

Lesson 13

HARDHAT DEFİ & AAVE

Hikmah Nisya - 1103184094
Radzis Araaf Jaya Jamaludin - 1103184234
Raudhatul Rafiqah Assyahiddini - 1103180225

```
patrick@iMac: [~/hh-fcc/hardhat-defi-fcc] $ yarn add --dev hardhat
yarn add v1.22.17
info No lockfile found.
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[#####] 4312/12314
```

```

      "88b 888P"  d88" 888 888 "88b      "88b 888
      888 .d888888 888 888 888 888 .d888888 888
      888 888 888 888 Y88b 888 888 888 888 Y88b.
      888 888 "Y888888 888 "Y888888 888 888 "Y888888 "Y888

👤 Welcome to Hardhat v2.9.3 👤

? What do you want to do? ...
  Create a basic sample project
  Create an advanced sample project
  Create an advanced sample project that uses TypeScript
> Create an empty hardhat.config.js
  Quit

```

```

> node_modules JS hardhat.config.js > ...
JS hardhat.config.js 1 /**
2 * @type import('hardhat/config').HardhatUserConfig
3 */
4 module.exports = {
5   solidity: "0.7.3",
6 };
7

```

Buatlah Hardhat Terlebih dahulu dengan code “ yarn add –dev hardhat” lalu enter, dan tunggu sampai instalasi nya selesai.

Setelah itu jalan kan hardhatnya denga code “ yarn hardhat”, tunggu sampai muncul pilihan, lalu pilih “ buat hardhat.config.js” setelah itu enter

Lalu akan muncul tampilan seperti gambar di samping, dan selanjutnya buat project yang sama seperti Lesson 9.

```
scripts > JS aaveBorrow.js > main
```

```
1
2
3  async function main(){
4
5  }
6
7  main()
8      .then(() => process.exit(0))
9      .catch((error) => {
10         console.error(error)
11         process.exit(1)
12     })
```

selanjutnya membuat Scripts baru dengan nama "aaveBorrows.js" dan membuat function main seperti gambar di ini.

WETH

Wrapped ETH

```
scripts > JS aaveBorrow.js >  main
```

```
1
2
3  async function main(){
4
5  }
6
7  main()
8      .then(() => process.exit(0))
9      .catch((error) => {
10         console.error(error)
11         process.exit(1)
12     })
```

Langkah selanjutnya adalah membuat Scripts baru dengan nama “aaveBorrows.js” dan membuat function main seperti gambar di ini.

```

1  const { ethers, getNamedAccounts, network } = require("hardhat")
2  const { networkConfig } = require("../helper-hardhat-config")
3
4  const AMOUNT = ethers.utils.parseEther("0.1")
5
6  async function getWeth() {
7      const { deployer } = await getNamedAccounts()
8      const iWeth = await ethers.getContractAt(
9          "IWeth",
10         networkConfig[network.config.chainId].wethToken,
11         deployer
12     )
13     const txResponse = await iWeth.deposit({
14         value: AMOUNT,
15     })
16     await txResponse.wait(1)
17     const wethBalance = await iWeth.balanceOf(deployer)
18     console.log(`Got ${wethBalance.toString()} WETH`)
19 }
20
21 module.exports = { getWeth, AMOUNT }

```

Lalu buat script baru dengan nama "getWeth.js", buat async function getWeth untuk membuat token yang di masukan kedalam data atau web aave

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.6.0;
3
4 interface AggregatorV2Interface {
5     function decimals() external view returns (uint8);
6
7     function description() external view returns (string memory);
8
9     function version() external view returns (uint256);
10
11     // getRoundData and latestRoundData should both raise "No data present"
12     // if they do not have data to report, instead of returning unset values
13     // which could be misinterpreted as actual reported values.
14     function getRoundData(uint80 _roundId)
15         external
16         view
17         returns (
18             uint80 roundId,
19             int256 answer,
20             uint256 startedAt,
21             uint256 updatedAt,
22             uint80 answeredInRound
23         );
24
25     function latestRoundData()
26         external
27         view
28         returns (
29             uint80 roundId,
30             int256 answer,
31             uint256 startedAt,
32             uint256 updatedAt,
33             uint80 answeredInRound
34         );
35 }
36 Footer

```

```

1 pragma solidity ^0.4.25;
2
3 interface Iweth {
4     function allowance(address owner, address spender) external view returns (uint256 remaining);
5
6     function approve(address spender, uint256 value) external returns (bool success);
7
8     function balanceOf(address owner) external view returns (uint256 balance);
9
10    function decimals() external view returns (uint8 decimalPlaces);
11
12    function name() external view returns (string memory tokenName);
13
14    function symbol() external view returns (string memory tokenSymbol);
15
16    function totalSupply() external view returns (uint256 totalTokensIssued);
17
18    function transfer(address to, uint256 value) external returns (bool success);
19
20    function transferFrom(
21        address from,
22        address to,
23        uint256 value
24    ) external returns (bool success);
25
26    function deposit() external payable;
27
28    function withdraw(uint256 wad) external;
29 }

```

Selanjutnya membuat “AggregatorV3Interface.sol”, ERC20, Ilendingpool,IWeth untuk interfacenya dalam contrakctions.

```

1 pragma solidity ^0.6.0;
2
3 interface IERC20 {
4     function allowance(address owner, address spender) external view returns (uint256 remaining);
5
6     function approve(address spender, uint256 value) external returns (bool success);
7
8     function balanceOf(address owner) external view returns (uint256 balance);
9
10    function decimals() external view returns (uint8 decimalPlaces);
11
12    function decreaseApproval(address spender, uint256 addedValue) external returns (bool success);
13
14    function increaseApproval(address spender, uint256 subtractedValue) external;
15
16    function name() external view returns (string memory tokenName);
17
18    function symbol() external view returns (string memory tokenSymbol);
19
20    function totalSupply() external view returns (uint256 totalTokensIssued);
21
22    function transfer(address to, uint256 value) external returns (bool success);
23
24    function transferFrom(
25        address from,
26        address to,
27        uint256 value
28    ) external returns (bool success);
29 }

```

```

1 // SPDX-License-Identifier: Apache-2.0
2 pragma solidity 0.6.12;
3 pragma experimental ABIEncoderV2;
4
5 import {LendingPoolAddressProvider} from "aave/protocols/v2/contracts/interfaces/LendingPoolAddressProvider.sol";
6 import {DataTypes} from "aave/protocols/v2/contracts/interfaces/LendingPoolAddressProvider.sol";
7
8 interface ILendingPool {
9     /*
10      * @dev Deposit on deposit()
11      * @param reserve The address of the underlying asset of the reserve
12      * @param user The address initiating the deposit
13      * @param amountRef the beneficiary of the deposit, receiving the tokens
14      * @param amount The amount deposited
15      * @param referral The referral code used
16      */
17     event Deposit(
18         address indexed reserve,
19         address user,
20         address indexed amountRef,
21         uint256 amount,
22         uint256 indexed referral
23     );
24
25     /*
26      * @dev Deposit on withdraw()
27      * @param reserve The address of the underlying asset being withdrawn
28      * @param user The address initiating the withdrawal, owner of tokens
29      * @param to Address that will receive the underlying
30      * @param amount The amount to be withdrawn
31      */
32     event Withdraw(address indexed reserve, address indexed user, address indexed to, uint256 amount);
33
34     /*
35      * @dev Initiated on borrow() and flashloan() when debt needs to be created
36      * @param reserve The address of the underlying asset being borrowed
37      * @param user The address of the user initiating the borrow(), receiving the funds on borrow() or flash
38      * @param initiator the transaction on flashloans()
39      * @param amountRef the address that will be getting the debt
40      * @param amount The amount borrowed
41      * @param borrowRateMode the rate mode: 1 for Stable, 2 for Variable
42      * @param borrowRate the variable rate of which the user has borrowed
43      * @param referral The referral code used
44      */

```

Dengan begitu DeFi bias di gunakan ke dalam Aave


```

1  pragma solidity ^0.6.6;
2
3  interface IERC20 {
4      function allowance(address owner, address spender) external view returns (uint256 remaini
5
6      function approve(address spender, uint256 value) external returns (bool success);
7
8      function balanceOf(address owner) external view returns (uint256 balance);
9
10     function decimals() external view returns (uint8 decimalPlaces);
11
12     function decreaseApproval(address spender, uint256 addedValue) external returns (bool s
13
14     function increaseApproval(address spender, uint256 subtractedValue) external;
15
16     function name() external view returns (string memory tokenName);
17
18     function symbol() external view returns (string memory tokenSymbol);
19
20     function totalSupply() external view returns (uint256 totalTokensIssued);
21
22     function transfer(address to, uint256 value) external returns (bool success);
23
24     function transferFrom(
25         address from,
26         address to,
27         uint256 value
28     ) external returns (bool success);
29 }

```

Selanjutnya kita akan membuat
“IERC20.sol”