



**Centurion**  
**UNIVERSITY**  
Shaping Lives...  
Empowering Communities...

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 Experiment :** Debugging Deep – Using Hardhat Console & Logs

### \* **Coding Phase: Pseudo Code / Flow Chart / Algorithm**

#### **Algorithm**

Step 1: Install and set up the Hardhat environment using: npm install --save-dev hardhat

Step 2: Create a new Hardhat project and add the contract file (e.g., storage.sol).

Step 3: Add debugging logs using: import "hardhat/console.sol"; console.log("Debug Value:", variableName);

Step 4: Write deployment and test scripts under the scripts/ and test/ folders.

Step 5: Run the Hardhat network and deploy the contract: npx hardhat run scripts/deploy.js

Step 6: Observe the console outputs to verify the flow of execution and identify any logical or arithmetic issues.

Step 7: Fix the errors and re-run tests until the contract executes successfully.

### \* **Softwares used**

- 1.Solidity
- 2.Hardhat
- 3.vs code

Page No.....

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

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

Steps :

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.19;
3
4 import "hardhat/console.sol";
5
6 contract Counter {
7     uint256 private _count;
8
9 constructor() {
10     console.log("Deploying Counter with initial count:", _count);
11 }
12
13 function increment() public {
14     console.log("Before increment, count was:", _count);
15     _count += 1;
16     console.log("After increment, count is:", _count);
17 }
18
19 function getCount() public view returns (uint256) {
20     console.log("Current count is:", _count);
21     return _count;
22 }
23 }
```

- PS D:\WEB 3\Test> **npm install --save-dev --legacy-peer-deps @nomicfoundation/hardhat-toolbox**  
added 1 package, and audited 61 packages in 2s  
16 packages are looking for funding  
run `npm fund` for details  
found 0 vulnerabilities

- PS D:\WEB 3\Test> **npx hardhat test**  
Hardhat only supports ESM projects.  
Please make sure you have `"**type": "module"**` in your package.json.  
You can set it automatically by running:  
**npm pkg set type="module"**

- PS D:\WEB 3\Test> **npm install --save-dev hardhat**  
added 59 packages in 55s  
16 packages are looking for funding  
run `npm fund` for details

- PS D:\WEB 3\Test>

- PS D:\WEB 3\fresh-test> **npm install --save-dev @nomiclabs/hardhat-ethers ethers@5.7.2 chai@1.10.1**  
added 27 packages, changed 9 packages, and audited 384 packages in 38s  
107 packages are looking for funding  
run `npm fund` for details  
**11 vulnerabilities (7 low, 3 high, 1 critical)**  
To address issues that do not require attention, run:  
**npm audit fix**  
Some issues need review, and may require choosing  
a different dependency.  
Run `npm audit` for details.

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

Applied and Action Learning

- 1.Error Detection & Log Analysis: Implemented structured debugging using Hardhat console logs to identify and resolve compilation, deployment, and runtime errors in smart contracts efficiently.
- 2.Performance & Execution Tracking: Enhanced debugging accuracy by monitoring transaction flow, gas usage, and function outputs through Hardhat's console and network tracing tools.
- 3.Code Validation & Testing Alignment: Ensured smart contract reliability by cross-verifying console outputs with expected behaviors during test execution, maintaining consistency and transparency in results.
- 4.Scalability & Continuous Improvement: Developed a modular debugging approach allowing quick iteration, improving developer efficiency, and supporting scalable project development across multiple contract versions.

## \* Observations

1. Identified and resolved deployment and runtime issues effectively using Hardhat console logs.
2. Observed detailed transaction flow and gas usage for better performance analysis.
3. Verified contract behavior and improved debugging accuracy through structured log outputs.

## ASSESSMENT

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		
<b>Total</b>	<b>50</b>		

*Signature of the Student:*

Name :

Regn. No. :

Page No.....

*Signature of the Faculty:*

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