**Crowd Funding Smart Contract Deployment Document**

Soumil Vavikar

Purdue University Global

IN532 – Blockchain Application Development (dApps)

Dr. David Ostrowski

November 19, 2024

**Crowd Funding Smart Contract Deployment Document**

This document contains the development, testing, and deployment-related documentation for the “Crowd Funding” and “Crowd Funding Factory” smart contracts developed as part of Unit 6, Unit 7, Unit 9, and Unit 10 assignments.

**Business Use-Case**

A crowdfunding contract will swap Ether (ETH) for non-fungible tokens (NFTs), representing a creative approach to decentralized fundraising on the Ethereum blockchain. This smart contract lets crowdfunding creators raise funds by offering custom NFTs to supporters in exchange for their ETH contributions. Participants can contribute by transmitting ETH to the contract address and receiving one project-specific token per successful contribution.

**Smart Contract and Supporting Files**

This smart contract is written in Solidity, and the "Hardhat" framework is used to develop, test, and deploy the smart contract.

A GitHub repository (<https://github.com/soumilvavikar/hardhat-crowd-funding>) has been created, and the smart contract, supporting files (test files, interaction files, and library files), test evidence, and commands required to start the local chain, deploy the smart contract, and test the smart contract have been pushed to the repository.

The entire workspace for the crowdfunding contract has also been submitted with this deployment document.

**Crowd Funding and Crowd Funding Factory Smart Contract Code**

Two libraries have been created to make crowdfunding smart contracts readable and maintainable. The sol files for the libraries are attached to the smart contract code.

  

**Deployment and Execution Commands**

|  |
| --- |
| ## Initial Repository Setup  *# Initialize NPM*  npm init  # Install hardhat if not done already  npm install --save-dev hardhat  # Initialize the hardhat project (select a valid option)  npx hardhat init  # Install the openzeppelin contract to use ERC721 interface  npm install --save-dev @openzeppelin/contracts  ## Compiling and Testing the Contract  # Compile the project  npx hardhat compile  # Run the tests  npx hardhat test  ## Starting the local chain and Deploying the Contract  # Spin the local chain  npx hardhat node  # Deploy the contract / hardhat project  npx hardhat ignition deploy ignition/modules/CrowdFundingModule.js --network localhost  # Deploying the crowd funding factory contract  npx hardhat ignition deploy ignition/modules/CrowdFundingFactoryModule.js --network localhost  ### Command for End-to-End Testing via Interactions  # Run all the functions to test the flow end to end.  npx hardhat run interactions/end-to-end/InteractingWithCrowdFundingContract.js --network localhost  # Run all the functions to test the flow end to end using the factory  npx hardhat run interactions/end-to-end/InteractingWithCrowdFundingFactoryContract.js --network localhost |

**Test Evidence(s)**

Extensive testing (including unit testing and end-to-end functional testing) of the developed smart contract has been done. The project's README.md contains quick links to the “.md” files containing the test evidence (logs and screenshots) captured within the workspace under the “test evidences” folder. The test evidence can also be found below.

**Unit Test Evidence**

Five tests have been added to assert that the factory smart contract and its functions work as intended. The unit test file and the unit testing evidence text file, which contains the command and the logs generated from the unit test run, are attached below.

 

The unit test readme file “README\_UNIT\_TESTING.md” is in the workspace's “testevidences” folder.

A screenshot of a computer

Description automatically generated

**Deployment Test Evidence**

No changes were made to the deployment process/steps from the process setup during the smart contract development work done during the Unit 6 and Unit 7 assignments. The crowdfunding smart contract has been deployed using the ignition module. The file containing the module code required for deployment and the test evidence of successful local chain startup and deployment are attached below.



A screen shot of a computer

Description automatically generated

A screen shot of a computer screen

Description automatically generated

A new “CrowdFundingFactoryModule.js” file has been created to deploy the “CrowdFundingFactory” contract to the local chain. PFB, the module file, and the test evidence of successful deployment to it.

 

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The deployment test evidence readme file “README\_E2E\_TEST\_EVIDENCES.md” is in the workspace's “testevidences" folder.

**End-to-End Test Evidence**

The end-to-end test interactions files and the end-to-end test evidence text files, which contain the logs generated from the end-to-end test runs, are attached below.

   

The end-to-end test readme file “README\_E2E\_TEST\_EVIDENCES.md” is in the workspace's “testevidences” folder.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated