

**TRƯỜNG ĐẠI HỌC ĐÀ LẠT**

**KHOA CÔNG NGHỆ THÔNG TIN**

**--🙞🕮🙜--**



**BÀI BÁO CÁO MÔN**

**MẪU THIẾT KẾ**

**ĐỀ TÀI: Math Quiz – Ứng dụng luyện tập Toán học sử dụng mẫu**

**thiết kế Strategy trong Java**

Giảng Viên: Đoàn Minh Khuê

Sinh viên thực hiện:Nguyễn Thị Ánh Trúc

MSSV: 221279

Lớp: CTK46A

Đà Lạt, ngày 29 tháng 10 năm 2025

Nhận Xét Của Giảng Viên

Đà lạt, ngày......tháng 10 năm 2025

**Giảng viên bộ môn**

(Ký tên)

Lời cảm ơn

Em xin gửi lời cảm ơn chân thành và sâu sắc đến thầy Đoàn Minh Khuê, giảng viên phụ trách môn Mẫu Thiết kế, người đã tận tình giảng dạy, hướng dẫn và hỗ trợ em trong suốt quá trình học tập cũng như thực hiện bài báo cáo cuối kỳ này.

Nhờ sự chỉ dẫn tận tâm và những góp ý quý báu của thầy, em đã hiểu rõ hơn về cách áp dụng các mẫu thiết kế hướng đối tượng trong lập trình Java, đặc biệt là khả năng vận dụng vào việc xây dựng và mở rộng ứng dụng thực tế.

Đây là những kiến thức có giá trị, giúp em rèn luyện tư duy phân tích, thiết kế phần mềm và trau dồi kỹ năng lập trình chuyên nghiệp hơn.

Em xin chân thành cảm ơn thầy đã dành thời gian, tâm huyết và sự tận tình để giúp em hoàn thiện bài báo cáo này.  
Em kính chúc thầy luôn dồi dào sức khỏe, thành công trong công tác giảng dạy và nghiên cứu.

**Sinh viên thực hiện**

Nguyễn Thị Ánh Trúc

Mục lục

[Danh mục hình ảnh 6](#_Toc212400608)

[Mở Đầu 7](#_Toc212400609)

[Chương 1. Mục tiêu và công cụ sử dụng 8](#_Toc212400610)

[1.1 Mục tiêu của đề tài 8](#_Toc212400611)

[1.1.1 Mục tiêu tổng quát 8](#_Toc212400612)

[1.1.2 Mục tiêu cụ thể 9](#_Toc212400613)

[1.2 Công cụ sử dụng và môi trường phát triển 10](#_Toc212400614)

[1.2.1 Công cụ sử dụng 10](#_Toc212400615)

[1.2.2 Môi trường chạy chương trình 11](#_Toc212400616)

[1.2.3 Cấu trúc thư mục dự án 12](#_Toc212400617)

[Chương 2. Cơ sở lý thuyết 14](#_Toc212400618)

[2.1 Lập trình hướng đối tượng OOP 14](#_Toc212400619)

[2.2 Mẫu thiết kế Strategy Pattern 14](#_Toc212400620)

[2.3 Mẫu thiết kế Decorator Pattern 15](#_Toc212400621)

[2.4 Mẫu thiết kế Composite Pattern 16](#_Toc212400622)

[2.5 Cơ sở dữ liệu (SQL Server) 17](#_Toc212400623)

[Chương 3. Phân tích và thiết kế hệ thống 19](#_Toc212400624)

[3.1 Phân tích chức năng 19](#_Toc212400625)

[3.2 Yêu cầu chức năng 19](#_Toc212400626)

[3.3 Sơ đồ use case 20](#_Toc212400627)

[3.4 Sơ đồ luồng hoạt động 21](#_Toc212400628)

[3.5 Sơ đồ lớp 22](#_Toc212400629)

[3.6 Cấu trúc tổng thể của hệ thống 23](#_Toc212400630)

[Chương 4. Thiết kế giao diện 27](#_Toc212400631)

[4.1 Tổng quan giao diện 27](#_Toc212400632)

[4.2 Cấu trúc màn hình chính 28](#_Toc212400633)

[4.2.1 Màn hình khởi động 28](#_Toc212400634)

[4.2.2 Màn hình chính 28](#_Toc212400635)

[4.3 Màn hình xếp hạng 33](#_Toc212400636)

[4.4 Chế độ hiển thị sáng và tối (Light / Dark Mode) 35](#_Toc212400637)

[4.5 Các đoạn mã minh họa 37](#_Toc212400638)

[4.6 Cơ sở dữ liệu SQL 39](#_Toc212400639)

[4.6.1 Cấu trúc cơ sở dữ liệu 39](#_Toc212400640)

[4.6.2 Kết nối SQL 41](#_Toc212400641)

[Chương 5. Hướng phát triển và kết luận 44](#_Toc212400642)

[5.1 Hướng phát triển 44](#_Toc212400643)

[5.2 Kết luận 45](#_Toc212400644)

[Tài liệu tham khảo 46](#_Toc212400645)

# Danh mục hình ảnh

[Hình 1: Cấu trúc thư mục 12](#_Toc212400059)

[Hình 2: Sơ đồ use case 21](#_Toc212400060)

[Hình 3: Sơ đồ luồng hoạt động 21](#_Toc212400061)

[Hình 4: Sơ đồ lớp 22](#_Toc212400062)

[Hình 5: Core 23](#_Toc212400063)

[Hình 6: Strategy.question 23](#_Toc212400064)

[Hình 7: Strategy.score 24](#_Toc212400065)

[Hình 8: Difficulty 25](#_Toc212400066)

[Hình 9: db 25](#_Toc212400067)

[Hình 10: Leaderboard 25](#_Toc212400068)

[Hình 11: UI 26](#_Toc212400069)

[Hình 12: Màn hình chờ 28](#_Toc212400070)

[Hình 13: Màn hình chính 29](#_Toc212400071)

[Hình 14: Đặt lại phiên 29](#_Toc212400072)

[Hình 15: Các chế độ 30](#_Toc212400073)

[Hình 16: Hướng dẫn 31](#_Toc212400074)

[Hình 17: Trả lời đúng 31](#_Toc212400075)

[Hình 18: trả lời sai 32](#_Toc212400076)

[Hình 19: Minh họa cho Footer 33](#_Toc212400077)

[Hình 20: Màn hình xếp hạng 33](#_Toc212400078)

[Hình 21: Thông báo 34](#_Toc212400079)

[Hình 22: Chế độ tối 36](#_Toc212400080)

[Hình 23: Chế độ tối 37](#_Toc212400081)

[Hình 24: Sơ đồ diagram 40](#_Toc212400082)

[Hình 25: Kết nối 42](#_Toc212400083)

[Hình 26: Cấu hình 42](#_Toc212400084)

[Hình 27: Đoạn mã 43](#_Toc212400085)

[Hình 28: Thông báo kết nối 43](#_Toc212400086)

# Mở Đầu

Trong thời đại ngày nay, công nghệ thông tin đang đóng vai trò quan trọng trong mọi lĩnh vực, đặc biệt là trong giáo dục. Việc kết hợp học tập với yếu tố trò chơi (game-based learning) đang trở thành xu hướng phổ biến vì giúp người học cảm thấy hứng thú hơn, tiếp thu kiến thức nhanh hơn và ghi nhớ lâu hơn.

Môn Toán học là một môn học nền tảng nhưng thường được cho là khô khan, khó tiếp cận. Chính vì vậy, nếu có một trò chơi nhỏ giúp người học vừa rèn luyện tư duy toán học vừa được giải trí, việc học sẽ trở nên nhẹ nhàng và thú vị hơn.

Từ ý tưởng đó, em chọn thực hiện đề tài *“Math Quiz – Ứng dụng luyện tập Toán học sử dụng mẫu thiết kế Strategy trong Java”*. Ứng dụng này vừa mang tính học tập, vừa giúp em rèn luyện kỹ năng lập trình hướng đối tượng (OOP) và cách áp dụng các mẫu thiết kế phần mềm trong thực tế.

Đề tài được xây dựng với các mục tiêu chính:

* Sử dụng mẫu Strategy Pattern để quản lý linh hoạt nhiều dạng câu hỏi toán học khác nhau.
* Cho phép thay đổi cách tính điểm và độ khó mà không cần chỉnh sửa cấu trúc chương trình.
* Giúp người dùng có trải nghiệm dễ chịu nhờ giao diện đơn giản, có chế độ sáng/tối và bảng xếp hạng (Leaderboard) để tạo động lực chơi.
* Rèn luyện khả năng tư duy, phân tích và thiết kế phần mềm theo hướng mở rộng.

Ứng dụng *“Math Quiz”* có thể được dùng để luyện tập Toán cơ bản hàng ngày, hoặc dùng làm minh họa cho các mẫu thiết kế hướng đối tượng trong môn học *Mẫu Thiết Kế*. Với nền tảng Java, ứng dụng có thể dễ dàng mở rộng hoặc chuyển sang dạng game di động trong tương lai.

# Mục tiêu và công cụ sử dụng

## Mục tiêu của đề tài

### Mục tiêu tổng quát

Đề tài *“Math Quiz”* được thực hiện với mục tiêu chính là xây dựng một ứng dụng trò chơi luyện tập Toán học đơn giản, thân thiện và có tính tương tác. Ứng dụng giúp người dùng rèn luyện các phép tính cơ bản như cộng, trừ, nhân, chia thông qua hình thức trả lời nhanh các câu hỏi ngẫu nhiên.

Qua đó, việc học Toán trở nên nhẹ nhàng, gần gũi hơn và tạo cảm giác hứng thú thay vì khô khan như cách học truyền thống.

Bên cạnh yếu tố học tập, đồ án còn hướng tới việc áp dụng các kiến thức về lập trình hướng đối tượng (OOP) vào thực tế, đặc biệt là các mẫu thiết kế phần mềm (Design Patterns) như Strategy, Decorator và Composite. Việc vận dụng các mẫu này giúp chương trình dễ dàng mở rộng thêm loại câu hỏi, cách tính điểm hoặc cấp độ khó mà không phải chỉnh sửa mã nguồn hiện có.

Đề tài cũng đã được mở rộng thêm phần lưu trữ dữ liệu bằng Microsoft SQL Server, thay thế cho phương pháp lưu bằng file truyền thống.

Điều này giúp ứng dụng:

* Quản lý thông tin người chơi, điểm số và chuỗi đúng một cách ổn định và an toàn hơn.
* Dễ dàng thực hiện các truy vấn thống kê, xếp hạng thông qua Stored Procedures và Views.
* Có khả năng mở rộng quy mô dữ liệu khi triển khai trên môi trường thực tế hoặc nhiều người dùng.

Thông qua quá trình thực hiện, sinh viên có cơ hội:

* Củng cố kiến thức về OOP, hiểu rõ vai trò của việc tách lớp, đóng gói chức năng và tái sử dụng mã nguồn.
* Làm quen với quy trình xây dựng một phần mềm hoàn chỉnh, từ khâu thiết kế, cài đặt đến chạy thử và kiểm tra.
* Sử dụng thành thạo các công cụ phát triển hiện đại như NetBeans IDE, Maven, và các thư viện hỗ trợ giao diện Java Swing.
* Nâng cao khả năng tư duy logic, kỹ năng trình bày và viết báo cáo kỹ thuật.

Mục tiêu cuối cùng của đề tài là giúp sinh viên không chỉ tạo ra một sản phẩm chạy được, mà còn hiểu rõ tư duy thiết kế và quy tắc mở rộng phần mềm một cách khoa học.

Ứng dụng “Math Quiz” vì vậy vừa là bài học về kỹ thuật lập trình, vừa là minh chứng cho tư duy sáng tạo và tinh thần tự học trong lĩnh vực phát triển phần mềm.

### Mục tiêu cụ thể

Từ mục tiêu tổng quát đã nêu, đề tài “Math Quiz” hướng đến các mục tiêu cụ thể sau:

1. *Xây dựng ứng dụng luyện tập Toán học trực quan:*  
   Ứng dụng được thiết kế theo dạng trò chơi hỏi – đáp, giúp người chơi luyện tập các phép tính cơ bản (cộng, trừ, nhân, chia) thông qua những câu hỏi ngẫu nhiên.  
   Giao diện được thiết kế thân thiện, dễ sử dụng và có hỗ trợ tiếng Việt để phù hợp với người học ở nhiều độ tuổi.
2. *Áp dụng mẫu thiết kế hướng đối tượng trong Java*:  
   Sử dụng Strategy Pattern để quản lý các loại câu hỏi khác nhau, cho phép mở rộng thêm các dạng bài mới mà không cần sửa đổi phần lõi.  
   Bên cạnh đó, kết hợp Decorator Pattern để bổ sung tính năng thưởng điểm hoặc giảm điểm tùy điều kiện, và Composite Pattern để cộng gộp nhiều chiến lược tính điểm lại với nhau.
3. *Phát triển hệ thống linh hoạt, dễ mở rộng:*Ứng dụng được tổ chức thành nhiều gói (package) độc lập, mỗi gói đảm nhiệm một chức năng cụ thể. Điều này giúp chương trình dễ dàng nâng cấp, thêm câu hỏi, hoặc chỉnh sửa giao diện mà không ảnh hưởng đến toàn bộ hệ thống.
4. *Tạo bảng xếp hạng (Leaderboard):*  
   Lưu trữ và quản lý thành tích người chơi thông qua cơ sở dữ liệu SQL Server, thay vì sử dụng file JSON.

Hệ thống sử dụng bảng dữ liệu (Players, GameSession) cùng các View và Stored Procedure để truy xuất, tổng hợp và sắp xếp kết quả theo điểm số, thời gian hoàn thành và chuỗi đúng.  
Chức năng này giúp người chơi có thêm động lực, tạo yếu tố thi đua nhẹ trong quá trình học và cho phép mở rộng lưu trữ trên nhiều thiết bị một cách an toàn, bền vững.

1. *Tăng cường kỹ năng lập trình thực tế:*Qua việc thực hiện đề tài, sinh viên rèn luyện khả năng phân tích bài toán, tư duy thiết kế hệ thống, cũng như kỹ năng xử lý lỗi, quản lý mã nguồn bằng Git và Maven.  
   Ngoài ra, việc làm việc với thư viện giao diện Swing giúp sinh viên làm quen với lập trình ứng dụng có giao diện đồ họa (GUI) trong Java.
2. *Đảm bảo tính thân thiện và dễ sử dụng:*  
   Giao diện được xây dựng với hai chế độ hiển thị (sáng và tối), bố cục rõ ràng, màu sắc dễ nhìn, nút bấm có hiệu ứng và phản hồi tốt.  
   Mục tiêu là để người dùng có trải nghiệm tốt nhất khi sử dụng phần mềm.

## Công cụ sử dụng và môi trường phát triển

### Công cụ sử dụng

Trong quá trình phát triển đề tài **“Math Quiz”**, sinh viên sử dụng các công cụ và môi trường sau:

* **Ngôn ngữ lập trình:** Java  
  → Ngôn ngữ hướng đối tượng phổ biến, phù hợp cho lập trình ứng dụng desktop và áp dụng các mẫu thiết kế.
* **Trình quản lý dự án:** Apache Maven  
  → Quản lý thư viện, tự động biên dịch, giúp tổ chức cấu trúc dự án rõ ràng, dễ bảo trì.
* **Môi trường phát triển (IDE):** NetBeans  
  → Hỗ trợ thiết kế giao diện bằng Java Swing, biên dịch, chạy và gỡ lỗi hiệu quả.
* **Thư viện giao diện:** Java Swing  
  → Tạo giao diện trực quan, dễ sử dụng, có thể tùy chỉnh màu sắc (chế độ sáng/tối).
* **Hệ quản trị cơ sở dữ liệu:** Microsoft SQL Server  
  → Lưu trữ và quản lý dữ liệu người chơi, điểm số, chuỗi đúng, thời gian và bảng xếp hạng.  
  → Kết nối thông qua JDBC, có hỗ trợ **Stored Procedure** và **View** giúp truy vấn nhanh và an toàn.
* **Công cụ quản lý mã nguồn:** Git và GitHub  
  → Theo dõi lịch sử thay đổi, đồng bộ dự án, và sao lưu mã nguồn trực tuyến.
* **PlantUML: Hỗ trợ vẽ sơ đồ bằng mã giúp sinh viên thực hiện nhanh hơn.**

### Môi trường chạy chương trình

Đề tài được phát triển và kiểm thử trên môi trường máy tính cá nhân với cấu hình trung bình, đảm bảo có thể chạy ổn định trên hầu hết các thiết bị phổ thông.

Cấu hình khuyến nghị:

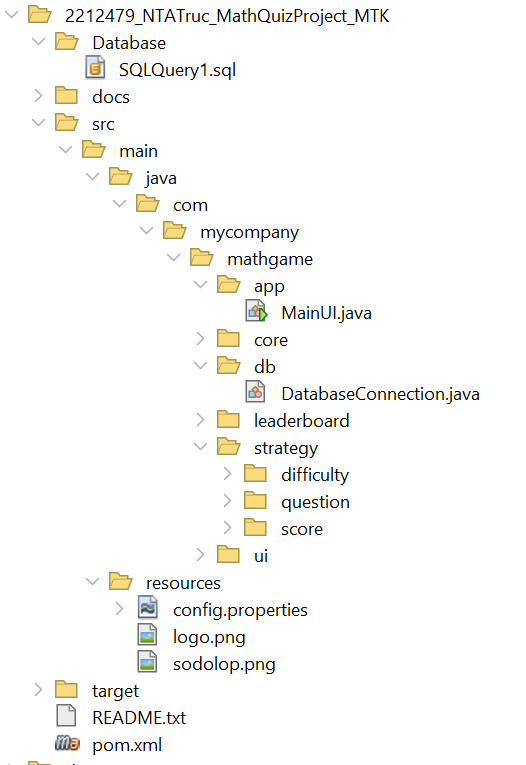
* Hệ điều hành: Windows 10 hoặc Windows 11.
* Bộ nhớ RAM: Tối thiểu 4GB.
* Bộ xử lý: Intel Core i3 hoặc tương đương (khuyến nghị Core i5 trở lên).
* Độ phân giải màn hình: 1280×720 trở lên.
* Công cụ build: Apache Maven (tích hợp trong NetBeans IDE).

Ứng dụng hoạt động hoàn toàn offline, không yêu cầu kết nối Internet

Tất cả dữ liệu người chơi, điểm số và bảng xếp hạng được lưu trữ cục bộ trong cơ sở dữ liệu SQL Server, sử dụng **tệp cấu hình SQL (.sql)** cụ thể là file *SQLQuery1.sql* đặt trong thư mục db của dự án để khởi tạo và quản lý dữ liệu.

### Cấu trúc thư mục dự án

Dự án được tổ chức theo *chuẩn Maven*, giúp tách biệt rõ ràng giữa mã nguồn, tài nguyên và file biên dịch.



Hình 1: Cấu trúc thư mục

* Trong đó:
* Database: chứa file databse SQL server(SQLQuery1.sql)
* dos: Thư mục chứa tài liệu bài báo cáo.
* app: Thư mục chứa lớp khởi động ứng dụng.
* core: Các lớp lõi như Question, GameState, Timer.
* db: Chứa lớp quản lý kết nối SQL Server
* leaderboard: Quản lý bảng xếp hạng và lưu kết quả.
* strategy: Chứa các mẫu thiết kế Strategy, Decorator, Composite.
* difficulty: Điều chỉnh độ khó dựa vào kết quả người chơi.
* question: Sinh câu hỏi (cộng, nhân, chia, trộn…).
* ui: Giao diện chính và bảng xếp hạng (MathQuizFrame, LeaderboardPanel)
* config.properties: file cấu hình kết nối SQL Server (ví dụ user, password, database).
* logo.png: Hình ảnh logo hiển thị khi khởi động (Splash Screen).
* target: Mục sinh ra sau khi biên dịch Maven.
* README.txt: Hướng dẫn chạy và mô tả nhanh về dự án.
* pom.xml: File cấu hình Maven (khai báo thư viện và project info).

# Cơ sở lý thuyết

## Lập trình hướng đối tượng OOP

Lập trình hướng đối tượng (Object-Oriented Programming – OOP) là một phương pháp phát triển phần mềm dựa trên các đối tượng (objects).

Mỗi đối tượng chứa dữ liệu (thuộc tính) và hành vi (phương thức), giúp mô phỏng các thực thể trong thế giới thực.

Mục tiêu của OOP là tăng khả năng tái sử dụng, dễ bảo trì và mở rộng hệ thống mà không làm thay đổi cấu trúc cốt lõi.

Các nguyên lý cơ bản của OOP bao gồm:

* Tính đóng gói (Encapsulation): Giấu đi chi tiết bên trong của đối tượng, chỉ cho phép truy cập thông qua các phương thức công khai.
* Tính kế thừa (Inheritance): Cho phép một lớp kế thừa thuộc tính và hành vi của lớp khác, giúp giảm trùng lặp mã nguồn.
* Tính đa hình (Polymorphism): Một phương thức có thể được định nghĩa lại (override) để hoạt động khác nhau tùy vào đối tượng cụ thể.
* Tính trừu tượng (Abstraction): Che giấu các chi tiết không cần thiết và chỉ hiển thị các tính năng quan trọng.

Trong đề tài này, OOP là nền tảng giúp sinh viên xây dựng hệ thống “Math Quiz” có cấu trúc rõ ràng, tách biệt từng chức năng (quản lý câu hỏi, tính điểm, xếp hạng), đồng thời dễ mở rộng nhờ các mẫu thiết kế như Strategy, Decorator, và Composite.

## Mẫu thiết kế Strategy Pattern

Strategy Pattern là một mẫu thiết kế hành vi (Behavioral Pattern) cho phép định nghĩa một họ các thuật toán, đóng gói riêng từng thuật toán, và hoán đổi chúng linh hoạt trong lúc chương trình chạy.

Thay vì viết nhiều câu lệnh *if...else* để xử lý các trường hợp khác nhau, Strategy Pattern giúp chương trình chọn chiến lược phù hợp mà không cần sửa mã gốc.

Trong dự án *Math Quiz*, Strategy Pattern được áp dụng để:

* Sinh các loại câu hỏi toán học khác nhau: cộng, trừ, nhân, chia, và hỗn hợp.
* Tạo ra chiến lược tính điểm riêng biệt: tính điểm theo thời gian, theo chuỗi đúng, hoặc tổng hợp.

Mỗi loại câu hỏi hoặc cách tính điểm là một “chiến lược” (strategy) riêng.  
Người chơi có thể thay đổi cách chơi hoặc cách tính điểm mà không cần thay đổi toàn bộ chương trình.

* Ưu điểm:
* Dễ mở rộng: thêm chiến lược mới mà không ảnh hưởng đến phần còn lại.
* Giảm phức tạp, loại bỏ nhiều cấu trúc if...else.
* Linh hoạt, có thể thay đổi thuật toán ngay trong quá trình chạy.
* Nhược điểm:
* Có thể làm tăng số lượng lớp trong hệ thống.
* Cần tổ chức tốt để tránh nhầm lẫn giữa các chiến lược.
* Khi nên sử dụng:
* Khi có nhiều cách khác nhau để thực hiện một hành vi.
* Khi muốn thay đổi thuật toán mà không ảnh hưởng đến mã cũ.
* Khi cần tách biệt phần “cách thực hiện” khỏi phần “logic điều khiển”.

## Mẫu thiết kế Decorator Pattern

Decorator Pattern là mẫu thiết kế cấu trúc (Structural Pattern) cho phép thêm chức năng mới cho đối tượng mà không cần sửa mã gốc.

Trong *Math Quiz*, Decorator được sử dụng để thêm phần thưởng điểm:

* Thưởng khi trả lời đúng liên tiếp (Streak Bonus).
* Thưởng khi trả lời nhanh trong thời gian quy định (Timed Bonus).

Ví dụ:  
Một chiến lược tính điểm cơ bản có thể được **bọc thêm (decorate)** bằng lớp TimedBonusDecorator để cộng thêm điểm thưởng.  
Điều này giúp hệ thống mở rộng linh hoạt, dễ bảo trì, và có thể kết hợp nhiều lớp Decorator để tạo ra các hiệu ứng cộng dồn.

* Ưu điểm:
* Dễ mở rộng chức năng mà không cần chỉnh sửa lớp gốc.
* Có thể kết hợp nhiều Decorator khác nhau linh hoạt.
* Tuân thủ nguyên tắc “Open/Closed” (mở để mở rộng, đóng để sửa đổi).
* Nhược điểm:
* Tăng số lượng lớp và mức độ phức tạp.
* Có thể khó debug khi nhiều Decorator lồng nhau.
* Khi nên sử dụng:
* Khi cần mở rộng chức năng của lớp hiện tại mà không muốn hoặc không thể sửa đổi mã gốc.
* Khi muốn tạo ra nhiều “biến thể” hành vi của cùng một đối tượng mà không cần kế thừa.

## Mẫu thiết kế Composite Pattern

Composite Pattern là mẫu thiết kế cấu trúc giúp kết hợp nhiều đối tượng nhỏ thành một đối tượng tổng hợp (composite) nhưng vẫn xử lý như một đối tượng duy nhất.

Trong *Math Quiz*, Composite được dùng để:

* Kết hợp nhiều chiến lược tính điểm lại với nhau, ví dụ: tổng hợp giữa điểm cố định, thưởng thời gian và thưởng chuỗi đúng.
* Quản lý danh sách chiến lược thống nhất, giúp chương trình dễ dàng thêm hoặc bớt cách tính điểm mà không làm thay đổi logic tổng thể.

Nhờ đó, ứng dụng có thể tạo ra nhiều biến thể của cách tính điểm mà vẫn giữ nguyên cấu trúc chính.

* Ưu điểm:
* Giúp xử lý tập hợp các đối tượng thống nhất như đối tượng đơn lẻ.
* Dễ mở rộng: thêm hoặc xóa thành phần mà không ảnh hưởng đến hệ thống.
* Giúp tổ chức mã nguồn rõ ràng khi có nhiều phần tử tương tự nhau.
* Nhược điểm:
* Thiết kế phức tạp hơn so với cách cài đặt đơn giản.
* Có thể khó quản lý khi cấu trúc phân cấp quá sâu.
* Khi nên sử dụng:
* Khi muốn nhóm nhiều đối tượng có cùng bản chất để xử lý như một.
* Khi cần tổ chức các phần tử theo dạng cây (Tree Structure).

## Cơ sở dữ liệu (SQL Server)

* Khái niệm:

SQL Server là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển.

Nó cho phép lưu trữ và quản lý dữ liệu có cấu trúc dưới dạng bảng (table), dùng ngôn ngữ *SQL (Structured Query Language)* để truy vấn, thêm, sửa và xóa dữ liệu.

Trong đề tài *Math Quiz*, SQL Server được dùng để lưu thông tin người chơi, điểm số, thời gian và bảng xếp hạng — giúp dữ liệu được quản lý tập trung, đáng tin cậy và dễ truy xuất.

SQL Server hỗ trợ tốt trong môi trường Windows, dễ tích hợp với Java thông qua *JDBC Driver (mssql-jdbc).*

* Ưu điểm:
* **Độ tin cậy cao:** Dữ liệu được lưu cố định trong cơ sở dữ liệu, không bị mất khi tắt chương trình.
* **Truy vấn linh hoạt:** Có thể lọc, sắp xếp, thống kê hoặc tính toán dữ liệu nhanh chóng bằng các câu lệnh SQL.
* **Bảo mật tốt:** Hỗ trợ phân quyền người dùng và xác thực khi kết nối.
* **Dễ tích hợp:** Kết nối trực tiếp với ứng dụng Java thông qua **JDBC driver**, không cần phần mềm trung gian.
* **Hỗ trợ xử lý phức tạp:** Có thể dùng Stored Procedure, View, Trigger để xử lý logic ngay trong cơ sở dữ liệu.
* Nhược điểm:
* **Cài đặt nặng hơn so với các cách lưu dữ liệu đơn giản (như file JSON hoặc text).**
* **Cần cấu hình kết nối (user, password, port) trước khi sử dụng.**
* **Khó triển khai nếu người dùng không quen với môi trường SQL Server.**
* **Khi nên sử dụng:**
* **Khi cần lưu dữ liệuổn định, lâu dài.**
* **Khi ứng dụng có nhiều người chơi hoặc nhiều bản ghi, không thể quản lý bằng file thông thường.**
* **Khi muốn thống kê, sắp xếp, tìm kiếm dữ liệu nhanh chóng bằng câu lệnh SQL.**
* **Khi cần mở rộng ứng dụng trong tương lai (ví dụ: nhiều người dùng, máy chủ riêng, hoặc web server).**

# Phân tích và thiết kế hệ thống

## Phân tích chức năng

Ứng dụng Math Quiz là một trò chơi luyện tập toán học giúp người chơi ôn tập kỹ năng tính toán cơ bản thông qua các câu hỏi được tạo ngẫu nhiên.

Hệ thống được xây dựng theo hướng lập trình hướng đối tượng (OOP) và áp dụng các mẫu thiết kế phần mềm (Design Patterns) để tăng khả năng mở rộng, tái sử dụng, và bảo trì.

Mục tiêu của hệ thống là giúp người chơi học toán một cách thú vị hơn thông qua hình thức trò chơi tương tác.

Người chơi vừa được rèn luyện tư duy logic, vừa cảm thấy hứng thú khi điểm số và cấp độ của họ thay đổi theo quá trình chơi.

Ngoài ra, dữ liệu người chơi và thành tích được lưu trữ trực tiếp trong cơ sở dữ liệu SQL Server, giúp đảm bảo tính ổn định, an toàn và có thể thống kê, mở rộng trong tương lai.

## Yêu cầu chức năng

* **FR1: Tạo câu hỏi toán học ngẫu nhiên:**

Hệ thống tự động sinh câu hỏi theo cấp độ (Level) và chiến lược (Strategy Pattern). Mỗi cấp độ sẽ có độ khó tăng dần.

* **FR2: Kiểm tra câu trả lời:**  
  Người chơi nhập đáp án, hệ thống tự động chấm đúng/sai và hiển thị kết quả ngay lập tức.
* **FR3: Tính điểm và cập nhật chuỗi đúng (Streak):**  
  Hệ thống tính điểm dựa trên nhiều chiến lược khác nhau (thời gian, chuỗi đúng liên tiếp...) để tăng tính hấp dẫn và động lực học tập.
* **FR4: Quản lý độ khó tự động:**  
  Hệ thống điều chỉnh cấp độ dựa vào kết quả người chơi (StreakBasedDifficulty).
* **FR5: Lưu trữ dữ liệu:**  
  Kết quả chơi, thời gian, điểm và chuỗi đúng được lưu trong **SQL Server**.
* **FR6: Hiển thị bảng xếp hạng:**  
  Toàn bộ kết quả chơi (tên người chơi, điểm số, thời gian, chuỗi đúng) được lưu vào cơ sở dữ liệu SQL Server, giúp dễ dàng truy xuất và thống kê.
* **FR7: Giao diện thân thiện:**  
  Ứng dụng Java Swing, có bố cục rõ ràng, dễ thao tác.
* **FR8: Khả năng mở rộng:**  
  Dễ thêm chiến lược tính điểm, dạng câu hỏi, hoặc chế độ chơi mới mà không sửa mã nguồn cũ.

## Sơ đồ use case

* Các tác nhân (Actors):
* *Người chơi:* Là người duy nhất tương tác trực tiếp với hệ thống.
* *Hệ thống:* Sinh câu hỏi, kiểm tra đáp án, tính điểm, lưu kết quả và hiển thị bảng xếp hạng.
* Các Use Case chính:
* **UC1 – Bắt đầu trò chơi:** Người chơi khởi động ứng dụng và bắt đầu phiên chơi mới.
* **UC2 – Sinh câu hỏi:** Hệ thống tạo ngẫu nhiên câu hỏi toán học dựa trên cấp độ.
* **UC3 – Trả lời câu hỏi:** Người chơi nhập đáp án và gửi để kiểm tra.
* **UC4 – Kiểm tra kết quả:** Hệ thống xác định đúng/sai, tính điểm và cập nhật chuỗi đúng (streak).
* **UC5 – Cập nhật điểm số:** Hệ thống tính tổng điểm, thời gian và lưu kết quả vào cơ sở dữ liệu SQL Server.
* **UC6 – Xem bảng xếp hạng:** Người chơi mở tab “Xếp hạng” để xem danh sách điểm cao nhất.
* **UC7 – Thoát ứng dụng:** Người chơi kết thúc phiên làm việc, hệ thống lưu lại dữ liệu nếu có thay đổi.

A diagram of a math quiz system

AI-generated content may be incorrect.

Hình 2: Sơ đồ use case

## Sơ đồ luồng hoạt động

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 3: Sơ đồ luồng hoạt động

## Sơ đồ lớp

A diagram of a company

AI-generated content may be incorrect.

Hình 4: Sơ đồ lớp

## Cấu trúc tổng thể của hệ thống

Hệ thống được chia thành các gói (package) chính, mỗi gói đảm nhiệm một chức năng riêng biệt, đảm bảo tính phân tách trách nhiệm (Single Responsibility Principle).

* **core:** Chứa các lớp trung tâm của trò chơi như:

A group of black text

AI-generated content may be incorrect.

Hình 5: Core

* Trong đó:
* Question: Mô tả câu hỏi toán học (đề bài, đáp án, độ khó).
* GameState: Lưu trạng thái hiện tại của người chơi (điểm, chuỗi đúng, cấp độ).
* Timer: Đo thời gian trả lời của người chơi.
* AnswerResult: Lưu kết quả của từng lần trả lời, gồm thông tin đúng/sai, thời gian trả lời và điểm đạt được.
* **strategy.question:** Các chiến lược sinh câu hỏi, áp dụng **Strategy Pattern**, gồm:

A close up of a computer screen

AI-generated content may be incorrect.

Hình 6: Strategy.question

* Trong đó
* AdditionStrategy – tạo câu hỏi cộng.
* MultiplicationStrategy – tạo câu hỏi nhân.
* DivisionStrategy – tạo câu hỏi chia.
* MixWithCarryStrategy – tạo câu hỏi hỗn hợp.
* RandomComposite – kết hợp nhiều chiến lược ngẫu nhiên.
* **strategy.score:** Các chiến lược tính điểm, sử dụng Decorator và Composite Pattern:

A screenshot of a computer code

AI-generated content may be incorrect.

Hình 7: Strategy.score

* Trong đó:
* FixedScoreStrategy: Điểm cố định.
* ScoreStrategy – Giao diện (interface) định nghĩa phương thức score() dùng để tính điểm cho mỗi câu trả lời.
* TimedBonusDecorator: Cộng thêm điểm khi trả lời nhanh.
* StreakBonusDecorator: Thưởng chuỗi đúng.
* SumScoreComposite: Tổng hợp nhiều chiến lược tính điểm.
* **strategy.difficulty:** Quản lý độ khó của trò chơi, gồm:

A close up of words

AI-generated content may be incorrect.

Hình 8: Difficulty

* Trong đó:
* Difficulty: Giao diện (interface) định nghĩa phương thức điều chỉnh độ khó cho trò chơi.
* StreakBasedDifficulty: Thay đổi cấp độ dựa trên chuỗi trả lời đúng.
* Db: Quản lý kết nối SQL Server

A close-up of a company logo

AI-generated content may be incorrect.

Hình 9: db

* Trong đó:
* DatabaseConnection.java: Mở kết nối, đóng kết nối, xử lý lỗi.
* SQLQuery1: khởi tạo CSDL.
* **leaderboard:** Quản lý bảng xếp hạng người chơi:

A group of black text

AI-generated content may be incorrect.

Hình 10: Leaderboard

* Trong đó:
* LeaderboardEntry: Lưu thông tin người chơi và điểm số.
* LeaderboardPort: Interface giúp tách rời phần UI và database
* LeaderboardServiceDb: Lưu/xóa/tải dữ liệu trong SQL Server.
* **ui:** Giao diện người dùng, sử dụng thư viện Swing:

A close up of words

AI-generated content may be incorrect.

Hình 11: UI

* Trong đó:
* MathQuizFrame: Giao diện chính.
* LeaderboardPanel: Bảng xếp hạng.



* MainUI: Khởi chạy ứng dụng kèm màn hình Splash Screen.

# Thiết kế giao diện

## Tổng quan giao diện

Giao diện người dùng của ứng dụng *“Math Quiz”* được phát triển bằng **Java Swing**, một thư viện phổ biến trong lập trình ứng dụng desktop của ngôn ngữ Java.

Việc lựa chọn Swing giúp chương trình hoạt động ổn định, dễ dàng tùy chỉnh bố cục, màu sắc và các thành phần hiển thị, đồng thời không cần cài đặt thêm công cụ hỗ trợ ngoài môi trường Java cơ bản.

Mục tiêu khi thiết kế giao diện là tạo nên một không gian học tập nhẹ nhàng, thân thiện và gần gũi với người dùng.  
Người học có thể vừa ôn tập các phép toán cơ bản (cộng, trừ, nhân, chia), vừa cảm thấy hứng thú như đang tham gia một trò chơi rèn luyện trí tuệ.  
Giao diện được thiết kế trực quan, dễ thao tác, giúp người chơi nhanh chóng hiểu cách sử dụng mà không cần hướng dẫn phức tạp.

Phong cách thiết kế tổng thể của ứng dụng đi theo hướng tối giản và hiện đại, ưu tiên sự rõ ràng, tinh gọn trong từng chi tiết.

Các thành phần được bố trí hợp lý: phần tiêu đề ở phía trên, khu vực chơi chính ở giữa và phần thông tin – điểm số ở phía dưới.

Tông màu chủ đạo là các gam sáng nhẹ và xanh lam, tạo cảm giác dịu mắt, phù hợp với tính chất học tập và thi đấu nhẹ nhàng của trò chơi.

Ngoài ra, ứng dụng được thiết kế để đáp ứng linh hoạt các thao tác chính của người chơi, bao gồm:

* Nhập đáp án cho câu hỏi hiện tại.
* Chuyển sang câu hỏi tiếp theo.
* Quan sát bảng xếp hạng tổng hợp điểm.
* Thay đổi chế độ hiển thị sáng hoặc tối tùy theo môi trường sử dụng.

Nhờ đó, giao diện không chỉ phục vụ mục đích học tập, mà còn tạo ra trải nghiệm tương tác vui vẻ, thoải mái, góp phần giúp người dùng duy trì động lực khi rèn luyện kỹ năng toán học mỗi ngày.

## Cấu trúc màn hình chính

### Màn hình khởi động

Khi người dùng khởi chạy ứng dụng Math Quiz, hệ thống sẽ hiển thị màn hình khởi động (Splash Screen) trong vài giây đầu tiên.

Màn hình này có logo ứng dụng, tiêu đề “Math Quiz – Trò chơi luyện tập Toán học”, cùng thanh tiến trình (Progress Bar) chạy trong khoảng 2 giây.

Mục đích của màn hình này là tạo cảm giác chuyên nghiệp, đồng thời giúp người chơi nhận diện thương hiệu của ứng dụng trước khi vào giao diện chính.

Sau khi quá trình khởi tạo hoàn tất, chương trình tự động chuyển đến màn hình chơi chính.

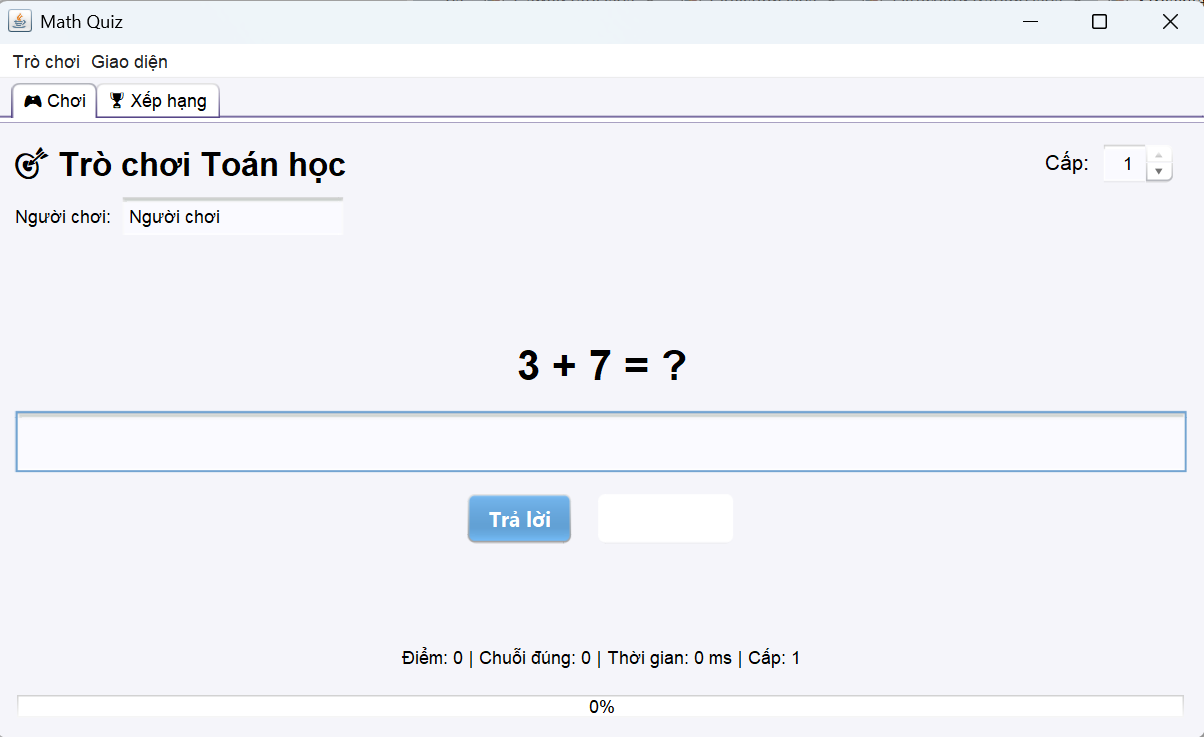
A screenshot of a computer

AI-generated content may be incorrect.

Hình 12: Màn hình chờ

### Màn hình chính

Sau khi khởi động, màn hình chính của trò chơi được bố trí theo hướng rõ ràng, trực quan và dễ thao tác, giúp người chơi dễ dàng hiểu cách sử dụng ngay từ lần đầu.



Hình 13: Màn hình chính

Giao diện được chia thành các khu vực sau:

* *Thanh menu:*

Gồm hai mục chính là *“Trò chơi”* và *“Giao diện”*.

Mục “Trò chơi” chứa chức năng đặt lại phiên → khi nhấn vào đặt lại phiên thì nó sẽ khởi động lại trò chơi cho người.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 14: Đặt lại phiên

Mục *“Giao diện”* cho phép thay đổi giữa *chế độ sáng và tối (Light/Dark Mode),* giúp người dùng chọn phong cách hiển thị phù hợp.

Mặc định của trò chơi sẽ là chế độ sáng.

A close-up of a white screen

AI-generated content may be incorrect.

Hình 15: Các chế độ

* *Phần trên (Header):*
* Hiển thị tiêu đề *“Math Quiz”*: Trò chơi toán học.
* Ô nhập tên người chơi: Có thể nhập tên người chơi khác nhau.
* Cấp độ hiện tại (Level): Khi trả lời đúng liên tục 5 câu thì nó sẽ tăng lên 1 level. Có tổng là 10 level.

Việc hiển thị tên cá nhân cũng tạo cảm giác gắn kết, giúp người dùng cảm thấy mình là nhân vật chính trong trò chơi.

Khi mở ứng dụng, hệ thống mặc định hiển thị tab *“Chơi”*.

* *Phần giữa (Main Panel):*

Đây là nơi diễn ra hoạt động chính của trò chơi.

Người chơi nhìn thấy câu hỏi toán học được tạo ngẫu nhiên theo cấp độ, nhập đáp án vào ô trống và sử dụng hai nút chức năng:

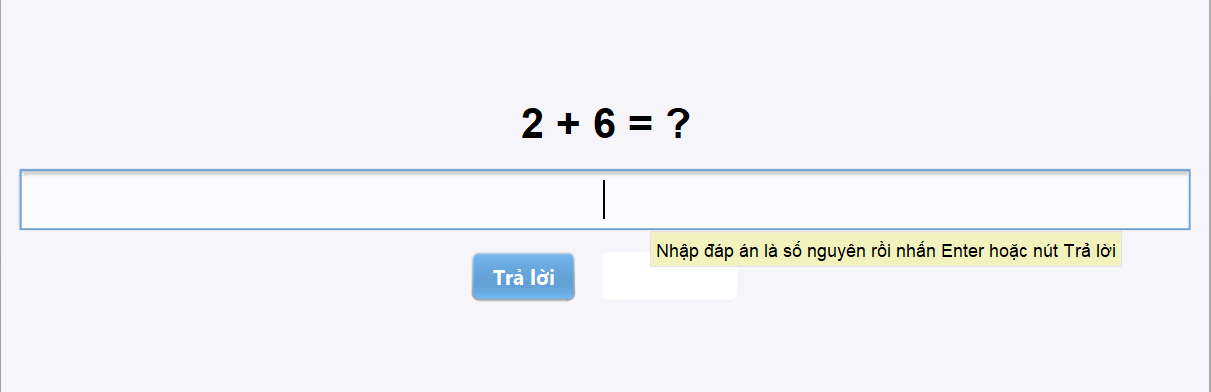
* *“Trả lời”*: Gửi kết quả để hệ thống kiểm tra đúng/sai. Màu nút *“Trả lời”* sẽ là màu xanh dương.

→ Nếu trả lời đúng thì nút “Tiếp theo” mới hiển thị chữ và màu nút, màu nút “Tiếp theo” là hiển thị màu xanh lá đó lên.

* *“Tiếp theo”*: Chuyển sang câu hỏi mới sau khi đã trả lời xong. Khi nút “Tiếp theo” hiển thị lên thì nút *“Trả lời”* không hiển thị lên.

Bố cục phần giữa được sắp xếp cân đối, kích thước chữ lớn và màu sắc dễ nhìn, giúp người chơi tập trung vào nội dung mà không bị rối mắt.

Khi người chơi nhấp vào ô trả lời kết quả thì nó sẽ hiện ra hương dẫn cho người chơi nhập đáp án.



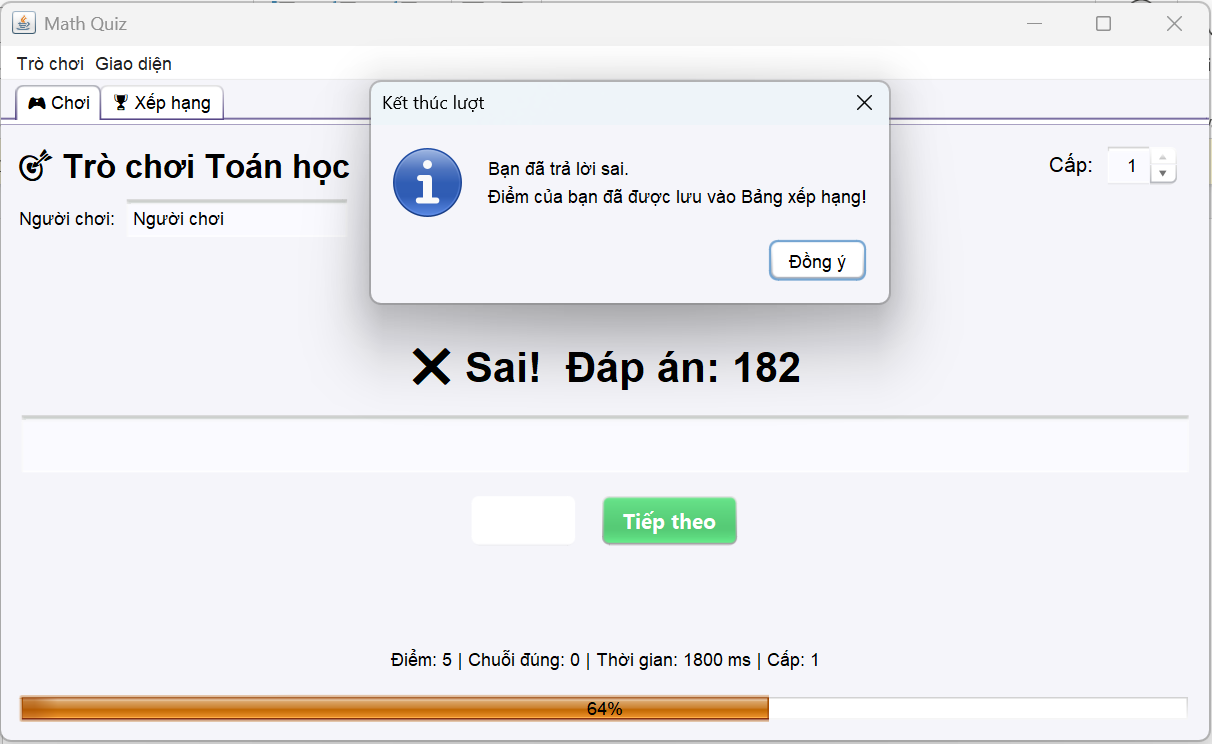
Hình 16: Hướng dẫn

Giao diện khi trả lời đúng câu hỏi đã cho. Khi đáp án đúng thì nút *“Tiếp theo”* hiển thị lên cho người chơi nhấn vào để người chơi có thể tiếp tục câu hỏi tiếp theo.



Hình 17: Trả lời đúng

Màn hình trả lời khi người chơi trả lời sai.

Khi trả lời sai sẽ được thông báo và kết quả đó được lưu ở bảng xếp hạng.

Hình 18: trả lời sai

* *Phần dưới (Footer):*

Hiển thị thông tin trạng thái của trò chơi, bao gồm:

* Điểm hiện tại (Score): Nếu trả lời đúng cộng điểm. Điểm cộng sẽ tùy thuộc vào mức độ trả lời nhanh hay chậm dựa vào thanh tiến trình để cộng điểm. Trả lời sai thì trừ đi điểm. Điểm trừ cũng sẽ tùy thuộc vào mức độ trả lời nhanh hay chậm dựa vào thanh tiến trình, và sẽ lưu kết quả đã trừ điểm bài đó để xếp hạng.
* Chuỗi đúng liên tiếp (Streak): Số câu trả lời đúng liên tiếp. Có nhiều câu đúng điểm cộng càng nhiều.
* Thời gian trả lời (Time): Thời gian mà người chơi trả lời câu hỏi.
* Cấp độ đang chơi (Level): Cấp độ hiện tại của câu hỏi đó. Khi trả lời đúng liên tiếp 7 câu thì cấp đọ sẽ tăng lên.

Phần này cũng chứa thanh tiến trình (Progress Bar), thể hiện tốc độ hoặc mức độ hoàn thành câu hỏi nhanh hay chậm của người chơi trong câu hỏi đó.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 19: Minh họa cho Footer

## Màn hình xếp hạng

A screenshot of a computer

AI-generated content may be incorrect.

Hình 20: Màn hình xếp hạng

Tab *“Xếp hạng”* là màn hình phụ của ứng dụng *Math Quiz*, nơi hiển thị danh sách những người chơi đạt điểm cao nhất sau các lượt chơi.

Tính năng này được thiết kế với mục tiêu tăng tính cạnh tranh và khuyến khích người học luyện tập thường xuyên hơn, đồng thời lưu lại tiến trình học tập của người dùng.

Bảng xếp hạng được trình bày trong tab riêng biệt *“Xếp hạng”*, song song với tab *“Chơi”* trên thanh điều hướng của ứng dụng. Khi người dùng mở tab này, hệ thống sẽ tự động tải dữ liệu từ cơ sở dữ liệu SQL Server, cụ thể từ bảng quiz.GameSession và quiz.Players, thông qua lớp trung gian LeaderboardServiceDb.Khi người chơi trả lời sai câu hỏi, hệ thống sẽ thông báo sai sau đó tự động lưu lại kết quả vào bảng xếp hạng.

Khi người chơi trả lời sai trong quá trình chơi, hệ thống sẽ hiển thị thông báo *“Sai”* và tự động lưu lại kết quả (tên người chơi, điểm số, chuỗi đúng và thời gian) vào cơ sở dữ liệu. Sau đó, người dùng có thể chuyển sang tab *“Xếp hạng”* để xem thành tích của mình cùng với các người chơi khác.

A white background with black text

AI-generated content may be incorrect.

Hình 21: Thông báo

Giao diện được bố trí rõ ràng với các thành phần chính:

* Tiêu đề “BẢNG XẾP HẠNG NGƯỜI CHƠI” được căn giữa và in đậm để tạo điểm nhấn.
* Bảng danh sách (JTable) hiển thị thông tin của từng người chơi với bốn cột chính:
* #: Thứ hạng của người chơi, tự động sắp xếp theo điểm giảm dần.
* Người chơi: Tên do người dùng nhập khi bắt đầu trò chơi.
* Điểm số: Tổng điểm đạt được sau các lượt chơi.
* Chuỗi đúng: Số câu trả lời đúng liên tiếp cao nhất (streak) của người chơi.
* Thời gian (s): Tổng thời gian người chơi hoàn thành bài, tính bằng giây. (Thời gian được đo bằng mili-giây trong hệ thống và hiển thị cho người dùng theo đơn vị giây (s) với hai chữ số thập phân).
* Phía dưới có nút chức năng:
* “Tải lại”: Làm mới dữ liệu, hiển thị lại danh sách người chơi mới nhất được lấy từ SQL Server.

## Chế độ hiển thị sáng và tối (Light / Dark Mode)

Một điểm đặc biệt trong giao diện của Math Quiz là khả năng chuyển đổi linh hoạt giữa hai chế độ hiển thị — sáng (Light Mode) và tối (Dark Mode).

Tính năng này được thêm vào nhằm nâng cao trải nghiệm người dùng, giúp họ có thể luyện tập toán học trong nhiều điều kiện ánh sáng khác nhau, đồng thời tạo cảm giác thân thiện và hiện đại cho ứng dụng.

Khi thay đổi chế độ sáng tối thì tất cả các trang giao diện đều thay đổi.

* *Chế độ sáng (Light Mode)*
* Giao diện sử dụng tông màu sáng chủ đạo, nền trắng hoặc xám nhạt, giúp tạo cảm giác thoáng đãng và dễ nhìn.
* Phù hợp với môi trường lớp học, hoặc khi người dùng chơi game vào ban ngày.
* Các thành phần như nút *“Trả lời”* và *“Tiếp theo”* có màu xanh lam dịu, nổi bật trên nền sáng, giúp dễ thao tác và quan sát.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 22: Chế độ tối

* *Chế độ tối (Dark Mode)*
* Giao diện sử dụng nền xám đậm hoặc đen, kết hợp chữ trắng giúp dễ đọc và giảm chói sáng.
* Rất hữu ích khi người dùng học hoặc luyện tập vào buổi tối, giúp giảm mỏi mắt và tăng khả năng tập trung.
* Các nút chức năng được giữ nguyên màu nhưng tinh chỉnh độ sáng, đảm bảo vẫn nổi bật trên nền tối.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 23: Chế độ tối

## Các đoạn mã minh họa

* *MathQuizFrame.java:* Áp dụng Strategy Pattern cho câu hỏi: Minh họa rõ mẫu thiết kế Strategy, cho phép thay đổi cách sinh câu hỏi mà không cần sửa mã chính.

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 23: Code minh họa

* *MainUI.java: Hiển thị màn hình khởi động (Splash Screen):*Thể hiện khả năng tạo trải nghiệm người dùng (UI/UX) chuyên nghiệp, có logo và thanh tiến trình.

A screenshot of a computer code

AI-generated content may be incorrect.

Hình 24: Code minh họa

* *FixedScoreStrategy.java:* Chiến lược tính điểm cơ bản: Minh họa cách tách logic tính điểm ra khỏi phần giao diện, thể hiện tính đóng gói (Encapsulation).

A screen shot of a computer code

AI-generated content may be incorrect.

Hình 25: Code minh họa

* *TimedBonusDecorator.java:* Decorator Pattern: Minh họa cách thêm tính năng thưởng điểm mà không làm thay đổi lớp gốc.

A screen shot of a computer code

AI-generated content may be incorrect.

Hình 26: Code minh họa

* *StreakBasedDifficulty.java* – Chiến lược độ khó (Difficulty Strategy): Thể hiện tính tự động điều chỉnh độ khó, giúp trò chơi thông minh hơn.

A screenshot of a computer code

AI-generated content may be incorrect.

Hình 27: Code minh họa

## Cơ sở dữ liệu SQL

Để đảm bảo dữ liệu của trò chơi được lưu trữ an toàn, có khả năng mở rộng và truy xuất nhanh, ứng dụng Math Quiz sử dụng SQL Server làm hệ quản trị cơ sở dữ liệu.

Hệ thống cơ sở dữ liệu được thiết kế theo mô hình quan hệ (Relational Model), có ràng buộc khóa chính – khóa ngoại rõ ràng, giúp đảm bảo tính toàn vẹn dữ liệu và dễ dàng thực hiện các truy vấn thống kê, cập nhật hoặc phân tích kết quả người chơi.

### Cấu trúc cơ sở dữ liệu

Cơ sở dữ liệu gồm 4 bảng chính, được thiết kế trong SQLQuery1.sql như sau:

* Bảng quiz.Players
* Lưu thông tin người chơi.
* Mỗi người chơi có mã định danh (PlayerId), tên người chơi (PlayerName) và thời điểm tạo tài khoản (CreatedAt).
* Được sử dụng để liên kết với bảng GameSession.
* Bảng quiz.GameSession
* Ghi lại thông tin của từng phiên chơi (Session).
* Bao gồm PlayerId, tổng điểm đạt được, tổng thời gian chơi, chuỗi đúng cao nhất (MaxStreak), và thời gian bắt đầu – kết thúc phiên.
* Liên kết 1-nhiều với bảng Players.
* Bảng quiz.Question
* Lưu nội dung của các câu hỏi đã được sinh ra trong quá trình chơi.
* Gồm nội dung câu hỏi (Prompt), đáp án đúng (Answer), và loại chiến lược (StrategyCode) dùng để tạo câu hỏi.
* Hỗ trợ cho việc phân tích loại câu hỏi và thống kê độ khó.
* Bảng quiz.Attempt
* Lưu từng lần trả lời của người chơi trong mỗi phiên.
* Gồm mã phiên (SessionId), mã câu hỏi (QuestionId), thời gian trả lời, độ khó (Level), điểm đạt được, và trạng thái đúng/sai (IsCorrect).
* Hỗ trợ tính điểm và cập nhật chuỗi đúng.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 24: Sơ đồ diagram

Ngoài ra, hệ thống còn có hai view hỗ trợ:

* v\_Leaderboard\_AllTime: Tổng hợp bảng xếp hạng người chơi theo điểm cao nhất.
* v\_PlayerStats: Thống kê lượt chơi, điểm cao nhất và chuỗi đúng tốt nhất cho từng người chơi.

Để thao tác với dữ liệu nhanh và an toàn, hệ thống sử dụng các stored procedure như:

* sp\_UpsertPlayer – kiểm tra hoặc thêm mới người chơi.
* sp\_BeginSession – khởi tạo phiên chơi mới.
* sp\_RecordAttempt – ghi lại kết quả từng câu hỏi và cập nhật điểm.
* sp\_EndSession – kết thúc phiên chơi, cập nhật thời gian hoàn thành.

Việc sử dụng **SQL Server** giúp hệ thống lưu trữ dữ liệu tập trung, đáng tin cậy, truy xuất nhanh, dễ mở rộng và dễ kết nối với các công cụ phân tích như Power BI hoặc Excel.

### Kết nối SQL

* DatabaseConnection.java: Mở kết nối, đóng kết nối, xử lý lỗi.

A computer code with many colorful text

AI-generated content may be incorrect.

Hình 25: Kết nối

* config.properties: file cấu hình kết nối SQL Server (ví dụ user, password, database).

A screenshot of a computer

AI-generated content may be incorrect.

Hình 26: Cấu hình

Trong đó:

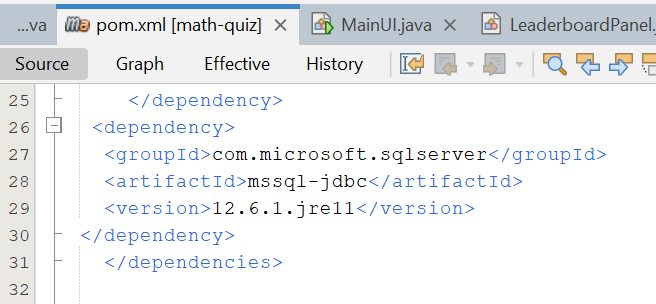
* **db.url=...** → Đường dẫn kết nối đến cơ sở dữ liệu SQL Server.

TRUC\\SQLEXPRESS: tên máy và tên instance SQL Server.

databaseName=MathQuiz: tên database cần kết nối.

encrypt=true;trustServerCertificate=true: bật mã hóa và cho phép chứng chỉ cục bộ (để tránh lỗi bảo mật khi chạy local)

* **db.user=sa** → Tên tài khoản đăng nhập SQL Server (mặc định là **sa** – System Administrator).
* **db.password=123** → Mật khẩu của tài khoản **sa** để truy cập cơ sở dữ liệu.
* Ở file pom.xml sẽ thêm 1 đoạn mã. Đoạn mã này giúp Maven tự động tải và thêm thư viện kết nối SQL Server (mssql-jdbc) vào dự án.



Hình 27: Đoạn mã

* Khi kết nối thành công với nhau thì giao diện sẽ hiện thị lên thông báo *“SQL Server connected”* và bắt đầu chơi được.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 28: Thông báo kết nối

# Hướng phát triển và kết luận

## Hướng phát triển

Đề tài *“Math Quiz – Ứng dụng luyện tập Toán học sử dụng mẫu thiết kế Strategy trong Java”* đã hoàn thành tốt các mục tiêu ban đầu đặt ra, từ việc áp dụng các mẫu thiết kế hướng đối tượng đến xây dựng giao diện người dùng trực quan và dễ sử dụng.

Tuy nhiên, để nâng cao trải nghiệm và mở rộng tính ứng dụng trong thực tế, ứng dụng có thể được phát triển thêm ở các hướng sau:

* **Tích hợp thêm nhiều dạng bài và chế độ chơi**
* Bổ sung các dạng câu hỏi khác như tìm x, so sánh biểu thức, hoặc toán đố tư duy.
* Thêm chế độ “Thử thách thời gian” (Time Challenge), “Học theo cấp độ” (Level Mode) để tăng sự hấp dẫn.
* **Thêm âm thanh, hiệu ứng và hình ảnh minh họa**
* Khi người chơi trả lời đúng/sai hoặc lên cấp, có thể phát âm thanh hoặc hiển thị hiệu ứng động.
* Giúp trò chơi sinh động hơn, tạo cảm giác gần gũi như một trò chơi giáo dục thực thụ.
* **Hỗ trợ đa ngôn ngữ (Multi-language)**
* Ngoài tiếng Việt, có thể thêm tiếng Anh để người học vừa luyện Toán vừa trau dồi ngoại ngữ.
* Điều này cũng mở rộng khả năng sử dụng của ứng dụng cho nhiều đối tượng học sinh hơn.

## Kết luận

Đề tài *“Math Quiz”* là một ví dụ điển hình về việc áp dụng các mẫu thiết kế hướng đối tượng trong phát triển phần mềm thực tế.

Thông qua dự án này, sinh viên không chỉ nắm vững kiến thức lý thuyết về **OOP**, Strategy, Decorator và Composite Pattern, mà còn biết cách vận dụng chúng vào một ứng dụng có tính thực tiễn cao. Đồng thời kết hợp **SQL Server** để quản lý dữ liệu người chơi.

Kết quả đạt được bao gồm:

* Ứng dụng hoạt động ổn định, giao diện thân thiện, dễ sử dụng.
* Có thể sinh câu hỏi ngẫu nhiên, tính điểm linh hoạt và lưu bảng xếp hạng.
* Thể hiện rõ tính mở rộng và tái sử dụng trong thiết kế phần mềm hiện đại.

Quá trình thực hiện đề tài giúp sinh viên:

* Hiểu rõ mối liên hệ giữa phân tích – thiết kế – cài đặt trong vòng đời phát triển phần mềm.
* Củng cố kiến thức về lập trình hướng đối tượng và mẫu thiết kế phần mềm.
* Nâng cao khả năng tư duy logic, kỹ năng tổ chức mã nguồn, và kỹ năng viết báo cáo kỹ thuật.
* Rèn luyện thái độ làm việc nghiêm túc, cẩn thận và sáng tạo trong việc giải quyết vấn đề.

# Tài liệu tham khảo

[1] Nguyễn Hữu Minh (2020), *Kết nối Java với Microsoft SQL Server bằng JDBC*, CodeJava.net, truy cập tại: [https://www.codejava.net/java-se/jdbc/connect-to-microsoft-sql-server-via-jdbc](https://www.codejava.net/java-se/jdbc/connect-to-microsoft-sql-server-via-jdbc?utm_source=chatgpt.com).

[2] Microsoft (2024), *Kết nối cơ sở dữ liệu SQL bằng Java – Proof of Concept*, Microsoft Docs, truy cập tại: <https://learn.microsoft.com/en-us/sql/connect/jdbc/>.

[3] Oracle (n.d.), *Xử lý câu lệnh SQL với JDBC – The Java Tutorials*, Oracle Documentation, truy cập tại: <https://docs.oracle.com/javase/tutorial/jdbc/basics/>.

[4] Refactoring.Guru (n.d.), *Mẫu thiết kế Strategy trong Java*, truy cập tại: [https://refactoring.guru/design-patterns/strategy/java/example](https://refactoring.guru/design-patterns/strategy/java/example?utm_source=chatgpt.com).

[5] GeeksforGeeks (n.d.), *Strategy Design Pattern trong Java – Lý thuyết và ví dụ*, truy cập tại: [https://www.geeksforgeeks.org/system-design/strategy-pattern-set-1/](https://www.geeksforgeeks.org/system-design/strategy-pattern-set-1/?utm_source=chatgpt.com).

[6] Baeldung (n.d.), *Hướng dẫn sử dụng Strategy Pattern trong Java*, truy cập tại: [https://www.baeldung.com/java-strategy-pattern](https://www.baeldung.com/java-strategy-pattern?utm_source=chatgpt.com).

[7] DigitalOcean (2022), *Ví dụ mẫu thiết kế Strategy trong Java*, truy cập tại: [https://www.digitalocean.com/community/tutorials/strategy-design-pattern-in-java-example-tutorial](https://www.digitalocean.com/community/tutorials/strategy-design-pattern-in-java-example-tutorial?utm_source=chatgpt.com).

[8] JournalDev (n.d.), *Decorator Pattern trong Java với ví dụ minh họa*, truy cập tại: https://www.journaldev.com/1540/decorator-design-pattern-in-java-example.

[9] SourceMaking (n.d.), *Composite Pattern – Cấu trúc và ứng dụng trong thiết kế phần mềm*, truy cập tại: https://sourcemaking.com/design\_patterns/composite.

[10] Head First Design Patterns (2004), *Design Patterns Explained in Java*, O’Reilly Media.