**Requirement Specifications**

**Project: Cursus**

Version: 0.2

**Approval Page**

The endorsement on this document, by the authorized <<client>> representative, indicates that <<client>> and FPT have agreed on the document “Cursus- Software Requirement Specifications”.

|  |  |  |  |
| --- | --- | --- | --- |
| **Prepared by :** | FPT  Business Analyst | Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  | Date: | \_\_\_\_ /\_\_\_\_ / \_\_\_\_ |
|  |  |  |  |
|  |  |  |  |
| **Reviewed by :** | FPT  Project Manager | Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | Consultant | Date: | \_\_\_\_ /\_\_\_\_ / \_\_\_\_ |
|  |  |  |  |
| **Supported by :** | FPT  Project Manager | Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  | Date: | \_\_\_\_ /\_\_\_\_ / \_\_\_\_ |
|  |  |  |  |
|  |  |  |  |
| **Approved by :** | FPT  Project Manager | Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  | Date: | \_\_\_\_ /\_\_\_\_ / \_\_\_\_ |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Change Description** |
| 30 Apr- 22 | 0.1 | Nguyen Phuong Nhat Khoa | Create SRS for Cursus |
| 5 May –24 | 0.2 | Tuan Tran | Modify SRS use case, paint point. |

**Mục lục**

[1. Introduction 4](#_1fob9te)

[1.1 Purpose 5](#_3znysh7)

[1.2 Overview 5](#_2et92p0)

[1.3 Intended Audience and Reading Suggestions 5](#_tyjcwt)

[Abbreviations 5](#_3dy6vkm)

[2. High Level Requirements 6](#_1t3h5sf)

[2.1 System structure 6](#_4d34og8)

[2.2 Workflow 6](#_2s8eyo1)

[2.3 Use Case Diagram 6](#_17dp8vu)

[2.3.1 General users 6](#_3rdcrjn)

[2.3.2 Administrator 8](#_lnxbz9)

[2.4 Permission matrix 10](#_35nkun2)

[3. Use Case Specifications 13](#_1ksv4uv)

[3.1 General functions 13](#_44sinio)

[3.1.1 UC 01: Sign up for Student 13](#_2jxsxqh)

[3.1.2 UC 02: Sign up for Instructor 15](#_z337ya)

[3.1.3 UC 03: Sign in 17](#_3j2qqm3)

[3.1.4 UC 04: Verify email 19](#_1y810tw)

[3.1.5 UC 05: Change password 21](#_4i7ojhp)

[3.1.6 UC 06: Forgot password 23](#_2xcytpi)

[3.2 General user 24](#_1ci93xb)

[3.2.1 UC 07: View Homepage 24](#_3whwml4)

[3.2.2 UC 08: Update profile 25](#_2bn6wsx)

[3.2.3 UC 09: View and Search course 26](#_qsh70q)

[3.2.4 UC 10: View enrolled course 27](#_3as4poj)

[3.2.5 UC 11: View course by category 27](#_1pxezwc)

[3.2.6 UC 12: View list of subscriptions 28](#_49x2ik5)

[3.2.7 UC 13: Search instructor 29](#_2p2csry)

[3.2.8 UC 14: Purchase the course 29](#_147n2zr)

[3.2.9 UC 15: Enroll course 30](#_3o7alnk)

[3.2.10 UC 16: Review course 30](#_23ckvvd)

[3.2.11 UC 17: Report course 31](#_ihv636)

[3.2.12 UC 18: View Dashboard 32](#_32hioqz)

[3.2.13 UC 19: View course/ student analytics 32](#_1hmsyys)

[3.2.14 UC 20: Create course 33](#_41mghml)

[3.2.15 UC 21: Edit course 34](#_2grqrue)

[3.2.16 UC 22: Re-submit course 34](#_vx1227)

[3.2.17 UC 23: Delete course 35](#_3fwokq0)

[3.2.18 UC 24: View reviews 36](#_1v1yuxt)

[3.2.19 UC 25: View earnings analytics 36](#_4f1mdlm)

[3.2.20 UC 26: Payout 37](#_2u6wntf)

[3.3 Admin 38](#_19c6y18)

[3.3.1 UC 27: Login 38](#_3tbugp1)

[3.3.2 UC 28: View dashboard 38](#_28h4qwu)

[3.3.3 UC 29: View list of students/instructors 39](#_nmf14n)

[3.3.4 UC 30: Approve or Reject Instructor registration 39](#_37m2jsg)

[3.3.5 UC 31: View list of categories 40](#_1mrcu09)

[3.3.6 UC 32: Create a category 40](#_46r0co2)

[3.3.7 UC 33: Edit a category 41](#_2lwamvv)

[3.3.8 UC 34: Delete category 42](#_111kx3o)

[3.3.9 UC 35: Block/ unblock student/ instructor 42](#_3l18frh)

[3.3.10 UC 36: View list of courses 43](#_206ipza)

[3.3.11 UC 37: View details of a course 43](#_4k668n3)

[3.3.12 UC 38: Approve/Reject course 44](#_2zbgiuw)

[4. Non-functional requirements 44](#_1egqt2p)

[4.1 Performance requirements 45](#_3ygebqi)

[4.2 Security Requirements 45](#_2dlolyb)

[4.3 Software quality attributes 45](#_sqyw64)

[5. Other requirements 45](#_3cqmetx)

[5.1 Localize rules 45](#_1rvwp1q)

[5.2 Common field 45](#_4bvk7pj)

[5.2.1 Freetext 45](#_2r0uhxc)

[5.2.2 Dropdown list 45](#_1664s55)

[5.2.3 Scroll list for selecting Category 46](#_3q5sasy)

[5.2.4 Image attachment 46](#_25b2l0r)

[5.2.5 Rich text 46](#_kgcv8k)

[5.2.6 Text area 47](#_34g0dwd)

[5.2.7 Date picker 47](#_1jlao46)

[5.2.8 Label display 47](#_43ky6rz)

[5.3 Message Configuration 47](#_2iq8gzs)

[5.4 Internal System Error 47](#_xvir7l)

[5.5 Pagination 47](#_3hv69ve)

[6. Appendices 47](#_1x0gk37)

[6.1 Message list 48](#_4h042r0)

[6.2 Email template 48](#_pkwqa1)

# Introduction

## Purpose

The Functional Requirements Specification will:

* Define the scope of business objectives, business functions, and organizational units covered,
* Identify the business processes that the solution must facilitate,
* Facilitate a common understanding of what the functional requirements are for all parties involved,
* Establish a basis for defining the acceptance tests for the solution to confirm that what is delivered meets requirements.
* The purpose of the document is to collect and analyse all assorted ideas that have come up to define the system, its requirements with respect to consumers. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outline concepts that may be developed later, and document ideas that are being considered, but may be discarded as the product develops.

## Overview

Overview - Litware, Inc. is a medium-sized Education technology company that has a main office in Boston.   
Litware researched about the Edtech domain and want to build an Online Course Management with name **Cursus** as [MVP](https://www.projectmanagement.com/blog-post/61937/Defining-MVP--MBI--MMF--and-MMR?__cf_chl_tk=UxPzcg2O.AGiRnz1MZ3q5CZH7iyC5KYWI6DmUjm24I0-1715072932-0.0.1.1-1407) (Minimum Value Project - simple) version to test the EU market trend.

Cursus system will allow user to be a part of Student/Studier and Instructor. Coacher will publish the course and student can purchase and study. Student could interact with Instructor as well.

Litware consider about the Client-Side architect for this Cursus to improve the user experience.

## Intended Audience and Reading Suggestions

This document is intended for:

* Development team: Responsible to develop detailed design, implement and perform unit test. Also perform test manual.
* Project Owner: Responsible to manage the Product backlog for the application.

## Abbreviations

|  |  |
| --- | --- |
| **Acronym** | **Reference** |
| SRS | Software requirement specification |
| UC | Use case |
|  |  |
|  |  |

# High Level Requirements

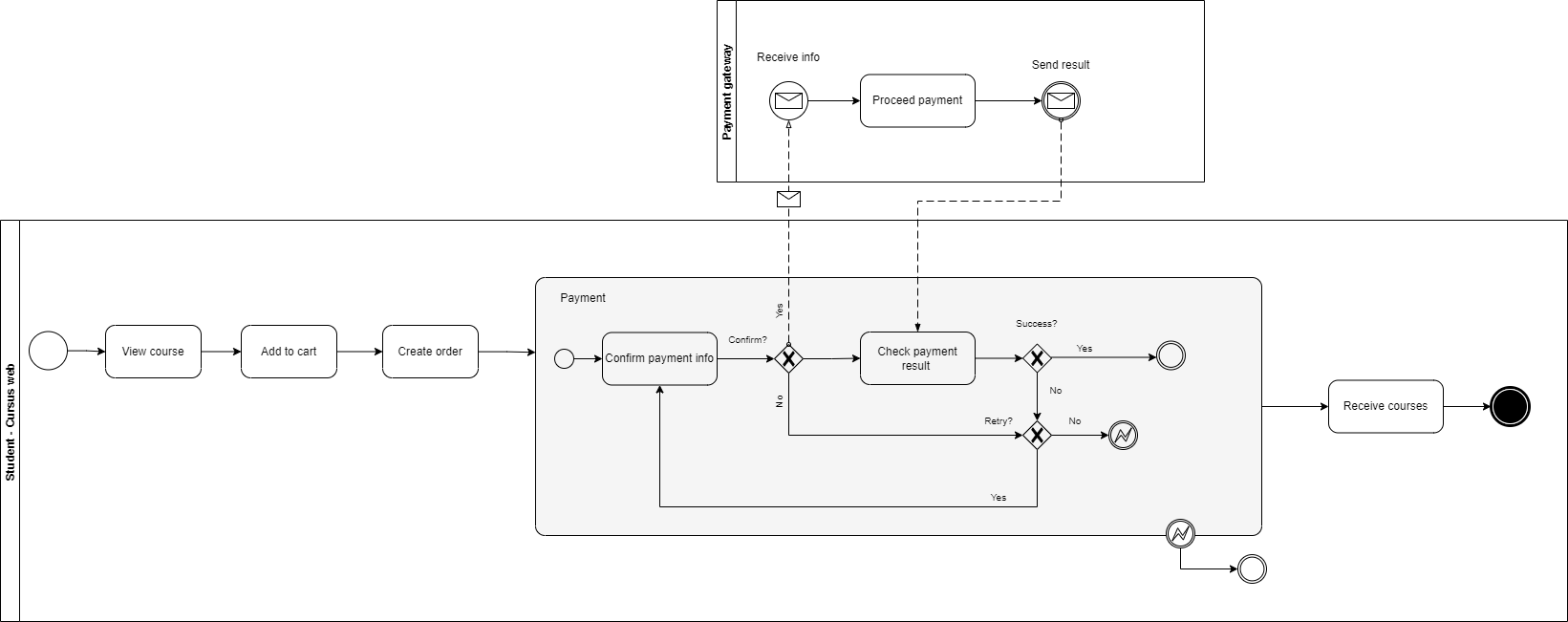
This section describes the general overview of the system functions or business processes which are depicted in different diagrams. It shows the types of users, their permissions granted to perform specific system functions and the sequence required to complete a business workflow (if any). As the section name implies, it is high-level which means it is not detailed enough. For detailed requirement specification, please see **3 Use Case Specifications** section below.

## System structure

The system architecture will be described in the software development phase.

## Workflow

This section shows the flow of tasks or steps taken by each users of the system in-order to complete a business process. The user’s actions are shown in each business process stage of the system and what happens before it can move to the next stage or revert to the previous.



## Use Case Diagram

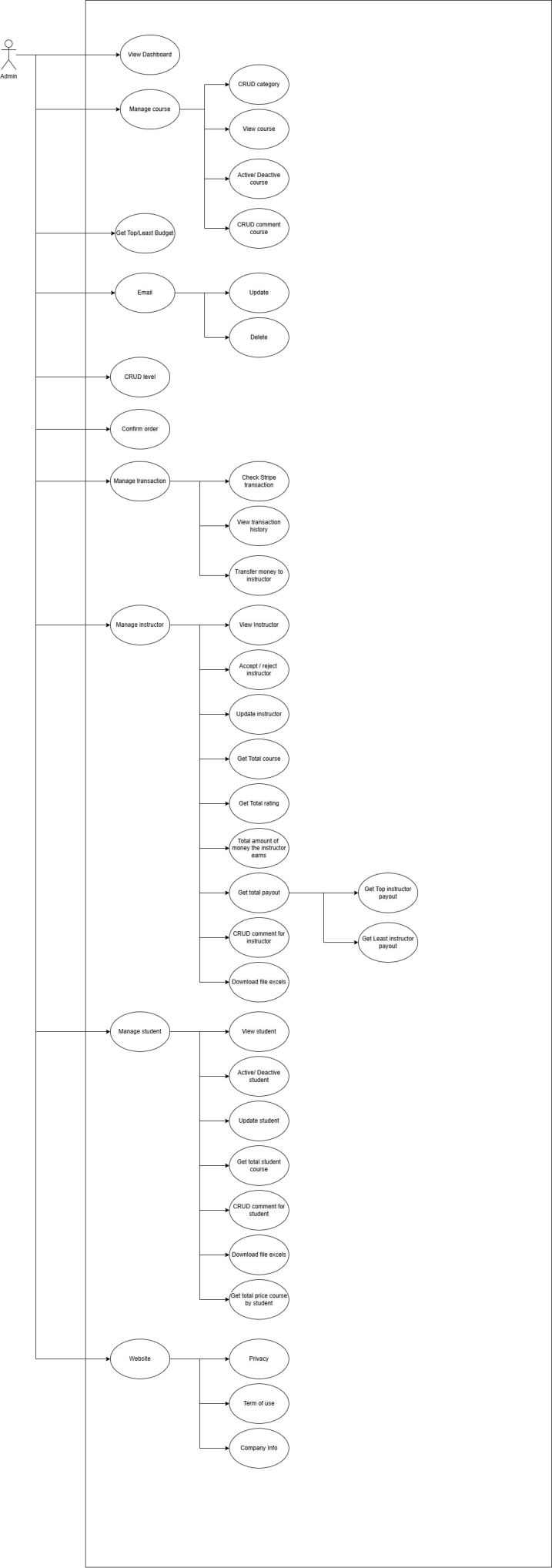
The use case diagram here shows the specific goal and objective or how the user interacts with the system. The eclipse in the system boundary represents the system use case/functions while the stickman represents the actor/user of the system. The line connecting the actor and the use case shows that the actor can perform that function in the system to achieve a goal.

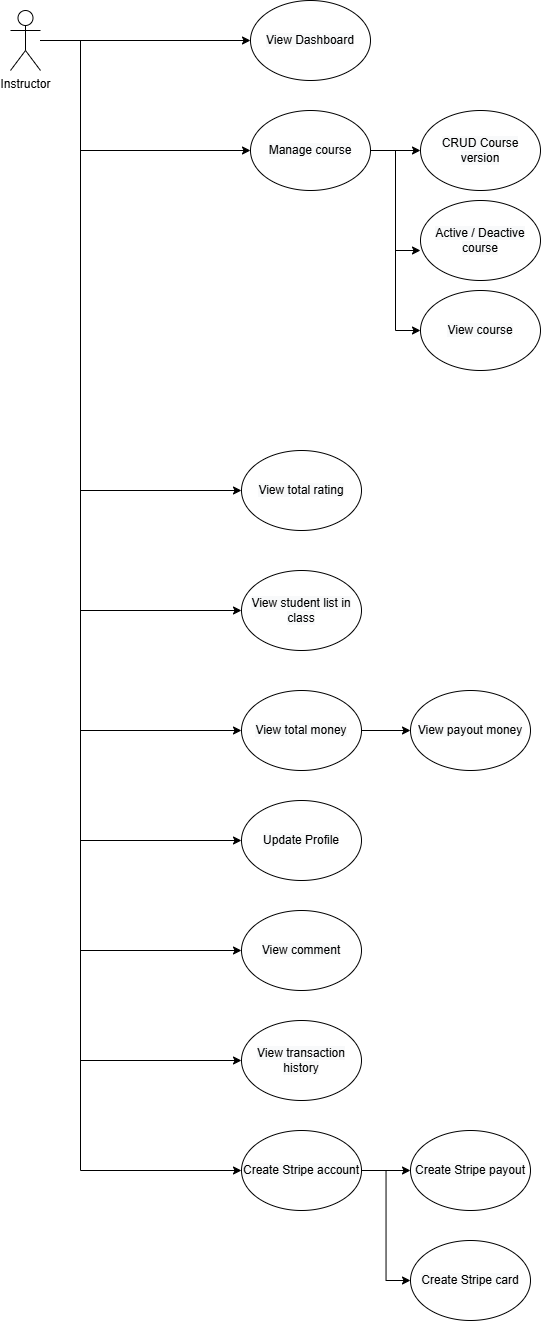
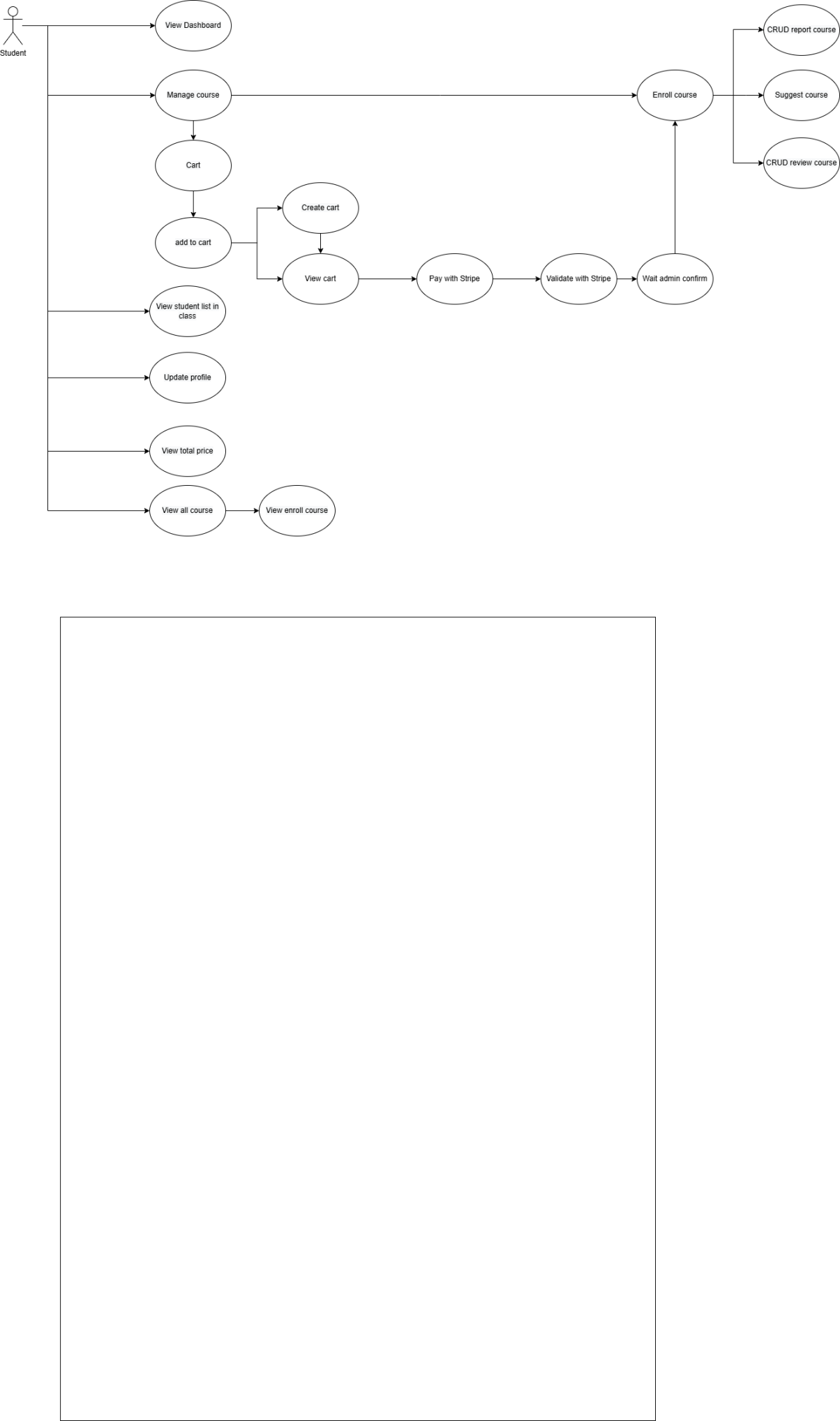
### General users

**Student:** the person who is in Cursus as Student Role, can view eligible courses, purchase and enroll into the courses, give feedback, manage personal user information.

**Instructor:** the person who provides the courses like teacher, mentor, support the course. This actor can get back the course fee that Student are paid.

**Guest:** the person who visits the Cursus and looks in to be the Student or Instructor.





### Administrator

The person who managed the Cursus system. This role mainly can support to approve/reject the Instructor’s courses and manage all courses, also can take care of the health of Cursus system.

## Permission matrix

Permission Matrix mapping functions and user roles for <<Cursus>> application is described as below:

Remark:

“O” means that user has permission on corresponding function. For more information about what the actor can do on that function, please refer to corresponding use case.

“X” means that user does not have permission on corresponding function.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **Guest** | **Student** | **Instructor** | **Admin** |
| **Authentication** |  |  |  |  |
| Register | O | O | O | X |
| Create password | X | O | O | X |
| Log in/ log out | O | O | O | O |
| **Homepage/ Dashboard** |  |  |  |  |
| View homepage as guest | O | O | O | X |
| View homepage by student | X | O | X | X |
| View Dashboard | O | O | O | O |
| View course enrolled. | X | O | X | X |
| View total of student/ course | X | X | O | X |
| View profile analytics | X | X | O | X |
| **Setting** |  |  |  |  |
| View/ edit profile | X | O | O | X |
| Change password | X | O | O | X |
| **Purchase courses** |  |  |  |  |
| View/ add/ edit/ delete shopping cart | X | O | X | X |
| Proceed purchase course | X | O | X | X |
| **Enroll course** |  |  |  |  |
| Enroll into a course | X | O | X | X |
| View tracking of the course | X | O | X | X |
| Review course | X | O | X | X |
| Report course | X | O | X | X |
| Saved course | X | O | X | X |
| **Dashboard for Instructor** |  |  |  |  |
| View course/ student analytics | X | X | O | X |
| **Manage course** |  |  |  |  |
| View list course | X | X | O | X |
| Create/ edit/ delete course | X | X | O | X |
| Submit course for approval | X | X | O | X |
| **Earning Affiliate** |  |  |  |  |
| View earning money | X | X | O | X |
| Payout | X | X | O | X |
| **Dashboard for admin** |  |  |  |  |
| View dashboard | X | X | X | O |
| **Manage categories** |  |  |  |  |
| Create/ view/ search/ edit/ delete categories | X | X | X | O |
| **Manage sub-categories** |  |  |  |  |
| Create/ view/ search/ edit/ delete sub-categories | X | X | X | O |
| **Manage student** |  |  |  |  |
| View list of students | X | X | X | O |
| Block/ unblock student | X | X | X | O |
| **Manage instructor** |  |  |  |  |
| View list of instructors | X | X | X | O |
| Block/ unblock instructor | X | X | X | O |
| Approve/ reject the course. | X | X | X | O |
| **Manage course** |  |  |  |  |
| View list of course | X | X | X | O |
| Review/ Approve/ Reject/ Block/ Unblock course | X | X | X | O |
| **Manage admin** |  |  |  |  |
| View list of admins | X | X | X | O |

# Use Case Specifications

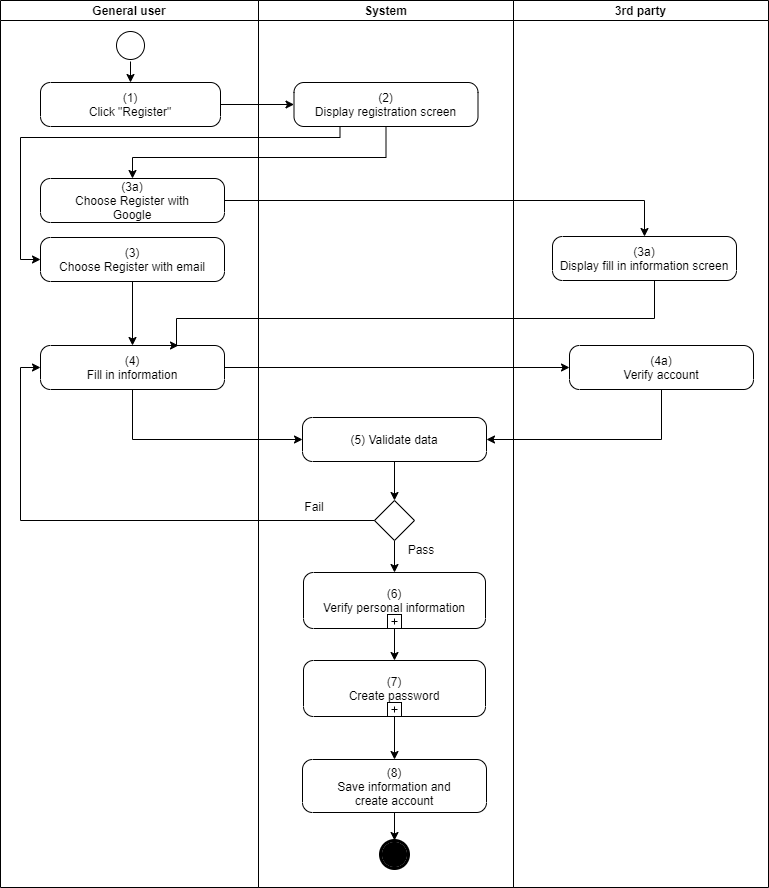
This section covers the system’s functional requirements which details what the system must do in terms of input, behavior and the expected output. It elicits the interaction between the actor(s) and the system, the system’s behavior and the results of their interactions.

## General functions

### UC 01: Sign up for Student

|  |  |
| --- | --- |
| **Objective:** | Allow general user to create new account on Cursus |
| **Actor:** | 1. Guest |
| **Trigger:** | 1. User chooses Sign up on Log in pop up 2. User chooses “Start Learning” at homepage |
| **Pre-condition:** | User has no account yet. |
| **Post-condition:** | User creates new account successfully |
| **Basic flow:** | 1. At Log in pop up/ Start learning, user choose the Register. 2. System displays registration screen. 3. User chooses:  - Register (with email), system will show the registration form.   - Register with Google (External registration), system redirects to registration form with detected email from Google.   1. User fills other information (email (if no), full name, address, password...) and submit. 2. System processes the data with validation as .NET default. 3. System response the result: - If success, send email confirmation (content of email: ET 1) & redirect to confirmation page - If failed, show the reason in the page. |
| **Alternative flow** | N/A |

**Activity flow**



**Business rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (2) | BR 1 | **Display rules:**   * When user click Log in button at homepage, website displays Log in pop up. At this pop up, user click Register to show registration screen. * Button “Register with Google” always shows above other fields. User clicked this button, redirect to log in screen of 3rd party – Google * Full name – max length: 200 characters * Email: user’s email. If user input, format must correct format of email as [abc@abc.xyz](mailto:abc@abc.xyz). If user enters wrong format 🡪 show MSG1 in red and below the field. * Button “Submit”: disable until user fills all information |
| (5) | BR 2 | **Validate rules:**  When user fills all information and click “Submit” button, system validates data as the rules below:   * If any required fields are left blank, the system displays error message MSG 2. MSG 2 is displayed below the blank field in red, italic format. * If username or email existed in system 🡪 show MSG 3 |
| (7) | BR 3 |  |

### UC 02: Sign up for Instructor

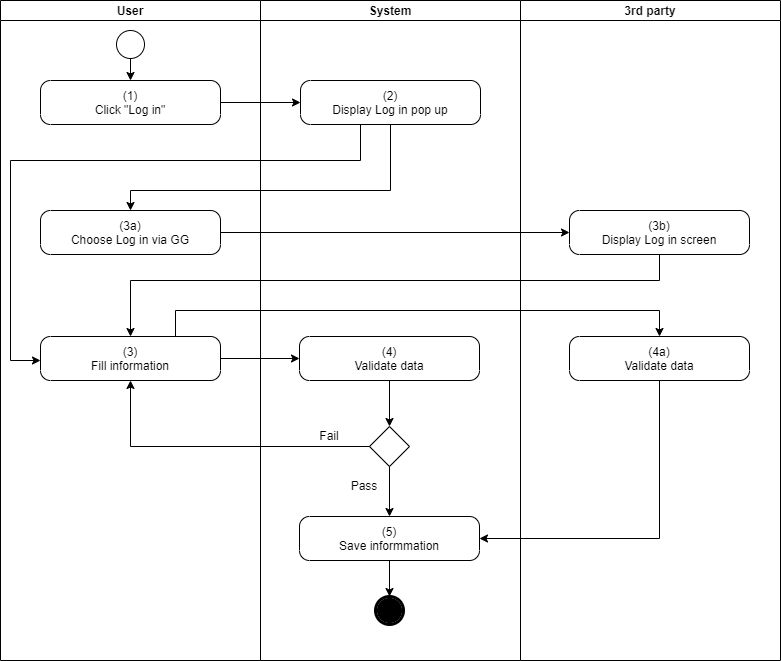
|  |  |
| --- | --- |
| **Objective:** | Allow general user to create new account on Cursus |
| **Actor:** | 1. Guest |
| **Trigger:** | 1. User chooses "Join to be Instructor” on Log in pop up |
| **Pre-condition:** | User has no account yet. |
| **Post-condition:** | User creates new account successfully |
| **Basic flow:** | 1. Click “Join to be Instructor” to redirect to Instructor registration screen. 2. System displays registration screen. 3. User chooses:  - Register (with email), system will show the registration form.   - Register with Google (External registration), system redirects to registration form with detected email from Google.   1. User fills other information (email (if no), full name, address, password, professional experience) and submit. 2. System processes the data with validation as .NET default. 3. System response the result: - If success, send email confirmation (content of email: ET 1) & redirect to confirmation page that inform that “Bạn đã đăng ký làm Giáo viên, chúng tôi sẽ xem xét thông tin của bạn trong X ngày.” - If failed, show the reason in the page. |
| **Alternative flow** | N/A |

**Business rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (2) | BR 1 | **Display rules:**   * When user click Log in button at homepage, website displays Log in pop up. At this pop up, user click Register to show registration screen. * Button “Register with Google” always shows above other fields. User clicked this button, redirect to log in screen of 3rd party – Google * Full name – max length: 200 characters * Email: user’s email. If user input, format must correct format of email as [abc@abc.xyz](mailto:abc@abc.xyz). If user enters wrong format 🡪 show MSG1 in red and below the field and notice “An error has occurs, please contact admin”. * Button “Submit”: disable until user fills all information |
| (5) | BR 2 | **Validate rules:**  When user fills all information and click “Submit” button, system validates data as the rules below:   * If any required fields are left blank, the system displays error message MSG 2. MSG 2 is displayed below the blank field in red, italic format. * If username or email existed in system 🡪 show MSG 3 |
| (7) | BR 3 | **Create rules:**  When system validate information successfully, the system proceeds to create new account:   * Display message MSG 4 * Add record for new user in DB * Save user information in DB * Send confirmation email to be corresponding. |

### UC 03: Sign in

|  |  |
| --- | --- |
| **Objective:** | Allow user Login to website. |
| **Actor:** | All types of users |
| **Trigger:** | User clicks “Log in” button at homepage |
| **Pre-condition:** | User has already had account in system |
| **Post-condition:** | User signs in successfully |
| **Basic flow** | (1). User clicks Log in at homepage  (2). System displays Log in pop up.  (3). User fills information: email + password  (4). System validates data & response |
| **Alternative flow** | (1). User clicks Log in at homepage  (2). System displays Log in pop up  (3a). User chooses Log in with Google (3b) 3rd party displays Log in screen  (3). User fills information  (4a). 3rd party validates data & response the data.  (5). System will direct to main menu after login success |

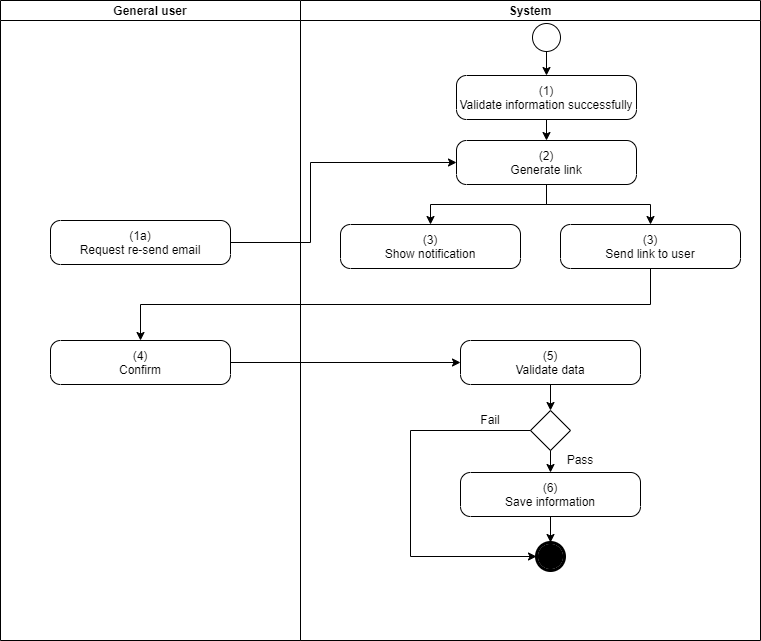


**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (2) | BR 4 | **Display rules:**  System displays Log in pop up with these fields:   * Button “Login with Google” is always on the top of pop up * Field “Email”: free text. This field allows user to enter the email which user signed up before. * Field “Password”: free text. * “Remember me”: check box. If user ticks this box, user will not enter email and password again. System will remember user information and allow to log in automatically. * Button “Log in”: call API to validate user’s information |
| (4) | BR 5 | **Validate rules:**  After user clicks “Log in” at Log in pop-up, system will validate based on the existing rules. The validation logic is:   * If the value of any mandatory fields is blank, system will show an error message for the required fields as MSG * If the email does not exist in database, the system will show an error message as MSG * If the password is not correct, the system will show an error message as MSG.  1. Error message will be show on top. For example: |
| (5) | BR 6 | **Save rules:**  System records information with last success login datetime for the user. |

### UC 04: Verify email

|  |  |
| --- | --- |
| **Objective:** | Allow user to verify email used to register account |
| **Actor:** | Guest. |
| **Trigger:** | System verifies user information successfully |
| **Pre-condition:** | User registered successfully with UC 01 with Register (with email) type. |
| **Post-condition:** | User account is activated. |
| **Basic flow:** | (1). User received the confirmation email from UC 01  (2) User clicks on the email link and redirect to the Cursus system confirmation page.  (3). System response the result:  - Success, active account.  - Failed: show reason and do not activate the account. |
| **Alternative flow:** | *N/A* |
| **Exception flow:** |  |

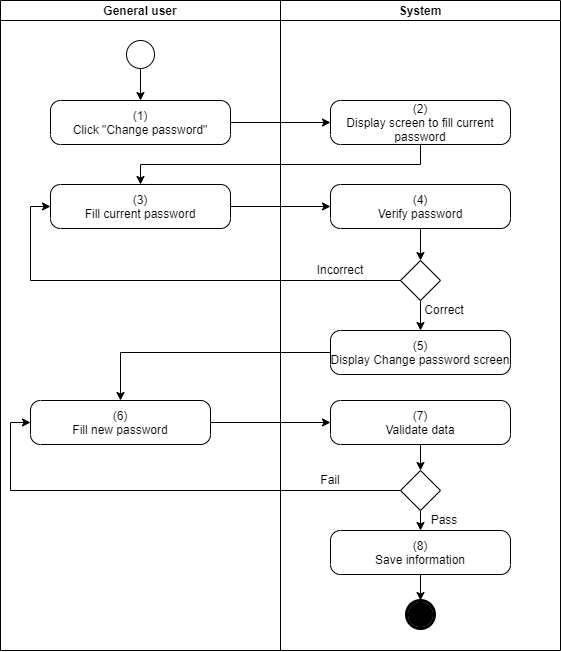
****

**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (1) | BR 11 | Content of email need to follow the template - ET1 |
| (2) | BR 13 | * Email link should be valid within 24hrs, otherwise, return the MSG 5 in Confirmation Result page. * If successful, return to the login page of UC 02 after 5 seconds. |

### UC 05: Change password

|  |  |
| --- | --- |
| **Objective:** | Allow user changes the password. |
| **Actor:** | All types of users |
| **Trigger:** | User chooses “Password” at Setting screen. |
| **Pre-condition:** | User logs in successfully. |
| **Post-condition:** | User changes the password successfully. |
| **Basic flow:** | 1. Login your account. 2. Go to “Your profile” 3. Change password 4. Enter your old password and add new password follow BR22 |



**Business Rules**

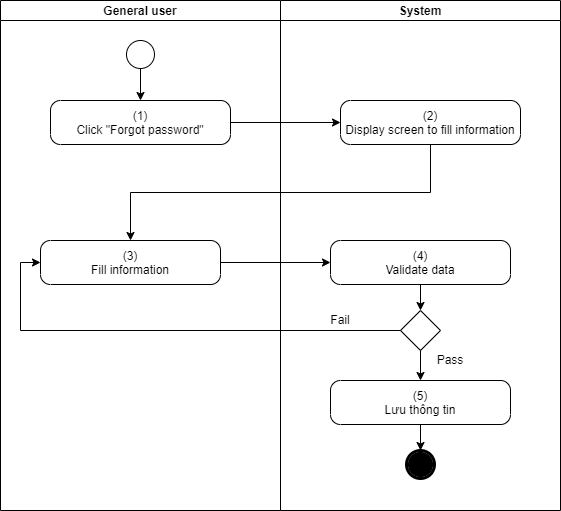
|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (1) | BR 21 | Show Change password form with information:   * Current password. * New password. * Confirm password. |
| (2) |  | **Validation rule:**   * New password can’t be same with current password. Show message MSG 22 if the same. |

### UC 06: Forgot password

|  |  |
| --- | --- |
| **Objective:** | Allow user creates new password if user forgot the current password. |
| **Actor:** | General user |
| **Trigger:** | User clicks “Forgot password” at Log in screen. |
| **Pre-condition:** | User already has account on Cursus. |
| **Post-condition:** | User creates new password successfully. |
| **Basic flow:** | 1. The user selects Forgot password.  2. The system displays a screen with email input.  3. User inputs the email then submits.  4. System checks the email:  - If existed in system, sends the email ET 2 to the input email.  - If not exist, response message MSG 8. |

**Activity flow**

**Step 1:**

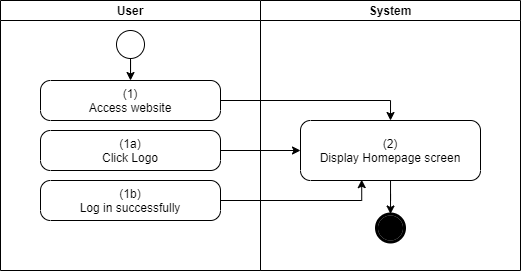


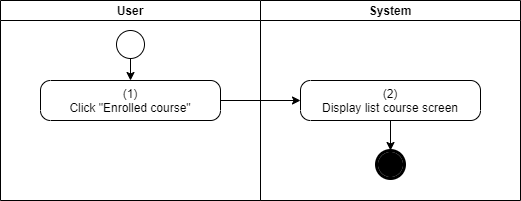
## General user

### UC 07: View Homepage

|  |  |
| --- | --- |
| **Objective:** | Allow user to view information at homepage |
| **Actor:** | General user |
| **Trigger:** | 1. User logs in successfully 2. User accesses website successfully |
| **Pre-condition:** | User accesses website successfully |
| **Post-condition:** | User views information at homepage |
| **Basic flow:** | 1. User can click on the logo  2. System redirects to the user info page. |

**Activity flow**





**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **Description** |
| (2) | BR 35 | * **Payment the course before can start course.** |

### **UC 08: Update profile**

|  |  |
| --- | --- |
| **Objective:** | Allow user to update phone number/email to personal information above |
| **Actor:** | Regular users |
| **Trigger:** | User selects email at the account screen |
| **Pre-conditions:** | User logged in successfully |
| **Post-conditions:** | User successfully added phone number/email |
| **Basic flows:** | 1. User selects email  2. Screen display system Enter email  3. User enters email  4. Information checking system  5. The system performs verification via email  6. The system stores information |

**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (1) | BR 30 | **Display rules:**  The system displays the email input screen |
| (2) | BR 31 | **Rules for checking information:**   * Users can change the information, except email. * After changed, the system should be updated. |
| (3) | BR 32 | **Save information**  After the email authentication process is successful, the system adds email information to the DB. |

### **UC 09: View and Search course (Admin and instructor \*)**

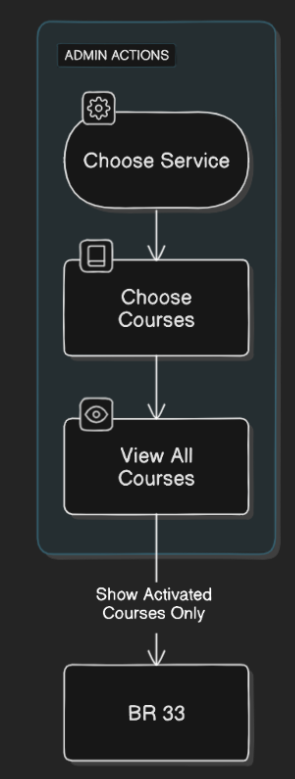
|  |  |
| --- | --- |
| **Objective:** | Allow users to view all courses on Cursus. |
| **Actor:** | General users |
| **Trigger:** | User clicks “Courses” or click on each category. |
| **Pre-conditions:** | User accesses the website successfully. |
| **Post-conditions:** | User views course information successfully. |
| **Basic flows:** | Instructor:   1. Click the logo 2. Choose workspace 3. Choose Service 4. Choose Courses and view all of your courses.   Admin:   1. Choose Service   2. Choose Courses and view all of your courses. |

**Activity flow**

· **Step 1:** Admin chooses Service.

· **Step 2:** Admin chooses Courses and views all courses.

* **BR 33:** Only activated courses can be shown in the list.
* **BR 34:** Admin can filter by search keyword (course name, instructor name).
* **BR 35:** Default has paging (10, 25, 50) per page. Default sorting is descending by Course Name and review points.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (1) | BR 33 | Only activated courses can be shown in the list. |
| (2) | BR 34 | Can filter by search keyword (course name, instructor name) |
| (3) | BR 35 | Default has paging (10, 25, 50) per page.  Short default as descending by Course Name, review points. |

### **UC 10: View enrolled course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view all courses that user registered before |
| **Actor:** | student |
| **Trigger:** | User clicks “Enrolled course” on the left menu |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User can view all of enrolled courses |
| **Basic flows:** | 1. Log in as student 2. Click Enrolled course for view. |
| **Alternative flows:** | · **AF1:** No Enrolled Courses: If the user is not enrolled in any courses, the system displays a message indicating no courses are enrolled.  · **AF2:** Data Retrieval Error: If there is an error retrieving the course list, the system shows an error message and options to retry or contact support. |

**Activity flow**

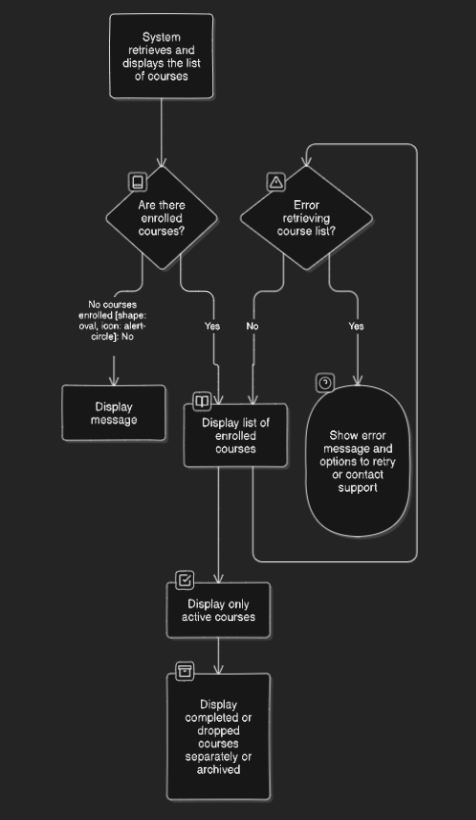
· **Step 1:** User logs in successfully as a student.

· **Step 2:** User clicks “Enrolled course” on the left menu.

* **BR 35:** The system must display only the courses the user is actively enrolled in. Courses that have been completed or dropped should be displayed separately or archived.

· **Step 3:** The system retrieves and displays the list of courses the user is actively enrolled in.

* **AF1:** If there are no enrolled courses, the system displays a message indicating no courses are enrolled.
* **AF2:** If there is an error retrieving the course list, the system shows an error message and options to retry or contact support.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 35 | * The system must display only the courses the user is actively enrolled in. Courses that have been completed or dropped should be displayed separately or archived. |

### **UC 11: View course by category**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view courses by category |
| **Actor:** | student |
| **Trigger:** | User clicks “Category” on the left menu |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User view courses by category successfully |
| **Basic flows:** | 1. Click the logo 2. Choose “my course” to view the list. 3. Search category course. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

· **Step 2:** User clicks the logo.

· **Step 3:** User chooses “My Course” to view the list.

· **Step 4:** User searches for a category course.

* **BR 36:** The system should categorize courses based on predefined themes or topics, ensuring that each course is accurately classified.

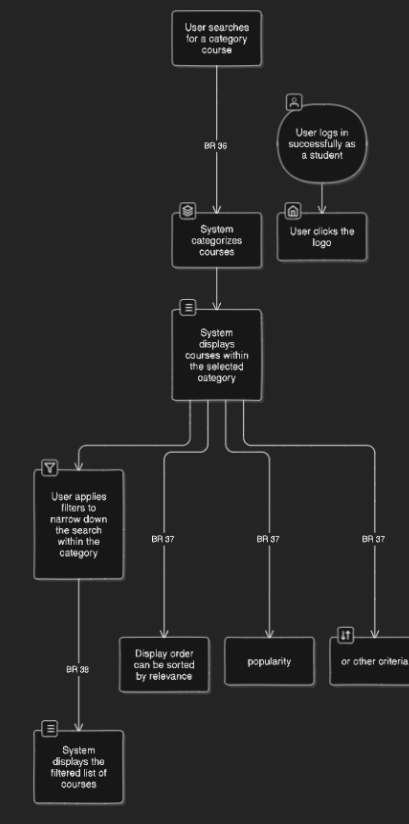
· **Step 5:** The system displays courses within the selected category.

* **BR 37:** The display order of courses within a category can be sorted by relevance, popularity, or other criteria.

· **Step 6:** User applies filters to narrow down the search within the category.

* **BR 38:** Users should be able to filter courses within a category by various factors such as level, duration, or rating.

· **Step 7:** The system displays the filtered list of courses



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 36 | The system should categorize courses based on predefined themes or topics, ensuring that each course is accurately classified. |
| (4) | BR 37 | The display order of courses within a category can be sorted by relevance, popularity, or other criteria. |
| (6) | BR 38 | Users should be able to filter courses within a category by various factors such as level, duration, or rating. |

### **UC 12: View list of subscriptions**

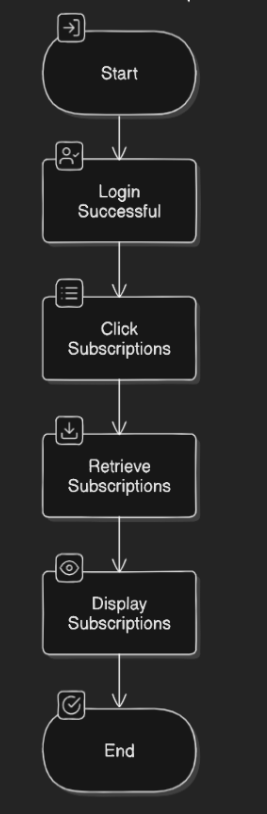
|  |  |
| --- | --- |
| **Objective:** | Allow user to view list of instructors that user subscribed to before. |
| **Actor:** | student |
| **Trigger:** | User clicks “Subscriptions” on the left menu. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User views course by category successfully. |
| **Basic flows:** | 1. Login as student 2. Choose “My course”. 3. View the list of all course and instructor that student has subscribed. |
| **Alternative flows:** | · **AF1:** No Subscriptions: If the user has not subscribed to any courses, the system displays a message indicating no subscriptions and suggests available courses.  · **AF2:** User Not Logged In: If the user is not logged in, they are prompted to log in before accessing the subscriptions list.  · **AF3:** Data Retrieval Error: If there's an issue retrieving the subscription data, an error message is displayed with options to retry or contact support. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

· **Step 2:** User clicks “Subscriptions” on the left menu.

· **Step 3:** The system retrieves and displays the list of all courses and instructors the student has subscribed to.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 39 | The system must display only active subscriptions and exclude expired or canceled ones |

### **UC 13: Search instructor**

|  |  |
| --- | --- |
| **Objective:** | Allow user to search instructor by name |
| **Actor:** | student |
| **Trigger:** | User clicks Search icon. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User searches instructor by name successfully |
| **Basic flows:** | 1. Login as student. 2. Use search bar to get instructor name. 3. Click enter to view instructor profile. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

· **Step 2:** User clicks the Search icon.

· **Step 3:** User enters the instructor's name in the search bar.

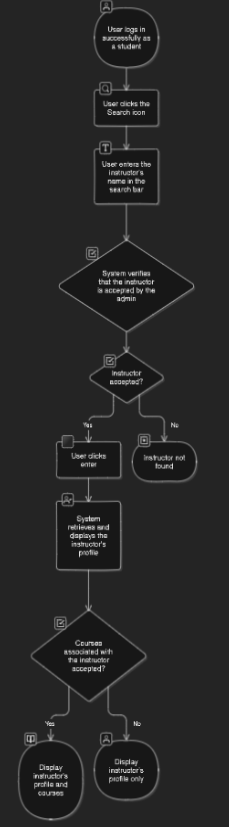
· **Step 4:** The system verifies that the instructor is accepted by the admin.

* **BR 40:** Instructor must be accepted by admin.

· **Step 5:** User clicks enter.

· **Step 6:** The system retrieves and displays the instructor's profile.

* **BR 41:** The courses associated with the instructor must be accepted by admin.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (4) | BR 40 | Instructor must be accepted by admin. |
| (6) | BR 41 | The course must be accepted by admin |

### **UC 14: Purchase the course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to purchase courses |
| **Actor:** | student |
| **Trigger:** | User clicks Add to cart icon |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User buys courses successfully |
| **Basic flow** | 1. Select a course for adding to cart 2. Check out the course payment 3. The course will review enroll button to start learning. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

· **Step 2:** User browses and selects a course to purchase.

· **Step 3:** User clicks the Add to Cart icon for the selected course.

· **Step 4:** The system processes the selected course and adds it to the cart.

* **BR 42:** The system must handle payment processing, including validation and handling of various payment methods.

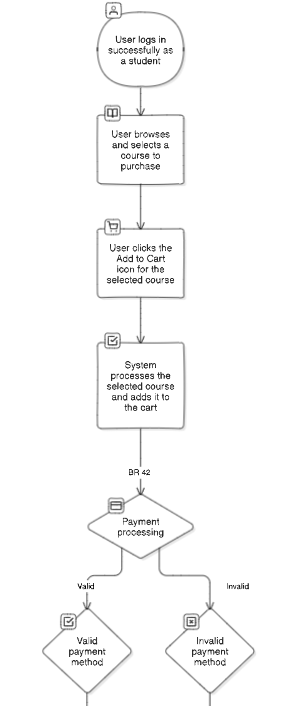
· **Step 5:** User proceeds to the checkout page.

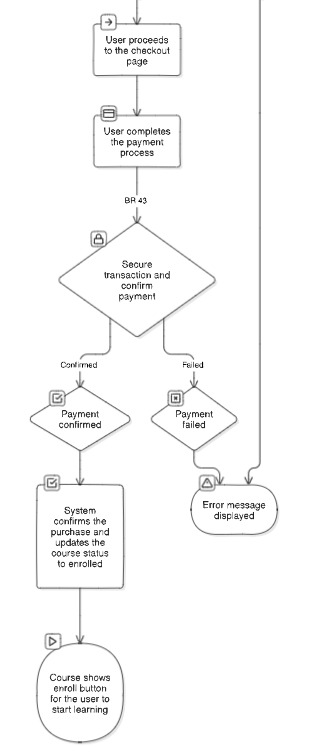
· **Step 6:** User completes the payment process.

* **BR 43:** The system must ensure that the transaction is secure and confirm the payment.

· **Step 7:** The system confirms the purchase and updates the course status to enrolled.

· **Step 8:** The course will now show an enroll button for the user to start learning.





**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (4) | BR 42 | Description of rule related to payment processing, such as validation or handling of payment methods |
| (6) | BR 43 |  |

### **UC 15: Enroll course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to Enroll course that user bought before. |
| **Actor:** | student |
| **Trigger:** | User chooses a course which was bought before |
| **Pre-conditions:** | 1. User logs in successfully as actor above. 2. The course was bought successfully |
| **Post-conditions:** | User can enroll the course |
| **Basic flows:** | 1. Login as student 2. Buy the course. 3. Go to the course page and you can click Enroll course to start the course. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

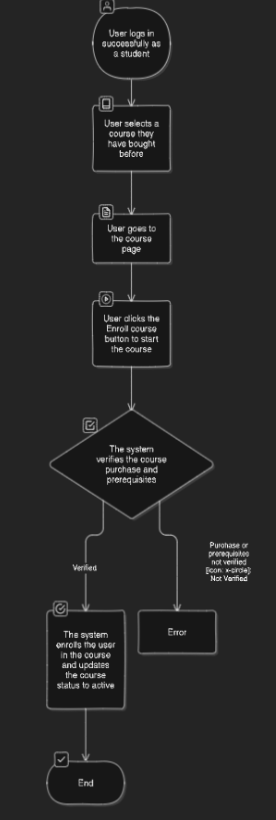
· **Step 2:** User selects a course they have bought before.

· **Step 3:** User goes to the course page.

· **Step 4:** User clicks the Enroll course button to start the course.

· **Step 5:** The system verifies the course purchase and prerequisites.

· **Step 6:** The system enrolls the user in the course and updates the course status to active.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 44 | Only courses that have been fully purchased and are available can be enrolled in. |
| (4) | BR 45 | Courses may have prerequisites or restrictions on enrollment, such as time limits for access after purchase. |

### **UC 16: Review course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to review the course that user has finished. |
| **Actor:** | student |
| **Trigger:** | User clicks “Review the course” at thank you screen |
| **Pre-conditions:** | 1. User logs in successfully as actor above. 2. The course has been completed by user |
| **Post-conditions:** | User reviews the course successfully |
| **Basic flows:** | 1. Login as student 2. Complete the course 3. Choose star and send comment for instructor. |

**Activity flow**

· **Step 1:** User logs in successfully as a student.

· **Step 2:** User completes the course.

· **Step 3:** User clicks “Review the course” at the thank you screen.

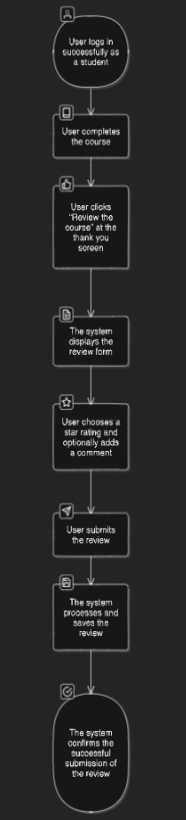
· **Step 4:** The system displays the review form.

· **Step 5:** User chooses a star rating and optionally adds a comment.

· **Step 6:** User submits the review.

· **Step 7:** The system processes and saves the review.

· **Step 8:** The system confirms the successful submission of the review.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 85 | Reviews can only be submitted for courses that have been marked as completed. |
| (4) | BR 86 | Reviews must include a star rating and may optionally include a comment. The system may enforce guidelines on comment length and content. |

### **UC 17: Report course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to report the courses which are violating the policy or the content does not match the description. |
| **Actor:** | student |
| **Trigger:** | User clicks “Report” button at each course. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User reports the course successfully. |
| **Basic flows:** | 1. **Login as Student:** The user logs in successfully.  2. **Access Course:** The user navigates to the course page.  3. **Submit Report:**   * **Click Report:** The user clicks the “Report” button. * **Select Reason:** The user selects a reason for the report, such as policy violation or misleading content. * **Provide Details:** The user may add comments or additional information. * **Submit Report:** The user submits the report, which is then logged by the system.   4. **Confirmation:** The system displays a message confirming that the report has been received and will be reviewed. |

**Activity flow**

· S**tep 1:** User logs in successfully as a student.

· **Step 2:** User navigates to the course page.

· **Step 3:** User clicks the “Report” button on the course page.

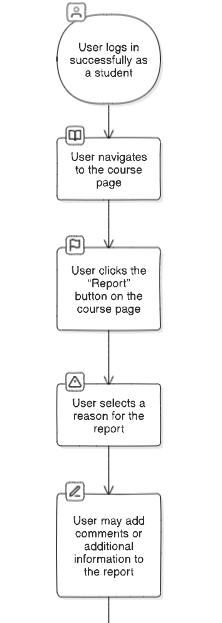
· **Step 4:** User selects a reason for the report (e.g., policy violation, misleading content).

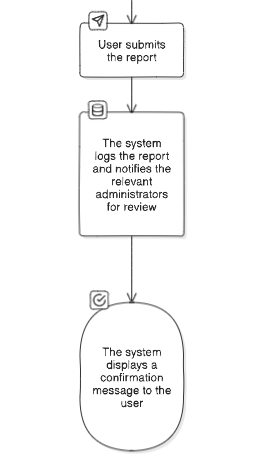
· **Step 5:** User may add comments or additional information to the report.

· **Step 6:** User submits the report.

· **Step 7:** The system logs the report and notifies the relevant administrators for review.

· **Step 8:** The system displays a confirmation message to the user, indicating that the report has been received and will be reviewed.





**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 133 | Reports must include a valid reason and may be subject to review by administrators. |
| (4) | BR 134 | Users can report each course only once unless new issues arise. Inappropriate reports may be flagged or rejected by the system. |

### **UC 18: View Dashboard**

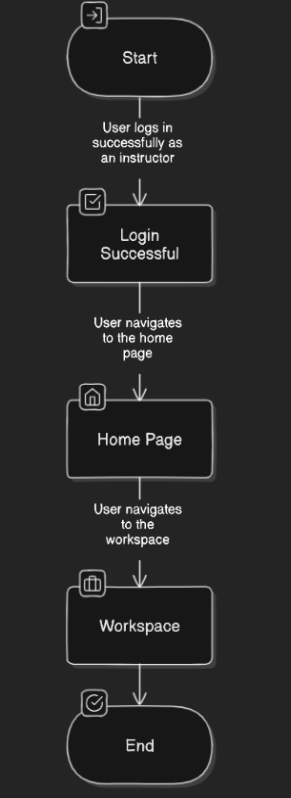
|  |  |
| --- | --- |
| **Objective:** | To allow instructors to view overall information on their dashboard, such as course statistics, student engagement, and other relevant metrics. |
| **Actor:** | Instructor |
| **Trigger:** | User logs in successfully. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User views dashboard successfully |
| **Basic flows:** | 1. Login as instructor 2. Go to home page 3. Go to workspace. |
| **Alternative flows:** |  |

**Activity flow**

· **Step 1:** User logs in successfully as an instructor.

· **Step 2:** User navigates to the home page.

· **Step 3:** User navigates to the workspace.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 60 | The system should ensure that the dashboard displays up-to-date and accurate information, including course statistics and student engagement metrics. |
| (4) | BR 61 | The system should provide tools for instructors to filter, sort, and customize the displayed data on their dashboard for more relevant insights. |

### **UC 19: View course/ student analytics**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view all information about user's course and view analytics about students |
| **Actor:** | Instructor |
| **Trigger:** | User chooses Course/ Analytics |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User views course/ analytic successfully. |
| **Basic flows:** | 1. Instructor view the list of student 2. Check student information |
| **Alternative flows:** |  |

**Activity flow**

· **Step 1:** User logs in successfully as an instructor.

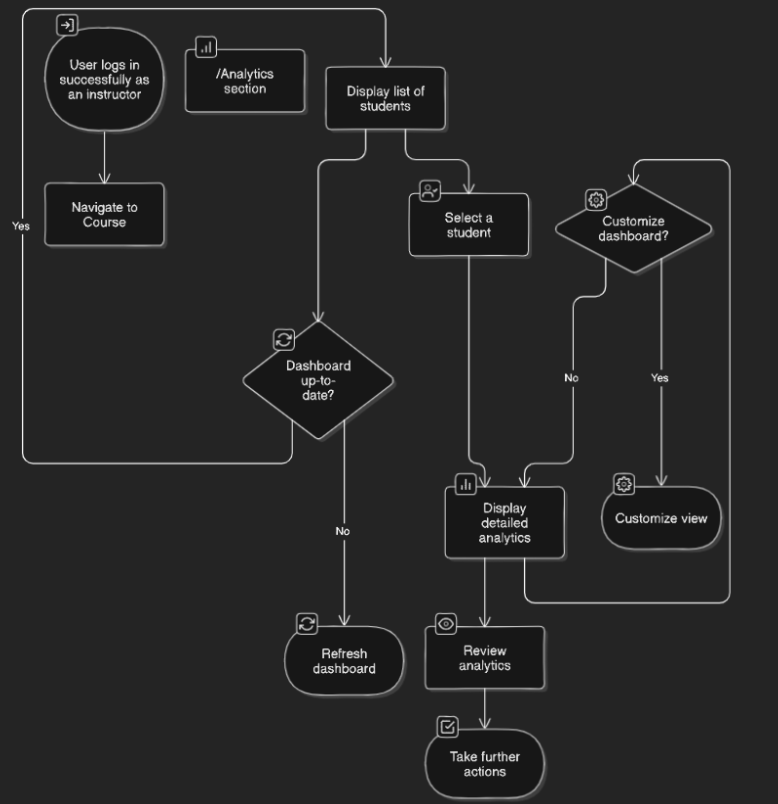
· **Step 2:** User navigates to the Course/Analytics section.

· **Step 3:** System displays a list of students enrolled in the instructor's courses.

· **Step 4:** Instructor selects a student from the list to view detailed information and analytics.

· **Step 5:** System displays detailed analytics about the selected student's engagement and performance in the course.

· **Step 6:** Instructor reviews the analytics and may take further actions based on the insights.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 60 | The dashboard should display up-to-date information, refreshed at regular intervals or upon instructor request. |
| (4) | BR 61 | Instructors should have the ability to customize the dashboard view, selecting which metrics to display and how to arrange them. |

### **UC 20: Create course**

|  |  |
| --- | --- |
| **Objective:** | Allow users to create new courses |
| **Actor:** | Instructor |
| **Trigger:** | 1. User clicks “Create course” button at homepage 2. User clicks “Create course” on the left menu |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User Create course successfully |
| **Basic flows:** | 1. Login as instructor 2. Go to workspace 3. Choose services. 4. Choose course. 5. Click Create course button. 6. Fill all field to create new course: Title, Code, Learning time, price, description, category, level. 7. Enter Submit button and wait for admin checking. 8. After check. The course will be available in the Course page. |

**Activity flow**

· **Step 1:** Instructor logs in.

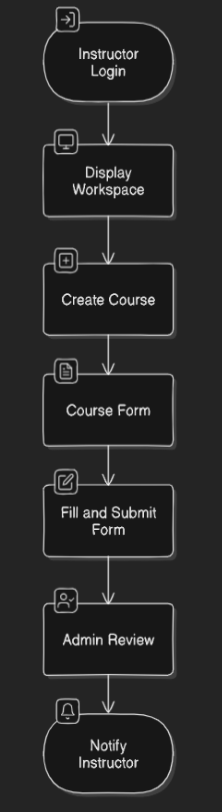
· **Step 2:** System displays workspace.

· **Step 3:** Instructor selects “Create course.”

· **Step 4:** System presents a form for course details.

· **Step 5:** Instructor fills in and submits the form.

· **Step 6:** System sends course for admin review and notifies the instructor of the status.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 48 | Courses must meet quality and content guidelines set by the platform and are subject to review and approval by an admin. |
| (5) | BR 49 | Instructors must complete all mandatory fields in the course creation form. Incomplete submissions will prompt the user to provide the missing information before submission. |

### **UC 21: Edit course**

|  |  |
| --- | --- |
| **Objective:** | Allow users to edit course information |
| **Actor:** | Instructor |
| **Trigger:** | User clicks Edit icon at list course screen |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User edits the course successfully. |
| **Basic flows:** | 1. Click the course in the Course page. 2. Click Clone button and edit that clone. 3. Enter new value fields and can create section. |

**Activity flow**

**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 50 |  |
| (4) | BR 51 |  |
| (6) | BR 52 |  |

### **UC 22: Re-submit course**

|  |  |
| --- | --- |
| **Objective:** | Allow users to modify and resubmit course |
| **Actor:** | Instructor |
| **Trigger:** | User goes to the course |
| **Pre-conditions:** | Course is submitted and got rejection from admin. |
| **Post-conditions:** | User resubmit the course successfully. |
| **Basic flows:** | 1. User access the rejected course. 2. User modifies the information. 3. User submit the course again. |

**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 50 | Instructors must provide valid data for all required fields. Incomplete or incorrect data will prevent saving changes. |
| (4) | BR 51 | The system should validate all entered data for format and consistency before allowing the changes to be saved. |
| (6) | BR 52 | Edited courses must be reviewed and approved by an admin if required by the platform’s policies before being made live. |

### 

### **UC 23: Delete course**

|  |  |
| --- | --- |
| **Objective:** | Allow user to delete course. |
| **Actor:** | Instructor |
| **Trigger:** | User clicks Delete icon at list course screen. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User deletes course successfully |
| **Basic flows:** | 1. Click the course in the Course page.  2.Click Delete button and edit that clone. |

**Activity flow**

· **Step 1:** Instructor logs in.

· **Step 2:** System displays the Course page.

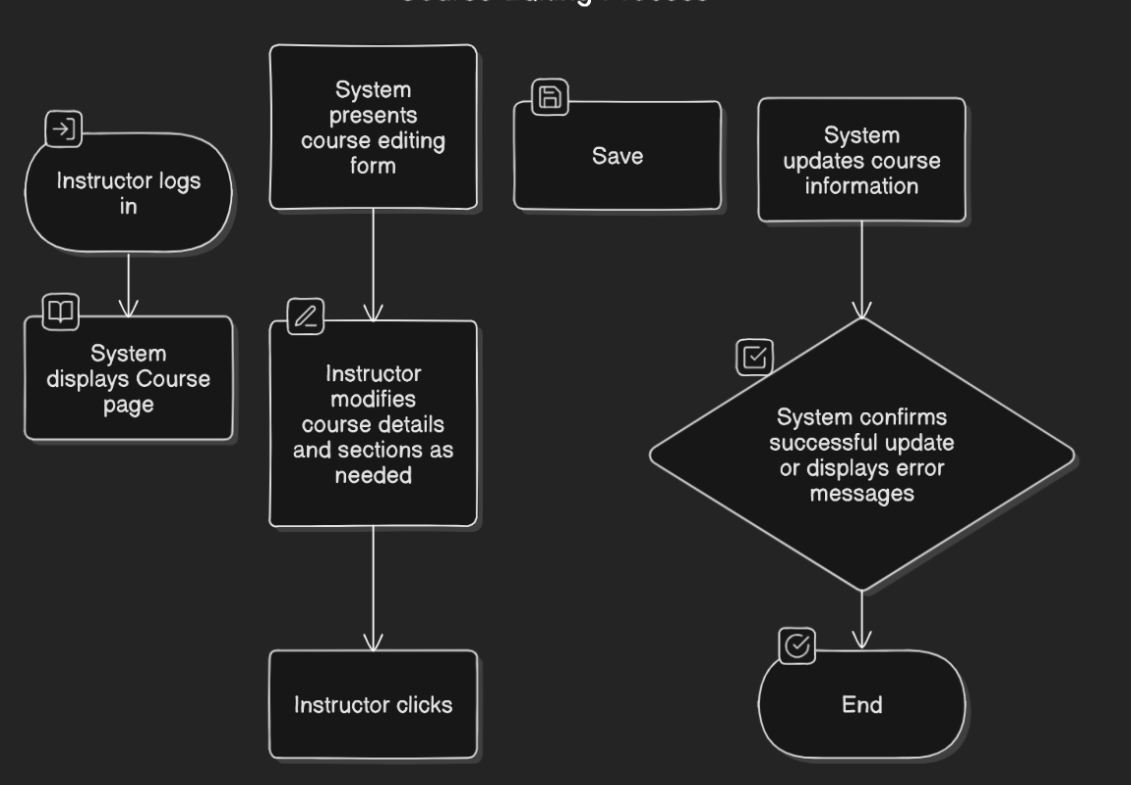
· **Step 3:** Instructor selects a course and clicks the “Edit” icon.

· **Step 4:** System presents the course editing form.

· **Step 5:** Instructor modifies course details and sections as needed.

· **Step 6:** Instructor clicks “Save,” and the system updates the course information.

· **Step 7:** System confirms the successful update or displays error messages if applicable.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 53 | **Display rules**  The system displays a confirmation message MSG 13. MSG 13 has 2 buttons "Agree" and "Cancel" |
| (4) | BR 54 | **Rules for saving information**   * If the user selects “Cancel”, the use case stops and the system displays the previous screen. * If the user chooses "Agree", the system will: Delete community information from the system * Cannot delete the course that has student enrollment. |

### **UC 24: View reviews**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view all reviews about user's course |
| **Actor:** | Instructor |
| **Trigger:** | User clicks “review” on the left menu |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User views all reviews successfully |
| **Basic flows:** | 1. Instructor login and select their course to view all reviews from student. |

**Activity flow**

· **Step 1:** Instructor logs in.

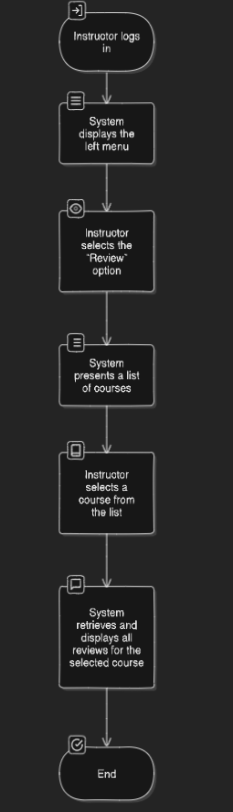
· **Step 2:** System displays the left menu.

· **Step 3:** Instructor selects the “Review” option.

· **Step 4:** System presents a list of courses.

· **Step 5:** Instructor selects a course from the list.

· **Step 6:** System retrieves and displays all reviews for the selected course.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 58 | The system must display reviews in chronological order or according to any filtering criteria set by the instructor. |
| (4) | BR 59 | Reviews should include all relevant details such as rating, comment, and reviewer information. The system should handle any inconsistencies or errors in review data appropriately. |

### **UC 25: View earnings analytics**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view the amount of money that earned from courses |
| **Actor:** | Instructor |
| **Trigger:** | User clicks “Earning” on the left menu |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User view earnings analytic successfully |
| **Basic flows:** | 1. *User goes to Earning page.* 2. *User can see the earning list by months as list.* |
| **Alternative flows:** | · **AF1:** No Earnings Data: If there is no earnings data available for the selected period, the system displays a message indicating that no data is available.  · **AF2:** Data Retrieval Error: If there is an error retrieving or displaying earnings data, the system shows an error message and provides options to retry or contact support. |

**Activity flow**

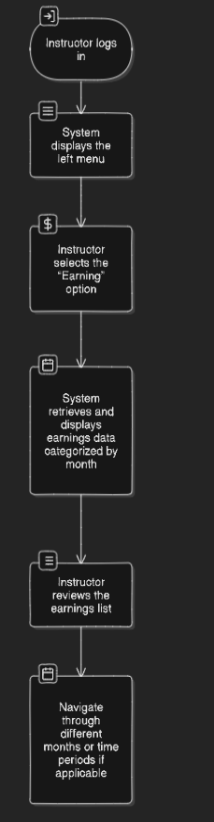
· **Step 1:** Instructor logs in.

· **Step 2:** System displays the left menu.

· **Step 3:** Instructor selects the “Earning” option.

· **Step 4:** System retrieves and displays earnings data categorized by month.

· **Step 5:** Instructor reviews the earnings list and can navigate through different months or time periods if applicable.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 60 | The system must display earnings data accurately and ensure it is updated regularly to reflect the latest earnings. |
| (4) | BR 61 | Earnings data should be categorized by month, and the system should provide options to filter or sort the data as needed. Any discrepancies or errors in data should be handled appropriately. |

### **UC 26: Payout**

|  |  |
| --- | --- |
| **Objective:** | Allow user to withdraw money to the bank account |
| **Actor:** | Instructor |
| **Trigger:** | User clicks “Payout” on the the earnings analysis page. |
| **Pre-conditions:** | User logs in successfully as actor above. |
| **Post-conditions:** | User withdraws money successfully |
| **Basic flows:** | 1. Click Stripe button, sign in Stripe. 2. Instructor can get money from Stripe to bank. 3. Input amount of money, Strike will send money to the bank. |

**Activity flow**

**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 62 | User cannot withdraw money if zero. |
| (4) | BR 63 | User will receive the email for the Payout |

## **Admin**

### **UC 27: Login**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to login into administration site. |
| **Actor:** | System admin |
| **Trigger:** | Admin logs in successfully. |
| **Pre-conditions:** | N/A |
| **Post-conditions:** | Admin has account. |
| **Basic flows:** | 1. User access to admin site (ex: abc.com\admin) then input the email & password.  2. User can access the admin site. |

### **UC 28: View dashboard**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to view dashboard |
| **Actor:** | System admin |
| **Trigger:** | Admin logs in successfully. |
| **Pre-conditions:** | Admin logs in successfully. |
| **Post-conditions:** | Admin views dashboard successfully. |
| **Basic flows:** | 1. User logs in successfully  2. System displays dashboard screen |

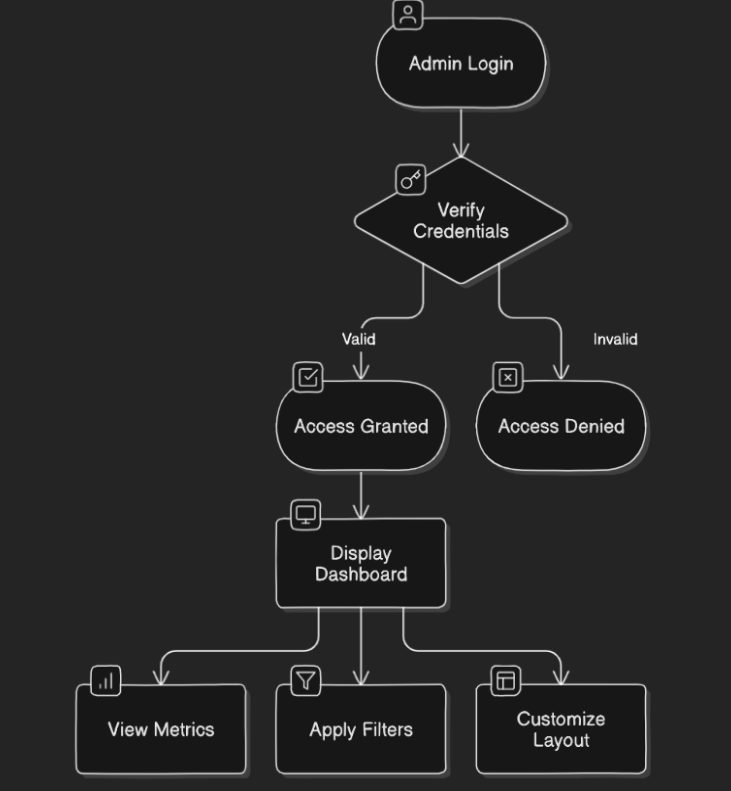
**Activity flow**

· **Step 1:** Admin logs in to the system.

· **Step 2:** System verifies credentials and grants access.

· **Step 3:** System displays the dashboard screen with key metrics and analytic.

· **Step 4:** Admin interacts with the dashboard, including viewing metrics, applying filters, or customizing the layout if applicable.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 87 | The dashboard must provide real-time or regularly updated information on key system metrics and status. The data should be accurate and reflect the current state of the system. |

### **UC 29: View list of students/instructors**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to manage the user of this system by each type. |
| **Actor:** | Admin |
| **Trigger:** | Admin logs in successfully |
| **Pre-conditions:** | Admin is logged in. |
| **Post-conditions:** | The admin can view and manage the list of students and instructors. |
| **Basic flows:** | 1. Admin accesses the admin page.  2. User can see the Users menu. At here:  - Choose the “Student” sub-menu to manage student.  - Choose the “Instructor” sub-menu to manage the instructor. |

**Activity flow**

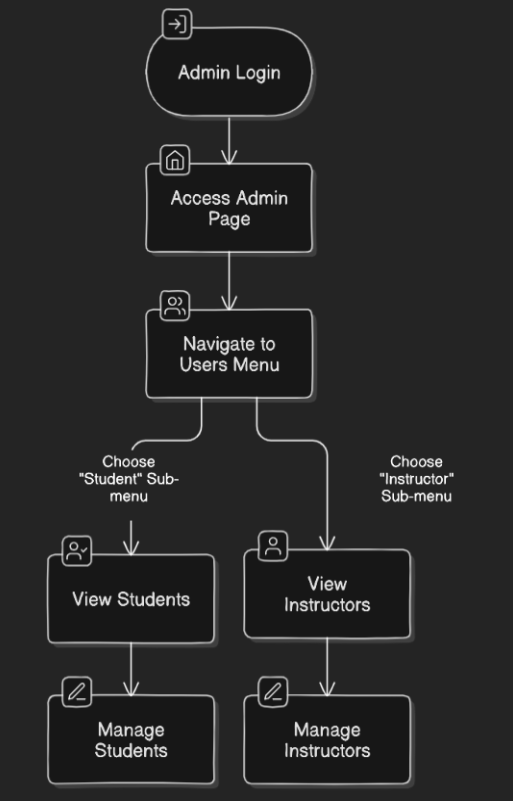
· **Step 1:** Admin logs in successfully.

· **Step 2:** Admin accesses the admin page.

· **Step 3:** Admin navigates to the Users menu.

· **Step 4:** Admin chooses the “Student” sub-menu to view and manage the list of students.

· **Step 5:** Admin chooses the “Instructor” sub-menu to view and manage the list of instructors.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 91 | The system must accurately display lists of students and instructors, including all relevant user details. |
| (4) | BR 92 | Admins should be able to perform actions on user accounts, such as viewing details, editing information, or managing statuses. |
| (6) | BR 93 | The system should provide options for searching, filtering, and sorting the lists of students and instructors for easier management. |
| (8) | BR 94 | Any issues with retrieving or displaying user data must be handled gracefully, with appropriate error messages and options for retrying or contacting support. |

### **UC 30: Approve or Reject Instructor registration**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to approve/reject the instructor of this system by each type. |
| **Actor:** | Admin |
| **Trigger:** | Admin log in successfully |
| **Pre-conditions:** | Admin access to the instructor submenu under User menu. |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Admin selects the instructor that is registered and has not activated at the instructor list. 2. Admin choses:  * Approve: instructor account is activated with email ET 3 * Reject: instructor account keeps inactivate with ET 4. |

### **UC 31: View list of categories**

|  |  |
| --- | --- |
| **Objective:** | Allow user to view list of category |
| **Actor:** | User (Student, Instructor, Admin) |
| **Trigger:** | User log in successfully |
| **Pre-conditions:** | User access to the Services sub-menu |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Click Service button 2. Click Categories button to show |

**Activity flow**

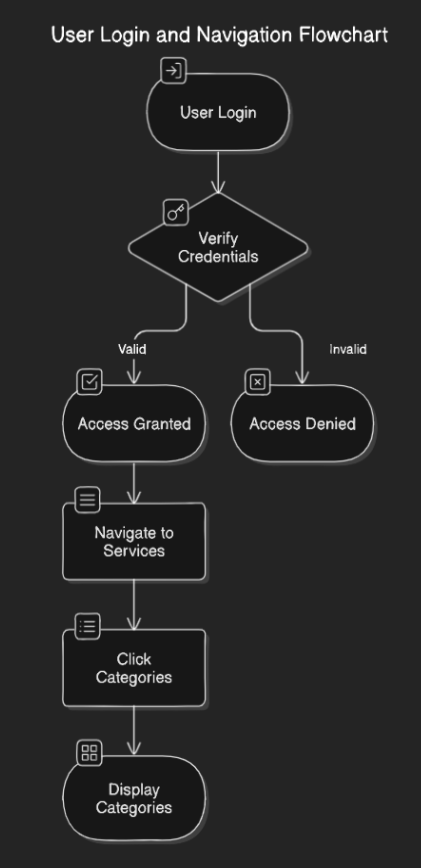
· **Step 1:** User logs in to the system.

· **Step 2:** System verifies the user’s credentials and grants access.

· **Step 3:** User navigates to the Services sub-menu.

· **Step 4:** User clicks the "Categories" button.

· **Step 5:** System retrieves and displays the list of categories.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 95 | The system must display all available categories accurately and completely, including any relevant details such as category name and description. |
| (4) | BR 96 | Users should be able to interact with the category list, including actions like viewing details, filtering, or sorting categories based on specific criteria. |

### **UC 32: Create a category**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to create list of category |
| **Actor:** | Admin |
| **Trigger:** | Admin Login Successfully |
| **Pre-conditions:** | User access to the Services sub-menu |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Go to the Categories 2. Create category 3. Fill name, description and parent 4. Click submit |

**Activity flow**

· **Step 1:** Admin logs in successfully.

· **Step 2:** Admin navigates to the Services sub-menu.

· **Step 3:** Admin selects the "Categories" option.

· **Step 4:** Admin clicks the "Create Category" button.

· **Step 5:** The system displays a form for creating a new category.

· **Step 6:** Admin fills in the category name, description, and selects a parent category.

* **BR 97:** The system validates the uniqueness of the category name to prevent duplicates.

· **Step 7:** Admin clicks the "Submit" button.

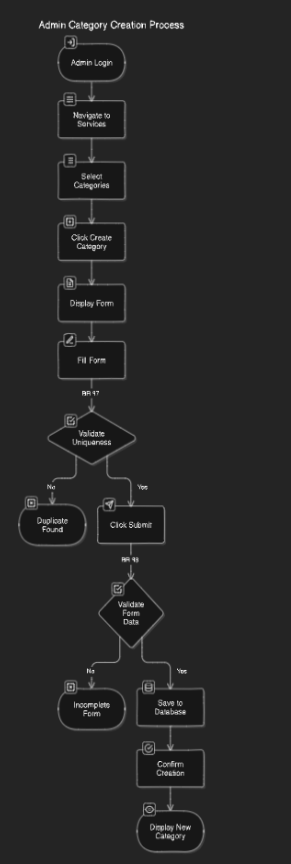
· **Step 8:** The system validates the form data:

* **BR 98:** The system ensures that all required fields (name, description, parent category) are completed before the category can be submitted.

· **Step 9:** If validation passes, the system saves the new category to the database.

· **Step 10:** The system confirms the successful creation to the admin.

· **Step 11:** The new category is now visible in the list of categories.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 97 | The system must validate the uniqueness of the category name to prevent duplicates. |
| (4) | BR 98 | All required fields (name, description, parent category) must be completed before the category can be submitted. |

### **UC 33: Edit a category**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to edit list of category |
| **Actor:** | Admin |
| **Trigger:** | Admin Login Successfully |
| **Pre-conditions:** | User access to the Services sub-menu |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Click edit the categories show in the list. 2. Edit field of that category 3. Click submit button |

**Activity flow**

· Admin logs in successfully.

· Admin navigates to the Services sub-menu.

· Admin clicks the "Edit" button next to a category in the list.

· The system displays the category's current details in editable fields.

· Admin makes changes to the category fields.

· Admin clicks the "Submit" button.

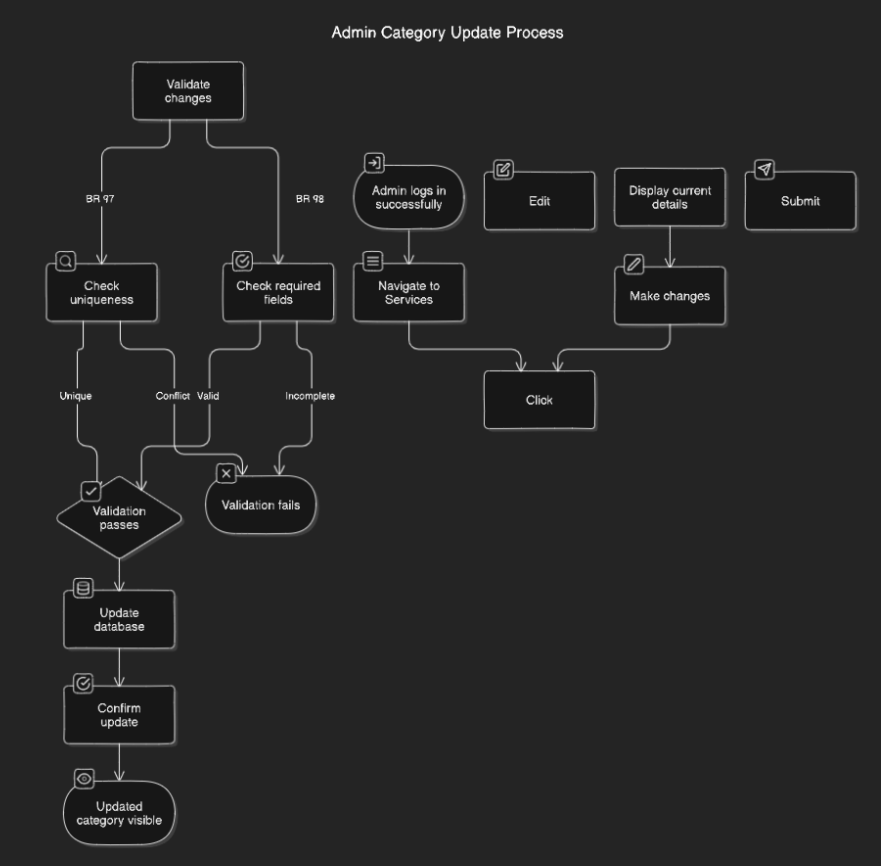
· The system validates the changes:

* **BR 97:** The system ensures that the updated category name is unique and does not conflict with other existing categories.
* **BR 98:** The system ensures that all required fields (name, description, parent category) are valid and completed before allowing changes to be submitted.

· If validation passes, the system updates the category details in the database.

· The system confirms the successful update to the admin.

· The updated category is now visible in the list of categories.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 97 | The system must ensure that the updated category name is unique and does not conflict with other existing categories. |
| (4) | BR 98 | All required fields (name, description, parent category) must be valid and completed before changes can be submitted. |

### **UC 34: Deactivate category**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to deactivate list of category |
| **Actor:** | Admin |
| **Trigger:** | Admin Login Successfully |
| **Pre-conditions:** | User access to the Services sub-menu |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Click edit the categories show in the list. 2. Edit status of that category to ”Deactivate” |

**Activity flow**

· **Step 1**: Admin logs in.

· **Step 2**: System verifies admin credentials.

· **Step 3**: Admin navigates to the Services sub-menu.

· **Step 4**: System displays a list of categories.

· **Step 5**: Admin selects a category to edit.

· **Step 6**: System displays category details.

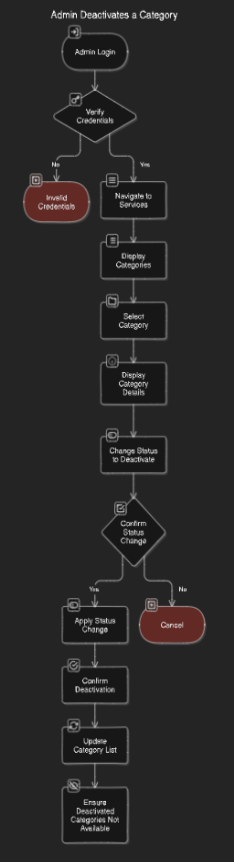
· **Step 7**: Admin changes the status to “Deactivate”.

· **Step 8**: Admin confirms the status change.

· **Step 9**: System applies the status change to “Deactivate”. (BR 97)

· **Step 10**: System confirms the category has been deactivated and updates the list.

· **Step 11**: System ensures deactivated categories are no longer available for selection in new content or courses but remain visible in historical data. (BR 98)



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 97 | The system must ensure that the status change to “Deactivate” is applied correctly and reflects the category as inactive. |
| (4) | BR 98 | Deactivated categories should no longer be available for selection in any new content or courses but should still be visible in historical data if applicable. |

### **UC 35: Block/ unblock student/ instructor**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to block/unlock student/instructor of category |
| **Actor:** | Admin |
| **Trigger:** | Admin Login Successfully |
| **Pre-conditions:** | User access to the Services sub-menu |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Click Student or Instructor to view the list of user. 2. Press Detail button 3. Click lock to locked the account |

**Activity flow**

· **Step 1:** Admin logs in to the system.

· **Step 2:** System verifies credentials and grants access.

· **Step 3:** Admin accesses the Services sub-menu and navigates to the Users section.

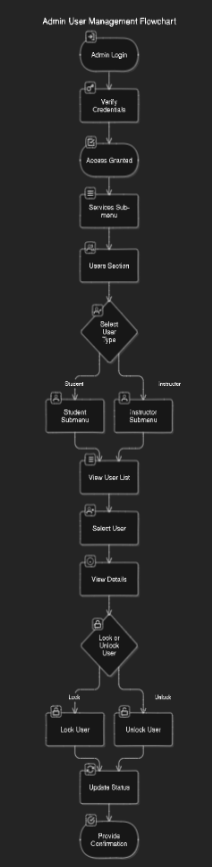
· **Step 4:** Admin selects either "Student" or "Instructor" submenu.

· **Step 5:** Admin views the list of users.

· **Step 6:** Admin selects a user and clicks the "Detail" button to view more information.

· **Step 7:** Admin clicks the "Lock" button to block the user or the "Unlock" button to unblock the user.

· **Step 8:** System updates the user’s status accordingly and provides confirmation.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 106 | The system must ensure that the block or unblock action is applied correctly to the user’s account. |
| (4) | BR 107 | Blocked users should not have access to the system until their account is unblocked, while unblocked users should regain full access. |

### **UC 36: View list of courses**

|  |  |
| --- | --- |
| **Objective:** | To allow the admin to view a comprehensive list of all courses available in the system. |
| **Actor:** | Admin |
| **Trigger:** | The action is triggered when the admin logs in successfully. |
| **Pre-conditions:** | The admin must be logged in and have access to the appropriate section of the system where courses are managed. |
| **Post-conditions:** | The admin is able to view and browse the list of courses. |
| **Basic flows:** | · **Log In:**   * · The admin logs in successfully to the system.   · **Access Courses Section:**   * · The admin navigates to the Courses section within the admin interface.   · **View Course List:**   * · The admin views the list of all available courses. This may include course details such as title, description, instructor, and status.   · **Filter/Sort Courses (Optional):**   * · The admin can filter or sort the course list based on various criteria (e.g., category, status, instructor) to find specific courses more efficiently. |

**Activity flow**

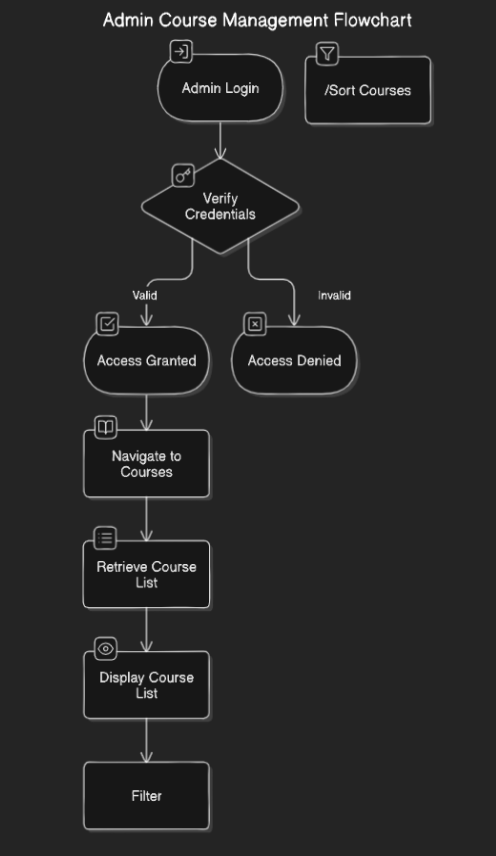
· **Step 1:** Admin logs in to the system.

· **Step 2:** System verifies credentials and grants access.

· **Step 3:** Admin navigates to the Courses section.

· **Step 4:** System retrieves and displays the list of all courses.

· **Step 5:** Admin views the course list and uses any available filtering or sorting options to organize the view.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 108 | The system must display all courses with their relevant details, including title, description, instructor, and status. |
| (4) | BR 109 | The admin should be able to use filtering and sorting options to efficiently navigate and manage the list of courses. |

### **UC 37: View details of a course**

|  |  |
| --- | --- |
| **Objective:** | To allow the admin to view detailed information about a specific course. |
| **Actor:** | Admin |
| **Trigger:** | The action is triggered when the admin selects a course to view its details |
| **Pre-conditions:** | The admin must be logged in and have access to the Courses section where course details can be viewed. |
| **Post-conditions:** | The admin views the detailed information of the selected course. |
| **Basic flows:** | · **Log In:**   * · The admin logs in successfully to the system.   · **Access Courses Section:**   * · The admin navigates to the Courses section within the admin interface.   · **Select Course:**   * · The admin selects a specific course from the list to view its details.   · **View Course Details:**   * · The system displays detailed information about the selected course, including title, description, instructor, category, enrollment status, and other relevant data. |

**Activity flow**   
· **Step 1:** Admin logs in to the system.

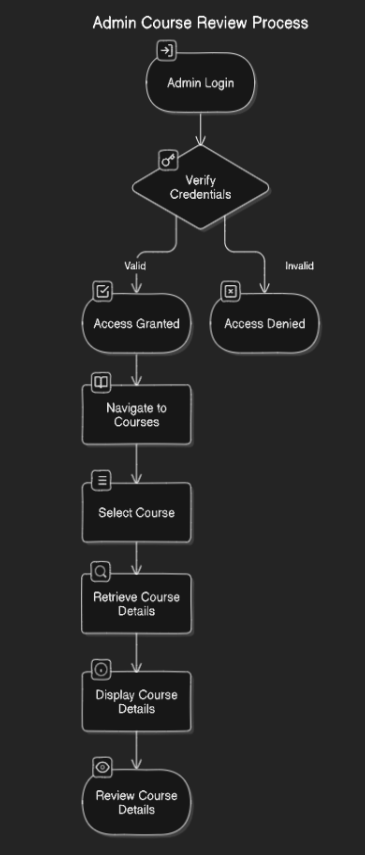
· **Step 2:** System verifies credentials and grants access.

· **Step 3:** Admin navigates to the Courses section.

· **Step 4:** Admin selects a course from the list to view its details.

· **Step 5:** System retrieves and displays detailed information about the selected course.

· **Step 6:** Admin reviews the course details.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 108 | The system must display all relevant details of the selected course, including title, description, instructor, category, enrollment status, and any other pertinent information. |
| (4) | BR 109 | The system should handle any errors in retrieving or displaying course details, providing appropriate feedback to the admin. |

### **UC 38: Approve/Reject course**

|  |  |
| --- | --- |
| **Objective:** | Allow admin to accept or reject instructor course |
| **Actor:** | Admin |
| **Trigger:** | Admin Login Successfully |
| **Pre-conditions:** | Instructor must create course |
| **Post-conditions:** |  |
| **Basic flows:** | 1. Admin will view the course 2. Click accept or reject the course |

**Activity flow**

· Admin logs in.

· System verifies admin credentials.

· Admin navigates to the course approval section.

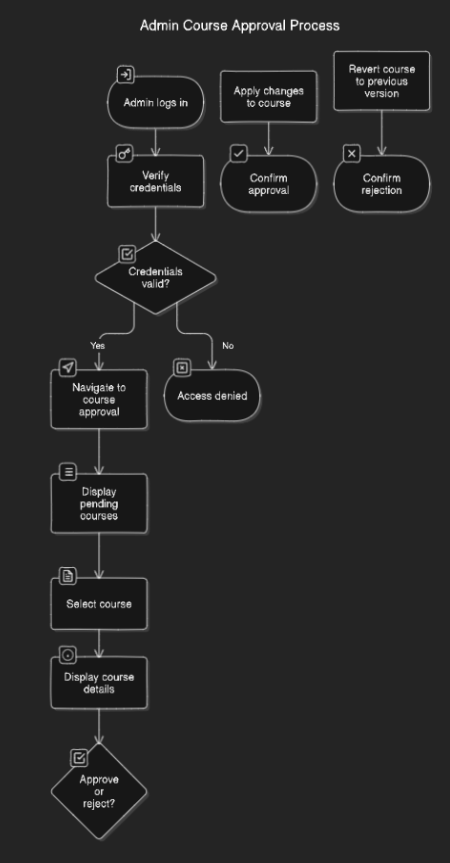
· System displays a list of courses pending approval.

· Admin selects a course to view details.

· System displays course details.

· Admin decides to approve or reject the course.

* **If approve**:
  + Admin clicks on the "Approve" button.
  + System applies the changes to the course. (BR 109)
  + System confirms the course has been approved.
* **If reject**:
  + Admin clicks on the "Reject" button.
  + System reverts the course to the previous version. (BR 108)
  + System confirms the course has been rejected.



**Business Rules**

|  |  |  |
| --- | --- | --- |
| **Step** | **BR Code** | **description** |
| (2) | BR 108 | The course is rejected will back to the previous version. |
| (4) | BR 109 | The course is approved will applied the changes. |

# **Non-functional requirements**

## **Performance requirements**

* Screen loading time should not exceed 5 seconds.

## **Security Requirements**

* The system must encrypt the user's personal information: password
* The system must comply with applicable security standards and policies
* The system should be designed to ensure that data stored in the system is protected against data loss, unauthorized access, use, modification, disclosure or misuse. Only authorized people have access to the data

## **Software quality attributes**

* The system can handle the amount of data as the number of users increases by optimizing storage and access methods.
* The application is guaranteed to run on all operating systems
  + Android: from 8.0 and up
  + iOs: from 13.1 and above
* The admin portal is guaranteed to run on:
  + Windows 7 and above
  + Chrome v88.0

# **Other requirements**

## **Localize rules**

* Number format:
  + Decimal digits: 2
  + Date format: DD/MM/YYYY
* Application language: Vietnamese

## **Common field**

### **Freetext**

All free text (single line of text input) displayed in the application, unless otherwise stated, has max length of 255 characters. This includes all text fields & standard search boxes. Once 255 characters have been inputted into the field, additional input is no longer received (not receiving input despite button pressed).

Can not use Vietnamese Text.

This input only accepts ASCII characters, and does not accept special character types (eg Unicode).

### **Dropdown list**

* Auto focus and display the selected value when clicked to show value list.
* Default text when there is no selected value: “Please select item”.
* Display 5 records, if there are more than 5, show vertical scrollbar.
* Sorted alphabetically.
* When the text to be displayed (in each entry, and also in the field after value has been selected) is longer than the text display box, cut the text at the end of first line, and replace the missing characters with “…”. Also applicable with text suggestions during Brand input.
* Show tooltip to when hovering the text to display full text.
* Does not allow duplicate entries.
* Add 1 blank selection on top, when selected, set the place holder as “Please select item”.

### **Scroll list for selecting Category**

* Auto focus and display the selected value.
* When parent category is selected, the value list of child category will be displayed.
* Sort by alphabet order.
* Display 6 records, if there are more than 6, show vertical scrollbar.
* When click again to a selected value, reload the selection of category tree. Select on the selected value and all its child categories will be non-selected.

### **Image attachment**

* Allow selecting and uploading 1 image at a time.
* When selecting another image to upload, this will overwrite the old image.
* After image is uploaded, display the image as thumbnail above the button.
* Allow only image files:
  + JPEG (.jpg, .jpeg)
  + PNG (.png)

User can delete uploaded image by clicking on option to delete it.

* Maximum allowed size: 1MB, show message “Maximum image attachment size is 1MB.” under the field when user uploaded an image which its size over 1MB.
* If user selects not supported file type or format, show message MSG 28

### **Rich text**

* Allow copy and paste with html format.
* Allow to format text with header.
* Allow text format as:
  + Bold
  + Italic
  + Underline
  + Strikethrough
* Allow text alignment.
  + Right
  + Left
  + Middle
* Allow change text color.
* Allow bulleted lists, numbered lists.
* Allow attach link (URL hyper link).
  + Anchor Link
  + URL
  + Mail Link
* Allow “Block Quote”
* Allow insert images
* Allow insert table.
* Show tooltip when hovering to each action
* Allow user edit and format using keyboard.

### **Text area**

### **Date picker**

* Validate immediately after losing focus.
* Format MM/DD/YYYY
  + Show message “The format is invalid.” under the field if user inputted date value with wrong format.
* Place holder: MM/DD/YYYY
* Default value when clicking on :<<Today>>.

### **Label display**

* When the text to be displayed is longer than the text display box (Width), move the redundancy to the line below. When moving the text to new line, keep the whole word.
* When the text to be displayed is longer than the text display box (Height), cut the text at the end of last line, and replace the missing characters with “…”.
* When the text to be displayed is longer than the text display box (Width) in case of display only one line, replace the missing characters with “…”.
* Show tooltip to when hovering the text to display full text

## **Message Configuration**

1. In-field error messages: Displayed in red italic text.
2. All pop-up messages: Always displayed with an “X” button on the top-right corner of the pop-up. Clicking outside of the pop-up does not close it, only the “X” button or corresponding buttons on the pop-up does.
3. After closing the popup, return to the current webpage.

## **Internal System Error**

All internal system errors will share the same error message, as was defined in MSG 3

## **Pagination**

Unless stated otherwise, search-related behavior in all views of the system will follow the following specifications:

* When user performs a new search (both completely new and searching within the current search results), refresh table, go back to page 1.
* When user changes the pagination setting (number of items to show per page), refresh table, go back to page 1.
* When user in page 1, disable buttons of “<<” and “Previous”
* When user in last page, disable buttons of “>>” and “Next”
* Otherwise, enable all buttons.

If there is 1 page, still display the pagination with 1 page.

# Appendices

## Message list

|  |  |  |
| --- | --- | --- |
| **#** | **Mã số thông báo** | **Nội dung** |
|  | MSG 1 | Vui lòng điền đầy đủ thông tin |
|  | MSG 2 | Người dùng đã tồn tại trong hệ thống |
|  | MSG 3 | Mã xác thực đã hết hiệu lực |
|  | MSG 4 | Mã xác thực không chính xác. Vui lòng thử lại |
|  | MSG 5 | Xác thực không thành công. Vui lòng thử lại |
|  | MSG 6 | Xác thực thành công |
|  | MSG 7 | Đăng ký thành công. Vui lòng kiểm tra email để hoàn thành xác nhận tài khoản. Nếu bạn không nhận được email. Ấn vào đây. |
|  | MSG 8 | Tài khoản không tồn tại |
|  | MSG 9 | Mật khẩu không chính xác. Bạn đã nhập sai mật khẩu <số lân> lần. Tài khoản của bạn sẽ bị vô hiệu hóa nếu nhập sai mật khẩu 3 lần |
|  | MSG 10 | Mật khẩu không chính xác. Tài khoản của bạn đã bị vô hiệu hóa. Vui lòng xác thực tài khoản. |
|  | MSG 11 | Mật khẩu không trùng khớp |
|  | MSG 12 | Mật khẩu không chính xác |
|  | MSG 13 | Bạn có chắc chắn xóa < thông báo> không? |
|  | MSG 14 | Bạn có muốn lưu thay đổi không? |
|  | MSG 15 | Bạn nhận được lời mời tham gia vào nhóm < tên nhóm> |
|  | MSG 16 | [tên người dùng] đã yêu cầu tham gia nhóm |
|  | MSG 17 | [tên người dùng] đã yêu cầu nhận sản phẩm [tên sản phẩm] |
|  | MSG 18 | [Tên người dùng] đã chấp nhận yêu cầu nhận sản phẩm của bạn |
|  | MSG 19 | [Tên người dùng] đã từ chối yêu cầu nhận sản phẩm của bạn |
|  | MSG 20 | Nhóm quyền này đã tồn tại |
|  | MSG 21 | Đã xảy ra lỗi. Vui lòng thử lại sau |
| 22. | MSG 22 | Mật khẩu mới không được trùng với mật khẩu hiện tại. |

## Email template

**ET 1: Welcome user template**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Welcome Email |
| Body | Thank you for signing up!  {FirstName}, {LastName}  Dear [UserFullName],<br><br>Welcome to Cursus! We are excited to have you join our learning community.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 2: Instructor course approved template**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Your course has been approved! |
| Body | The New Courses is available now  {FirstName}, {LastName}  New course has been approved by Admin, please check in the main page.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 3: Inactive instructor course template:**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Inactive Course |
| Body | Hello friends, {FullName}  <p>Your {courseTitle} course led by {instructorName} is inactive.</p>  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 4: Create instructor course template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | New course has been created! |
| Body | The New Courses is available, {FirstName}, {LastName}  New course has been added by Instructor, please check in the main page.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 5: Remind delete account template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Remind Delete Account! |
| Body | Hello! {FirstName}, {LastName}  Dear [UserFullName],<br><br>Your account will be deleted after 14 days.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 6: Verify email for instructor approval template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Circus Verify Email For Instructor Approval |
| Body | Circus Verify Email For Instructor Approval  {FirstName}, {LastName}  <h2>-Your Account has been aprroval!</h2>  <p>Thank you for your waiting! Click this to go to the main page</p>  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 7: Congratulations on completing the course template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Circus Verify Email For Instructor Approval |
| Body | Hello!  {FirstName}, {LastName}  Dear [UserFullName],<br><br>You have completed our course program, you can take new courses to increase your knowledge and skills.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 8: Reset your password template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Reset Your Password |
| Body | Reset your password to regain access.  {FirstName}, {ResetLink}  <p>Hello {FirstName},</p><p>Click <a href="{ResetLink}">here</a> to reset your password.</p>  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 9: Verify email template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Cursus Verify Email |
| Body | User Account Verified!  {FirstName}, {LinkLogin}  <p>Thank you for registering your Cursus account. Click here to go back the page</p>  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 10: Delete account template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Delete Account! |
| Body | Hello!  {FirstName}, {LinkLogin}  Dear [UserFullName],<br><br>Your account has been deleted.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 11: Notice for instructor template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Your course has been rejected! |
| Body | The New Courses is not available now  {FirstName}, {LinkLogin}  New course has been rejected by Admin, please check in the main page.  <p>Contact us at cursusservicetts@gmail.com</p> |

**ET 12: Reset your password template.**

|  |  |
| --- | --- |
| Gửi đến | <email người dùng> |
| Subject | Reset Your Password |
| Body | Reset your password to regain access  [UserFullName], [ResetPasswordLink]  Hi [UserFullName],<br><br>We received a request to reset your password. Click the link below to reset your password.  <p>Contact us at cursusservicetts@gmail.com</p> |