Online Music Store:
Analysis and Design

Software Engineering and Testing

Institute of Technology Blanchardstown

2009

Adeola Rahmon Sanni B00029278

Juan Jesús Cervera Hinojosa B00029823

Table of Contents

1. Project Description

The aim of this project is to develop a online music store to the client *International Music Store*.

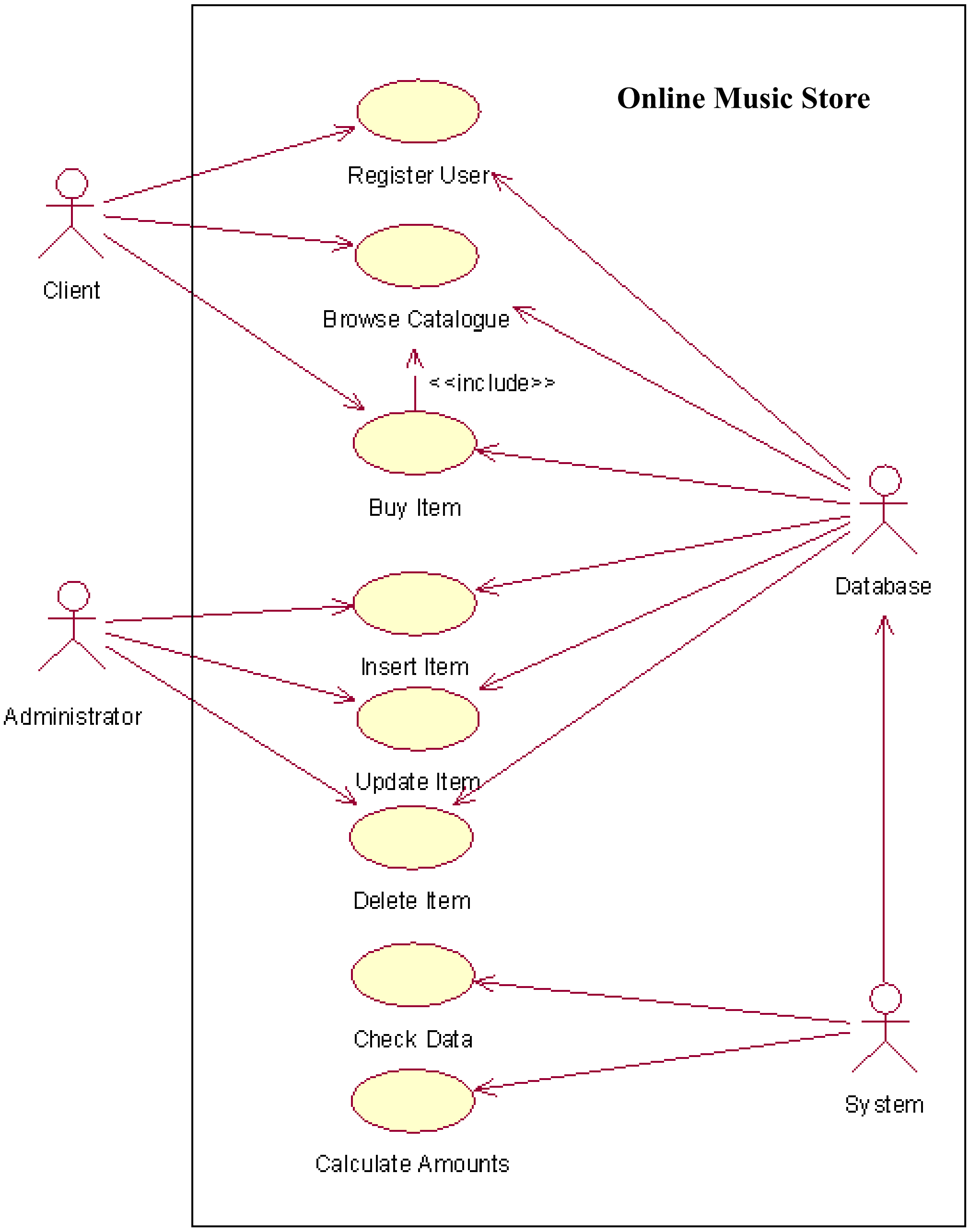
Via an online webpage the application will allow *International Music Store*'s users to access to music items descriptions, to make a selection of their favorite items and order their particular choice; besides they will receive an order confirmation.

2. UML Use Case Diagrams

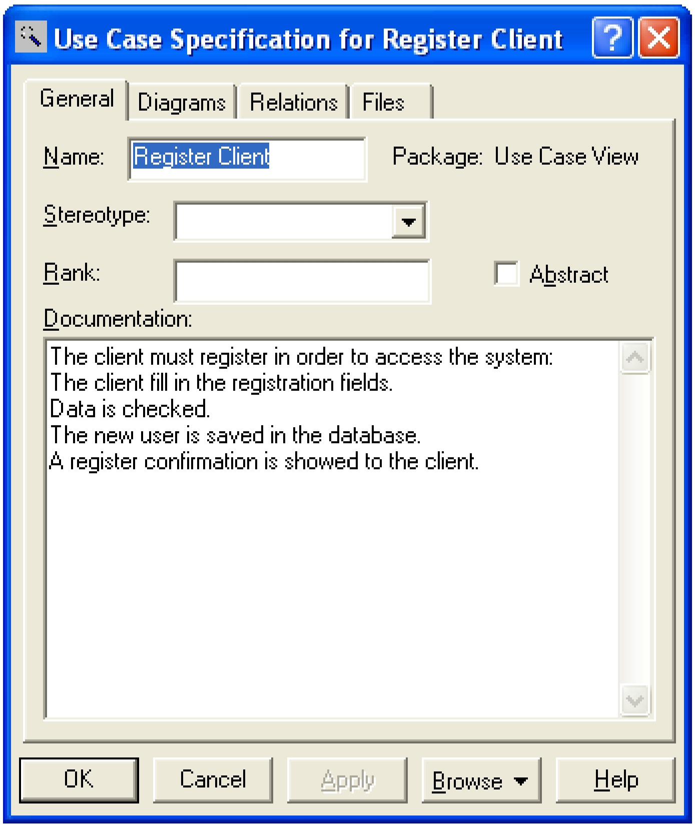
Use case diagrams describe interactions between the users of the system and the system itself.

The following use case diagram capture the functional requirements of the application for *International Music Store* from a external view of the system:

(Diagram in next page)


Online Music Store
Client
Administrator
Database
System
Register User
Browse Catalogue
<<include>>
Buy Item
Insert Item
Update Item
Delete Item
Check Data
Calculate Amounts

3. Use Case Specifications



Name: Register Client

Package: Use Case View

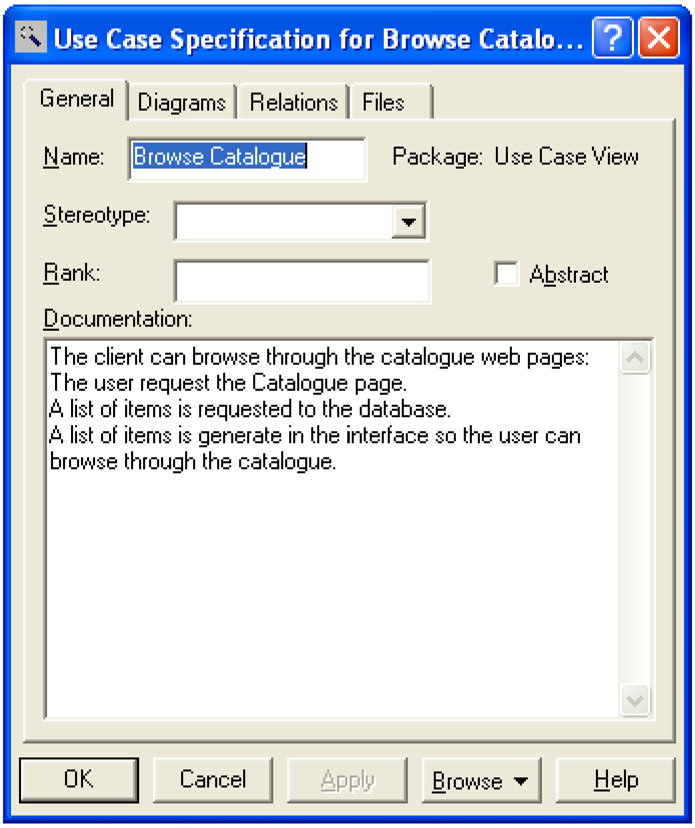
Stereotype:

Rank:

□ Abstract

Documentation:

The client must register in order to access the system:
The client fill in the registration fields.
Data is checked.
The new user is saved in the database.
A register confirmation is showed to the client.



Name: Browse Catalogue

Package: Use Case View

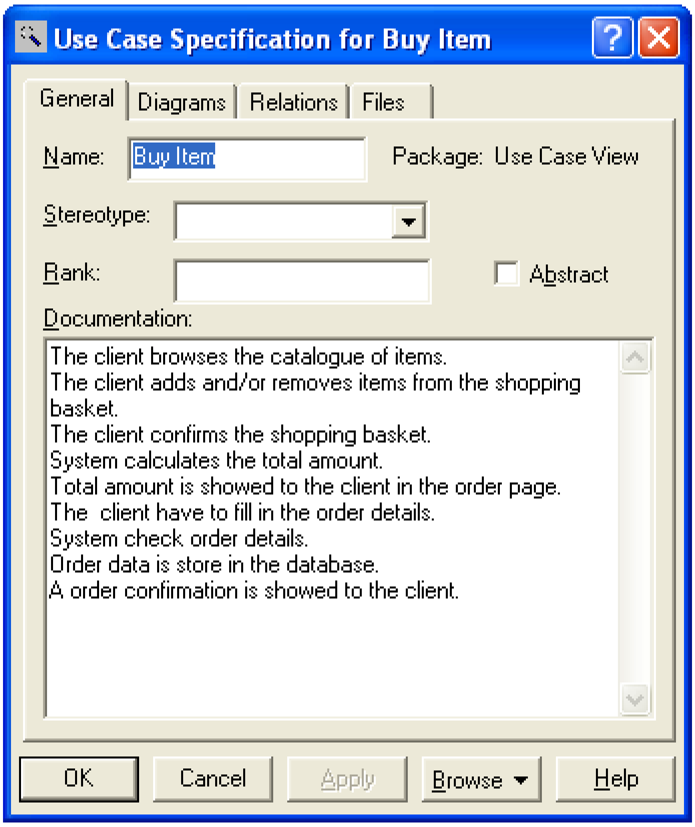
Stereotype:

Rank:

□ Abstract

Documentation:

The client can browse through the catalogue web pages:
The user request the Catalogue page.
A list of items is requested to the database.
A list of items is generate in the interface so the user can browse through the catalogue.



Name: Buy Item

Package: Use Case View

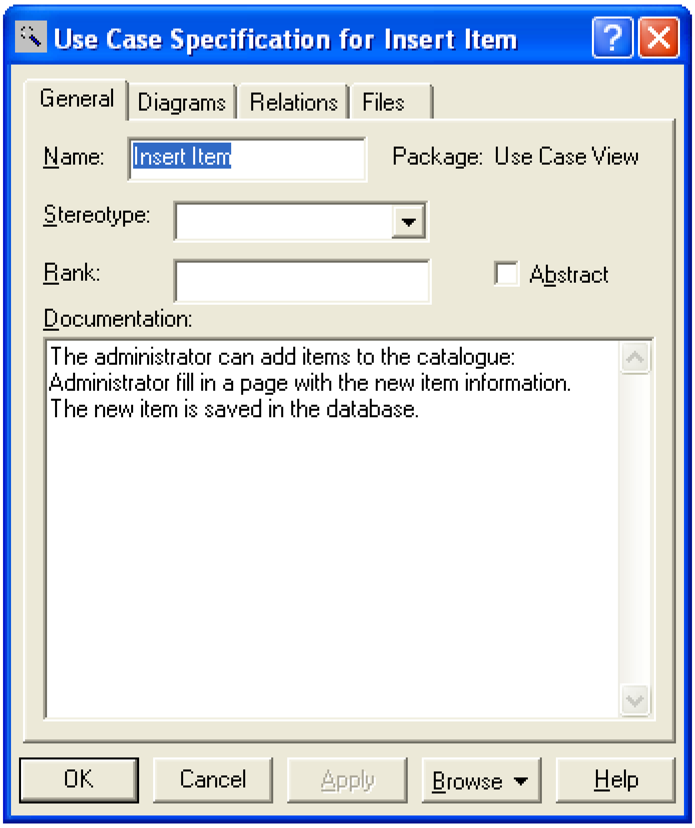
Stereotype:

Rank:

□ Abstract

Documentation:

The client browses the catalogue of items.
The client adds and/or removes items from the shopping basket.
The client confirms the shopping basket.
System calculates the total amount.
Total amount is showed to the client in the order page.
The client have to fill in the order details.
System check order details.
Order data is store in the database.
A order confirmation is showed to the client.



Name: Insert Item

Package: Use Case View

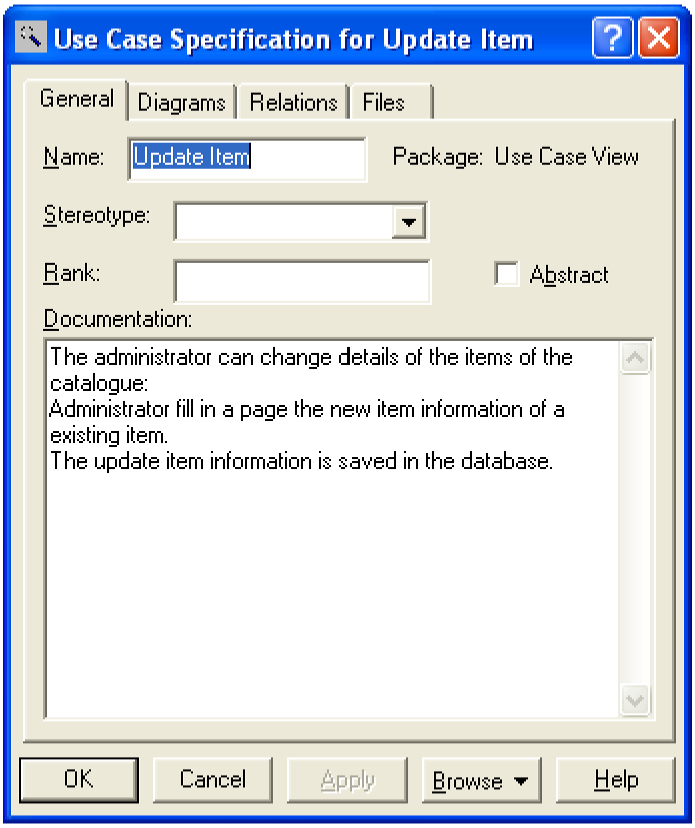
Stereotype:

Rank:

□ Abstract

Documentation:

The administrator can add items to the catalogue:
Administrator fill in a page with the new item information.
The new item is saved in the database.



Name: Update Item

Package: Use Case View

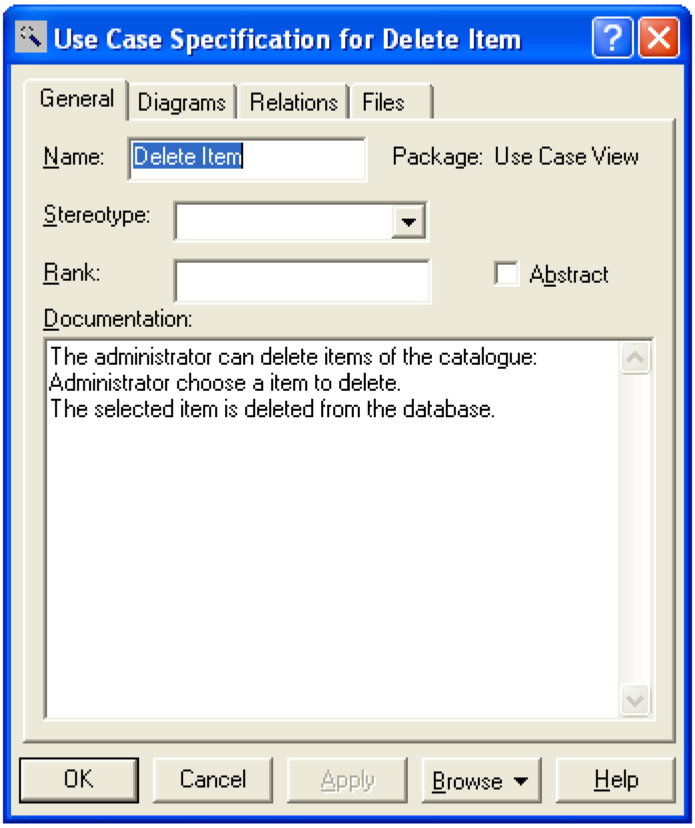
Stereotype:

Rank:

□ Abstract

Documentation:

The administrator can change details of the items of the catalogue:
Administrator fill in a page the new item information of a existing item.
The update item information is saved in the database.



Name: Delete Item

Package: Use Case View

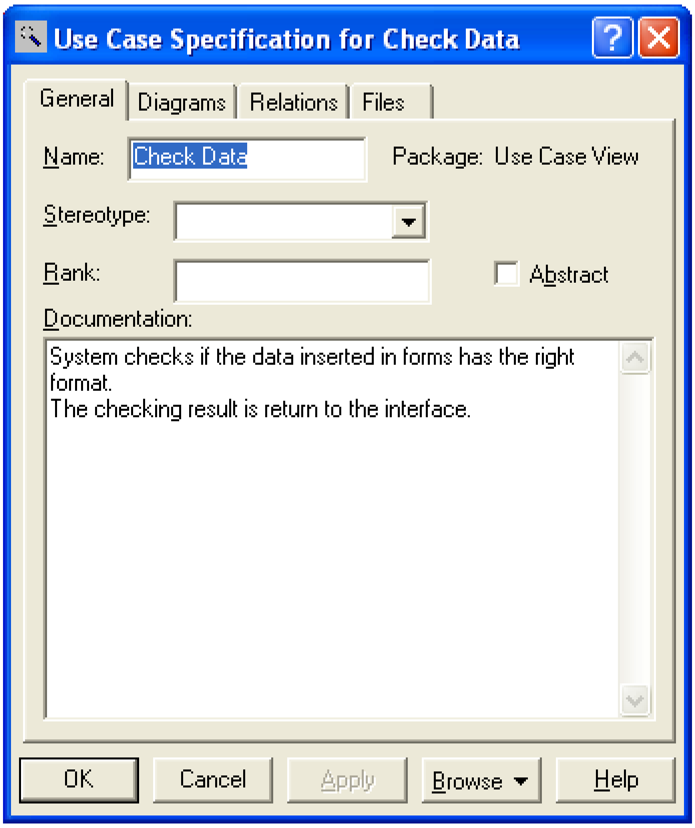
Stereotype:

Rank:

□ Abstract

Documentation:

The administrator can delete items of the catalogue:
Administrator choose a item to delete.
The selected item is deleted from the database.



Name: Check Data

Package: Use Case View

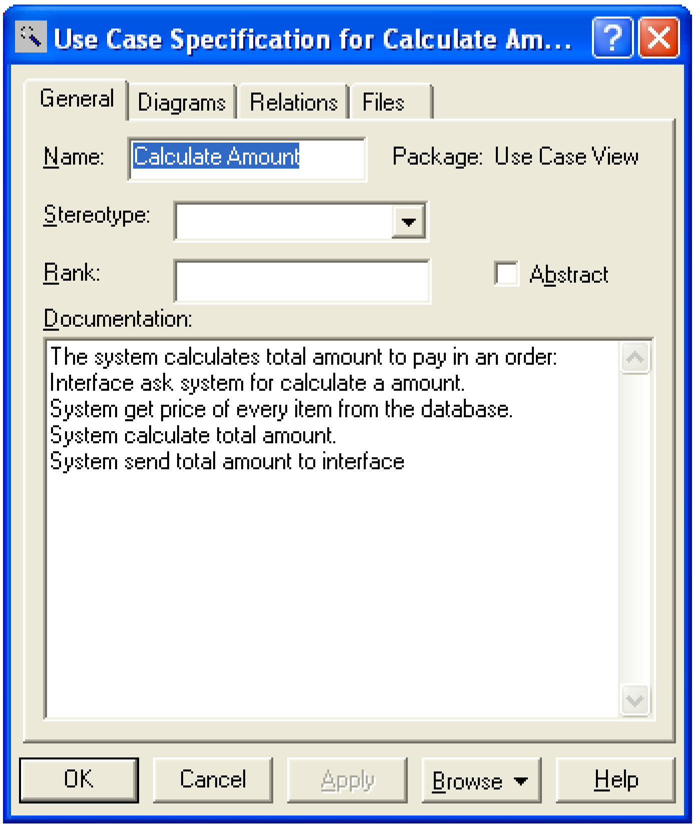
Stereotype:

Rank:

□ Abstract

Documentation:

System checks if the data inserted in forms has the right format.
The checking result is return to the interface.



Name: Calculate Amount

Package: Use Case View

Stereotype:

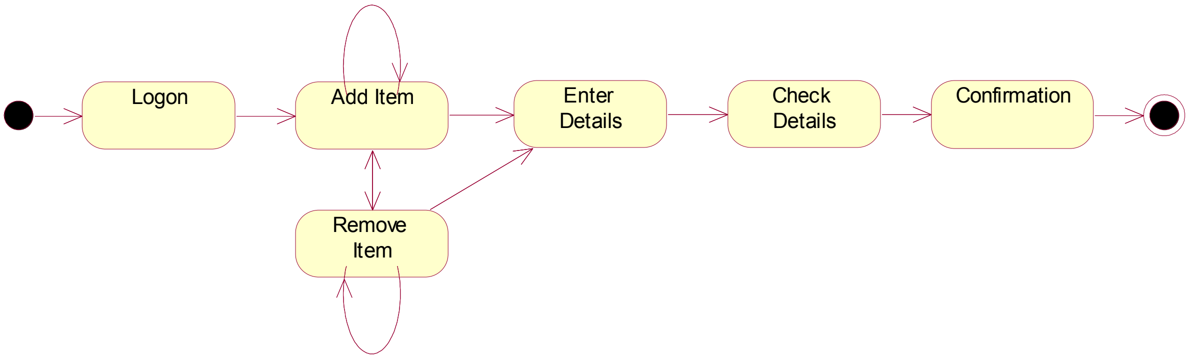
Rank:

□ Abstract

Documentation:

The system calculates total amount to pay in an order:
Interface ask system for calculate a amount.
System get price of every item from the database.
System calculate total amount.
System send total amount to interface

Activity diagram for the use case *Buy Item*:



4. Objects/Classes

After study and elaborate the system descriptions i.e. Use cases and their specifications, the next step is identify classes, attributes and operations by means of those system descriptions.

We have identified nouns and verbs in the specifications, this is a list of the results:

Nouns/things	**Verbs**
Shopping Basket	Buy
Username/password	Register
Client	Browse
Administrator	Calculate amount
Database	Logon
Catalogue	Add
Item	Remove
Data	Confirm
Amount	Insert
Credit card	Update
Address	Delete
Order	Check data
Interface	Authorize
Email	
Description/Image	
Price	

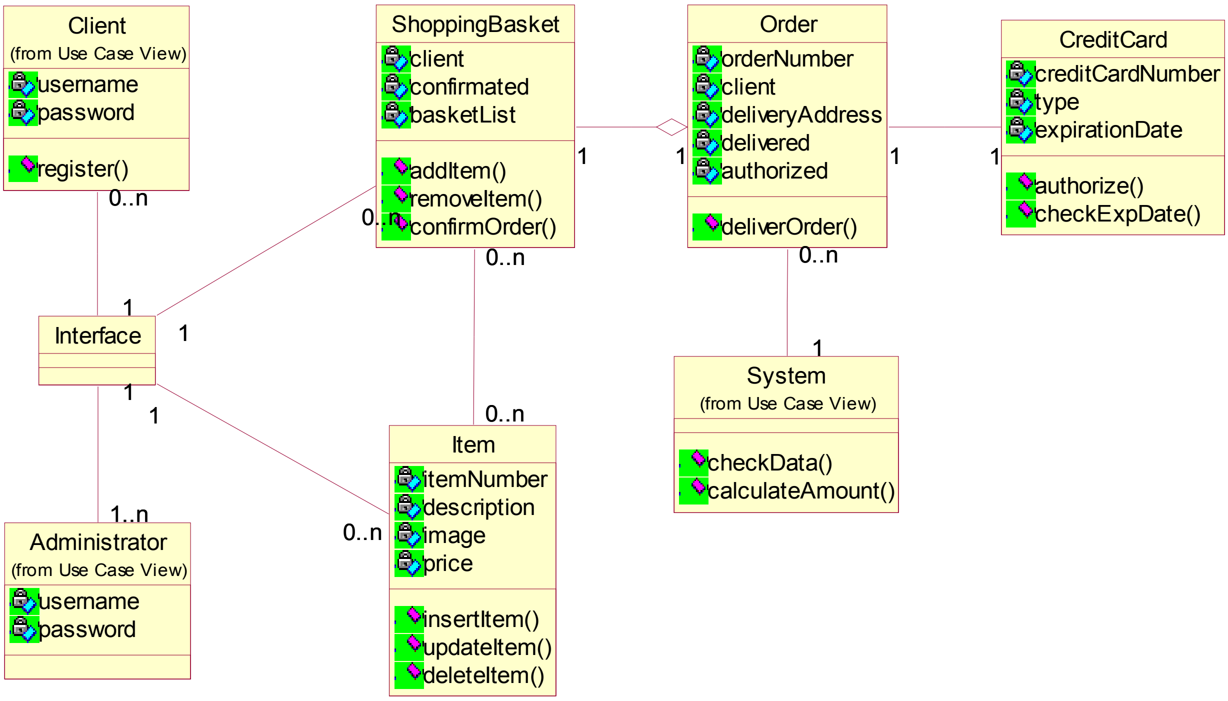
The nouns suggest classes and class attributes. The verbs suggest class operations:

- *Client*: It has username, password, email, address. It has to register in the application and logon.
- *ShoppingBasket*: It has an associated client. In the application we can add or remove items of the shopping basket.
- *Order*: It has an associated client, a delivery address and an associated shopping basket. We can confirm the order.
- *CreditCard*: We need to authorize a credit card to confirm an order.
- *Interface*: It defines store's clients perception of the application, it sends data between pages (HTML).
- *Item*: It has a description/image, a price. The administrator can insert, update or delete items.
- *Administrator*: It has username and password. It has to logon in the application.
- *System*: It check data of order and registration process and calculates the amount of every order.
- *Username/Password*: They are going to be attributes of client and administrator.
- *Catalogue*: It is a list of items, we consider this as a synonymous of Item class.

After analyze the list above we identify the classes and the relationships between the selected classes and the rest of nouns and verbs to get the classes' attributes and classes' operations. The final result is reflected in the UML Class Diagram.

5. UML Class Diagram

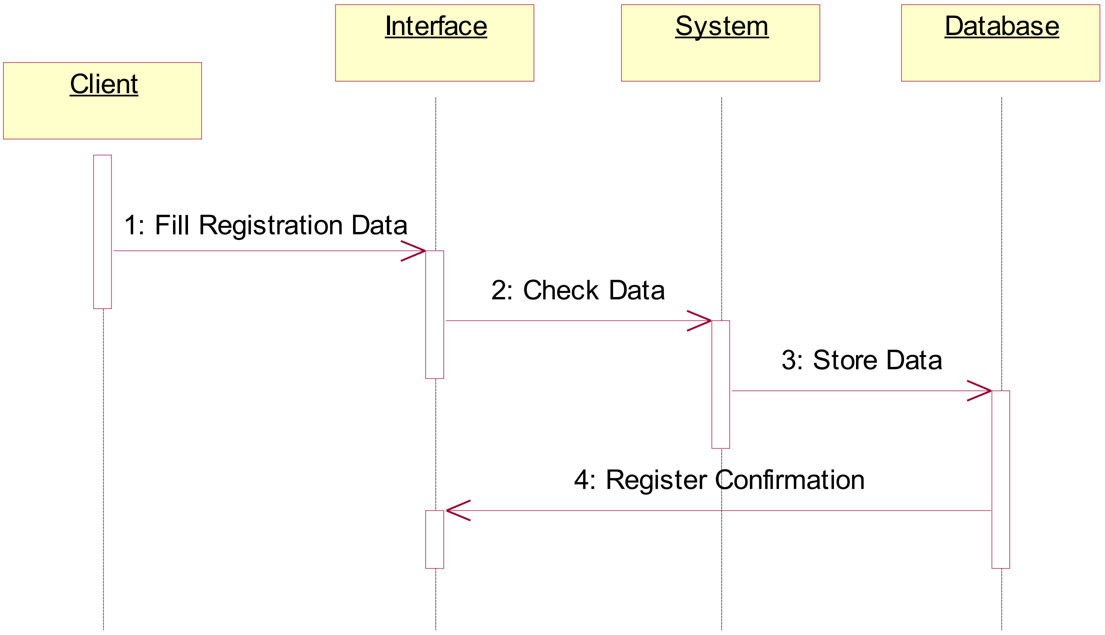
By means of the following diagram we show the classes and the relationships among them, getting an overview of the application.



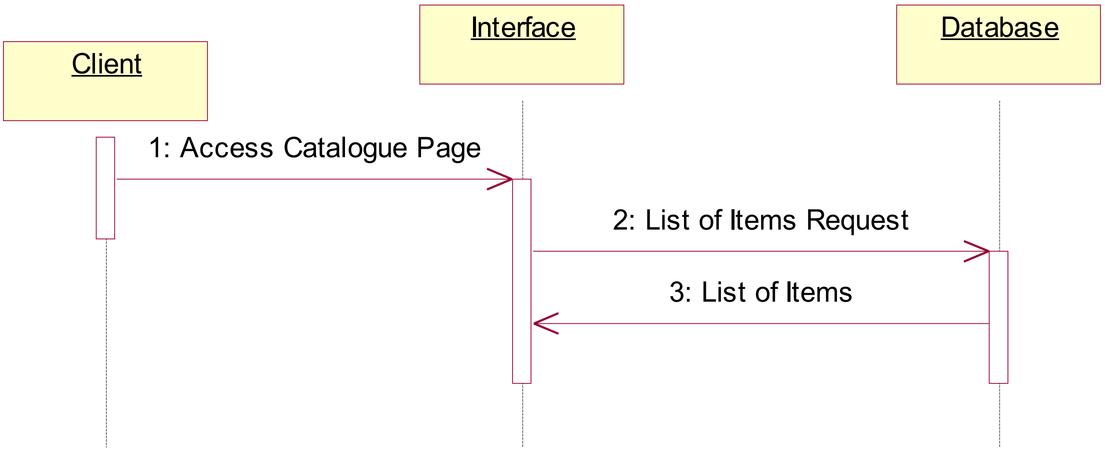
6. UML Interaction Diagram

By means of these interaction diagrams (sequence diagrams) we get a description of how the groups of objects in the application collaborate in the different behaviors.

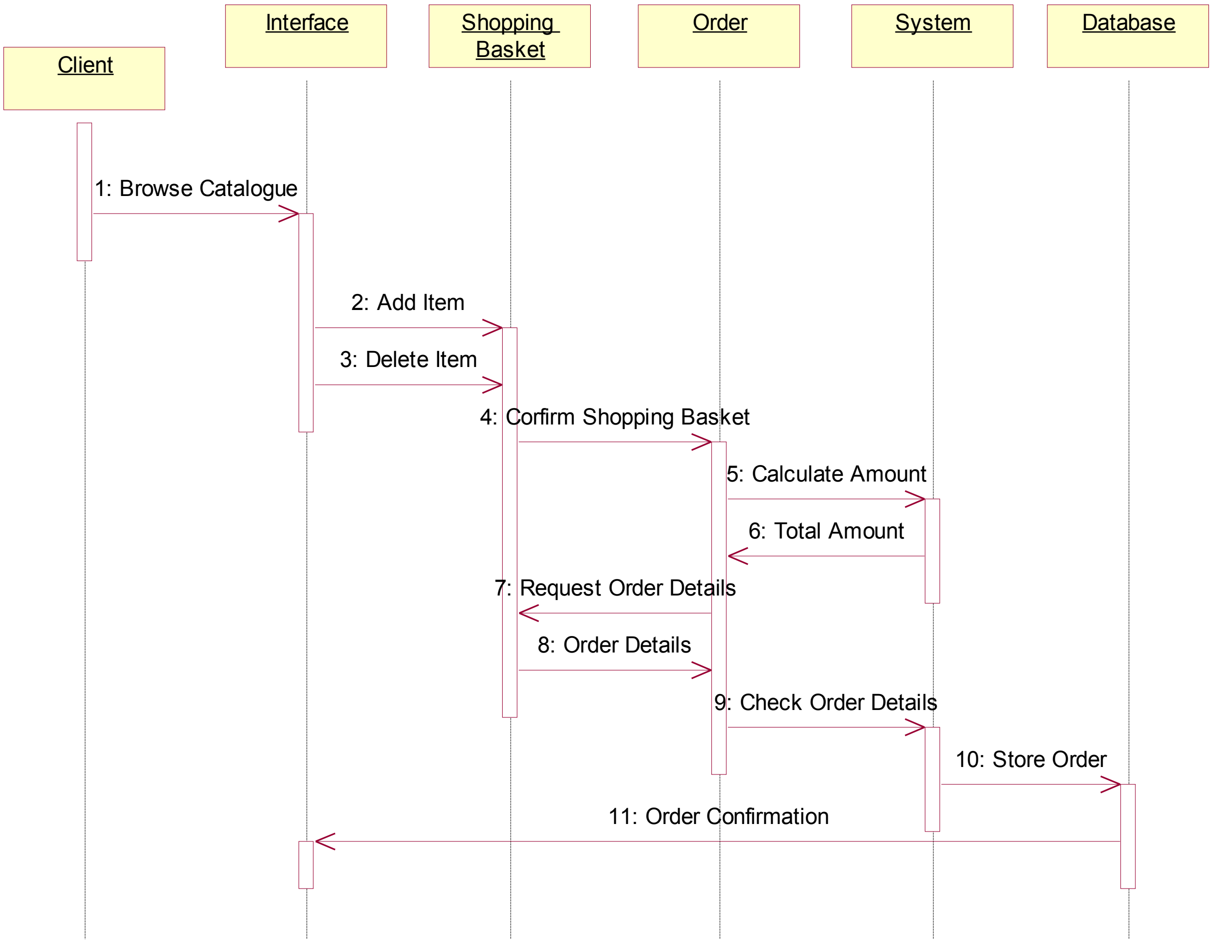
6.1 Register User



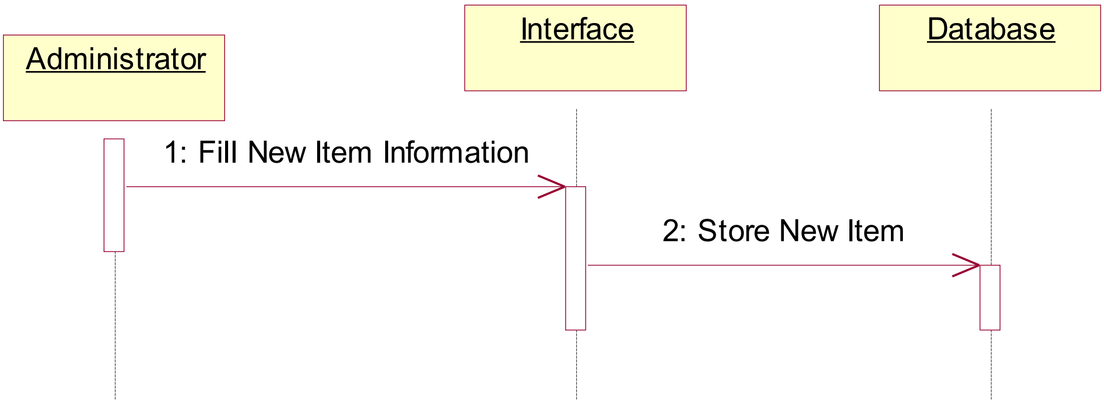
6.2 Browse Catalogue



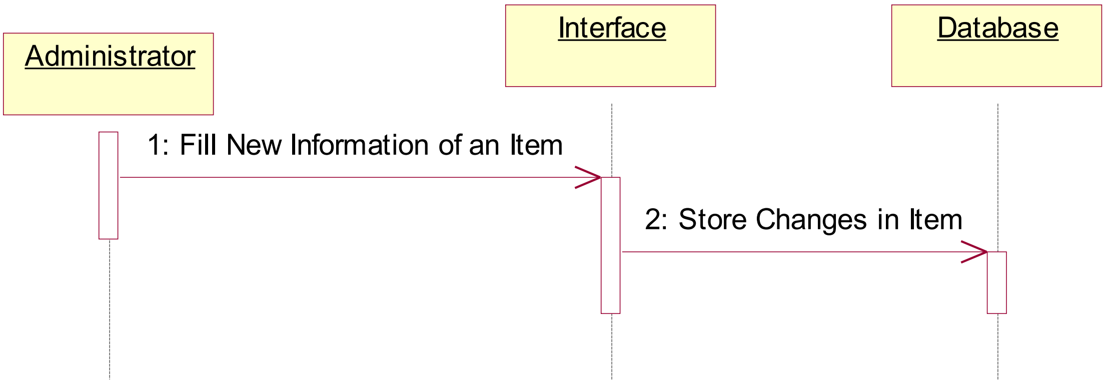
6.3 Buy Item



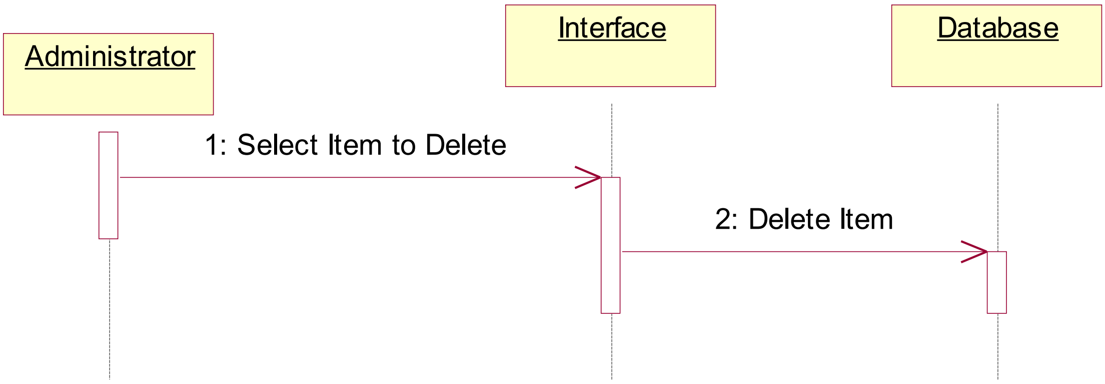
6.4 Insert Item



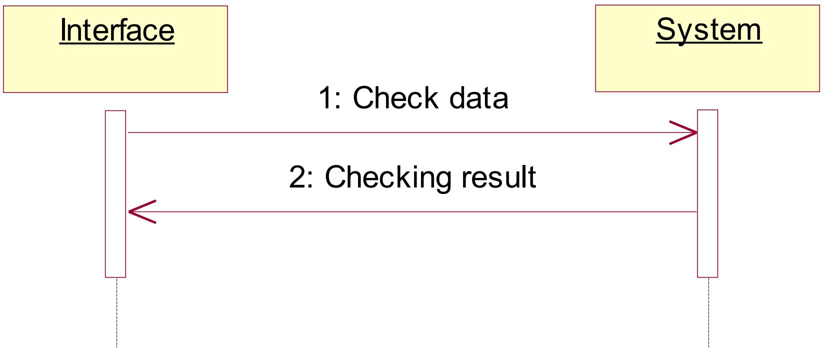
6.5 Update Item



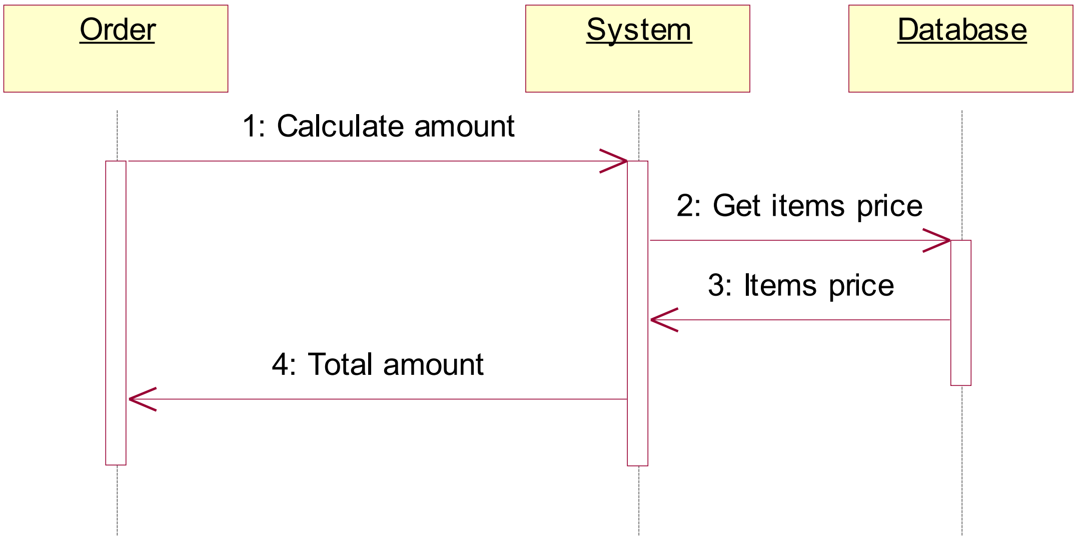
6.6 Delete Item



6.7 Check Data



6.8 Calculate Amount



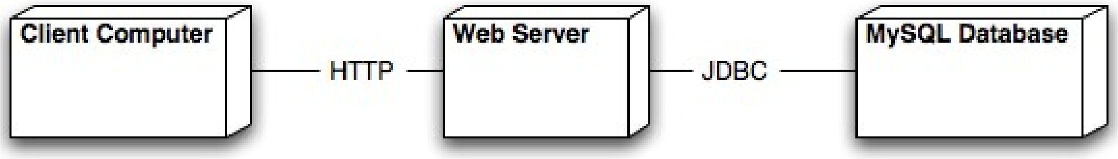
7. System Deployment Diagram

The system software will be implemented in Java SE using JDK 1.6. A standard PC will be used for system prototype (demonstration model) development and the JDK are readily available. A database will be used to store the users, the catalogue, the orders, etc. This will be developed in MySQL and will be interacted with through JDBC (Java Database Connectivity), all of which are readily available.

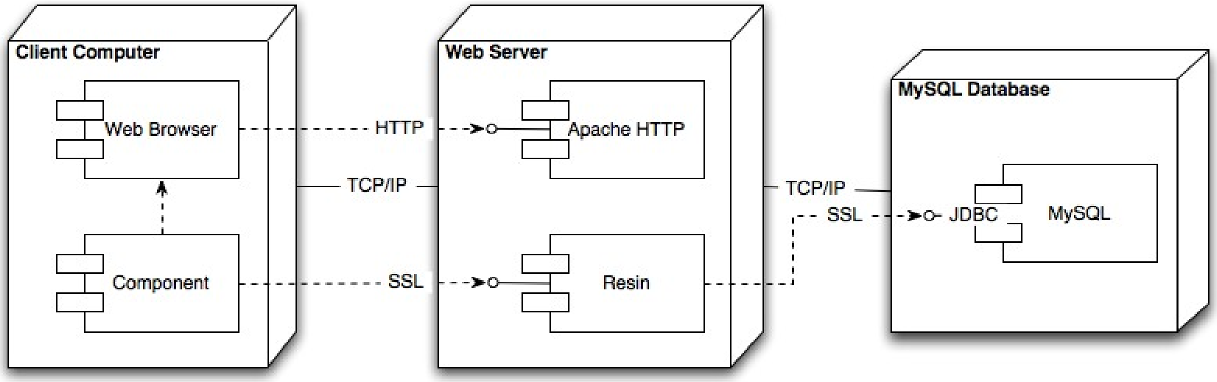
The application will work with the servlet container Resin 3.0.12 (Microsoft Windows). The servlet container will generate the HTML code that the user's web browser will interpreted.

The following deployment diagrams show the physical configurations of software and hardware of the application.

Deployment diagram with a basic view:



Deployment diagram with a detail view of components and nodes:

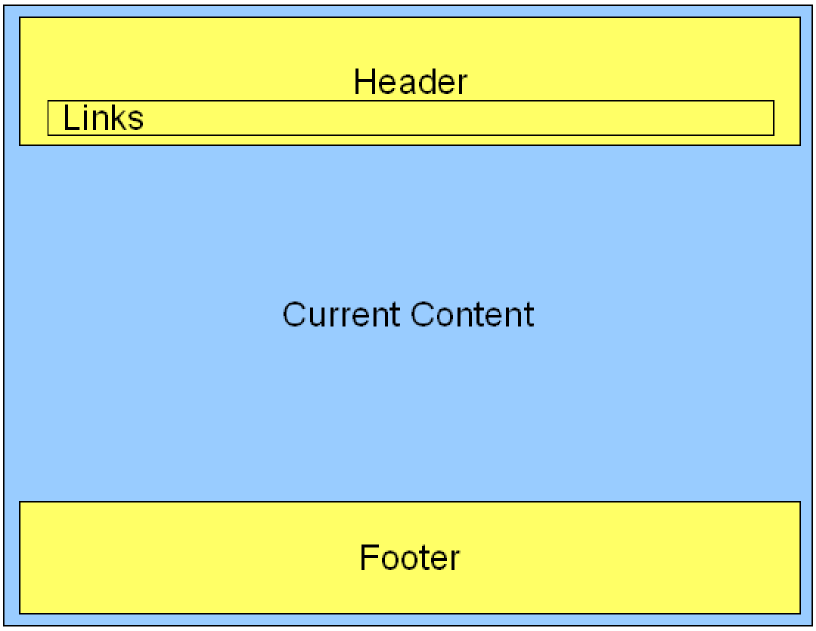


8. User Interface Design

User Interface is a really important part as it defines store's clients perception of the application.

“An application is only as good or bad as its user interface”. [McLoughlin, S., 2009]

In every page of the application we are going to keep the following layout to have a consistent placement of the different elements. In the header we will the name of the store and a list of links to access to all the functions of the application. In the center the user will show the information related to the last function selected. In the footer we will have the contact details of the store (physical and virtual addresses).



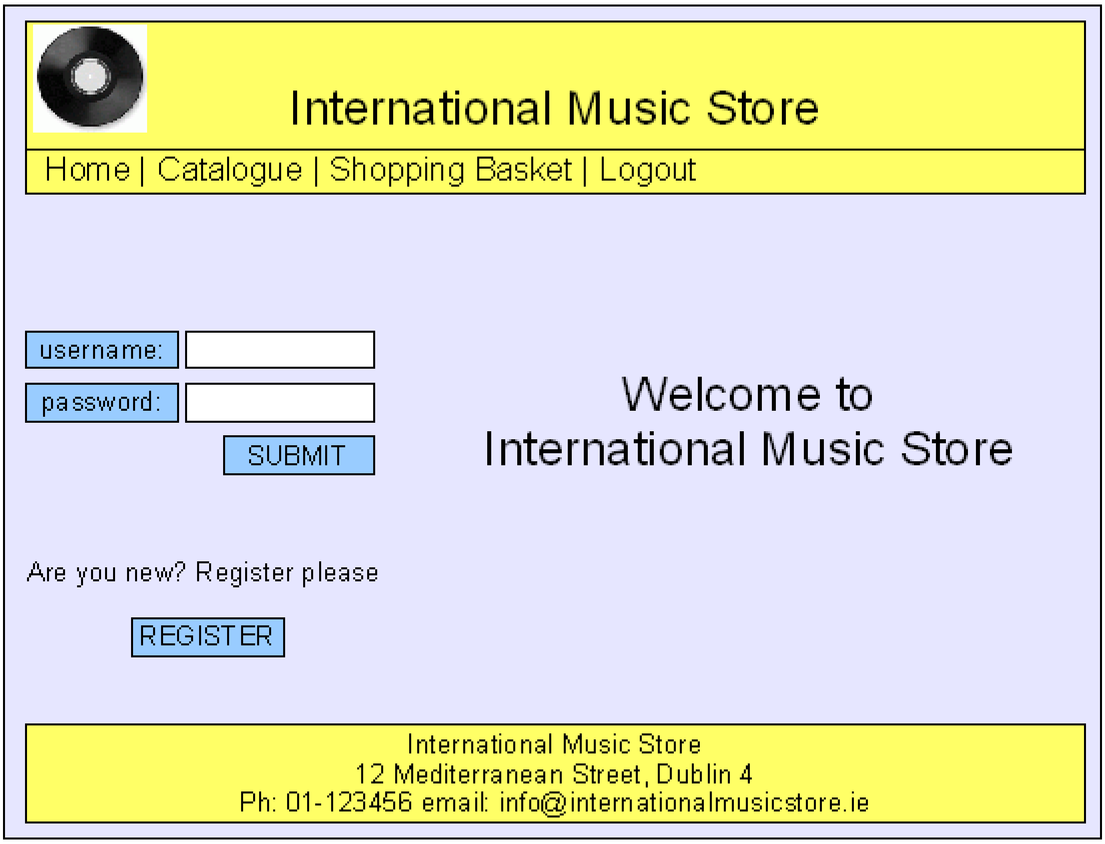
In the forms we will align fields properly by means of tables, keeping the same format in everyone of them for a symmetrical finish.

We will group different or complicate user's operations effectively, there are not going to be more than one different operation in the same page and if a operation is complicated with too much information we will try to break it up into multiple pages.

To make readable the pages the text will be dark and big and the background will be bright. So we help people color blind and get an accessible webpage.

The data inserted by the users is checked by the system. If the user makes a mistake or forgets to fill a necessarily field he will be redirect to the previous page to fill in it again.

Snapshot:



International Music Store

Welcome to International Music Store

Are you new? Register please

International Music Store
12 Mediterranean Street, Dublin 4
Ph: 01-123456 email: info@internationalmusicstore.ie

9. Database Schema

9.2 Entities Description

The database of Online Music Store is called *musicstore* and it is constituted by these entities:

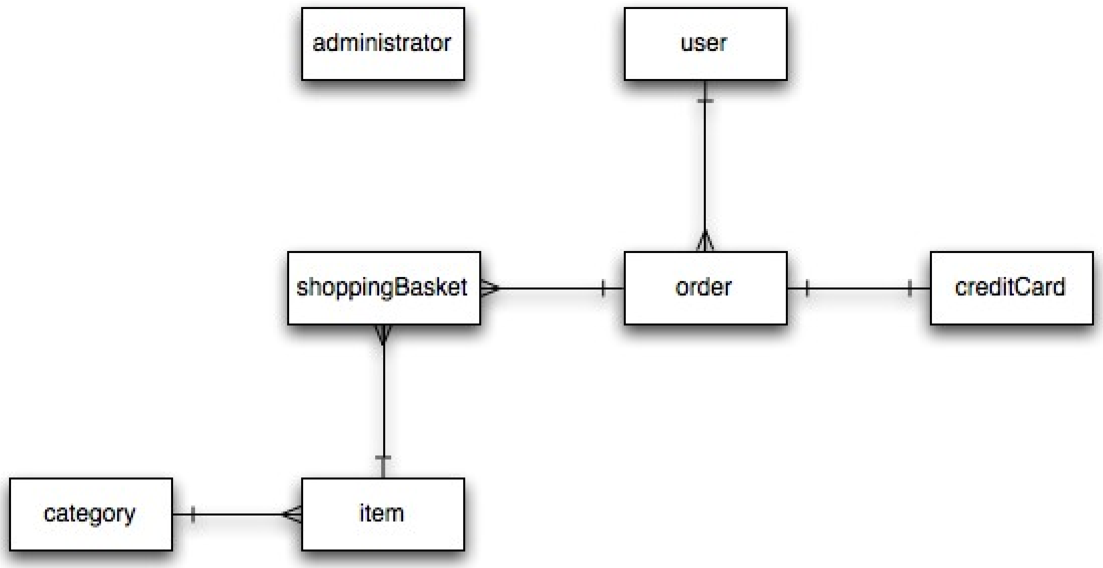
- *user*: Contains the users that can buy items in the store. The main fields have an user identification, the name of the user, his e-mail and his address.
- *administrator*: Contains the administrators of the system.
- *order*: Contains the orders processed in the store.
- *shoppingBasket*: Contains a list of items of every order and resolves a potential “many-to-many” relationship problem between *order* and *item*.
- *item*: Contains the items available to buy in the store.
- *category*: Allows organized the items in different categories.
- *creditCard*: Contains the related information of every credit card used in the orders.

9.3 ERD (Entity Relationship Diagram)

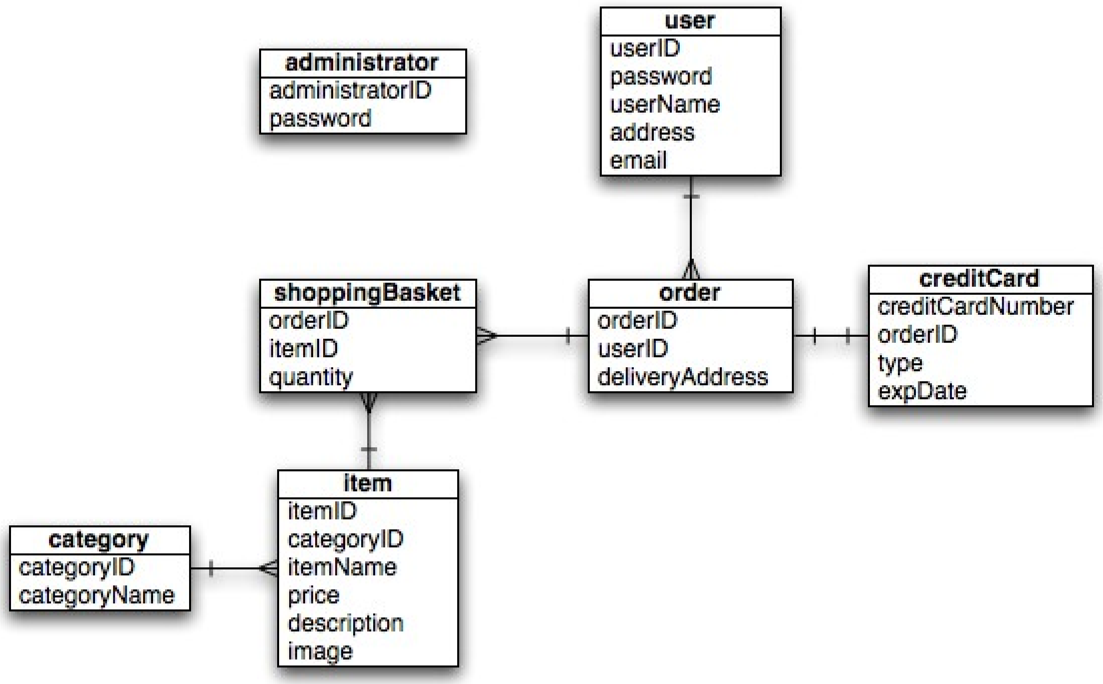
This approximation of the database implements the entities listed in section 9.2, and provides the main functionalities of the *Preliminary Software Requirements Analysis* showed in the *Feasibility Study*. With this design the system should be functional.

The following Entity Relationship Diagrams (ERD), shows the logical and the physical models of the database design to store the data for the *Music Online Store* project, developing its schema.

9.3.1 Logical Model



9.3.2 Physical Model (Schema)



10. Algorithm Design

After examine all class methods, we have check that there are not complicated algorithms, all of them are loops to list elements, establish connection to the database, etc. What is really complicated in the code of this application is write the queries to the database and integrate them into the java code.

Some of the queries of this application and the way to display their results will be:

- Change price of a item

```
String itemID = request.getParameter(“itemID”);
String price = request.getParameter(“price”);
String sqlQuery = “UPDATE Item SET price =“;
String quote = “’”;
String whereItemID= ‘ where itemID = ‘;
String andPrice = ‘ and price = ‘;
String sqlQuery = sqlQuery + quote + price + quote +
whereItemID +
quote + itemID + quote;
```

- Delete a item

```
String itemID = request.getParameter(“itemID”);
String sqlQuery = “DELETE FROM Item WHERE itemID = “;
String quote = “’”;
String sqlQuery = sqlQuery + quote + itemID + quote;
```

- Displaying query results in a table

```
ResultSetMetaData rm = rs.getMetaData();
String result = null;
int columns = rm.getColumnCount();
result = "<tr>";
for ( int i = 1; i <= columns; i++){
result += "<td>" + rm.getColumnLabel(i) + "</td>";
}
result += "</tr>";
```

11. References

Fowler, M., 2004. UML Distilled: A Brief Guide To The Standard Object Modeling Language. 3rd ed. Boston, Massachusetts: Addison-Wesley. Pearson Education, Inc. ISBN 0-321-19368-7.

McLoughlin, S., 2009. Software Engineering and Testing Notes: Session 2 – 4. Blanchardstown, Dublin: Institute of Technology Blanchardstown.

McGrath, M., 2005. SQL in easy steps. 1st ed. Southam, Warwickshire: Computer Step. ISBN I-84078-296-X.

McGrath, M., 2002. Java Server Pages in easy steps. 1st ed. Southam, Warwickshire: Computer Step. ISBN 1-84078-197-1.