QuillAudits

# Audit Report
# October, 2023

For

# Table of Content

# Executive Summary

| | |
|---|---|
| **Project Name** | AiPepe Coin |
| **Project URL** | *https://aipepecoineth.net* |
| **Overview** | AiPepe is an ERC20 token contract which at deployment, a maximum supply of 9, 999, 999, 999 tokens is minted into the address deploying the contract. |
| **Audit Scope** | *https://github.com/aipepecoineth/Aipepe/blob/main/Aipepe.sol* |
| **Contracts in Scope** | AIPEPE |
| **Commit Hash** | 4ddde9de90c0f901c91b10e79a62c6912b1cfb7a |
| **Language** | Solidity |
| **Blockchain** | Ethereum |
| **Method** | Manual Analysis, Functional Testing, Automated Testing |
| **Review 1** | 28th September 2023 - 4th October 2023 |
| **Updated Code Received** | 4th October 2023 |
| **Review 2** | 5th October 2023 |
| **Fixed In** | 7dbe2ecf6cdacc43be7da1a19ee0e4847a078e9b |

# Number of Security Issues per Severity


High
Medium
Low
Informational

| | High | Medium | Low | Informational |
|---|---|---|---|---|
| Open Issues | 0 | 0 | 0 | 0 |
| Acknowledged Issues | 0 | 0 | 0 | 0 |
| Partially Resolved Issues | 0 | 0 | 0 | 0 |
| Resolved Issues | 0 | 0 | 0 | 1 |

# Checked Vulnerabilities

- Access Management
- Arbitrary write to storage
- Centralization of control
- Ether theft
- Improper or missing events
- Logical issues and flaws
- Arithmetic Correctness
- Race conditions/front running
- SWC Registry
- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Malicious libraries
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- ERC's conformance
- Dangerous strict equalities
- Tautology or contradiction
- Return values of low-level calls
- Missing Zero Address Validation
- Private modifier
- Revert/require functions
- Multiple Sends
- Using suicide
- Using delegatecall
- Upgradeable safety
- Using throw

# Checked Vulnerabilities

- Using inline assembly
- Style guide violation
- Unsafe type inference
- Implicit visibility level

# Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behaviour.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

### Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

### Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

### Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

### Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

### Tools and Platforms used for Audit

Remix IDE, Truffle, Solhint, Mythril, Slither, Solidity statistic analysis.

## Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

### High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

### Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

### Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

### Informational

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

## Types of Issues

### Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

### Resolved

These are the issues identified in the initial audit and have been successfully fixed.

### Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

### Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

# A. Contract - AiPepe

## High Severity Issues

No issues were found.

## Medium Severity Issues

No issues were found.

## Low Severity Issues

No issues were found.

## Informational Issues

### A.1: Unused Inherited Library

**Description**

The token contract inherits the Ownable contract imported from Openzeppelin library but none of its properties was used. After the token contract is deployed, no specific function would be a privileged function because the onlyOwner modifier from the Ownable contract was not used in any. However this will also form the bytecode.

**Remediation**

Remove the unused ownable library.

**Status**

Resolved

# General Recommendation

After scrutinizing the contract implementation and the whitepaper of the project, we realized that the contract does not have the burning features because there are no specific functions designed for it, despite being mentioned on the white paper. Clients clarified that the model of burning tokens would be done by any holder sending to the 0xdead address.

# Functional Tests

**Some of the tests performed are mentioned below:**

- Should get the name of the token
- Should get the symbol of the token
- Should get the decimal of the token
- Should get the total supply of the token when deployed
- Should get balance of the owner when contract is deployed
- Should transfer tokens to other address
- Should approve another account to spend token
- Should access the balance of allowance allowed to an address
- Should increase the balance of allowance allowed to an address
- Should decrease the balance of allowance allowed to an address

## Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

## Closing Summary

In this report, we have considered the security of AiPepe Coin. We performed our audit according to the procedure described above.

## Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in AiPepe Coin smart contracts. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of AiPepe Coin smart contracts. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of the AiPepe Coin to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

# About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.

**850+**
Audits Completed

**$30B**
Secured

**$30B**
Lines of Code Audited

## Follow Our Journey

# Audit Report
# October, 2023

For

QuillAudits


Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com