

# Assure DeFi<sup>®</sup>

THE VERIFICATION **GOLD STANDARD**



## Security Assessment

### AlToken

Date: 18/04/2025

Audit Status: PASS

Audit Edition: Advanced

# Risk Analysis

## Vulnerability summary

Classification	Description
 High	High-level vulnerabilities can result in the loss of assets or manipulation of data.
 Medium	Medium-level vulnerabilities can be challenging to exploit, but they still have a considerable impact on smart contract execution, such as allowing public access to critical functions.
 Low	Low-level vulnerabilities are primarily associated with outdated or unused code snippets that generally do not significantly impact execution, sometimes they can be ignored.
 Informational	Informational vulnerabilities, code style violations, and informational statements do not affect smart contract execution and can typically be disregarded.

## Executive Summary

According to the Assure assessment, the Customer's smart contract is **Secured**.



# Scope

## Target Code And Revision

For this audit, we performed research, investigation, and review of the AIToken contracts followed by issue reporting, along with mitigation and remediation instructions outlined in this report.

## Target Code And Revision

<b>Project</b>	Assure
<b>Language</b>	Solidity
<b>Codebase</b>	ERCAI.sol [SHA256]: <a href="#">cc02dcc265444cf58500d3036c0530f7bbdaea4a709aa0f81d75505639a3644a</a> ERC ERCAI_V2.sol [SHA256]: <a href="#">ee17d43b4b84db6826f0c56236ab5af16394184937ac62be523bcc327a63c2eb</a>
<b>Audit Methodology</b>	Static, Manual [excluding libraries, audit scope: from line 938]



# Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

Category	Item
Code review & Functional Review	<ul style="list-style-type: none"><li>• Compiler warnings.</li><li>• Race conditions and Reentrancy. Cross-function race conditions.</li><li>• Possible delays in data delivery.</li><li>• Oracle calls.</li><li>• Front running.</li><li>• Timestamp dependence.</li><li>• Integer Overflow and Underflow.</li><li>• DoS with Revert.</li><li>• DoS with block gas limit.</li><li>• Methods execution permissions.</li><li>• Economy model.</li><li>• Private user data leaks.</li><li>• Malicious Event log.</li><li>• Scoping and Declarations.</li><li>• Uninitialized storage pointers.</li><li>• Arithmetic accuracy.</li><li>• Design Logic.</li><li>• Cross-function race conditions.</li><li>• Safe Zeppelin module.</li><li>• Fallback function security.</li><li>• Overpowered functions / Owner privileges</li></ul>

# AUDIT OVERVIEW



## **1. Full Balance added to Liquidity [Fixed ✓]**

Function: launch

Issue: Uses address(this).balance when calling addLiquidityETH, which includes all ETH the contract holds (e.g., task deposits). The owner can appropriate funds reserved for rewards or other purposes.

Recommendation: Specify the exact ETH amount to add as liquidity rather than full balance; segregate user-funds into a separate vault or track reserved balances to prevent misuse.

Fix: Swapped to msg.value when calling addLiquidityETH.



## **1. Linear Scan in Task Completion [Partially Fixed 🟡]**

Function: completeTask()

Issue: Validates assignment by an  $O(n)$  scan of agentAssignedTasks[agentId]. With many tasks, this linear check can hit the block gas limit

Recommendation: Use a mapping taskId => assigned for  $O(1)$  lookup, or impose a cap on tasks per agent.

Fix: V2 introduces agentTaskAssignments[taskId][agentId] and checks it, but still retains the old for-loop afterward. That loop should be removed.

## **2. Unbounded Message Pagination [Fixed ✓]**

Function: getAgentMessages

Issue: Unbounded array slicing: reading count messages loops and copies each one; large count may exceed gas limits.

Recommendation: Enforce a safe maximum count per call (e.g.,  $\leq 100$ ), or paginate with fixed page sizes.

Fix: V2 enforces count = min(count, 100) before slicing.

## **3. Loop DoS in ERC20/ERC721 Transfer [Partially Fixed 🟡]**

Function: \_transferERC20WithERC721

Issue: Unbounded loops over queued NFTs and fractional-token adjustments can be forced to iterate excessively, leading to gas-limit DoS on transfers.

Recommendation: Before entering loops, require( $\text{nftsToTransfer} \leq \text{MAX\_BATCH}$ ) (and similarly for other loops) or batch the operation.

Fix: V2 adds MAX\_BATCH\_SIZE for the whole-NFT transfer loop, but the exemption-case loops (tokensToRetrieveOrMint, tokensToWithdrawAndStore) remain unchecked.



### **1. Uncapped Task Array Growth [Fixed ✓]**

Function: assignTask, completeTask

Issue: Unlimited growth of agentAssignedTasks without any cap allows endless array pushes, exacerbating DoS vectors.

Recommendation: Impose a reasonable limit on tasks per agent, or require staking to discourage spam.

Fix: V2 enforces  $\text{agentAssignedTasks}[\text{agentId}].\text{length} < \text{MAX\_TASKS\_PER\_AGENT}$  in assignTask.

### **2. Missing Reentrancy Guard on Execute [Fixed ✓]**

Function: execute()

Issue: Forwards to the external TBA's execute() without its own reentrancy guard, relying solely on the TBA for protection.

Recommendation: Mark execute as nonReentrant (in addition to using the TBA's guard) to harden reentrancy resistance.

Fix: V2 marks both execute() and executeAsAgent() as nonReentrant.

### **3. Unverified Message Recipient [Fixed ✓]**

Function: sendAgentMessage

Issue: Doesn't check that toTokenId actually exists/minted, letting messages be sent to non-existent agents.

Recommendation: Add require(\_isValidTokenId(toTokenId + ID\_ENCODING\_PREFIX), "Recipient not minted").

Fix: V2 adds require(toTokenId <= minted, "Recipient token does not exist")

### **4. Uncapped Outbound Messages [Not Fixed ✗]**

Function: sendAgentMessage

Issue: No limit on how many messages an agent can send, enabling spam and DoS of storage/gas.

Recommendation: Add require(agentMessages[fromTokenId].length < MAX\_MESSAGES) to throttle per-agent messages.



## INFORMATIONAL

---

### **1. Silent Account Creation Failures [Fixed ✓]**

Function: createAccount() during the minting

Issue: Silent try {} catch {} on registry createAccount swallows all errors so failures go unnoticed, possibly leaving NFTs unaccounted.

Recommendation: Emit an event or revert on creation failure to ensure visibility and proper troubleshooting

Fix: V2 emits AccountCreationFailed(tokenId, implementation, salt) in the catch.

# Testing coverage

During the testing phase, custom use cases were written to cover all the logic of contracts. \*Check “Annexes” to see the testing code.

**AIToken contract tests:**



tests/test\_ercai.py::test\_tasks **RUNNING**

Transaction sent: **0x3b0e9194b11bbb78e3f4a1841c4208d7cfe90ed2a076998af5ad39816a11dd16**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **10**  
UniswapV2Factory.constructor confirmed Block: **12** Gas used: **2412742 (20.11%)**  
UniswapV2Factory deployed at: **0xb6286fAFd0451320ad6A8143089b216C2152c025**

Transaction sent: **0x8d0fffe33b6a8dbd41653e6d6c3736a9662efa46253f53fd0357d06dc94a50f7**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **11**  
WETH9.constructor confirmed Block: **13** Gas used: **476546 (3.97%)**  
WETH9 deployed at: **0x7a3d735ee6873f17Dbdcab1d51B604928dc10d92**

Transaction sent: **0xd74bb5f258adf92348d5e26d382cf2fd45ef280ce868123f2ae8f692421ecdce**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **12**  
UniswapV2Router02.constructor confirmed Block: **14** Gas used: **3895430 (32.46%)**  
UniswapV2Router02 deployed at: **0x2c15A315610Bfa5248E4CbCbd693320e9D8E03Cc**

Transaction sent: **0xe74b72341d7d43aec4608ea274fb68b2bd33bb7c6adc6abcefcea128cee519ed**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **13**  
ERC6551Registry.constructor confirmed Block: **15** Gas used: **276741 (2.31%)**  
ERC6551Registry deployed at: **0xe692Cf21B12e0B2717C4bF647F9768Fa58861c8b**

Transaction sent: **0x722e110a0f87163ac6097b843039aa886c0faab4b8e03651eff03482001c5169**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **14**  
ERC6551Account.constructor confirmed Block: **16** Gas used: **971011 (8.09%)**  
ERC6551Account deployed at: **0xe65A7a341978d59d40d30FC23F5014FACB4f575A**

Transaction sent: **0xb9b7f98a3266a69db2e9deec162b7cfc188c407c481e2ba726eeaae427eb4be**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **15**  
ERCAI.constructor confirmed Block: **17** Gas used: **5118568 (42.65%)**  
ERCAI deployed at: **0x30375B532345B01cB8c2AD12541b09E9Aa53A93d**

Transaction sent: **0x1c737f8a39935cbd642d5a2aacdd0be8b0a98828a619e5f563eb634a838f2c2**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **17**  
ERCAI.launch confirmed Block: **19** Gas used: **2384911 (19.87%)**

Transaction sent: **0xac062a01b5b138ab398d0a51fec77756241941b8c5703964a5962513d3790da4**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **1**  
ERCAI.transfer confirmed Block: **20** Gas used: **82097 (0.68%)**

Transaction sent: **0x37ec9a59afe4102c5cf2f36e9733f74a9c19e0f110bb6f274adbaf70765fb7d7**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **18**  
ERCAI.liveNFTs confirmed Block: **21** Gas used: **43007 (0.36%)**

Transaction sent: **0x70f684189efe016b22cb7d70ca2a88b6d2e79ac687f6ac906946c82c0fad5149**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **1**  
ERCAI.approve confirmed Block: **22** Gas used: **46520 (0.39%)**

Transaction sent: **0xcbc2c348a0b350cb71eeb37dce299d97061f072733722744465f74eac096ac22**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **0**  
ERCAI.erc20TransferFrom confirmed Block: **23** Gas used: **240030 (2.00%)**

Transaction sent: **0x5a592e4c2cdf20582e5bf011f49ac1cc62e9d1487259775d982c4295b8043b5a**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **2**  
ERCAI.createTask confirmed (**Deadline must be in the future**) Block: **24** Gas used: **28320 (0.24%)**

Transaction sent: **0x0147d4923b93ff8801dbafeb74d1f133f10d1f03b63cc79c06e1a9989ac306e9**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **3**  
ERCAI.createTask confirmed Block: **25** Gas used: **112922 (0.94%)**

Transaction sent: **0x79a4d5401baeb2d6d6254737ae1b873e3551c03cce79fb0662a09b0a615bdf4f**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **1**  
ERCAI.createTask confirmed Block: **26** Gas used: **97922 (0.82%)**

Transaction sent: **0xb1ealebc92341f24cdfbadcca1881436360777e6823244a9d0eb421d6b999020**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **4**  
ERCAI.assignTask confirmed (**Not agent owner**) Block: **27** Gas used: **22801 (0.19%)**

Transaction sent: **0x989b8b95439614a1694d5d4422dd0a9a8aaac0755f80c2c6fd8609783583e673**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **2**  
ERCAI.assignTask confirmed Block: **28** Gas used: **47706 (0.40%)**

tests/test\_ercai.py::test\_launch **RUNNING**

Transaction sent: **0x630175e2b8028df55fadb3f2e43c4b8fd480cfcbd10f45155fb85cb188b4a285**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **0**  
UniswapV2Factory.constructor confirmed Block: **1** Gas used: **2412742** (20.11%)  
UniswapV2Factory deployed at: **0x3194cBDC3dbcd3E11a07892e7bA5c3394048Cc87**

Transaction sent: **0xf441fd91b57539cf07605aeld89ac28b9d879054c215c3ae6ffceb5827765d51**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **1**  
WETH9.constructor confirmed Block: **2** Gas used: **476546** (3.97%)  
WETH9 deployed at: **0x602C71e4DAC47a042Ee7f46E0aee17F94A3bA0B6**

Transaction sent: **0xf99c34447c2dec1afc53e0e268dee0ab9253f622233d4b39a4c948d17ad8c81a**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **2**  
UniswapV2Router02.constructor confirmed Block: **3** Gas used: **3895430** (32.46%)  
UniswapV2Router02 deployed at: **0xE7eD6747FaC5360f88a2EFC03E00d25789F69291**

Transaction sent: **0x0233fce1d941af7d55f0b4297a8c940b4d421f2b7c63de92b3737e2cffda054a**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **3**  
ERC6551Registry.constructor confirmed Block: **4** Gas used: **276741** (2.31%)  
ERC6551Registry deployed at: **0x6951b5Bd815043E3F842c1b026b0Fa888Cc2DD85**

Transaction sent: **0xa2d3e349d20d0f5b96d8d974d23672c5496f5c0d571082d8abc05dbf7db1e4f8**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **4**  
ERC6551Account.constructor confirmed Block: **5** Gas used: **971011** (8.09%)  
ERC6551Account deployed at: **0xe0aA552A10d7EC8760Fc6c246D391E698a82dDf9**

Transaction sent: **0xe82818bdfa32697d35930d736880fb066135c6012f0189fe84ec4d8822f444f6**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **5**  
ERCAI.constructor confirmed Block: **6** Gas used: **5118556** (42.65%)  
ERCAI deployed at: **0x6b4BDe1086912A6Cb24ce3dB43b3466e6c72AFd3**

Transaction sent: **0x0850766a9bddd43ac79cc779ebe7367dfae69de72912197d6c10654dd28e245b**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **0**  
ERCAI.launch confirmed (**reverted**) Block: **7** Gas used: **22719** (0.19%)

Transaction sent: **0xadc77c82ec9fff9c9e3fbf56eca7d673627c731d94d3fb5181b8974a5978f784**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **6**  
ERCAI.launch confirmed (**ds-math-sub-underflow**) Block: **8** Gas used: **2267738** (18.90%)

Transaction sent: **0xb3d0e6e331f83ed0d4fef83276184f747762dc1be7ec6aedb944337663422766**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **8**  
ERCAI.launch confirmed Block: **10** Gas used: **2384961** (19.87%)

Transaction sent: **0x13b04d291a21b4dd8832c3f180cb8ed700252f2e43df32ea116d63645bb37149**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **9**  
ERCAI.launch confirmed (**Already launched**) Block: **11** Gas used: **23591** (0.20%)

tests/test\_ercai.py::test\_launch **PASSED**

```
Transaction sent: 0x989b8b95439614a1694d5d4422dd0a9a8aaac0755f80c2c6fd8609783583e673
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 2
ERCAI.assignTask confirmed Block: 28 Gas used: 47706 (0.40%)

Transaction sent: 0x227fa208871769355be817a67fdbbe9cda4f0521a89ceff9fb79a7e1c5f6a1a7
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 3
ERCAI.completeTask confirmed (Agent not assigned to task) Block: 29 Gas used: 41383 (0.34%)

Transaction sent: 0x9b664493fe472a005c6f39ddb444651306546bddd121312f2c9568134d2d32e8
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 5
ERCAI.completeTask confirmed (Not authorized) Block: 30 Gas used: 36061 (0.30%)

Transaction sent: 0x993f46144e52a5ef2adc12f13e70741726eb7005e37564db042fd55adb0ff752
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 4
ERCAI.completeTask confirmed (Not authorized) Block: 31 Gas used: 36061 (0.30%)

Transaction sent: 0x3c7797e073f3d778f82a84ab2979c3aa44348c8456c190baf14e0f2686675248
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 5
ERCAI.completeTask confirmed Block: 32 Gas used: 125628 (1.05%)

Transaction sent: 0x9052db2ff085bbb655470aef733d56dbe004c832723d0907619f92d4ef47f395
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 6
ERCAI.assignTask confirmed (Task already completed) Block: 33 Gas used: 23702 (0.20%)

Transaction sent: 0x6da96c4514b2af280cc10477a7c40a943c8d130980ec2c172321e32c6fae0248
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 7
ERCAI.completeTask confirmed (Task already completed) Block: 34 Gas used: 36941 (0.31%)

Transaction sent: 0xc9a4d6503495bae6f82669500ee0bf7c8cdf546080cbb36a12f3f9c19f41f790
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 8
ERCAI.assignTask confirmed (Task deadline passed) Block: 36 Gas used: 24542 (0.20%)

Transaction sent: 0xb54d165f68ec642debf95249f669d272cf64e0c807b98b9125f5b420c2ba91da
Gas price: 0.0 gwei Gas limit: 12000000 Nonce: 9
ERCAI.completeTask confirmed (Task deadline passed) Block: 37 Gas used: 37784 (0.31%)

tests/test_ercai.py::test_tasks PASSED
```



tests/test\_ercai.py::test\_agent\_message **RUNNING**

Transaction sent: **0x8fe76809664dd619043faad839f4a9fa47a0cea3286efe6c085ce36530361695**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **19**  
UniswapV2Factory.constructor confirmed Block: **38** Gas used: **2412742 (20.11%)**  
UniswapV2Factory deployed at: **0xDae02e4fE488952cFB8c95177154D188647a0146**

Transaction sent: **0xece75357220dc7ae93e0cc72254267fc379473053dcbadd423d63e18af504382**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **20**  
WETH9.constructor confirmed Block: **39** Gas used: **476546 (3.97%)**  
WETH9 deployed at: **0xdCF93F11ef216cEC9C07fd31dD801c9b2b39Afb4**

Transaction sent: **0x63f831a9ebdeab7673265856d3ce97ed4d8c7cdb50e914b83aba05222de233a7**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **21**  
UniswapV2Router02.constructor confirmed Block: **40** Gas used: **3895430 (32.46%)**  
UniswapV2Router02 deployed at: **0xBcb61491F1859f53438918F1A5aFCA542Af9D397**

Transaction sent: **0xe85de7fddb6ef2fd9f3fa782fb6de1e96cbb794b463215f7bc21484f941a6c5a**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **22**  
ERC6551Registry.constructor confirmed Block: **41** Gas used: **276741 (2.31%)**  
ERC6551Registry deployed at: **0xD22363efee93190f82b52FCD62B7Dbcb920eF658**

Transaction sent: **0x57efb9cd0d46b21b74ef8599d46a2523fec2265c8d79a86109b2f39f25e8ac4c**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **23**  
ERC6551Account.constructor confirmed Block: **42** Gas used: **971011 (8.09%)**  
ERC6551Account deployed at: **0x4D1B781ce59B8C184F63B99D39d6719A522f46B5**

Transaction sent: **0x6326e73139e419c2f39f0a7256bdc3b9d5d54c8eaa11bd5a5697efe10e17e5c1**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **24**  
ERCAI.constructor confirmed Block: **43** Gas used: **5118568 (42.65%)**  
ERCAI deployed at: **0xf9C8Cf55f2E520B08d869df7bc76aa3d3ddDF913**

Transaction sent: **0x9a15564e3a5cfe4404302de41ec3fff193c73a8282597808eb87aece5c1565d1**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **26**  
ERCAI.launch confirmed Block: **45** Gas used: **2384961 (19.87%)**

Transaction sent: **0xc3ed4a860fbb73c751d440b84e5acd7fd33311f8b0da1d06579ef8dc7770fc64**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **1**  
ERCAI.transfer confirmed Block: **46** Gas used: **82097 (0.68%)**

Transaction sent: **0x939f448c00f306d47f481dbbffe79637f4c3fe23f3a5808c07f46ce8c5c20c95**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **27**  
ERCAI.liveNFTs confirmed Block: **47** Gas used: **43007 (0.36%)**

Transaction sent: **0xf80ef7a9cb7365837b038e56217080c10787a05458460d7370414ab14719ed68**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **6**  
ERCAI.approve confirmed Block: **48** Gas used: **46520 (0.39%)**

Transaction sent: **0x7ff48236e5b7daa0ef936cac533c6b7d6ccd491592c45b883a4b10712c64e32d**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **10**  
ERCAI.erc20TransferFrom confirmed Block: **49** Gas used: **240030 (2.00%)**

Transaction sent: **0x00027833192363b79d0bbda4c52e4e2b34c803f083fd23261f2017846c0968a2**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **7**  
ERCAI.sendAgentMessage confirmed (**Not authorized**) Block: **50** Gas used: **30244 (0.25%)**

Transaction sent: **0xe8d791476c2fcb2b8696ace89753b97818b13cec67421059858582f68f85fb02**  
Gas price: **0.0** gwei Gas limit: **12000000** Nonce: **11**  
ERCAI.sendAgentMessage confirmed Block: **51** Gas used: **113044 (0.94%)**

tests/test\_ercai.py::test\_agent\_message **PASSED**

# Annexes

Testing code:

Testing\_AIToken:

```
from brownie import (

    reverts,

)

from scripts.helpful_scripts import (

    ZERO_ADDRESS,

    DAY_TIMESTAMP,

    get_timestamp,

    increase_timestamp,

    get_account,

    random_salt,

    get_chain_number,

)

from scripts.deploy import (

    # MOCK

    deploy_weth,

    deploy_router,

    deploy_factory,

    deploy_liquidity,

    # Contracts

    deploy_ercai

)
```

```

def test_launch(only_local):

    # Arrange

    owner = get_account(0)

    other = get_account(1)

    extra = get_account(2)


    factory = deploy_factory(owner, owner)

    weth = deploy_weth(owner)

    router = deploy_router(owner, factory.address, weth.address)

    ercai = deploy_ercai(owner, random_salt(), router.address)


    # not owner

    with reverts():

        ercai.launch(1000000, True, {"from": other})


    # not enough balance

    with reverts():

        ercai.launch(1000000, True, {"from": owner})


    owner.transfer(ercai.address, "1 ether")

    tx = ercai.launch(1000000, True, {"from": owner})

    assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS

    assert tx.events['Transfer'][0]['to'] == ercai.address

    assert tx.events['Transfer'][0]['amount'] == 1000000e18


    with reverts("Already launched"):

        ercai.launch(1000000, True, {"from": owner})


def test_tasks(only_local):

```



```
# Arrange

owner = get_account(0)

other = get_account(1)

extra = get_account(2)


factory = deploy_factory(owner, owner)

weth = deploy_weth(owner)

router = deploy_router(owner, factory.address, weth.address)

ercai = deploy_ercai(owner, random_salt(), router.address)


owner.transfer(ercai.address, "1 ether")

tx = ercai.launch(1000000, True, {"from": owner})

pair_addr = tx.events['Transfer'][1]['to']


# send some tokens

ercai.transfer(other, 2e18, {"from": pair_addr})


ercai.liveNFTs({"from": owner})

ercai.approve(extra, 2e18, {"from": other})

tx = ercai.erc20TransferFrom(other, extra, 1e18, {"from": extra})

assert tx.events['Transfer'][0]['from'] == other

assert tx.events['Transfer'][0]['to'] == extra

assert tx.events['Transfer'][0]['amount'] == 1e18


token_id = tx.events['ERC6551AccountCreated'][0]['tokenId']


with reverts("Deadline must be in the future"):

    ercai.createTask("some_description", 1, {"from": other})
```

```

tx = ercai.createTask("some_description", get_timestamp(7), {"from": other})

assert tx.events['TaskCreated'][0]['taskId'] == 0

assert tx.events['TaskCreated'][0]['requester'] == other


tx = ercai.createTask("some_description", get_timestamp(14), {"from": extra})

assert tx.events['TaskCreated'][0]['taskId'] == 1

assert tx.events['TaskCreated'][0]['requester'] == extra


with reverts("Not agent owner"):

    ercai.assignTask(0, token_id, {"from": other})

tx = ercai.assignTask(0, token_id, {"from": extra})

assert tx.events['TaskAssigned'][0]['taskId'] == 0

assert tx.events['TaskAssigned'][0]['agentId'] == token_id


with reverts("Agent not assigned to task"):

    ercai.completeTask(1, token_id, b"outcome", {"from": extra})


with reverts("Not authorized"):

    ercai.completeTask(0, token_id, b"outcome", {"from": other})


with reverts("Not authorized"):

    ercai.completeTask(0, 2, b"outcome", {"from": extra})


tx = ercai.completeTask(0, token_id, b"outcome", {"from": extra})

assert tx.events['TaskCompleted'][0]['taskId'] == 0

assert tx.events['TaskCompleted'][0]['agentId'] == token_id


with reverts("Task already completed"):

    ercai.assignTask(0, token_id, {"from": extra})

```

```

with reverts("Task already completed"):

    ercai.completeTask(0, token_id, b"outcome", {"from": extra})

increase_timestamp(DAY_TIMESTAMP * 20)

with reverts("Task deadline passed"):

    ercai.assignTask(1, token_id, {"from": extra})

with reverts("Task deadline passed"):

    ercai.completeTask(1, token_id, b"outcome", {"from": extra})

def test_agent_message(only_local):

    # Arrange

    owner = get_account(0)

    other = get_account(1)

    extra = get_account(2)

    factory = deploy_factory(owner, owner)

    weth = deploy_weth(owner)

    router = deploy_router(owner, factory.address, weth.address)

    ercai = deploy_ercai(owner, random_salt(), router.address)

    owner.transfer(ercai.address, "1 ether")

    tx = ercai.launch(1000000, True, {"from": owner})

    pair_addr = tx.events['Transfer'][1]['to']

    # send some tokens

    ercai.transfer(other, 2e18, {"from": pair_addr})

```

```
ercai.liveNFTs({"from": owner})

ercai.approve(extra, 2e18, {"from": other})

tx = ercai.erc20TransferFrom(other, extra, 1e18, {"from": extra})

assert tx.events['Transfer'][0]['from'] == other
assert tx.events['Transfer'][0]['to'] == extra
assert tx.events['Transfer'][0]['amount'] == 1e18

token_id = tx.events['ERC6551AccountCreated'][0]['tokenId']

with reverts("Not authorized"):
    ercai.sendAgentMessage(2, 3, b'message', {"from": other})

tx = ercai.sendAgentMessage(token_id, 3, b'message', {"from": extra})

assert tx.events['AgentMessageSent'][0]['fromTokenId'] == token_id
assert tx.events['AgentMessageSent'][0]['toTokenId'] == 3

messages = ercai.getAgentMessages(3, 0, 1, {"from": other})

assert len(messages) == 1
```

# Technical Findings Summary

## Findings

Vulnerability Level	Total	Not Fixed	Not Apply	Acknowledged	Partially Fixed	Fixed
<div><div></div>High</div>	1					1
<div><div></div>Medium</div>	3				2	1
<div><div></div>Low</div>	4	1				3
<div><div></div>Informational</div>	1					1

# Assessment Results

## Score Results

Review	Score
<b>Global Score</b>	<b>85/100</b>
Assure KYC	Not completed
Audit Score	85/100

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must pass and provide all the data needed for the assessment. Our Passing Score has been changed to 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below. The Global Score is a combination of the evaluations obtained between having or not having KYC and the type of contract audited together with its manual audit.

## Audit PASS

Following our comprehensive security audit of the token contract for the AIToken project, we inform you that the cybersecurity audit has failed due to multiple critical issues identified during the review, which pose significant risks to the contract's functionality and security. Immediate remediation is required to address these vulnerabilities.

After reviewing the issues, the development team applies all necessary corrective measures.



# Disclaimer

Assure Defi has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocating for the Project, and users relying on this report should not consider this as having any merit for financial AIToken in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment in AIToken, nor should it be used to signal that any person reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and Assure Defi is under no covenant to audit completeness, accuracy, or solidity of the contracts. In no event will Assure Defi or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment of AITokens provided by Assure Defi are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any AITokens, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The AIToken may access, and depend upon, multiple layers of third parties.