

# BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**------------------------**

****

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC NGÀNH CÔNG NGHỆ THÔNG TIN

XÂY DỰNG HỆ THỐNG HỖ TRỢ TỰ HỌC VÀ ĐÁNH GIÁ TRỰC TUYẾN

GVHD : TS. Đặng Trọng Hợp

Sinh viên : Đinh Văn Duy Mã số sinh viên : 2021604611

Hà Nội - Năm 2025

i

# MỤC LỤC

DANH MỤC TỪ VIẾT TẮT.................................................................................

v

DANH MỤC HÌNH ẢNH......................................................................................

vi

DANH MỤC BẢNG BIỂU....................................................................................

viii

v

**DANH MỤC TỪ VIẾT TẮT**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Từ viết tắt** | **Tiếng Anh** | **Nghĩa tiếng việt** |
| 1 | API |  |  |

**DANH MỤC HÌNH ẢNH**

vi

Hình 1.1 :API

**DANH MỤC BẢNG BIỂU**

Bảng 1.1: 10

# MỞ ĐẦU

### Lý do chọn đề tài

Hiện tại là thời kỳ chuyển đổi số rất mạnh mẽ, giáo dục trực tuyến là một phần không thể thiếu trong việc nâng cao chất lượng đào tạo, chất lượng học tập dành cho các sinh viên. Các trường đại học cần triển khai mạnh mẽ các hệ thống hỗ trợ học trực tuyến cho sinh viên. Tuy nhiên hầu hết các hệ thống chỉ hỗ trợ các bài tập dạng trắc nghiệm hoặc tự luận, chưa có tính năng thực hành lập trình trực tuyến trên nền tảng. Điều này gây khó khăn cho các sinh viên trong việc học tập các môn có liên quan tới lập trình, do phải sử dụng phần mềm (IDE) bên ngoài để viết mã nguồn.

Nhắm vào những yếu tố đó, em muốn phát triển mạnh mẽ một tính năng có thể trực tiếp lập trình trên chính trang website của trường đại học Công Nghiệp Hà Nội của mình. Chính vì vậy đề tài “Hệ thống hỗ trợ tự học và đánh giá trực tuyến” được thực hiện nhằm xây dựng một nền tảng tự học trực tuyến có tích hợp tính năng lập trình trực tuyến trên website.

### Mục tiêu nghiên cứu

* + Xây dựng nền tảng học tập trực tuyến, cho phép sinh viên truy cập các học phần, bài học lý thuyết và thực hành lập trình.
  + Tích hợp trình soạn thảo code trực tuyến, hỗ trợ nhiều ngôn ngữ lập trình.
  + Cho phép sinh viên viết, chạy thử và nộp bài tập ngay trên nền tảng.
  + Tự động chấm điểm bài tập lập trình.
  + Hỗ trợ giảng viên trong quản lý và đánh giá sinh viên.
  + Ứng dụng công nghệ hiện đại để phát triển hệ thống: Docker, ASP.NET Core Web API, ReactJs,..

### Đối tượng nghiên cứu

* + Website học kết hợp của trường đại học Công Nghiệp Hà Nội.
  + Sinh viên, giảng viên.
  + Mô hình học tập kết hợp.

### Phạm vi nghiên cứu

* + Sinh viên: Sử dụng hệ thống để học lý thuyết, thực hành lập trình trực tuyến và nộp bài tập.
  + Giảng viên: Quản lý bài học, tạo bài tập lập trình, theo dõi và đánh giá kết quả của sinh viên.
  + Hệ thống chấm điểm tự động: Xử lý bài làm của sinh viên, kiểm tra tính đúng đắn của mã nguồn dựa trên test case, thời gian xử lý,..
  + Lập trình trực tuyến: Hỗ trợ sinh viên viết code, chạy thử và nộp bài ngay trên trình duyệt mà không cần cài đặt phần mềm bên ngoài.
  + Học trực tuyến: Sinh viên có thể truy cập học phần, bài giảng lý thuyết và tài liệu tham khảo.

## Nội dung nghiên cứu

* + Nghiên cứu mô hình học kết hợp của trường Đại học Công Nghiệp Hà Nội.
  + Phân tích và thiết kế xây dựng theo các học phần.
  + Xác định các yêu cầu chức năng và phi chức năng của website.
  + Đề xuất mô hình hệ thống phù hợp với nhu cầu của sinh viên và giảng viên.
  + Thiết kế cơ sở dữ liệu để lưu trữ thông tin học phần, bài giảng, bài tập và kết quả nộp bài.
  + Xây dựng API trên nền tảng ASP.NET Core Web API.
  + Phát triển Frontend bằng ReactJS, tối ưu hiệu suất.
  + Tích hợp trình soạn thảo code online hỗ trợ nhiều ngôn ngữ lập trình để sinh viên có thể chọn ngôn ngữ tùy thích.

### Bố cục bản báo cáo

Báo cáo gồm các phần sau đây:

# 

# CHƯƠNG 1. TỔNG QUAN HỆ THỐNG HỖ TRỢ TỰ HỌC VÀ ĐÁNH GIÁ TRỰC TUYẾN

## Giới thiệu về hệ thống hỗ trợ tự học và đánh giá trực tuyến.

Trong thời đại công nghệ số phát triển mạnh mẽ, việc ứng dụng công nghệ thông tin vào giáo dục ngày càng trở nên phổ biến và cần thiết. Đặc biệt là trong bối cảnh dịch bệnh Covid-19 vừa qua, hình thức học tập trực tuyến đã dần thay thế cho phương thức học truyền thống tại nhiều cơ sở giáo dục. Tuy nhiên, các hệ thống học trực tuyến hiện nay đa số chỉ hỗ trợ trình bày nội dung bài học mà thiếu đi sự tương tác thực hành, đặc biệt là đối với các môn học lập trình, đòi hỏi người học phải thường xuyên luyện tập và nhận được phản hồi tức thì.

Trước nhu cầu đó, đề tài ***"Xây dựng hệ thống hỗ trợ tự học và đánh giá trực tuyến"*** ra đời với mục tiêu xây dựng một nền tảng học trực tuyến tích hợp chức năng lập trình online và hệ thống chấm điểm tự động. Hệ thống không chỉ giúp sinh viên có thể tiếp cận nội dung bài học ở bất kỳ đâu, mà còn cho phép họ thực hành lập trình trực tiếp trên trình duyệt, nộp bài và nhận kết quả đánh giá ngay lập tức.

Với mô hình tổ chức theo từng học phần, mỗi học phần gồm nhiều bài học lý thuyết đi kèm bài tập thực hành, hệ thống này hỗ trợ giảng viên dễ dàng tạo nội dung giảng dạy và bài kiểm tra, đồng thời theo dõi tiến độ học tập của sinh viên. Qua đó, hệ thống góp phần nâng cao hiệu quả tự học và tạo môi trường học tập chủ động, linh hoạt cho người học.

## Mô tả bài toán

Bài toán đặt ra là xây dựng một hệ thống học tập và đánh giá trực tuyến, đặc biệt dành cho các môn học lập trình, cho phép người học không chỉ tiếp cận nội dung bài giảng mà còn có thể thực hành và được chấm điểm một cách tự động. Cụ thể, hệ thống cần đáp ứng các yêu cầu sau:

Tổ chức học phần và bài học: Các môn học (học phần) được chia thành các bài học, mỗi bài học bao gồm phần lý thuyết (tài liệu, video, văn bản) và bài tập lập trình đi kèm.

Soạn thảo và thực hành lập trình trực tuyến: Hệ thống tích hợp trình soạn thảo code (code editor) cho phép sinh viên viết, chạy thử và nộp bài trực tiếp trên nền tảng, không cần cài đặt phần mềm bên ngoài.

Chấm điểm tự động: Bài làm của sinh viên được kiểm tra bằng hệ thống test case, trả về kết quả đúng/sai, điểm số, thời gian thực thi và số lượng test case vượt qua.

**Quản lý người dùng:**

Sinh viên: Có thể đăng ký, đăng nhập, tham gia học phần, làm bài tập và theo dõi kết quả học tập.

Giảng viên: Có thể tạo học phần, bài học, bài tập, theo dõi tiến độ sinh viên và xem thống kê kết quả học tập.

Hệ thống quản trị: Quản trị viên có quyền quản lý học phần, người dùng, bài học, bài tập, kết quả và các báo cáo thống kê hệ thống.

**Yêu cầu về kỹ thuật:**

Backend sử dụng ASP.NET Core Web API để cung cấp dữ liệu và xử lý nghiệp vụ.

Frontend sử dụng ReactJS để xây dựng giao diện tương tác hiện đại, dễ sử dụng.

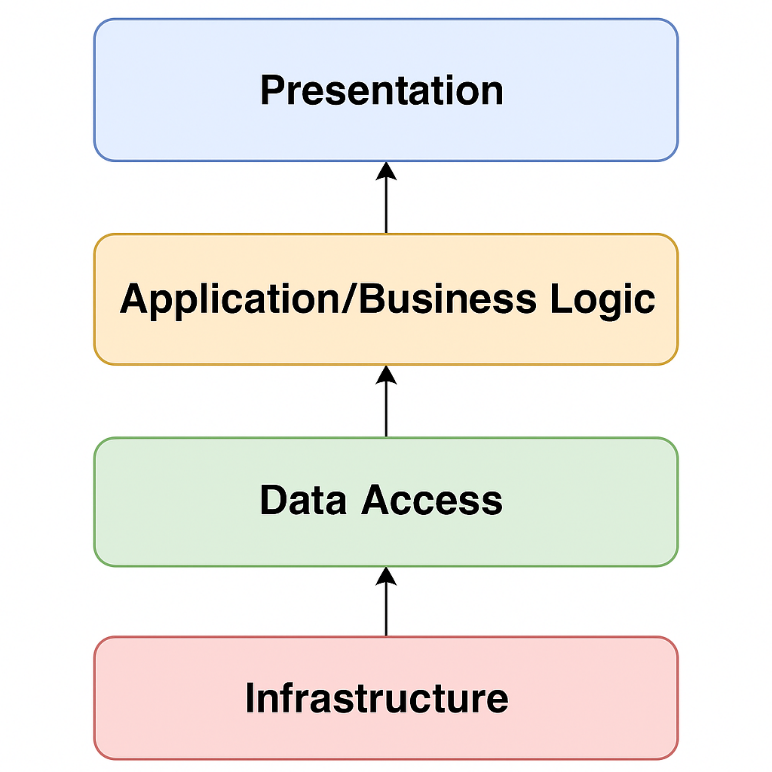
Cơ sở dữ liệu SQL Server để lưu trữ toàn bộ thông tin người dùng, học phần, bài tập, kết quả.

Tích hợp Docker + Judge để xử lý và đánh giá bài nộp một cách an toàn, cách ly môi trường thực thi.

Mục tiêu của hệ thống là xây dựng một nền tảng học tập toàn diện, trực quan, dễ sử dụng, đồng thời nâng cao tính chủ động của sinh viên trong việc học và thực hành lập trình, tạo ra môi trường học tập thực tế và hiệu quả hơn trong giáo dục đại học.

## Mô hình và công nghệ Web

## Kiến trúc phân lớp (Layered Architecture)

Kiến trúc phân lớp là một mô hình tổ chức phần mềm phổ biến, trong đó hệ thống được chia thành nhiều lớp riêng biệt, mỗi lớp đảm nhiệm một vai trò cụ thể trong quy trình xử lý dữ liệu và nghiệp vụ. Việc chia nhỏ hệ thống thành các lớp giúp tăng tính mô-đun, dễ bảo trì, mở rộng và kiểm thử phần mềm.

Trong đề tài này, hệ thống được xây dựng theo kiến trúc phân lớp gồm các lớp chính sau:

**Lớp Presentation (Giao diện người dùng)**

Là phần giao tiếp giữa người dùng và hệ thống.

Trong đề tài, lớp này được phát triển bằng ReactJS, hiển thị các giao diện học tập, bài học, bài tập và cho phép sinh viên tương tác như gửi code, xem kết quả, theo dõi tiến độ.

**Lớp Business Logic (Xử lý nghiệp vụ)**

Chịu trách nhiệm xử lý các logic nghiệp vụ như tính điểm, kiểm tra kết quả bài làm, xử lý các yêu cầu từ frontend.

Lớp này được tổ chức thành các service trong ASP.NET Core, đảm bảo rằng mọi thao tác từ giao diện đều đi qua lớp xử lý nghiệp vụ.

**Lớp Data Access (Truy cập dữ liệu)**

Chịu trách nhiệm giao tiếp với cơ sở dữ liệu, thực hiện các thao tác thêm, sửa, xóa, truy vấn dữ liệu.

Lớp này sử dụng mô hình Repository và UnitOfWork để tách biệt logic truy cập dữ liệu khỏi logic xử lý nghiệp vụ, đồng thời tăng khả năng tái sử dụng và kiểm thử.

**Lớp Infrastructure (Hạ tầng)**

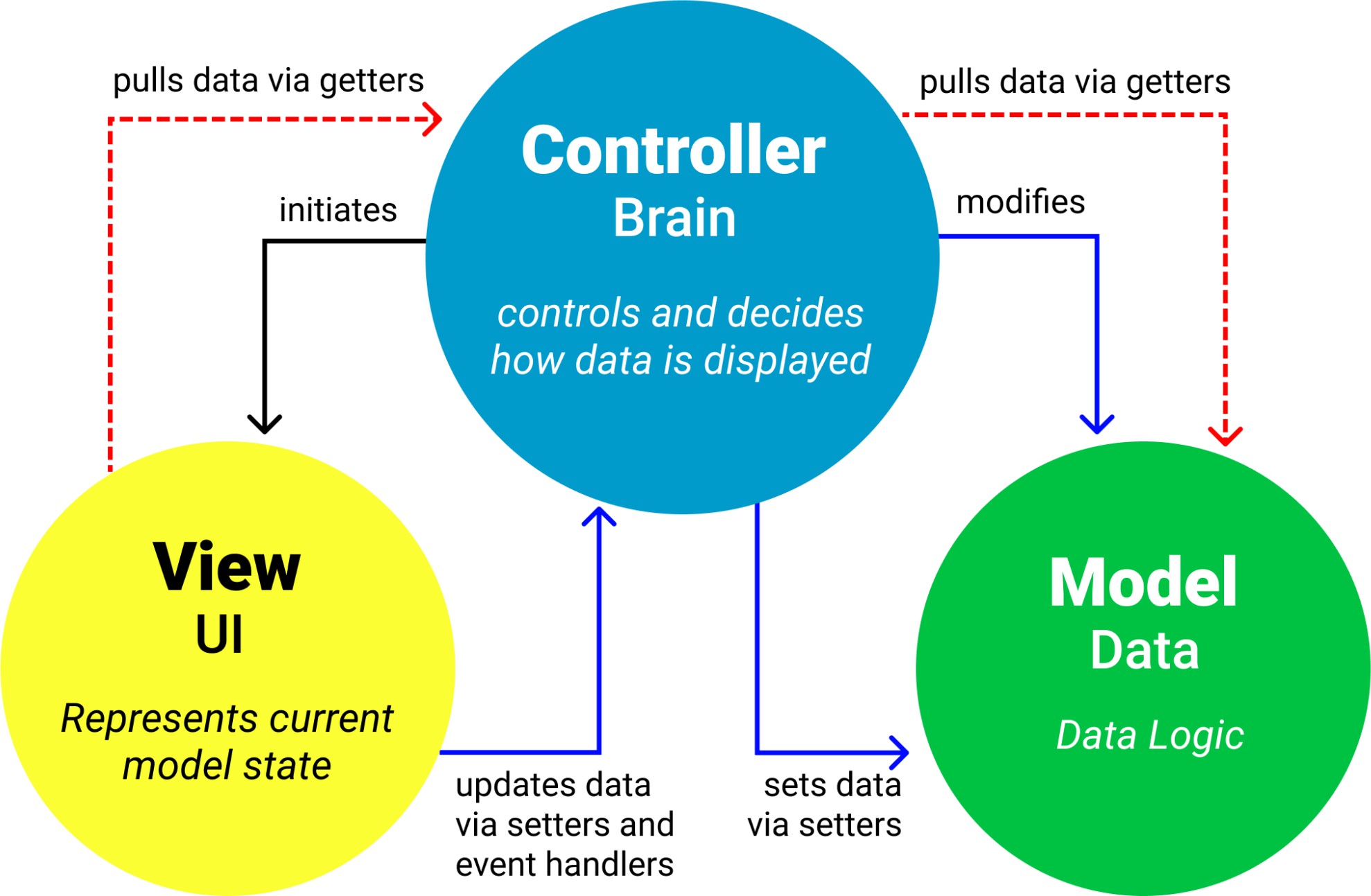
Bao gồm các thành phần hỗ trợ tích hợp với các dịch vụ bên ngoài như hệ thống chấm bài tự động (Judge API).

Giúp hệ thống mở rộng linh hoạt và dễ dàng tích hợp với các công nghệ khác trong tương lai.

Việc áp dụng kiến trúc phân lớp giúp hệ thống đảm bảo được tính rõ ràng về cấu trúc, dễ dàng bảo trì, mở rộng và tái sử dụng mã nguồn. Đồng thời giúp nhóm phát triển phân chia công việc hiệu quả hơn giữa các thành viên frontend và backend.

## Mô hình kiến trúc MVC

MVC [2] (Model-View-Controller) là một mô hình kiến trúc phần mềm được sử dụng để tổ chức và quản lý mã nguồn trong quá trình phát triển ứng dụng. Kiến trúc này tách biệt ứng dụng thành ba thành phần chính có thể tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dùng.



*Hình 1.1 : Kiến trúc phần mềm Model-View-Controller*

* Model: là nơi lưu trữ thông tin, thực hiện các thao tác cập nhật và truy vấn dữ liệu mà không quan tâm đến cách dữ liệu được hiển thị hoặc tương tác với người dùng.
* View: là thành phần hiển thị giao diện người dùng. View chịu trách nhiệm hiển thị dữ liệu một cách đẹp mắt và dễ thao tác đối với người dùng.
* Controller: là thành phần tiếp nhận và xử lý sự kiện từ người dùng. Controller nhận yêu cầu từ người dùng thông qua View, sau đó thao tác với Model và điều hướng hiển thị trở lại View.

## Ngôn ngữ lập trình C# và ASP.NET

C# là một ngôn ngữ lập trình đa năng, được phát triển bởi Microsoft, thuộc dòng ngôn ngữ lập trình .NET. C# được sử dụng phổ biến trong phát triển phần mềm, đặc biệt là trong việc xây dựng các ứng dụng web với ASP.NET. ASP.NET là một framework phát triển ứng dụng web mạnh mẽ, được Microsoft phát triển, cho phép tạo ra các ứng dụng web, API, và các dịch vụ web hiệu quả.

Một số ưu điểm khiến ngôn ngữ lập trình C# và ASP.NET trở nên phổ biến:

* Hiệu suất cao và bảo mật tốt: C# và ASP.NET cung cấp hiệu suất vượt trội nhờ vào việc biên dịch mã nguồn sang mã máy (C#) và tối ưu hóa quá trình xử lý dữ liệu trong ASP.NET. Ngoài ra, ASP.NET còn tích hợp các tính năng bảo mật mạnh mẽ giúp bảo vệ các ứng dụng web khỏi các cuộc tấn công như SQL Injection, XSS, CSRF, v.v.
* Mã nguồn mở và miễn phí: C# và ASP.NET Core là mã nguồn mở, miễn phí và có thể sử dụng trên nhiều nền tảng như Windows, Linux và macOS. Điều này giúp tiết kiệm chi phí phát triển và triển khai ứng dụng.
* Tính linh hoạt: C# có thể được sử dụng để phát triển ứng dụng trên nhiều nền tảng, từ các ứng dụng máy tính để bàn, di động đến các ứng dụng web phức tạp. ASP.NET hỗ trợ phát triển ứng dụng web đa dạng từ các trang web động đến các API RESTful, WebSocket, và hơn thế nữa.
* Cộng đồng và thư viện hỗ trợ: C# và ASP.NET có một cộng đồng phát triển mạnh mẽ và số lượng thư viện hỗ trợ phong phú, giúp các nhà phát triển dễ dàng giải quyết các bài toán phức tạp và tối ưu hóa quá trình phát triển ứng dụng.
* Tính tương thích với các hệ quản trị cơ sở dữ liệu: ASP.NET có thể kết nối với nhiều hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, PostgreSQL, SQLite và Oracle, giúp xây dựng các ứng dụng với cơ sở dữ liệu linh hoạt và mạnh mẽ.

Mặc dù có nhiều ưu điểm, C# và ASP.NET cũng có một số hạn chế, đặc biệt là về khối lượng tài nguyên hệ thống khi sử dụng cho các ứng dụng quy mô lớn. Tuy nhiên, những vấn đề này có thể được giải quyết nhờ vào các công cụ tối ưu hóa và cấu hình đúng đắn trong quá trình phát triển.

## Cơ sở dữ liệu MS SQL

MS SQL (Microsoft SQL Server) là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) được phát triển bởi Microsoft, sử dụng ngôn ngữ truy vấn có cấu trúc (SQL). MS SQL quản lý dữ liệu hiệu quả nhờ việc tổ chức dữ liệu dưới dạng các bảng và mối quan hệ giữa các bảng, giúp giảm thiểu sự lặp lại dữ liệu, tăng tính rõ ràng và hiệu quả của dữ liệu.

**Ưu điểm của MS SQL:**

Tốc độ cao, ổn định, dễ sử dụng: MS SQL là cơ sở dữ liệu tốc độ cao, ổn định, dễ sử dụng và hoạt động trên các hệ điều hành Windows, cung cấp nhiều tính năng tiện ích mạnh mẽ.  
Độ bảo mật cao: MS SQL rất thích hợp cho các ứng dụng có truy cập CSDL trên Internet, với các tính năng bảo mật mạnh mẽ như mã hóa, kiểm tra quyền truy cập và chính sách bảo mật nâng cao.  
Khả năng mở rộng: MS SQL hỗ trợ quy mô dữ liệu lớn và có khả năng mở rộng linh hoạt, cho phép xử lý các yêu cầu phức tạp và dữ liệu lớn trong môi trường doanh nghiệp.

**Nhược điểm của MS SQL:**

Chi phí cao: MS SQL yêu cầu giấy phép sử dụng, điều này có thể gây tốn kém cho các tổ chức nhỏ hoặc các dự án cần tiết kiệm chi phí.  
Yêu cầu tài nguyên hệ thống lớn: MS SQL yêu cầu phần cứng và tài nguyên hệ thống mạnh mẽ, đặc biệt là khi xử lý các cơ sở dữ liệu lớn hoặc các tác vụ phức tạp.

# Ngôn ngữ lập trình ReactJs

ReactJS là một thư viện JavaScript được phát triển bởi Facebook, chuyên dùng để xây dựng giao diện người dùng (UI) cho các ứng dụng web. React cho phép phát triển ứng dụng đơn trang (SPA) với khả năng tái sử dụng các thành phần giao diện, tăng hiệu quả phát triển và dễ bảo trì. ReactJS nổi bật với tính năng Virtual DOM, giúp tối ưu hóa hiệu suất của ứng dụng bằng cách giảm thiểu số lần cập nhật giao diện thực sự trên trình duyệt.

**Ưu điểm của ReactJS:**

Dễ sử dụng và học hỏi: ReactJS có cú pháp đơn giản, dễ hiểu và rất dễ tiếp cận đối với lập trình viên JavaScript.  
Tái sử dụng component: ReactJS cho phép tái sử dụng các component giao diện, giúp giảm thiểu sự trùng lặp mã nguồn và tăng tính mô-đun cho ứng dụng.  
Tốc độ và hiệu suất cao: Nhờ vào Virtual DOM, ReactJS có khả năng cập nhật giao diện nhanh chóng và tối ưu hóa hiệu suất của ứng dụng web.  
Cộng đồng mạnh mẽ: ReactJS có một cộng đồng phát triển lớn mạnh, với rất nhiều tài liệu, plugin và thư viện hỗ trợ.

**Nhược điểm của ReactJS:**

SEO khó khăn: Mặc dù có thể cải thiện SEO với các kỹ thuật như server-side rendering (SSR), nhưng ReactJS không tối ưu cho việc tối ưu hóa công cụ tìm kiếm (SEO) mặc định.  
Tài liệu chưa đầy đủ: Một số phần của ReactJS có tài liệu chưa đầy đủ hoặc khó hiểu, đòi hỏi người học phải tự nghiên cứu hoặc tìm kiếm từ cộng đồng.  
Cập nhật nhanh chóng: Với sự phát triển nhanh chóng của ReactJS, việc cập nhật và làm quen với các thay đổi mới trong thư viện có thể gây khó khăn cho lập trình viên.

## Kết luận chương 1

Chương 1 đã trình bày tổng quan về hệ thống hỗ trợ tự học và đánh giá trực tuyến, nhu cầu xây dựng hệ thống thi lập trình trực tuyến cho các trường đại học. Từ đó, đề tài nghiên cứu được xác định với mục tiêu phát triển một hệ thống cho phép sinh viên lập trình trực tiếp trên web, gửi bài và chấm điểm tự động bằng test case. Để đáp ứng các yêu cầu này, các công nghệ phù hợp được lựa chọn là ngôn ngữ lập trình ASP.NET Core, ReactJS, hệ quản trị CSDL MS SQL, và sử dụng Docker để triển khai hệ thống.

# CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Yêu cầu chức năng và phi chức năng

### Yêu cầu chức năng đối với sinh viên:

* Đăng nhập: Sinh viên có thể đăng nhập vào hệ thống bằng mã sinh viên và mật khẩu đã được trường cấp trước đó.
* Xem danh sách học phần: Sinh viên có thể xem danh sách các học phần mà đã đăng ký học trước đó.
* Xem bài học và tài liệu: Sinh viên có thể xem các bài học, tài liệu hướng dẫn và các bài tập liên quan đến từng học phần.
* Làm bài tập lập trình: Sinh viên có thể viết mã nguồn và làm các bài tập lập trình trực tiếp trên hệ thống.
* Xem kết quả chấm điểm: Sinh viên có thể xem kết quả chấm điểm tự động sau khi nộp bài tập, bao gồm số test case đã pass, điểm số và thời gian thực thi.
* Quản lý thành tích học tập: xem các thành tích học tập, làm bài tập của mình qua các học phần.

### Yêu cầu đối với giảng viên:

* Quản lý bài tập: Giảng viên có thể thêm, sửa, xóa bài tập lập trình, thiết lập các test case và ví dụ cho từng bài tập.
* Quản lý sinh viên: Quản trị viên có thể quản lý tài khoản sinh viên, xem thông tin và theo dõi tiến độ học tập của sinh viên.
* Chấm bài tự động: Quản trị viên có thể cấu hình hệ thống chấm bài tự động và theo dõi kết quả của sinh viên.
* Báo cáo và thống kê: Giảng viên có thể xem báo cáo thống kê kết quả làm bài của sinh viên, các vấn đề nổi bật trong quá trình học tập và đánh giá.

### Các yêu cầu phi chức năng:

* Giao diện thân thiện với người dùng: Giao diện người dùng phải dễ sử dụng, trực quan, đẹp mắt và dễ dàng thao tác cho sinh viên và giảng viên.
* Tốc độ và hiệu suất cao: Hệ thống phải có tốc độ tải trang nhanh, có khả năng xử lý và tương tác mượt mà, đặc biệt trong việc chấm điểm tự động và xử lý bài tập lập trình.
* Độ khả dụng và tin cậy cao: Hệ thống phải đảm bảo luôn sẵn sàng và ổn định, tránh xảy ra lỗi trong quá trình làm bài và chấm điểm.
* Bảo mật và toàn vẹn dữ liệu: Dữ liệu của sinh viên và kết quả bài tập phải được bảo mật và đảm bảo tính toàn vẹn, không bị mất mát hoặc sai lệch.
* Khả năng mở rộng và bảo trì dễ dàng: Hệ thống phải có khả năng mở rộng để đáp ứng nhu cầu người dùng trong tương lai và dễ dàng bảo trì, cập nhật các tính năng mới.

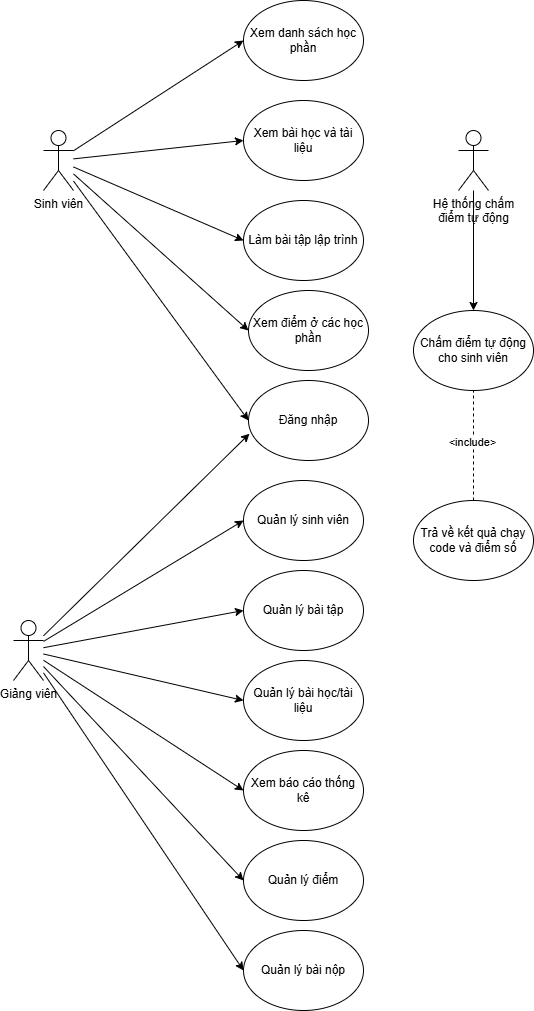
## Mô hình hóa chức năng

## Danh sách các Actor

*Bảng 2.1: Danh sách các Actor*

|  |  |  |
| --- | --- | --- |
| **STT** | **Actor** | **Ghi chú** |
| 1 | Sinh viên | Là người sử dụng chính hệ thống để tham gia học các học phần và làm bài tập lập trình. |
| 2 | Giảng viên | Là người quản lý hệ thống, tạo các bài tập ở các học phần và các bài test lập trình. |
| 3 | Hệ thống chấm điểm tự động | Là hệ thống tích hợp vào để tự động chấm điểm các bài tập lập trình của sinh viên. |

## Biểu đồ use case

****

*Hình 2.1: Biểu đồ Use case tổng quát của hệ thống*

## Mô tả chi tiết các Use Case

## Use case Đăng nhập hệ thống

*Bảng 2.2 : Use case Đăng nhập hệ thống*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Đăng nhập hệ thống | | |
| Tác nhân | | Sinh viên, giảng viên | | |
| Mô tả | | Use case này cho phép sinh viên, giảng viên sử dụng tài khoản để đăng nhập vào hệ thống | | |
| Sự kiện kích hoạt | | Sinh viên truy cập trang đăng nhập và gửi thông tin tài khoản | | |
| Tiền điều kiện | | Tài khoản sinh viên đã được đăng ký | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Sinh viên, giảng viên truy cập trang đăng nhập | Hiển thị form đăng nhập |
| 2 | Sinh viên, giảng viêng nhập mã và mật khẩu, ấn “Đăng nhập” | Kiểm tra thông tin hợp lệ và cho phép truy cập hệ thống |
| Luồng sự kiện rẽ nhánh | 2.1 | Nhập sai mật khẩu | Hiển thị thông báo lỗi "Mật khẩu không đúng" |
| 1  .. 2 |  | Tại bất kỳ bước nào trong luồng sự kiện chính, nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Sinh viên được đăng nhập vào hệ thống và chuyển đến giao diện trang chính | | |

### Use case Xem danh sách học phần

*Bảng 2.3 : Use case Xem danh sách học phần*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Xem danh sách học phần | | |
| Tác nhân | | Sinh viên | | |
| Mô tả | | Sau khi đăng nhập, sinh viên có thể xem danh sách các học phần mà mình tham gia | | |
| Sự kiện kích hoạt | | Sinh viên chọn mục "Học phần" trên giao diện | | |
| Tiền điều kiện | | Sinh viên đã đăng nhập vào hệ thống | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Tác nhân hành động** | **Hệ thống phản hồi** |
| 1 | Truy cập mục "Học phần" |  |
| 2 |  | Truy vấn và hiển thị danh sách học phần tương ứng với sinh viên |
| Luồng sự kiện rẽ nhánh | 2.1 |  | Nếu chưa có học phần nào Hiển thị thông báo "Bạn chưa được phân vào học phần nào" |
| 1  .. 2 |  | Tại bất kỳ bước nào trong luồng sự kiện chính, nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Sinh viên có thể chọn học phần để vào xem nội dung | | |

### Use case Truy cập bài học

*Bảng 2.4 : Use case Truy cập bài học*

|  |  |
| --- | --- |
| Tên use case | Truy cập bài học |
| Tác nhân | Sinh viên |
| Mô tả | Sinh viên chọn học phần và truy cập các bài học lý thuyết |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sự kiện kích hoạt | | Sinh viên chọn học phần cụ thể | | |
| Tiền điều kiện | | Học phần có ít nhất một bài học | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Nhấp vào bài học cụ thể |  |
| 2 |  | Hiển thị nội dung bài học với các tài liệu |
| Luồng sự kiện rẽ nhánh | 1  .. 2 |  | Tại bất kỳ bước nào trong luồng sự kiện chính, nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Không | | |

### Use case Làm bài tập lập trình

*Bảng 2.5 : Use case làm bài tập lập trình*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Làm bài tập lập trình | | |
| Tác nhân | | Sinh viên | | |
| Mô tả | | Sinh viên thực hiện các bài tập lập trình trực tiếp trên hệ thống sau khi học xong lý thuyết | | |
| Sự kiện kích hoạt | | Sinh viên chọn một bài tập từ bài học | | |
| Tiền điều kiện | | Sinh viên đã đăng nhập và chọn một bài học có bài tập lập trình | | |
| Luồng sự kiện | Luồng sự kiện | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn bài học và nhấn vào bài tập | Hiển thị giao diện bài tập lập trình gồm: đề bài, ví dụ, khung viết code |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | chính | 2 | Đọc đề, viết code vào ô soạn thảo | Cho phép chọn ngôn ngữ lập trình, hiển thị nút “Gửi bài” |
| Luồng sự kiện rẽ nhánh | 2.1 |  | Sinh viên không code gì mà bấm “gửi bài” hiển thị cảnh báo “Hãy viết code mới được gửi” |
| 2.2 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Sinh viên hoàn thành việc viết code, sẵn sàng gửi bài để hệ thống chấm điểm | | |

### Use case Gửi code và chấm bài

*Bảng 2.6 : Use case Gửi code và chấm bài*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Gửi code và chấm bài | | |
| Tác nhân | | Sinh viên | | |
| Mô tả | | Sau khi làm bài, sinh viên gửi code lên hệ thống để được chấm điểm tự động | | |
| Sự kiện kích hoạt | | Sinh viên nhấn nút “Gửi bài” sau khi hoàn thành phần code | | |
| Tiền điều kiện | | Bài làm đã được viết và có thể biên dịch | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Nhấn nút “Gửi bài” |  |
| 2 |  | Gửi code đến Piston API để thực thi |
| 3 |  | Nhận kết quả chấm (pass/fail theo test case) |
|  | 4 |  | Hiển thị mã đã nộp và kết quả chấm (test case, điểm, thời gian...) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Luồng sự kiện rẽ nhánh | 1 | Chưa làm bài nào | Hiển thị thông báo “Chưa có bài nộp nào” |
| 3.2 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Không | | |

## Use case Theo dõi tiến độ học tập

*Bảng 2.7 : Use case Theo dõi tiến độ học tập*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Theo dõi tiến độ học tập | | |
| Tác nhân | | Sinh viên | | |
| Mô tả | | Hệ thống cung cấp thông tin tổng quan về các bài học đã hoàn thành, số bài tập đã làm, điểm trung bình,... | | |
| Sự kiện kích hoạt | | Sinh viên chọn mục “Tiến độ học tập” trên giao diện | | |
| Tiền điều kiện | | Sinh viên đã thực hiện ít nhất một hoạt động học hoặc làm bài tập | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Truy cập mục “Tiến độ học tập” |  |
| 2 |  | Lấy dữ liệu về các bài học đã mở, bài tập đã làm, điểm trung bình |
| 3 |  | Hiển thị bảng tiến độ học tập dưới dạng biểu đồ hoặc danh sách |
| Luồng sự kiện rẽ nhánh | 3.1 |  | Nếu sinh viên chưa làm bài tập nào Hiển thị thông báo “Bạn chưa bắt đầu học phần nào” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 3.2 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Sinh viên có thể biết được mình đang học tới đâu và cần cải thiện phần nào | | |

### Use case Tạo bài học và tài liệu dạy

*Bảng 2.8 : Use case Tạo bài học và tài liệu dạy*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Tạo bài học và tài liệu | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Giảng viên tạo các bài học mới, biên soạn nội dung tài liệu, thêm vào hệ thống để sinh viên học tập | | |
| Sự kiện kích hoạt | | Giảng viên đăng nhập và truy cập vào mục quản lý bài học | | |
| Tiền điều kiện | | Giảng viên đã đăng nhập vào hệ thống | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn thêm bài học | Hiển thị form nhập thông tin bài học (tiêu đề, mô tả, tài liệu PDF, video, v.v.) |
| 2 | Nhập đầy đủ thông tin và tải lên tài liệu |  |
| 3 |  | Lưu bài học vào hệ thống, hiển thị trong danh sách bài học cho sinh viên |
| Luồng sự kiện rẽ nhánh | 2.1 |  | Nếu quên nhập thông tin gì đó hệ thống hiển thị “Vui lòng nhập .. cho bài học” |
| 3.1 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Bài học mới được tạo thành công và có thể hiển thị cho sinh viên | | |

### Use case Tạo bài tập lập trình

*Bảng 2.9 : Use case Tạo bài tập lập trình*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Tạo bài tập lập trình | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Giảng viên tạo các bài tập lập trình, thêm đề bài, ví dụ, test case chấm điểm | | |
| Sự kiện kích hoạt | | Giảng viên chọn bài học và nhấn “Thêm bài tập lập trình” | | |
| Tiền điều kiện | | Bài học đã tồn tại trong hệ thống | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn bài học cần thêm bài tập | Hiển thị giao diện thêm bài tập (tiêu đề, mô tả, ví dụ input/output) |
| 2 | Nhập nội dung đề bài, ví dụ minh họa và test case | Lưu bài tập vào hệ thống, liên kết với bài học tương ứng |
| Luồng sự kiện rẽ nhánh | 2.1 |  | Nếu Không nhập test case, Cảnh báo “Vui lòng nhập ít nhất một test case để chấm điểm” |
| 2.2 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Bài tập lập trình đã sẵn sàng cho sinh viên làm và hệ thống có thể tự động chấm điểm | | |

### Use case Xem kết quả bài làm của Sinh viên

*Bảng 2.10 : Use case Xem kết quả của Sinh viên*

|  |  |
| --- | --- |
| Tên use case | Xem kết quả làm bài của sinh viên |
| Tác nhân | Giảng viên |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mô tả | | Giảng viên theo dõi, đánh giá kết quả bài tập sinh viên đã nộp | | |
| Sự kiện kích hoạt | | Giảng viên truy cập trang quản lý kết quả bài tập | | |
| Tiền điều kiện | | Có sinh viên đã nộp bài | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Truy cập phần “Kết quả bài làm” |  |
| 2 |  | Hiển thị danh sách sinh viên và các bài tập đã nộp |
| 3 | Chọn sinh viên hoặc bài cụ thể để xem chi tiết |  |
| 4 |  | Hiển thị mã nguồn, kết quả test case, thời gian nộp |
| Luồng sự kiện rẽ nhánh | 1 | Chọn một sinh viên chưa nộp bài nào |  |
| 2 |  | Hiển thị thông báo “Chưa có bài nộp” |
| 3 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 3 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Giảng viên có thể đánh giá tiến độ và năng lực của sinh viên | | |

### Use case Quản lý bài học và bài tập

*Bảng 2.11 : Use case Quản lý bài học và bài tập*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Quản lý bài học và bài tập | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Giảng viên có thể sửa, xóa bài học hoặc bài tập không còn phù hợp | | |
| Sự kiện kích hoạt | | Giảng viên truy cập phần danh sách bài học hoặc bài tập | | |
| Tiền điều kiện | | Bài học hoặc bài tập đã tồn tại | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn bài học hoặc bài tập cần chỉnh sửa |  |
| 2 |  | Hiển thị nội dung chi tiết |
| 3 | Thực hiện cập nhật nội dung hoặc xóa bài |  |
| 4 |  | Lưu thay đổi hoặc xóa dữ liệu tương ứng |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Luồng sự kiện rẽ nhánh | 1 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Dữ liệu bài học/bài tập được cập nhật chính xác trong hệ thống | | |

### Use case Quản lý sinh viên

*Bảng 2.12 : Use case Quản lý sinh viên*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Quản lý sinh viên | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Xem danh sách sinh viên, thông tin tài khoản, theo dõi tiến độ học tập và kết quả làm bài của từng sinh viên | | |
| Sự kiện kích hoạt | | Giảng viên truy cập mục "Quản lý Sinh viên" từ trang quản trị | | |
| Tiền điều kiện | | Sinh viên đã đăng ký tài khoản và tham gia vào hệ thống | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Truy cập trang quản lý sinh viên |  |
| 2 |  | Hiển thị danh sách sinh viên đang hoạt động |
| 3 | Chọn sinh viên cụ thể để xem chi tiết |  |
| 4 |  | Hiển thị thông tin cá nhân, bài học đã học, số bài đã làm và điểm số |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Luồng sự kiện rẽ nhánh | 1 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Thông tin sinh viên được hiển thị, giảng viên có thể đánh giá quá trình học tập | | |

### Use case Thống kê và báo cáo

*Bảng 2.13 : Use case Thống kê và báo cáo*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Thống kê và báo cáo | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Xem thống kê về số lượng bài nộp, tỷ lệ hoàn thành, điểm trung bình theo từng học phần hoặc bài tập | | |
| Sự kiện kích hoạt | | Truy cập chức năng thống kê từ trang quản trị | | |
| Tiền điều kiện | | Có dữ liệu bài nộp từ sinh viên | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn học phần hoặc bài tập cụ thể |  |
| 2 |  | Truy vấn và hiển thị các số liệu thống kê gồm: tổng số sinh viên, số bài nộp, số bài hoàn thành, điểm trung bình |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 3 | Chọn xuất báo cáo |  |
| 4 |  | Tạo file PDF hoặc Excel và cho phép tải xuống |
| Luồng sự kiện rẽ nhánh | 1 | Chọn học phần chưa có sinh viên tham gia |  |
| 2 |  | Thông báo “Không có dữ liệu để thống kê” |
| 3 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Không | | |

### Use case Quản lý học phần

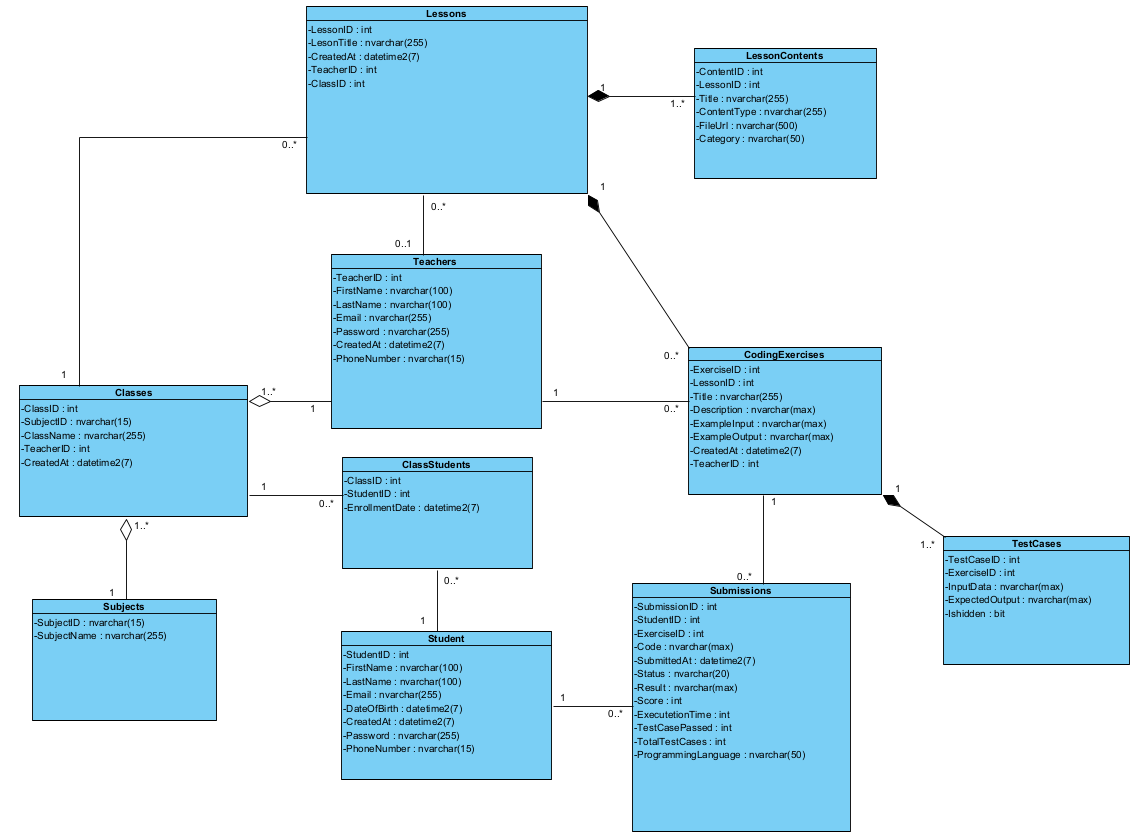
*Bảng 2.14 : Use case Quản lý học phần*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | Quản lý học phần | | |
| Tác nhân | | Giảng viên | | |
| Mô tả | | Tạo, cập nhật, xóa các học phần (môn học) và liên kết với bài học, bài tập tương ứng | | |
| Sự kiện kích hoạt | | Giảng viên truy cập chức năng quản lý học phần | | |
| Tiền điều kiện | | Tài khoản có quyền quản trị | | |
| Luồng sự kiện | Luồng sự kiện chính | # | **Hành động của tác nhân** | **Hệ thống phản hồi** |
| 1 | Chọn menu “Quản lý học phần”. |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 2 |  | Hiển thị danh sách các học phần lên màn hình |
| 3 | Nếu chọn “Thêm học phần” thực hiện các bước 4, 5 |  |
| 4 | Nhập thông tin học phần và chọn Thêm mới |  |
| 5 |  | Lưu thông tin Học phần và hiển thị danh sách học phần mới lên màn hình, UC kết thúc. |
| 6 | Nếu chọn 1 Học phần đã có để cập nhật thông tin, thực hiện bước 7,8. |  |
| 7 | Thực hiện sửa thông tin Học phần chọn cập nhật. Hoặc chọn xóa học phần |  |
| 8 |  | Cập nhật thông tin học phần. Hiển thị danh sách Học phần mới lên màn hình, UC kết thúc. |
| Luồng sự kiện rẽ nhánh | 2.1 |  | Nếu không có học phần nào, hiển thị thông báo “không có học phần nào”. |
| 1..  8 |  | Nếu không kết nối được với CSDL thì hệ thống sẽ thông báo lỗi và UC kết thúc. |
| Hậu điều kiện | | Học phần được tạo và có thể gán bài học, bài tập | | |

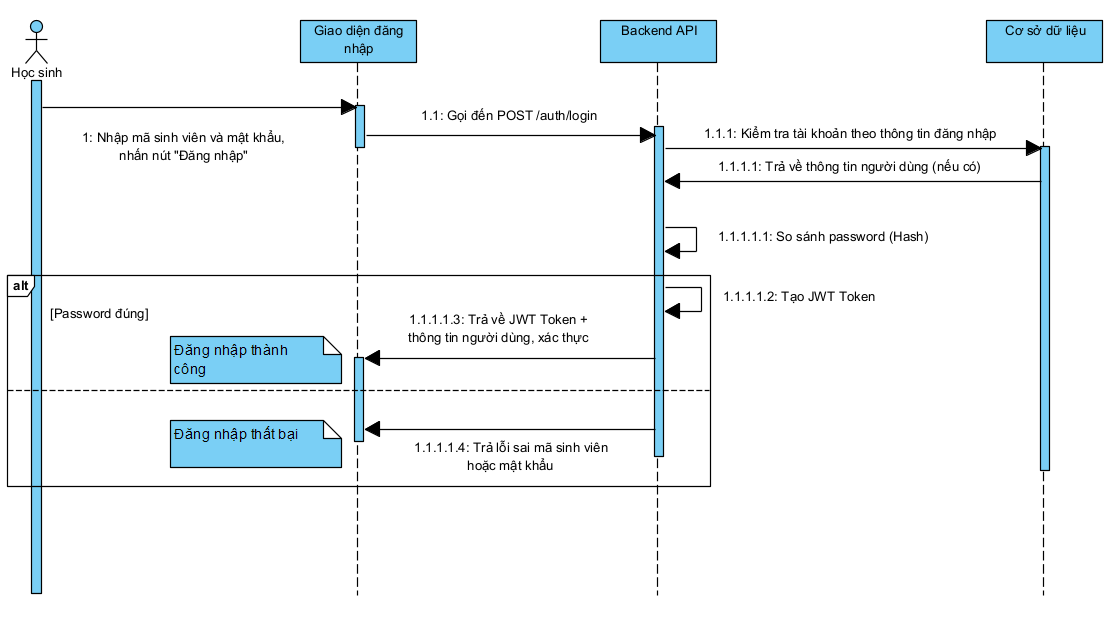
## Thiết kế chi tiết

## Mô hình hóa dữ liệu

****

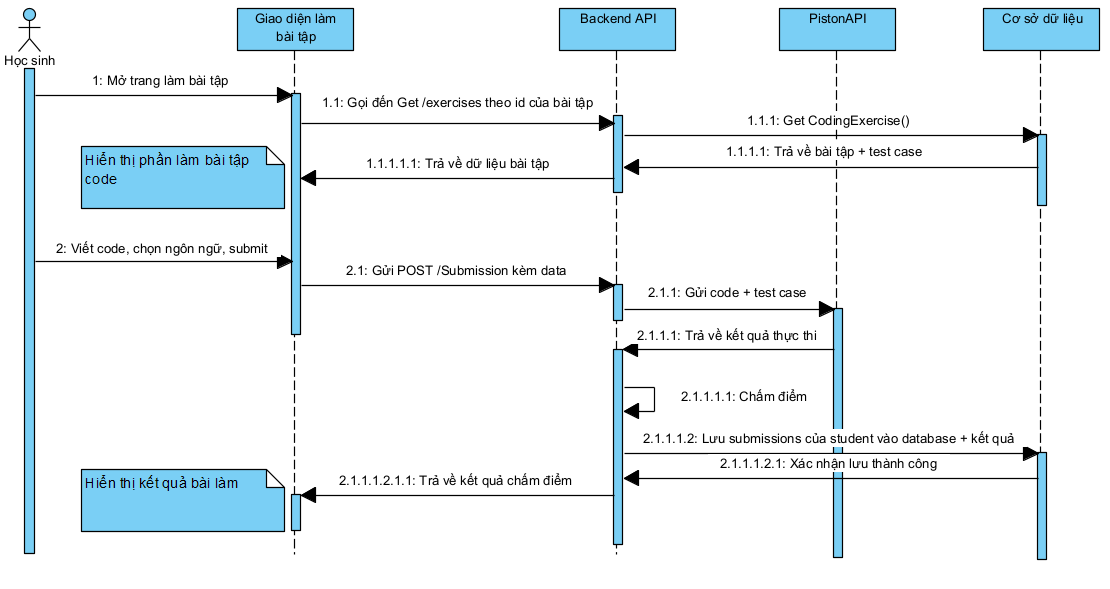
*Hình 2.2: Biểu đồ lớp thiết kế hệ thống*

## Biểu đồ trình tự use case Đăng nhập

****

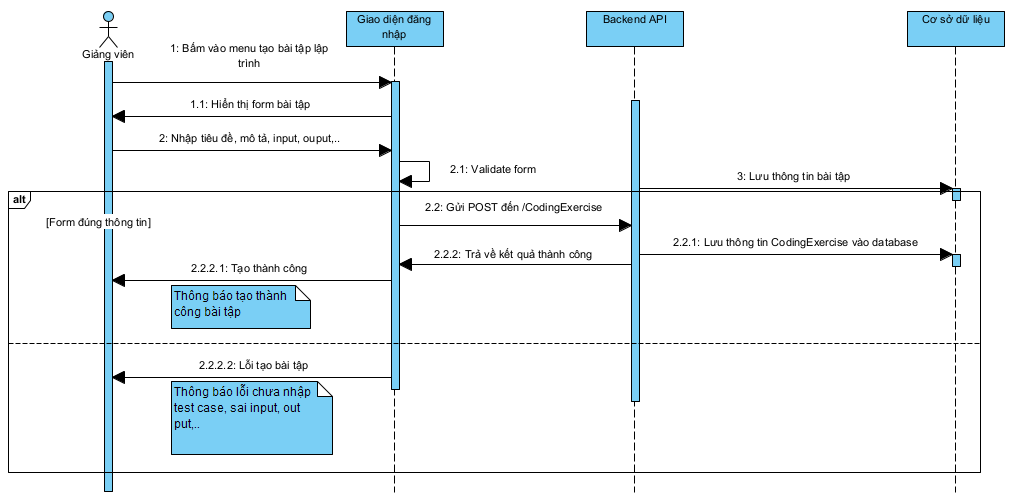
*Hình 2.3: Biểu đồ trình tự UC Đăng nhập*

## Biểu đồ trình tự use case Làm bài tập lập trình

****

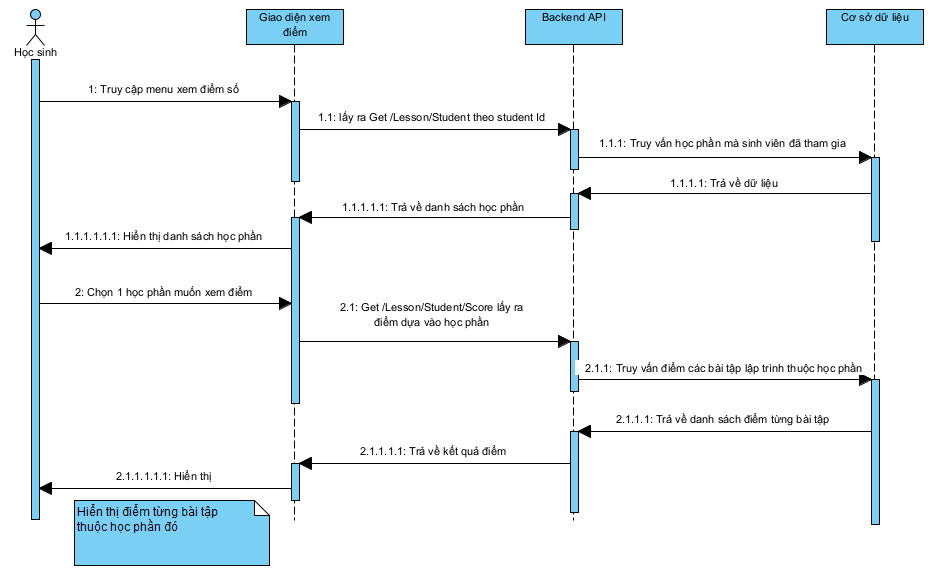
*Hình 2.4: Biểu đồ trình tự UC Làm bài tập lập trình*

## Biểu đồ trình tự use case Tạo bài tập lập trình

****

*Hình 2.5: Biểu đồ trình tự UC Tạo bài tập lập trình*

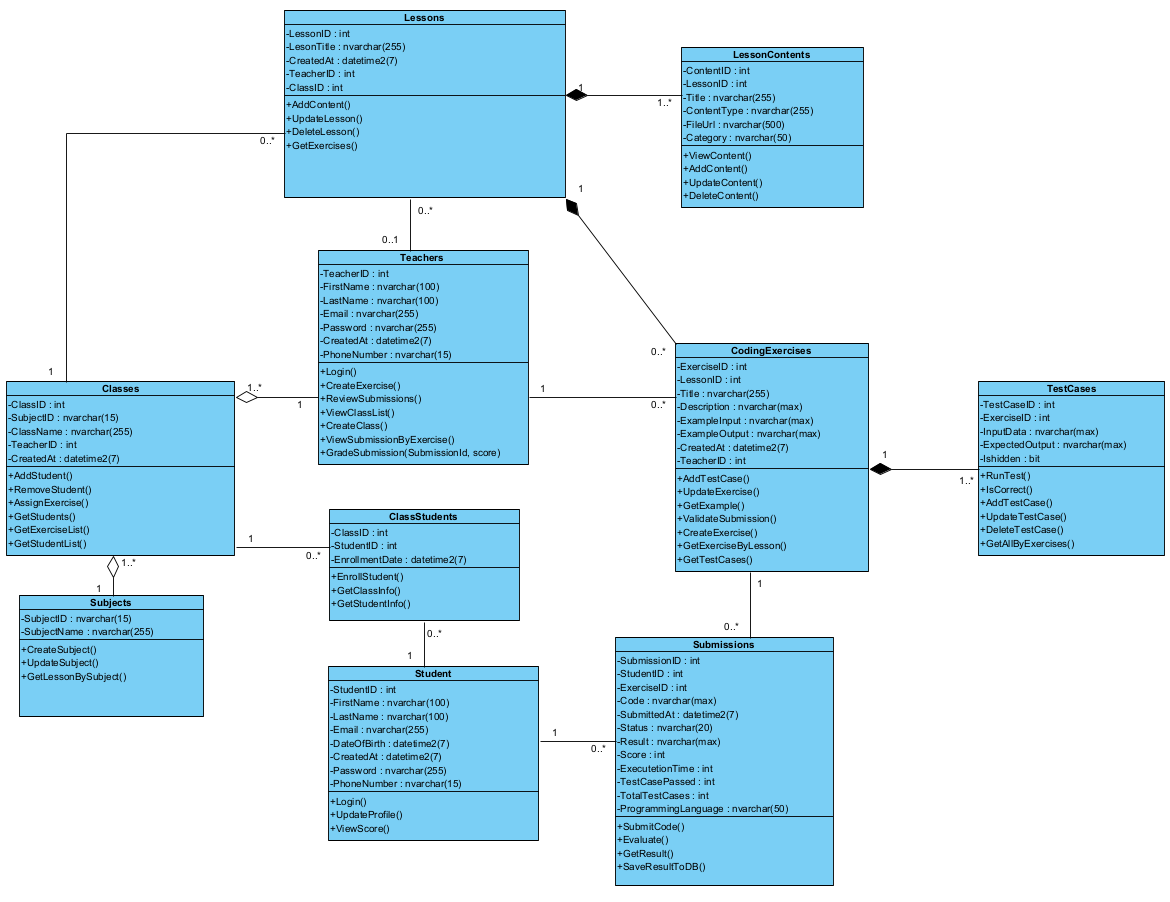
## Biểu đồ trình tự use case Theo dõi kết quả học

****

*Hình 2.6: Biểu đồ trình tự UC Theo dõi kết quả học*

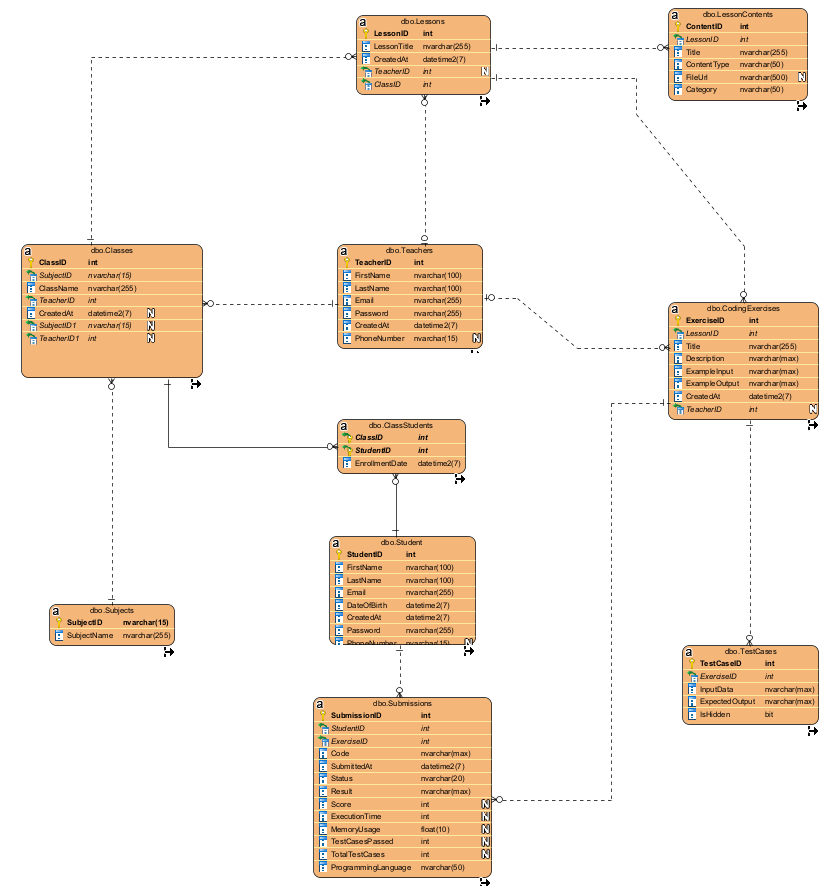
## Biểu đồ lớp chi tiết

Hình 2.7 là biểu đồ lớp sau khi thêm các phương thức từ biểu đồ tuần tự.



*Hình 2.7: Biểu đồ lớp chi tiết.*

## Thiết kế cơ sở dữ liệu mức vật lý

****

*Hình 2.8 : Biểu đồ thực thể liên kết mức vật lý*

## Bảng Subjects (Học phần)

*Bảng 2.20: Bảng Subjects*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | SubjectId | nvarchar(15) | Mã học phần | Khóa chính |
| 2 | SubjectName | nvarchar(255) | Tên học phần |  |

## Bảng Student (Sinh viên)

*Bảng 2.21: Bảng Student*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | StudentID | int | Mã sinh viên | Khóa chính |
| 2 | FirstName | Nvarchar(100) | Họ |  |
| 3 | LastName | Nvarchar(100) | Tên |  |
| 4 | Email | Nvarchar(255) | Email |  |
| 5 | DateOfBirth | Datetime2(7) | Ngày sinh |  |
| 6 | CreatedAt | Datetime2(7) | Thời gian tạo |  |
| 7 | Password | Nvarchar(255) | Mật khẩu |  |
| 8 | PhoneNumber | Nvarchar(15) | Số điện thoại |  |

## Bảng ClassStudents (Lớp học chứa student)

*Bảng 2.22: Bảng ClassStudent*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | ClassID | int | Mã lớp học | Khóa chính |
| 2 | StudentID | int | Mã sinh viên | Khóa chính |
| 3 | EnrollmentDate | Datetime2(7) | Ngày tham gia học |  |

## Bảng Classes (Bảng lớp học)

*Bảng 2.23: Bảng Classes*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | ClassID | int | Mã lớp học | Khóa chính |
| 2 | SubjectID | Nvarchar(15) | Mã học phần | Khóa ngoại |
| 3 | ClassName | Nvarchar(255) | Tên lớp học |  |
| 4 | TeacherID | Int | Mã giảng viên | Khóa ngoại |
| 5 | CreatedAt | Datetime2(7) | Ngày tạo lớp |  |

## Bảng Teachers (Giảng viên)

*Bảng 2.24: Bảng Teachers*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | TeacherID | Int | Mã giảng viên | Khóa chính |
| 2 | FirstName | Nvarchar(100) | Họ |  |
| 3 | LastName | Nvarchar(100) | Tên |  |
| 4 | Email | Nvarchar(255) | Email |  |
| 5 | Password | Nvarchar(255) | Password |  |
| 6 | CreatedAt | Datetime2(7) | Ngày tạo |  |
| 7 | PhoneNumber | Nvarchar(15) | Số điện thoại |  |

## Bảng Lessons (Bài học)

*Bảng 2.25: Bảng Lessons*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | LessonID | Int | Mã bài học | Khóa chính |
| 2 | LessonTitle | Nvarchar(255) | Tên tiêu đề |  |
| 3 | CreateAt | Datetime2(7) | Ngày tạo bài học |  |
| 4 | TeacherID | int | Mã giảng viên | Khóa ngoại |
| 5 | ClassID | int | Mã lớp học | Khóa ngoại |

## Bảng LessonContents (Nội dung bài học)

*Bảng 2.26: Bảng LessonContents*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | ContentID | int | Mã nội dung bài học | Khóa chính |
| 2 | LessonID | Int(11) | Mã bài học | Khóa ngoại |
| 3 | Title | Nvarchar(255) | Tên tiêu đề |  |
| 4 | ContentType | Nvarchar(255) | Thể loại (File,PDF) |  |
| 5 | FileUrl | Nvarchar(500) | Đường dẫn |  |
| 6 | Category | Nvarchar(50) | Thể loại |  |

## Bảng TestCases (Kiểm thử)

*Bảng 2.27: Bảng TestCases*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | TestCaseID | Int | Mã test case | Khóa chính |
| 2 | ExerciseID | Int | Mã bài tập | Khóa ngoại |
| 3 | InputData | Nvarchar(max) | Đầu vào |  |
| 4 | ExpectedOutput | Nvarchar(max) | Mong đợi đầu ra |  |
| 5 | IsHidden | bit | Ẩn/hiện test case |  |

## Bảng CodingExercises (Bài tập code)

*Bảng 2.28: Bảng CodingExercises*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | ExerciseID | Int | Mã bài tập | Khóa chính |
| 2 | LessonID | Int | Mã học phần | Khóa ngoại |
| 3 | Title | Nvarchar(255) | Tiêu đề |  |
| 4 | Description | Nvarchar(max) | Mô tả bài tập |  |
| 5 | ExampleInput | Nvarchar(max) | Ví dụ đầu vào |  |
| 6 | ExampleOutput | Nvarchar(max) | Ví dụ đầu ra |  |
| 7 | CreatedAt | Datetime2(7) | Ngày tạo |  |
| 8 | TeacherID | int | Mã giảng viên | Khóa ngoại |

## Bảng Submission (Bài nộp)

*Bảng 2.29: Bảng Submission*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Kiểu dữ liệu** | **Giải thích** | **Ghi chú** |
| 1 | SubmissionID | Int | Mã bài nộp | Khóa chính |
| 2 | StudentID | Int | Mã sinh viên | Khóa ngoại |
| 3 | ExerciseID | Int | Mã bài tập | Khóa ngoại |
| 4 | Code | Nvarchar(max) | Đoạn code |  |
| 5 | SubmittedAt | Datetime2(7) | Ngày nộp |  |
| 6 | Status | varchar(20) | Trạng thái |  |
| 7 | Result | Nvarchar(max) | Kết quả trả về |  |
| 8 | Score | Int | Điểm của sinh viên |  |
| 9 | ExcutetionTime | Int | Thời gian thực thi |  |
| 10 | TestCasePassed | Int | Số test case vượt |  |
| 11 | TotalTestCases | Int | Tổng test case |  |
| 12 | ProgramingLanguage | varchar(50) | Ngôn ngữ lập trình |  |