**AUCTION CONTRACT:**

The Practical Work for Module 2 required the creation of a Smart Contract for an auction, the characteristics and considerations of which were outlined in the Campus. Therefore, the steps taken to complete the assigned task will be explained.

CONTRACT STRUCTURE: OVERVIEW

The contract manages an Ethereum auction, allowing:

* Bidding with ETH.
* Reimbursing non-winners.
* Automatically determining the highest bidder.
* If a bidder does not win by higher bid, rebidding takes their first bid into account and adds subsequent bids to the auction.
* Applying fees.
* Extending the auction if there are late bids.
* Managing auction completion.
* Determining who can perform actions, such as ending the auction.

Global Variables:

* public address owner; Auction owner (the person who created it).
* public uint deadline; Auction deadline.
* public uint commissionPercent = 2; Fixed commission of 2%.
* public uint timeExtension = 10 minutes; If someone bids in the last 10 minutes, the bid is extended for 10 minutes.

Structure of an Offer:

* struct Bid { address bidder; uint amount;}; Represents an individual offer: who bid and how much.

Storage Structures:

* Bid[] public bidHistory; List of all valid registered bids.
* Bid public highestBid; Stores the highest accumulated bid.
* mapping(address => uint) public accumulatedBids; Stores the total amount bid by each user.
* mapping(address => uint[]) public userBidHistory; Stores an array of all bids per user.
* mapping(address => uint) public refunds; Amounts that can be withdrawn for partial refunds.

Modifiers:

* modifier onlyBeforeEnd() { require(block.timestamp < deadline, "The auction has ended.") \_; }; Only accept bids before the deadline.
* modifier onlyOwner() { require(msg.sender == owner, "Only the owner can perform this action."); \_; }; Only the owner can end the auction.

Builder:

* constructor(uint \_durationSeconds) { owner = msg.sender; deadline = block.timestamp + durationSeconds;}; Defines the owner and calculates the deadline.

Functions:

function placeBid():

* Validates that the amount sent is greater than 0.
* Adds the bid to the user's running total.
* Updates the user's history.
* Checks if the total exceeds the current best bid by at least 5%.
* Updates the best bid and the overall history.
* Extends the deadline if there are less than 10 minutes remaining.

Function endAuction():

* Only the owner can call it.
* Verifies that the deadline has passed.
* Applies a 2% fee.
* Transfers the remainder to the owner.
* Emits the AuctionEnded event.

function withdrawDeposit():

* If you are not the winner, you can withdraw your accumulated bid after the bid closes.

function getWinner(): Current winner.

function getBidHistory(): Bid history.

function timeRemaining(): Time remaining.

function bidsOf(address user): Individual bids for a user.

Events:

* event NewBid(address indexed bidder, uint amount);
* event AuctionEnded(address winner, uint winningAmount);

They emit messages to be captured by dApps or web interfaces (frontend).