School: .... Campus: ....

Academic Year: .... Subject Name: .... Subject Code: ....

Semester: .... Program: .... Branch: .... Specialization: ....

Date: ....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement :

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Contract QA – Testing Smart Contracts

Smart Contract Testing Setup

Step 1: Create Test Directory Structure

```
# Create test directory if it doesn't exist
New-Item -ItemType Directory -Force -Path test

# Create test utilities directory
New-Item -ItemType Directory -Force -Path test\utils
```

Step 2: Create Test Utilities :- File: test/utils/fixtures.js

Step 3: Create Main Test File :- File: test/Greeter.test.js

Step 4: Create Security & Integration Tests :- File: test/Greeter.security.test.js

Step 5: Create Performance Tests :- File: test/Greeter.performance.test.js

Step 6: Update Hardhat Config for Better Testing :- Update hardhat.config.js:

Step 7: Run the Comprehensive Tests

```
# Run all tests
npx hardhat test

# Run specific test files
npx hardhat test test/Greeter.test.js
npx hardhat test test/Greeter.security.test.js
npx hardhat test test/Greeter.performance.test.js

# Run tests with gas reporting
REPORT_GAS=true npx hardhat test

# Run tests with detailed output
npx hardhat test --verbose
```

Step 8: Create Test Script :- File: scripts/run-tests.js

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

Coding Phase: Pseudo Code / Flow Chart / Algorithm

Test Categories Covered

1. Unit Tests
- Deployment verification
- Functionality testing
- Return value validation

2. Integration Tests
- Multi-contract interactions
- Cross-function calls
- State persistence

3. Security Tests
- Input validation
- Access control
- Edge case handling

4. Performance Tests
- Gas optimization
- Transaction throughput
- Resource usage

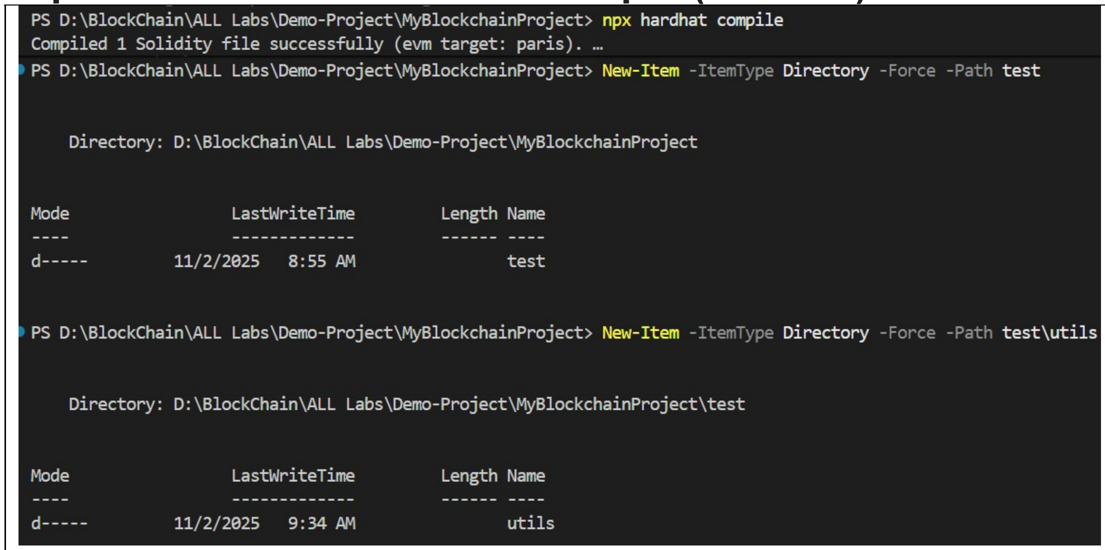
5. Negative Tests
- Error conditions
- Invalid inputs
- Boundary cases

* Softwares used

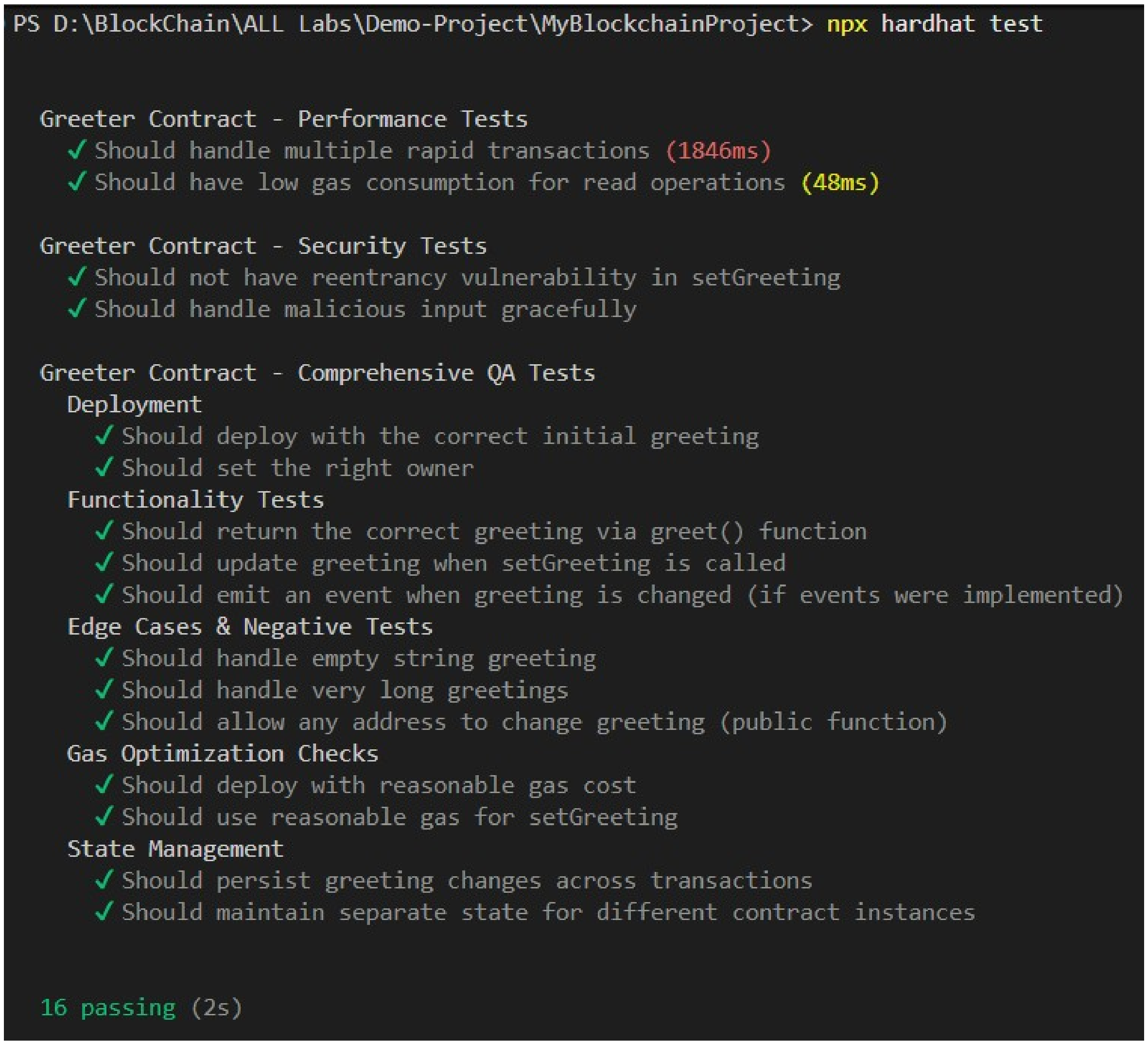
- **Hardhat**
- **Solidity**
- **VS Code**
- **JavaScript**

******As applicable according to the experiment.*
Two sheets per experiment (10-20) to be used.

* **Implementation Phase: Final Output (no error)**



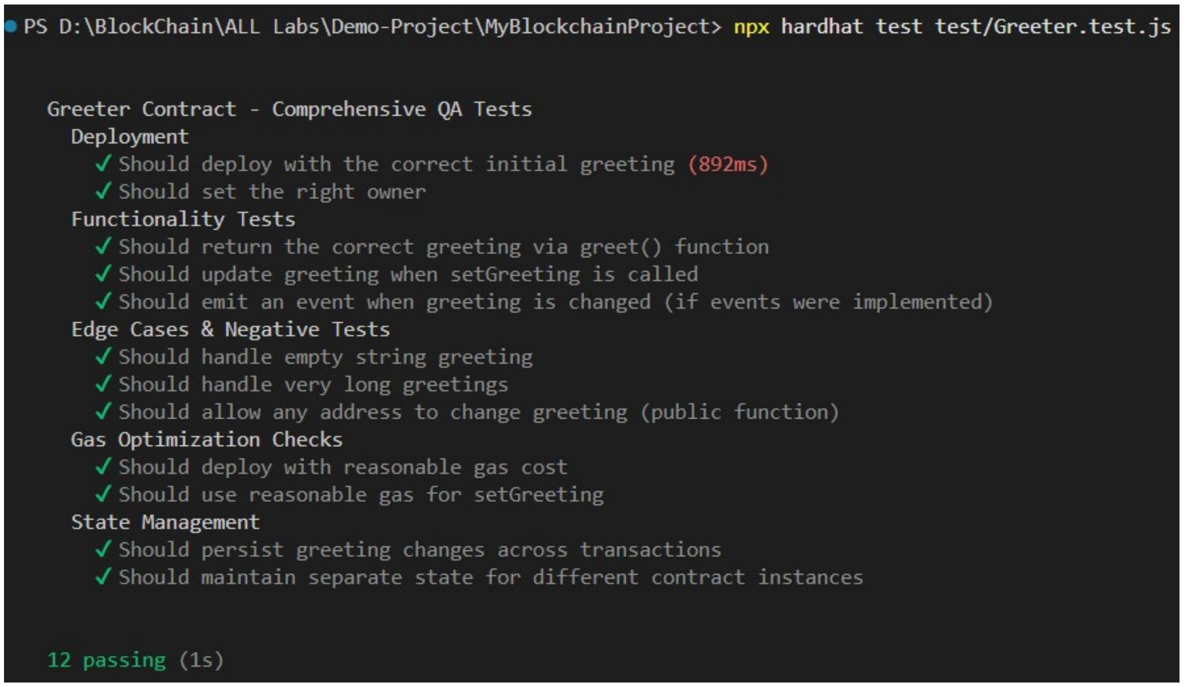
```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat compile
Compiled 1 Solidity file successfully (evm target: paris). …
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> New-Item -ItemType Directory -Force -Path test


    Directory: D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        11/2/2025   8:55 AM                test


PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> New-Item -ItemType Directory -Force -Path test\utils


    Directory: D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject\test


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        11/2/2025   9:34 AM                utils
```



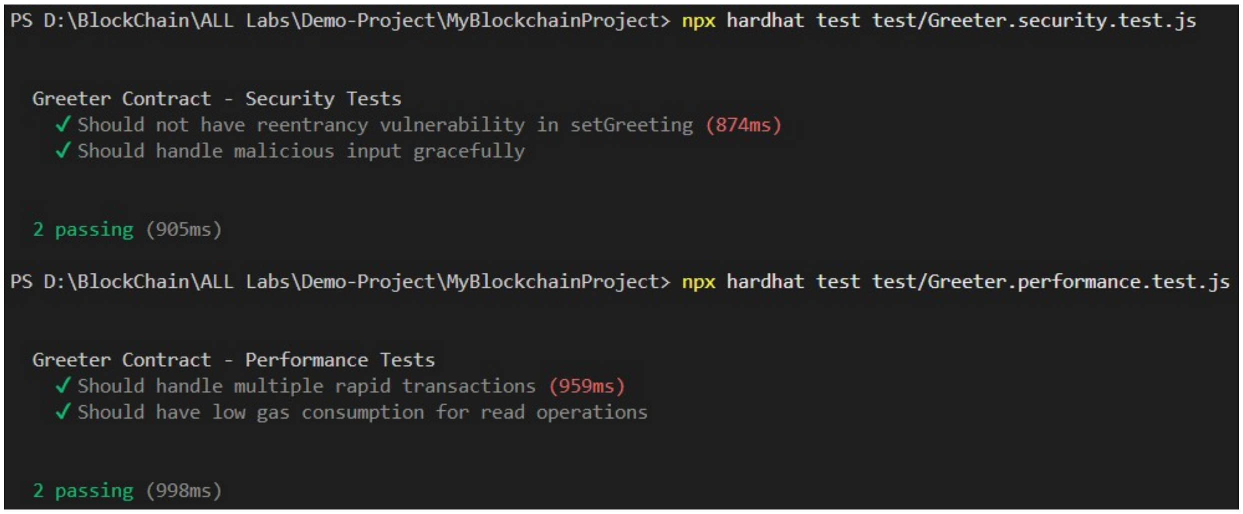
```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test


  Greeter Contract - Performance Tests
    ✓ Should handle multiple rapid transactions (1846ms)
    ✓ Should have low gas consumption for read operations (48ms)

  Greeter Contract - Security Tests
    ✓ Should not have reentrancy vulnerability in setGreeting
    ✓ Should handle malicious input gracefully

  Greeter Contract - Comprehensive QA Tests
    Deployment
      ✓ Should deploy with the correct initial greeting
      ✓ Should set the right owner
    Functionality Tests
      ✓ Should return the correct greeting via greet() function
      ✓ Should update greeting when setGreeting is called
      ✓ Should emit an event when greeting is changed (if events were implemented)
    Edge Cases & Negative Tests
      ✓ Should handle empty string greeting
      ✓ Should handle very long greetings
      ✓ Should allow any address to change greeting (public function)
    Gas Optimization Checks
      ✓ Should deploy with reasonable gas cost
      ✓ Should use reasonable gas for setGreeting
    State Management
      ✓ Should persist greeting changes across transactions
      ✓ Should maintain separate state for different contract instances


  16 passing (2s)
```

**As applicable according to the experiment.*
Two sheets per experiment (10-20) to be used.

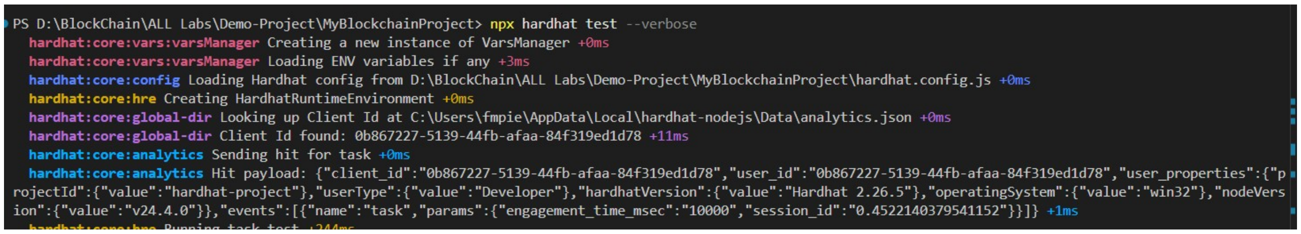
Implementation Phase: Final Output (no error)



```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test test/Greeter.test.js


  Greeter Contract - Comprehensive QA Tests
    Deployment
      ✓ Should deploy with the correct initial greeting (892ms)
      ✓ Should set the right owner
    Functionality Tests
      ✓ Should return the correct greeting via greet() function
      ✓ Should update greeting when setGreeting is called
      ✓ Should emit an event when greeting is changed (if events were implemented)
    Edge Cases & Negative Tests
      ✓ Should handle empty string greeting
      ✓ Should handle very long greetings
      ✓ Should allow any address to change greeting (public function)
    Gas Optimization Checks
      ✓ Should deploy with reasonable gas cost
      ✓ Should use reasonable gas for setGreeting
    State Management
      ✓ Should persist greeting changes across transactions
      ✓ Should maintain separate state for different contract instances


  12 passing (1s)
```



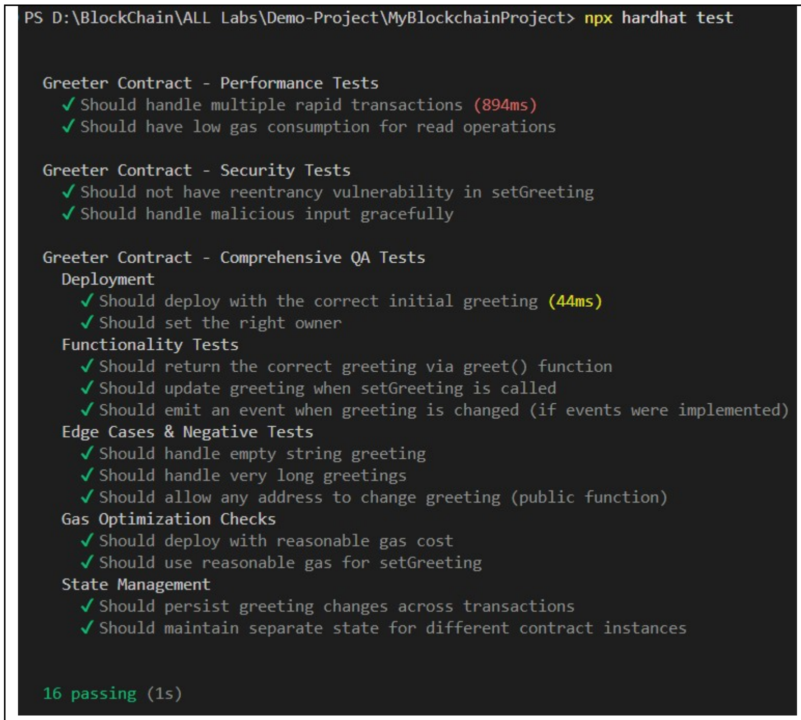
```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test test/Greeter.security.test.js


  Greeter Contract - Security Tests
    ✓ Should not have reentrancy vulnerability in setGreeting (874ms)
    ✓ Should handle malicious input gracefully


  2 passing (905ms)

PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test test/Greeter.performance.test.js


  Greeter Contract - Performance Tests
    ✓ Should handle multiple rapid transactions (959ms)
    ✓ Should have low gas consumption for read operations


  2 passing (998ms)
```



```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test --verbose
  hardhat:core:vars:varsManager Creating a new instance of VarsManager +0ms
  hardhat:core:vars:varsManager Loading ENV variables if any +3ms
  hardhat:core:config Loading Hardhat config from D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject\hardhat.config.js +0ms
  hardhat:core:hre Creating HardhatRuntimeEnvironment +0ms
  hardhat:core:global-dir Looking up Client Id at C:\Users\fmpie\AppData\Local\hardhat-nodejs\Data\analytics.json +0ms
  hardhat:core:global-dir Client Id found: 0b867227-5139-44fb-afaa-84f319ed1d78 +11ms
  hardhat:core:analytics Sending hit for task +0ms
  hardhat:core:analytics Hit payload: {"client_id":"0b867227-5139-44fb-afaa-84f319ed1d78","user_id":"0b867227-5139-44fb-afaa-84f319ed1d78","user_properties":{"p
rojectId":{"value":"hardhat-project"},"userType":{"value":"Developer"},"hardhatVersion":{"value":"Hardhat 2.26.5"},"operatingSystem":{"value":"win32"},"nodeVers
ion":{"value":"v24.4.0"}},"events":[{"name":"task","params":{"engagement_time_msec":"10000","session_id":"0.4522140379541152"}}]} +1ms
  hardhat:core:hre Running task test +244ms
```

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.*

* Implementation Phase: Final Output (no error)



```
PS D:\BlockChain\ALL Labs\Demo-Project\MyBlockchainProject> npx hardhat test


  Greeter Contract - Performance Tests
    ✓ Should handle multiple rapid transactions (894ms)
    ✓ Should have low gas consumption for read operations

  Greeter Contract - Security Tests
    ✓ Should not have reentrancy vulnerability in setGreeting
    ✓ Should handle malicious input gracefully

  Greeter Contract - Comprehensive QA Tests
    Deployment
      ✓ Should deploy with the correct initial greeting (44ms)
      ✓ Should set the right owner
    Functionality Tests
      ✓ Should return the correct greeting via greet() function
      ✓ Should update greeting when setGreeting is called
      ✓ Should emit an event when greeting is changed (if events were implemented)
    Edge Cases & Negative Tests
      ✓ Should handle empty string greeting
      ✓ Should handle very long greetings
      ✓ Should allow any address to change greeting (public function)
    Gas Optimization Checks
      ✓ Should deploy with reasonable gas cost
      ✓ Should use reasonable gas for setGreeting
    State Management
      ✓ Should persist greeting changes across transactions
      ✓ Should maintain separate state for different contract instances


  16 passing (1s)
```

* Observations

1. Comprehensive Test Coverage:
   - Multiple test categories (unit, integration, security, performance)
   - Progressive test complexity from basic to advanced scenarios
   - Modular test structure with separate files for different concerns
2. Testing Pyramid Implementation:

```
    /\     Performance Tests (few)
   /  \
  /__\  Security Tests (some)
 /      \
/____\ Unit & Integration Tests (many)
```

ASSESMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

**As applicable according to the experiment. Two sheets per experiment (10-20) to be used.*