



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 :

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

- **Start** – Begin Ethereum smart contract setup using **Truffle** and **Hardhat**.
- **Install Node.js & npm** – Required for running blockchain tools.
- **Install Truffle** – `npm install -g truffle` to set up the Truffle framework.
- **Create Truffle Project** – Run `truffle init`, then add and compile a sample contract using `truffle compile` and deploy with `truffle migrate`.
- **Install Hardhat** – `npm install --save-dev hardhat` to set up Hardhat in the project.
- **Create Hardhat Project** – `npx hardhat` to generate folders and config files.
- **Compile & Deploy Contract** – Place the same contract in **contracts/**, run `npx hardhat compile` and `npx hardhat run scripts/deploy.js`.
- **Record Outputs** – Capture screenshots of compilation and deployment results.
- **End** – Verify successful deployment and compare Truffle vs Hardhat behavior.

* **Softwares used**

1. npm (Node Package Manager)
2. Truffle Suite – Ethereum development framework
3. Hardhat – Ethereum development environment
4. VS Code – Code editor
5. Ganache

* Testing Phase: Compilation of Code (error detection)

For installing truffle
npm install -g truffle

```
C:\Users\HP>npm install -g truffle
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated mkdirp-promise@5.0.1: This package is broken and no longer maintained. 'mkdirp' itself supports promises now, please switch to that.
npm warn deprecated har-validator@5.1.5: this library is no longer supported
npm warn deprecated yaeti@0.0.6: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm warn deprecated memdown@1.4.1: Superseded by memory-level (https://github.com/Level/community#faq)
npm warn deprecated glob@7.2.0: Glob versions prior to v9 are no longer supported
npm warn deprecated level-errors@2.0.1: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated encoding-down@6.3.0: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated deferred-leveledown@5.3.0: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated levelup@4.4.0: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated level-js@5.0.2: Superseded by browser-level (https://github.com/Level/community#faq)
npm warn deprecated level-packager@5.1.1: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated level-codec@9.0.2: Superseded by level-transcoder (https://github.com/Level/community#faq)
npm warn deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm warn deprecated multibase@0.6.1: This module has been superseded by the multiformats module
npm warn deprecated apollo-server-errors@3.3.1: The 'apollo-server-errors' package is part of Apollo Server v2 and v3, which are now end-of-life (as of October 22nd 2023 and October 22nd 2024, respectively). This package's functionality is now found in the '@apollo/server' package. See https://www.apollographql.com/docs/apollo-server/previous-versions/ for more details.
```

```
C:\Users\HP>npm install -g ganache-cli
npm warn deprecated ganache-cli@6.12.2: ganache-cli is now ganache; visit https://trfl.io/g7 for details
added 1 package in 6s

2 packages are looking for funding
  run 'npm fund' for details

C:\Users\HP>
```

Install Hardhat:

```
C:\Users\HP>cd hardhat-project
C:\Users\HP\hardhat-project>npm init -y
Wrote to C:\Users\HP\hardhat-project\package.json:

{
  "name": "hardhat-project",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

```
C:\Users\HP\hardhat-project>npm install --save-dev hardhat
added 57 packages, and audited 58 packages in 22s

14 packages are looking for funding
  run 'npm fund' for details
```

```
C:\Users\HP\hardhat-project>npm install --save-dev hardhat
added 57 packages, and audited 58 packages in 22s

14 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities

C:\Users\HP\hardhat-project>
```

* Implementation Phase: Final Output (no error)

Truffle project deployment process

1. Install Ganache CLI (for local blockchain)

```
npm install -g ganache-cli
```

2. Create a new Truffle project

```
mkdir truffle-project
```

```
cd truffle-project
```

```
truffle init
```

3. Write smart contract (contracts/SimpleStorage.sol)

4. Configure network in truffle-config.js

5. Start local blockchain

```
npx ganache-cli
```

6. Compile the contract

```
truffle compile
```

8. Deploy (migrate) the contract

9. truffle migrate --network development

10. Open Truffle console for check deploy successfully

The screenshot shows the Visual Studio Code interface with the Truffle project structure on the left and the SimpleStorage.sol smart contract code on the right.

SimpleStorage.sol:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SimpleStorage {
    uint256 private number;

    function set(uint256 _num) public {
        number = _num;
    }

    function get() public view returns (uint256) {
        return number;
    }
}
```

Available Accounts:

```
(0) 0x8eE1DED08257615C21985D4F55022b6A2f2033E6 (100 ETH)
(1) 0xb61B60C9e15C9D1BCbb46F4953902371ddBD5Ee2 (100 ETH)
(2) 0x99Fc5a7ce49BE9693E2056A8d19EB64B05153062 (100 ETH)
(3) 0xF5c1bfEc0152FA3585200e21CA27Da5bB31f079E (100 ETH)
(4) 0x692c9B7927cb79954F0E65F791ea343E648432b4 (100 ETH)
(5) 0xF42370af0806B36A2857B100a5404B9241bB15f7 (100 ETH)
```

TERMINAL:

```
Compiling your contracts...
=====
✓ Fetching solc version list from solc-bin. Attempt #1
✓ Downloading compiler. Attempt #1.
> Compiling .\contracts\SimpleStorage.sol
> Artifacts written to C:\Users\HP\truffle-project\build\contracts
> Compiled successfully using:
  - solc: 0.8.20+commit.a1b79de6.Emscripten.clang
```

* Implementation Phase: Final Output (no error)

Steps to Deploy Smart Contract in Hardhat

1.Create a new folder for project

```
mkdir hardhat-project
cd hardhat-project
```

2.Initialize npm

```
npm init -y
```

4.Install Hardhat

```
npm install --save-dev hardhat
```

Setup Hardhat project

5.npx hardhat

Select “Create a JavaScript project”, press Enter for defaults.

6.Write smart contract (contracts/SimpleStorage.sol)

7.Add deployment script (scripts/deploy.js)

8.Compile the contract

9.npx hardhat compile

10.Start local Hardhat blockchain

The screenshot shows the Visual Studio Code interface with the Truffle Project workspace open. The Explorer sidebar on the left lists the project structure, including build, contracts, migrations, test, and node_modules. The contracts folder contains SimpleStorage.json and SimpleStorage.sol. The SimpleStorage.sol file is selected and shown in the main editor area. The code defines a Solidity contract named SimpleStorage with two functions: set(uint256 _num) and get().

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SimpleStorage {
    uint256 private number;

    function set(uint256 _num) public {
        number = _num;
    }

    function get() public view returns (uint256) {
        return number;
    }
}
```

The screenshot shows a terminal window with a dark theme. It displays the deployment of the SimpleStorage contract and its interaction using ethers.js. The code creates a contract factory, deploys the contract, interacts with it by setting a value and getting the current value, and handles errors.

```
1 async function main() {
2     const SimpleStorage = await ethers.getContractFactory("SimpleStorage");
3     const simpleStorage = await SimpleStorage.deploy();
4     await simpleStorage.deployed();
5     console.log("SimpleStorage deployed at:", simpleStorage.address);
6
7     // Interact with contract
8     await simpleStorage.set(42);
9     const value = await simpleStorage.get();
10    console.log("Stored value:", value.toString());
11 }
12
13 main().catch((error) => {
14     console.error(error);
15     process.exitCode = 1;
16 });
17
```

* Implementation Phase: Final Output (no error)

Applied and Action Learning

```
C:\Users\HP\hardhat-project>> Block gas limit: 6721975 (0x6691b7)
'gas' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\HP\hardhat-project>
C:\Users\HP\hardhat-project>
C:\Users\HP\hardhat-project>l_initial_migration.js
'l_initial_migration.js' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\HP\hardhat-project>=====
C:\Users\HP\hardhat-project> Deploying 'Migrations'
'Deploying' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\HP\hardhat-project> -----
'-----' is not recognized as an internal or external command,
operable program or batch file.
```

* Observations

1. Truffle provides a simple setup with migration scripts and is suitable for beginners, but it has slower compilation and limited debugging features.
2. Hardhat offers faster compilation, detailed error stack traces, and better debugging tools, making it more developer-friendly for production projects.
3. Both frameworks achieve the same goal of compiling, deploying, and testing smart contracts, but Hardhat is more modern and efficient, while Truffle is easier to start with.

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		
Total	50		

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.