copies of this report that you download and/or print if you do not agree to these conditions. Scan and verify report's presence in the GitHub repository by a qr-code at the title page. This report is for non-reliability information only and does not represent investment advice. No one shall be entitled to depend on the report or its contents, and Soken and its affiliates shall not be held responsible to you or anyone else, nor shall Soken provide any guarantee or representation to any person with regard to the accuracy or integrity of the report.

Without any terms, warranties or other conditions other than as set forth in that exclusion and Soken excludes hereby all representations, warrants, conditions and other terms (including, without limitation, guarantees implied by the law of satisfactory quality, fitness for purposes and the use of reasonable care and skills).

The report is provided as "as is" and does not contain any terms and conditions. Except as legally banned, Soken disclaims all responsibility and responsibilities and no claim against Soken is made to any amount or type of loss or damages (without limitation, direct, indirect, special, punitive, consequential or pure economic losses or losses) that may be caused by you or any other person, or any damages or damages, including without limitations (whether innocent or negligent).

Security analysis is based only on the smart contracts. No applications or operations were reviewed for security. No product code has been reviewed.

# Procedure

## Our analysis contains following steps:

1. Project Analysis;
2. Unit Testing:
  - Smart contract functions will be unit tested on multiple parameters and under multiple conditions to ensure that all paths of functions are functioning as intended.
  - In this phase intended behaviour of smart contract is verified.
  - In this phase, we would also ensure that smart contract functions are not consuming unnecessary gas.
  - Gas limits of functions will be verified in this stage.
3. Automated Testing:
  - Mythril
  - Oyente
  - Manticore
  - Solgraph
4. Testing code with artificial intelligence

# Terminology

**We categorize the finding into 4 categories based on their vulnerability:**

- Low-severity issue — less important, must be analyzed
- Medium-severity issue — important, needs to be analyzed and fixed
- High-severity issue — important, might cause vulnerabilities, must be analyzed and fixed
- Critical-severity issue — serious bug causes, must be analyzed and fixed.

## Limitations

The security audit of Smart Contract cannot cover all vulnerabilities. Even if no vulnerabilities are detected in the audit, there is no guarantee that future smart contracts are safe. Smart contracts are in most cases safeguarded against specific sorts of attacks. In order to find as many flaws as possible, we carried out a comprehensive smart contract audit. Audit is a document that is not legally binding and guarantees nothing.

## Basic Security Recommendation

Unlike hardware and paper wallets, hot wallets are connected to the internet and store private keys online, which exposes them to greater risk. If a company or an individual holds significant amounts of cryptocurrency in a hot wallet, they should consider using MultiSig addresses. Wallet security is enhanced when private keys are stored in different locations and are not controlled by a single entity.

# Token Contract Details for 03.07.2024

Contract Name: **FENIXCOIN**

Deployed address: **0x8691BFe12e932C0d82ef59F37c7c0b772AEFd27D**

Total Supply: **420,000,000,000,000,000**

Token Tracker: **FXC**

Decimals: **9**

Token holders: **145**

Transactions count: **2,442**

Top 100 holders dominance: **100.00%**

## Audit Details



Project Name: **FENIX COIN**

Language: **Solidity**

Compiler Version: **v0.8.19**

Blockchain: **BSC**

# Social Profiles

Project Website: <https://fxcx1000.website/>

Project Twitter: <https://x.com/Fenix100000X>

Project Telegram: <https://t.me/+AX8Cf7txrIRkMzdh>

## Project Website Overview



- ✓ JavaScript errors hasn't been found.
- ✓ Malware pop-up windows hasn't been detected.
- ✓ No issues with loading elements, code, or stylesheets.

# Recommendations for the website

The **FENIX COIN** project website lacks essential legal documents. This omission poses significant risks, including:

## 1. Jurisdictional Blocking:

- Absence of legal documentation may lead to the website being blocked in various jurisdictions due to non-compliance with local regulations.

## 2. Lack of Protection Against Claims:

- Without proper legal documentation, the project owner is vulnerable to potential claims from unscrupulous users. This could result in legal disputes and financial liabilities.

## Recommendations:

### 1. Implement Essential Legal Documents:

- **Terms of Use:** Define the rules and guidelines for using the website and services.
- **Privacy Policy:** Outline how user data is collected, used, and protected.
- **Disclaimer:** Clarify the limitations of the project's liability.
- **Risk Disclosure:** Inform users about the risks associated with using the platform and investing in cryptocurrencies.
- **Anti-Money Laundering (AML) and Know Your Customer (KYC) Policies:** Ensure compliance with regulatory requirements to prevent illicit activities.

### 2. Regular Legal Audits:

- Conduct periodic legal audits to ensure ongoing compliance with international laws and regulations.

### 3. Consult Legal Experts:

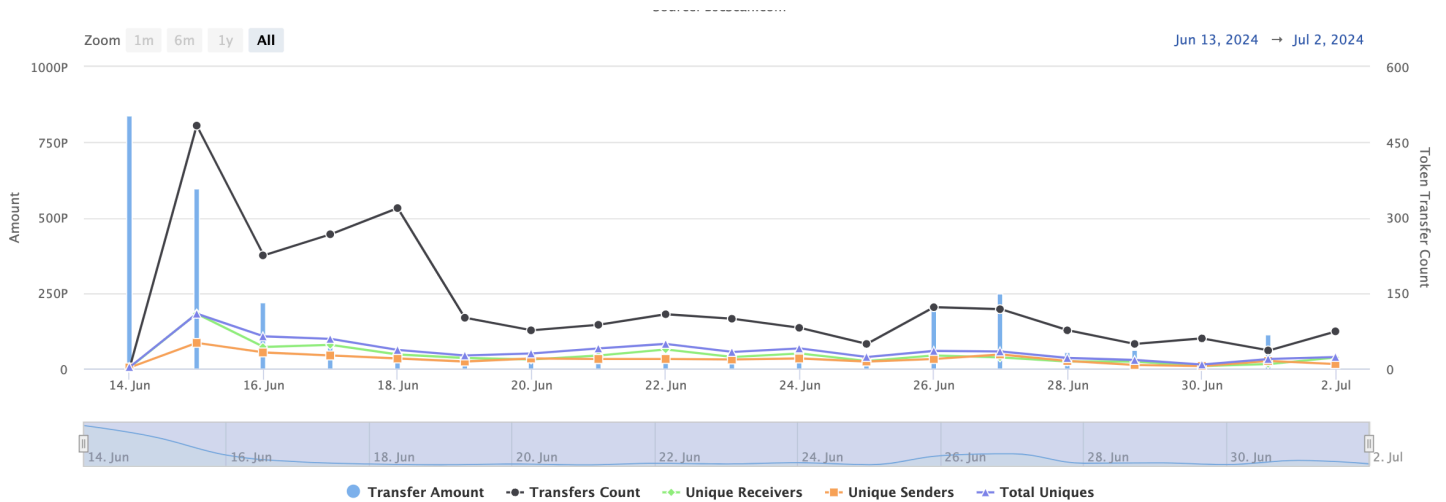
- Engage with legal professionals specializing in cryptocurrency and blockchain to draft and review the necessary legal documents.

By addressing these issues, the **FENIX COIN** project can enhance its legal standing, ensure compliance with various jurisdictions, and protect itself from potential legal challenges.

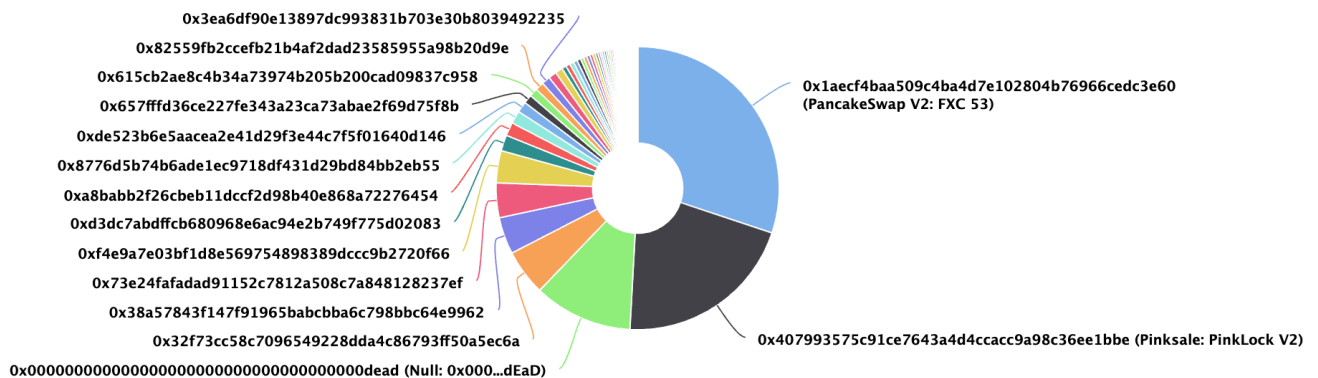


This report highlights the critical need for robust legal documentation on the **FENIX COIN** project website to safeguard against jurisdictional blocking and protect the project owner from possible claims.











## Token Analytics



## FXC Token Distribution



# FXC Top 10 Holders

1	<a href="#">PancakeSwap V2: FXC 53</a> 	126,552,606,068,822,714.474995568	30.1316%
2	<a href="#">Pinksale: PinkLock V2</a> 	87,168,932,845,240,978.299791142	20.7545%
3	<a href="#">Null: 0x000...dEaD</a> 	47,416,696,399,418,881.045591126	11.2897%
4	<a href="#">0x32f73Cc5...F50a5Ec6A</a> 	22,041,946,956,122,544.442508109	5.2481%
5	<a href="#">0x38A57843...Bc64E9962</a> 	17,774,786,414,888,054.162998652	4.2321%
6	<a href="#">0x73e24Faf...8128237EF</a> 	16,615,136,587,632,943.305590301	3.9560%
7	<a href="#">0xf4E9A7e0...9b2720F66</a> 	15,457,184,626,640,848.94569394	3.6803%
8	<a href="#">0xD3DC7ABd...775d02083</a> 	7,575,458,140,136,753.992668941	1.8037%
9	<a href="#">0xa8bAbb2f...A72276454</a> 	6,510,710,340,285,422.624213386	1.5502%
10	<a href="#">0x8776d5b7...84bb2eb55</a> 	6,159,355,322,528,809.112104548	1.4665%

## Vulnerabilities checking

Issue Description	Checking Status
Compiler Errors	Completed
Delays in Data Delivery	Completed
Re-entrancy	Completed
Transaction-Ordering Dependence	Completed
Timestamp Dependence	Completed
Shadowing State Variables	Completed
DoS with Failed Call	Completed
DoS with Block Gas Limit	Completed
Outdated Compiler Version	Completed
Assert Violation	Completed
Use of Deprecated Solidity Functions	Completed
Integer Overflow and Underflow	Completed
Function Default Visibility	Completed
Malicious Event Log	Completed
Math Accuracy	Completed
Design Logic	Completed
Fallback Function Security	Completed
Cross-function Race Conditions	Completed
Safe Zeppelin Module	Completed

# Security Issues

## 2) INCORRECT ACCESS CONTRO

**Critical**

**L248-L251, L253-L256, L265-L271**

### Description

Access control plays an important role in the segregation of privileges in smart contracts and other applications. If this is misconfigured or not properly validated on sensitive functions, it may lead to loss of funds, tokens and in some cases compromise of the smart contract.

The contract is importing an access control library but the function is missing the modifier.

## 2) ACCOUNT EXISTENCE CHECK FOR LOW LEVEL CALLS

**Medium**

**L427, L428.**

### Description

The low-level calls such as the `delegatecall`, `call`, or `callcode`, do not validate prior to the call if the destination account exists or not. They will always return true even if the account is non-existent, therefore, giving invalid output.

## 3) HARDCODED SLIPPAGE FOR UNISWAP

**Medium**

**L382-L387, L413-L419**

### Description

This contract was found to be using hardcoded slippage. This can lead to a sandwich attack where the attacker can track market orders and replicate

them whenever the order amount to be executed is large enough to compensate for exchange manipulation costs.

## 4) LIMITATIONS OF SOLIDITY'S TRYCATCH IN EXTERNAL CALLS

**Medium**

**L382-L387, L391 L398, L413-L421**

### Description

Solidity's `try-catch` feature aims to manage errors during external calls within smart contracts. However, it has limitations. It won't catch errors if the target isn't a contract expecting a return value, if the return value has a different number of arguments, or if the target lacks the method being called.

## 5) PRECISION LOSS DURING DIVISION BY LARGE NUMBERS

**Medium**

**L358**

### Description

In Solidity, when dividing large numbers, precision loss can occur due to limitations in the Ethereum Virtual Machine EVM. Solidity lacks native support for decimal or fractional numbers, leading to truncation of division results to integers. This can result in inaccuracies or unexpected behaviors, especially when the numerator is not significantly larger than the denominator.

## 6) EMPTY TRY/CATCH BLOCK

**Low**

**L382-L387, L391 L398, L413 L421**

### Description

The contract was found to be defining an empty try/catch block which is not doing anything useful.

## 7) MISSING EVENTS

**Low**

**L253-L256, L258-L263, L276-L278, L403 L429**

### Description

Events are inheritable members of contracts. When you call them, they cause the arguments to be stored in the transaction's log—a special data structure in the blockchain.

These logs are associated with the address of the contract which can then be used by developers and auditors to keep track of the transactions.

The contract was found to be missing these events on the function which would make it difficult or impossible to track these transactions off-chain.

## 8) OUTDATED COMPILER VERSION

**Low**

**L7**

### Description

Using an outdated compiler version can be problematic especially if there are publicly disclosed bugs and issues that affect the current compiler version.

## 9) USE OWNABLE2STEP

**Low**

**L150**

### Description



Ownable2Step is safer than Ownable for smart contracts because the owner cannot accidentally transfer the ownership to a mistyped address. Rather than directly transferring to the new owner, the transfer only completes when the new owner accepts ownership.

## Conclusion for project owner

Critical, Medium and Low-severity issues exist within smart contracts.

NOTE: Please check the disclaimer above and note, that audit makes no statements or warranties on business model, investment attractiveness or code sustainability. Contract security report for community



## Soken Contact Info

Website: [www.soken.io](http://www.soken.io)

128 City Road, London, England, United  
Kingdom

Telegram: @soken\_support

Twitter: @soken\_team

