pragma solidity ^0.4.26; //We have to specify what version of the compiler this code will use

contract InvoiceProc {

string public enterp; // MGH

address public enterPAddress;

string public vendorName;

address public vendorAddress;

string public goods;

uint public Qty;

uint public cost;

uint public counter = 0;

address public benef;

enum POState {

PO,

Invoice,

Paid

}

struct Invoice{

string nameFrom;

string nameTo;

uint256 value;

address from;

address to;

uint id;

string status;

}

//mapping(address=>Invoice[]) invoices;

Invoice[] invoices;

event LogDepositMade(address accountAddress, uint amount);

POState public PState = POState.PO;

function PurchaseOrd (string compName,address compAdd,string vend,address vendadd,string item, uint Qty) returns (uint id)

{

enterp = compName;

enterPAddress =compAdd;

vendorName = vend;

vendorAddress = vendadd;

goods = item;

Qty = Qty;

cost = Qty \* 2;

Invoice memory Inv ;//create a local Invoice structure that will contain the invoice info for ONE invoice

Inv.nameFrom=compName;

Inv.from=compAdd;

Inv.to = vendadd;

Inv.nameTo = vend;

Inv.value=cost;

Inv.id = counter++;

Inv.status = 'Inprogress';

invoices.push(Invoice({nameFrom:compName,from:compAdd,nameTo:vend,to:vendadd,value:cost,id:counter++,status: 'inprogress'}));

uint j = invoices.length-1;

return (invoices[j].id);

PState = POState.Invoice;

}

function getInvoiceValue(uint \_invoiceID) returns (address Toadd,string \_vendName,uint invid, uint val, string stat)

{

for(uint i=0; i<invoices.length;i++){

if(invoices[i].id==\_invoiceID){

\_vendName = invoices[i].nameTo;

invid = invoices[i].id;

val = invoices[i].value;

stat = invoices[i].status;

benef = invoices[i].to;

return (benef,\_vendName,invid,val,stat);

}

}

//uint i = invoices[\_comp].length;

//invid = invoices[i].id;

//val = invoices[i].value;

//\_companyName = invoices[i].compName;

//stat = invoices[i].status;

}

modifier isInvoice() {

require(PState == POState.Invoice);

\_;

}

//function pay(address Add, uint amount, uint \_invoiceID) public payable

function pay( uint \_invoiceID) public payable isInvoice() returns (address benef,string stat)

{

//LogDepositMade(Add,amount);

for(uint i=0; i<invoices.length;i++){

if(invoices[i].id==\_invoiceID){

benef = invoices[i].to;

invoices[i].status="Paid";

stat = invoices[i].status;

benef.transfer(this.balance);

return(benef,stat);

break;

}

}

}

}