

**COURSE E-COMMERCE**

**SYSTEM**

**Software Design Specification**

**Trần Quang Minh**

**Lead software engineering**

– Summer 2023 –

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 10/6/2023 | Version 1 | Trần Quang Minh | <First Version> |
| 20/6/2023 | Version 2 | Trần Quang Minh |  |
| 10/7/2023 | Version 3 | Trần Quang Minh |  |
|  |  |  |  |

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | Trần Quang Minh | Leader |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[I. Introductions](#_3dy6vkm)

[1.1. Overview](#_1t3h5sf) 4

[1.2. Purpose](#_1t3h5sf) 4

[1.3. Scope](#_1t3h5sf) 5

[III. Overview](#_3dy6vkm) 5

[1. Code Packages](#_1t3h5sf) 6

[2. Database Design](#_4d34og8) 8

[a. Database Schema](#_2s8eyo1) 8

[b. Table Description](#_17dp8vu) 8

[IIII. Code Designs](#_3rdcrjn) 9

[1. Public Account](#_26in1rg) 9

[a. Class Diagram](#_lnxbz9) 10

[b. Class Specifications](#_35nkun2) 11

[c. Sequence Diagram(s)](#_1ksv4uv) 12

[2. Public Order 17](#_2jxsxqh)

[a. Class Diagram 17](#_z337ya)

[b. Class Specifications 17](#_3j2qqm3)

[c. Sequence Diagram(s) 18](#_1y810tw)

[3.Public Insurance 20](#_4f1mdlm)

[a. Class Diagram 20](#_2u6wntf)

[b. Class Specifications 20](#_19c6y18)

[c. Sequence Diagram(s) 21](#_3tbugp1)

[4. Public Product 22](#_3whwml4)

[a. Class Diagram 22](#_2bn6wsx)

[b. Class Specifications 22](#_qsh70q)

[c. Sequence Diagram(s) 23](#_3as4poj)

[5. Public Category 25](#_41mghml)

[a. Class Diagram 25](#_2grqrue)

[b. Class Specifications 25](#_vx1227)

[c. Sequence Diagram(s) 26](#_3fwokq0)

# INTRODUCTIONs

## 1.1 Overview

Welcome to Website The COURSE E-COMMERCE SYSTEM(TCES) - where you can find top programming courses to develop your skills. We are committed to providing you with high-quality courses from industry-leading experts.

With the remarkable advancement of technology and the digital revolution, programming has become an essential skill for many professions. At Website TCES, we understand that learning programming can be challenging, especially for beginners. Therefore, we have created a convenient, flexible, and effective learning platform to help you progress quickly.

Website TCES offers a variety of diverse programming courses, including popular programming languages such as Python, JavaScript, Java, C++, and various development platforms like web, mobile, and desktop. You can easily search and select courses that align with your personal goals.

At Website TCES, we place special emphasis on delivering quality content. The courses are designed by experienced experts, ensuring that you learn the latest knowledge and cutting-edge programming techniques. You will have opportunities to apply your knowledge through exercises, real-life projects, and high-quality reference materials.

Additionally, Website TCES provides a strong support community to assist you throughout your learning journey. You can exchange ideas, ask questions, and share experiences with other members in the community, building professional networks and enhancing your communication skills.

Start your programming journey today with Website TCES! We believe that with our support, you can develop programming skills confidently and effectively.

**1.2 Purpose**

The COURSE E-COMMERCE SYSTEM(TCES) system consists of an Administrator Web site and a Client Web site. The Admin page supports Admins and Employees as a fully functional CMS website in CRUD for Products, Accounts,...

The Client's Website will have various software components, such as the main page, product listing, shopping cart, checkout process, and user profile management. User interfaces should be created to be easy to use, intuitive, and in line with modern web design standards. In addition, they should be responsive to different devices and ensure a consistent experience across the entire site.

This Software Requirements Specification illustrates, in clear terms, the main uses of the system and the required functionality specified by our customers. The intended audience of this document is us

Main customers of Computer E-Commerce System: Mr. Nguyen Thanh Y - SWP391 Instructor and members of SE1715-NET SUMMER 2023, as well as other students studying SE1715 will request permission to access the materials there.

**1.3 Scope**

The scope of Programming Courses (TCES) in relation to course sales is to provide a comprehensive and user-friendly platform for users to browse, select and purchase programming courses.

The site will feature an extensive catalog of programming courses, covering a variety of programming languages, frameworks, and platforms. Each course will have a dedicated page with detailed information, including course syllabus, instructor details, prerequisites, duration, and reviews from previous students.

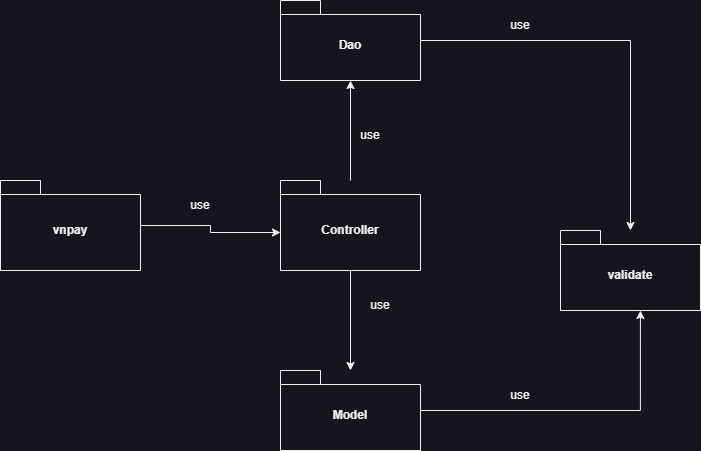
The TCES website will provide a streamlined and secure checkout process, allowing users to easily add the desired courses to their cart and proceed with the payment. Multiple payment options will be available to satisfy different user preferences.

User accounts will be created for registered users, allowing them to track their course progress, access purchased courses and receive updates on new course releases or events upcoming. Users will have the ability to rate and review the courses they have completed, providing valuable feedback to other future learners.

Overall, TCES's selling point is to create a seamless and enjoyable experience for users, empowering them to explore and gain programming skills through the high-quality courses offered on the web. communication.

# II. Overview

## 1. Code Packages

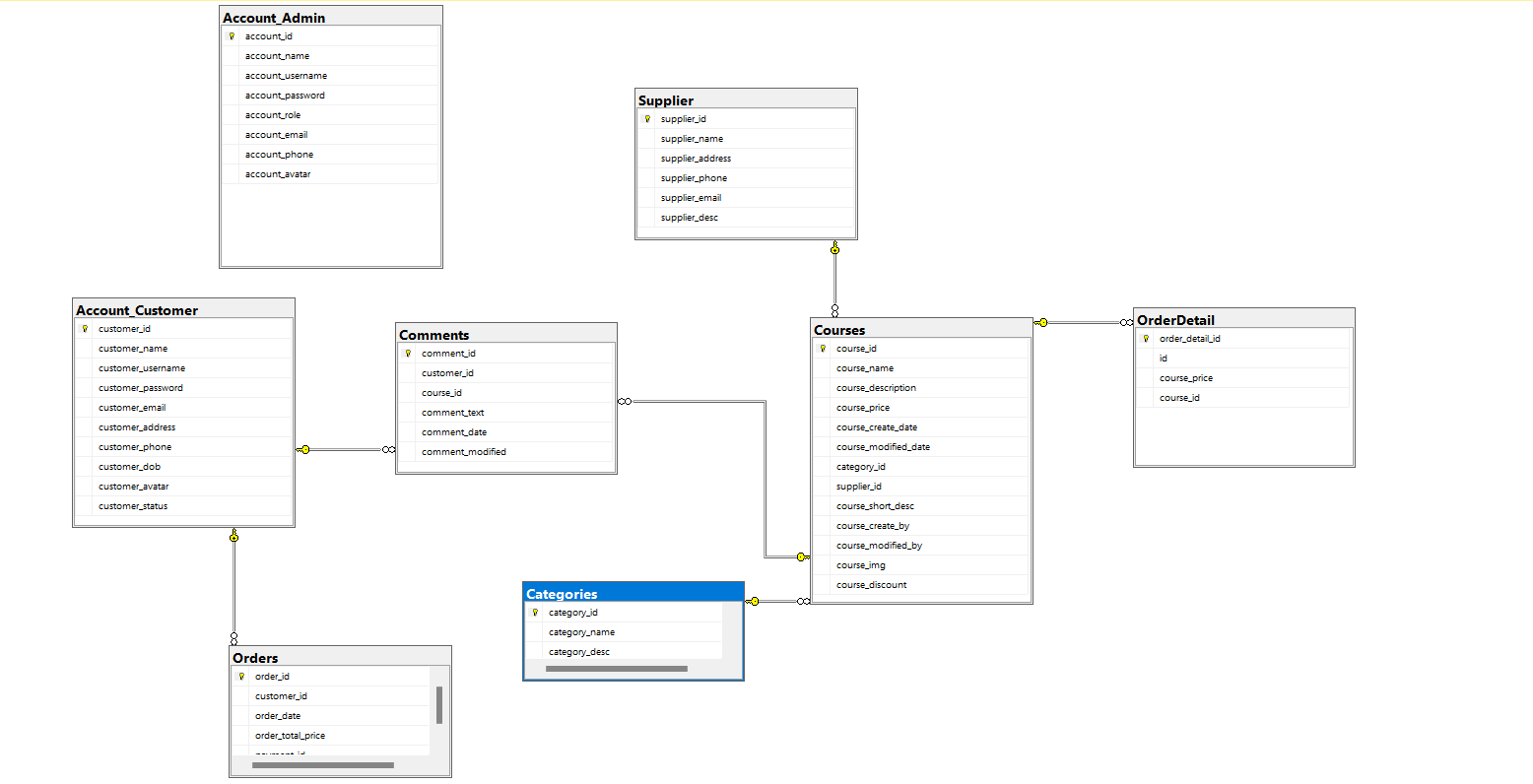


***Package descriptions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| 1 | Model | The Model package typically contains classes or components that represent the application's data model or domain objects. These classes define the structure, behavior, and relationships of the data entities used in the application. The Model package often embodies the business logic and rules associated with the data entities. It may include classes representing entities, value objects, enumerations, or other components required to model the problem domain accurately. The Model package plays a vital role in implementing the application's core functionality and interacting with the data stored in the database. |
| 2 | Controller | The Controller package deals with the handling of requests, coordinating actions, and managing the flow of data in an application. It acts as an intermediary between the user interface (UI) or external systems and the internal components of the application. The Controller receives input from the user or external sources, invokes appropriate methods or services from the Model package to process the request, and prepares the response to be sent back to the UI or external systems. It plays a pivotal role in implementing the application's business logic, orchestrating actions, and ensuring proper data flow between different components. |
| 3 | Dao | *The Dao package is responsible for handling data access and database operations. It provides an interface between the application and the underlying data storage system, such as a relational database or a NoSQL database. The Dao package typically includes classes that encapsulate CRUD (Create, Retrieve, Update, Delete) operations on the data entities. It abstracts away the specific details of data storage and retrieval, allowing the rest of the application to work with the data through a higher-level API.* |
| 4 | validate | The Validate package is often used for input validation and data integrity checks. It contains classes or functions that validate user inputs or data before processing them further. This package helps ensure that the input values are in the expected format and meet certain criteria or constraints. For example, it can check the length of a string, verify that a number falls within a certain range, or validate email addresses. Input validation is crucial for maintaining data quality and preventing security vulnerabilities like SQL injection or cross-site scripting attacks. |
| 5 | vnpay | This package to control the payment through VNPay |

## 2. Database Design

### a. Database Schema



### b. Table Description

Course:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | course\_id | int |  | x | x | x | PK |
| 2 | course\_name | nvarchar | 255 |  |  |  |  |
| 3 | course\_description | text |  |  |  |  |  |
| 4 | course\_price | decimal(10, 2) |  |  |  |  |  |
| 5 | course\_create\_date | date |  |  |  |  |  |
| 6 | course\_modified\_date | date |  |  |  |  |  |
| 7 | category\_id | int |  |  |  | x | FK |
| 8 | supplier\_id | int |  |  |  | x | FK |
| 9 | course\_short\_desc | nvarchar | 255 |  |  |  |  |
| 10 | course\_create\_by | nvarchar | 255 |  |  |  |  |
| 11 | course\_modified\_by | nvarchar | 255 |  |  |  |  |
| 12 | course\_img | nvarchar | 50 |  |  |  |  |
| 13 | course\_discount | int |  |  |  |  |  |

Account\_Customer:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | customer\_id | int |  | x | x | x | PK |
| 2 | customer\_name | nvarchar | 255 |  |  |  |  |
| 3 | customer\_username | nvarchar | 255 |  |  |  |  |
| 4 | customer\_password | nvarchar | 50 |  |  |  |  |
| 5 | customer\_email | nvarchar | 255 |  |  |  |  |
| 6 | customer\_address | nvarchar | 255 |  |  |  |  |
| 7 | customer\_phone | nvarchar | 20 |  |  |  |  |
| 8 | customer\_dob | date |  |  | x |  |  |
| 9 | customer\_avatar | nvarchar |  |  | x |  |  |
| 10 | customer\_status | int |  |  | x |  |  |

Account\_Admin:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | account\_id | int |  | x | x | x | PK |
| 2 | account\_name | nvarchar | 255 |  |  |  |  |
| 3 | account\_username | nvarchar | 50 |  |  |  |  |
| 4 | account\_password | nvarchar | 255 |  |  |  |  |
| 5 | account\_role | int |  |  |  |  |  |
| 6 | account\_email | nvarchar | 255 |  |  |  |  |
| 7 | account\_phone | int |  |  |  |  |  |
| 8 | account\_avatar | nvarchar | 50 |  |  |  |  |

Category:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | category\_id | int |  | x | x | x | PK |
| 2 | category\_name | nvarchar | 255 |  |  |  |  |
| 3 | category\_desc | nvarchar | 1000 |  |  |  |  |

Comments:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | comment\_id | int |  | x | x | x | PK |
| 2 | customer\_id | int |  |  |  | x | FK |
| 3 | course\_id | int |  |  |  | x | FK |
| 4 | comment\_text | text |  |  |  |  |  |
| 5 | comment\_date | datetime |  |  |  |  |  |
| 6 | comment\_modified | datetime |  |  | x |  |  |

News:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | news\_id | int |  | x |  | x | PK |
| 2 | news\_title | nvarchar | 255 |  |  |  |  |
| 3 | news\_content | text |  |  |  |  |  |
| 4 | news\_created\_date | date |  |  |  |  |  |
| 5 | news\_modified\_date | date |  |  |  |  |  |
| 6 | news\_created\_by | nvarchar | 255 |  |  |  |  |
| 7 | news\_modified\_by | nvarchar | 255 |  |  |  |  |

Orders:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | order\_id | int |  | x | x | x | PK |
| 2 | customer\_id | int |  |  | x | x | FK |
| 3 | order\_date | date |  |  |  |  |  |
| 4 | order\_total\_price | decimal(10, 2) |  |  |  |  |  |
| 8 | payment\_id | int |  |  |  |  |  |

OrderDetail:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | order\_detail\_id | int |  | x | x | x | PK |
| 2 | id | int |  |  |  |  |  |
| 3 | course\_price | decimal(10, 2) |  |  |  |  |  |
| 4 | course\_id | int |  |  |  | x | FK |

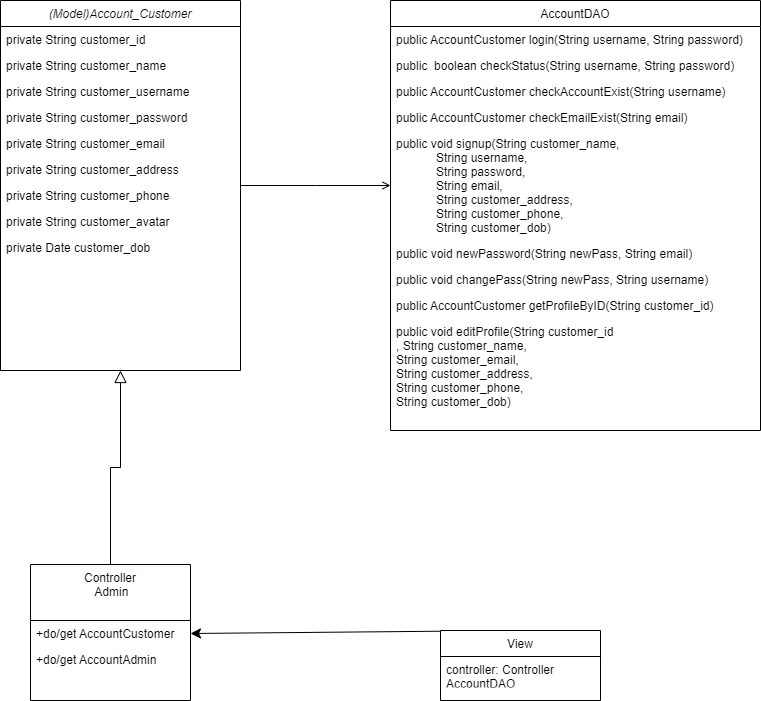
Supplier:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field name** | **Type** | **Size** | **Unique** | **Not null** | **PK/ FK** | **Notes** |
| 1 | supplier\_id | int |  | x | x | x | PK |
| 2 | supplier\_name | nvarchar | 255 |  |  |  |  |
| 3 | supplier\_address | nvarchar | 255 |  |  |  |  |
| 4 | supplier\_phone | int |  |  |  |  |  |
| 5 | supplier\_email | nvarchar | 255 |  |  |  |  |
| 6 | supplier\_desc | nvarchar | 500 |  | x |  |  |

# III. Code Designs

## 1. Public Account

### a. Class Diagram



### b. Class Specifications

#### AccountDAO.java

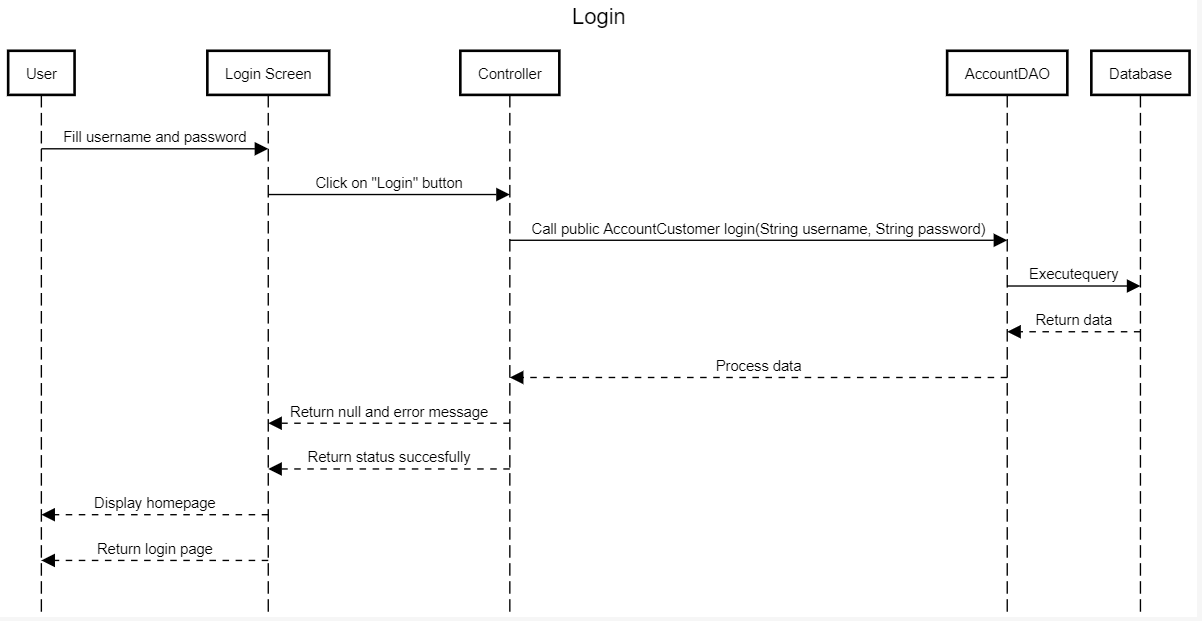
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 01 | public AccountCustomer login(String username, String password) | Handle get AccountCustomer by customer\_username and customer\_password |
| 02 | public boolean checkStatus(String username, String password) | Handle status of account |
| 03 | public AccountCustomer checkAccountExist(String username) | Handle AccountCustomer exist or not |
| 04 | public AccountCustomer checkEmailExist(String email) | Handle email have in database or not |
| 05 | public void signup(String customer\_name,  String username,  String password,  String email,  String customer\_address,  String customer\_phone,  String customer\_dob) | Handle signup new accountcustomer |
| 06 | public void newPassword(String newPass, String email) | Handle edit newpassword when user forgot password |
| 07 | public void changePass(String newPass, String username) | Handle change password |
| 08 | public AccountCustomer getProfileByID(String customer\_id) | Handle get profile of accountcustomer |
| 09 | public void editProfile(String customer\_id, String customer\_name, String customer\_email, String customer\_address, String customer\_phone, String customer\_dob) | Handle edit profile |

#### AccountCustomerDAO.java

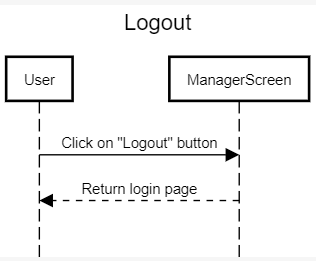
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 01 | public AccountCustomer getAccountCustomerById(String customer\_id) | Handle get AccountCustomer by ID off account |

### c. Sequence Diagram(s)

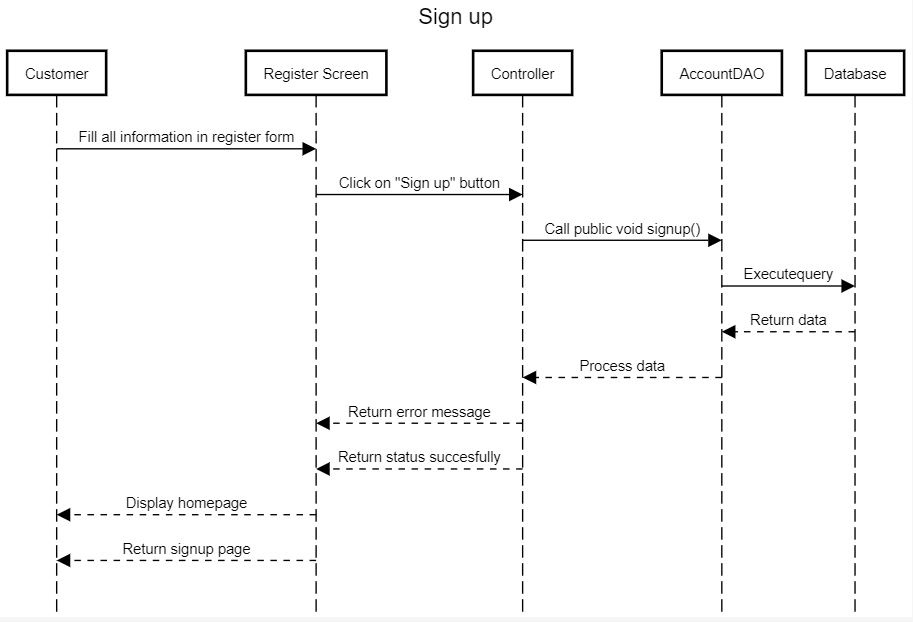
#### Login



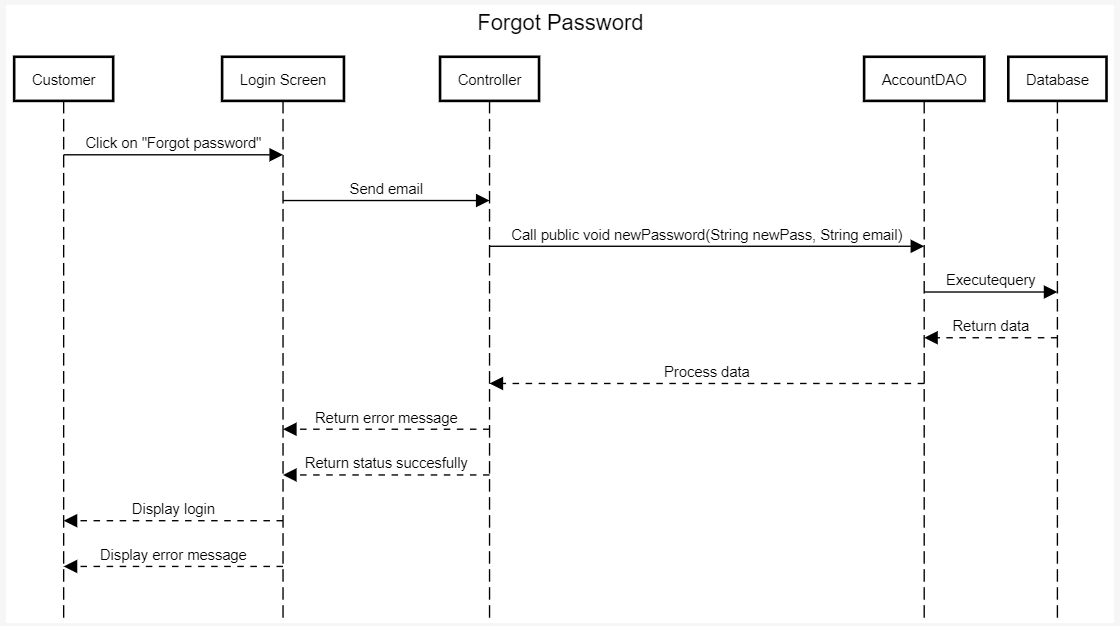
#### Logout



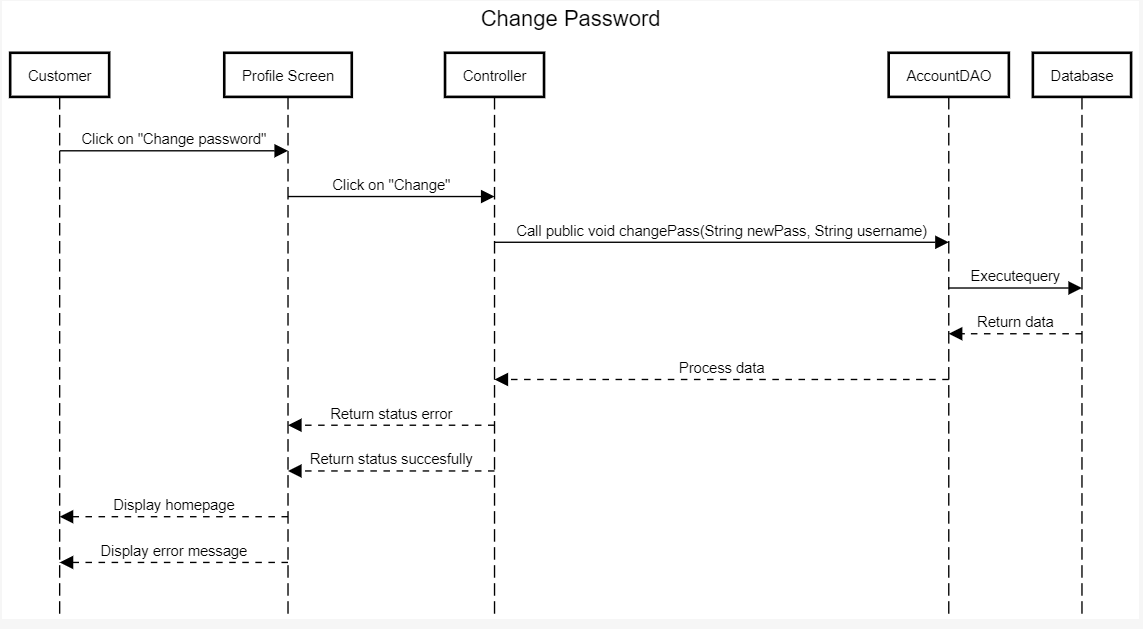
#### Signup



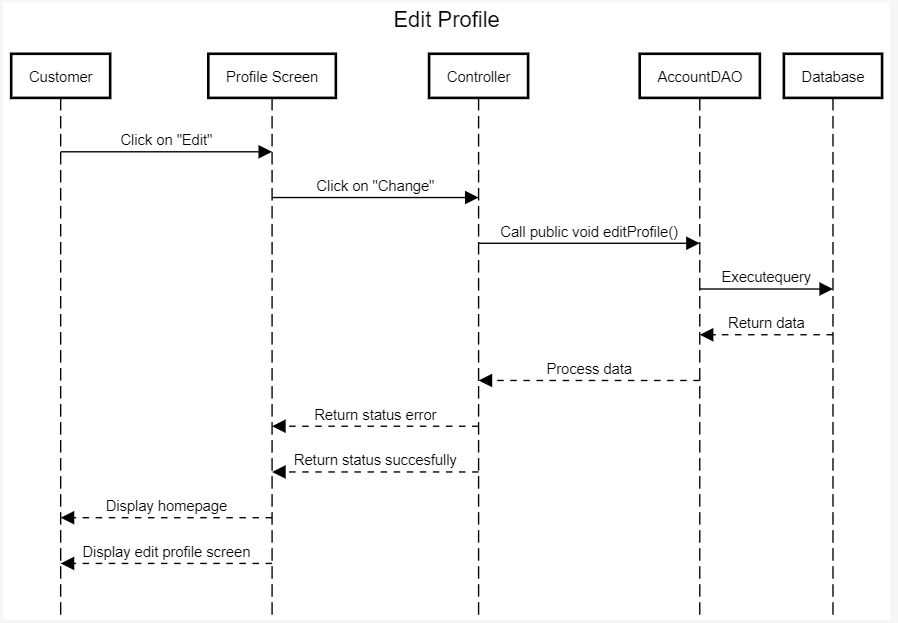
#### Forgot Password



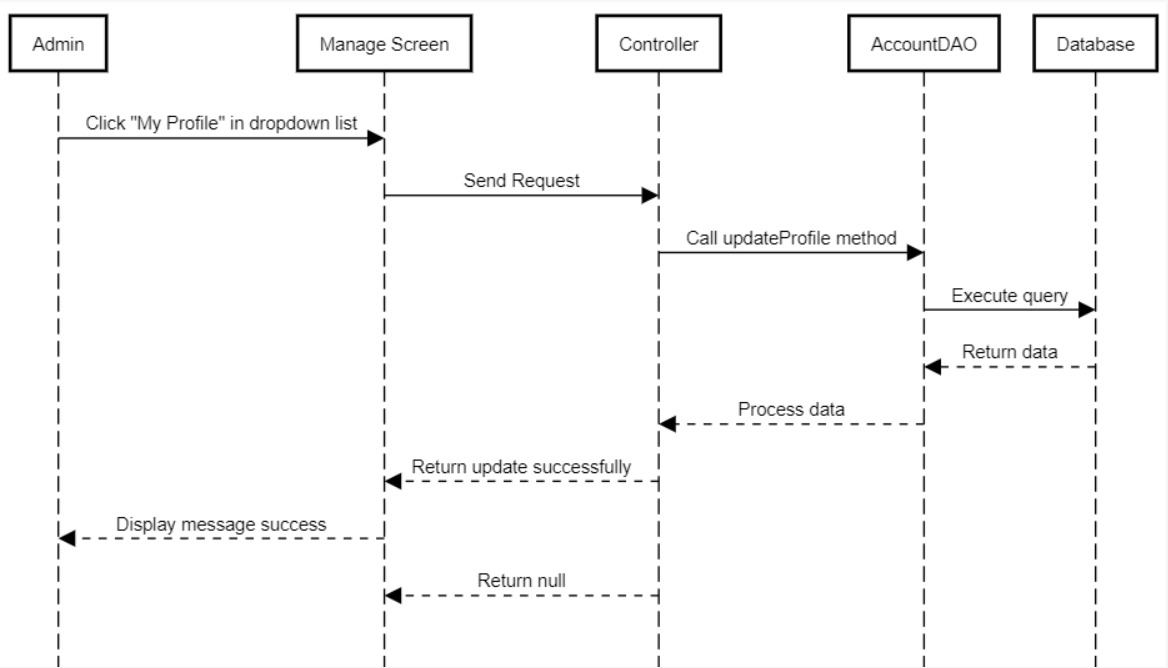
#### Change Password



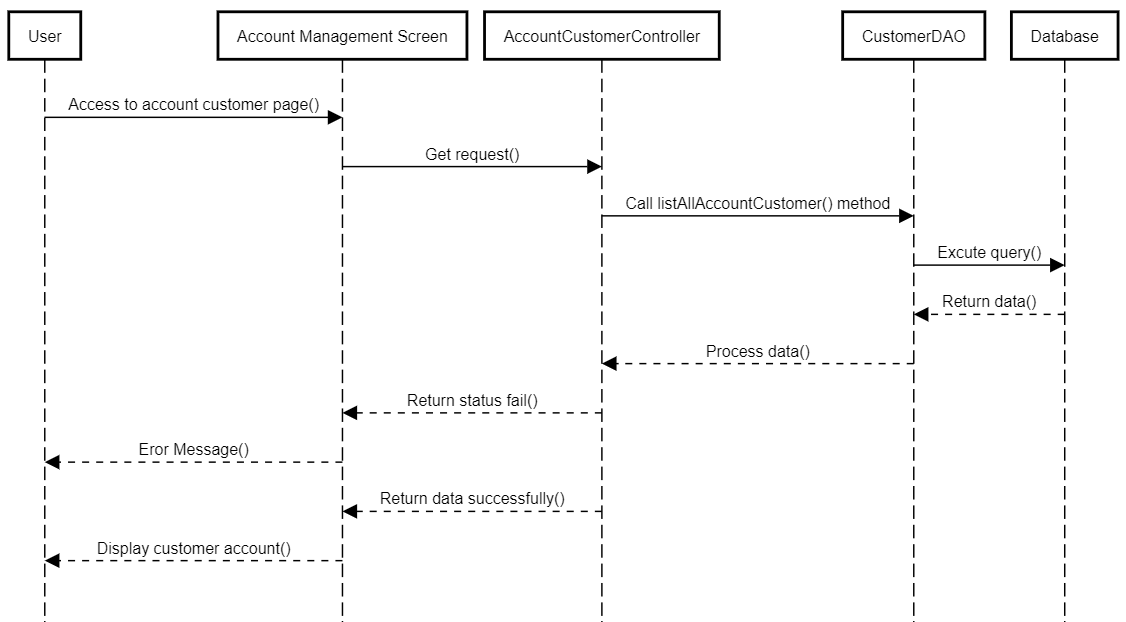
#### Edit Profile



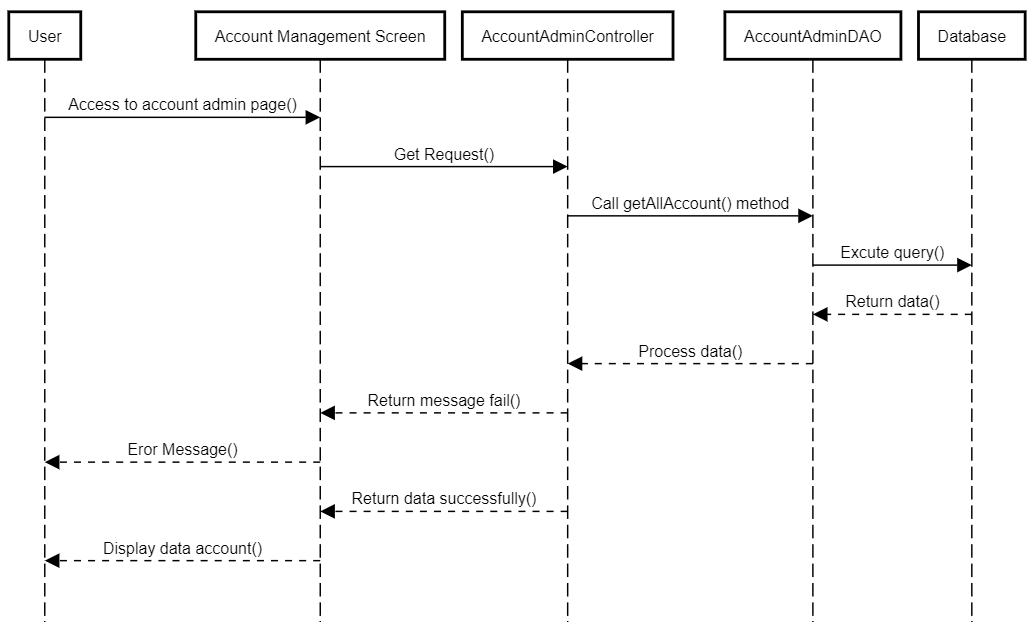
#### Edit Profile Admin



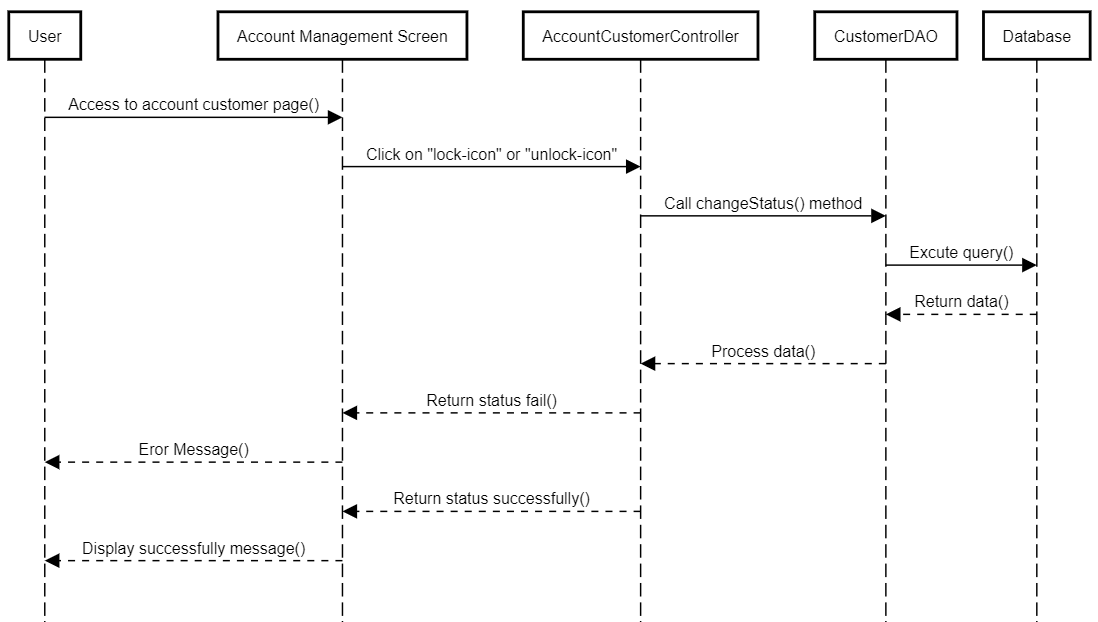
#### View customer account



#### View admin account

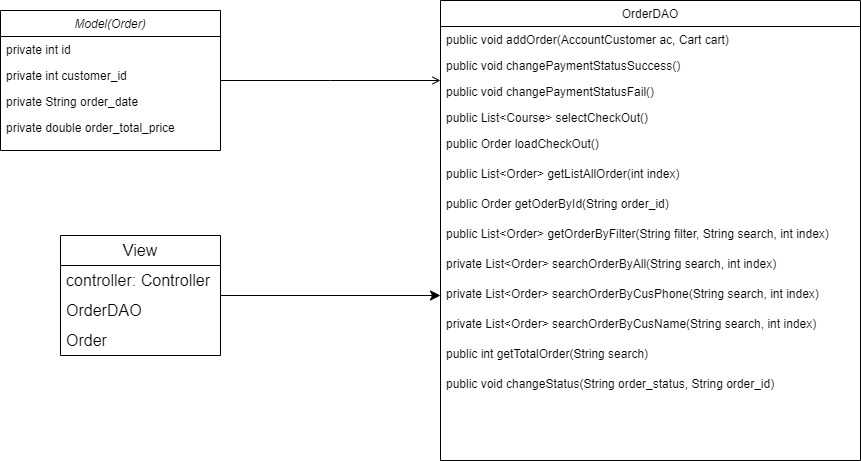


#### Update account status



## 2. Public Order

### a. Class Diagram



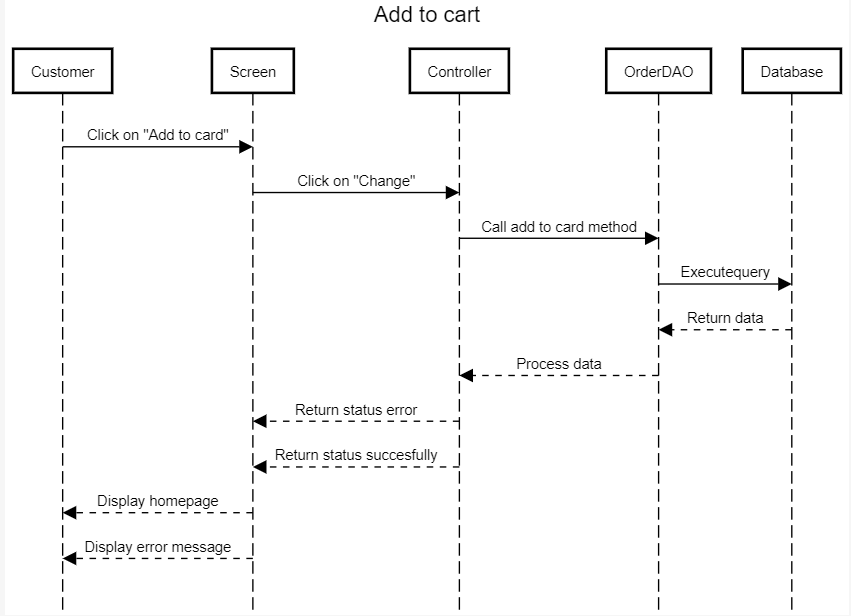
### b. Class Specifications

#### OrderDAO.java

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 01 | public void addOrder(AccountCustomer ac, Cart cart) | Handle add order to cart |
| 02 | public void changePaymentStatusSuccess() | Handle change payment status to success |
| 03 | public void changePaymentStatusFail() | Handle change payment status to fail |
| 04 | public List<Course> selectCheckOut() | Handle to select checkout |
| 05 | public Order loadCheckOut() | Handle load checkout |
| 06 | public List<Order> getListAllOrder(int index) | Handle request get list order |
| 07 | public Order getOderById(String order\_id) | Handle request get order by ID |
| 08 | public List<Order> getOrderByFilter(String filter, String search, int index) | Handle request get order with filter |
| 09 | private List<Order> searchOrderByCusPhone(String search, int index) | Handle request get list order by customer phone |
| 10 | private List<Order> searchOrderByCusName(String search, int index) | Handle request get list order by customer name |
| 11 | public int getTotalOrder(String search) | Get total order |

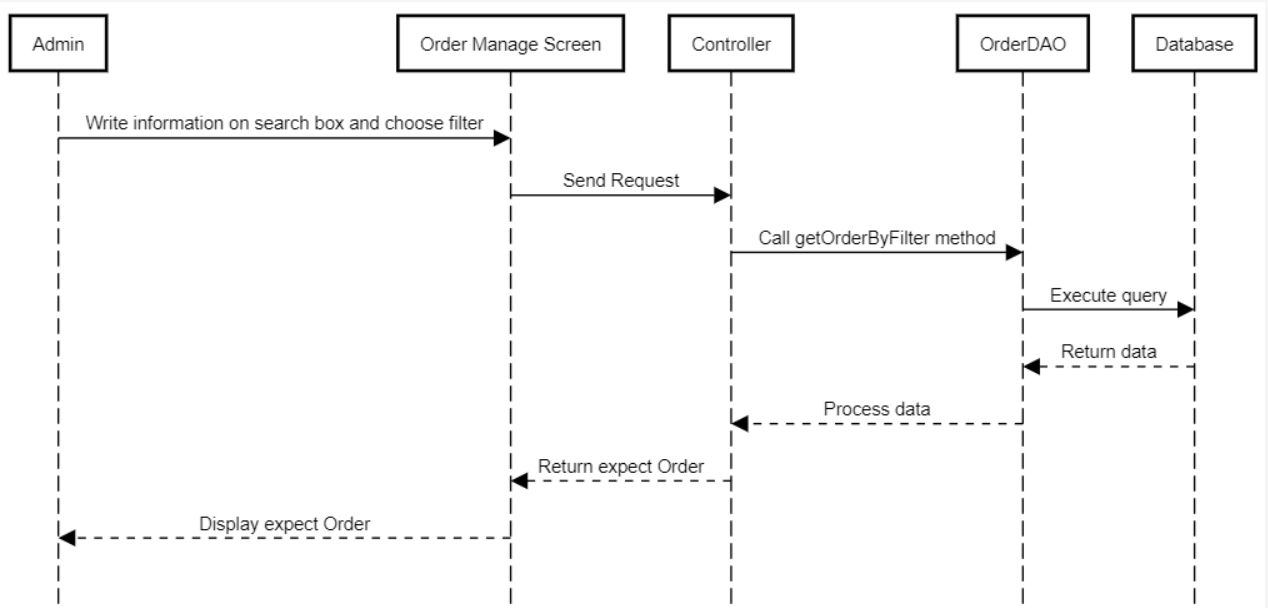
### c. Sequence Diagram(s)

#### Add to cart

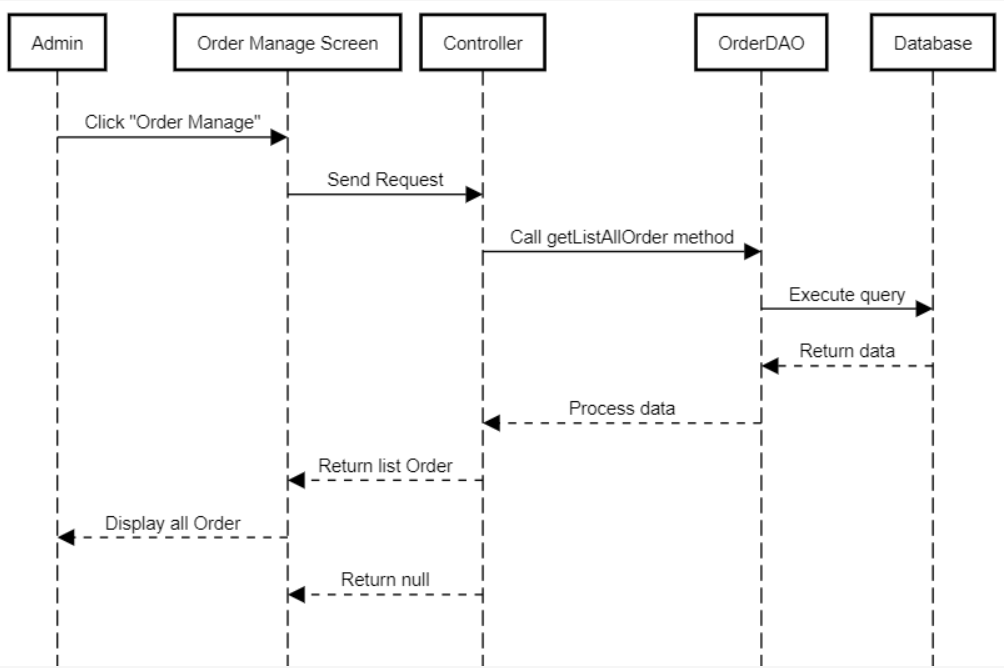


#### View order detail

#### Search Order



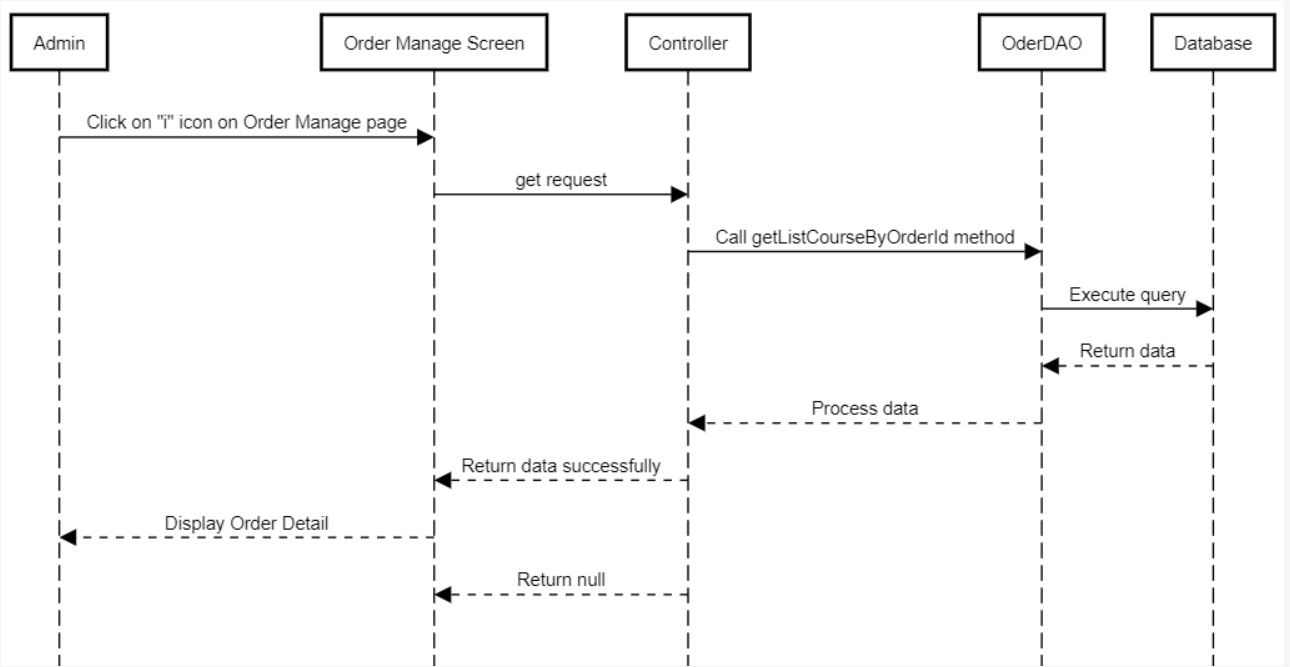
#### View all order



#### Change order status

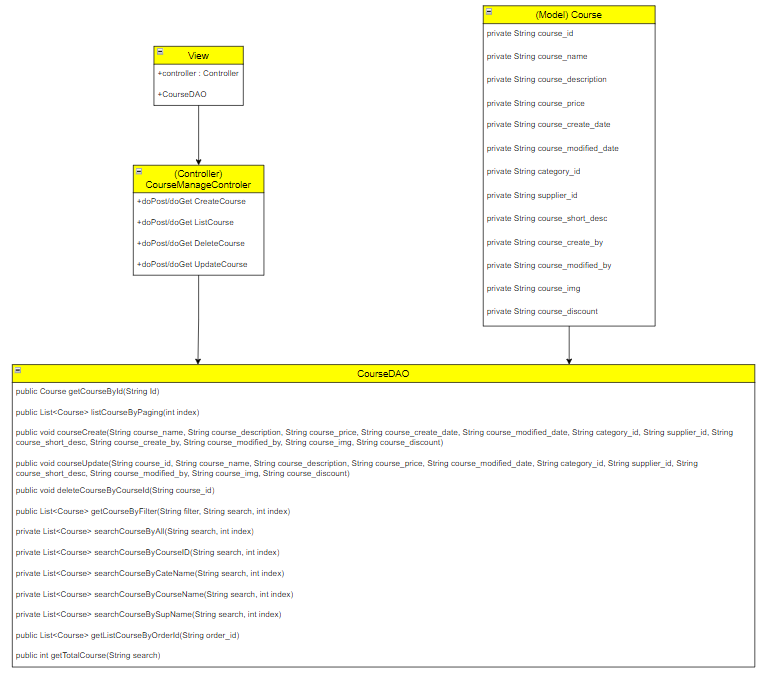
## 

#### View order detail



## 3. Public Product

### a. Class Diagram



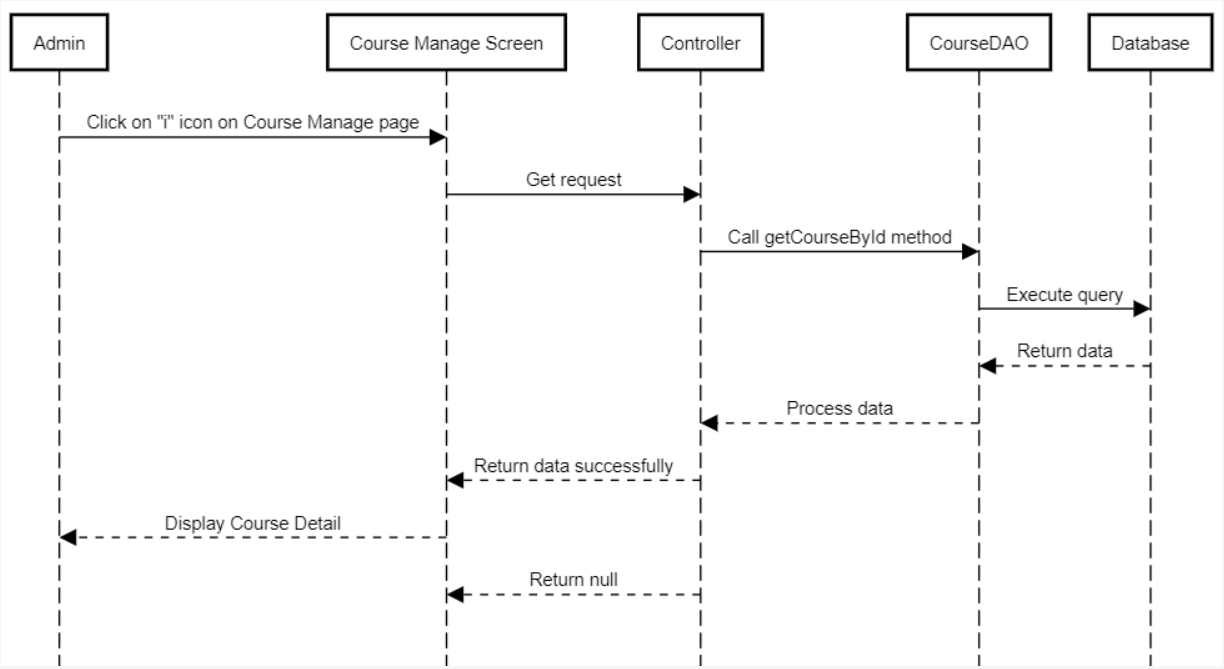
### b. Class Specifications

#### CourseDAO

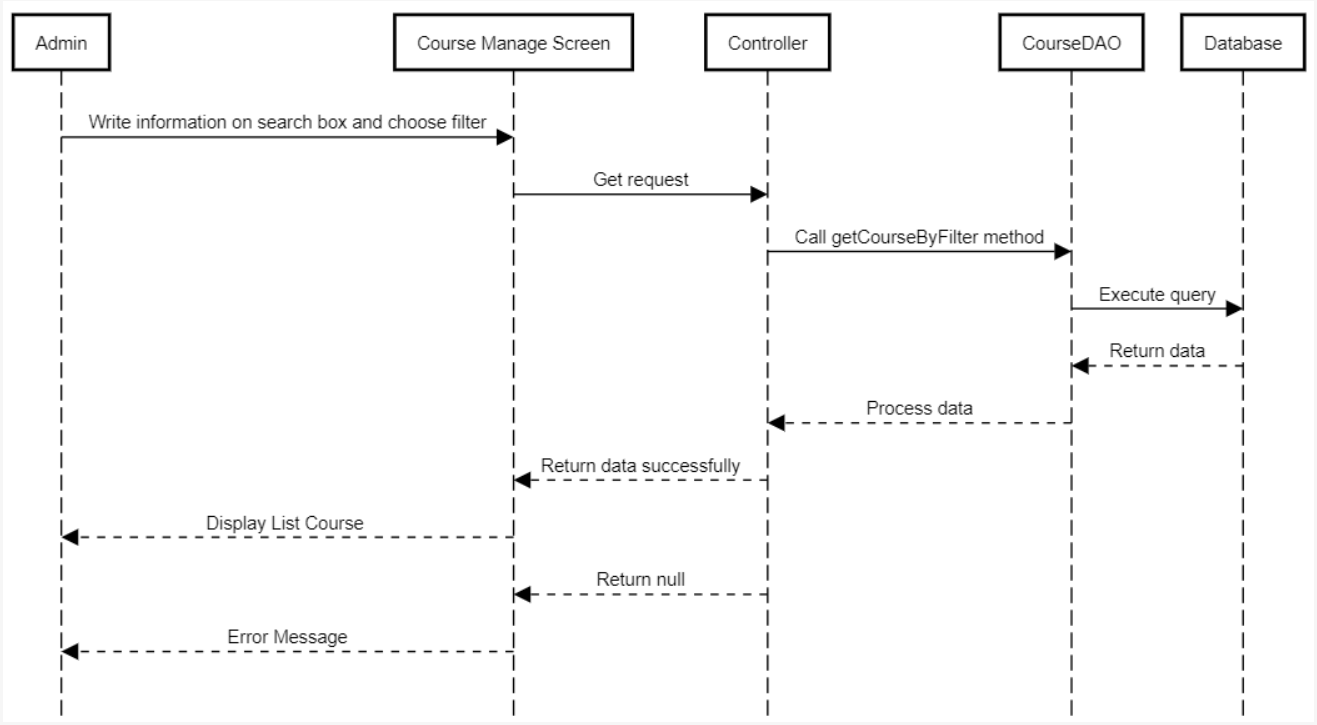
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 01 | public Course getCourseById(String Id) | Handle request get course by id |
| 02 | public List<Course> listCourseByPaging(int index) | Handle request get list course |
| 03 | public void courseCreate(String course\_name, String course\_description, String course\_price, String course\_create\_date, String course\_modified\_date, String category\_id, String supplier\_id, String course\_short\_desc, String course\_create\_by, String course\_modified\_by, String course\_img, String course\_discount) | Handle request add a course |
| 04 | public void courseUpdate(String course\_id, String course\_name, String course\_description, String course\_price, String course\_modified\_date, String category\_id, String supplier\_id, String course\_short\_desc, String course\_modified\_by, String course\_img, String course\_discount) | Handle request update a product |
| 05 | public void deleteCourseByCourseId(String course\_id) | Handle request delete course by id |
| 06 | public List<Course> getCourseByFilter(String filter, String search, int index) | Handle request search course |
| 07 | private List<Course> searchCourseByAll(String search, int index) | Handle request search all course |
| 08 | private List<Course> searchCourseByCourseID(String search, int index) | Handle request search course by id course |
| 09 | private List<Course> searchCourseByCateName(String search, int index) | Handle request search course by category name |
| 10 | private List<Course> searchCourseByCourseName(String search, int index) | Handle request search course by course name |
| 11 | private List<Course> searchCourseBySupName(String search, int index) | Handle request search course by supplier name |
| 12 | public List<Course> getListCourseByOrderId(String order\_id) | Handle request get list course by order id |
| 13 | public int getTotalCourse(String search) | Handle request total course |

### c. Sequence Diagram(s)

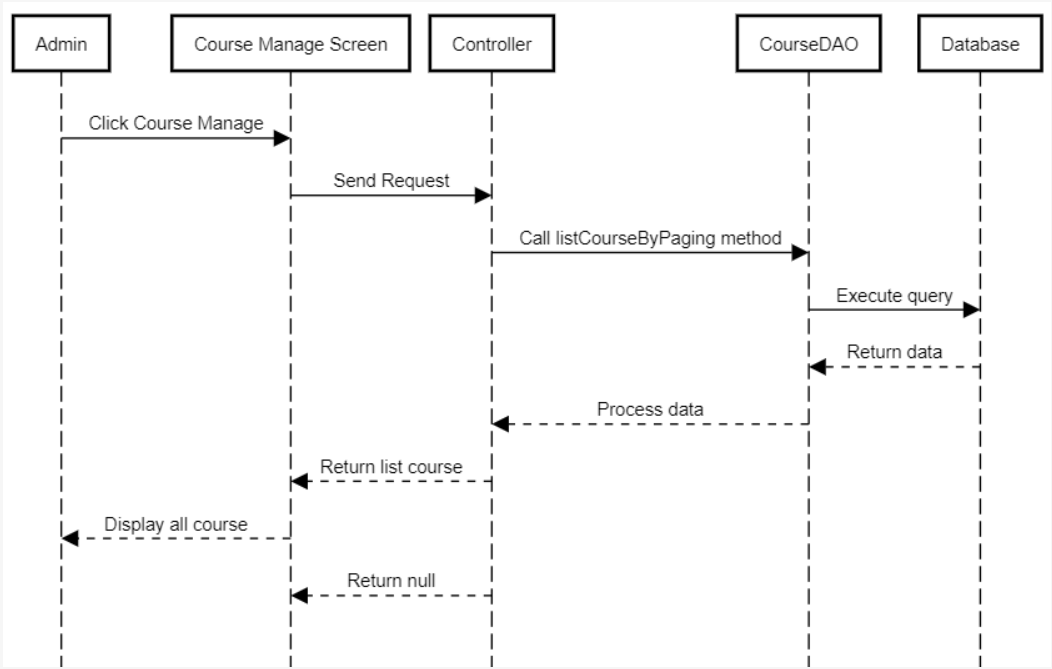
#### View product detail



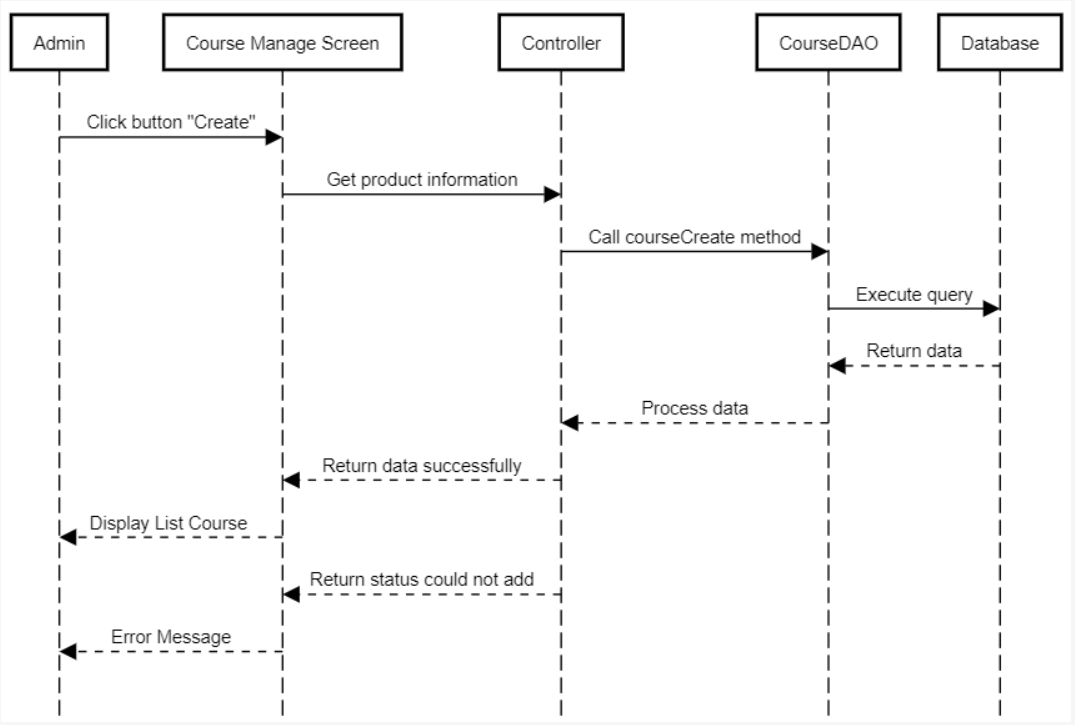
#### Search product



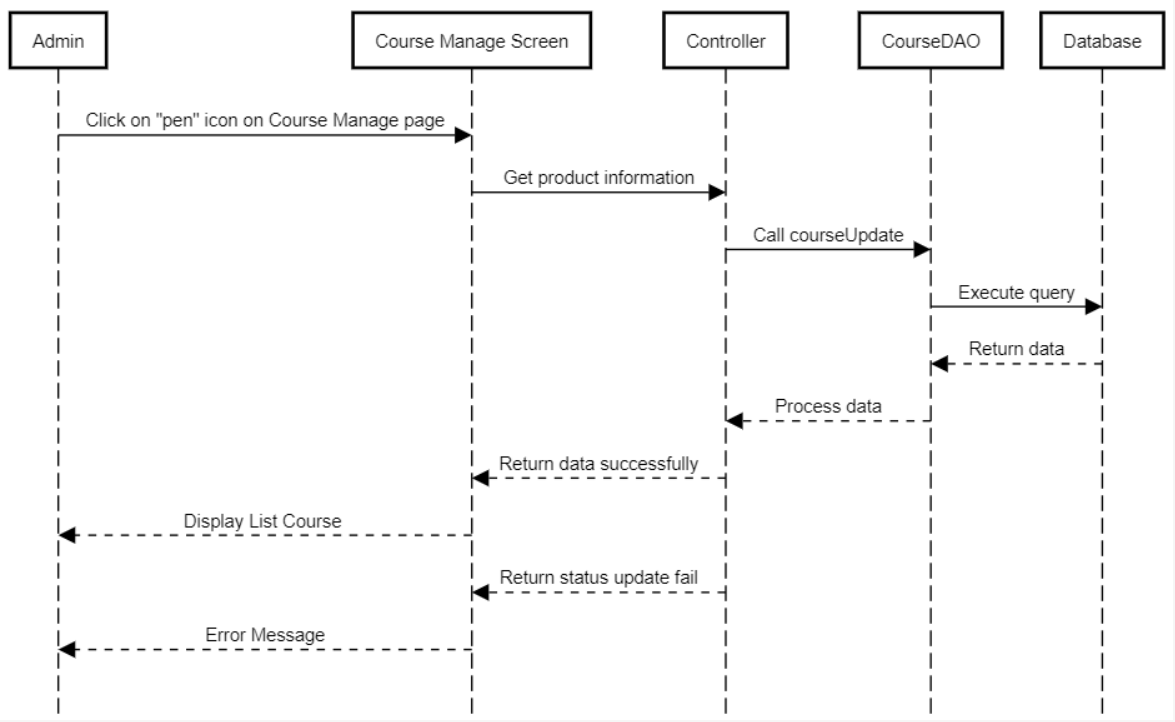
#### View all product



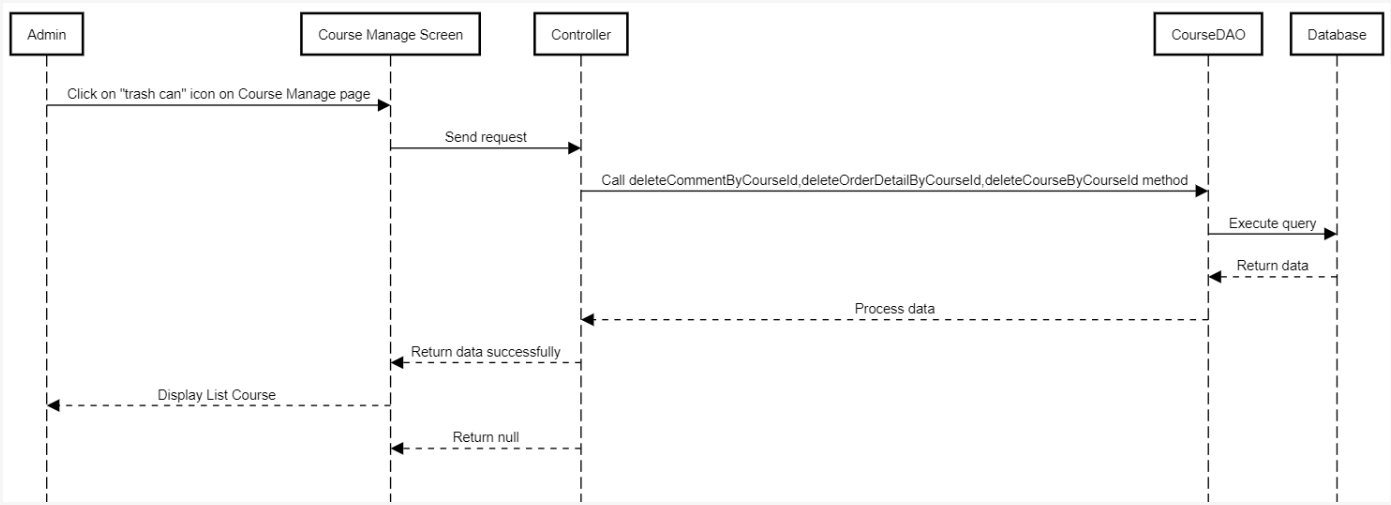
#### Add Product



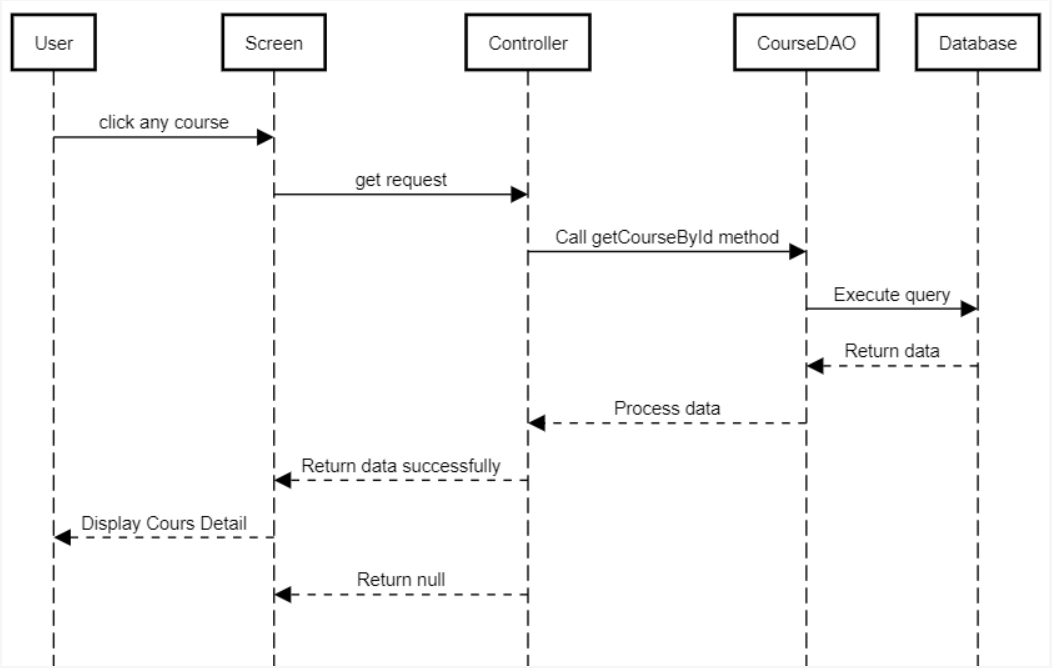
#### Edit product



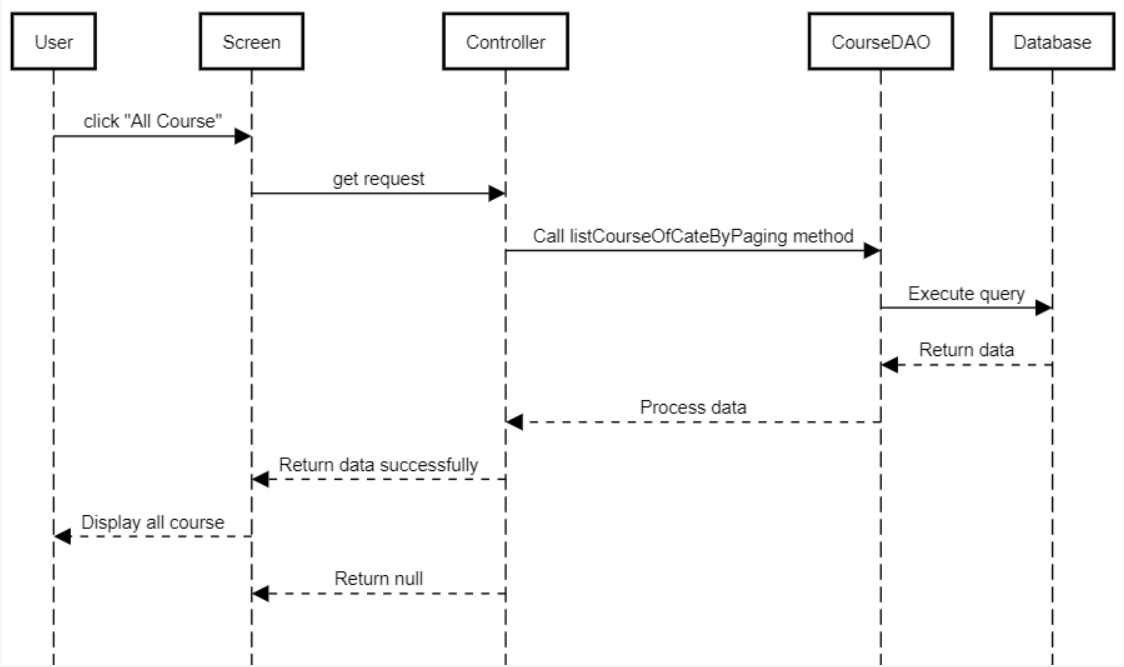
#### Delete product



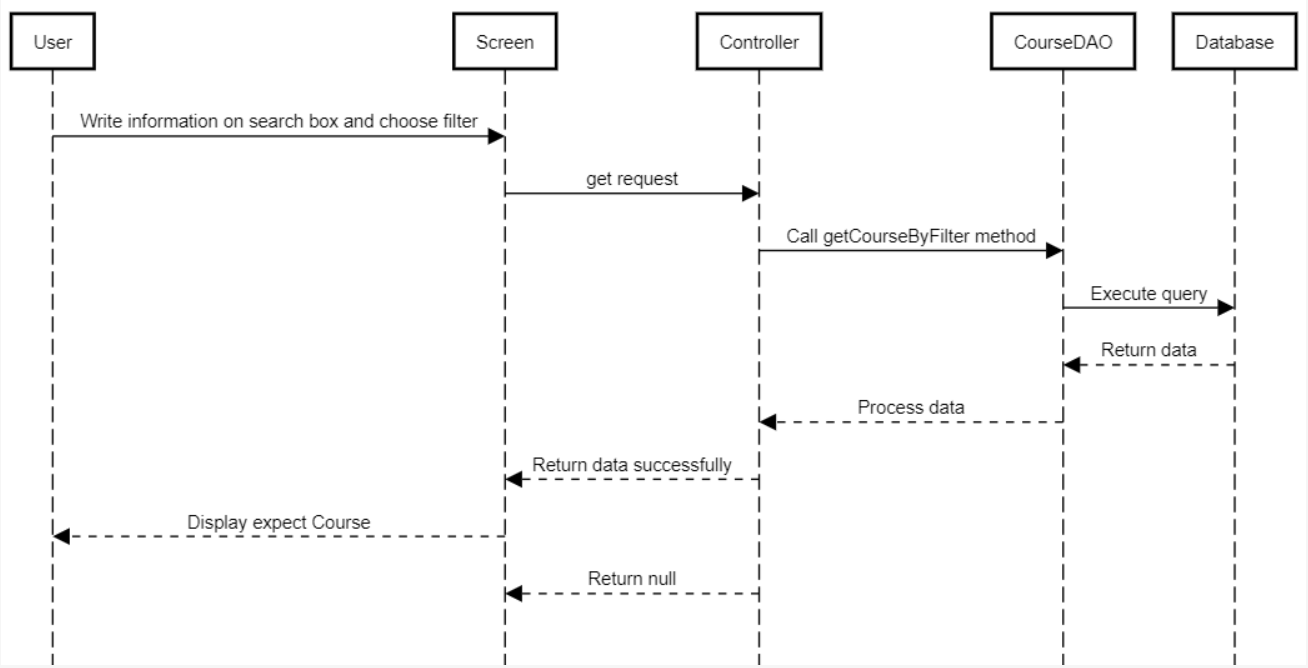
#### View product detail User



#### View all product User



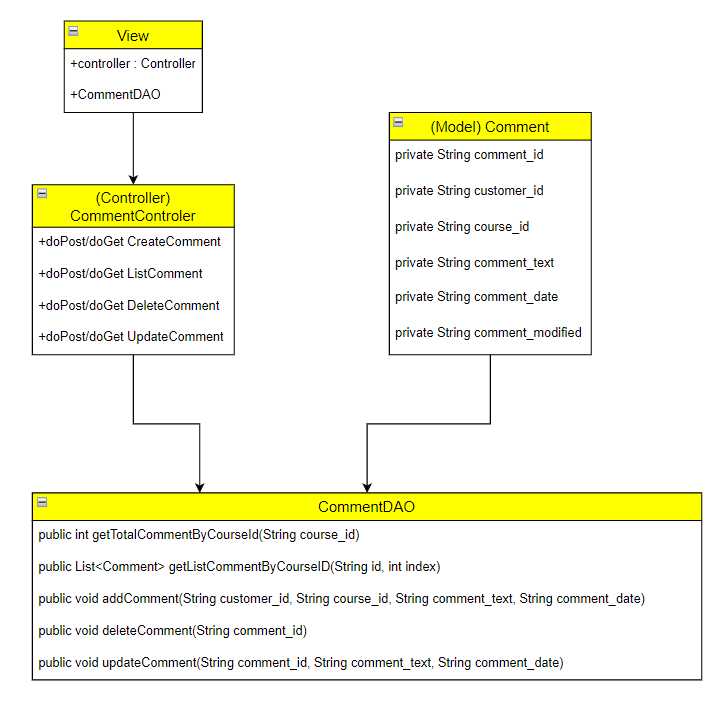
#### Search product User



#### 

## 4. Public Comment

### a. Class Diagram



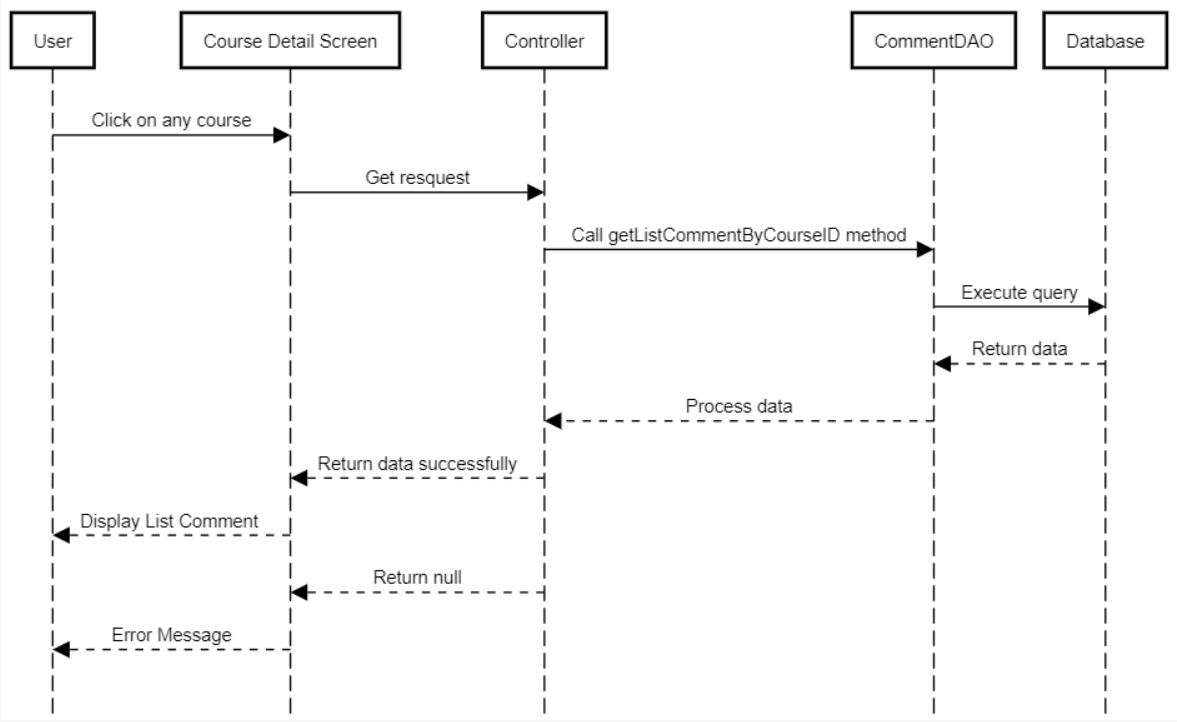
### b. Class Specifications

#### CommentDAO

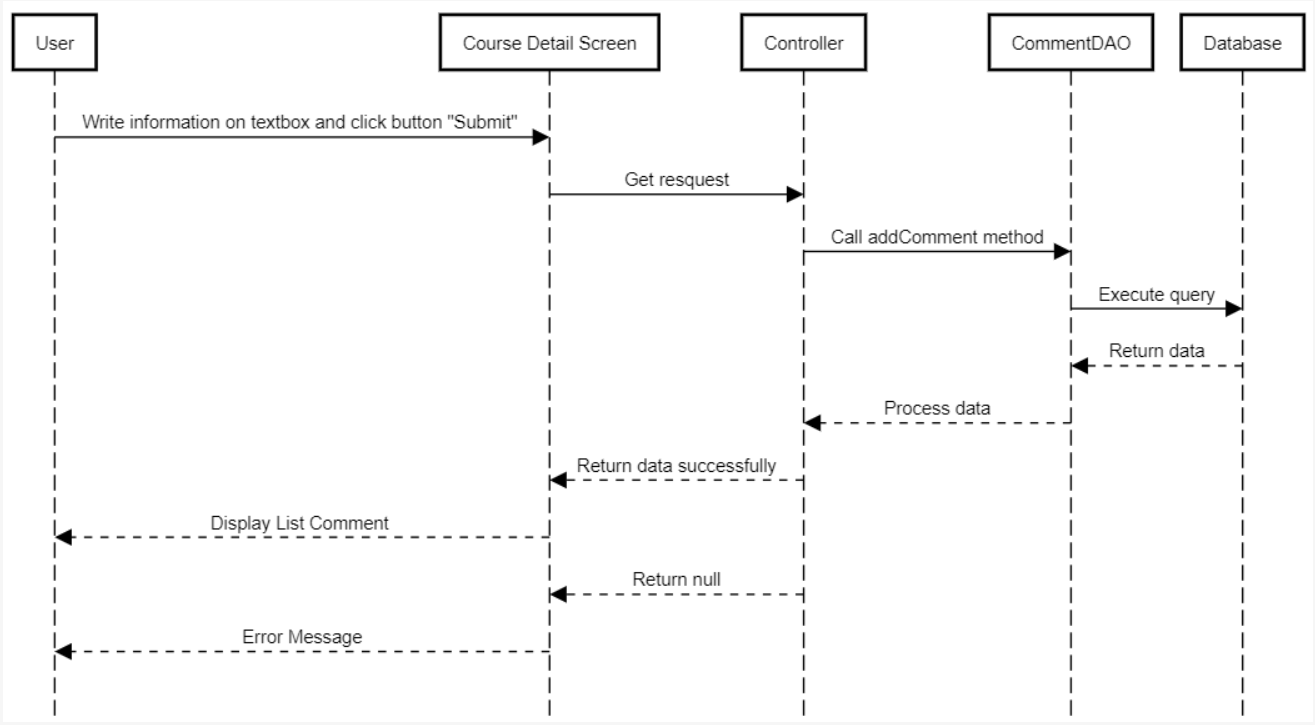
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 01 | public int getTotalCommentByCourseId(String course\_id) | Handle requests to receive total comments by id course |
| 02 | public List<Comment> getListCommentByCourseID(String id, int index) | Handle request get list comment by id course |
| 03 | public void addComment(String customer\_id, String course\_id, String comment\_text, String comment\_date) | Handle request add comment |
| 04 | public void deleteComment(String comment\_id) | Handle request delete comment |
| 05 | public void updateComment(String comment\_id, String comment\_text, String comment\_date) | Handle request update comment |

### c. Sequence Diagram(s)

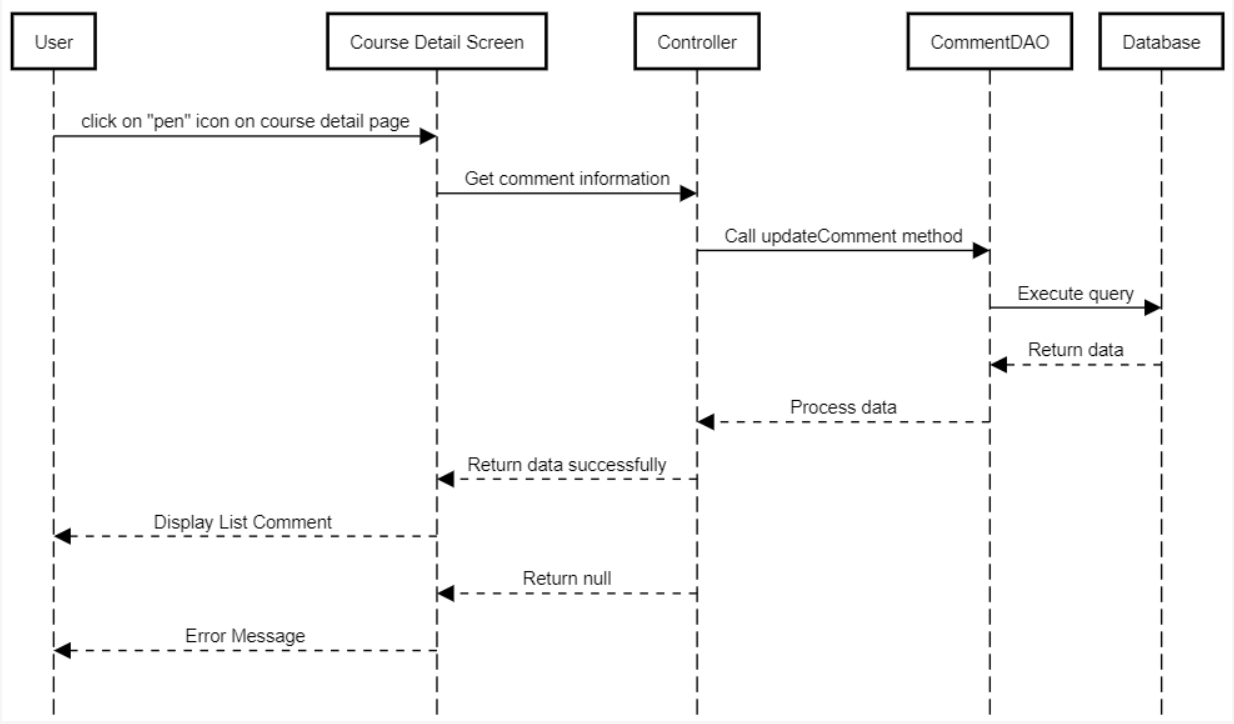
#### View list comment

****

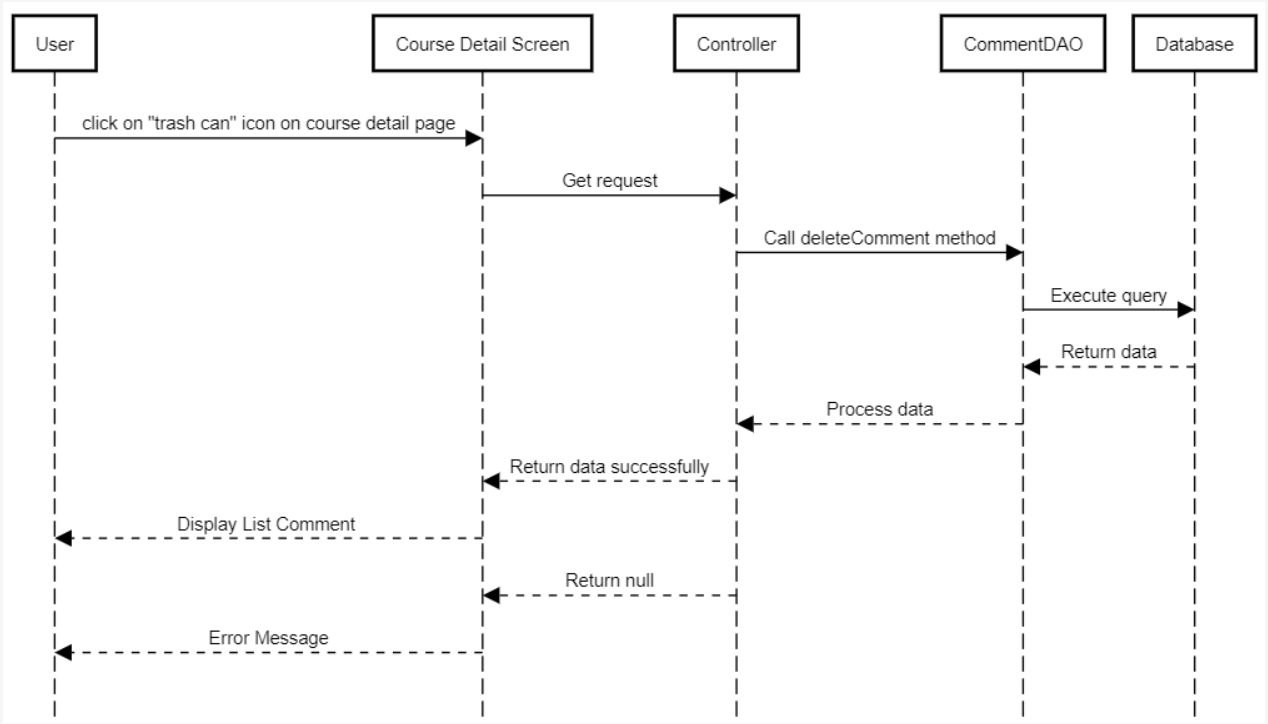
#### Add comment

****

#### Edit comment

****

#### Delete comment

****