

**MINISTRY OF**

**EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Corporate Training System**

|  |  |
| --- | --- |
| **GROUP 02** | |
| **Group members** | Lê Phúc Thịnh – SE61759 (Leader)  Võ Thạch Nguyên – SE61778  Phạm Chánh Hưng – SE61807  Đặng Thế Anh – SE61372 |
| **Supervisor** | Nguyễn Huy Hùng |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | CTS |

- Ho Chi Minh City, ***8th January, 2018*** -



**CAPSTONE PROJECT REGISTER**

Class: Duration time: From 08/01/2018 To 25/04/2018

(\*) Profession: <Software Engineer> Specialty: <ES> <IS>

x

(\*) Kinds of person make registers: Lecturer Students

x

1. Register information for supervisor (if have)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Full name** | **Phone** | **E-Mail** | **Title** |
| Supervisor 1 | Nguyễn Huy Hùng |  | hungnh@fpt.edu.vn | Mr. |

2. Register information for students (if have)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Full name** | **Code** | **Phone** | **E-mail** | **Role** |
| Student 1 | Lê Phúc Thịnh | SE61759 | 0961173093 | [thinhlpse61759@fpt.edu.vn](mailto:thinhlpse61759@fpt.edu.vn) | Leader |
| Student 2 | Võ Thạch Nguyên | SE61778 | 01233461809 | [nguyenvtse61778@fpt.edu.vn](mailto:nguyenvtse61778@fpt.edu.vn) | Member |
| Student 3 | Phạm Chánh Hưng | SE61807 | 0936210777 | [hungpcse61807@fpt.edu.vn](mailto:hungpcse61807@fpt.edu.vn) | Member |
| Student 4 | Đặng Thế Anh | SE61372 | 0981434867 | [anhdtse61372@fpt.edu.vn](mailto:anhdtse61372@fpt.edu.vn) | Member |

3. Register content of Capstone Project

(\*) 3.1. Capstone Project name:

**English**: Corporate Training System

**Vietnamese:** Hệ Thống Đào Tạo Cho Doanh Nghiệp

**Abbreviation:** CTS

(\*) 3.2. Main proposal content (including result and product)

1. Theory and practice (document):
   * Student should apply the software development process and the UML 2.0 in modeling the system
   * Software artifacts include User Requirement, Software Requirement Specification, Architecture Design, Detail Design, System Implementation and Testing Document, Installation Guide, sources code, and deployable software packages
   * Server-side technique:
     + Database design, OOA, OOD, OOP, MVC, Java or .Net technology, …
     + Apache Lucene, ElasticSearch
   * Client-side technique
     + HTML5, CSS, JavaScript, JQuery, Ajax
     + Mobile Platform (iOS, Android)
2. Program:

Building a web platform that allows corporate to deploy their own training programs. The following main features should be implemented:

* + Manage corporate brand: each corporate should be about to customize their own brand name, logo, and related information
  + Manage users: there are 4 types of roles: Admin, Manager, Author, Learner. Admin and Manager should be able to create accounts for Author and Learners, as well as manually enroll Learners to a specific course. Admin can change user’s roles
  + Manage programs: Admin/Manager can create a training program. Each program comprises of sequence of courses. A program can be self-paced or deadline
  + Manage training plans: Admin/Manager can create training plans: starting and ending programs/courses
  + Manage course: Author can create courses.
    - Courses can include: video, text, PowerPoint, quiz.
    - Passing a quiz might be required before proceeding to next lecture/course
    - Each course belongs to skills
  + Manage reports: Admin/Manager can view Learners’ report on learning activities

1. Other products:

* All of management functions of the system must be implemented to support the operating system.

4. Other comment (propose all relative thing if have)

|  |  |
| --- | --- |
| **Supervisor (If have)**  *(Sign and full name)* | HCM city, date 12/12/2017  **On behalf of Registers**  *(Sign and full name)* |

Nguyễn Huy Hùng

# Table of Contents

[List of Tables 6](#_Toc512037311)

[List of Figures 7](#_Toc512037312)

[Definitions, Acronyms and Abbreviations 8](#_Toc512037313)

[A. Introduction 9](#_Toc512037314)

[1. Project Overview 9](#_Toc512037315)

[2. Introduction 9](#_Toc512037316)

[3. Current Situation 9](#_Toc512037317)

[4. Problem Definition 10](#_Toc512037318)

[5. Proposed Solution 10](#_Toc512037319)

[5.1 Feature functions 10](#_Toc512037320)

[5.2 Advantages and Disadvantages 10](#_Toc512037321)

[B. Software Project Management Plan 11](#_Toc512037322)

[1. Problem Definition 11](#_Toc512037323)

[1.1 Name of this Capstone Project 11](#_Toc512037324)

[1.2 Problem Abstract 11](#_Toc512037325)

[1.3 Project Overview 11](#_Toc512037326)

[**1.3.1 Current Situation** 11](#_Toc512037327)

[**1.3.2 The Proposed System** 12](#_Toc512037328)

[**1.3.3 Development Environment** 12](#_Toc512037329)

[1.3.3.1 Hardware requirement 12](#_Toc512037330)

[1.3.3.2 Software requirement 13](#_Toc512037331)

[2. Project organization 13](#_Toc512037332)

[2.1 Software Process Model 13](#_Toc512037333)

[2.2 Roles and Responsibilities 14](#_Toc512037334)

[2.3 Tools and Techniques 15](#_Toc512037335)

[3. Project Management Plan 16](#_Toc512037336)

[3.1 Product Backlog 16](#_Toc512037337)

[3.2 Sprint Backlog 16](#_Toc512037338)

[3.3 Meeting Minutes 16](#_Toc512037339)

[C. Software Requirement Specifications 16](#_Toc512037340)

[1. User Requirement Specification 16](#_Toc512037341)

[1.1 Guest Requirement 16](#_Toc512037342)

[1.2 Learner Requirement 16](#_Toc512037343)

[1.3 Author Requirement 16](#_Toc512037344)

[1.4 System Admin Requirement 17](#_Toc512037345)

[1.5 Manager Requirement 17](#_Toc512037346)

[2. System Requirement Specification 17](#_Toc512037347)

[2.1 External Interface Requirement 17](#_Toc512037348)

[**2.1.1 User Interface** 17](#_Toc512037349)

[**2.1.2 Hardware Interface** 17](#_Toc512037350)

[**2.1.3 Software Interface** 17](#_Toc512037351)

[**2.1.4 Communication Protocol** 17](#_Toc512037352)

[2.2 System Overview Use Case 17](#_Toc512037353)

[3. Conceptual Diagram 19](#_Toc512037354)

[D. Software Design Description 20](#_Toc512037355)

[1. Design Overview 20](#_Toc512037356)

[2. System Architectural Design 21](#_Toc512037357)

[2.1 System Architecture Overview 21](#_Toc512037358)

[3. Component Diagram 23](#_Toc512037359)

[4. Detail Description 23](#_Toc512037360)

[4.1 Class Diagram 23](#_Toc512037361)

[5. Database Design 26](#_Toc512037362)

[Entity Relationship Diagram (ERD) 26](#_Toc512037363)

[Data Dictionary 27](#_Toc512037364)

[6. Algorithms 27](#_Toc512037365)

[6.1 Create course URL 27](#_Toc512037366)

[6.1.1 Definition 27](#_Toc512037367)

[6.1.2 Define problem 28](#_Toc512037368)

[6.1.3 Solution 28](#_Toc512037369)

[6.1.4 Complexity 28](#_Toc512037370)

[6.1.5 Flow chart 29](#_Toc512037371)

[Tasks sheet 30](#_Toc512037372)

[APPENDIX 34](#_Toc512037373)

# List of Tables

[Table 1: Definitions, Acronyms and Abbreviations 9](#_Toc511591717)

[Table 2: Hardware Requirement for Server 14](#_Toc511591718)

[Table 3: Hardware Requirement for PC. 14](#_Toc511591719)

[Table 4: Software requirement. 15](#_Toc511591720)

[Table 5: Roles and Responsibilities 17](#_Toc511591721)

[Table 6: Tools and Techniques 18](#_Toc511591722)

[Table 7: Conceptual diagram data dictionary 22](#_Toc511591723)

[Table 8: Component Dictionary 25](#_Toc511591724)

[Table 9: Class Diagram Dictionary 29](#_Toc511591725)

[Table 10: ERD dictionary 32](#_Toc511591726)

# List of Figures

[Figure 1: Scrum Process 16](#_Toc511591698)

[Figure 2: System Overview Use Case 20](#_Toc511591699)

[Figure 3: Conceptual diagram 21](#_Toc511591700)

[Figure 4: System architecture overview 23](#_Toc511591701)

[Figure 5: Component diagram 25](#_Toc511591702)

[Figure 6: Class diagram 27](#_Toc511591703)

[Figure 7: Entity Relationship Diagram 30](#_Toc511591704)

[Figure 8: Create course URL flow chart 34](#_Toc511591705)

# Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| CTS | Corporate Training System |

Table 1: Definitions, Acronyms and Abbreviations

# Introduction

## Project Overview

* Project name: **Corporate Training System**
* Project Code: **CTS**
* Product Type: **Website application**
* Start Date: **8th January, 2018**
* End Date: **April, 2018**

## Introduction

Nowadays, online learning turns out to be more and more practiced. Many corporations start to post their training programs and courses online for their employees. It represents an easy and comfortable method to achieve knowledge in almost every field, from law and accounting, to even human sciences, such as psychology and physics. We can see that e-learning is a great alternative for corporations, especially for companies which can’t afford the time or human resources to train their employees. Currently, many corporations are having problems to find a suitable system that can provide them an environment to train staffs, manage training plans and view employees report easier and more effective.

In this document, we will introduce a solution for this problem. We build a system, which allows any corporates to post their training program with ease and privacy on our website. Through our system, staffs can access to training anywhere and anytime, making learning and training more effective, faster and easier. For corporations’ admin and manager, they can manage their programs effortlessly, and view their employees’ statistics.

Besides, this document also shows our working process and our perspective in the system, designs, architecture and workflow.

## Current Situation

Corporations with an online training system find out their employees are training faster and better than those who do not have access to online learning. However, it seems not easy for each company to build their own training system.

Whenever a corporate want to develop a system to train their employees, they usually hire a software company to build a system that suitable for their needs with a cheap price. Nevertheless, it is hard to seek a software company to acquire the two requirements above.

In conclusion, corporates train their employee can be summarized through the following ways:

1. Corporations train their employees in a traditional way: Trainers/mentors train employees directly through meetings and classes.
2. Corporations train their employees online by:
   1. Build their own system to train their employees.
   2. Share their training programs online for their employees.
   3. Use the current available training system.

## Problem Definition

From above current situation, there are some disadvantages:

* Traditional training is not efficiency for corporations.

For current available training system:

* There are not many systems for corporates to train its employees online.
* Learner cannot communicate with author to understand the content of that course if they get trouble in learning.
* Learner has to wait for their manager to accept the request before they can begin to learn a course.
* When an author creates a course, they do not need to receive permission from manager or administrator. Therefore, that course may have a bad influence on learners or inappropriate content.
* An author maybe wonders what plan or program that their courses belongs to.

## Proposed Solution

Our proposed solution is to build a system named Corporate Training System (CTS) who has troubles with finding a place to train their employees online.

Corporate Training System is a web application with following functions:

### 5.1 Feature functions

* + Manage plan, programs, skills and courses.
  + Manage users.
  + Manage corporate brand and information.
  + Participate to a plan or program.
  + Enroll in a course.
  + Learn a lecture.
  + Take quiz and view quiz result.
  + Make statistic report.
  + View learners’ report on learning activities
  + Manage personal information
  + Manage discussion.

### 5.2 Advantages and Disadvantages

* Advantages:
  + - Necessary for all corporate
    - New idea
    - Easy to use application
    - Friendly interface
* Disadvantages:
  + - Finding a suitable price for customer.
    - The cost in web application maintenance is high
    - Users’ demand change

# Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

- Official name: Corporate Training System

- Vietnamese name: Hệ thống đào tạo cho doanh nghiệp

- Abbreviation: CTS

### Problem Abstract

This project is our concern about development of training online for corporation. We call it Corporate Training System (Traisy). As current in Vietnam, as well as other developed countries, websites and applications for e-study are limited in functions and often focus on only certain companies. For example, many platforms for learning online only contains video lecture. Meanwhile, some systems support many functions but only available to a specific company, such as SAKAI for FPT SOFTWARE.

Those websites that focus on several tasks for e-learning are often designed for international users and thus, are not particularly suitable for Vietnamese people.

The solution is to provide a system for all corporates, especially Vietnamese Corporates, a way to satisfy all their needs of a simple and effective e-training for employees.

### Project Overview

**1.3.1 Current Situation**

Below are some problems that we encountered in this project:

* **Business logic:** Some scopes of project are not clear to every team member in early stage of developments. For example, learners cannot self-enroll their selves into a training program but being enrolled by their manager or admin.
* **Schedule of team member:** team members can have a conflict in meeting schedule because of sick, or class and work schedule, etc.
* **Framework study:** team members have a problem when applying the play framework into project. The team needs an amount of time to get familiar new techniques.
* **Some tools require premium package to access:** For convenience for play framework, we use Intellij IDE, which requires premium account and expensive for our team members.
* **Lack of UI (user interface), UX (user experience) design skill:** Our team members all study IS major and no one has studied UI, UX design. Therefore, the UI may misunderstand or hard to use with normal user.
* **Knowledge about training plan:** this project is about how a corporation or company trains their employees. The knowledge of this issue is new to members.

**1.3.2 The Proposed System**

The proposed system is Corporate Training System. The system contains a web application for learner, author, manager and admin.

In detail, the web application provides following features:

* For learner:
  + Learners can view study courses. In this function, they can view all resources and lectures of a course they are assigned and through this, they can study any lecturer.
  + Learners can view their learning activities. After learners have enrolled in a course, they can view process of their study.
  + Learners can update their info and manage their discussion.
* For author:
  + Authors can manage their courses. Author has full function to edit their lessons such as update, delete and add.
  + Author can view learner statistic. In this function, author can view all learners’ progress who takes their lectures. By doing this, author can easily improve their lesions.
* For manager:
  + Manager can manage learners. Manager can add a learner to a program/plan or remove learner from any program/plan of their company.
  + Manager can create, edit or delete a skill.
* For admin:
  + Manage corporate’s information. Admin can edit their corporate’s related information and change company’s logo or brand name.
  + Admin can edit all users’ role.

**1.3.3 Development Environment**

##### 1.3.3.1 Hardware requirement

|  |  |  |
| --- | --- | --- |
| **Server** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window Server 2008 | Window Server 2008 |
| **Computer Processor** | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz) |
| **Computer Memory** | 1GB RAM | 2GB or more |

Table 2: Hardware Requirement for Server

|  |  |  |
| --- | --- | --- |
| **PC** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window 7 | Window 7 or more. |
| **Computer Processor** | Intel® Core i3 1.4GHz | Intel® Core i5 2.50GHz |
| **Computer Memory** | 1GB RAM | 2GB RAM or more |
| **Web Browser** | Firefox (v52 or higher), Chromes (v28 or higher) | Chrome latest stable version |

Table 3: Hardware Requirement for PC.

##### 1.3.3.2 Software requirement

|  |  |  |
| --- | --- | --- |
| **Name** | **Name/Version** | **Description** |
| **Environment** | Java JDK 7 | Specification for developing web application |
| **Operating System** | Window 10 | Operating system and platform for development |
| **Integrated development environment** | IntelliJ IDEA 2017.2.3  Bracket (Front-end Application Development) | Programming tools |
| **Modeling tool** | StarUML 2.8,  Lucidchart.com | Software modeling tool |
| **Database Management System** | MySQL 5.7 | Used to create & manage the database for system |
| **Source control** | Git hosted by Github.com | Used to source control & version control. |
| **Web browser** | Chrome 28 or higher. | Testing browser |

Table 4: Software requirement.

## Project organization

### Software Process Model

This project is developed under Scrum model (Scrum Methodology). We apply Scrum model to capable with current situation in our team. We choose this model due to following reasons:

+ Easy in tracking process: In Scrum, a product backlog (a wish list of tasks that need to be prioritized in a project) is created. Thanks to product backlog, our team can create sprint backlog so that we can tracking member's tasks easier in each sprint.

+ Quality: We agreed with each other that sprint is one week. In specific week, we always have a sprint planning meeting that takes place at the beginning of each sprint, a daily meeting that is held for a brief period (about 15 minutes) each day to allow each member to brief the team on what they did the previous day and a sprint review meeting, this meeting is held at the end of each sprint. With all of this meeting, we can review and guarantee quality of all our works.

+ Fast and convenient: Scrum Methodology is very suitable with our project and team because scrum can work with any technology/ programming language.

+ Easy to manage: Due to short sprints and constant feedback, it becomes easier to cope with the changes. Besides, it is also a lightly controlled method which insists on frequent updating of work progress through regular meetings. Thus, there is clear visibility of the development project.



Figure 1: Scrum Process

For more information:

* <http://scrummethodology.com/>

### Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Nguyễn Huy Hùng | Project manager | * Specify user requirement * Control the development process * Give out technique and business analysis support |
| **2** | Lê Phúc Thịnh | Team Leader,  BA, DEV, Tester | * Managing process * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **3** | Võ Thạch Nguyên | Team member, BA, DEV, Tester | * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **4** | Phạm Chánh Hưng | Team member, BA, DEV, Tester | * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **5** | Đặng Thế Anh | Team member, BA, DEV, Tester | * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |

Table 5: Roles and Responsibilities

### Tools and Techniques

|  |  |  |
| --- | --- | --- |
|  | **Tools** | **Techniques** |
| **Front-end** | Bracket | HTML5  CSS3  Javascript  jQuery  Bootstrap |
| **Back-end** | IntelliJ IDEA | JavaEE  Play Framework  JPA  Wowza streaming engine |

Table 6: Tools and Techniques

## Project Management Plan

### Product Backlog

All product backlogs could be found here

### Sprint Backlog

All sprint backlogs could be found here

### Meeting Minutes

All meeting documents could be found here

# Software Requirement Specifications

## User Requirement Specification

**1.1 Guest Requirement**

Guest is a person who doesn’t access to the system. Guest can use some functions in the system, to use all functions, guest must login. These are some functions guest can use:

* Login

**1.2 Learner Requirement**

Learner is a person who access to the system as learner role. Learner can use following functions:

* Login
* View course
* View plan
* View program
* Learn lecture
* Take quiz
* View learning activity
* Manage discussion

**1.3 Author Requirement**

* View learner statistic
* Manage course
* Manage lecture
* Manage section
* Manage quiz

**1.4 System Admin Requirement**

* Manage brand

**1.5 Manager Requirement**

* View report
* Manage user
* Manage plan
* Manage program
* Manage skill
* Manage feedback

## **System Requirement Specification**

**2.1 External Interface Requirement**

**2.1.1 User Interface**

* + - * The user interface uses Vietnamese.
      * The user interface displays best on 1366x768 resolutions on desktop.

**2.1.2 Hardware Interface**

Desktop PC.

**2.1.3 Software Interface**

Web application: work with browsers Firefox (v52 or higher), Chromes (v28 or higher), Internet Explorer (v10 or above) or with any web browser that supports HTML5 & CSS3

**2.1.4 Communication Protocol**

Use HTTP protocol 1.1 for communication between the web browser and the web.

**2.2 System Overview Use Case**

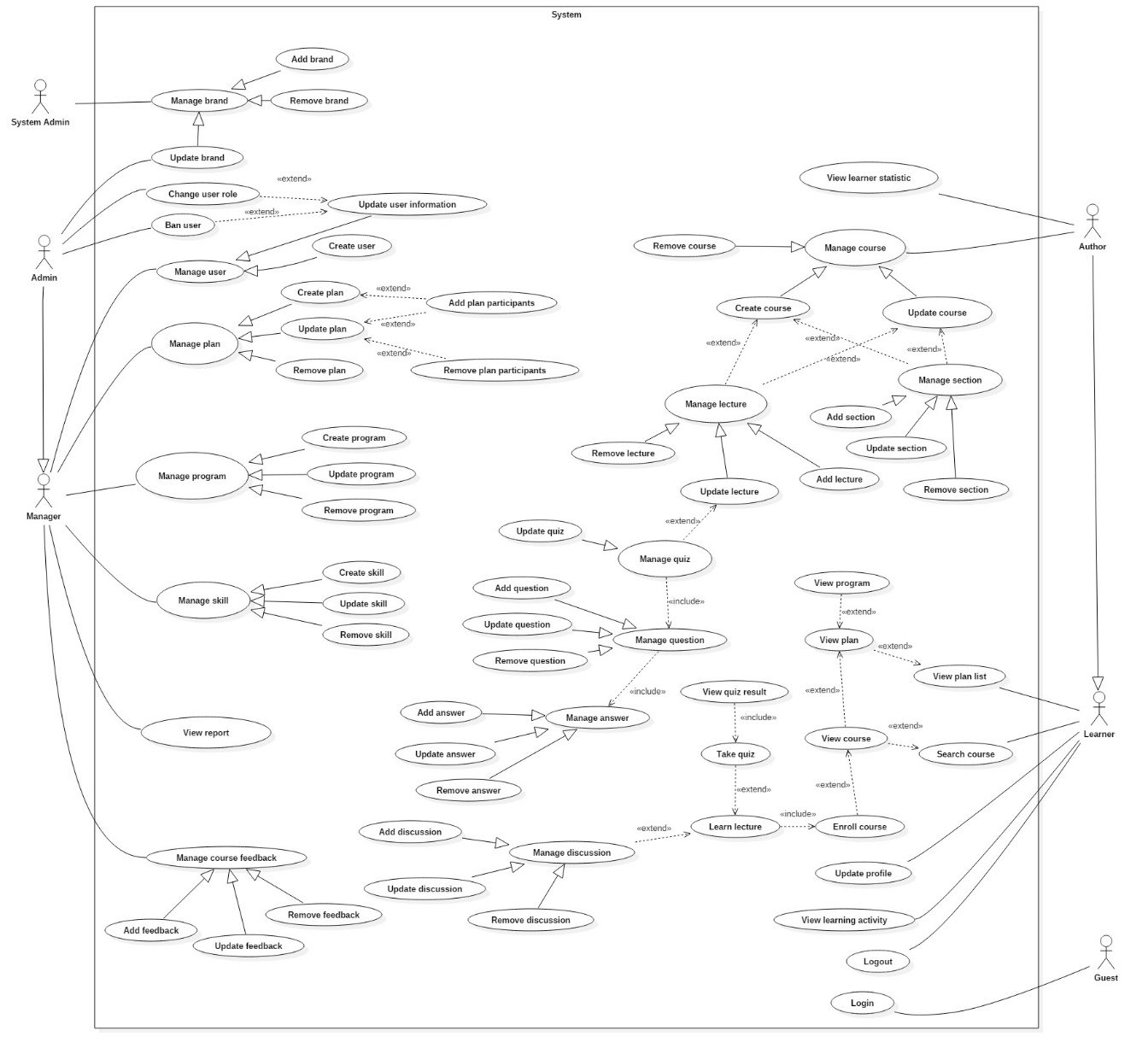


Figure 2: System Overview Use Case

## 3. Conceptual Diagram

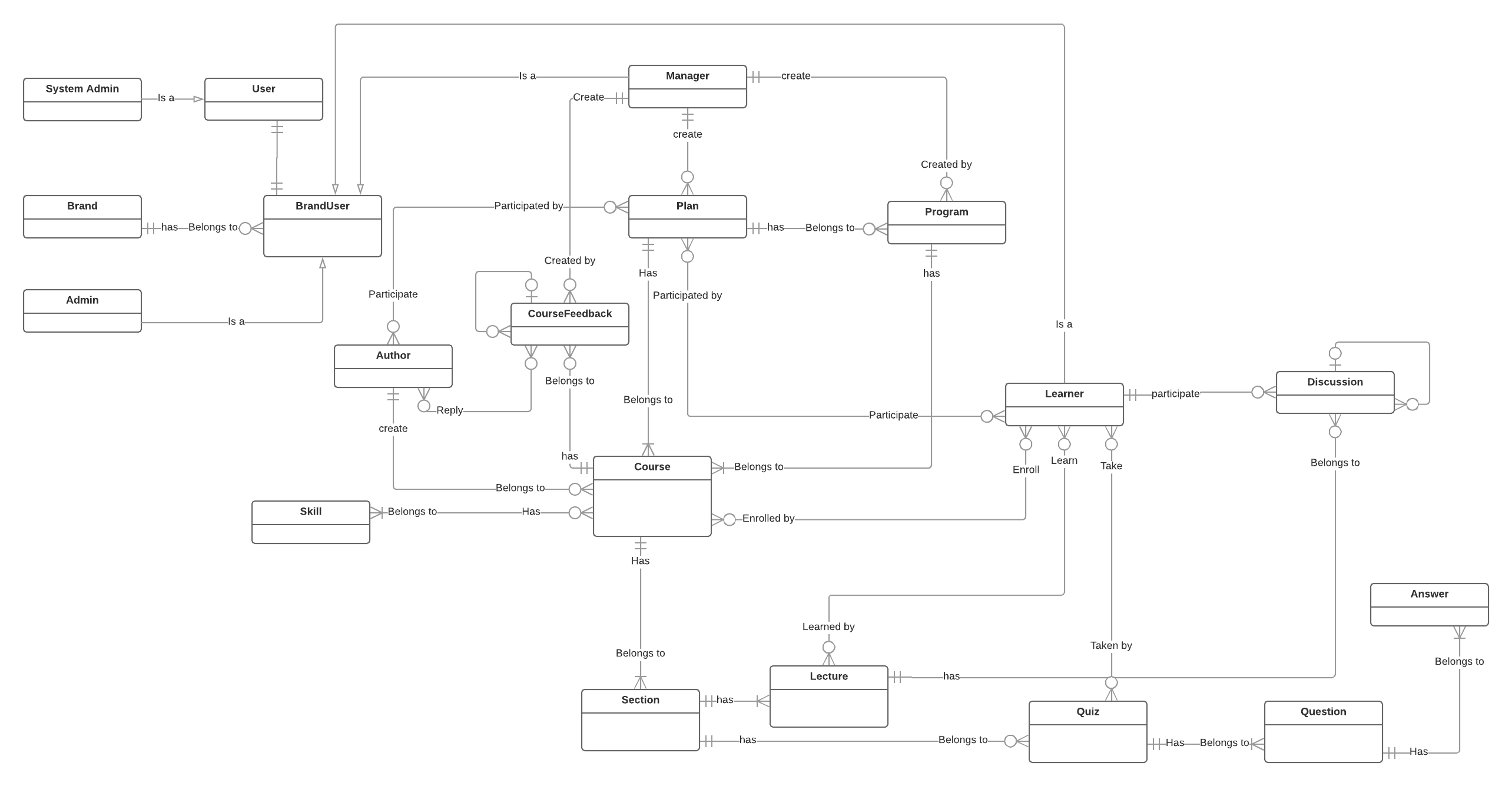


Figure 3: Conceptual diagram

|  |  |
| --- | --- |
| **ENTITY DATA DICTIONARY: DESCRIBE ALL CONTENT OF ALL ENTITIES** | |
| **Entity Name** | **Description** |
| **User** | Abstract entity describes a user in system |
| **System Admin** | Contains the system admin information |
| **Admin** | Contains the admin user information |
| **Manager** | Contains the manager information |
| **Author** | Contains the author information |
| **Learner** | Contains the learner information |
| **Brand** | Contains the brand information |
| **BrandUser** | Contains the user information in a brand |
| **Plan** | Contains the plan information |
| **Program** | Contains the program information |
| **Course** | Contains the course information |
| **CourseFeedback** | Contains the information of feedback |
| **Skill** | Contains the information of skill |
| **Section** | Contains the section information |
| **Lecture** | Contains the lecture information |
| **Quiz** | Contains the quiz information |
| **Question** | Contains the information of question |
| **Answer** | Contains the information of an answer |
| **Discussion** | Contains the information of a discussion |

Table 7: Conceptual diagram data dictionary

# Software Design Description

## 1. Design Overview

This document describes the technical and user interface design of TRAISY Web Application. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.

The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

The database design describes the relationships between entities and details of each entity.

Document overview:

Section 2: gives an overall description of the system architecture design.

Section 3: gives component diagrams that describe the connection and integration of the system.

Section 4: gives the detail design description, which includes class diagram, class explanation, and sequence diagram to details the application functions.

Section 5: describe screens design.

Section 6: describe a fully attributed ERD.

Section 7: describe algorithms.

## 2. System Architectural Design

**2.1 System Architecture Overview**

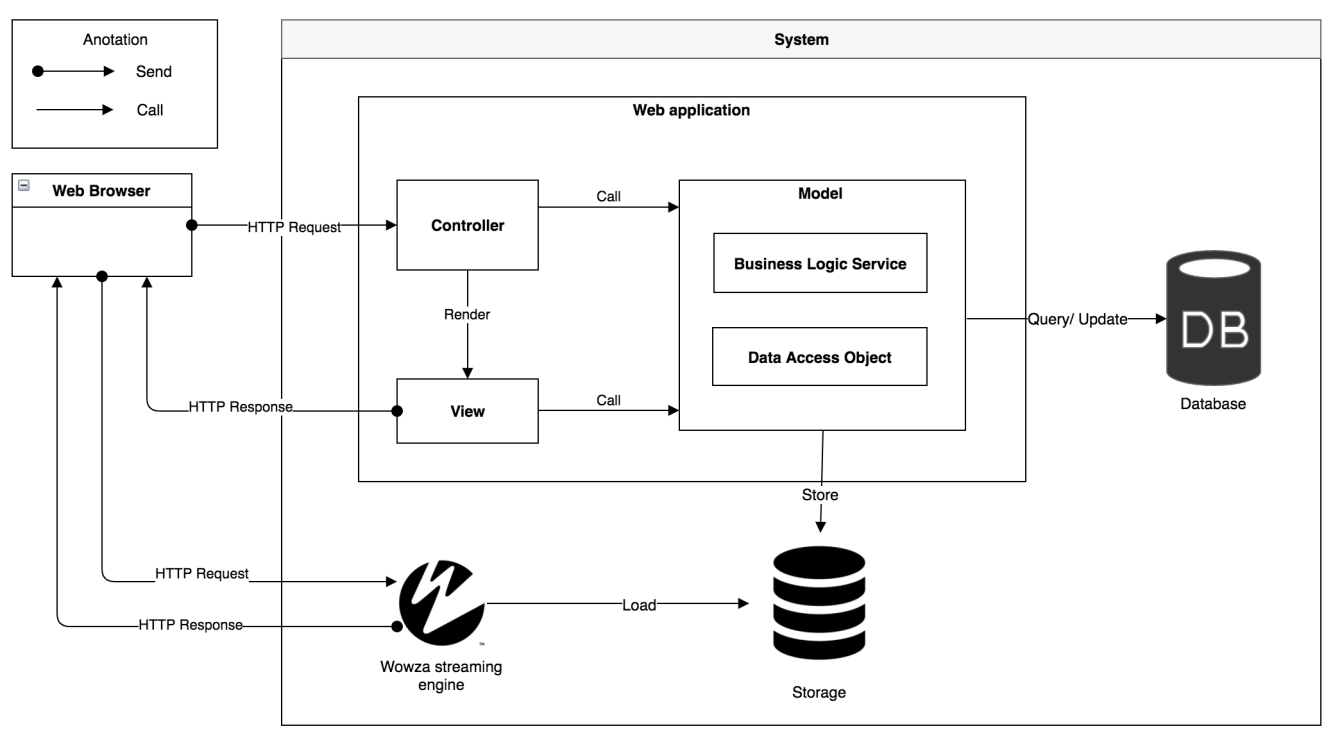


Figure 4: System architecture overview

Our application is developed mainly on Play Framework 2.5 which follows the MVC architectural pattern. We choose this framework for web application because of following advantages:

* **Hot reload:** Make a change, refresh the page and we can see the change immediately. As a result, we don’t waste time to redeploy the application.
* **Open source:** Play framework is open source; we can see how everything works in the open source code. There is also a large community using it, contributing plugins, asking questions on StackOverflow, and so on.
* **Reactive:** play is built on top of Akka and Netty, so it’s fully asynchronous: that is, you can use non-blocking I/O when making calls to remote services. This makes it possible to support many concurrent connections without dealing with thread pools.

This pattern splits the application into separate layers: The model layer, view layer, and controller layer.

* The model layer: Contains data structures and operations that needed for system to operate. Whenever model objects need to be saved into database, they contain JPA annotation for making these operations easier and more flexible.
* The view layer: a HTML file that presents the content and interaction with user. In play framework, views are generated using a templating system. The Controller gets data from the model layer, and then applies a template to render these objects.
* The controller responds to user actions and processes them. For more details, the controller listens for HTTP requests, then extracts relevant data from the request, such as parameter, request headers… and invokes the suitable methods that contains read, update or create the model object. After finishing the process, controllers render the view and then written as an HTTP Response to the user.

Besides the MVC architectural, we also use Wowza streaming engine with HLS streaming protocol for an easier and more efficient way to deliver video content or video lecture.

HTTP Live streaming (HSL) is a streaming protocol that generates multiple versions of the same content (but different resolution/bit rate) and chops these versions into chunks or segments (e.g., two seconds). The segments are provided on a web server and can be downloaded through HTTP standard compliant GET requests. Through multiple versions of same content, HSL also provides “adaptive streaming”, the client can switch to a higher resolution – if bandwidth permits, or to a lower resolution – if bandwidth decreases.

## Component Diagram

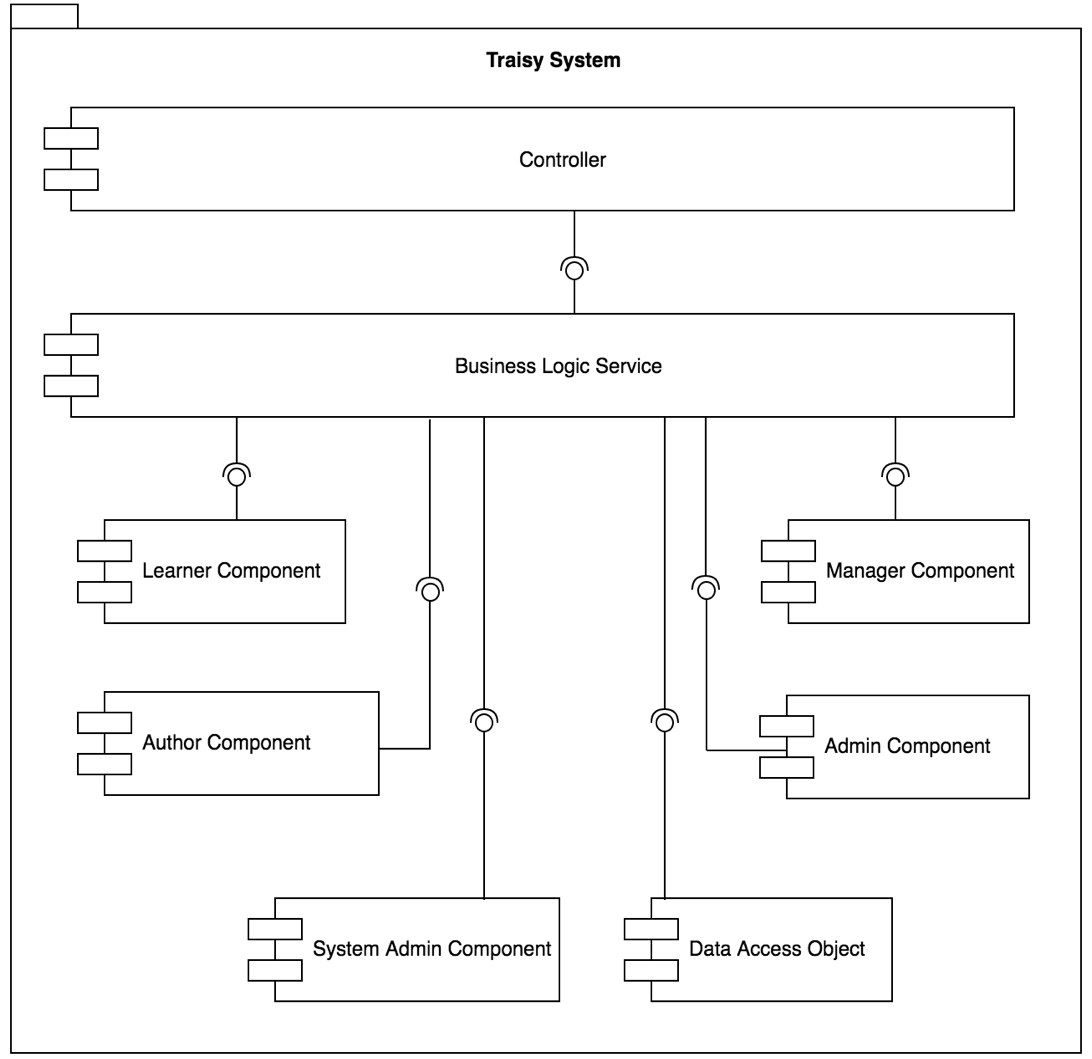


Figure 5: Component diagram

|  |  |
| --- | --- |
| COMPONENT DICTIONARY: DESCRIBES COMPONENTS | |
| Traisy System | Corporate Training System |
| Controller | Component to handle HTTP request |
| Business Logic Service | Component to handle system’s business operations |
| Learner Component | Component to handle learner activities in the system |
| Author Component | Component to handle author activities in the system |
| Manager Component | Component to handle manager activities in the system |
| Admin Component | Component to handle admin activities in the system |
| System Admin Component | Component to handle system admin activities in the system |
| Data access object | Component to handle interaction between the system and database |

Table 8: Component Dictionary

## Detail Description

**4.1 Class Diagram**

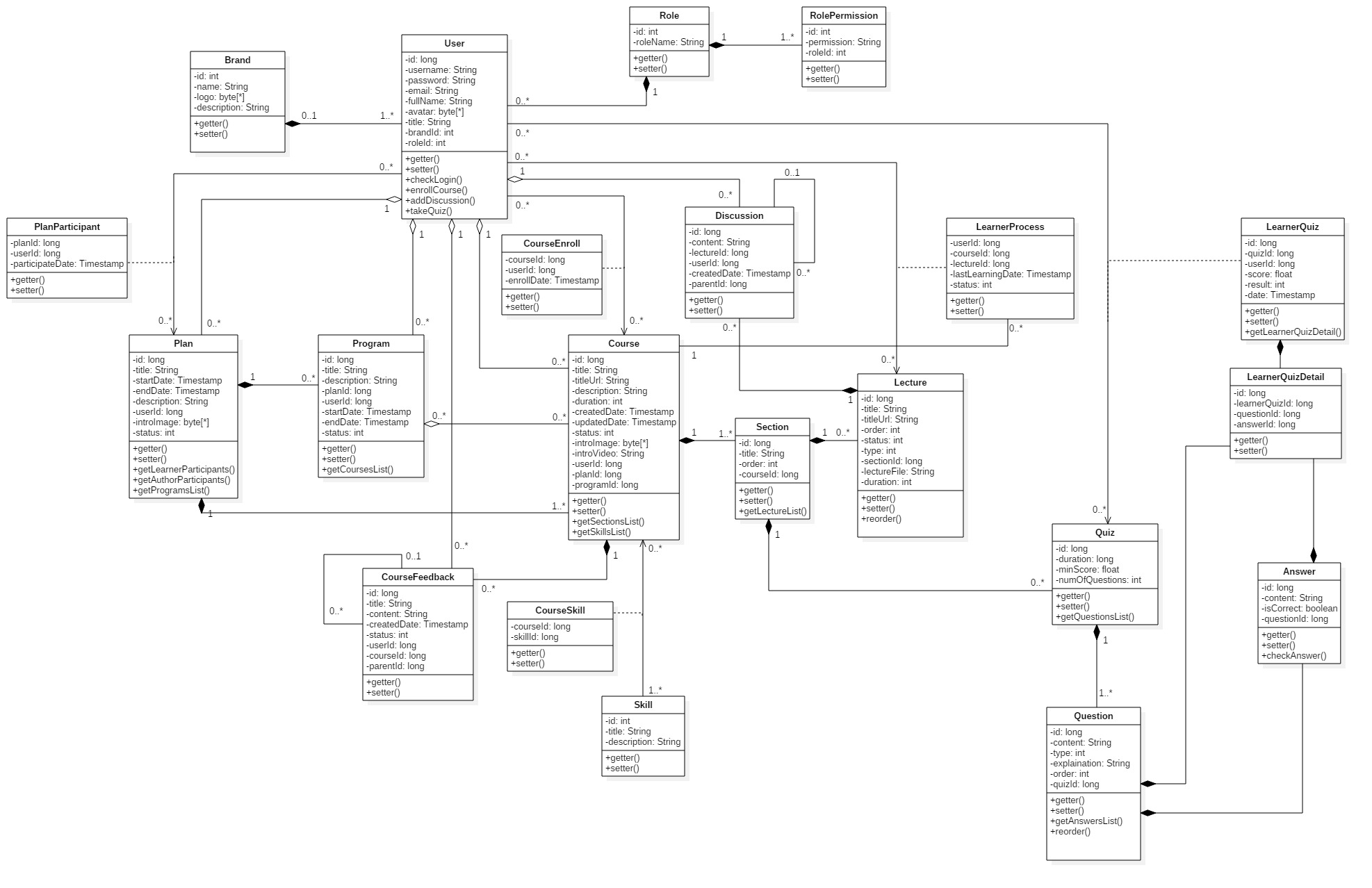


Figure 6: Class diagram

|  |  |  |
| --- | --- | --- |
| **CLASS DICTIONARY: DESCRIBE CLASS** | | |
| **Class Name** | **Mapping column with Conceptual diagram** | **Description** |
| **User** | user | Contains the user information |
| **Brand** | brand | Contains the brand information |
| **PlanParticipant** | N/A | Not exist in conceptual diagram but need this table to contain the information of a user participated in a plan |
| **Plan** | Plan | Contains the plan information |
| **Program** | Program | Contain the program information |
| **CourseEnroll** | N/A | Not exist in conceptual diagram but need this table to contain the information of a user enrolled in a plan |
| **Course** | Course | Contains the course information |
| **Discussion** | Discussion | Contains the discussion information |
| **Section** | Section | Contains the section information |
| **Skill** | Skill | Contains the skill information |
| **CourseSkill** | N/A | Not exist in the conceptual diagram but need this table to contain the information of a skill in a specific course |
| **Lecture** | Lecture | Contains the lecture information |
| **LearnerProcess** | N/A | Not exist in the conceptual diagram but need this table to contain the process information when learner study a lecture |
| **Quiz** | Quiz | Contains the quiz information |
| **Question** | Question | Contains the question information |
| **LearnerQuiz** | N/A | Contains the information of learner when that learner take quiz |
| **LearnerQuizDetail** | N/A | Contains the result of learner’s quiz |
| **Answer** | Answer | Contains the information of answer |
| **Role** | N/A | Not exist in conceptual diagram but need this table to contain the information of a role |
| **RolePermission** | N/A | Not exist in conceptual diagram but need this table to contain the permission of each role |

Table 9: Class Diagram Dictionary

## Database Design

### Entity Relationship Diagram (ERD)

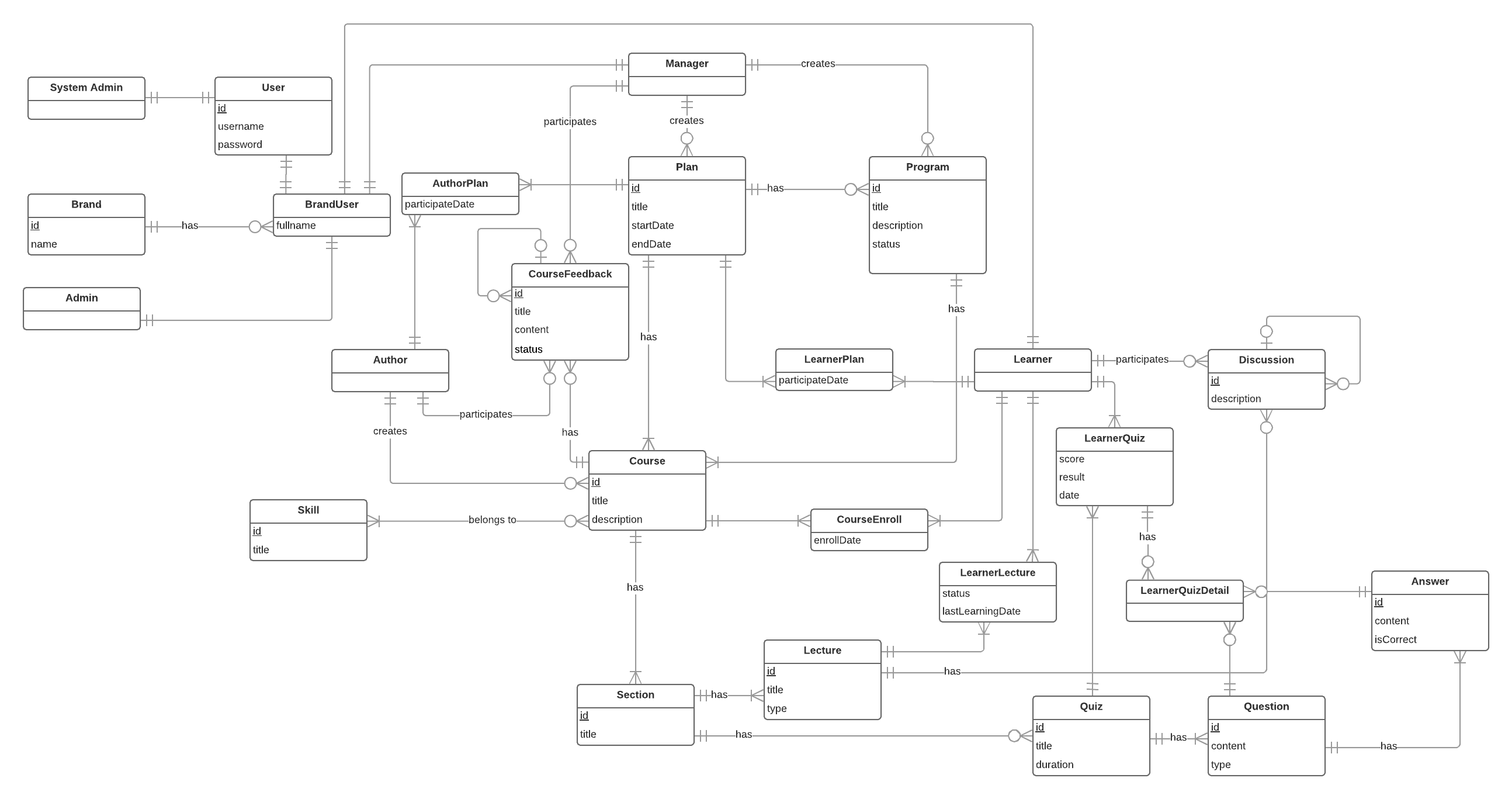
****

Figure 7: Entity Relationship Diagram

### Data Dictionary

|  |  |
| --- | --- |
| Entity Data Dictionary: describe content of all entities | |
| Entity name | **Description** |
| User | Contains the user’s information |
| System Admin | Contains the system admin’s information |
| Brand | Contains the brand’s information |
| Brand User | Shows relationship between Brand and User |
| Admin | Contains the admin’s information |
| Manager | Contains the manager’s information |
| Author | Contains the author’s information |
| AuthorPlan | Shows relationship between Author and Plan |
| Plan | Contains the plan’s information |
| Program | Contains the program’s information |
| CourseFeedback | Contains the feedback’s information |
| Course | Contains the course’s information |
| Skill | Contains the skill’s information |
| Section | Contains the section’s information |
| LearnerPlan | Shows relationship between Learner and Plan |
| CourseEnroll | Contains the information about what course that learner was enrolled |
| Lecture | Contains the lecture’s information |
| LearnerLecture | Contains the information about what lecture that learner was learned |
| Learner | Contains the learner’s information |
| LearnerQuiz | Contains the information about what quiz that learner was taken |
| Quiz | Contains quiz’s information |
| Discussion | Contains discussion’s information |
| Question | Contains the question’s information |
| Answer | Contains the answer’s information |
| LearnerQuizDetail | Contains the result of quiz that learner take |

Table 10: ERD dictionary

## Algorithms

### 6.1 Create course URL

#### Definition

When a user visits a course, the web will navigate to that course’s page with unique URL for each course. Moreover, the course’s URL needs to be related to the course’s title.

#### Define problem

Each course has its own title and content. However, we need a unique URL that related to the title for web to navigate between courses. The problem is that when an author creates a course, the course title may be duplicated to other course; therefore, we need an algorithm to differentiate between courses that have the same title but different content.

#### Solution

To solve this problem, we should follow these steps:

Each course will be distinguished by its URL that saved in database, which is created through:

* + 1. Lowercase all the course’s title
    2. Transfer the course title into non-Unicode characters by removing the Unicode accents and diacritics
    3. Replace special characters such as white space with hyphen (-)
    4. If the URL is already existed in database, the URL will be appended with a number behind.
    5. The URL structure will be as bellow:
       - Khoá học HTML: khoa-hoc-html (if course title is not duplicated)
       - Khoá học HTML: khoa-hoc-html-1 (if course title is duplicated)

#### Complexity

In total, the complexity of this algorithm is **N**

#### Flow chart

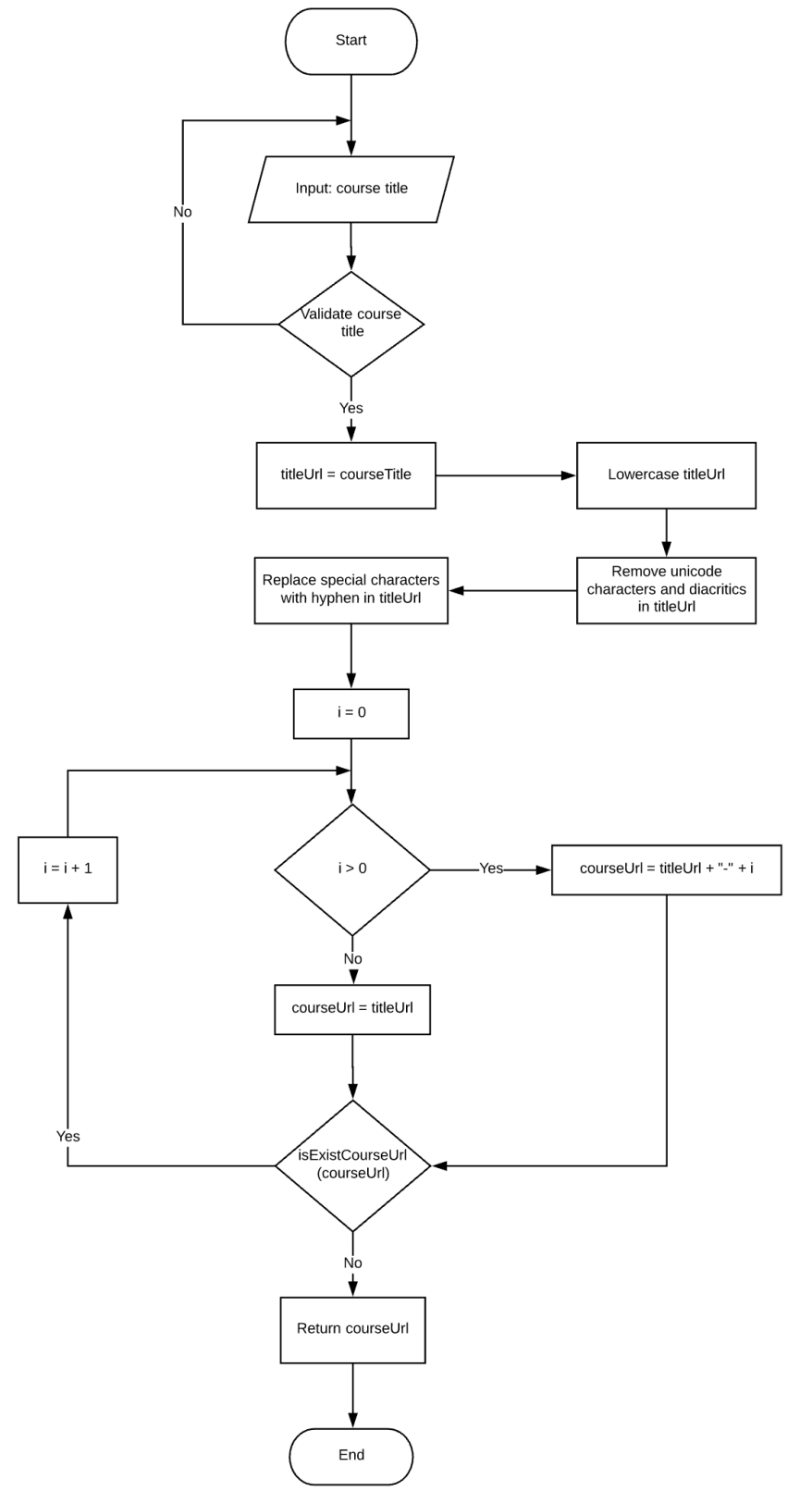
****

Figure 8: Create course URL flow chart

# Tasks sheet

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Product Deliverables** | **Task** | **ThinhLP** | **NguyenVT** | **HungPC** | **AnhDT** | **Unit** | **Size** |
| 1 | Report1 - Introduction | Project Information |  | **O** |  |  |  | 1 |
| Introduction |  | **O** |  |  |  | 1 |
| Current Situation |  | **O** |  |  |  | 1 |
| Problem Definition |  |  | **O** |  |  | 1 |
| Proposed Solution |  | **O** |  |  |  | 1 |
| Functional Requirement |  |  | **O** |  |  | 1 |
| Role and Responsibility |  |  | **O** |  |  | 1 |
| 2 | Report2- Software Project Management Plan | Problem Definition |  |  | **O** |  |  | 1 |
| Project organization |  |  | **O** |  |  | 1 |
| Project management plan |  | **O** |  |  |  | 1 |
| Coding Convention |  | **O** |  |  |  | 1 |
| 3 | Report 3- Software Requirement Specification | User Requirement Specification |  | **O** |  |  |  | 1 |
| **System Requirement Specification** |  |  |  |  |  | 1 |
| External Interface Requirements |  |  | **O** |  |  | 1 |
| System overview usecase | **O** |  |  |  |  | 4 |
| **Functional Requirement** |  |  |  |  |  |  |
| Guest - Login |  |  |  | **O** |  | 1 |
| **Learner** |  |  |  |  |  |  |
| Log out |  |  |  | **O** |  | 1 |
| View plan list | **O** |  |  |  |  | 1 |
| View plan | **O** |  |  |  |  | 1 |
| Update profile | **O** |  |  |  |  | 1 |
| View learning activity | **O** |  |  |  |  | 1 |
| Search course | **O** |  |  |  |  | 1 |
| View course | **O** |  |  |  |  | 1 |
| View program | **O** |  |  |  |  | 1 |
| Enroll course | **O** |  |  |  |  | 2 |
| Learn lecture | **O** |  |  |  |  | 2 |
| Take quiz | **O** |  |  |  |  | 2 |
| View quiz result | **O** |  |  |  |  | 2 |
| Add discussion | **O** |  |  |  |  | 1 |
| Update discussion | **O** |  |  |  |  | 1 |
| Remove discussion | **O** |  |  |  |  | 1 |
| **Author** |  |  |  |  |  |  |
| View learner statistic |  |  | **O** |  |  | 3 |
| Create course |  |  | **O** |  |  | 1 |
| Remove course |  |  | **O** |  |  | 1 |
| Update course |  |  | **O** |  |  | 1 |
| Add section |  |  | **O** |  |  | 1 |
| Update section |  |  | **O** |  |  | 1 |
| Remove section |  |  | **O** |  |  | 1 |
| Add lecture |  |  | **O** |  |  | 1 |
| Update lecture |  |  | **O** |  |  | 1 |
| Remove lecture |  |  | **O** |  |  | 1 |
| Update quiz |  |  | **O** |  |  | 1 |
| Add question |  |  | **O** |  |  | 1 |
| Update question |  |  | **O** |  |  | 1 |
| Remove question |  |  | **O** |  |  | 1 |
| Add answer |  |  | **O** |  |  | 1 |
| Update answer |  |  | **O** |  |  | 1 |
| Remove answer |  |  | **O** |  |  | 1 |
| **Manager** |  |  |  |  |  |  |
| View report |  | **O** |  |  |  | 1 |
| Update user's information |  | **O** |  |  |  | 1 |
| Create user |  | **O** |  |  |  | 1 |
| Create plan |  | **O** |  |  |  | 1 |
| Update plan |  | **O** |  |  |  | 1 |
| Add plan participants |  | **O** |  |  |  | 2 |
| Remove plan participants |  | **O** |  |  |  | 2 |
| Remove plan |  | **O** |  |  |  | 1 |
| Create program |  | **O** |  |  |  | 1 |
| Update program |  | **O** |  |  |  | 1 |
| Remove program |  | **O** |  |  |  | 1 |
| Create skill |  | **O** |  |  |  | 1 |
| Update skill |  | **O** |  |  |  | 1 |
| Remove skill |  | **O** |  |  |  | 1 |
| Add feedback |  | **O** |  |  |  | 1 |
| Update feedback |  | **O** |  |  |  | 1 |
| Remove feedback |  | **O** |  |  |  | 1 |
| **Admin** |  |  |  |  |  |  |
| Update brand |  |  |  | **O** |  | 1 |
| Change user role |  |  |  | **O** |  | 1 |
| Ban user |  |  |  | **O** |  | 1 |
| **System Admin** |  |  |  |  |  |  |
| Add brand |  |  |  | **O** |  | 1 |
| Update brand |  |  |  | **O** |  | 1 |
| Remove brand |  |  |  | **O** |  | 1 |
| **Non-functional Requirement** |  |  |  |  |  |  |
| Usability |  |  |  | **O** |  | 1 |
| Reliability |  |  |  | **O** |  | 1 |
| Availability |  |  |  | **O** |  | 1 |
| Security |  |  |  | **O** |  | 1 |
| Maintainability |  |  |  | **O** |  | 1 |
| Portability |  |  |  | **O** |  | 1 |
| Performance |  |  |  | **O** |  | 1 |
| Conceptual diagram |  |  | **O** |  |  | 1 |
| 4 | Report 4- Software Design Description | Design overview |  |  |  | **O** |  | 1 |
| System Architectural Design |  | **O** |  |  |  | 4 |
| Component Diagram | **O** |  |  |  |  | 3 |
| Detailed Description of Components | **O** |  |  |  |  | 1 |
| Class diagram | **O** |  |  |  |  | 5 |
| **Class specification (states, behaviors)** |  |  |  |  |  |  |
| User | **O** |  |  |  |  | 1 |
| Brand | **O** |  |  |  |  | 1 |
| Plan | **O** |  |  |  |  | 1 |
| PlanParticipant | **O** |  |  |  |  | 1 |
| Program | **O** |  |  |  |  | 1 |
| Course |  | **O** |  |  |  | 1 |
| CourseEnroll |  | **O** |  |  |  | 1 |
| Skill |  | **O** |  |  |  | 1 |
| CourseSkill |  | **O** |  |  |  | 1 |
| CourseSection |  | **O** |  |  |  | 1 |
| CourseDiscussion |  |  | **O** |  |  | 1 |
| CourseLecture |  |  | **O** |  |  | 1 |
| LearnerProcess |  |  | **O** |  |  | 1 |
| Quiz |  |  | **O** |  |  | 1 |
| Question |  |  | **O** |  |  | 1 |
| Answer |  |  |  | **O** |  | 1 |
| LearnerQuiz |  |  |  | **O** |  | 1 |
| LearnerQuizDetail |  |  |  | **O** |  | 1 |
| Role |  |  |  | **O** |  | 1 |
| RolePermission |  |  |  | **O** |  | 1 |
| **Sequence Diagram** |  |  |  |  |  |  |
| **System Admin** |  |  |  |  |  |  |
| Create new brand |  |  |  | **O** |  | 2 |
| Remove brand |  |  |  | **O** |  | 2 |
| **Admin** |  |  |  |  |  |  |
| Update brand |  |  |  | **O** |  | 2 |
| **Manager** |  |  |  |  |  |  |
| Update user information |  |  | **O** |  |  | 2 |
| Create new user |  |  | **O** |  |  | 2 |
| Create plan |  |  | **O** |  |  | 2 |
| Update plan |  |  | **O** |  |  | 2 |
| Remove plan |  | **O** |  |  |  | 2 |
| Add author to plan |  | **O** |  |  |  | 2 |
| Create program |  | **O** |  |  |  | 2 |
| Update program |  | **O** |  |  |  | 2 |
| Remove program |  | **O** |  |  |  | 2 |
| View report | **O** |  |  |  |  | 2 |
| **Author** |  |  |  |  |  |  |
| Create lecture | **O** |  |  |  |  | 2 |
| Update lecture | **O** |  |  |  |  | 2 |
| Remove lecture | **O** |  |  |  |  | 2 |
| **Learner** |  |  |  |  |  |  |
| Login |  |  |  | **O** |  | 2 |
| Learn lecture | **O** |  |  |  |  | 2 |
| **User Interface Design** |  |  |  |  |  |  |
| **Learner Interface Design** |  |  |  |  |  |  |
| Login |  |  |  | **O** |  | 1 |
| View plan list |  |  |  | **O** |  | 1 |
| View plan overview |  |  |  | **O** |  | 1 |
| Update profile |  |  |  | **O** |  | 1 |
| Learn quiz lecture |  |  |  | **O** |  | 1 |
| **Manager Interface Design** |  |  |  |  |  |  |
| Update brand |  | **O** |  |  |  | 1 |
| View user list |  | **O** |  |  |  | 1 |
| View list plan |  | **O** |  |  |  | 1 |
| Update/Remove plan |  | **O** |  |  |  | 1 |
| Add author to plan |  | **O** |  |  |  | 1 |
| View list program |  |  | **O** |  |  | 1 |
| Create program |  |  | **O** |  |  | 1 |
| Manage course feedback |  |  | **O** |  |  | 1 |
| View list skills |  |  | **O** |  |  | 1 |
| Create skill |  |  | **O** |  |  | 1 |
| View plan report |  |  | **O** |  |  | 1 |
| **Author Interface Design** |  |  |  |  |  |  |
| Manage course |  |  |  | **O** |  | 1 |
| Manage section |  |  |  | **O** |  | 1 |
| Manage lecture |  |  |  | **O** |  | 1 |
| Create skill |  |  |  | **O** |  | 1 |
| View plan report |  |  |  | **O** |  | 1 |
| **Database Design** |  |  |  |  |  |  |
| Entity Relationship Diagram | **O** |  |  |  |  | 4 |
| Entity Dictionary |  | **O** |  |  |  | 1 |
| **Algorithms** |  |  |  |  |  |  |
| Create course URL | **O** |  |  |  |  | 5 |
| 5 | Report 5 - Software Implementation and Test Document | **Introduction** |  |  |  | **O** |  |  |
| **Database Relationship Diagram** |  |  |  |  |  |  |
| Physical diagram | **O** |  |  |  |  | 4 |
| Data dictionary |  | **O** |  |  |  | 1 |
| Test plan |  |  | **O** |  |  | 1 |
| **Write Test case** |  |  |  |  |  |  |
| System admin - Add brand |  | **O** |  |  |  | 1 |
| Manager - Create user |  | **O** |  |  |  | 1 |
| Author - Create course |  |  | **O** |  |  | 1 |
| Author - Create lecture |  |  | **O** |  |  | 1 |
| Author - Create quiz |  |  | **O** |  |  | 1 |
| Learner - Search course |  | **O** |  |  |  | 1 |
| Learner - Update profile |  | **O** |  |  |  | 1 |
| **Perform testing** |  |  |  |  |  |  |
| System admin - Add brand |  |  |  | **O** |  | 1 |
| Manager - Create user |  |  |  | **O** |  | 1 |
| Author - Create course |  |  |  | **O** |  | 1 |
| Author - Create lecture |  |  |  | **O** |  | 1 |
| Author - Create quiz |  |  |  | **O** |  | 1 |
| Learner - Search course |  |  |  | **O** |  | 1 |
| Learner - Update profile |  |  |  | **O** |  | 1 |
| 6 | Report 6 - Software User's Manual | Installation Guide |  |  |  | **O** |  | 2 |
| User’s Guide |  |  |  |  |  |  |
| Admin - Update brand |  |  |  | **O** |  | 1 |
| Manager - Create plan |  |  | **O** |  |  | 1 |
| Manager - Update plan |  |  | **O** |  |  | 1 |
| Manager - Create program |  |  | **O** |  |  | 1 |
| Author - Create course |  | **O** |  |  |  | 1 |
| Author - Update course |  | **O** |  |  |  | 1 |
| Author - Create quiz | **O** |  |  |  |  | 1 |
| Learner - Update profile | **O** |  |  |  |  | 1 |
| Learner - Learning quiz |  |  |  | **O** |  | 1 |

|  |  |  |
| --- | --- | --- |
| **Notes** |  |  |
| No. | Function Types | Function Point Size |
| 1 | Insert, Delete, Update, Simple Search, Login, Register, simple function or simple query on one table | 1 |
| 2 | Computed functions or joined query | 2 |
| 3 | Complex functions or multiple subquery | 3 |
| 4 | Constraint process and data integrity | 4 |
| 5 | Functions with implementing algorithms | 5 |

# APPENDIX

**UML Documentation of IBM**

<http://www.ibm.com/developerworks/rational/library/769.html?ca=drs->

**Play Framework 2.5**

<https://playframework.com/documentation/2.5.x/Home>

**MySQL**

<https://dev.mysql.com/doc/>

**Wowza Streaming Engine**

<https://www.wowza.com/products/streaming-engine/documentation>