PG BC Capstone Project 1 Code

```
pragma solidity ^0.5.11;

contract KYC{

    address public contractOwner;
    constructor() public {
        contractOwner = msg.sender;
    }
    modifier onlyOwner(){
        require(msg.sender == contractOwner);
        _;//similar to continue keyword
    }

    struct Bank{
        string bankName;
        address bankAddress;
        bool isAddingCustomerAllowed;
        bool isInitiatingKycAllowed;
    }
     mapping(address=>Bank) public banks;


    function addBank(string memory _bankName) public onlyOwner{
        address bankAddress = msg.sender;
        banks[bankAddress].bankName = _bankName;
        banks[bankAddress].bankAddress = msg.sender;
        banks[bankAddress].isAddingCustomerAllowed = false;
        banks[bankAddress].isInitiatingKycAllowed = false;

    }
    struct Customer{
        string custName;
        string custAddress;
        string custMobile;
        bool kycStatus;
        address bankAddress;
    }
    mapping(string=>Customer) public customers;

    function addCustomer(string memory _custName,
    string memory custAddress,string memory custMobile, address bankAddress)
public onlyOwner returns (string memory message){
        bool val = banks[bankAddress].isAddingCustomerAllowed;
        if(val){
        customers[_custName].custName = _custName;
        customers[_custName].custAddress = custAddress;
        customers[_custName].custMobile = custMobile;
        customers[_custName].bankAddress = bankAddress;
        customers[_custName].kycStatus = false;
        message =  "Customer Added Successfully!!!";
        }
        else{
        message =  "Bank is not allowed to add customer";
        }
        return message;

    }

    function checkKyc(string memory _custName) public view returns(bool){
        return customers[_custName].kycStatus;
    }

    function doKYC(string memory _custName) public returns (string memory 
message) {
        address bankAddress = customers[_custName].bankAddress;
        bool check = banks[bankAddress].isInitiatingKycAllowed;

        if(check){
            customers[_custName].kycStatus = true;
            message = "Hurray!! KYC Completed";
        }
        else{
            message = "Bank not allowed to do kyc";
        }
        return message;
    }

    function blockBankToAddCustomers(address add) public onlyOwner{
        banks[add].isAddingCustomerAllowed = false;
    }
      function allowBankToAddCustomers(address add) public onlyOwner{
        banks[add].isAddingCustomerAllowed = true;
    }
      function blockBankToDoKYC(address add) public onlyOwner{
        banks[add].isInitiatingKycAllowed = false;
    }
      function allowBankToDoKYC(address add) public onlyOwner{
        banks[add].isInitiatingKycAllowed = true;
    }
    function queryCustomerByName(string memory _custName)
    public view returns(
        string memory custName,
        string memory custAddress,
        string memory custMobile,
        bool kycStatus,
        address bankAddress) {
            return(
                customers[_custName].custName,
                customers[_custName].custAddress,
                customers[_custName].custMobile,
                customers[_custName].kycStatus,
                customers[_custName].bankAddress
            );
    }






}
```