**1) Write tests for your smart contract and demonstrate**

**the process of testing using Hardhat Development**

**Environment.**Ans. The steps for testing your smart contract and how to test it using the Hardhat development environment are:  
  
Steps to Demonstrate the Testing Process:  
 Setting up hardhat environment:

* Initialize a Node.js project
* Install Hardhat and set up a basic project structure.
* Write a sample smart contract
* Write unit tests for the smart contract using Mocha
* Run the tests  
    
  **Step 1**: Install Hardhat and initialize the project.  
  1. Install Hardhat - Open a terminal and navigate to your project directory, then run:  
  npm init -y  
  Npm install –save-dev hardhat  
    
  2. Set up a hardhat project:  
  npx hardhat  
    
  Choose "Create a basic sample project" and follow the prompts. This will create the basic directory structure for the project.

**Step 2.** Write a sample smart contract  
Here, we will write a sample contract called SimpleStorage.sol that stores and retrieves a value.  
contracts/SimpleStorage.sol  
  
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract SimpleStorage {

uint256 private storedValue;

function set(uint256 value) public {

storedValue = value;

}

function get() public view returns (uint256) {

return storedValue;

}

}  
  
**Step 3.** Write Unit Tests  
Now let's write a unit test for this contract using Mocha as the testing framework and Chai for assertions.  
  
test/SimpleStorageTest.js:  
  
const { expect } = require("chai");

const { ethers } = require("hardhat");

describe("SimpleStorage Contract", function () {

let SimpleStorage;

let simpleStorage;

let owner;

beforeEach(async function () {

SimpleStorage = await ethers.getContractFactory("SimpleStorage");

[owner] = await ethers.getSigners();

simpleStorage = await SimpleStorage.deploy();

await simpleStorage.deployed();

});

it("Should return the stored value after setting it", async function () {

await simpleStorage.set(42);

expect(await simpleStorage.get()).to.equal(42);

});

it("Should return zero as the initial stored value", async function () {

expect(await simpleStorage.get()).to.equal(0);

});

it("Should update the stored value when set is called", async function () {

await simpleStorage.set(100);

expect(await simpleStorage.get()).to.equal(100);

});

});  
  
**Step 4.** Run the tests  
To run the tests, execute the following command:  
npx hardhat test  
  
Expected output: If everything is correctly configured, you should see something like:  
 *SimpleStorage Contract*

*Should return the stored value after setting it (38ms)*

*Should return zero as the initial stored value (38ms)*

*Should update the stored value when set is called (41ms)*

*3 passing (300ms)*