POS SYSTEM – ARCHITECT DESIGN DOCUMENT



HIT Team

Consulting

Sales

Staffing

Support

Information of document

|  |  |
| --- | --- |
| Title | Architecture Driver Document |
| Author(s) | All team |
| Reviewer(s) | All team |
| Team name | HIT Team |
| Team members | Thanh Giang, Hiep Ta, Phuc Nguyen, Giang Nguyen, Dat Tran, Huy Huynh |
| Project mentors | Ms. Huong Nguyen, Mr. Huyen Pham |
| Editor | Hiep Ta |
| Type of report | Architecture Driver Document |
| Software used | MS Word |

Document Reviewer Information

|  |  |  |
| --- | --- | --- |
| **Reviewer Name** | **Review Attendance (R/S)** | **Comments** |
|  |  |  |
|  |  |  |
|  |  |  |

Document Approver Information

|  |  |  |
| --- | --- | --- |
| **Approver Name** | **Approver Function** | **Comments** |
|  |  |  |
|  |  |  |
|  |  |  |

Document Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Revision | Status | Change Summary | Revised by |
| 06/03/2012 | 1.0 |  | Consume Team member’s tasks | Thanh Giang |
| 15/04/2012 | 1.1 |  | Update Quality Attributes | Giang Nguyen |
| 18/04/2012 | 1.2 |  | Use case | Thanh Giang |
| 24/05/2012 |  | start | Use case Description | Hiep Ta |
| 09/06/2012 |  |  | Module View | Huy Huynh |
| 21/06/2012 | 3.3 |  | Static Perspective & Dynamic Perspective | Huy Huynh |

Table of Contents

[1. Document description 4](#_Toc329248612)

[1.1. Purpose and audience: 4](#_Toc329248613)

[1.2. Document organization: 4](#_Toc329248614)

[1.3. Common notation: 4](#_Toc329248615)

[1.4. Terminology and definitions: 4](#_Toc329248616)

[1.5. References and relevant document: 5](#_Toc329248617)

[2. Project overview 6](#_Toc329248618)

[2.1. Project Context 6](#_Toc329248619)

[2.2. System Context 6](#_Toc329248620)

[3. Architectural drivers 8](#_Toc329248621)

[4. System context 9](#_Toc329248622)

[5. Level X decomposition 10](#_Toc329248623)

[6. Mapping between perspectives 11](#_Toc329248624)

[7. Document directory 12](#_Toc329248625)

# Document description

## Purpose

Intending to capture and convey the architectural decisions that have been made in order to implement POS system, the Software Architecture Design Document (SADD) formally provides a comprehensive overview of the proposed system. It uses a number of architectural decompositions to depict the different aspects, corresponding with the architecture driver specified. This document serves as a basis for the detailed design, which will establish the design in increased detail.

## Audience

The audiences for this document include:

Supervisor

|  |  |
| --- | --- |
| Name | E-mail |
| Mrs. Nguyen Thuy Huong | ngthuong02@yahoo.com |

HIT Team

|  |  |
| --- | --- |
| Name | E-mail |
| Giang Thi Ha Thanh | giangthihathanh@gmail.com |
| Ta Quang Hiep | hiep.taquang@gmail.com |
| Huynh Chan Huy | huynhchanhuy@gmail.com |
| Nguyen Tran Hong Phuc | phucnguyen10121991@gmail.com |
| Tran Dung Dat | dattran2525@gmail.com |
| Nguyen Vu Trong Giang | chan.tinh157@gmail.com |

Administrators

Administrators of POS system may use this design to understand the structure of the system.

Future Developers of the product

This document is written such that any future developers employed for enhancements or modifications to the POS system code may use it to understand the existing system.

## Document organization:

Describe the overall organization of the document. List the major sections of the document and describe what concerns each section addresses.

## Terminology and definitions:

|  |  |
| --- | --- |
| ACDM | Architecture Centric Design Method |
| Mapping | A mapping is created either between different views or between elements or relationships and groups of elements and relationships. A mapping can be hierarchical, or on the same level. Mappings between elements or relationships and groups of elements and relationships can span across views. |
| Entity | The ACDM definition for an actor which could be a person or a system. |
| Relationship Type | A relationship type is a connector in the palette, which is used to create a relationship in the canvas. |
| Relationship | A relationship is an instance of a relationship type, and used to connect element(s) on the canvas. |
| Perspective | 3 perspectives prescribed by ACDM; static, dynamic and physical. |
| View | A view is a drawing in a perspective. There can be multiple views in a perspective. |
| Design | This is the drawing the architect does in the tool. The term design is used in this document because architecture is used in many other ways which might confuse the reader. |
| Architecture | All of the diagrams including their elements and relationships as a representation of a software system. The architecture also includes the architectural drivers which are assigned to the elements and relationships. |
| Architectural drivers | These consist of functional requirements, quality attributes, business constraints and technical constraints. Can be assigned to elements, relationships and mappings. |
|  |  |

## References and relevant document:

### References

[Sandcastle] - Architectural Drivers Specification

[Lat08] Lattanze, A. *Architecting Software Intensive Systems: A Practitioners Handbook*, New York, NY: Auerbach, 2008

Software Architecture Design Document - Collaborative Problem Solver

### Relevant document

POS System – Architecture Driver Document

# Project overview

* This section describes the project and its purpose and scope.
* Describe why the system is being built.
* Describe business, organizational, mission, or marketing concerns that are relevant to the project.
* List the relevant stakeholders, their organizations, and how they will interact with the system.

## Project Context

It is Final project that requires HIT Team to develop POS System with the following constrains:

* Team size: 6 members
* The time for completing: 3 months
* Software Development Model assigned: Water fall
* Tools assigned: ASP .NET Framework, MS SQL 2008, Visual studio 2010.
* Hardware: Laptops

## System Context

The project team will develop a sale system on websites with the purpose bring the convenience and helpful in the sale for Company A, a retail chain (hereinafter, the system). POS system will contain a number of functions that help manage product as well as sale process.

* Administrator can do following function:
* Add New User
* Search/ View User List
* View User Detail Information
* Update User Information, Assign Authorize
* Add New POST
* Search/ View POST List
* View POST Detail Information
* Update POST Information
* Staff can do following function:
* Add New Product
* Search/ View Product List
* View Product Detail Information
* Update Product Information
* Add New Type
* Search/ View Type List
* View Type Detail Information
* Add New Member
* Search/ View Member List
* View Member Detail Information
* Update Member Information
* View Member Point Log
* Cashier can do following function:
* Add New Store
* Search/ View Store List
* View Store Detail Information
* Update Store Information
* Analysis Statistic
* Cashier can do following function:
* Add New Bill
* Search/ View Bill List
* View Bill Detail Information
* Print Bill
* Member can do following function:
* View point

# Architectural drivers

These architectural drivers will influence the architectural design and implementation of the project. Additionally, they will impact the schedule and quality of the project. As a whole these architectural drivers define the scope of the project.

The architectural drivers presented in this document include:

* **Functional Requirements:** These requirements are presented in the form of specifications and use cases. These are a refinement of the requirements documented in the raw requirements specification document of step 1 ACDM.
* **Quality Attribute Requirements:** These requirements are presented in the form of quality attribute scenarios. These scenarios are based on the quality attributes documented in the raw requirements specification document of step 1 ACDM.
* **Business Constraints:** These are the business constraints documented in the raw requirements specification document of step 1 ACDM.
* **Technical Constraints:** These are the technical constraints documented in the raw requirements specification document of step 1 ACDM.

# System context



## Users and roles:

Stakeholders who interact to Retail system were described in section 2.3.Stakeholder of this document.

## Channels:

Users will use different channels to access the system.

* Staff: Access website to system by PC locates at Head Office.
* Manager: Access website to system by every PC.
* Cashier: Use Bar code reader at computer using POS website to interacts with system
* Customer: Use online website to see their point
* Administrator: Use PC at Head Office to access the system

## Relationship Describe:

The context diagram shows the input of stakeholders and output from system, direction of the arrows show the direction of information.

: Show that the input from user to the system

: Show that the output from system to the user

# Physic Perspective

**ALLOCATION VIEW (Deployment Style)**

## Primary presentation



## Element catalog:

### Elements and their properties

.

|  |  |  |  |
| --- | --- | --- | --- |
| Associated Drawings:  Fig2 | | | Perspective:  Physic |
| No | **Name** | **Properties** | **Responsibilities** |
| 1 | Main Database | * It‘s a database run in database management system SQL server 2008 | * Contains all general data of system |
| 2 | Backup database | * It‘s a database run in database management system SQL server 2008 | * Contains all general data of system. Run parallel with the main database |
| 3 | WEB POS Application | * It’s a WEB application |  |
| 5 | Database Server | * OS: Windows Server 2008 * Processor: 1 x Intel® Xeon® Processor E5606 * Memory: 1 x 2GB DDR3 1333 240- * Hard Disk: DELL 250GB SATA 7.2K 3.0Gbs 3.5" Enterprise * Software: Microsoft SQL Server 2008 Enterprise, .NET Framework 4.0 | * Run Main Database * Run Backup Database |
| 6 | WEB server | * Software: .Net Framework 4, IIS 7 |  |
| 6 | User PC | * Operation System : Genuine Windows® 7 Home Basic, * Processor: AMD AM3 For Phenom™ II/Athlon™ II Family /Processors * Chipset: AMD SB710 * Graphics: ATI Radeon HD 3200, * Hard Drive SATA: 3.5" 320G * Software: Web browser | * Head office and Retail Store   + Run WEB POS Application |
| 7 | Fiber router | * Vigor2950 | * Connect LAN with WAN |
| 8 | LAN | * Topology: Star * Use switch to connect elements in LAN | * Connect computers in a store |
| 11 | Fiber cable |  | * Connect fiber router in WAN by fiber port |

## Architecture background

### Rationale design

Deployment design satisfies quality attributes following:

* + **Performance**: there is main data base is stored at head office, so request from Retails Store will be very fast because it access directly to the main database. We use load balancer to disperse the requests, balance the requirements for the two servers that ensure the performance and availability of the system
  + **Security**: The Firewall will prevent unauthorized or unwanted communications so that we can save database and web server. POS Database is placed in internal network, not with Webserver.
  + **Availability**: Our system has two databases to be able to process multiple requests: maintain multiple copies of either data or computations all data store in database server and two web servers: disperse the requests

# Static Perspective

**Module View (Layered Style and Uses Style)**

## Primary presentation:

### Layered Style

### Uses Style

## Element catalog:

### Elements and their properties

|  |  |  |
| --- | --- | --- |
| Elements | | Properties |
| Controller | **ProductControllers** | The ProductController class contains action methods that render view pages (AddProduct, EditProduct, ViewProduct) |
| **Category**  **Controllers** | The CategoryController class contains action methods that render view pages (AddCategory, EditCategory, ViewCategory) |
| **StatisticsControllers** | The StatisticsController class contains action methods that render view pages (Statistics) |
| **SaleControllers** | The SaleController class contains action methods that render view pages (BillManagement, Checkout, PriceLog) |
| **StoreControllers** | The StoreController class contains action methods that render view pages (AddStore, ViewStore, EditStore) |
| **LoyalMember**  **Controllers** | The LoyalMemberController class contains action methods that render view pages (AddLoyalMember, ViewLoyalMember, EditLoyalMember) |
| **StoreCategory**  **Controllers** | The StoreCategoryController class contains action methods that render view pages (AddStoreCategory, ViewStoreCategory, EditStoreCategory) |
| **ComputerControllers** | The POSController class contains action methods that render view pages (AddPOS, ViewPOS, EditPOS) |
| **UserControllers** | The UserController class contains action methods that render view pages (AddUser, ViewUser, EditUser) |
| View | **AddStore** | This GUI helps the user to add a new Store. |
| **ViewStore** | This GUI helps the user to view the Store. |
| **EditStore** | This GUI helps the user to modify some information about the Store. |
| **AddCategory** | This GUI helps the user to add a new Product Category. |
| **ViewCategory** | This GUI helps the user to view the Product Category. |
| **EditCategory** | This GUI helps the user to modify some information about the Product Category. |
| **AddProduct** | This GUI helps the user to add a new Product. |
| **ViewProduct** | This GUI helps the user to view the Product. |
| **EditProduct** | This GUI helps the user to modify some information about the Product. |
| **BillManagement** | This GUI helps the user to view the list of Bills and the user can see some detail information of each Bill. |
| **PriceLog** | This GUI shows all Price history that was used for each product |
| **Checkout** | This GUI helps the user check bill and make a payment. |
| **AddUser** | This GUI helps the user to add a new User. |
| **ViewUser** | This GUI helps the user to view the User. |
| **EditUser** | This GUI helps user to modify some information about the User. |
| **AddLoyalMember** | This GUI helps the user to add a new loyal Customer. |
| **ViewLoyalMember** | This GUI helps the user to view the loyal Customer. |
| **EditLoyalMember** | This GUI helps user to modify some information about the loyal Customer. |
| **AddStoreCategory** | This GUI helps the user to add a new Retail Store Category. |
| **ViewStoreCategory** | This GUI helps the user to view the Retail Store Category. |
| **EditStoreCategory** | This GUI helps user to modify some information about the Retail Store Category. |
| **AddComputer** | This GUI helps the user to add a new Computer. |
| **EditComputer** | This GUI helps user to modify some information about the Computer. |
| **ViewComputer** | This GUI helps the user to view the Computer. |
| **Statistics** | To make statistics about the total amount of product (or product category) was bought on month. |
| Model | **ProductModels** | Storing and retrieving the Product information and return a message back to view pages (AddProduct, EditProduct, ViewProduct) |
| **SaleModels** | Storing and retrieving the Sale information and return a message back to view pages (BillManagement, Checkout, PriceLog) |
| **StoreModels** | Storing and retrieving the Retail Store information and return a message back to view pages (AddStore, EditStore, ViewStore) |
| **LoyalMember**  **Models** | Storing and retrieving the Customer information and return a message back to view pages (AddLoyalMember, EditLoyalMember, ViewLoyalMember) |
| **UserModels** | Storing and retrieving the Userinformation and return a message back to view pages (AddUser, EditUser, ViewUser) |
| **CategoryModels** | Storing and retrieving the Product Categoryinformation and return a message back to view pages (AddProductCategory, EditProductCategory, ViewProductCategory) |
| **StoreCategory**  **Models** | Storing and retrieving the Store Category information and return a message back to view pages (AddStoreCategory, EditStoreCategory, ViewStoreCategory) |
| **ComputerModels** | Storing and retrieving the POSinformation and return a message back to view pages (AddPOS, EditPOS, ViewPOS) |
| **StatisticsModels** | Storing and retrieving the some information related Statistics and return a message back to view pages (Statistics) |
| **DBContext** | Context class is the primary class for interacting with data as objects that are instances of entity types that are defined in an Entity Data Model (EDM). |

* + 1. Relations and their properties

|  |  |
| --- | --- |
| Connector | Properties |
| Allowed to use | The layers are related to each other by the strictly ordered relation allowed to use. |
| Uses | The uses style shows how modules depend on each other; it is helpful for planning because it helps define subsets and increments of the system being developed. |

## Architecture background:

Why decision design meets the requirement will be explained as below:

* A product have a specific point that will be awarded when the member purchase products with cash.
* Users can enter these codes by using computer keyboard if they don’t have any barcode reader.
* The loyal member either pays with cash, points, or a combination of the two on Sale UI.
* When the points are used, the number of points used is immediately subtracted from the number of points accrued by the member
* Products are classified into categories (kind of product types) such as food, general merchandise. And user can manage categories
* For each product, its standard price, common to all stores, is set as a part of the product data. Each store, however, can set and use its own actual retail price instead of the standard price. The actual retail price must be set in advance. User can

# Data Model

## Primary presentation:



## Element Catalog

### Elements and their properties

|  |  |  |  |
| --- | --- | --- | --- |
| Entity | Attributes | Data Type | Description |
| Bill | **BillID** | Varchar(9) |  |
| **ComputerMAC** | VARCHAR(17) | Attribute said bill is made in which Computer well as the general store |
| **CustomerID** | Varchar(9) | Customer pays the invoice. |
| **UserID** | Varchar(9) | Cashier |
| TotalCost | FLOAT | The total cost of the bill, ensuring the implementation of Statistical Performance Data |
| Date | DATETIME | Paid Bill Date |
| PlusPoint | INT | Minus and plus points in a session will be stored here. |
| MinusPoint | INT |
| Product | **ProductID** | Varchar(9) |  |
| Product\_Name | NVARCHAR(50) |  |
| BasicCost | FLOAT | AttributeBasic Cost show the default price of the product |
| **CategoryID** | Varchar(9) |  |
| Stock |  | The merchandise that a shop has on hand |
| PrStatus | BIT |  |
| Bill\_Detail | **BillID** | Varchar(9) |  |
| **ProductID** | Varchar(9) |  |
| Quantity | INT | Sum Loyal Point of customer |
| Customer | **CustomerID** | Varchar(9) |  |
| Customer\_Name | NVARCHAR(50) |  |
| Customer\_Address | NVARCHAR(50) |  |
| Customer\_Phone | VARCHAR(15) |  |
| SumPoint | INT | Sum Loyal Point of customer |
| CuStatus | BIT |  |
| RetailStore | **RetailStoreID** | Varchar(9) |  |
| RetailStore\_Name | NVARCHAR(50) |  |
| ReStatus | BIT |  |
| Cost | **ProductID** | Varchar(9) | Entity Cost said that Retail Store Retail Store selling a certain product and pricing individual products within a certain time. |
| **RetailStoreID** | Varchar(9) |
| DateStart | DATETIME |
| DateEnd | DATETIME |
| Cost | FLOAT |
| Category | **CategoryID** | Varchar(9) |  |
| Category\_Name | NVARCHAR(50) |  |
| CaStatus | BIT |  |
| RetailStore\_Category | **RetailStoreID** | Varchar(9) |  |
| **CategoryID** | Varchar(9) |  |
| Quantity | INT |  |
| Computer | **ComputerMAC** | VARCHAR(17) |  |
| **RetailStoreID** | Varchar(9) | This Attribute tells us this POS Terminal is placed at which Retail Store |
| CoStatus | BIT |  |
| User | **UserID** | Varchar(9) |  |
| User\_Name | NVARCHAR(50) |  |
| User\_Address | NVARCHAR(50) |  |
| User\_Phone | VARCHAR(15) |  |
| Password | VARCHAR(32) |  |
| **RetailStoreID** | Varchar(9) | This Attribute tells us this user works at which Retail Store |
| UsStatus | BIT |  |

## Architecture Background

* Data model design satisfy business constrains following:
  + Not all stores carry every product type, and the range of product types carried is designated for each store.
  + Can set and use its own actual retail price instead of the standard price during the limited period specified be each store.

# Dynamic Perspective

**Component and Connector View**

## Primary presentation

## Element catalog:

### Elements and their properties

|  |  |  |
| --- | --- | --- |
| Elements | | Properties |
| Client tier | **Web Browser** | A component that send a request to web server and receive a reply is sent by web server. And then it displays UI to user. |
| Server Tier | **IIS 7** | IIS 7 is a major enhancement to the Windows web platform and plays a central role in unifying Microsoft web platform technologies - ASP.NET, Windows Communication Foundation web services, and Windows SharePoint Services.  It handle requests and replies that are send between client tier and server tier. |
| **ProductControllers** | The ProductController class contains action methods that render view pages (AddProduct, EditProduct, ViewProduct) |
| **Category**  **Controllers** | The CategoryController class contains action methods that render view pages (AddCategory, EditCategory, ViewCategory) |
| **StatisticsControllers** | The StatisticsController class contains action methods that render view pages (Statistics) |
| **SaleControllers** | The SaleController class contains action methods that render view pages (BillManagement, Checkout, PriceLog) |
| **StoreControllers** | The StoreController class contains action methods that render view pages (AddStore, ViewStore, EditStore) |
| **LoyalMember**  **Controllers** | The LoyalMemberController class contains action methods that render view pages (AddLoyalMember, ViewLoyalMember, EditLoyalMember) |
| **StoreCategory**  **Controllers** | The StoreCategoryController class contains action methods that render view pages (AddStoreCategory, ViewStoreCategory, EditStoreCategory) |
| **ComputerControllers** | The POSController class contains action methods that render view pages (AddPOS, ViewPOS, EditPOS) |
| **UserControllers** | The UserController class contains action methods that render view pages (AddUser, ViewUser, EditUser) |
| **AddStore** | This GUI helps the user to add a new Store. |
| **ViewStore** | This GUI helps the user to view the Store. |
| **EditStore** | This GUI helps the user to modify some information about the Store. |
| **AddCategory** | This GUI helps the user to add a new Product Category. |
| **ViewCategory** | This GUI helps the user to view the Product Category. |
| **EditCategory** | This GUI helps the user to modify some information about the Product Category. |
| **AddProduct** | This GUI helps the user to add a new Product. |
| **ViewProduct** | This GUI helps the user to view the Product. |
| **EditProduct** | This GUI helps the user to modify some information about the Product. |
| **BillManagement** | This GUI helps the user to view the list of Bills and the user can see some detail information of each Bill. |
| **PriceLog** | This GUI shows all Price history that was used for each product |
| **Checkout** | This GUI helps the user check bill and make a payment. |
| **AddUser** | This GUI helps the user to add a new User. |
| **ViewUser** | This GUI helps the user to view the User. |
| **EditUser** | This GUI helps user to modify some information about the User. |
| **AddLoyalMember** | This GUI helps the user to add a new loyal Customer. |
| **ViewLoyalMember** | This GUI helps the user to view the loyal Customer. |
| **EditLoyalMember** | This GUI helps user to modify some information about the loyal Customer. |
| **AddStoreCategory** | This GUI helps the user to add a new Retail Store Category. |
| **ViewStoreCategory** | This GUI helps the user to view the Retail Store Category. |
| **EditStoreCategory** | This GUI helps user to modify some information about the Retail Store Category. |
| **AddComputer** | This GUI helps the user to add a new Computer. |
| **EditComputer** | This GUI helps user to modify some information about the Computer. |
| **ViewComputer** | This GUI helps the user to view the Computer. |
| **Statistics** | To make statistics about the total amount of product (or product category) was bought on month. |
| **ProductModels** | Storing and retrieving the Product information and return a message back to view pages (AddProduct, EditProduct, ViewProduct) |
| **SaleModels** | Storing and retrieving the Sale information and return a message back to view pages (BillManagement, Checkout, PriceLog) |
| **StoreModels** | Storing and retrieving the Retail Store information and return a message back to view pages (AddStore, EditStore, ViewStore) |
| **LoyalMember**  **Models** | Storing and retrieving the Customer information and return a message back to view pages (AddLoyalMember, EditLoyalMember, ViewLoyalMember) |
| **UserModels** | Storing and retrieving the Userinformation and return a message back to view pages (AddUser, EditUser, ViewUser) |
| **CategoryModels** | Storing and retrieving the Product Categoryinformation and return a message back to view pages (AddProductCategory, EditProductCategory, ViewProductCategory) |
| **StoreCategory**  **Models** | Storing and retrieving the Store Category information and return a message back to view pages (AddStoreCategory, EditStoreCategory, ViewStoreCategory) |
| **ComputerModels** | Storing and retrieving the POSinformation and return a message back to view pages (AddPOS, EditPOS, ViewPOS) |
| **StatisticsModels** | Storing and retrieving the some information related Statistics and return a message back to view pages (Statistics) |
| **TempDB.xml** | When client send a request to get data from DB to display UI. The system will store this data in TempDB as a temporary data. Afterwards, if client need this data, and sent request to Model, it will read TempDB file and return data to client to display UI, needless to access database. |
| Database tier | **Primary Database** | It’s main database server which uses frequently when the system work properly. It’s responsible for store data such as sales data, user data, customer data, store data, product data, and category data. In the certain time, it will synch with temporary database as a backup data. |
| **Backup Database** | It’s a temporary database which uses rarely. It only uses when the system doesn’t work properly, crash or not available. In the certain time, it will synch all data with primary. |

### Relations and their properties

|  |  |
| --- | --- |
| Connector | Properties |
| Request/ Reply | Connector between client and server style, used by a client to invoke services on a server. |
| Call and return | Responsible for conveying the service request from the requester to the provider and for returning any results. Use by interface to request data from Filter/Object |
| Synchronous Replication | The technique for replicating data by two or more databases (or file systems) where the system being replicated does waits for the data to have been recorded on the duplicate system before proceeding. |
| File I/O | Refer the communication in access data in a file to perform operation such as: read file and write file. |
| ODBC | Stands for Open Database Connectivity. It is the standard method which allows any application to connect data. ODBC uses a middle layer called the database driver to handles the connection in between the application and the relational database management system. |

### Element behavior

* + - 1. Add Store



* + - 1. Add Loyal Member



* + - 1. AddUser



* + - 1. AddUserComputer



* + - 1. Statistic



* + - 1. AddNewCategogy



* + - 1. AddNewProduct



* + - 1. Sale



# Mapping between perspectives

* Show mappings between structures in different perspectives if necessary.
* Show how specific structures in one perspective map to specific structures in other perspectives.
* Be careful in the mappings and use prose to describe any mappings you show.

# Document directory

In this section include an index, glossary, and acronym list