Design of Shopping Mall Management System

Course Project Report: CS686 – Object Oriented Systems

Kamlesh Laddhad (05329014)

Akshay Ukey (05305045)

Under Guidance of
Prof. R. K. Joshi.

Project Description:

In this project, we will be designing a simple shopping mall using object oriented technology. The mall will provide a soothing shopping experience for customers, while at the same time allowing us to explore design patterns and other features object oriented technology.

.

The requirements:

The system will allow more than one shop owner to set up different shops, to sell various products under one roof i.e. mall. The concept, at its very basic, provides for an environment that allows the following:

- Shop Owner:
  - Any person wishing to setup shop in the mall can send a proposal to the mall owner.
  - The mall owner approves the proposal and confirms the deal.
  - Shop owners can then setup and maintain their own shop(s) in the Mall.
- Customers
  - Customers when enter the mall have to authenticate themselves on a central server.
  - After authentication, the customer is allocated a shopping cart and can enter a particular shop of his/her choice for shopping.
  - After entering a shop, customer can brows through the products available in the shop, can select some of them and put into the shopping cart.
  - Customer can anytime change the items in the cart either by adding new items or by removing gexsting items. Customer proceeds towards the payment counter. Finalize product list of items he finally wish to buy and make the final payment.
  - He/She then leaves the shop and can either enter another shop or leave the mall.
- The prototype presented allows customers to purchase products from all the shops in the Mall.

Report Organization:

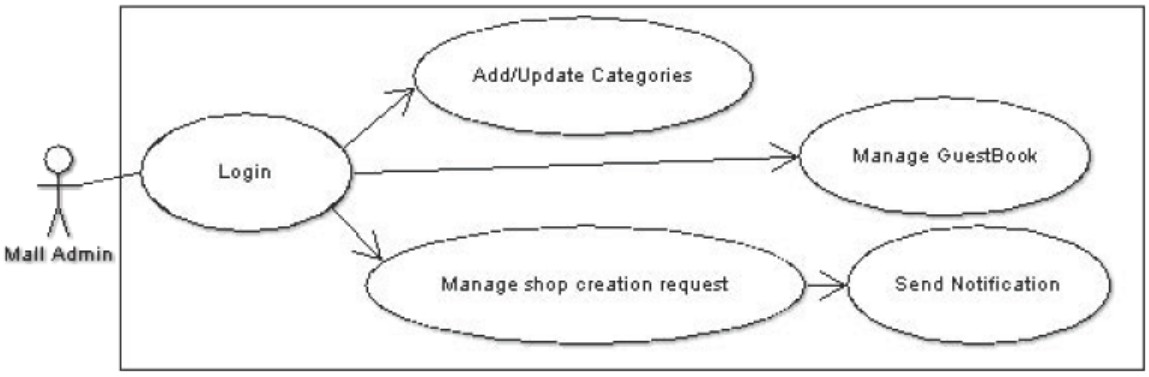
We organize our report in the following order.

- Use Case diagram for capturing all the possible scenarios.
  - Mall Owner
  - Shop Owner
  - Customer
- State Machines for different Objects
  - Shopping Proposal
  - Shop Setup/Contents
  - Customer
  - Item
  - Shopping Cart
  - Guestbook
- Object Identification
  - Mall
  - MallOwner
  - Shop
  - ShopOwner
  - Item
  - ShoppingCart
  - Customer
  - Guestbook
- Class Diagram showing relationships between different classes.
- CRC for all classes.
- Sequence Diagram capturing different scenarios.
  - Customer entering the mall and entering a selected shop.
  - Customer buying a product at a shop.
  - ShopOwner proposing for the putting a shop in the mall.
  - Guestbook States.
- Various design patterns incorporated in the design.
  - Singleton
  - Iterator
  - Observer
  - States
- Conclusion

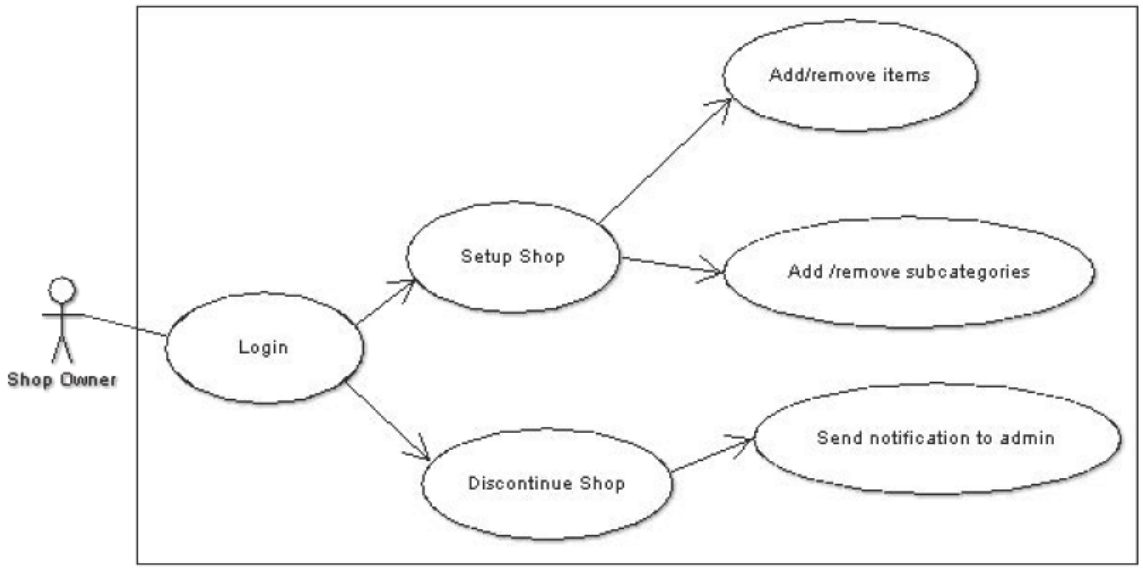
Use Case Diagrams:

Following are the use case diagram required for the project:

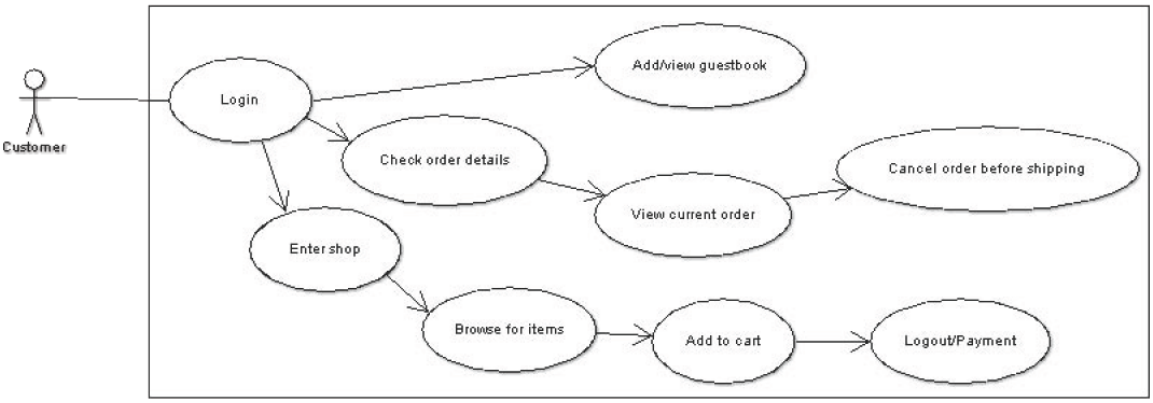
Mall Admin:



Shop Owner:



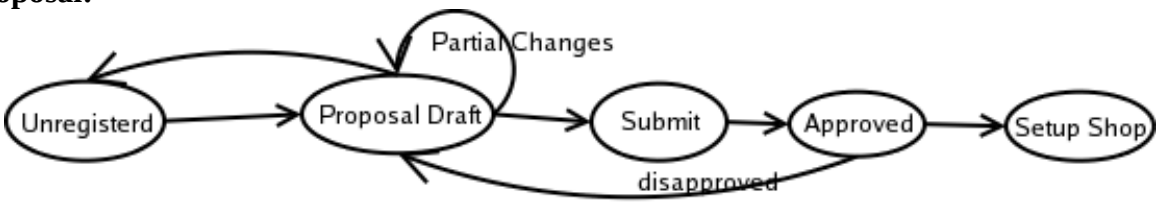
Customer:



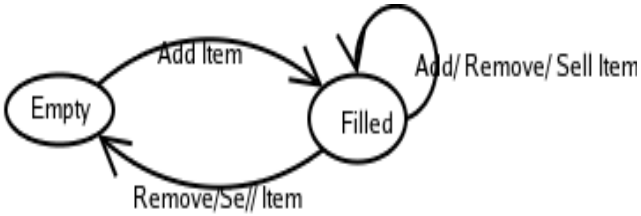
State Machines:

Adhering to the requirements, followings are the state machines for this project.

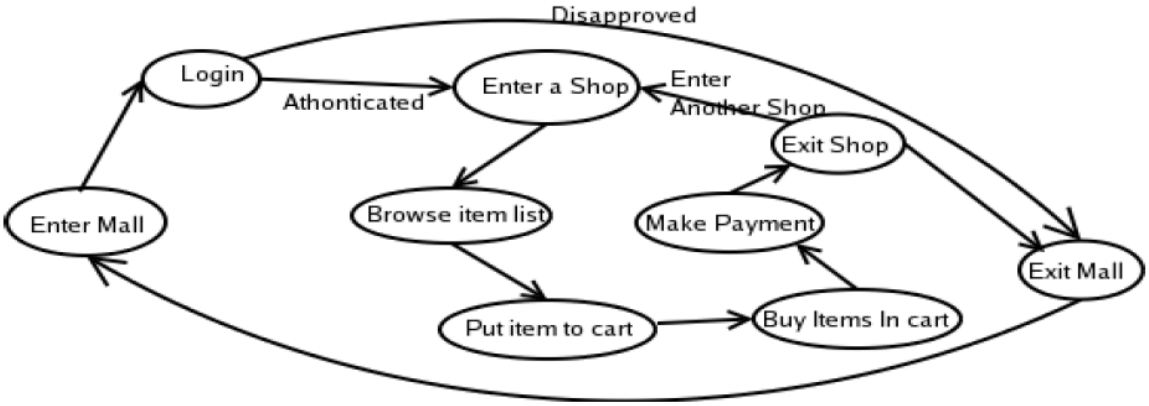
Shop Proposal:



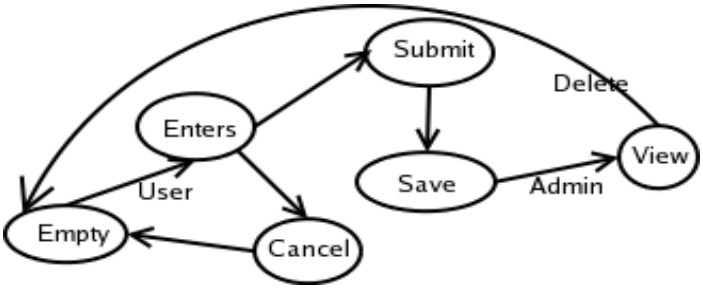
Shop Contents/Setup:



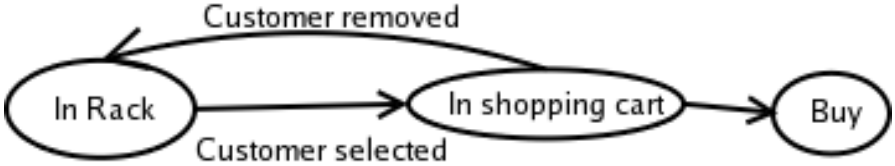
Customer:



Guestbook



Item:



Object Identification:

Here's a brief description of the objects we identified with respect to our project.

Mall

Mall will provide a single roof for various shops. The mall performs the creation of a set of different shops, such as a book store, a shoe store, etc. The mall greets an arriving customer, performs authentication for him/her and allocates him/her the shopping cart. Mall presents the customer with a list of different stores availble and allows the customer to shop at any of stores in the mall.

Some of the things a customer can do at the mall are:

- Get a list of available stores
- Get a shopping cart
- Enter a store /shop

Notable Attributes:

- name - the name of the mall
- shops - a collection of stores of different types
- customers - the customers currently in the mall
- owner – the owner of the mall.
- guestbook – a collection of comments by different customers

Possible Methods:

- void enter(Customer c) - customer c enters the mall
- void exit(Customer c) - customer c exits the mall
- ShoppingCart getShoppingCart - returns an empty shopping cart
- Enumeration customers() - returns an enumeration of the customers in the mall
- void checkout(shoppingCart cart) - checkout and purchase the items in the shopping cart

Shop: The Abstract superclass for a shop

Notable Attributes:

- name - the name of the store
- storeId - unique ID for the store
- items - items available for sale in the store
- customers - the customers currently in the store
- owner – the owner of the shop

Possible Methods:

- abstract void enter(Customer c) - customer c enters the store
- abstract void exit(Customer c) - customer c exits the store
- Enumeration customers() - returns an enumeration of the customers in the store
- Enumeration items() - returns an enumeration of the items available for sale in the store
- abstract void addToCart(shopingCart, item) - add an item to the shopping cart
- abstract void removeFromCart(shopingCart, item) - remove an item from the shopping cart

BookStore: A possible subclass of Store
ShoeStore: A possible subclass of Store
GameStore: A possible subclass of Store

Item: An item for sale in a store

Notable Attributes:

- itemName - the name of the item
- itemId - unique ID for the item
- storeId - the ID of the store from which the item came
- price - the price of the item

Customer: A customer visiting the mall. This class extends the general person class.
Notable Attributes:
shoppingCart - the shopping cart being used by the customer
store - the store the customer is currently in.

ShoppingCart : A shopping cart for the customer.
Notable Attributes:
items - items currently in the shopping cart
Possible Methods:
Enumeration items() - returns an enumeration of the items currently in the cart
addItem()- adds a given item to collection
removeItem() – removes the given item from the collection.
calcSubTotal() – calculates the incremental total of all the items in the cart.

GuestBook:
Customers visiting the mall can send there comments about the shopping experience at the mall to mall admin via guestbook. Each cutomer will have comment as an attribute field and the guestbook is the collection of such comments and is an attribute of the Mall class.

Notable Attributes:
comments
customer
date
Possible functions:
Add comment
View comments
Delete comments

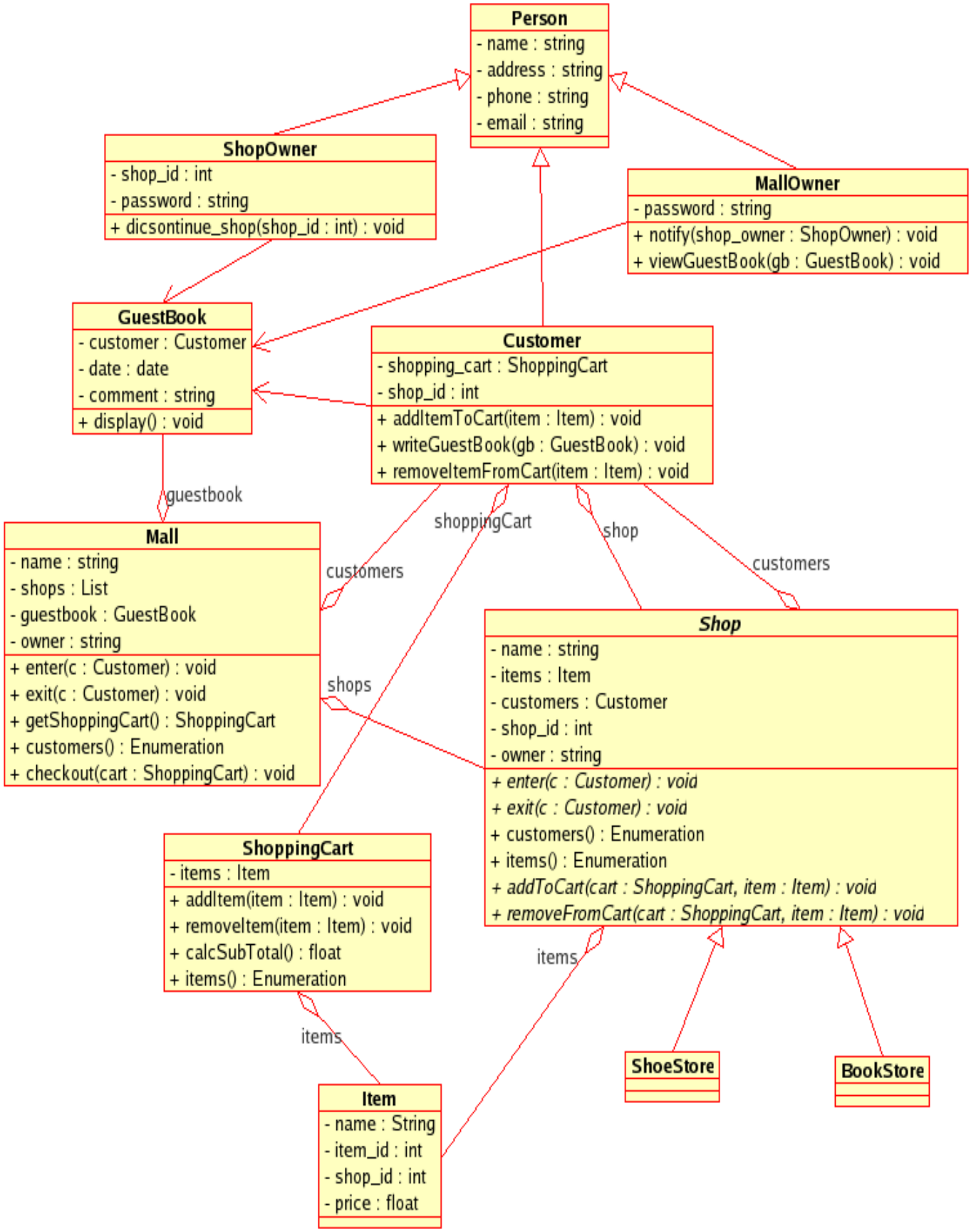
MallOwner: The owner of the mall or mall admin as referred in the class diagram above. This class also extends the person class.
Notable Atttributes:
password

ShopOwner: The owner of a shop. Each shop has a owner. This class also extends the person class.
Notable Atttributes:
password
shop_id

Person:
name
email
phone
address

For the sake of brevity, the required accessors and mutators are not listed in the above class descriotion.

Class Diagram: Following is the class diagram for our project.



Class Responsibility Collaborations (CRC):

Class: Person
Attributes: name, address, phone, email.
Responsibilities:
Collaborations:

Class: ShopOwner
Super Class : Person
Attributes: shop_id, password.
Responsibilities: Maintain Shop: Add/remove/sell items. Discontinue shop.
Collaborations: Shop, Item: Add/Remove/Sell items. MallOwner: Discontinue shop.

Class: MallOwner
Super Class: Person
Attributes: password.
Responsibilities: Maintain Mall: Approve/Disapprove shop proposals, send notification to shop owners. Maintain Guestbook: View/Delete guestbook entries.
Collaborations: Mall: Add/Delete Shops. ShopOwner: Send Notification about approve/disapprove proposals Guestbook: View/Delete guestbook entries

Class: Customer
Super Class: Person
Attributes:
Responsibilities: Take comments about products and shopping experience. Get list of shops in the mall. Get list of items in the shop. Add, delete items to/from shopping cart
Collaborations: ShoppingCart, Item: Add, Delete items from/to shopping ca Mall: Ask list of shops, take cart from mall. Shop: Ask list of items. Guestbook:Comments about shopping experience.

Class: Mall
Attributes: name, shops, customers, guestbook, owner.
Responsibilities: Set Owner after authentication. Create Shops. Display list of existing shops. Allocate shopping cart to each customer entering the shop. Take User Comment and add to guestbook. Welcome new Customer. Maintain Customer record in mall.
Collaborations: Shop: Maintain record of shops, Create new shop. Owner: Authentication. Guestbook: Add new Comments by Customers. Customer: Add new Customer when he/she enters. ShoppingCart: Allocate to new user.

Class: Shop
Attributes: name, items, customers, owner, shop_id.
Responsibilities: Set Owner after authentication. Maintain Customer record in shop. Maintain Item record in shop. Add new items to shop. Reject defected items. Do billing of customer selected items. Display items in the shop.
Collaborations: Owner: Authentication. Customer: Add new Customer when he/she enters. Item, ShoppingCart: Maintain items, bill items selected from this shop.

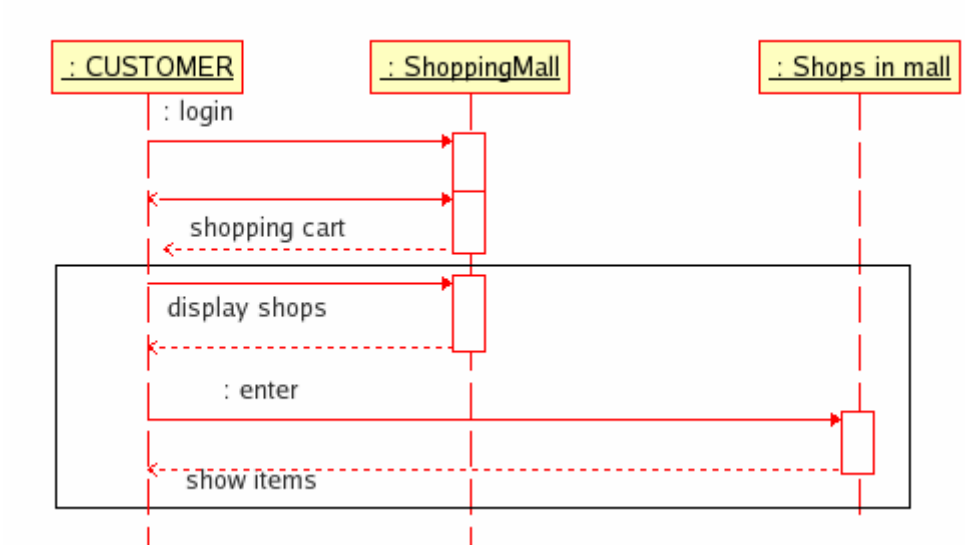
Class: Item
Attributes: name, item_id, shop_id, price.
Responsibilities: Display price.
Collaborations:

Class: ShoppingCart
Attributes: items
Responsibilities: Add item. Delete item.
Collaborations: Item: Add, delete item.

Class: Guestbook
Attributes: customer, date, comment.
Responsibilities: Enlist Comments. Add /Delete Comments.
Collaborations:

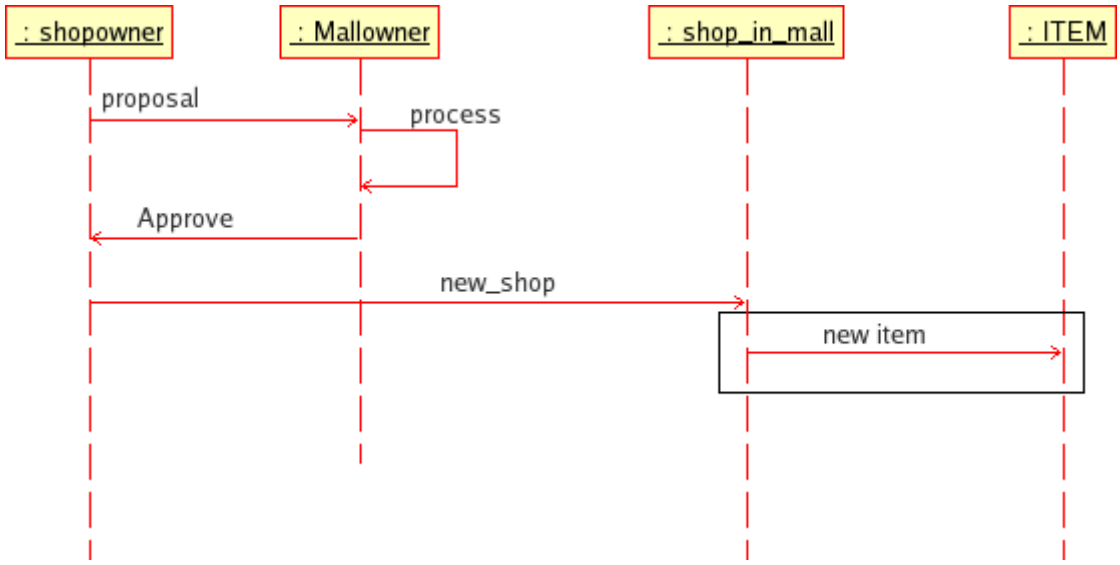
Sequence Diagram:

Customer entering the mall gets authenticated and will be allocated a shopping cart. On successful authentication, he/she will be presented with the list of stores available. On entering the store he/she will be presented with the list of items available at the shop. This whole sequence has been captured by following sequence diagram.

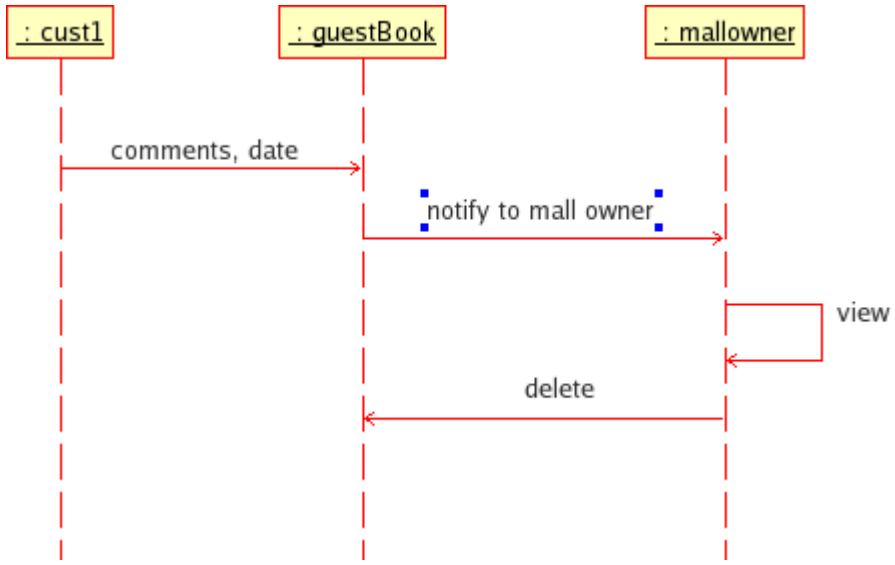


Customer buying a product at a shop in the mall is being captured by following sequence diagram.

ShopOwner, who is proposing for the putting a shop in the mall sends a proposal to MallOwner, MallOwner in turn approves or disapproves the proposal. This sequence has been captured in following sequence diagram.



Guestbook: Every customer visiting the mall can send his/her comments to MallOwner. This sequence has been captured in following sequence diagram.



Design Patterns: We propose to capture following design patterns in our design.

1. **Singleton:** Mall object is a singleton class. Only one object of this type exists during the execution of program. Also, all the stores contained in the mall are singleton classes. Whenever the mall is getting initialize, all the stores are also getting initialized
This is being captured by following code.

Mall.java

```
package data;

public class Mall {
    private static Mall uniqueInstance = null;
    public static Mall instance() {
        if(uniqueInstance == null)
            uniqueInstance = new Mall();
        return uniqueInstance;
    }
    private Mall() {
        bookStore=BookStore.instance();
        shoeStore=ShoeStore.instance();
    }
    public Store bookStore;
    public Store shoeStore;
}
```

2. **Iterator:** Whenever the mall is supposed to present list of stores it contains, we are presenting the user with the Iterator class provided by java which captures the Iterator design patterns.

Customer.java

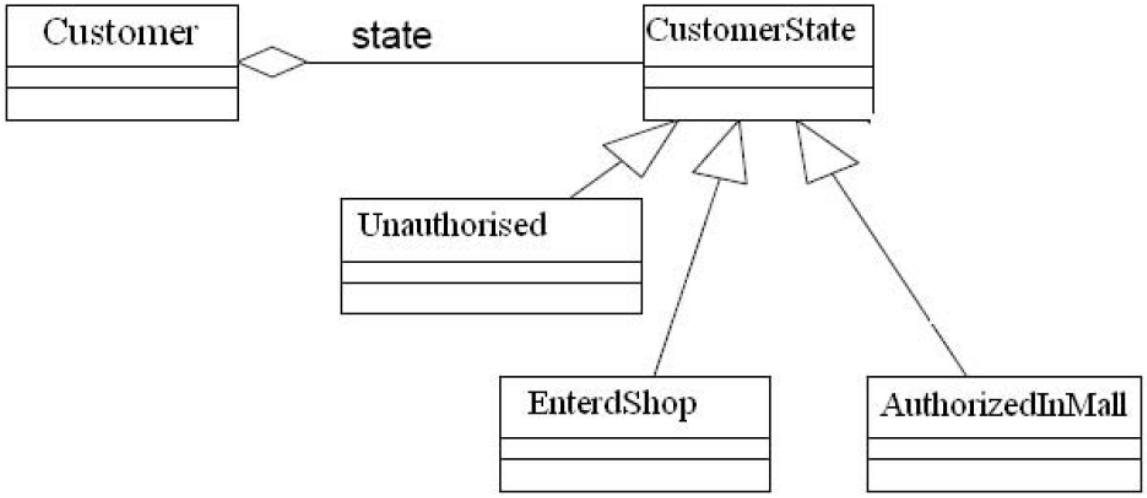
```
package data;

import java.util.Iterator;
import data.Mall;
import data.Store;

public class Customer{
    private Mall mall;
    public void printList(){
        mall=Mall.instance();
        Iterator iter=mall.getStoreList();
        while(iter.hasNext()){
            Store store=(Store)iter.next();
            System.out.println(store.getName());
        }
    }

    public Customer(Customer customer){
        mall=Mall.instance();
    }
}
```

Also, whenever the shop is supposed to present the list of items it has we are using the Iterator class in java to do this.

3. **Observer:** All stores are observable objects. In particular, whenever a new item gets added to any of the shops in the mall, customer is being notified about the change. All customers by default are assumed to have requested for this facility.

4. **State:** The customer object can have different states while visiting the mall and shopping in different stores of the mall. Following diagram captures three different states the customer object can go into.

    a. Unauthorized: The customer has just entered the mall and is yet to login.
    b. EnteredShop: The customer has selected one of the shops for shopping and has entered into it. He may brows and select items in the store. Put the selected items in the cart which is allocated to him when he entered the shop.
    c. AuthorizedInMall: The customer either just logged in or has come out of one of the shops and roaming around in the mall. Either he will leave the mall or he/she will enter another shop in the mall.



Conclusion and future work:

The project enabled us to understand all the design patterns thoroughly. The Iterator, Singleton, Observer are essential design patterns in order to capture the software design of such shopping malls. Various techniques like use case analysis, state machine, CRC, sequence diagram are helpful in prototyping software design.

The project can be improved by incorporating the MVC design technique. More of design patterns such as factory patterns can be included in the project.

Reference:

Design Patterns: Elements of Reusable object oriented software, Erich Gamma, Richard Helm, et al.