

# **Computer Sale**

**Software Design Specification** 

# **Record of changes**

Date	A* M, D	In charge	Change Description
11 thg 5, 2024	M·	Tô Việt Hoàng	Code Packages
11 thg 5, 2024	Α -	Vũ Đặng Quang Vinh	Database Schema_Database Schema
12 thg 5, 2024	M·	Vũ Đặng Quang Vinh	Database Schema
12 thg 5, 2024	Α -	Vũ Đặng Quang Vinh	Table Description
22 thg 7, 2024	Α •	Trần Khánh Linh	Code Designs
26 thg 7, 2024	M·	Trần Khánh Linh	
27 Jul 2024	M·	Vũ Đặng Quang Vinh	OverViews,Code Designs,
27 Jul 2024	M·	Tô Việt Hoàng	

<sup>\*</sup>A - Added

M - Modified

D - Deleted

# **Table of Contents**

Record of changes	2
Table of Contents	3
Document convention	9
Basic Content Font:	9
Heading Font	9
I. Overview	10
1. Code Packages	10
1.1: Diagram	10
1.2 Package descriptions	10
2. Database Design	11
2.1. Database Schema	11
2.2. Table Description	12
II. Code Designs	15
1. Guest	15
1.1 Home	15
a. Class Diagram	15
b. Class Specifications	15
AddNews	15
CategoryDAO	
DiscountDAO	16
BrandDAO	16
ProductDAO	16
NewsDAO	
c. Sequence Diagram(s)	17
d. Database Queries	17
1.2 Product List	18
a. Class Diagram	18
b. Class Specifications	18
ProductDAO	18
MyUtils	18
c. Sequence Diagram(s)	19
d. Database Queries	19
1.3 Product Details	20
a. Class Diagram	20

b. Class Specifications	20
ProductDAO	20
DiscountDAO	20
c. Sequence Diagram(s)	21
d. Database Queries	21
1.5 Login	22
a. Class Diagram	22
b. Class Specifications	22
CustomerDAO	22
EmployeeDAO	22
c. Sequence Diagram(s)	23
d. Database Queries	23
1.6 Signup	24
a. Class Diagram	24
b. Class Specifications	24
CustomerDAO	24
EmployeeDAO	24
c. Sequence Diagram(s)	25
d. Database Queries	25
1.7 Forgot password	26
a. Class Diagram	26
b. Class Specifications	26
CustomerDAO	26
EmployeeDAO	26
MyUtils	26
EmailUtility	27
c. Sequence Diagram(s)	27
d. Database Queries	27
2. Customer	28
2.1 Profile	28
a. Class Diagram	28
b. Class Specifications	28
CustomerDAO	28
c. Sequence Diagram(s)	28
d. Database Queries	29
3. Employee	31
3.0 Employee Profile	32
a. Class Diagram	32

b. Class Specifications	32
EmployeeDAO	32
c. Sequence Diagram(s)	32
Edit Profile	32
Change Password	33
d. Database Queries	33
3.1 Staff	35
3.1.1 Add News	35
a. Class Diagram	35
b. Class Specifications	35
controller.AddNews	35
News	35
MyUtils	36
c. Sequence Diagram(s)	36
d. Database Queries	36
3.1.2 Add New Discount	38
a. Class Diagram	38
b. Class Specifications	38
controller. Add Discount. java	38
Discount	38
MyUtils	38
c. Sequence Diagram(s)	39
d. Database Queries	39
3.1.3 Add New Products	40
a. Class Diagram	40
b. Class Specifications	40
controller.ManageProducts.java	40
ProductDAO	40
Product_ImageDAO	41
CategoryDAO	41
BrandDAO	41
Employee	41
c. Sequence Diagram(s)	
d. Database Queries	42
3.1.4 Manage Product	43
a. Class Diagram	43
b. Class Specifications	
controller.ManageProducts.java	43

ProductDAO	44
Product_ImageDAO	44
CategoryDAO	44
BrandDAO	44
Employee	44
c. Sequence Diagram(s)	45
d. Database Queries	45
3.1.5. Manage Brand	46
a. Class Diagram	46
b. Class Specifications	46
controller. Manage Brand. java	46
CategoryDAO	46
MyUtils	47
c. Sequence Diagram(s)	48
d. Database Queries	48
3.1.6. Manage Product Category	49
a. Class Diagram	49
b. Class Specifications	49
controller.ManageCategory.java	49
CategoryDAO	50
MyUtils	50
c. Sequence Diagram(s)	51
d. Database Queries	52
3.1.7 Manage Orders	53
a. Class Diagram	53
b. Class Specifications	53
controller. Manage Orders. java	53
OrderDAO	53
MyUtils	53
c. Sequence Diagram(s)	54
d. Database Queries	54
3.1.8 Manage News	56
a. Class Diagram	56
b. Class Specifications	56
controller.ManageNews.java	56
NewsDAO	56
MyUtils	57
c. Sequence Diagram(s)	57

d. Database Queries	57
3.1.9 Manage News Category	58
a. Class Diagram	58
b. Class Specifications	58
controller.ManageNews.java	58
NewsDAO	58
MyUtils	58
c. Sequence Diagram(s)	59
d. Database Queries	59
3.1.10 Manage Discount	60
a. Class Diagram	60
b. Class Specifications	60
controller.ManageNews.java	60
DiscountDAO	60
MyUtils	61
c. Sequence Diagram(s)	61
d. Database Queries	62
3.2 Manager	63
3.2.1 Add new employee	63
a. Class Diagram	63
b. Class Specifications	63
controller.ManageNews.java	63
EmployeeDAO	63
c. Sequence Diagram(s)	64
d. Database Queries	64
3.2.2 Manage Feedback	65
a. Class Diagram	65
b. Class Specifications	65
controller.ManageNews.java	65
NewsDAO	65
MyUtils	65
c. Sequence Diagram(s)	66
d. Database Queries	66
3.3 Admin	67
3.3.1 View statistic	67
a. Class Diagram	67
b. Class Specifications	67
controller.ManageNews.java	67

OrderDAO	67
DAOjoin	68
c. Sequence Diagram(s)	68
d. Database Queries	69
3.3.2 Manage Account	69
a. Class Diagram	69
b. Class Specifications	69
controller.ManageNews.java	69
NewsDAO	70
MyUtils	70
c. Sequence Diagram(s)	71
d. Database Queries	71

# **Document convention**

#### **Basic Content Font:**

• Font: Calibri

• **Size:** 12pt

#### **Heading Font**

• Font: Roboto

#### • Sizes:

- Heading 1 (H1): 30pt
- o Heading 2 (H2): 24pt
- o Heading 3 (H3): 20pt
- o Heading 4 (H4): 18pt
- Heading 5 (H5): 16pt
- Heading 6 (H6): 14pt

#### • Align

- o Heading 1 (H1): 0
- o Heading 2 (H2): 0
- o Heading 3 (H3): 0
- o Heading 4 (H4): 1
- o Heading 5 (H5): 1.5

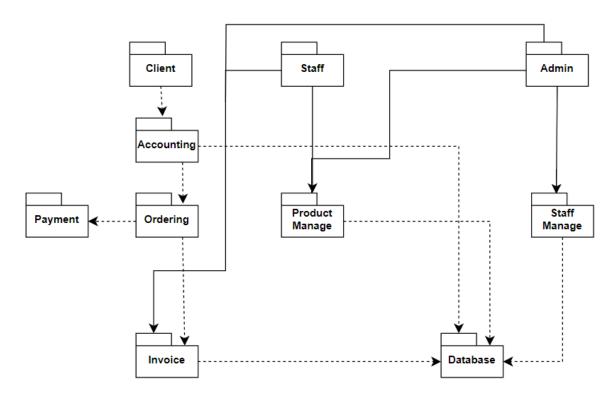
#### Heading 6 (H6):

# I. Overview

# 1. Code Packages

# 1.1: Diagram

<u>UML</u>

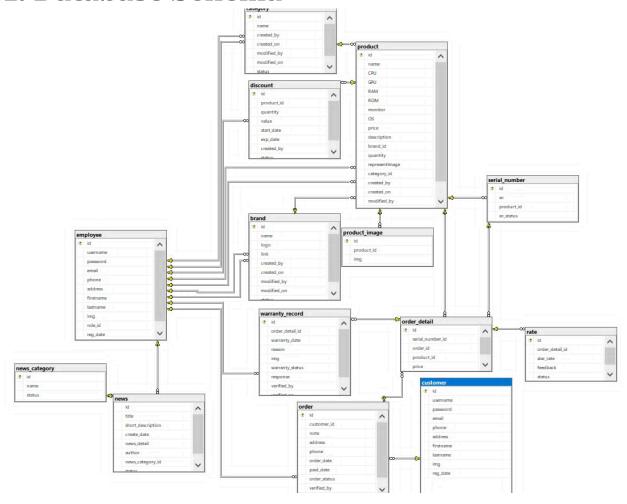


# 1.2 Package descriptions

No	Package	Description	
1.	Admin	The highest role of manage system	
2.	Staff	The Employees who manage the detail of products	
3.	Client	The customer, buyer, viewer	
4.	Accounting	About login or register account Login	
5.	Ordering	A product or a list of products which are ordered	
6.	Payment	The way to pay for invoice	
7.	Invoice	Including information of products are bought and the detail of warranty	
8.	Product Manage	Can CRUD the product	
9.	Staff Manage	Can CRUD the staff	
10.	Database	Saving system data	

# 2. Database Design

### 2.1. Database Schema



# 2.2. Table Description

No	Table		Description	
1.	customer	id	int identity(1,1)	primary key
		username	varchar(50)	. , ,
		password	varchar(50)	
		email	varchar(100)	
		phone	varchar(20)	
		address	nvarchar(100)	
		firstname	nvarchar(50)	
		lastname	nvarchar(50)	
		img	nvarchar(100)	
		reg_date	date	
2.	employee	id	int identity(1,1)	primary key
		username	varchar(50)	
		password	varchar(50)	
		email	varchar(100)	
		phone	varchar(20)	
		address	nvarchar(100)	
		firstname	nvarchar(50)	
		lastname	nvarchar(50)	
		img	nvarchar(100)	
		role_id	int	
		reg_date	date	
		status	tinyint	
3.	category	id	int identity(1,1)	primary key
		name	nvarchar(255)	
		created_by	int	reference key
		created_on	date	
		modified_by	int	reference key
		modified_on	date	
		status	tinyint	
4.	brand	id	int identity(1,1)	primary key
		name	nvarchar(255)	
		logo	varchar(100)	
		link	varchar(100)	
		created_by	int	reference key
		created_on	date	
		modified_by	int	reference key
		modified_on	date	
		status	int	
5.	product	id	int identity(1,1)	primary key
		name	nvarchar(255)	
		CPU	varchar(100)	
		GPU	varchar(100)	

		T D A A A	1 (100)	
		RAM	varchar(100)	
		ROM	varchar(100)	
		monitor	varchar(100)	
		OS	varchar(100)	
		price	int	
		[description]	ntext	
		brand_id	int	reference key
		quantity	int	
		img	varchar(100)	
		category_id	int	reference key
		created_by	int	reference key
		created_on	date	
		modified by	int	reference key
		modified on	date	
		status	int	
6.	product image	id	int	primary key
		product_id	int	reference key
		img	varchar(100)	
7.	discount	id	int identity(1,1)	primary key
′′	discourre	product id	int	reference key
		quantity	int	Tereference key
		value	tinyint	
		exp_date	date	
		created by	int	reference key
		created on	date	Telefeliee key
		status	int	
8.	serial number	id	int identity(1,1)	primary key
0.	Serial_Hullibel		varchar(100)	primary key
		sn product id	int	reference key
		product_id		Telefelice key
	[ordor <sup>1</sup>	sn_status	tinyint	ngina agai leas
9.	[order]	id	int identity(1,1)	primary key
		customer_id	int	reference key
		note	nvarchar(255)	
		[address]	nvarchar(255)	
		phone	varchar(20)	
		order_date	date	
		paid_date	date	
		order_status	tinyint	
		verified_by	int	reference key
		verified_on	date	
10.	order_detail	id	int identity(1,1)	primary key
		serial_number_id	int	reference key
		order_id	int	reference key
		product_id	int	reference key
1		price	int	

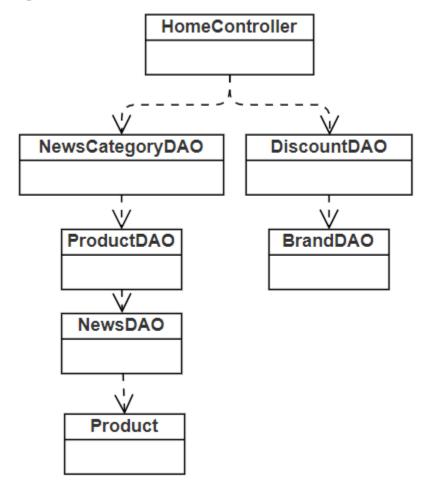
11.	rate	id order_detail_id star_rate feedback	int identity(1,1) int tinyint nvarchar(255)	primary key reference key
		status	int	
12.	warranty_record	id order_detail_id warranty_date reason img warranty_status response verified_by	int identity(1,1) int date nvarchar(255) varchar(255) tinyint nvarchar(255) int	primary key reference key reference key
		verified_on	date	
13.	news_category	id [name] status	int identity(1,1) varchar(100) int	primary key
14.	news	id title short_description create_date news_detail author news_category_id status	int identity(1,1) nvarchar(100) nvarchar(255) date text int int int	reference key reference key

# II. Code Designs

## 1. Guest

#### **1.1** *Home*

a. Class Diagram



### b. Class Specifications

#### AddNews

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the Home . jsp view.

request, HttpServletResponse	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.
response)	

CategoryDAO

No	Method	Description
1	getAll()	Retrieves all Category categories.

### DiscountDAO

No	Method	Description
1	getDiscountProductID()	Retrieves a map of discount IDs and their corresponding values for active discounts.

#### BrandDAO

No	Method	Description
1	getAll()	Retrieves all Brand categories.

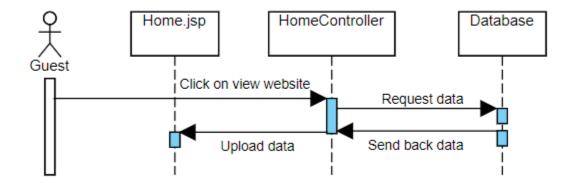
### ProductDAO

No	Method	Description
1	getHomeProduct()	Retrieves a list of products that have the top 8 highest discount values and are currently active.

### NewsDAO

No	Method	Description
1	getHomeNews()	Retrieves a list of the top 3 news articles in a specific category (category ID 1), ordered by their creation date.

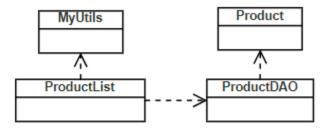
#### c. Sequence Diagram(s)



- getAll()
  - select \* from category
- getDiscountProductID()
  - SELECT \* FROM discount where (start\_date < ? or start\_date = ?) and exp\_date > ?
- getAll()
  - select \* from brand
- getHomeProduct()
  - SELECT \* FROM product \n"
  - + "WHERE id IN (\n"
  - + "SELECT TOP 8 product id FROM discount\n"
  - + "WHERE (start\_date < ? OR start\_date = ? ) AND exp\_date > ?\n"
  - + "GROUP BY product\_id,value,start\_date\n"
  - + "ORDER BY [value] )
- getHomeNews()
  - SELECT top 3 \* FROM news where news\_category\_id =1 order by create\_date

### 1.2 Product List

### a. Class Diagram



### b. Class Specifications

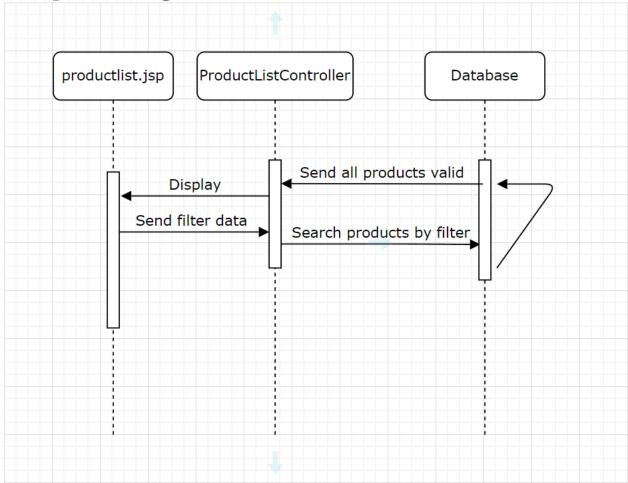
#### ProductDAO

No	Method	Description
1	getAvailable(String extend)	Retrieves a list of available products with an optional filter condition.
2	getAllByFilter(String filter)	Retrieves a list of available products filtered by the given condition.

### MyUtils

No	Method	Description
1	getArrayListByPaging(ArrayLi st <t> list, int pageNumber, int itemsPerPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.

### c. Sequence Diagram(s)



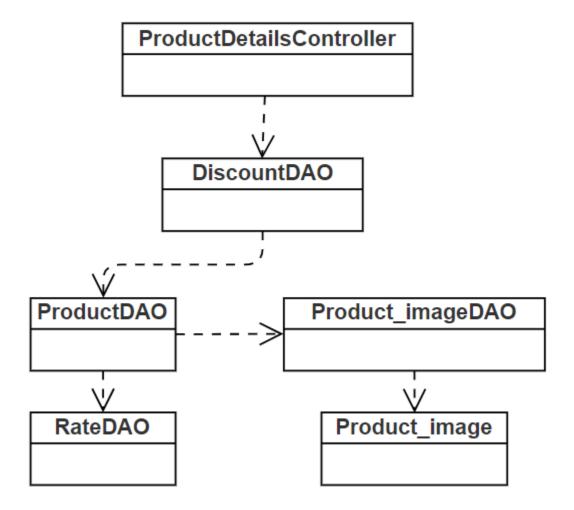
### d. Database Queries

- getAvailable(String extend)
  - select \* from product where status=1

•

### 1.3 Product Details

### a. Class Diagram



### b. Class Specifications

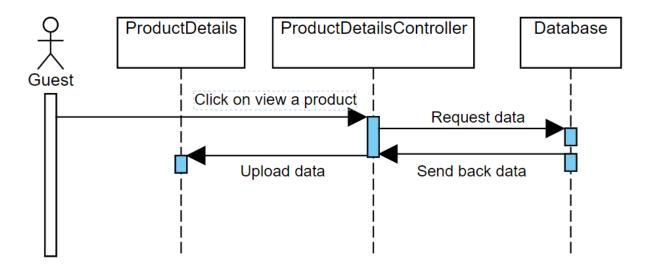
#### ProductDAO

	ToddotBilo	
No	Method	Description
1	getById(int id)	Retrieves a product by its ID.
2	getRelatedProducts(int id)	Retrieves a list of up to 8 products related to the product with the specified ID, based on the same brand.

#### **DiscountDAO**

No	Method	Description
1	getArrayListByPaging(ArrayLi st <t> list, int pageNumber, int itemsPerPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.

### c. Sequence Diagram(s)

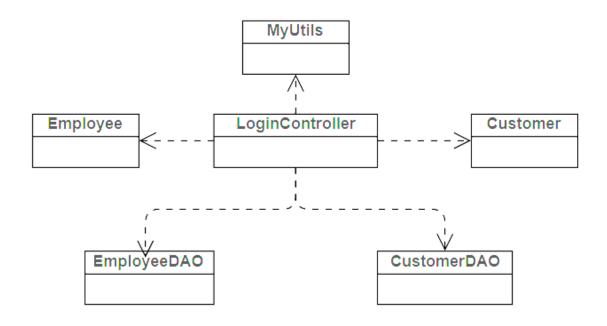


- getById(int id)
  - o select \* from product where id =
- getRelatedProducts(int id)

```
    String sql = "SELECT top 8 p.* \n"
    + "FROM product p\n"
    + "JOIN product p1 ON p.brand_id = p1.brand_id\n"
    + "WHERE p1.id = ?;
```

## 1.5 Login

### a. Class Diagram



### b. Class Specifications

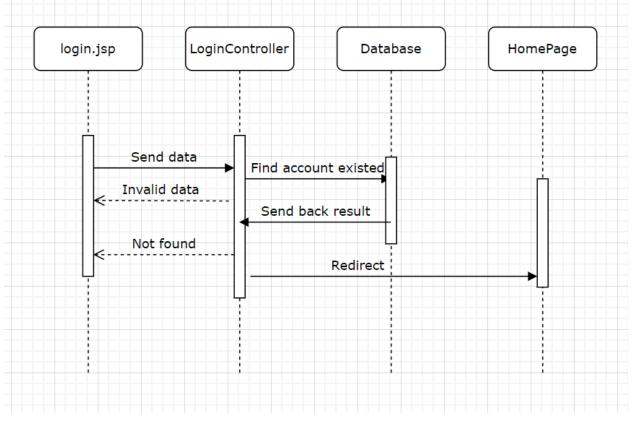
#### CustomerDAO

No	Method	Description
1	checkLogin(String username, String password)	Checks if a customer with the provided username and password exists in the database and returns the corresponding Customer object if found.

EmployeeDAO

No	Method	Description
1	checkLogin(String username, String password)	Checks if a customer with the provided username and password exists in the database and returns the corresponding Employee object if found.

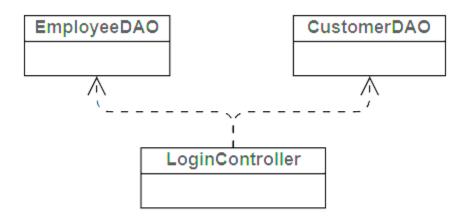
#### c. Sequence Diagram(s)



- checkLogin(String username, String password)
  - select \* from customer where username like "" + username + "" and password like
     "" + MyUtils.getMd5(password) + ""
- checkLogin(String username, String password)
  - select \* from employee where username like '" + username + "' and password like '" + MyUtils.getMd5(password) + "'

# 1.6 Signup

### a. Class Diagram



### b. Class Specifications

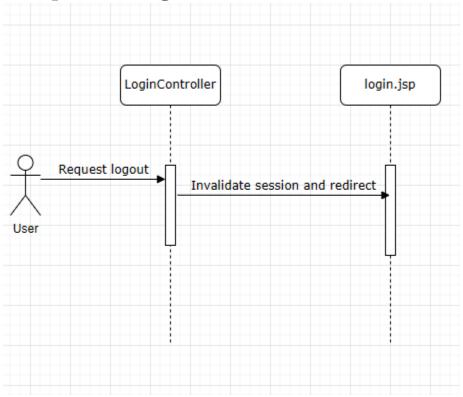
#### CustomerDAO

No	Method	Description
1	checkSignUp(String username, String email)	Checks if a username or email is already in use by another customer

**EmployeeDAO** 

No	Method	Description
1	checkSignUp(String username, String email)	Checks if a username or email is already in use by another customer

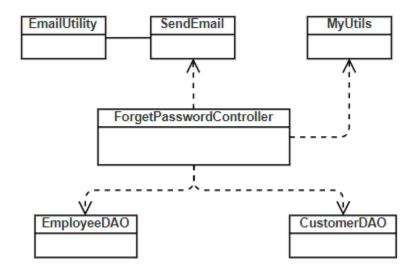
### c. Sequence Diagram(s)



- checkSignUp(String username, String email)
  - select \* from customer where username like "" + username + "" or email like "" + email + ""
- checkSignUp(String username, String email)
  - select \* from employee where username like '" + username + "' or email like '" + email + "'

# 1.7 Forgot password

### a. Class Diagram



### b. Class Specifications

#### CustomerDAO

No	Method	Description
1	getByEmail(email)	Checks if a username or email is already in use by another customer
2	changePassword(int id, String newPassword)	Updates the password for a customer with the specified ID

**EmployeeDAO** 

No	Method	Description
1	getByEmail(email)	Checks if a username or email is already in use by another customer
2	changePassword(int id, String newPassword)	Updates the password for a employee with the specified ID

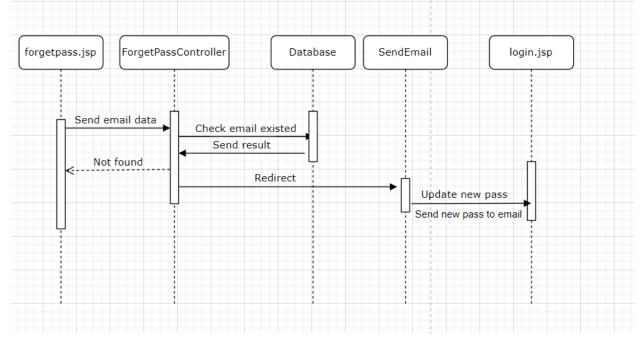
**MyUtils** 

No	Method	Description
1	genCode(length)	Gen a random string which have the length given

**EmailUtility** 

No	Method	Description
1	sendMail(to,subject,body,co ntent)	Using library to send an email

### c. Sequence Diagram(s)

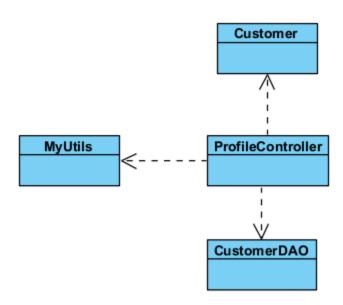


- getByEmail(email)
  - elect \* from customer where email like '" + email + "'
- changePassword(int id, String newPassword)
  - O UPDATE [dbo].[customer] SET [password] = ? WHERE [id] = ?

# 2. Customer

### 2.1 Profile

## a. Class Diagram



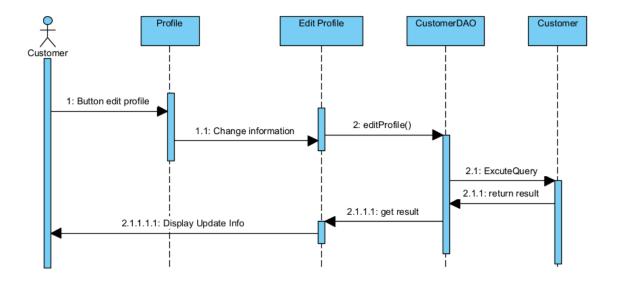
### b. Class Specifications

#### CustomerDAO

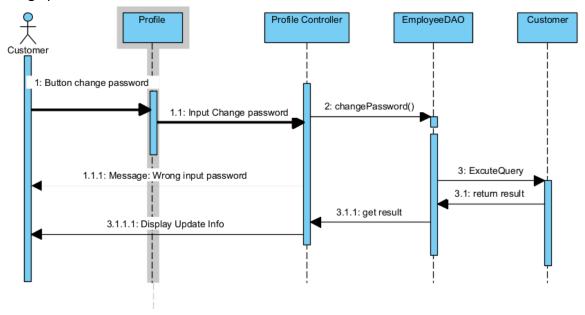
	0.000011012110	
No	Method	Description
1	editProfile()	Checks if a username or email is already in use by another customer
2	checkPassWord(String oldPassword, int id)	Check the password of the selected id

## c. Sequence Diagram(s)

Edit profile



#### Change password



### d. Database Queries

editProfile()

```
    UPDATE [dbo].[customer]\n"
    +" SET [email] = ?\n"
    +" ,[phone] = ?\n"
    +" ,[address] = ?\n"
    +" ,[firstname] = ?\n"
    +" ,[lastname] = ?\n"
    +" ,[img] = ?\n"
    +" WHERE [id] = ?
```

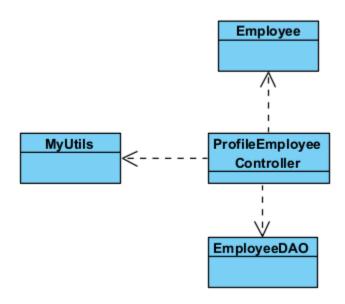
checkPassWord(String oldPassword, int id)

○ SELECT password FROM [dbo].[customer] WHERE id = ?

# 3. Employee

# 3.0 Employee Profile

## a. Class Diagram



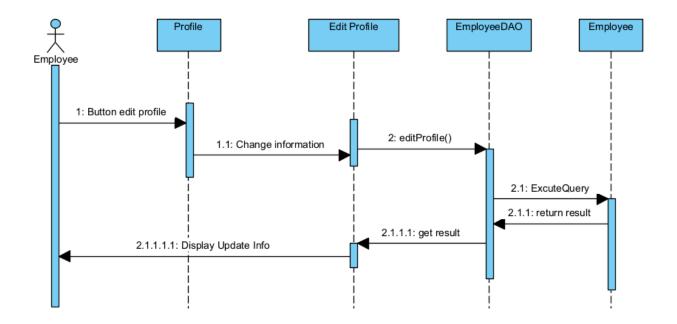
# b. Class Specifications

**EmployeeDAO** 

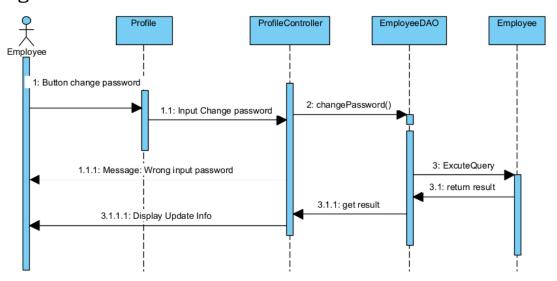
No	Method	Description
1	editProfile()	Checks if a username or email is already in use by another customer
2	checkPassWord(String oldPassword, int id)	Check the password of the selected id

# c. Sequence Diagram(s)

#### **Edit Profile**



#### **Change Password**



### d. Database Queries

editProfile()

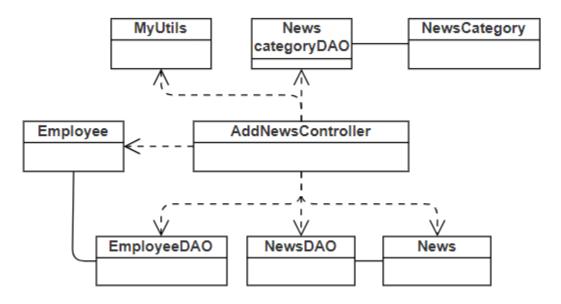
```
    UPDATE [dbo].[employee]\n"
    +" SET [email] = ?\n"
    +" ,[phone] = ?\n"
    +" ,[address] = ?\n"
    +" ,[firstname] = ?\n"
    +" ,[lastname] = ?\n"
    +" ,[img] = ?\n"
```

- + " WHERE [id] = ?
- checkPassWord(String oldPassword, int id)
  - SELECT password FROM [dbo].[employee] WHERE id = ?

### **3.1 Staff**

### 3.1.1 Add News

### a. Class Diagram



### b. Class Specifications

#### controller.AddNews

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddNews . j sp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### News

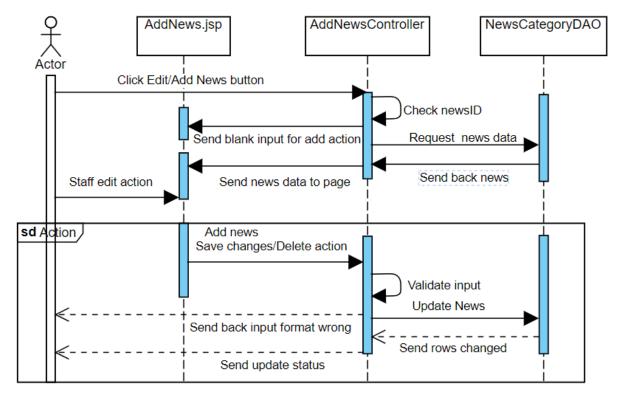
No	Method	Description
1	getManageNews()	Retrieves all news entries from the database, ordered by ID and creation date in descending order.
2	searchByNewString(String search)	Searches for news entries where the title or author's name matches the provided search string.

3	searchByNewsCategory(int	Searches for news entries that belong to a specific news
	search)	category ID.

#### **MyUtils**

No	Method	Description
1	validateParameters(String params)	throws an IllegalArgumentException if any of the parameters are null, ensuring that all required parameters are provided.
2	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.

### c. Sequence Diagram(s)

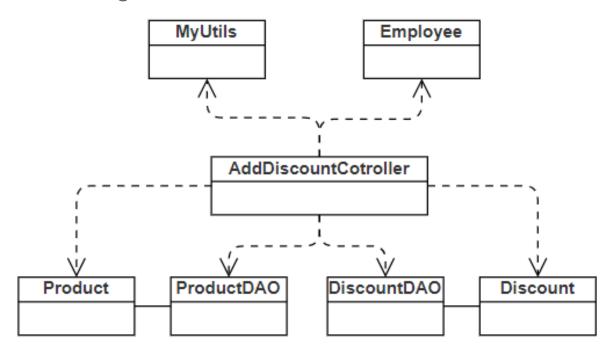


- 1, getManageNews()
  - SELECT \* FROM news ORDER BY id desc, create\_date DESC
- 2,searchByNewString(String search)
  - SELECT \* FROM news where title LIKE '%" + search + "%' or author = (select id from employee where username like '%" + search + "%' or firstname like '%" + search + "%'

- searchByNewsCategory(int search)
  - SELECT \* FROM news where news\_category\_id = " + search

### 3.1.2 Add New Discount

### a. Class Diagram



# b. Class Specifications

controller.AddDiscount.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

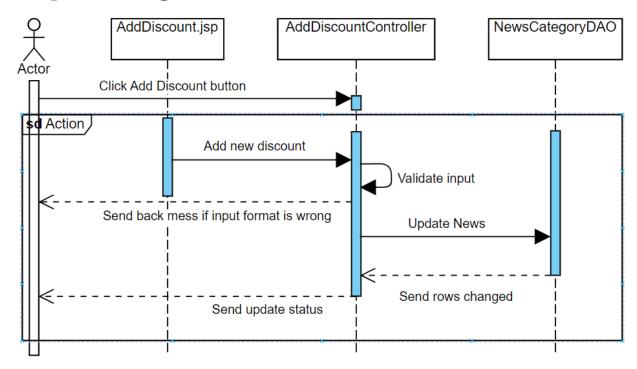
#### Discount

No	Method	Description
1	addDiscount(Discount discount)	Inserts a new discount record into the database

**MyUtils** 

	$\mathcal{J}$	
No	Method	Description

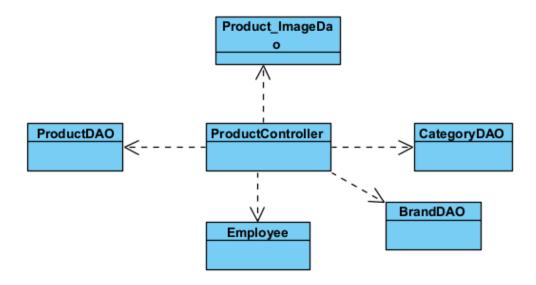
1	validateParameters(String params)	throws an IllegalArgumentException if any of the parameters are null, ensuring that all required parameters are provided.
2	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.



- addDiscount(Discount discount)
  - INSERT INTO discount (product\_id, quantity, value, start\_date, exp\_date, created\_by) VALUES (?, ?, ?, ?, ?)

### 3.1.3 Add New Products

### a. Class Diagram



### b. Class Specifications

controller.ManageProducts.java

	TOTTOTIMATIA GOT TO GAOCO	· j · · · · ·
No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news product then forwards to the AddProduct . j sp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or hidden/display products based on the provided action.

#### **ProductDAO**

No	Method	Description
1	addProduct()	Inserts a new Product record into the database
2	getLastInsertedProductId()	Retrieves the ID of the most recently inserted product from the product table.

# Product\_ImageDAO

No	Method	Description
1	addProductImage()	Inserts a image name with customer id into database
2	getAll()	Get all image from product_image table

### CategoryDAO

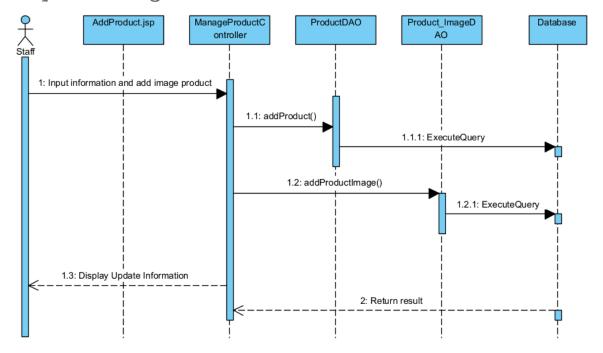
No	Method	Description
1	getAll()	Get all category from category table

### BrandDAO

No	Method	Description
1	getAll()	Get all category from brand table

# Employee

No	Method	Description
1	(Employee) session.getAttribute("currentE mployee")	Set the current id of Employee as soon as the user login as Employee roles



### d. Database Queries

addProduct()

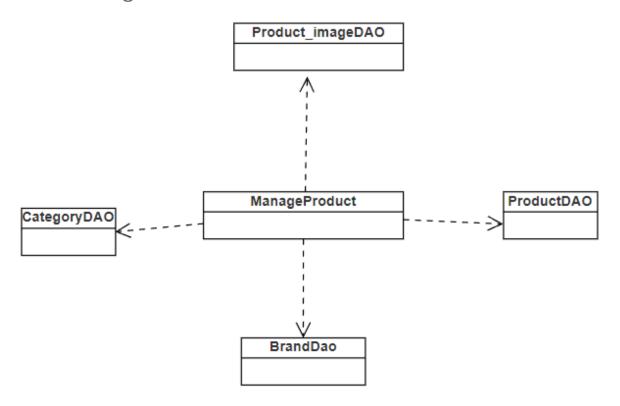
```
INSERT INTO [dbo].[product]\n"
0
                       ([name]\n"
                       ,[CPU]\n"
0
                       ,[GPU]\n"
0
                       ,[RAM]\n"
                       ,[ROM]\n"
0
                       ,[monitor]\n"
                       ,[OS]\n"
0
                       ,[price]\n"
                       ,[quantity]\n"
0
                       ,[representImage]\n"
                       ,[description]\n"
0
                       ,[brand id]\n"
                       ,[category_id]\n"
0
                       ,[created_by]\n"
0
                       ,[created on])\n"
                    VALUES (?,?,?,?,?,?,?,?,?,?,?,?)
```

- getLastInsertedProductId()
   "SELECT MAX(id) AS id FROM product"
- getAllByProductID()"select \* from product image where product id = ?"

addProductImage(int productId, String imageName)
 "INSERT INTO [dbo].[product\_image] (product\_id, img) VALUES (?, ?)"

# 3.1.4 Manage Product

### a. Class Diagram



### b. Class Specifications

controller.ManageProducts.java

	ioner.manager roadets	y = 1 = 1
No	Method	Description
1	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news product then forwards to the AddProduct.jsp view.
2	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or hidden/display products based on the provided action.

ProductDAO

No	Method	Description
1	editProductById()	Change product properties into data base
2	getLastInsertedProductId()	Retrieves the ID of the most recently inserted product from the product table.
3	hiddenProduct()	Change product status into hidden
4	displayProduct	Display product which is hidden

Product\_ImageDAO

No	Method	Description
1	addProductImage()	Inserts a image name with customer id into database
2	getAll()	Get all image from product_image table

### CategoryDAO

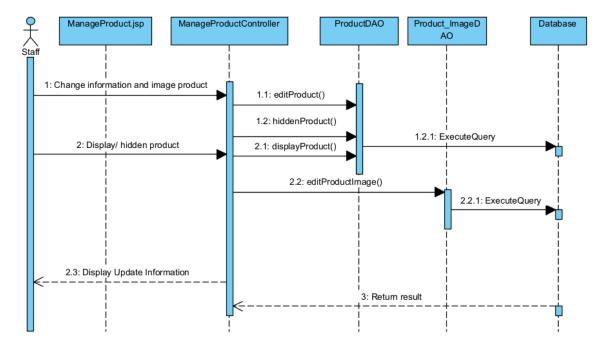
No	Method	Description
1	getAll()	Get all category from category table

#### **BrandDAO**

No	Method	Description
1	getAll()	Get all category from brand table

### Employee

No	Method	Description
1	(Employee) session.getAttribute("currentE mployee")	Set the current id of Employee as soon as the user login as Employee roles



### d. Database Queries

editProductById()

```
"UPDATE [dbo].[product]\n"
          + " SET [name] = ?\n"
                [CPU] = n
                ,[GPU] = ?\n"
                [RAM] = ?\n"
                [ROM] = n
          + "
                ,[monitor] = ?\n"
          + "
                ,[OS] = ?\n"
          + "
                ,[price] = ?\n"
                ,[description] = ?\n"
          + "
                [brand_id] = ?\n"
                ,[category id] = ?\n"
          + "
                ,[quantity]= ?\n"
          + "
                ,[representImage]= ?\n"
                [modified_by] = ?\n"
                ,[modified on] = ?\n"
          + " WHERE [id] = ?"
```

hiddenProduct()

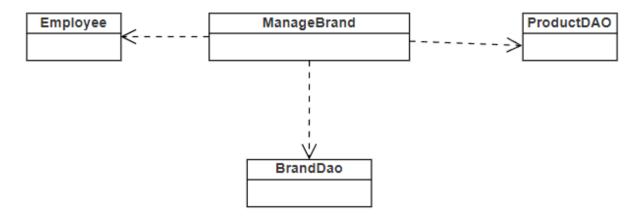
```
"UPDATE [dbo].[product]\n"
+ " SET [status] = 0\n"
+ " WHERE [id] = ?"
displayProduct()
```

"UPDATE [dbo].[product] SET [status] = 1 WHERE [id] = (SELECT p.id \n" +

- " FROM product p \n" +
- " JOIN brand b ON b.id=p.brand id\n" +
- Join category c on c.id=p.category\_id\n" +
- " WHERE b.status = 1 and c.status=1 and p.id=?)"
- getAllProductByCategory()"select \* from product where category = ?"

### 3.1.5. Manage Brand

### a. Class Diagram



### b. Class Specifications

controller.ManageBrand.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the ManageCategory . jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

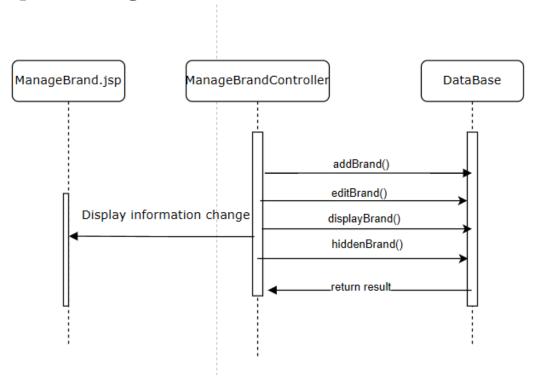
CategoryDAO

No	Method	Description
1	getAll()	This method retrieves all brands from the database by

		calling the get method with null as the parameter
2	addBrand(String name, String logo, String link, int created_by)	This method adds a new brand to the database. It constructs and executes an SQL INSERT statement with the given brand name, logo, link, and creator ID.
3	editBrandById(String name, String logo, String link, int modified_by, int id)	This method updates an existing brand in the database. It constructs and executes an SQL UPDATE statement with the new brand name, logo, link, modifier ID, and brand ID.
4	hiddenBrand(int id)	This method hides a brand and all associated products in the database by setting their status to 0. It uses a transaction to ensure both updates are performed atomically.
5	displayBrand(int id)	This method makes a brand and all associated products with active categories visible in the database by setting their status to 1. It uses a transaction to ensure both updates are performed atomically.

### MyUtils

No	Method	Description
1	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
2	getArrayListByPaging(ArrayLis t <t> list, int pageNumber, int ItemsOfPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.



#### d. Database Queries

```
addBrand()
```

"INSERT INTO [dbo].[brand] "

+ "([name], [logo], [link], [created\_by], [created\_on]) "

+ "VALUES (?, ?, ?, ?, ?)"

#### editBrandById()

"UPDATE [dbo].[brand]\n"

+ " ,[modified\_by] = ?\n"

+ " ,[modified\_on] =  $?\n$ "

+ " WHERE [id] = ?"

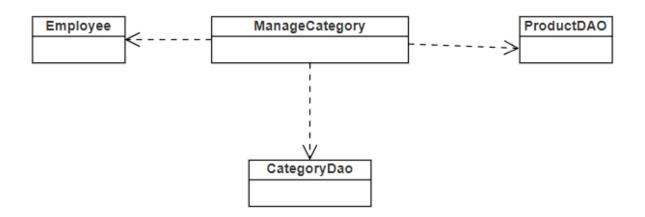
#### hiddenBrand()

"UPDATE [dbo].[brand]\n"

"UPDATE [dbo].[product]\n"

+ " WHERE [brand\_id] = ?"

# 3.1.6. Manage Product Category a. Class Diagram



### b. Class Specifications

controller.ManageCategory.java

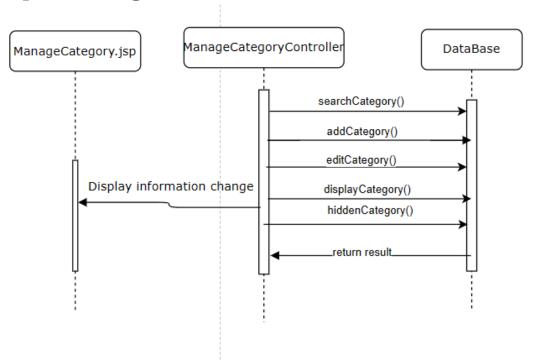
No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the ManageCategory . jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

CategoryDAO

No	Method	Description
1	getAll()	This method retrieves all categories from the database by calling the get method with null as the parameter.
2	addCategory(String name, int created_by)	This method adds a new category to the database. It constructs and executes an SQL INSERT statement with the given category name and creator ID.
3	editCategory(String name, int modified_by, int id)	This method updates an existing category in the database. It constructs and executes an SQL UPDATE statement with the new category name, modifier ID, and category ID.
4	hiddenCategory(int id)	This method hides a category and all associated products in the database by setting their status to 0. It uses a transaction to ensure both updates are performed atomically.
5	displayCategory(int id)	This method makes a category and all associated products with active brands visible in the database by setting their status to 1. It uses a transaction to ensure both updates are performed atomically.

### <u>MyUtils</u>

No	Method	Description
1	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
2	getArrayListByPaging(ArrayLis t <t> list, int pageNumber, int ItemsOfPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.

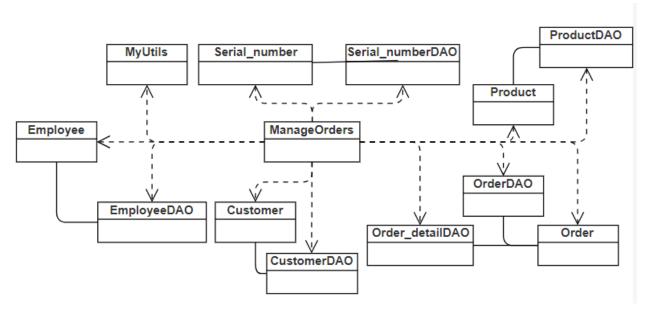


#### d. Database Queries

addCategory() "INSERT INTO [dbo].[category] " + "([name], [created\_by], [created\_on]) " + "VALUES (?, ?, ?)" editCategory() "UPDATE [dbo].[category]\n" + " SET [name] = ?\n" ,[modified by] =  $?\n$ "  $[modified_on] = n$ + " WHERE [id] = ?" displayCategory() "UPDATE [dbo].[category]\n" + " SET [status] = 1\n" + " WHERE [id] = ?" """UPDATE product SET status = 1 WHERE id IN ( SELECT p.id FROM product p JOIN brand b ON b.id=p.brand\_id WHERE b.status = 1 ) and [category\_id] = ?""" hiddenCategory() "UPDATE [dbo].[category]\n" + " SET [status] = 0\n" + " WHERE [id] = ?" "UPDATE [dbo].[product]\n" + " SET [status] = 0\n" + " WHERE [category id] = ?"

# 3.1.7 Manage Orders

### a. Class Diagram



### b. Class Specifications

controller.ManageOrders.java

	oner on or manage or dors if a va		
No	Method	Description	
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.	
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.	

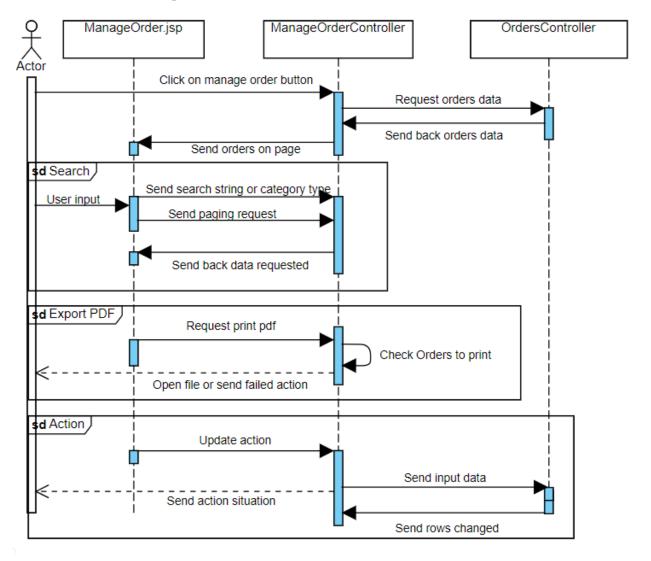
#### OrderDAO

No	Method	Description
1	getOrders()	Retrieves orders that have been paid and have order details.
2	searchByOrderStatus(int order_status)	Retrieves orders with a specific status.

#### **MyUtils**

No	Method	Description
----	--------	-------------

1	removeDiacritics(String input)	replaces characters with diacritical marks with their non-diacritical equivalents based on a predefined map
2	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
3	getArrayListByPaging(ArrayLis t <t> list, int pageNumber, int ItemsOfPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.

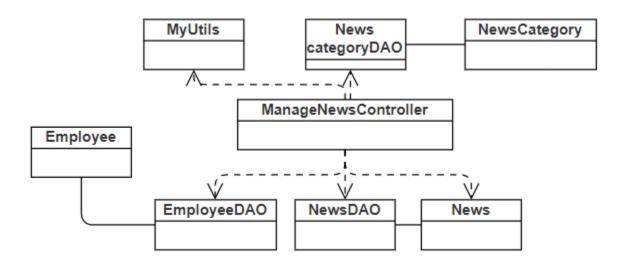


- getOrders()
  - SELECT \* FROM [order] where paid\_date is not null and id in (select od.order\_id from order\_detail od) order by id desc

- searchByOrderStatus(int order\_status)
   SELECT \* FROM [order] where order\_status = " + order\_status

# 3.1.8 Manage News

### a. Class Diagram



### b. Class Specifications

controller.ManageNews.java

	<u> </u>	
No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### NewsDAO

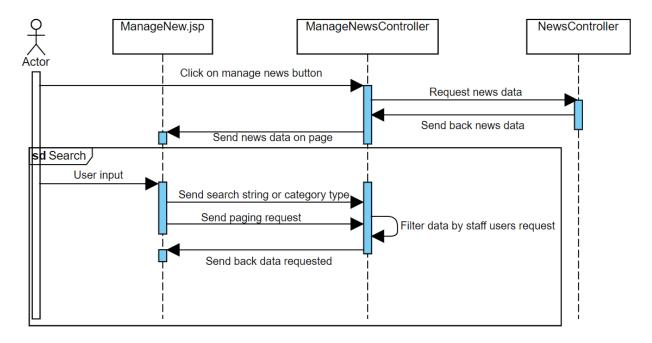
	101102110	
No	Method	Description
1	getManageNews()	Retrieves all news sorted by ID and creation date in descending order.
2	searchByNewString(String search)	Searches for news items by a search string in the title or author's name.
3	searchByNewsCategory(int search)	Searches for news items by a specific news category.

1	
1	
1	

#### MyUtils

No	Method	Description
1	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
2	getArrayListByPaging(ArrayLis t <t> list, int pageNumber, int ItemsOfPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.

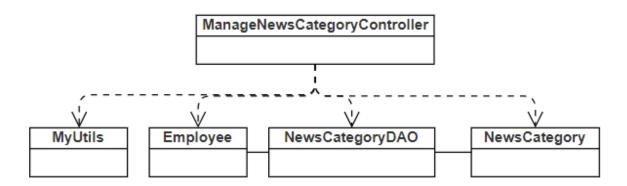
#### c. Sequence Diagram(s)



- getManageNews()
  - SELECT \* FROM news ORDER BY id desc, create date DESC
- searchByNewString(String search)
  - SELECT \* FROM news where title LIKE '%" + search + "%' or author = (select id from employee where username like '%" + search + "%' or firstname like '%" + search + "%' or lastname like '%" + search + "%')
- searchByNewsCategory(int search)
  - SELECT \* FROM news where news\_category\_id = " + search

# 3.1.9 Manage News Category

### a. Class Diagram



### b. Class Specifications

controller.ManageNews.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### **NewsDAO**

No	Method	Description
1	searchNewCategoryString(Stri ng search)	Searches for news categories by a search string in their name.
2	getAll()	Retrieves all news categories.

*MyUtils* 

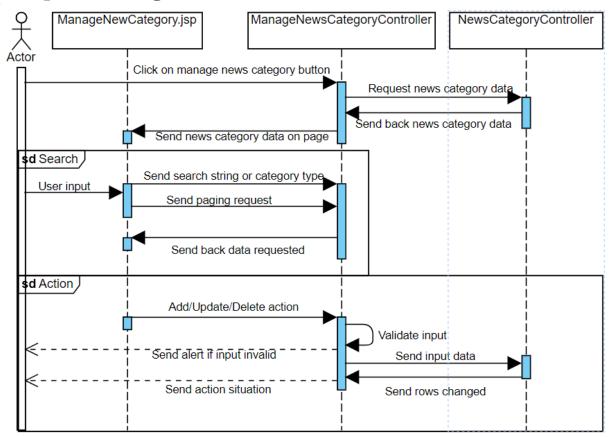
No	Method	Description
1	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.

getArrayListByPaging(ArrayLis t<T> list, int pageNumber, int ItemsOfPage)

2

calculates the start and end indices for the desired page and returns the corresponding sublist.

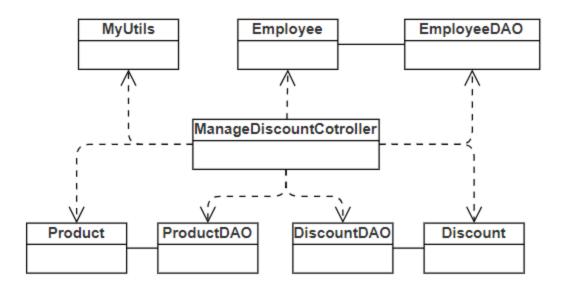
#### c. Sequence Diagram(s)



- getAll()
  - select \* from news category
- searchNewCategoryString(String search)
  - select \* from news category where name like '%" + search + "%'

# 3.1.10 Manage Discount

### a. Class Diagram



### b. Class Specifications

controller.ManageNews.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### DiscountDAO

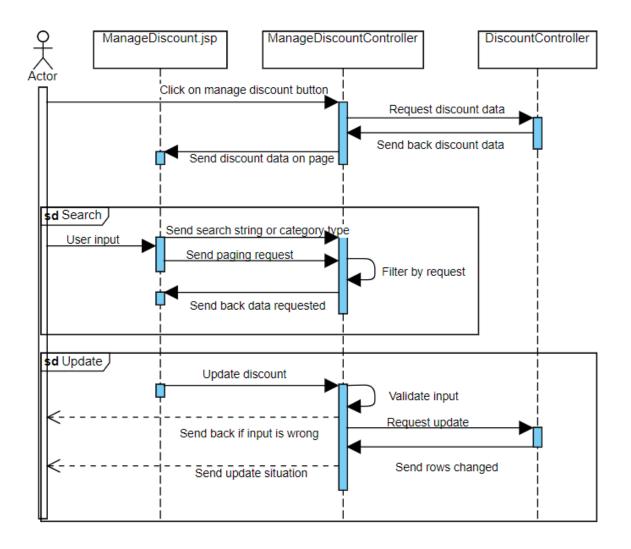
No	Method	Description
1	updateDiscount(Discount discount)	Updates an existing discount record in the database.
2	addDiscount(Discount discount)	Adds a new discount record to the database.

3	deleteDiscount(int discountID)	Deletes a discount record from the database based on
		its ID.

#### MyUtils

No	Method	Description
1	setAlertAttributes(HttpServlet Request request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
2	validateParameters(String params)	throws an IllegalArgumentException if any of the parameters are null, ensuring that all required parameters are provided.

### c. Sequence Diagram(s)



- updateDiscount(Discount discount)
  - UPDATE [dbo].[discount]\n"

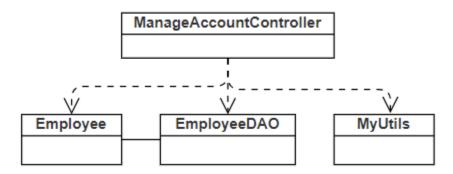
```
+ " SET [product_id] = ?\n"
```

- o + " ,[quantity] = ?\n"
- o + " ,[value] = ?\n"
- o + " ,[start\_date] = ?\n"
- o + " ,[exp\_date] = ?\n"
- o + " ,[created\_by] = ?\n"
- o + " WHERE [id] = ?
- addDiscount(Discount discount)
  - INSERT INTO discount (product\_id, quantity, value, start\_date, exp\_date, created\_by) VALUES (?, ?, ?, ?, ?)
- deleteDiscount(int discountID)
  - DELETE FROM [dbo].[discount] WHERE [id] = ?

# 3.2 Manager

# 3.2.1 Add new employee

### a. Class Diagram



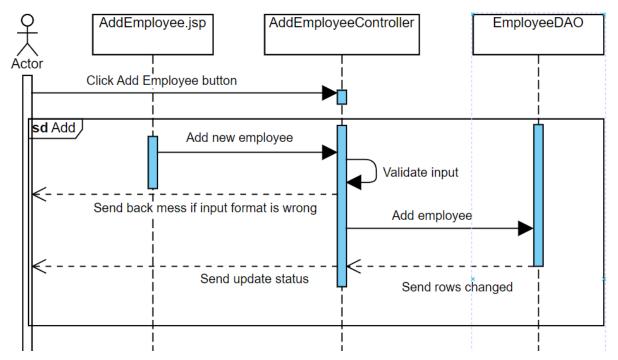
### b. Class Specifications

controller.ManageNews.java

	<u> </u>	
No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

**EmployeeDAO** 

	mp10 y 00 20 10		
No	Method	Description	
1	checkSignUp(String username, String email)	Checks if a username or email already exists in the database	
2	addEmployee(Employee e)	Adds a new employee to the database.	



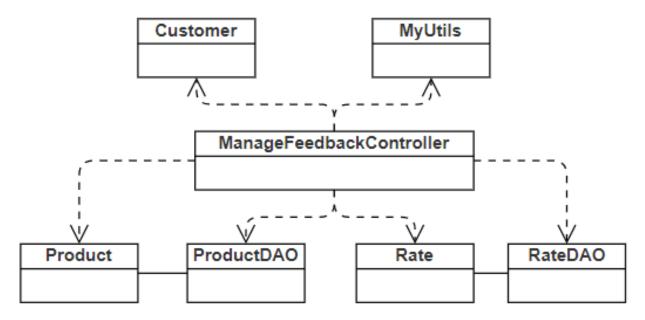
### d. Database Queries

• addEmployee(Employee e)

```
INSERT INTO [dbo].[employee]\n"
                      ([username]\n"
0
                      ,[password]\n"
0
                      ,[email]\n"
0
                      ,[phone]\n"
0
                      ,[address]\n"
0
                      ,[firstname]\n"
0
                      ,[lastname]\n"
0
0
                      ,[reg_date]\n"
                      ,[role_id])\n"
              + "
                   VALUES\n"
0
                      (?,?,?,?,?,?,?)
```

# 3.2.2 Manage Feedback

### a. Class Diagram



### b. Class Specifications

controller.ManageNews.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### **NewsDAO**

No	Method	Description
1	deleteRateByld(int rateId)	Deletes a rate record from the database based on the given rate ID.

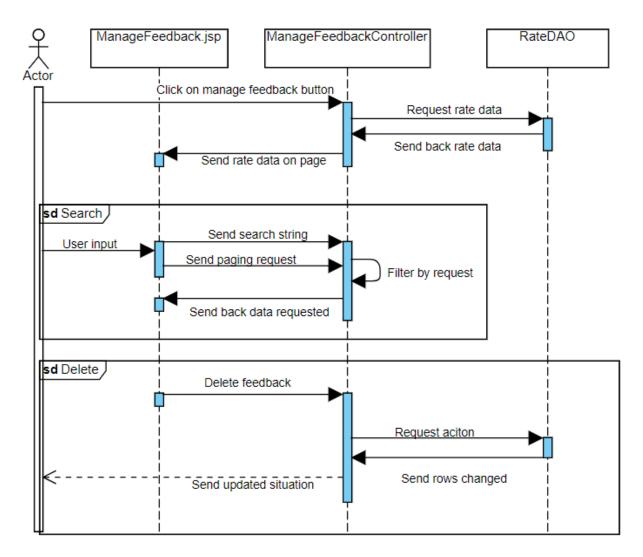
MyUtils

No	Method	Description
1	setAlertAttributes(HttpServlet	sets an alert message and type based on the success or

Request request, boolean status, String action)

failure of an action, which can be used for displaying user feedback in a web application.

### c. Sequence Diagram(s)

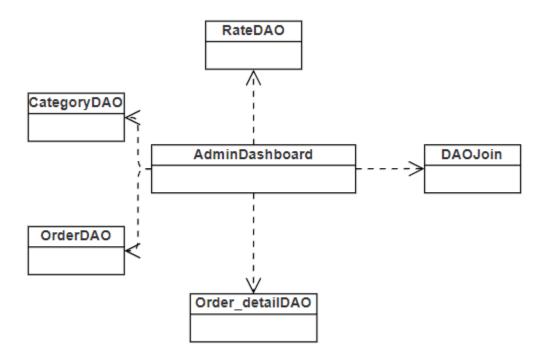


- deleteRateById(int rateId)
  - O DELETE FROM rate WHERE id = ?

# 3.3 Admin

# 3.3.1 View statistic

### a. Class Diagram



### b. Class Specifications

controller.ManageNews.java

No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

#### OrderDAO

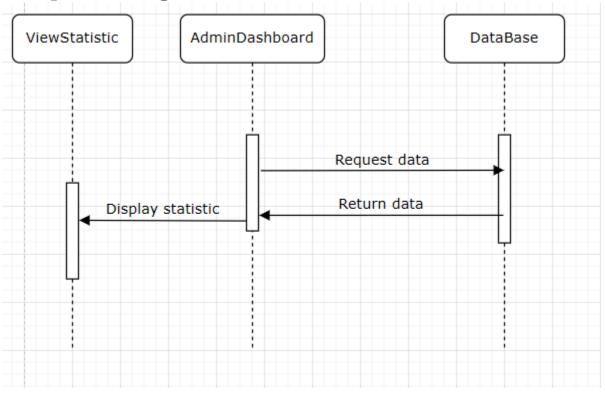
No	Method	Description

1	getTotalSales()	Retrieves the total sales amount from the orders.
2	getTotalSalesByDay(String date)	Retrieves the total sales amount for a specific day from the orders

#### DAOjoin

	, <del></del>	
No	Method	Description
1	getNumberOfProductsSoldBy Category(int categoryId)	Retrieves the number of products sold for a specific category
2	getProductRate()	Retrieves a list of product ratings for display in the admin dashboard.
3	getOrderAdminStatistic()	Retrieves order statistics for the admin dashboard.
4	getNumberProductSoldByDay (String date)	Retrieves the number of products sold on a specific day.
5	getTotalNumberProductSold()	Retrieves the total number of products sold.
6	getProductQuality(int type)	Retrieves the product quality data based on the specified type

# c. Sequence Diagram(s)

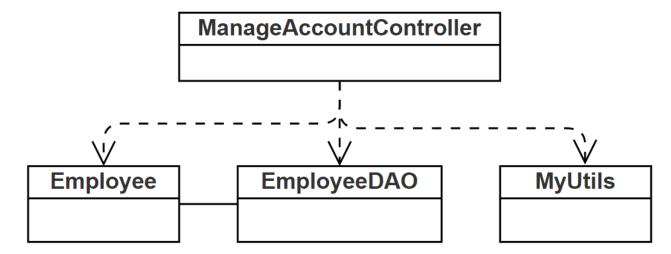


#### d. Database Queries

- getProductRate()
- select p.name ,ROUND(cast(sum(r.star\_rate)as float)/cast(count(r.id) as float),2) as star from product p join
- order detail od on od.product id = p.id join
- rate r on r.order\_detail\_id = od.id
- group by p.id,p.name
- order by star desc
- getTotalNumberProductSold()
- select COUNT(od.id) as NumberSold from [order] o join
- order detail od on o.id=od.order id
- where o.order\_status=2
- getOrderAdminStatistic()
- select TOP 5 o.order\_date, o.id, c.firstname+' '+c.lastname as fullname, sum(od.price) as total, o.paid\_date from [order] o join
- order detail od on o.id=od.order id join
- customer c on o.customer id=c.id
- group by o.id, o.order\_date, c.firstname, c.lastname, o.paid\_date
- order by o.order\_date desc
- getNumberProductSoldByDay(day)
  - o select o.order\_date , COUNT(od.id) as NumberSold from [order] o join
  - order detail od on o.id=od.order id
  - o where o.order status=2 and o.order date=?
  - o group by o.order date

# 3.3.2 Manage Account

#### a. Class Diagram



#### b. Class Specifications

controller.ManageNews.java

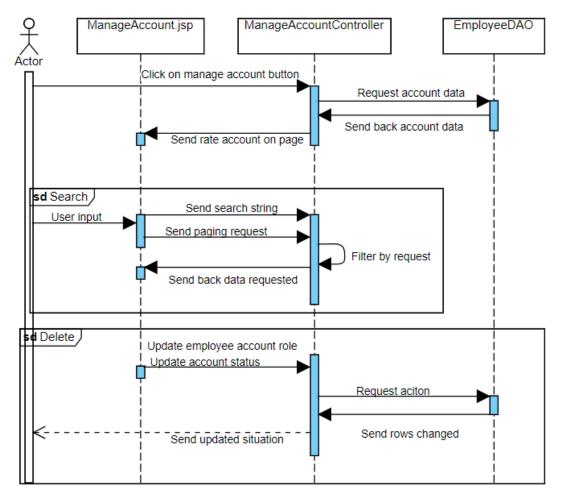
No	Method	Description
	doGet(HttpServletRequest request, HttpServletResponse response)	Handles HTTP GET requests; fetches news categories and existing news for editing, then forwards to the AddDiscount.jsp view.
	doPost(HttpServletRequest request, HttpServletResponse response)	Handles HTTP POST requests; processes form submissions for adding, updating, or deleting news based on the provided action.

### NewsDAO

No	Method	Description
1	getAll()	Retrieves all employee records from the database
2	updateEmployeeRole(int id, int roleId)	Updates the role of an employee in the database.
3	updateEmployeeStatus(int id, int statusID)	Updates the status of an employee in the database.

### MyUtils

No	Method	Description
1	setAlertAttributes(HttpServle tRequest request, boolean status, String action)	sets an alert message and type based on the success or failure of an action, which can be used for displaying user feedback in a web application.
2	getArrayListByPaging(ArrayLi st <t> list, int pageNumber, int ItemsOfPage)</t>	calculates the start and end indices for the desired page and returns the corresponding sublist.



- updateEmployeeRole(int id, int roleId)
  - UPDATE employee SET role id = ? WHERE id = ?
- updateEmployeeStatus(int id, int statusID)
  - O UPDATE employee SET [status] = ? WHERE id = ?