BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

A blue and white logo

Description automatically generated

**BÀI TẬP LỚN**

**HỌC PHẦN: PHÁT TRIỂN DỰ ÁN DI ĐỘNG**

**ĐỀ TÀI: Xây dựng ứng dụng luyện thi bằng lái xe (Java DriverExam App)**

|  |  |  |
| --- | --- | --- |
| Mã Sinh Viên | Họ và Tên | Lớp |
| 2151170553 | Trần Trường Long | 63KTPM1 |
| 2151173798 | Nguyễn Đức Minh | 63KTPM1 |
| 2151173821 | Hoàng Đăng Tú Tài | 63KTPM1 |

### 

**Hà Nội, năm 2024**

BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

A blue and white logo

Description automatically generated

**BÀI TẬP LỚN**

**HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**ĐỀ TÀI: Xây dựng ứng dụng luyện thi bằng lái xe (Java DriverExam App)**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Sinh Viên | Họ và Tên | Điểm | |
| Bằng Số | Bằng Chữ |
| 2151170553 | Trần Trường Long |  |  |
| 2151173798 | Nguyễn Đức Minh |  |  |
| 2151173821 | Hoàng Đăng Tú Tài |  |  |

### 

### CÁN BỘ CHẤM THI 1 CÁN BỘ CHẤM THI 2

**Hà Nội, năm 2024**

**LỜI NÓI ĐẦU**

Trong bối cảnh giao thông ngày càng phát triển và trở nên phức tạp, việc sở hữu giấy phép lái xe không chỉ là một nhu cầu cần thiết mà còn là một trách nhiệm lớn lao đối với mỗi người lái. Nhằm hỗ trợ người dùng trong quá trình ôn tập và chuẩn bị cho kỳ thi cấp giấy phép lái xe hạng B1, B2 tại Việt Nam, chúng tôi đã phát triển ứng dụng "Thi Bằng Lái Xe".

Ứng dụng được thiết kế với giao diện thân thiện và dễ sử dụng, giúp người dùng dễ dàng tiếp cận và làm quen với các dạng câu hỏi trắc nghiệm. Bên cạnh việc cung cấp các bài thi thử, ứng dụng còn tích hợp nhiều tính năng hữu ích như lưu lịch sử thi, giải thích chi tiết cho mỗi câu hỏi, và phân tích các câu hỏi người dùng đã làm sai.

Chúng tôi hy vọng rằng ứng dụng này sẽ trở thành một công cụ hữu hiệu, hỗ trợ người dùng không chỉ trong việc ôn tập kiến thức lý thuyết giao thông mà còn giúp nâng cao ý thức và trách nhiệm khi tham gia giao thông. Với 600 câu hỏi trắc nghiệm chính thức và nhiều chế độ thi đa dạng, "Thi Bằng Lái Xe" sẽ mang đến cho người dùng những trải nghiệm học tập thú vị và hiệu quả.

**MỤC LỤC**

[CHƯƠNG 1. MÔ TẢ BÀI TOÁN 7](#_Toc181311179)

[1.1. Giới thiệu 7](#_Toc181311180)

[1.2. Chức năng chính 7](#_Toc181311181)

[1. Thi đề ngẫu nhiên 7](#_Toc181311182)

[2. Thi Theo bộ đề 7](#_Toc181311183)

[3. Các câu bị sai 8](#_Toc181311184)

[4. Ôn tập câu hỏi 8](#_Toc181311185)

[5. Các biển báo 8](#_Toc181311186)

[6. Các mẹo ghi nhớ 8](#_Toc181311187)

[7. 60 Câu điểm liệt. 8](#_Toc181311188)

[8. Top 50 câu dễ sai. 8](#_Toc181311189)

[9. Xóa dữ liệu 8](#_Toc181311190)

[10. Lưu trữ dữ liệu. 9](#_Toc181311191)

[1.3. Yêu cầu phi chức năng 9](#_Toc181311192)

[Dễ Sử Dụng 9](#_Toc181311193)

[Hiệu Năng 9](#_Toc181311194)

[Độ Tin Cậy 10](#_Toc181311195)

[Tính Tương Thích 10](#_Toc181311196)

[Khả Năng Mở Rộng 10](#_Toc181311197)

[CHƯƠNG 2. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG 11](#_Toc181311198)

[2.1. Phân tích yêu cầu: 11](#_Toc181311199)

[Xác định người dùng: 11](#_Toc181311200)

[Thu thập yêu cầu: 11](#_Toc181311201)

[Phân tích yêu cầu: 11](#_Toc181311202)

[2.2. Thiết kế hệ thống: 13](#_Toc181311203)

[2.2.1. Biểu đồ lớp cho mô hình miền 13](#_Toc181311204)

[2.2.2. Biểu đồ lớp cho MVC 14](#_Toc181311205)

[2.2.3. Thiết kế cơ sở dữ liệu: 28](#_Toc181311206)

[2.2.4. Thiết kế giao diện: 30](#_Toc181311207)

[CHƯƠNG 3. KẾT QUẢ THỰC HIỆN 30](#_Toc181311208)

[3.1. Công nghệ đã sử dụng 30](#_Toc181311209)

[3.2. Tiến độ thực hiện 30](#_Toc181311210)

[3.3 Hình ảnh sản phẩm 32](#_Toc181311211)

[CHƯƠNG 4: MÃ NGUỒN ĐẦY ĐỦ 32](#_Toc181311212)

**BẢNG PHÂN CHIA CÔNG VIỆC THEO CHỨC NĂNG CHÍNH:**

|  |  |  |
| --- | --- | --- |
| **STT** | **THÀNH VIÊN** | **CHỨC NĂNG ĐẢM NHẬN** |
| **1** | **Trần Trường Long** | * **Thi theo đề ngẫu nhiên** * **Thi theo bộ đề** * **Xóa dữ liệu** * **120 mô phỏng** |
| **2** | **Nguyễn Đức Minh** | * **Các câu bị sai** * **Các biển báo** * **60 câu điểm liệt** |
| **3** | **Hoàng Đăng Tú Tài** | * **Ôn tập theo câu hỏi** * **Mẹo ghi nhớ** * **Top 50 câu sai** |

**BẢNG CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **TỪ VIẾT TẮT** | **VIẾT ĐẦY ĐỦ** |
| 1 | CSDL | Cơ sở dữ liệu |

CHƯƠNG 1. MÔ TẢ BÀI TOÁN

## 1.1. Giới thiệu

Ứng dụng Thi Bằng Lái Xe là một phần mềm di động được phát triển nhằm hỗ trợ người dùng ôn tập và chuẩn bị cho kỳ thi cấp giấy phép lái xe tại Việt Nam. Phần mềm này cung cấp các tính năng quan trọng như làm bài thi thử, xem danh sách các câu hỏi trắc nghiệm, lưu lịch sử thi và giải thích chi tiết cho mỗi câu hỏi.

Với giao diện thân thiện và dễ sử dụng, ứng dụng cho phép người dùng dễ dàng lựa chọn các bộ câu hỏi từ 600 câu trắc nghiệm chính thức. Ứng dụng cũng hỗ trợ sinh ngẫu nhiên câu hỏi và đáp án, giúp người dùng làm quen với nhiều dạng đề khác nhau, từ đó nắm vững kiến thức lý thuyết giao thông.

***Tổng quan giới thiệu phần mềm:***

https://www.canva.com/design/DAGVImpxtrM/3TQQDgkaDC-FmR9oyoGzkA/view

## 1.2. Chức năng chính

### 1. Thi đề ngẫu nhiên

Ứng dụng cung cấp danh sách đầy đủ 600 câu hỏi trắc nghiệm chính thức, được sử dụng trong kỳ thi cấp giấy phép lái xe tại Việt Nam. Chúng được chia ngẫu nhiên thành các bộ đề và tính năng này cho phép người dùng làm bài kiểm tra với 1 bộ đề bất kì. Khi làm sẽ có giới hạn thời gian tổng thể, người dùng có thể nộp bài bất cứ khi nào chứ không bắt buộc làm hết. Sau khi nộp bài sẽ trả về kết quả xem người dùng có hoàn thành bài thi hay không, với 1 số tiêu chí như không được làm sai câu điểm liệt, số câu đúng phải thỏa mãn với yêu cầu tối thiểu số câu đúng của bài thi.

### 2. Thi Theo bộ đề

Trái nghiệm với tính năng thi theo đề ngẫu nhiên, thì thi theo bộ đề lại cho phép người dùng có thể tự chọn 1 bộ đề và làm cho đến khi thông thạo nó. Nhìn chung thì hai tính năng này giống như hai dạng luyện tập để người dùng có thể tự do lựa chọn cho phù hợp với phương pháp học của họ. Khi làm sẽ có giới hạn thời gian tổng thể, người dùng có thể nộp bài bất cứ khi nào chứ không bắt buộc làm hết. Sau khi nộp bài sẽ trả về kết quả xem người dùng có hoàn thành bài thi hay không, với 1 số tiêu chí như không được làm sai câu điểm liệt, số câu đúng phải thỏa mãn với yêu cầu tối thiểu số câu đúng của bài thi.

### 3. Các câu bị sai

Hệ thống sẽ lưu lại các câu hỏi người dùng đã làm sai vào CSDL. Khi người dùng chọn làm mục này , họ sẽ tập trung học những kiến thức hổng, giúp cho việc luyện tập toàn diện hơn.

### 4. Ôn tập câu hỏi

Nếu như hai tính năng Thi theo bộ đề ngẫu nhiên hoặc cụ thể, sẽ giúp người dùng có trải nghiệm thi thử, với số câu và thời gian tương đồng với khi thi thật. Thì tính năng này lại phân chia câu hỏi theo mức độ khó, nội dung và bối cảnh để phân thành những dạng bài cụ thể cho người dùng.

### 5. Các biển báo

Tập hợp danh sách các biển báo giao thông, có phân nhóm thành biển cảnh báo, biển cấm,.. Để người dùng tập trung vào những dạng bài có biển báo, vì đây cũng là 1 dạng bài khó.

### 6. Các mẹo ghi nhớ

Tập hợp các mẹo làm bài, ghi nhớ giúp người dùng có thể tiết kiệm thời gian khi học, tuy nhiên người dùng chỉ nên sử dụng tính năng này sau khi đã nắm cơ bản được kiến thức, không nên lợi dụng để học qua loa, học vẹt.

### 7. 60 Câu điểm liệt.

Tính năng này tập trung phát triển nhằm hướng đến 1 yêu cầu vô cùng quan trọng khi thi. Đó là người dùng bắt buộc phải hoàn thành đúng tất cả các câu điểm liệt trong đề, thì mới xét đến các tiêu chí tiếp theo để xét người dùng có qua hay không. Người dùng chọn tính năng này để ôn tâp 60 câu điểm liệt.

### 8. Top 50 câu dễ sai.

Tính năng này tập trung vào 50 câu hỏi mà đa số người dùng thường xuyên làm sai, dễ làm sai khi làm bài thi. Người dùng chọn tính năng này để tập trung ôn 50 câu dễ sai.

### 9. Xóa dữ liệu

Tính năng xóa dữ liệu giúp người dùng reset lại lịch sử thi của phiên sử dụng trước, hỗ trợ cho việc luyện tập diễn ra đúng với mong muốn của người dùng.

### 10. Lưu trữ dữ liệu.

Ứng dụng Thi Bằng Lái Xe Ô Tô sử dụng **SQLite** làm giải pháp lưu trữ dữ liệu chính. SQLite là một cơ sở dữ liệu nhúng nhẹ, cung cấp cách thức lưu trữ và truy xuất dữ liệu hiệu quả mà không cần máy chủ cơ sở dữ liệu phức tạp. Dữ liệu trong ứng dụng này bao gồm các câu hỏi, thuộc tính phân loại và các thông tin liên quan đến kết quả thi của người dùng.

#### Ưu điểm của việc sử dụng SQLite

* **Hiệu năng cao:** SQLite hoạt động rất nhanh và hiệu quả ngay cả với lượng dữ liệu lớn như 600 câu hỏi trắc nghiệm và nhiều thông tin phân loại.
* **Dễ dàng tích hợp:** Là một cơ sở dữ liệu nhúng, SQLite không yêu cầu máy chủ riêng biệt, do đó dễ dàng tích hợp vào các ứng dụng di động và hoạt động tốt ngay cả khi ngoại tuyến.
* **Bảo mật và riêng tư:** Dữ liệu được lưu trữ cục bộ trên thiết bị người dùng, đảm bảo tính bảo mật và không yêu cầu kết nối mạng để truy cập dữ liệu.
* **Khả năng mở rộng:** SQLite dễ dàng mở rộng cho các bản cập nhật sau này khi cần thêm nhiều câu hỏi, thuộc tính hoặc thông tin mới.

## 1.3. Yêu cầu phi chức năng

### Dễ Sử Dụng

Ứng dụng được thiết kế với giao diện trực quan và thân thiện với người dùng. Tất cả các tính năng như xem câu hỏi, làm bài thi thử, và xem lại lịch sử thi đều dễ dàng truy cập và thao tác. Người dùng ở mọi cấp độ kỹ năng, kể cả những người không quen thuộc với công nghệ, có thể nhanh chóng làm quen và sử dụng ứng dụng một cách dễ dàng.

### Hiệu Năng

Ứng dụng đảm bảo hoạt động mượt mà và nhanh chóng, ngay cả khi xử lý một lượng lớn dữ liệu câu hỏi và bài thi. Nhờ việc sử dụng Room Database để lưu trữ dữ liệu cục bộ, người dùng có thể truy cập danh sách câu hỏi và kết quả thi một cách nhanh chóng mà không cần chờ tải dữ liệu từ server.

### Độ Tin Cậy

Dữ liệu của người dùng như kết quả bài thi và lịch sử thi được lưu trữ an toàn trong cơ sở dữ liệu cục bộ. Ứng dụng đảm bảo rằng thông tin này sẽ không bị mất mát ngay cả khi ứng dụng bị tắt đột ngột hoặc thiết bị gặp sự cố. Việc lưu trữ cục bộ cũng giúp ứng dụng hoạt động mà không cần kết nối internet, nâng cao tính khả dụng và độ tin cậy.

### Tính Tương Thích

Ứng dụng được phát triển để tương thích với nhiều loại thiết bị và kích cỡ màn hình khác nhau, từ điện thoại di động đến máy tính bảng. Điều này đảm bảo rằng người dùng có thể sử dụng ứng dụng trên nhiều nền tảng mà không gặp phải vấn đề về giao diện hay trải nghiệm.

### Khả Năng Mở Rộng

Ứng dụng được thiết kế linh hoạt và dễ dàng mở rộng thêm các tính năng mới hoặc cập nhật nội dung câu hỏi khi cần. Điều này đảm bảo rằng ứng dụng có thể phát triển và cải thiện theo nhu cầu của người dùng trong tương lai.

CHƯƠNG 2. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG

## 2.1. Phân tích yêu cầu:

### Xác định người dùng:

* Người dùng cuối là những người đang chuẩn bị tham gia kỳ thi cấp giấy phép lái xe hạng B1, B2 tại Việt Nam. Họ có thể là học sinh, sinh viên, người đi làm, hoặc bất kỳ ai có nhu cầu thi và ôn tập các câu hỏi lý thuyết về luật giao thông đường bộ.
* Đặc điểm người dùng: Ứng dụng cần dễ sử dụng và thân thiện với người dùng, vì người dùng có thể có các mức độ hiểu biết khác nhau về công nghệ và điện thoại di động. Ứng dụng phải đảm bảo rằng người dùng có thể dễ dàng điều hướng và truy cập các tính năng như xem câu hỏi, thi thử, và xem lại lịch sử thi.

### Thu thập yêu cầu:

* Chức năng chính: Dựa trên mô tả bài toán, các chức năng chính của ứng dụng bao gồm:
* Thi theo đề ngẫu nhiên
* Thi theo bộ đề
* Làm các câu đã bị sai
* Ôn tập câu hỏi
* Các biển báo
* Các mẹo ghi nhớ
* Ôn tập 60 câu điểm liệt
* Ôn tập 50 câu sai điển hình
* Các câu hỏi có hình ảnh(nếu đề bài yêu cầu).
* Giới hạn thời gian cho bài thi theo đề.
* Xóa dữ liệu
* Lưu trữ dữ liệu.
* Yêu cầu phi chức năng: Ứng dụng cần đảm bảo tính dễ sử dụng, hiệu suất cao, độ tin cậy, tính tương thích với nhiều loại thiết bị và khả năng mở rộng trong tương lai.

### Phân tích yêu cầu:

* **Thi theo bộ đề ngẫu nhiên:** Người dùng có thể truy cập tính năng này để thi ngẫu nhiên 1 bộ đề. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Thi theo bộ đề cụ thể:** Người dùng có thể truy cập tính năng này để chọn và luyện tập cụ thể, để thông thạo từng bộ đề 1 cách nhanh chóng. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Làm các câu đã bị sai:** Người dùng có thể truy cập tính năng này để chọn và luyện tập tập trung vào các câu đã làm sai trước đây, phù hợp để khắc phục lỗ hổng kiến thức. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Ôn tập câu hỏi theo dạng:** Người dùng có thể truy cập tính năng này để chọn và luyện tập cụ thể theo từng dạng câu hỏi. Các câu hỏi sẽ phân loại theo độ khó, hoặc dạng bài.Phù hợp với những người muốn luyện tập theo dạng bài cụ thể. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Các biển báo:** Người dùng có thể truy cập tính năng này để học và ghi nhớ hệ thống biển báo giao thông, các biển báo có phân loại thành các loại như (biển báo cấm, biển báo nguy hiểm, biển báo hiệu lệnh, biển báo chỉ dẫn,…)
* **Các mẹo ghi nhớ:** Tổng hợp những mẹo để người dùng có thể học, hoặc làm bài thi 1 cách nhanh chóng hơn, tiết kiệm thời gian khi luyện tập và thực hành.
* **60 Câu điểm liệt:** Tính năng này tập trung phát triển nhằm hướng đến 1 yêu cầu vô cùng quan trọng khi thi. Đó là người dùng bắt buộc phải hoàn thành đúng tất cả các câu điểm liệt trong đề, thì mới xét đến các tiêu chí tiếp theo để xét người dùng có qua hay không. Người dùng chọn tính năng này để ôn tâp 60 câu điểm liệt. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Top 50 câu dễ sai:** Tính năng này tập trung vào 50 câu hỏi mà đa số người dùng thường xuyên làm sai, dễ làm sai khi làm bài thi. Người dùng chọn tính năng này để tập trung ôn 50 câu dễ sai. Với mỗi câu hỏi bao gồm nội dung câu hỏi, các đáp án lựa chọn (A, B, C, D) và phần giải thích. Các câu hỏi có thể bao gồm hình ảnh minh họa.
* **Xóa dữ liệu:** Khi 1 phiên sử dụng hệ thống kết thúc, do tính năng lưu trữ dữ liệu sẽ lưu trữ những kết quả(câu sai, đề đã làm) của phiên sử dụng đó, và người dùng sẽ dựa vào đó để luyện tập tập trung vào những mảng khuyết điểm của họ. Tuy nhiên đôi khi người dùng muốn bắt đầu lại, hoặc nhiều lí do khác khiến lịch sử dữ liệu làm bài không được khách quan với họ, thì tính năng này giúp họ reset hệ thống và bắt đầu lại từ đầu.
* **Lưu trữ dữ liệu:** Ứng dụng lưu trữ các câu hỏi thi, thông tin phân loại, lịch sử thi và các câu trả lời sai vào cơ sở dữ liệu SQLite để dễ dàng truy xuất và xử lý trong các chức năng khác nhau.

## 2.2. Thiết kế hệ thống:

### 2.2.1. Biểu đồ lớp cho mô hình miền

Biểu đồ lớp mô hình miền tập trung vào các **thực thể nghiệp vụ chính** của ứng dụng, bao gồm các lớp đại diện cho câu hỏi, loại bài thi, và dữ liệu người dùng. Dưới đây là mô tả cho các thực thể chính:

**Lớp chính trong mô hình miền:**

* **Question**: Đại diện cho một câu hỏi trắc nghiệm trong ứng dụng.
  + Thuộc tính: id, questionText, answers, options ,suggest, image.
* **DriverType**: Đại diện cho loại bằng lái xe (B1, B2,...).
  + Thuộc tính: id, title, quantity, description.
* **Options**: Đại diện cho các đáp án
  + Thuộc tính: id, content
* **Sign**: Đại diện cho các biển báo
  + Thuộc tính: id, image, description

### 2.2.2. Biểu đồ lớp cho MVC

Biểu đồ lớp MVC mô tả kiến trúc tổng thể của ứng dụng, bao gồm các lớp thuộc về **Model (nghiệp vụ)**, **View (giao diện)**, và **Controller (điều khiển)**.

**Các lớp trong MVC:**

* **Model**:
  + **Question**: Lớp mô tả câu hỏi, bao gồm thông tin về câu hỏi, đáp án, hình ảnh liên quan.
  + **DriverType**: Lớp mô tả các loại bằng lái xe.
  + **Options**: Lớp mô tả các đáp án của câu hỏi, bao gồm nội dung của đáp án.
  + **Sign:** Lớp đại diện cho biển báo giao thông, bao gồm ảnh và mô tả
* **View**:
  + **QuestionAdapter**: Hiển thị danh sách câu hỏi và các tuỳ chọn trả lời.
  + **DriverTypeAdapter**: Xử lý việc hiển thị và tương tác với danh sách loại câu hỏi
  + **SignAdapter**: Hiển thị danh sách biển báo và tùy chọn chi tiết của biển báo.
* **Controller**:
  + **MainActivity**: Điều hướng chính của ứng dụng, khởi tạo các hoạt động và chuyển đổi giữa các màn hình.
  + **DatabaseHelper**: Xử lý việc lưu trữ và truy vấn dữ liệu trong SQLite.
  + **DriverTypeActivity**: Xử lí logic danh sách loại câu hỏi để người dùng chọn.
  + **ExamsActivity**: Xử lí logic danh sách bộ đề và cho phép người dùng bắt đầu thi.
  + **TestActivity**: Xử lí logic câu hỏi, đáp án trong quá trình làm bài thi.
  + **ResultActivity**: Xử lí logic kết quả thi sau khi hoàn thành.
  + **SignActivity**: Xử lí logic danh sách biển báo
  + **SignDetailActivity**: Xử lí logic dữ liệu chi tiết của các biển báo
  + **TipsActivity**: Xử lí logic danh sách mẹo khi làm bài thi

**Mô tả chi tiết biểu đồ MVC:**

* **Model**: Gồm các lớp Question, DriverType, Options, và Sign – chịu trách nhiệm lưu trữ và xử lý dữ liệu nghiệp vụ chính.
* **View**: Các lớp giao diện như DriverTypeAdapter, QuestionAdapter, SignAdapter sẽ hiển thị dữ liệu từ model và nhận tương tác từ người dùng.
* **Controller**: Các lớp như MainActivity, TestActivity,... và DatabaseHelper đảm nhiệm việc điều khiển luồng dữ liệu giữa Model và View, cũng như xử lý thao tác của người dùng.

**Dựa trên kiến trúc MVC đã chọn cho ứng dụng ôn thi GPLX, đây là các lớp có thể xác định:**

*Các lớp Model*

**Lớp Question:**

• **Thuộc tính:**

* private String id: Mã định danh của câu hỏi.
* private String questionText: Nội dung của câu hỏi.
* private Options options: Các lựa chọn trả lời.
* private String answer: Đáp án chính xác của câu hỏi.
* private String suggest: Gợi ý để trả lời câu hỏi (nếu có).
* private Image image: Hình ảnh đi kèm câu hỏi (nếu có), được định nghĩa là một lớp lồng trong Question.

• **Phương thức:**

* getId(): Trả về mã định danh của câu hỏi.
* getQuestionText(): Trả về nội dung của câu hỏi.
* getOptions(): Trả về các lựa chọn của câu hỏi (đối tượng Options).
* getAnswer(): Trả về đáp án của câu hỏi.
* getSuggest(): Trả về gợi ý cho câu hỏi (nếu có).
* getImage(): Trả về hình ảnh đi kèm của câu hỏi (nếu có).

• **Lớp lồng Image:**

* **Thuộc tính:**
  + private String img1: Đường dẫn hoặc mã nhận diện hình ảnh.
* **Phương thức:**
  + getImg1(): Trả về đường dẫn hoặc mã của hình ảnh.

Lớp Options:

• **Thuộc tính:**

* private String A: Nội dung của lựa chọn A.
* private String B: Nội dung của lựa chọn B.
* private String C: Nội dung của lựa chọn C.
* private String D: Nội dung của lựa chọn D.
* private String E: Nội dung của lựa chọn E.

• **Phương thức:**

* getA(): Trả về nội dung của lựa chọn A.
* getB(): Trả về nội dung của lựa chọn B.
* getC(): Trả về nội dung của lựa chọn C.
* getD(): Trả về nội dung của lựa chọn D.
* getE(): Trả về nội dung của lựa chọn E.

Lớp DriverType:

• **Thuộc tính:**

* private int id: Mã định danh cho loại câu hỏi.
* private String title: Tiêu đề hoặc tên của loại câu hỏi.
* private int quantity: Số lượng câu hỏi thuộc loại này.
* private String description: Mô tả chi tiết về loại câu hỏi.

• **Phương thức:**

* getId(): Trả về mã định danh của loại câu hỏi.
* getTitle(): Trả về tiêu đề hoặc tên của loại câu hỏi.
* getQuantity(): Trả về số lượng câu hỏi thuộc loại này.
* getDescription(): Trả về mô tả chi tiết về loại câu hỏi.

Lớp Sign:

• **Thuộc tính:**

* private String name: Tên của biển báo.
* private String des: Mô tả về biển báo.
* private String imagePath: Đường dẫn đến hình ảnh biển báo.
* private int titleIndex: Chỉ số tiêu đề, dùng để phân loại khi đối tượng là tiêu đề.

• **Phương thức:**

* Sign(String name, String des, String imagePath): Constructor tạo đối tượng biển báo thông thường với giá trị mặc định cho titleIndex là -1.
* Sign(String name, String des, String imagePath, int titleIndex): Constructor tạo đối tượng với chỉ số tiêu đề.
* getName(): Trả về tên biển báo.
* getDes(): Trả về mô tả của biển báo.
* getImagePath(): Trả về đường dẫn hình ảnh hoặc ảnh mặc định bienbao.png nếu không có hình ảnh.
* getTitleIndex(): Trả về chỉ số tiêu đề để phân loại khi đối tượng là tiêu đề.

Các lớp hỗ trợ giao diện:

**Lớp DriverTypeAdapter:**

* **Thuộc tính:**
  + private Context context: Ngữ cảnh mà adapter đang hoạt động, thường là Activity hoặc Fragment.
  + private List<DriverType> driverTypeList: Danh sách các loại câu hỏi (DriverType) mà adapter sẽ hiển thị.
* **Phương thức:**
  + public DriverTypeAdapter(Context context, List<DriverType> driverTypeList): Constructor để khởi tạo các thuộc tính của adapter.
  + @NonNull @Override public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType): Phương thức này tạo ra một ViewHolder mới bằng cách inflate layout cho một item trong danh sách.
  + @Override public void onBindViewHolder(@NonNull ViewHolder holder, int position): Phương thức này gán dữ liệu cho ViewHolder từ danh sách driverTypeList:
    - Lấy một đối tượng DriverType từ danh sách theo vị trí position.
    - Thiết lập tiêu đề và mô tả cho titleTextView và descriptionTextView.
    - Cập nhật số lượng câu hỏi cho quantityTextView.
    - Thiết lập một listener cho itemView, khi người dùng nhấn vào sẽ chuyển đến TestActivity và truyền theo chỉ số examsIndex (ID loại câu hỏi).
  + @Override public int getItemCount(): Phương thức này trả về kích thước của danh sách driverTypeList, cho biết số lượng item mà adapter sẽ hiển thị.
* **Lớp ViewHolder:**
  + public static class ViewHolder extends RecyclerView.ViewHolder: Đây là lớp lồng bên trong, dùng để giữ các view cho mỗi item trong danh sách.
    - TextView titleTextView: TextView hiển thị tiêu đề loại câu hỏi.
    - TextView descriptionTextView: TextView hiển thị mô tả loại câu hỏi.
    - TextView quantityTextView: TextView hiển thị số lượng câu hỏi thuộc loại đó.

**Lớp QuestionAdapter**

**Thuộc tính**

* **List<Question> questions**: Danh sách các câu hỏi được hiển thị.
* **Map<String, String> selectedAnswers**: Lưu trữ các câu trả lời đã chọn của người dùng theo định dạng khóa-giá trị.
* **List<String> criticals**: Danh sách các câu hỏi điểm liệt (nếu có).

**Phương thức**

* **QuestionAdapter(List<Question> questions, List<String> criticals)**: Khởi tạo adapter với danh sách câu hỏi và các câu hỏi điểm liệt.
* **@NonNull QuestionViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)**: Tạo một QuestionViewHolder mới cho từng item trong danh sách câu hỏi.
* **void onBindViewHolder(@NonNull QuestionViewHolder holder, int position)**:
  + Gán dữ liệu cho từng câu hỏi vào ViewHolder.
  + Hiển thị câu hỏi, hình ảnh (nếu có), và các tùy chọn đáp án.
  + Thiết lập sự kiện lắng nghe cho RadioGroup để lưu lựa chọn của người dùng.
* **int getItemCount()**: Trả về số lượng câu hỏi trong danh sách.
* **Map<String, String> getSelectedAnswers()**: Trả về bản đồ các câu trả lời đã chọn.

**Lớp con QuestionViewHolder**

* **TextView questionTextView**: Hiển thị văn bản câu hỏi.
* **ImageView questionImageView**: Hiển thị hình ảnh liên quan đến câu hỏi.
* **RadioGroup answerGroup**: Nhóm các nút radio cho các đáp án.
* **RadioButton optionAButton, optionBButton, optionCButton, optionDButton, optionEButton**: Các nút radio cho các tùy chọn đáp án A, B, C, D, và E.

**Lớp SignAdapter**

* **Thuộc tính:**
  + private Context context: Context được truyền vào, dùng để thực hiện các tác vụ liên quan đến UI.
  + private List<Sign> signList: Danh sách các đối tượng Sign mà adapter sẽ hiển thị.
  + private int titleCount = 0: Biến đếm số lượng tiêu đề đã gặp trong danh sách.
* **Constructor:**
  + public SignAdapter(Context context, List<Sign> signList): Khởi tạo adapter với context và danh sách các đối tượng Sign.
* **Phương thức:**
  + @NonNull @Override public SignViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType): Tạo một ViewHolder mới bằng cách inflate layout cho mỗi item của biển báo.
  + @Override public void onBindViewHolder(@NonNull SignViewHolder holder, int position): Gán dữ liệu cho ViewHolder:
    - Lấy đối tượng Sign theo vị trí và kiểm tra xem đó có phải là tiêu đề hay không (dựa vào mô tả và đường dẫn hình ảnh).
    - **Nếu là tiêu đề:**
      * Cấu hình văn bản tiêu đề (cỡ chữ, kiểu chữ, màu chữ).
      * Đặt màu nền dựa trên chỉ số tiêu đề (titleIndex).
      * Đảm bảo rằng mô tả và hình ảnh không hiển thị.
    - **Nếu không phải là tiêu đề:**
      * Cấu hình văn bản biển báo thông thường (cỡ chữ, kiểu chữ).
      * Hiển thị hoặc ẩn mô tả và hình ảnh tùy thuộc vào dữ liệu của đối tượng Sign.
      * Sử dụng Glide để tải hình ảnh nếu có đường dẫn hợp lệ.
  + holder.itemView.setOnClickListener(...): Xử lý sự kiện nhấp chuột vào item để chuyển đến SignDetailActivity, gửi thông tin biển báo (tên, mô tả, đường dẫn hình ảnh).
  + @Override public int getItemCount(): Trả về số lượng đối tượng trong danh sách signList.
* **Lớp SignViewHolder:**
  + public static class SignViewHolder extends RecyclerView.ViewHolder: Lớp lồng bên trong dùng để giữ các view cho mỗi biển báo.
    - TextView nameTextView: TextView để hiển thị tên biển báo.
    - TextView descriptionTextView: TextView để hiển thị mô tả biển báo.
    - ImageView imageView: ImageView để hiển thị hình ảnh biển báo.

*Các lớp Controller:*

**Lớp DriverTypeActivity**

* **Thuộc tính:**
  + private RecyclerView recyclerView: RecyclerView để hiển thị danh sách các loại lái xe.
  + private DriverTypeAdapter adapter: Adapter cho RecyclerView để quản lý và hiển thị dữ liệu.
  + private List<DriverType> driverTypeList: Danh sách các loại lái xe.
  + private ImageButton backButton: Nút để quay lại màn hình trước đó.
* **Phương thức:**
  + @Override protected void onCreate(@Nullable Bundle savedInstanceState):
    - Thiết lập giao diện cho activity và ánh xạ các view.
    - Khởi tạo driverTypeList và gọi phương thức loadDriverTypes() để tải dữ liệu loại lái xe từ tệp JSON.
    - Thiết lập adapter và layout manager cho RecyclerView.
    - Thiết lập sự kiện cho nút Back để quay lại màn hình chính (MainActivity).
  + private void loadDriverTypes():
    - Tải dữ liệu từ tệp JSON có tên driver\_type.json từ thư mục assets.
    - Đọc nội dung tệp và phân tích cú pháp JSON để lấy danh sách các loại lái xe.
    - Duyệt qua mảng các loại lái xe, tạo đối tượng DriverType cho từng loại và thêm vào danh sách driverTypeList.
    - Nếu có lỗi xảy ra trong quá trình tải dữ liệu, hiển thị thông báo lỗi.

**Lớp ExamsActivity**

* **Thuộc tính:**
  + private DatabaseHelper databaseHelper: Khai báo một đối tượng DatabaseHelper để tương tác với cơ sở dữ liệu.
* **Phương thức:**
  + @Override protected void onCreate(Bundle savedInstanceState):
    - Gọi phương thức khởi tạo của lớp cha và thiết lập giao diện cho activity với layout activity\_exams.
    - Khởi tạo databaseHelper để sử dụng các phương thức truy cập cơ sở dữ liệu.
    - Ánh xạ đến nút Back và thiết lập sự kiện cho nó để quay lại màn hình chính (MainActivity).
    - Ánh xạ đến GridLayout để chứa các nút bài kiểm tra.
  + **Thiết lập các nút bài kiểm tra:**
    - Vòng lặp qua tất cả các con cái của GridLayout để xác định và cấu hình các nút bài kiểm tra.
    - Lưu chỉ số của từng nút (tương ứng với bài kiểm tra) vào biến examIndex.
    - Lấy trạng thái bài kiểm tra từ cơ sở dữ liệu bằng phương thức getExamStatus(examIndex) của databaseHelper.
    - Dựa vào trạng thái, thay đổi màu nền và màu chữ của nút:
      * **Trạng thái 1**: Nút màu xanh lá cây với chữ trắng (đã hoàn thành).
      * **Trạng thái -1**: Nút màu đỏ với chữ trắng (không hoàn thành).
      * **Trạng thái khác**: Nút màu xanh lam với chữ đen (chưa kiểm tra).
    - Thiết lập sự kiện click cho từng nút để chuyển đến TestActivity và truyền chỉ số bài kiểm tra (examsIndex).

**Lớp MainActivity**

* **Phương thức:**
  + @Override protected void onCreate(Bundle savedInstanceState):
    - Gọi phương thức khởi tạo của lớp cha và thiết lập giao diện cho activity với layout activity\_main.
    - Sử dụng EdgeToEdge.enable(this) để làm cho activity có thể mở rộng đến các cạnh của màn hình (tùy thuộc vào phiên bản Android).

**Lớp ResultActivity**

**Thuộc tính**

* **resultText**: TextView hiển thị điểm số của người dùng.
* **totalQuestionsText**: TextView hiển thị tổng số câu hỏi trong bài thi.
* **resultMessage**: TextView hiển thị thông báo kết quả thi (đậu hoặc rớt).
* **backButton**: Nút để người dùng quay về trang chính.
* **toolbar**: Thanh công cụ ở phía trên giao diện.
* **toolbarTitle**: TextView cho tiêu đề trên thanh công cụ.
* **databaseHelper**: Đối tượng quản lý cơ sở dữ liệu.

**Phương thức**

* **onCreate(Bundle savedInstanceState)**: Khởi tạo giao diện và nhận dữ liệu từ Intent, thiết lập hiển thị cho các thuộc tính.
* **goToMain()**: Quay về trang chính của ứng dụng khi người dùng nhấn nút quay lại.

**Lớp SignActivity**

**Thuộc tính**

* **recyclerView**: RecyclerView để hiển thị danh sách biển báo giao thông.
* **adapter**: Đối tượng SignAdapter để quản lý và hiển thị dữ liệu trong RecyclerView.
* **signList**: Danh sách chứa các đối tượng Sign đại diện cho thông tin biển báo.

**Phương thức**

* **onCreate(Bundle savedInstanceState)**: Khởi tạo giao diện, thiết lập RecyclerView, và gọi phương thức để tải dữ liệu biển báo.
* **loadSignData()**: Tải dữ liệu biển báo từ tệp JSON (BienBao.json) trong thư mục assets, phân tích cú pháp và thêm vào signList. Nếu có lỗi trong quá trình tải, hiển thị thông báo Toast cho người dùng.

**Lớp SignDetailActivity**

**Thuộc tính**

* **nameTextView**: TextView để hiển thị tên của biển báo.
* **descriptionTextView**: TextView để hiển thị mô tả của biển báo.
* **imageView**: ImageView để hiển thị hình ảnh của biển báo.

**Phương thức**

* **onCreate(Bundle savedInstanceState)**:
  + Khởi tạo giao diện và liên kết các thuộc tính với các view trong layout.
  + Nhận dữ liệu từ Intent bao gồm tên, mô tả, và đường dẫn hình ảnh của biển báo.
  + Hiển thị dữ liệu vào các view tương ứng.
  + Sử dụng thư viện Glide để tải hình ảnh vào imageView, có placeholder và hình ảnh lỗi.
  + Thiết lập nút "Quay lại" để kết thúc activity hiện tại.
  + Cập nhật tiêu đề cho toolbar bằng tên của biển báo.

**Lớp TipsActivity**

**Phương thức**

* **onCreate(Bundle savedInstanceState)**:
  + Khởi tạo giao diện và thiết lập layout cho activity.
  + Liên kết nút "Quay lại" với một ImageButton trong layout.
  + Thiết lập sự kiện click cho nút "Quay lại", điều hướng về MainActivity khi nút được nhấn.

**Lớp DatabaseHelper**

Lớp DatabaseHelper được sử dụng để quản lý cơ sở dữ liệu SQLite cho ứng dụng quiz. Nó định nghĩa các bảng và cung cấp các phương thức để thao tác với dữ liệu.

**Các thuộc tính**

* **DATABASE\_NAME**: Tên của cơ sở dữ liệu.
* **DATABASE\_VERSION**: Phiên bản của cơ sở dữ liệu.
* **TABLE\_INCORRECT\_ANSWERS**: Tên bảng lưu trữ các câu trả lời sai.
* **COL\_1, COL\_2**: Các cột trong bảng incorrect\_answers.
* **TABLE\_EXAM\_RESULTS**: Tên bảng lưu trữ trạng thái bài thi.
* **COL\_EXAM\_INDEX, COL\_EXAM\_STATUS**: Các cột trong bảng exam\_results.

**Phương thức**

* **DatabaseHelper(Context context)**: Constructor khởi tạo cơ sở dữ liệu với tên và phiên bản đã định nghĩa.
* **void onCreate(SQLiteDatabase db)**:
  + Tạo các bảng incorrect\_answers và exam\_results.
  + Chèn 18 bản ghi vào bảng exam\_results với trạng thái mặc định là 0.
* **void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)**:
  + Xóa các bảng nếu chúng tồn tại và tạo lại chúng với cấu trúc mới.
* **void insertIncorrectAnswer(int questionId)**: Chèn một câu trả lời sai vào bảng incorrect\_answers.
* **void deleteCorrectAnswer(int questionId)**: Xóa một câu trả lời sai khỏi bảng incorrect\_answers theo questionId.
* **ArrayList<Integer> getIncorrectAnswers()**: Trả về danh sách các ID câu hỏi mà người dùng đã trả lời sai.
* **void insertOrUpdateExamResult(int examIndex, int examStatus)**: Chèn hoặc cập nhật trạng thái bài thi vào bảng exam\_results.
* **Integer getExamStatus(int examIndex)**: Lấy trạng thái bài thi theo examIndex.
* **void resetData()**: Đặt lại dữ liệu trong bảng incorrect\_answers và exam\_results, khởi tạo lại trạng thái bài thi về 0.

*Mối quan hệ giữa các lớp:*

**1. MainActivity**

* **Mối quan hệ với các lớp khác**:
  + Điều hướng tới:
    - SignDetailActivity: Xem chi tiết về việc đăng ký hoặc thông tin người dùng.
    - TipsActivity: Cung cấp các mẹo và thông tin hữu ích.
    - QuestionActivity: Hiển thị câu hỏi cụ thể (nếu có).
    - TestActivity: Thực hiện bài kiểm tra, nơi người dùng có thể trả lời các câu hỏi.

**2. TestActivity**

* **Mối quan hệ với các lớp khác**:
  + Sử dụng QuestionAdapter để hiển thị danh sách câu hỏi trong RecyclerView.
  + Quản lý danh sách Question (List<Question>), cung cấp các câu hỏi từ nhiều nguồn khác nhau.
  + Sử dụng DatabaseHelper để quản lý và lưu trữ câu trả lời sai cũng như trạng thái bài thi.

**3. QuestionAdapter**

* **Mối quan hệ với Question**:
  + Sử dụng các đối tượng Question để hiển thị thông tin câu hỏi trong RecyclerView của TestActivity.
* **Lưu trữ các câu trả lời đã chọn** thông qua Map<String, String> selectedAnswers.

**4. DatabaseHelper**

* **Mối quan hệ với TestActivity**:
  + Cung cấp các phương thức để lưu trữ câu trả lời sai (insertIncorrectAnswer) và truy xuất trạng thái bài thi (getExamStatus).
  + Quản lý các bảng dữ liệu như incorrect\_answers và exam\_results.

**5. Question**

* **Mối quan hệ**:
  + Mô tả cấu trúc và dữ liệu của một câu hỏi trong quiz, bao gồm các thuộc tính như questionText, options, và image.

*Sơ đồ quan hệ:*

MainActivity

├─> SignDetailActivity

├─> TipsActivity

├─> QuestionActivity

└─> TestActivity

├─> QuestionAdapter

│ └─> Question

└─> DatabaseHelper

### 2.2.3. Thiết kế cơ sở dữ liệu:

Các bảng dữ liệu chính trong cơ sở dữ liệu SQLite của ứng dụng bao gồm:

**1. Bảng TABLE\_INCORRECT\_ANSWERS**

* **Tên bảng**: incorrect\_answers
* **Mục đích**: Lưu trữ các câu hỏi mà người dùng đã trả lời sai.
* **Cấu trúc cột**:
  + ID: Khóa chính (Primary Key) của bảng, kiểu **INTEGER**, tự động tăng (AUTOINCREMENT).
  + QUESTION\_ID: Lưu trữ ID của câu hỏi mà người dùng trả lời sai, kiểu **INTEGER**.
* **Chức năng liên quan**:
  + **insertIncorrectAnswer(int questionId)**: Thêm ID của câu hỏi sai vào bảng.
  + **deleteCorrectAnswer(int questionId)**: Xóa một câu trả lời khỏi bảng nếu người dùng đã trả lời đúng câu hỏi đó sau khi làm lại.
  + **getIncorrectAnswers()**: Lấy danh sách các câu hỏi sai từ bảng này.

Bảng này giúp lưu lại các câu hỏi mà người dùng làm sai để có thể đánh giá lại hoặc ôn tập sau.

**2. Bảng TABLE\_EXAM\_RESULTS**

* **Tên bảng**: exam\_results
* **Mục đích**: Theo dõi trạng thái của mỗi bài thi mà người dùng đã làm, bao gồm việc họ đã vượt qua (đậu), thất bại (rớt), hoặc chưa làm.
* **Cấu trúc cột**:
  + ID: Khóa chính (Primary Key) của bảng, kiểu **INTEGER**, tự động tăng (AUTOINCREMENT).
  + EXAM\_INDEX: Chỉ số bài thi, kiểu **INTEGER** và phải là duy nhất (UNIQUE). Được dùng để định danh từng bài thi.
  + EXAM\_STATUS: Trạng thái của bài thi, kiểu **INTEGER**, với các giá trị có ý nghĩa như sau:
    - 0: Mặc định là chưa làm bài thi.
    - 1: Bài thi đã đậu (người dùng đạt đủ yêu cầu).
    - -1: Bài thi không đạt (người dùng rớt).
* **Chức năng liên quan**:
  + **insertOrUpdateExamResult(int examIndex, int examStatus)**: Cập nhật trạng thái của bài thi bằng cách thêm mới hoặc thay thế (nhờ vào SQLiteDatabase.CONFLICT\_REPLACE).
  + **getExamStatus(int examIndex)**: Lấy trạng thái của bài thi theo examIndex để xác định bài thi đó đã đậu, rớt hay chưa làm.
  + **resetData()**: Đặt lại dữ liệu về trạng thái ban đầu. Nó xóa toàn bộ dữ liệu trong cả hai bảng và đặt lại tất cả các bài thi về trạng thái chưa làm (EXAM\_STATUS = 0).

Bảng này hỗ trợ quản lý tiến trình học và làm bài của người dùng, cho phép họ xem lại hoặc ôn tập các bài thi đã làm và đánh giá hiệu suất của mình.

### 2.2.4. Thiết kế giao diện:

**Link figma:** https://www.figma.com/design/LtfRmDwFw5WTiyThdvc1gG/App-Driver-Exams?node-id=0-1&node-type=canvas&t=UZPFAmEWFxFtrl7A-0

CHƯƠNG 3. KẾT QUẢ THỰC HIỆN

## 3.1. Công nghệ đã sử dụng

* **Ngôn ngữ lập trình**: Java
* **Công cụ**: Android Studio
* **Cơ sở dữ liệu**: SQLite
* **Thư viện**:
  + RecyclerView: sử dụng để hiển thị danh sách câu hỏi trong một giao diện người dùng cuộn.
  + Glide: sử dụng để tải và hiển thị hình ảnh từ internet (nếu có).
  + CountDownTimer: để quản lý thời gian làm bài.

## 3.2. Tiến độ thực hiện

Link GitHub tới dự án: https://github.com/truonglongtran/CSE441\_APP\_DRIVER\_EXAMS

Hướng dẫn các bước đã thực hiện:

**B1. Tạo dự án mới:**

* Mở Android Studio.
* Chọn "New Project".
* Chọn "Empty Activity" làm mẫu dự án.
* Nhập tên dự án (“GPLXB2”).
* Chọn ngôn ngữ là Java.
* Nhấn "Finish" để tạo dự án.

**B2. Tạo các package:**

* Trong cửa sổ "Project", click chuột phải vào thư mục "java".
* Chọn "New" -> "Package".
* Tạo các package sau:
  + model: chứa các lớp Question , Sign, Options, DriverType.
  + view: chứa lớp DiverTypeAdapter, SignAdapter, QuestionAdapter và giao diện người dùng.
  + controller: chứa lớp DatabaseHelper để quản lý cơ sở dữ liệu, và các Activity để xử lí logic.

**B3. Tạo các lớp:**

* Trong mỗi package, click chuột phải và chọn "New" -> "Java Class" để tạo các lớp tương ứng.
* Cài đặt các thuộc tính và phương thức cho từng lớp dựa trên thiết kế đã phân tích.

**B4. Cài đặt cơ sở dữ liệu SQLite:**

* Tạo lớp DatabaseHelper trong package controller.
* Định nghĩa cấu trúc cơ sở dữ liệu cho bảng incorrect\_answers và exam\_results.
* Thực hiện các phương thức để thêm, xóa, và truy xuất dữ liệu.

**B5. Viết code:**

* Bắt đầu viết code cho từng lớp, thực hiện các chức năng của ứng dụng:
  + Question: cài đặt các thuộc tính và phương thức cơ bản cho câu hỏi.
  + TestActivity: cài đặt các phương thức để quản lý giao diện, timer và logic làm bài.
  + QuestionAdapter: cài đặt phương thức để liên kết dữ liệu với RecyclerView.
  + DatabaseHelper: Khai báo và khởi tạo bảng, tương tác với cơ sở dữ liệu SQLite

**B6. Tải câu hỏi từ JSON:**

* Thêm tài liệu JSON chứa các câu hỏi vào thư mục assets.
* Viết các phương thức trong TestActivity để tải và phân tích cú pháp dữ liệu từ file JSON.

**B7. Chạy và kiểm thử:**

* Kết nối thiết bị Android hoặc sử dụng trình giả lập. Nhấn nút "Run" trong Android Studio để biên dịch và chạy ứng dụng.
* Kiểm tra các chức năng của ứng dụng, sửa lỗi và hoàn thiện code.

**B8. Triển khai (tùy chọn):**

* Nếu muốn đóng gói ứng dụng thành file APK để chia sẻ và cài đặt trên các thiết bị khác, bạn có thể sử dụng chức năng "Build APK" của Android Studio.

## 3.3 Hình ảnh sản phẩm

CHƯƠNG 4: MÃ NGUỒN ĐẦY ĐỦ

* + **DatabaseHelper.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.ContentValues; import android.content.Context; import android.database.Cursor; import android.database.sqlite.SQLiteDatabase; import android.database.sqlite.SQLiteOpenHelper;  import java.util.ArrayList;  public class DatabaseHelper extends SQLiteOpenHelper {  private static final String *DATABASE\_NAME* = "quiz.db";  private static final int *DATABASE\_VERSION* = 2;   // Bảng cho câu trả lời sai  private static final String *TABLE\_INCORRECT\_ANSWERS* = "incorrect\_answers";  private static final String *COL\_1* = "ID";  private static final String *COL\_2* = "QUESTION\_ID";   // Bảng cho trạng thái bài thi  private static final String *TABLE\_EXAM\_RESULTS* = "exam\_results";  private static final String *COL\_EXAM\_INDEX* = "EXAM\_INDEX";  private static final String *COL\_EXAM\_STATUS* = "EXAM\_STATUS";   public DatabaseHelper(Context context) {  super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  }   @Override  public void onCreate(SQLiteDatabase db) {  db.execSQL("CREATE TABLE " + *TABLE\_INCORRECT\_ANSWERS* + " ("  + *COL\_1* + " INTEGER PRIMARY KEY AUTOINCREMENT, "  + *COL\_2* + " INTEGER)");   db.execSQL("CREATE TABLE " + *TABLE\_EXAM\_RESULTS* + " ("  + *COL\_1* + " INTEGER PRIMARY KEY AUTOINCREMENT, "  + *COL\_EXAM\_INDEX* + " INTEGER UNIQUE, "  + *COL\_EXAM\_STATUS* + " INTEGER)");   for (int i = 0; i <= 17; i++) {  ContentValues contentValues = new ContentValues();  contentValues.put(*COL\_EXAM\_INDEX*, i);  contentValues.put(*COL\_EXAM\_STATUS*, 0); // Mặc định là 0  db.insert(*TABLE\_EXAM\_RESULTS*, null, contentValues);  }  }   @Override  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_INCORRECT\_ANSWERS*);  db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_EXAM\_RESULTS*);  onCreate(db); // Tạo lại bảng mới  }   public void insertIncorrectAnswer(int questionId) {  SQLiteDatabase db = this.getWritableDatabase();  ContentValues contentValues = new ContentValues();  contentValues.put(*COL\_2*, questionId);  db.insert(*TABLE\_INCORRECT\_ANSWERS*, null, contentValues);  db.close();  }   public void deleteCorrectAnswer(int questionId) {  SQLiteDatabase db = this.getWritableDatabase();  db.delete(*TABLE\_INCORRECT\_ANSWERS*, *COL\_2* + " = ?", new String[]{String.*valueOf*(questionId)});  db.close();  }   public ArrayList<Integer> getIncorrectAnswers() {  ArrayList<Integer> incorrectAnswers = new ArrayList<>();  SQLiteDatabase db = this.getReadableDatabase();  Cursor cursor = db.rawQuery("SELECT \* FROM " + *TABLE\_INCORRECT\_ANSWERS*, null);  if (cursor.moveToFirst()) {  do {  incorrectAnswers.add(cursor.getInt(1)); // Lấy ID câu hỏi  } while (cursor.moveToNext());  }  cursor.close();  db.close();  return incorrectAnswers;  }   public void insertOrUpdateExamResult(int examIndex, int examStatus) {  SQLiteDatabase db = this.getWritableDatabase();  ContentValues contentValues = new ContentValues();  contentValues.put(*COL\_EXAM\_INDEX*, examIndex);  contentValues.put(*COL\_EXAM\_STATUS*, examStatus);  db.insertWithOnConflict(*TABLE\_EXAM\_RESULTS*, null, contentValues, SQLiteDatabase.*CONFLICT\_REPLACE*);  db.close();  }  public Integer getExamStatus(int examIndex) {  SQLiteDatabase db = this.getReadableDatabase();  Cursor cursor = db.query(*TABLE\_EXAM\_RESULTS*,  new String[]{*COL\_EXAM\_STATUS*},  *COL\_EXAM\_INDEX* + "=?",  new String[]{String.*valueOf*(examIndex)},  null, null, null);   Integer examStatus = null;  if (cursor.moveToFirst()) {  examStatus = cursor.getInt(0);  }  cursor.close();  db.close();  return examStatus;  }  public void resetData() {  SQLiteDatabase db = this.getWritableDatabase();  db.delete(*TABLE\_INCORRECT\_ANSWERS*, null, null);  db.delete(*TABLE\_EXAM\_RESULTS*, null, null);   for (int i = 0; i <= 17; i++) {  ContentValues contentValues = new ContentValues();  contentValues.put(*COL\_EXAM\_INDEX*, i);  contentValues.put(*COL\_EXAM\_STATUS*, 0); // Mặc định là 0  db.insert(*TABLE\_EXAM\_RESULTS*, null, contentValues);  }  db.close();  } } |

* + **DriverType.java**

|  |
| --- |
| * package com.example.gplxb2;  public class DriverType {  private int id;  private String title;  private int quantity;  private String description;   public DriverType(int id, String title, int quantity, String description) {  this.id = id;  this.title = title;  this.quantity = quantity;  this.description = description;  }   public int getId() {  return id;  }   public String getTitle() {  return title;  }   public int getQuantity() {  return quantity;  }   public String getDescription() {  return description;  } } |

* + **Options.java**

|  |
| --- |
| * package com.example.gplxb2;  public class Options {  private String A;  private String B;  private String C;  private String D;  private String E;   public Options(String A, String B, String C, String D, String E) {  this.A = A;  this.B = B;  this.C = C;  this.D = D;  this.E = E;  }   public String getA() {  return A;  }   public String getB() {  return B;  }   public String getC() {  return C;  }   public String getD() {  return D;  }   public String getE() {  return E;  } } |

* + **Question.java**

|  |
| --- |
| * package com.example.gplxb2;  public class Question {  private String id;  private String questionText;  private Options options;  private String answer;  private String suggest;  private Image image;   public Question(String id, String questionText, Options options, String answer, String suggest, Image image) {  this.id = id;  this.questionText = questionText;  this.options = options;  this.answer = answer;  this.suggest = suggest;  this.image = image;  }   public String getId() {  return id;  }   public String getQuestionText() {  return questionText;  }   public Options getOptions() {  return options;  }   public String getAnswer() {  return answer;  }   public String getSuggest() {  return suggest;  }   public Image getImage() {  return image;  }   // Lớp Image lồng trong lớp Question  public static class Image {  private String img1;   public Image(String img1) {  this.img1 = img1;  }   public String getImg1() {  return img1;  }  } } |

* + **Sign.java**

|  |
| --- |
| * package com.example.gplxb2;  public class Sign {  private String name;  private String des;  private String imagePath;  private int titleIndex; // Thêm thuộc tính titleIndex   // Constructor cho các loại đối tượng khác nhau  public Sign(String name, String des, String imagePath) {  this.name = name;  this.des = des;  this.imagePath = imagePath;  this.titleIndex = -1; // Giá trị mặc định nếu không phải là tiêu đề  }   // Constructor cho tiêu đề  public Sign(String name, String des, String imagePath, int titleIndex) {  this.name = name;  this.des = des;  this.imagePath = imagePath;  this.titleIndex = titleIndex; // Gán chỉ số tiêu đề  }   public String getName() {  return name;  }   public String getDes() {  return des;  }   public String getImagePath() {  return (imagePath != null && !imagePath.isEmpty()) ? imagePath : "bienbao.png"; // Hình ảnh mặc định  }   public int getTitleIndex() {  return titleIndex; // Thêm phương thức getter cho titleIndex  } } |

* + **DriverTypeActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.ImageButton; import android.widget.Toast; import androidx.annotation.Nullable; import androidx.appcompat.app.AppCompatActivity; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView; import org.json.JSONArray; import org.json.JSONObject; import java.io.InputStream; import java.nio.charset.StandardCharsets; import java.util.ArrayList; import java.util.List;  public class DriverTypeActivity extends AppCompatActivity {  private RecyclerView recyclerView;  private DriverTypeAdapter adapter;  private List<DriverType> driverTypeList;  private ImageButton backButton; // Nút Back   @Override  protected void onCreate(@Nullable Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_driver\_type*);   recyclerView = findViewById(R.id.*recyclerView*);  backButton = findViewById(R.id.*back\_button*); // Tham chiếu đến nút Back  driverTypeList = new ArrayList<>();   loadDriverTypes();   adapter = new DriverTypeAdapter(this, driverTypeList);  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setAdapter(adapter);   // Thiết lập sự kiện cho nút Back  backButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(DriverTypeActivity.this, MainActivity.class);  startActivity(intent);  finish();  }  });  }   private void loadDriverTypes() {  try {  InputStream is = getAssets().open("driver\_type.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray typesArray = jsonObject.getJSONArray("types");   for (int i = 0; i < typesArray.length(); i++) {  JSONObject typeObject = typesArray.getJSONObject(i);   int id = typeObject.getInt("id");  String title = typeObject.getString("title");  int quantity = typeObject.getInt("quantity");  String description = typeObject.getString("description");   DriverType driverType = new DriverType(id, title, quantity, description);  driverTypeList.add(driverType);  }  } catch (Exception e) {  e.printStackTrace();  Toast.*makeText*(this, "Lỗi khi tải dữ liệu", Toast.*LENGTH\_SHORT*).show();  }  } } |

* + **ExamActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.graphics.drawable.Drawable; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.GridLayout; import android.widget.ImageButton; import androidx.activity.EdgeToEdge; import androidx.appcompat.app.AppCompatActivity; import androidx.core.content.ContextCompat;  public class ExamsActivity extends AppCompatActivity {   private DatabaseHelper databaseHelper; // Declare DatabaseHelper   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  EdgeToEdge.*enable*(this);  setContentView(R.layout.*activity\_exams*);   databaseHelper = new DatabaseHelper(this); // Initialize DatabaseHelper   ImageButton backButton = findViewById(R.id.*back\_button*);   backButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(ExamsActivity.this, MainActivity.class);  startActivity(intent);  finish();  }  });   GridLayout buttonGrid = findViewById(R.id.*button\_grid*);   for (int i = 0; i < buttonGrid.getChildCount(); i++) {  final int examIndex = i; // Store button index  View child = buttonGrid.getChildAt(i);  if (child instanceof Button) {  Button examButton = (Button) child;   int examStatus = databaseHelper.getExamStatus(examIndex);   switch (examStatus) {  case 1:  examButton.setBackgroundTintList(ContextCompat.*getColorStateList*(this, R.color.*green*));  examButton.setTextColor(ContextCompat.*getColor*(ExamsActivity.this, android.R.color.*white*));  break;  case -1:  examButton.setBackgroundTintList(ContextCompat.*getColorStateList*(this, R.color.*red*));  examButton.setTextColor(ContextCompat.*getColor*(ExamsActivity.this, android.R.color.*white*));  break;  default:  examButton.setBackgroundTintList(ContextCompat.*getColorStateList*(this, R.color.*MyBlue*));  examButton.setTextColor(ContextCompat.*getColor*(ExamsActivity.this, android.R.color.*black*));  break;  }   examButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Navigate to TestActivity and pass the exam index  Intent intent = new Intent(ExamsActivity.this, TestActivity.class);  intent.putExtra("examsIndex", examIndex); // Pass examsIndex to TestActivity  startActivity(intent);  }  });  }  }  } } |

* + **ResultActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.graphics.Color; import android.os.Bundle; import android.widget.Button; import android.widget.TextView;  import androidx.appcompat.app.AppCompatActivity; import androidx.appcompat.widget.Toolbar;  public class ResultActivity extends AppCompatActivity {   private TextView resultText;  private TextView totalQuestionsText;  private TextView resultMessage;  private Button backButton;  private Toolbar toolbar;  private TextView toolbarTitle; // TextView cho tiêu đề trên toolbar  private DatabaseHelper databaseHelper; // Thêm biến cho DatabaseHelper   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_result*);   // Initialize views  toolbar = findViewById(R.id.*toolbar*);  toolbarTitle = findViewById(R.id.*toolbar\_title*); // Khởi tạo TextView tiêu đề  resultText = findViewById(R.id.*result\_text*);  totalQuestionsText = findViewById(R.id.*total\_questions\_text*);  resultMessage = findViewById(R.id.*result\_message*);  backButton = findViewById(R.id.*back\_btn*);   // Initialize DatabaseHelper  databaseHelper = new DatabaseHelper(this);   // Receive data from Intent  Intent intent = getIntent();  int score = intent.getIntExtra("SCORE", 0);  int totalQuestions = intent.getIntExtra("TOTAL\_QUESTIONS", 0);  int incorrectCriticalCount = intent.getIntExtra("INCORRECT\_CRITICAL\_COUNT", 0);  int examIndex = getIntent().getIntExtra("EXAMS\_INDEX", -1);  String examType = intent.getStringExtra("EXAM\_TYPE"); // Nhận loại bộ đề  String examTitle = intent.getStringExtra("TITLE"); // Nhận tiêu đề bộ đề   // Set toolbar title  if (examTitle != null) {  toolbarTitle.setText(examTitle); // Thiết lập tiêu đề lên toolbar  } else {  toolbarTitle.setText("Kết quả thi"); // Tiêu đề mặc định nếu không có  }   // Display the score and total questions  resultText.setText("Your Score: " + score);  totalQuestionsText.setText("Total Questions: " + totalQuestions);   // Determine the result message and update the database status  if (incorrectCriticalCount > 0) {  resultMessage.setText("Bạn đã thi trượt vì làm sai câu điểm liệt");  resultMessage.setTextColor(Color.*RED*);  databaseHelper.insertOrUpdateExamResult(examIndex, -1); // Update status to -1 (fail)  } else if (score < 32) {  resultMessage.setText("Bạn đã thi trượt do không đủ số câu đúng tối thiểu");  resultMessage.setTextColor(Color.*RED*);  databaseHelper.insertOrUpdateExamResult(examIndex, -1); // Update status to -1 (fail)  } else {  resultMessage.setText("Chúc mừng bạn đã thi đạt");  resultMessage.setTextColor(Color.*GREEN*);  databaseHelper.insertOrUpdateExamResult(examIndex, 1); // Update status to 1 (pass)  }   // Set up the back button event  backButton.setText("Quay về trang chủ");  backButton.setBackgroundColor(Color.*parseColor*("#329BF1")); // Đặt màu cho nút  backButton.setOnClickListener(v -> goToMain());  }   // Method to return to the main page  public void goToMain() {  Intent intent = new Intent(this, MainActivity.class);  intent.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TOP* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  startActivity(intent);  finish();  } } |

* + **SignActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.os.Bundle; import android.util.Log; import android.view.View; import android.widget.Button; import android.widget.ImageButton; import android.widget.Toast; import androidx.appcompat.app.AppCompatActivity; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView; import org.json.JSONArray; import org.json.JSONObject; import java.io.InputStream; import java.nio.charset.StandardCharsets; import java.util.ArrayList; import java.util.List;  public class SignActivity extends AppCompatActivity {  private RecyclerView recyclerView;  private SignAdapter adapter;  private List<Sign> signList;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_sign*);   recyclerView = findViewById(R.id.*recyclerView*);  signList = new ArrayList<>();   loadSignData();   adapter = new SignAdapter(this, signList);  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setAdapter(adapter);   // Thiết lập sự kiện cho nút "Quay lại"  ImageButton backButton = findViewById(R.id.*back\_button*);  backButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  finish(); // Kết thúc activity hiện tại  }  });  }   private void loadSignData() {  try {  InputStream is = getAssets().open("BienBao.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray names = jsonObject.names(); // Lấy danh sách các khóa   if (names != null) {  for (int i = 0; i < names.length(); i++) {  String key = names.getString(i); // Lấy khóa  JSONArray signArray = jsonObject.getJSONArray(key);   // Thêm một tiêu đề cho mỗi mảng  signList.add(new Sign(key, "", "", i)); // Sử dụng tên mảng làm tiêu đề và truyền chỉ số   for (int j = 0; j < signArray.length(); j++) {  JSONObject signObject = signArray.getJSONObject(j);  String name = signObject.getString("name");  String des = signObject.optString("des", "");   String imagePath;  if (signObject.has("image") && !signObject.isNull("image")) {  imagePath = signObject.getString("image");  } else {  imagePath = "bienbao.png"; // Hình ảnh mặc định  }   Sign sign = new Sign(name, des, imagePath);  signList.add(sign);  }  }  }   } catch (Exception e) {  e.printStackTrace();  Toast.*makeText*(this, "Lỗi khi tải dữ liệu: " + e.getMessage(), Toast.*LENGTH\_SHORT*).show();  }  } } |

* + **SignDetailActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.os.Bundle; import android.view.View; import android.widget.ImageView; import android.widget.TextView; import androidx.appcompat.app.AppCompatActivity; import com.bumptech.glide.Glide;  public class SignDetailActivity extends AppCompatActivity {  private TextView nameTextView;  private TextView descriptionTextView;  private ImageView imageView;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_sign\_detail*);   nameTextView = findViewById(R.id.*text\_name*);  descriptionTextView = findViewById(R.id.*text\_description*);  imageView = findViewById(R.id.*image\_view*);   // Lấy dữ liệu từ intent  Intent intent = getIntent();  String name = intent.getStringExtra("name");  String description = intent.getStringExtra("description");  String imagePath = intent.getStringExtra("imagePath");   // Hiển thị dữ liệu  nameTextView.setText(name);  descriptionTextView.setText(description);  Glide.*with*(this)  .load(imagePath)  .placeholder(R.drawable.*bienbao*) // Hình ảnh mặc định khi đang tải  .error(R.drawable.*bienbao*) // Hình ảnh khi có lỗi tải  .into(imageView);   // Thiết lập nút Back  findViewById(R.id.*back\_button*).setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  finish(); // Kết thúc activity hiện tại  }  });   // Cập nhật tiêu đề cho Toolbar  TextView toolbarTitle = findViewById(R.id.*txt\_detail\_sign*);  toolbarTitle.setText(name); // Cập nhật tên biển báo vào TextView  } } |

* + **TipsActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.ImageButton;  import androidx.appcompat.app.AppCompatActivity;  public class TipsActivity extends AppCompatActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_tips*);   // Nút Back  ImageButton backButton = findViewById(R.id.*back\_button*);  backButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Điều hướng về MainActivity  Intent intent = new Intent(TipsActivity.this, MainActivity.class);  startActivity(intent);  finish();  }  });  } } |

* + **TestActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.DialogInterface; import android.content.Intent; import android.os.Bundle; import android.os.CountDownTimer; import android.util.Log; import android.widget.Button; import android.widget.ImageButton; import android.widget.TextView;  import androidx.appcompat.app.AlertDialog; import androidx.appcompat.app.AppCompatActivity; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView;  import org.json.JSONArray; import org.json.JSONObject;  import java.io.InputStream; import java.nio.charset.StandardCharsets; import java.util.ArrayList; import java.util.Arrays; import java.util.Collections; import java.util.List; import java.util.Map; import java.util.stream.Collectors;  public class TestActivity extends AppCompatActivity {   private TextView timerTextView, questionCounterTextView;  private Button submitButton;  private ImageButton nextButton, backButton;  private RecyclerView questionRecyclerView;  private LinearLayoutManager layoutManager;  private CountDownTimer countDownTimer;  private int currentPosition = 0;  private int totalQuestions = 0;  private List<Question> questions; // List of questions  private List<String> criticals; // List of critical questions  private QuestionAdapter adapter; // Adapter for RecyclerView  private ArrayList<Integer> incorrectAnswers;  private DatabaseHelper databaseHelper;  private String title = ""; // Biến để lưu tiêu đề   // Exam index  private int examIndex = -1; // Mặc định -1 để random   // Array to hold predefined question sets for different exams  int[][] arrays = {  {10031, 10089, 10114, 10121, 10134, 10151, 10158, 10160, 10163, 10186, 10193, 10254, 10258, 10260, 10294, 10342, 10343, 10348, 10366, 10393, 10415, 10423, 10466, 10479, 10482, 10489, 10498, 10508, 10516, 10521, 10528, 10532, 10541, 10542, 10558},  {10001, 10002, 10061, 10065, 10128, 10144, 10150, 10154, 10156, 10189, 10208, 10210, 10223, 10242, 10271, 10308, 10324, 10326, 10343, 10350, 10361, 10375, 10378, 10400, 10429, 10503, 10504, 10517, 10522, 10544, 10553, 10563, 10567, 10592, 10593},  {10019, 10030, 10112, 10113, 10117, 10129, 10140, 10143, 10158, 10162, 10180, 10208, 10217, 10262, 10273, 10361, 10388, 10404, 10420, 10449, 10454, 10464, 10465, 10482, 10485, 10489, 10498, 10503, 10525, 10532, 10537, 10554, 10555, 10575, 10581},  {10043, 10060, 10079, 10107, 10130, 10143, 10153, 10155, 10161, 10162, 10168, 10199, 10240, 10247, 10290, 10306, 10345, 10350, 10358, 10373, 10395, 10413, 10431, 10462, 10470, 10493, 10494, 10535, 10537, 10545, 10553, 10556, 10574, 10576, 10590},  {10019, 10063, 10068, 10084, 10089, 10133, 10144, 10147, 10148, 10153, 10192, 10209, 10232, 10244, 10278, 10316, 10350, 10383, 10392, 10412, 10417, 10427, 10431, 10467, 10484, 10495, 10532, 10533, 10542, 10543, 10550, 10572, 10576, 10577, 10593},  {10030, 10068, 10097, 10116, 10126, 10149, 10153, 10158, 10164, 10172, 10193, 10224, 10248, 10250, 10294, 10339, 10357, 10392, 10393, 10412, 10425, 10431, 10441, 10463, 10465, 10501, 10506, 10514, 10519, 10552, 10554, 10562, 10565, 10568, 10596},  {10054, 10066, 10072, 10077, 10127, 10141, 10142, 10148, 10150, 10152, 10180, 10212, 10238, 10256, 10304, 10323, 10358, 10379, 10392, 10400, 10405, 10411, 10449, 10454, 10476, 10505, 10516, 10524, 10529, 10536, 10573, 10578, 10580, 10582, 10595},  {10047, 10058, 10080, 10109, 10112, 10130, 10140, 10141, 10143, 10147, 10175, 10200, 10220, 10221, 10299, 10329, 10342, 10397, 10404, 10407, 10413, 10423, 10428, 10470, 10476, 10488, 10492, 10499, 10506, 10514, 10531, 10537, 10540, 10575, 10577},  {10009, 10011, 10093, 10114, 10116, 10130, 10142, 10146, 10150, 10166, 10170, 10199, 10256, 10258, 10291, 10315, 10340, 10343, 10346, 10368, 10397, 10404, 10427, 10440, 10480, 10488, 10494, 10512, 10531, 10539, 10541, 10543, 10553, 10557, 10595},  {10044, 10062, 10066, 10114, 10119, 10134, 10144, 10151, 10154, 10161, 10188, 10200, 10220, 10250, 10295, 10313, 10323, 10338, 10364, 10373, 10421, 10431, 10435, 10445, 10482, 10487, 10506, 10507, 10535, 10536, 10547, 10563, 10566, 10576, 10577},  {10028, 10056, 10073, 10107, 10126, 10140, 10153, 10155, 10156, 10166, 10187, 10199, 10221, 10238, 10288, 10314, 10318, 10323, 10360, 10379, 10401, 10450, 10452, 10461, 10473, 10491, 10494, 10507, 10518, 10523, 10524, 10537, 10546, 10586, 10596},  {10019, 10046, 10066, 10086, 10100, 10131, 10151, 10153, 10156, 10161, 10184, 10205, 10246, 10251, 10275, 10345, 10362, 10389, 10398, 10435, 10441, 10445, 10446, 10453, 10467, 10488, 10513, 10536, 10539, 10560, 10573, 10586, 10587, 10595, 10600},  {10001, 10010, 10021, 10028, 10043, 10131, 10140, 10150, 10151, 10162, 10189, 10196, 10251, 10263, 10273, 10319, 10325, 10332, 10368, 10388, 10408, 10410, 10440, 10451, 10465, 10495, 10501, 10535, 10536, 10539, 10547, 10561, 10562, 10587, 10589},  {10003, 10032, 10051, 10079, 10109, 10126, 10141, 10145, 10156, 10159, 10186, 10202, 10229, 10266, 10290, 10324, 10335, 10345, 10371, 10384, 10390, 10434, 10444, 10456, 10457, 10506, 10511, 10512, 10522, 10542, 10558, 10560, 10574, 10580, 10591},  {10033, 10041, 10056, 10057, 10138, 10144, 10146, 10156, 10161, 10188, 10201, 10215, 10227, 10241, 10274, 10309, 10311, 10326, 10360, 10366, 10372, 10377, 10471, 10473, 10482, 10504, 10519, 10521, 10532, 10533, 10537, 10556, 10564, 10582, 10591},  {10004, 10037, 10045, 10114, 10121, 10129, 10151, 10156, 10163, 10165, 10175, 10209, 10236, 10245, 10280, 10345, 10348, 10358, 10368, 10396, 10421, 10427, 10431, 10466, 10472, 10499, 10517, 10542, 10545, 10566, 10574, 10578, 10585, 10597, 10600},  {10052, 10067, 10078, 10080, 10081, 10137, 10144, 10149, 10153, 10161, 10190, 10211, 10254, 10264, 10302, 10352, 10404, 10406, 10411, 10437, 10439, 10462, 10476, 10480, 10482, 10511, 10515, 10516, 10520, 10551, 10562, 10564, 10567, 10582, 10587},  {10019, 10031, 10054, 10111, 10132, 10141, 10152, 10153, 10161, 10165, 10176, 10209, 10224, 10236, 10274, 10349, 10351, 10415, 10416, 10418, 10452, 10470, 10473, 10476, 10485, 10494, 10499, 10507, 10519, 10566, 10571, 10578, 10590, 10594, 10600} };   private int[] khainiemQuytac ={10001, 10002, 10003, 10004, 10005, 10006, 10007, 10008, 10009, 10010, 10011, 10012, 10013, 10014, 10015, 10016, 10017, 10018, 10019, 10020, 10021, 10022, 10023, 10024, 10025, 10026, 10027, 10028, 10029, 10030, 10031, 10032, 10033, 10034, 10035, 10036, 10037, 10038, 10039, 10040, 10041, 10042, 10043, 10044, 10045, 10046, 10047, 10048, 10049, 10050, 10051, 10052, 10053, 10054, 10055, 10056, 10057, 10058, 10059, 10060, 10061, 10062, 10063, 10064, 10065, 10066, 10067, 10068, 10069, 10070, 10071, 10072, 10073, 10074, 10075, 10076, 10077, 10078, 10079, 10080, 10081, 10082, 10083, 10084, 10085, 10086, 10087, 10088, 10089, 10090, 10091, 10092, 10093, 10094, 10095, 10096, 10097, 10098, 10099, 10100, 10101, 10102, 10103, 10104, 10105, 10106, 10107, 10108, 10109, 10110, 10111, 10112, 10113, 10114, 10115, 10116, 10117, 10118, 10119, 10120, 10121, 10122, 10123, 10124, 10125, 10126, 10127, 10128, 10129, 10130, 10131, 10132, 10133, 10134, 10135, 10136, 10137, 10138, 10139, 10140, 10141, 10142, 10143, 10144, 10145, 10146, 10147, 10148, 10149, 10150, 10151, 10152, 10153, 10154, 10155, 10156, 10157, 10158, 10159, 10160, 10161, 10162, 10163, 10164, 10165, 10166};  private int[] nghiepvuVantai = {10167, 10168, 10169, 10170, 10171, 10172, 10173, 10174, 10175, 10176, 10177, 10178, 10179, 10180, 10181, 10182, 10183, 10184, 10185, 10186, 10187, 10188, 10189, 10190, 10191, 10192};  private int[] vanhoaGiaothong = {10193, 10194, 10195, 10196, 10197, 10198, 10199, 10200, 10201, 10202, 10203, 10204, 10205, 10206, 10207, 10208, 10209, 10210, 10211, 10212, 10213};  private int[] kithuatLaixe = {10214, 10215, 10216, 10217, 10218, 10219, 10220, 10221, 10222, 10223, 10224, 10225, 10226, 10227, 10228, 10229, 10230, 10231, 10232, 10233, 10234, 10235, 10236, 10237, 10238, 10239, 10240, 10241, 10242, 10243, 10244, 10245, 10246, 10247, 10248, 10249, 10250, 10251, 10252, 10253, 10254, 10255, 10256, 10257, 10258, 10259, 10260, 10261, 10262, 10263, 10264, 10265, 10266, 10267, 10268, 10269};  private int[] cautaoSuachua = {10270, 10271, 10272, 10273, 10274, 10275, 10276, 10277, 10278, 10279, 10280, 10281, 10282, 10283, 10284, 10285, 10286, 10287, 10288, 10289, 10290, 10291, 10292, 10293, 10294, 10295, 10296, 10297, 10298, 10299, 10300, 10301, 10302, 10303, 10304};  private int[] sahinh = {10305, 10306, 10307, 10308, 10309, 10310, 10311, 10312, 10313, 10314, 10315, 10316, 10317, 10318, 10319, 10320, 10321, 10322, 10323, 10324, 10325, 10326, 10327, 10328, 10329, 10330, 10331, 10332, 10333, 10334, 10335, 10336, 10337, 10338, 10339, 10340, 10341, 10342, 10343, 10344, 10345, 10346, 10347, 10348, 10349, 10350, 10351, 10352, 10353, 10354, 10355, 10356, 10357, 10358, 10359, 10360, 10361, 10362, 10363, 10364, 10365, 10366, 10367, 10368, 10369, 10370, 10371, 10372, 10373, 10374, 10375, 10376, 10377, 10378, 10379, 10380, 10381, 10382, 10383, 10384, 10385, 10386, 10387, 10388, 10389, 10390, 10391, 10392, 10393, 10394, 10395, 10396, 10397, 10398, 10399, 10400, 10401, 10402, 10403, 10404, 10405, 10406, 10407, 10408, 10409, 10410, 10411, 10412, 10413, 10414, 10415, 10416, 10417, 10418, 10419, 10420, 10421, 10422, 10423, 10424, 10425, 10426, 10427, 10428, 10429, 10430, 10431, 10432, 10433, 10434, 10435, 10436, 10437, 10438, 10439, 10440, 10441, 10442, 10443, 10444, 10445, 10446, 10447, 10448, 10449, 10450, 10451, 10452, 10453, 10454, 10455, 10456, 10457, 10458, 10459, 10460, 10461, 10462, 10463, 10464, 10465, 10466, 10467, 10468, 10469, 10470, 10471, 10472, 10473, 10474, 10475, 10476, 10477, 10478, 10479, 10480, 10481, 10482, 10483, 10484, 10485, 10486};  private int[] bienbao = {10305, 10306, 10307, 10308, 10309, 10310, 10311, 10312, 10313, 10314, 10315, 10316, 10317, 10318, 10319, 10320, 10321, 10322, 10323, 10324, 10325, 10326, 10327, 10328, 10329, 10330, 10331, 10332, 10333, 10334, 10335, 10336, 10337, 10338, 10339, 10340, 10341, 10342, 10343, 10344, 10345, 10346, 10347, 10348, 10349, 10350, 10351, 10352, 10353, 10354, 10355, 10356, 10357, 10358, 10359, 10360, 10361, 10362, 10363, 10364, 10365, 10366, 10367, 10368, 10369, 10370, 10371, 10372, 10373, 10374, 10375, 10376, 10377, 10378, 10379, 10380, 10381, 10382, 10383, 10384, 10385, 10386, 10387, 10388, 10389, 10390, 10391, 10392, 10393, 10394, 10395, 10396, 10397, 10398, 10399, 10400, 10401, 10402, 10403, 10404, 10405, 10406, 10407, 10408, 10409, 10410, 10411, 10412, 10413, 10414, 10415, 10416, 10417, 10418, 10419, 10420, 10421, 10422, 10423, 10424, 10425, 10426, 10427, 10428, 10429, 10430, 10431, 10432, 10433, 10434, 10435, 10436, 10437, 10438, 10439, 10440, 10441, 10442, 10443, 10444, 10445, 10446, 10447, 10448, 10449, 10450, 10451, 10452, 10453, 10454, 10455, 10456, 10457, 10458, 10459, 10460, 10461, 10462, 10463, 10464, 10465, 10466, 10467, 10468, 10469, 10470, 10471, 10472, 10473, 10474, 10475, 10476, 10477, 10478, 10479, 10480, 10481, 10482, 10483, 10484, 10485, 10486};   private int[] arraystop50 = {10001, 10004, 10011, 10016, 10022, 10029, 10038, 10052, 10065, 10090,  10149, 10154, 10162, 10166, 10171, 10176, 10179, 10182, 10205, 10214,  10232, 10236, 10241, 10251, 10262, 10265, 10275, 10292, 10307, 10313,  10339, 10343, 10375, 10380, 10397, 10402, 10403, 10411, 10419, 10438,  10442, 10450, 10457, 10508, 10533, 10550, 10551, 10569, 10583, 10592};      @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_test*);   databaseHelper = new DatabaseHelper(this);  incorrectAnswers = databaseHelper.getIncorrectAnswers();  // Initialize views  timerTextView = findViewById(R.id.*timer*);  questionCounterTextView = findViewById(R.id.*question\_counter*);  submitButton = findViewById(R.id.*submit\_btn*);  nextButton = findViewById(R.id.*next\_btn*);  backButton = findViewById(R.id.*back\_btn*);  questionRecyclerView = findViewById(R.id.*question\_list*);  LinearLayoutManager layoutManager = new LinearLayoutManager(this, LinearLayoutManager.*HORIZONTAL*, false);  questionRecyclerView.setOnTouchListener((v, event) -> true);  questionRecyclerView.setLayoutManager(layoutManager);     // Get the examIndex from intent  examIndex = getIntent().getIntExtra("examsIndex", -1);  Log.*d*("TestActivity", "examsIndex: " + examIndex);   if (examIndex > 1000) {  if (examIndex == 2001) {  questions = loadCriticalQuestions();  title = "Các câu Điểm liệt"; // Gán tiêu đề cho câu hỏi quan trọng  } else {  questions = loadQuestionsByExamIndex(examIndex);  switch (examIndex) {  case 2002:  title = "Khái niệm quy tắc";  break;  case 2003:  title = "Nghiệp vụ vận tải";  break;  case 2004:  title = "Văn hóa giao thông";  break;  case 2005:  title = "Kỹ thuật lái xe";  break;  case 2006:  title = "Cấu tạo sửa chữa";  break;  case 2007:  title = "Biển báo";  break;  case 2008:  title = "Sa hình";  break;  default:  title = "Đề thi " + examIndex; // Đặt tiêu đề mặc định  break;  }  }  } else if (examIndex == 60) {  questions = loadCriticalQuestions(); // Load only critical questions  title = "Các câu Điểm liệt"; // Tiêu đề cho câu hỏi quan trọng  } else if (examIndex == 40) {  questions = loadIncorrectQuestions(); // Load only top 50 questions  title = "Các câu bị sai"; // Tiêu đề cho câu bị sai  } else if (examIndex == 50) {  questions = loadTop50(); // Load only top 50 questions  title = "Top các câu hay sai"; // Tiêu đề cho top câu hỏi  } else if (examIndex != -1) {  questions = loadQuestionsFromArray(arrays[examIndex]); // Load specific set of questions  int realExamIndex = examIndex + 1;  title = "Đề thi " + realExamIndex ; // Tiêu đề cho đề thi cụ thể  } else {  questions = loadQuestionsFromJson(); // Load random questions  title = "Đề ngẫu nhiên"; // Tiêu đề cho đề ngẫu nhiên  }   criticals = loadCriticalQuestionsFromJson();  totalQuestions = questions.size();   // Setup adapter and pass the callback for answer selection  adapter = new QuestionAdapter(questions, criticals); // Pass criticals to adapter  questionRecyclerView.setAdapter(adapter);   // Update question counter  updateQuestionCounter();   // Start timer  startTimer();   // Set Next button click listener  nextButton.setOnClickListener(v -> {  if (currentPosition < totalQuestions - 1) {  currentPosition++;  questionRecyclerView.smoothScrollToPosition(currentPosition);  updateQuestionCounter();  }  });   // Set Back button click listener  backButton.setOnClickListener(v -> {  if (currentPosition > 0) {  currentPosition--;  questionRecyclerView.smoothScrollToPosition(currentPosition);  updateQuestionCounter();  }  });   // Set Submit button click listener  submitButton.setOnClickListener(v -> showConfirmationDialog(title)); // Gửi tiêu đề khi nộp bài  }   // Update question counter  private void updateQuestionCounter() {  questionCounterTextView.setText((currentPosition + 1) + "/" + totalQuestions);  }   // Start the timer  private void startTimer() {  countDownTimer = new CountDownTimer(10 \* 60 \* 1000, 1000) { // 10 minutes  @Override  public void onTick(long millisUntilFinished) {  long minutes = millisUntilFinished / 60000;  long seconds = (millisUntilFinished % 60000) / 1000;  String timeLeft = String.*format*("%02d:%02d", minutes, seconds);  timerTextView.setText("Time: " + timeLeft);  }   @Override  public void onFinish() {  submitAnswers(title);  }  }.start();  }  // @Override // public void onBackPressed() { // // Gọi hàm submit khi nhấn nút quay lại // submitAnswers(); // }  @Override  protected void onUserLeaveHint() {  super.onUserLeaveHint();  // Gọi hàm submit khi người dùng rời khỏi Activity  submitAnswers(title);  }  //hộp thoại xác nhận trước khi nộp bài  private void showConfirmationDialog(final String title) {  AlertDialog.Builder builder = new AlertDialog.Builder(this);  builder.setTitle("Xác nhận nộp bài");  builder.setMessage("Bạn có chắc chắn muốn nộp bài?");   builder.setPositiveButton("Có", new DialogInterface.OnClickListener() {  @Override  public void onClick(DialogInterface dialog, int which) {  // Gọi hàm submitAnswers khi người dùng xác nhận  submitAnswers(title);  }  });   builder.setNegativeButton("Không", new DialogInterface.OnClickListener() {  @Override  public void onClick(DialogInterface dialog, int which) {  dialog.dismiss(); // Đóng hộp thoại nếu người dùng chọn không  }  });   AlertDialog dialog = builder.create();  dialog.show();  }    // Gửi các câu trả lời và tính điểm  private void submitAnswers(String title) {  int score = 0; // Khởi tạo điểm số  int incorrectCriticalCount = 0; // Số lượng câu trả lời sai cho các câu hỏi quan trọng  Map<String, String> selectedAnswers = adapter.getSelectedAnswers(); // Lấy các câu trả lời đã chọn từ adapter   // Lấy danh sách các câu hỏi từ cơ sở dữ liệu  ArrayList<Integer> incorrectAnswersFromDB = databaseHelper.getIncorrectAnswers();   // So sánh câu trả lời của người dùng với các câu trả lời đúng  for (Question question : questions) {  String userAnswer = selectedAnswers.get(question.getId()); // Lấy câu trả lời của người dùng   // Kiểm tra nếu câu hỏi là câu hỏi quan trọng  if (criticals.contains(question.getId())) {  // Nếu câu trả lời sai hoặc không được chọn  if (userAnswer == null || !userAnswer.equals(question.getAnswer())) {  incorrectCriticalCount++; // Tăng số lượng câu hỏi quan trọng trả lời sai  }  }   // Kiểm tra nếu câu trả lời sai  if (userAnswer == null || !userAnswer.equals(question.getAnswer())) {  // Chuyển đổi ID câu hỏi sang Integer  Integer questionId = Integer.*parseInt*(question.getId());   // Kiểm tra nếu ID chưa có trong danh sách  if (!incorrectAnswers.contains(questionId)) {  incorrectAnswers.add(questionId); // Thêm ID câu hỏi sai vào danh sách  databaseHelper.insertIncorrectAnswer(questionId); // Thêm vào cơ sở dữ liệu  }  } else {  // Nếu câu trả lời đúng, kiểm tra và xóa ID nếu có trong danh sách  Integer questionId = Integer.*parseInt*(question.getId());   if (incorrectAnswers.contains(questionId)) {  incorrectAnswers.remove(questionId); // Xóa ID câu hỏi đúng khỏi danh sách nếu có  databaseHelper.deleteCorrectAnswer(questionId); // Xóa khỏi cơ sở dữ liệu  }  // Tăng điểm cho câu trả lời đúng  score++; // Tăng điểm cho câu trả lời đúng  }  }   // Sắp xếp danh sách các câu trả lời sai theo thứ tự tăng dần  Collections.*sort*(incorrectAnswers); // Sắp xếp danh sách   // Log ra các ID câu trả lời sai  Log.*d*("TestActivity", "Các câu trả lời sai: " + incorrectAnswers);   // Hiển thị điểm số và chuyển đến ResultActivity  Intent intent = new Intent(TestActivity.this, ResultActivity.class);  intent.putExtra("TITLE", title); // Gửi tiêu đề đề thi  intent.putExtra("SCORE", score); // Truyền điểm số đến ResultActivity  intent.putExtra("TOTAL\_QUESTIONS", questions.size()); // Truyền tổng số câu hỏi  intent.putExtra("INCORRECT\_CRITICAL\_COUNT", incorrectCriticalCount); // Truyền số lượng câu hỏi quan trọng sai  intent.putExtra("EXAMS\_INDEX", examIndex); // Truyền examsIndex đến ResultActivity  startActivity(intent); // Chuyển đến ResultActivity  finish(); // Tùy chọn để kết thúc activity này  }    private List<Question> loadQuestionsByExamIndex(int examIndex) {  List<Question> questionList = new ArrayList<>();  List<Integer> questionIdList;   // Xác định mảng câu hỏi dựa trên examIndex  switch (examIndex) {  case 2002:  questionIdList = Arrays.*stream*(khainiemQuytac).boxed().collect(Collectors.*toList*());  break;  case 2003:  questionIdList = Arrays.*stream*(nghiepvuVantai).boxed().collect(Collectors.*toList*());  break;  case 2004:  questionIdList = Arrays.*stream*(vanhoaGiaothong).boxed().collect(Collectors.*toList*());  break;  case 2005:  questionIdList = Arrays.*stream*(kithuatLaixe).boxed().collect(Collectors.*toList*());  break;  case 2006:  questionIdList = Arrays.*stream*(cautaoSuachua).boxed().collect(Collectors.*toList*());  break;  case 2007:  questionIdList = Arrays.*stream*(bienbao).boxed().collect(Collectors.*toList*());  break;  case 2008:  questionIdList = Arrays.*stream*(sahinh).boxed().collect(Collectors.*toList*());  break;  default:  // Trường hợp không hợp lệ  return questionList;  }   try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Phân tích câu hỏi từ JSON và kiểm tra ID trong questionIdList  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");   // Chuyển ID từ String sang Integer để so sánh  int questionId = Integer.*parseInt*(id);   // Kiểm tra nếu ID câu hỏi có nằm trong questionIdList  if (questionIdList.contains(questionId)) {  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   // Tạo một đối tượng Question mới và thêm vào danh sách  Question question = new Question(id, questionText, options, answer, suggest, image);  questionList.add(question);  }  }   } catch (Exception e) {  e.printStackTrace();  }  return questionList;  }   private List<Question> loadIncorrectQuestions() {  List<Question> incorrectQuestionList = new ArrayList<>();  try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Log để xem giá trị của incorrectAnswers  Log.*d*("IncorrectAnswers", "List of incorrect answers: " + incorrectAnswers.toString());   // Lọc câu hỏi dựa trên ID trong incorrectAnswers  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");   // Kiểm tra nếu ID nằm trong mảng incorrectAnswers  if (incorrectAnswers.contains(Integer.*parseInt*(id))) {  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   // Tạo đối tượng Question và thêm vào danh sách  Question question = new Question(id, questionText, options, answer, suggest, image);  incorrectQuestionList.add(question);  }  }   } catch (Exception e) {  e.printStackTrace();  }  return incorrectQuestionList;  }     // Load critical questions based on IDs from the criticals list  private List<Question> loadCriticalQuestions() {  List<Question> criticalQuestionList = new ArrayList<>();  criticals = loadCriticalQuestionsFromJson(); // Load critical question IDs   try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Parse questions from JSON and match with critical IDs  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");   // Check if the current question ID is in the criticals list  if (criticals.contains(id)) {  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   // Create a new Question object and add it to the list  Question question = new Question(id, questionText, options, answer, suggest, image);  criticalQuestionList.add(question);  }  }   } catch (Exception e) {  e.printStackTrace();  }  return criticalQuestionList;  }  private List<Question> loadTop50() {  List<Question> top50QuestionList = new ArrayList<>();  // Chuyển arraystop50 thành List<Integer>  List<Integer> arrayStop50List = Arrays.*stream*(arraystop50).boxed().collect(Collectors.*toList*());   try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Parse questions from JSON and match with IDs in arrayStop50  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");   // Chuyển ID từ String sang Integer để so sánh  int questionId = Integer.*parseInt*(id);   // Kiểm tra nếu ID câu hỏi có nằm trong arrayStop50  if (arrayStop50List.contains(questionId)) {  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   // Create a new Question object and add it to the list  Question question = new Question(id, questionText, options, answer, suggest, image);  top50QuestionList.add(question);  }  }   } catch (Exception e) {  e.printStackTrace();  }  return top50QuestionList;  }    // Load critical questions from JSON  private List<String> loadCriticalQuestionsFromJson() {  List<String> criticalList = new ArrayList<>();  try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray criticalArray = jsonObject.getJSONArray("critical");   // Parse critical IDs from JSON  for (int i = 0; i < criticalArray.length(); i++) {  String criticalId = criticalArray.getString(i);  criticalList.add(criticalId);  }   } catch (Exception e) {  e.printStackTrace();  }  return criticalList;  }   // Load questions from JSON file for random selection  private List<Question> loadQuestionsFromJson() {  List<Question> questionList = new ArrayList<>();  try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Parse questions from JSON  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   Question question = new Question(id, questionText, options, answer, suggest, image);  questionList.add(question);  }   // Shuffle questions and limit to 35  java.util.Collections.*shuffle*(questionList);  int numberOfQuestionsToTake = Math.*min*(35, questionList.size());  questionList = questionList.subList(0, numberOfQuestionsToTake);   } catch (Exception e) {  e.printStackTrace();  }  return questionList;  }   // Load questions based on specific array for non-random selection  private List<Question> loadQuestionsFromArray(int[] questionIds) {  List<Question> questionList = new ArrayList<>();  try {  InputStream is = getAssets().open("driver\_data.json");  byte[] buffer = new byte[is.available()];  is.read(buffer);  is.close();  String json = new String(buffer, StandardCharsets.*UTF\_8*);   JSONObject jsonObject = new JSONObject(json);  JSONArray questionArray = jsonObject.getJSONArray("question");   // Parse questions from JSON and match with questionIds  for (int i = 0; i < questionArray.length(); i++) {  JSONObject questionObj = questionArray.getJSONObject(i);  String id = questionObj.getString("id");   // Only add questions that match the given array of IDs  for (int questionId : questionIds) {  if (id.equals(String.*valueOf*(questionId))) {  String questionText = questionObj.getString("question");   JSONObject optionsJson = questionObj.getJSONObject("option");  Options options = new Options(  optionsJson.getString("a"),  optionsJson.getString("b"),  optionsJson.getString("c"),  optionsJson.optString("d", null),  optionsJson.optString("e", null)  );   String answer = questionObj.getString("answer");  String suggest = questionObj.optString("suggest", "");   // Load image if available  JSONObject imageJson = questionObj.optJSONObject("image");  Question.Image image = null;  if (imageJson != null) {  String img1 = imageJson.getString("img1");  image = new Question.Image(img1);  }   Question question = new Question(id, questionText, options, answer, suggest, image);  questionList.add(question);  break;  }  }  }   } catch (Exception e) {  e.printStackTrace();  }  return questionList;  } } |

* + **MainActivity.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Intent; import android.net.Uri; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.ImageButton;  import androidx.activity.EdgeToEdge; import androidx.appcompat.app.AlertDialog; import androidx.appcompat.app.AppCompatActivity;  import java.util.ArrayList;  public class MainActivity extends AppCompatActivity {   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  EdgeToEdge.*enable*(this);  setContentView(R.layout.*activity\_main*);   ImageButton btnRandom = findViewById(R.id.*btnrandom*);  btnRandom.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, TestActivity.class);  intent.putExtra("examsIndex", -1); // Truyền giá trị -1 cho btnRandom  startActivity(intent);  }  });   // Thiết lập sự kiện cho nút chính xác  ImageButton btnExact = findViewById(R.id.*btnexact*);  btnExact.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, ExamsActivity.class);  startActivity(intent);  }  });   //btn liet  ImageButton btnLiet = findViewById(R.id.*btnliet*);  btnLiet.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, TestActivity.class);  intent.putExtra("examsIndex", 60);  startActivity(intent);  }  });   //btn btntopfalseds  ImageButton btntopfalseds = findViewById(R.id.*btntopfalseds*);  btntopfalseds.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, TestActivity.class);  intent.putExtra("examsIndex", 50); // Truyền giá trị -1 cho btnRandom  startActivity(intent);  }  });   //btn btntopfalseds  ImageButton btnfalseds = findViewById(R.id.*btnfalseds*);  btnfalseds.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  DatabaseHelper databaseHelper = new DatabaseHelper(MainActivity.this);   // Lấy danh sách các câu trả lời sai  ArrayList<Integer> incorrectAnswers = databaseHelper.getIncorrectAnswers();   if (incorrectAnswers.isEmpty()) {  // Nếu không có câu sai nào, hiển thị hộp thoại thông báo  new AlertDialog.Builder(MainActivity.this)  .setTitle("Thông báo")  .setMessage("Bạn không có câu sai nào chưa được sửa.")  .setPositiveButton("Đồng ý", (dialog, which) -> {  // Trở lại MainActivity  dialog.dismiss();  })  .show();  } else {  // Nếu có câu sai, chuyển đến TestActivity  Intent intent = new Intent(MainActivity.this, TestActivity.class);  intent.putExtra("examsIndex", 40); // Truyền giá trị examsIndex = 40  startActivity(intent);  }  }  });    //btn btntrain  ImageButton btntrain = findViewById(R.id.*btntrain*);  btntrain.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, DriverTypeActivity.class);  intent.putExtra("examsIndex", 40); // Truyền giá trị -1 cho btnRandom  startActivity(intent);  }  });   //btn btnbienbao  ImageButton btnBienBao = findViewById(R.id.*btnbienbao*);  btnBienBao.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Intent intent = new Intent(MainActivity.this, SignActivity.class);  startActivity(intent);  }  });  //btn btnremind  ImageButton btnRemind = findViewById(R.id.*btnremind*);  btnRemind.setOnClickListener(v->{  Intent intent = new Intent(MainActivity.this, TipsActivity.class);  startActivity(intent);  });  //btn btnmophong  ImageButton btnmophong = findViewById(R.id.*btnmophong*);  btnmophong.setOnClickListener(v -> {  // Tạo Intent với hành động ACTION\_VIEW và URL của Facebook  Intent intent = new Intent(Intent.*ACTION\_VIEW*, Uri.*parse*("https://play.google.com/store/apps/details?id=com.AnG.Pro.Mo.Phong&hl=vi"));  startActivity(intent); // Mở trình duyệt với URL đã cung cấp  });  //btn btndeleteapp  ImageButton btndeleteapp = findViewById(R.id.*btndeleteapp*);  btndeleteapp.setOnClickListener(v -> {  // Tạo Intent với hành động ACTION\_VIEW và URL của Facebook  Intent intent = new Intent(MainActivity.this, TipsActivity.class);  startActivity(intent);  });   Button btnDeleteData = findViewById(R.id.*btnDeleteData*);  btnDeleteData.setOnClickListener(v -> {  // Tạo hộp thoại xác nhận  new AlertDialog.Builder(MainActivity.this)  .setTitle("Xác nhận xóa dữ liệu")  .setMessage("Bạn có chắc chắn muốn xóa tất cả dữ liệu không?")  .setPositiveButton("Đồng ý", (dialog, which) -> {  // Người dùng chọn "Đồng ý" => Xóa dữ liệu  DatabaseHelper databaseHelper = new DatabaseHelper(MainActivity.this);  databaseHelper.resetData(); // Gọi phương thức để reset dữ liệu   // Tạo Intent trở lại MainActivity  Intent intent = new Intent(MainActivity.this, MainActivity.class);  intent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TOP* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*); // Xóa các Activity trước đó  startActivity(intent);  finish(); // Kết thúc Activity hiện tại để tránh quay lại  })  .setNegativeButton("Hủy", (dialog, which) -> {  // Người dùng chọn "Hủy" => Đóng hộp thoại và trở về MainActivity  dialog.dismiss(); // Đóng hộp thoại  Intent intent = new Intent(MainActivity.this, MainActivity.class);  intent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TOP* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  startActivity(intent);  finish();  })  .show();  });    } } |

* + **DriverTypeAdapter.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Context; import android.content.Intent; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.TextView; import androidx.annotation.NonNull; import androidx.recyclerview.widget.RecyclerView; import java.util.List;  public class DriverTypeAdapter extends RecyclerView.Adapter<DriverTypeAdapter.ViewHolder> {  private Context context;  private List<DriverType> driverTypeList;   public DriverTypeAdapter(Context context, List<DriverType> driverTypeList) {  this.context = context;  this.driverTypeList = driverTypeList;  }   @NonNull  @Override  public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View view = LayoutInflater.*from*(context).inflate(R.layout.*item\_driver\_type*, parent, false);  return new ViewHolder(view);  }   @Override  public void onBindViewHolder(@NonNull ViewHolder holder, int position) {  DriverType driverType = driverTypeList.get(position);  holder.titleTextView.setText(driverType.getTitle());  holder.descriptionTextView.setText(driverType.getDescription());   holder.quantityTextView.setText("Gồm: " + driverType.getQuantity() + " câu");   holder.itemView.setOnClickListener(v -> {  Intent intent = new Intent(context, TestActivity.class);  intent.putExtra("examsIndex", driverType.getId());  context.startActivity(intent);  });  }   @Override  public int getItemCount() {  return driverTypeList.size();  }   public static class ViewHolder extends RecyclerView.ViewHolder {  TextView titleTextView, descriptionTextView, quantityTextView;   public ViewHolder(@NonNull View itemView) {  super(itemView);  titleTextView = itemView.findViewById(R.id.*titleTextView*);  descriptionTextView = itemView.findViewById(R.id.*descriptionTextView*);  quantityTextView = itemView.findViewById(R.id.*quantityTextView*);  }  } } |

* + **QuestionAdapter.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.util.Log; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.ImageView; import android.widget.RadioButton; import android.widget.RadioGroup; import android.widget.TextView;  import androidx.annotation.NonNull; import androidx.recyclerview.widget.RecyclerView;  import com.bumptech.glide.Glide;  import java.util.HashMap; import java.util.List; import java.util.Map;  public class QuestionAdapter extends RecyclerView.Adapter<QuestionAdapter.QuestionViewHolder> {  private final List<Question> questions; // Danh sách câu hỏi  private final Map<String, String> selectedAnswers = new HashMap<>(); // Lưu trữ câu trả lời đã chọn  private final List<String> criticals;   // Constructor  public QuestionAdapter(List<Question> questions, List<String> criticals) {  this.questions = questions;  this.criticals = criticals; // Gán giá trị cho criticals  }   @NonNull  @Override  public QuestionViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*question\_item*, parent, false);  return new QuestionViewHolder(view);  }   @Override  public void onBindViewHolder(@NonNull QuestionViewHolder holder, int position) {  Question question = questions.get(position);   // Log thông tin câu hỏi và vị trí  Log.*d*("QuestionAdapter", "Hiển thị câu hỏi tại vị trí: " + position + ", ID câu hỏi: " + question.getId());  String title = "Câu " + (position + 1) + ": " + question.getQuestionText();  holder.questionTextView.setText(title);   // Load the image if it exists  String img1Url = question.getImage().getImg1();  if (img1Url != null && !img1Url.isEmpty()) {  holder.questionImageView.setVisibility(View.*VISIBLE*);  Glide.*with*(holder.itemView.getContext())  .load(img1Url)  .into(holder.questionImageView);  } else {  holder.questionImageView.setVisibility(View.*GONE*);  }   // Set options visibility and text  holder.optionAButton.setVisibility(View.*GONE*);  holder.optionBButton.setVisibility(View.*GONE*);  holder.optionCButton.setVisibility(View.*GONE*);  holder.optionDButton.setVisibility(View.*GONE*);  holder.optionEButton.setVisibility(View.*GONE*);   // Set options dynamically  if (!question.getOptions().getA().equals("null")) {  holder.optionAButton.setText(question.getOptions().getA());  holder.optionAButton.setVisibility(View.*VISIBLE*);  }   if (!question.getOptions().getB().equals("null")) {  holder.optionBButton.setText(question.getOptions().getB());  holder.optionBButton.setVisibility(View.*VISIBLE*);  }   if (!question.getOptions().getC().equals("null")) {  holder.optionCButton.setText(question.getOptions().getC());  holder.optionCButton.setVisibility(View.*VISIBLE*);  }   if (!question.getOptions().getD().equals("null")) {  holder.optionDButton.setText(question.getOptions().getD());  holder.optionDButton.setVisibility(View.*VISIBLE*);  }   if (!question.getOptions().getE().equals("null")) {  holder.optionEButton.setText(question.getOptions().getE());  holder.optionEButton.setVisibility(View.*VISIBLE*);  }   // Lắng nghe sự thay đổi của RadioGroup  holder.answerGroup.setOnCheckedChangeListener((group, checkedId) -> {  RadioButton selectedButton = holder.itemView.findViewById(checkedId);  if (selectedButton != null) {  String selectedAnswerText = selectedButton.getText().toString(); // Text của đáp án được chọn  String selectedKey = getSelectedKey(selectedAnswerText, holder); // Lấy khóa đáp án được chọn   // Lưu lựa chọn của người dùng  selectedAnswers.put(question.getId(), selectedKey);   // Log đáp án đã chọn  Log.*d*("QuestionAdapter", "Câu hỏi ID: " + question.getId() + ", Khóa đáp án được chọn: " + selectedKey);  }  });   // Xóa lựa chọn nếu chưa có lựa chọn  holder.answerGroup.clearCheck();  }   private String getSelectedKey(String selectedAnswerText, QuestionViewHolder holder) {  if (selectedAnswerText.equals(holder.optionAButton.getText().toString())) {  return "a";  } else if (selectedAnswerText.equals(holder.optionBButton.getText().toString())) {  return "b";  } else if (selectedAnswerText.equals(holder.optionCButton.getText().toString())) {  return "c";  } else if (selectedAnswerText.equals(holder.optionDButton.getText().toString())) {  return "d";  } else if (selectedAnswerText.equals(holder.optionEButton.getText().toString())) {  return "e";  }  return ""; // Trả về chuỗi rỗng nếu không tìm thấy khóa  }   @Override  public int getItemCount() {  return questions.size();  }   // Lấy các đáp án đã chọn  public Map<String, String> getSelectedAnswers() {  return selectedAnswers;  }   public static class QuestionViewHolder extends RecyclerView.ViewHolder {  TextView questionTextView;  ImageView questionImageView;  RadioGroup answerGroup;  RadioButton optionAButton;  RadioButton optionBButton;  RadioButton optionCButton;  RadioButton optionDButton;  RadioButton optionEButton;   public QuestionViewHolder(View itemView) {  super(itemView);  questionTextView = itemView.findViewById(R.id.*question\_text*);  questionImageView = itemView.findViewById(R.id.*question\_image*);  answerGroup = itemView.findViewById(R.id.*answer\_group*);  optionAButton = itemView.findViewById(R.id.*option\_a*);  optionBButton = itemView.findViewById(R.id.*option\_b*);  optionCButton = itemView.findViewById(R.id.*option\_c*);  optionDButton = itemView.findViewById(R.id.*option\_d*);  optionEButton = itemView.findViewById(R.id.*option\_e*);  }  } } |

* + **SignAdapter.java**

|  |
| --- |
| * package com.example.gplxb2;  import android.content.Context; import android.content.Intent; import android.graphics.Color; import android.graphics.Typeface; import android.util.TypedValue; import android.view.Gravity; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.ImageView; import android.widget.TextView; import androidx.annotation.NonNull; import androidx.recyclerview.widget.RecyclerView; import com.bumptech.glide.Glide; import java.util.List;  public class SignAdapter extends RecyclerView.Adapter<SignAdapter.SignViewHolder> {  private Context context;  private List<Sign> signList;  private int titleCount = 0; // Biến đếm để theo dõi số tiêu đề đã gặp   public SignAdapter(Context context, List<Sign> signList) {  this.context = context;  this.signList = signList;  }   @NonNull  @Override  public SignViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View view = LayoutInflater.*from*(context).inflate(R.layout.*item\_sign*, parent, false);  return new SignViewHolder(view);  }   @Override  public void onBindViewHolder(@NonNull SignViewHolder holder, int position) {  Sign sign = signList.get(position);   // Kiểm tra nếu đây là tiêu đề (không có mô tả và ảnh)  boolean isTitle = (sign.getDes() == null || sign.getDes().isEmpty()) &&  (sign.getImagePath() == null || sign.getImagePath().equals("bienbao.png"));   if (isTitle) {  // Tăng biến đếm số tiêu đề lên  titleCount++;   // Cấu hình cho tiêu đề  holder.nameTextView.setText(sign.getName());  holder.nameTextView.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 25); // Kích thước chữ là 25  holder.nameTextView.setTypeface(null, Typeface.*BOLD*); // Đặt chữ in đậm  holder.nameTextView.setTextColor(Color.*BLACK*); // Màu chữ đen   // Đặt màu nền dựa trên chỉ số tiêu đề  int titleIndex = sign.getTitleIndex();  switch (titleIndex) {  case 0:  holder.itemView.setBackgroundColor(Color.*RED*); // Nền màu đỏ cho tiêu đề đầu tiên  break;  case 1:  holder.itemView.setBackgroundColor(Color.*YELLOW*); // Nền màu vàng cho tiêu đề thứ hai  break;  default:  holder.itemView.setBackgroundColor(Color.*TRANSPARENT*); // Nền trong suốt cho các tiêu đề khác  break;  }   // Căn giữa theo chiều dọc và chiều ngang  holder.nameTextView.setGravity(Gravity.*CENTER*);   // Đảm bảo ẩn mô tả và hình ảnh  holder.descriptionTextView.setVisibility(View.*GONE*);  holder.imageView.setVisibility(View.*GONE*);  } else {  // Cấu hình cho phần tử thông thường  holder.nameTextView.setText(sign.getName());  holder.nameTextView.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, 16); // Kích thước chữ bình thường  holder.nameTextView.setTypeface(null, Typeface.*BOLD*); // Đặt chữ in đậm   // Hiển thị hoặc ẩn mô tả  if (sign.getDes() == null || sign.getDes().isEmpty()) {  holder.descriptionTextView.setVisibility(View.*GONE*);  } else {  holder.descriptionTextView.setText(sign.getDes());  holder.descriptionTextView.setVisibility(View.*VISIBLE*);  }   // Hiển thị hoặc ẩn hình ảnh  if (sign.getImagePath() != null && !sign.getImagePath().equals("bienbao.png")) {  holder.imageView.setVisibility(View.*VISIBLE*);  holder.itemView.setBackgroundColor(Color.*TRANSPARENT*); // Đặt nền trong suốt cho phần tử thông thường  // Sử dụng Glide để tải hình ảnh  Glide.*with*(context)  .load(sign.getImagePath())  .placeholder(R.drawable.*bienbao*) // Hình ảnh mặc định khi đang tải  .error(R.drawable.*bienbao*) // Hình ảnh khi có lỗi tải  .into(holder.imageView);  } else {  holder.imageView.setVisibility(View.*GONE*);  }  }   holder.itemView.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Khởi tạo Intent để chuyển đến SignDetailActivity  Intent intent = new Intent(context, SignDetailActivity.class);  intent.putExtra("name", sign.getName());  intent.putExtra("description", sign.getDes());  intent.putExtra("imagePath", sign.getImagePath());  context.startActivity(intent);  }  });  }    @Override  public int getItemCount() {  return signList.size();  }   public static class SignViewHolder extends RecyclerView.ViewHolder {  TextView nameTextView;  TextView descriptionTextView;  ImageView imageView;   public SignViewHolder(@NonNull View itemView) {  super(itemView);  nameTextView = itemView.findViewById(R.id.*text\_name*);  descriptionTextView = itemView.findViewById(R.id.*text\_description*);  imageView = itemView.findViewById(R.id.*image\_view*);  }  } } |

* + **activity\_main.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/main"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  tools:context=".MainActivity">   <!-- Fixed Top Row -->  <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="16dp">   <TextView  android:id="@+id/textView"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#247AA8"  android:gravity="center"  android:paddingTop="10dp"  android:paddingBottom="10dp"  android:text="600 Câu GPLX Bằng B2"  android:textColor="#FFFFFF"  android:textSize="20sp"  android:textStyle="bold" />  </TableRow>    <!-- Scrollable Rows -->  <ScrollView  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1">   <TableLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:stretchColumns="\*">   <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="16dp"  android:paddingStart="16dp"  android:paddingEnd="16dp">   <ImageButton  android:id="@+id/btnrandom"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginEnd="8dp"  android:paddingStart="16dp"  app:srcCompat="@drawable/ngaunhien" />   <ImageButton  android:id="@+id/btnexact"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginStart="8dp"  android:paddingEnd="16dp"  app:srcCompat="@drawable/bode" />  </TableRow>   <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="16dp"  android:paddingStart="16dp"  android:paddingEnd="16dp">   <ImageButton  android:id="@+id/btnfalseds"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginEnd="8dp"  android:paddingStart="16dp"  app:srcCompat="@drawable/causai" />   <ImageButton  android:id="@+id/btntrain"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginStart="8dp"  android:paddingEnd="16dp"  app:srcCompat="@drawable/ontap" />  </TableRow>   <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="16dp"  android:paddingStart="16dp"  android:paddingEnd="16dp">   <ImageButton  android:id="@+id/btnbienbao"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginEnd="8dp"  android:paddingStart="16dp"  app:srcCompat="@drawable/bienbao" />   <ImageButton  android:id="@+id/btnremind"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginStart="8dp"  android:paddingEnd="16dp"  app:srcCompat="@drawable/meo" />  </TableRow>   <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="16dp"  android:paddingStart="16dp"  android:paddingEnd="16dp">   <ImageButton  android:id="@+id/btnliet"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginEnd="8dp"  android:paddingStart="16dp"  app:srcCompat="@drawable/cauliet" />   <ImageButton  android:id="@+id/btntopfalseds"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginStart="8dp"  android:paddingEnd="16dp"  app:srcCompat="@drawable/topsai" />  </TableRow>   <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:paddingStart="16dp"  android:paddingEnd="16dp">   <ImageButton  android:id="@+id/btnmophong"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginEnd="8dp"  android:paddingStart="16dp"  app:srcCompat="@drawable/mophong" />   <ImageButton  android:id="@+id/btndeleteapp"  android:layout\_width="0dp"  android:layout\_height="130dp"  android:layout\_weight="1"  android:layout\_marginStart="8dp"  android:paddingEnd="16dp"  app:srcCompat="@drawable/noad" />  </TableRow>  <TableRow  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:padding="16dp"  android:gravity="center\_horizontal">   <!-- Nút để xóa dữ liệu -->  <Button  android:id="@+id/btnDeleteData"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:backgroundTint="#FF4444"  android:paddingStart="40dp"  android:paddingEnd="40dp"  android:paddingTop="10dp"  android:layout\_marginTop="5dp"  android:paddingBottom="10dp"  android:text="Xóa Dữ Liệu"  android:textColor="#FFFFFF"  android:textSize="18sp"  android:textStyle="bold"  android:layout\_marginBottom="16dp"  android:layout\_gravity="center" />  </TableRow>  </TableLayout>  </ScrollView> </LinearLayout> |

* + **activity\_exams.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:orientation="vertical"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">   <!-- Toolbar phía trên -->   <!-- Phần chứa các nút với khả năng cuộn xuống -->  <androidx.appcompat.widget.Toolbar  android:id="@+id/top\_toolbar"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#329BF1"  android:elevation="4dp">   <!-- Nút Back --> <!-- <ImageButton--> <!-- android:id="@+id/back\_button"--> <!-- android:layout\_width="wrap\_content"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:layout\_gravity="start"--> <!-- android:gravity="center\_vertical"--> <!-- android:text="Quay lại"--> <!-- android:backgroundTint="#329BF1"--> <!-- android:textColor="@android:color/white"--> <!-- android:textSize="16sp"--> <!-- android:layout\_marginStart="16dp" />-->  <ImageButton  android:id="@+id/back\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:src="@drawable/back"  android:background="?attr/selectableItemBackground"  android:contentDescription="Quay lại"  android:layout\_marginStart="16dp"  android:padding="8dp"/>    <TextView  android:id="@+id/txt\_title\_topic"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="Các bộ đề"  android:textColor="@android:color/white"  android:textSize="18sp"/>  </androidx.appcompat.widget.Toolbar>   <ScrollView  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1"  android:padding="16dp">   <GridLayout  android:id="@+id/button\_grid"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:columnCount="2"  android:padding="8dp"  android:orientation="horizontal">   <!-- Các nút xếp thành 2 cột -->  <!-- Tăng kích thước chữ -->   <Button  android:layout\_width="0dp"  android:layout\_height="97dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 1"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 2"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 3"  android:background="@drawable/blue\_button\_background"  android:textSize="20sp"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 4"  android:background="@drawable/blue\_button\_background"  android:textSize="20sp"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 5"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 6"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 7"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 8"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 9"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 10"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 11"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 12"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 13"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 14"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 15"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 16"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 17"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->   <Button  android:layout\_width="0dp"  android:layout\_height="100dp"  android:layout\_columnWeight="1"  android:layout\_margin="8dp"  android:text="Đề 18"  android:textSize="20sp"  android:background="@drawable/blue\_button\_background"  android:textColor="@android:color/black" /> <!-- Màu chữ đen -->  </GridLayout>  </ScrollView>  </LinearLayout> |

* + **activity\_test.xml**

|  |
| --- |
| * <LinearLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical">   <!-- Toolbar -->  <androidx.appcompat.widget.Toolbar  android:id="@+id/toolbar"  android:layout\_width="match\_parent"  android:layout\_height="?attr/actionBarSize"  android:background="#329BF1"  android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">   <!-- TextView hiển thị số câu hiện tại / tổng số câu -->  <TextView  android:id="@+id/question\_counter"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:gravity="center\_vertical"  android:text="1/35"  android:textColor="@android:color/white"  android:textSize="16sp"  android:layout\_marginStart="16dp"/>   <!-- TextView hiển thị bộ đếm thời gian -->  <TextView  android:id="@+id/timer"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="10:00"  android:textColor="@android:color/white"  android:textSize="18sp"/>   <!-- Nút submit -->  <Button  android:id="@+id/submit\_btn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="end"  android:text="Chấm điểm"  android:textColor="@android:color/white"  android:background="@android:color/transparent"  android:textSize="16sp"  android:layout\_marginEnd="16dp"/>  </androidx.appcompat.widget.Toolbar>   <!-- RecyclerView để hiển thị các câu hỏi -->  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/question\_list"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1"  android:isScrollContainer="false"  android:orientation="horizontal"/>   <!-- Toolbar mới với các nút điều khiển -->  <androidx.appcompat.widget.Toolbar  android:id="@+id/control\_toolbar"  android:layout\_width="match\_parent"  android:layout\_height="?attr/actionBarSize"  android:background="#329BF1"  android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">   <!-- Nút Back -->  <ImageButton  android:id="@+id/back\_btn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:background="@android:color/transparent"  android:contentDescription="Back"  android:src="@drawable/back"  android:layout\_gravity="start"  android:layout\_marginStart="16dp" />   <!-- Khoảng trống để đẩy nút Next sang bên phải -->  <View  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_weight="1" />   <!-- Nút Next -->  <ImageButton  android:id="@+id/next\_btn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:background="@android:color/transparent"  android:contentDescription="Next"  android:src="@drawable/next"  android:layout\_gravity="end"  android:layout\_marginEnd="16dp" />  </androidx.appcompat.widget.Toolbar> </LinearLayout> |

* + **activity\_result.xml**

|  |
| --- |
| * <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  > <!-- Canh giữa toàn bộ nội dung -->   <!-- Toolbar -->  <androidx.appcompat.widget.Toolbar  android:id="@+id/toolbar"  android:layout\_width="match\_parent"  android:layout\_height="?attr/actionBarSize"  android:background="#329BF1"  android:layout\_marginBottom="150dp"  android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">   <!-- TextView hiển thị tiêu đề -->  <TextView  android:id="@+id/toolbar\_title"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textColor="@android:color/white"  android:textSize="20sp"  android:layout\_gravity="center"  android:textStyle="bold"/>  </androidx.appcompat.widget.Toolbar>   <!-- Sử dụng android:layout\_weight để căn giữa theo chiều dọc -->  <TextView  android:id="@+id/result\_text"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="24sp"  android:textStyle="bold"  android:layout\_marginBottom="8dp"  android:layout\_gravity="center"  /> <!-- Phân bổ không gian đồng đều -->   <TextView  android:id="@+id/total\_questions\_text"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="18sp"  android:layout\_marginBottom="8dp"  android:layout\_gravity="center"  /> <!-- Phân bổ không gian đồng đều -->   <TextView  android:id="@+id/result\_message"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:paddingTop="16dp"  android:textSize="20sp"  android:layout\_gravity="center"  android:layout\_marginBottom="150dp"  android:gravity="center"  /> <!-- Phân bổ không gian đồng đều -->   <Button  android:id="@+id/back\_btn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Quay về trang chủ"  android:textColor="@android:color/white"  android:backgroundTint="#329BF1"  android:layout\_marginTop="16dp"  android:layout\_gravity="center"/> <!-- Canh giữa nút -->  </LinearLayout> |

* + **question\_item.xml**

|  |
| --- |
| * <LinearLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical">   <TextView  android:id="@+id/question\_text"  android:layout\_width="wrap\_content"  android:textStyle="bold"  android:textSize="20dp"  android:layout\_height="wrap\_content" />   <ImageView  android:id="@+id/question\_image"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:maxHeight="300dp"  android:layout\_gravity="center"  android:adjustViewBounds="true" />   <RadioGroup  android:id="@+id/answer\_group"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content">   <RadioButton  android:id="@+id/option\_a"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />   <RadioButton  android:id="@+id/option\_b"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />   <RadioButton  android:id="@+id/option\_c"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />   <RadioButton  android:id="@+id/option\_d"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />   <RadioButton  android:id="@+id/option\_e"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />  </RadioGroup>  </LinearLayout> |

* + **activity\_driver\_type.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">   <!-- Toolbar -->   <!-- RecyclerView -->   <androidx.appcompat.widget.Toolbar  android:id="@+id/top\_toolbar"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:background="#329BF1"  android:elevation="4dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintHorizontal\_bias="0.0"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent">   <!-- Nút Back --> <!-- <Button--> <!-- android:id="@+id/back\_button"--> <!-- android:layout\_width="wrap\_content"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:layout\_gravity="start"--> <!-- android:layout\_marginStart="16dp"--> <!-- android:backgroundTint="#329BF1"--> <!-- android:gravity="center\_vertical"--> <!-- android:text="Quay lại"--> <!-- android:textColor="@android:color/white"--> <!-- android:textSize="16sp" />-->  <ImageButton  android:id="@+id/back\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:src="@drawable/back"  android:background="?attr/selectableItemBackground"  android:contentDescription="Quay lại"  android:layout\_marginStart="16dp"  android:padding="8dp"/>   <!-- Tiêu đề -->  <TextView  android:id="@+id/txt\_title\_train"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="Ôn tập câu hỏi"  android:textColor="@android:color/white"  android:textSize="18sp" />  </androidx.appcompat.widget.Toolbar>   <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recyclerView"  android:layout\_width="0dp"  android:layout\_height="0dp"  app:layout\_constraintTop\_toBottomOf="@id/top\_toolbar"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> |

* + **item\_driver\_type.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:padding="16dp">   <TextView  android:id="@+id/titleTextView"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:textSize="18sp"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toStartOf="@id/quantityTextView"  app:layout\_constraintHorizontal\_chainStyle="packed" />   <TextView  android:id="@+id/descriptionTextView"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:textSize="14sp"  app:layout\_constraintTop\_toBottomOf="@id/titleTextView"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toStartOf="@id/quantityTextView" />   <TextView  android:id="@+id/quantityTextView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="16sp"  android:textColor="@color/green"  android:textStyle="italic"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> |

* + **activity\_sign.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical">   <androidx.appcompat.widget.Toolbar  android:id="@+id/top\_toolbar"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#329BF1"  android:elevation="4dp">   <!-- Nút Back --> <!-- <Button--> <!-- android:id="@+id/back\_button"--> <!-- android:layout\_width="wrap\_content"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:layout\_gravity="start"--> <!-- android:gravity="center\_vertical"--> <!-- android:text="Quay lại"--> <!-- android:backgroundTint="#329BF1"--> <!-- android:textColor="@android:color/white"--> <!-- android:textSize="16sp"--> <!-- android:layout\_marginStart="16dp" />-->  <ImageButton  android:id="@+id/back\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:src="@drawable/back"  android:background="?attr/selectableItemBackground"  android:contentDescription="Quay lại"  android:layout\_marginStart="16dp"  android:padding="8dp"/>   <TextView  android:id="@+id/txt\_title\_topic"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="Biển báo đường bộ"  android:textColor="@android:color/white"  android:textSize="18sp"/>  </androidx.appcompat.widget.Toolbar>   <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recyclerView"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"/>  </LinearLayout> |

* + **item\_sign.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical"  android:padding="8dp">  <!-- <TextView--> <!-- android:id="@+id/txt\_space"--> <!-- android:layout\_width="wrap\_content"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:textSize="18sp"--> <!-- android:textStyle="bold"--> <!-- android:paddingBottom="4dp" />-->  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal">   <ImageView  android:id="@+id/image\_view"  android:layout\_width="100dp"  android:layout\_height="100dp"  android:scaleType="centerCrop" />   <LinearLayout  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  android:orientation="vertical"  android:paddingStart="8dp"  android:paddingLeft="8dp">   <TextView  android:id="@+id/text\_name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:layout\_gravity="center"  android:paddingTop="4dp" />   <TextView  android:id="@+id/text\_description"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:paddingTop="2dp" />  </LinearLayout>  </LinearLayout> </LinearLayout> |

* + **activity\_tips.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  android:background="#FFFFFF">   <!-- Toolbar -->  <androidx.appcompat.widget.Toolbar  android:id="@+id/top\_toolbar"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#329BF1"  android:elevation="4dp">   <!-- Nút Back --> <!-- <Button--> <!-- android:id="@+id/back\_button"--> <!-- android:layout\_width="80dp"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:layout\_gravity="start"--> <!-- android:gravity="center\_vertical"--> <!-- android:text="Quay lại"--> <!-- android:backgroundTint="#329BF1"--> <!-- android:textColor="@android:color/white"--> <!-- android:textSize="11sp"--> <!-- android:layout\_marginStart="0dp"--> <!-- />-->  <ImageButton  android:id="@+id/back\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:src="@drawable/back"  android:background="?attr/selectableItemBackground"  android:contentDescription="Quay lại"  android:layout\_marginStart="16dp"  android:padding="8dp"/>   <!-- Tiêu đề -->  <TextView  android:id="@+id/txt\_detail\_sign"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="Mẹo 600 câu hỏi ôn thi GPLX"  android:textColor="@android:color/white"  android:textSize="18sp"/>  </androidx.appcompat.widget.Toolbar>   <!-- Nội dung -->  <ScrollView  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:padding="16dp">   <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical">   <!-- Mục nội dung -->  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Cấp phép"  android:textSize="20sp"  android:textStyle="bold"  android:textColor="#000000"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Đường cấm dừng, cấm đỗ, cấm đi do UBND cấp tỉnh cấp"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Xe quá khổ, quá tải do: cơ quan quản lý đường bộ có thẩm quyền cấp phép"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Nồng độ cồn"  android:textSize="20sp"  android:textStyle="bold"  android:textColor="#000000"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Người điều khiển xe mô tô, ô tô, máy kéo trên đường mà trong máu hoặc hơi thở có nồng độ cồn: Bị nghiêm cấm"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Khoảng cách an toàn tối thiểu"  android:textSize="20sp"  android:textStyle="bold"  android:textColor="#000000"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="35m nếu vận tốc lưu hành (V) = 60 (km/h)"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="55m nếu 60 &#60; V &#8804; 80"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="70m nếu 80 &#60; V &#8804; 100"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="100m nếu 100 &#60; V &#8804; 120"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Dưới 60km/h: Chủ động và đảm bảo khoảng cách."  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hỏi về tuổi (T)"  android:textSize="20sp"  android:textStyle="bold"  android:textColor="#000000"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Tuổi tối đa hạng E: nam 55, nữ 50"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Tuổi lấy bằng lái xe (cách nhau 3 tuổi)"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Gắn máy: 16T (dưới 50cm3)"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Mô tô + B1 + B2: 18T"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="C, FB: 21T"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="D, FC: 24T"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="E, FD: 27T"  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Trên đường cao tốc, trong đường hầm, đường vòng, đầu dốc, nơi tầm nhìn hạn chế"  android:textSize="20sp"  android:textStyle="bold"  android:textColor="#000000"  android:paddingBottom="8dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Không được quay đầu xe, không lùi, không vượt."  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Không được vượt trên cầu hẹp có một làn xe."  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Không được phép quay đầu xe ở phần đường dành cho người đi bộ qua đường."  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="4dp" />   <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Không được dừng, đỗ xe gần bến xe buýt, nơi có biển báo cấm dừng."  android:textSize="16sp"  android:textColor="#333333"  android:paddingBottom="8dp" />   </LinearLayout>  </ScrollView> </LinearLayout> |

* + **activity\_sign\_detail.xml**

|  |
| --- |
| * <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical" >   <androidx.appcompat.widget.Toolbar  android:id="@+id/top\_toolbar"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#329BF1"  android:elevation="4dp">   <!-- Nút Back --> <!-- <Button--> <!-- android:id="@+id/back\_button"--> <!-- android:layout\_width="wrap\_content"--> <!-- android:layout\_height="wrap\_content"--> <!-- android:layout\_gravity="start"--> <!-- android:gravity="center\_vertical"--> <!-- android:text="Quay lại"--> <!-- android:backgroundTint="#329BF1"--> <!-- android:textColor="@android:color/white"--> <!-- android:textSize="16sp"--> <!-- android:layout\_marginStart="16dp" />-->  <ImageButton  android:id="@+id/back\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="start"  android:src="@drawable/back"  android:background="?attr/selectableItemBackground"  android:contentDescription="Quay lại"  android:layout\_marginStart="16dp"  android:padding="8dp"/>   <TextView  android:id="@+id/txt\_detail\_sign"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:gravity="center\_vertical"  android:text="Chi tiết biển báo"  android:textColor="@android:color/white"  android:textSize="18sp"/>  </androidx.appcompat.widget.Toolbar>   <ImageView  android:id="@+id/image\_view"  android:layout\_width="match\_parent"  android:layout\_height="200dp"  android:layout\_marginTop="16dp"  android:scaleType="fitCenter" /> <!-- Chỉnh scaleType thành fitCenter -->   <TextView  android:id="@+id/text\_name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="24sp"  android:textStyle="bold" />   <TextView  android:id="@+id/text\_description"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="16sp"  android:layout\_marginTop="8dp"/>   </LinearLayout> |