Software Requirements Specification

for

Furniture Shopping Cart System

Version 1.0

Prepared By :

Divyansh Verma

**CONTENTS : Page No.**

1. SCOPE 1
   1. AUDIENCE
   2. ORGANISATION
   3. APPLICABLE DOCUMENTS
2. SOFTWARE REQUIREMENT SPECIFICATION 2
3. FUNCTIONAL MODEL 3
4. ACTIVITY DIAGRAM 8
5. USE CASE DIAGRAM 11
6. INTERACTION DIAGRAM
   * 1. SEQUENCE DIAGRAM 19
     2. COLLABORATION DIAGRAM 23
7. CLASS DIAGRAM 28
8. IMPLEMENTATION DIAGRAM
   1. COMPONENT DIAGRAM 29
   2. DEPLOYMENT DIAGRAM 31
9. **Scope :**

The following subsections describe the scope of the Furniture Shopping system in terms of its audience, organization, and applicable documents

* 1. **Audience :**

Customers, Administrators, Sellers.

* 1. **Organization :**

FURNITURE SHOPPING CART SYSTEM (Furniture Avenue)

* 1. **Applicable Documents :**

The following documents provide information necessary to understand this documentation:

1. **Software Requirement Specification:**

* **System Abstract –**

The Furniture Shopping Cart system facilitates the Customer to view the Items Products, inquire about the item details, availability of seats and many more. The major functionality of system is to allow the user to book and cancels the flights as per user requirements. It also provides the administrator or manager to modify existing flights or to introduce a new flight’s in the schedule.

* Major features provided by the system are :

1. Customer

The system allows the user or member to perform query including furniture price, material and other specific details, etc.

1. Administrator

It allows the customer to register in order to give the services to its member .

1. Furniture shopping system

The system allows the member to order the product as per his/her requirements. The customer is prompt to enter the customer details and credit card details. The member then receives the unique cart number .

1. Product Listing

This process gives the details about the purchasing process made by the customer at the details of the shipping and taxes involved with the cost

1. Administrator

The administrator module of the system allows the admin/manager to manage the products item It provides the admin /manger to modify or change the existing items or to introduce a new items. Apart from modifying the category it also allow the admin/manager to take the taxes and include the shipping charges along with the h/w s/w and service requirements.

1. **Functional Model.**



**Fig 3.1 Data Flow Diagram for Furniture Shopping Cart Software**

 **Fig 3.2 Data Flow Diagram for Product Listing**

 **Fig 2.3 Fig : 3.3Data Flow Diagram for Administrator.**

 **Fig Fig: Data Flow Diagram for Secure Gateway Provider.**

1. **Activity Diagram.**

* An Activity diagram is a visual representation of any system's activities and flows of data or decisions between activities.
* Activity diagrams provide a very broad view of a business process.
* They represent the dynamics of a system.
* They are flow charts that are used to show the work flow of a system.
* They show the flow of control from activity to activity in the system.
* They show what activities can be done in parallel, and any alternative paths through the

flow.

* Purpose
  + Model business workflows
  + Model operations
* Activity diagrams commonly contain
  + Activity states and action states
    - Transitions
      * Objects

**Action States and Activity States**

* Action states are atomic and cannot be decomposed
* Work of the action state is not interrupted.
* Activity states can be further decomposed
* Their activity being represented by other activity diagrams
* They may be interrupted
* Represented in UML by a rounded rectangle.
* *Activity* represents the performance of some behavior in the work flow.

**Transitions**

* Transitions are used to show the passing of the flow of control from activity to activity.

They are typically triggered by the completion of the behavior in the originating activity**.**

* When the action or activity of a state completes, flow of control passes immediately to the next action or activity state
* A flow of control has to start and end at some place

initial state -- a solid ball stop state -- a solid ball inside a circle

 

**Branching**

* When modeling the workflow of a system, it is often necessary to show where the

flow of control branches based on a decision point.

* The transition from a decision point contain a guard condition.
* The guard condition is used to determine which path from the decision point is taken.
* Decisions along with their guard conditions allow you to show alternative paths

through a work flow**.**

* A branch specifies alternate paths taken based on some Boolean expression
* A branch may have one incoming transition and two or more outgoing ones

**Synchronization Bars**

* In a workflow there are typically some activities that may be done in parallel.
* A synchronization bar allows you to specify what activities may be done

concurrently.

* A synchronization bar may have
  + many incoming transition and one outgoing transition, or
  + one incoming transition and many outgoing transitions.
* It is use to specify the forking and joining of parallel flows of control
* A synchronization bar is rendered as a thick horizontal or vertical line
* A fork may have one incoming transitions and two or more outgoing transitions
  + each transition represents an independent flow of control
  + conceptually, the activities of each of outgoing transitions are concurrent
    - either truly concurrent (multiple nodes)
    - or sequential yet interleaved (one node)
* A join may have two or more incoming transitions and one outgoing transition
  + above the join, the activities associated with each of these paths continues in

parallel

* + at the join, the concurrent flows synchronize
  + each waits until all incoming flows have reached the join, at which point one

flow of control continues on below the join



**Fig 4.1 Activity Diagram :**

**5. Use-Case Diagram.**

* The use-case concept was introduced by Ivar Jacobson in the object-oriented software engineering (OOSE) method.
* A use-case diagram is a graph of actors, a set of use cases enclosed by a system boundary, communication (participation) associations between the actors and the use cases, and generalization among the cases.
* Use case diagrams show how users interact with the system.
* Use case diagrams describe what a system does from the standpoint of an external observer. The emphasis is on *what* a system does rather than *how.*
* Use case diagrams are closely connected to scenarios. A **scenario** is an example of what happens when someone interacts with the system. Here is a scenario for a medical clinic.

"A patient calls the clinic to make an appointment for a yearly checkup. The receptionist finds the nearest empty time slot in the appointment book and schedules the appointment for that time slot. "

* Used during requirements elicitation to represent external behavior
* ***Actors*** represent roles, that is, a type of user of the system
* ***Use cases*** represent a sequence of interaction for a type of functionality
* The use case model is the set of all use cases. It is a complete description of the functionality of the system and its environment

**Actors**

* An actor models an external entity which communicates with the system:
  + User
  + External system
  + Physical environment
* An actor has a unique name and an optional description.
* Examples:

**Customer**

* Passenger: A person in the train
* GPS satellite: Provides the system with GPS coordinates

**Use Cases**

A use case represents a class of functionality provided by the system as an event flow.

A use case consists of:

* Unique name
* Participating actors
* Pre-conditions
* Entry conditions (triggers)

**Item Purchased**

* Flow of events
* Exit conditions (success ‘guarantees’)
* Special requirements

The Use Case diagram for whole system is as shown in fig 5.3

**The *<<extend>>* Relationship**

* <<extend>> relationships represent exceptional or seldom invoked cases.
* The exceptional event flows are factored out of the main event flow for clarity.
* Use cases representing exceptional flows can extend more than one use case.
* The direction of a <<extend>> relationship is to the extended use case

**The <<include>> Relationship**

* An <<include>> relationship represents behavior that is factored out of the use case.
* An <<include>> represents behavior that is factored out for reuse, not because it is an exception.
* The direction of a <<include>> relationship is to the using use case (unlike <<extend>> relationships).

Select shipping address

Add shipping address

Add Credit card type

Add Credit card number

Add credit card

Add Netbanking details

Choose shipping type

Furniture customer

Select credit card

<<include>>

<<include>>

<<include>>

<<include>>

<<include>>

<<include>>

**Fig :5.1 Use Case Diagram**



**Fig 5.2 Use case Diagram**

**USE CASE DESCRIPTION :**

**Actors:**

* 1. New Customers
  2. Registered Customer
  3. Administrator

**Use cases:**

1. Search Item
2. Register Item
3. Buy Item
4. Update Item
5. **Search Item**

Any user can search the item. User can search the item by name or company.

1. **Register Customer**

To buy the item, the customer has to be a registered customer. This process is used to register a customer**.**

1. **Buy Item**

A registered customer can buy a item using this process. This process include search item, validate customer, prepare bill, pay bill.

**Use Case Table**

|  |  |
| --- | --- |
| Use case ID | 1 |
| Use case name | View shopping cart |
| Actor | Furniture customer |
| Pre-condition | Open the website |
| Post-condition | Search for product |
| Flow of events | open the website of Furniture shopping cart |

|  |  |
| --- | --- |
| Use case ID | 2 |
| Use case name | Search for product |
| Actor | Furniture customer |
| Pre-condition | View shopping cart |
| Post-condition | Select the product |
| Flow of events | Open the website, search the product from list |

|  |  |
| --- | --- |
| Use case ID | 3 |
| Use case name | Create new user |
| Actor | Furniture customer |
| Pre-condition | Select the product |
| Post-condition | Select credit card |
| Flow of events | Open the website, search the product from list, select the product, create new user |

|  |  |
| --- | --- |
| Use case ID | 4 |
| Use case name | Purchase item |
| Actor | Furniture customer |
| Pre-condition | Select credit card |
| Post-condition | Review order information |
| Flow of events | Open the website, search the product from list, select the product, create new user, select credit card, purchase the selected item |

|  |  |
| --- | --- |
| Use case ID | 5 |
| Use case name | Login as registered user |
| Actor | Furniture seller and Furniture customer |
| Pre-condition | Open the website |
| Post-condition | Sell or purchase the product |
| Flow of events | Open the website, Login as registered user |

|  |  |
| --- | --- |
| Use case ID | 6 |
| Use case name | Item registration |
| Actor | Furniture seller |
| Pre-condition | Login into the shopping cart system |
| Post-condition | Confirm the registration |
| Flow of events | Open the website, Login as registered user, register the item into system |

|  |  |
| --- | --- |
| Use case ID | 7 |
| Use case name | Change the information |
| Actor | administrator |
| Pre-condition | Login into the system as administrator |
| Post-condition | Update the system |
| Flow of events | Open the website, Login as administrator, make changes in the information |

|  |  |
| --- | --- |
| Use case ID | 8 |
| Use case name | Block users |
| Actor | administrator |
| Pre-condition | Select the user |
| Post-condition | Update the system |
| Flow of events | Open the website, Login as administrator, select the user, block the user |

|  |  |
| --- | --- |
| Use case ID | 9 |
| Use case name | Inform the deal to seller |
| Actor | Furniture Seller and shopping cart system |
| Pre-condition | Check the availability of item |
| Post-condition | Make a deal |
| Flow of events | Open the website, login as shopping cart system , seller sell the item to shopping cart system, inform the deal to seller |

Register Item

Search for product

Choose shipping type

Select credit card

Purchase item

Review order confirmation

View shopping cart

Add item to shopping cart

Login as registered user

Furniture seller

Shopping Cart system

Inform the deal to seller

Block users

Administrator

Change item information

Furniture Customer

Create new user

**fig 5.3. Use case diagram for Furniture Shopping System**

1. **UML Interaction Diagram** 
   1. **Sequence Diagram :**



**Fig.6.1.1 Sequence Diagram for User registeration**



**Fig.6.1.2 Sequence Diagram for Seller registeration**



**Fig.6.1.3 Sequence Diagram for Furniture Customer**



**Fig.6.1.4 Sequence Diagram for Furniture Administrator**

* 1. **Collaboration diagrams:**

They are also interaction diagrams. They convey the same information as sequence diagrams, but they focus on object roles instead of the times that messages are sent. In a sequence diagram, object roles are the vertices and messages are the connecting links.

* Collaboration diagrams show (used to model) how objects interact and their roles.
* They are very similar to sequence diagrams. Actually they are considered as a **cross** between class and sequence diagram.
* *Sequence Diagrams* are arranged according to Time.
* *Collaboration Diagrams* represent the structural organization of object*.*
* [Both sequence and collaboration diagrams are called interaction diagrams]
* Forms a context for interactions
  + May *realize* use cases
  + May be associated with operations
  + May describe the static structure of classes
* Collaboration diagrams contain the following:
  + Class roles (subsystems/objects/classes/actors/ external systems) as before.
  + Association roles (pathways or links over which messages flow)
  + Message flows (messages sent between class roles)

 **Fig.6.2.1 Collaboration Diagram for User registration**



**Fig.6.2.2Collaboration Diagram for Seller registeration**

 **Fig.6.2.3 Collaboration Diagram for Furniture Customer**



**Fig.6.2.4 Collaboration Diagram for Furniture Administrator**

**7. CLASS DIAGRAM :**

TRANSACTION



transid



pnrno



transdate



amt



commit()



rollback()

Payment System



custno



custid



custname



creditcardno



BillingAddress



VeiwOrder()



PaymentDetails()

FURNITURE SHOPPING CART

SYSTEM

HTTP

CartAddition



Creidtno : String



Description : String



ItemId : String



Amount : Integer



Category : String



Get\_to\_cart()



Check\_out()

CANCELLATION



cusno : Integer



Item id : Integer



amount : Integer



retrive()



update()

CREDIT CARD HOLDER



cardno



issuedate



expirydate



verifydtls()



perform\_transaction()

CUSTOMER



Number of Item : Integer



cusid : Integer



Shipnname : String



Company : String



Address : String



Payment()



CartAddition()



Cancellation()

USER



Email id : String



password : String



First name : String



Last Name : String



Street Address : String



ZipCode : Integer



City : String



State : String



Country : String



Phone : String



Login()



Registration()



Logout()

SELLER



Item name : String



Item id : Integer



Discount : Integer



Categoty : Integer

Fig 7.1 **Class diagram for Furniture Shopping System**

**8.Implementation Diagram.**

**8.1Component Diagram :**

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model physical aspects of a system.

Physical aspects are the elements like executables, libraries, files, documents etc that resides in a node. So component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

Description of the component diagram for Furniture shopping system is as follows :

Data base server contains all the database tables. It contains Administrator, Registered Customer, Seller, Item and Payment.

Application server contains Access classes package and Business classes package and view layer classes, i.e. view classes package.

Clients are the nodes having no processing capabilities. Only browser is there on this node to send a request



**8.2 Deployment Diagram :**

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

So deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

.



Fig 8.2 Deployment diagram for Furniture Shopping System.