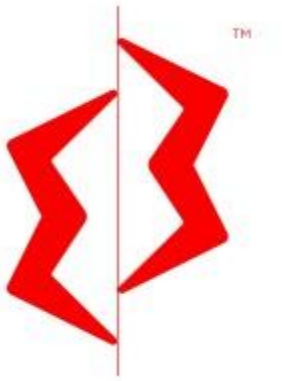

Web3Bridge Cohort (V)

Code Review Assignment

Code Reviewed by:

Augustine Onuora

Code URL: https://github.com/bugduino/idle-contracts/blob/master/contracts/IdleTokenV3_1.sol

Title: IdleToken

Version: Version 3.1

Language: Solidity

Facilitator: Akinnusotu Temitayo Daniel

Line 1-7: A comment explaining the name, version of the contract and what the contract is expected to do.

Line 8: used to inform solidity the compiler version to use when compiling the contract.

Line 9-19: uses the import statement to import external contracts that will interact with the current. The current contract will inherit properties of these external contracts. Files imported are SafeMath.sol, SafeERC20.sol, /ERC20.sol, ERC20Detailed.sol, Ownable.sol, Pausable.sol, ReentrancyGuard.sol and Initializable.sol.

Line 21-28: uses interfaces which do not have any definition or any state variables, constructors, or any function with implementation, they only contain function declarations. Interfaces used are iERC20Fulcrum.sol, ILendingProtocol.sol. IGovToken.sol, IIdleTokenV3_1.sol, IIdleRebalancerV3.sol, Comptroller.sol and CERC20.sol

Line 30: imports GST2ConsumerV2.sol

Line 32: this is where the name of the contract was state as IdleTokenV3_1. This line also initialized the inherited (imported) contracts such as ERC20, ERC20Detailed, ReentrancyGuard, Ownable, Pausable, IIdleTokenV3_1, GST2ConsumerV2,

Line 33: declared that SafeERC20 will be used for IERC20

Line 34: declared that SafeMath will be used for uint256.

Line 36: The constant name ONE_18 was declared as private with type unit256 and was assigned a value of 10**18.

Line 39: declared State variable token as type address with scope Public.

Line 41: declared State variable iToken as type address with scope private.

Line 43: declared State variable cToken as type address with scope private.

Line 45-59: State variable declaration of rebalancer, feeAddress, lastITokenPrice, TokenDecimal, maxUnlentPerc, fee, allAvailableTokens and govTokens as type address.

Line 62: lastAllocation was declared as type uint256.

Line 64: uses mapping to map userAvgprices from Address to uint256.

Line 66: uses mapping to map protocolWrappers from address to address.

Line 68: uses mapping to map govTokensLastBalances from Address to uint256.

Line 70: uses nested mapping to map usersGovTokensIndexes from address to (Address to uint256).

Line 72: uses mapping to map govTokensIndexes from Address to uint256.

Line 74: uses mapping to map userNoFeeQty from Address to uint256.

Line 76: Declared _minterBlock as type byte32 and scoped as private

Line 79: created event Rebalance that will be called by the front-end with two variables _rebalancer and _amount .

Line 80: created event Referral that will be called by the front-end with two variables _amount and _ref.

Line 91-98: function Initializes variables _name, _Symbol as type string while _token, _iToken, _cToken and _rebalancer as type address.

Line 99-100: Declared a constructor named initializer

Line 101: uses the require keyword to make sure that address of _rebalancer is not equal to address Zero (0).

Line 103: initialized the variable _name, symbol and assigned 18 decimal places to decimals.

Line 104: assigned ownable to msg.sender

Line 105: assigned Pausable to msg.sender.

Line 106: initialized inherited contracts ReentrancyGuard and GST2ConsumerV2.

Line 109: 1000 was assigned to maxUnlentPerc.

Line 111: _token was assigned to token

Line 112: number of decimal places was assigned to tokenDecimals.

Line 113: _itoken was assigned to itoken.

Line 114: _ctoken was assigned to ctoken.

Line 115: _rebalancer was assigned to rebalancer.

Line 116: closing brace for the constuctor initialize

Line 123-139: presents a modifier named whenITokenPriceHasNotDecreased()whose function is has two checks. Firstly, it checks the address of the idle token to make sure it is not address(0) then the last token price is assigned. Secondly, it checks the idle token price to know if it is greater than the last tokenprice then lastITokenPrice = iTokenPrice.

Line 150-161: function setAllAvailableTokensAndWrappers allows onlyowner to change available tokens when there is a bug on interest bearing tokens. This function uses an array of protocols tokens called protocolTokens and array of wrapper address called wrapper.

Line 168-171: Function setToken used by owner to set idle token address.

Line 179-183: function setGovTokens: is used by owner to set governance token array.

Line 190-194: function setRebalancer allows owner to set the address of the idle rebalancer.

Line 201-206: function setFee allows owner eto set fee. The fee should be less than or equal to 10%.

Line 212-216: fuction setMaxUnlengthPerc allows owner to set the Maximum Unlent percentage which should be less than or equal to 100%.

Line 223-227: function setFeeAddress allows owner to set fee address. it should not be address(0).

Line 235-239: function tokenPrice: returns a view of the token price.

Line 247-258: function getAPRs returns arrays of token addresses and array of aprs from the ilendingProtocol.

Line 265-282: gets the avg APR of the token and returns the view of the avg APR.

Line 293-299: function transferFrom is used to transfer and update the avgPrice paid for the recipient, and update the user gov idx.

Line 309-314: function transfer updates the avgPrice paid for the recipeint and update the user gov idx and return a true or false is action is successful or not.

Line 323-349: function _updateUserGovIdxTransfer uses the _rom, _to and amount parameters for transfer and transferFrom and updates recipient gov indexes.

Line 357-365: function getGovTokensAmounts gets how many token a user is entitled to and returns a view of the amount.

Line 379-405: function mintIdleToken mints and returns idletoken, give an underlying amount. Keccak256 is used to encode the block. it calls the rebalancer if _skiprebalancer is false and also uses gastoken to get a gas discount.

Line 413-429: function _updateUserGovIdx calacualtes user index and upates minter gov indexes.

Line 440-476: function redeemIdleToken checks the block and makes sure no mint have been made on same block, it then calculate the pool share one can withdraw given the amount of idleToken they want to burn.

Line 485-506: function redeemInterestBearingTokens takes care of of the amount of interest one can withdraw given the amount of idletoken they want to burn.

Line 517-521: function rebalanceWithGST returns rebalance and shold be approve by msg.sender.

Line 531-533: function rebalance returns a true or false for _rebalance.

Line 541-561: function _tokenPrice gets the current idle token price and returns a view of it.

Line 570-652: function _rebalance() returns a bool of the rebalance by checking if there is nede to rebalance by looking at the rebalancer contract.

Line 661-729: function _redeemGovTokens() is used to redeem unclaimed governane tokens and update governance global index and user index. it sends all accrrued token by a user to the user when called during redeem.

Line 738-748: function _updateUserFeeInfo is used to update avg price paid for each idle token of a user.

Line 757-776: function _getFee is used to calculate fee and send it to the feAddress.

Line 785-800: function _mintWithAmounts mints specific amounts of protocols tokens.

Line 809-825: function _amountsFromAllocations returns calculated newAmount from perentage allocations.

Line 836-875: function _redeemAllNeeded takes care of redeeming all underlying needed from each protocols. it checks the difference between amount and newAmount and redeem the difference. The amount is converted from underlying to protocol token.

Line 886-907: function _getCurrentAllocations() is used to get the contrat balance of every protocol currently been used. Return token addresses, respective amounts in underlying and total Amount.

Line 914-928: function _getCurrentPoolValue() gets the current pool value in underlying and rturns total in underlying.

Line 939-949: function _redeemProtocolTokens is used to redeem underlying tokens through protocol wrappers.