**Online Shopping System and Banking System**

**Data and Class Definitions**

By:

Pedro Arredondo, Luke Brennan,

Daemon Nelson, and Brandon Villalobos

**Class Definitions**

**Core**

|  |
| --- |
| <<control>>  Core.Core |
| \_DEBUG : Boolean |
| loadConfig(type : String) : HashMap<String, String>  main(args : String[]) : void |

|  |
| --- |
| <<control>>  Core.Buffer |
| messageBuffer : byte[]  responseBuffer : byte[]  messageBufferFull : boolean  responseBufferFull : boolean |
| send(message : byte[]) : byte[]  receive() : byte[]  reply(response : byte[]) : voide |

|  |
| --- |
| <<control>>  Core.SystemThread, extends Thread |
| buffer : Core.Buffer  type : String  config : HashMap<String, String>  isExiting : boolean |
| constructor(type : String, buffer : Core.Buffer)  setExiting(exiting : boolean) : void  run() : void |

**Banking System**

|  |
| --- |
| <<application logic>>  BankingSystem.Data.DataObject |
| No attributes |
| fill(record : byte[]) : void  id() : int  setID(int id) : void  recordLength() : int  serialize() : byte[]  datafile() : String |

|  |
| --- |
| <<application logic>>  BankingSystem.Data.DataManager |
| \_DEBUG : boolean  \_CLEANUP : boolean  \_CONFIG : HashMap<String, String>  \_BUFFER : Core.Buffer |
| encode(v : int) : byte[]  encode(v : long) : byte[]  encode(v : float) : byte[]  encode(v : String, l : int) : byte[]  decodeFloat(b : byte[]) : float  decodeLong(b : byte[]) : long  decodeInt(b : byte[]) : int  decodeString(b : byte[]) : String  main(args : String[]) : void  initialize(config : HashMap<String, String>, buffer : Core.Buffer) : void  getBuffer() : Core.Buffer  getDir(id : String) : String  getDBDir(id : String) : String  create(data : BankingSystem.Data.DataObject) : boolean  read(data : BankingSystem.Data.DataObject, id : int) : byte[]  read(data : BankingSystem.Data.DataObject, s : int, n : int) : ArrayList<byte[]>  update(data : BankingSystem.Data.DataObject, id : int) : boolean  delete(data : BankingSystem.Data.DataObject, id : int) : boolean |

|  |
| --- |
| <<entity>>  BankingSystem.Data.BankAccount  implements BankingSystem.Data.DataObject |
| accountID : int  creditCard : String  creditCardLength : int  balance : float  recordLength : int  datafile : String |
| constructor(accountID : int, creditCard : String, balance : float)  constructor()  getCreditCard() : String  getBalance() : float  addBalance(amount : float) : void |

|  |
| --- |
| <<entity>>  BankingSystem.Data.Request  implements BankingSystem.Data.DataObject |
| action : byte  idtype : byte  accountID : int  creditCard : String  creditCardLength : int  balance : float  recordLength : int  datafile : String |
| constructor(action : String, idType : String, accountID : int, creditCard : String, balance : float)  constructor(record : byte[])  constructor()  execute() : long |

|  |
| --- |
| <<application logic>>  BankingSystem.RequestManager.Main |
| \_DEBUG : boolean |
| main(config : HashMap<String, String>, buffer : Core.Buffer) : void |

**Online Shopping System**

|  |
| --- |
| <<application logic>>  OnlineShoppingSystem.Data.DataObject |
| No attributes |
| fill(record : byte[]) : void  id() : int  setID(int id) : void  recordLength() : int  serialize() : byte[]  datafile() : String |

|  |
| --- |
| <<application logic>>  OnlineShoppingSystem.Data.DataManager |
| \_DEBUG : boolean  \_CLEANUP : boolean  \_CONFIG : HashMap<String, String>  \_BUFFER : Core.Buffer |
| encode(v : int) : byte[]  encode(v : long) : byte[]  encode(v : float) : byte[]  encode(v : String, l : int) : byte[]  decodeFloat(b : byte[]) : float  decodeLong(b : byte[]) : long  decodeInt(b : byte[]) : int  decodeString(b : byte[]) : String  main(args : String[]) : void  initialize(config : HashMap<String, String>, buffer : Core.Buffer) : void  getBuffer() : Core.Buffer  getDir(id : String) : String  getDBDir(id : String) : String  create(data : BankingSystem.Data.DataObject) : boolean  read(data : BankingSystem.Data.DataObject, id : int) : byte[]  read(data : BankingSystem.Data.DataObject, s : int, n : int) : ArrayList<byte[]>  update(data : BankingSystem.Data.DataObject, id : int) : boolean  delete(data : BankingSystem.Data.DataObject, id : int) : boolean |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.BankRequest  implements OnlineShoppingSystem.Data.DataObject |
| action : byte  idtype : byte  accountID : int  creditCard : String  creditCardLength : int  balance : float  recordLength : int  datafile : String |
| constructor(action : String, idType : String, accountID : int, creditCard : String, balance : float)  constructor()  execute() : long |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.CustomerAccount  implements OnlineShoppingSystem.Data.DataObject |
| customerID : int  username : String  usernameLength : int  password : String  passwordLength : int  name : String  nameLength : int  address : String  addressLength : int  creditCard : String  creditCard : int  isPremium : boolean  premPaid : boolean  recordLength : int  dataFile : String |
| constructor(action : String, idType : String, accountID : int, creditCard : String, balance : float)  constructor()  getUsername() : String  getPassword() : String  setPassword(password : String) : void  getName() : String  setName(name : String) : void  getAddress() : String  setAddress(address : String) : void  getCreditCard() : String  setCreditCard(creditCard : String) : void  isPremium() : boolean  setPremium(isPremium : boolean) : void  premPaid() : boolean  setPremPaid(premPaid : boolean) : void  create() : boolean  update() : boolean  delete() : boolean  Login(username : String, password : String) : OnlineShoppingSystem.Data.CustomerAccount |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.Item  implements OnlineShoppingSystem.Data.DataObject |
| itemID : int  supplierID : int  itemName : String  itemNameLength : int  itemDesc : String  itemDescLength : int  itemRegCost : float  itemPremCost : float  itemQty : int  reservedQty : int  recordLength : int  dataFile : String |
| constructor(supplierID : int, itemName : String, itemDesc : String, itemRegCost : float,  itemPremCost : float itemQty : int)  constructor(record : byte[])  constructor()  getSupplierID() : int  getItemDesc() : String  setItemDesc(itemDesc : String)  getItemRegCost() : float  setItemRegCost(itemRegCost : float) : void  getItemPremCost() : float  setItemPremCost(itemPremCost : float) : void  getItemQty() : int  setItemQty(itemQty : int) : void  getReservedQty() : int  setReservedQty(reservedQty : int) : void  create() : boolean  update() : boolean  delete() : boolean  getItem(itemID : int) : OnlineShoppingSystem.Data.Item  getItems(supplierID : int) : ArrayList<OnlineShoppingSystem.Data.Item> |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.Order  implements OnlineShoppingSystem.Data.DataObject |
| orderID : int  customerID : int  orderStatus : byte  orderDate : int  orderCost : float  orderDelivery : byte  invoiceID : int  purchaseAuth : long  recordLength : int  dataFile : String |
| constructor(customerID : int, orderStatus : byte, orderDate : int, orderCost : float, orderDelivery : byte)  constructor(record : byte[])  constructor()  getCustomerID() : int  getOrderStatus() : byte  setOrderStatus(orderStatus : byte) : void  getOrderDate() : int  setOrderDate(orderDate : int)  getOrderCost() : float  setOrderCost(orderCost : float)  getInvoiceID() : int  getPurchaseAuth() : long  create() : boolean  update() : boolean  delete() : boolean  purchase(creditCard : String) : boolean  getOrder(orderID : int) : OnlineShoppingSystem.Data.Order  getOrders(customerID : int) : ArrayList<OnlineShoppingSystem.Data.Order> |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.OrderItem  implements OnlineShoppingSystem.Data.DataObject |
| orderID : int  customerID : int  orderStatus : byte  orderDate : int  orderCost : float  orderDelivery : byte  invoiceID : int  purchaseAuth : long  recordLength : int  dataFile : String |
| constructor(orderID : int, itemID : int, itemQty : int, orderItemStatus : byte)  constructor(record : byte[])  constructor()  getOrderID() : int  getItemID() : int  getItemQty() : int  setItemQty(itemQty : int) : void  getOrderItemStatus() : byte  setOrderItemStatus(orderItemStatus : byte) : void  create() : boolean  update() : boolean  delete() : boolean  getOrderItem(orderItemID : int) : OnlineShoppingSystem.Data.OrderItem  getOrderItems(id : int, mode : char) : ArrayList<OnlineShoppingSystem.Data.OrderItem> |

|  |
| --- |
| <<entity>>  OnlineShoppingSystem.Data.SupplierAccount  implements OnlineShoppingSystem.Data.DataObject |
| supplierID : int  username : String  usernameLength : int  password : String  passwordLength : int  name : String  nameLength : int  recordLength : int  dataFile : String |
| constructor(action : String, idType : String, accountID : int, creditCard : String, balance : float)  constructor()  getUsername() : String  getPassword() : String  setPassword(password : String) : void  getName() : String  setName(name : String) : void  create() : boolean  update() : boolean  delete() : boolean  Login(username : String, password : String) : OnlineShoppingSystem.Data.SupplierAccount  getSupplier(supplierID : int) : OnlineShoppingSystem.Data.SupplierAccount  getSuppliers() : ArrayList<OnlineShoppingSystem.Data.SupplierAccount> |

|  |
| --- |
| <<application logic>>  OnlineShoppingSystem.UI.Main  extends Application |
| No attributes |
| main(config : HashMap<String, String>, buffer : Core.Buffer) : void  start(primaryStage : Stage) : void |

**Database Files**

The various data files for both the Online Shopping System and Banking System act as distinct databases. Both systems have a Data.DataManager class that acts as a control mechanism for CRUD operations with minimal query mechanics, e.g. read one or read all. Class-implementations of data table records contain wrapper methods for the centralized CRUD operations. Both BankingSystem.Data.Request and OnlineShoppingSystem.Data.BankRequest are designed to use their respective Data.DataManager: however, they are not stored in database files and act only as processing constructs. Default save directories are listed: however, they are customizable using the configuration files generated when running Core.Core.

|  |
| --- |
| [*BankingSystem*].[*Data*].[*BankAccount*]  BankingSystem/DataBase/BankAccounts.dat |
| accountID : int (index)  creditCard : nvarchar  balance : decimal |

|  |
| --- |
| [*OnlineShoppingSystem*].[*Data*].[*CustomerAccount*]  OnlineShoppingSystem/DataBase/CustomerAccounts.dat |
| customerID : int (index)  username : nvarchar  password : nvarchar  name : nvarchar  address : nvarchar  creditCard : nvarchar  isPremium : bit  premPaid : bit |

|  |
| --- |
| [*OnlineShoppingSystem*].[*Data*].[*Item*]  OnlineShoppingSystem/DataBase/Items.dat |
| itemID : int (index)  supplierID : int (foreign key : [*OnlineShoppingSystem*].[*Data*].[*SupplierAccount*].supplierID)  itemName : nvarchar  itemDesc : nvarhar  itemRegCost : decimal  itemPremCost : decimal  itemQty : int  reservedQty : int |

|  |
| --- |
| [*OnlineShoppingSystem*].[*Data*].[*Order*]  OnlineShoppingSystem/DataBase/Orders.dat |
| orderID : int (index)  customerID : int (foreign key : [*OnlineShoppingSystem*].[*Data*][*CustomerAccount*].customerID)  orderStatus : tinyInt  orderDate : int  orderCost : decimal  orderDelivery : tinyint  invoiceID : int  purchaseAuth : bigint |

|  |
| --- |
| [*OnlineShoppingSystem*].[*Data*].[*OrderItem*]  OnlineShoppingSystem/DataBase/OrderItems.dat |
| orderItemID : int (index)  orderID : int (foreign key : [*OnlineShoppingSystem*].[*Data*].[*Order*].orderID)  itemID : int (foreign key : [*OnlineShoppingSystem*].[*Data*].[*Item*].itemID)  itemQty : int  orderItemStatus : tinyint |

|  |
| --- |
| [*OnlineShoppingSystem*].[*Data*].[*SupplierAccount*]  OnlineShoppingSystem/DataBase/SupplierAccounts.dat |
| supplierID : int  username : nvarchar  password : nvarchar  name : nvarchar |

**Message Buffer Details**

The message buffering process is split between multiple classes. The Core.Buffer class is used as a synchronization point for the one producer and one consumer of the buffering process, which exist within the OnlineShoppingSystem.Data.BankRequest and BankingSystem.Data.Request classes respectively. In order for the downstream classes to appropriately reference their buffer, the buffer is assigned statically within the OnlineShoppingSystem.Data.DataManager and BankingSystem.Data.DataManager classes and referenced by their respective producing/consuming classes through appropriately accessible methods.