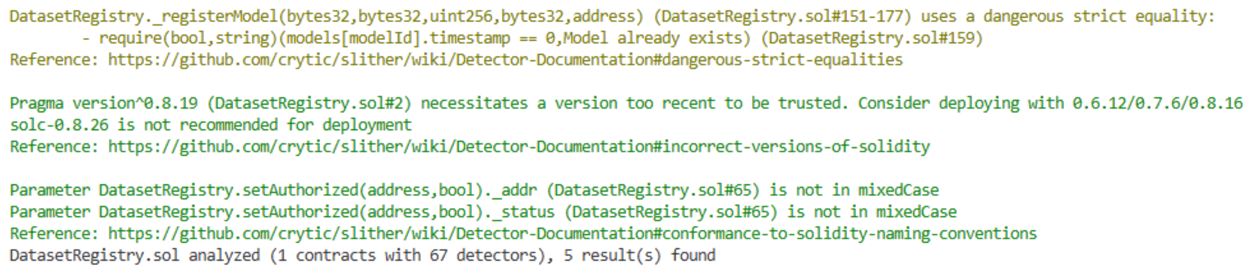
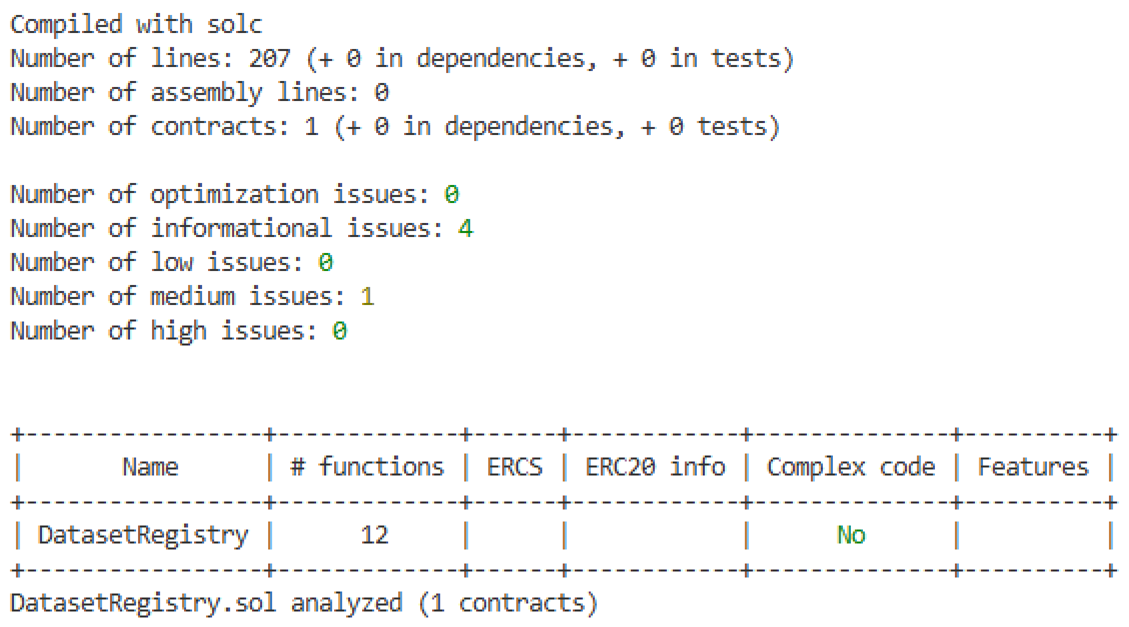
DatasetRegistry Smart Contract

1. *Existing vulnerabilities*

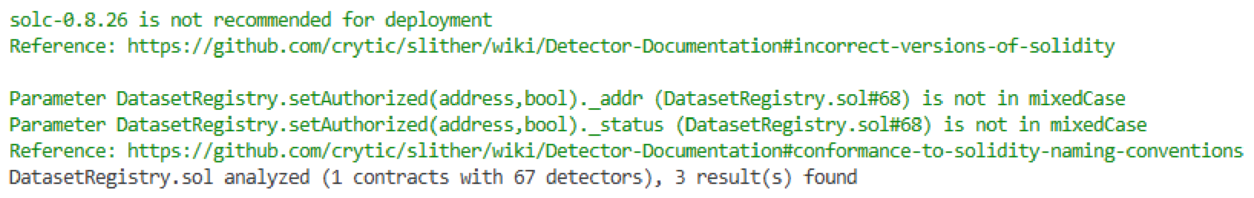


```
DatasetRegistry._registerModel(bytes32,bytes32,uint256,bytes32,address) (DatasetRegistry.sol#151-177) uses a dangerous strict equality:
        - require(bool,string)(models[modelId].timestamp == 0,Model already exists) (DatasetRegistry.sol#159)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Pragma version^0.8.19 (DatasetRegistry.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.26 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Parameter DatasetRegistry.setAuthorized(address,bool)._addr (DatasetRegistry.sol#65) is not in mixedCase
Parameter DatasetRegistry.setAuthorized(address,bool)._status (DatasetRegistry.sol#65) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
DatasetRegistry.sol analyzed (1 contracts with 67 detectors), 5 result(s) found
```

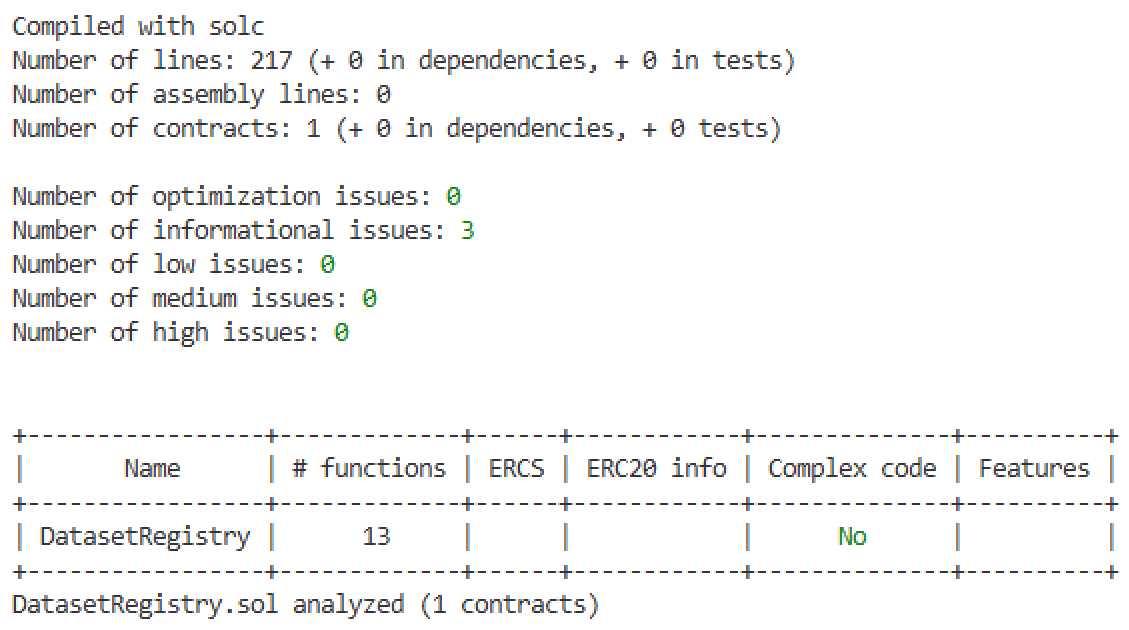


```
Compiled with solc
Number of lines: 207 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 1 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 4
Number of low issues: 0
Number of medium issues: 1
Number of high issues: 0


+-----------------+-------------+------+------------+--------------+----------+
|       Name      | # functions | ERCS | ERC20 info | Complex code | Features |
+-----------------+-------------+------+------------+--------------+----------+
| DatasetRegistry |      12     |      |            |      No      |          |
+-----------------+-------------+------+------------+--------------+----------+
DatasetRegistry.sol analyzed (1 contracts)
```

2. *Results after fixing the issues:*



```
solc-0.8.26 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Parameter DatasetRegistry.setAuthorized(address,bool)._addr (DatasetRegistry.sol#68) is not in mixedCase
Parameter DatasetRegistry.setAuthorized(address,bool)._status (DatasetRegistry.sol#68) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
DatasetRegistry.sol analyzed (1 contracts with 67 detectors), 3 result(s) found
```



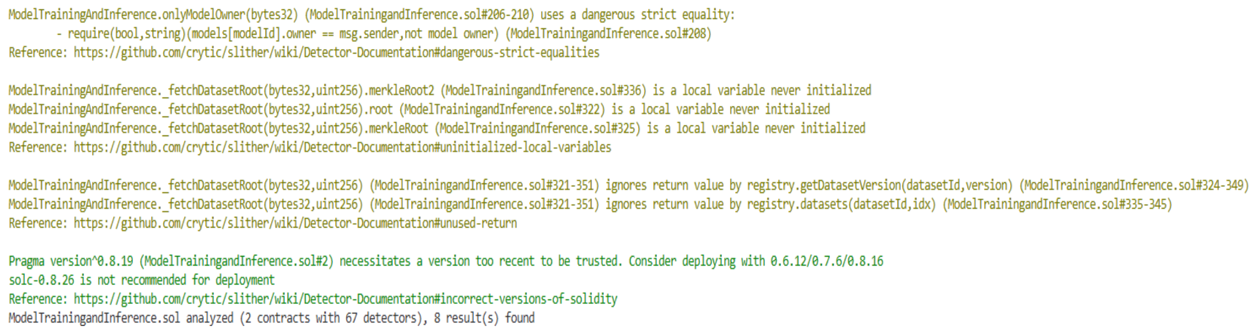
```
Compiled with solc
Number of lines: 217 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 1 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 3
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0


+-----------------+-------------+------+------------+--------------+----------+
|       Name      | # functions | ERCS | ERC20 info | Complex code | Features |
+-----------------+-------------+------+------------+--------------+----------+
| DatasetRegistry |      13     |      |            |      No      |          |
+-----------------+-------------+------+------------+--------------+----------+
DatasetRegistry.sol analyzed (1 contracts)
```

Vulnerabilities explanation: During our security analysis, we identified and addressed two key vulnerabilities in the smart contracts. First, the use of strict equality checks on uninitialized storage was flagged as dangerous by Slither, as it could introduce logical inconsistencies. To mitigate this, we replaced the equality check with a dedicated boolean existence flag, ensuring safer and more explicit validation of registered entities. Second, we resolved risks related to compiler mismatches by aligning the pragma directive with a stable and recommended version, preventing unintended behaviors from nightly or deprecated releases.

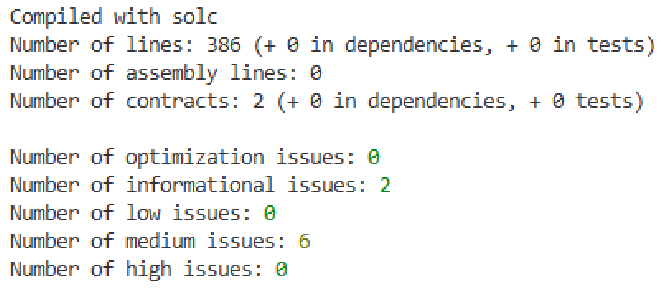
Other warnings raised during the inspection were not considered vulnerabilities that required changes. Specifically, Slither flagged the use of Solidity 0.8.26 as “not recommended for deployment.” However, this is a precautionary advisory rather than a security flaw, and our choice of compiler was retained due to compatibility and stability with the overall toolchain. Additionally, non-mixedCase parameter naming (e.g., _addr, _status) was reported as a style issue rather than a security vulnerability. Since naming conventions do not affect functionality, correctness, or security of the contract, we elected to leave them unchanged for consistency with the existing codebase.

ModelTrainingandInference Smart Contract

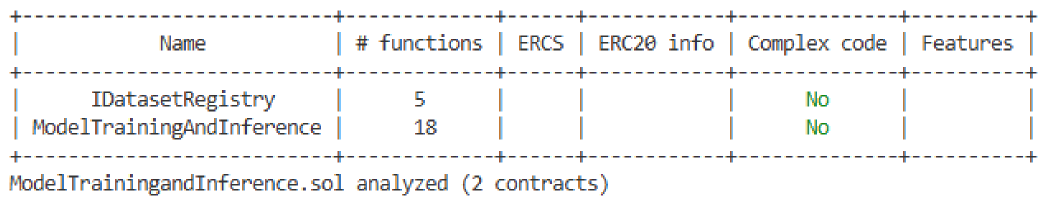
1. Existing vulnerabilities



```
ModelTrainingAndInference.onlyModelOwner(bytes32) (ModelTrainingandInference.sol#206-210) uses a dangerous strict equality:
        - require(bool,string)(models[modelId].owner == msg.sender,not model owner) (ModelTrainingandInference.sol#208)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

ModelTrainingAndInference._fetchDatasetRoot(bytes32,uint256).merkleRoot2 (ModelTrainingandInference.sol#336) is a local variable never initialized
ModelTrainingAndInference._fetchDatasetRoot(bytes32,uint256).root (ModelTrainingandInference.sol#322) is a local variable never initialized
ModelTrainingAndInference._fetchDatasetRoot(bytes32,uint256).merkleRoot (ModelTrainingandInference.sol#325) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

ModelTrainingAndInference._fetchDatasetRoot(bytes32,uint256) (ModelTrainingandInference.sol#321-351) ignores return value by registry.getDatasetVersion(datasetId,version) (ModelTrainingandInference.sol#324-349)
ModelTrainingAndInference._fetchDatasetRoot(bytes32,uint256) (ModelTrainingandInference.sol#321-351) ignores return value by registry.datasets(datasetId,idx) (ModelTrainingandInference.sol#335-345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Pragma version^0.8.19 (ModelTrainingandInference.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.26 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
ModelTrainingandInference.sol analyzed (2 contracts with 67 detectors), 8 result(s) found
```

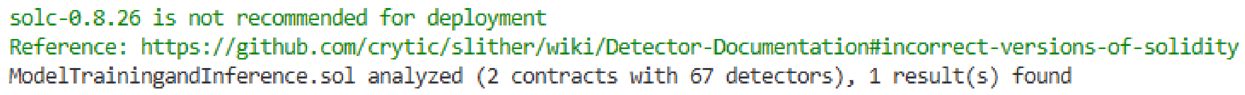


```
Compiled with solc
Number of lines: 386 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 2 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 2
Number of low issues: 0
Number of medium issues: 6
Number of high issues: 0
```

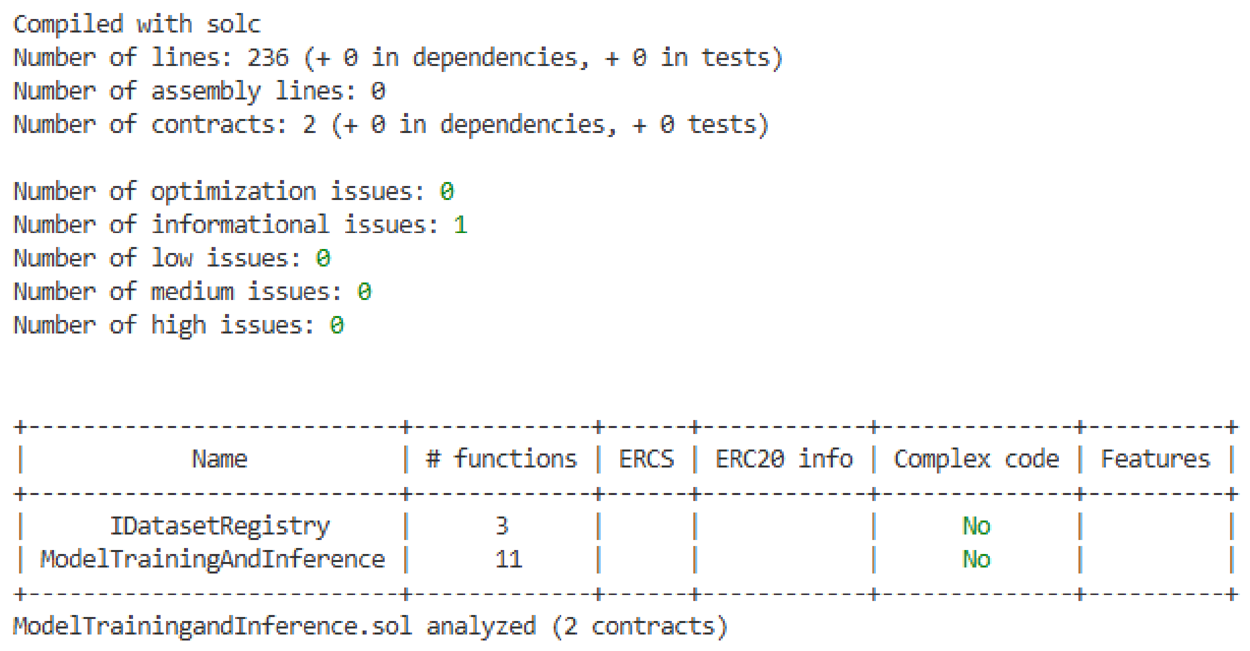


```
+---------------------------+-------------+------+------------+--------------+----------+
|            Name           | # functions | ERCS | ERC20 info | Complex code | Features |
+---------------------------+-------------+------+------------+--------------+----------+
|      IDatasetRegistry     |      5      |      |            |      No      |          |
| ModelTrainingAndInference |      18     |      |            |      No      |          |
+---------------------------+-------------+------+------------+--------------+----------+
ModelTrainingandInference.sol analyzed (2 contracts)
```

2. Results after fixing the issues:



```
solc-0.8.26 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
ModelTrainingandInference.sol analyzed (2 contracts with 67 detectors), 1 result(s) found
```

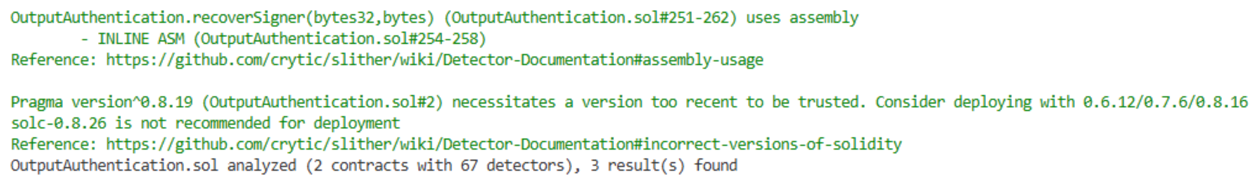


```
Compiled with solc
Number of lines: 236 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 2 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 1
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0


+---------------------------+-------------+------+------------+--------------+----------+
|            Name           | # functions | ERCS | ERC20 info | Complex code | Features |
+---------------------------+-------------+------+------------+--------------+----------+
|      IDatasetRegistry     |      3      |      |            |      No      |          |
| ModelTrainingAndInference |      11     |      |            |      No      |          |
+---------------------------+-------------+------+------------+--------------+----------+
ModelTrainingandInference.sol analyzed (2 contracts)
```

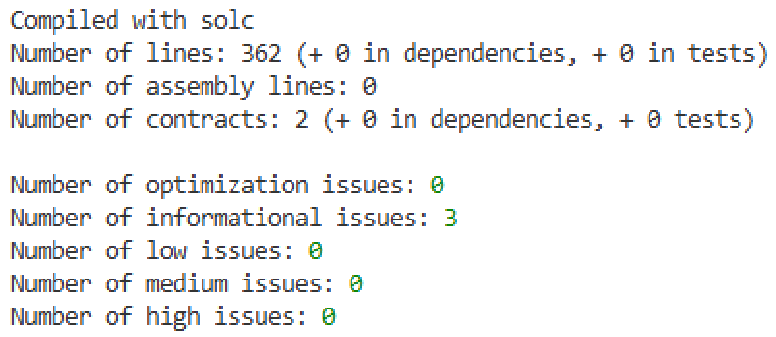
Vulnerabilities explanation: In reviewing the **ModelTrainingAndInference** contract, we identified and remediated the key security concerns. First, we replaced the dangerous strict equality in onlyModelOwner with a safer ownership check to avoid potential bypass scenarios. Next, we resolved issues with uninitialized local variables in _fetchDatasetRoot by ensuring all returned values from the registry are explicitly assigned before use, and we corrected unused return values by capturing and leveraging the data where needed. These changes eliminate the possibility of silent logic flaws or reliance on uninitialized memory. Finally, Slither continued to raise an informational warning regarding the local compiler (solc-0.8.26), which is not recommended for deployment. This does **not** indicate a vulnerability in the contract itself but rather reflects the environment where analysis was performed. To align with best practices, our target deployment uses solc-0.8.16, a stable and widely trusted version, so this warning can be safely disregarded.

OutputAuthentication Smart Contract

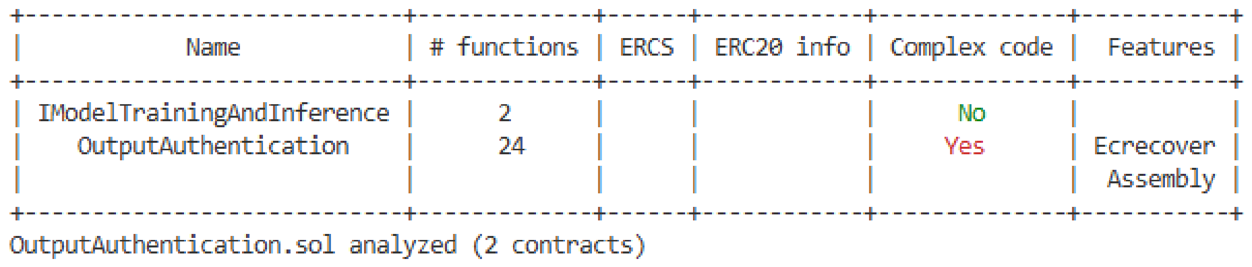
1. *Existing vulnerabilities*



```
OutputAuthentication.recoverSigner(bytes32,bytes) (OutputAuthentication.sol#251-262) uses assembly
        - INLINE ASM (OutputAuthentication.sol#254-258)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Pragma version^0.8.19 (OutputAuthentication.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.26 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
OutputAuthentication.sol analyzed (2 contracts with 67 detectors), 3 result(s) found
```



```
Compiled with solc
Number of lines: 362 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 2 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 3
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0
```



```
+----------------------------+-------------+------+------------+--------------+-----------+
|            Name            | # functions | ERCS | ERC20 info | Complex code |  Features |
+----------------------------+-------------+------+------------+--------------+-----------+
| IModelTrainingAndInference |      2      |      |            |      No      |           |
|    OutputAuthentication    |      24     |      |            |     Yes      | Ecrecover |
|                            |             |      |            |              |  Assembly |
+----------------------------+-------------+------+------------+--------------+-----------+
OutputAuthentication.sol analyzed (2 contracts)
```

Vulnerabilities explanation: In the case of the OutputAuthentication contract, the static analysis identified three issues. The first was the use of inline assembly in the recoverSigner function. While assembly can introduce risks due to bypassing compiler safety checks, its usage here is restricted to a standard and well-established pattern for signature recovery that is widely adopted in Solidity libraries such as OpenZeppelin. Because this implementation is both minimal and secure, we determined that no modification was necessary, though the decision to retain assembly was explicitly documented. The second and third issues were related to compiler versioning: the use of pragma solidity ^0.8.19 and deployment under solc-

0.8.26. These warnings were not treated as critical because the 0.8.x compiler series includes built-in safety against integer overflows and other vulnerabilities, and version 0.8.26 is a stable release actively supported by the Solidity team. Consequently, while these warnings were acknowledged, they were deemed safe to ignore in the context of our deployment environment. Overall, no unmitigated security-critical vulnerabilities remained after inspection.