

**Addis Ababa Institute of Technology**

**Center of Information Technology and Scientific Computing**

**Department of ITSC**

**eMazon**

**Software Design Specification**

Submitted to: Instructor Natnael

Team Members

1. Elbetel Gezahegn ATR/3445/08
2. Kimiya Mohammed ATR/2765/08
3. Lulit Mulugeta ATR/3806/08
4. Rahel Getachew ATR/0621/08
5. Yeabsira Gashaw ATR/6749/08
6. Yonas Alem ATR//08
7. Abel Girma ATR/1442/08

Date: 24 March 2018

Table of Contents

[List of Tables ii](#_Toc451061527)

[List of figures iii](#_Toc451061528)

[Definitions, Acronyms, Abbreviations iv](#_Toc451061529)

[1. Introduction 1](#_Toc451061530)

[1.1 Purpose 1](#_Toc451061531)

[1.2 General Overview 1](#_Toc451061532)

[1.3 Development Methods & Contingencies 2](#_Toc451061533)

[2. System design model 2](#_Toc451061534)

[2.1 Subsystem decomposition 2](#_Toc451061535)

[2.2 Hardware/software mapping 3](#_Toc451061536)

[3. Object Model 4](#_Toc451061538)

[3.1 Class Diagram 4](#_Toc451061539)

[3.2 Sequence Diagram 5](#_Toc451061540)

[4. Detailed Design 11](#_Toc451061542)

[References 21](#_Toc451061543)

# List of Tables

Table 1: Design description of guest class…………………………………13

Table 2: Attribute description for the guest class…………………………13

Table 3: Operation description for the guest class………………………..14

Table 4: Design description of item class………………………………….15

Table 5: Attribute description for the item class………………………….15

Table 6: Design description of admin class………………………………..16

Table 7: Attribute description for the admin class………………………..16

Table 8: Operation description for the admin class……………………….17

Table 9: Design description of authorized user class……………………..17

Table 10: Attribute description for the authorized class………………….18

Table 11: Operation description for the authorized class…………………19

Table 12: Design description of database class……………………………..19

Table 13: Attribute description for the database class……………………..19

Table 14: Operation description for the database class……………………20

Table 15: Design description of report class………………………………...20

Table 16: Attribute description for the report class………………………...20

Table 17: Operation description for the report class……………………….20

# List of figures2

Figure 1: Context diagram of MVC architecture……………………………1

Figure 2: Component diagram layer 1………………………………………..2

Figure 3: Component diagram layer 2………………………………………..3

Figure 4: Deployment diagram………………………………………………..3

Figure 5: Class diagram ………………………………………………………..4

Figure 6: Sequence diagram login……………………………………………..5

Figure 7: Sequence diagram log out…………………………………………...5

Figure 8: Sequence diagram view profile……………………………………..6

Figure 9: Sequence diagram edit profile………………………………………6

Figure 10: Sequence diagram giving comment……………………………….7

Figure 11: Sequence diagram deactivate account…………………………….7

Figure 12: Sequence diagram search item……………………………………..8

Figure 13: Sequence diagram wish items………………………………………8

Figure 14: Sequence diagram add to cart………………………………………9

Figure 15: Sequence diagram ask refund………………………………………9

Figure 16: Sequence diagram delete product………………………………….10

Figure 17: Sequence diagram add product…………………………………….10

Figure 18: Sequence diagram signup……………………………………………11

Figure 19: Sequence diagram checkout…………………………………………12

# Definitions, Acronyms, Abbreviations

HTTP: Hyper Text Transfer Protocol

OOP: object oriented programming

TCP: Transmission Communication Protocol

PHP: Hypertext Pre-processor

HTML: Hyper-Text Markup Language

CSS: Cascading Style Sheet

SQL: Structured Query Language

DBMS: Database Management System

# Introduction

## 1.1 Purpose

The purpose of this designed document is to provide a guideline throughout the process of developing eMazone online shopping website. It is going to be used to translate the business requirements and business processes into a technical design and verify whether these requirements have been addressed.

## General Overview

The architecture we are going to use to implement eMazon is MVC, since it is one of the best architectures to separate application logic from the user interface. MVC mainly divides the application in to three independent but interconnected components: model, view and controller. The model expresses the application’s behavior in terms of the problem domain. The view is responsible for presenting the model in a particular format. The Controller represents classes connecting the model and the view, and is used to communicate between classes in the model and view.

MVC increases the usability, security and maintainability of an application because it provides discrete tiers which are self-contained and separated components that have clear boundaries. As a result we will be able to fulfill some of our non-functional constraints.

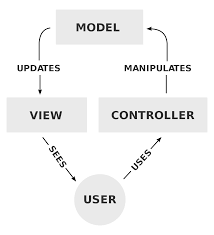


Figure 1: Context diagram for MVC architecture

## Development Methods & Contingencies

* It will be written in HTML5 and PHP7.2 . HTML5 is the largest version of HTML which allows writing clear and descriptive codes. PHP is known for OOP which makes it preferable to use. It suits the eMazon online shop development because it allows to create dynamic web pages easily, quickly, and also to handle user interactions. Additionally, the latest version of web styling, CSS3, Jquery will be used to make the website interface attractive.
* Since the system will be database driven, the system will use MongoDB database which is good enough for performance and has decent documentation.
* The tool that will be used to write the codes is visual studio code. This IDE provides a rich code editor for PHP and HTML with syntax highlighting and auto code completion. It is preferable because it is very fast considering its size, supports many frameworks, and allow comparing other codes and compiling these different codes.
* This system will be running on Apache web server. Apache is closely available as it is cheap. Finally it will be tested on live linux server on hosted website.

# System Architecture

## 2.1 Subsystem decomposition

## C:\Users\das\Desktop\p1.png

Figure2: Component diagram layer 1

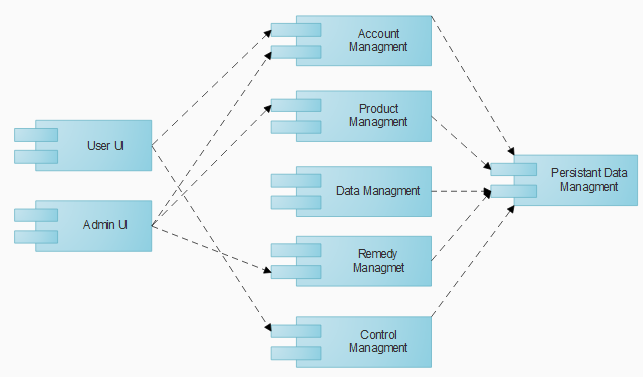


Figure 3: Component diagram layer 2

## Hardware/software mapping

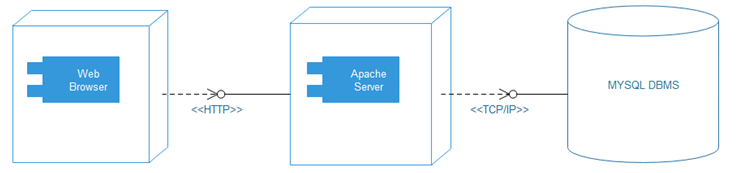


Figure 4: Deployment diagram

# 

# 3. Object Model

## 3.1 Class Diagram

## C:\Users\das\Desktop\classdig.png

## Figure 5: class diagram

## 3.2 Sequence Diagram

# D:\IMG_20180323_234236.jpg

# Figure 6: Sequence diagram login

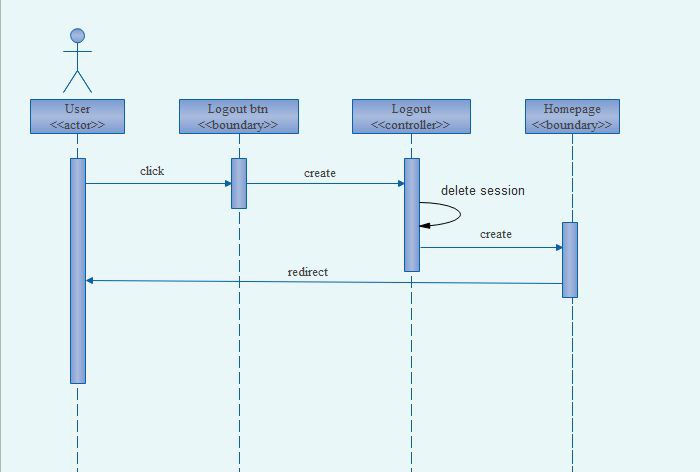


Figure 7: sequence diagram logout

# D:\IMG_20180323_235346.jpg

Figure 8: Sequence diagram view profile

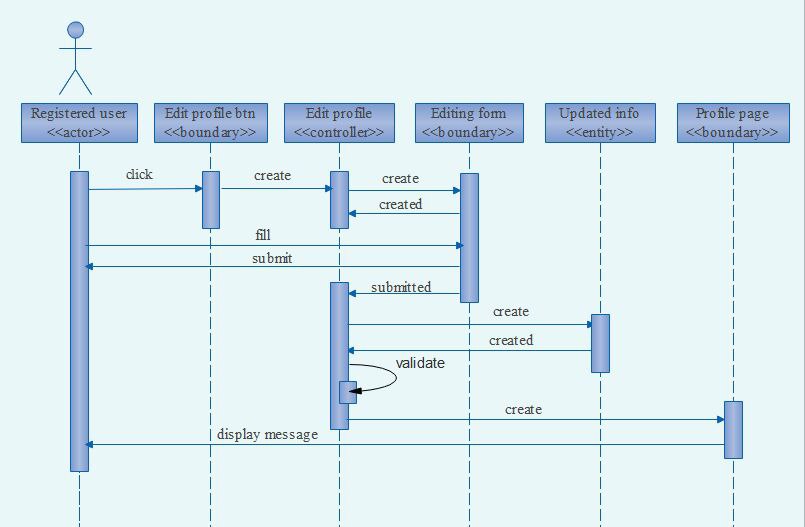


Figure 9: Sequence diagram Edit profile

# D:\IMG_20180323_235319.jpg

Figure 10: sequence diagram giving comment

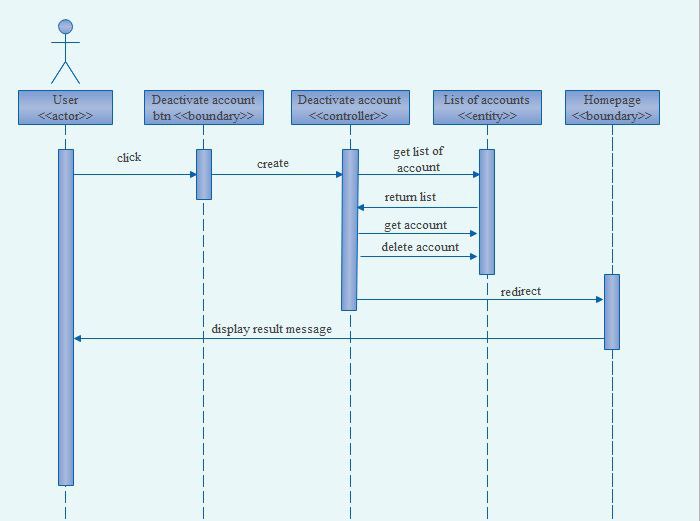


Figure 11: Sequence diagram deactivate account

# D:\IMG_20180323_235317.jpg

Figure 12: Sequence diagram searching item

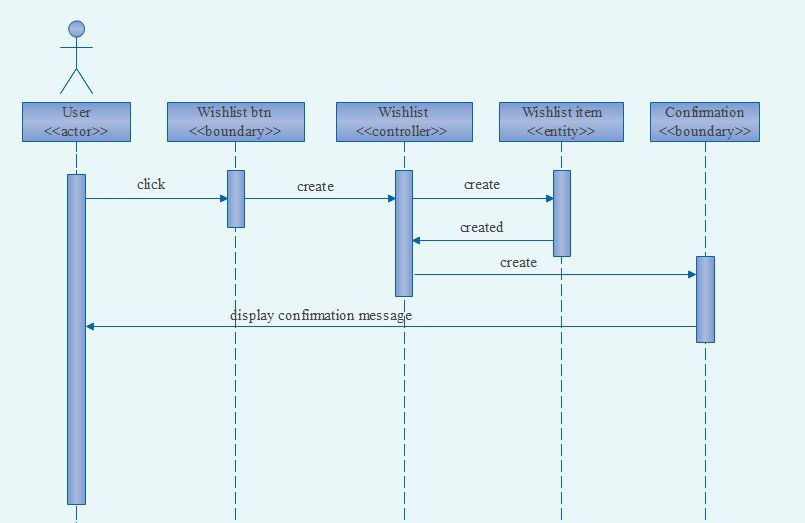


Figure 13: Sequence diagram wish items

# D:\IMG_20180323_235338.jpg

Figure 14: Sequence diagram add to cart

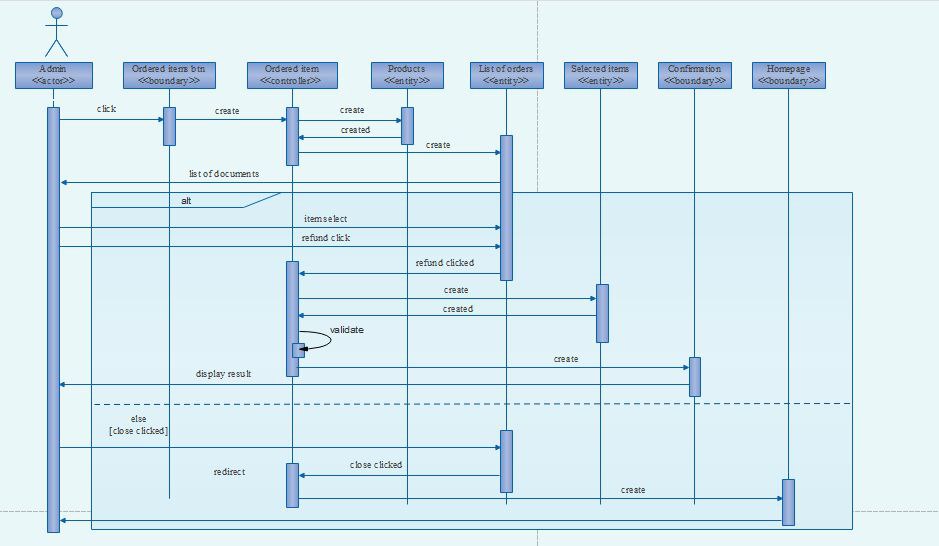


Figure 15: Sequence diagram ask refund

# D:\IMG_20180323_235336.jpg

Figure 16: Sequence diagram Delete product

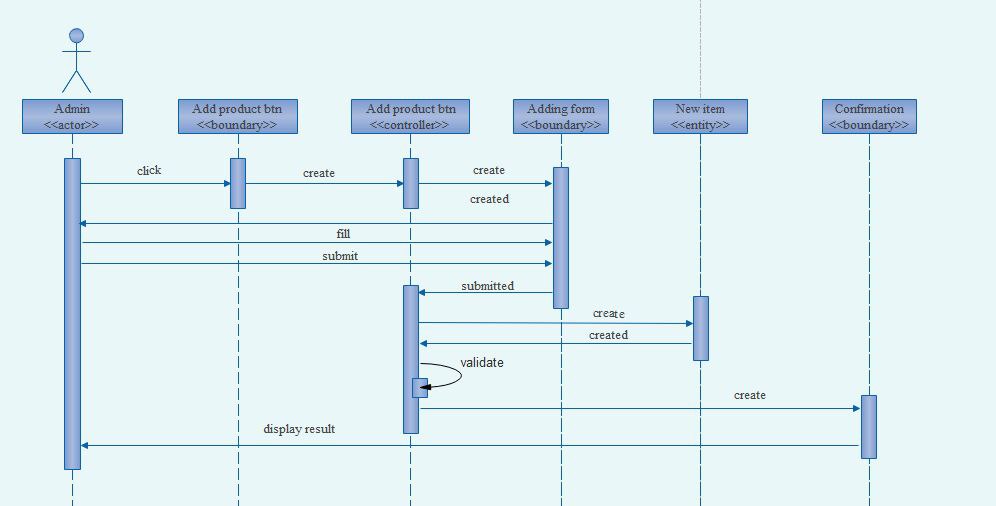


Figure 17: Sequence diagram add product

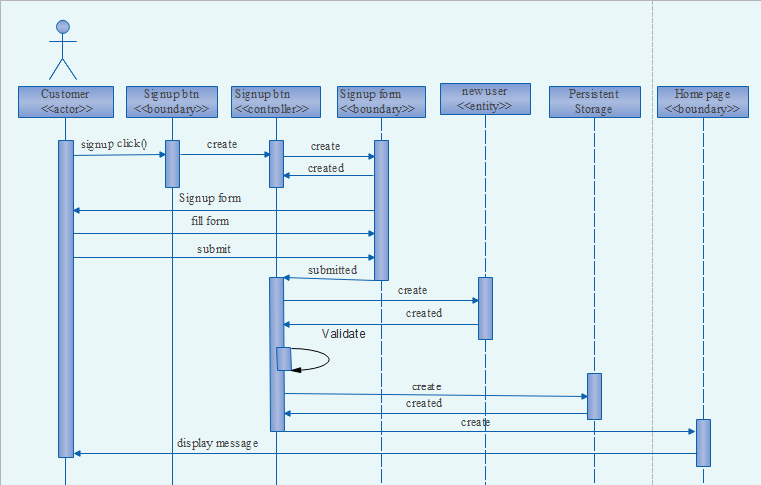


Figure 18: sequence diagram for Signup

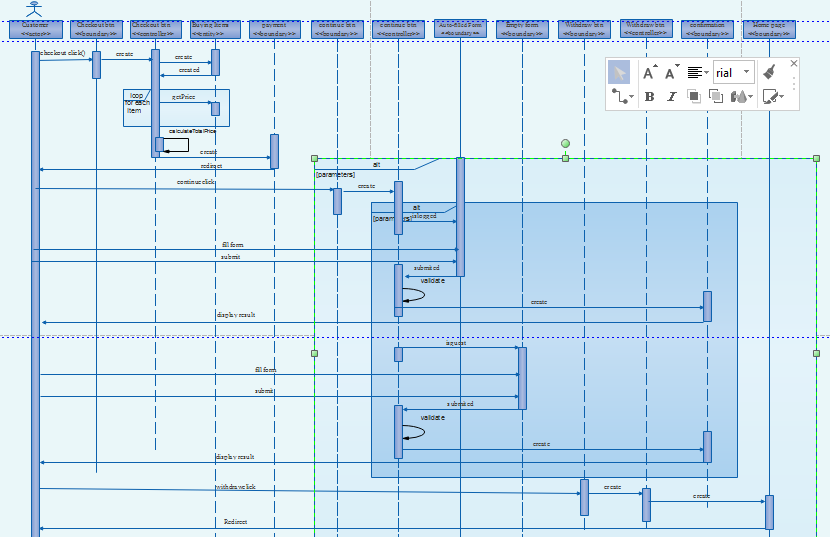


Figure 19: sequence diagram for checkout

# 4. Detailed Design

|  |
| --- |
| Guests |
| -accountId :String  -fullName:String  -email:String  -phoneNo.:String |
| +searchitem(itemID):void  +compareItem(itemPrice,itemType,ItemRating):bool  +buyItem(Fullname,email,accountId):void  #isVerified(userName):bool  +sendFeedback():void  +rateService():float  -register(fullName,email,String newPassword,String new userName):void  -checkOut(itemId,itemPice):void  -askRemed(itemId,Date checkoutDate,phoneNO.):void  -getHotDeals(ItemName,itemType,itemPrice,itemRate):void |

Table1: Guest class

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Invariant |
| fullName | String | private | fullName<>Null ,it must must contain full and descriptive Name |
| Accounted | String | private | accountId<>Null |
| Email | String | private | email <>NULL  ->should contain “@” character and after it and some strings it should contain “.” |
| phoneNo. | String | private | phoneNo<>NULL |

Table 2: Attribute description for the Guest class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation | Visibility | Return type | argument | Pre-Condition | Post-condition |
| searchItem | public | void | itemId:string | many items thier butnot the prefered one | Get the desired item |
| compareItem | public | void | itemPrice:String  itemRate:float  itemType:String | All kind of item displayed with no comparison | Items compared based on the parameters given. |
| buyItem | private | void | fullName:String  AccountID:String | Item are compared or displayed | Item successfully bought. |
| isVerified | protected | Boolean | userName:String | The user previledge is unknown. | Categorize the user as guest or authorized. |
| sendFeedback | private | void |  | Browse the page | Send their idea about the system. |
| rateService | public | float | itemType:String  itemPrice:float  itemID:String | Must buy and see the product. | Rate the item |
| Register | private | void | fullName:String  Email:String  phoneNO.:String  New UserName:String  New password:String | the user use services as a guest. | User authorized successfully and get legal account. |
| Checkout | private | void | itemId:String  itemPrice:Float | The users categorize or search the item. | The user checks the out successfully. |
| askRemedy | private | void | itemId:String  checkoutDate:Date  phoneNO.:String | Make sure the item have warranty and damaged. | Set the remedy based on the damage. |
| getHotDealItems | public | void | itemName:String  itemType:String  itemPrice:Float  itemRating:Float | Item displayed catagorically or randomly | Get the best wanted items. |

Table3: Operation description for the Guest class

|  |
| --- |
| Items |
| +ItemName:String  -itemtId:String  +itemType:String  +itemPrice:float  +itemComment:String  +itemRating:float  +itemState:bool  +itemDetail:String |

Table 4: Item class

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Invariant |
| itemName | String | public | itemName<>  Full and descriptive |
| itemId | String | protected | itemId<>NULL |
| itemType | String | public | itemType<>NULL  Should describe thier category efficiently. |
| itemPrice | Float | public | itemPrice<>NULL |
| itemComment | String | public | itemComment<>  Should bes short and brief. |
| itemRating | Float | public | itemRating<>NULL |
| itemState | bool | public | itemState<>NULL |
| itemDetial | String | public | itemDetail<>  Should provide the necessary information about the item. |

Table 5: Attribute description for the Product class

|  |
| --- |
| Admin |
| -userName:String  -password:String |
| +Login(userName,password):void  -addItem(itemId,itemName,itemPrice,itemRate,itemStatus):void  -updateItem(itemId,itemName,itemPrice,itemRate,itemStatus):void  -deleteItem(itemId):void  +searchItem(itemId):void  -manageUsers(userName,timeStamp):void  -categorizeProduct(productId,productType,productPrice,productRating):void  -manageRemedy(userName,Email,phoneNo.):void |

Table 6: Admin class

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | type | visibility | invariant |
| Username | String | private | userName<>NULL |
| Password | String | private | Password<>NULL |

Table 7: Attribute description for Admin class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation | visibility | Return type | argument | Pre-Condition | Post-Condition |
| Login | public | void | userName:String  Password:string | The Admin enter to the Admin browser page | The Admin logged into Admin interface. |
| Additem | private | void | itemId:String  itemName:String  ItemPrice:Float  itemRate:Float  itemStatus:bool | No product item or kind their | Item added |
| updateItem | private | void | itemId:String  itemName:String  ItemPrice:Float  itemRate:Float  itemStatus:bool | Item out stoke or add other. | Item updated and available for users. |
| deleteItem | private | void | itemId:String | Item out Stock | Item removed from the page and database. |
| searchItem | private | void | itemId:String | Item not displayed in the admin page | Item found or not found because if status. |
| manageUsers | private | void | userName:String  TimeStamp:Date | User doesnot used the account overTime or the account have some problem. | Delete the user if he does not user overtime and resolve the account problem. |
| categorizeProduct | private | void | itemType:String  itemName:String  itemPrice:Float | Product added to database. | Catagorize items based on the parameters given. |
| mangeRemedy | private | void | checkoutDate:Date  reportId:String | Feedback from users that the item is damaged. | Check the item date and report or receipt and give the users remedy. |

Table 8: Operation Description for Admin class

|  |
| --- |
| Authorized users |
| -userName:String  -Password:String |
| +searchItem(itemId,itemName,ItemPrice,itemRating):void  -buyProduct(accountId):void  -Login(userName,password):void  -resetPassword(email):void  -Addwishlist(itemName,itemId):void  -deletewishlist(itemId):void  -Updatewishlist(itemId):void  -sendFeedback():void  +rateProduct(itemId,itemName):void  -Askremedy(itemid,Date checkoutDate):void |

Table 9: class design for Authorized user class

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | type | Visibility | invariant |
| Username | String | Private | userName<>NULL |
| Password | String | Private | Password<>NULL |

Table 10: Attribute Description for the authorized user class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation | visibility | Return type | argument | Pre-Condition | Post-Condition |
| Login | public | void | userName:String  Password:string | The authorized enter to the Admin browser page | The Authorized user logged into Admin interface. |
| resetPassword | private | void | email:String | Forget password or user name  Or password is weak. | Reset the previous one and get the new one. |
| searchItem | private | void | itemId:String  itemName:String  ItemPrice:Float  itemRate:Float  itemStatus:bool | Item unavailable in the front | Item gotten and available for users. |
| addtoWishlist | private | void | itemId:String  itemName:String | Item in Stock  Or out stock but unable to but based on money shortage. | Item added to wishlist and used as memo for the future. |
| Updatewishlist | private | void | itemId:String  itemName:String | Item needed but out of stock. | Item found and added to wishlist. |
| deleteWishlist | private | void | itemName:String | Item bought or unavailable anymore or get as a gift. | Item deleted from wishlist. |
| categorizeProduct | private | void | itemType:String  itemName:String  itemPrice:Float | Product added to database. | Catagorize items based on the parameters given. |
| askRemedy | private | void | checkoutDate:Date  reportId:String | the item is damaged. | Get the remedy based on damage. |
| sendFeedback | private | void | . | Buy or browse the items in stock | Feedback sent. |
| rateItem | public | void | ItemName:String  itemId:String | Item bought | Rate item successful |

Table 11: Operation description for authorized users

|  |
| --- |
| Database |
| #databaseName:string  #databaseId:string |
| +Connect():void  -addRecord(itemId,itemName,itemPrice,itemType):void  -updateRecord(itemId,itemName,itemPrice):void  -deleteRecord(itemId):void |

Table12: design class for database

|  |  |  |  |
| --- | --- | --- | --- |
| attribute | type | Visibility | invariant |
| databaseName | String | Private | databaseName<>NULL |
| databaseId | String | Private | databaseID<>NULL |

Table 13: Attribute description for the database class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation | visibility | Return type | arguments | Pre-condition | Post-Condition |
| connect | private | void | databaseName:String  userName:String  Password:String | Database created. | Connection successful. |
| addRecord | private | void | recordName:String  recordId:String  recordType:String | Data to be inserted | Record inserted. |
| updateRecord | private | void | recordId:String | Data to be inserted. | Data updated. |
| deleteRecord | private | void | Recordid:String | Data no longer needed | Data deleted from database. |

Table 14: Operation description for database class

|  |
| --- |
| Report and VAT |
| #itemId:String  #itemName:String  #itemPrice:String |
| -generateReport(itemId,itemName,itemPrice):void |

Table15: Design description for report class

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | invariant |
| itemId | String | Private | itemId<> NULL |
| itemName | String | Private | itemName<>NULL |
| itemPrice | String | Private | itemPrice<>NULL |

Table16: Attribute Description for report class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| operation | visibility | Return type | arguments | Pre-conditions | Post-Conditions |
| generateReport | private | Void | RecordId:String | Item bought | Report generated. |

Table 17: Operation description for the report class

**References**

Software-Engineering-Ian-Sommerville