TRƯỜNG ĐẠI HỌC TRÀ VINH

TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ

ISO 9001:2015

XÂY DỰNG HỆ THỐNG LUYỆN THI TRỰC TUYẾN TIẾNG ANH KỸ NĂNG READING VỚI CƠ SỞ DỮ LIỆU NOSQL

ĐỒ ÁN TỐT NGHIỆP

NGÀNH CÔNG NGHỆ THÔNG TIN

Giảng viên hướng dẫn: **ThS. PHAN THỊ PHƯƠNG NAM**
Sinh viên thực hiện: **TRẦM NGỌC MAI**
Mã số sinh viên: **110121062**
Lớp: **DA21TTA**
Khoá: **2021**

Vĩnh Long, tháng 8 năm 2025

TRƯỜNG ĐẠI HỌC TRÀ VINH
TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ

ISO 9001:2015

XÂY DỰNG HỆ THỐNG LUYỆN THI TRỰC TUYẾN TIẾNG ANH KỸ NĂNG READING VỚI CƠ SỞ DỮ LIỆU NOSQL

ĐỒ ÁN TỐT NGHIỆP

NGÀNH CÔNG NGHỆ THÔNG TIN

Giảng viên hướng dẫn: **ThS. PHAN THỊ PHƯƠNG NAM**
Sinh viên thực hiện: **TRẦM NGỌC MAI**
Mã số sinh viên: **110121062**
Lớp: **DA21TTA**
Khoá: **2021**

Vĩnh Long, tháng 8 năm 2025

LỜI MỞ ĐẦU

Trong bối cảnh hội nhập toàn cầu diễn ra mạnh mẽ, tiếng Anh đã trở thành một công cụ thiết yếu, đóng vai trò cầu nối quan trọng trong giao tiếp, học tập và phát triển sự nghiệp. Trong số bốn kỹ năng ngôn ngữ cơ bản, kỹ năng đọc hiểu (Reading) được xem là nền tảng quan trọng, không chỉ giúp người học tiếp cận và xử lý thông tin hiệu quả mà còn góp phần nâng cao tư duy phản biện và mở rộng kiến thức chuyên môn.

Cùng với sự phát triển mạnh mẽ của công nghệ thông tin, đặc biệt là các hệ thống luyện thi trực tuyến và cơ sở dữ liệu NoSQL với hệ quản trị MongoDB, việc xây dựng các nền tảng học tập thông minh, dễ mở rộng và phù hợp với nhu cầu cá nhân hóa đã trở nên thuận lợi hơn bao giờ hết.

Từ những lý do đó, tôi đã chọn đề tài "Xây dựng hệ thống luyện thi trực tuyến tiếng Anh kỹ năng Reading với cơ sở dữ liệu NoSQL". Đề tài này được thực hiện với mong muốn góp phần tạo ra một công cụ thực hành hiệu quả, giúp người học có thể tự đánh giá năng lực của mình. Đồng thời, đây cũng là cơ hội để tôi vận dụng các kiến thức đã học để xây dựng một hệ thống thực tiễn, có khả năng triển khai và ứng dụng rộng rãi.

LỜI CẢM ƠN

Trước hết, tôi xin bày tỏ lòng biết ơn chân thành và sâu sắc đến cô Phan Thị Phương Nam, giảng viên hướng dẫn đã tận tình chỉ bảo, định hướng và hỗ trợ tôi trong suốt quá trình thực hiện khóa luận tốt nghiệp. Những góp ý quý báu của cô đã giúp tôi hoàn thiện đề tài một cách tốt nhất cả về nội dung lẫn hình thức.

Tôi xin gửi lời cảm ơn đến Quý Thầy, Cô trong Trường Kỹ thuật và Công nghệ - Trường Đại học Trà Vinh đã truyền đạt cho tôi kiến thức và kỹ năng chuyên môn trong suốt thời gian học tập tại trường. Những bài giảng, bài thực hành và định hướng nghề nghiệp từ các Thầy, Cô là nền tảng vững chắc giúp tôi hoàn thành khóa luận này.

Mặc dù đã cố gắng hết sức nhưng do giới hạn về thời gian, năng lực còn hạn chế nên bài khóa luận không thể tránh khỏi những thiếu sót. Tôi rất mong nhận được sự thông cảm và những góp ý chân thành từ Quý Thầy, Cô và Hội đồng chấm khóa luận để tôi có thể hoàn thiện hơn trong các công trình nghiên cứu sau này.

Vĩnh Long, ngày… tháng .... năm 2025

Sinh viên thực hiện

Trầm Ngọc Mai

NHẬN XÉT

(Của giảng viên hướng dẫn trong đồ án, khoá luận của sinh viên)

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

Giảng viên hướng dẫn

(ký và ghi rõ họ tên)

UBND TỈNH VĨNH LONG
TRƯỜNG ĐẠI HỌC TRÀ VINH

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh Phúc

BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP

(Của giảng viên hướng dẫn)

Họ và tên sinh viên: Trầm Ngọc Mai MSSV: 110121062

Ngành: Công nghệ thông tin Khóa: 2021

Tên đề tài: Xây dựng hệ thống luyện thi trực tuyến tiếng anh kỹ năng reading với cơ sở dữ liệu NoSQL.

Họ và tên Giáo viên hướng dẫn: Phan Thị Phương Nam

Chức danh: Giảng viên chính ..... Học vị: Thạc sĩ

NHẬN XÉT

1. Nội dung đề tài:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

2. Ưu điểm:

.....
.....
.....
.....

3. Khuyết điểm:

.....
.....
.....
.....
.....

4. Điểm mới đề tài:

.....
.....

.....
.....
.....

5. Giá trị thực trên đề tài:

.....
.....
.....
.....
.....
.....
.....

7. Đề nghị sửa chữa bổ sung:

.....
.....
.....
.....
.....
.....
.....

8. Đánh giá:

.....
.....
.....
.....

Vĩnh Long*, ngày tháng năm 2025*
Giảng viên hướng dẫn
(Ký & ghi rõ họ tên)

NHẬN XÉT
(Của giảng viên chấm trong đồ án, khoá luận của sinh viên)

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

Giảng viên chấm

(ký và ghi rõ họ tên)

UBND TỈNH VĨNH LONG **TRƯỜNG ĐẠI HỌC TRÀ VINH**	**CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM** **Độc lập - Tự do - Hạnh phúc**

BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP

(Của cán bộ chấm đồ án, khóa luận)

Họ và tên người nhận xét: .....

Chức danh: ..... Học vị: .....

Chuyên ngành: .....

Cơ quan công tác: .....

Tên sinh viên: .....

Tên đề tài đồ án, khóa luận tốt nghiệp: .....

.....

.....

I. Ý KIẾN NHẬN XÉT

1. Nội dung:

.....

.....

.....

.....

.....

.....

.....

.....

.....

2. Điểm mới các kết quả của đồ án, khóa luận:

.....

.....

.....

3. Ứng dụng thực tế:

.....

.....

.....

.....

.....

II. CÁC VẤN ĐỀ CẦN LÀM RÕ

(Các câu hỏi của giáo viên phản biện)

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

III. KẾT LUẬN

(Ghi rõ đồng ý hay không đồng ý cho bảo vệ đồ án khóa luận tốt nghiệp)

.....
.....
.....
.....
.....

……, ngày …… tháng …… năm 20…

Người nhận xét

(Ký & ghi rõ họ tên)

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1
1.1 Lý do chọn đề tài 1
1.2 Mục tiêu nghiên cứu 1
1.3 Phạm vi nghiên cứu 2
1.4 Phương pháp nghiên cứu 2
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ TỔNG QUAN TÀI LIỆU 3
2.1 Tổng quan các dạng bài đọc hiểu (Reading) 3
2.1.1 Cấu trúc bài thi IELTS Reading 3
2.1.2 Các dạng bài thường gặp 4
2.1.2.1 Hoàn thành dán nhãn biểu đồ (Diagram label completion) 4
2.1.2.2 Câu hỏi xác định thông tin (Information identification questions) 5
2.1.2.3 Nối nội dung (Matching features) 6
2.1.2.4 Nối tiêu đề (Matching headings) 7
2.1.2.5 Nối thông tin (Matching information) 9
2.1.2.6 Kết hợp các vế câu (Matching sentence endings) 9
2.1.2.7 Câu hỏi trắc nghiệm (Multiple choice) 10
2.1.2.8 Hoàn thành câu (Sentence completion) 11
2.1.2.9 Dạng câu hỏi với câu trả lời ngắn (Short-answer questions) 12
2.1.3 Tiêu chí chấm điểm 13
2.2 Tổng quan về cơ sở dữ liệu NoSQL 16
2.2.1 Cơ sở dữ liệu NoSQL 16
2.2.2 Ưu điểm của cơ sở dữ liệu NoSQL 16
2.2.3 Các trường hợp sử dụng của cơ sở dữ liệu NoSQL 17
2.2.4 Cách thức hoạt động của cơ sở dữ liệu NoSQL 18
2.2.5 Cơ sở dữ liệu NoSQL gồm những loại 19
2.2.6 Sự khác biệt giữa cơ sở dữ liệu NoSQL và SQL 21
2.2.7 Khi nào nên chọn cơ sở dữ liệu NoSQL thay vì cơ sở dữ liệu SQL 24
2.3 Nghiên cứu cách thiết kế cơ sở dữ liệu với mô hình dữ liệu trong cơ sở dữ liệu NoSQL 25
2.3.1 Thiết kế cơ sở dữ liệu NoSQL 25
2.3.2 Điểm mạnh của thiết kế cơ sở dữ liệu NoSQL 25
2.3.3 Các quy tắc khi thiết kế cơ sở dữ liệu NoSQL 26
2.3.4 Những mẫu thiết kế cơ sở dữ liệu NoSQL thông dụng nhất 26
2.3.5 Các bước thiết kế schema cho cơ sở dữ liệu NoSQL 28
2.3.6 Dữ liệu được lưu trữ trong NoSQL 30
2.4 Các công nghệ thiết kế, cài đặt hệ thống. 31
2.4.1 Nghiên cứu RESTfull API 31
2.4.2 Các thành phần của RESTfull API 31
2.4.3 Cách thức RESTful hoạt động 32
2.4.4 NodeJs 32
2.4.5 Những ứng dụng nên viết bằng Nodejs 33
2.4.6 Lý do để tự tin sử dụng Nodejs 33
2.4.7 Kiến thức về framework ExpressJs 34
2.4.8 ExpressJs được dùng cho các công việc 34
2.4.9 Tính năng và lợi ích của ExpressJs 35
2.4.10 Tailwind CSS 36
2.4.11 Ưu điểm của Tailwind Css 36

2.4.12 Nhược điểm của Tailwind Css 37
2.4.13 Hệ quản trị cơ sở dữ liệu MonggoDB 37
2.5 Tổng quan về các hệ thống chấm điểm tự động hiện nay.....38
2.5.1 Lý do nên chọn hệ thống chấm điểm tự động 38
CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU 41
3.1. Kiến trúc tổng thể 41
3.1.1 Mô hình Kiến trúc MVC (Model-View-Controller) 41
3.1.2. Công nghệ sử dụng và Lý do lựa chọn.....42
3.2. Các mô-đun chức năng 43
3.2.1. Mô-đun Xác thực (Authentication Module).....44
3.2.2. Mô-đun Quản trị (Admin Module).....45
3.2.3. Mô-đun Học viên (User Module).....46
3.2.4. Mô-đun Xử lý Bài Test (Test Engine).....47
3.3. Thiết kế Cơ sở dữ liệu và Giao diện.....48
3.3.1. Thiết kế Cơ sở dữ liệu 48
3.3.2. Thiết kế Sơ đồ cấu trúc Giao diện Người dùng.....55
CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU 82
4.1. Cài đặt công nghệ 82
4.1.1. Môi trường phát triển 82
4.1.1.1. Yêu cầu hệ thống.....82
4.1.1.2. Cấu trúc công nghệ Backend.....82
4.1.1.3. Cấu trúc công nghệ Frontend 83
4.1.2. Cài đặt và triển khai.....84
4.1.2.1. Quy trình cài đặt 84
4.1.2.2. Cấu hình cơ sở dữ liệu.....84
4.1.2.3. Cấu hình session và authentication.....85
4.2. Kết nối Frontend - Backend 86
4.2.1. Kiến trúc MVC 86
4.2.2. API và Cấu hình Route (Route Configuration) 87
4.2.2.1. Cấu trúc API RESTful (Ví dụ) 87
4.2.2.2. Lưu trữ câu trả lời thời gian thực (Real-time Answer Saving) 87
4.2.3. Template Engine và Render Giao diện (View Rendering).....87
4.2.3.1. Cấu hình EJS Template Engine.....88
4.2.3.2. Render Nội dung Động (Dynamic Content Rendering).....88
4.2.3.3 Giao diện và chức năng của hệ thống.....88
4.2.4. Tích hợp JavaScript phía Frontend.....115
4.2.4.1. Chức năng Tự động Lưu (Auto-save Functionality).....115
4.2.4.2. Triển khai Đồng hồ Đếm ngược (Timer Implementation).....115
4.3. Kiểm thử hệ thống 116
4.3.1. Phương pháp kiểm thử được áp dụng dựa trên phân tích mã nguồn.....116
4.3.1.1. Kiểm thử Xác thực Dữ liệu Đầu vào (Input Validation Testing).....116
4.3.1.2. Kiểm thử Xác thực & Bảo mật (Authentication & Security Testing).....116
4.3.1.3. Kiểm thử Xử lý Lỗi (Error Handling Testing) 116
4.3.1.4. Kiểm thử Tính Toàn vẹn Dữ liệu (Data Integrity Testing) 116
4.3.2. Kết quả Kiểm thử Chức năng (Dựa trên bằng chứng mã nguồn).....117
4.3.3. Kiểm thử Hiệu suất (Performance Testing) - Khía cạnh Tối ưu hóa từ Mã nguồn 117
4.3.4. Kiểm thử Bảo mật được triển khai 117

4.4. So sánh với các hệ thống tương tự 117
4.4.1. Phân tích hệ thống cạnh tranh 118
4.4.2. Định vị sản phẩm.....118
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 119
5.1. Tổng kết kết quả đạt được 119
5.2. Hạn chế và đề xuất cải tiến trong tương lai 120
5.2.1. Hạn chế của hệ thống hiện tại.....120
5.2.2. Đề xuất cải tiến và hướng phát triển trong tương lai.....121
DANH MỤC TÀI LIỆU THAM KHẢO 123

MỤC LỤC HÌNH ẢNH

Hình 2.1 Ví dụ cho dạng bài dán nhãn biểu đồ 5
Hình 2.2 Ví dụ cho dạng bài xác định thông tin 6
Hình 2.3 Ví dụ cho dạng bài nối nội dung … 7
Hình 2.4 Ví dụ minh họa dạng bài nối tiêu đề … 8
Hình 2.5 Ví dụ cho dạng bài nối thông tin … 9
Hình 2.6 Ví dụ dạng bài kết hợp các vế câu …....10
Hình 2.7 Ví dụ dạng bài trắc nghiệm …. 11
Hình 2.8 Ví dụ dạng câu hỏi hoàn thành câu … 12
Hình 2.9 Mẫu câu hỏi cho dạng bài yêu cầu câu trả lời ngắn 13
Hình 2.10 Cách Container chứa cơ sở dữ liệu NoSQL 29
Hình 2.11 Dùng tham chiếu (Referencing) để giảm bớt sự chuẩn hóa dữ liệu 30
Hình 2.12 RESTfull API.....31
Hình 2.13 NodeJS.....32
Hình 2.14 Tính năng của ExpressJs 35
Hình 2.15 Tailwind CSS.....36
Hình 2.16 MogoDB 37
Hình 3.1 Mô hình cơ sơ dữ liệu.....49
Hình 3.2 Sơ đồ cấu trúc trang chủ 57
Hình 3.3 Sơ đồ cấu trúc trang đăng nhập 58
Hình 3.4 Sơ đồ cấu trúc trang đăng ký 59
Hình 3.5 Sơ đồ cấu trúc trang dashboard admin 61
Hình 3.6 Sơ đồ cấu trúc trang danh sách bài đọc 62
Hình 3.7 Sơ đồ cấu trúc trang tạo bài đọc 63
Hình 3.8 Sơ đồ cấu trúc sửa bài đọc.....64
Hình 3.9 Sơ đồ cấu trúc danh sách câu hỏi 66
Hình 3.10 Sơ đồ cấu trúc tạo câu hỏi 67
Hình 3.11 Sơ đồ cấu trúc sửa câu hỏi.....68
Hình 3.12 Sơ đồ cấu trúc trang danh sách bài kiểm tra.....70
Hình 3.13 Sơ đồ cấu trúc tạo bài kiểm tra 72
Hình 3.14 Sơ đồ cấu trúc sửa bài kiểm tra 73
Hình 3.15 Sơ đồ cấu trúc dashboard của người dùng.....75
Hình 3.16 Sơ đồ cấu trúc danh sách bài kiểm tra người dùng 76
Hình 3.17 Sơ đồ cấu trúc cài đặt bài kiểm tra 77
Hình 3.18 Sơ đồ cấu trúc trang luyện tập.....78
Hình 3.19 Sơ đồ cấu trúc kết quả bài kiểm tra 80
Hình 3.20 Sơ đồ cấu trúc trang lịch sử 81
Hình 4.1 Giao diện trang chủ……90
Hình 4.2 Giao diện trang đăng nhập.....91
Hình 4.3 Giao diện trang đăng ký 92
Hình 4.4 Giao diện dashboard của admin 94
Hình 4.5 Giao diện quản lý bài đọc.....95
Hình 4.6 Giao diện tạo bài đọc.....96
Hình 4.7 Giao diện sửa bài đọc 97
Hình 4.8 Giao diện danh sách câu hỏi.....99
Hình 4.9 Giao diện tạo câu hỏi.....100
Hình 4.10 Giao diện chỉnh sửa câu hỏi 102
Hình 4.11 Giao diện quản lý bài kiểm tra 103

Hình 4.12 Giao diện tạo bài kiểm tra 105
Hình 4.13 Giao diện chỉnh sửa bài kiểm tra.....107
Hình 4.14 Giao diện dashboard của người dùng.....109
Hình 4.15 Giao diện danh sách bài kiểm tra của người dùng 110
Hình 4.16 Giao diện cài đặt bài kiểm tra.....111
Hình 4.17 Giao diện chính thực hiện bài kiểm tra 112
Hình 4.18 Giao diện kết quả bài kiểm tra.....114
Hình 4.19 Giao diện lịch sửa bài kiểm tra.....115

MỤC LỤC BẢNG

Bảng 2.1 Thang điểm IELTS Reading Academic 13
Bảng 2.2 Thang điểm IELTS Reading General Training.....14
Bảng 2.4 So sánh sự khác biệt giữa cơ sở dữ liệu NoSQL và SQL 22
Bảng 3.1 Bảng Collection users 50
Bảng 3.2 Bảng Collection readingpassages.....51
Bảng 3.3 Bảng Collection questions 51
Bảng 3.5 Bảng Collection userattempts 54

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Ý nghĩa (Tiếng Anh)	Ý nghĩa (Tiếng Việt)
ACID	Atomicity, Consistency, Isolation, Durability	Tính nguyên tố, Nhất quán, Tách biệt, Bền vững
AI	Artificial Intelligence	Trí tuệ nhân tạo
API	Application Programming Interface	Giao diện lập trình ứng dụng
CRUD	Create, Read, Update, Delete	Tạo, đọc, cập nhật, xóa
CSS	Cascading Style Sheets	N/A
CTA	Call to Action	N/A
DAX	DynamoDB Accelerator	N/A
EJS	Embedded JavaScript	N/A
ERD	Entity Relationship Diagram	Sơ đồ Quan hệ Thực thể
I/O	Input/Output	N/A
IELTS	International English Language Testing System	N/A
IoT	Internet of Things	Internet vạn vật
JSON	JavaScript Object Notation	N/A
JWT	JSON Web Tokens	N/A
LMS	Learning Management System	Hệ thống quản lý học tập
ML	Machine Learning	Học máy
MVC	Model-View-Controller	N/A
NPM	Node Package Manager	Trình quản lý gói Node
ODM	Object Data Modeling	N/A
OLAP	Online Analytical Processing	Xử lý phân tích trực tuyến
OLTP	Online Transaction Processing	Xử lý giao dịch trực tuyến
OMR	Optical Mark Recognition	Nhận dạng điểm quang học
PWA	Progressive Web App	N/A
RDBMS	Relational Database Management System	Hệ thống quản lý cơ sở dữ liệu quan hệ

Từ viết tắt	Ý nghĩa (Tiếng Anh)	Ý nghĩa (Tiếng Việt)
RDF	Resource Description Framework	Khung mô tả tài nguyên
SPA	Single Page Application	Ứng dụng một trang
SPARQL	N/A	Ngôn ngữ truy vấn cho RDF
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
SSPL	Server Side Public License	Giấy phép Công cộng phía Máy chủ
UI	User Interface	Giao diện người dùng
UX	User Experience	Trải nghiệm người dùng

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1 Lý do chọn đề tài

Trong bối cảnh toàn cầu hóa diễn ra ngày càng mạnh mẽ, khi các quốc gia và nền văn hóa đang xích lại gần nhau hơn bao giờ hết, ngoại ngữ đặc biệt là tiếng Anh đã trở thành một công cụ thiết yếu, đóng vai trò cầu nối quan trọng trong giao tiếp, học tập, nghiên cứu khoa học và phát triển sự nghiệp. Trong số bốn kỹ năng ngôn ngữ cơ bản là nghe, nói, đọc và viết, kỹ năng đọc hiểu (Reading) được xem là một nền tảng quan trọng, không chỉ giúp người học tiếp cận và xử lý thông tin một cách hiệu quả, mà còn góp phần nâng cao tư duy phản biện và mở rộng kiến thức chuyên môn trong nhiều lĩnh vực.

Bên cạnh đó, với sự phát triển mạnh mẽ của công nghệ thông tin, đặc biệt là các hệ thống luyện thi trực tuyến và cơ sở dữ liệu NoSQL với hệ quản trị cơ sở dữ liệu MongoDB vốn có khả năng mở rộng và linh hoạt trong việc lưu trữ dữ liệu phi cấu trúc đã tạo điều kiện thuận lợi để xây dựng các nền tảng học tập thông minh, dễ mở rộng và phù hợp với nhu cầu học tập cá nhân hóa.

Từ những lý do đó, tôi chọn đề tài: **"**Xây dựng hệ thống luyện thi trực tuyến tiếng Anh kỹ năng Reading với cơ sở dữ liệu NoSQL**"**, với mong muốn góp phần hỗ trợ người học có công cụ thực hành hiệu quả, đồng thời vận dụng các kiến thức đã học để xây dựng hệ thống thực tiễn, có khả năng triển khai và ứng dụng rộng rãi.

1.2 Mục tiêu nghiên cứu

Đề tài hướng đến việc xây dựng một hệ thống luyện thi trực tuyến kỹ năng Reading cho kỳ thi tiếng Anh với các mục tiêu cụ thể như sau:

Thiết kế hệ thống luyện tập các dạng câu hỏi reading thường gặp như:

- Multiple Choice, Matching headings, Fill in blank, True/False/Not Given và Short answer;
- Sử dụng cơ sở dữ liệu NoSQL với hệ quản trị cơ sở dữ liệu MongoDB nhằm đảm bảo khả năng mở rộng và dễ dàng quản lý câu hỏi, bài đọc và kết quả;
- Tự động chấm điểm và thống kê kết quả luyện tập nhằm hỗ trợ người học tự đánh giá năng lực;
- Phân quyền quản trị viên và người dùng một cách rõ ràng, để hệ thống vận hành an toàn, hiệu quả.

1.3 Phạm vi nghiên cứu

Đề tài tập trung vào việc xây dựng hệ thống luyện thi tiếng Anh, kỹ năng Reading trực tuyến với các chức năng chính như:

- Đăng ký, đăng nhập, phân quyền.
- Chức năng của quản trị viên như: Xem, thêm, xóa, sửa và tìm kiếm cho bài đọc, câu hỏi và bài kiểm tra.
- Chức năng cho người luyện thi như: Làm bài, chấm điểm tự động và thống kê kết quả.

Phần backend được xây dựng bằng Node.js và Express, frontend sử dụng EJS kết hợp với TailwindCSS. Cơ sở dữ liệu sử dụng MongoDB với các mô hình dữ liệu phù hợp.

Hệ thống được thiết kế để phục vụ luyện kỹ năng Reading cho các kỳ thi đánh giá năng lực tiếng Anh như IELTS Academic/General Training.

Hệ thống được xây dựng phục vụ mục đích nghiên cứu học thuật và thực hành công nghệ, chưa triển khai rộng rãi ở quy mô thực tế có lưu lượng truy cập lớn.

1.4 Phương pháp nghiên cứu

Để hoàn thành đề tài, tôi sử dụng các phương pháp nghiên cứu sau:

- **Nghiên cứu lý thuyết**: Tìm hiểu về các dạng bài đọc trong tiếng Anh, mô hình cơ sở dữ liệu NoSQL, kiến trúc hệ thống web và các công nghệ thiết kế – lập trình web hiện đại;
- **Phân tích và thiết kế hệ thống**: Phân tích chức năng và thiết kế kiến trúc tổng thể, cơ sở dữ liệu và giao diện người dùng;
- **Phương pháp thực nghiệm**: Cài đặt hệ thống dựa trên các công nghệ đã chọn, triển khai thử nghiệm, kiểm thử tính năng và đánh giá hiệu quả;
- **So sánh đối chiếu**: Đối chiếu hệ thống đã xây dựng với các nền tảng luyện thi trực tuyến hiện nay để rút ra điểm mạnh, điểm yếu và hướng phát triển trong tương lai.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ TỔNG QUAN TÀI LIỆU

2.1 Tổng quan các dạng bài đọc hiểu (Reading)

2.1.1 Cấu trúc bài thi IELTS Reading

Bài thi IELTS Reading là một trong bốn phần thi quan trọng trong kỳ thi IELTS (International English Language Testing System), được thiết kế để đánh giá toàn diện khả năng đọc hiểu của thí sinh trong môi trường sử dụng tiếng Anh học thuật hoặc đời sống hàng ngày. Thí sinh sẽ có tổng thời gian là 60 phút để hoàn thành phần thi này, bao gồm việc đọc hiểu ba đoạn văn (Passages) và trả lời tổng cộng 40 câu hỏi thuộc nhiều dạng khác nhau.

Thông qua phần thi Đọc, giám khảo sẽ đánh giá khả năng của thí sinh trong các kỹ năng đọc thiết yếu như: nắm bắt ý chính, đọc để ghi nhớ chi tiết, đọc lướt (skimming) để tìm thông tin nhanh chóng, đọc quét (scanning) để định vị thông tin cụ thể, cũng như khả năng hiểu lập luận logic, nhận biết quan điểm, thái độ và mục đích của tác giả. Những kỹ năng này đều phản ánh năng lực sử dụng tiếng Anh thực tế trong môi trường học thuật hoặc giao tiếp hàng ngày.

Bài thi IELTS Reading có hai hình thức tùy thuộc mục đích dự thi của thí sinh

- **IELTS Reading – Hình thức Academic (Học thuật)**

Đây là hình thức thi dành cho những thí sinh có nhu cầu du học bậc Đại học, Sau đại học, hoặc tham gia vào các chương trình học mang tính nghiên cứu chuyên sâu. Trong phần thi này, đề bài gồm 3 đoạn văn học thuật có độ dài trung bình khoảng 1500 từ mỗi đoạn, với độ khó tăng dần từ đoạn 1 đến đoạn 3. Các văn bản thường được trích từ các nguồn tài liệu chính thống như: sách chuyên khảo, tạp chí khoa học, bài viết học thuật, báo chí chuyên ngành, v.v.

Nội dung của các đoạn văn thường xoay quanh các chủ đề phổ biến trong đời sống học thuật như: khoa học, giáo dục, công nghệ, môi trường, xã hội học, tâm lý học, nhân văn... Mặc dù mang tính học thuật cao, nhưng các bài đọc này được viết cho đối tượng người đọc không chuyên, vì vậy không đòi hỏi kiến thức chuyên sâu về lĩnh vực được đề cập, mà chủ yếu kiểm tra khả năng đọc hiểu và phân tích văn bản học thuật.

- **IELTS Reading – Hình thức General Training (Tổng quát)**

Hình thức này được thiết kế dành cho những thí sinh có mục đích di cư, làm việc, hoặc học tập không thuộc bậc đại học tại các quốc gia sử dụng tiếng Anh là ngôn ngữ

chính. Nội dung các đoạn văn trong hình thức General thường thực tế hơn, gần gũi với đời sống hàng ngày.

Các văn bản được trích từ sách, tạp chí, báo, thông báo, quảng cáo, sổ tay công ty, hướng dẫn sử dụng, và các loại tài liệu có thể gặp thường xuyên trong môi trường giao tiếp tiếng Anh hàng ngày. Bài thi cũng bao gồm 3 phần, với các đoạn văn ngắn ở phần đầu (có thể là mẫu tin hoặc hướng dẫn), và một đoạn văn dài hơn, có mức độ khó tăng dần ở phần cuối. Mục tiêu chính của hình thức này là đánh giá khả năng đọc hiểu trong bối cảnh xã hội và môi trường làm việc thực tế.

- Các dạng câu hỏi thường gặp và chiến thuật làm bài

Cả hai hình thức IELTS Reading đều bao gồm nhiều dạng câu hỏi khác nhau, yêu cầu thí sinh phải vận dụng linh hoạt các kỹ năng đọc hiểu. Một số dạng câu hỏi phổ biến bao gồm:

- True/False/Not Given hoặc Yes/No/Not Given
- Matching headings (Nối tiêu đề với đoạn văn)
- Matching features (Nối thông tin với người, địa điểm, hoặc nhóm)
- Multiple choice questions
- Sentence completion (Hoàn thành câu)
- Summary/note/flow-chart/table completion
- Short answer questions

Để làm bài hiệu quả, thí sinh cần xây dựng các chiến thuật làm bài phù hợp như: xác định từ khóa, luyện kỹ năng scanning và skimming, quản lý thời gian hợp lý, tránh "bẫy từ vựng", và rèn luyện tư duy phản biện để hiểu quan điểm tác giả. Việc luyện tập thường xuyên, kết hợp giữa lý thuyết và thực hành, sẽ giúp nâng cao khả năng làm bài và đạt kết quả cao trong phần thi này.[1]

2.1.2 Các dạng bài thường gặp

2.1.2.1 Hoàn thành dán nhãn biểu đồ (Diagram label completion)

Yêu cầu của dạng bài: Hoàn thành dán nhãn trên biểu đồ dựa trên thông tin cung cấp trong đoạn văn.

Mô tả dạng bài: Thí sinh sẽ được cung cấp biểu đồ, đoạn văn tóm tắt để trống có các phần thông tin quan trọng. Nhiệm vụ của thí sinh là phải chọn đúng đáp án để điền vào chỗ trống cho phù hợp. [1]

Ví dụ dạng bài:

Questions 20 and 21

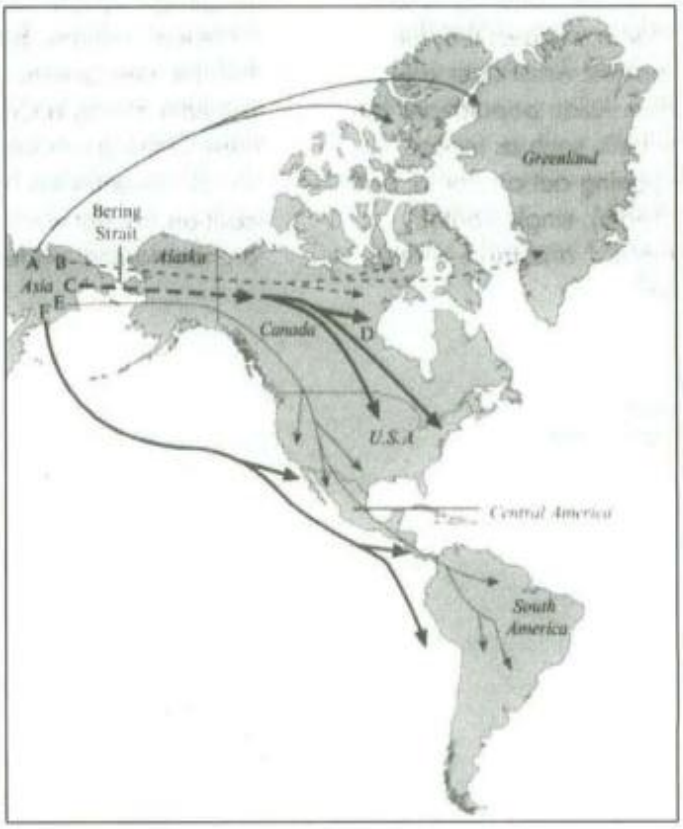
The discussion of Williams's research indicates the periods at which early people are thought to have migrated along certain routes. There are six routes, **A–F**, marked on the map below.

Complete the table below.

*Write the correct letter, **A–F**, in boxes 20 and 21 on your answer sheet.*

Route	Period (number of years ago)
20	15,000 or more
21	600 to 700

Early Population Movement to the Americas



Hình 2.1 Ví dụ cho dạng bài dán nhãn biểu đồ [11]

2.1.2.2 Câu hỏi xác định thông tin (Information identification questions)

Yêu cầu về dạng bài: Information identification questions là một trong các dạng bài IELTS Reading phổ biến. Dạng bài này yêu cầu thí sinh phải xác định xem

thông tin, ý kiến của người viết trong câu hỏi đúng/ sai/ thông tin có được cung cấp trong đoạn văn không để trả lời Yes/No/Not Given hoặc True/ False /Not given.

Dạng bài này đánh giá khả năng:

- Tổng hợp và xác định đúng thông tin trong đoạn văn
- Quét và hiểu nghĩa sâu xa của thông tin đó
- Hiểu quan điểm, nhận định được nêu của người viết

Mô tả dạng bài: Trong đề bài sẽ đưa ra khoảng 4-5 câu chứa các thông tin được cho trong bài đọc. Thí sinh sẽ đọc từng câu và xác định câu đó đúng hay là sai hoặc có xuất hiện trong bài hay không. [1]

Ví dụ dạng bài:

Test 2

Questions 1–4

Do the following statements agree with the claims of the writer in Reading Passage 1?

In boxes 1–4 on your answer sheet, write

YES	*if the statement agrees with the claims of the writer*
NO	*if the statement contradicts the claims of the writer*
NOT GIVEN	*if it is impossible to say what the writer thinks about this*

1 Only two Japanese pagodas have collapsed in 1400 years.

2 The Hanshin earthquake of 1995 destroyed the pagoda at the Toji temple.

3 The other buildings near the Toji pagoda had been built in the last 30 years.

4 The builders of pagodas knew how to absorb some of the power produced by severe weather conditions.

Hình 2.2 Ví dụ cho dạng bài xác định thông tin [11]

2.1.2.3 Nối nội dung (Matching features)

Yêu cầu của dạng bài: Tìm thông tin tương thích với đặc điểm của chủ thể trong danh sách. Sau đó, nối các câu nói hoặc thông tin với một danh sách các lựa chọn đã cho.

Dạng bài này đánh giá khả năng:

- Kỹ năng đọc lướt và đọc dò (Skimming and Scanning).
- Kỹ năng tìm và xác định được bố cục bài viết.
- Xác định sự liên hệ của các sự kiện trong văn bản.
- Có khả năng nhận biết lý thuyết và các ý kiến được nêu trong bài.

Yêu cầu của dạng bài: Tìm thông tin tương thích với đặc điểm của chủ thể trong danh sách. Mô tả dạng bài: Thí sinh sẽ được cung cấp câu hỏi bao gồm chủ thể là người, sự việc, sự vật hoặc địa điểm nổi bật trong bài. Nhiệm vụ của thí sinh là tìm thông tin phù hợp với từng chủ đề và nối chúng lại với nhau. [1]

Ví dụ dạng bài

Questions 34–39

Look at the following statements issued by the conference.

Which six of the following statements, **A–J**, refer to the resolutions that were issued?

Match the statements with the appropriate resolutions (Questions 34–39).

*Write the correct letter, **A–J**, in boxes 34–39 on your answer sheet.*

A	All kinds of species of trees should be preserved.
B	Fragile mountain forests should be given priority in research programs.
C	The surviving natural forests of Europe do not need priority treatment.
D	Research is to be better co-ordinated throughout Europe.
E	Information on forest fires should be collected and shared.
F	Loss of leaves from trees should be more extensively and carefully monitored.
G	Resources should be allocated to research into tree diseases.
H	Skiing should be encouraged in thinly populated areas.
I	Soil imbalances such as acidification should be treated with compounds of nitrogen and sulphur.
J	Information is to be systematically gathered on any decline in the condition of forests.

34 Resolution 1

35 Resolution 2

36 Resolution 3

37 Resolution 4

38 Resolution 5

39 Resolution 6

Hình 2.3 Ví dụ cho dạng bài nối nội dung [11]

2.1.2.4 Nối tiêu đề (Matching headings)

Yêu cầu: Dạng bài Matching headings yêu cầu thí sinh nối danh sách các tiêu đề và các đoạn hoặc phần chính xác

Dạng bài này đánh giá khả năng:

- Nhận biết ý tưởng/ chủ đề chính trong các đoạn/ phần văn bản
- Phân biệt sự khác nhau và xác định được ý chính với các ý bổ trợ.
- Yêu cầu của dạng bài: Thí sinh phải chọn ra một trong các tiêu đề được cung cấp và đặt vào đoạn văn thích hợp.

Mô tả dạng bài: Đề bài sẽ cung cấp các tiêu đề khác nhau, phù hợp với các đoạn văn có trong bài đọc. Thí sinh cần lựa chọn tiêu đề nào phù hợp với đoạn văn nào. [1]

Ví dụ dạng bài

Questions 14–19

Reading Passage 2 has seven sections, **A–G**.

*Choose the correct headings for sections **A–F** from the list of headings below.*

*Write the correct number, **i–x**, in boxes 14–19 on your answer sheet.*

List of Headings

i The results of the research into blood-variants
ii Dental evidence
iii Greenberg's analysis of the dental and linguistic evidence
iv Developments in the methods used to study early population movements
v Indian migration from Canada to the U.S.A.
vi Further genetic evidence relating to the three-wave theory
vii Long-standing questions about prehistoric migration to America
viii Conflicting views of the three-wave theory, based on non-genetic evidence
ix Questions about the causes of prehistoric migration to America
x How analysis of blood-variants measures the closeness of the relationship between different populations

14 Section **A**
15 Section **B**
16 Section **C**
17 Section **D**
18 Section **E**
19 Section **F**

Example	*Answer*
Section **G**	**viii**

Hình 2.4 Ví dụ minh họa dạng bài nối tiêu đề [11]

2.1.2.5 Nối thông tin (Matching information)

Yêu cầu: Tìm kiếm chính xác các thông tin trong các đoạn văn A,B,C,... Thông tin mà thí sinh có thể được yêu cầu tìm: chi tiết cụ thể, ví dụ, lý do, mô tả, so sánh, tóm tắt, giải thích.

Dạng bài này đánh giá khả năng: Đọc hiểu, định vị chi tiết, nhận biết một bản tóm tắt hoặc định nghĩa.

Yêu cầu của dạng bài: Thí sinh cần tìm thông tin cụ thể trong các đoạn văn A, B, C và nối chung với đáp án phù hợp.

Mô tả dạng bài: Đề bài yêu cầu thí sinh tìm kiếm thông tin về các chi tiết cụ thể được đặt để trong mỗi đoạn văn. Sau đó, thí sinh sẽ đặt các ký tự Alphabet tương ứng với mỗi đoạn vào đáp án thí sinh cho là phù hợp. [1]

Ví dụ dạng bài

Questions 37–40

The text has nine paragraphs, **A–I**.

Which paragraph contains the following information?

*Write the correct letter, **A–I**, in boxes 37–40 on your answer sheet.*

37 why a bridge was required across the River Severn

38 a method used to raise money for the bridge

39 why Coalbrookdale became attractive to iron makers

40 how the sections of the bridge were connected to each other

Hình 2.5 Ví dụ cho dạng bài nối thông tin [11]

2.1.2.6 Kết hợp các vế câu (Matching sentence endings)

Yêu cầu: Kết hợp nửa đầu của một câu dựa trên bài đọc hiểu với phương án tốt nhất để tạo ra một câu hoàn chỉnh.

Dạng bài này đánh giá khả năng:

- Hiểu các ý chính của thí sinh trong một câu.
- Kỹ năng Skimming, Scanning, xác định từ khóa.

Mô tả dạng bài: Thí sinh sẽ được cung cấp một danh sách hai vế câu chưa hoàn chỉnh. Nhiệm vụ của thí sinh là nối chúng lại thành một câu có nghĩa từ thông tin trong bài. [1]

(Ví dụ dạng bài kết hợp các vế câu ở trang sau)

Ví dụ dạng bài:

Questions 21–26

*Complete each sentence with the correct ending, **A–K**, below.*

*Write the correct letter, **A–K**, in boxes 21–26 on your answer sheet.*

21 In Alaska, biologists keep a check on adult fish

22 Biologists have the authority

23 In-Season Abundance-Based Management has allowed the Alaska salmon fisheries

24 The Marine Stewardship Council (MSC) was established

25 As a result of the collapse of the salmon runs in 1999, the state decided

26 In September 2000, the MSC allowed seven Alaska salmon companies

A to recognise fisheries that care for the environment.
B to be successful.
C to stop fish from spawning.
D to set up environmental protection laws.
E to stop people fishing for sport.
F to label their products using the MSC logo.
G to ensure that fish numbers are sufficient to permit fishing.
H to assist the subsistence communities in the region.
I to freeze a huge number of salmon eggs.
J to deny certification to the Alaska fisheries.
K to close down all fisheries.

Hình 2.6 Ví dụ dạng bài kết hợp các vế câu [11]

2.1.2.7 Câu hỏi trắc nghiệm (Multiple choice)

Yêu cầu: Dạng bài Multiple choice yêu cầu thí sinh chọn đáp án chính xác từ những lựa chọn được cho trước (A, B, C hoặc D).

Dạng bài này đánh giá khả năng:

- Khả năng quét thông tin, keyword
- Hiểu ý nghĩa sâu xa trong đoạn văn

Mô tả dạng bài: Đề bài sẽ đưa ra câu hỏi kèm theo một danh sách đáp án, thí sinh sẽ dựa trên thông tin trong bài đọc và chọn những đáp án chính xác. [1]

Ví dụ dạng bài:

Questions 27–29

*Choose the correct letter, **A**, **B**, **C** or **D**.*

Write the correct letter in boxes 27–29 on your answer sheet.

27 The writer suggests that people may have difficulty sleeping in the mountains because

- **A** humans do not prefer peace and quiet to noise.
- **B** they may be exposed to short bursts of very strange sounds.
- **C** humans prefer to hear a certain amount of noise while they sleep.
- **D** they may have adapted to a higher noise level in the city.

28 In noise experiments, Glass and Singer found that

- **A** problem-solving is much easier under quiet conditions.
- **B** physiological arousal prevents the ability to work.
- **C** bursts of noise do not seriously disrupt problem-solving in the long term.
- **D** the physiological arousal of control subjects declined quickly.

29 Researchers discovered that high noise levels are not likely to interfere with the

- **A** successful performance of a single task.
- **B** tasks of pilots or air traffic controllers.
- **C** ability to repeat numbers while tracking moving lines.
- **D** ability to monitor three dials at once.

Hình 2.7 Ví dụ dạng bài trắc nghiệm [11]

2.1.2.8 Hoàn thành câu (Sentence completion)

Yêu cầu: Trong các dạng bài IELTS Reading, Hoàn thành câu (Sentence completion) được xem là một dạng khó. Dạng này yêu cầu thí sinh hoàn thành các câu được lấy từ bài đọc hiểu.

Dạng bài này đánh giá khả năng:

- Xác định vị trí của chi tiết hoặc thông tin cụ thể.
- Kỹ năng Scanning, vốn từ vựng, ngữ pháp.

Mô tả dạng bài: Đề bài cung cấp các câu có để trống, thí sinh cần chọn từ phù hợp để điền vào chỗ trống đó. Các thông tin hoàn toàn được lấy từ bài đọc. [1]

Ví dụ dạng bài:

Questions 15–21

Complete the sentences below.

*Choose **NO MORE THAN TWO WORDS AND/OR A NUMBER** from the text for each answer.*

Write your answers in boxes 15–21 on your answer sheet.

15 The law on minimum pay doesn't cover you if you are working in your ..... or if you are a volunteer.

16 You may be paid under £5 an hour if you are receiving ..... at the start of a job.

17 There are different rules for people who are provided with ..... with their jobs.

18 If you earn extra money, for example for working longer hours or in tips, this counts as part of your wage when you receive it via ..... .

19 Anyone being paid below the National Minimum Wage should speak to their ..... if they can.

20 According to the law, you can ask to look at your boss's ..... .

21 You have a period of ..... to complain if your boss does not co-operate within the specified period of time.

Hình 2.8 Ví dụ dạng câu hỏi hoàn thành câu [11]

2.1.2.9 Dạng câu hỏi với câu trả lời ngắn (Short-answer questions)

Yêu cầu: Trả lời một câu hỏi bằng một câu trả lời ngắn/ rút gọn.

Dạng bài này đánh giá khả năng:

- Xác định vị trí và hiểu chính xác thông tin
- Sử dụng ngữ pháp, từ vựng để rút gọn câu

Mô tả dạng bài: Đề bài sẽ cung cấp các câu hỏi và số lượng từ cần trả lời cho mỗi câu hỏi để thí sinh làm bài. [1]

Ví dụ dạng bài:

Questions 28–31

Answer the questions below.

*Choose **ONE NUMBER ONLY** from the text for each answer.*

Write your answers in boxes 28–31 on your answer sheet.

28 When was the furnace bought by Darby originally constructed?

29 When were the roads leading to the bridge completed?

30 When was the bridge closed to traffic?

31 When was a model of the bridge built?

Hình 2.9 Mẫu câu hỏi cho dạng bài yêu cầu câu trả lời ngắn [11]

2.1.3 Tiêu chí chấm điểm

Điểm IELTS Reading sẽ được tính bằng số câu trả lời đúng trong tổng 40 câu hỏi của bài thi. Với một câu trả lời đúng, thí sinh sẽ nhận được một điểm. Band điểm IELTS Reading là 1.0 – 9.0 dựa trên tổng số câu trả lời đúng. [2]

Bảng 2.1 Thang điểm IELTS Reading Academic

Số câu đúng	Điểm IELTS Reading Academic
39-40	9.0
37-38	8.5
35-36	8.0
33-34	7.5
30-32	7.0
27-29	6.5
23-26	6.0
19-22	5.5

Số câu đúng	Điểm IELTS Reading Academic
15-18	5.0
13-14	4.5
10-12	4.0
8-9	3.5
6-7	3.0

Bảng 2.2 Thang điểm IELTS Reading General Training

Số câu đúng	Điểm IELTS General Training
40	9.0
39	8.5
37-38	8.0
36	7.5
34-35	7.0
32-33	6.5
30-31	6.0
27-29	5.5
23-26	5.0
19-22	4.5
15-18	4.0
12-14	3.5
9-11	3.0

Dựa vào kết quả bảng điểm IELTS sẽ đánh giá năng lực tiếng Anh của thí sinh như sau: [2]

- **Cấp độ kỹ năng: Chuyên gia**

Thí sinh có đầy đủ khả năng sử dụng ngôn ngữ. Việc sử dụng tiếng Anh phù hợp, chính xác, trôi chảy và thể hiện sự hiểu biết đầy đủ.

- **Cấp độ kỹ năng: Rất tốt**

Thí sinh có khả năng sử dụng ngôn ngữ hoàn toàn, chỉ thỉnh thoảng có những sai sót thiếu hệ thống và sử dụng không phù hợp. Họ có thể hiểu nhầm một số điều trong những tình huống không quen thuộc. Họ xử lý tốt những lập luận phức tạp và chi tiết.

- **Cấp độ kỹ năng: Tốt**

Thí sinh có khả năng sử dụng ngôn ngữ tốt, mặc dù đôi khi có những điểm không chính xác, cách sử dụng không phù hợp và hiểu lầm trong một số tình huống. Họ thường xử lý tốt ngôn ngữ phức tạp và hiểu lý luận chi tiết.

- **Cấp độ kỹ năng: Có năng lực**

Thí sinh có khả năng sử dụng ngôn ngữ hiệu quả mặc dù có một số điểm không chính xác, cách sử dụng không phù hợp và hiểu lầm. Họ có thể sử dụng và hiểu được ngôn ngữ phức tạp, đặc biệt trong những tình huống quen thuộc.

- **Cấp độ kỹ năng: Khiêm tốn**

Thí sinh có khả năng sử dụng ngôn ngữ một phần và xử lý được ý nghĩa tổng thể trong hầu hết các tình huống, mặc dù họ có thể mắc nhiều lỗi. Họ phải có khả năng xử lý các giao tiếp cơ bản trong lĩnh vực riêng của họ.

- **Cấp độ kỹ năng: Hạn chế**

Năng lực cơ bản của Thí sinh chỉ giới hạn ở những tình huống quen thuộc. Họ thường xuyên bộc lộ những vấn đề trong việc hiểu và diễn đạt.

- **Cấp độ kỹ năng: Vô cùng hạn chế**

Thí sinh chỉ truyền đạt và hiểu ý nghĩa chung trong những tình huống rất quen thuộc. Thường xuyên có sự cố trong giao tiếp.

- **Cấp độ kỹ năng: Không liên tục**

Thí sinh gặp khó khăn lớn trong việc hiểu tiếng Anh nói và viết.

- **Cấp độ kỹ năng: Không sử dụng**

Thí sinh không có khả năng sử dụng ngôn ngữ ngoại trừ một vài từ đơn lẻ.

- **Cấp độ kỹ năng: Chưa kiểm tra**

Thí sinh không trả lời các câu hỏi. [2]

2.2 Tổng quan về cơ sở dữ liệu NoSQL

2.2.1 Cơ sở dữ liệu NoSQL

Cơ sở dữ liệu NoSQL còn được gọi là cơ sở dữ liệu được xây dựng cho mục đích nhất định chuyên dành cho các mô hình dữ liệu cụ thể và lưu trữ dữ liệu trong các sơ đồ linh hoạt dễ dàng điều chỉnh quy mô cho các ứng dụng hiện đại. Nhiều khối lượng công việc cơ sở dữ liệu có thể hưởng lợi từ hiệu quả chi phí và hiệu suất của cơ sở dữ liệu NoSQL. Ngoài ra, cơ sở dữ liệu NoSQL được công nhận rộng rãi vì khả năng dễ phát triển, chức năng cũng như hiệu năng ở quy mô lớn. Trang này có các tài nguyên giúp người dùng hiểu rõ hơn về cơ sở dữ liệu NoSQL và bắt đầu sử dụng. [3]

2.2.2 Ưu điểm của cơ sở dữ liệu NoSQL

Các ứng dụng theo mô hình dữ liệu quan hệ có thể đối mặt với một số thách thức có thể được giải quyết bằng cơ sở dữ liệu NoSQL. Ví dụ, các ứng dụng xử lý một khối lượng dữ liệu lớn từ các nguồn khác nhau như phương tiện truyền thông xã hội, cảm biến thông minh và cơ sở dữ liệu của bên thứ ba. Tất cả dữ liệu khác nhau này không phù hợp với mô hình quan hệ. Việc thực thi các cấu trúc dạng bảng có thể dẫn đến dư thừa, sao chép dữ liệu và các vấn đề về hiệu suất trên quy mô lớn.

Cơ sở dữ liệu NoSQL chuyên dành cho các mô hình dữ liệu phi quan hệ và có sơ đồ linh hoạt để xây dựng các ứng dụng hiện đại. Chúng được công nhận rộng rãi vì khả năng dễ phát triển, chức năng cũng như hiệu năng ở quy mô lớn. Lợi ích của cơ sở dữ liệu NoSQL được liệt kê dưới đây. [3]

- **Tính linh hoạt**

Cơ sở dữ liệu NoSQL thường cung cấp các sơ đồ linh hoạt giúp công đoạn phát triển nhanh hơn và có khả năng lặp lại cao hơn. Mô hình dữ liệu linh hoạt biến cơ sở dữ liệu NoSQL thành lựa chọn lý tưởng cho dữ liệu không được tổ chức thành cấu trúc hoặc có cấu trúc chưa hoàn chỉnh.

- **Khả năng mở rộng**

Cơ sở dữ liệu NoSQL thường được thiết kế để tăng quy mô bằng cách sử dụng các cụm phần cứng được phân phối thay vì tăng quy mô bằng cách bổ sung máy chủ mạnh và tốn kém. Một số nhà cung cấp dịch vụ đám mây xử lý các hoạt động này một cách không công khai dưới dạng dịch vụ được quản lý đầy đủ.

- **Hiệu năng cao**

Cơ sở dữ liệu NoSQL được tối ưu hóa cho các mô hình dữ liệu cụ thể và các mẫu truy cập. Chúng cho phép hiệu suất cao hơn so với khi người dùng đang cố gắng thực hiện chức năng tương tự với cơ sở dữ liệu quan hệ.

- **Chức năng cao**

Cơ sở dữ liệu NoSQL cung cấp các API và kiểu dữ liệu cực kỳ thiết thực được xây dựng riêng cho từng mô hình dữ liệu tương ứng. [3]

2.2.3 Các trường hợp sử dụng của cơ sở dữ liệu NoSQL

Người dùng có thể sử dụng cơ sở dữ liệu NoSQL để xây dựng nhiều ứng dụng di động, Internet vạn vật (IoT), chơi game và web hiệu suất cao cung cấp trải nghiệm người dùng tuyệt vời trên quy mô lớn. Phạm vi cơ sở dữ liệu NoSQL và các trường hợp sử dụng tương ứng rất rộng. Mặc dù rất khó để trình bày một tập hợp các trường hợp sử dụng đại diện, dưới đây chúng tôi sẽ cung cấp một vài ví dụ minh họa để bắt đầu suy nghĩ và khuyến khích người dùng tìm hiểu thêm về từng cơ sở dữ liệu NoSQL và các trường hợp sử dụng tương ứng. [3]

- **Quản lý dữ liệu theo thời gian thực**

Người dùng có thể cung cấp các đề xuất thời gian thực, cá nhân hóa và cải thiện trải nghiệm người dùng với cơ sở dữ liệu NoSQL.

Ví dụ: Disney+ cung cấp thư viện nội dung kỹ thuật số mở rộng cho hơn 150 triệu người đăng ký sử dụng công nghệ cơ sở dữ liệu NoSQL. Công ty có thể mở rộng quy mô và cung cấp các tính năng thông dụng như Tiếp tục xem, Danh sách xem và Đề xuất cá nhân với Amazon DynamoDB.

- **Bảo mật trên đám mây**

Người dùng có thể sử dụng cơ sở dữ liệu đồ thị để nhanh chóng khám phá các mối quan hệ phức tạp trong dữ liệu của mình.

Ví dụ: Wiz đã hình dung lại bảo mật đám mây như một biểu đồ sử dụng Amazon Neptune. Wiz giúp khách hàng của họ cải thiện trạng thái bảo mật của họ bằng cách nhanh chóng xác định và khắc phục các rủi ro quan trọng nhất. Họ sử dụng mô hình đồ thị được lưu trữ trong Amazon Neptune để khám phá sự kết hợp độc hại của các yếu tố rủi ro đại diện cho các rủi ro quan trọng. Các công cụ rủi ro của Wiz đi qua đồ thị và trong vài giây có thể kết hợp một loạt các yếu tố rủi ro liên kết với nhau trong một đồ thị bảo mật.

- **Các ứng dụng có độ sẵn sàng cao**

Cơ sở dữ liệu NoSQL phân tán rất tuyệt vời để xây dựng các ứng dụng có độ sẵn sàng cao, độ trễ thấp dùng cho nhắn tin, mạng xã hội, chia sẻ tệp và hơn thế nữa.

Ví dụ: Snapchat có hơn 290 triệu người dùng gửi hàng tỷ hình ảnh và tin nhắn video mỗi ngày. Công ty sử dụng hệ thống cơ sở dữ liệu NoSQL để giảm 20% độ trễ trung bình của việc gửi tin nhắn. [3]

2.2.4 Cách thức hoạt động của cơ sở dữ liệu NoSQL

Cơ sở dữ liệu NoSQL sử dụng nhiều mô hình dữ liệu để truy cập và quản lý dữ liệu. Các loại cơ sở dữ liệu này được tối ưu hóa dành riêng cho các ứng dụng yêu cầu mô hình dữ liệu linh hoạt, lượng dữ liệu lớn và độ trễ thấp, có thể đạt được bằng cách giảm bớt một số hạn chế về tính nhất quán của dữ liệu của các cơ sở dữ liệu quan hệ. Có sự khác biệt trong việc thực hiện dựa trên mô hình dữ liệu. Tuy nhiên, nhiều cơ sở dữ liệu NoSQL sử dụng Javascript Object Notation (JSON), một định dạng trao đổi dữ liệu mở đại diện cho dữ liệu dưới dạng tập hợp các cặp tên-giá trị. [3]

❖ **Ví dụ cơ sở dữ liệu NoSQL**

Hãy xem ví dụ về mô hình sơ đồ dành cho cơ sở dữ liệu sách đơn giản:

Trong cơ sở dữ liệu quan hệ, hồ sơ về một cuốn sách thường được phân tách (hay còn gọi là "chuẩn hóa") và lưu trữ trong các bảng tách biệt nhau, còn mối quan hệ được quy định bằng các ràng buộc khóa ngoại và khóa chính. Trong ví dụ này, bảng Sách có các cột cho ISBN, Tên sách và Số phiên bản, bảng Tác giả có các cột cho ID tác giả và Tên tác giả và cuối cùng, bảng Tác giả–ISBN có các cột cho ID tác giả và ISBN. Mô hình quan hệ được thiết kế để cho phép các cơ sở dữ liệu này thực thi tính toàn vẹn tham chiếu giữa nhiều bảng trong cơ sở dữ liệu, được chuẩn hóa để giảm dư thừa và thường được tối ưu hóa cho mục đích lưu trữ.

Trong cơ sở dữ liệu NoSQL, hồ sơ về một cuốn sách thường được lưu trữ dưới dạng văn bản. Với từng quyển sách, mục, ISBN, Tên sách, Số phiên bản, Tên tác giả và ID tác giả được lưu trữ dưới dạng thuộc tính trong một văn bản duy nhất. Trong mô hình này, dữ liệu được tối ưu hóa cho việc phát triển trực quan và khả năng điều chỉnh quy mô theo chiều ngang.

❖ **Thuật ngữ SQL so với NoSQL**

Bảng sau sẽ so sánh thuật ngữ được sử dụng trong cơ sở dữ liệu NoSQL chọn lọc với thuật ngữ được sử dụng trong cơ sở dữ liệu SQL. [3]

Bảng 2.3 Bảng sau so sánh thuật ngữ được sử dụng trong cơ sở dữ liệu NoSQL

SQL	MongoDB	DynamoDB	Cassandra	Couchbase
Bảng	Bộ sưu tập (Collection)	Bảng	Bảng	Bộ chứa dữ liệu (Data Bucket)
Hàng (Row)	Tài liệu (Document)	Mục (Item)	Hàng (Row)	Tài liệu (Document)
Cột (Column)	Trường (Field)	Thuộc tính (Attribute)	Cột (Column)	Trường (Field)
Khóa chính (Primary Key)	Id đối tượng (_id)	Khóa chính (Primary Key)	Khóa chính (Primary Key)	ID văn bản (Document ID)
Chỉ mục (Index)	Chỉ mục (Index)	Chỉ mục thứ cấp (Secondary Index)	Chỉ mục (Index)	Chỉ mục (Index)
Chế độ xem (View)	Chế độ xem (View)	Chỉ mục thứ cấp toàn cục (Global Secondary Index)	Chế độ xem cụ thể hóa (Materialized View)	Chế độ xem (View)
Bảng/Đối tượng lồng nhau	Văn bản nhúng (Embedded Document)	Bản đồ (Map)	Bản đồ (Map)	Bản đồ (Map)
Mảng (Array)	Mảng (Array)	Danh sách (List)	Danh sách (List)	Danh sách (List)

2.2.5 Cơ sở dữ liệu NoSQL gồm những loại

Có một số hệ thống cơ sở dữ liệu NoSQL khác nhau do nhiều điểm khác biệt trong cách quản lý và lưu trữ dữ liệu không lược đồ. Chúng tôi giải thích một số loại phổ biến dưới đây.

❖ **Cơ sở dữ liệu khóa-giá trị**

Cơ sở dữ liệu khóa-giá trị rất dễ phân chia và cho phép thay đổi quy mô theo chiều ngang tại các cấp độ mà các loại cơ sở dữ liệu NoSQL không thể làm được. Cơ sở dữ liệu khóa–giá trị lưu trữ dữ liệu dưới dạng một tập hợp các cặp khóa-giá trị mà tại đó, khóa đóng vai trò là mã định danh duy nhất. Khóa và giá trị đều có thể là bất cứ thứ gì, từ đối tượng đơn giản đến đối tượng phức hợp phức tạp. Thiết kế dữ liệu lưu trữ khóa–giá trị được dùng cho trò chơi, công nghệ quảng cáo và đặc biệt thích hợp cho IoT. Amazon DynamoDB được thiết kế để có hiệu suất ổn định với độ trễ ổn định chỉ vài mili giây cho khối lượng công việc thuộc quy mô bất kỳ. [3]

❖ **Cơ sở dữ liệu tài liệu**

Cơ sở dữ liệu tài liệu có cùng một định dạng mô hình tài liệu mà các nhà phát triển sử dụng trong mã ứng dụng của họ. Chúng lưu trữ dữ liệu dưới dạng các đối tượng JSON với bản chất linh hoạt, bán cấu trúc và phân cấp. Tính chất linh hoạt, bán cấu trúc và phân cấp của các tài liệu và cơ sở dữ liệu tài liệu cho phép chúng phát triển phù hợp với yêu cầu của ứng dụng. Mô hình cơ sở dữ liệu tài liệu phát huy hiệu quả với danh mục, hồ sơ người dùng và hệ thống quản lý nội dung, nơi từng văn bản là duy nhất và phát triển theo thời gian. Amazon DocumentDB (với khả năng tương thích MongoDB) và MongoDB là các cơ sở dữ liệu tài liệu phổ biến cung cấp các API mạnh mẽ và trực quan để phát triển linh hoạt và lặp lại.

❖ **Cơ sở dữ liệu đồ thị**

Mục đích của cơ sở dữ liệu đồ thị là giúp người dùng dễ dàng xây dựng và chạy ứng dụng hoạt động với tập dữ liệu kết nối cao. Chúng sử dụng nút để lưu trữ các thực thể dữ liệu và biên để lưu trữ mối quan hệ giữa các thực thể. Biên luôn có một nút bắt đầu, nút kết thúc, kiểu và hướng. Biên có thể mô tả các mối quan hệ cha-con, hành động và quyền sở hữu và các mối quan hệ tương tự. Không giới hạn về số lượng cũng như loại mối quan hệ mà một nút có thể sở hữu. Người dùng có thể sử dụng cơ sở dữ liệu đồ thị để xây dựng và chạy ứng dụng hoạt động với bộ dữ liệu kết nối cao. Các trường hợp sử dụng điển hình của cơ sở dữ liệu đồ thị bao gồm mạng xã hội, công cụ đề xuất, phát hiện gian lận và đồ thị kiến thức. Amazon Neptune là một dịch vụ cơ sở dữ liệu đồ thị được quản lý đầy đủ, hỗ trợ cả mô hình Đồ thị thuộc tính và Khung mô tả tài nguyên (RDF), cho phép người dùng lựa chọn hai API đồ thị: TinkerPop và RDF/SPARQL.

❖ **Cơ sở dữ liệu nằm trong bộ nhớ**

Trong khi các cơ sở dữ liệu phi quan hệ khác lưu trữ dữ liệu trên đĩa hoặc SSD, các kho dữ liệu trong bộ nhớ được thiết kế để loại bỏ nhu cầu truy cập đĩa. Chúng lý tưởng cho các ứng dụng yêu cầu thời gian phản hồi micro giây hoặc có lưu lượng truy cập tăng đột biến lớn. Người dùng có thể sử dụng chúng trong các ứng dụng trò chơi và công nghệ quảng cáo cho các tính năng như bảng xếp hạng, cửa hàng phiên và phân tích thời gian thực. Amazon MemoryDB dành cho Redis là một dịch vụ cơ sở dữ liệu nằm trong bộ nhớ, bền, tương thích với Redis có chức năng đảm bảo độ trễ của thao tác đọc tính bằng micro giây, độ trễ của thao tác ghi chưa đến mười mili giây và độ bền trên nhiều vùng sẵn sàng. Amazon ElastiCache là dịch vụ bộ nhớ đệm trong bộ nhớ được quản lý hoàn toàn, tương thích với cả Redis và Memcached để phục vụ khối lượng công việc có độ trễ thấp, thông lượng cao. Trình tăng tốc Amazon DynamoDB (DAX) là một ví dụ khác về kho dữ liệu chuyên dụng giúp DynamoDB đọc nhanh hơn rất nhiều.

❖ **Tìm kiếm cơ sở dữ liệu**

Cơ sở dữ liệu công cụ tìm kiếm là một loại cơ sở dữ liệu phi quan hệ được dành riêng cho việc tìm kiếm nội dung dữ liệu, chẳng hạn như bản ghi đầu ra ứng dụng được sử dụng bởi các nhà phát triển để khắc phục sự cố. Chúng sử dụng các chỉ mục để phân loại các đặc điểm tương tự giữa dữ liệu và tạo điều kiện cho khả năng tìm kiếm. Cơ sở dữ liệu công cụ tìm kiếm được tối ưu hóa để sắp xếp dữ liệu phi cấu trúc như hình ảnh và video. Dịch vụ OpenSearch của Amazon được xây dựng nhằm mục đích cung cấp hình ảnh trực quan và phân tích gần như trong thời gian thực về các dữ liệu do máy tạo bằng cách lập chỉ mục, tập hợp và tìm kiếm trong các bản ghi và chỉ số bán cấu trúc. [3]

2.2.6 Sự khác biệt giữa cơ sở dữ liệu NoSQL và SQL

Trong nhiều thập kỷ, mô hình dữ liệu chiếm ưu thế trong phát triển ứng dụng là mô hình dữ liệu quan hệ lưu trữ dữ liệu trong các bảng được tạo thành từ các hàng và cột. Ngôn ngữ truy vấn có cấu trúc (SQL) được sử dụng để tạo và chỉnh sửa các bảng quan hệ này. Cơ sở dữ liệu SQL mô hình hóa các mối quan hệ dữ liệu dưới dạng bảng.

Các hàng trong bảng đại diện cho một tập hợp các giá trị có liên quan của một đối tượng hoặc thực thể. Mỗi cột trong bảng đại diện cho một thuộc tính dữ liệu và một trường (hoặc ô bảng) lưu trữ giá trị thực tế của thuộc tính. Người dùng có thể sử

dụng hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS) để truy cập dữ liệu theo nhiều cách khác nhau mà không cần tổ chức lại các bảng cơ sở dữ liệu.

Mãi cho đến cuối những năm 2000, các mô hình dữ liệu linh hoạt khác mới bắt đầu được đưa vào áp dụng và sử dụng nhiều hơn. Để phân biệt và phân loại các cơ sở dữ liệu và mô hình dữ liệu mới này, từ NoSQL đã được đặt ra. NoSQL là viết tắt của không chỉ SQL hoặc non-SQL. Thông thường, thuật ngữ NoSQL được sử dụng tương đương với thuật ngữ phi quan hệ. Sự khác biệt chính giữa cơ sở dữ liệu quan hệ và không quan hệ được đưa ra trong bảng dưới đây. [3]

Bảng 2.4 So sánh sự khác biệt giữa cơ sở dữ liệu NoSQL và SQL

Tiêu chí	**Cơ sở dữ liệu quan hệ**	**Cơ sở dữ liệu NoSQL**
Khối lượng công việc tối ưu	Cơ sở dữ liệu quan hệ được thiết kế dành cho các ứng dụng xử lý giao dịch trực tuyến (OLTP) trong giao dịch có độ ổn định cao. Chúng cũng hữu ích để xử lý phân tích trực tuyến (OLAP).	Các cơ sở dữ liệu NoSQL được thiết kế cho các mẫu truy cập dữ liệu, bao gồm các ứng dụng có độ trễ thấp. Cơ sở dữ liệu tìm kiếm NoSQL được thiết kế để phục vụ phân tích dữ liệu có cấu trúc chưa hoàn chỉnh.
Mô hình dữ liệu	Mô hình quan hệ chuẩn hóa dữ liệu vào bảng được hình thành từ hàng và cột. Sơ đồ quy định rõ ràng bảng, hàng, cột, chỉ mục, mối quan hệ giữa các bảng và các thành tố cơ sở dữ liệu khác. Cơ sở dữ liệu sẽ thực thi tính toàn vẹn tham chiếu trong mối quan hệ giữa các bảng.	Các cơ sở dữ liệu NoSQL cung cấp nhiều mô hình dữ liệu khác nhau như khóa-giá trị, tài liệu, biểu đồ và cột, được tối ưu hóa để đạt hiệu năng và quy mô tối ưu.

Tiêu chí	Cơ sở dữ liệu quan hệ	Cơ sở dữ liệu NoSQL
Thuộc tính ACID	Cơ sở dữ liệu quan hệ có các thuộc tính mang tính nguyên tố, nhất quán, tách biệt và bền vững (ACID): • Tính nguyên tố đòi hỏi giao dịch phải được thực thi đầy đủ hoặc hoàn toàn không thực hiện. • Tính nhất quán đòi hỏi rằng khi dữ liệu phải nhất quán với sơ đồ cơ sở dữ liệu giao dịch được thực hiện. • Tính tách biệt đòi hỏi rằng các giao dịch diễn ra đồng thời phải được thực thi tách biệt với nhau. • Tính bền vững đòi hỏi phải có khả năng phục hồi từ tình trạng hư hỏng hệ thống hoặc mất điện đột ngột về trạng thái đã biết cuối cùng.	Hầu hết cơ sở dữ liệu NoSQL thường phải đánh đổi bằng cách nới lỏng một số thuộc tính ACID này của cơ sở dữ liệu quan hệ để có mô hình dữ liệu linh hoạt hơn có khả năng thay đổi quy mô theo chiều ngang. Việc này biến các cơ sở dữ liệu NoSQL thành lựa chọn tuyệt vời cho các trường hợp sử dụng cần thông lượng cao, độ trễ thấp cần thay đổi quy mô theo chiều ngang vượt qua giới hạn của một phiên bản duy nhất.

Tiêu chí	Cơ sở dữ liệu quan hệ	Cơ sở dữ liệu NoSQL
Hiệu năng	Hiệu năng thường phụ thuộc vào hệ thống con của ổ đĩa. Thông thường, việc tối ưu hóa các truy vấn, chỉ mục và cấu trúc bảng bắt buộc phải được thực hiện để đạt mức hiệu năng tối đa.	Hiệu năng thường được xem là chức năng của kích cỡ cụm phần cứng ngầm, độ trễ mạng và ứng dụng đưa ra lệnh gọi.
Điều chỉnh quy mô	Cơ sở dữ liệu quan hệ thường tăng quy mô bằng cách tăng năng lực điện toán của phần cứng hoặc tăng quy mô bằng cách thêm bản sao của khối lượng công việc chỉ đọc.	Cơ sở dữ liệu NoSQL thường có thể phân mảnh. Điều này là do các mẫu truy cập khóa-giá trị có khả năng tăng quy mô bằng cách sử dụng kiến trúc được phân phối để tăng thông lượng, đem đến hiệu năng ổn định với quy mô gần như không giới hạn.
API	Yêu cầu lưu trữ và truy xuất dữ liệu được truyền đạt bằng cách sử dụng các truy vấn nhất quán với ngôn ngữ truy vấn có cấu trúc (SQL). Các truy vấn này được phân tích và thực thi bởi cơ sở dữ liệu quan hệ.	API trên cơ sở đối tượng cho phép các nhà phát triển ứng dụng dễ dàng lưu trữ và truy xuất cấu trúc dữ liệu trong bộ nhớ. Khóa phân mảnh tìm kiếm các cặp khóa-giá trị, tập hợp cột hoặc văn bản có cấu trúc chưa hoàn chỉnh có chứa đối tượng và thuộc tính của ứng dụng được xếp theo chuỗi.

2.2.7 Khi nào nên chọn cơ sở dữ liệu NoSQL thay vì cơ sở dữ liệu SQL

Cơ sở dữ liệu NoSQL là tốt nhất để xử lý dữ liệu không xác định, không liên quan hoặc thay đổi nhanh chóng. Nó rất trực quan để sử dụng cho các nhà phát triển

khi ứng dụng có ảnh hưởng đến lược đồ cơ sở dữ liệu. Người dùng có thể sử dụng nó cho các ứng dụng:

- Cần các sơ đồ linh hoạt giúp công đoạn phát triển nhanh hơn và có khả năng lặp lại cao hơn.
- Ưu tiên hiệu suất hơn tính nhất quán của dữ liệu mạnh mẽ và duy trì mối quan hệ giữa các bảng dữ liệu (tính toàn vẹn tham chiếu).
- Yêu cầu mở rộng theo chiều ngang bằng cách phân mảnh trên các máy chủ.
- Hỗ trợ dữ liệu bán cấu trúc và phi cấu trúc.

Người dùng không phải lúc nào cũng phải lựa chọn giữa lược đồ cơ sở dữ liệu phi quan hệ và lược đồ cơ sở dữ liệu quan hệ. Người dùng có thể sử dụng kết hợp cơ sở dữ liệu SQL và NoSQL trong các ứng dụng của mình. Cách tiếp cận kết hợp này khá phổ biến và đảm bảo mỗi khối lượng công việc được ánh xạ vào cơ sở dữ liệu phù hợp để có hiệu suất giá tối ưu. [3]

2.3 Nghiên cứu cách thiết kế cơ sở dữ liệu với mô hình dữ liệu trong cơ sở dữ liệu NoSQL

2.3.1 Thiết kế cơ sở dữ liệu NoSQL

Thiết kế cơ sở dữ liệu NoSQL không chỉ đơn thuần là việc xác định cách lưu trữ dữ liệu, mà còn bao gồm một loạt các quyết định chiến lược về cách tổ chức và quản lý dữ liệu trong môi trường phi cấu trúc và phân tán. Trái ngược với cơ sở dữ liệu quan hệ truyền thống (RDBMS), nơi mà mô hình dữ liệu thường được xác định trước và cố định, thiết kế NoSQL database thường tập trung vào việc tối ưu hóa truy cập dữ liệu và khả năng mở rộng của hệ thống.

Thông qua việc phân tích yêu cầu của dự án và đánh giá các mẫu truy vấn, các lập trình viên cơ sở dữ liệu NoSQL xác định cách tổ chức dữ liệu sao cho phù hợp với nhu cầu của ứng dụng. Điều này bao gồm việc chọn lựa khóa chính (primary key), quyết định các trường (field) cần được index và xác định cách tổ chức dữ liệu theo các mô hình như cây (tree), mạng lưới (graph) hoặc chuỗi thời gian (time-series). [4]

2.3.2 Điểm mạnh của thiết kế cơ sở dữ liệu NoSQL

Một điểm mạnh của thiết kế cơ sở dữ liệu NoSQL là tính linh hoạt, cho phép dễ dàng thay đổi cấu trúc dữ liệu để đáp ứng các yêu cầu kinh doanh mới hoặc mô hình truy vấn khác nhau. Điều này thường dẫn đến quá trình thiết kế được

thực hiện theo cách lặp lại và có thể điều chỉnh theo thời gian, khi yêu cầu của ứng dụng thay đổi hoặc khi có sự phát triển trong dữ liệu hoặc môi trường.

Do đó, việc điều chỉnh và cải thiện schema là một phần quan trọng của việc duy trì và phát triển các hệ thống cơ sở dữ liệu NoSQL. Trong tổ chức dữ liệu NoSQL, những quyết định về thiết kế không chỉ dựa trên cấu trúc dữ liệu mà còn phản ánh nhu cầu về hiệu suất, mở rộng và tính nhất quán của hệ thống. [4]

2.3.3 Các quy tắc khi thiết kế cơ sở dữ liệu NoSQL

Thiết kế cơ sở dữ liệu NoSQL tập trung vào cách mà ứng dụng sẽ truy vấn dữ liệu, thay vì tập trung vào các mối quan hệ trong dữ liệu.Thiết kế cơ sở dữ liệu NoSQL nhấn mạnh vào các mẫu truy cập hơn là các mô hình dữ liệu trừu tượng. Đó là lý do tại sao các quy tắc tốt nhất cho thiết kế NoSQL database yêu cầu một biểu đồ về các cách mà ứng dụng sẽ truy vấn dữ liệu, và sự hỗ trợ cần thiết cho khối lượng công việc.

Thiết kế cơ sở dữ liệu NoSQL cũng xem xét tần suất mà bộ dữ liệu sẽ thay đổi, lượng dữ liệu sẽ được lưu trữ và các yêu cầu về tính sẵn có, hiệu suất và nhất quán.

Thiết kế cơ sở dữ liệu NoSQL có nghĩa là lựa chọn loại cơ sở dữ liệu phù hợp cho một ứng dụng cụ thể. Các loại cơ sở dữ liệu này có thể là lưu trữ theo key-value, theo column-family, cơ sở dữ liệu hướng tài liệu và cơ sở dữ liệu dạng mạng lưới. [4]

2.3.4 Những mẫu thiết kế cơ sở dữ liệu NoSQL thông dụng nhất

Cơ sở dữ liệu NoSQL đang trở nên phổ biến hơn vì chúng cung cấp tính linh hoạt, khả năng mở rộng và hiệu suất cho việc xử lý các tập dữ liệu lớn và phức tạp. Tuy nhiên, thiết kế một cơ sở dữ liệu NoSQL đòi hỏi một cách tiếp cận khác biệt so với các cơ sở dữ liệu quan hệ truyền thống.

❖ **Mẫu Key-value**

Một trong những mẫu thiết kế NoSQL database đơn giản và phổ biến nhất là mẫu key-value. Trong mẫu này, mỗi danh sách dữ liệu được lưu trữ dưới dạng một cặp khóa-giá trị với mỗi khoá là độc nhất. Giá trị có thể là bất cứ thứ gì, chẳng hạn như một chuỗi, một số, một tài liệu hoặc là nhị phân.

Mẫu key-value lý tưởng cho các tình huống mà người dùng cần truy cập nhanh chóng và đơn giản đến dữ liệu bằng cách sử dụng khóa làm định danh.

Ví dụ, người dùng có thể sử dụng mẫu key-value để lưu trữ các thiết lập người dùng, dữ liệu phiên, hoặc dữ liệu caching.

❖ **Mẫu Tài Liệu (Document)**

Trong mẫu này, mỗi mục dữ liệu được lưu trữ dưới dạng một tài liệu độc lập, tự chứa các thông tin liên quan đến nó, không cần phải tham chiếu đến tài liệu khác. Mẫu này còn bao gồm một collection các trường và giá trị mà không theo một cấu trúc cố định và có thể linh động tùy thuộc vào nhu cầu cụ thể của dữ liệu.

Mẫu tài liệu phù hợp cho các tình huống mà người dùng cần lưu trữ dữ liệu phức tạp và đa dạng, có thể thay đổi về cấu trúc và kích thước.

Ví dụ, người dùng có thể sử dụng mẫu tài liệu để lưu trữ các danh mục sản phẩm, hồ sơ khách hàng, hoặc bài đăng trên blog.

❖ **Mẫu Tổ Hợp (Aggregate)**

Mẫu tổ hợp là một biến thể của mẫu tài liệu nhưng tập trung vào việc nhóm các mục dữ liệu liên quan vào một tài liệu duy nhất.

Mẫu tổ hợp hữu ích cho các tình huống mà người dùng cần thực hiện các thao tác dữ liệu một cách liền mạch, trên một tập dữ liệu mà tất cả các mục thuộc cùng một nhóm. Tức là nếu một phần của thao tác dữ liệu không thành công, toàn bộ quá trình thực hiện sẽ bị hủy, đảm bảo rằng dữ liệu không bị sai lệch hay mất mát.

Ví dụ, người dùng có thể sử dụng mẫu tổ hợp để lưu trữ chi tiết đơn hàng, các mặt hàng tồn kho, hoặc bài đăng trên mạng xã hội.

❖ **Mẫu Bucket**

Mẫu bucket là một biến thể khác của mẫu tài liệu nhằm giảm số lượng tài liệu trong một collection bằng cách nhóm các mục dữ liệu tương tự vào một tài liệu duy nhất.

Mẫu bucket hữu ích cho các tình huống khi người dùng cần lưu trữ một lượng lớn dữ liệu mà có ít truy vấn được chọn, hoặc có ít sự lựa chọn trong truy vấn.

Ví dụ, người dùng có thể sử dụng mẫu bucket để lưu trữ dữ liệu cảm biến, dữ liệu nhật ký, hoặc dữ liệu lịch sử.

❖ **Mẫu mạng lưới (Graph)**

Mẫu mạng lưới là một mẫu thiết kế cơ sở dữ liệu NoSQL chuyên biệt, biểu diễn dữ liệu dưới dạng một mạng lưới các nút và cạnh. Mẫu mạng lưới có thể kích hoạt các truy vấn mạnh mẽ và hiệu quả, đi qua các kết nối giữa các mục dữ liệu nhưng cũng có thể đòi hỏi nhiều tài nguyên và kỹ năng để triển khai và duy trì.

Mẫu mạng lưới là lựa chọn tối ưu cho các tình huống mà người dùng cần mô hình phân tích các mối quan hệ có sự phụ thuộc phức tạp giữa các mục dữ liệu. [4]

2.3.5 Các bước thiết kế schema cho cơ sở dữ liệu NoSQL

Mặc dù một số có thể cho rằng NoSQL không cần thiết kế schema do tính linh hoạt và không ràng buộc của nó nhưng thực tế thì thiết kế schema vẫn mang lại nhiều lợi ích trong việc tổ chức dữ liệu. Schema giúp tối ưu hóa hiệu suất truy vấn, đảm bảo tính nhất quán và tổ chức logic của dữ liệu cũng như hỗ trợ quá trình phát triển và bảo trì ứng dụng. Dưới đây là một số bước và phương pháp phổ biến để thiết kế schema trong NoSQL:

❖ **Thiết kế Containers**

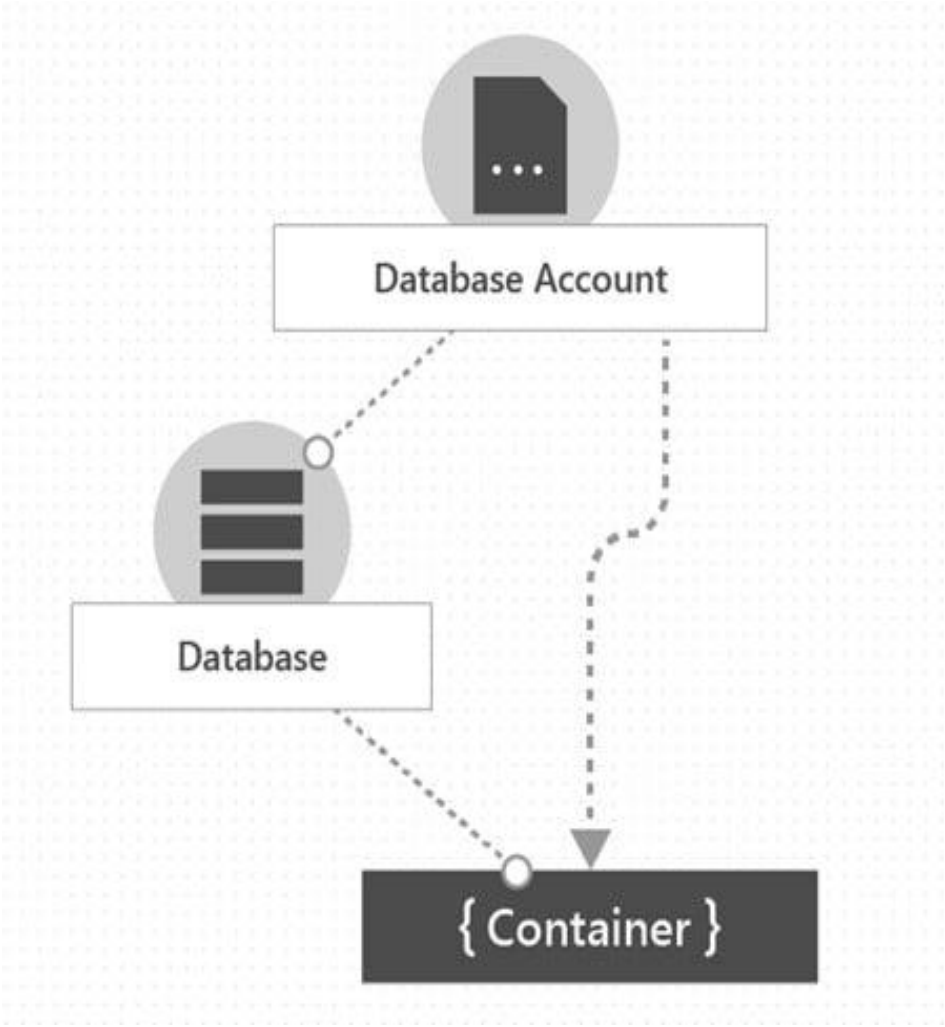
Trong ngữ cảnh này, “container” đề cập đến các đơn vị lưu trữ dữ liệu trong cơ sở dữ liệu NoSQL, tương tự như bảng trong RDBMS.

Xác định các Containers dựa trên Query Patterns: Xây dựng các container dựa trên mẫu truy vấn để mỗi container phục vụ một chức năng duy nhất.

❖ **Cân bằng thiết kế Container**

Đảm bảo container tương ứng với một module/class code để thực thi một chức năng duy nhất, giúp trong việc phát triển và bảo trì hệ thống. Điều quan trọng là phải cân nhắc giữa việc thiết kế một container cho mỗi truy vấn so với thiết kế một container để thực hiện nhiều truy vấn. Do đó, việc sử dụng thuật toán chuẩn hóa ngược (denormalization) đóng vai trò quan trọng trong thiết kế NoSQL. [4]

(Hình ảnh minh họa như hình 2.10 ở trang sau)



Hình 2.10 Cách Container chứa cơ sở dữ liệu NoSQL

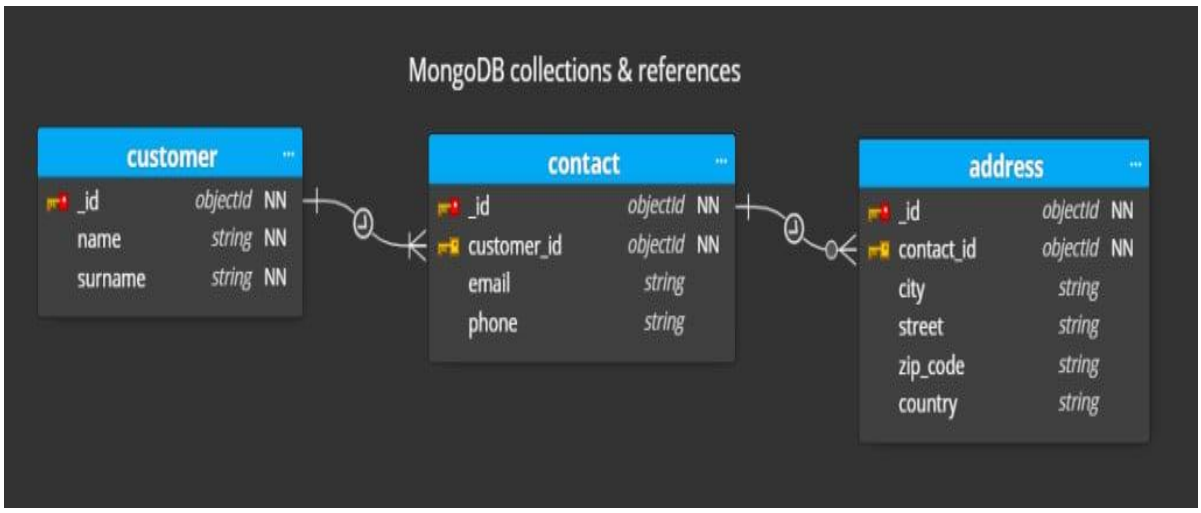
❖ **Điều chỉnh Dữ liệu**

Sử dụng kỹ thuật nhúng (Embedding) hoặc tham chiếu (Referencing) để giảm bớt sự chuẩn hoá dữ liệu giữa các containers.

Xác định mối quan hệ , phép nối cần thực hiện là quan hệ 1-1 (One-to-One), quan hệ 1-nhiều (One-to-Many) hay quan hệ nhiều-nhiều (Many-to-Many) giữa các đối tượng (entities) và áp dụng kỹ thuật giảm chuẩn hóa tương ứng.

Lựa chọn giữa Embedding và Referencing dựa trên số lượng và tần suất truy cập của các thuộc tính. [4]

(Hình ảnh minh họa như hình 2.11 ở trang sau)



Hình 2.11 Dùng tham chiếu (Referencing) để giảm bớt sự chuẩn hóa dữ liệu

❖ **Thiết kế Primary Keys**

Lựa chọn các thuộc tính liên quan đến doanh nghiệp làm Primary Keys để tối ưu hóa việc truy cập dữ liệu.

Phân loại các mẫu truy vấn và thiết kế Primary Keys để đáp ứng yêu cầu của các truy vấn quan trọng và chính xác.

Thiết kế Primary Keys để hỗ trợ việc tham gia dữ liệu trên ứng dụng mà không cần sử dụng các chỉ số phức tạp.

❖ **Thiết kế Indexes**

Tạo các Indexes dựa trên các mẫu truy vấn để cải thiện hiệu suất của các truy vấn không quan trọng hoặc dùng cho mục đích khai thác dữ liệu.

Tránh việc tạo quá nhiều Indexes cho mỗi truy vấn và tối ưu hóa các Indexes để giảm bớt tác động đến hiệu suất hệ thống.

Thiết kế các loại Indexes phù hợp với nhu cầu cụ thể của ứng dụng như B-Tree, Array, Geo-spatial và Full Text Search. [4]

2.3.6 Dữ liệu được lưu trữ trong NoSQL

NoSQL bao gồm bốn loại thiết kế cơ sở dữ liệu chính, vì vậy chi tiết cách lưu trữ dữ liệu của từng loại sẽ khác nhau. Điểm chung của chúng là bộ nhớ vật lý sẽ được phân phối và phân vùng theo chiều ngang.

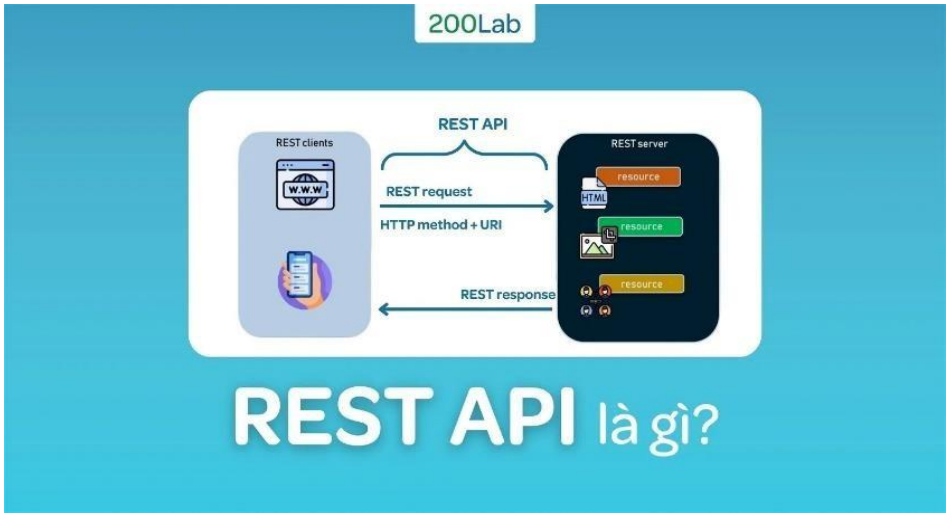
- Cơ sở dữ liệu Key-value: mỗi danh sách dữ liệu được lưu trữ dưới dạng một cặp khóa-giá trị, với mỗi khóa là một định danh duy nhất.

- Cơ sở dữ liệu Document: tổ chức dữ liệu trong các tài liệu có cấu trúc như JSON hoặc XML. Mỗi mục dữ liệu được lưu trữ dưới dạng một tài liệu độc lập.
- Cơ sở dữ liệu Graph: biểu diễn dữ liệu dưới dạng một mạng lưới các nút và cạnh, thể hiện mối quan hệ giữa chúng.
- Cơ sở dữ liệu Column-Family: lưu trữ dữ liệu dưới dạng các cột thay vì hàng như cơ sở dữ liệu quan hệ, giúp tối ưu hóa cho các truy vấn theo cột. [4]

2.4 Các công nghệ thiết kế, cài đặt hệ thống.

2.4.1 Nghiên cứu RESTfull API

RESTfull API là viết tắt của Representational State Transferful Application Programming Interface. RESTful API là một tiêu chuẩn được sử dụng để thiết kế các dịch vụ web dựa trên các phương thức HTTP như GET, POST, PUT và DELETE. Đây là một kiểu thiết kế giao diện lập trình ứng dụng (API) cho phép các ứng dụng web giao tiếp với nhau. [5]



Hình 2.12 RESTfull API

2.4.2 Các thành phần của RESTfull API

RESTful API được thiết kế dựa trên kiến trúc client-server, stateless và cacheable. Bản thân kiến trúc này rất đơn giản và dễ hiểu, trong đó client gửi yêu cầu đến server, server xử lý yêu cầu và trả về kết quả cho client. Tuy nhiên, để thiết kế một RESTful API tốt thì cần phải biết rõ các thành phần chính của nó.

- **Resource:** Đây là đối tượng mà chúng ta muốn truy xuất từ server. Resource được định danh bởi một URI (Uniform Resource Identifier) và có thể được truy xuất bằng HTTP methods như GET, POST, PUT, DELETE.

- **HTTP Method:** Đây là cách client gửi yêu cầu đến server. RESTful API hỗ trợ 4 phương thức chính: GET, POST, GET, DELETE. Những phương thức này phù hợp với các tác vụ cụ thể như lấy thông tin từ server (GET), tạo mớiresource (POST), cập nhật resource (PUT) hay xóa resource (DELETE).
- **Representation:** Mỗi resource có thể có nhiều dạng representation khác nhau như HTML, XML, JSON,… Chúng ta có thể yêu cầu server trả về representation của resource bằng cách sử dụng HTTP header Accept.
- **Status code:** Là mã trạng thái mà server trả về sau khi xử lý yêu cầu từ client. RESTful API sử dụng mã trạng thái HTTP để chỉ ra kết quả của yêu cầu, ví dụ như 200 OK cho thành công, 404 Not Found cho không tìm thấy resource,…
- **Hypermedia:** Là các liên kết giữa các resource. Nó giúp cho việc điều hướng giữa các resource dễ dàng hơn, cũng như giúp cho client có thể tự động tìm kiếm và truy xuất các resource liên quan. Hypermedia có thể được định dạngbằng các định dạng như HTML, Atom hay JSON. [5]

2.4.3 Cách thức RESTful hoạt động

RESTful API hoạt động bằng cách sử dụng phương thức HTTP để truyền tải dữ liệu. Khi có yêu cầu từ ứng dụng web, API sẽ xử lý yêu cầu này và trả về kết quả dưới dạng chuẩn như XML hoặc JSON. Kiến trúc này đã được sử dụng rộng rãi trong việc phát triển các ứng dụng di động và web. RESTful API được xây dựng dựa trên các nguyên tắc thiết kế đơn giản và dễ hiểu. [5]

2.4.4 NodeJs

Hình 2.13 NodeJS

Nodejs là một nền tảng (Platform) phát triển độc lập được xây dựng ở trên Javascript Runtime của Chrome mà chúng ta có thể xây dựng được các ứng dụng mạng một cách nhanh chóng và dễ dàng mở rộng.

Nodejs được xây dựng và phát triển từ năm 2009, bảo trợ bởi công ty Joyent, trụ sở tại California, Hoa Kỳ.

Phần Core bên dưới của Nodejs được viết hầu hết bằng C++ nên cho tốc độ xử lý và hiệu năng khá cao.

Nodejs tạo ra được các ứng dụng có tốc độ xử lý nhanh, realtime thời gian thực.

Nodejs áp dụng cho các sản phẩm có lượng truy cập lớn, cần mở rộng nhanh, cần đổi mới công nghệ, hoặc tạo ra các dự án Startup nhanh nhất có thể. [6]

2.4.5 Những ứng dụng nên viết bằng Nodejs

Rõ ràng, không phải cứ hot và mới là Nodejs làm gì cũng tốt, ví dụ như một ứng dụng cần tính ổn định cao, logic phức tạp thì các ngôn ngữ PHP hay Ruby… vẫn là sự lựa chọn tốt hơn. Còn dưới đây là những ứng dụng có thể và nên viết bằng Nodejs:

- **Websocket server:** Các máy chủ web socket như là Online Chat, Game Server…
- **Fast File Upload Client:** là các chương trình upload file tốc độ cao.
- **Ad Server:** Các máy chủ quảng cáo.
- **Cloud Services:** Các dịch vụ đám mây.
- **RESTful API:** đây là những ứng dụng mà được sử dụng cho các ứng dụng khác thông qua API.
- **Any Real-time Data Application:** bất kỳ một ứng dụng nào có yêu cầu về tốc độ thời gian thực.
- **Micro Services:** Ý tưởng của micro services là chia nhỏ một ứng dụng lớn thành các dịch vụ nhỏ và kết nối chúng lại với nhau. Nodejs có thể làm tốt điều này. [6]

2.4.6 Lý do để tự tin sử dụng Nodejs

Các ứng dụng Nodejs được viết bằng javascript, ngôn ngữ này là một ngôn ngữ khá thông dụng. Theo tác giả của ngôn ngữ Javascript, Ryan Dahl: "Javascript có những đặc tính mà làm cho nó rất khác biệt so với các ngôn ngữ lập trình động còn lại, cụ thể là nó không có khái niệm về đa luồng, tất cả là đơn luồng và hướng sự kiện."

Nodejs chạy đa nền tảng phía Server, sử dụng kiến trúc hướng sự kiện Event-driven, cơ chế non-blocking I/O làm cho nó nhẹ và hiệu quả.

Có thể chạy ứng dụng Nodejs ở bất kỳ đâu trên máy Mac – Window – Linux, hơn nữa cộng đồng Nodejs rất lớn và hoàn toàn miễn phí. [2]

Người dùng có thể thấy cộng đồng Nodejs lớn như thế nào tại đây, các package đều hoàn toàn free

Các ứng dụng NodeJS đáp ứng tốt thời gian thực và chạy đa nền tảng. [6]

2.4.7 Kiến thức về framework ExpressJs

ExpressJS là một framework miễn phí và mã nguồn mở dành cho việc xây dựng ứng dụng web trên nền tảng Node.js. Với ExpressJS, người dùng có thể nhanh chóng thiết kế và phát triển các ứng dụng web và API một cách dễ dàng và tiện lợi. Điều này đặc biệt hữu ích cho các nhà phát triển và lập trình viên đã quen thuộc với JavaScript.

Vì ExpressJS là một framework phát triển ứng dụng web trên Nodejs, các lập trình viên có thể sử dụng mã đã có sẵn để xây dựng các ứng dụng web đơn trang (SPA), đa trang hoặc kết hợp cả hai. Ngoài ra, ExpressJS còn cung cấp một kiến trúc MVC (Model-View-Controller) hữu ích để tổ chức các ứng dụng web phía máy chủ.

Với các tính năng hỗ trợ nâng cao của Nodejs, ExpressJS giúp giảm bớt sự phức tạp của việc xây dựng API hiệu quả. Nếu không sử dụng ExpressJS, lập trình viên sẽ phải viết rất nhiều mã để xử lý các yêu cầu khác nhau.

ExpressJS giúp cho việc này trở nên đơn giản hơn đáng kể. Với sự hỗ trợ từ cộng đồng và các thư viện bên thứ ba, ExpressJS là một công cụ hữu ích cho việc xây dựng các ứng dụng web và API trên Nodejs. [6]

2.4.8 ExpressJs được dùng cho các công việc

ExpressJS là một framework giúp cho việc tổ chức kiến trúc back-end trở nên dễ dàng hơn. Đặc biệt, những nhà phát triển web thường sử dụng ExpressJS để triển khai kiến trúc MVC, từ đó giúp cho việc viết mã back-end trở nên dễ bảo trì hơn.

Vì ExpressJS hoạt động trên backend, nên nó có thể được coi như một “bộ não” đằng sau một trang web. Nó có thể xác định cách những trang được định tuyến trên một trang web và cung cấp các chức năng quản lý xác thực cho một trang web.

Một số trường hợp sử dụng phổ biến của ExpressJS bao gồm:

- Sử dụng cookie trên một trang web nào đó
- Triển khai xác thực trên một trang web

- Tạo thêm thanh tìm kiếm vào một trang web
- Cung cấp thêm các tệp tĩnh như hình ảnh vào đó

Với ExpressJS, các nhà phát triển có thể dễ dàng thêm các tính năng như xử lý yêu cầu HTTP, xác thực người dùng và quản lý định tuyến trong ứng dụng web của họ. Tất cả những điều này làm cho việc phát triển ứng dụng web trở nên nhanh chóng và dễ dàng hơn, đặc biệt là với những lập trình viên đã quen với JavaScript.

2.4.9 Tính năng và lợi ích của ExpressJs

❖ **Tính năng của ExpressJs**

Hình 2.14 Tính năng của ExpressJs

ExpressJS có thể giúp phát triển máy chủ nhanh hơn bằng việc cung cấp các tính năng phổ biến của Nodejs dưới dạng hàm có thể tái sử dụng. Ngoài ra, ExpressJS cũng đóng vai trò là phần mềm trung gian, giúp tổ chức các chức năng khác nhau của ứng dụng.

ExpressJS cũng cung cấp một cơ chế định tuyến nâng cao có thể giúp duy trì vững trạng thái của trang web. Nó cũng cung cấp các công cụ tạo khuôn mẫu cho phép các nhà phát triển tạo được nội dung động trên các trang web này bằng việc bắt đầu xây dựng các mẫu HTML ở phía máy chủ.

Gỡ lỗi là yếu tố quan trọng để phát triển các ứng dụng web và ExpressJS cung cấp cơ chế giúp xác định chính xác phần ứng dụng web có lỗi, giúp cho quá trình gỡ lỗi trở nên dễ dàng hơn. [7]

❖ **Lợi ích của ExpressJs**

ExpressJS là một công nghệ rất dễ học và sử dụng nếu người dùng đã biết JavaScript. Điều này giúp cho việc phát triển back-end dễ dàng hơn rất nhiều. Bởi vì

mã JavaScript được diễn giải thông qua Google V8 JavaScript Engine của Nodejs, ExpressJS cho phép mã được thực hiện một cách nhanh chóng và hiệu quả.

Một trong những điểm mạnh của ExpressJS là tính linh hoạt cao và dễ dàng tùy chỉnh theo nhu cầu của người dùng. ExpressJS cung cấp một module phần mềm trung gian linh hoạt và rất hữu ích để thực hiện các tác vụ bổ sung theo phản hồi và yêu cầu.

Ngoài ra, ExpressJS hỗ trợ phát triển ứng dụng theo mô hình MVC, một mô hình phổ biến cho việc lập trình web hiện nay. Điều này giúp cho việc phát triển ứng dụng được tổ chức và dễ bảo trì hơn. [7]

2.4.10 Tailwind CSS

Hình 2.15 Tailwind CSS

Tailwind CSS là một framework CSS theo hướng utility-first, được phát triển bởi Adam Wathan và phát hành lần đầu vào tháng 10/2019. Nó giúp xây dựng giao diện người dùng một cách nhanh chóng và linh hoạt bằng cách cung cấp các lớp tiện ích (utility classes) thay vì các component cố định. [8]

2.4.11 Ưu điểm của Tailwind Css

- Tăng tốc độ phát triển nhờ khai báo trực tiếp các lớp CSS trên HTML.
- Giảm kích thước file CSS bằng cách chỉ xuất ra các lớp đã sử dụng.
- Hỗ trợ thiết kế Responsive ngay trên file HTML.
- Cung cấp các lớp CSS cho các trạng thái tương tác như hover, focus.
- Khuyến khích sử dụng các lớp tái sử dụng, giảm thiểu mã lặp.
- Dễ dàng mở rộng, chỉnh sửa và tùy biến.
- Hỗ trợ chế độ tối (Dark Mode).

- Tích hợp tốt với nhiều framework front-end khác như React, Vue.js. [8]

2.4.12 Nhược điểm của Tailwind Css

- Số lượng mã HTML có thể trở nên rất dài và khó quản lý.
- Cần có kiến thức vững về CSS để sử dụng hiệu quả.
- Tốn thời gian để làm quen và nhớ các tên lớp. [8]

2.4.13 Hệ quản trị cơ sở dữ liệu MonggoDB

Hình 2.16 MogoDB

MongoDB là một cơ sở dữ liệu tài liệu đa nền tảng. Được phân loại là cơ sở dữ liệu NoSQL, MongoDB sử dụng lược đồ giống tài liệu JSON. MongoDB được phát triển bởiMongoDB Inc. và được cấp phép theo Giấy phép Công cộng phía Máy chủ (SSPL).

MongoDB lưu trữ dữ liệu trong các tài liệu linh hoạt, giống như JSON, có nghĩa là các trường có thể thay đổi từ tài liệu này sang tài liệu khác và cấu trúc dữ liệu có thể được thay đổi theo thời gian.

Mô hình tài liệu ánh xạ tới các đối tượng trong mã ứng dụng của người dùng, giúp dữ liệu dễ dàng làm việc với.

- Truy vấn đặc biệt, lập chỉ mục và tổng hợp thời gian thực cung cấp các cách mạnh mẽ để truy cập và phân tích dữ liệu của người dùng.
- MongoDB là một cơ sở dữ liệu phân tán ở cốt lõi của nó, vì vậy tính sẵn sàng cao, tỷ lệ ngang và phân phối địa lý được xây dựng và dễ sử dụng.

MongoDB là miễn phí và nguồn mở. Các phiên bản được phát hành trước ngày 16 tháng 10 năm 2018 được xuất bản theo AGPL. Tất cả các phiên bản được phát hành sau ngày 16 tháng 10 năm 2018. [9]

2.5 Tổng quan về các hệ thống chấm điểm tự động hiện nay

2.5.1 Lý do nên chọn hệ thống chấm điểm tự động

Việc lựa chọn hệ thống chấm điểm tự động có thể cải thiện đáng kể hiệu quả và tính nhất quán trong các bối cảnh giáo dục. Các hệ thống này xử lý khối lượng lớn đánh giá một cách "nhanh chóng", cho phép các nhà giáo dục dành nhiều thời gian hơn cho việc giảng dạy tương tác và hỗ trợ học sinh. Bằng cách áp dụng các tiêu chí thống nhất cho bài làm của mọi học sinh, chấm điểm tự động làm giảm nguy cơ thiên vị và không nhất quán có thể xảy ra trong quá trình chấm điểm thủ công. Phản hồi ngay lập tức cũng giúp học sinh hiểu được hiệu suất của mình một cách nhanh chóng, thúc đẩy môi trường học tập phản ứng nhanh hơn.

Việc lựa chọn hệ thống chấm điểm tự động mang lại một số lợi thế có thể cải thiện đáng kể quá trình giáo dục. Ở đây, hệ thống chấm điểm tự động được cho là có một số lợi thế sẽ làm cho quá trình giáo dục thậm chí còn hiệu quả hơn. Sau đây là tâm lý do cần thiết nhất: [10]

- **Tốc độ**

Hệ thống chấm điểm tự động có thể nhanh chóng đánh giá một số lượng lớn bài tập cho giáo viên và học sinh. Việc cung cấp phản hồi nhanh chóng có tầm quan trọng lớn đối với quá trình học tập.

- **Sự nhất quán**

Hệ thống tự động áp dụng cùng một tiêu chí thống nhất cho tất cả các bài tập đã nộp. Sự thống nhất này loại bỏ mọi định kiến và điểm số một cách công bằng và khách quan.

- **Khả năng mở rộng**

Nó sẽ chạy hiệu quả khi nhiều sinh viên cung cấp một lớp học nhất định hoặc một tổ chức có nhiều sinh viên. Họ có thể chấm điểm hàng chục nghìn bài tập và bài tập của sinh viên mà không gây thêm căng thẳng và tác động tốn kém.

- **Phản hồi chi tiết**

Các hệ thống phần mềm tinh vi có thể cung cấp mô tả chi tiết hơn về cách học sinh thực hiện và do đó xác định được những lĩnh vực tốt và những lĩnh vực học sinh

còn chậm. Nó cũng có thể rộng hơn so với những gì phản hồi truyền thống có thể cung cấp theo cách thủ công.

- **Hiệu quả về chi phí**

Chi phí hành chính trong kiểm tra có thể được giảm đáng kể nhờ hệ thống chấm điểm tự động, vì việc sử dụng chúng làm giảm số lượng người chấm điểm cần thiết. Điều này đặc biệt hữu ích trong các bối cảnh lớn với nhiều học sinh.

- **Khả năng thích nghi**

Các hệ thống hiện đại có thể được thiết kế để xử lý các loại câu hỏi trắc nghiệm, trả lời ngắn và bài luận trong đánh giá. Hơn nữa, chúng có thể thích ứng để phù hợp với các tiêu chí và tiêu chuẩn khác nhau được áp dụng trong quá trình này.

- **Phân tích dữ liệu**

Hệ thống chấm điểm tự động thường đi kèm với các công cụ tích hợp hỗ trợ các nhà giáo dục theo dõi xu hướng về thành tích của học sinh, do đó chỉ ra những lĩnh vực mà học sinh có nhiều khả năng gặp khó khăn nhất. Điều này đảm bảo rằng các biện pháp can thiệp được nhắm mục tiêu theo cách tiếp cận dựa trên dữ liệu như vậy.

- **Giảm thiểu sai lệch và lỗi**

Do đó, hệ thống tự động sẽ cắt giảm mức độ lỗi của con người và sự thiên vị chủ quan liên quan đến việc chấm điểm. Theo cách này, mọi học sinh đều được đánh giá theo các tiêu chí và tiêu chuẩn đã cho. [10]

2.5.2 Lịch sử phát triển của AI chấm điểm

Lịch sử của hệ thống chấm điểm tự động có từ thời kỳ hình thành công nghệ giáo dục. Trên thực tế, mặc dù ở dạng đơn giản, nó bắt đầu với các phương pháp chấm điểm dựa trên máy vào đầu và giữa thế kỷ 20. Các hệ thống ban đầu sử dụng thẻ đục lỗ hoặc công nghệ nhận dạng điểm quang học (OMR) để tự động chấm điểm các bài kiểm tra trắc nghiệm. Các hệ thống này, mang tính sáng tạo vào thời điểm đó, lại bị hạn chế về khả năng, tập trung vào đánh giá khách quan và nhu cầu thiết lập và bảo trì thủ công.

Hệ thống chấm điểm tự động đã đi một chặng đường dài từ những ngày đầu khi chúng bắt đầu là những công cụ rất đơn giản để trở thành một số công nghệ AI tiên tiến. Các hệ thống chấm điểm ban đầu chủ yếu dựa trên các phương pháp đơn giản như bảng chấm điểm. Đây là những tờ giấy mà học sinh sẽ trả lời các câu hỏi bằng cách tô màu hoặc tô vào các vòng tròn. Những tờ giấy này sau đó sẽ được

quét và chấm điểm bằng các thuật toán đơn giản có khả năng chấm điểm chỉ các câu trả lời trắc nghiệm.

Với những tiến bộ to lớn trong trí tuệ nhân tạo và máy học trong nhiều thập kỷ qua, các hệ thống chấm điểm tự động cũng đã chuyển đổi thành các hình thức đánh giá phức tạp và đa dạng hơn nhiều. Các hệ thống AI hiện đại hiện có thể chấm điểm các câu hỏi trả lời ngắn hoặc thậm chí là câu trả lời dài hoặc các hình thức bài tập khác lên đến mức độ khó tăng dần trong đánh giá, như bài tập lập trình. Các hệ thống này có thể sử dụng xử lý ngôn ngữ tự nhiên học sâu để đánh giá nội dung hiệu quả hơn và dài hơn, với phản hồi chi tiết cung cấp thông tin chi tiết được cá nhân hóa. Nhìn chung, các xu hướng này đại diện cho những thay đổi hướng tới các công cụ giáo dục được tích hợp AI trong tự nhiên, do đó tăng khả năng mở rộng với độ chính xác cao hơn trong việc chấm điểm và đồng thời liên tục cải thiện khả năng chấm điểm các phần sắc thái hơn trong các bài tập của học sinh.

Khi công nghệ tiến triển, hệ thống chấm điểm cũng vậy. Các chương trình máy tính được tích hợp, do đó, có thể thực hiện các đánh giá phức tạp hơn, chẳng hạn như các câu hỏi trả lời ngắn. Trong các hệ thống này, việc kiểm tra câu trả lời được thực hiện bằng cách khớp từ khóa. Các học sinh khác nhau có thể viết cùng một câu trả lời bằng các từ khác nhau.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1. Kiến trúc tổng thể

3.1.1 Mô hình Kiến trúc MVC (Model-View-Controller)

Hệ thống được xây dựng dựa trên mô hình kiến trúc MVC, một mô hình phổ biến trong phát triển ứng dụng web, nhằm mục đích phân tách rõ ràng các thành phần và logic nghiệp vụ của ứng dụng. Sự phân tách này mang lại nhiều lợi ích như tăng cường khả năng bảo trì, dễ dàng quản lý mã nguồn, hỗ trợ phát triển song song và tái sử dụng mã nguồn hiệu quả.

- **Models (/models):** Các Models chịu trách nhiệm quản lý dữ liệu và logic nghiệp vụ liên quan trực tiếp đến cơ sở dữ liệu. Chúng tương tác với cơ sở dữ liệu MongoDB thông qua Mongoose ODM để định nghĩa schema, thực hiện các thao tác truy vấn (CRUD - Create, Read, Update, Delete), kiểm tra tính hợp lệ của dữ liệu (validation), và định nghĩa các mối quan hệ giữa các thực thể dữ liệu. Các models chính được xây dựng bao gồm: User, ReadingPassage, Question, Test và UserAttempt.
- **Views (/views):** Phần Views đảm nhiệm việc hiển thị giao diện người dùng (UI - User Interface). Hệ thống sử dụng EJS (Embedded JavaScript templating) kết hợp với express-ejs-layouts để tạo ra các trang HTML động, hiển thị dữ liệu được truyền từ Controllers. Tailwind CSS được sử dụng để áp dụng styling, giúp xây dựng giao diện người dùng nhanh chóng và nhất quán. Cấu trúc thư mục views được tổ chức theo từng module để dễ dàng quản lý.
- **Controllers (/controllers):** Controllers đóng vai trò trung gian, điều phối luồng dữ liệu và xử lý logic nghiệp vụ chính của ứng dụng. Chúng tiếp nhận các yêu cầu HTTP từ người dùng (thông qua Routes), tương tác với Models để truy xuất, xử lý hoặc cập nhật dữ liệu từ cơ sở dữ liệu. Sau khi xử lý, Controllers sẽ lựa chọn View thích hợp và truyền dữ liệu cần thiết cho View để hiển thị cho người dùng, hoặc trả về phản hồi dưới dạng JSON cho các yêu cầu API.
- **Routes (/routes):** Các Routes chịu trách nhiệm định tuyến các URL của ứng dụng đến các hàm xử lý tương ứng trong Controllers. Mỗi file route định nghĩa một tập hợp các endpoints, ánh xạ mỗi URL và phương thức HTTP (GET, POST, PUT, DELETE) tới một action cụ thể trong Controller. Các Routes cũng có thể bao gồm các middleware để thực hiện các tác vụ trung gian như xác thực người dùng, kiểm tra quyền truy cập trước khi yêu cầu được chuyển đến Controller.

3.1.2. Công nghệ sử dụng và Lý do lựa chọn

Việc lựa chọn công nghệ được cân nhắc kỹ lưỡng để đáp ứng các yêu cầu về hiệu năng, khả năng mở rộng và tính bảo mật.

❖ **Backend - Node.js + Express.js:**

- **Node.js:** Được lựa chọn là nền tảng JavaScript runtime xây dựng trên Chrome V8 engine, cho phép chạy JavaScript phía server. Node.js sở hữu mô hình I/O không đồng bộ (non-blocking I/O) và hướng sự kiện (event-driven), rất hiệu quả cho các ứng dụng web có nhiều kết nối đồng thời và thao tác đọc/ghi dữ liệu như hệ thống luyện thi này. Cộng đồng phát triển lớn mạnh và kho thư viện npm phong phú cung cấp nhiều giải pháp sẵn có.
- **Express.js:** Là một framework web tối giản, linh hoạt và phổ biến nhất cho Node.js. Express.js cung cấp các công cụ mạnh mẽ để xây dựng web server, định tuyến (routing), quản lý middleware, giúp đơn giản hóa và tăng tốc quá trình phát triển backend. Framework này đặc biệt phù hợp cho việc xây dựng các RESTful API.

❖ **Database - MongoDB + Mongoose ODM:**

- **MongoDB:** Là một cơ sở dữ liệu NoSQL hướng tài liệu (document-oriented). MongoDB được lựa chọn vì khả năng lưu trữ dữ liệu linh hoạt, không yêu cầu schema cố định, rất phù hợp với việc quản lý các loại câu hỏi đa dạng và có cấu trúc khác nhau trong kỹ năng Reading. Khả năng mở rộng theo chiều ngang (horizontal scaling) tốt, đáp ứng nhu cầu tăng trưởng dữ liệu và lưu lượng truy cập trong tương lai.
- **Mongoose ODM:** Một thư viện Object Data Modeling (ODM) cho MongoDB và Node.js. Mongoose cung cấp một giải pháp dựa trên schema để mô hình hóa dữ liệu ứng dụng, tích hợp sẵn tính năng validation, casting kiểu dữ liệu, middleware cho schema, query building và các business logic hooks. Điều này giúp việc tương tác với MongoDB trở nên có cấu trúc, dễ quản lý và ít lỗi hơn.

❖ **Template Engine - EJS với Express-EJS-Layouts:**

- **EJS (Embedded JavaScript):** Một template engine đơn giản cho phép nhúng mã JavaScript trực tiếp vào trong các file HTML. EJS giúp việc render dữ liệu động từ phía server trở nên dễ dàng và trực quan, đặc biệt khi đội ngũ phát triển đã quen thuộc với JavaScript.

- **Express-EJS-Layouts:** Một middleware cho Express.js giúp quản lý các layout chung (ví dụ: header, footer, sidebar) cho các trang EJS một cách hiệu quả. Điều này giúp tránh lặp lại mã HTML ở nhiều trang và dễ dàng duy trì cấu trúc giao diện nhất quán.

❖ **Styling - Tailwind CSS:** Tailwind CSS là một framework CSS utility-first, cung cấp một tập hợp lớn các lớp CSS tiện ích có thể tái sử dụng. Nó cho phép xây dựng giao diện người dùng một cách nhanh chóng trực tiếp trong HTML mà không cần viết nhiều mã CSS tùy chỉnh. Tailwind CSS hỗ trợ mạnh mẽ cho việc tạo responsive design và dễ dàng tùy biến giao diện theo yêu cầu.

❖ **Authentication - JWT + bcrypt + express-session:**

- **JWT (JSON Web Tokens):** Một chuẩn mở (RFC 7519) được sử dụng để truyền tải thông tin một cách an toàn giữa các bên dưới dạng một đối tượng JSON đã được ký số. Thường được dùng cho việc xác thực API, đặc biệt trong các kiến trúc stateless.
- **bcrypt:** Một thư viện phổ biến và đáng tin cậy để mã hóa (hashing) mật khẩu. bcrypt sử dụng thuật toán Blowfish với một yếu tố chi phí (cost factor) có thể điều chỉnh, giúp tạo ra các hash mạnh, chống lại các cuộc tấn công brute-force và rainbow table.
- **express-session:** Middleware cho Express.js dùng để quản lý phiên làm việc của người dùng. Nó cho phép lưu trữ dữ liệu phiên ở phía server và gửi một cookie chứa session ID về phía client, giúp duy trì trạng thái đăng nhập và lưu trữ thông tin tạm thời qua các request.

❖ **Session Storage - connect-mongo (MongoDB session store):** connect-mongo là một module cho phép lưu trữ dữ liệu phiên của express-session trực tiếp vào trong cơ sở dữ liệu MongoDB, thay vì lưu trữ trong bộ nhớ (mặc định của express-session). Điều này rất quan trọng trong môi trường production, đặc biệt khi ứng dụng được triển khai trên nhiều instance (ví dụ: sử dụng load balancer), đảm bảo tính nhất quán và bền bỉ của phiên làm việc ngay cả khi server khởi động lại.

3.2. Các mô-đun chức năng

Hệ thống được thiết kế theo hướng module hóa, chia thành các khối chức năng chính để đảm bảo tính tổ chức, dễ phát triển, bảo trì và nâng cấp. Mỗi module đảm nhiệm một tập hợp các tác vụ và logic nghiệp vụ cụ thể.

3.2.1. Mô-đun Xác thực (Authentication Module)

❖ **Mục tiêu chính:** Quản lý toàn bộ quy trình liên quan đến việc đăng ký, đăng nhập, đăng xuất và phân quyền người dùng trong hệ thống.

❖ **Các chức năng cụ thể và luồng xử lý:**

- **Đăng ký tài khoản người dùng:** Giao diện (views/auth/register.ejs) cung cấp form cho người dùng nhập thông tin (username, email, password, fullName). routes/auth.js định nghĩa endpoint (ví dụ: POST /register) để nhận dữ liệu từ form. authController.js xử lý yêu cầu bằng cách kiểm tra tính hợp lệ của dữ liệu (validation), kiểm tra username/email đã tồn tại trong User Collection, mã hóa mật khẩu người dùng bằng thư viện bcrypt trước khi lưu trữ, tạo một bản ghi người dùng mới trong User Collection với role mặc định là 'user', và phản hồi kết quả đăng ký cho người dùng.
- **Đăng nhập/Đăng xuất:**
  - **Đăng nhập:** Giao diện (views/auth/login.ejs) cung cấp form cho người dùng nhập username (hoặc email) và password. routes/auth.js định nghĩa endpoint (ví dụ: POST /login). authController.js xử lý yêu cầu bằng cách tìm kiếm người dùng trong User Collection, so sánh mật khẩu bằng bcrypt.compare(), tạo một phiên làm việc mới cho người dùng sử dụng express-session (lưu trữ userId và role), và chuyển hướng người dùng đến trang dashboard tương ứng.
  - **Đăng xuất:** routes/auth.js định nghĩa endpoint (ví dụ: GET /logout). authController.js hủy phiên làm việc hiện tại của người dùng (req.session.destroy()) và chuyển hướng người dùng về trang đăng nhập hoặc trang chủ.
- **Xác thực phiên làm việc:** Một middleware xác thực được sử dụng trong routes để kiểm tra sự tồn tại và tính hợp lệ của session người dùng (req.session.user hoặc req.session.userId) cho các trang hoặc API yêu cầu đăng nhập. Nếu không hợp lệ, người dùng sẽ bị chuyển hướng về trang đăng nhập.
- **Phân quyền người dùng (user/admin):** Sau khi xác thực phiên, middleware phân quyền sẽ kiểm tra vai trò (role) của người dùng được lưu trong session. Dựa vào role ('user' hoặc 'admin'), middleware sẽ quyết định người dùng có được phép truy cập vào một tài nguyên hoặc thực hiện một hành động cụ thể hay không (ví dụ, chỉ người dùng có role là 'admin' mới được truy cập vào các routes trong routes/admin.js).

- **Mã hóa mật khẩu với bcrypt:** Khi đăng ký hoặc thay đổi mật khẩu, thư viện bcrypt được sử dụng để tạo ra một chuỗi hash từ mật khẩu gốc. Chuỗi hash này, cùng với "salt" được tạo tự động, sẽ được lưu trữ trong cơ sở dữ liệu thay vì mật khẩu gốc, giúp bảo vệ thông tin người dùng ngay cả khi cơ sở dữ liệu bị xâm nhập.

3.2.2. Mô-đun Quản trị (Admin Module)

❖ **Mục tiêu chính:** Cung cấp cho quản trị viên một giao diện tập trung và các công cụ mạnh mẽ để quản lý toàn bộ nội dung học liệu (đoạn văn, câu hỏi, bài test) và theo dõi, phân tích hoạt động của hệ thống cũng như kết quả học tập của học viên.

❖ **Các chức năng cụ thể và luồng xử lý:**

- **Dashboard quản trị tổng quan:** routes/admin.js định nghĩa endpoint cho trang dashboard (ví dụ: GET /admin/dashboard). adminController.js truy vấn và tổng hợp dữ liệu từ nhiều collection (ví dụ: số lượng người dùng, số lượng bài test, số lượt làm bài gần đây) để chuẩn bị dữ liệu cho view. views/admin/dashboard.ejs hiển thị các thông tin này dưới dạng số liệu thống kê, biểu đồ, cung cấp cái nhìn tổng quan về tình hình hệ thống.
- **Quản lý đoạn văn đọc (Reading Passages):** adminController.js xử lý các thao tác CRUD (Create, Read, Update, Delete) cho ReadingPassage Collection. Chức năng bao gồm tạo mới (form nhập title và content), xem danh sách (hiển thị thông tin cơ bản, tùy chọn sửa/xóa), và cập nhật/xóa một đoạn văn.
- **Tạo và quản lý câu hỏi (Questions) với 5 loại:** adminController.js chứa logic phức tạp để xử lý việc tạo và cập nhật các loại câu hỏi khác nhau, đảm bảo dữ liệu được lưu trữ chính xác theo questionType. Giao diện cho phép chọn questionType (Multiple Choice, Fill in the Blank, Matching, True/False/Not Given, Short Answer). Dựa trên questionType được chọn, form sẽ hiển thị động các trường nhập liệu đặc thù. Mỗi câu hỏi được liên kết với một passageId (nếu thuộc đoạn văn cụ thể) và có score, order.
- **Quản lý bài test và test pools:** Quản lý Bài Test: adminController.js cho phép admin tạo một Test mới, bao gồm: đặt title, description, timeLimit, totalScore. Quan trọng nhất là việc lựa chọn các ReadingPassages và sau đó chọn các Questions cụ thể (thông qua ID) từ những passages đó để đưa vào cấu trúc passages: [{ passageId, questions: [questionId] }] của Test Collection. Admin có thể kích hoạt (active: true) hoặc hủy kích hoạt bài test.

- **Thống kê và báo cáo kết quả học viên:** adminController.js truy vấn UserAttempt Collection và các collection liên quan (User, Test, Question) để tổng hợp dữ liệu. Cung cấp các báo cáo về: kết quả làm bài của từng học viên, thống kê hiệu suất theo từng bài test, và phân tích độ khó của câu hỏi. Dữ liệu có thể được hiển thị dưới dạng bảng biểu hoặc biểu đồ.

3.2.3. Mô-đun Học viên (User Module)

❖ **Mục tiêu chính:** Cung cấp một môi trường trực quan, tương tác và hiệu quả để học viên thực hành kỹ năng đọc hiểu tiếng Anh, theo dõi tiến độ và nhận phản hồi chi tiết về kết quả làm bài.

❖ **Các chức năng cụ thể và luồng xử lý:**

- **Dashboard cá nhân:** Sau khi đăng nhập, userController.js lấy thông tin cá nhân của người dùng và các dữ liệu liên quan đến quá trình học tập (từ UserAttempt Collection). views/user/dashboard.ejs hiển thị thông tin tóm tắt: số bài test đã hoàn thành, tỷ lệ làm đúng trung bình, tổng thời gian luyện tập. Có thể có biểu đồ thể hiện sự tiến bộ điểm số qua các lần làm bài, và danh sách các bài test đang làm dở hoặc các bài test mới/gợi ý.
- **Danh sách bài test có sẵn:** userController.js truy vấn Test Collection (hoặc TestPool Collection) để lấy danh sách các bài test đang ở trạng thái active. views/user/tests.ejs (hoặc tương tự) hiển thị danh sách này, mỗi bài test có thể kèm theo thông tin: tiêu đề, mô tả ngắn, timeLimit, độ khó (nếu có). Nút "Bắt đầu làm bài" được cung cấp cho mỗi bài test.
- **Làm bài test với timer:** Khi người dùng chọn "Bắt đầu làm bài", userController.js tạo một bản ghi UserAttempt mới và tải toàn bộ cấu trúc của Test. views/user/attempt.ejs hiển thị giao diện làm bài với nội dung ReadingPassage (nếu có), câu hỏi hiện tại, các trường/lựa chọn để nhập/chọn câu trả lời, và đồng hồ đếm ngược (timer) dựa trên timeLimit của Test.
- **Xem kết quả và phân tích chi tiết:** Khi người dùng nộp bài (hoặc timer hết giờ), userController.js thực hiện quá trình chấm điểm tự động cho UserAttempt. Bản ghi UserAttempt được cập nhật với status: "completed", completedAt, score, percentageScore, và chi tiết các answers. views/user/result.ejs hiển thị tóm tắt kết quả, phân tích chi tiết từng câu (câu trả lời của người dùng, đáp án đúng, giải thích), và đánh dấu rõ ràng câu trả lời đúng/sai.

- **Lịch sử làm bài:** userController.js truy vấn UserAttempt Collection để lấy danh sách tất cả các lượt làm bài của người dùng hiện tại, sắp xếp theo ngày làm. views/user/history.ejs (hoặc tương tự) hiển thị danh sách này, mỗi mục có thông tin tóm tắt và liên kết để xem lại chi tiết kết quả.
- **Thống kê tiến độ học tập:** Có thể là một phần của User Dashboard hoặc một trang riêng. userController.js tổng hợp dữ liệu từ các UserAttempt của người dùng để tạo biểu đồ hoặc bảng thống kê về: sự thay đổi điểm số trung bình qua thời gian, tỷ lệ làm đúng theo từng questionType, và thời gian làm bài trung bình.

3.2.4. Mô-đun Xử lý Bài Test (Test Engine)

❖ **Mục tiêu chính:** Đảm bảo toàn bộ quy trình làm bài test của học viên, từ khi bắt đầu cho đến khi nhận kết quả, diễn ra một cách chính xác, mượt mà và đáng tin cậy.

❖ **Các thành phần logic và quy trình xử lý chính:**

- **Hệ thống timer tự động:** Khi trang làm bài (views/user/attempt.ejs) được tải, JavaScript phía client sẽ đọc giá trị timeLimit của bài test và khởi tạo một bộ đếm ngược hiển thị trên giao diện. Client-side sử dụng setInterval() để cập nhật timer mỗi giây. Khi timer về 0, JavaScript sẽ tự động kích hoạt hành động nộp bài. Server-side (userController.js) cũng tính toán thời gian làm bài và có thể có logic để xử lý các trường hợp nộp bài muộn.
- **Lưu câu trả lời real-time (hoặc cơ chế lưu thường xuyên để tránh mất dữ liệu):** JavaScript phía client gắn các event listener vào các trường nhập liệu/lựa chọn câu trả lời. Khi một event được kích hoạt, JavaScript sẽ thu thập questionId và userAnswer, sau đó gửi một AJAX request đến server (ví dụ: POST/api/user/attempts/:attemId/answers/:questionId). Endpoint này sẽ xác thực yêu cầu, tìm bản ghi UserAttempt và cập nhật câu trả lời vào mảng answers của UserAttempt.
- **Chấm điểm tự động (Logic trong userController.js khi người dùng nộp bài):** Quá trình này được kích hoạt khi người dùng nhấn nút "Nộp bài" hoặc khi timer hết giờ. userController.js lấy bản ghi UserAttempt hiện tại, duyệt qua từng câu trả lời đã lưu, truy xuất thông tin Question tương ứng (bao gồm questionType, correctAnswer, score), và thực hiện so sánh userAnswer với correctAnswer tùy theo loại câu hỏi. Dựa trên kết quả so sánh, các trường isCorrect và scoreAchieved cho từng câu trả lời được cập nhật. Sau đó, tính toán các giá trị tổng hợp cho UserAttempt (tổng

điểm, tỷ lệ phần trăm) và cập nhật status thành "completed". Cuối cùng, bản ghi UserAttempt đã được chấm điểm hoàn chỉnh được lưu vào cơ sở dữ liệu.

- **Phân tích kết quả chi tiết để hiển thị cho người dùng:** Sau khi chấm điểm và lưu UserAttempt, userController.js chuẩn bị một đối tượng dữ liệu đầy đủ bao gồm thông tin về Test, UserAttempt (với các câu trả lời đã chấm điểm), và chi tiết của từng Question (bao gồm questionText, options, correctAnswer gốc, và explanation). Dữ liệu này được truyền đến template view (views/user/result.ejs) để hiển thị một báo cáo kết quả rõ ràng, chi tiết và hữu ích cho học viên.

3.3. Thiết kế Cơ sở dữ liệu và Giao diện

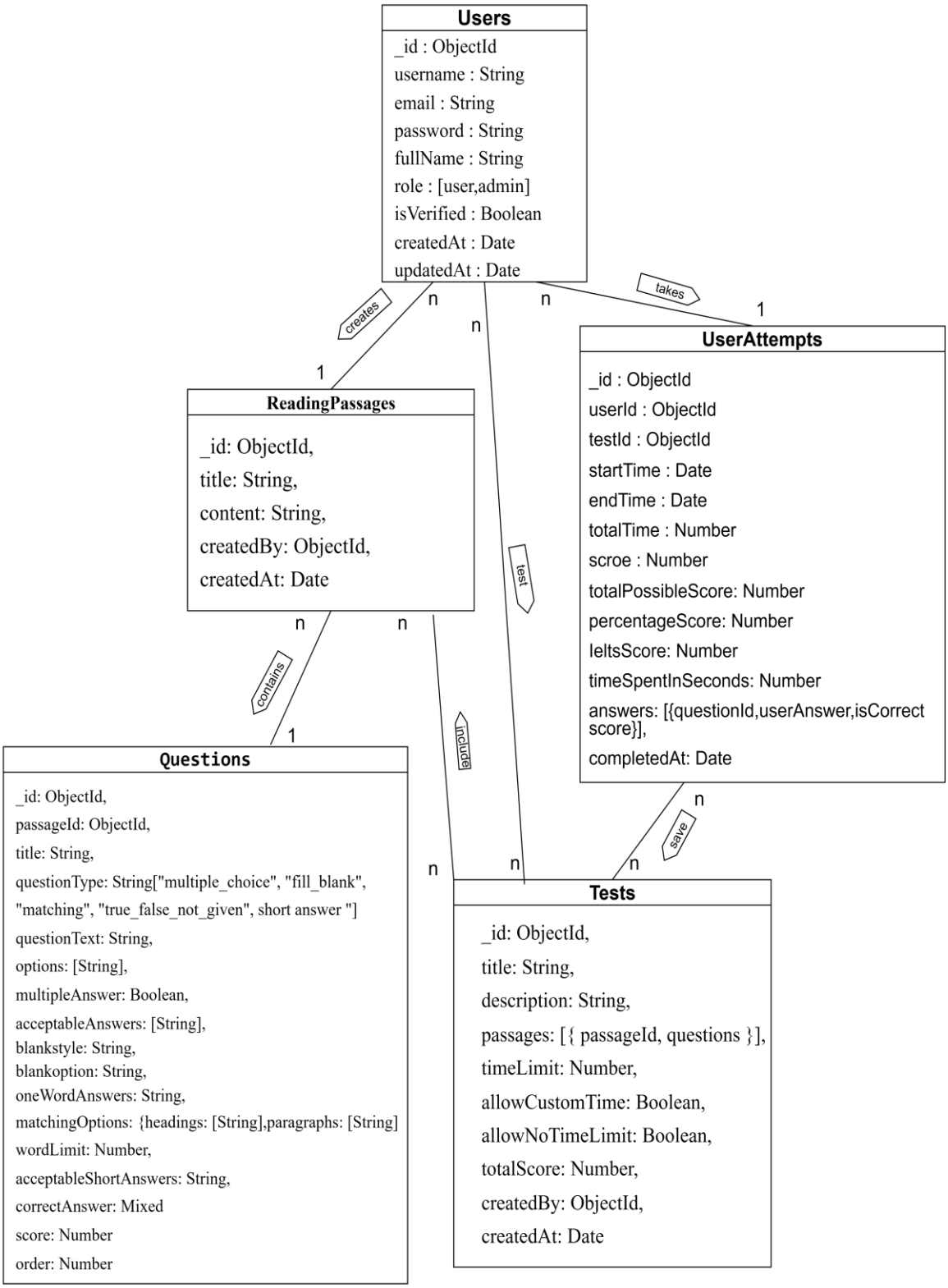
3.3.1. Thiết kế Cơ sở dữ liệu

Lựa chọn MongoDB làm hệ quản trị cơ sở dữ liệu NoSQL mang lại sự linh hoạt trong việc định nghĩa và thay đổi schema, đặc biệt phù hợp với yêu cầu lưu trữ đa dạng các loại câu hỏi và dữ liệu phi cấu trúc/bán cấu trúc khác của hệ thống.

3.3.1.1. Sơ đồ Quan hệ Thực thể (Entity Relationship Diagram - ERD - Dạng khái niệm cho NoSQL)

Mặc dù MongoDB là NoSQL và không có các mối quan hệ ràng buộc (enforced relationships) như trong cơ sở dữ liệu quan hệ, việc hình dung một sơ đồ ERD khái niệm giúp làm rõ mối liên kết logic giữa các collection thông qua các trường tham chiếu (ObjectId references).

(Mô hình cơ sở dữ liệu trang sau)



Hình 3.1 Mô hình cơ sơ dữ liệu

Mô tả các thực thể chính và mối liên kết logic:

- **User:** Thực thể trung tâm, lưu trữ thông tin người dùng. Một User có thể tạo ra nhiều ReadingPassage, Question, Test (nếu là Admin) và thực hiện nhiều UserAttempt.

- **ReadingPassage:** Lưu trữ nội dung các đoạn văn đọc. Một ReadingPassage có thể chứa nhiều Question.
- **Question:** Lưu trữ chi tiết từng câu hỏi. Một Question thường thuộc về một ReadingPassage và có thể xuất hiện trong nhiều Test.
- **Test:** Định nghĩa một bài kiểm tra hoàn chỉnh. Một Test được cấu thành từ một hoặc nhiều ReadingPassage và một tập hợp các Question được chọn từ những passage đó. Một Test có thể thuộc về nhiều TestPool và có nhiều UserAttempt.
- **UserAttempt:** Ghi lại một lượt làm bài cụ thể của một người dùng đối với một bài test. Mỗi UserAttempt thuộc về một User và một Test, chứa danh sách các câu trả lời của người dùng.

3.3.1.2. Mô tả chi tiết các Collections và Schema

Dưới đây là mô tả chi tiết về cấu trúc (schema) của từng collection chính trong MongoDB, sử dụng Mongoose ODM để định nghĩa. Các trường timestamps: true sẽ tự động thêm createdAt và updatedAt vào mỗi document.

❖ **User Collection (users)**

Bảng 3.1 Bảng Collection users

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
1	username	String	Duy nhất, Bắt buộc, Cắt khoảng trắng đầu/cuối, Độ dài tối thiểu 3 ký tự, Index duy nhất	Tên người dùng
2	email	String	Duy nhất, Bắt buộc, Cắt khoảng trắng đầu/cuối, Chuyển thành chữ thường, Phải khớp định dạng email hợp lệ, Index duy nhất	Địa chỉ email
3	password	String	Bắt buộc, Độ dài tối thiểu 6 ký tự, Sẽ được hash bằng bcrypt trước khi lưu	Mật khẩu người dùng
4	fullName	String	Bắt buộc, Cắt khoảng trắng đầu/cuối	Họ và tên đầy đủ của người dùng
5	role	String	Liệt kê là "user" hoặc "admin", Mặc định là "user"	Vai trò của người dùng trong hệ thống
6	isVerified	Boolean	Mặc định là false	Trạng thái xác thực email

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
7	createdAt	Date	Tự động thêm bởi timestamps	Thời điểm tạo bản ghi
8	updatedAt	Date	Tự động thêm bởi timestamps	Thời điểm cập nhật bản ghi gần nhất

❖ **ReadingPassage Collection (readingpassages)**

Bảng 3.2 Bảng Collection readingpassages

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
1	title	String	Bắt buộc, Cắt khoảng trắng đầu/cuối, Có thể index	Tiêu đề của đoạn văn
2	content	String	Bắt buộc	Nội dung chính của đoạn văn
3	createdBy	ObjectId	Tham chiếu đến User Collection, Bắt buộc	Người tạo đoạn văn này
4	createdAt	Date	Mặc định là thời gian hiện tại	Thời điểm tạo đoạn văn
5	Question Count	Virtual	Đếm số câu hỏi liên quan đến đoạn văn này	Số lượng câu hỏi thuộc đoạn văn

❖ **Question Collection (questions)**

Bảng 3.3 Bảng Collection questions

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
1	passageId	ObjectId	Tham chiếu đến ReadingPassage Collection, Bắt buộc, Có thể index	ID của đoạn văn mà câu hỏi thuộc về
2	title	String	Cắt khoảng trắng đầu/cuối	Tiêu đề hoặc phần mở đầu của câu hỏi
3	questionType	String	Bắt buộc, Liệt kê là "multiple_choice", "fill_blank", "matching", "true_false_not_given", "short_answer", Có thể index	Loại câu hỏi
4	questionText	String	Bắt buộc, Cắt khoảng trắng đầu/cuối	Nội dung chi tiết của câu hỏi

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
5	options	Array of String	Dành riêng cho multiple_choice, Phải có ít nhất hai lựa chọn	Các lựa chọn đáp án cho câu hỏi trắc nghiệm
6	multipleAnswers	Boolean	Dành riêng cho multiple_choice, Mặc định là false	Cho phép nhiều đáp án đúng (trong trắc nghiệm)
7	blankStyle	String	Dành riêng cho fill_blank, Liệt kê là "simple", "multiple", "limited_words", "one_word_only", Mặc định là "simple"	Kiểu của câu hỏi điền vào chỗ trống
8	acceptableAnswers	Array of String	Dành riêng cho fill_blank (kiểu "simple"), Phải có ít nhất một đáp án chấp nhận được	Các đáp án chấp nhận được cho kiểu "simple"
9	blankOptions	Array of String	Dành riêng cho fill_blank (kiểu "multiple"), Phải có ít nhất hai lựa chọn	Các lựa chọn cho kiểu "multiple" (điền vào chỗ trống chọn từ danh sách)
10	oneWordAnswers	Array of String	Dành riêng cho fill_blank (kiểu "one_word_only"), Phải có ít nhất một đáp án	Đáp án dạng một từ cho kiểu "one_word_only"
11	wordLimits	Array of Number	Dành riêng cho fill_blank (kiểu "one_word_only"), Số lượng bằng oneWordAnswers	Giới hạn từ cho các đáp án dạng một từ
12	blankNumbers	Array of Number	Dành riêng cho fill_blank (kiểu "one_word_only"), Số lượng bằng oneWordAnswers	Số thứ tự của chỗ trống trong câu hỏi
13	matchingOptions	Object	Dành riêng cho matching, Chứa headings (mảng String) và paragraphs (mảng String), cả hai phải có ít nhất một phần tử	Các lựa chọn cho câu hỏi nối
14	correctAnswer	Mixed	Bắt buộc, Cấu trúc đa dạng tùy theo questionType	Đáp án đúng của câu hỏi
15	explanation	String	Có thể có	Giải thích cho đáp án đúng
16	score	Number	Bắt buộc, Mặc định là 1, Giá trị tối thiểu là 0	Điểm của câu hỏi

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
17	order	Number	Bắt buộc, Mặc định là 0, Giá trị tối thiểu là 1	Thứ tự hiển thị của câu hỏi trong bài kiểm tra
18	wordLimit	Number	Dành riêng cho short_answer, Phải lớn hơn 0	Giới hạn từ cho câu trả lời ngắn
19	acceptableShortAnswers	Array of String	Dành riêng cho short_answer, Phải có ít nhất một đáp án chấp nhận được	Các đáp án chấp nhận được cho câu trả lời ngắn
20	createdAt	Date	Tự động thêm bởi timestamps	Thời điểm tạo bản ghi
21	updatedAt	Date	Tự động thêm bởi timestamps	Thời điểm cập nhật bản ghi gần nhất

❖ **Test Collection (tests)**

Bảng 3.4 Bảng Collection tests

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
1	title	String	Bắt buộc, Cắt khoảng trắng đầu/cuối	Tiêu đề của bài kiểm tra
2	description	String	Cắt khoảng trắng đầu/cuối	Mô tả về bài kiểm tra
3	timeLimit	Number	Mặc định là 60 (phút)	Giới hạn thời gian làm bài kiểm tra
4	allowCustomTime	Boolean	Mặc định là false	Cho phép người dùng tùy chỉnh thời gian làm bài
5	allowNoTimeLimit	Boolean	Mặc định là false	Cho phép người dùng làm bài không giới hạn thời gian
6	passages	Array of Objects	Mảng các object, mỗi object chứa passageId (tham chiếu đến ReadingPassage, bắt buộc) và questions (mảng các tham chiếu đến Question)	Các đoạn văn và câu hỏi được bao gồm trong bài kiểm tra
7	totalScore	Number	Mặc định là 0, Có thể tính từ tổng score của các câu hỏi	Tổng điểm tối đa của bài kiểm tra

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
8	active	Boolean	Mặc định là true, Có thể index	Bài kiểm tra có sẵn cho học viên hay không
9	createdBy	ObjectId	Tham chiếu đến User Collection, Bắt buộc, Có thể index	Người tạo bài kiểm tra này
10	createdAt	Date	Mặc định là thời gian hiện tại	Thời điểm tạo bài kiểm tra
11	questionCount	Virtual	Tổng số câu hỏi trong bài kiểm tra	Tổng số câu hỏi trong bài kiểm tra
12	Pre-save validation	N/A	Đảm bảo bài test có ít nhất một đoạn văn và mỗi đoạn văn có ít nhất một câu hỏi	Kiểm tra hợp lệ trước khi lưu

❖ **UserAttempt Collection (userattempts)**

Bảng 3.5 Bảng Collection userattempts

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
1	user	ObjectId	Tham chiếu đến User Collection, Bắt buộc, Cần index	Người dùng đã thực hiện bài kiểm tra
2	test	ObjectId	Tham chiếu đến Test Collection, Bắt buộc, Cần index	Bài kiểm tra mà người dùng đã làm
3	answers	Array of Objects	Mảng các object, mỗi object chứa questionId (tham chiếu đến Question), userAnswer (Mixed), isCorrect (Boolean), score (Number)	Chi tiết câu trả lời của người d ùng cho từng câu hỏi
4	score	Number	Mặc định là 0	Tổng điểm đạt được của người dùng trong bài kiểm tra này
5	totalPossibleScore	Number	Mặc định là 0	Tổng điểm tối đa có thể đạt được của bài kiểm tra này
6	percentageScore	Number	Mặc định là 0	Điểm phần trăm đạt được

STT	Tên thuộc tính	Kiểu dữ liệu	Các ràng buộc (nếu có)	Mô tả
7	IeltsScore	Number		Điểm quy đổi ielts
8	timeSpentInSeconds	Number	Mặc định là 0	Thời gian người dùng đã dành để làm bài kiểm tra (tính bằng giây)
9	startedAt	Date	Mặc định là thời gian hiện tại	Thời điểm người dùng bắt đầu bài kiểm tra
10	completedAt	Date	Mặc định là null	Thời điểm người dùng hoàn thành bài kiểm tra
11	status	String	Liệt kê là "in_progress", "completed", "abandoned", Mặc định là "in_progress", Cần index	Trạng thái của lần làm bài
12	createdAt	Date	Tự động thêm bởi timestamps	Thời điểm tạo bản ghi

3.3.2. Thiết kế Sơ đồ cấu trúc Giao diện Người dùng

Mục tiêu của thiết kế cấu trúc giao diện là tạo ra một cái nhìn tổng quát, trực quan về giao diện của hệ thống

3.3.2.1. Nguyên tắc và Đặc điểm chung

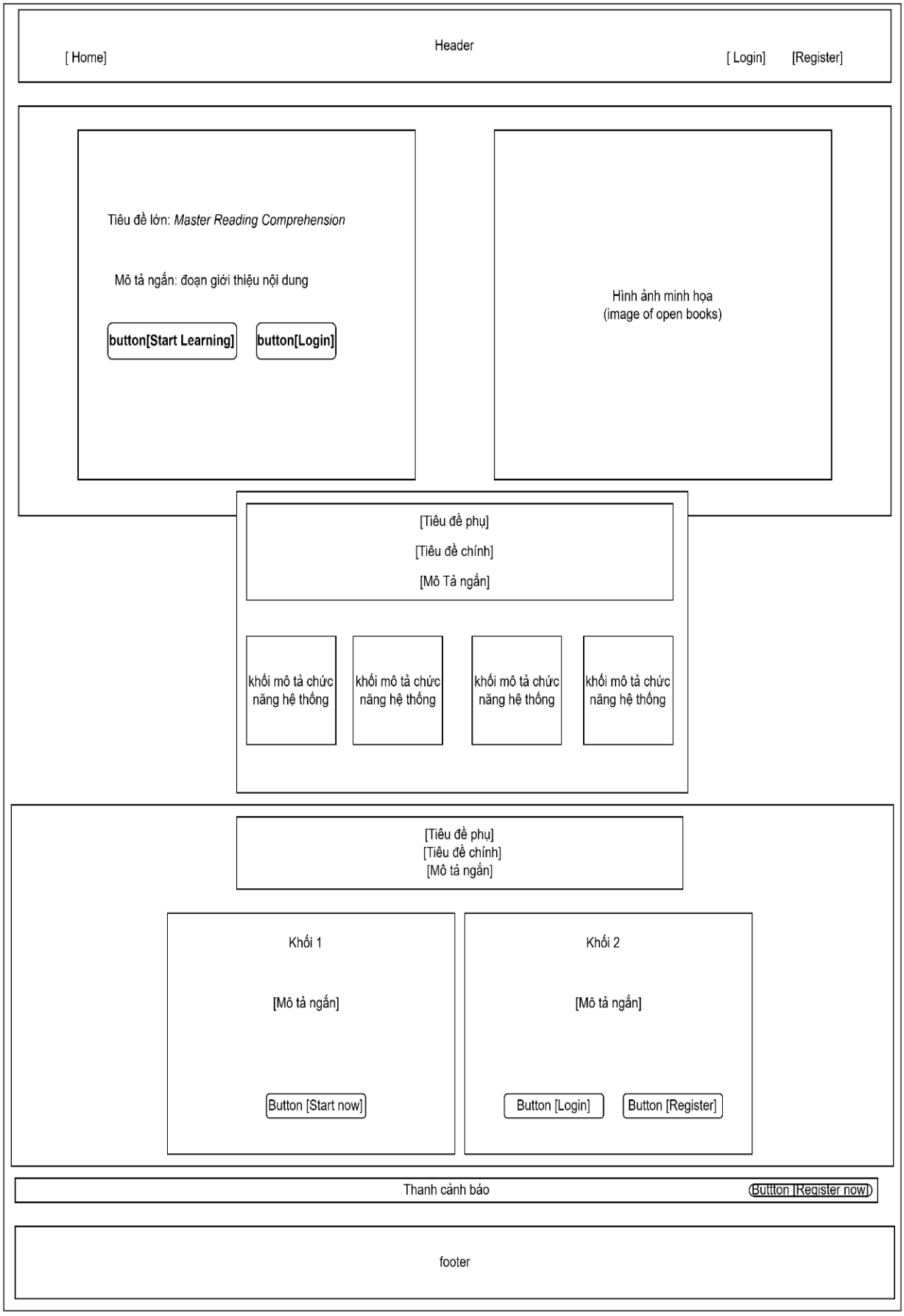
❖ **Triết lý Utility-First với Tailwind CSS:** Việc áp dụng Tailwind CSS giúp xây dựng giao diện nhanh chóng bằng cách kết hợp các lớp tiện ích trực tiếp trong mã HTML. Điều này tạo điều kiện cho việc tùy chỉnh cao độ và duy trì sự nhất quán trong thiết kế, đồng thời hỗ trợ mạnh mẽ cho responsive design.

❖ **Trải nghiệm người dùng (UX) được ưu tiên:** Tập trung vào tính nhất quán (sử dụng đồng bộ các yếu tố giao diện), phản hồi hệ thống (cung cấp phản hồi ngay lập tức cho các hành động của người dùng), tính đơn giản và trực quan (thiết kế luồng công việc logic và dễ hiểu), kiểm soát và tự do của người dùng (cho phép quay lại, hủy bỏ, sửa lỗi).

3.3.2.2. Thiết kế sơ đồ cấu trúc giao diện trang home (trang chủ)

❖ **Cấu trúc chung:**

- **Header (Đầu trang):** Chứa các liên kết điều hướng chính (Home, Login, Register).
- **Phần Hero (Phần nổi bật chính):** Khu vực đầu tiên và quan trọng nhất, nhằm thu hút sự chú ý và giới thiệu giá trị cốt lõi của website
  - Tiêu đề lớn (Master Reading Comprehension): Tiêu đề chính, nêu bật mục tiêu website.
  - Mô tả ngắn: Đoạn văn giới thiệu về phương pháp học, lợi ích mang lại.
  - button[Start Learning], button[Login]: Các nút kêu gọi hành động (CTA) chính, khuyến khích người dùng bắt đầu học hoặc đăng nhập.
  - Hình ảnh minh họa: Hình ảnh trực quan (cuốn sách đang mở) để tăng sự hấp dẫn.
- **Khối Tính năng (Features Section):** Trình bày các tính năng nổi bật của hệ thống.
  - Thành phần:
    + **[Tiêu đề chính] và [Tiêu đề phụ]:** Giới thiệu khu vực tính năng.
    + **Khối mô tả chức năng (4 khối):** Mỗi khối giới thiệu một tính năng cụ thể với icon và mô tả ngắn
- **Khối Lựa chọn người dùng (User Options):** Cung cấp hai lộ trình rõ ràng cho người dùng mới.
  - Thành phần:
    + **Khối 1 (Guest User):** Dành cho người dùng muốn trải nghiệm ngay không cần tài khoản. Cung cấp danh sách quyền lợi và nút [Start now].
    + **Khối 2 (Registered User):** Dành cho người dùng muốn đăng ký/đăng nhập để sử dụng đầy đủ tính năng như lưu lịch sử, theo dõi tiến độ. Có các nút [Login] và [Register].
- **Thanh Cảnh báo (Notification Bar):** Một dòng thông báo quan trọng cho người dùng để cảnh báo người dùng về việc mất dữ liệu nếu không đăng nhập. Nút Register now được tích hợp vào các khối CTA khác.

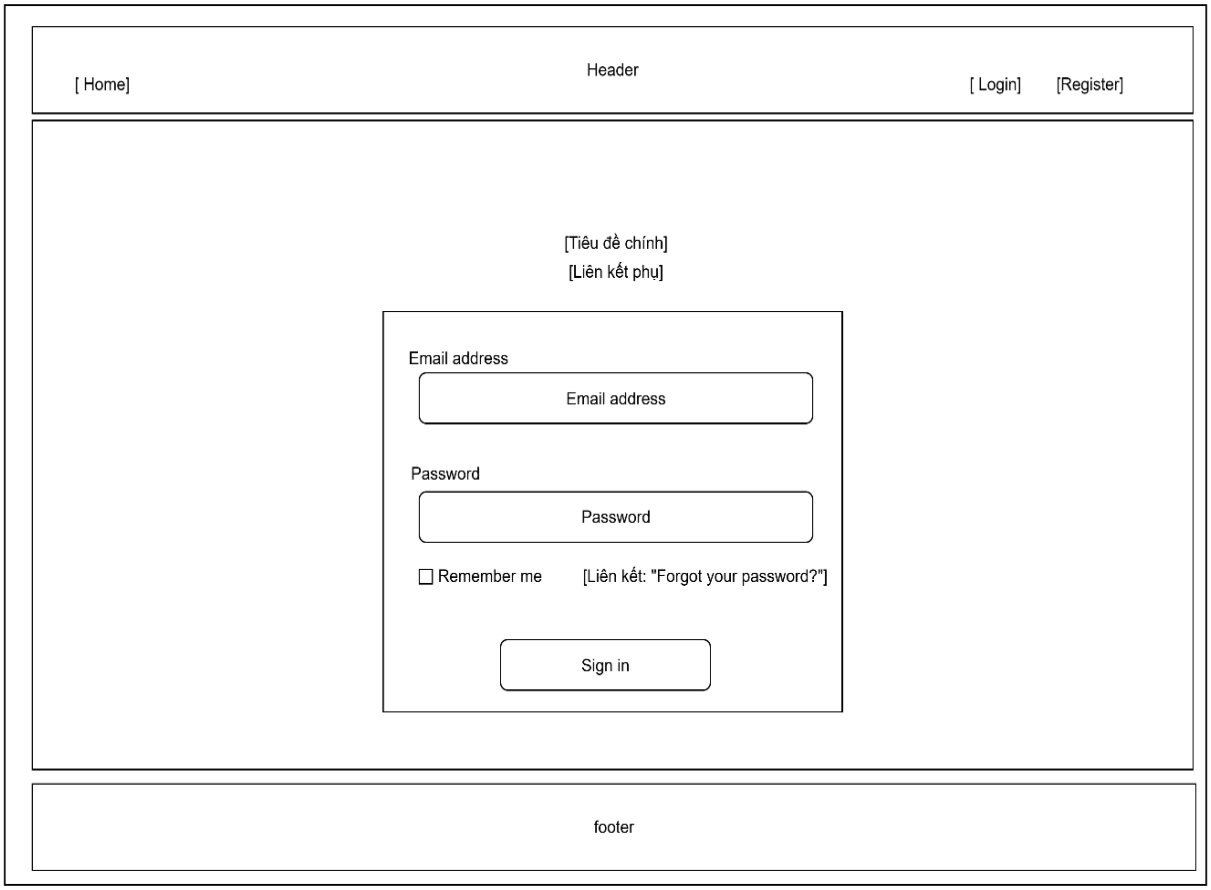


Hình 3.2 Sơ đồ cấu trúc trang chủ

3.3.2.3. Thiết kế giao diện đăng nhập và đăng ký tài khoản (login and sign in)

❖ **Trang đăng nhập**

- **Cấu trúc chính:**
- **Header (Đầu trang):** Chứa logo, liên kết "Home", và các tùy chọn "Login", "Register".
- **Phần nội dung chính (Form Đăng nhập):**
  - **Tiêu đề chính và liên kết phụ:** Có thể là tiêu đề của trang đăng nhập và một liên kết phụ liên quan.
  - **Trường nhập liệu Email:** Bao gồm nhãn "Email address" và ô nhập liệu.
  - **Trường nhập liệu Password:** Bao gồm nhãn "Password" và ô nhập liệu.
  - **Tùy chọn "Remember me" (Ghi nhớ tôi):** Một checkbox để người dùng có thể chọn ghi nhớ thông tin đăng nhập.
  - **Liên kết "Forgot your password?" (Quên mật khẩu):** Dẫn đến trang khôi phục mật khẩu.
  - **Nút "Sign in" (Đăng nhập):** Nút chính để gửi thông tin đăng nhập.

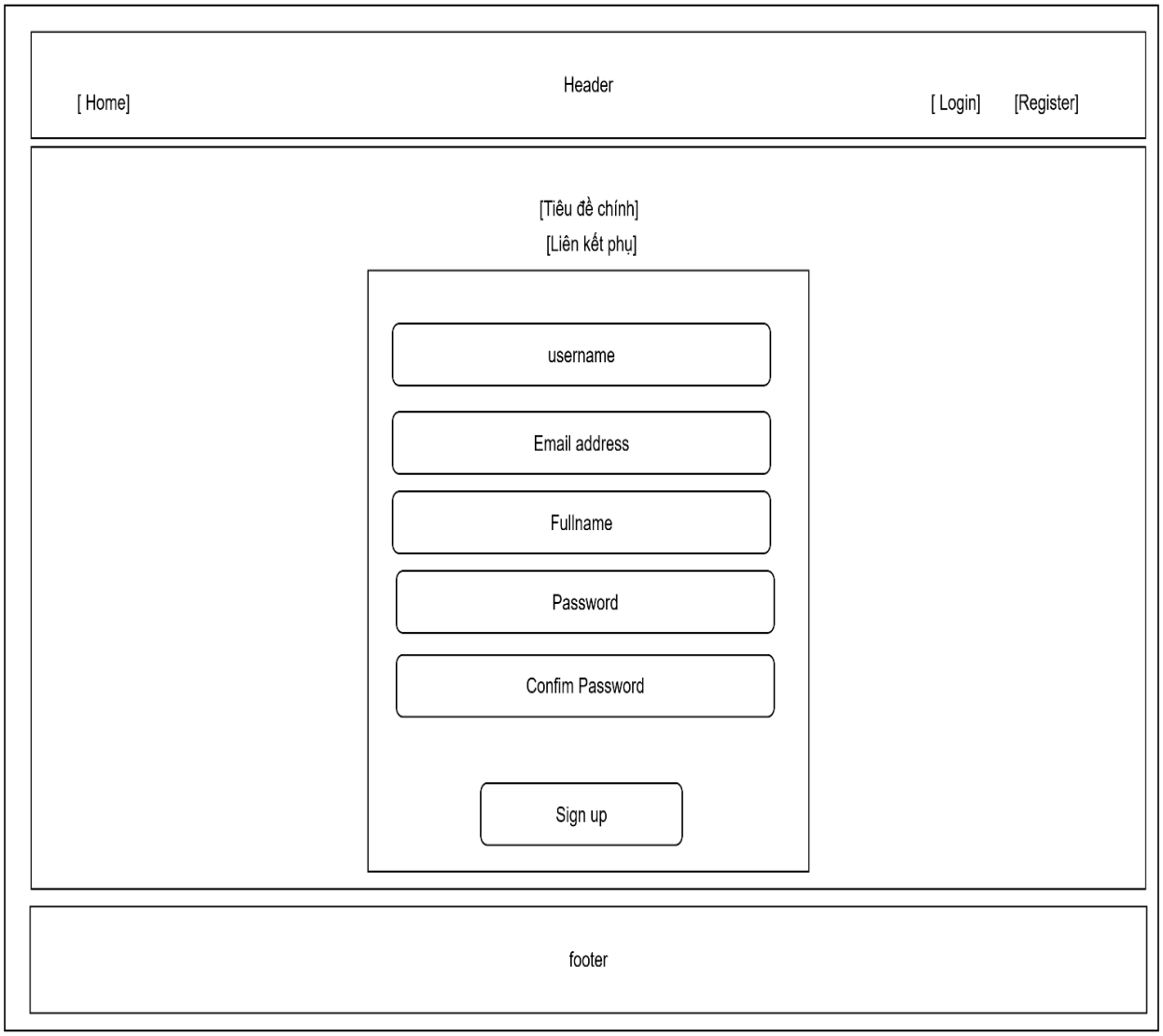


Hình 3.3 Sơ đồ cấu trúc trang đăng nhập

❖ **Trang đăng ký**

- **Phần nội dung chính (Form Đăng ký):**

- **Tiêu đề chính và liên kết phụ:** Có thể là tiêu đề của trang đăng ký và một liên kết phụ liên quan.
- **Trường nhập liệu "Fullname":** Để nhập họ và tên.
- **Trường nhập liệu "Username":** Để nhập tên đăng nhập.
- **Trường nhập liệu "Email address":** Để nhập địa chỉ email.
- **Trường nhập liệu "Password":** Để tạo mật khẩu.
- **Trường nhập liệu "Confirm Password":** Để nhập lại mật khẩu, đảm bảo khớp với mật khẩu đã nhập.
- **Nút "Sign up":** Nút chính để gửi thông tin đăng ký.



Hình 3.4 Sơ đồ cấu trúc trang đăng ký

3.3.2.4 thiết kế giao diện dashboard của quản trị viên (admin)

❖ **Cấu trúc chính:**

- **Header (Đầu trang):**
  - **Logo:** Biểu tượng của ứng dụng/hệ thống.
  - **Liên kết điều hướng:** Gồm các liên kết chính như "Home", "Dashboard", "Passages", "Test", "History".
  - **Thông tin người dùng:** Hiển thị "admin [username]" (tên người dùng admin) và nút "Button logout" (đăng xuất).
- **Phần nội dung chính (Dashboard):**
  - **Tiêu đề lớn:** Tiêu đề của trang tổng quan.
  - **Nút tạo mới:** "Button [New Passage]" (Tạo bài đọc mới) và "Button [New Test]" (Tạo bài kiểm tra mới), cho phép admin thêm nội dung nhanh chóng.
  - **Các khối thống kê nhanh:**
    + "Khối thống kê User": Hiển thị tổng quan về người dùng.
    + "Khối thống kê Reading Passage": Tổng quan về các bài đọc.
    + "Khối thống kê Test": Tổng quan về các bài kiểm tra.
  - **Các nút quản lý:**
    + "Button Manage Passage": Dẫn đến trang quản lý các bài đọc.
    + "Button Manage Test": Dẫn đến trang quản lý các bài kiểm tra.
  - **Các khối thống kê chi tiết/gần đây:**
    + "Thống kê Recent Users": Hiển thị danh sách hoặc biểu đồ về các người dùng mới/hoạt động gần đây.
    + "Thống kê Recent Test Results": Hiển thị kết quả các bài kiểm tra gần đây.
- **Footer (Chân trang):** Phần cuối trang, chứa thông tin bản quyền hoặc các liên kết phụ khác.

Sơ đồ cấu trúc trang dashboard admin như Hình 3.5 (Trang sau)

Hình 3.5 Sơ đồ cấu trúc trang dashboard admin

3.3.2.5 Thiết kế giao diện của quản lý bài đọc (manager passage)

❖ **Giao diện Danh sách Bài đọc (Reading Passages List)**

- **Phần nội dung chính (Quản lý Bài đọc):**

- **Tiêu đề lớn:** Tiêu đề của trang quản lý.
- **Nút "New Passage" (Bài đọc mới):** Cho phép người dùng tạo một bài đọc mới.
- **Thanh tìm kiếm và bộ lọc:**
  + Ô nhập liệu để tìm kiếm.

+ Nút "Search" (Tìm kiếm).

+ Nút "Clear" (Xóa) để xóa bộ lọc tìm kiếm.

- **Danh sách các bài đọc:** Được hiển thị dưới dạng các "card" hoặc "khối" riêng biệt. Mỗi khối bao gồm:

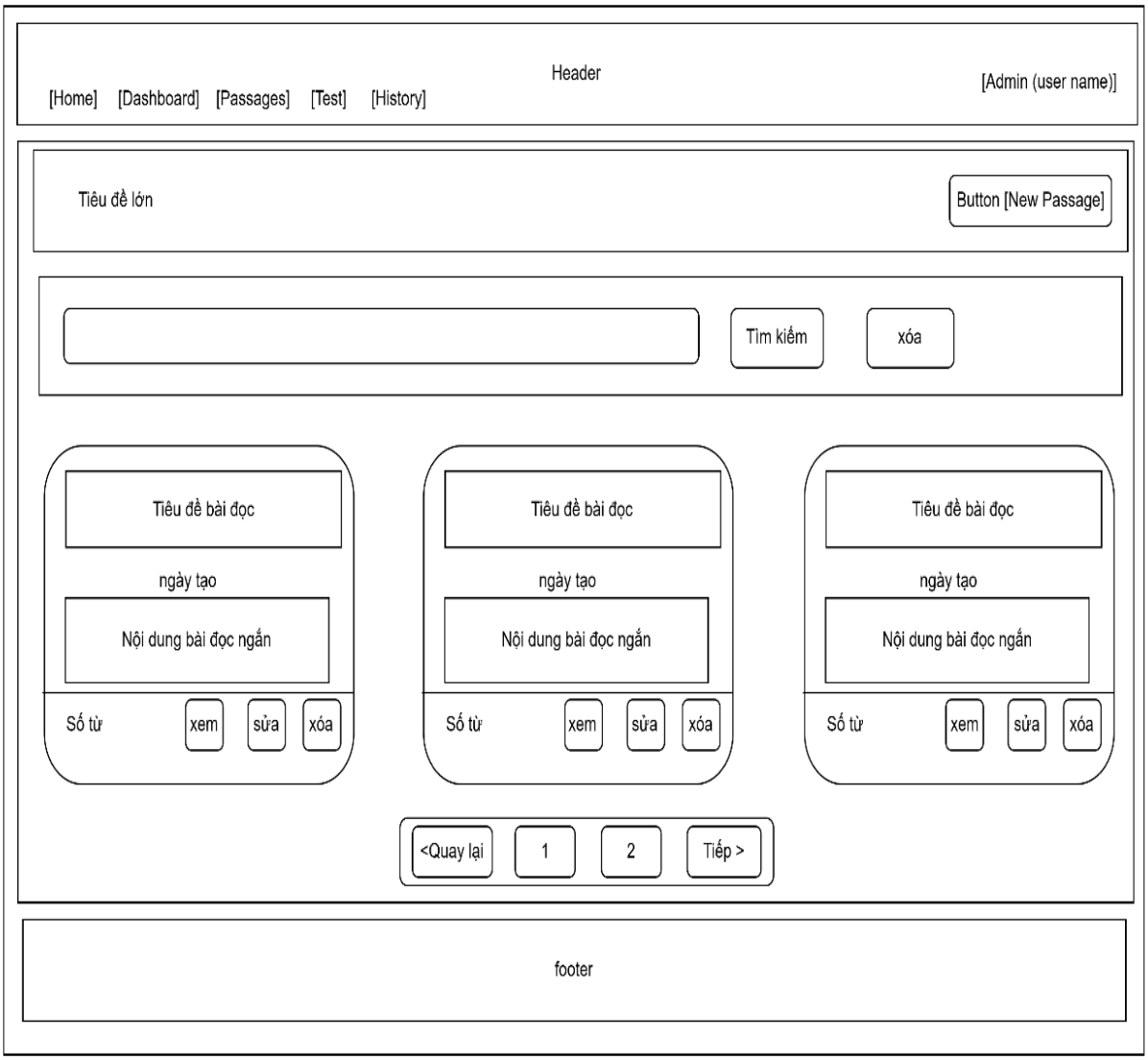
+ "[Tiêu đề bài đọc]": Tiêu đề của từng bài đọc.

+ "[Ngày tạo]": Hiển thị ngày tạo bài đọc

+ "[Nội dung một phần bài đọc]": Phần tóm tắt hoặc nội dung chính của bài đọc.

+ Các nút hành động: "xem" (xem chi tiết), "sửa" (chỉnh sửa), "xóa" (xóa bài đọc).

- **Phân trang:** Các nút điều hướng trang (ví dụ: "<", "1", "2", "3", ">") để duyệt qua danh sách bài đọc nếu có nhiều trang.



Hình 3.6 Sơ đồ cấu trúc trang danh sách bài đọc

❖ **Giao diện Tạo Bài đọc Mới (Create Reading Passage)**

- **Phần nội dung chính (Trang tạo/chỉnh sửa Bài đọc):**

- **Tiêu đề trang tạo:** Tiêu đề cho biết mục đích của trang (ví dụ: "Tạo bài đọc mới" hoặc "Chỉnh sửa bài đọc").
- **Nút "Back to Passage" (Quay lại trang Bài đọc):** Cho phép người dùng dễ dàng trở về trang danh sách các bài đọc.
- **Trường nhập liệu "Title" (Tiêu đề):** Một ô nhập liệu lớn để nhập tiêu đề của bài đọc.
- **Trường nhập liệu "Content" (Nội dung):** Một vùng văn bản lớn hơn để nhập toàn bộ nội dung của bài đọc.
- **Nút "Create" (Tạo bài đọc):** Cho phép người dùng tạo một bài đọc mới.

Hình 3.7 Sơ đồ cấu trúc trang tạo bài đọc

❖ **Giao diện Chỉnh sửa Bài đọc (Edit Reading Passage)**

- **Phần nội dung chính (Trang chỉnh sửa):**

- **Tiêu đề trang sửa:** Tiêu đề cho biết mục đích của trang.
- **Nút "View Details" (Xem chi tiết):** Cho phép xem phiên bản chi tiết của nội dung đang chỉnh sửa.
- **Nút "Back to Passage" (Quay lại trang Bài đọc):** Cho phép người dùng dễ dàng trở về trang danh sách.
- **Trường nhập liệu "Title cần sửa":** Một ô nhập liệu lớn để chỉnh sửa tiêu đề của nội dung.
- **Trường nhập liệu "Content cần sửa":** Một vùng văn bản lớn hơn để chỉnh sửa nội dung chính.
- **Nút "Update" (Sửa bài đọc):** Cho phép người dùng sửa một bài đọc.

Hình 3.8 Sơ đồ cấu trúc sửa bài đọc

Ba giao diện này phối hợp với nhau tạo thành một quy trình quản lý bài đọc trực quan và đầy đủ chức năng cho quản trị viên.

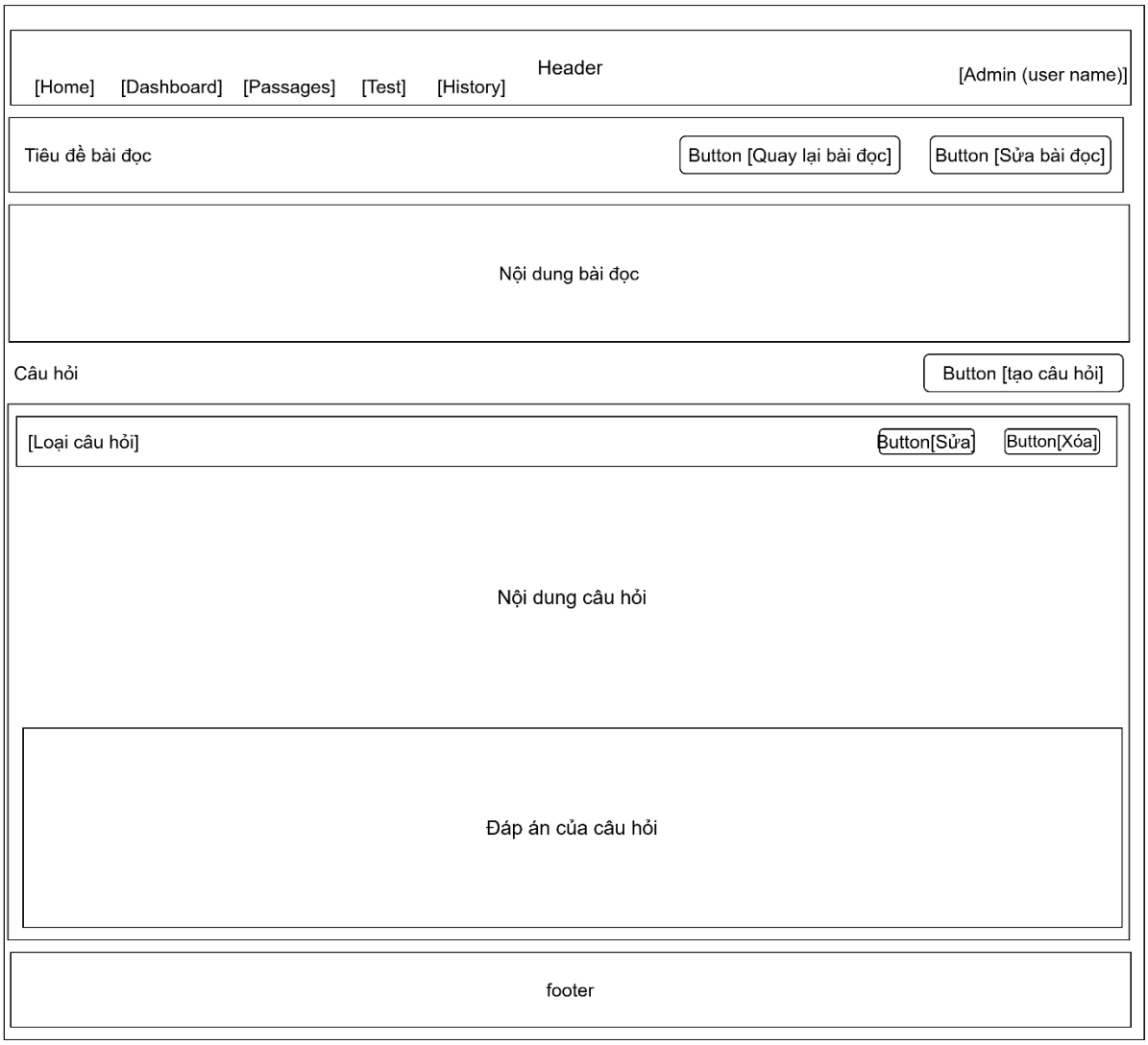
3.3.2.6 Thiết kế giao diện quản lý Câu hỏi (Questions Management)

Dưới đây là mô tả chi tiết cho từng giao diện trong quy trình quản lý câu hỏi, thường liên quan đến các bài đọc:

❖ **Giao diện Danh sách Câu hỏi (Questions List)**

- **Phần nội dung chính (Xem/Quản lý Bài đọc và Câu hỏi):**
  - **Tiêu đề "Bài đọc":** Tiêu đề chính của trang.
  - **Nút "Edit Passage" (Chỉnh sửa Bài đọc):** Cho phép người dùng chỉnh sửa nội dung của bài đọc hiện tại.
  - **Nút "Back to Passage" (Quay lại trang Bài đọc):** Dẫn người dùng trở lại trang danh sách các bài đọc.
  - **Khu vực hiển thị Bài đọc:**
    + "Tiêu đề bài đọc": Hiển thị tiêu đề của bài đọc.
    + "Nội dung bài đọc": Hiển thị toàn bộ nội dung của bài đọc.
  - **Khu vực "Question" (Câu hỏi):**
    + "Button [Add new question]" (Thêm câu hỏi mới): Nút để thêm một câu hỏi mới liên quan đến bài đọc.
    + "Nội dung question": Khu vực hiển thị nội dung của một câu hỏi (hoặc danh sách các câu hỏi).
    + Các nút hành động cho câu hỏi: "Sửa" (chỉnh sửa câu hỏi), "xóa" (xóa câu hỏi). Điều này ngụ ý rằng có thể có nhiều câu hỏi và mỗi câu hỏi sẽ có các nút này.

  Sơ đồ cấu trúc trang danh sách câu hỏi như Hình 3.9 (Trang sau)



Hình 3.9 Sơ đồ cấu trúc danh sách câu hỏi

❖ **Giao diện Tạo Câu hỏi Mới (Create New Question)**

- **Phần nội dung chính (Tạo Câu hỏi):**

- **Tiêu đề "Tạo câu hỏi":** Tiêu đề chính của trang.
- **Nút "Back to Passage" (Quay lại trang Bài đọc):** Dẫn người dùng trở lại trang danh sách các bài đọc.
- **Khu vực hiển thị Bài đọc:**

+ "Tiêu đề bài đọc": Hiển thị tiêu đề của bài đọc mà câu hỏi này sẽ thuộc về.

+ "Nội dung bài đọc": Hiển thị toàn bộ nội dung của bài đọc để người dùng có thể tham chiếu khi tạo câu hỏi.

- **Khu vực nhập liệu thông tin câu hỏi:**

+ "Chọn loại câu hỏi": Một dropdown hiển thị các loại câu hỏi để người dùng có thể chọn để tạo.

+ "Thứ tự câu hỏi": Ô nhập liệu để xác định thứ tự của câu hỏi.

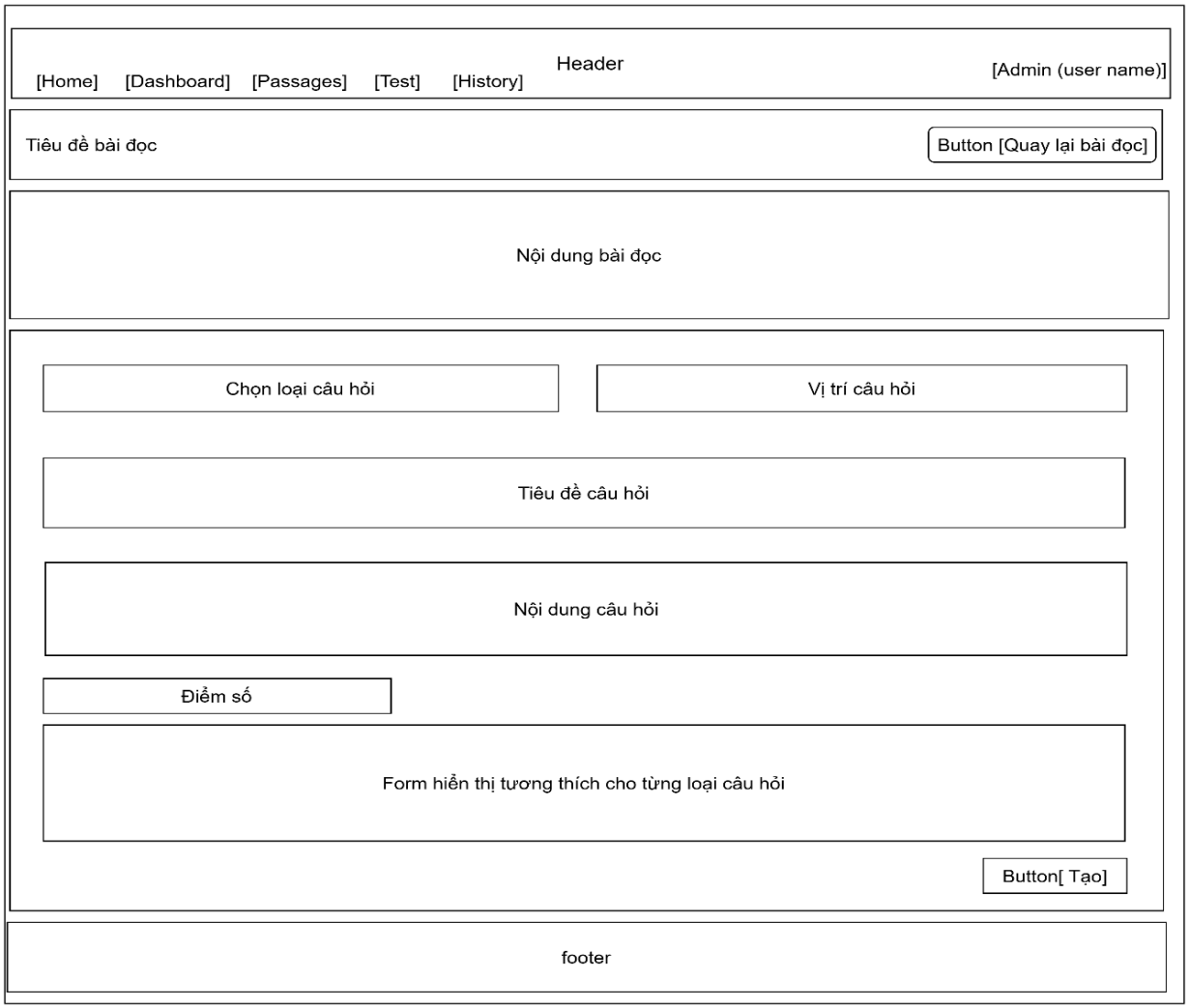
+ "tiêu đề câu hỏi": Ô nhập liệu cho tiêu đề hoặc phần mở đầu của câu hỏi.

+ "nội dung câu hỏi": Ô nhập liệu lớn hơn cho nội dung chi tiết của câu hỏi.

+ "điểm": Ô nhập liệu để gán điểm cho câu hỏi này.

+ "Form hiển thị tương thích cho từng loại câu hỏi": Đây là một khu vực động, sẽ thay đổi giao diện nhập liệu tùy thuộc vào loại câu hỏi đã chọn.

- **Nút "Create" (Tạo câu hỏi):** Cho phép người dùng sửa tạo câu hỏi.



Hình 3.10 Sơ đồ cấu trúc tạo câu hỏi

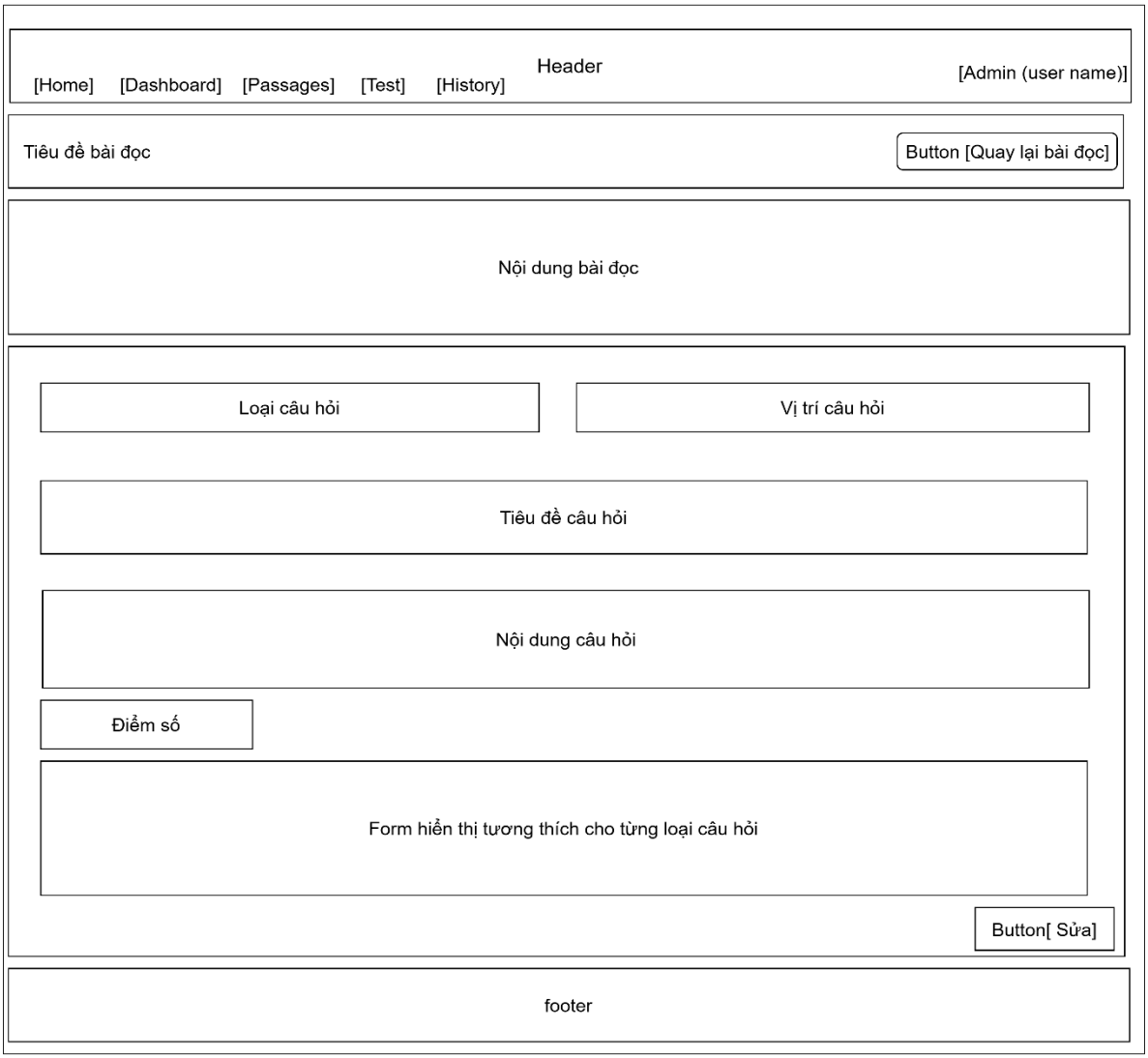
❖ **Giao diện Chỉnh sửa Câu hỏi (Edit Question)**

- **Phần nội dung chính (Chỉnh sửa Câu hỏi):**

- **Tiêu đề "Sửa câu hỏi":** Tiêu đề chính của trang, chỉ rõ mục đích là chỉnh sửa.
- **Nút "Back to Passage" (Quay lại trang Bài đọc):** Dẫn người dùng trở lại trang danh sách các bài đọc hoặc trang xem chi tiết bài đọc.
- **Khu vực hiển thị Bài đọc:**

+ "Tiêu đề bài đọc": Hiển thị tiêu đề của bài đọc mà câu hỏi này thuộc về.
+ "Nội dung bài đọc": Hiển thị toàn bộ nội dung của bài đọc để người dùng có thể tham chiếu khi chỉnh sửa câu hỏi.

- **Khu vực nhập liệu thông tin câu hỏi:**
  + "Chọn loại câu hỏi": Có thể là một ô hiển thị loại câu hỏi hiện tại
  + "Thứ tự câu hỏi": Ô nhập liệu để chỉnh sửa thứ tự của câu hỏi.
  + "tiêu đề câu hỏi": Ô nhập liệu cho tiêu đề hoặc phần mở đầu của câu hỏi.
  + "nội dung câu hỏi": Ô nhập liệu lớn hơn cho nội dung chi tiết của câu hỏi.
  + "điểm": Ô nhập liệu để chỉnh sửa điểm số gán cho câu hỏi này.
  + "Form hiển thị tương thích cho từng loại câu hỏi": Đây là một khu vực động, sẽ thay đổi giao diện nhập liệu tùy thuộc vào loại câu hỏi hiện tại.
- **Nút "Update" (Sửa câu hỏi):** Cho phép người dùng sửa một câu hỏi.



Hình 3.11 Sơ đồ cấu trúc sửa câu hỏi

Ba giao diện này tạo thành một hệ thống chức năng cho phép quản trị viên hoặc người soạn thảo nội dung có thể dễ dàng xây dựng, quản lý và tùy chỉnh ngân hàng câu hỏi liên quan đến các bài đọc hoặc chủ đề cụ thể.

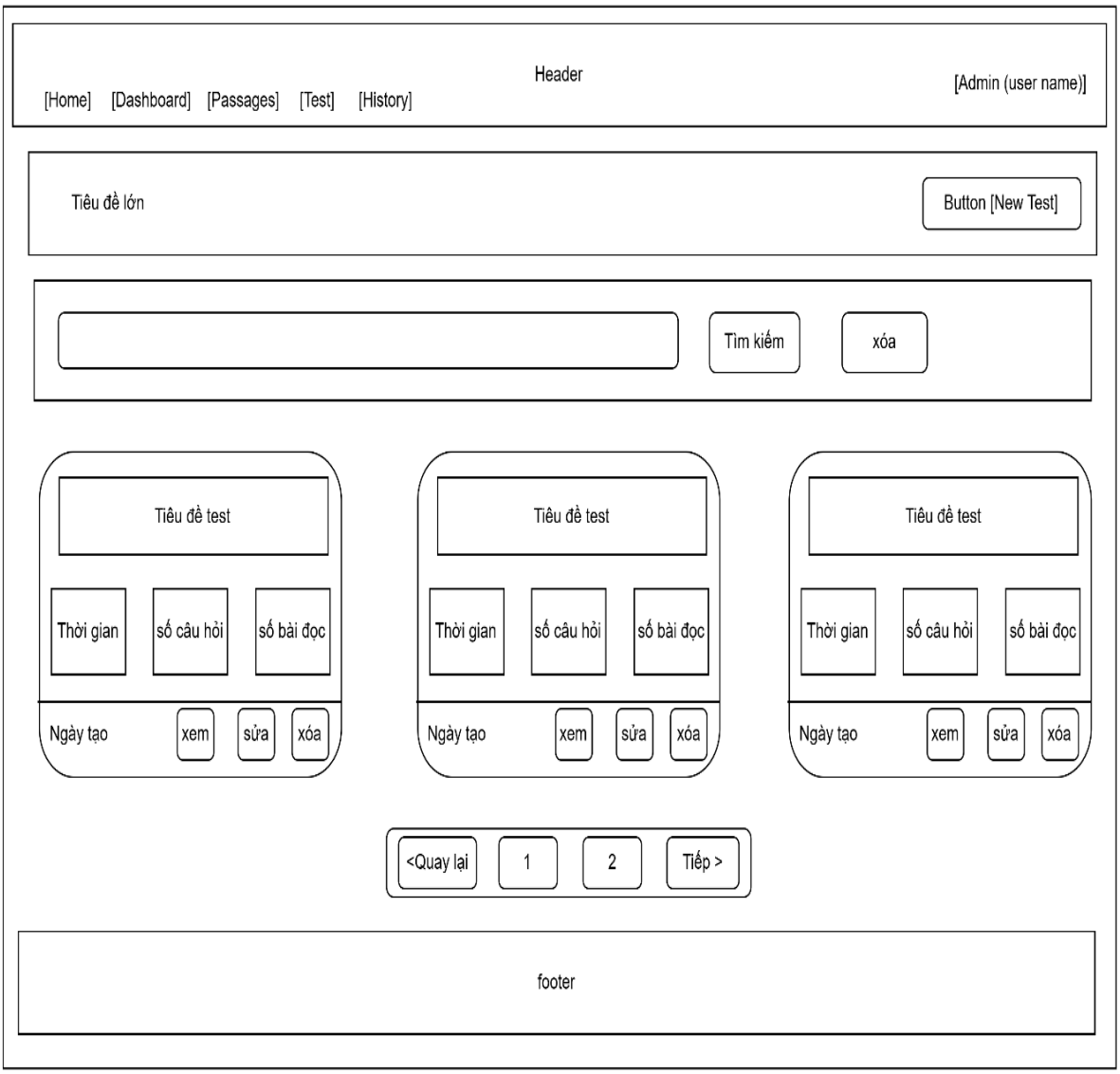
3.3.2.7 Thiết kế giao diện quản lý bài kiểm tra (Manager Test)

Ba giao diện này cung cấp một quy trình đầy đủ để tạo, xem, quản lý và chỉnh sửa các bài kiểm tra đọc trong hệ thống.

❖ **Giao diện Danh sách Bài kiểm tra Đọc (Reading Tests List)**

- **Phần nội dung chính (Quản lý Bài kiểm tra):**
  - **Tiêu đề "Test":** Tiêu đề chính của trang.
  - **Nút "Create Test" (Tạo bài kiểm tra):** Cho phép người dùng tạo một bài kiểm tra mới.
  - **Thanh tìm kiếm và bộ lọc:**
    + Ô nhập liệu để tìm kiếm.
    + Nút "Search" (Tìm kiếm).
    + Nút "Clear" (Xóa) để xóa bộ lọc tìm kiếm.
  - **Danh sách các bài kiểm tra:** Được hiển thị dưới dạng các "card" hoặc "khối" riêng biệt. Mỗi khối bao gồm:
    + "Tiêu đề Test": Tiêu đề của từng bài kiểm tra.
    + Các thông tin tóm tắt:
      - "thời gian": Thời lượng của bài kiểm tra.
      - "số câu hỏi": Điểm tối đa hoặc thông tin điểm khác.
      - "số bài đọc": Số lượng bài đọc liên quan đến bài kiểm tra này (nếu có).
    + Các nút hành động: "xem" (xem chi tiết bài kiểm tra), "sửa" (chỉnh sửa bài kiểm tra), "xóa" (xóa bài kiểm tra).

    Sơ đồ cấu trúc trang danh sách bài kiểm tra như Hình 3.12 (Trang sau)



Hình 3.12 Sơ đồ cấu trúc trang danh sách bài kiểm tra

❖ **Giao diện Tạo Bài kiểm tra Đọc Mới (Create Reading Test)**

- **Phần nội dung chính (Tạo bài kiểm tra):**
  - **Tiêu đề "Tạo Test":** Tiêu đề chính của trang.
  - **Nút "Back to Test" (Quay lại trang Test):** Cho phép người dùng quay lại trang danh sách các bài kiểm tra.
  - **Trường nhập liệu "Test title":** Ô để nhập tiêu đề của bài kiểm tra.
  - **Trường nhập liệu "Time limit":** Ô để đặt giới hạn thời gian cho bài kiểm tra.
  - **Bộ lọc loại câu hỏi để tạo test**
    + Nhãn: “Lọc câu hỏi để tạo test”
    + Các nút lọc câu hỏi theo loại:

+ All
+ Multiple Choice
+ Fill in the blank
+ Matching
+ True/False/Not Given
+ Short answers

- **Nút [Xóa tất cả lựa chọn]:** Xóa các câu hỏi đã chọn (nếu có)
- **Khu vực chọn bài đọc (Passages):**

  + Một ô nhập liệu và nút "Search" để tìm kiếm các bài đọc.

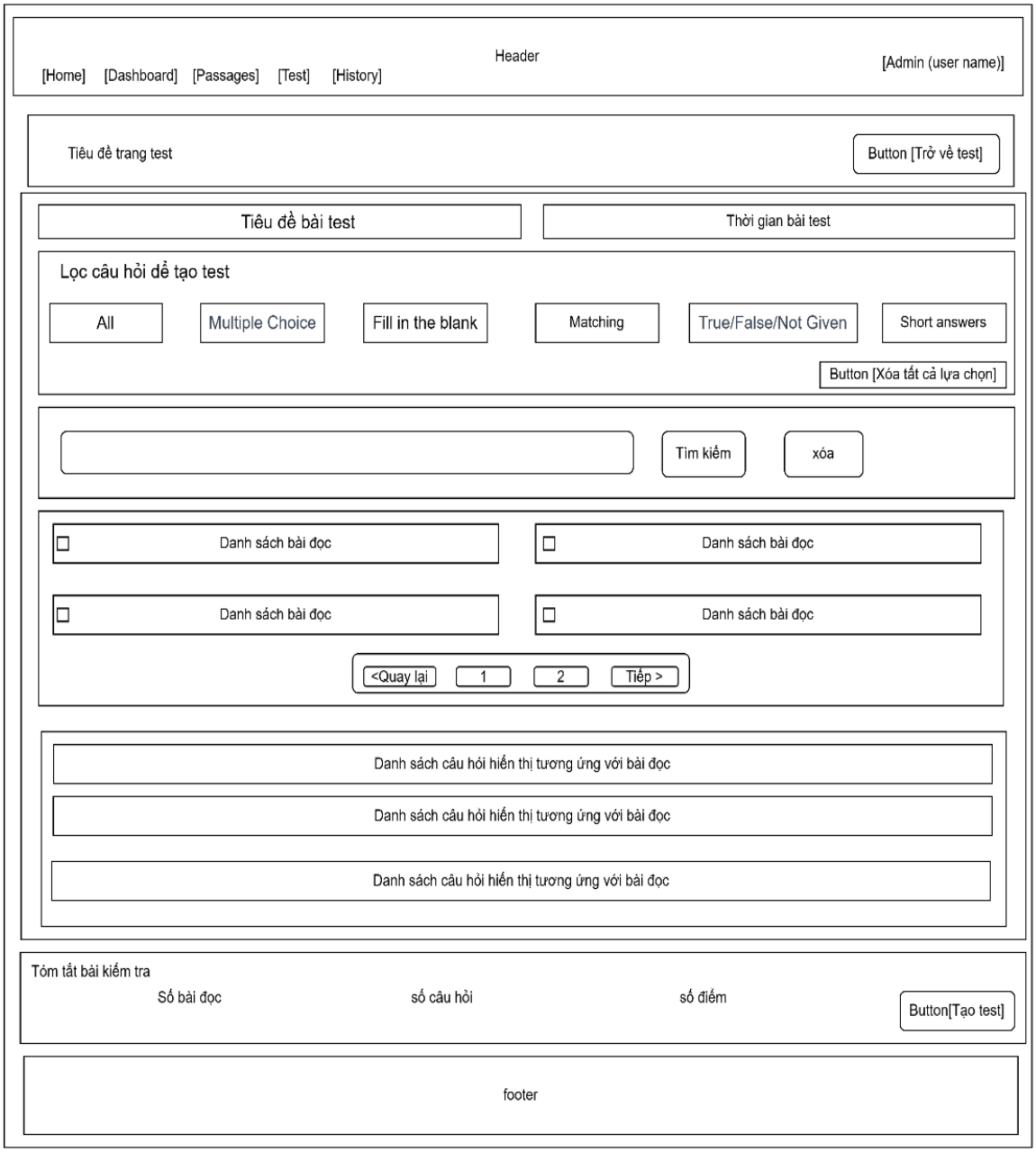
  + Một danh sách các "Tiêu đề passage" với checkbox bên cạnh mỗi tiêu đề, cho phép người dùng chọn các bài đọc sẽ được đưa vào bài kiểm tra.

  + Các nút phân trang ("<", "1", "2", "3", ">") để duyệt qua danh sách bài đọc nếu có nhiều trang.
- **Khu vực "Select Questions" (Chọn câu hỏi):**

  + "List Question": Một khu vực để hiển thị các câu hỏi đã được chọn

  + **Nút "Create Test" (Tạo bài kiểm tra):** Nút để hoàn tất quá trình tạo bài kiểm tra.

  Sơ đồ cấu trúc trang tạo bài kiểm tra như hình 3.13 (Trang sau)



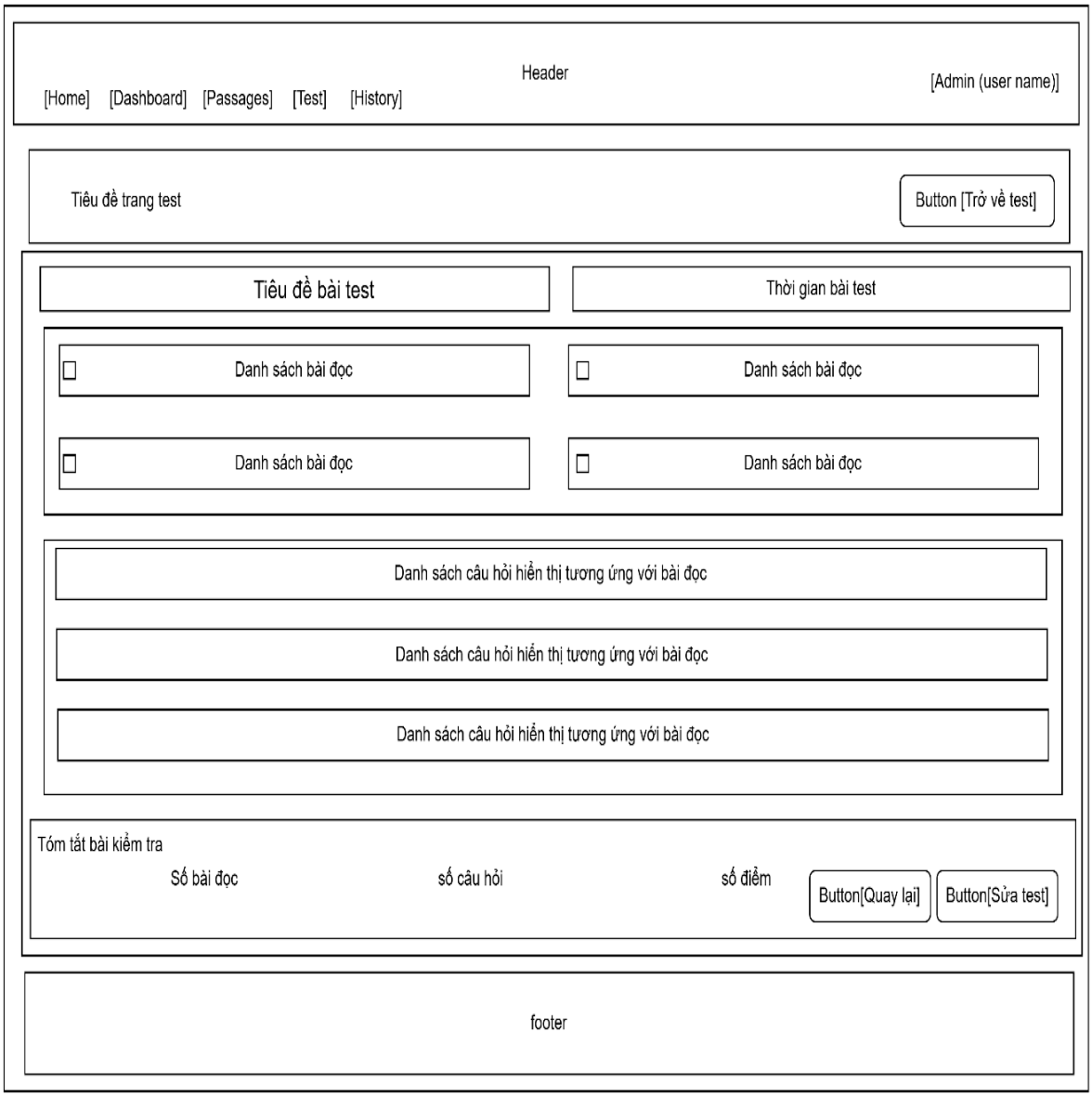
Hình 3.13 Sơ đồ cấu trúc tạo bài kiểm tra

❖ **Giao diện Chỉnh sửa Bài kiểm tra Đọc (Edit Reading Test)**

- **Phần nội dung chính (Chỉnh sửa Bài kiểm tra):**
  + **Tiêu đề "Sửa Test":** Tiêu đề chính của trang, chỉ rõ mục đích là chỉnh sửa.
  + **Nút "Back to Test" (Quay lại trang Test):** Cho phép người dùng quay lại trang danh sách các bài kiểm tra.
  + **Trường nhập liệu "Test title":** Ô để chỉnh sửa tiêu đề của bài kiểm tra.
  + **Trường nhập liệu "Time limit":** Ô để chỉnh sửa giới hạn thời gian cho bài kiểm tra.
  + **Khu vực chọn bài đọc (Passages):**

- Một danh sách các "Tiêu đề passage" với checkbox bên cạnh mỗi tiêu đề, cho phép người dùng chọn hoặc bỏ chọn các bài đọc sẽ được đưa vào bài kiểm tra. Các bài đọc đã chọn trước đó sẽ được đánh dấu sẵn.
- Các nút phân trang ("<", "1", "2", "3", ">") để duyệt qua danh sách bài đọc nếu có nhiều trang.

+ **Khu vực "Select Questions" (Chọn câu hỏi):**
  - "List Question": Một khu vực để hiển thị các câu hỏi đã được chọn cho bài kiểm tra này, hoặc để chọn/thêm/bớt các câu hỏi khác.
+ **Nút "cancel" (Hủy):** Hủy bỏ các thay đổi và quay lại trang trước.
+ **Nút "update Test" (Cập nhật Test):** Lưu lại các thay đổi đã thực hiện cho bài kiểm tra.



Hình 3.14 Sơ đồ cấu trúc sửa bài kiểm tra

Bộ ba giao diện này cung cấp một quy trình quản lý bài kiểm tra đọc khá toàn diện, từ việc tạo mới với các cấu hình chi tiết về bài đọc và câu hỏi, đến việc chỉnh sửa và theo dõi danh sách các bài kiểm tra.

3.3.2.8 Thiết kế giao diện trải nghiệm người dùng cho hệ thống luyện thi kỹ năng Reading trực tuyến

Bộ giao diện này hướng dẫn người dùng qua toàn bộ quá trình làm bài kiểm tra kỹ năng đọc, bao gồm lựa chọn bài kiểm tra, tùy chỉnh cài đặt, làm bài, xem kết quả và theo dõi lịch sử.

Ngoài ra hệ thống còn có chức năng cho người dùng không đăng nhập và người dùng đăng nhập. Đối với người dùng không đăng nhập sẽ không lưu được lịch sử làm bài, ngược lại đối với người dùng đăng nhập sẽ lưu được lịch sử làm bài.

❖ **Giao diện Trang chính của Người dùng (User Dashboard)**

- **Header (Thanh điều hướng)**

• **Các mục điều hướng chính hiển thị ở đầu trang:**

+ [Home] — Trang chủ
+ [Dashboard] — Bảng điều khiển tổng quan
+ [Passages] — Quản lý các bài đọc
+ [Test] — Quản lý bài kiểm tra
+ [History] — Lịch sử làm bài của người dùng

• **Phía bên phải**: hiển thị tên người dùng dưới dạng [(user name)]

- **Nội dung chính (Main Content)**

+ Tiêu đề chào mừng.
+ Một khối chứa nội dung chào đón người dùng: “Chào mừng người dùng trở lại, [Tên người dùng]!”.
+ Thống kê số lượng bài kiểm tra đã làm.
+ Hiển thị thông tin các bài kiểm tra gần đây mà người dùng đã thực hiện.

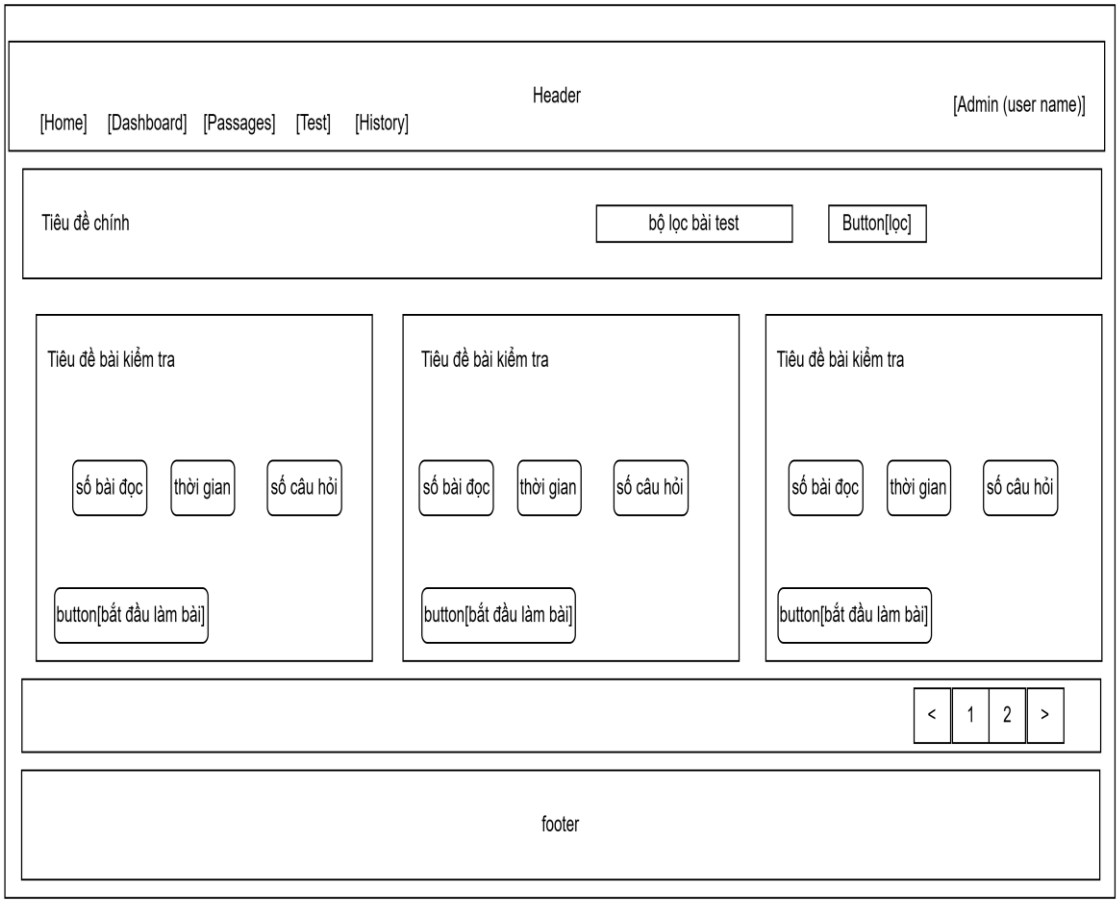
Sơ đồ cấu trúc dashboard của người dùng như hình 3.15 (Trang sau)

Hình 3.15 Sơ đồ cấu trúc dashboard của người dùng

❖ **Giao diện Danh sách Bài kiểm tra của người dùng (Test List)**

- **Phần nội dung chính (Danh sách bài kiểm tra):**
  - **Tiêu đề chính:** Tiêu đề của trang
  - Bộ lọc bài kiểm tra: cho phép người dùng có thể lọc bài kiểm tra theo dạng đề.
  - **Các khối bài kiểm tra:** Mỗi khối đại diện cho một bài kiểm tra riêng biệt và hiển thị các thông tin tóm tắt:
    + "Tiêu đề bài kiểm tra": Tên của bài kiểm tra.
    + "số lượng bài đọc": Số lượng bài đọc liên quan đến bài kiểm tra này.
    + "thời gian": Thời lượng dự kiến của bài kiểm tra.
    + "điểm": Thông tin về điểm số.
    + "button[bắt đầu làm bài]": Nút để người dùng bắt đầu làm bài kiểm tra này.

- **Phân trang:** Các nút điều hướng trang ("<", "1", "2", ">") để duyệt qua danh sách bài kiểm tra nếu có nhiều hơn một trang.

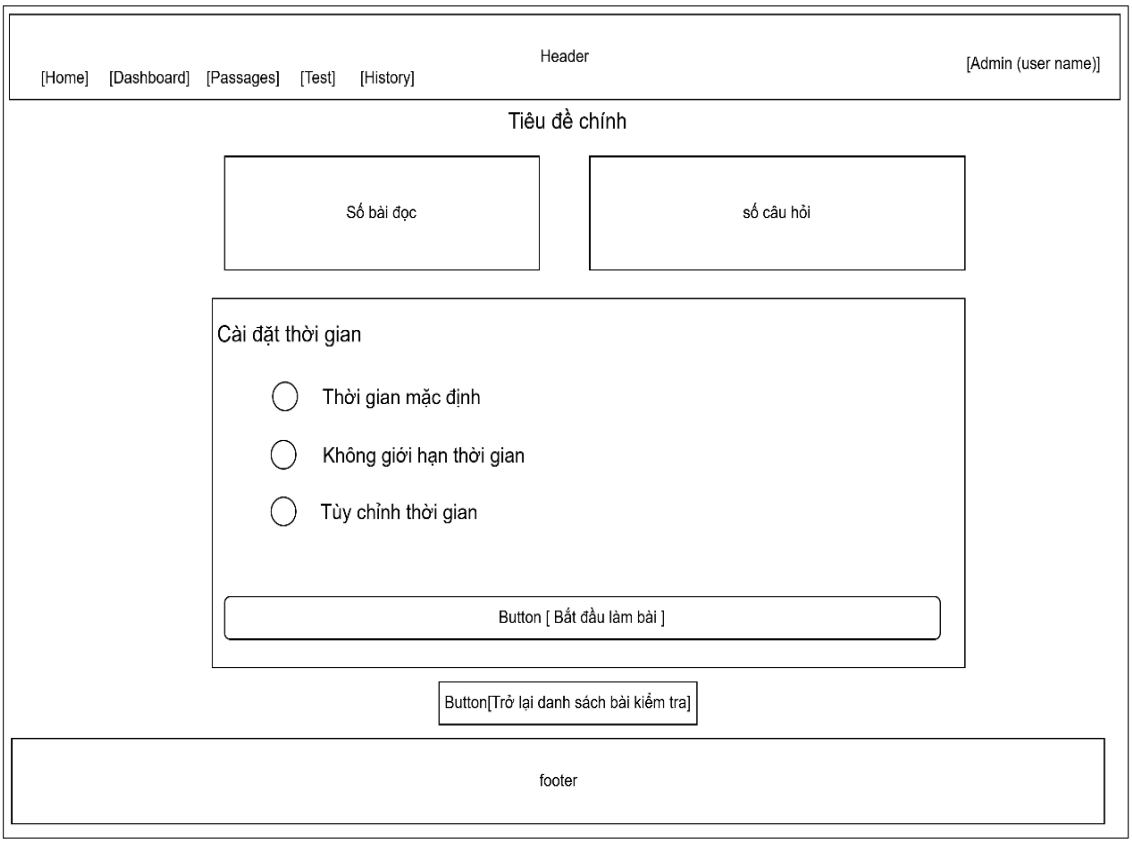


Hình 3.16 Sơ đồ cấu trúc danh sách bài kiểm tra người dùng

❖ **Giao diện Chuẩn bị/Cài đặt Bài kiểm tra (Test Preparation/Settings)**

- **Phần nội dung chính (Cấu hình Bài kiểm tra):**
  - **Tiêu đề bài kiểm tra:** Tên của bài kiểm tra mà người dùng sắp làm.
  - **Các khối thông tin tổng quan về bài kiểm tra:**
    + **Số lượng bài đọc:** Tổng số bài đọc có trong bài kiểm tra này.
    + **Số câu hỏi:** Tổng số câu hỏi trong bài kiểm tra.
  - **Khu vực "Time settings" (Cài đặt thời gian):** Cho phép người dùng tùy chỉnh thời gian làm bài:
    + **Default time (60 minutes):** Tùy chọn để sử dụng thời gian mặc định (ví dụ: 60 phút).
    + **No time limit:** Tùy chọn không giới hạn thời gian.
    + **Custom time: [ô nhập liệu] min:** Tùy chọn để người dùng tự nhập thời gian làm bài tùy chỉnh (ví dụ: 60 phút).

- **Button[Start test] (Bắt đầu bài kiểm tra):** Nút để bắt đầu bài kiểm tra với các cài đặt đã chọn.
- Button[Back to list] (Trở lại bài đọc): Nút dùng để trở lại danh sách bài kiểm tra

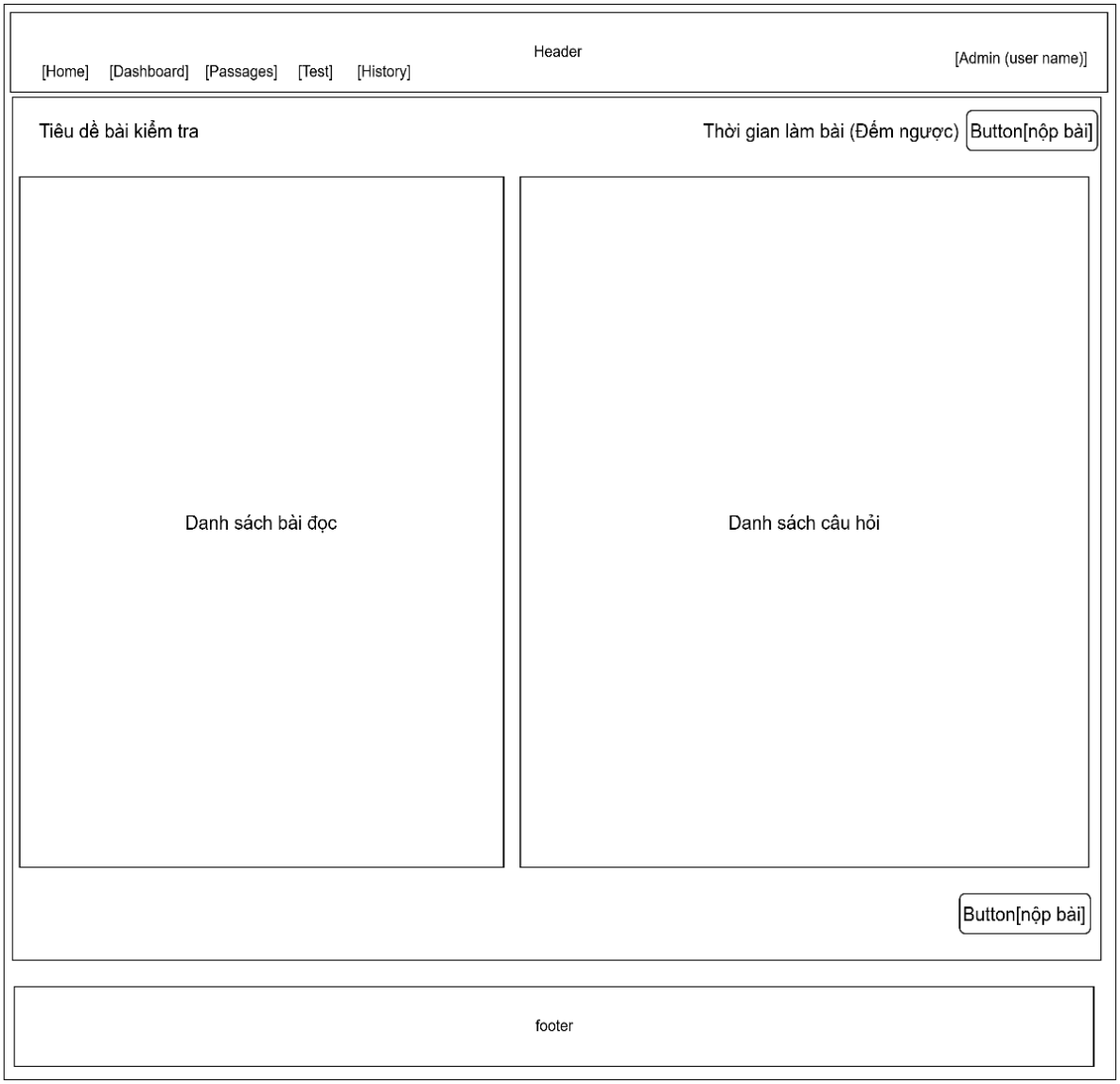


Hình 3.17 Sơ đồ cấu trúc cài đặt bài kiểm tra

❖ **Giao diện Làm Bài kiểm tra (Test Taking Interface)**

- **Phần nội dung chính (Trang làm bài kiểm tra):**
  - **Tiêu đề bài test:** Hiển thị tiêu đề của bài kiểm tra đang làm.
  - **Khối bên trái - Nội dung bài đọc:**
    - "Nội dung bài đọc": Khu vực lớn để hiển thị toàn bộ văn bản của bài đọc liên quan đến bài kiểm tra. Điều này cho phép người dùng đọc và tham khảo nội dung khi trả lời câu hỏi.
  - **Khối bên phải - Nội dung câu hỏi và form trả lời:**
    - **"Thời gian làm bài":** Một khu vực hiển thị đồng hồ đếm ngược hoặc thời gian còn lại cho bài kiểm tra.
    - **"Button[Nộp bài]":** Nút để nộp bài kiểm tra. Có một nút nộp bài ở đầu và một nút ở cuối, có thể là để tiện lợi cho người dùng.

- "Nội dung câu hỏi và form trả lời câu hỏi": Khu vực này sẽ chứa từng câu hỏi và các trường nhập liệu hoặc lựa chọn đáp án tương ứng (ví dụ: ô nhập văn bản, radio buttons, checkboxes, v.v.).



Hình 3.18 Sơ đồ cấu trúc trang luyện tập

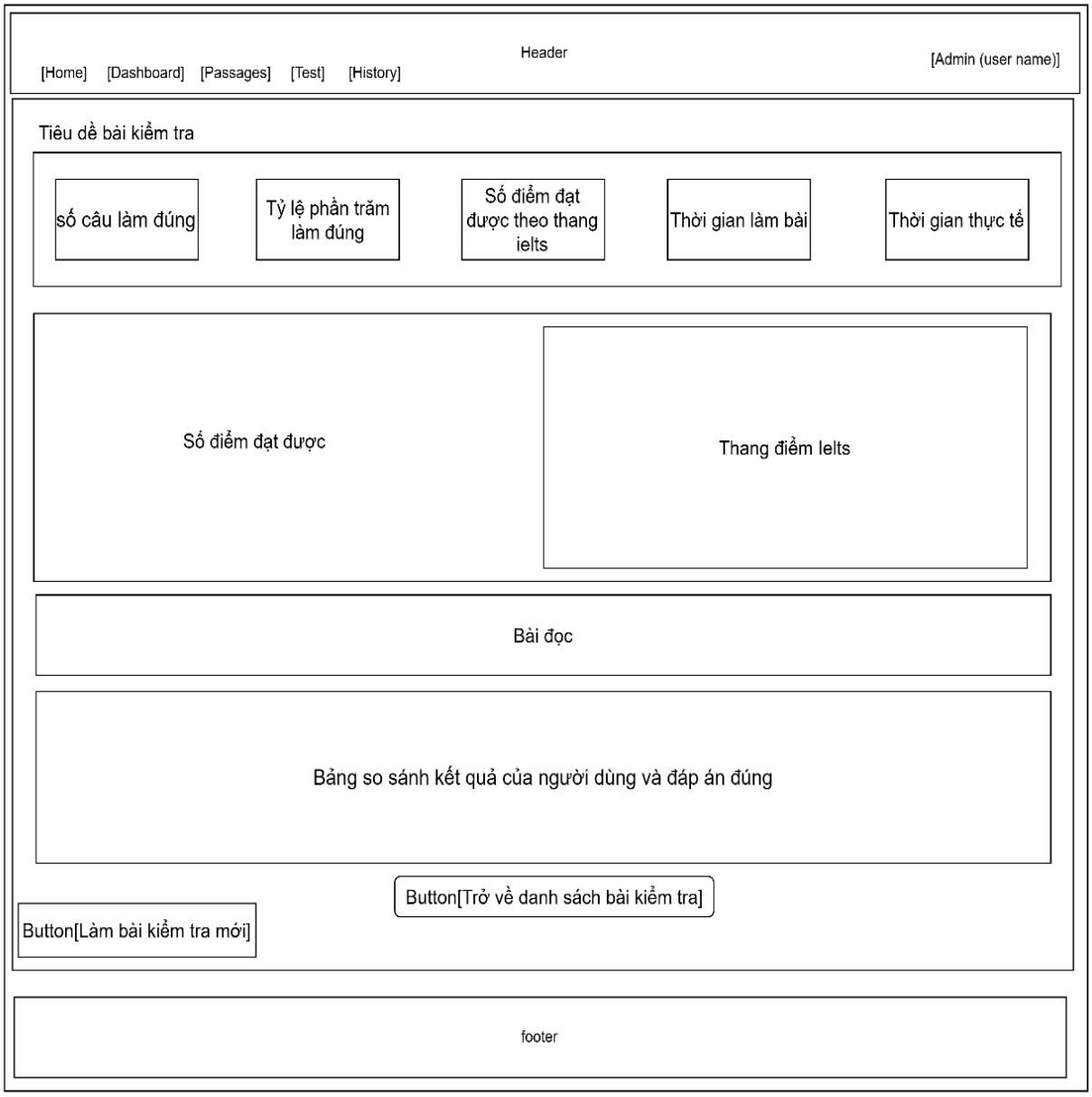
❖ **Giao diện Kết quả Bài kiểm tra (Test Result)**

- **Phần nội dung chính (Kết quả Bài kiểm tra):**
  - **Tiêu đề "Kết quả Test":** Tiêu đề chính của trang.
  - **Nút "Back to Test":** Nút để quay lại trang danh sách các bài kiểm tra hoặc trang tổng quan bài kiểm tra.
  - **Khu vực "Tiêu đề tóm tắt":** Cung cấp các thông tin tổng quan về kết quả:
    + **"Thông kê số câu đúng":** Hiển thị số câu trả lời đúng.

+ **"Thông kê điểm theo thang điểm ielts":** Hiển thị điểm số đạt được theo thang điểm ielts.
+ **"Thông kê phần trăm":** Hiển thị điểm số dưới dạng phần trăm.
+ **"Thời gian làm bài":** Thời gian đã dùng để hoàn thành bài kiểm tra.
+ **"Thời gian thực tế":** Có thể là thời gian giới hạn của bài kiểm tra.

- **Khu vực "Hiển thị thang điểm quy đổi ielts":**
  + **Số điểm đạt được**: Hiện thị điểm số.
  + **Thang điểm IELTS**: Hiển thị biểu đồ quy đổi điểm sang thang IELTS.
- **Khu vực "Tiêu đề bài test" và nội dung bài đọc:**
  + "Tiêu đề bài test": Tên của bài kiểm tra.
  + "Nội dung bài đọc": Hiển thị nội dung của bài đọc liên quan đến bài kiểm tra.
- **Khu vực "Thứ tự câu hỏi, kết quả đáp án và điểm":**
  + "Bảng so sánh kết quả người dùng và kết quả đúng": Đây là phần quan trọng nhất, hiển thị chi tiết từng câu hỏi, đáp án của người dùng, đáp án đúng và điểm đạt được cho mỗi câu.
- **Nút "Button[Trở lại]":** Nút quay lại trang trước.
- **Nút "Button bài test mới":** Nút để làm một bài kiểm tra mới.

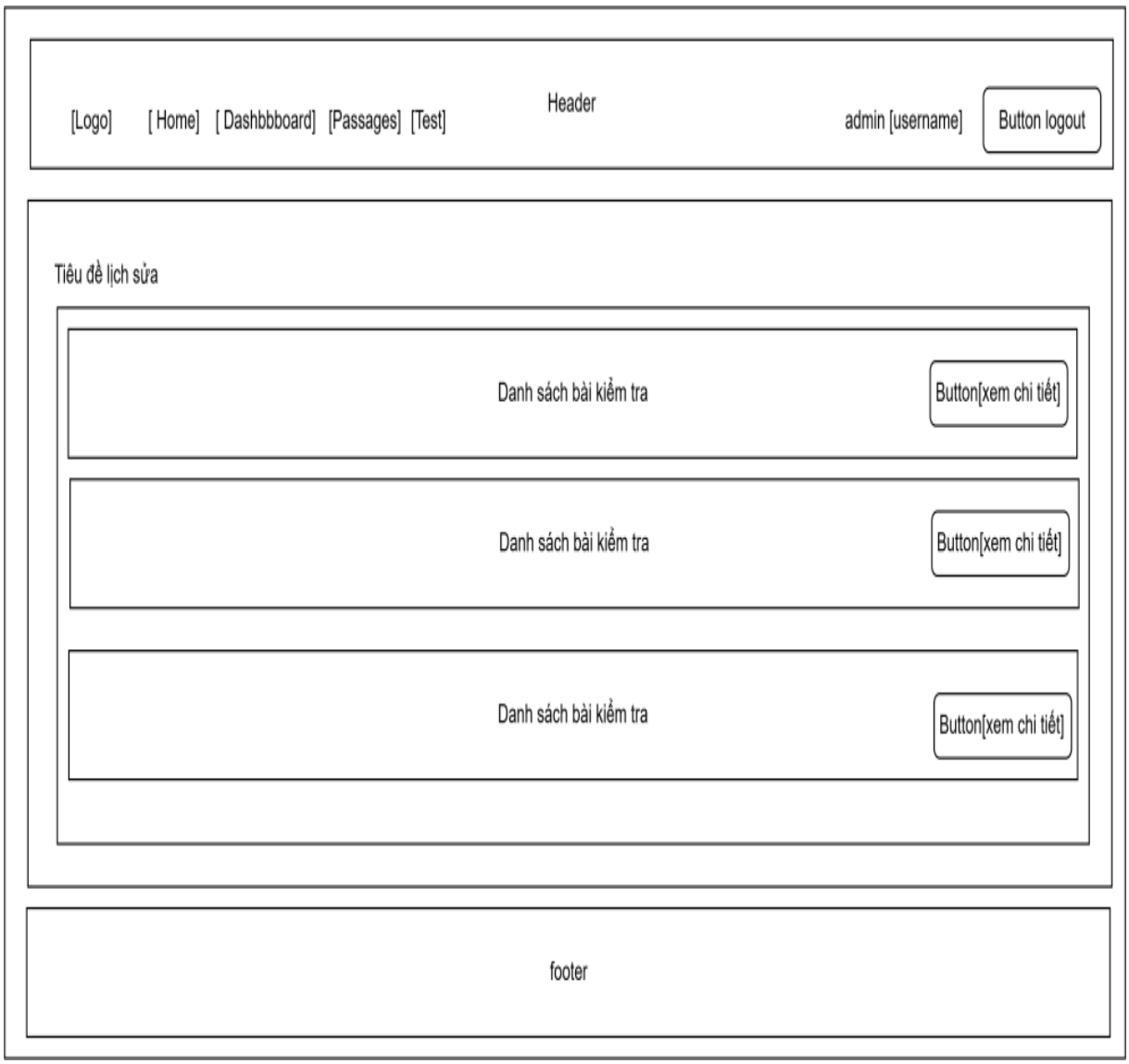
Sơ đồ cấu trúc kết quả bài kiểm tra như hình 3.19 (Trang sau)



Hình 3.19 Sơ đồ cấu trúc kết quả bài kiểm tra

❖ **Giao diện Lịch sử Làm bài (History – Chỉ hiển thị đối với người dùng đăng nhập)**

- **Phần nội dung chính (Lịch sử làm bài kiểm tra):**
  - **Tiêu đề "Lịch sử":** Tiêu đề chính của trang.
  - **Các khối "Danh sách bài kiểm tra":** Mỗi khối đại diện cho một bài kiểm tra đã hoàn thành. Trong mỗi khối có:
    + "Danh sách bài kiểm tra": Đây là khu vực sẽ hiển thị thông tin tóm tắt về bài kiểm tra đã làm (ví dụ: tên bài kiểm tra, ngày làm, điểm số, v.v.).
    + **"Button[xem chi tiết]":** Nút để xem kết quả chi tiết của bài kiểm tra đó (như trang kết quả đã phân tích trước đó).



Hình 3.20 Sơ đồ cấu trúc trang lịch sử

Toàn bộ các giao diện này tạo nên một luồng trải nghiệm người dùng khá hoàn chỉnh và chi tiết, từ việc chọn lựa, chuẩn bị, thực hiện bài kiểm tra cho đến việc nhận phản hồi về kết quả và theo dõi tiến trình học tập.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1. Cài đặt công nghệ

4.1.1. Môi trường phát triển

Hệ thống Reading Practice System được phát triển trên nền tảng Node.js, một môi trường thực thi JavaScript phía máy chủ mạnh mẽ, cho phép xây dựng các ứng dụng web hiệu năng cao và có khả năng mở rộng tốt. Kiến trúc phần mềm cốt lõi của hệ thống tuân theo mô hình MVC (Model-View-Controller), giúp phân tách rõ ràng các thành phần xử lý dữ liệu (Model), logic nghiệp vụ (Controller), và giao diện người dùng (View). Việc áp dụng kiến trúc này không chỉ giúp mã nguồn được tổ chức một cách khoa học mà còn tạo điều kiện thuận lợi cho việc bảo trì và phát triển các tính năng mới trong tương lai.

4.1.1.1. Yêu cầu hệ thống

Để đảm bảo quá trình phát triển và vận hành hệ thống diễn ra suôn sẻ, các yêu cầu về môi trường và phiên bản công nghệ cụ thể được đặt ra như sau:

- **Node.js**: Yêu cầu phiên bản 16.0 trở lên.
- **MongoDB**: Sử dụng phiên bản 5.0 trở lên làm hệ quản trị cơ sở dữ liệu NoSQL.
- **NPM (Node Package Manager)**: Yêu cầu phiên bản 8.0 trở lên để quản lý các gói phụ thuộc.
- **Trình duyệt web**: Hệ thống đảm bảo tương thích và hiển thị tối ưu trên các trình duyệt web hiện đại phổ biến như Chrome, Firefox, Safari, và Edge (phiên bản mới nhất).

4.1.1.2. Cấu trúc công nghệ Backend

Backend của hệ thống được xây dựng chủ yếu dựa trên Node.js và framework Express.js, cung cấp một nền tảng vững chắc để xây dựng các API và xử lý logic nghiệp vụ. Mongoose ODM được sử dụng để tương tác với cơ sở dữ liệu MongoDB một cách hiệu quả. Dưới đây là danh sách các thư viện (dependencies) chính được sử dụng phía backend, trích từ tệp package.json:

JavaScript

```
// package.json - Các dependency chính của Backend
{
  "dependencies": {
```

```
    "express": "^4.18.2",              // Web framework linh hoạt và tối giản cho
Node.js
    "mongoose": "^7.5.0",         // MongoDB ODM (Object Data Modeling)
    "ejs": "^3.1.9",              // Template engine (Embedded JavaScript)
    "express-session": "^1.17.3",  // Middleware quản lý session
    "bcrypt": "^5.1.1",           // Thư viện mã hóa mật khẩu
    "jsonwebtoken": "^9.0.1",      // Tạo và xác thực JSON Web Tokens
    "express-ejs-layouts": "^2.5.1",// Hỗ trợ sử dụng layout cho EJS
    "connect-mongo": "^5.0.0",     // Lưu trữ session vào MongoDB
    "method-override": "^3.0.0",   // Cho phép sử dụng các HTTP verbs như PUT,
DELETE từ forms
    "multer": "^1.4.5-lts.1",      // Middleware xử lý việc tải lên file
    "moment": "^2.29.4"            // Thư viện xử lý và định dạng ngày giờ
  }
}
```

Các thư viện này hỗ trợ việc xây dựng các tính năng cốt lõi như định tuyến (routing), quản lý cơ sở dữ liệu, xác thực người dùng, quản lý phiên làm việc, và nhiều tác vụ khác.

4.1.1.3. Cấu trúc công nghệ Frontend

Giao diện người dùng của hệ thống chủ yếu được render từ phía máy chủ (server-side rendering) sử dụng EJS làm template engine. Để xây dựng giao diện người dùng một cách nhanh chóng, nhất quán và đáp ứng (responsive), Tailwind CSS đã được lựa chọn làm CSS framework chính.

```
JavaScript
// package.json (phần liên quan đến build CSS cho frontend)
{
  "dependencies": {
    "tailwindcss": "^3.4.17",      // Utility-first CSS framework
    "@tailwindcss/vite": "^4.1.4"   // (Nếu sử dụng Vite) Tích hợp Tailwind với
    Vite
  },
  "devDependencies": {
```

```
    "autoprefixer": "^10.4.15",     // Công cụ PostCSS để tự động thêm vendor
    prefixes
    "postcss": "^8.4.28"          // Công cụ xử lý CSS bằng JavaScript plugins
  }
}
```

Tailwind CSS, cùng với các công cụ hỗ trợ như Autoprefixer và PostCSS, giúp tối ưu hóa quá trình phát triển giao diện và đảm bảo tính tương thích trên nhiều trình duyệt.

4.1.2. Cài đặt và triển khai

Quy trình cài đặt hệ thống từ mã nguồn và các cấu hình ban đầu được chuẩn hóa để đảm bảo tính đồng nhất và thuận tiện cho việc thiết lập môi trường.

4.1.2.1. Quy trình cài đặt

Các bước cơ bản để cài đặt và khởi chạy hệ thống:

```
Bash
# 1. Clone repository từ hệ thống quản lý phiên bản (ví dụ: Git)
git clone <repository-url>
cd reading-practice-system
# 2. Cài đặt các thư viện phụ thuộc cho dự án
npm install
# 3. Cấu hình các biến môi trường
# Sao chép tệp .env.example thành .env và cập nhật các giá trị cần thiết
# (MONGODB_URI, SESSION_SECRET, etc.)
cp .env.example .env
# Mở tệp .env và điền thông tin cấu hình
# 4. Build CSS với Tailwind (nếu quy trình build CSS được tách riêng)
npm run tailwind:build
# 5. Khởi chạy ứng dụng
npm run dev   # Chế độ phát triển (thường có hot-reloading)
npm start     # Chế độ sản phẩm
```

4.1.2.2. Cấu hình cơ sở dữ liệu

Kết nối tới cơ sở dữ liệu MongoDB được quản lý tập trung, thường trong một tệp cấu hình riêng (config/database.js), sử dụng Mongoose:

JavaScript

```
// config/database.js
const mongoose = require('mongoose');
const connectDB = async () => {
  try {
    const conn = await mongoose.connect(process.env.MONGODB_URI, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    });
    console.log(`MongoDB Connected: ${conn.connection.host}`);
  } catch (error) {
    console.error('Database connection error:', error);
    process.exit(1); // Thoát ứng dụng nếu không kết nối được CSDL
  }
};
module.exports = connectDB;
```

Hàm connectDB này được gọi khi ứng dụng khởi động để thiết lập kết nối tới MongoDB.

4.1.2.3. Cấu hình session và authentication

Quản lý phiên làm việc của người dùng được thực hiện thông qua express-session, với session data được lưu trữ trong MongoDB nhờ connect-mongo để đảm bảo tính bền vững.

JavaScript

```
// config/session.js (hoặc trong app.js)
const session = require('express-session');
const MongoStore = require('connect-mongo');
const sessionConfig = {
  secret: process.env.SESSION_SECRET, // Khóa bí mật cho session
  resave: false,
  saveUninitialized: true, // Lưu session mới dù chưa có dữ liệu
  cookie: {
```

```
    secure: process.env.NODE_ENV === 'production', // Cookie chỉ gửi qua HTTPS ở production
    maxAge: 24 * 60 * 60 * 1000 // Thời gian sống của cookie: 24 giờ
  },
  store: new MongoStore({ // Lưu trữ session vào MongoDB
    mongoUrl: process.env.MONGODB_URI,
    touchAfter: 24 * 3600 // Thời gian (giây) cập nhật lại session
  })
};
// Áp dụng middleware: app.use(session(sessionConfig));
```

Cấu hình này giúp quản lý trạng thái đăng nhập của người dùng một cách an toàn và hiệu quả.

4.2. Kết nối Frontend - Backend

Hệ thống Reading Practice System chủ yếu sử dụng mô hình server-side rendering với EJS, nơi backend Node.js/Express chịu trách nhiệm xử lý logic và render trực tiếp các trang HTML cho client. Kiến trúc MVC được áp dụng xuyên suốt để tổ chức mã nguồn.

4.2.1. Kiến trúc MVC

Mô hình MVC (Model-View-Controller) giúp phân tách rõ ràng các thành phần của ứng dụng:

- **Models (models/):** Đại diện cho dữ liệu và logic tương tác CSDL (ví dụ: User.js, Test.js, Question.js, ReadingPassage.js, UserAttempt.js, TestPool.js sử dụng Mongoose).
- **Views (views/):** Các tệp EJS chịu trách nhiệm hiển thị giao diện người dùng, được tổ chức theo layouts và các phân mục chức năng (auth, user, admin).
- **Controllers (controllers/):** Xử lý yêu cầu từ client, tương tác với Models và quyết định View nào sẽ được render (ví dụ: authController.js, testController.js).
- **Routes (routes/):** Định nghĩa các URL endpoint và ánh xạ chúng tới các hàm xử lý trong Controllers.

Cấu trúc thư mục dự án thể hiện rõ ràng việc áp dụng mô hình này:

```
reading-practice-system/
├── models/          # Tầng Dữ liệu
├── controllers/     # Tầng Logic Nghiệp vụ
├── views/           # Tầng Trình diễn
└── routes/          # Định nghĩa Đường dẫn
```

4.2.2. API và Cấu hình Route (Route Configuration)

Hệ thống định nghĩa các routes để xử lý yêu cầu HTTP, tuân theo các nguyên tắc RESTful cho việc quản lý tài nguyên và các routes phục vụ chức năng cụ thể của người dùng.

4.2.2.1. Cấu trúc API RESTful (Ví dụ)

Các tệp trong thư mục routes/ (ví dụ: test.js, user.js) định nghĩa các endpoints như:

- GET /tests: Lấy danh sách tất cả bài kiểm tra.
- POST /tests: Tạo một bài kiểm tra mới.
- GET /practice: Lấy danh sách bài kiểm tra cho người dùng luyện tập.
- POST /attempt: Bắt đầu một lượt làm bài.
- POST /submit: Nộp bài kiểm tra.
- GET /results/:id: Xem kết quả chi tiết của một lượt làm bài.

4.2.2.2. Lưu trữ câu trả lời thời gian thực (Real-time Answer Saving)

Để cải thiện trải nghiệm người dùng, một API endpoint (POST/api/attempts/:attemptId/answers) được thiết kế để cho phép client (JavaScript) gửi và lưu trữ câu trả lời của người dùng một cách định kỳ hoặc khi có thay đổi, mà không cần đợi đến khi nộp toàn bộ bài. Logic xử lý phía backend sẽ tìm lượt làm bài (UserAttempt) và cập nhật hoặc thêm mới câu trả lời vào CSDL.

4.2.3. Template Engine và Render Giao diện (View Rendering)

EJS (Embedded JavaScript) được sử dụng làm template engine chính. express-ejs-layouts được tích hợp để sử dụng các tệp layout chung, giúp tái sử dụng mã HTML và duy trì tính nhất quán cho giao diện.

4.2.3.1. Cấu hình EJS Template Engine

Trong tệp khởi tạo ứng dụng Express, EJS được thiết lập làm view engine mặc định, cùng với việc chỉ định thư mục chứa các tệp view và tệp layout chính (views/layouts/main.ejs).

4.2.3.2. Render Nội dung Động (Dynamic Content Rendering)

Dữ liệu từ controllers được truyền vào các tệp EJS. Bên trong view, các thẻ EJS (<% %> cho logic và <%= %> để hiển thị biến) được sử dụng để render nội dung HTML động, ví dụ như hiển thị danh sách các bài kiểm tra với thông tin chi tiết của từng bài.

4.2.3.3 Giao diện và chức năng của hệ thống

Giao diện hệ thống được thiết kế với tiêu chí hiện đại, trực quan, dễ sử dụng và thân thiện với người dùng. Việc kết hợp EJS template engine với Tailwind CSS giúp xây dựng giao diện linh hoạt, dễ tùy chỉnh và đáp ứng tốt trên nhiều thiết bị.

❖ **Giao diện trang chủ**

Trang chủ là nơi đầu tiên người dùng tiếp cận khi truy cập vào hệ thống. Giao diện được chia thành các khu vực chính, mỗi khu vực thể hiện một nhóm chức năng cụ thể:

❖ **Header**

- **Nút Home**: là liên kết của trang chủ.
- **Nút "Login" và "Register"**: chuyển hướng đến trang đăng nhập và đăng ký.
- Giao diện sử dụng màu sắc nổi bật để thu hút sự chú ý vào các hành động chính.

- **Khối chính – Giới thiệu hệ thống luyện thi**

+ **Tiêu đề chính**: Master Reading Comprehension – nhấn mạnh mục tiêu hệ thống là giúp người dùng nâng cao kỹ năng đọc hiểu.

+ **Nội dung giới thiệu ngắn gọn**: mô tả về các loại bài đọc và phương pháp học tập tương tác.

+ **Nút "Start Practice"**: chuyển đến phần luyện tập mà không cần đăng nhập (dành cho người dùng khách).

+ **Hình ảnh minh họa**: tạo cảm giác chuyên nghiệp và phù hợp với chủ đề học thuật.

- **Tính năng nổi bật**

Giao diện sử dụng các hộp nội dung biểu tượng hóa, giúp làm nổi bật các tính năng chính:

+ **Diverse Question Types**: hỗ trợ 5 loại câu hỏi (Multiple Choice, Fill in the Blank, Matching, True/False/Not Given, Short Answer).
+ **Progress Tracking**: theo dõi tiến trình học tập của người dùng qua thống kê.
+ **Automatic Scoring**: hệ thống chấm điểm tự động ngay sau khi làm bài.
+ **Extensive Library**: thư viện tài liệu phong phú, phân loại theo chủ đề và độ khó.
+ **So sánh quyền truy cập: Guest vs Registered Users**

  Người dùng được phân loại rõ ràng thành: Đăng nhập và không đăng nhập.
+ **Guest Users**:
  - Làm bài không cần tài khoản.
  - Xem điểm ngay sau khi nộp bài.
  - Không lưu lịch sử làm bài.
  - Dữ liệu sẽ mất nếu tải lại trang.
+ **Registered Users**:
  - Sử dụng đầy đủ chức năng hệ thống.
  - Lưu lại lịch sử làm bài, thống kê tiến độ.
  - Dữ liệu được bảo mật.

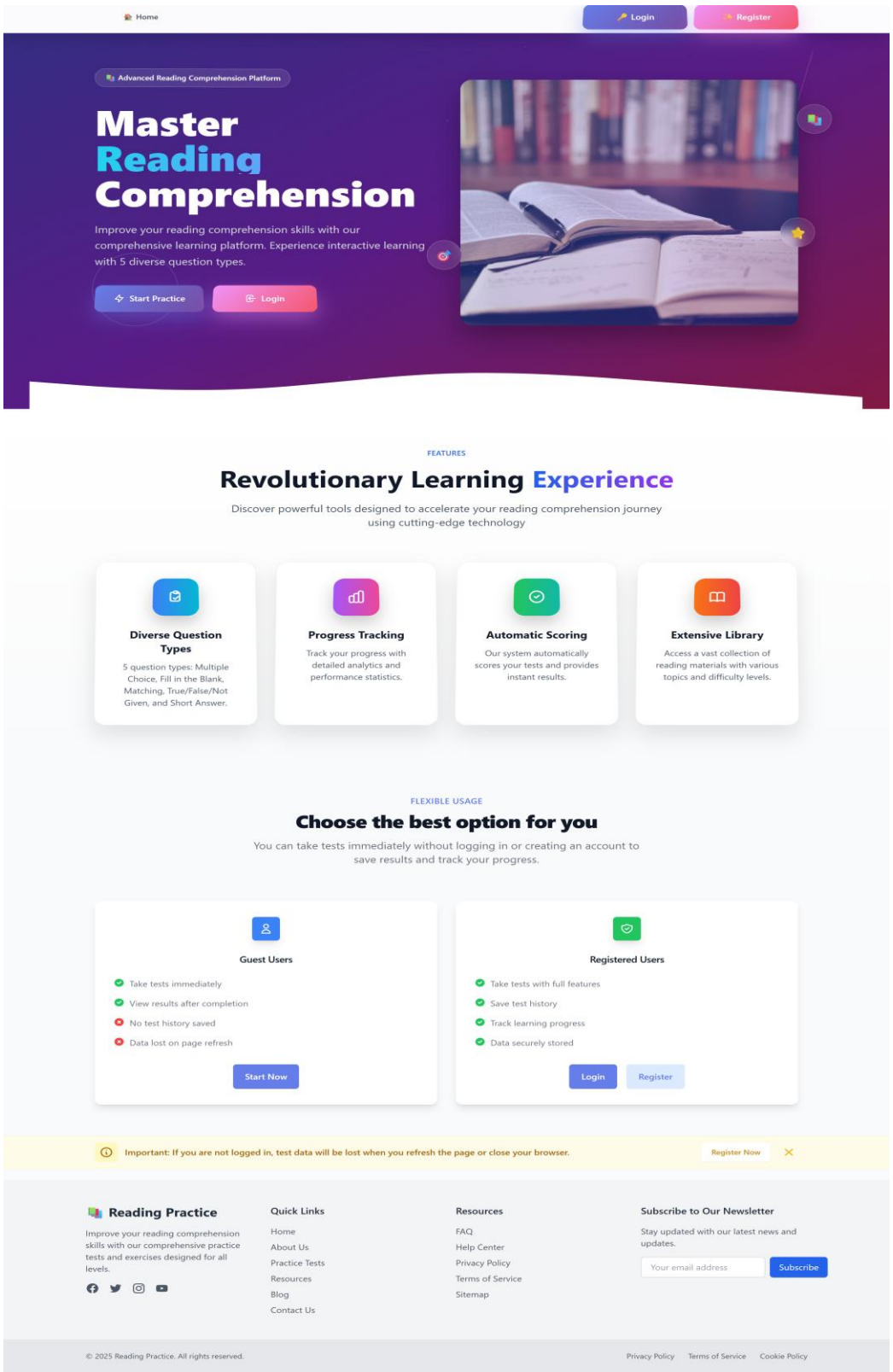
Phần này giúp người dùng tự lựa chọn cách sử dụng phù hợp, từ đó tăng trải nghiệm người dùng và tỷ lệ đăng ký tài khoản.

- **Footer – Thông tin bổ trợ**

Các liên kết ty cập nhanh (Quick Links), chính sách (Privacy, Terms), và ô đăng ký nhận bản tin.

Đảm bảo người dùng luôn có thể truy cập các nội dung liên quan.

Giao diện trang chủ được trình bày tại Hình 4.1 (Trang sau)



Hình 4.1 Giao diện trang chủ

❖ **Giao diện đăng nhập – Sign in to your account**

Trang đăng nhập là nơi người dùng đã đăng ký có thể truy cập vào hệ thống để sử dụng đầy đủ các chức năng như làm bài thi có lưu kết quả, theo dõi tiến độ học tập, và truy cập tài nguyên mở rộng.

- **Chức năng chính:**

+ **Nhập địa chỉ email** và **mật khẩu** để xác thực.

+ **Checkbox "Remember me"**: lưu thông tin đăng nhập cho những lần sau.

+ **Liên kết "Forgot your password?"**: hỗ trợ khôi phục mật khẩu trong trường hợp người dùng quên.

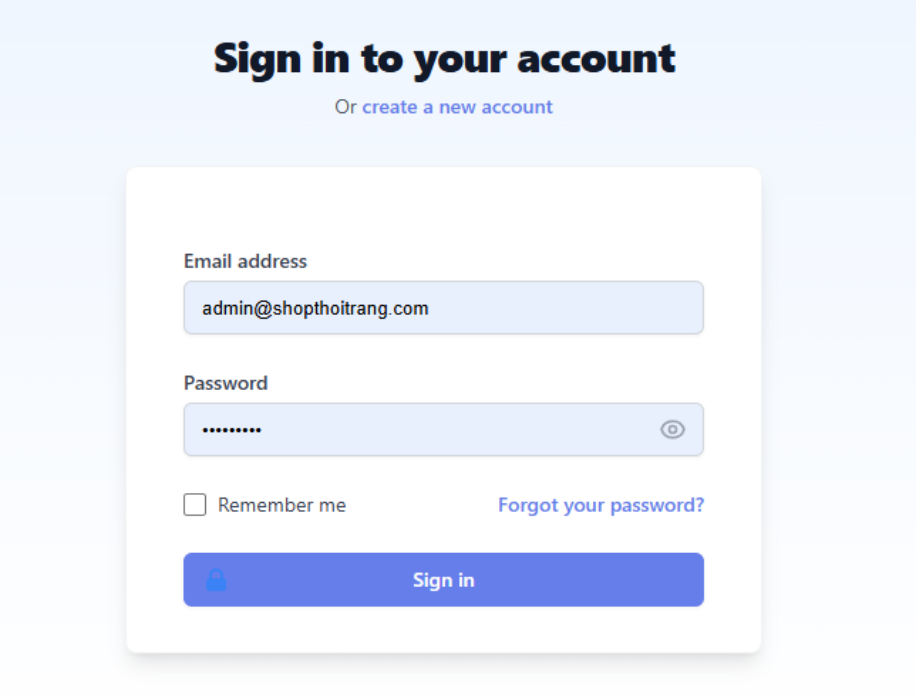
+ **Liên kết "Create a new account"**: chuyển hướng sang trang đăng ký nếu chưa có tài khoản.

- **Kỹ thuật triển khai:**

+ Giao diện được render bằng EJS, dữ liệu người dùng được gửi về backend bằng POST request.

+ Quá trình xác thực được xử lý thông qua middleware xác thực kết nối MongoDB để kiểm tra thông tin đăng nhập.

+ Khi đăng nhập thành công, hệ thống sử dụng JWT hoặc session để quản lý trạng thái người dùng.



Hình 4.2 Giao diện trang đăng nhập

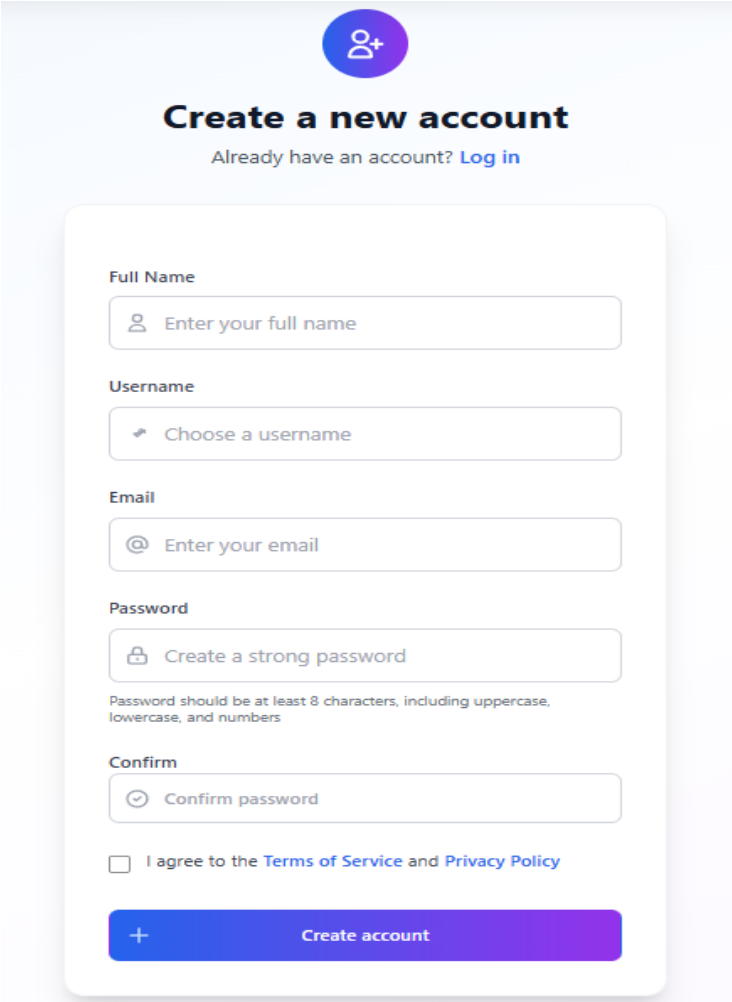
❖ **Giao diện đăng ký – Create a new account**

Trang đăng ký tài khoản cho phép người dùng mới khởi tạo tài khoản để sử dụng toàn bộ chức năng của hệ thống, bao gồm lưu kết quả làm bài, theo dõi tiến trình và truy cập tài nguyên nâng cao.

- **Chức năng chính:**

  Nhập thông tin cá nhân gồm:

  - **Full name**: tên đầy đủ của người dùng.
  - **Username**: tên định danh hệ thống.
  - **Email**: dùng để xác thực và phục hồi mật khẩu.
  - **Password + Confirm**: đảm bảo bảo mật và xác minh chính xác.
  - **Checkbox đồng ý điều khoản**: đảm bảo người dùng đồng ý với chính sách sử dụng và quyền riêng tư.
  - **Nút "Create account"**: gửi dữ liệu đến hệ thống để tạo tài khoản.
  - **Liên kết "Log in"**: quay lại trang đăng nhập nếu đã có tài khoản.
- **Kỹ thuật triển khai:**
  - Các trường dữ liệu được kiểm tra đầu vào (validation) phía client trước khi gửi.
  - Mật khẩu được mã hóa (hash) trước khi lưu vào cơ sở dữ liệu MongoDB.
  - Thông tin người dùng được lưu trữ có tổ chức trong collection users.



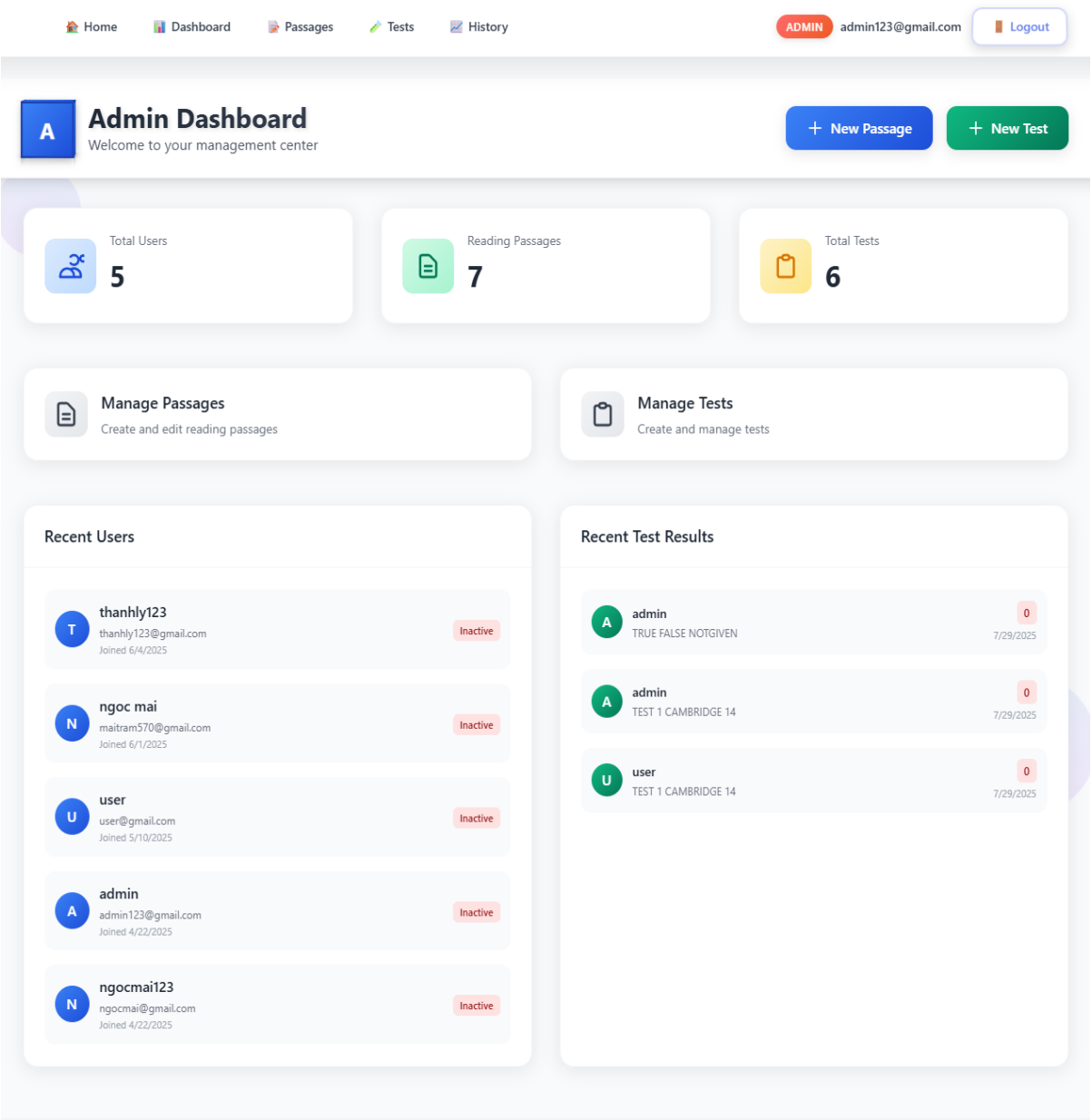
Hình 4.3 Giao diện trang đăng ký

❖ **Giao diện quản trị viên – Admin Dashboard**

Dashboard dành cho quản trị viên là trung tâm quản lý nội dung và người dùng trên hệ thống. Từ đây, admin có thể theo dõi tổng quan hoạt động, thêm bài đọc, tạo đề kiểm tra và quản lý người dùng.

- **Chức năng chính:**
  - **Thống kê nhanh**:
    + **Total Users**: tổng số tài khoản hiện có.
    + **Reading Passages**: tổng số bài đọc đã tạo.
    + **Total Tests**: số đề kiểm tra đã được thiết lập.
  - **Quản lý nội dung**:
    + **Manage Passages**: điều hướng đến trang tạo và chỉnh sửa bài đọc.
    + **Manage Tests**: điều hướng đến trang tạo và cập nhật bài kiểm tra.
    + **Danh sách người dùng gần đây**: Hiển thị email và trạng thái hoạt động (Active/Inactive), dễ dàng theo dõi các tài khoản mới tạo.
    + **Kết quả kiểm tra gần đây**: Hiển thị tên người làm bài, tên bài kiểm tra và thời gian làm gần nhất, Giúp admin theo dõi hoạt động học tập tổng quát.
  - **Kỹ thuật triển khai:**
    + Giao diện được render bằng EJS, dữ liệu được lấy từ MongoDB.
    + Các chức năng quản lý sử dụng các route riêng với quyền truy cập dành cho role admin.
    + Tích hợp các controller để tạo/sửa/xóa bài đọc và bài kiểm tra thông qua API nội bộ.

Giao diện dashboard của admin được trình bài tại Hình 4.4 (trang sau)



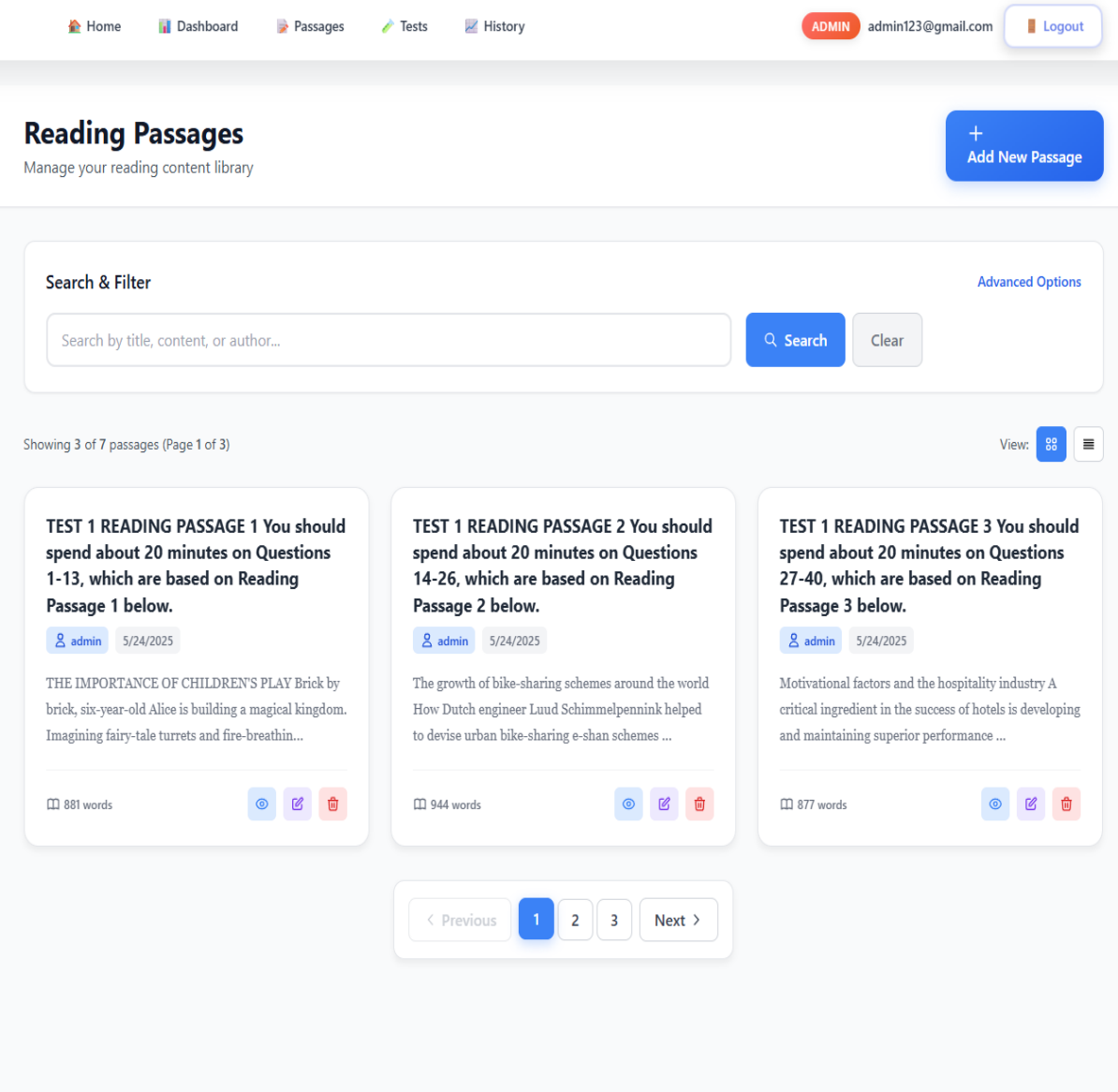
Hình 4.4 Giao diện dashboard của admin

❖ **Giao diện quản lý bài đọc – Reading Passages**

Trang này hiển thị toàn bộ các bài đọc đã được tạo và lưu trong hệ thống. Quản trị viên có thể tìm kiếm, lọc, chỉnh sửa hoặc xóa bài đọc một cách dễ dàng.

- **Chức năng chính**
  - **Thanh tìm kiếm**: Cho phép tìm kiếm bài đọc theo tiêu đề, nội dung hoặc tác giả, Có tùy chọn nâng cao (Advanced Options) để lọc kết quả chính xác hơn.
  - **Danh sách bài đọc**: Mỗi bài đọc được hiển thị dưới dạng thẻ (card) với các thông tin:
    + Tiêu đề bài đọc (ví dụ: TEST 1 READING PASSAGE 1…).
    + Tác giả (admin) và ngày đăng.

    + Tóm tắt nội dung (preview khoảng 2 dòng đầu).
    + Số lượng từ (word count).
    + Các nút chức năng: xem chi tiết, chỉnh sửa, xóa bài đọc.
- **Phân trang (pagination)**: nếu số lượng bài đọc nhiều, hệ thống tự động chia thành các trang.
- **Chuyển đổi chế độ xem**: Grid view hoặc List view.
- **Nút "Add New Passage"**: chuyển hướng đến trang tạo bài đọc mới.
- **Kỹ thuật triển khai:**
    - Nội dung được lấy từ MongoDB và hiển thị theo phân trang.
    - Tìm kiếm sử dụng truy vấn với điều kiện regex để lọc tiêu đề hoặc nội dung.
    - Các hành động chỉnh sửa/xóa được bảo vệ bằng phân quyền admin.

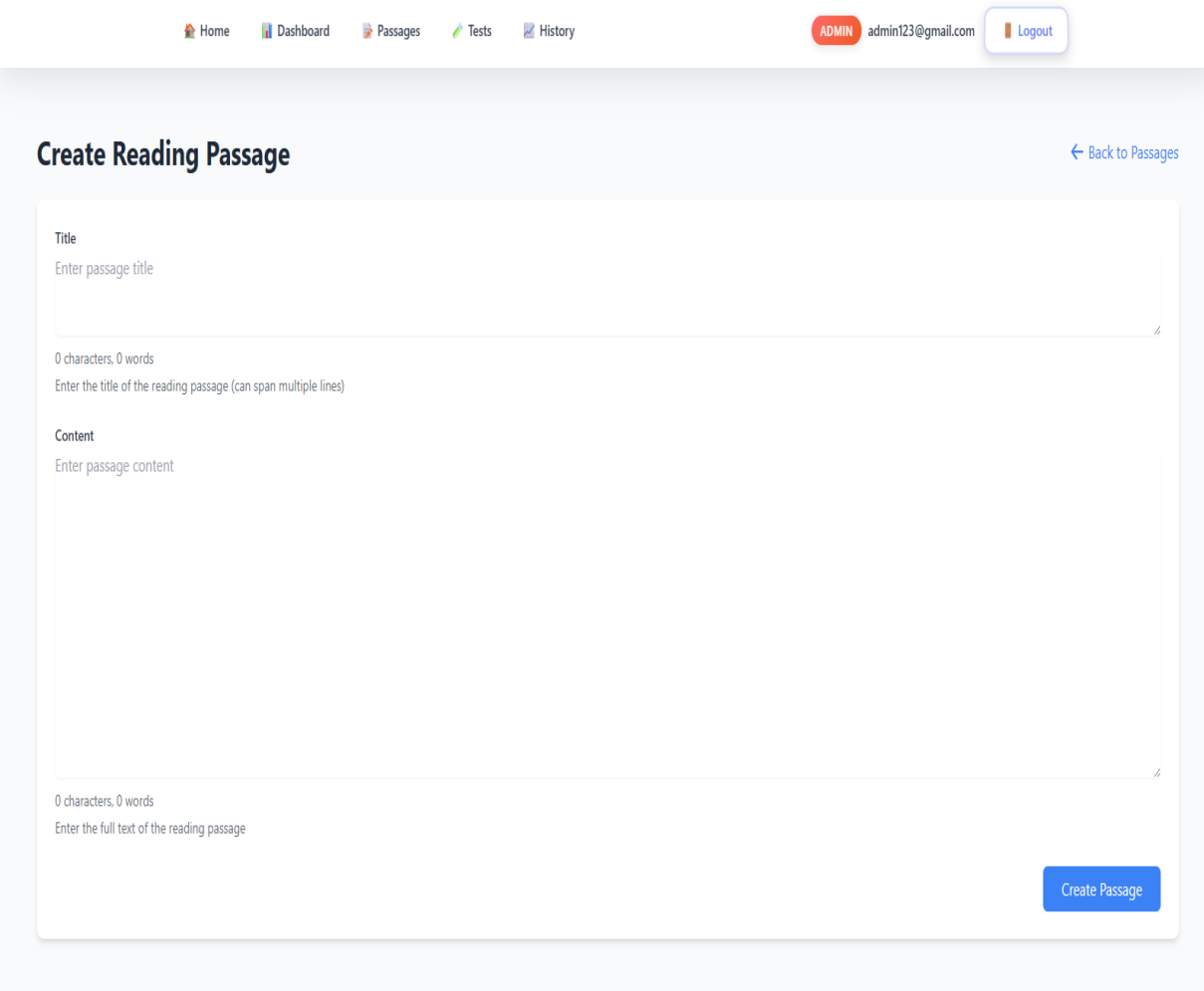


Hình 4.5 Giao diện quản lý bài đọc

❖ Giao diện tạo bài đọc – Create Reading Passage

Đây là trang giúp quản trị viên tạo nội dung cho các bài đọc hiểu trong hệ thống. Giao diện đơn giản, tập trung vào phần nhập liệu để đảm bảo việc tạo nội dung diễn ra nhanh chóng và thuận tiện.

- **Chức năng chính:**
  - **Title**: nhập tiêu đề của bài đọc. Cho phép nhập dài, có thể xuống dòng.
  - **Content**: nhập nội dung đầy đủ của bài đọc, có thể gồm nhiều đoạn.
  - **Nút "Create Passage"**: sau khi điền đầy đủ thông tin, admin nhấn nút này để lưu bài đọc vào hệ thống.
  - **Back to Passages**: quay lại danh sách các bài đọc đã tạo.
- **Kỹ thuật triển khai:**
  - Nội dung bài đọc được gửi đến backend qua POST request.
  - Dữ liệu sau khi nhập sẽ được lưu vào MongoDB trong collection passages.
  - Có kiểm tra số lượng ký tự, tránh lưu bài đọc trống.



Hình 4.6 Giao diện tạo bài đọc

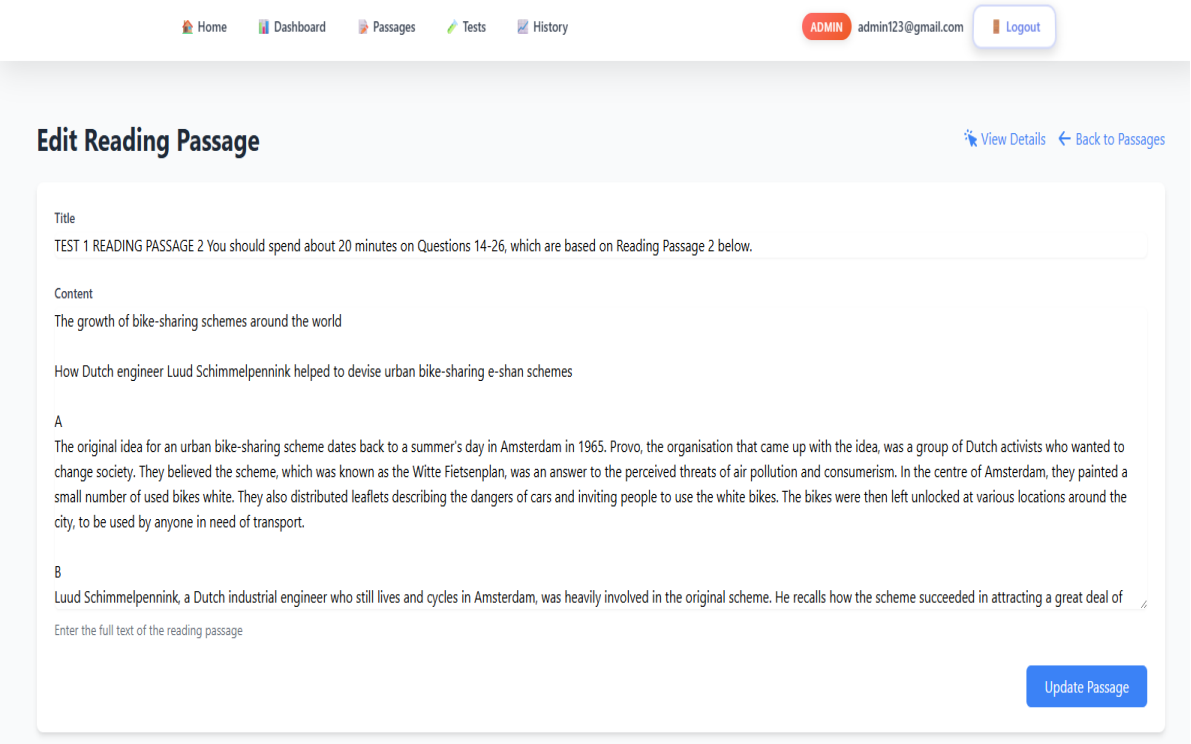
❖ Giao diện chỉnh sửa nội dung – Edit Reading Passage

Trang này cho phép quản trị viên cập nhật nội dung của một bài đọc đã có trong hệ thống. Chức năng này giúp duy trì chất lượng học liệu, dễ dàng sửa lỗi hoặc thay đổi nội dung theo nhu cầu.

- **Chức năng chính:**

  Hiển thị tiêu đề và nội dung hiện có của bài đọc.

  - **Các trường nhập liệu:**
    + **Title**: tiêu đề có thể chỉnh sửa trực tiếp.
    + **Content**: nội dung đầy đủ, có thể sửa ngay tại chỗ.
    + **Nút "Update Passage"**: lưu thay đổi vào hệ thống.
    + **View Details**: cho phép xem bài đọc ở chế độ xem hoàn chỉnh (read-only).
    + **Back to Passages**: quay lại danh sách bài đọc.
  - **Kỹ thuật triển khai:**
    + Giao diện được pre-fill dữ liệu từ database bằng GET request.
    + Sau khi chỉnh sửa, nội dung được gửi qua POST hoặc PUT để cập nhật vào MongoDB.
    + Có xác thực quyền (admin) để tránh người dùng thường truy cập trái phép.
    + Có xử lý kiểm tra ký tự tối thiểu và chống rỗng trước khi cập nhật.



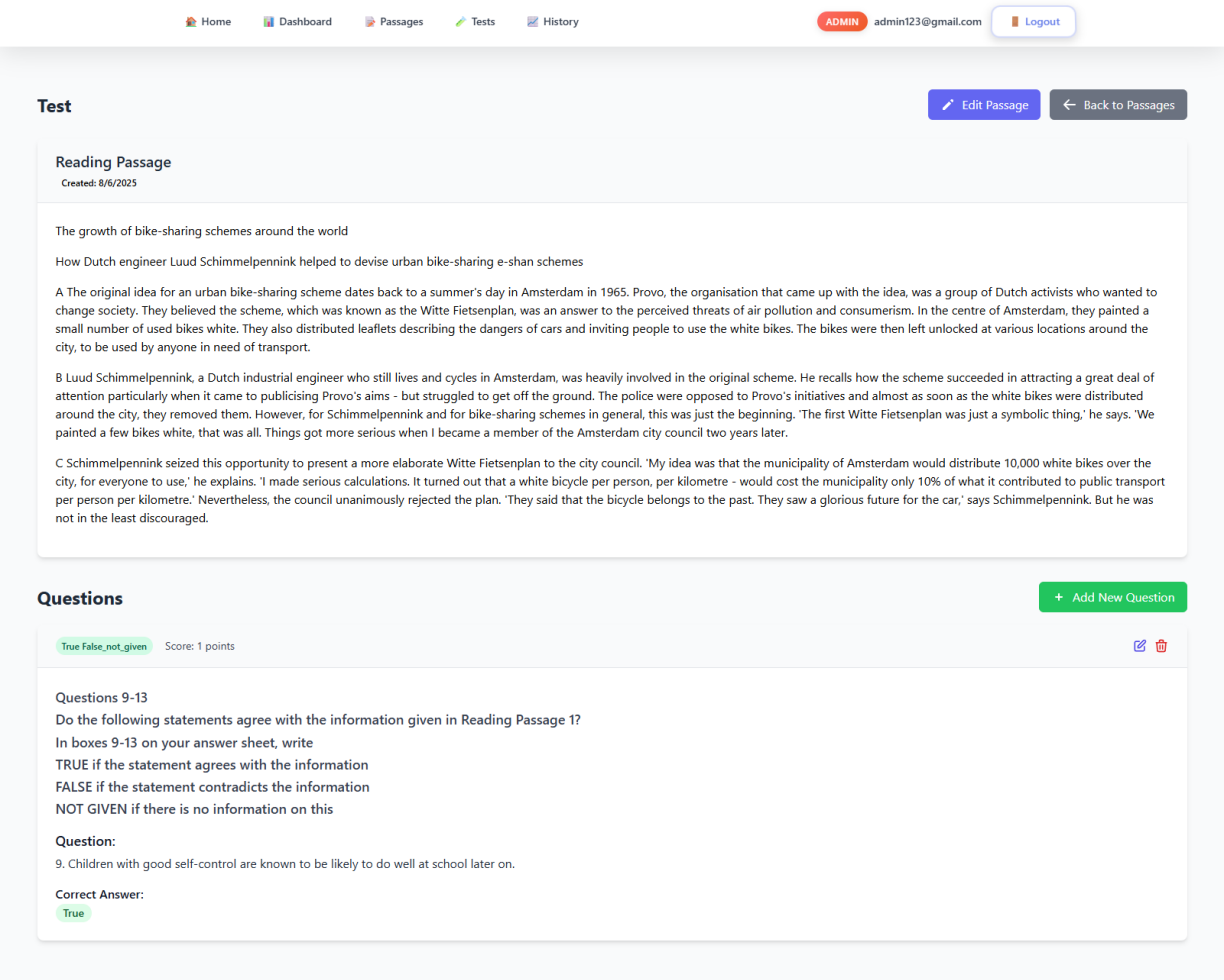
Hình 4.7 Giao diện sửa bài đọc

❖ **Giao diện danh sách câu hỏi – List questions**

Đây là khu vực cho phép quản trị viên tạo bộ câu hỏi trắc nghiệm dựa trên một đoạn văn cụ thể trong hệ thống. Giao diện này hỗ trợ thao tác dễ dàng với nội dung bài đọc và các câu hỏi liên quan.

- **Chức năng chính:**
  - **Hiển thị bài đọc đầy đủ**: Toàn bộ nội dung đoạn văn được hiển thị phía trên để người dùng tiện tham khảo trong lúc tạo câu hỏi, gồm tiêu đề bài đọc, thời gian tạo và đoạn văn.
  - **Thêm câu hỏi mới**: Nút "Add New Question" màu xanh lá cho phép thêm câu hỏi vào bài đọc hiện tại. Sau khi thêm, các câu hỏi sẽ hiển thị bên dưới với định dạng rõ ràng.
  - **chọn loại câu hỏi như:** True/False/Not Given, Multiple Choice, Fill in the blank, Matching, Short answer (hệ thống hỗ trợ tùy chọn này khi tạo).
  - **Chỉnh sửa hoặc xóa câu hỏi**: Mỗi câu hỏi đã tạo có nút chỉnh sửa (biểu tượng bút) và xóa (biểu tượng thùng rác) ở góc phải.
- **Điều hướng**
  - **Back to Passages**: quay về danh sách các bài đọc đã có trong hệ thống.
  - **Edit Passage**: chỉnh sửa nội dung bài đọc nếu phát hiện lỗi hoặc cần cập nhật.
- **Kỹ thuật triển khai:**
  - Mỗi câu hỏi được liên kết với một passage_id trong cơ sở dữ liệu để đảm bảo quản lý chính xác theo bài đọc.
  - Câu trả lời đúng (Correct Answer) được lưu kèm với mỗi câu hỏi và hiển thị rõ trong giao diện.
  - Hệ thống xác thực vai trò admin trước khi cho phép thêm/sửa/xóa dữ liệu.
  - Hỗ trợ nhiều định dạng câu hỏi, phù hợp với các dạng bài IELTS Reading

    Giao diện danh sách câu hỏi được trình bài tại Hình 4.8 (trang sau)



Hình 4.8 Giao diện danh sách câu hỏi

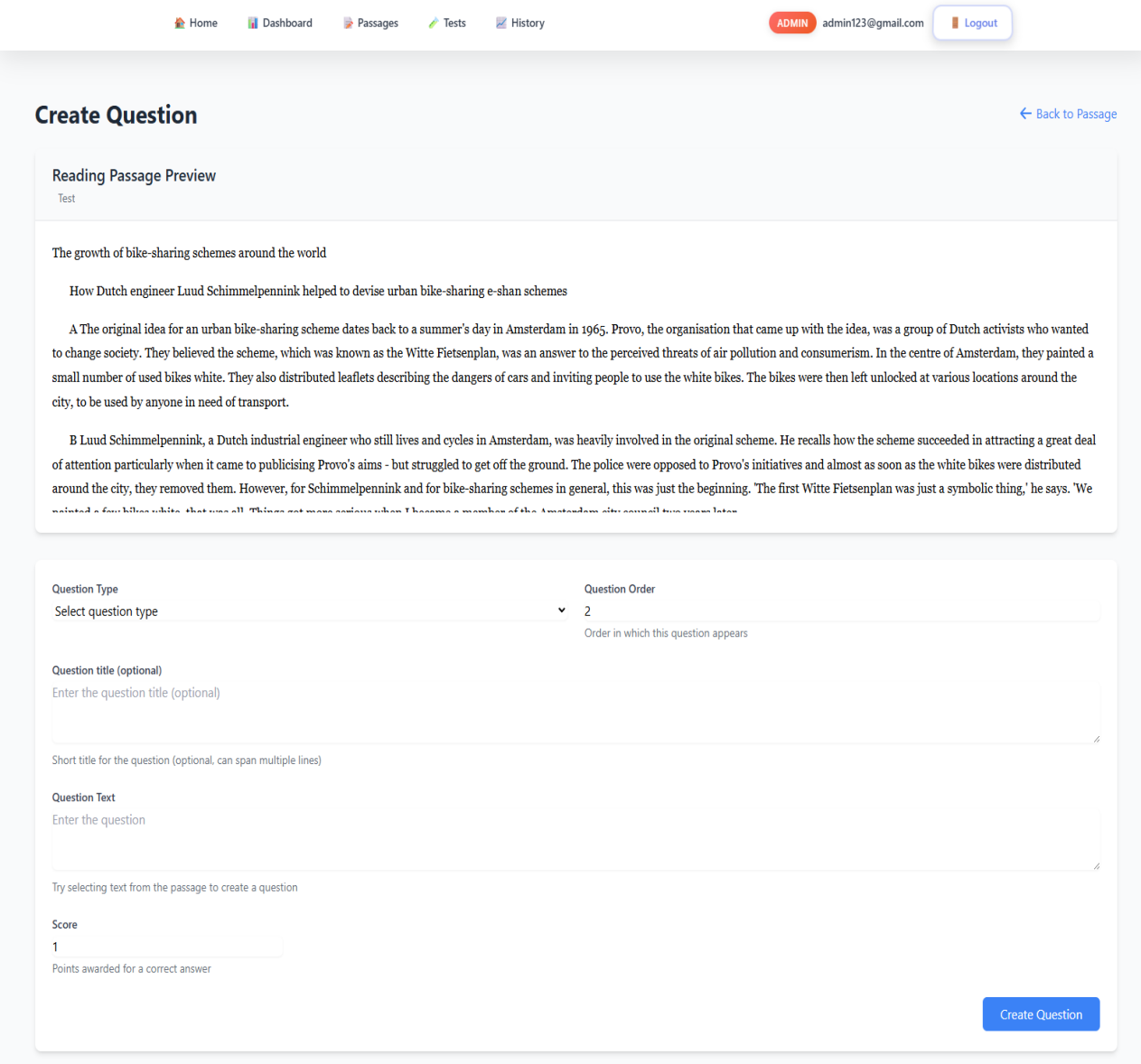
❖ Giao diện tạo câu hỏi chi tiết – Create Question

Đây là nơi quản trị viên tạo một câu hỏi cụ thể cho bài đọc đã chọn. Giao diện này nằm trong quy trình tạo đề kiểm tra từ một đoạn văn cụ thể (Reading Passage).

- **Chức năng chính:**
  - **Hiển thị bài đọc dạng xem trước (Preview)**: Toàn bộ bài đọc được hiển thị phía trên để tham khảo khi tạo câu hỏi. Người dùng có thể dễ dàng quét nội dung để viết câu hỏi phù hợp theo từng đoạn.
  - **Form tạo câu hỏi**
    + **Question Type**: menu chọn loại câu hỏi (Multiple Choice, True/False/Not Given, Fill in the Blank, Matching,...).
    + **Question Order**: số thứ tự câu hỏi xuất hiện trong đề kiểm tra.
    + **Question Title (optional)**: tên tiêu đề ngắn cho câu hỏi, có thể bỏ trống.
    + **Short Title (optional)**: tóm tắt tiêu đề giúp quản lý dễ dàng nếu danh sách dài.

+ **Question Text**: nội dung chính của câu hỏi cần tạo.
+ **Score**: điểm số tương ứng nếu trả lời đúng.
+ **Khu vực nội dung câu hỏi:** câu hỏi sẽ được hiển thị tùy theo loại câu hỏi mà người dùng muốn tạo.

- **Nút Create Question:** Khi điền xong thông tin, nhấn Create Question để lưu vào cơ sở dữ liệu và liên kết với bài đọc.
- **Kỹ thuật triển khai:**
  + Câu hỏi được lưu kèm theo passage_id, order, type, score, và các trường text liên quan.
  + Hệ thống tự kiểm tra tính hợp lệ trước khi lưu.
  + Sẽ có thông báo lỗi nếu thiếu dữ liệu cần thiết.



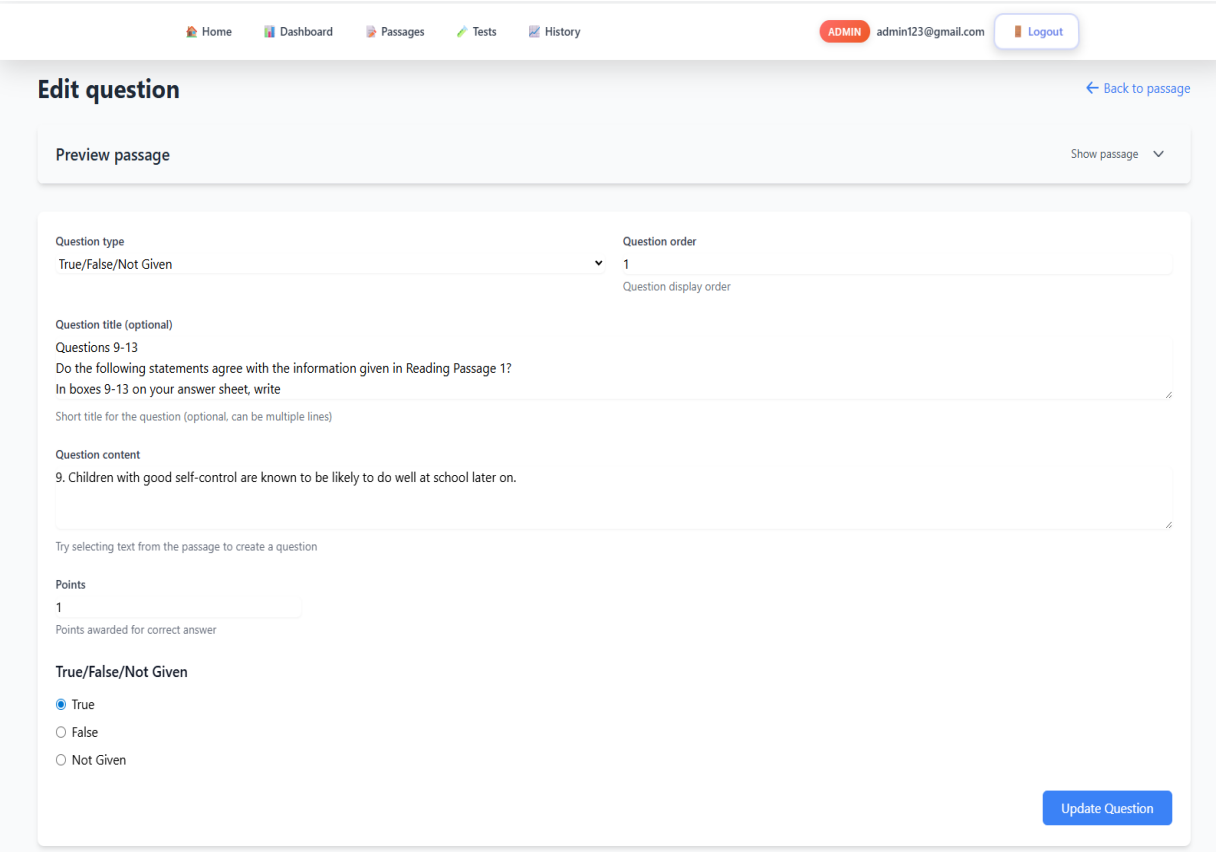
Hình 4.9 Giao diện tạo câu hỏi

❖ Giao diện chỉnh sửa câu hỏi – Edit Question

Đây là nơi cho phép quản trị viên cập nhật nội dung một câu hỏi đã tạo từ trước. Tính năng này hữu ích khi cần sửa lỗi chính tả, thay đổi đáp án, hoặc cập nhật lại thứ tự hiển thị câu hỏi trong bài kiểm tra.

- **Chức năng chính:**
  - **Xem trước bài đọc gốc (Preview passage)**: Bài đọc đầy đủ được ẩn/hiện theo nhu cầu (nút Show passage ở góc phải), giúp người dùng đối chiếu nội dung câu hỏi nhanh chóng.
  - **Thông tin câu hỏi**
    + Question Type: loại câu hỏi hiện tại không thể đổi khi chỉnh sửa.
    + Question Order: thứ tự hiển thị của câu hỏi trong đề (có thể thay đổi số thứ tự nếu muốn sắp xếp lại).
    + Question Title (optional): tiêu đề câu hỏi.
    + Short Title (optional): mô tả ngắn hoặc nhãn gọn dùng cho nội bộ.
    + Question Content: nội dung chính của câu hỏi.
    + Điểm số: số điểm được chấm nếu trả lời đúng.
    + Khu vực nội dung câu hỏi: câu hỏi sẽ được hiển thị tùy theo loại câu hỏi mà người dùng chỉnh sửa.
    + Nút Update Question: Nhấn để lưu các thay đổi vào cơ sở dữ liệu.
  - **Kỹ thuật triển khai:**
    + Câu hỏi được nhận dạng duy nhất qua question_id.
    + Khi chỉnh sửa, hệ thống kiểm tra hợp lệ tất cả các trường trước khi ghi đè dữ liệu.
    + Trạng thái và dữ liệu mới được cập nhật ngay trong bài test liên kết thông qua passage_id.

    Giao diện chỉnh sửa câu hỏi được trình bài tại Hình 4.10 (trang sau)



Hình 4.10 Giao diện chỉnh sửa câu hỏi

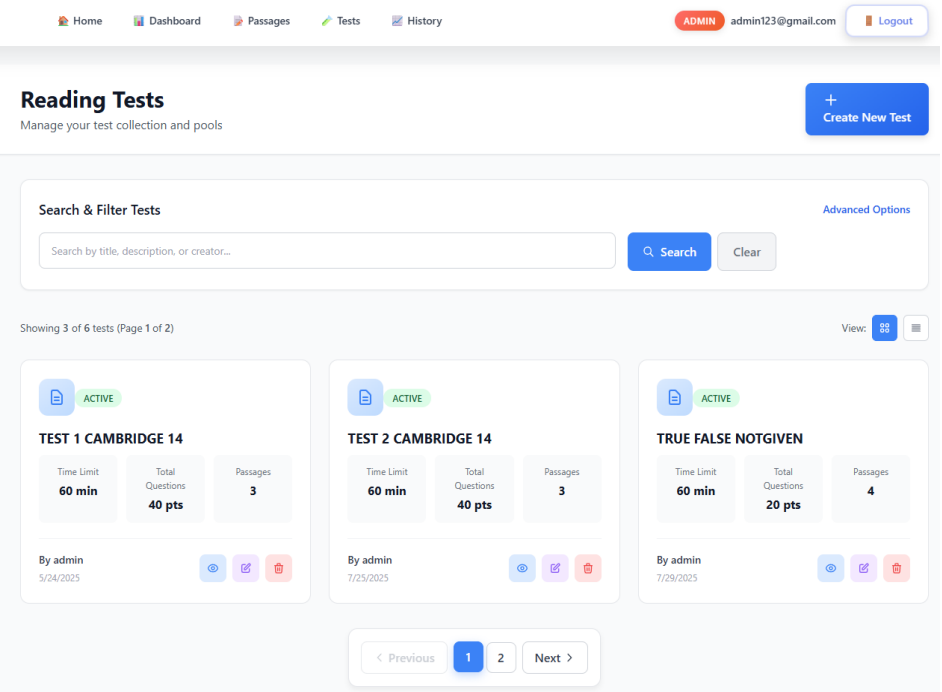
❖ **Giao diện quản lý đề kiểm tra – Reading Tests**

Đây là nơi quản trị viên có thể quản lý toàn bộ bộ sưu tập đề kiểm tra kỹ năng đọc hiểu (Reading), bao gồm tạo mới, tìm kiếm, chỉnh sửa và theo dõi trạng thái từng đề thi.

- **Chức năng chính:**
  - **Tìm kiếm & lọc đề kiểm tra**
    - **Search bar**: Cho phép tìm kiếm theo tiêu đề, mô tả, hoặc người tạo đề.
    - **Nút "Advanced Options"**: Mở rộng bộ lọc để tìm kiếm nâng cao (giả định trong chức năng ẩn hoặc chưa hiển thị).
    - **Nút Search & Clear**: Dùng để thực hiện tìm hoặc đặt lại bộ lọc.
  - **Danh sách đề kiểm tra hiện có**: Mỗi đề kiểm tra hiển thị dưới dạng thẻ đề (test card) với các thông tin:
    - **Tên bài kiểm tra:** (ví dụ: TEST 1 CAMBRIDGE 14).
    - **Trạng thái**: "ACTIVE" (hiện đang được sử dụng).
    - **Thời gian làm bài**: mặc định là 60 phút.

+ **Tổng số câu hỏi**: hiển thị tổng số câu hỏi có trong bài test (Ví dụ 40 questions).
+ **Số đoạn văn**: hiển thị tổng số bài đọc có trong bài test (Ví dụ 3 passages.)
+ **Ngày tạo và người tạo**: hiển thị rõ ràng bên dưới.

- **Các thao tác trên từng đề**
  + Biểu tượng nút thao tác:
  + Preview đề kiểm tra.
  + Edit đề kiểm tra.
  + Delete đề (có thể có xác nhận trước khi xoá).
  + View passage list liên kết với đề đó.
- **Tạo đề mới**: Nút "+ Create New Test" ở góc phải trên cùng cho phép tạo đề kiểm tra mới từ đầu.
- **Phân trang**: Giao diện hỗ trợ chia trang nếu có nhiều đề kiểm tra (ví dụ: hiển thị Page 1 of 2). Có thể điều hướng giữa các trang bằng nút Previous / Next.
- **Kỹ thuật triển khai:**
  + Mỗi đề kiểm tra liên kết với nhiều đoạn văn (passages) và câu hỏi.
  + Dữ liệu được phân loại theo trạng thái (ACTIVE / INACTIVE).
  + Các thao tác như sửa, xoá hoặc thêm mới được xác thực quyền admin trước khi thực hiện.



Hình 4.11 Giao diện quản lý bài kiểm tra

❖ **Giao diện tạo bài kiểm tra – Create Reading Test**

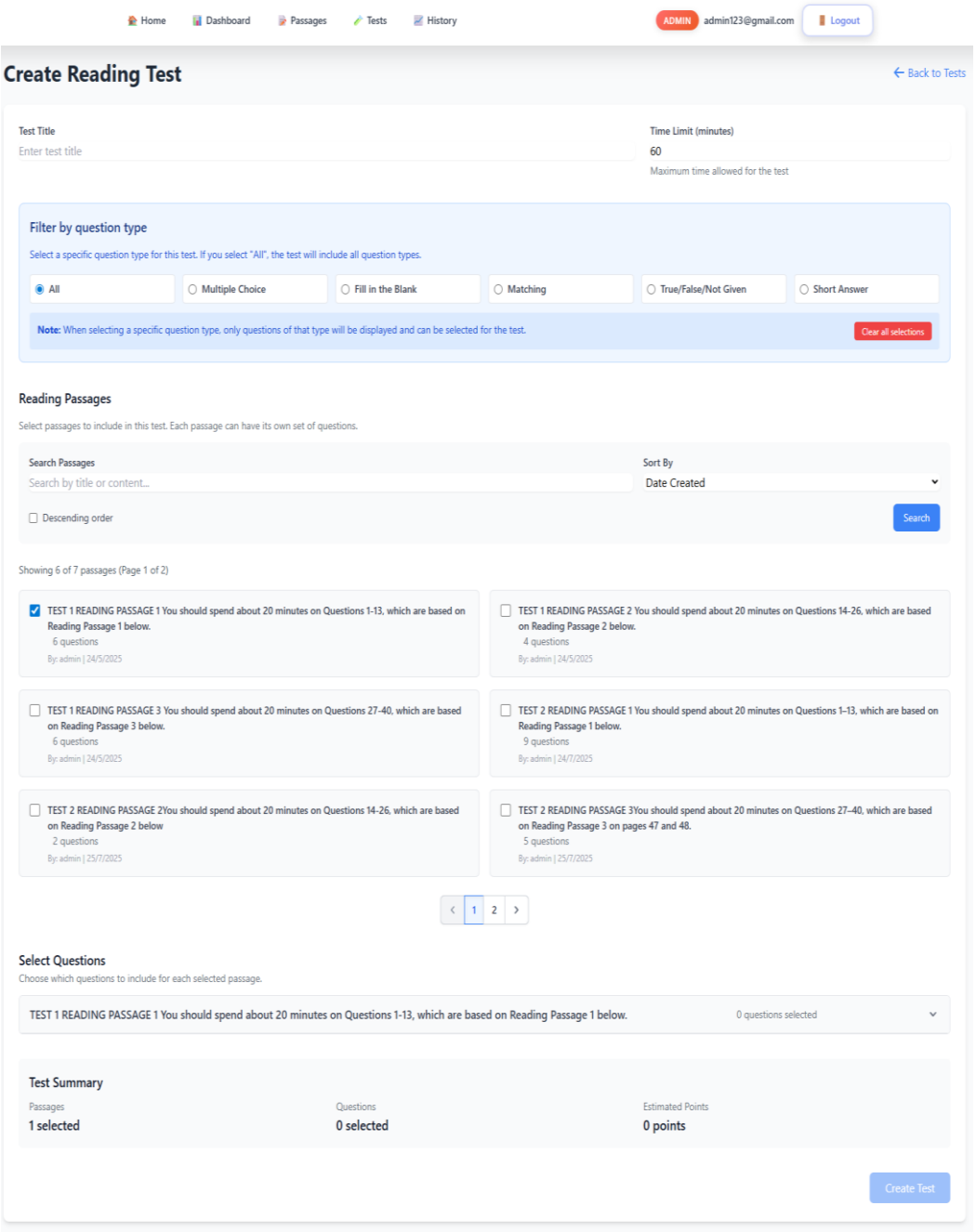
Đây là nơi quản trị viên tạo bài kiểm tra kỹ năng đọc hiểu (Reading) bằng cách chọn tiêu đề, loại câu hỏi và đoạn văn phù hợp để xây dựng cấu trúc đề kiểm tra.

- **Chức năng chính:**
  - **Nhập tiêu đề và thời gian làm bài:**
    + **Trường "Test Title"**: Nhập tên đề kiểm tra.
    + **Time Limit (minutes)**: Mặc định là 60 phút thời gian làm bài tối đa cho đề.
  - **Bộ lọc theo loại câu hỏi:** Lựa chọn loại câu hỏi muốn đưa vào đề:
    + All (Tất cả)
    + Multiple Choice
    + Fill in the Blank
    + Matching
    + True/False/Not Given
    + Short Answer
    + Nút "Clear all selections": Xóa toàn bộ lựa chọn đã chọn.
  - **Danh sách đoạn văn – Reading Passages:**
    + **Thanh tìm kiếm (Search Passages)**: Tìm theo tiêu đề hoặc nội dung.
    + **Tuỳ chọn sắp xếp (Sort By)**: Theo ngày tạo (Date Created).
    + **Tuỳ chọn hiển thị**: Sắp xếp theo thứ tự giảm dần (Descending order).
    + **Danh sách đoạn văn hiển thị dạng thẻ**, mỗi thẻ bao gồm:
      ▪ Tên đoạn văn (ví dụ: TEST 1 READING PASSAGE 1…)
      ▪ Mô tả về đoạn văn và phạm vi câu hỏi liên quan.
      ▪ Số lượng câu hỏi (ví dụ: 4 questions).
      ▪ Người tạo và ngày tạo (By admin | 24/6/2025).
  - **Phân trang:**
    + Giao diện hỗ trợ chia trang nếu có nhiều đoạn văn.
    + Hiển thị số trang và điều hướng bằng Previous / Next hoặc số trang.
  - **Chọn câu hỏi – Select Questions:** Sau khi chọn đoạn văn, quản trị viên có thể **chọn câu hỏi cụ thể** từ đoạn đó để hêm vào đề kiểm tra.
  - **Tổng quan đề kiểm tra – Test Summary:**
    + **Passages**: Số đoạn đã chọn.
    + **Questions**: Số câu đã chọn.

+ **Estimated Points**: Tổng điểm.

- **Tạo đề – Create Test:** Nút **"Create Test"** màu xanh dương ở cuối trang dùng để hoàn tất việc tạo đề kiểm tra sau khi đã chọn đầy đủ thông tin.

- **Kỹ thuật triển khai:**
  - Hệ thống chỉ hiển thị các câu hỏi tương ứng với loại đã chọn trong bộ lọc.
  - Mỗi đề kiểm tra có thể chứa nhiều đoạn văn và nhiều loại câu hỏi.
  - Các thao tác như chọn câu hỏi, thêm đoạn văn và tạo đề kiểm tra yêu cầu quyền admin.



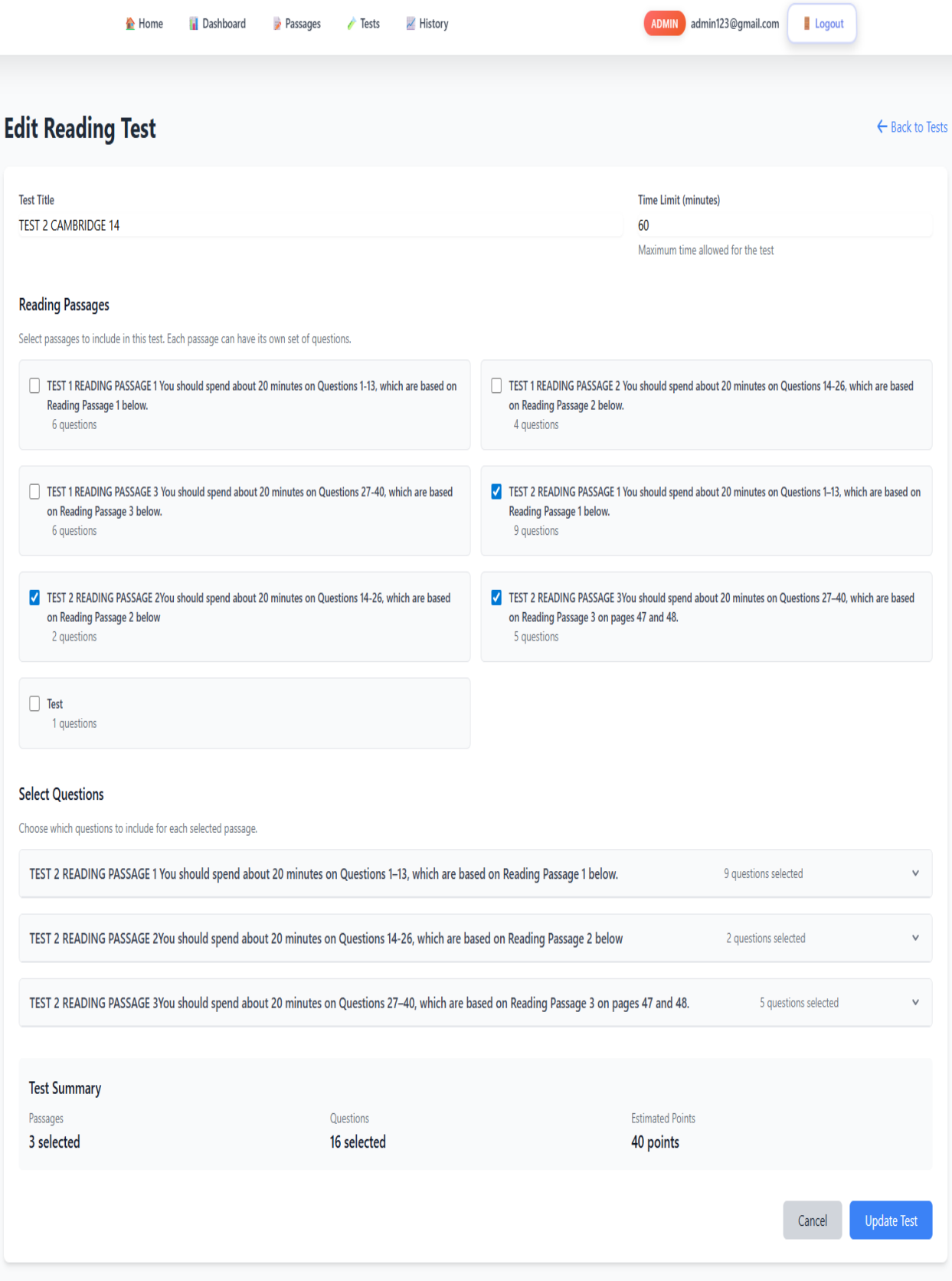
Hình 4.12 Giao diện tạo bài kiểm tra

❖ Giao diện chỉnh sửa bài kiểm tra – Edit Reading Test

Đây là nơi quản trị viên chỉnh sửa nội dung đề kiểm tra kỹ năng đọc hiểu đã tạo trước đó, bao gồm cập nhật tiêu đề, chỉnh sửa đoạn văn, lựa chọn câu hỏi và cập nhật đề kiểm tra.

- **Chức năng chính:**
  - Thông tin chung về đề kiểm tra:
    + **Test Title**: Hiển thị tiêu đề đề kiểm tra đã tạo ví dụ: TEST 2 CAMBRIDGE 14.
    + **Time Limit (minutes)**: Mặc định là 60 phút.
  - **Danh sách đoạn văn – Reading Passages:** Giao diện hiển thị các đoạn văn đã và chưa được thêm vào đề.
    + Mỗi thẻ đoạn văn hiển thị:
      - Tên đoạn văn (VD: *TEST 2 READING PASSAGE 1*).
      - Thông tin thời gian làm bài & phạm vi câu hỏi.
      - Số lượng câu hỏi (VD: *9 questions*).
      - Có thể chọn các đoạn văn để bao gồm trong đề kiểm tra.
  - Chọn câu hỏi – Select Questions: Cho phép chọn câu hỏi cho từng đoạn văn đã chọn. Mỗi đoạn có thể mở rộng (dropdown) để chọn cụ thể các câu hỏi. Ví dụ:
    - PASSAGE 1 có 9 questions selected**.**
    - PASSAGE 2 có 2 questions selected**.**
    - PASSAGE 3 có 5 questions selected**.**
- **Tổng quan đề kiểm tra – Test Summary:**
  - **Passages**: Tổng số đoạn đã chọn (VD: *3 selected*).
  - **Questions**: Tổng số câu hỏi đã chọn (VD: *16 selected*).
  - **Estimated Points**: Tổng điểm ước tính của bài kiểm tra (VD: *40 points*).
- **Các nút chức năng:**
  - **Update Test**: Nút màu xanh dương dùng để lưu các thay đổi đã thực hiện.
  - **Cancel**: Hủy thao tác chỉnh sửa và quay lại mà không lưu thay đổi.
- **Kỹ thuật triển khai:**
  - Hệ thống cho phép chọn hoặc bỏ chọn đoạn văn và câu hỏi tương ứng một cách linh hoạt.
  - Việc chỉnh sửa bài kiểm tra chỉ được thực hiện khi người dùng có quyền admin**.**

- Câu hỏi được đồng bộ theo đoạn văn được chọn, giúp đảm bảo tính nhất quán của bài test.

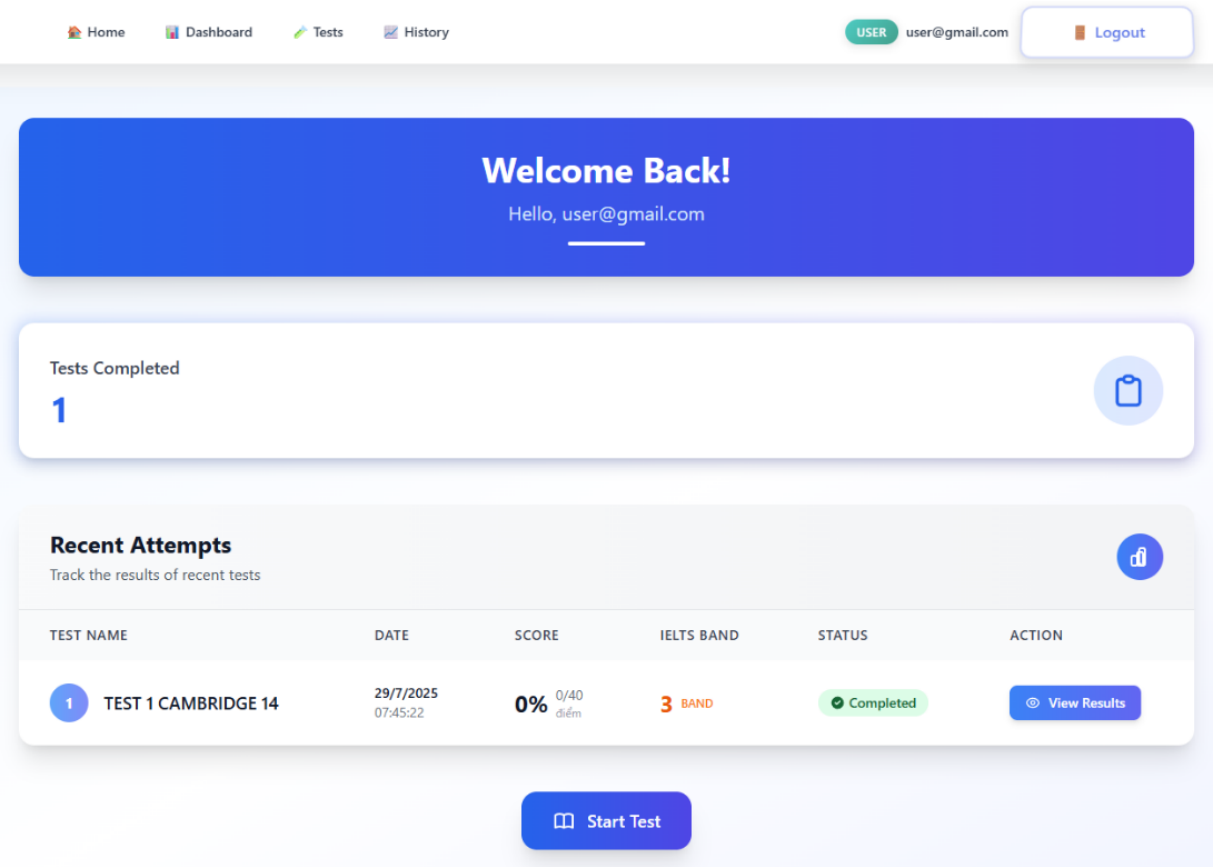


Hình 4.13 Giao diện chỉnh sửa bài kiểm tra

❖ **Giao diện tổng quan người dùng – User Dashboard**

Đây là giao diện hiển thị sau khi người học đăng nhập vào hệ thống. Trang cung cấp cái nhìn tổng quan về số lượng bài kiểm tra đã hoàn thành, các lần làm bài gần đây và cho phép bắt đầu bài kiểm tra mới.

- **Chức năng chính:**
  - **Chào mừng người dùng:**
    + Phần đầu trang hiển thị lời chào "Welcome Back!" kèm theo địa chỉ email của người dùng (VD: user@gmail.com).
    + Nổi bật bằng khung màu xanh đậm tạo cảm giác thân thiện và cá nhân hóa trải nghiệm.
  - **Thống kê tổng quan:**
    + **Tests Completed**: Hiển thị số lượng bài kiểm tra đã hoàn thành (**1** bài).
    + Giao diện trình bày bằng thẻ nổi bật với biểu tượng clipboard.
  - **Lịch sử làm bài – Recent Attempts:** Hiển thị danh sách các bài kiểm tra người dùng đã thực hiện gần đây. Mỗi hàng gồm các thông tin:
    + **Test Name**: Tên đề kiểm tra (VD: TEST 1 CAMBRIDGE 14).
    + **Date**: Ngày giờ làm bài (VD: 29/7/2025 07:45:22).
    + **Score**: Điểm số đạt được (VD: 0% – 0/40 điểm).
    + **IELTS Band**: Mức điểm quy đổi theo thang IELTS (VD: 3 BAND).
    + **Status**: Trạng thái hoàn thành (VD: Completed).
    + **Action**: Nút **View Results** để xem lại kết quả chi tiết.
- **Bắt đầu làm bài – Start Test:** Nút "Start Test" màu xanh dương đậm nằm ngay dưới phần thống kê. Thiết kế nổi bật, dễ nhận biết và kích hoạt nhanh chóng việc bắt đầu bài kiểm tra mới.
- **Kỹ thuật triển khai:**
  - Giao diện phân quyền rõ ràng: hiển thị vai trò là user (khác với admin).
  - Dữ liệu được lấy động từ hệ thống, cập nhật theo từng lần người dùng hoàn thành bài kiểm tra.
  - Mỗi bài kiểm tra đi kèm trạng thái & kết quả cụ thể, giúp người học theo dõi tiến trình luyện tập.



Hình 4.14 Giao diện dashboard của người dùng

❖ **Giao diện danh sách bài kiểm tra – Test List**

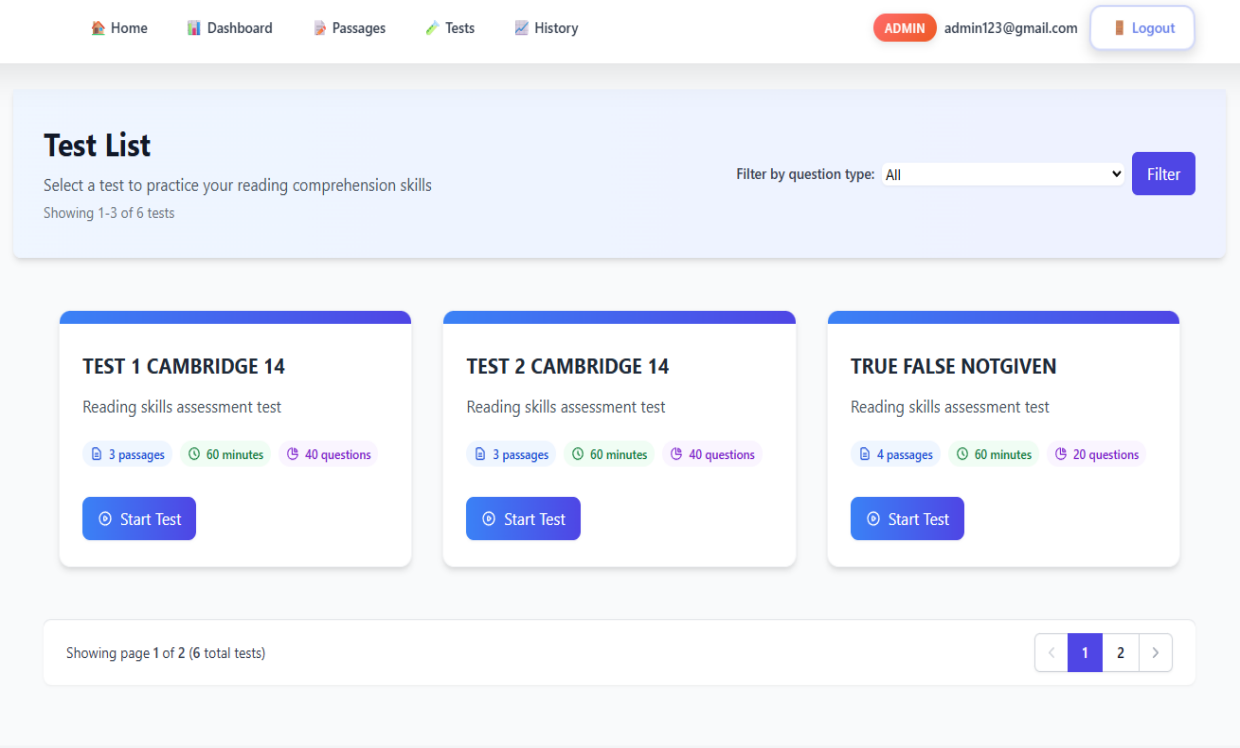
Đây là nơi người dùng (hoặc quản trị viên) có thể xem toàn bộ các bài kiểm tra kỹ năng đọc hiểu hiện có, lọc theo loại câu hỏi và bắt đầu làm bài kiểm tra mong muốn.

- **Chức năng chính:**
  - **Bộ lọc bài kiểm tra:**
    - + **Tiêu đề chính**: "Test List" kèm mô tả: Select a test to practice your reading comprehension skills.
    - + **Dropdown "Filter by question type"**: Cho phép lọc danh sách bài kiểm tra theo loại câu hỏi (VD: All, Multiple Choice, True/False/Not Given, v.v.).
    - + **Nút "Filter"**: Thực hiện lọc danh sách bài kiểm tra theo lựa chọn đã chọn.
  - **Danh sách các bài kiểm tra:** Mỗi bài kiểm tra được hiển thị dưới dạng thẻ test (test card), gồm các thông tin sau:
    - + **Tên bài kiểm tra**: VD: TEST 1 CAMBRIDGE 14,
    - + **Mô tả**: Reading skills assessment test.
    - + **Thông tin chi tiết** (dưới dạng biểu tượng):
      - ▪ Số đoạn văn (**passages**): VD: 3 passages / 4 passages.

- Thời gian làm bài: Mặc định là 60 minutes.
- Tổng số câu hỏi (**questions**): VD: 40 questions, 20 questions.

+ **Nút "Start Test"**: Cho phép bắt đầu bài kiểm tra ngay lập tức.

- **Phân trang:**
  + **Hiển thị tổng số bài kiểm tra:** Showing page 1 of 2 (6 total tests).
  + **Điều hướng phân trang**: Các nút <, 1, 2, > cho phép chuyển qua các trang tiếp theo của danh sách bài kiểm tra.
- **Kỹ thuật triển khai:**
  + Danh sách bài kiểm tra được hiển thị động dựa vào dữ liệu trong hệ thống.
  + Hệ thống hỗ trợ lọc bài kiểm tra theo loại câu hỏi để phục vụ nhu cầu luyện tập cụ thể.
  + Các thẻ bài kiểm tra được trình bày rõ ràng và nhất quán, giúp người dùng dễ dàng lựa chọn.



Hình 4.15 Giao diện danh sách bài kiểm tra của người dùng

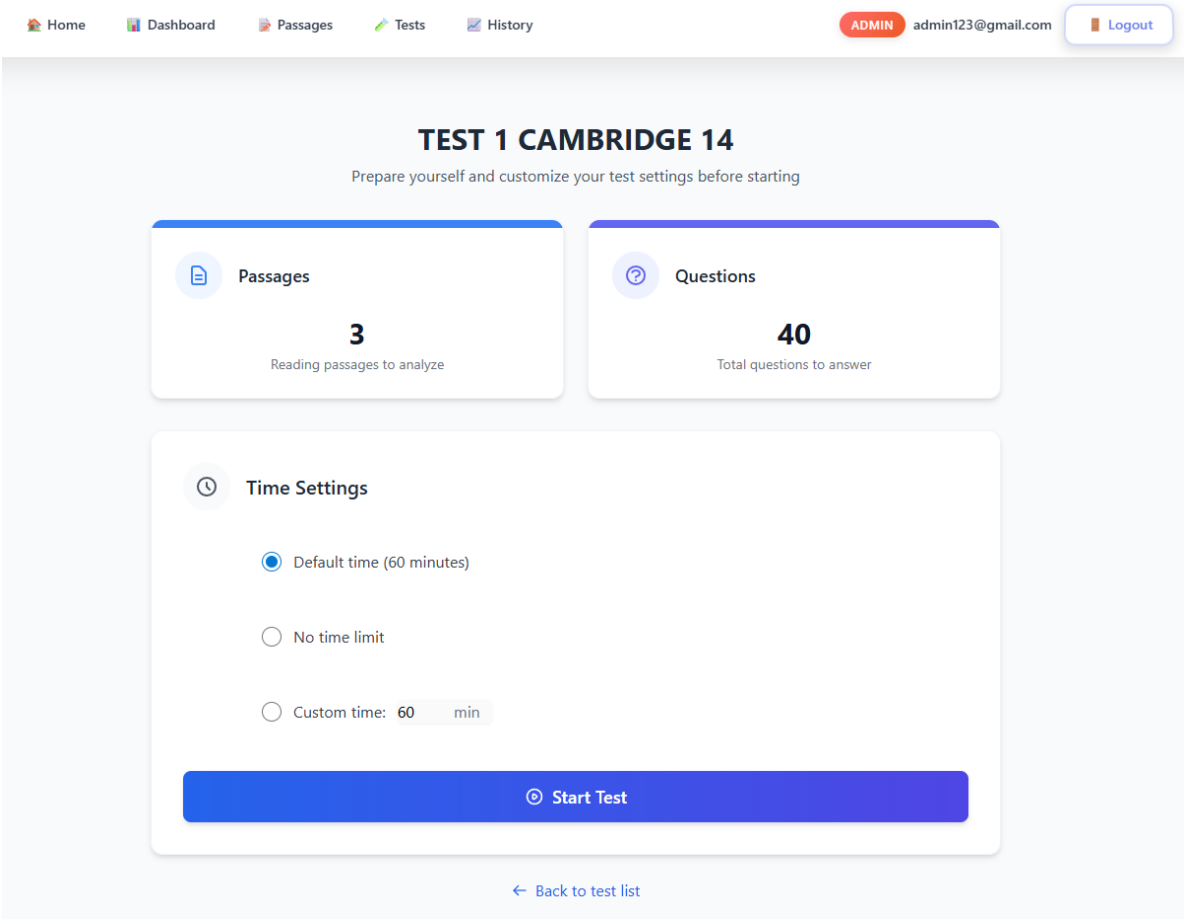
❖ **Giao diện tùy chỉnh cấu hình bài kiểm tra – Test Settings**

Trang này cho phép người dùng xem lại thông tin bài kiểm tra và thiết lập thời gian làm bài trước khi bắt đầu.

- **Chức năng chính:**
  - **Tiêu đề trang:** Hiển thị tên của bài kiểm tra

- **Thông tin bài kiểm tra:**
  + **Passages**: Hiển thị tổng số đoạn văn.
  + **Questions**: Hiển thị tổng số câu hỏi.
- **Cài đặt thời gian làm bài (Time Settings):** Người dùng có thể chọn 1 trong 3 tùy chọn:
  + **Default time (60 minutes)** – mặc định được chọn sẵn.
  + **No time limit** – không giới hạn thời gian.
  + **Custom time** – cho phép nhập thời gian tùy ý (đơn vị: phút).
- **Nút hành động: Start Test** Bắt đầu bài kiểm tra ngay với các thiết lập đã chọn.
- **Điều hướng: ← Back to test list** quay về trang danh sách bài kiểm tra.

- Kỹ thuật triển khai:
  - Toàn bộ lựa chọn thời gian đều được kiểm soát bằng radio button, giúp dễ chọn và đảm bảo chỉ có 1 thiết lập được áp dụng.
  - Cho phép người dùng điều chỉnh linh hoạt thời gian để luyện tập phù hợp với trình độ hoặc nhu cầu.

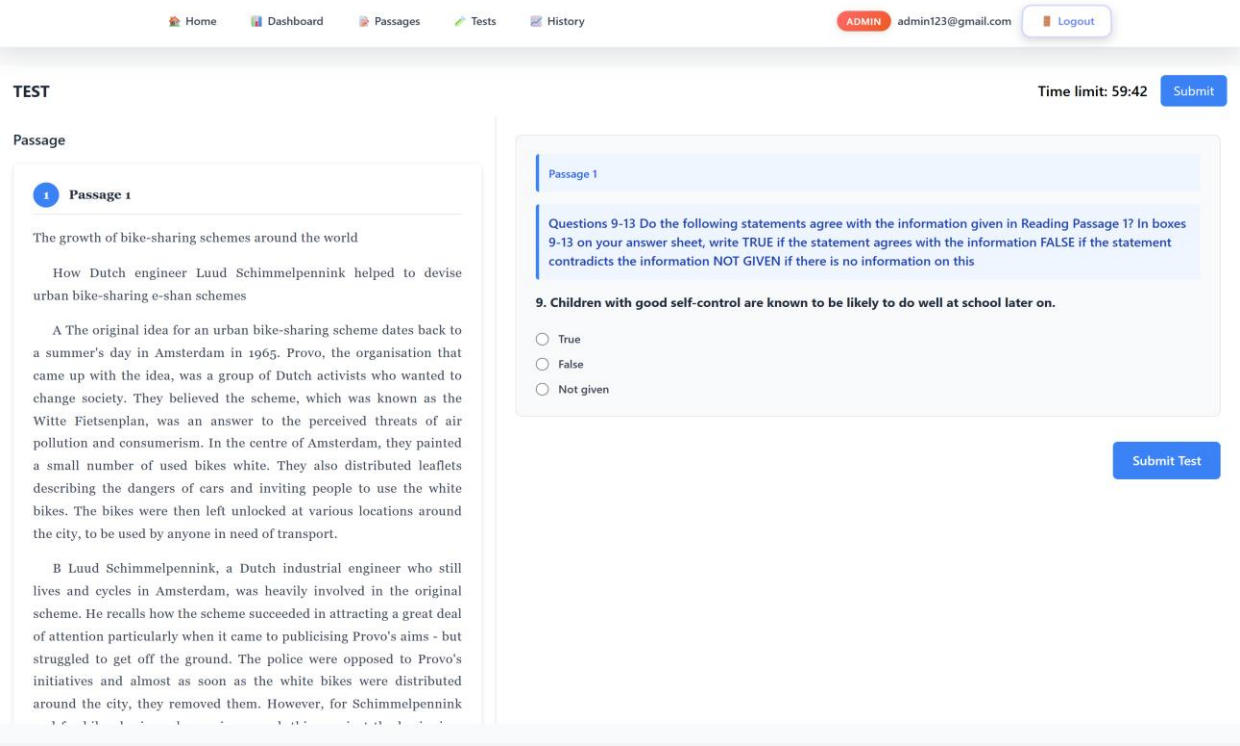


Hình 4.16 Giao diện cài đặt bài kiểm tra

❖ Giao diện làm bài kiểm tra – Test in Progress

Đây là màn hình chính khi người dùng đang làm bài kiểm tra kỹ năng đọc hiểu (IELTS Reading).

- **Chức năng chính:**
  - **Phần trên cùng:**
    - + Hiển thị tên của bài test.
    - + **Time limit**: Bộ đếm ngược thời gian làm bài (hiển thị ở góc trên bên phải, ví dụ: 59:45).
    - + **Nút Submit**: Dùng để nộp toàn bộ bài test (nút nhỏ nằm cạnh bộ đếm thời gian).
- **Giao diện nội dung bài đọc:**
  - **Vùng trái nội dung bài đọc (Reading Passage):**
    - + Thứ tự bài đọc với tiêu đề lớn và rõ ràng.
    - + Nội dung bài đọc được hiển thị đầy đủ, dạng cuộn.
    - + Các đoạn văn được chia rõ ràng, có cấu trúc tốt giúp người đọc dễ theo dõi.
  - **Vùng phải Câu hỏi và lựa chọn:** Hiển thị thông tin của câu hỏi và nơi để người dùng làm bài kiểm tra
- **Nút Submit Test** : cho phép người dùng nộp toàn bộ bài khi đã hoàn thành khi nhấn nút sẽ hiển thị lên cảnh báo để hỏi người dùng có chắc chắn nộp bài không.



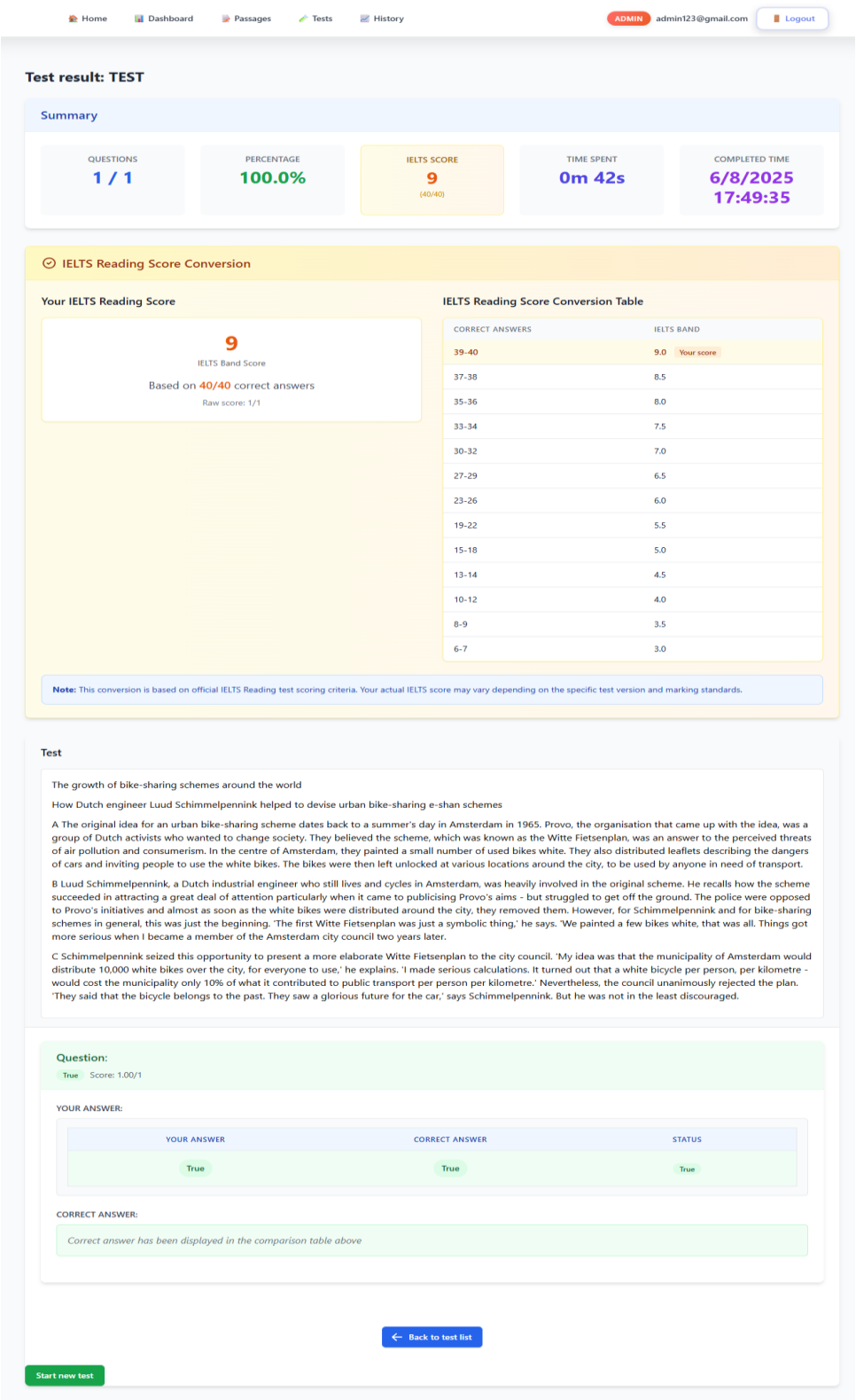
Hình 4.17 Giao diện chính thực hiện bài kiểm tra

❖ **Giao diện kết quả làm bài kiểm tra – Test Result**

Đây là màn hình hiển thị kết quả sau khi người dùng hoàn thành bài kiểm tra kỹ năng đọc hiểu (IELTS Reading).

- **Chức năng chính:**
  - **Phần tóm tắt kết quả (Summary):**
    + **Questions:** Tổng số câu hỏi và số câu trả lời đúng (ví dụ: 1/1).
    + **Percentage:** Tỉ lệ trả lời đúng (%) (ví dụ: 100.0%).
    + **IELTS Score:** Điểm quy đổi IELTS Reading (ví dụ: 9/40).
    + **Time Spent:** Thời gian làm bài (ví dụ: 0m 42s).
    + **Completed Time:** Ngày giờ hoàn thành bài (ví dụ: 6/8/2025 17:49:35).
  - **Phần quy đổi điểm IELTS Reading (IELTS Reading Score Conversion):**
    + Hiển thị Band Score của người dùng dựa trên số câu trả lời đúng và bảng quy đổi chuẩn của IELTS.
    + Bảng quy đổi điểm: Liệt kê số câu đúng tương ứng với band score từ 3.0 đến 9.0.
    + Ghi chú: Điểm quy đổi dựa trên tiêu chí chấm điểm IELTS thực tế, có thể thay đổi tùy theo dạng đề và tiêu chuẩn chấm.
- **Phần nội dung bài test:** Hiển thị bài đọc đã sử dụng trong bài kiểm tra.
  - Câu hỏi và thông tin chi tiết câu trả lời:
    + Hiển thị câu hỏi, số điểm đạt được (ví dụ: Score 1.00/1).
    + Bảng so sánh câu trả lời của người dùng và đáp án đúng.
    + Trạng thái câu trả lời (ví dụ: True – đúng).
- **Nút điều hướng:**
  - **Back to test list:** Quay lại danh sách bài kiểm tra.
  - **Start new test:** Bắt đầu một bài kiểm tra mới.

  Giao diện kết quả bài kiểm tra được trình bài tại Hình 4.18 (trang sau)

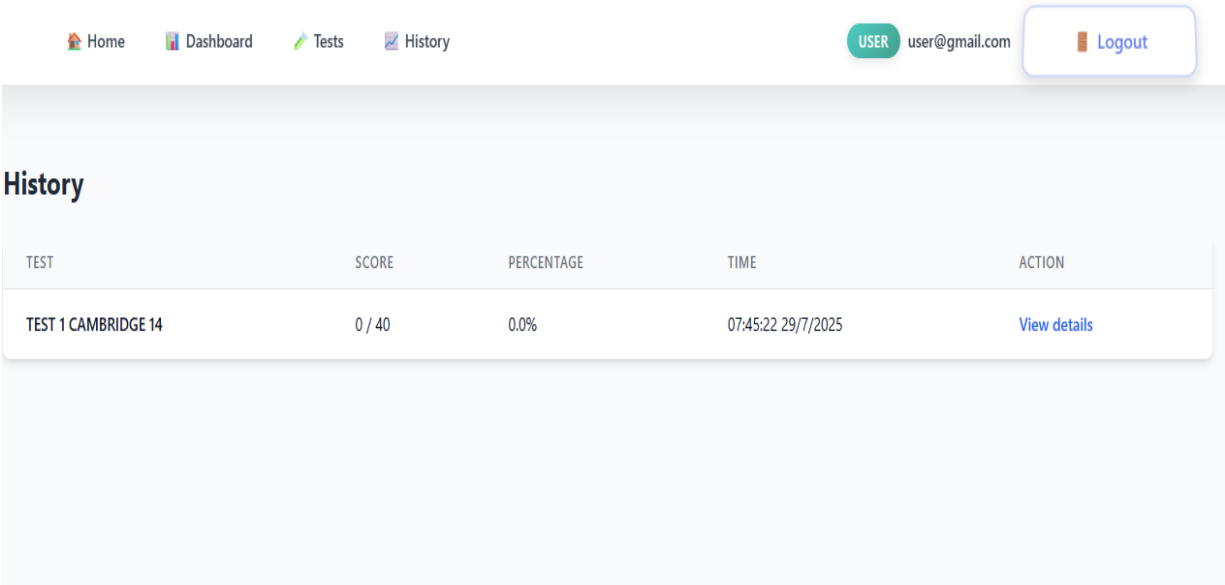


Hình 4.18 Giao diện kết quả bài kiểm tra

❖ **Giao diện lịch sử làm bài kiểm tra – History**

Đây là màn hình hiển thị danh sách các bài kiểm tra đã được người dùng thực hiện, kèm theo thông tin kết quả chi tiết và chức năng này chỉ áp dụng cho người dùng đăng nhập.

- **Chức năng chính:**
  - **Bảng lịch sử bài kiểm tra:**
    + **Test:** Tên bài kiểm tra đã làm (ví dụ: TEST 1 CAMBRIDGE 14).
    + **Score:** Số câu trả lời đúng trên tổng số câu (ví dụ: 0/40).
    + **Percentage:** Tỉ lệ phần trăm câu trả lời đúng (ví dụ: 0.0%).
    + **Time:** Thời điểm hoàn thành bài kiểm tra (ví dụ: 07:45:22 29/7/2025).
    + **Action:** Nút View details để xem chi tiết kết quả của bài kiểm tra đó.



Hình 4.19 Giao diện lịch sửa bài kiểm tra

4.2.4. Tích hợp JavaScript phía Frontend

JavaScript phía client được sử dụng để tăng cường tính tương tác và thực hiện các tác vụ động.

4.2.4.1. Chức năng Tự động Lưu (Auto-save Functionality)

Lớp TestManager (trong public/js/test-taking.js) được triển khai để tự động lưu câu trả lời của người dùng. Nó lắng nghe sự kiện input trên các ô trả lời, sau đó sử dụng fetch API để gửi câu trả lời đến endpoint /api/attempts/:attemptId/answers một cách không đồng bộ. Cơ chế debounce cũng được áp dụng để tối ưu số lần gọi API.

4.2.4.2. Triển khai Đồng hồ Đếm ngược (Timer Implementation)

Lớp TestTimer (trong public/js/timer.js) quản lý việc đếm ngược thời gian làm bài. Nó cập nhật hiển thị thời gian mỗi giây và có thể kích hoạt một hàm callback (ví dụ: tự động nộp bài) khi hết giờ. Đồng hồ cũng có cảnh báo khi thời gian sắp hết.

4.3. Kiểm thử hệ thống

Quá trình kiểm thử hệ thống "Reading Practice System" được thực hiện kỹ lưỡng, tập trung vào việc xác minh tính đúng đắn của dữ liệu đầu vào, bảo mật xác thực, xử lý lỗi và tính toàn vẹn dữ liệu. Các bằng chứng kiểm thử được trích xuất trực tiếp từ mã nguồn và quá trình phân tích hệ thống.

4.3.1. Phương pháp kiểm thử được áp dụng dựa trên phân tích mã nguồn

Các phương pháp kiểm thử chính sau đã được triển khai và xác minh thông qua phân tích mã nguồn thực tế của hệ thống:

4.3.1.1. Kiểm thử Xác thực Dữ liệu Đầu vào (Input Validation Testing)

Hệ thống đã triển khai các cơ chế xác thực dữ liệu ở nhiều tầng:

- **Xác thực ở tầng Model (Mongoose Schema):** Các trường dữ liệu trong UserSchema, QuestionSchema đều có các quy tắc ràng buộc như required, unique, minlength, match (cho email), enum (cho loại câu hỏi), và các hàm validate tùy chỉnh.
- **Xác thực phía Frontend (JavaScript):** Ví dụ, script trong public/js/admin /test/create.js kiểm tra việc chọn đủ bài đọc và câu hỏi khi tạo bài kiểm tra.

4.3.1.2. Kiểm thử Xác thực & Bảo mật (Authentication & Security Testing)

- **Bảo mật Mật khẩu:** Mật khẩu được băm bằng **bcrypt** (salt rounds = 10) trước khi lưu (UserSchema.pre('save')).
- **Quản lý Phiên làm việc và Phân quyền:** Middleware isAuthenticated và isAdmin (middleware/auth.js) kiểm tra session và vai trò người dùng để kiểm soát truy cập. Cấu hình session sử dụng secure cookies và có thời gian hết hạn.

4.3.1.3. Kiểm thử Xử lý Lỗi (Error Handling Testing)

- **Backend:** Khối try...catch được sử dụng trong controllers để xử lý lỗi và trả về thông báo phù hợp.
- **Frontend:** Yêu cầu AJAX (fetch) có kèm .catch() để xử lý lỗi phía client (public/js/user/test-attempt.js).

4.3.1.4. Kiểm thử Tính Toàn vẹn Dữ liệu (Data Integrity Testing)

- Các quy tắc nghiệp vụ phức tạp được định nghĩa trong QuestionSchema (ví dụ: pre('save') hook và validator tùy chỉnh) đảm bảo dữ liệu câu hỏi (loại, tùy chọn, đáp án) luôn hợp lệ và nhất quán.

4.3.2. Kết quả Kiểm thử Chức năng (Dựa trên bằng chứng mã nguồn)

Việc triển khai các logic kiểm tra và xác thực trong mã nguồn cho thấy các chức năng chính đã được xem xét về mặt đảm bảo chất lượng:

- **Hệ thống Xác thực:** Các quy tắc validate đầu vào, mã hóa mật khẩu, so sánh mật khẩu, quản lý session và phân quyền đều được triển khai trong mã nguồn.
- **Xác thực Dữ liệu Đầu vào:** Các quy tắc cho định dạng email, độ dài mật khẩu/username, trường bắt buộc, và logic phức tạp cho tùy chọn câu hỏi đều có trong schema.
- **Xử lý Lỗi:** Các cơ chế bắt lỗi ở cả backend (try-catch) và frontend (AJAX catch) đã được tích hợp.

4.3.3. Kiểm thử Hiệu suất (Performance Testing) - Khía cạnh Tối ưu hóa từ Mã nguồn

Mã nguồn thể hiện các nỗ lực tối ưu hóa hiệu suất ban đầu:

- **Tối ưu hóa Truy vấn Cơ sở dữ liệu:** Sử dụng phân trang (.skip(), .limit()) và .populate() để giảm số lượt và khối lượng truy vấn (controllers/userController.js).
- **Tối ưu hóa Hiệu suất Frontend:** Chức năng tự động lưu câu trả lời (public/js/user/test-attempt.js) sử dụng localStorage để lưu trữ tạm thời trước khi đồng bộ lên server, giảm độ trễ cảm nhận của người dùng và tần suất gọi API.

4.3.4. Kiểm thử Bảo mật được triển khai

Các biện pháp bảo mật cơ bản đã được tích hợp:

- **Bảo mật Xác thực:** Mã hóa mật khẩu bcrypt (salt rounds = 10), độ dài tối thiểu mật khẩu. Cấu hình session an toàn (secure cookies cho production, thời gian hết hạn).
- **Làm sạch Dữ liệu Đầu vào (Input Sanitization):** Mongoose schema có các tùy chọn trim, lowercase. EJS mặc định escape HTML output (<%= ... %>).

4.4. So sánh với các hệ thống tương tự

Phần này phân tích vị thế của Hệ thống Reading Practice System so với các giải pháp cạnh tranh trên thị trường, từ đó nhận diện điểm mạnh, điểm yếu và định hướng phát triển.

4.4.1. Phân tích hệ thống cạnh tranh

Hệ thống được so sánh với các nền tảng giáo dục và luyện tập phổ biến như Khan Academy, Duolingo, và Coursera/edX.

- **So với Khan Academy:** RPS có kiến trúc đơn giản hơn, dễ bảo trì cho quy mô nhỏ, nhưng thiếu các tính năng AI và sự đa dạng về loại câu hỏi.
- **So với Duolingo:** RPS tập trung vào học thuật hơn là game hóa, có toàn quyền quản lý nội dung nhưng trải nghiệm di động và yếu tố tạo động lực còn hạn chế.
- **So với Coursera/edX:** RPS chuyên biệt cho kỹ năng đọc, quy mô nhỏ hơn, không cấp chứng chỉ nhưng dễ tiếp cận và tập trung vào kỹ năng thực tế.

4.4.2. Định vị sản phẩm

- **Lợi thế Cạnh tranh chính:**
  - **Chuyên biệt hóa cao:** Tập trung vào kỹ năng đọc hiểu.
  - **Đơn giản, trực quan:** Giao diện dễ sử dụng.
  - **Linh hoạt nội dung:** Admin toàn quyền quản lý.
  - **Chi phí thấp (tiềm năng):** Công nghệ mã nguồn mở.
  - **Khả năng tùy chỉnh:** Dễ mở rộng tính năng.
- **Điểm yếu chính:**
  - Hạn chế tính năng nâng cao (gamification, social).
  - Thiếu AI hỗ trợ cá nhân hóa.
  - Trải nghiệm di động chỉ là responsive web.
  - Phân tích dữ liệu học tập còn cơ bản.
- **Vị trí thị trường:** "Đơn giản & Chuyên biệt", phù hợp cho người dùng cá nhân, trường học, trung tâm ngoại ngữ nhỏ cần công cụ luyện đọc hiểu tập trung.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Tổng kết kết quả đạt được

Qua quá trình thực hiện đề tài "Xây dựng hệ thống luyện thi trực tuyến tiếng Anh kỹ năng Reading với cơ sở dữ liệu NoSQL", các mục tiêu chính đề ra ban đầu đã được hoàn thành, cụ thể như sau:

❖ **Xây dựng thành công hệ thống luyện thi trực tuyến với các chức năng cốt lõi:**

- Hệ thống đã đáp ứng được nhu cầu luyện tập các dạng câu hỏi reading thường gặp trong các kỳ thi tiếng Anh như: trắc nghiệm nhiều lựa chọn, điền từ vào chỗ trống, matching headings, fill in blank, true/false/not given và short answer.
- **Chức năng quản trị (Admin Module):** Cho phép quản trị viên quản lý hiệu quả thư viện bài đọc (Reading Passages), ngân hàng câu hỏi (Questions) với 5 loại câu hỏi khác nhau, và tạo lập, cấu hình các bài kiểm tra (Tests). Giao diện dashboard quản trị cung cấp cái nhìn tổng quan về hoạt động hệ thống.
- **Chức năng người dùng/học viên (User Module):** Cung cấp trải nghiệm làm bài hoàn chỉnh từ việc lựa chọn bài kiểm tra từ danh sách, tùy chỉnh cài đặt thời gian làm bài (nếu có), thực hiện bài làm với giao diện tương tác, có đồng hồ đếm ngược và cơ chế lưu câu trả lời. Sau khi nộp bài, người dùng có thể xem kết quả chi tiết, phân tích câu trả lời và theo dõi lịch sử làm bài của mình (đối với người dùng đã đăng nhập).
- **Chức năng xác thực (Authentication Module):** Quản lý quy trình đăng ký, đăng nhập, đăng xuất và phân quyền người dùng (admin/user) một cách rõ ràng, đảm bảo an toàn và bảo mật cho hệ thống.

❖ **Ứng dụng thành công cơ sở dữ liệu NoSQL với hệ quản trị cơ sở dữ liệu MongoDB:**

- Hệ thống đã sử dụng MongoDB làm cơ sở dữ liệu chính, phát huy được tính linh hoạt của NoSQL trong việc lưu trữ đa dạng các loại câu hỏi và cấu trúc dữ liệu khác nhau của bài đọc, thông tin người dùng và kết quả làm bài.
- Việc thiết kế schema cho các collection (User, ReadingPassage, Question, Test, UserAttempt, TestPool) đã được thực hiện cẩn thận, đảm bảo tính toàn vẹn dữ liệu ở mức ứng dụng và tối ưu cho các truy vấn thường gặp.

❖ **Triển khai hệ thống trên nền tảng công nghệ hiện đại:**

- Backend được xây dựng bằng Node.js và framework Express.js theo kiến trúc MVC, giúp mã nguồn được tổ chức khoa học, dễ bảo trì và mở rộng.
- Frontend sử dụng EJS làm template engine cho việc render phía máy chủ, kết hợp với Tailwind CSS để xây dựng giao diện người dùng đáp ứng (responsive) và hiện đại.
- Các thư viện và công cụ hỗ trợ như Mongoose ODM, bcrypt, express-session, connect-mongo, multer, moment đã được tích hợp hiệu quả.

❖ **Đảm bảo chất lượng hệ thống thông qua kiểm thử:**

- Các phương pháp kiểm thử dựa trên phân tích mã nguồn đã được áp dụng, bao gồm kiểm thử xác thực dữ liệu đầu vào (cả ở tầng model và frontend), kiểm thử xác thực và bảo mật (mã hóa mật khẩu, quản lý session, phân quyền), kiểm thử xử lý lỗi, và kiểm thử tính toàn vẹn dữ liệu.
- Mặc dù đánh giá kiểm thử tự động và số liệu hiệu năng chi tiết còn là điểm cần cải thiện, các bằng chứng từ mã nguồn cho thấy hệ thống đã tích hợp nhiều biện pháp đảm bảo chất lượng, với đánh giá sơ bộ đạt 7.5/10.

Tóm lại, đề tài đã xây dựng thành công một hệ thống luyện thi trực tuyến kỹ năng Reading đáp ứng các mục tiêu nghiên cứu ban đầu, có khả năng hỗ trợ người học trong việc rèn luyện và tự đánh giá năng lực đọc hiểu tiếng Anh.

5.2. Hạn chế và đề xuất cải tiến trong tương lai

Bên cạnh những kết quả tích cực đã đạt được, hệ thống "Reading Practice System" vẫn còn một số hạn chế nhất định và nhiều tiềm năng để phát triển, hoàn thiện hơn trong tương lai.

5.2.1. Hạn chế của hệ thống hiện tại

- **Thiếu các tính năng nâng cao và tương tác:** Hệ thống hiện tại tập trung vào các chức năng cốt lõi mà chưa có các yếu tố "game hóa" (gamification) mạnh mẽ, các tính năng tương tác xã hội (social learning) để tăng sự hấp dẫn và động lực cho người dùng.
- **Chưa ứng dụng Trí tuệ Nhân tạo (AI):** Thiếu các tính năng học tập thích ứng (adaptive learning) dựa trên AI để cá nhân hóa lộ trình học, tự động điều chỉnh độ khó hay đưa ra gợi ý dựa trên năng lực người dùng.

- **Trải nghiệm di động còn hạn chế:** Hệ thống mới chỉ hỗ trợ giao diện web đáp ứng, chưa có ứng dụng di động native chuyên biệt, điều này có thể ảnh hưởng đến sự tiện lợi và trải nghiệm người dùng trên thiết bị di động.
- **Phân tích dữ liệu học tập còn cơ bản:** Khả năng phân tích và báo cáo chi tiết về quá trình học tập, điểm mạnh/yếu của người dùng còn ở mức cơ bản, chưa cung cấp được những insight sâu sắc.
- **Kiểm thử tự động và hiệu năng:** Việc thiếu một bộ kiểm thử tự động toàn diện (unit, integration, end-to-end) và các số liệu đo lường hiệu năng chính thức từ kiểm thử tải (load testing) là một hạn chế.
- **Khả năng mở rộng quy mô lớn:** Kiến trúc và công nghệ hiện tại phù hợp cho quy mô vừa và nhỏ. Việc mở rộng để phục vụ lượng người dùng rất lớn có thể đòi hỏi những thay đổi và tối ưu hóa đáng kể.

5.2.2. Đề xuất cải tiến và hướng phát triển trong tương lai

Dựa trên những hạn chế đã nêu và tiềm năng phát triển của hệ thống, các hướng cải tiến sau được đề xuất:

❖ **Cải tiến Ngắn hạn (1-3 tháng tới):**

- **Phát triển Progressive Web App (PWA):** Nâng cao trải nghiệm trên di động, cho phép "cài đặt" ứng dụng và có thể hỗ trợ một số tính năng offline cơ bản.
- **Tích hợp Yếu tố Game hóa Cơ bản:** Bổ sung điểm thưởng, huy hiệu, bảng xếp hạng đơn giản để tăng tính tương tác.
- **Nâng cấp Tính năng Phân tích Kết quả:** Cung cấp báo cáo chi tiết hơn cho người dùng và quản trị viên.
- **Tăng cường Hỗ trợ Ngoại tuyến cho PWA:** Cho phép cache bài kiểm tra để làm offline.
- **(Tùy chọn) Hỗ trợ Âm thanh cho Bài đọc:** Giúp luyện nghe hiểu hoặc hỗ trợ người dùng có khó khăn về thị giác.

❖ **Tầm nhìn Phát triển Dài hạn (6-12 tháng tới và xa hơn):**

- **Ứng dụng AI cho Gợi ý Cá nhân hóa và Học tập Thích ứng:** Phân tích hành vi học tập để đưa ra gợi ý phù hợp, tự động điều chỉnh độ khó.
- **Xây dựng các Tính năng Tương tác Xã hội:** Tạo nhóm học tập, diễn đàn thảo luận, so sánh kết quả.

- **Hỗ trợ Đa ngôn ngữ:** Mở rộng giao diện và nội dung cho nhiều ngôn ngữ.
- **Phát triển API để Tích hợp với Hệ thống LMS khác:** Giúp hệ thống hoạt động như một module chuyên biệt.
- **Phát triển Ứng dụng Di động Native (iOS/Android):** Mang lại trải nghiệm di động tốt nhất nếu có nhu cầu cao.

❖ **Định hướng Phát triển Công nghệ:**

- **Backend:** Cân nhắc nâng cấp lên NestJS, sử dụng Redis cho caching.
- **Frontend:** Chuyển sang Single Page Application (SPA) với React/Vue.js (sử dụng TypeScript).
- **AI/ML:** Nghiên cứu sử dụng TensorFlow.js hoặc thư viện Python (qua API).
- **Deployment:** Áp dụng container hóa (Docker), điều phối bằng Kubernetes, sử dụng CDN.
- **Kiểm thử:** Xây dựng bộ kiểm thử tự động toàn diện (unit, integration, end-to-end) và thực hiện kiểm thử hiệu năng định kỳ.

❖ **Kết luận cuối cùng:**

Đề tài "Xây dựng hệ thống luyện thi trực tuyến tiếng Anh kỹ năng Reading với cơ sở dữ liệu NoSQL" đã hoàn thành các mục tiêu chính, tạo ra một sản phẩm có giá trị ứng dụng thực tiễn. Hệ thống cung cấp một nền tảng cơ bản và ổn định cho việc luyện tập và đánh giá kỹ năng đọc hiểu tiếng Anh. Mặc dù còn những điểm cần cải thiện và phát triển thêm, nhưng với nền tảng hiện có và lộ trình phát triển rõ ràng, hệ thống có tiềm năng lớn để trở thành một công cụ học tập mạnh mẽ, đáp ứng tốt hơn nhu cầu của người học và đóng góp vào việc nâng cao năng lực ngoại ngữ trong cộng đồng. Việc tiếp tục nghiên cứu, áp dụng các công nghệ mới và lắng nghe phản hồi từ người dùng sẽ là yếu tố then chốt cho sự phát triển bền vững của hệ thống trong tương lai.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] K. Quỳnh, “IDP IELTS,” [Trực tuyến]. Available: https://ielts.idp.com/vietnam /prepare/article-ielts-reading-common-questions. [Đã truy cập 07 07 2023].

[2] Đ. Nguyễn Anh, "Smartcom," [Online]. Available:https://smartcom.vn/blog /thang-diem-ielts-reading/. [Accessed 2024 05 2024].

[3] Thanh, "Viblo," [Online]. Available: https://viblo.asia/p/mongodb-la-gi-co-so-du-lieu-phi-quan-he-bJzKmgoPl9N. [Accessed 09 03 2018].

[4] D. Mỹ, "It viec blog," [Online]. Available: https://itviec.com/blog/thiet-ke-co-so-du-lieu-nosql/. [Accessed 20 04 2024].

[5] T. V. Đoàn, "Viblo," [Online]. Available: https://viblo.asia/p/tim-hieu-ve-restful-api-OeVKBNYyKkW. [Accessed 17 09 2020].

[6] Q. Trung, "Viblo," [Online]. Available: https://viblo.asia/p/mot-cai-nhin-tong-quan-nhat-ve-nodejs-Ljy5VeJ3lra. [Accessed 23 04 2018].

[7] "CodeGym," [Online]. Available: https://codegym.vn/blog/expressjs-la-gi-huong-dan-cach-hoc-expressjs-chi-tiet/. [Accessed 21 01 2025].

[8] Pum, "200lab," [Online]. Available: https://200lab.io/blog/huong-dan-cai-dat-tailwind-css-co-ban. [Accessed 01 01 2025].

[9] "Amazon," [Online]. Available: https://aws.amazon.com/vi/nosql/.

[10] "OnlineExamMarker," [Online]. Available: https://onlineexammaker.com /kb/vi/8-automatic-grading-systems-that-will-save-your-time-in-teaching-and-training-15/. [Accessed 07 12 2024].

[11] V. Hào, Cambridge ielts 6, Hà Nội: Từ điển Bách khoa, Sách/ ĐH Trà Vinh, 2010.