<Company Name>

Computer Shop System

Software Architecture Document

Version 1.0

Revision History

Date	Version	Description	Author
25/12/2019	1.0	Final Draft.	Phung Minh Nguyet

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This document applies to the Computer Shop System which will be developed by CT Computer.

1.3 Definitions, Acronyms, and Abbreviations

Admin: the person who develop the system and manage all in the computer shop.

Staff: who work for the shop.

Customer: a user who is not logged in the system.

1.4 References

None

1.5 Overview

In the following section, architectural design of the Computer Shop Management System is provided in detail. First, the primary software architecture of the system will be defined. Then, there are further discussion about the goals and constraints that will be imposed upon the quality of the final product, which including but not limited to security, distribution and reuse. In the precedence sections, the key views of the system are demonstrated to depict different aspects of the system. Lastly, criteria concerning with size, performance and quality of the system will be proposed.

2. Architectural Representation

This documents presents the architectural as a series of mandatory views: Use-Case View, Logical View, Deployment View and Data View. These views are presented as Visual Paradigm Community Edition Models , StarUML and use the Unified Modeling Language (UML).

Use-Case View

- Audience: all the stakeholders of the system, including the end-users.
- Area: describes the set of scenarios and/or use cases that represent significant, central functionality to the system.
- Related artifacts: Use-Case Model, Analysis Model, Use-Case-Realization documents.

Logical View

- Audience: designers, programmers.
- Area: functional requirements: describes the design's object model.
- Related artifacts: Design Model.

Deployment View

- Audience: deployment managers, system administrators.
- Area: topology: describes the mapping of the software onto the hardware and shows the systems distributed aspects.
- Related artifacts: Deployment Model.

Data View

- Audience: data specialists, database administrators.
- Area: persistence: describes the architecturally significant persistent elements in the data model.
- Related artifacts: Data Model.

3. Architectural Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

- The Computer Shop Management System must be designed to fulfills all system requirements specified in requirements definition.
- The Computer Shop Management design must be structured to be robust, easy to change if and when functional requirements change.
- The Computer Shop Management System must be designed to allow the re-use of business logic across applications; therefore, the design separate the three components: model, view and controller.
- The separation of the three components: model, view and controller are also necessary to provide a convenient cooperation between different development teams.
- The Computer Shop Management System will run on a dedicated platform with access to a database.
- The Computer Shop Management website provides most of the content display. An interface to this system must be capable of handling large traffic volumes.

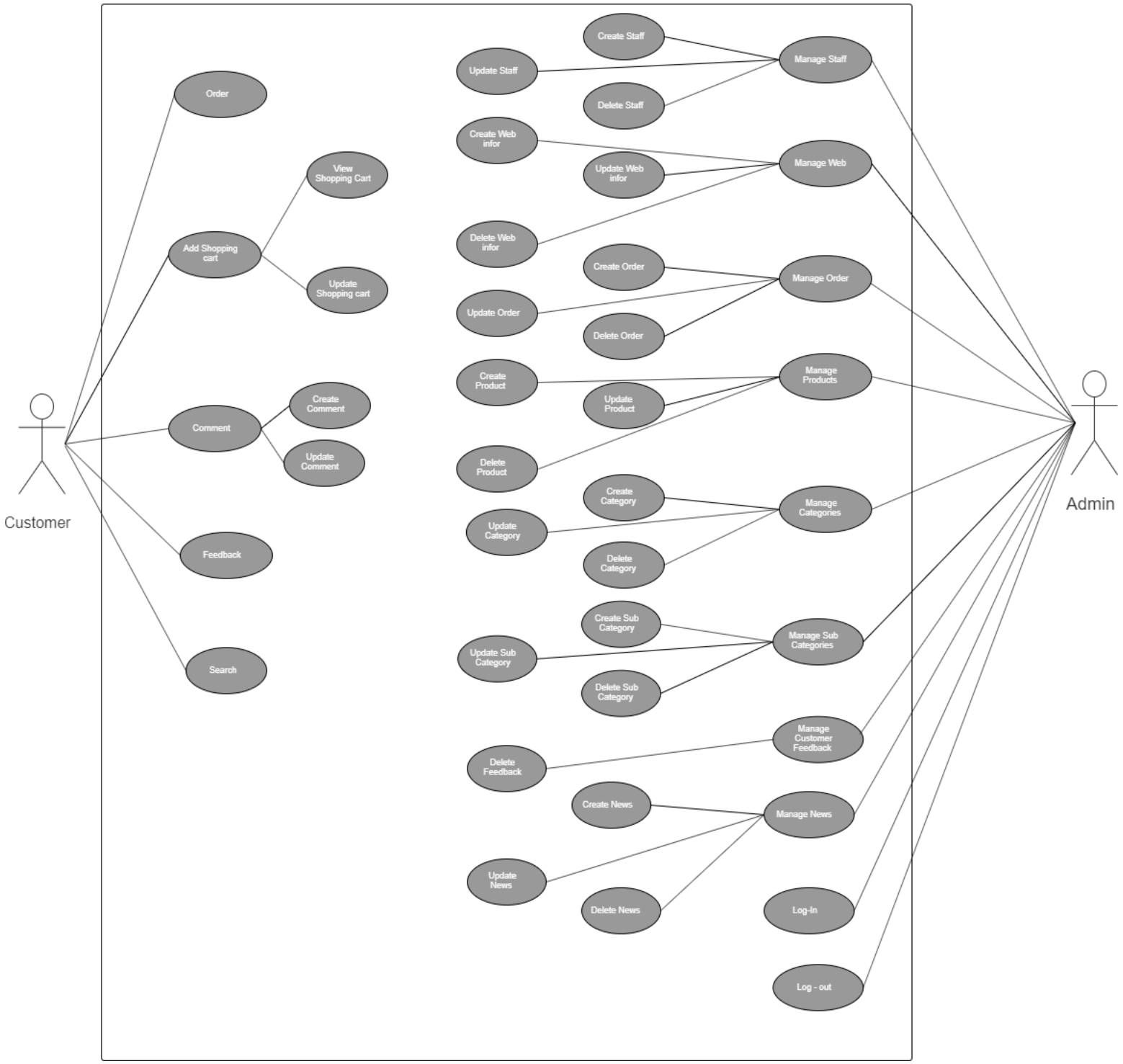
4. Use-Case View

A description of the Use-Case View of the system architecture. The Use Case View is important input to the selection of the set of scenarios and/or use cases that are the focus of an iteration. It describes the set of scenarios and/or use cases that represent some significant, central functionality. It also describes the set of scenarios and/or use cases that have a substantial architectural coverage (that exercise many architectural elements) or that stress or illustrate a specific, delicate point of the architecture.

The significant use cases in this system are listed below:

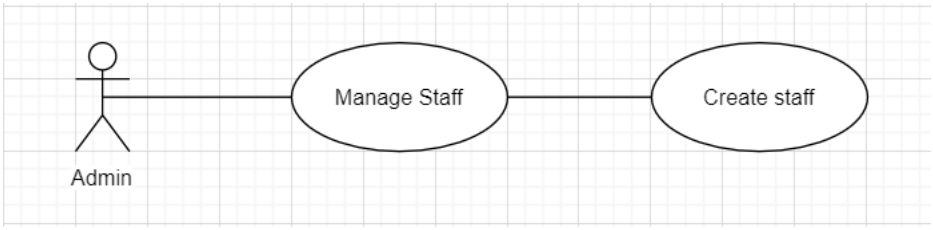
- Log-in
- Log-out
- Search
- Feedback
- Create comment
- Delete comment
- View shopping cart
- Create Staff
- Update Staff
- Delete Staff

- Manage web
- Create order
- Delete order
- Create product
- Update product
- Delete product
- Create Category
- Update Category
- Delete Category
- Create news
- Update News
- Delete News
- Create Sub Category
- Update Sub Category
- Delete Sub Category
- Create customer feedback
- Delete customer feedback


Customer
Admin
Order
View Shopping Cart
Add Shopping cart
Update Shopping cart
Comment
Create Comment
Update Comment
Feedback
Search
Create Staff
Update Staff
Delete Staff
Manage Staff
Create Web infor
Update Web infor
Delete Web infor
Manage Web
Create Order
Update Order
Delete Order
Manage Order
Create Product
Update Product
Delete Product
Manage Products
Create Category
Update Category
Delete Category
Manage Categories
Create Sub Category
Update Sub Category
Delete Sub Category
Manage Sub Categories
Delete Feedback
Manage Customer Feedback
Create News
Update News
Delete News
Manage News
Log-In
Log - out

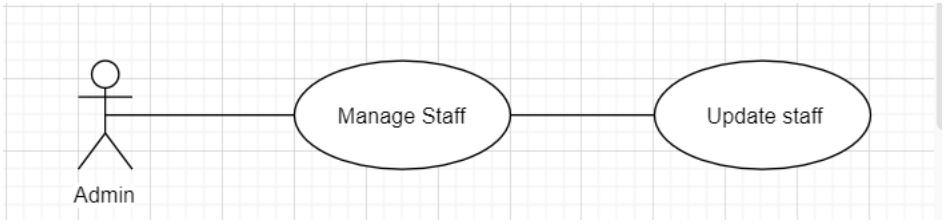
4.1 Use-Case Realizations

1. Create Staff:



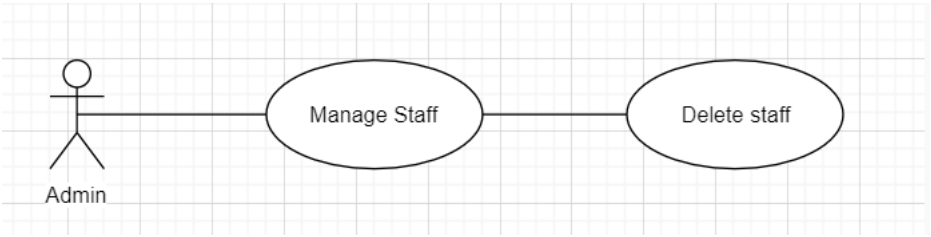
- Brief Description: Admin create staff's information.
- Specification: See Use-Case-Realization Specification: Create Staff

2. Update Staff:



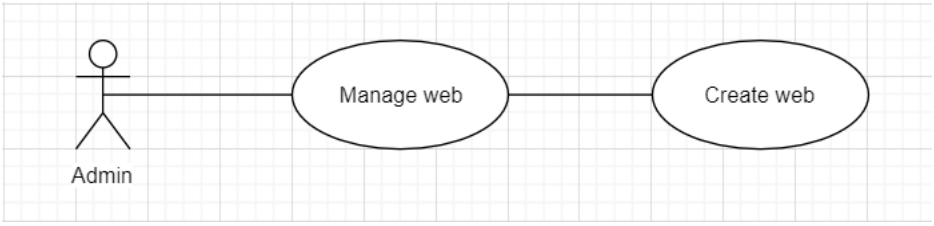
- Brief Description: Admin update staff's information.
- Specification: See Use-Case-Realization Specification: Update Staff

3. Delete Staff:



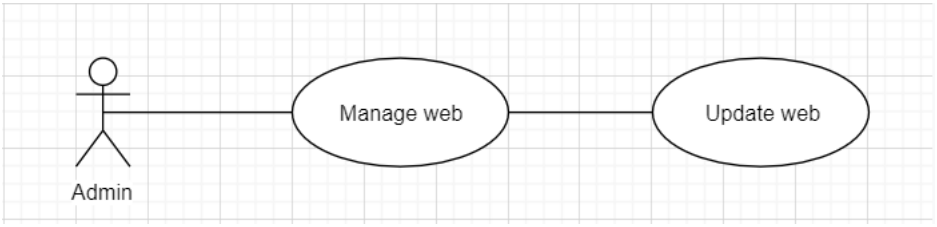
- Brief Description: Admin delete staff's information.
- Specification: See Use-Case-Realization Specification: Delete Staff

4. Create Web:



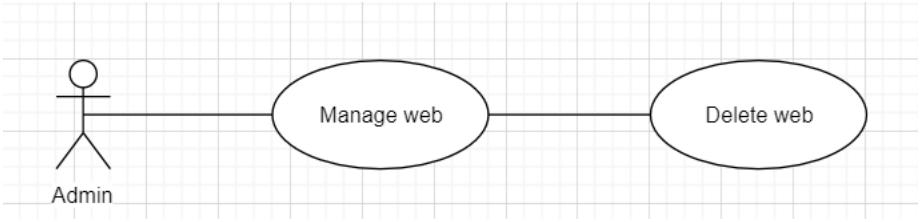
- Brief Description: Admin create web's information.
- Specification: See Use-Case-Realization Specification: Create Web

5. Update Web :



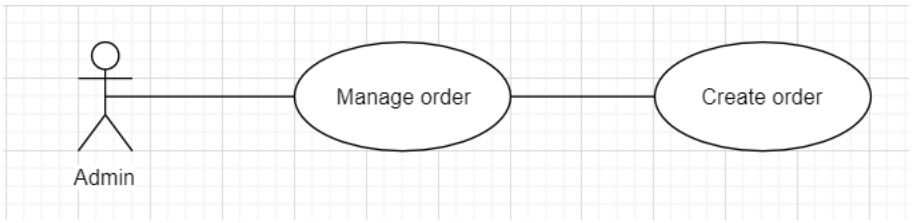
- Brief Description: Admin update web's information.
- Specification: See Use-Case-Realization Specification: Update Web

6. Delete Web:



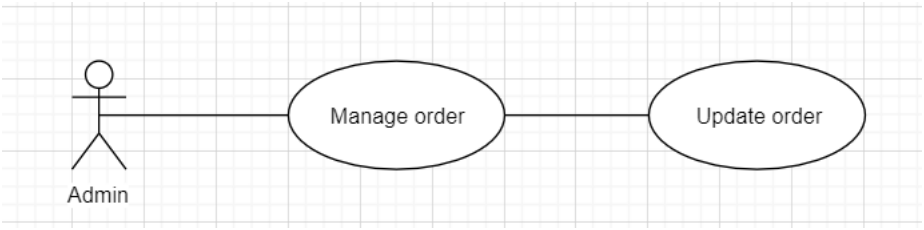
- Brief Description: Admin delete web's information.
- Specification: See Use-Case-Realization Specification: Delete Web

7. Create order:



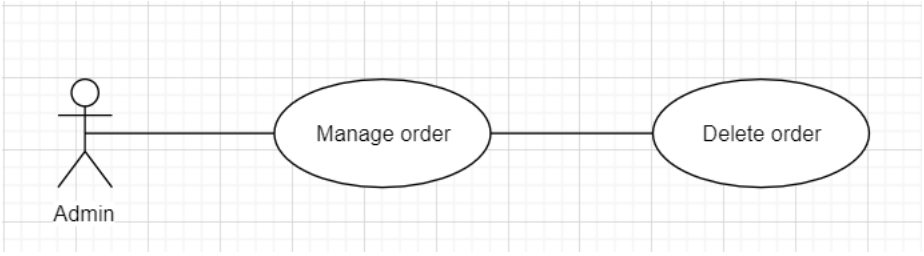
- Brief Description: Admin create order.
- Specification: See Use-Case-Realization Specification: Create order

8. Update order:



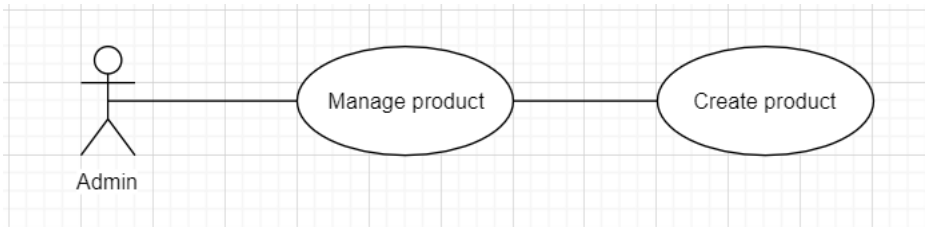
- Brief Description: Admin update order.
- Specification: See Use-Case-Realization Specification: Update order

9. Delete order:



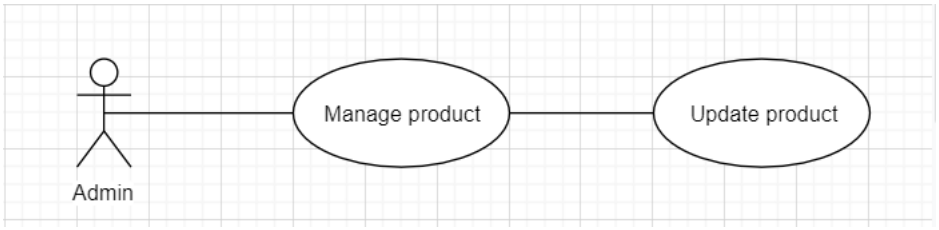
- Brief Description: Admin delete order.
- Specification: See Use-Case-Realization Specification: Delete order.

10. Create product:



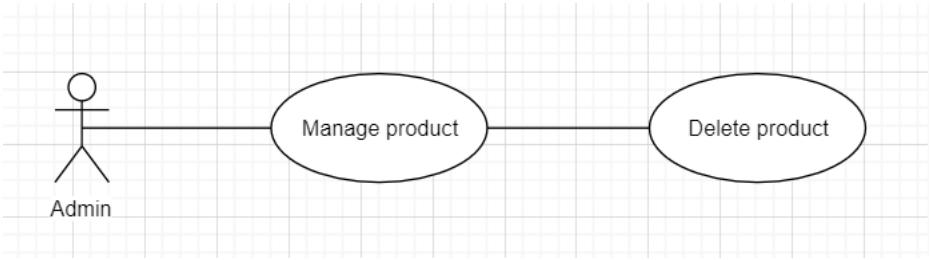
- Brief Description: Admin create product.
- Specification: See Use-Case-Realization Specification: Create product.

11. Update product:



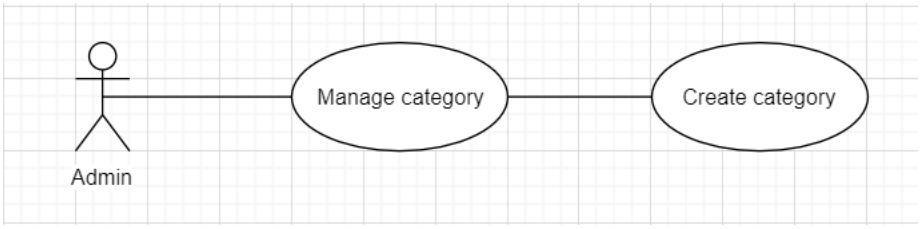
- Brief Description: Admin update product.
- Specification: See Use-Case-Realization Specification: Update product.

12. Delete product:



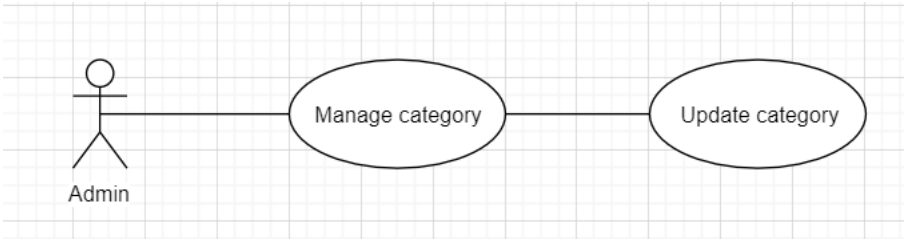
- Brief Description: Admin delete product.
- Specification: See Use-Case-Realization Specification: Delete product.

13. Create category:



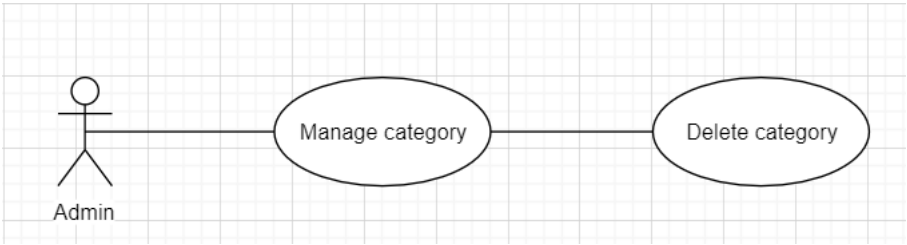
- Brief Description: Admin create category.
- Specification: See Use-Case-Realization Specification: Create category.

14. Update category:



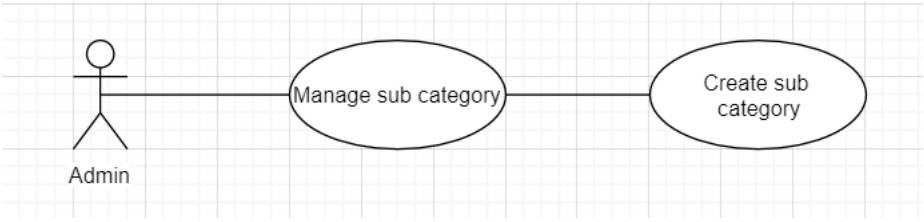
- Brief Description: Admin update category.
- Specification: See Use-Case-Realization Specification: Update category.

15. Delete category:



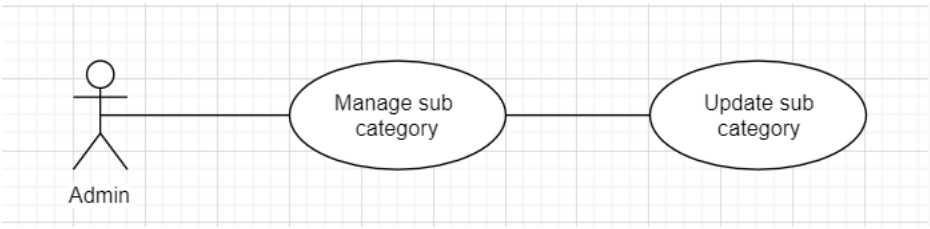
- Brief Description: Admin delete category.
- Specification: See Use-Case-Realization Specification: Delete category.

16. Create sub category:



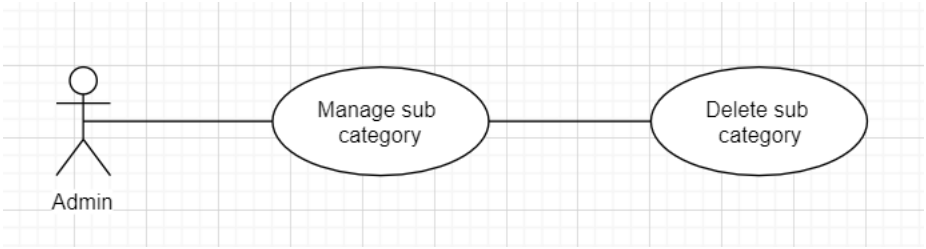
- Brief Description: Admin create sub category.
- Specification: See Use-Case-Realization Specification: Create sub category.

17. Update sub category:



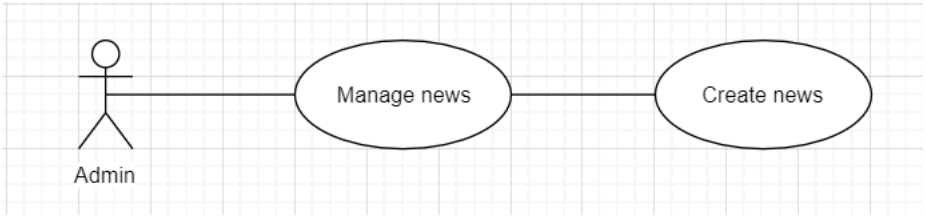
- Brief Description: Admin update sub category.
- Specification: See Use-Case-Realization Specification: Update sub category.

18. Delete sub category:



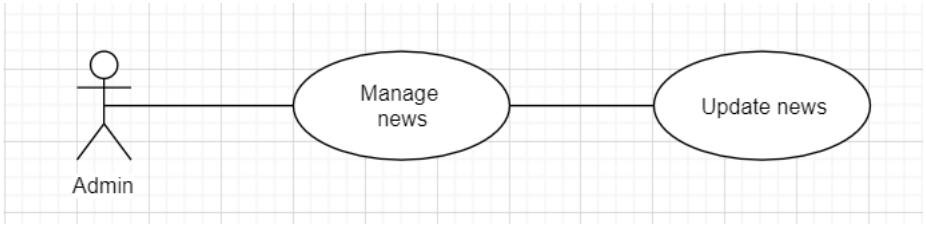
- Brief Description: Admin delete sub category.
- Specification: See Use-Case-Realization Specification: Delete sub category.

19. Create news:



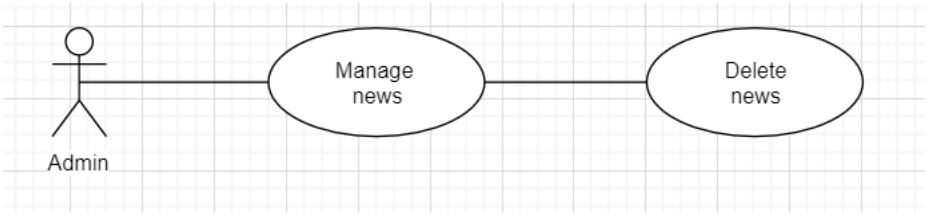
- Brief Description: Admin create news.
- Specification: See Use-Case-Realization Specification: Create news.

20. Update news:



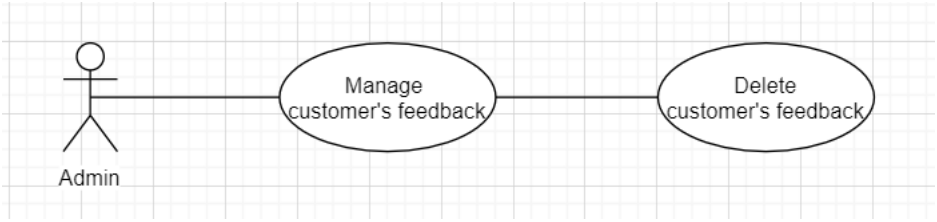
- Brief Description: Admin update news.
- Specification: See Use-Case-Realization Specification: Update news.

21. Delete news:



- Brief Description: Admin delete news.
- Specification: See Use-Case-Realization Specification: Delete news.

22. Delete customer's feedback:



- Brief Description: Admin delete customer's feedback.
- Specification: See Use-Case-Realization Specification: Delete customer's feedback.

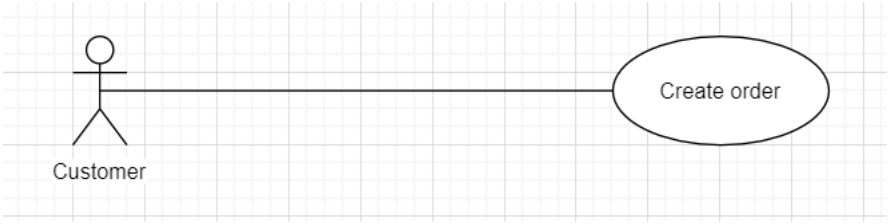
23. Log-in:

- Brief Description: Staff logging in to the system.
- Specification: See Use-Case-Realization Specification: Log-in

24. Log-out:

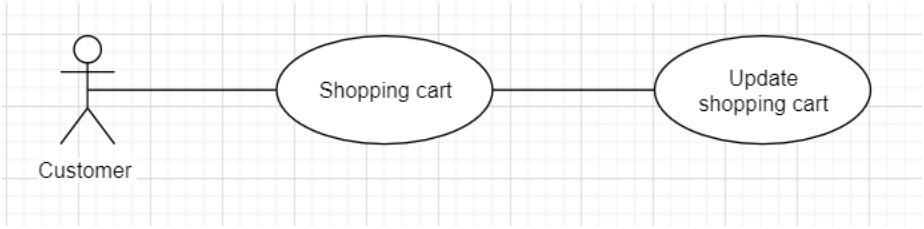
- Brief Description: Staff logging out to the system.
- Specification: See Use-Case-Realization Specification: Log-out.

25. Create order:



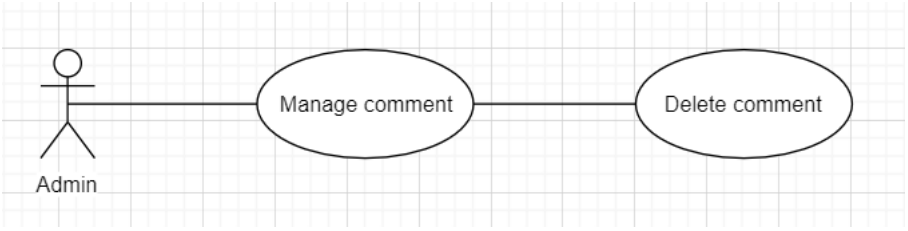
- Brief Description: Customer create the order.
- Specification: See Use-Case-Realization Specification: Create news.

26. Update shopping cart :



- Brief Description: Customer update shopping cart.
- Specification: See Use-Case-Realization Specification: Update shopping cart.

27. Delete comment:

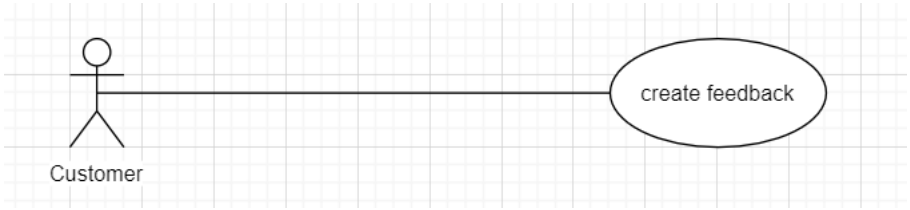


- Brief Description: Admin delete comment.
- Specification: See Use-Case-Realization Specification: Delete comment.

28. Create comment:

- Brief Description: Customer create comment.
- Specification: See Use-Case-Realization Specification: Create comment

29. Feedback:



- Brief Description: Customer create the feedback to the shop.
- Specification: See Use-Case-Realization Specification: Feedback

30. Search:

- Brief Description: Customer search the product.
- Specification: See Use-Case-Realization Specification: Search.

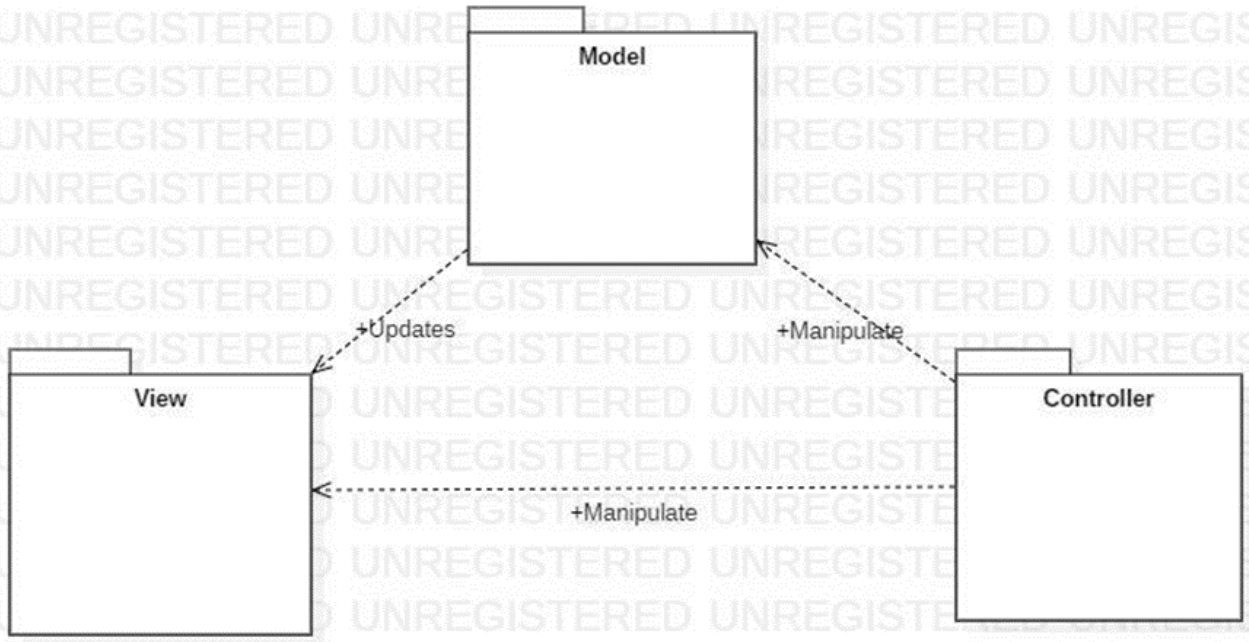
5. Logical View

5.1 Overview

A description of the logical view of the architecture. Describes the overall decomposition of the design model in terms of package hierarchy and layers.

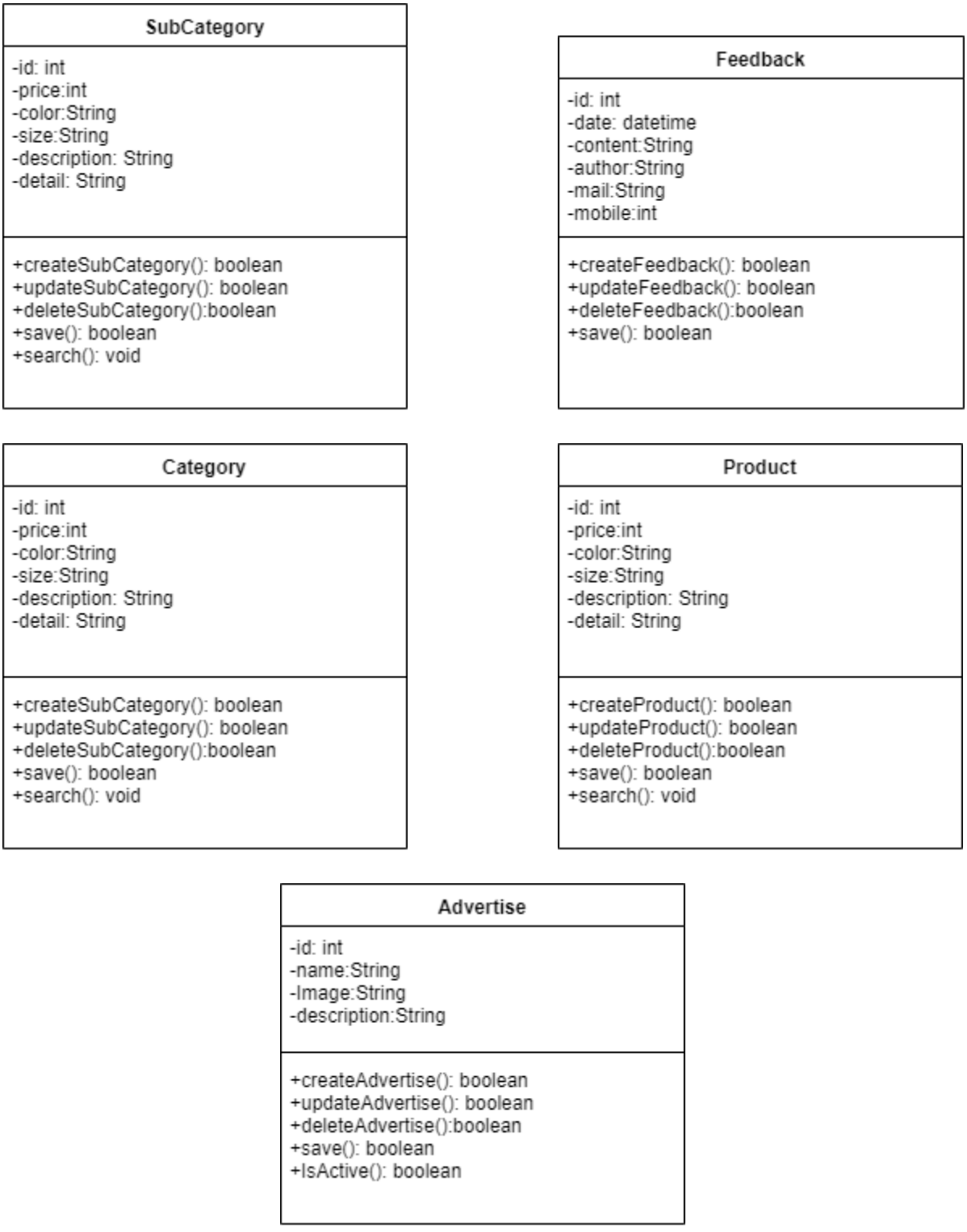
The logical view of the Computer Shop System is comprised of 3 significant packages:

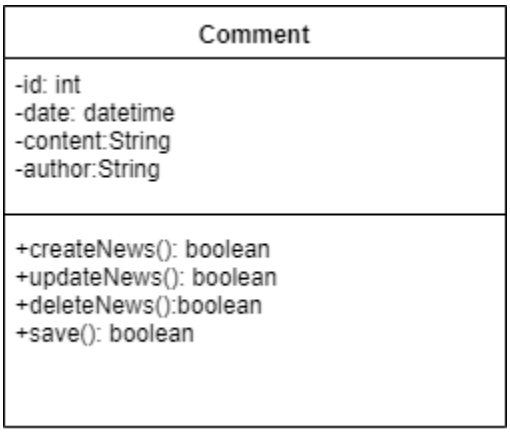
- Model: contains classes that directly manages the data, logic and rules of the Computer Shop System and displayed in the view.
- View: contains classes that generates output representation of information to the user based on changes in the model.
- Controller: contains classes that can send commands to the model to update the model's state (e.g., add a new computer); it can also send commands to its associated view to change the view's presentation of the model (e.g., scrolling through contract's reviews).

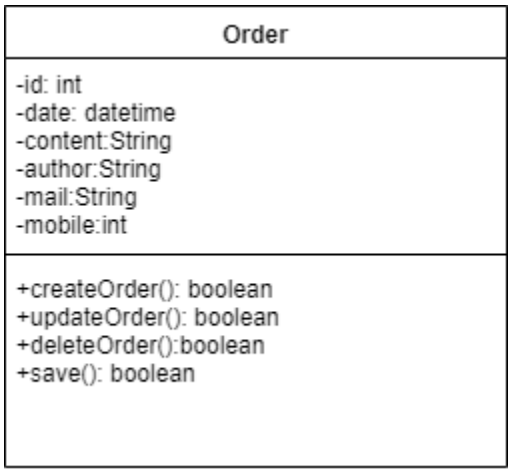


5.2 Architecturally Significant Design Packages

5.2.1 Package Model







Name	model
Brief Description	Contains classes that directly manages the data, logic and rules of the Computer Shop Management System and displayed in the view.
Classes	Computer, Rating, Order, OrderLine, Payment.

- Class Categories:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	250	N/A	N/A
position	int	Private	True	False	4	N/A	N/A

Operations				
Header	**Return Type**	**Access**	**Scope**	**Specification**
Create()	Boolean	Public	Instance	Create new Category Return true if success.
Update()	Boolean	Public	Instance	Update Category. Return true if success
Delete()	Boolean	Public	Instance	Delete Category
Display()	Boolean	Public	Instance	Display Categories list/information true if success.
Save()	Boolean	Public	Instance	Save Category in database .
Search()	void	Private	Instance	Search Category

- Class Sub Categories:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
category_id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	250	N/A	N/A
position	int	Private	True	False	4	N/A	N/A

Operations				
Header	**Return Type**	**Access**	**Scope**	**Specification**
Create()	boolean	Public	Instance	Create Sub Category Return true if success.
Update()	boolean	Public	Instance	Update Sub Category. Return true if success

Delete()	boolean	Public	Instance	Delete Sub Category
Display()	boolean	Public	Instance	Display Sub Category list/information true if success.
Save()	boolean	Public	Instance	Save Sub Category in database .
Search()	void	Private	Instance	Search Sub Category

- Class product:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	550	N/A	N/A
CategoryId	int	Private	False	False	11	1	N/A
SubCategoryId	int	Private	False	False	11	1	N/A
typeId	int	Private	False	False	11	1	N/A
Price	int	Private	False	False	11	1	N/A
Color	String	Private	True	False	250	N/A	N/A
Material	String	Private	True	False	250	N/A	N/A
Size	String	Private	True	False	20	N/A	N/A
Operations							
Header	**Return Type**	**Access**	**Scope**	**Specification**			
Create()	boolean	Public	Instance	Create new product Return true if success.			
Update()	boolean	Public	Instance	Update product. Return true if success			
Delete()	boolean	Public	Instance	Delete product			
Display()	boolean	Public	Instance	Display product list/information true if success.			
Save()	boolean	Public	Instance	Save product in database .			
Search()	void	Private	Instance	Search product			

- Class feedback:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	50	N/A	N/A
mail	String	Private	True	False	50	N/A	N/A
mobile	String	Private	True	False	20	N/A	N/A
Operations							

Header	**Return Type**	**Access**	**Scope**	**Specification**
Create()	boolean	Public	Instance	Create new feedback Return true if success.
Update()	boolean	Public	Instance	Update feedback .Return true if success
Delete()	boolean	Public	Instance	Delete feedback
Display()	boolean	Public	Instance	Display feedback list/information true if success.
Save()	boolean	Public	Instance	Save feedback in database .

- Class advertise:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	150	N/A	N/A
description	String	Private	True	False	250	N/A	N/A
Operations							

Header	**Return Type**	**Access**	**Scope**	**Specification**
Create()	boolean	Public	Instance	Create new advertise Return true if success.
Update()	boolean	Public	Instance	Update advertise. Return true if success
Delete()	boolean	Public	Instance	Delete advertise
Display()	boolean	Public	Instance	Display advertise list/information

				true if success.
Save()	boolean	Public	Instance	Save advertise in database .

- Class Comment:

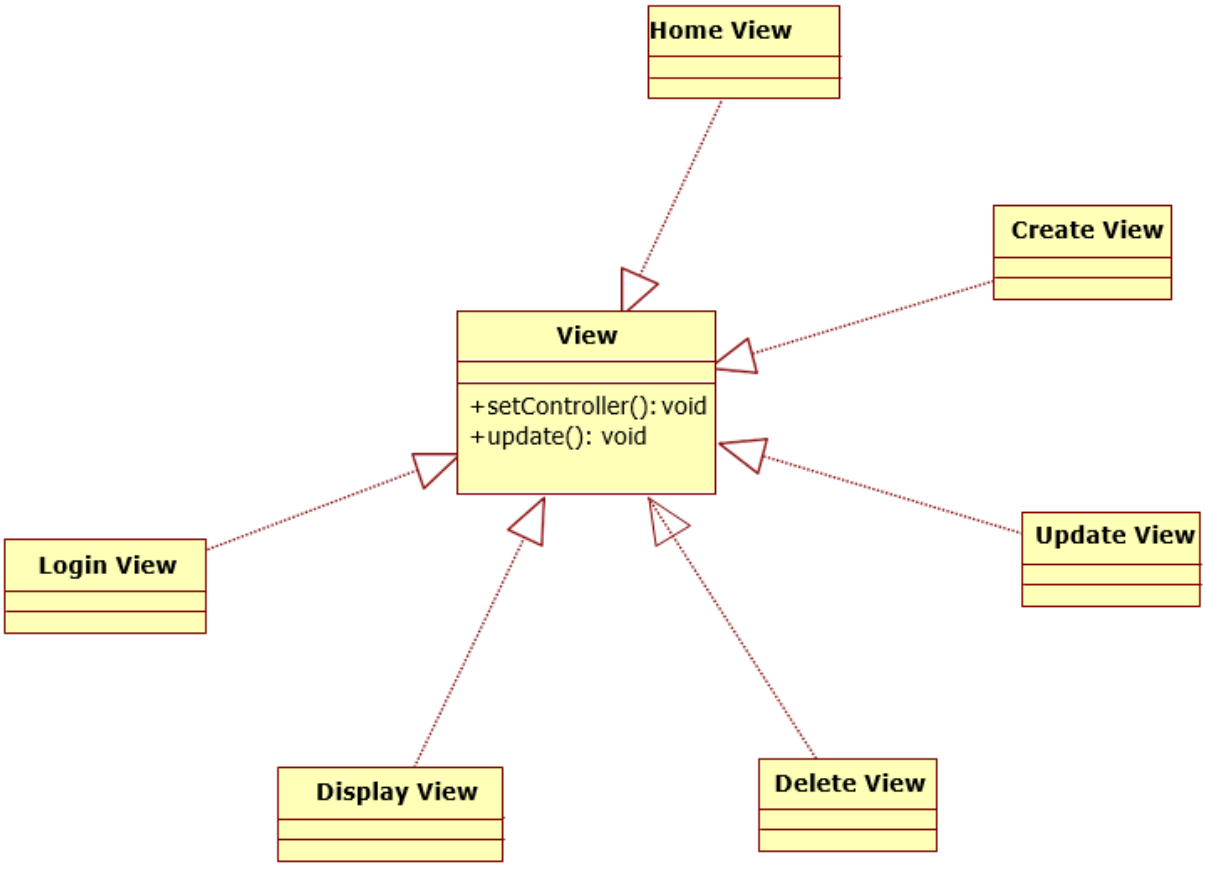
Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	250	N/A	N/A
position	int	Private	True	False	4	N/A	N/A
Operations							
Header	**Return Type**	**Access**	**Scope**	**Specification**			
Create()	boolean	Public	Instance	Create new Comment Return true if success.			
Update()	boolean	Public	Instance	Update Comment. Return true if success			
Delete()	boolean	Public	Instance	Delete Comment			
Display()	boolean	Public	Instance	Display Comment list/information true if success.			
Save()	boolean	Public	Instance	Save Comment in database .			

- Class order:

Name	User						
Brief Description	Data model for user table in database.						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
id	int	Private	False	False	11	1	N/A
name	String	Private	True	False	50	N/A	N/A
address	String	Private	True	False	200	N/A	N/A
phone	String	Private	True	False	20	N/A	N/A
email	String	Private	True	False	50	N/A	N/A
Operations							

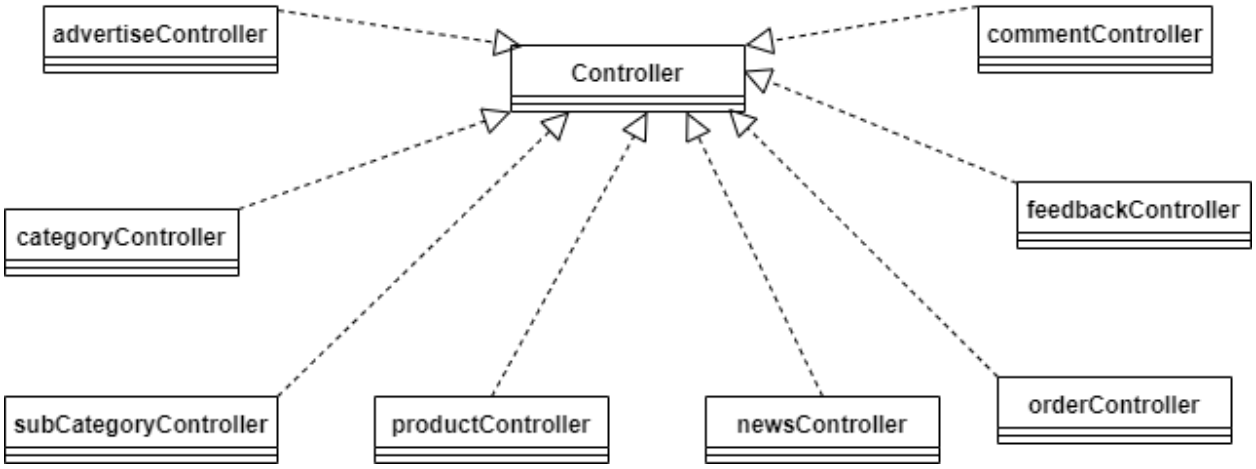
Header	Return Type	Access	Scope	Specification
Create()	boolean	Public	Instance	Create new order Return true if success.
Update()	boolean	Public	Instance	Update order. Return true if success
Delete()	boolean	Public	Instance	Delete order
Display()	boolean	Public	Instance	Display order list/information true if success.
Save()	boolean	Public	Instance	Save order in database .

5.2.2 Package View



Name	view
Brief Description	Contains classes that generates output representation of Information to the user based on changes in the model.
Interfaces	View.
Classes	HomeView, LoginView, DisplayView CreateView, UpdateView, DeleteView,

5.2.3 Package Controller



Name	Controller
Brief Description	Controls the data flow into model object and updates the view whenever data changes
Classes	Controller, advertiseController, categoryController, OrderController, subCategoryController, productController, newsController,feedbackController,commentController

- Class advertiseController

Name	advertiseController						
Brief Description	Class for handling operations related to advertise						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add advertise				
update	DBS	Public	Update advertise				
delete	DBS	Public	remove advertise				

- Class orderController

Name	orderController						
Brief Description	Class for handling operations related to order						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add order				
update	DBS	Public	Update order				
delete	DBS	Public	remove order				

- Class productController

Name	productController						
Brief Description	Class for handling operations related to product						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add product				
update	DBS	Public	Update product				

delete	DBS	Public	remove product

- Class categoryController

Name	categoryController						
Brief Description	Class for handling operations related to category						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add category				
update	DBS	Public	Update category				
delete	DBS	Public	remove category				

- Class subCategoryController

Name	subCategoryController						
Brief Description	Class for handling operations related to subCategory						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add subCategory				
update	DBS	Public	Update subCategory				
delete	DBS	Public	remove subCategory				

- Class newsController

Name	newsController						
Brief Description	Class for handling operations related to news						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				

add	DBS	Public	Add news
update	DBS	Public	Update news
delete	DBS	Public	remove news

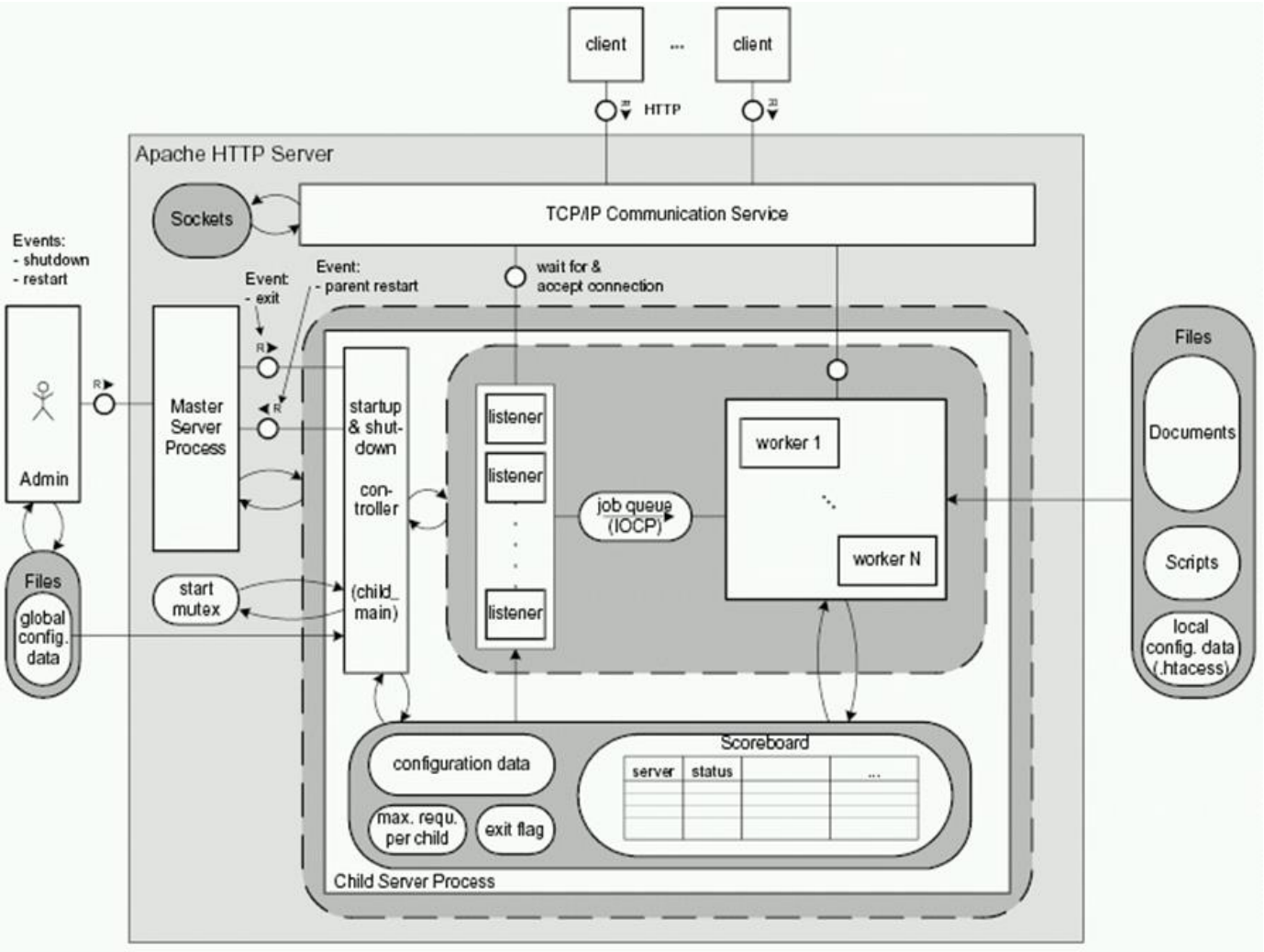
- Class feedbackController

Name	feedbackController						
Brief Description	Class for handling operations related to feedback						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add feedback				
update	DBS	Public	Update feedback				
delete	DBS	Public	remove feedback				

- Class commentController

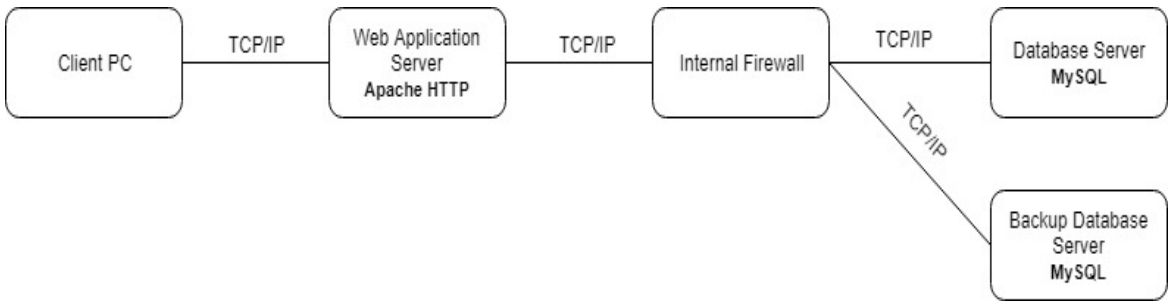
Name	commentController						
Brief Description	Class for handling operations related to comment						
Attributes							
Name	**Type**	**Access**	**Mutable**	**Optional**	**Length**	**Min**	**Max**
Operations							
Header	**Return Type**	**Access**	**Specification**				
add	DBS	Public	Add comment				
delete	DBS	Public	remove comment				

6. Process View



7. Deployment View

This section describes one or more physical network (hardware) configurations on which the warehouse management system is deployed and run. The system is comprised of these mandatory physical nodes: one firewall (internal), a web server, a database server and a backup database server. The diagram below is the simplicity version of the Warehouse Management System deployment view.



8. Implementation View

8.1 Overview

- The Implementation view depicts the physical composition of the implementation in terms of Implementation Subsystems, and Implementation Elements (directories and files, including source code, data, and executable files). Usually, the layers of the Implementation view do fit the layering defined in the Logical view

8.2 Layers

8.2.1 Presentation Layer
- The Presentation layer contains all the components needed to allow interactions with an end-user. It encompasses the user interface

8.2.2 Control Layer
- The Control layer contains all the components used to access the domain layer or directly the resource layer when this is appropriate

8.2.3 Resource Layer
- The Resource layer contains the components needed to enable communication between the business tier and the enterprise information systems (Database, external services, ERP, etc…)

8.2.4 Domain layer
- The Domain layer contains all the components related to the business logic. It gathers all the subsystems that meet the needs of a particular business domain. It also contains the business object model.

8.2.5 Common Element Layer
- The Common Element layer contains the components re-used within several layers.

9. Data View(Optional)

10. Size and Performance

The major dimensioning characteristics of the software that impact the architecture and performance constraints:
- The system shall support up to 200 orders at the time.
- The system must perform all functions with minimal time delays.
- The system must also accurately save all information transactions.

11. Quality

The system architecture supports the quality requirements:
- In order to maintain the highest degree of system integrity, the system is capable of ensuring that all information transitions are saved.
- Databases will be backed up on a daily basis in concern with safety implications.