FonAI

Smart Contract Audit Report

SYSFIXED

11 Feb 2023

TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
FonAI	FAI	Arbitrum

Addresses

Contract address	0xca848be669070a6a429a400edf2af8b95008951c
Contract deployer address	0xCa848be669070A6a429a400EdF2Af8B95008951c

Project Website

https://fonai.app/

Codebase

https://arbiscan.io/address/0xca848be669070a6a429a400edf2af8b95008951c#code

SUMMARY

FonAI is an Artificial Intelligence robot on blockchain that generates digital NFTs from the user's description starting on Arbitrum.

Contract Summary

Documentation Quality

FonAI provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by FonAI with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 475, 507, 530, 531, 566, 602, 668, 672, 684, 691, 700, 924, 924, 925, 930, 930, 931, 931, 932, 932, 1000, 1000, 1001, 1001, 1002, 1002, 1007, 1054, 1062, 1108, 1140, 1141, 1152 and 1152.
- SWC-110 SWC-123 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 1118 and 1119.

CONCLUSION

We have audited the FonAI project released in January 2023 to discover issues and identify potential security vulnerabilities in FonAI Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides satisfactory results with low-risk issues.

The issues found in the FonAI smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues and out-of-bounds array access. The index access expression can cause an exception in case of an invalid array index value.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	PASS
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	PASS
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	PASS
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	PASS
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Friday Feb 10 2023 12:22:11 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Feb 11 2023 04:02:18 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	FonAI.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 475

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
474   function add(uint256 a, uint256 b) internal pure returns (uint256) {
475   uint256 c = a + b;
476   require(c >= a, "SafeMath: addition overflow");
477
478   return c;
479
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 507

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
506   require(b <= a, errorMessage);
507   uint256 c = a - b;
508
509   return c;
510   }
511
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 530

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
529
530   uint256 c = a * b;
531   require(c / a == b, "SafeMath: multiplication overflow");
532
533   return c;
534
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 531

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
530   uint256 c = a * b;
531   require(c / a == b, "SafeMath: multiplication overflow");
532
533   return c;
534   }
535
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 566

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
565   require(b > 0, errorMessage);
566   uint256 c = a / b;
567   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
568
569   return c;
570
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 602

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
601   require(b != 0, errorMessage);
602   return a % b;
603   }
604   }
605
606
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 668

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
667   function mul(int256 a, int256 b) internal pure returns (int256) {
668   int256 c = a * b;
669
670   // Detect overflow when multiplying MIN_INT256 with -1
671   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
672
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 672

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
671   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
672   require((b == 0) || (c / b == a));
673   return c;
674   }
675
676
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 684

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
683   // Solidity already throws when dividing by 0.
684   return a / b;
685   }
686
687   /**
688
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 691

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
690   function sub(int256 a, int256 b) internal pure returns (int256) {
691   int256 c = a - b;
692   require((b >= 0 && c <= a) || (b < 0 && c > a));
693   return c;
694   }
695
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 700

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
699   function add(int256 a, int256 b) internal pure returns (int256) {
700   int256 c = a + b;
701   require((b >= 0 && c >= a) || (b < 0 && c < a));
702   return c;
703   }
704
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 924

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
923
924   uint256 totalSupply = 1 * 1e6 * 1e5;
925   supply += totalSupply;
926
927   walletDigit = 2;
928
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 924

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
923
924   uint256 totalSupply = 1 * 1e6 * 1e5;
925   supply += totalSupply;
926
927   walletDigit = 2;
928
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 925

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
924   uint256 totalSupply = 1 * 1e6 * 1e5;
925   supply += totalSupply;
926
927   walletDigit = 2;
928   transDigit = 1;
929
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 930

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
929
930   maxTransactionAmount = supply * transDigit / 100;
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 930

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
929
930   maxTransactionAmount = supply * transDigit / 100;
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 931

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
930   maxTransactionAmount = supply * transDigit / 100;
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934   buyTotalFees = 16;
935
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 931

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
930   maxTransactionAmount = supply * transDigit / 100;
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934   buyTotalFees = 16;
935
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 932

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934   buyTotalFees = 16;
935   sellTotalFees = 25;
936
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 932

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
931   maxWallet = supply * walletDigit / 100;
932   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
933
934   buyTotalFees = 16;
935   sellTotalFees = 25;
936
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1000

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
999   function updateLimits() private {
1000   maxTransactionAmount = supply * transDigit / 100;
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1000

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
999   function updateLimits() private {
1000   maxTransactionAmount = supply * transDigit / 100;
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1001

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1000   maxTransactionAmount = supply * transDigit / 100;
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
1005
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1001

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1000   maxTransactionAmount = supply * transDigit / 100;
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
1005
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1002

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
1005
1006
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1002

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1001   swapTokensAtAmount = supply * 5 / 10000; // 0.05% swap wallet;
1002   maxWallet = supply * walletDigit / 100;
1003   }
1004
1005
1006
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1007

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1006   function updateLimits(uint256 _amount) public onlyOwner {
1007   swapTokensAtAmount = _amount * 1e5; // 0.05% swap wallet;
1008   }
1009
1010   function setAutomatedMarketMakerPair(address pair, bool value) public onlyOwner {
1011
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1054

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1053   require(amount <= maxTransactionAmount, "Buy transfer amount exceeds the
maxTransactionAmount.");
1054   require(amount + balanceOf(to) <= maxWallet, "Max wallet exceeded");
1055   }
1056
1057   //when sell
1058
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1062

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1061   else if(!_isExcludedMaxTransactionAmount[to]){
1062   require(amount + balanceOf(to) <= maxWallet, "Max wallet exceeded");
1063   }
1064   }
1065   }
1066
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1108

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1107
1108   amount -= fees;
1109   }
1110
1111   super._transfer(from, to, amount);
1112
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1140

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1139
1140   if(contractBalance > swapTokensAtAmount * 20){
1141   contractBalance = swapTokensAtAmount * 20;
1142   }
1143
1144
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1141

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1140   if(contractBalance > swapTokensAtAmount * 20){
1141   contractBalance = swapTokensAtAmount * 20;
1142   }
1143
1144   swapTokensForEth(contractBalance);
1145
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1152

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1151   payable(adr).transfer(
1152   (amountWDOGE * amountPercentage) / 100
1153   );
1154   }
1155
1156
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1152

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- FonAI.sol

Locations

```
1151   payable(adr).transfer(
1152   (amountWDOGE * amountPercentage) / 100
1153   );
1154   }
1155
1156
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1118

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- FonAI.sol

Locations

```
1117   address[] memory path = new address[](2);
1118   path[0] = address(this);
1119   path[1] = uniswapV2Router.WETH();
1120
1121   _approve(address(this), address(uniswapV2Router), tokenAmount);
1122
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1119

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- FonAI.sol

Locations

```
1118   path[0] = address(this);
1119   path[1] = uniswapV2Router.WETH();
1120
1121   _approve(address(this), address(uniswapV2Router), tokenAmount);
1122
1123
```

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to, or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without Sysfixed's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model, or legal compliance.

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.