Hardhat DeFI & Aave

AggregatorV3Interface.sol

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.6.0;

interface AggregatorV3Interface {
  function decimals() external view returns (uint8);

  function description() external view returns (string memory);

  function version() external view returns (uint256);

  // getRoundData and latestRoundData should both raise "No data present"
  // if they do not have data to report, instead of returning unset values
  // which could be misinterpreted as actual reported values.
  function getRoundData(uint80 _roundId)
    external
    view
    returns (
      uint80 roundId,
      int256 answer,
      uint256 startedAt,
      uint256 updatedAt,
      uint80 answeredInRound
    );
```

IERC20.sol

```solidity
pragma solidity ^0.6.6;

interface IERC20 {
  function allowance(address owner, address spender) external view returns (uint256 remaining);

  function approve(address spender, uint256 value) external returns (bool success);

  function balanceOf(address owner) external view returns (uint256 balance);

  function decimals() external view returns (uint8 decimalPlaces);

  function decreaseApproval(address spender, uint256 addedValue) external returns (bool success);

  function increaseApproval(address spender, uint256 subtractedValue) external;

  function name() external view returns (string memory tokenName);

  function symbol() external view returns (string memory tokenSymbol);

  function totalSupply() external view returns (uint256 totalTokensIssued);

  function transfer(address to, uint256 value) external returns (bool success);

  function transferFrom(
    address from
```

ILendingPool.sol

```solidity
// SPDX-License-Identifier: agpl-3.0
pragma solidity 0.6.12;
pragma experimental ABIEncoderV2;

import {ILendingPoolAddressesProvider} from "@aave/protocol-v2/contracts/interfaces/ILendingPoolAddres
import {DataTypes} from "@aave/protocol-v2/contracts/protocol/libraries/types/DataTypes.sol";

interface ILendingPool {
  /**
   * @dev Emitted on deposit()
   * @param reserve The address of the underlying asset of the reserve
   * @param user The address initiating the deposit
   * @param onBehalfOf The beneficiary of the deposit, receiving the aTokens
   * @param amount The amount deposited
   * @param referral The referral code used
   **/
  event Deposit(
    address indexed reserve,
    address user,
    address indexed onBehalfOf,
    uint256 amount,
    uint16 indexed referral
  );
```

FILE EXPLORERS

Workspaces

default_workspace

- AggregatorV3Interface.sol
- IERC20.sol
- ILendingPool.sol
- IWeth.sol

Home   AggregatorV3Interface.sol   IERC20.sol   ILendingPool.sol ✕

```solidity
 96   event RebalanceStableBorrowRate(address indexed reserve, address indexed user);
 97
 98   /**
 99    * @dev Emitted on flashLoan()
100    * @param target The address of the flash loan receiver contract
101    * @param initiator The address initiating the flash loan
102    * @param asset The address of the asset being flash borrowed
103    * @param amount The amount flash borrowed
104    * @param premium The fee flash borrowed
105    * @param referralCode The referral code used
106    **/
107   event FlashLoan(
108     address indexed target,
109     address indexed initiator,
110     address indexed asset,
111     uint256 amount,
112     uint256 premium,
113     uint16 referralCode
114   );
115
116   /**
117    * @dev Emitted when the pause is triggered.
118    */
119   event Paused();
```

FILE EXPLORERS

Workspaces

default_workspace

- AggregatorV3Interface.sol
- IERC20.sol
- ILendingPool.sol
- IWeth.sol

Home   AggregatorV3Interface.sol   IERC20.sol   ILendingPool.sol   IWeth.sol ✕

```solidity
 1   pragma solidity ^0.4.19;
 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);
```