**CSED490U Blockchain & Cryptocurrency**

**Assignment 10**

****

**Submitted by- Sajan Maharjan**

**POVIS id-** [**thesajan@postech.ac.kr**](mailto:thesajan@postech.ac.kr)

**Registration Number- 20182095**

**CustomToken.Sol**

|  |
| --- |
| pragma solidity ^0.4.18;  contract CustomToken {  string public name;  string public symbol;  uint8 public decimals;  mapping (address => uint256) public balanceOf;  event Transfer(address \_from, address \_to, uint \_value);  constructor(string \_tokenName,string \_tokenSymbol,uint8 \_decimalUnits, uint256 \_initialSupply) public {  name = \_tokenName;  symbol = \_tokenSymbol;  decimals = \_decimalUnits;  balanceOf[msg.sender] = \_initialSupply  }  function transfer(address \_to, uint256 \_value) public {  require(balanceOf[msg.sender] >=\_value);  require(balanceOf[\_to] + \_value >= balanceOf[\_to] );  balanceOf[msg.sender] -= \_value;  balanceOf[\_to] += \_value;  emit Transfer(msg.sender,\_to,\_value);  } } |

Crowdfunding.sol

|  |
| --- |
| pragma solidity ^0.4.18;  interface token {  function transfer(address receiver, uint amount) external;  }  contract CrowdFunding {  address public owner;  uint public goalAmount;  uint public totalAmount;  uint public deadline;  uint public price;  token public tokenReward;  mapping(address => uint256) public balanceOf;  bool public goalReached;  bool public ended  event GoalReached(address ownerAddress, uint amountRaisedValue);  event FundTransfer(address backer, uint amount, bool isContribution);  modifier onlyOwner() {  require(msg.sender == owner);  \_; }  modifier afterDeadline() {  require (now >= deadline);  \_; }  constructor(uint \_goalAmount, uint \_durationMinutes, uint \_costOfToken, address \_tokenAddress) public {  owner = msg.sender;  goalAmount = \_goalAmount \* 1 ether;  deadline =now + \_durationMinutes \* 1 minutes;  price = \_costOfToken \* 1 ether;  tokenReward = token(\_tokenAddress);  totalAmount = 0;  goalReached = false;  ended = false; }  function () payable external {  require(!ended);  uint amount = msg.value;  balanceOf[msg.sender] += amount;  totalAmount += amount;  tokenReward.transfer(msg.sender, amount / price);  emit FundTransfer(msg.sender, amount, true); }  function checkGoalReached () external afterDeadline {  require(!ended);  if (totalAmount >= goalAmount){  goalReached = true;  emit GoalReached(owner, totalAmount);  }  ended = true; }  function withdraw() external afterDeadline {  if (!goalReached) {  uint amount = balanceOf[msg.sender];  balanceOf[msg.sender] = 0;  if (amount > 0) {  if (msg.sender.send(amount)) {  emit FundTransfer(msg.sender, amount, false);  }  else {  balanceOf[msg.sender] = amount;  }  }  }  if (goalReached && owner == msg.sender) {  if (owner.send(totalAmount)) {  emit FundTransfer(owner, totalAmount, false);  }  else {  goalReached = false;  }  } }  function kill() public onlyOwner {  selfdestruct(owner); }  } |

(to be completed soon)