|  |
| --- |
| **FPT- aptech computer education** |
| eProject Report |
| Course Online |
|  |
| |  |  |  | | --- | --- | --- | | **Group- Phạm Đức Thắng** | | **Contribution percentage** | | **Group Member** | Phạm Đức Thắng - TH2009031  Đặng Văn Tiến – TH2009015  Hà Duy Nhật - TH2009032  Lê Thành Đạt – TH2009017  Nguyễn Tiến Dũng – TH2009038 | 100%  100%  100%  100%  100% | | **Batch** | T2009M | | | **Instructor** | Đào Hồng Luyến | | | **Semester** | 3 | | |
|  |

|  |
| --- |
| - Hanoi, 3/2022 |

[**1.** **Acknowledgements** 5](#_Toc97766425)

[**1.** **Acknowledgements** 5](#_Toc97766426)

[**2.** **Introduction** 6](#_Toc97766427)

[**3.** **Problem Definition** 6](#_Toc97766428)

[**3.1.** **Problem Abstraction** 6](#_Toc97766429)

[**3.2.** **The Current System** 6](#_Toc97766430)

[**3.3.** **The Proposed System** 6](#_Toc97766431)

[**3.4.** **Boundaries of the System** 6](#_Toc97766432)

[**3.5.** **Development Environment** 6](#_Toc97766433)

[**4.** **Requirements and Business Flows** 7](#_Toc97766434)

[**4.1.** **Customer Requirements** 7](#_Toc97766435)

[**4.2.** **Activity Diagram** 7](#_Toc97766436)

[4.2.1 [Admin] Create Category 7](#_Toc97766437)

[4.2.2 [Admin] Edit Category 8](#_Toc97766438)

[4.2.3 [Admin] Delete Category 9](#_Toc97766439)

[4.2.4 [Admin] Manage Category 10](#_Toc97766440)

[4.2.5 [Admin] Create Course 10](#_Toc97766441)

[4.2.6 [Admin] Edit Course 10](#_Toc97766442)

[4.2.7 [Admin] Delete Course 10](#_Toc97766443)

[4.2.8 [Admin] Manage Course 10](#_Toc97766444)

[4.2.9 [Admin] Create Learners 10](#_Toc97766445)

[4.2.10 [Admin] Edit Learners 10](#_Toc97766446)

[4.2.11 [Admin] Delete Learners 10](#_Toc97766447)

[4.2.12 [Admin] Manage Learners 10](#_Toc97766448)

[4.2.13 [Admin] Statistics income 10](#_Toc97766449)

[4.2.14 [User] Login 10](#_Toc97766450)

[4.2.15 [User] Logout 10](#_Toc97766451)

[4.2.16 [User] Register 10](#_Toc97766452)

[4.2.17 [User] Filte course 10](#_Toc97766453)

[4.2.18 [User] Change password 10](#_Toc97766454)

[4.2.19 [User] Search course 10](#_Toc97766455)

[4.2.20 [User] Learn by lesson 10](#_Toc97766456)

[4.2.21 [User] Update profile 10](#_Toc97766457)

[4.2.22 [User] Buy the course 10](#_Toc97766458)

[4.2.23 [User] My course 10](#_Toc97766459)

[4.2.24 [User] Register course 10](#_Toc97766460)

[4.2.25 [User] Operation history 10](#_Toc97766461)

[4.2.26 [User] Account information 10](#_Toc97766462)

[4.2.27 [User] Completion schedule 10](#_Toc97766463)

[**4.3.** **Use Case Diagram** 10](#_Toc97766464)

[**4.4** **Use Case Specification** 12](#_Toc97766465)

[*<Write down all non-trivial use cases. This should reflect what you get when your team does the system analysis. Use the template to write the detailed specification for use cases>* 12](#_Toc97766466)

[4.4.1[Admin] Create category 12](#_Toc97766467)

[4.4.2 [Admin] Edit category 13](#_Toc97766468)

[4.4.3[Admin] Remove category 14](#_Toc97766469)

[4.4.4[Admin] Manage category 14](#_Toc97766470)

[4.4.5[Admin]Create course 15](#_Toc97766471)

[4.4.6 [Admin] Edit course 16](#_Toc97766472)

[4.4.7[Admin] Remove Course 17](#_Toc97766473)

[4.4.8[Admin] Manage category 18](#_Toc97766474)

[4.4.9[Admin] Create learners 19](#_Toc97766475)

[4.4.10 [Admin] Edit learners 20](#_Toc97766476)

[4.4.11[Admin] Remove learners 21](#_Toc97766477)

[4.4.12[Admin] Manage learners 22](#_Toc97766478)

[4.4.13 [Admin] Statistics income 22](#_Toc97766479)

[4.4.14 Login 23](#_Toc97766480)

[4.4.15 Logout 24](#_Toc97766481)

[4.4.16 Register 24](#_Toc97766482)

[4.4.17 Filte course 26](#_Toc97766483)

[4.4.18 Change password 26](#_Toc97766484)

[4.4.19 Search course 27](#_Toc97766485)

[4.4.20 Learn by lesson 28](#_Toc97766486)

[4.4.21 Update profile 29](#_Toc97766487)

[4.4.22 Buy the course 30](#_Toc97766488)

[4.4.23 My course 31](#_Toc97766489)

[4.4.24 Register course 32](#_Toc97766490)

[4.4.25 Operation history 32](#_Toc97766491)

[4.4.26 Account information 33](#_Toc97766492)

[4.4.27 Competion schedule 34](#_Toc97766493)

[**4.5 Other Concerns<Optional>** 35](#_Toc97766494)

[**5.** **Design** 36](#_Toc97766495)

[**5.1.** **System Architecture** 36](#_Toc97766496)

[**5.2.** **Class Diagram** 36](#_Toc97766497)

[**5.3.** **Class Diagram Explanation** 36](#_Toc97766498)

[**5.4.** **Sequence Diagram (Optional)** 36](#_Toc97766499)

[**5.5.** **Collaboration Diagram (Optional)** 37](#_Toc97766500)

[**5.6.** **State Diagram (Optional)** 37](#_Toc97766501)

[**5.7.** **Entity Relationship Diagram** 37](#_Toc97766502)

[**5.8.** **Database Design** 37](#_Toc97766503)

[**5.9.** **Algorithms (optional)** 42](#_Toc97766504)

[**5.10.** **Others (optional)** 42](#_Toc97766505)

[6. System Prototype 43](#_Toc97766506)

[6. Management and Project Planning 44](#_Toc97766507)

[6.1. Management Approach 44](#_Toc97766508)

[6.2. Project Plan 44](#_Toc97766509)

[6.3. Task Sheet 44](#_Toc97766510)

[6.4. Meeting Minutes (Optional) 44](#_Toc97766511)

[7. Checklists 45](#_Toc97766512)

[7.1. Check List of Validation 45](#_Toc97766513)

[7.2. Submission Checklist 45](#_Toc97766514)

[8. Screenshots 45](#_Toc97766515)

[9. Coding Convention 45](#_Toc97766516)

[10. User Guide 46](#_Toc97766517)

[11. Other Concerns<Optional> 47](#_Toc97766518)

[Appendix 48](#_Toc97766519)

[Glossary [Optional] 48](#_Toc97766520)

[References [Optional] 48](#_Toc97766521)

[Others<Optional> 48](#_Toc97766522)

# **Acknowledgements**

# **Introduction**

## **3.1. The urgency of the topic and the practical significance of the topic**

**- The urgency of the subject:**

Information technology is developing day by day and being applied in many fields

In particular, the Internet covers almost the whole world, creating favorable conditions in

people's life, work and study. The application of technology is widespread

widely changed in industries, especially in the field of learning, In fact, just with a device such as a phone, a computer with an internet connection, updating news, and viewing lessons can be done. Posting online is very easy, from the above reason I decided to build a website "**selling online courses**". Therefore, the design of the course series website system is extremely necessary for people today.

**- Practical significance:**

+ Can deploy and apply the course series website in practice, meeting the needs

learning of today's learners

+ Not only learners, students, and working people nowadays also pay great attention

to self-study Learn to improve your ability

+ Information technology is developing more and more, especially the Internet is

increasing day by day widely used. Widely used technology devices. Public

equipment. Technology using the Internet is also producing more and more

dvantages in online learning

line.

+ Therefore, the design of the course series website system is extremely necessary

for people people today.

+ Website will provide useful courses in many fields. Help learners have

more choices, more active in learning time and increasing self-study ability at all

levels place.

# **Problem Definition**

## **Problem Abstraction**

*< Write down the problem abstract. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## **The Current System**

*< Describe the current system\situation. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## **The Proposed System**

*< Describe the system under developed. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## **Boundaries of the System**

*< List the scope\boundaries of the System-under-developed. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## **Development Environment**

*<Describe the environment for system development. Include software and hardware requirements>*

# **Requirements and Business Flows**

*<In this section you will provide the requirements you developed against. This section should be useful, so only provide enough information (requirements, diagrams) to help with the implementation of the software. Focus on the “non-obvious” aspects. If you are dealing with a business process, activity diagrams are good. Making a set of requirements that is compact, simple and covers all the key aspects of the software can. Don’t do the work if you don’t need it!>*

## **Customer Requirements**

### 5.1.1 Customer Requirement Specification

- Customers can:

1. Learners

2. Filte course

3. Search

4. Login, register, logout

5. Update profile

6. Buy the course

7. My course

8. Account Information

9. My course

- Admin can:

1. Manage category

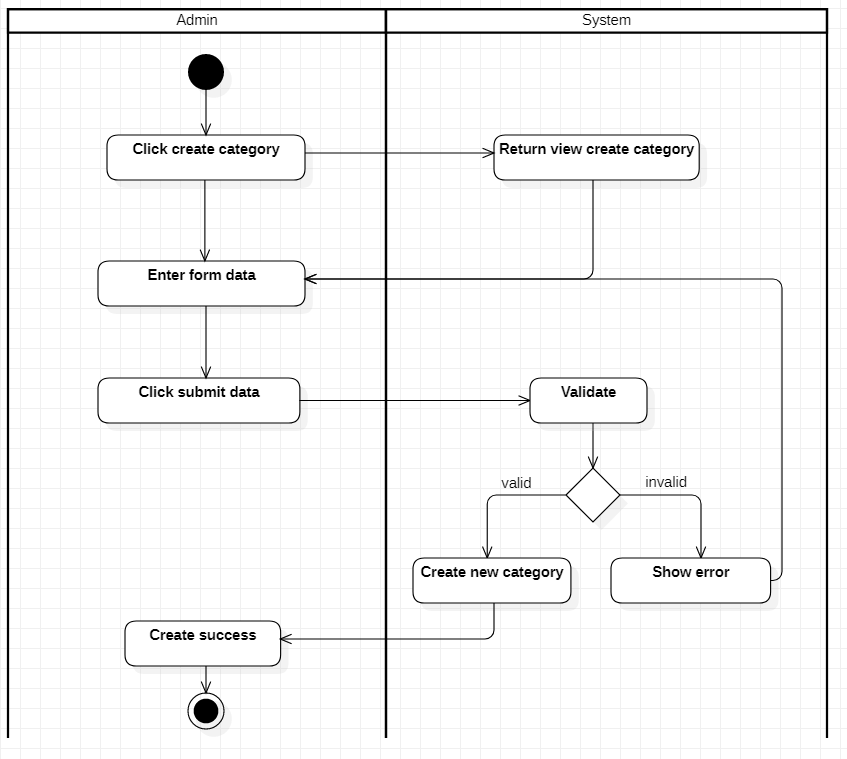
2. Manage learners

3. Manage course

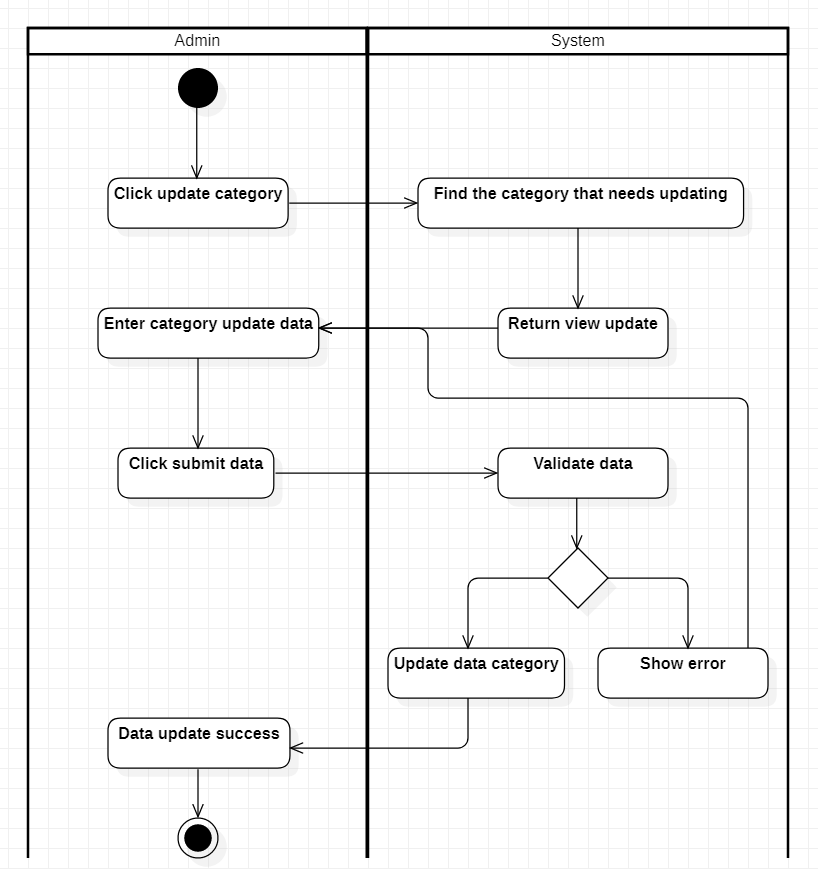
4. Manage statistics income

## **Activity Diagram**

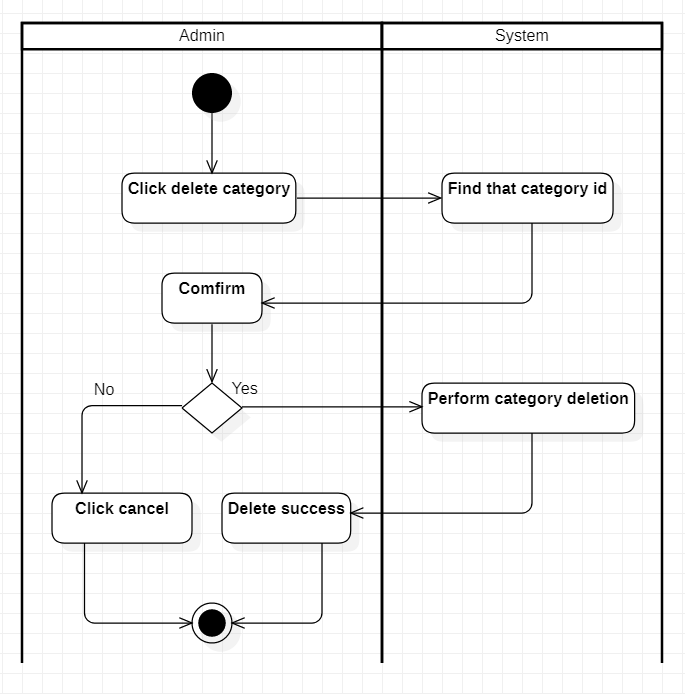
### 4.2.1 [Admin] Create Category



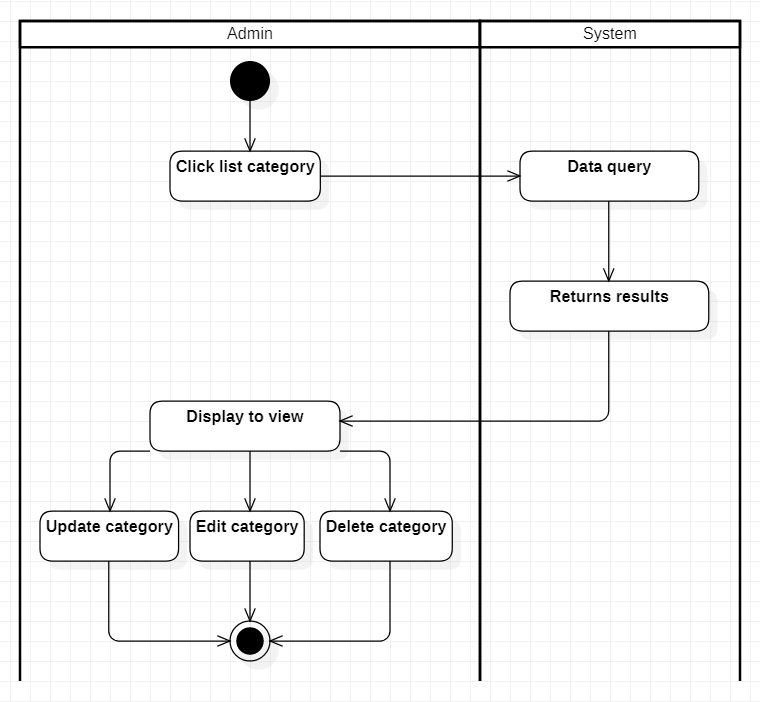
### 4.2.2 [Admin] Edit Category



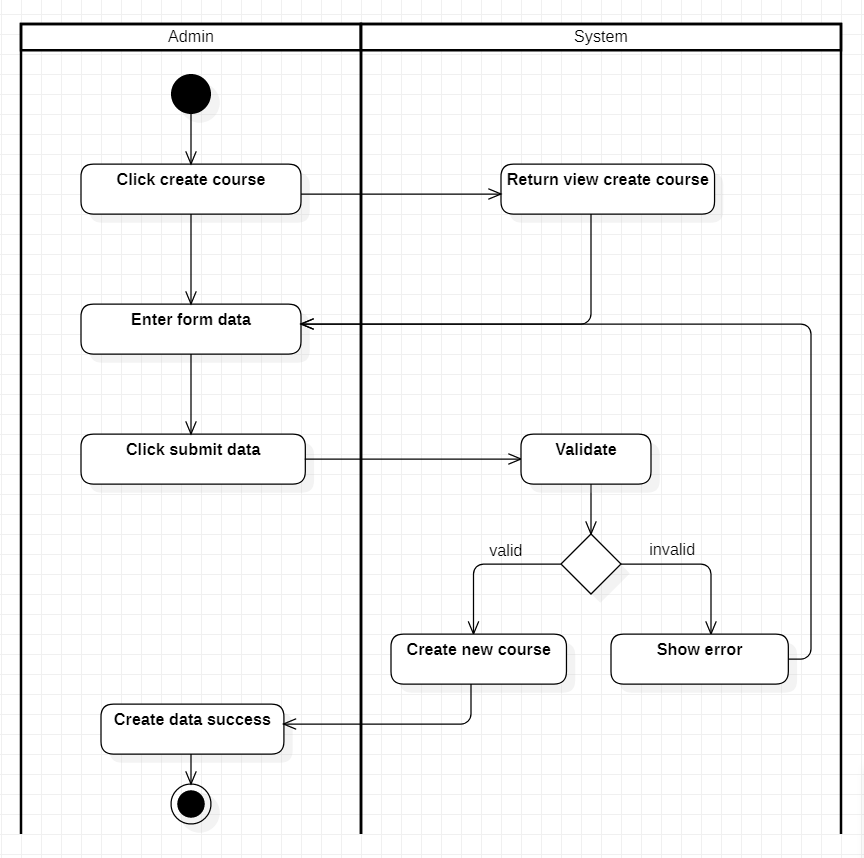
### 4.2.3 [Admin] Delete Category



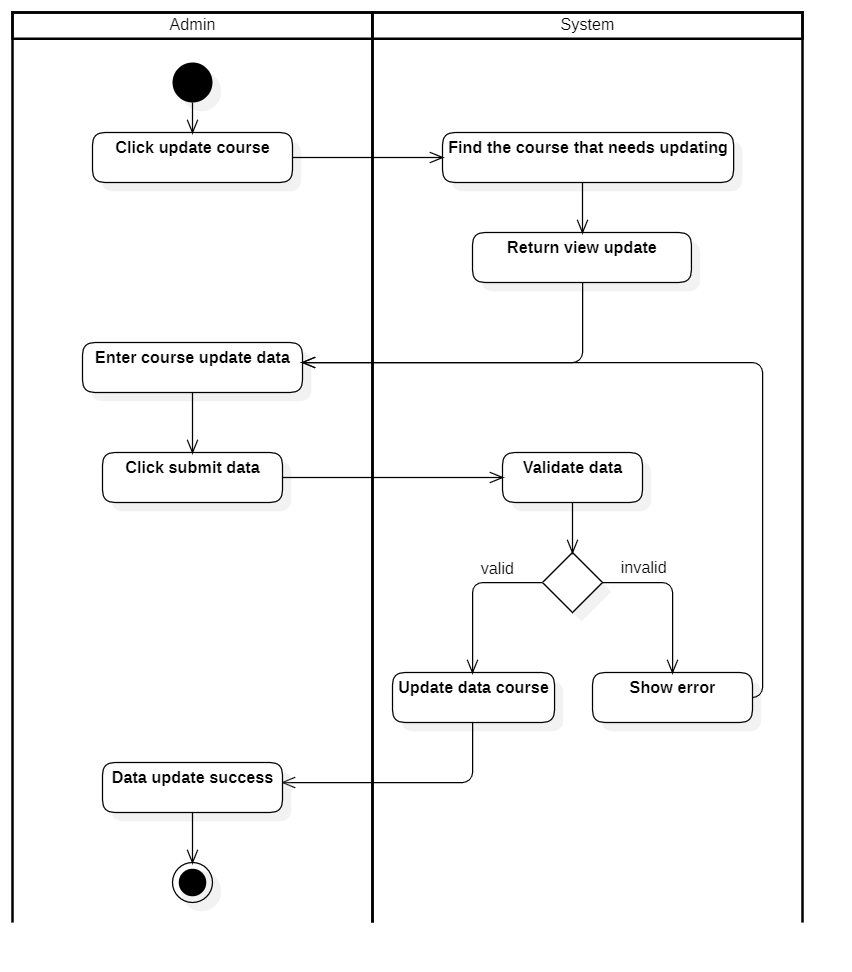
### 4.2.4 [Admin] Manage Category



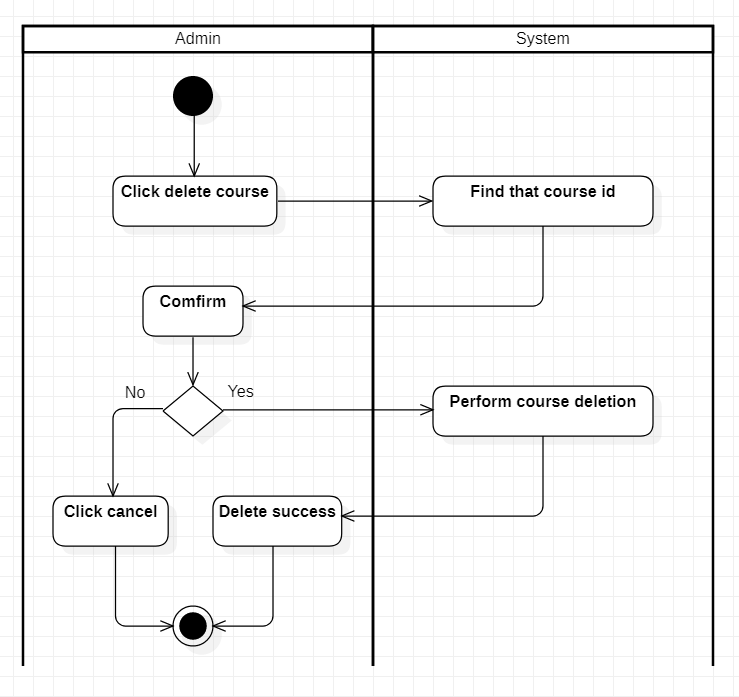
### 4.2.5 [Admin] Create Course



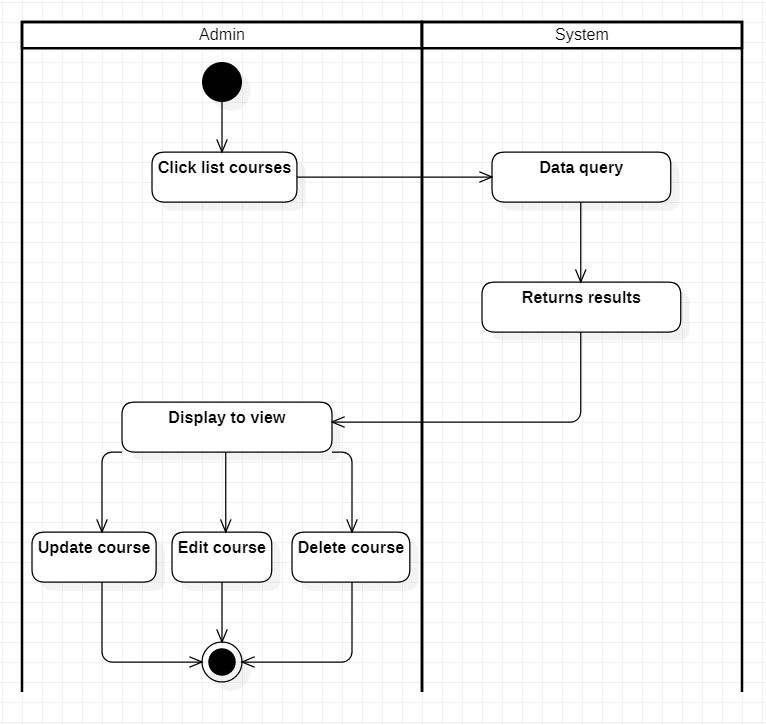
### 4.2.6 [Admin] Edit Course



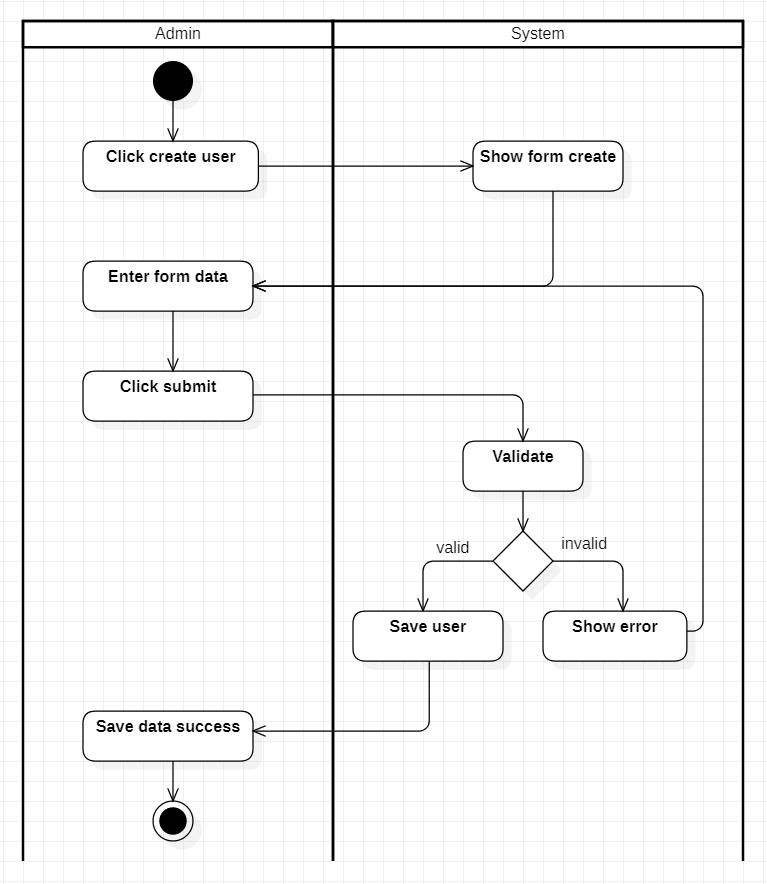
### 4.2.7 [Admin] Delete Course



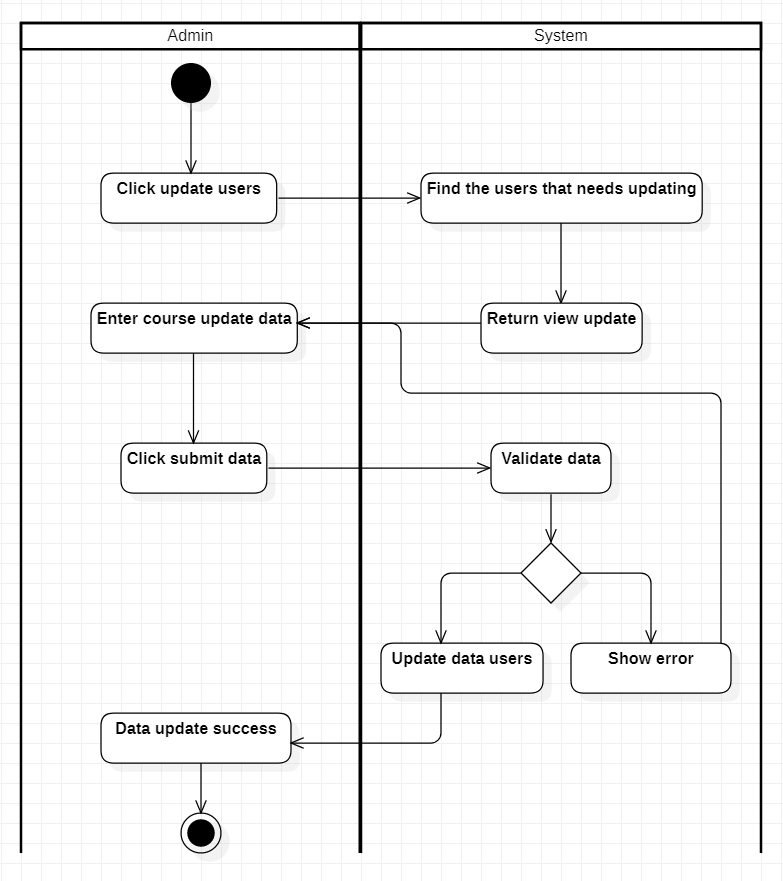
### 4.2.8 [Admin] Manage Course



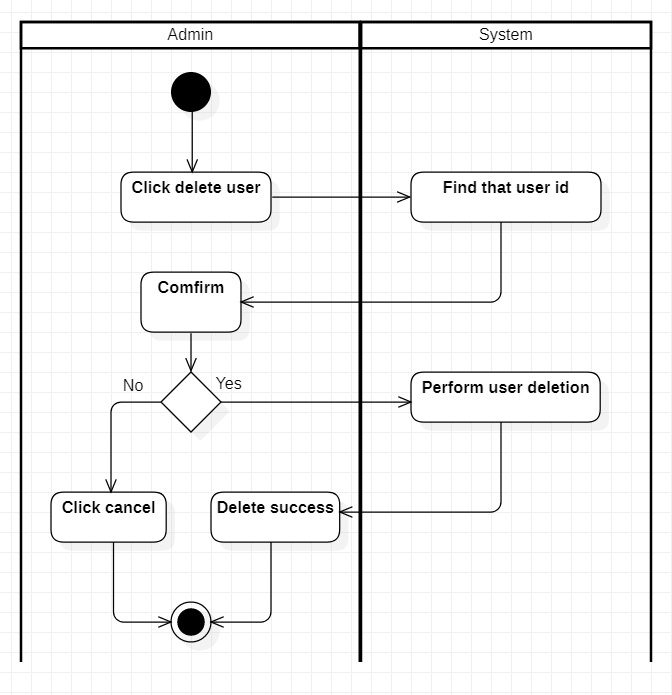
### 4.2.9 [Admin] Create Learners



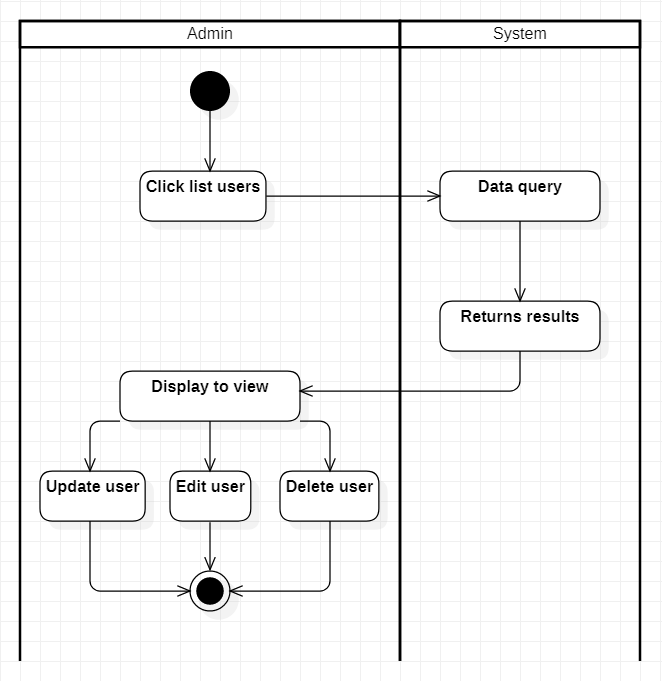
### 4.2.10 [Admin] Edit Learners



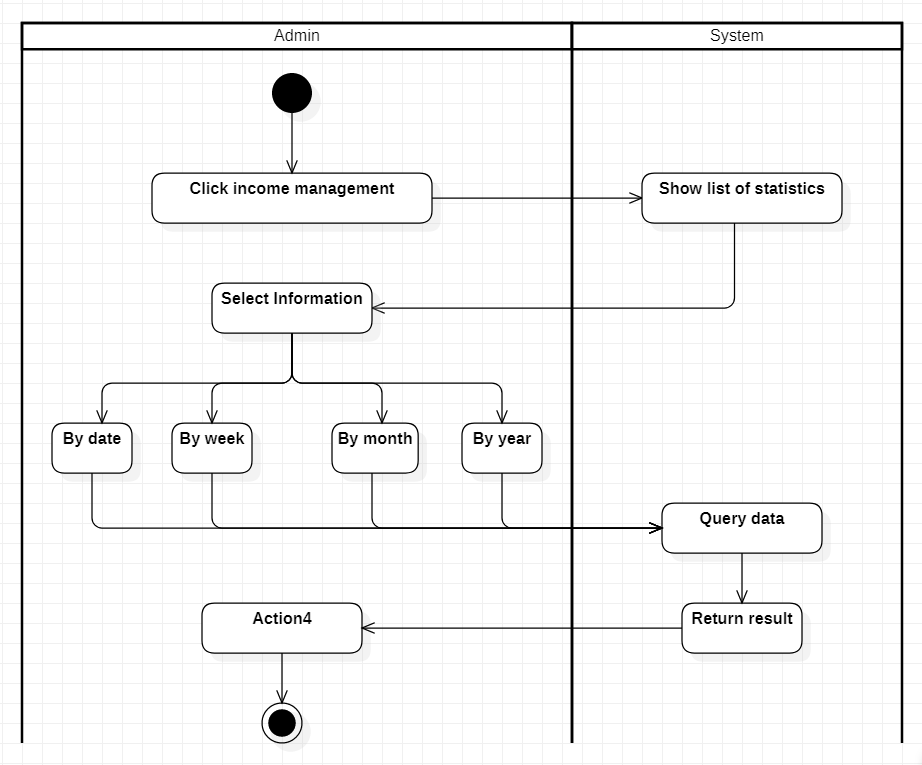
### 4.2.11 [Admin] Delete Learners



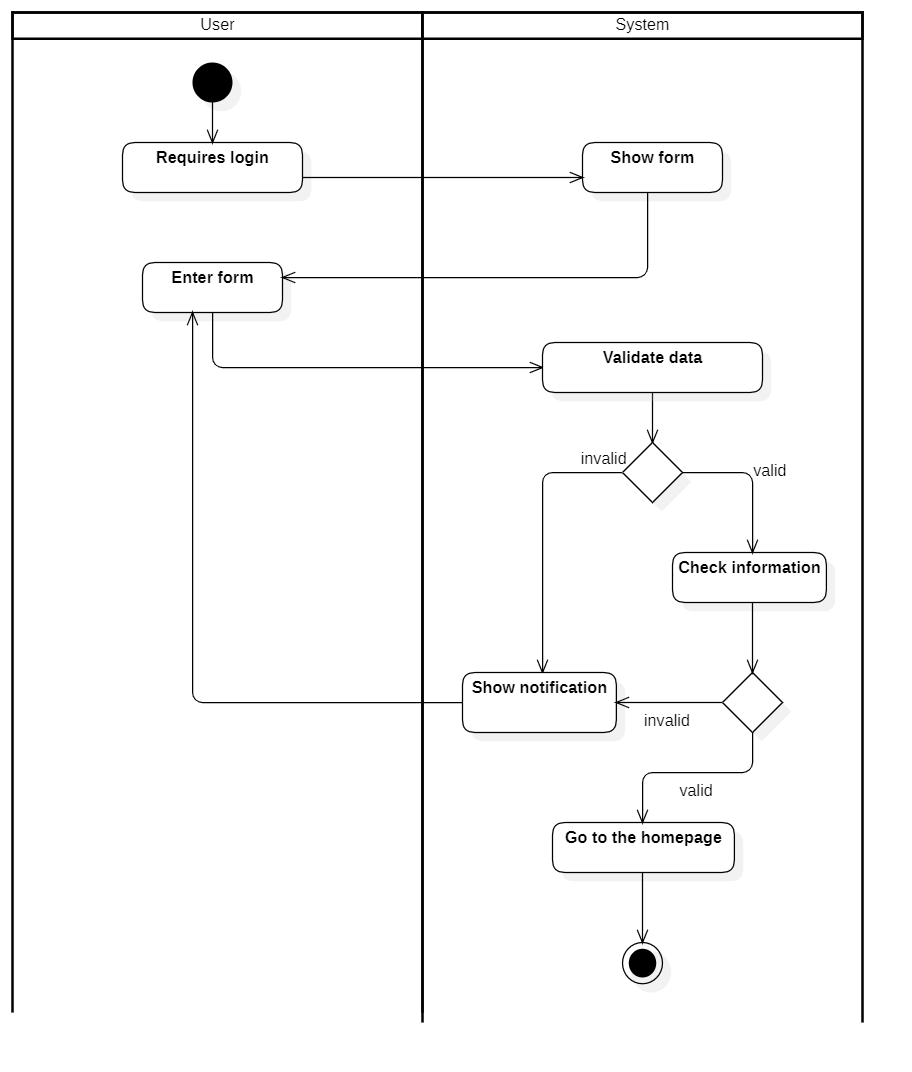
### 4.2.12 [Admin] Manage Learners



### 4.2.13 [Admin] Statistics income



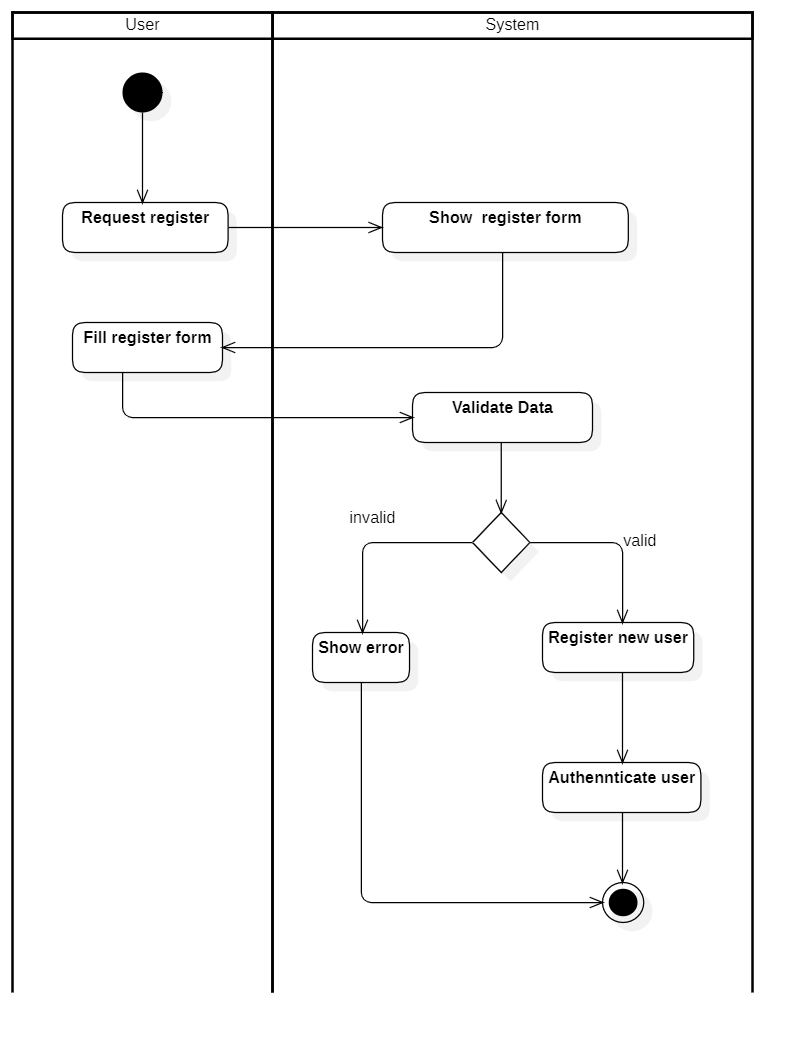
### 4.2.14 [User] Login



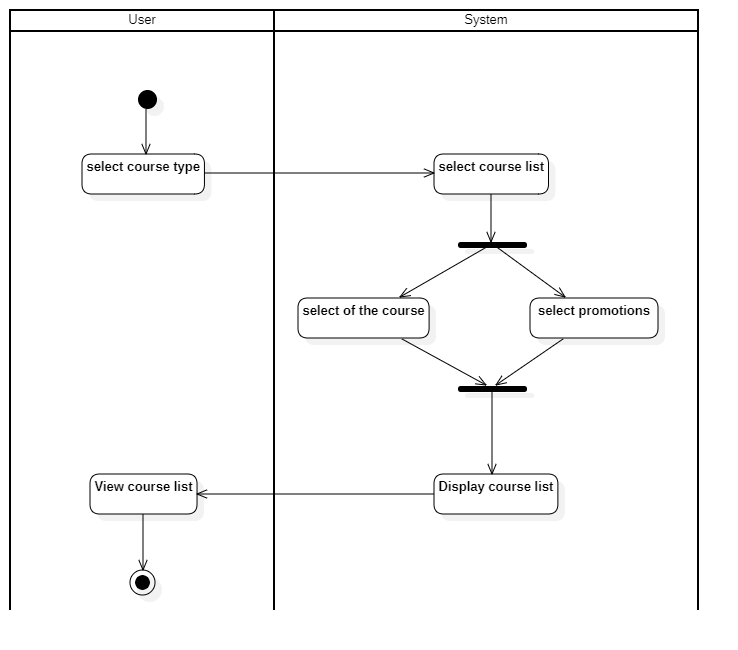
### 4.2.15 [User] Logout



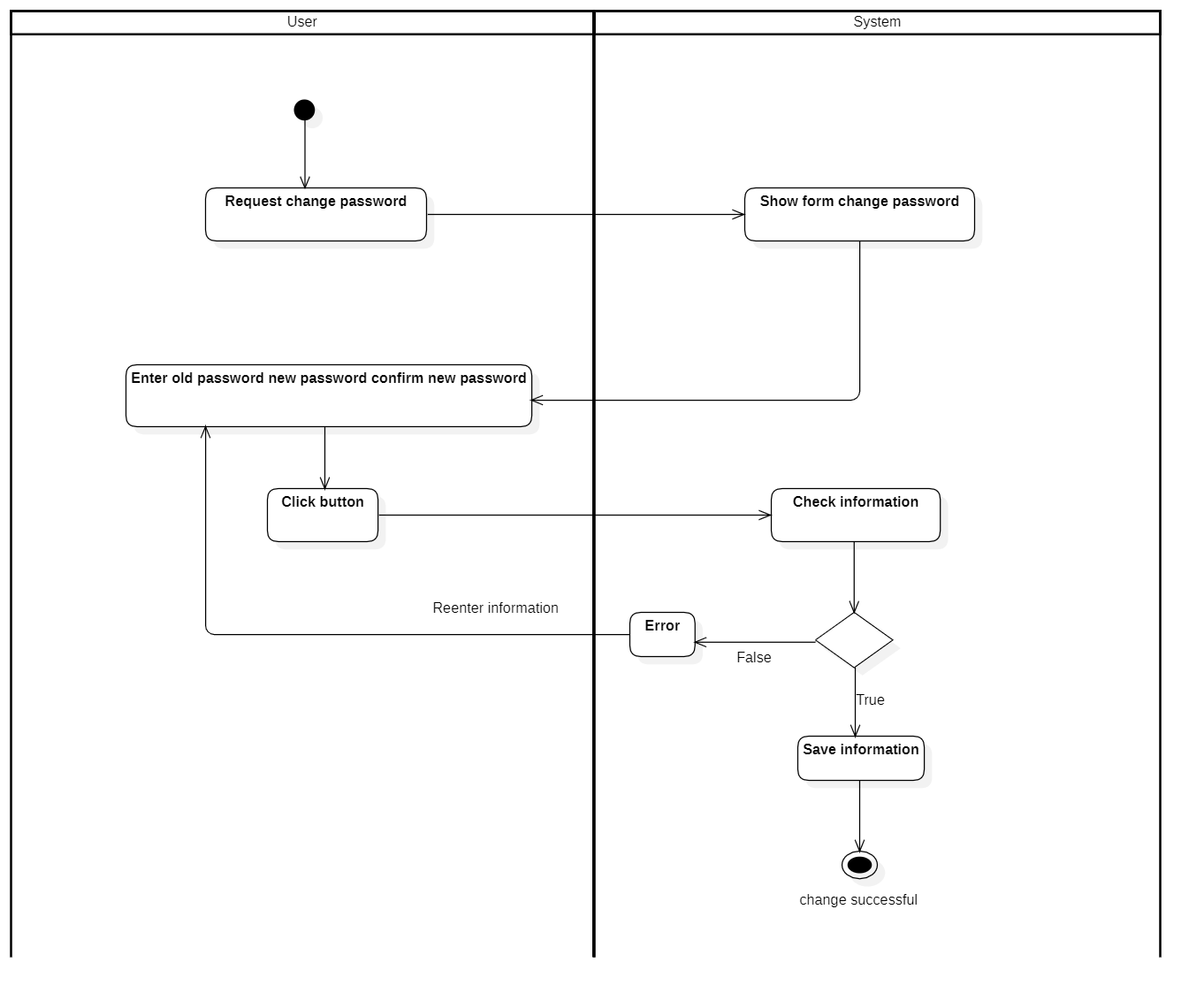
### 4.2.16 [User] Register



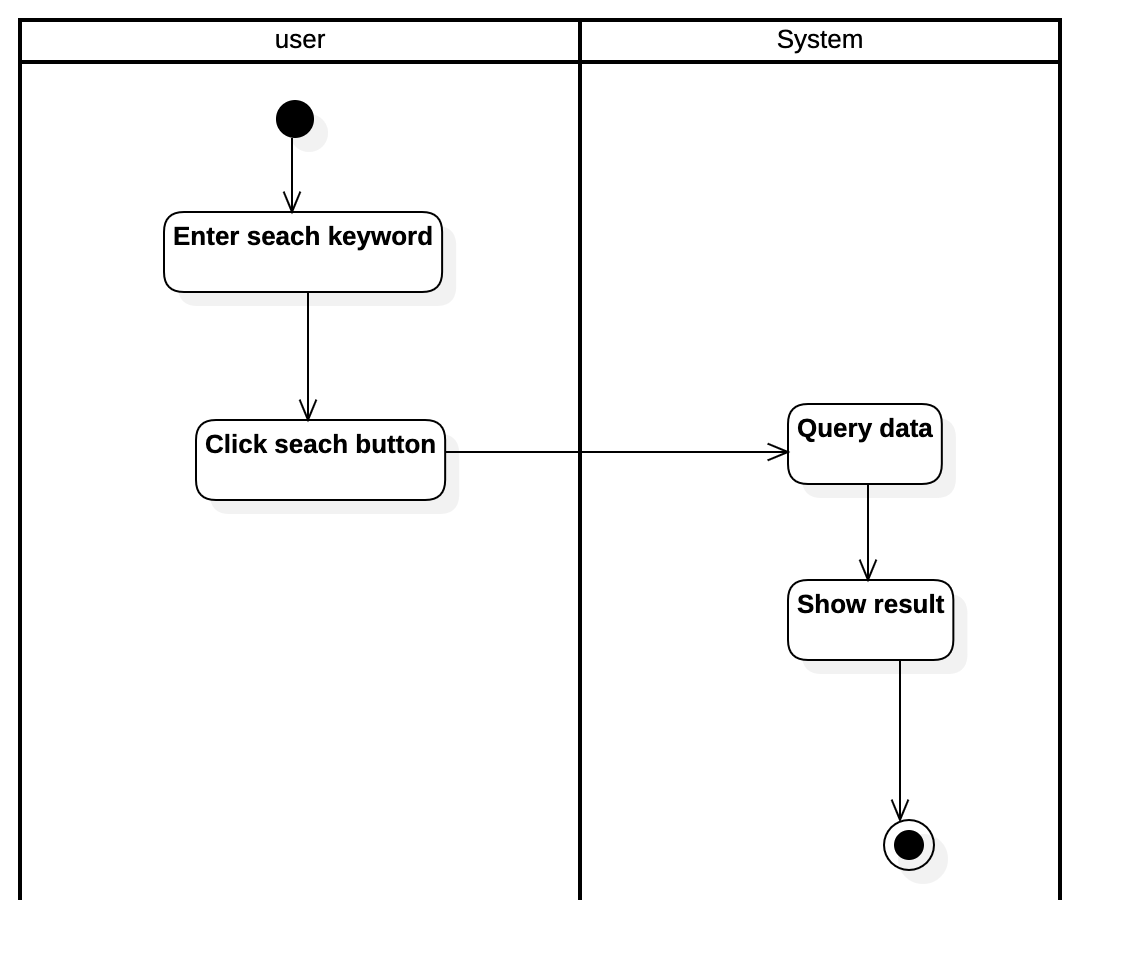
### 4.2.17 [User] Filte course



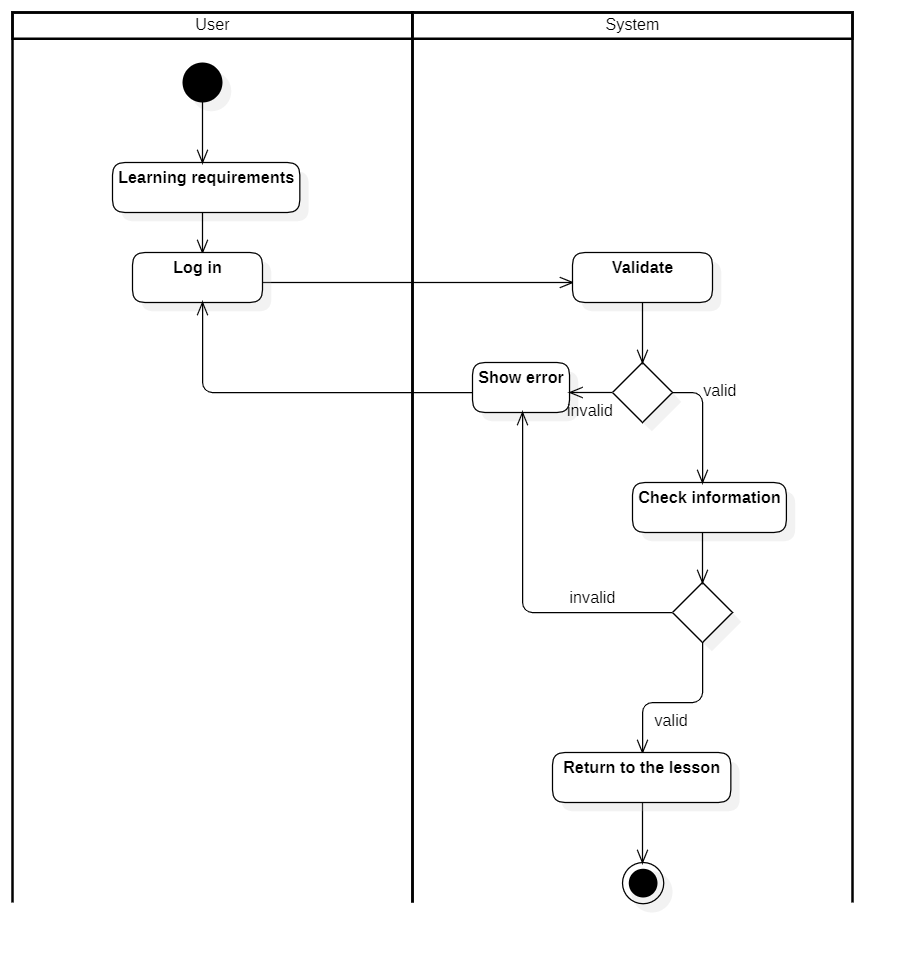
### 4.2.18 [User] Change password



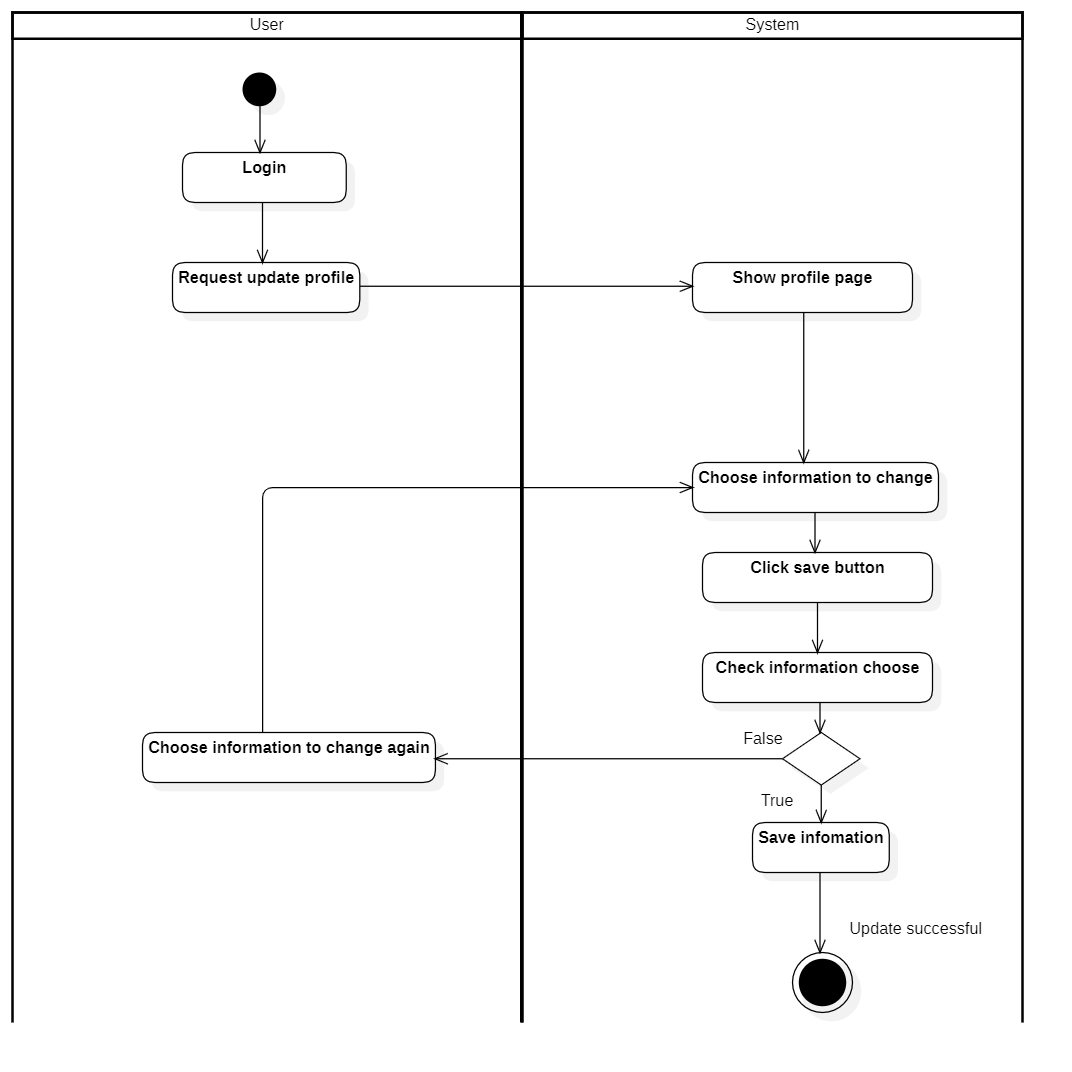
### 4.2.19 [User] Search course



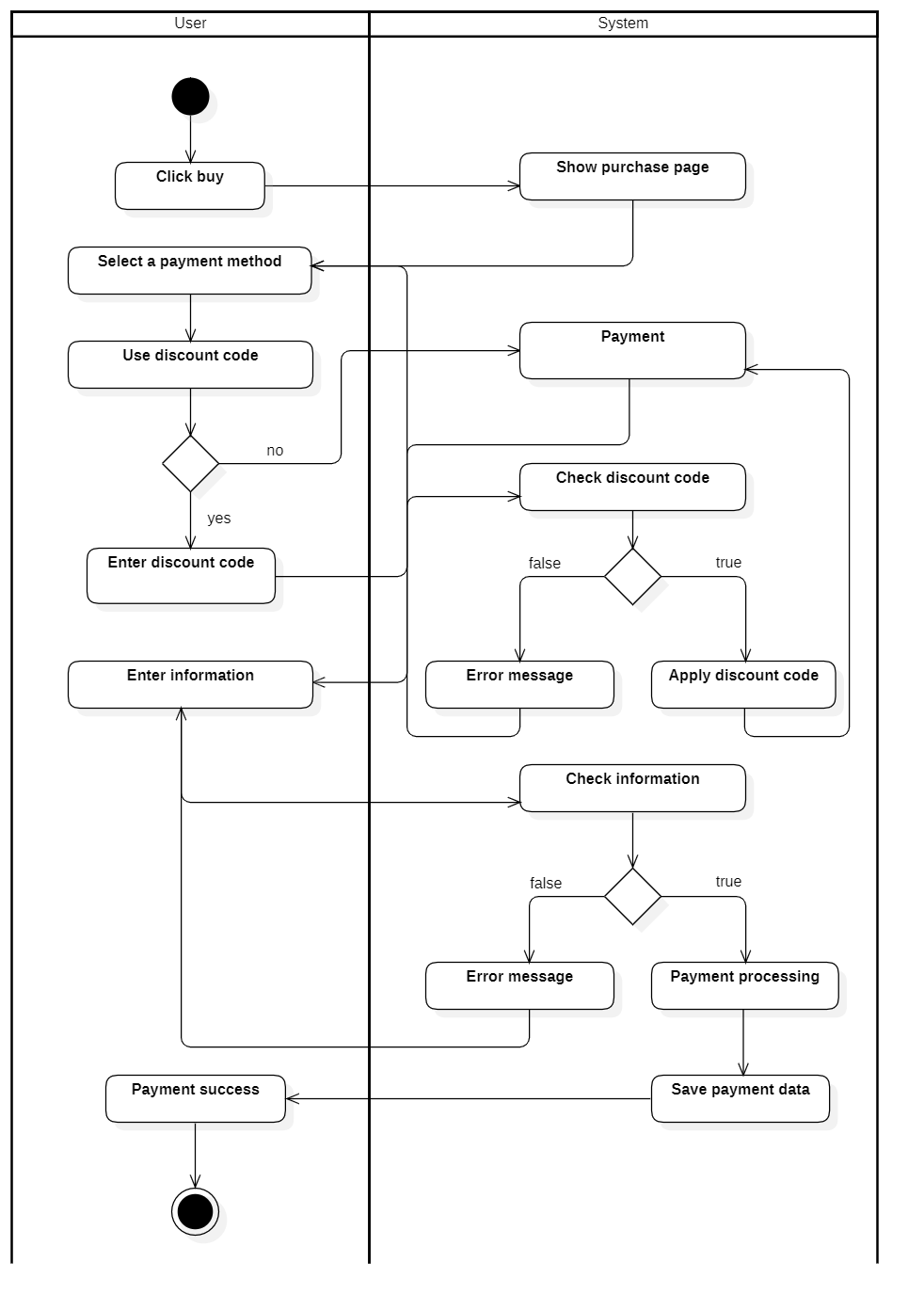
### 4.2.20 [User] Learn by lesson



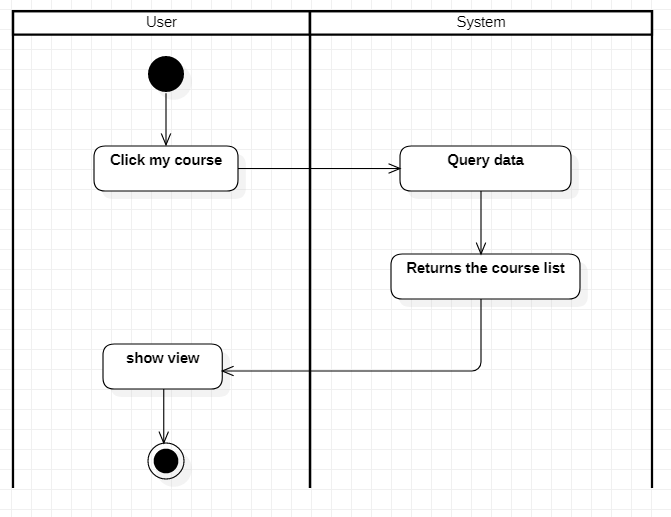
### 4.2.21 [User] Update profile



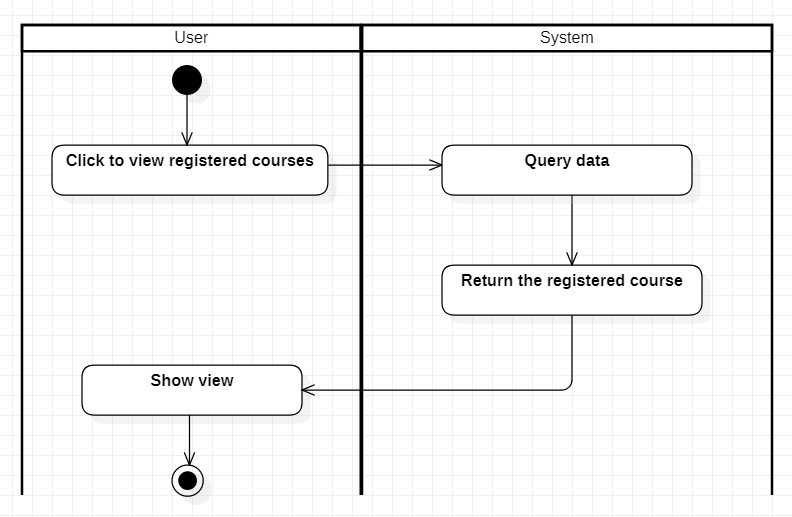
### 4.2.22 [User] Buy the course



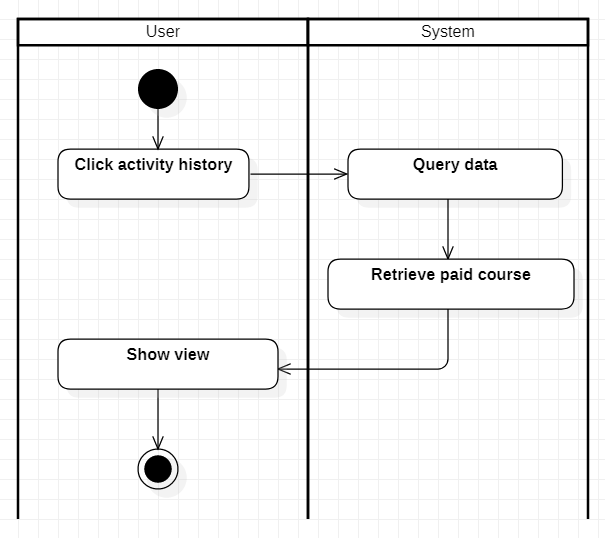
### 4.2.23 [User] My course



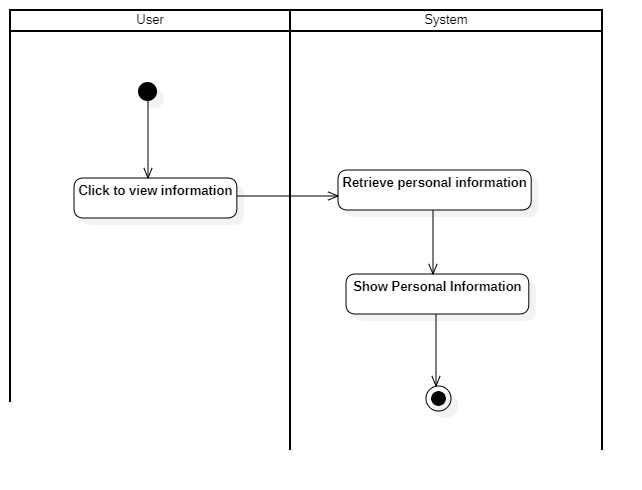
### 4.2.24 [User] Register course



### 4.2.25 [User] Recent Activity

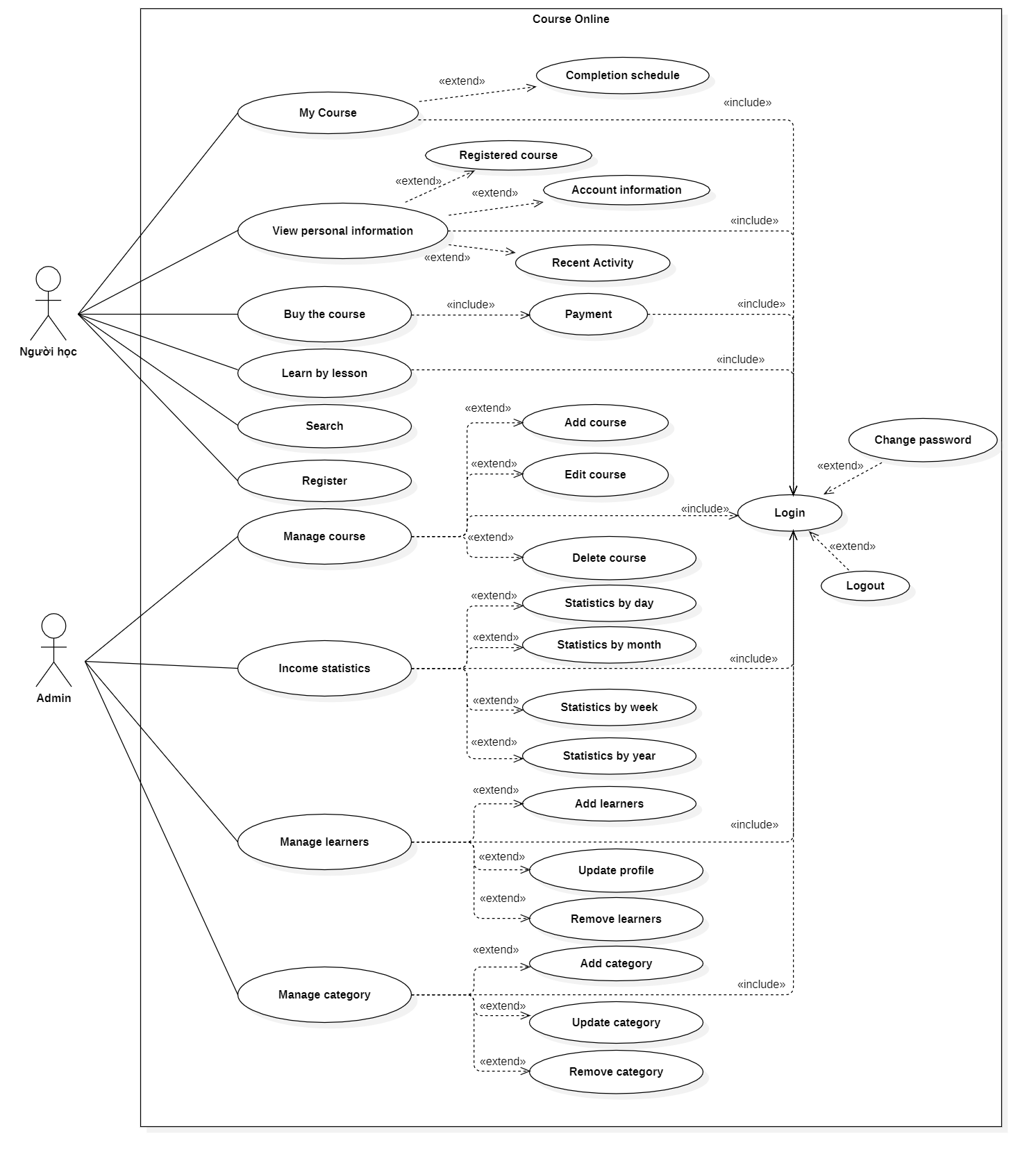


### 4.2.26 [User] Account information



## **Use Case Diagram**

4.3.1 Usecase drawing



## **Use Case Specification**

### 4.4.1[Admin] Create category

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Add category** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin add category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The usecase begins when actor click on “add new categroy” button.  2. The system show add new course form contains  Fields:  ✓ Name  ✓ ParentId  ✓ Image  3. Admin enter information into form  4. The system check the validity of the information.  5. The system will save course and  6. System display successful message |
| **Alternative Flows** | A1.Name is null:  If in step 4 of basic stream agent do not enter name of categroy, image, parentId then:  1. Error message  2. Go back to the form and re-enter |
| **Exit Options** | - Admin click on “Cancel” button |
| **Special Requirements** |  |
| **Post Condition** | - A new course is create |

### 4.4.2 [Admin] Edit category

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Edit category** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin edit category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The use case start when the admin selects the category he wants to edit, then clicks the “Edit Category” button.  2. The system loads information of the list into the form, including:  ✓ Name.  3. Admin renames category to form.  4. The system checks the validity of information.  5. Save catalog information to the database  6. The system displays a successful message. |
| **Alternative Flows** | A1.Name is null:  If in step 4 of the underlying stream agent the name of the category is not entered then:  1. System display error message.  2. Use Case continues at step 3 of Basic Flow, enter the name of the category. |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A category updated. |

### 4.4.3[Admin] Remove category

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Remove category** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin remove category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. System display list list.  2. Admin has selected the category you want to remove then click the “remove” button.  3. The system confirms whether the admin wants to remove it or not  4. The system removes the category from the database.  5. The system displays a success message. |
| **Alternative Flows** |  |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A category is removed. |

### 4.4.4[Admin] Manage category

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Manage category** |
| **Actors** | Admin |
| **Brief Description** | - Use case allows the admin manage category contain: add category,edit category, remove category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. This use case begin when admin click on “Manage category” menu.  2. System display list all categy in database and buttons: “add new category” ,”edit category”, ”remove category”.  3.  -If actor click “add category” use case “**add category**” is executed.  -If actor click “edit category” use case “**edit category**” is executed.  -If actor click “remove category” use case “**remove category**” is executed. |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Category is added,edited or removed |

### 4.4.5[Admin]Create course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Add course** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin add course |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The usecase begins when actor click on “add new course” button.  2. The system show add new course form contains  Fields:  ✓ Name  ✓ Category Select  ✓ Price  ✓ Image  ✓ Description  ✓ Content  ✓ Benefits  ✓ Document  ✓ Status  3. Admin enter information into form  4. The system check the validity of the information.  5. The system will save course and  6. System display successful message |
| **Alternative Flows** | A1.Name is null:  If in step 4 of basic stream agent do not enter name of course, image, content, description, benefits then:  1. Error message  2. Go back to the form and re-enter |
| **Exit Options** | - Admin click on “Cancel” button |
| **Special Requirements** |  |
| **Post Condition** | - A new course is create |

### 4.4.6 [Admin] Edit course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Edit course** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin edit course |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The use case start when the admin selects the course he wants to edit, then clicks the “Edit Course” button.  2. The system loads information of the list into the form, including:  ✓ Name  ✓ Category Select  ✓ Price  ✓ Image  ✓ Description  ✓ Content  ✓ Benefits  ✓ Document  ✓ Status  3. Admin enter new name, new image, ...  4. The system checks the validity of information.  5. Save course information to the database  6. The system displays a successful message. |
| **Alternative Flows** | **Leave the fields empty:**  If in step 3 of the basic thread agent do not enter the name, image, description, benefits, price of the course then:  1. Error message.  2. Use case continues at step 3 of Basic Flow, continue to enter the fields. |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A course updated. |

### 4.4.7[Admin] Remove Course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Remove course** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin remove course |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. System display list list.  2. Admin has selected the course you want to remove then click the “remove” button.  3. The system confirms whether the admin wants to remove it or not  4. The system removes the category from the database.  5. The system displays a success message. |
| **Alternative Flows** |  |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A category is removed. |

### 4.4.8[Admin] Manage course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Manage category** |
| **Actors** | Admin |
| **Brief Description** | - Use case allows the admin manage category contain: add category,edit category, remove category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. This use case begin when admin click on “Manage course” menu.  2. System display list all course in database and buttons: “add new course” ,”edit course”, ”remove course”.  3.  -If actor click “add course” use case “**add course**” is executed.  -If actor click “edit course” use case “**edit course**” is executed.  -If actor click “remove caurse” use case “**remove caurse**” is executed. |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Course is added,edited or removed |

### 4.4.9[Admin] Create learners

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Add learners** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin add category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The usecase begins when actor click on “add new learners” button.  2. The system show add new learners form contains  Fields:  ✓ UserName  ✓ Password  ✓ Email  ✓ Phone  ✓ Address  ✓ Avatar  ✓ Birthday  ✓ Gender  3. Admin enter information into form  4. The system check the validity of the information.  5. The system will save course and  6. System display successful message |
| **Alternative Flows** | A1.Name is null:  If in step 4 of basic stream agent do not enter userName of password, email, phone, address, avatar, birthday, gender then:  1. Error message  2. Go back to the form and re-enter |
| **Exit Options** | - Admin click on “Cancel” button |
| **Special Requirements** |  |
| **Post Condition** | - A new learners is create |

### 4.4.10 [Admin] Edit learners

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Edit course** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin edit category |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. The use case start when the admin selects the learners he wants to edit, then clicks the “Edit Learners” button.  2. The system loads information of the list into the form, including:  ✓ UserName  ✓ Password  ✓ Email  ✓ Phone  ✓ Address  ✓ Avatar  ✓ Birthday  ✓ Gender  ✓ Status  ✓ RoleId  3. Admin changes the fields in the form.  4. The system checks the validity of information.  5. Save catalog information to the database  6. The system displays a successful message. |
| **Alternative Flows** | **Leave the fields empty:**  If in step 3 of the basic thread agent do not enter the userName, password, email, phone, address, avatar, birthday, gender then:  1. Error message.  2. Use case continues at step 3 of Basic Flow, continue to enter the fields. |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A learners updated. |

### 4.4.11[Admin] Remove learners

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Remove learners** |
| **Actors** | Admin |
| **Brief Description** | - Usecase allows the admin remove learners |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. System display list list.  2. Admin has selected the learners you want to remove then click the “remove” button.  3. The system confirms whether the admin wants to remove it or not  4. The system removes the category from the database.  5. The system displays a success message. |
| **Alternative Flows** |  |
| **Exit Options** | - Admin click on “Cancel” button. |
| **Special Requirements** |  |
| **Post Condition** | - A learners is removed. |

### 4.4.12[Admin] Manage learners

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Manage learners** |
| **Actors** | Admin |
| **Brief Description** | - Use case allows the admin manage learners contain: add learners, edit learners, remove leraners |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. This use case begin when admin click on “Manage laearners” menu.  2. System display list all laerners in database and buttons: “add new learners” ,”edit learners”, ”remove learners”.  3.  -If actor click “add learners” use case “**add learners**” is executed.  -If actor click “edit learners” use case “**edit learners**” is executed.  -If actor click “remove learners” use case “**remove learners**” is executed. |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Learners is added,edited or removed |

### 4.4.13 [Admin] Statistics income

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Statistics income** |
| **Actors** | Admin |
| **Brief Description** | - Use case allows the admin statistics income. |
| **Preconditions** | - Admin must be logged in |
| **Basic Flow of Events** | 1. This use case starts when the agent clicks the "Earnings Statistics" menu.  2. Enter selective statistics by day, week, month, year.  3. Income statistics system for that period  4. The system displays the results for the admin |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Income is display |

### 4.4.14 Login

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Login** |
| **Actors** | Admin, user |
| **Brief Description** | - Actors use this use case to login to system |
| **Preconditions** |  |
| **Basic Flow of Events** | 1. Actor clicks the "Login" button in the web page.  2.The system displays a "login" form containing  Field :  ✓ Username.  ✓ Password.  3. Actor enters a username and password into the form.  4. The system checks the validity of the username and password.  5. Go to homepage |
| **Alternative Flows** | If in the step 4 Basic Flow,Customer enter invalid user name and/or password then:  1.System display error message.  2.The use case resumes at step 3 of Basic Flow. |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Actors login to system |

### 4.4.15 Logout

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Logout** |
| **Actors** | Admin, user |
| **Brief Description** | - Actors use this usecase to log out of the system |
| **Preconditions** | - Actors must be logged in |
| **Basic Flow of Events** | 1. Actor clicks the "Logout" button in the web page.  2. System confirms if admin wants to log out  3. Go back to the login page |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Actors log out of the system |

### 4.4.16 Register

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Register** |
| **Actors** | Admin, user |
| **Brief Description** | - Use case allows the customer register to system |
| **Preconditions** |  |
| **Basic Flow of Events** | 1. The use case begins when customer click on “Register “ button in website.  2.The system show register form contains  Fields:  ✓ UserName  ✓ Password  ✓ Email  ✓ Phone  ✓ Address  ✓ Avatar  ✓ Birthday  ✓ Gender  3. Customer enter information into form.  4.The system checks the validity of the information.  5. Save information of customer to database  6.System display successful register message. |
| **Alternative Flows** | **A1.Exist use name:**  If in step 4 of the basic flow customer enter a usename existed in the database then:  1. System display error message.  2. The use case resumes at step 3 of Basic Flow enter other username.  **A2.Email not valid:**  If in step 4 of the basic flow customer enter a usename existed in the database then:  1. System display error message.  2. The use case resumes at step 3 of Basic Flow re-enter email address.  **A3.(Full name, password, Address, BirthDay) is null:**  If in step 4 of the basic flow customer not enter (Full name, password, Address, BirthDay) then:  1. System display error message.  2. The use case resumes at step 3 of Basic Flow enter requiment information. |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Customer have a new account in system. |

### 4.4.17 Filte course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Filte course** |
| **Actors** | User |
| **Brief Description** | - Use case to filter courses from categories |
| **Preconditions** |  |
| **Basic Flow of Events** | 1. User selects required category  2. Then the system returns a list of courses  3. Users choose to select by price, by promotion  4. On-demand course processing and return system |
| **Alternative Flows** | - When there is no such course as filtered, it returns a message that the course is not found |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Show courses to users when filtered |

### 4.4.18 Change password

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Change password** |
| **Actors** | User, Admin |
| **Brief Description** | - Actors use this use case to chage password login to system |
| **Preconditions** | - The Actors must be logged into system before this use case begins. |
| **Basic Flow of Events** | 1. The use case begins when Actors click on “Change password” menu on the website.  3.The system display ” change password” form contains Fields:  ✓ Old password.  ✓ New password.  ✓ Confirm new password.  4. Actors enter infomation in form.  5. Actors click on “save” button.  6.The system checks the validity of the information.  7.System save information to data base.  8. System display successful message. |
| **Alternative Flows** | **A1. Old password is incorrect**  - If in Basic Procedure step 6, the Customer enters the old  password incorrectly:  1. System display error message.  2. The use case continues at step 4 of the Basic Flow,  re-entering the old password.  **A2. Confirm old password new password does not match.**  - If in step 6 Basic Procedure, the Actor enters the new password and confirms the new password is not the same:  1. System display error message.  2. The use case continues at step 4 of the New Password Reconfirmation Basic Flow. |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Customer's password is changed |

### 4.4.19 Search course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Change password** |
| **Actors** | User |
| **Brief Description** | - Users use this use case to search courses |
| **Preconditions** |  |
| **Basic Flow of Events** | 1. The system displays the course list.  2. The system displays search options  3. Users can choose the information they want to search  4. Executable user clicks "Search" button.  5. The system displays the list of suitable courses |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - The user has a matching list of courses |

### 4.4.20 Learn by lesson

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Learn by lesson** |
| **Actors** | User |
| **Brief Description** | - Users can use this use case to learn |
| **Preconditions** | - Learners must be logged in |
| **Basic Flow of Events** | 1. First, the user presses the "start learning" button of the course  2. The system will return the user's chapters of the course  3. In the chapters are lessons that the user can choose  4. Users can choose to learn by video, learn by image or document  5. When the user selects the lesson type, the system will clearly display the left side |
| **Alternative Flows** | **A1. When the user clicks "start learning" but has not paid:**  1. Ask users to pay  2. Paying user returns to the payment page |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Learners can study according to their chosen lesson types |

### 4.4.21 Update profile

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Update profile** |
| **Actors** | User, Admin |
| **Brief Description** | - Actors can use this use case to correct personal information |
| **Preconditions** | - Actors must be logged in |
| **Basic Flow of Events** | 1. The use case begins when customer click on “Update information” menu on the website.  2.The system load personal information of customer.  3.The system display information into ” Update” form contains Fields:  ✓ FullName  ✓ Address  ✓ Birthday  ✓ Gender  ✓ Phone  ✓ Email  4. Customer change information in form.  5. Customer click on “Save” button.  6. The system checks the validity of the information.  7. System save information to data base.  8. System display successful message. |
| **Alternative Flows** | **A1. Customer enters invalid information:**  1. System display error message.  2. The Use Case continues in step 4 of the Basic Process. |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Learners can study according to their chosen lesson types |

### 4.4.22 Buy the course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Buy the course** |
| **Actors** | User |
| **Brief Description** | - Learners can use this use case to purchase the course |
| **Preconditions** | - Learners must be logged in |
| **Basic Flow of Events** | 1. When learners click the "**buy course**" button  2. The system will redirect to the course purchase page  3. The system provides payment methods for users  4. User selects payment method  5. User clicks "**continue**"  6. The system will return a form to enter information  7. User enters information  8. User clicks "send" button  9. The system will check the accuracy of user input  10. The system will process the payment  11. Then the information into the database  12. Show success message to buyer |
| **Alternative Flows** | **A1. In case the user has a discount code:**  1. In the step of choosing a payment method  2. User can enter discount code  3. The system will check the accuracy of the discount  code  4. The system will apply the discount code and deduct  money  **A1.1 In case the discount code is wrong**  1. The system will report an error to the user  2. The usecase resumes at step 4 of Basic Flow.  **A2. In case of incorrect information input**  1. The system will report an error  2. The usecase resumes at step 7 of Basic Flow. |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Users can purchase courses and save history in the database |

### 4.4.23 My course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **My course** |
| **Actors** | User |
| **Brief Description** | - Users can view the courses they have registered for |
| **Preconditions** | - User must be logged in |
| **Basic Flow of Events** | 1. When learners click the "my course" button  2. The system will check the existence of user  3. The system will retrieve the courses and registered users  4. Return the list of courses of that user |
| **Alternative Flows** | **Haven't signed up for any courses yet:**  1. The system will display a message to the user  2. The system creates a button to link to the course list  page for users to register for the course |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Returns the list of courses the user has registered for |

### 4.4.24 Register course

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Register course** |
| **Actors** | User |
| **Brief Description** | - User can view booked but not paid course |
| **Preconditions** | - User must be logged in |
| **Basic Flow of Events** | 1. The use case begins when user click on “Register  course” menu on the website.  2. The system will query the database and then get the  user's information  3. The system will retrieve all the courses that the user has  attended  4. The system will return a view of all the courses you have  ordered |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Retrieve the registered course |

### 4.4.25 Recent Activity

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Recent Activity** |
| **Actors** | User |
| **Brief Description** | - This use case allows user to view course purchase  history, course enrollment history |
| **Preconditions** | - User must be logged in |
| **Basic Flow of Events** | 1. The use case begins when user click on “Operation  history” menu on the website.  2. The system will query the database to get your  information  3. The system will retrieve the transaction history  4. The system will list and display it to the view |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Get the user's transaction history |

### 4.4.26 Account information

|  |  |
| --- | --- |
| **USE CASE SPECIFICATION** | |
| **Use-case Name** | **Account information** |
| **Actors** | User, Admin |
| **Brief Description** | - The use case is used to display user account information |
| **Preconditions** | - Actors must be logged in |
| **Basic Flow of Events** | 1. The use case starts when actors click on the "Account Info" menu on a website.  2. The database query system retrieves actors' account information  3. Account information display system |
| **Alternative Flows** |  |
| **Exit Options** |  |
| **Special Requirements** |  |
| **Post Condition** | - Display information at the user's account |

## **4.5 Other Concerns<Optional>**

*<You can list here all other concerns about the business or the requirements if needed>*

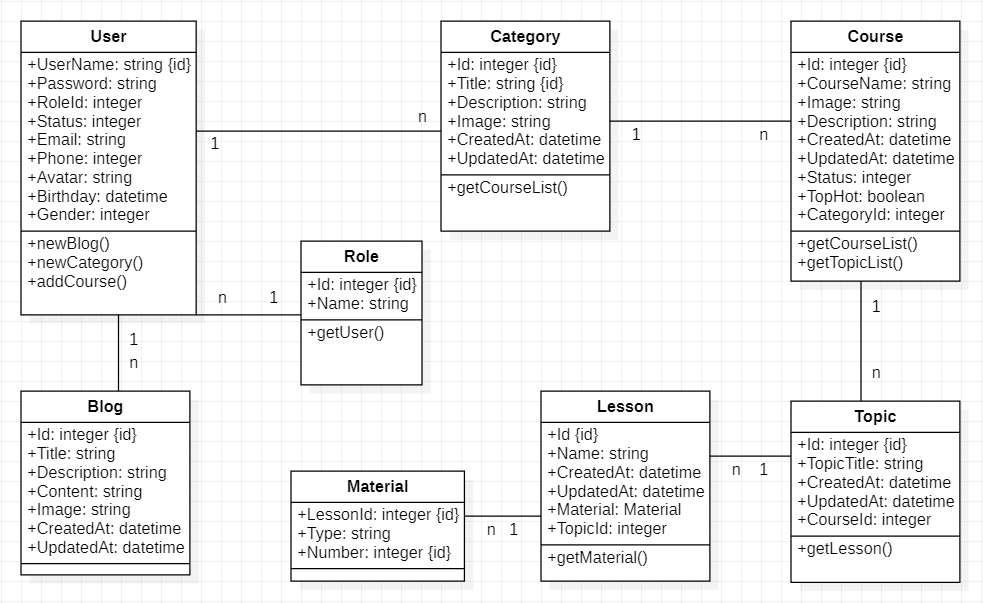
# **Design**

*[This section shows design of the system. This could be a part of the Developers Manual]*

## **System Architecture**

*<Explain and present the architecture of the system using texts or diagams>*

## **Class Diagram**

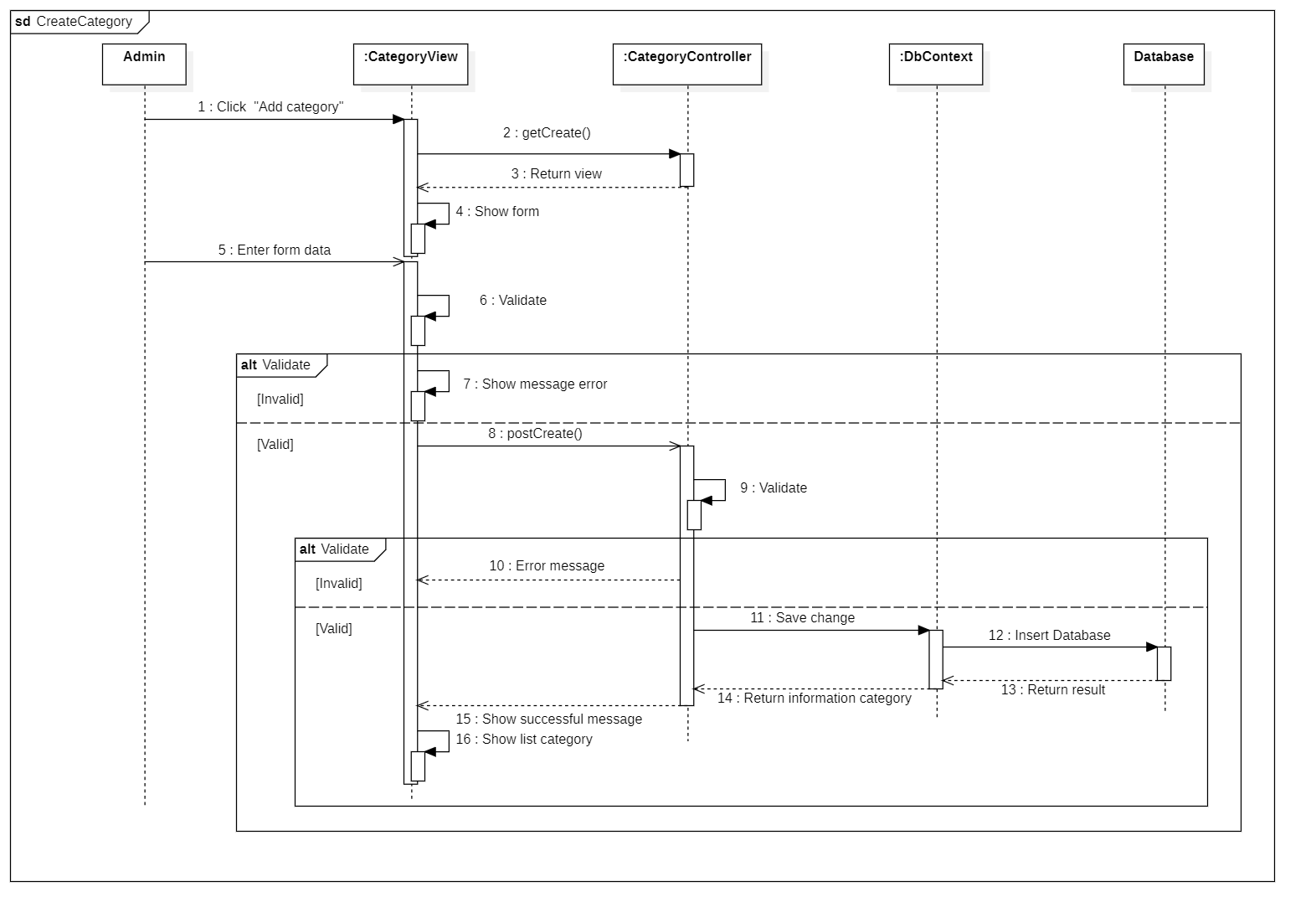


## **Class Diagram Explanation**

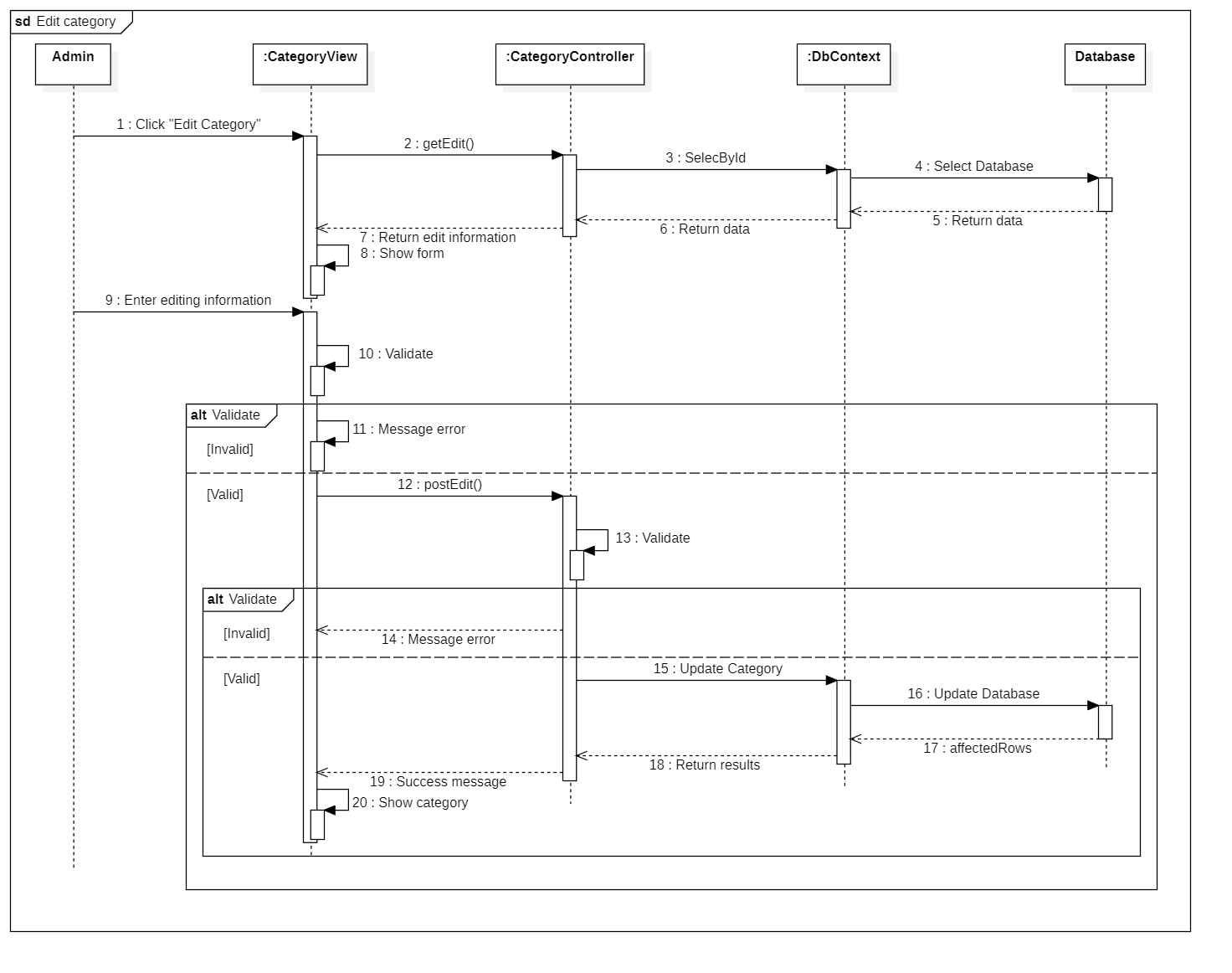
*<Provide brief explanation about the class diagram above. You do not need to explain “obvious” parts of your class diagram. For example, I know what a “Login” class is. Don’t say “The login class was created to store login information.”>*

## **Sequence Diagram (Optional)**

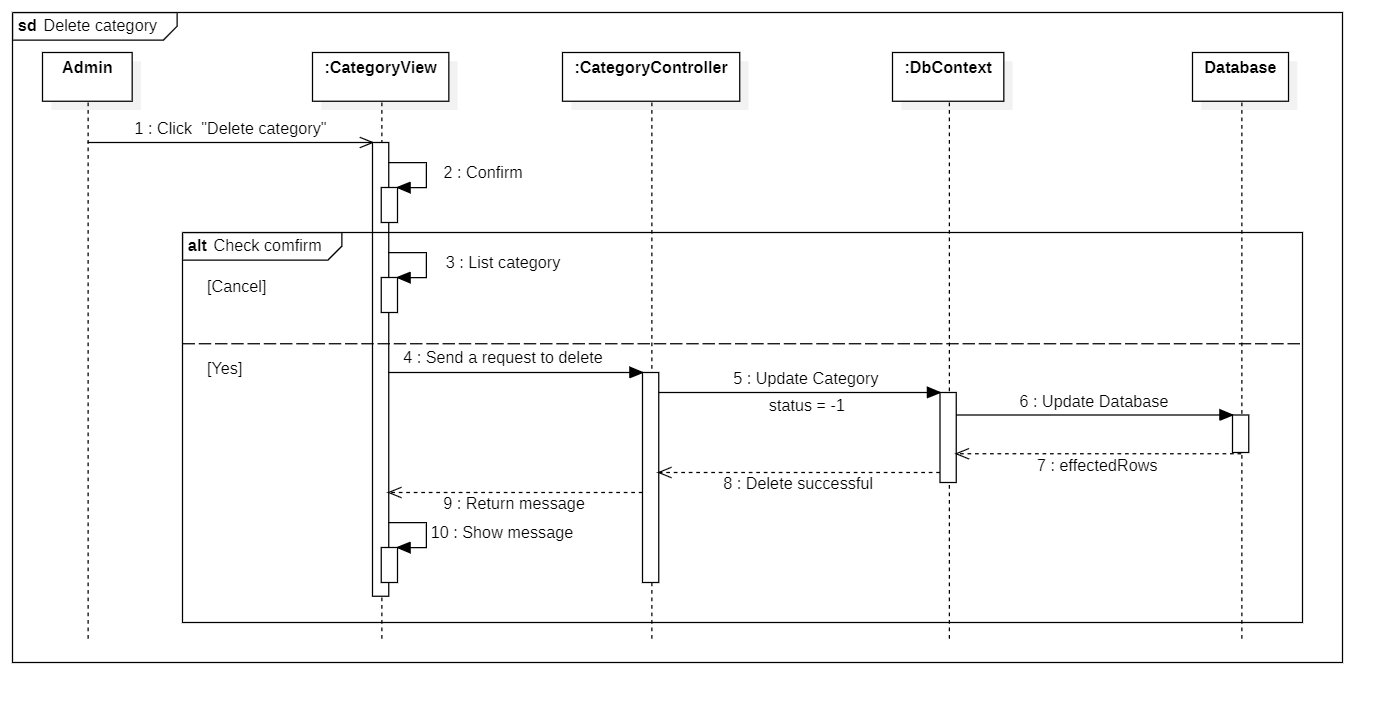
### 6.4.1 [Admin] Create Category



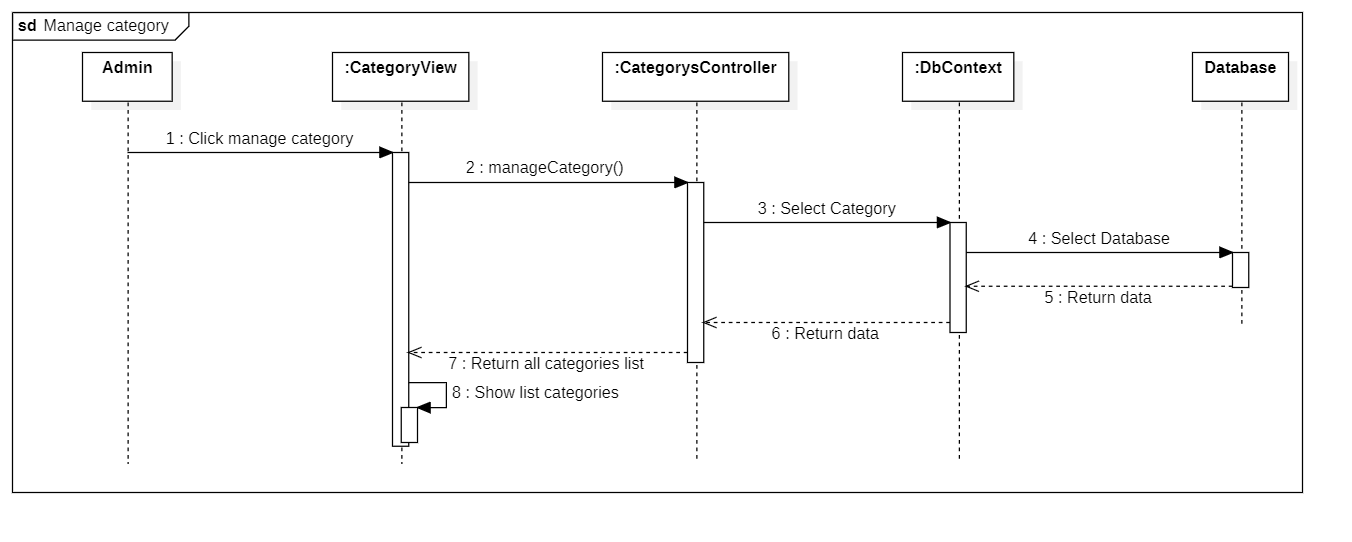
### 6.4.2 [Admin] Edit Categroy



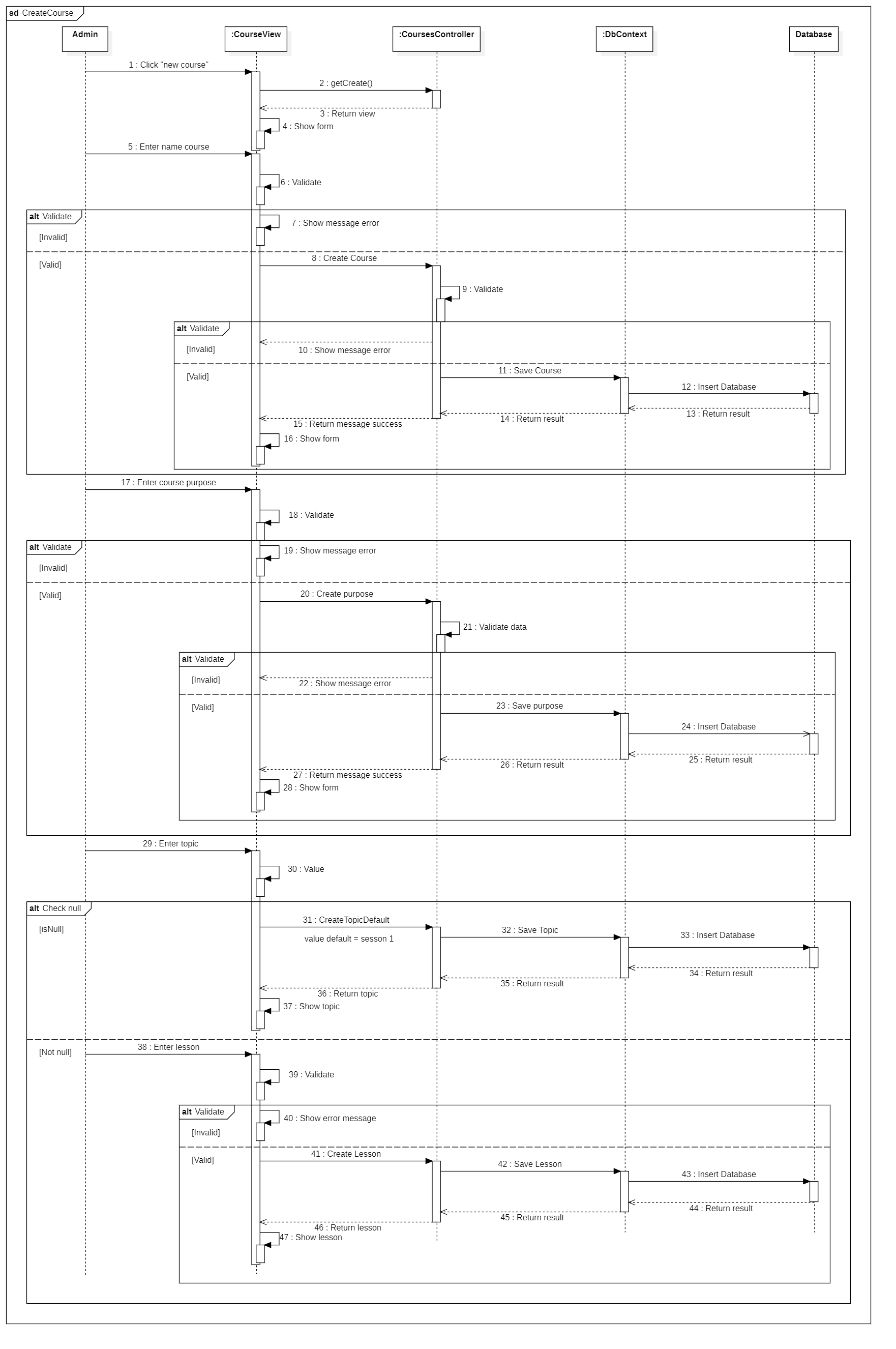
### 6.4.3 [Admin] Remove Category



### 6.4.4 [Admin] Manage Category



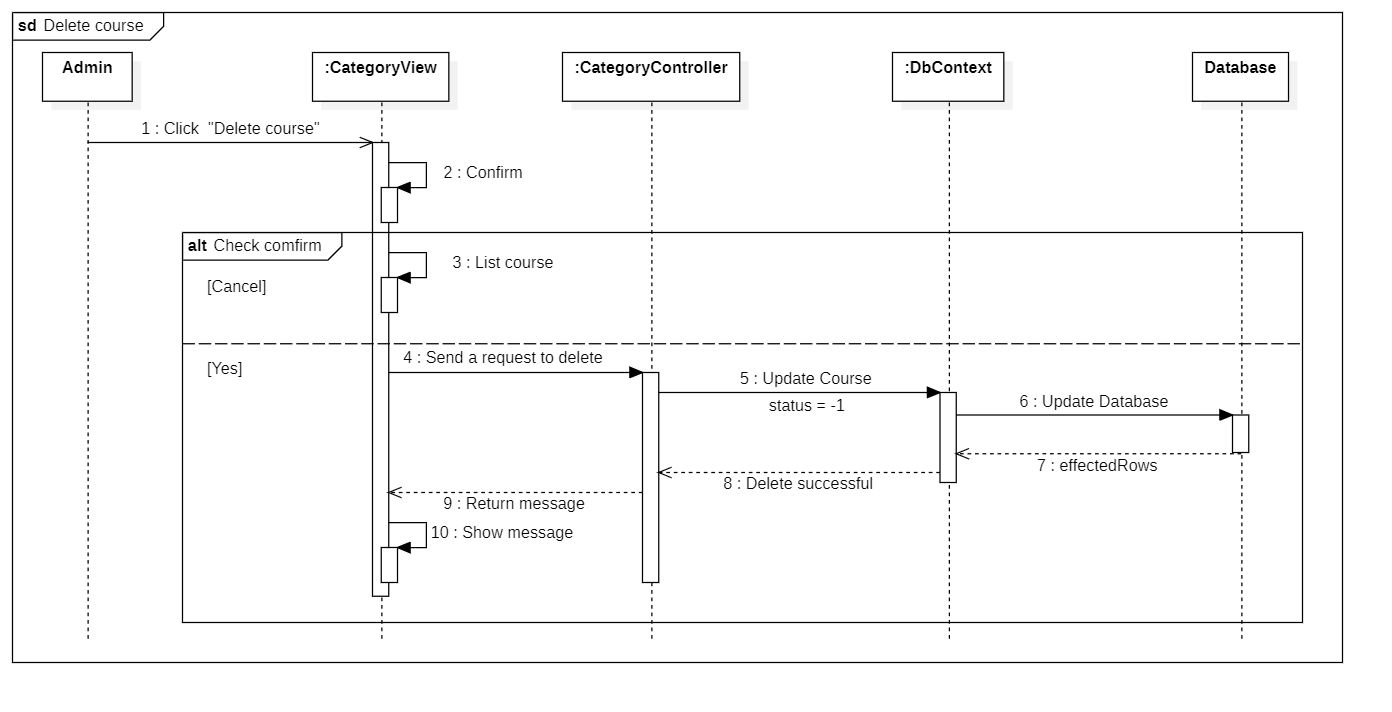
### 6.4.5 [Admin] Create Course



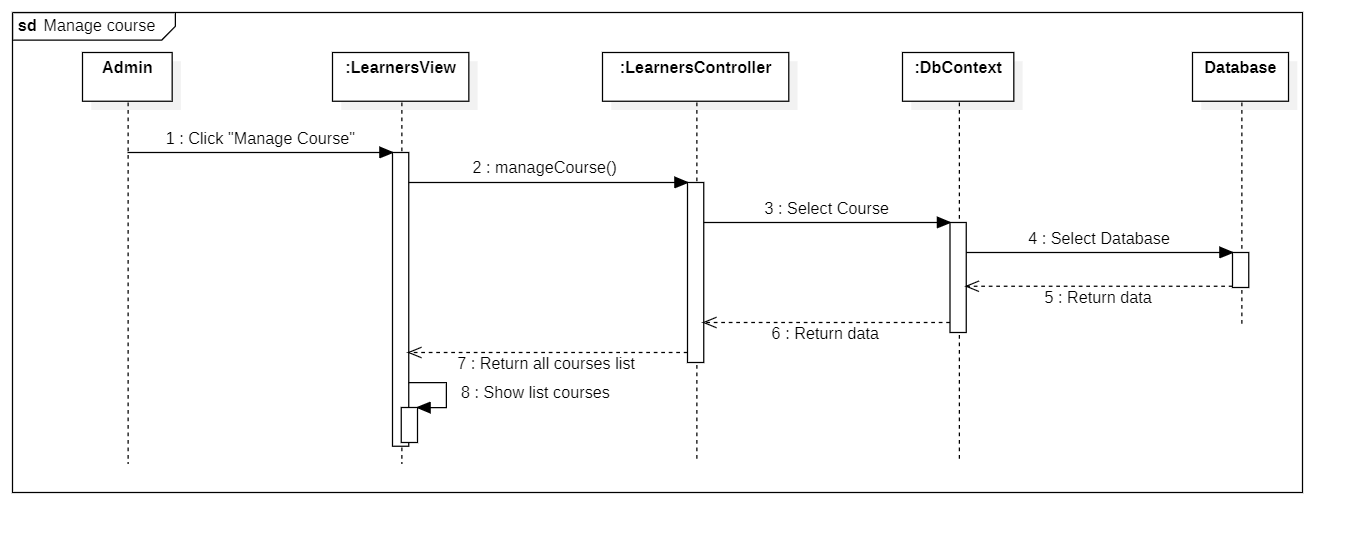
### 6.4.6 [Admin] Edit Course



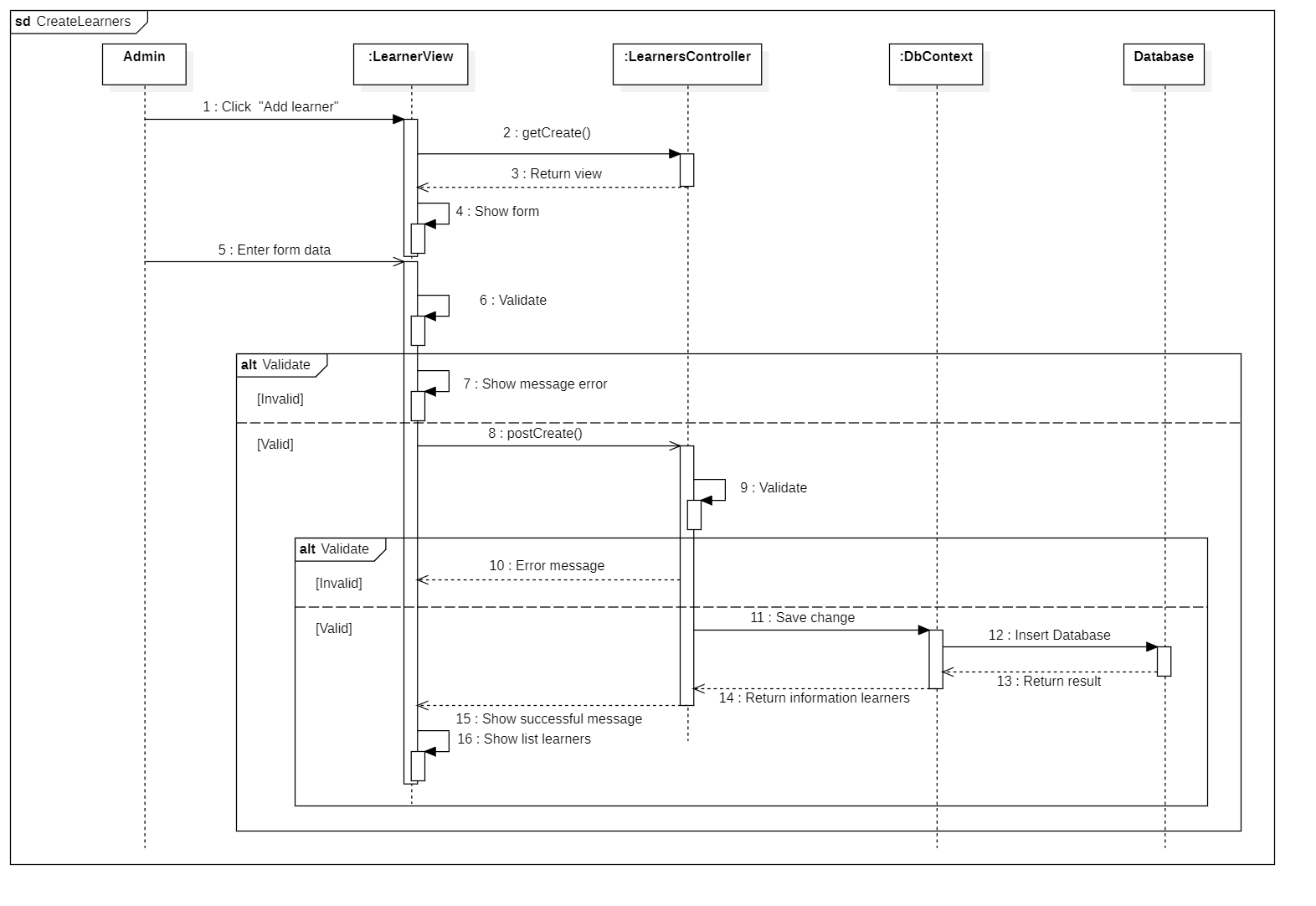
### 6.4.7 [Admin] Remove Course



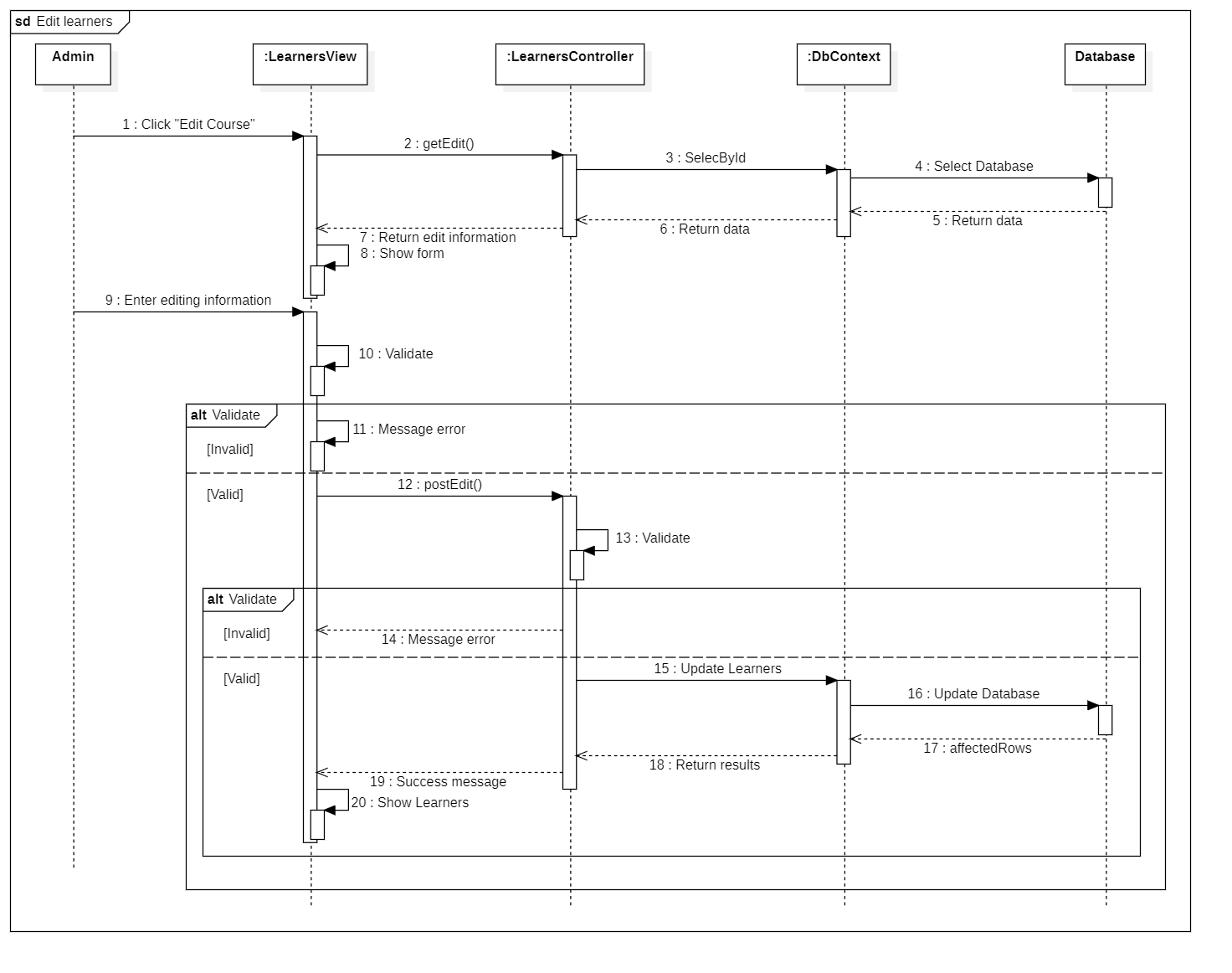
### 6.4.8 [Admin] Manage Course



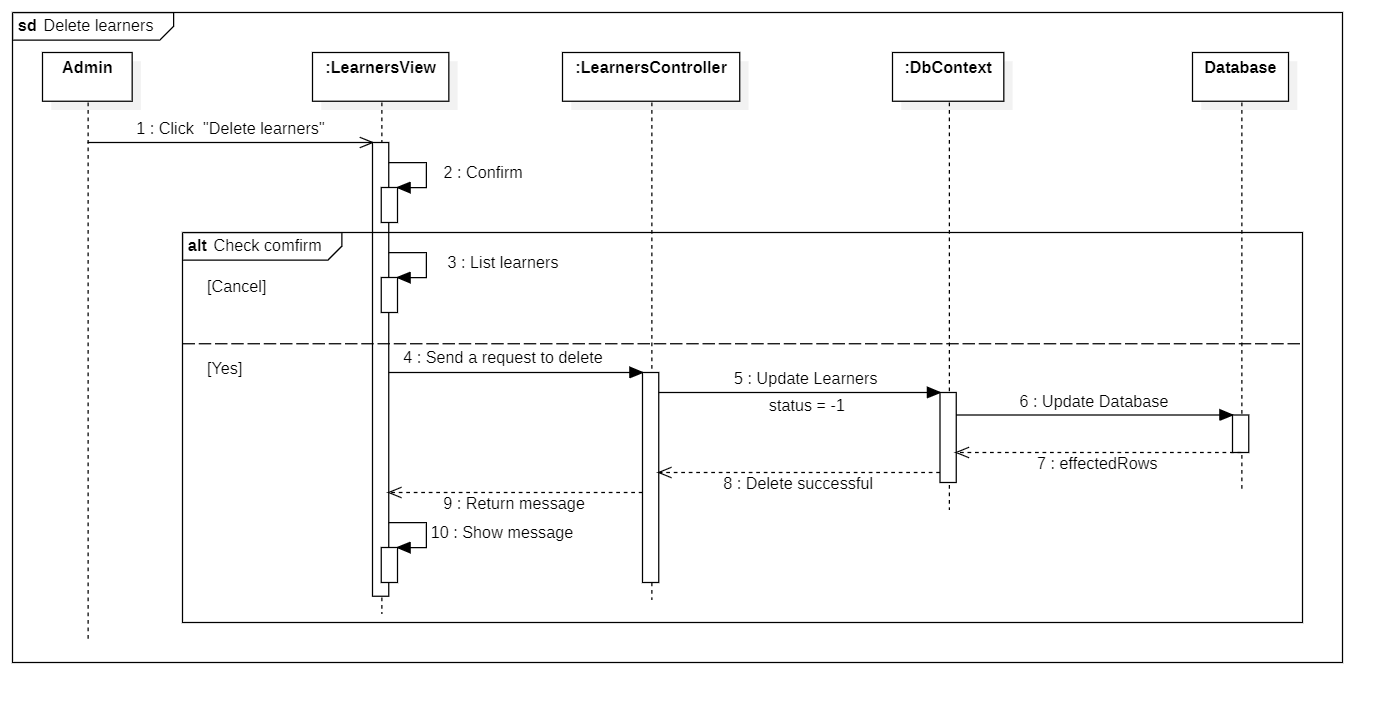
### 6.4.9 [Admin] Create learners



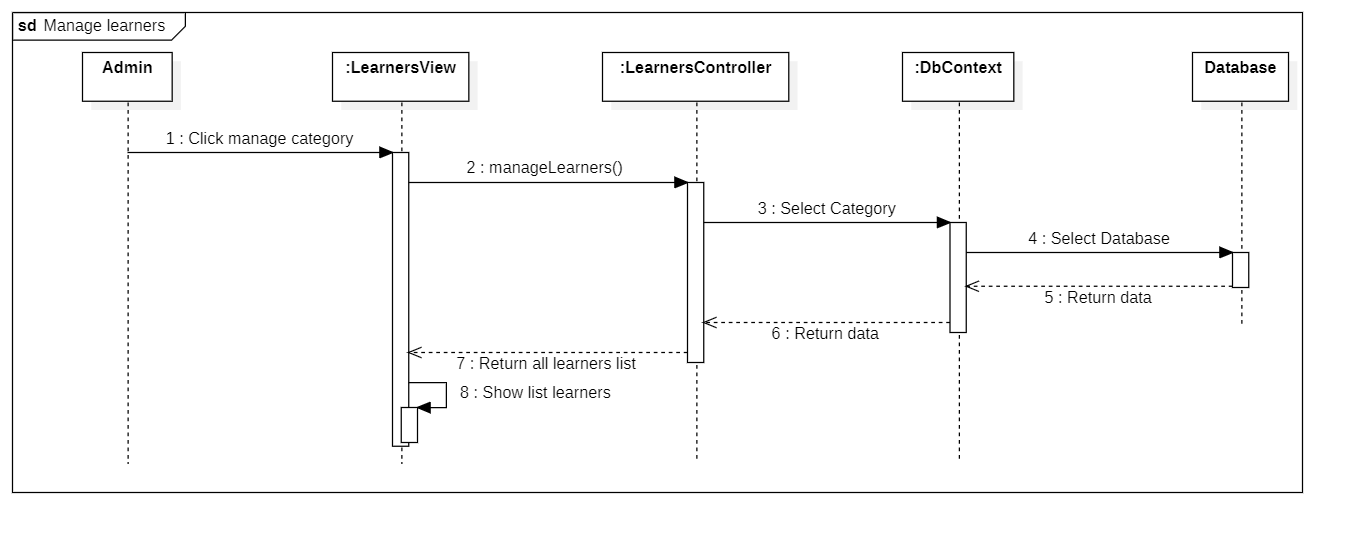
### 6.4.10 [Admin] Edit learners



### 6.4.11 [Admin] Remove learners



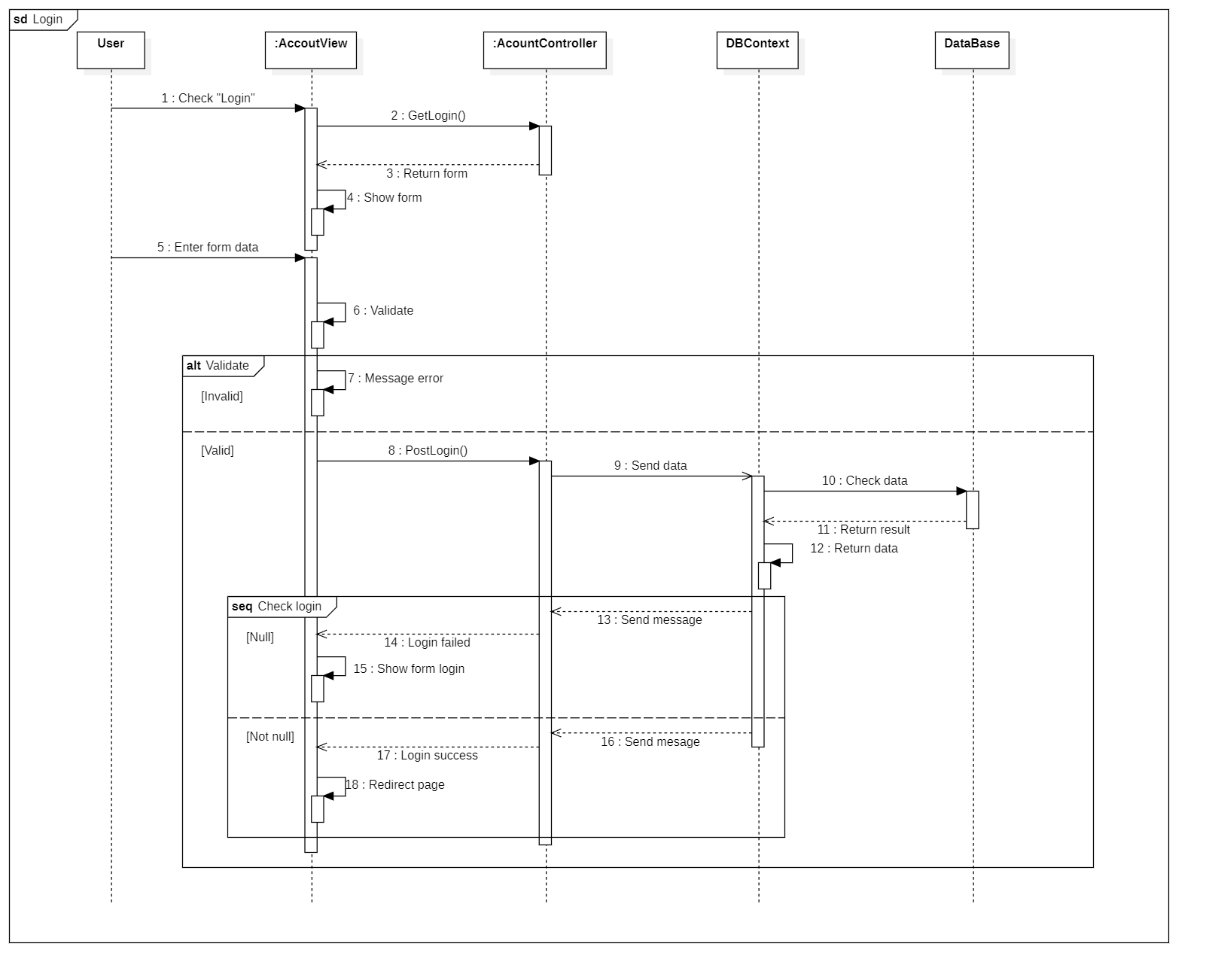
### 6.4.12 [Admin] Manage learners



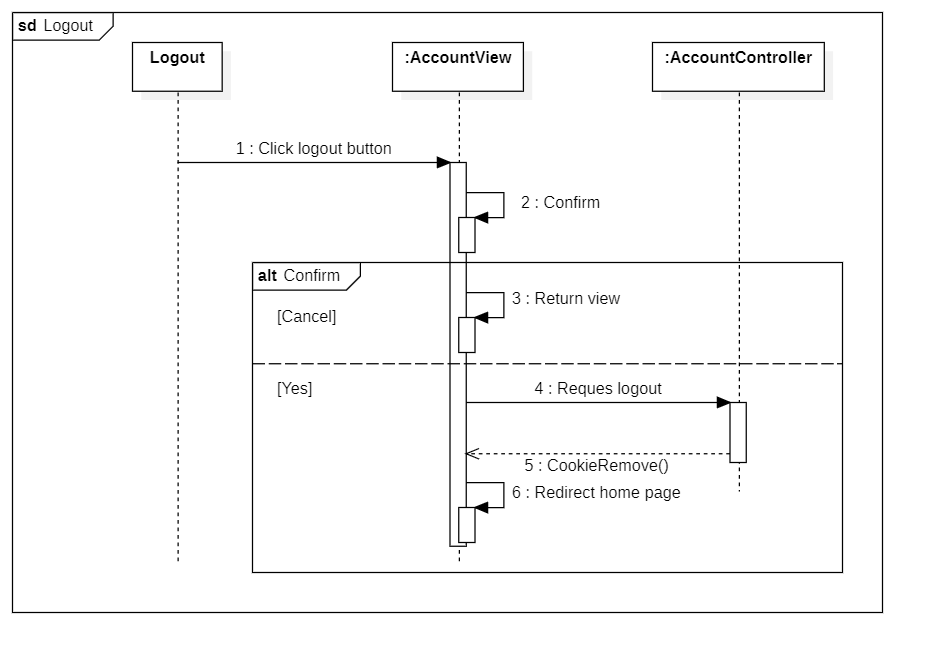
### 6.4.13 [Admin] Statistics income



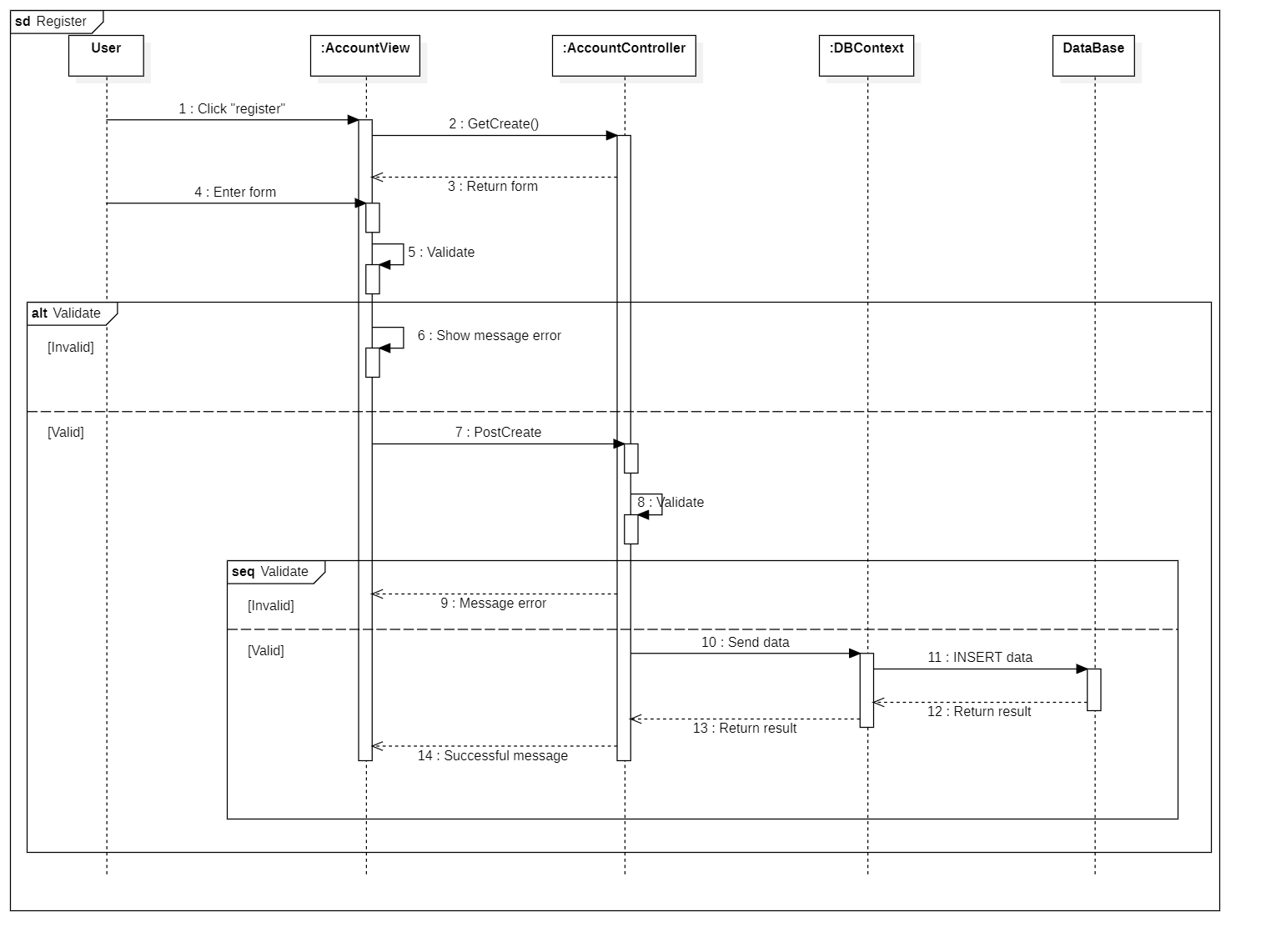
### 6.4.14 Login



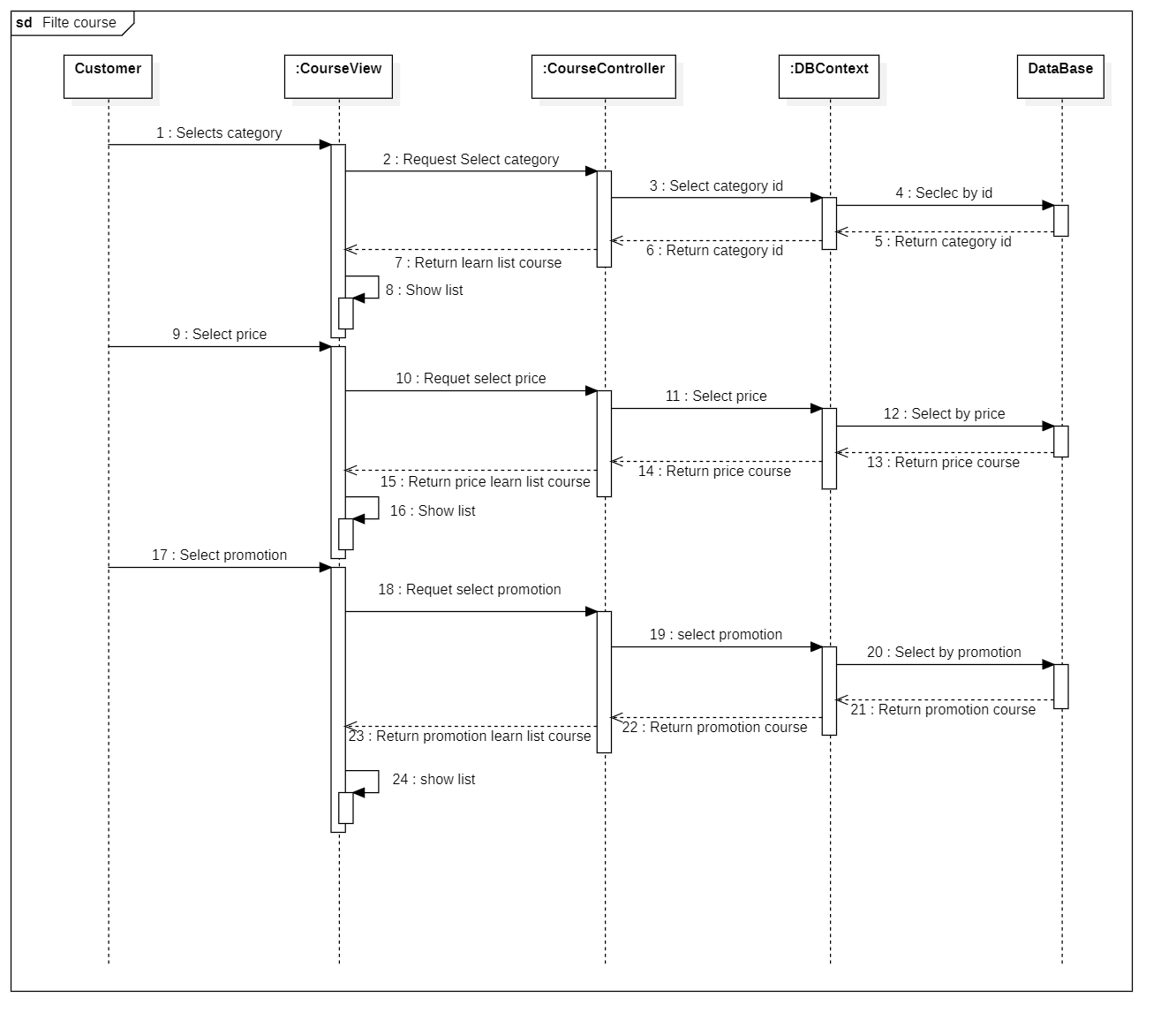
### 6.4.15 Logout



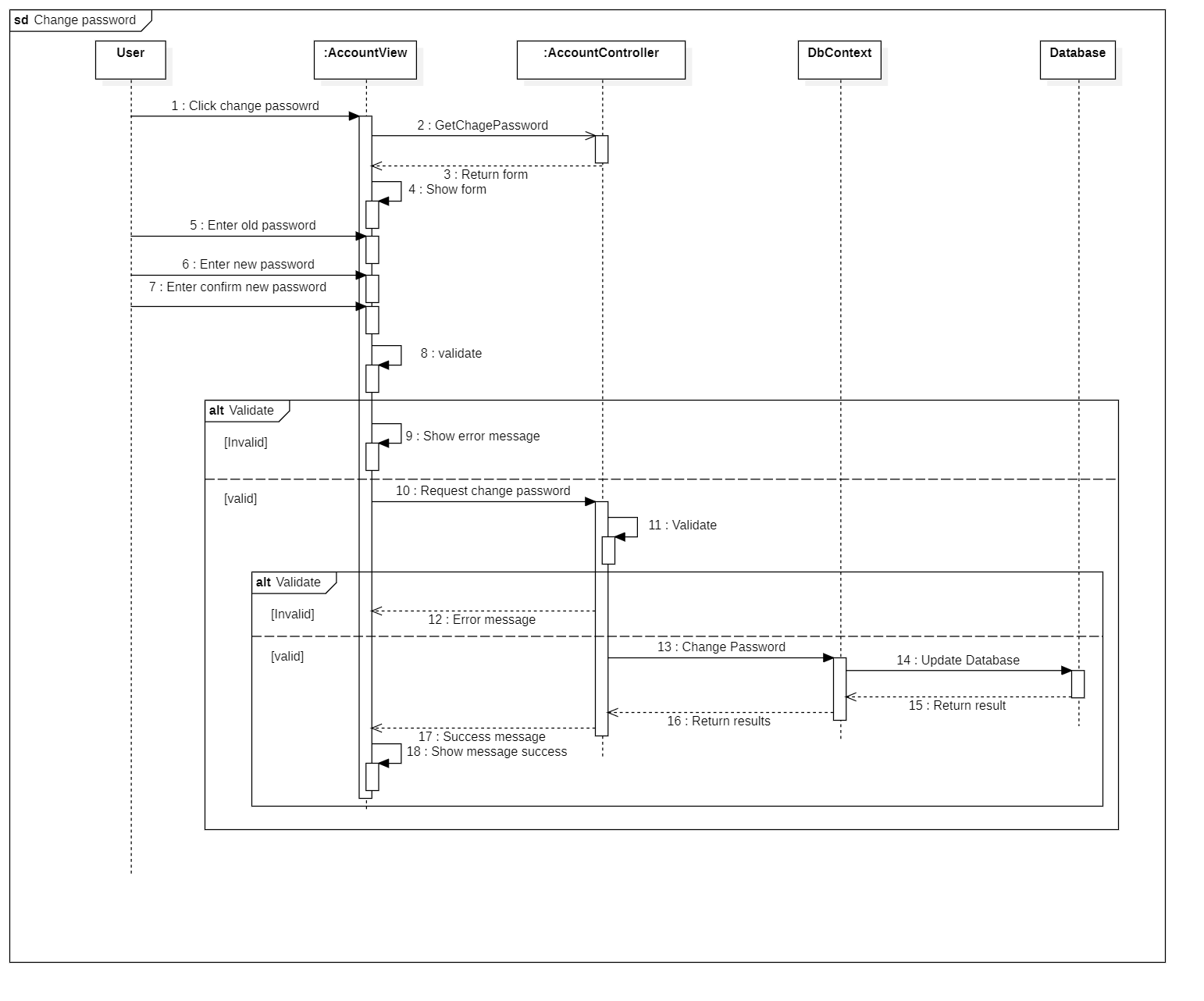
### 6.4.16 Register



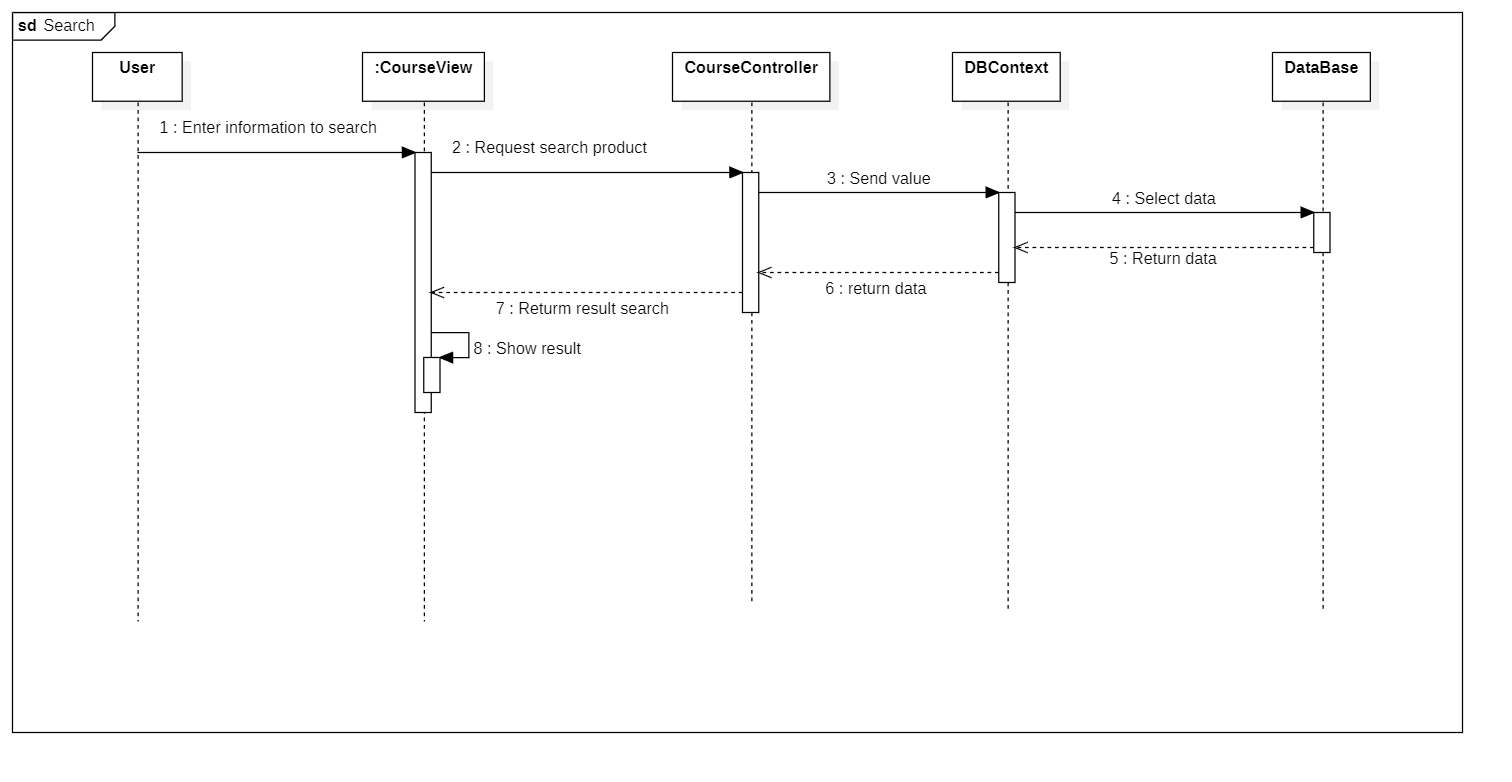
### 6.4.17 Filter



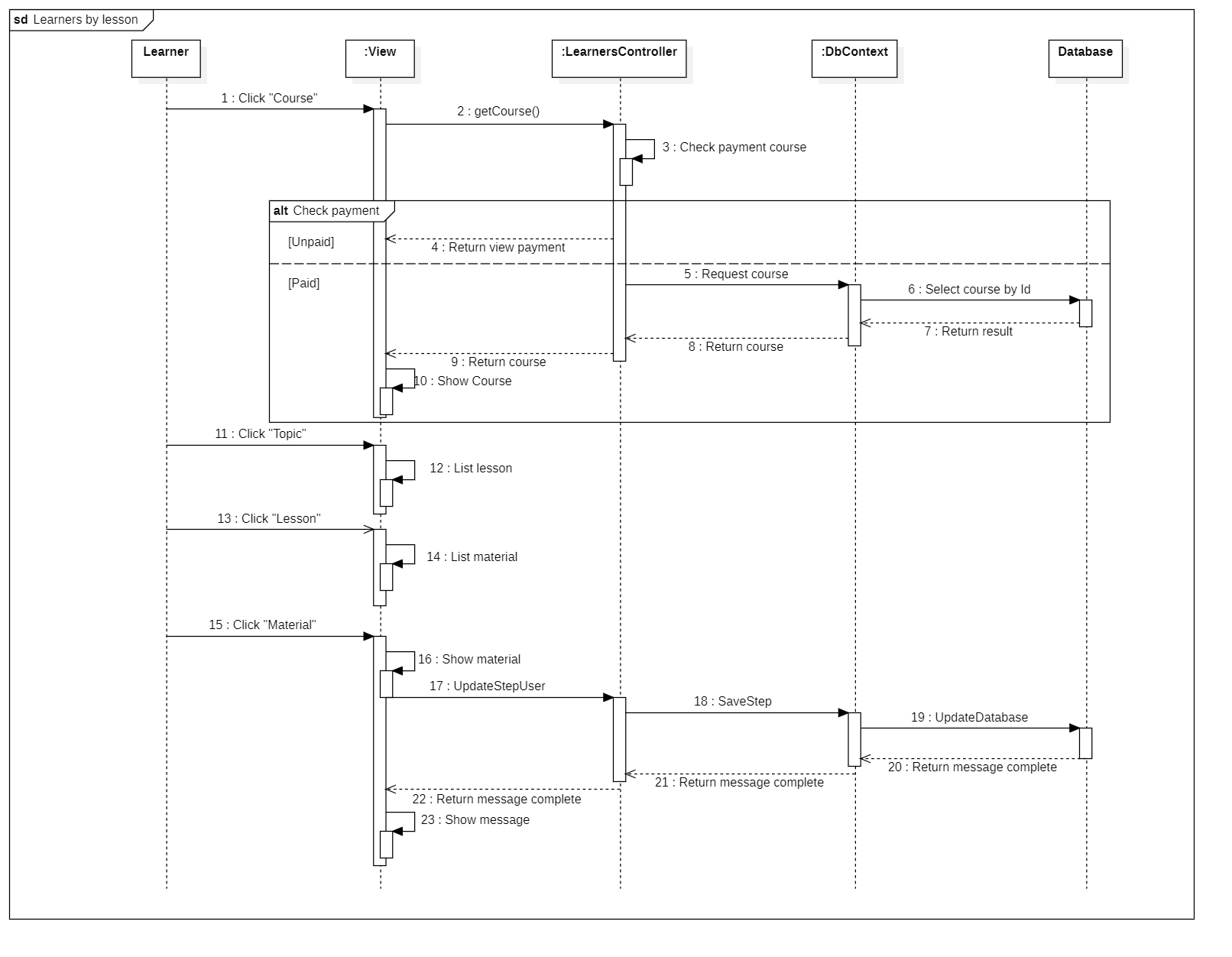
### 6.4.18 Change password



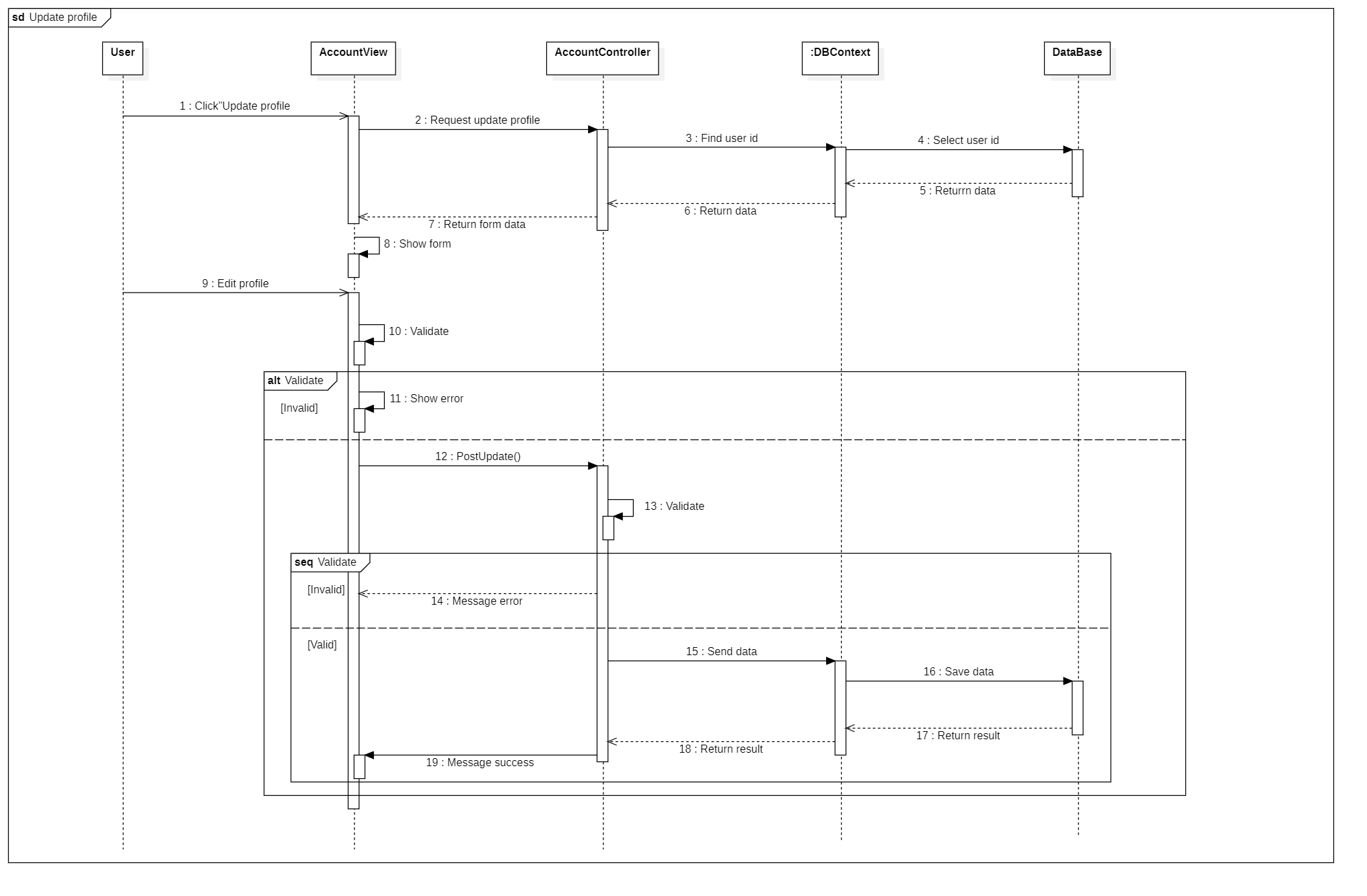
### 6.4.19 Search course



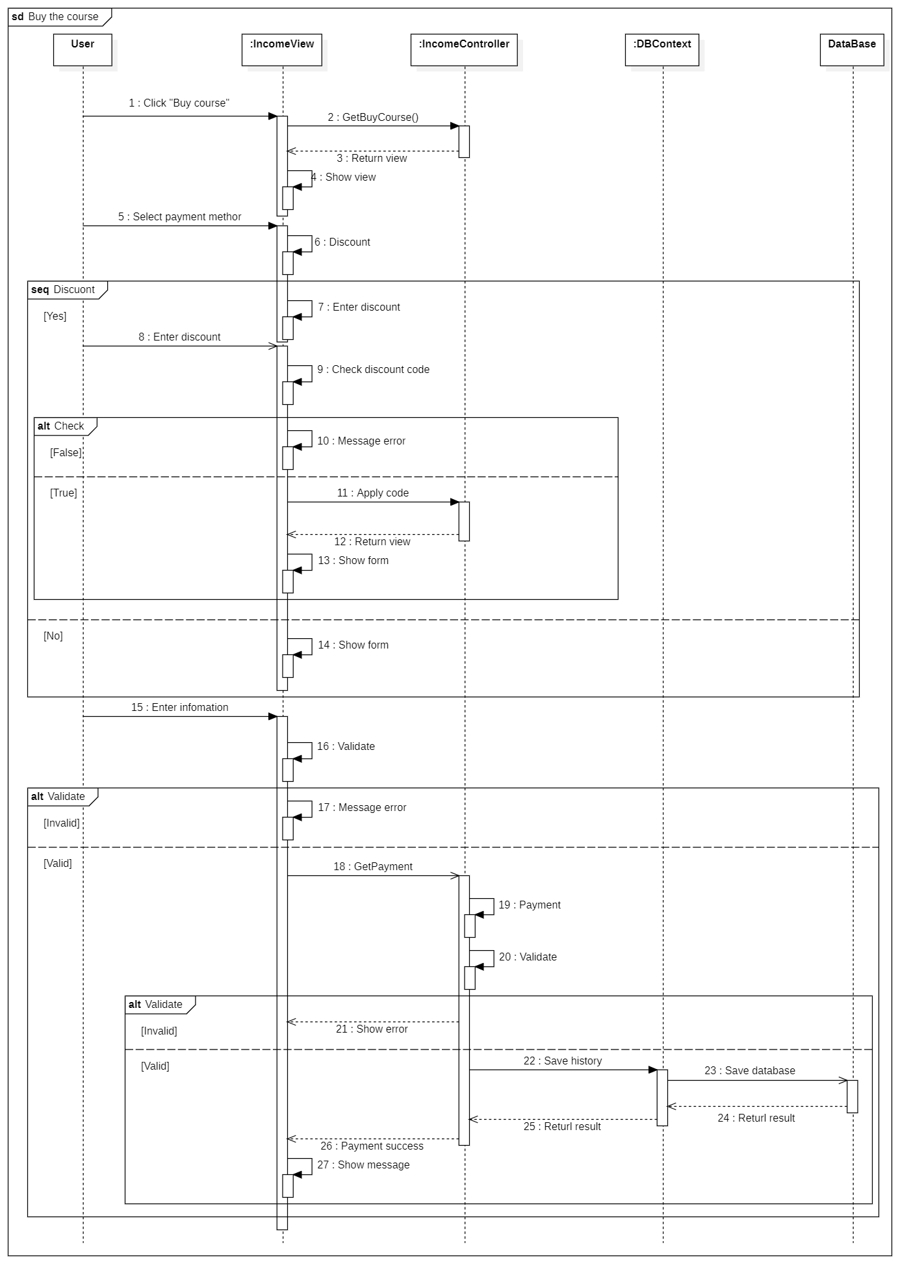
### 6.4.20 Learn by lesson



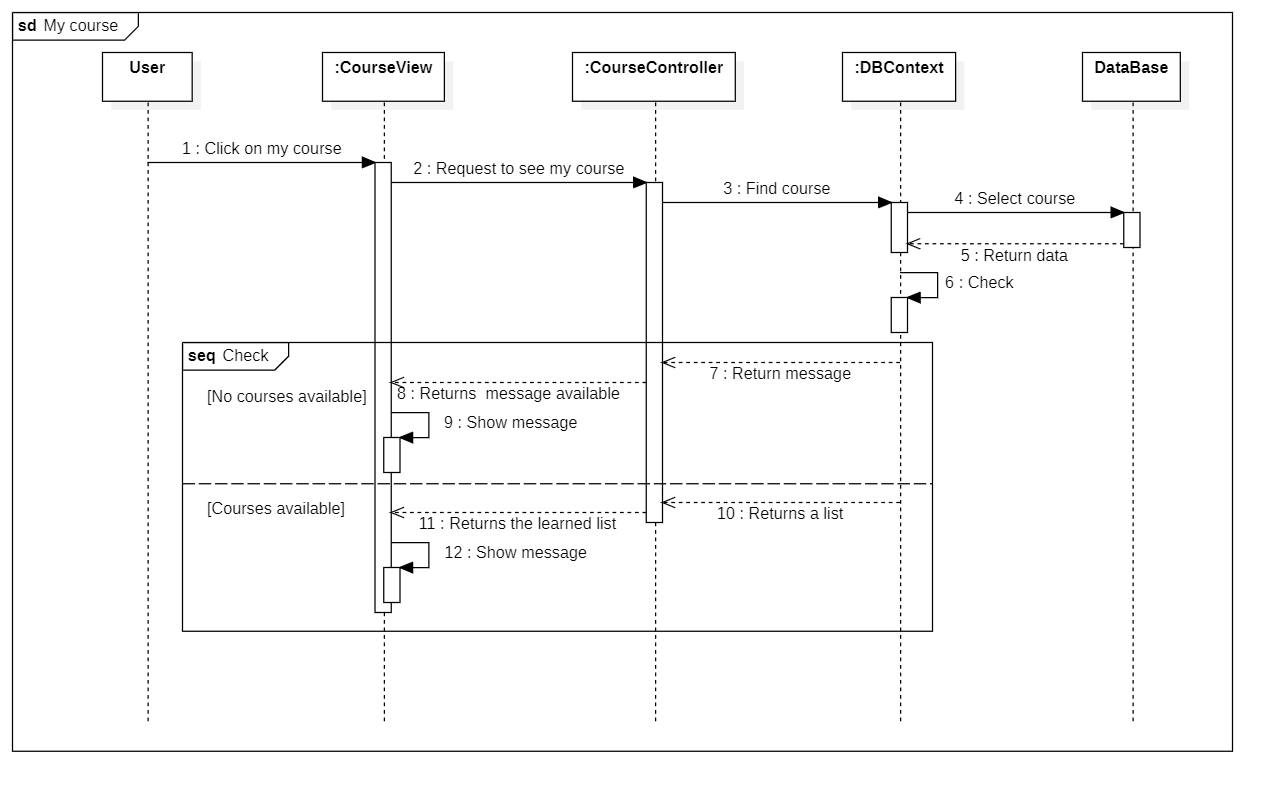
### 6.4.21 Update profile



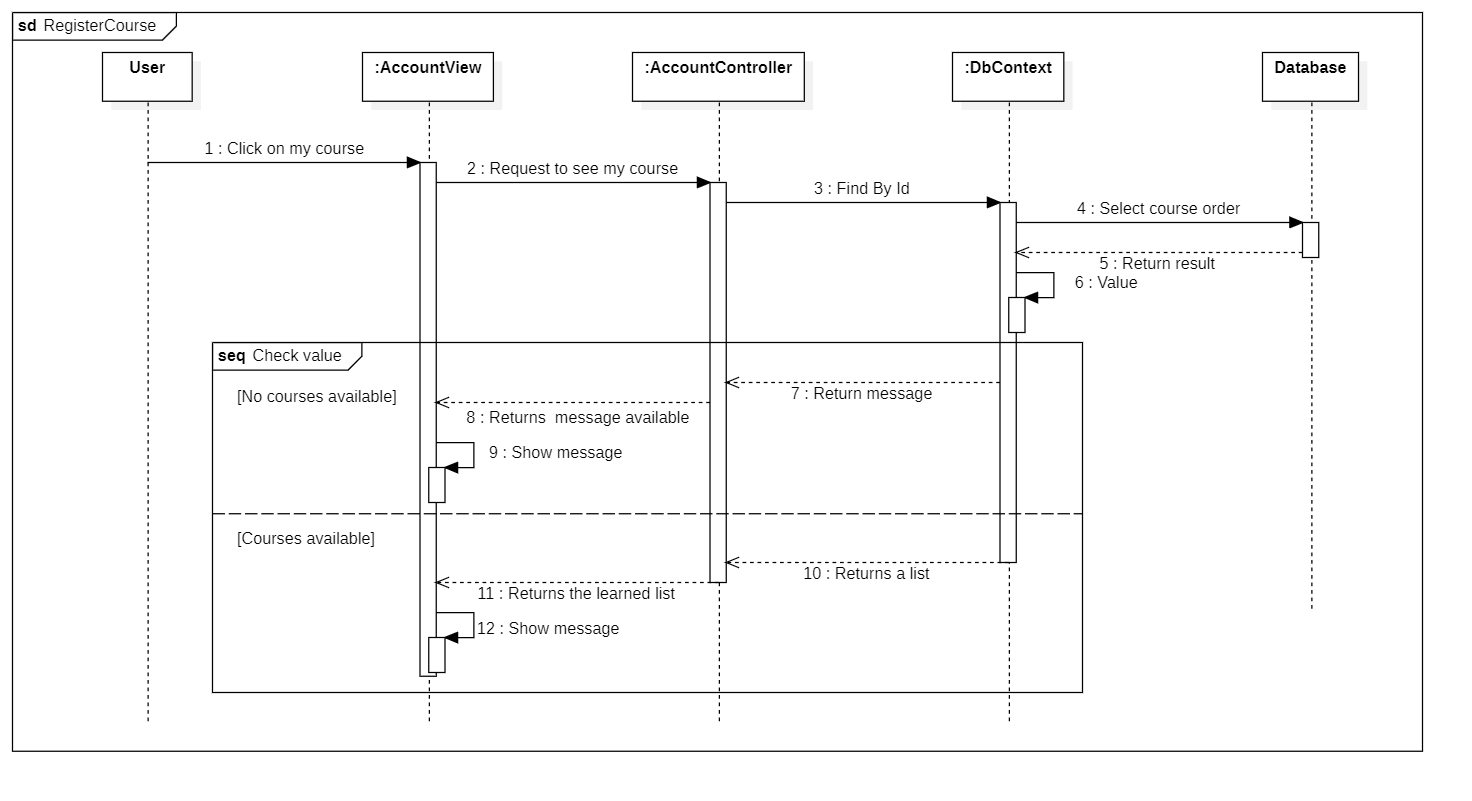
### 6.4.22 Buy the course



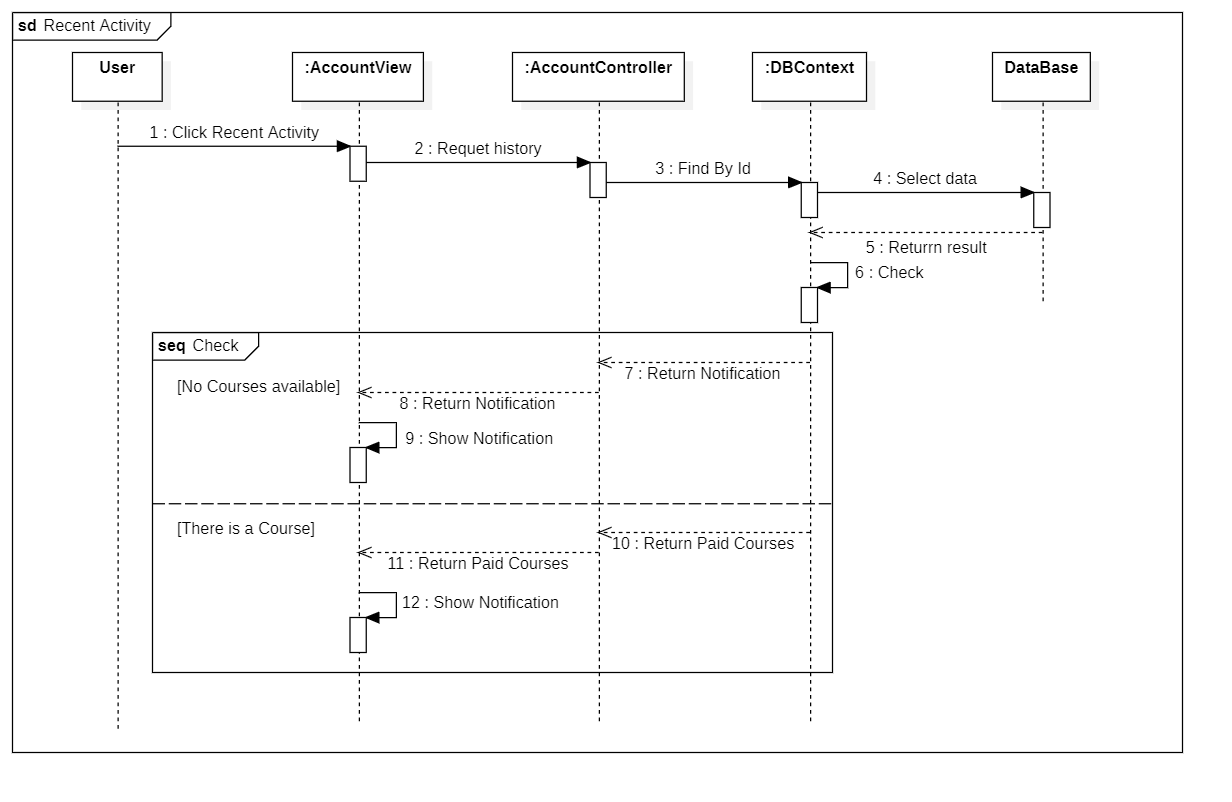
### 6.4.23 My course



### 6.4.24 Register course



### 6.4.25 Recent Activity



### 6.4.26 Account information



## **Collaboration Diagram (Optional)**

*<for important and complex interactions, collaboration diagrams should be drawn for clearing the details and supporting the system implementation. This section is optional>*

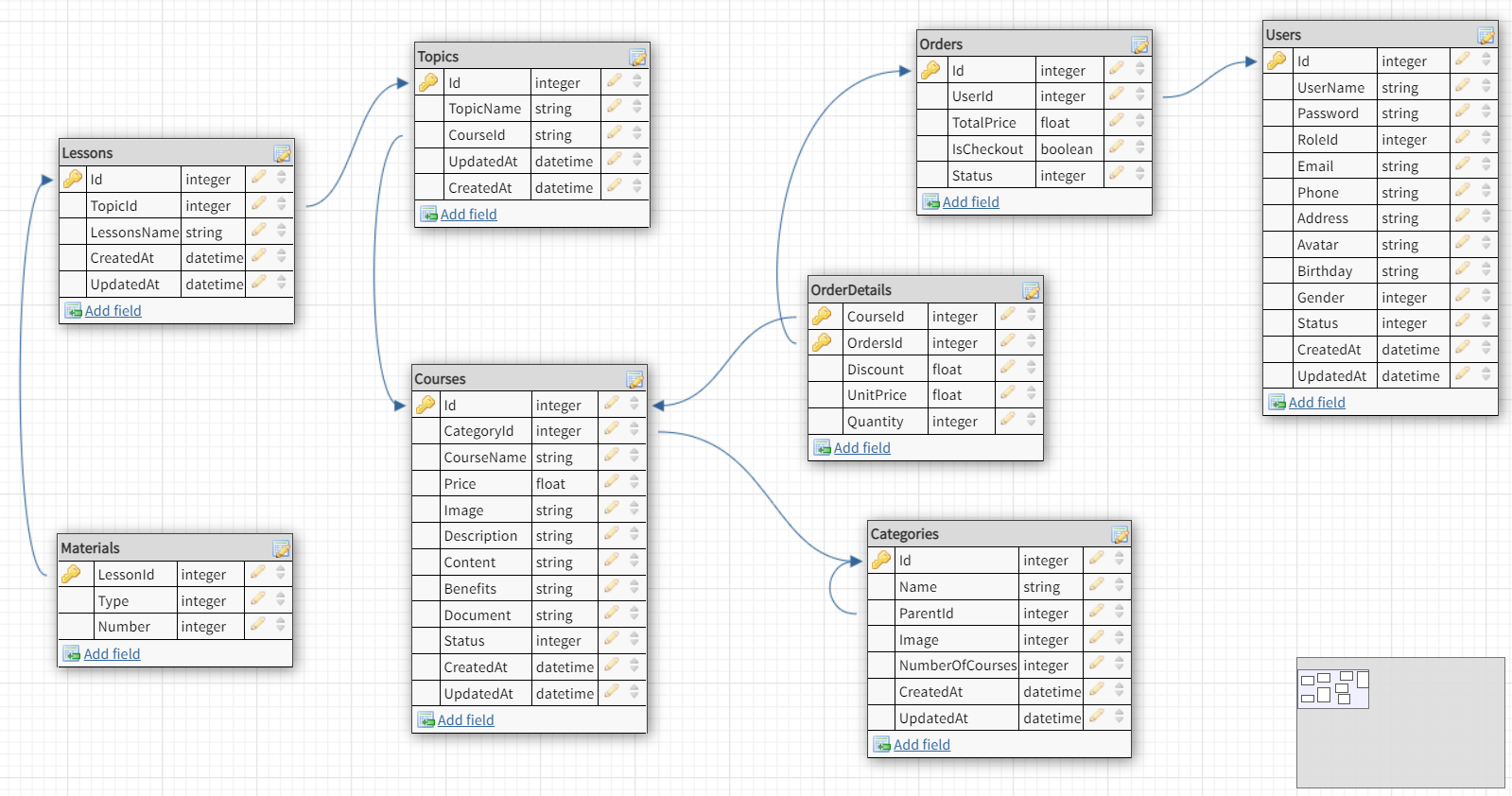
## **State Diagram (Optional)**

*<put all state diagrams here>*

## **Entity Relationship Diagram**

*<Provide the ERD Diagram for the system here. If your team uses file or in-memory storage facility instead of database, replace this section by ‘Data Structures’>*

## **Database Design**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Materials** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| LessonId | Int | No | PK,  FK | LessonId of Materials, LessonId is the  foreign key of the Id field in the  Lessons table |
| Type | Int | No |  | Type of materials |
| Number | Int | No |  | Number of materials |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Lessons** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | Id is the primary key of table |
| TopicId | Int | No | FK | TopicId is the foreign key of the Id field  in the Topics table |
| LessonName | nvarchar(max) | No |  | Name of lesson |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Topics** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | Id is the primary key of Topics |
| CourseId | Int | No | FK | CourseId is the foreign key of the Id  field in the Courses table |
| TopicName | nvarchar(max) | No |  | Name of topic |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Courses** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | Id is the primary key of Courses |
| CategoryId | Int | No | FK | CategoryId is the foreign key of the Id  field in the Categories table |
| CourseName | nvarchar(max) | No |  | Name of course |
| Price | float | No |  | Price of course |
| Image | narchar(max) | No |  | Image of course |
| Description | nvarchar(max) | No |  | Description of course |
| Content | nvarchar(max) | No |  | Content of course |
| Benefits | nvarchar(max) | No |  | Benefits of course |
| Document | varchar(max) | No |  | Document of course |
| Status | int | No |  | Status of course |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Categories** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | Id is the primary key of table |
| ParentId | Int | No | FK | ParentId is the foreign key of the Id  field in the Categories table |
| Name | nvarchar(max) | No |  | Name of category |
| Image | nvachar(max) | No |  | Image of category |
| NumberOfCourse | int | No |  | Number of courses |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : OrderDetails** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| CourseId | Int | No | PK, FK | CourseId is the primary key of table,  CourseId is the foreign key of the Id  field in the Courses table |
| OrderId | Int | No | PK,  FK | OrderId is the primary key of table,  OrderId is the foreign key of the Id  field in the Orders table |
| Discount | float | No |  | Discount of order |
| UnitPrice | float | No |  | Unit price of order |
| Quantity | int | No |  | Quantity of order |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Orders** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | Id is the primary key of table |
| UserId | Int | No | FK | UserId is the foreign key of the Id  field in the Users table |
| TotalPrice | float | No |  | Total price of order |
| IsCheckout | bool | No |  | Is checkout of order |
| Status | int | No |  | Status of order |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table : Users** | | | | |
| **Field Name** | **Data Type** | **Null** | **Key** | **Description** |
| Id | Int | No | PK | CourseId is the primary key of table |
| UserName | nvachar(max) | No |  | User name of user |
| Password | varchar(250) | No |  | Password of user |
| RoleId | int | No |  | Role of user |
| Email | varchar(250) | No |  | Email of user |
| Phone | varchar(20) | No |  | Phone of user |
| Address | nvarchar(max) | No |  | Address of user |
| Avatar | varchar(max) | No |  | Avatar of user |
| Birthday | datetime | No |  | Birthday of user |
| Gender | int | No |  | Gender of user |
| Status | int | No |  | Status of user |

## **Algorithms (optional)**

*<Provide the detailed description about algorithms used in the system. You can use Flow Chart or Activity Diagram to represent algorithms. Focus on the important and complex algorithms>*

## **Others (optional)**

*<Any design concerns or diagrams can be put here>*

# System Prototype

*<Put the system prototype or mock UI here. Focus on* ***important forms*** *and the* ***screen flows*** *between forms. If you use RAD and .NET, prototyping really help you to reach your development goals quickly. If you draw the UI manually or use a dedicated software, take snapshots and put them here.>*

# Management and Project Planning

## Management Approach

*< Briefly describe the management approach that your team selects. Is your team self-managed or managed by one leader? How do you assign tasks to team members? How often do you meet? What do you do during meeting? Etc.>*

## Project Plan

*<The detailed project plan is put here. You can use WBS Excel sheet, Sprint Backlog (see Sprint Backlog\_Template.xls), Task sheet, Gantt chart, etc. to present your team’s plan. You can capture the Gantt chart in PMS if you use it to plan your project>*

## Task Sheet

*<Write down the tasks in Task Sheet-compatible format, this Task Sheet works as the activity report of the projec or the plan of the project (not recommended); see eProject Guide for detailed Task Sheet>*

## Meeting Minutes (Optional)

*<Put all minutes of your team meetings here>*

# Checklists

## Check List of Validation

*< Put the checklist here; describe how it is used and the resulted checklist>*

## Submission Checklist

*< Put the checklist here; describe how it is used and the resulted checklist>*

# Screenshots

*<Capture some intuitive and main screens of the software and put them here>*

# Coding Convention

*<Provide the coding convention for your team. If you simply want to use the existing code standard(s) such as ‘Java Code Convention’, you can refer to it\them by name or URL>*

# User Guide

*<Provide guidelines for using the system >*

# Other Concerns<Optional>

*<If you have any other information you want to add to this document, place it here. This could include thoughts on the eProject, improvements, etc.>*

# Appendix

## Glossary [Optional]

*<Place all definitions or abbreviation used in this document >*

## References [Optional]

*<Place all referenced materials used in this document >*

## Others<Optional>