**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT ­- HÀN**

**KHOA KĨ THUẬT MÁY TÍNH**



**ĐỒ ÁN CƠ SỞ III**

**XÂY DỰNG PHẦN MỀM LUYỆN THI TRẮC NGHIỆM TRÊN THIẾT BỊ DI ĐỘNG**

Ngành **:AN TOÀN THÔNG TIN**

Giảng Viên Hướng Dẫn **: Ths. Ninh Khánh Chi**

Sinh viên thực hiện **:** **Đinh Hữu Đức**

Lớp **: 20NS**

*TP.Đà Nẵng , tháng 5 năm 2022*

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT ­- HÀN**

**KHOA KĨ THUẬT MÁY TÍNH**

**ĐỒ ÁN CƠ SỞ III**

**XÂY DỰNG PHẦN MỀM LUYỆN THI TRẮC NGHIỆM TRÊN THIẾT BỊ DI ĐỘNG**

Ngành **:AN TOÀN THÔNG TIN**

Giảng Viên Hướng Dẫn **: Ths. Ninh Khánh Chi**

Sinh viên thực hiện **:** **Đinh Hữu Đức**

Lớp **: 20NS**

*TP.Đà Nẵng , tháng 5 năm 2022*

**LỜI CẢM ƠN**

Với sự phát triển nhanh chóng của khoa học công nghệ và đặc biệt là ngành công nghệ thông tin, làm từng bước thay đổi cuộc sống của con người bởi các ứng dụng sản phẩm của công nghệ. Đi cùng với sự phát triển của ngành công nghệ thông tin là sự ra đời ngày càng nhiều thiết bị cầm tay, smartphone. Nhu cầu của con người về việc sử dụng và giải trí trên các thiết bị di động cũng ngày càng cao và đa dạng. Một trong những bộ phận ứng dụng công nghệ thông tin chính là thay đổi và giúp đỡ việc sinh hoạt hàng ngày của mọi người trong cuộc sống. Vì vậy em đã lựa chọn đề tài “**Xây dựng phần mềm luyện thi trắc nghiệm trên thiết bị di động**” để làm đồ án với ý nghĩa mang tính thực tế.

Với mong muốn người dùng có thể ôn lại những kiến thức mình đã học một cách tốt nhất, những kiến thức quan trọng và sẽ làm bài đạt kết quả cao trong các kỳ thi.

Trong thời gian làm đồ án em đã gặp không ít khó khăn nhưng được sự giúp đỡ hướng dẫn tận tình. Em xin chân thành cảm ơn cô **Ths. Ninh Khánh Chi** – người trực tiếp hướng dẫn đã tận tình hướng dẫn và giúp đỡ em hoàn thành đồ án.

**NHẬN XÉT**

**(Của giảng viên hướng dẫn)**

**……………………………………………………………………………………..**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

**……………………………………………………………………………..………**

*Chữ ký của GVHD*

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 11](#_Toc106047430)

[1.1. Mục tiêu đề tài 11](#_Toc106047431)

[1.1.1. Lý do chọn đề tài 11](#_Toc106047432)

[1.1.2. Mục tiêu đề tài 11](#_Toc106047433)

[1.1.3. Giới han phạm vi nghiêm cứu 11](#_Toc106047434)

[1.1.4. Ý nghĩa khoa học thực tiễn 12](#_Toc106047435)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT & THIẾT KẾ HỆ THỐNG 13](#_Toc106047436)

[2.1 Giới thiệu hệ điều hành Android 13](#_Toc106047437)

[2.1.1 Android là gì? 13](#_Toc106047438)

[2.1.2 Lịch sử phát triển 14](#_Toc106047439)

[2.1.3 Cộng đồng mã nguồn mở 15](#_Toc106047440)

[2.1.4 Kiến trúc hệ điều hành Android 17](#_Toc106047441)

[2.2 Các thành phần trong 1 ứng dụng android 19](#_Toc106047442)

[2.2.1 Activities 20](#_Toc106047443)

[2.2.2 Services 22](#_Toc106047444)

[2.2.3 Content Providers 23](#_Toc106047445)

[2.2.4 Broadcast Receivers 23](#_Toc106047446)

[2.2.5 Intent 23](#_Toc106047447)

[2.2.6 File Manifest 24](#_Toc106047448)

[2.2.7 Resources 28](#_Toc106047449)

[2.3 Các thành phần giao diện trong Android 29](#_Toc106047450)

[2.3.1 View 29](#_Toc106047451)

[2.3.2 ViewGroup 30](#_Toc106047452)

[2.3.3 Button 31](#_Toc106047453)

[2.3.4 ImageView 32](#_Toc106047454)

[2.3.5 ListView 32](#_Toc106047455)

[2.3.6 TextView 32](#_Toc106047456)

[2.3.7 EditText 33](#_Toc106047457)

[2.3.8 CheckBox 33](#_Toc106047458)

[2.4 Phân tích & thiết kế hệ thống 33](#_Toc106047459)

[2.4.1 Sơ sơ đồ phân cấp chức năng phía frontend 33](#_Toc106047460)

[2.4.2 Sơ đồ phân cấp chức năng phía Backend 34](#_Toc106047461)

[2.4.3 Biểu đồ Use – Case tổng quát 35](#_Toc106047462)

[2.4.4 Biểu đồ lớp 37](#_Toc106047463)

[2.4.4.1 Danh sách lớp Subject 37](#_Toc106047464)

[2.4.4.2 Danh sách lớp ChooseNumExam 37](#_Toc106047465)

[2.4.4.3 Danh sách lớp Exam 37](#_Toc106047466)

[2.4.4.4 Danh sách lớp Save points 38](#_Toc106047467)

[2.4.4.5 Danh sách lớp seeTestScores 38](#_Toc106047468)

[2.4.4.6 Danh sách lớp Formula 39](#_Toc106047469)

[2.4.4.7 Sơ đồ lớp của Exam 39](#_Toc106047470)

[2.4.4.8 Sơ đồ lớp của Formula 40](#_Toc106047471)

[2.4.4.9 Sơ đồ trạng thái của Exam 41](#_Toc106047472)

[2.4.4.10 Sơ đồ tuần tự của Exam 42](#_Toc106047473)

[2.4.5 Thiết kế cơ sở dữ liệu 43](#_Toc106047474)

[2.4.5.1 Bảng công thức 43](#_Toc106047475)

[2.4.5.2 Bảng trắc nghiệm 43](#_Toc106047476)

[2.4.5.3 Bảng SQLite\_Sequence 44](#_Toc106047477)

[2.4.5.4 Bảng tbScore 44](#_Toc106047478)

[2.4.5.5 Bảng android\_metadata 45](#_Toc106047479)

[CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM 46](#_Toc106047480)

[3.1 Giao diện chương trình 46](#_Toc106047481)

[3.1.1 Giao diện màn hình chính: 46](#_Toc106047482)

[3.1.2 Giao diện menu: 47](#_Toc106047483)

[3.1.3 Giao diện đề: 48](#_Toc106047484)

[3.1.4 Giao diện trả lời: 49](#_Toc106047485)

[3.1.5 Giao diện tổng kết câu đã trả lời: 49](#_Toc106047486)

[3.1.6 Giao diện đáp án: 50](#_Toc106047487)

[3.1.7 Giao diện xem điểm: 52](#_Toc106047488)

[3.1.8 Giao diện lưu điểm: 53](#_Toc106047489)

[3.1.9 Giao diện tìm kiếm công thức: 55](#_Toc106047490)

[3.1.10 Giao diện danh sách điểm: 57](#_Toc106047491)

[3.1.11 Giao diện thông tin tài khoản: 58](#_Toc106047492)

[CHƯƠNG 4: KẾT LUẬN VÀ KIẾN NGHỊ 61](#_Toc106047493)

[4.1 Kết quả đạt được 61](#_Toc106047494)

[4.2 Hướng phát triển 61](#_Toc106047495)

**DANH SÁCH HÌNH ẢNH**

[Hình 2.1: Kiến trúc hệ điều hành Android 17](#_Toc105186757)

[Hình 2.2: Vòng đời hoạt động của Android 21](#_Toc105186760)

[Hình 2.3: Thư mục Android Manifest 25](#_Toc105186766)

[Hình 2.4: File AndroidManifest.xml 26](#_Toc105186767)

[Hình 2.5: Ví dụ về Label 28](#_Toc105186768)

[Hình 2.6: Sơ đồ phân cấp chức năng phía frontend 34](#_Toc105186781)

[Hình 2.7: Sơ đồ phân cấp chức năng phía backend 35](#_Toc105186783)

[Hình 2.8: Sơ đồ UseCase 36](#_Toc105186785)

[Hình 2.9: Sơ đồ lớp chức năng làm bài kiểm tra 40](#_Toc105186804)

[Hình 2.10: Sơ đồ lớp chức năng xem công thức 41](#_Toc105186806)

[Hình 2.11: Sơ đồ trạng thái chức năng làm bài kiểm tra 42](#_Toc105186808)

[Hình 2.12: Sơ đồ tuần tự chức năng làm bài kiểm tra 43](#_Toc105186810)

[Hình 2.13: Bảng cơ sở dữ liệu công thức 43](#_Toc105186813)

[Hình 2.14: Bảng cơ sở dữ liệu trắc nghiệm 44](#_Toc105186815)

[Hình 2.15: Bảng cơ sở dữ liệu sqlite\_sequence 44](#_Toc105186817)

[Hình 2.16: Bảng cơ sở dữ liệu tbScore 45](#_Toc105186819)

[Hình 2.17: Bảng cơ sở dữ liệu android\_metadata 45](#_Toc105186821)

[Hình 3.1: Giao diện màn hình chính 46](#_Toc105186825)

[Hình 3.2: Giao diện Menu 48](#_Toc105186827)

[Hình 3.3: Giao diện chọn đề 49](#_Toc105186829)

[Hình 3.4: Giao diện chọn đáp án 49](#_Toc105186831)

[Hình 3.5: Giao diện xem các đáp án đã chọn 50](#_Toc105186833)

[Hình 3.6: Giao diện đáp án đúng 51](#_Toc105186835)

[Hình 3.7: Giao diện đáp án sai 52](#_Toc105186836)

[Hình 3.8: Giao diện xem điểm và sô câu đúng sai 53](#_Toc105186838)

[Hình 3.9: Giao diện lưu điểm 54](#_Toc105186840)

[Hình 3.10: Giao diện tất cả công thức 56](#_Toc105186842)

[Hình 3.11: Giao diện công thức đã tìm 57](#_Toc105186843)

[Hình 3.12: Giao diện danh sách điểm 58](#_Toc105186845)

[Hình 3.13: Giao diện cập nhật thông tin 59](#_Toc105186847)

[Hình 3.13: Giao diện thay đổi mật khẩu 60](#_Toc105186848)

**DANH MỤC BẢNG BIỂU**

[Bảng 2.1: Danh sách các thuộc tính lớp Subject 37](#_Toc105186971)

[Bảng 2.2: Danh sách các phương thức lớp Subject 37](#_Toc105186973)

[Bảng 2.3: Danh sách các thuộc tính lớp Exam 38](#_Toc105186975)

[Bảng 2.4: Danh sách các phương thức lớp Exam 38](#_Toc105186976)

[Bảng 2.5: Danh sách các thuộc tính lớp lớp Save points 38](#_Toc105186978)

[Bảng 2.6: Danh sách các phương thức lớp lớp Save points 38](#_Toc105186979)

[Bảng 2.7: Danh sách các thuộc tính lớp lớp seeTestScores 39](#_Toc105186981)

[Bảng 2.8: Danh sách các phương thức lớp seeTestScores 39](#_Toc105186982)

[Bảng 2.9: Danh sách các thộc tính lớp Formula 39](#_Toc105186984)

[Bảng 2.10: Danh sách các phương thứ lớp Formula 39](#_Toc105186985)

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

* 1. Mục tiêu đề tài
     1. Lý do chọn đề tài

Những năm gần đây, hình thức thi trắc nghiệm rất phổ biến và ngảy càng được ưa chuộng hơn. Nó giúp kiểm tra được kiến thức tổng quát thuộc nhiều khía cạnh, cũng như thời gian chấm thi nhanh hơn. Bởi vậy nhu cầu của mọi người về ôn luyện trắc nghiệm cũng tăng cao, càng ngày càng có nhiều người tìm đến những phần mền thi trắc nghiệm. Đó chính là lý do chúng em chọn đề tài này

Với sự phát triển nhanh chóng của khoa học công nghệ và đặc biệt là ngành công nghệ thông tin, làm từng bước thay đổi cuộc sống của con người bởi các ứng dụng sản phẩm của công nghệ. Đi cùng với sự phát triển của ngành công nghệ thông tin là sự ra đời ngày càng nhiều thiết bị cầm tay, smartphone. Nhu cầu của con người về việc sử dụng và giải trí trên các thiết bị di động cũng ngày càng cao và đa dạng. Một trong những bộ phận ứng dụng công nghệ thông tin chính là thay đổi và giúp đỡ việc sinh hoạt hàng ngày của mọi người trong cuộc sống. Vì vậy em đã lựa chọn đề tài “**Xây dựng phần mềm luyện thi trắc nghiệm trên thiết bị di động**” để làm đồ án với ý nghĩa mang tính thực tế. Với mong muốn người dung

* + 1. Mục tiêu đề tài

- Xây dựng giao diện Menu, giao diện test bài trắc nghiệm  
- Thiết kế giao diện cho phép xử lý các chức năng chính  
- Thiết kế giao diện chương trình chính thực hiện các công việc:  
+ Tổng hợp các câu hỏi với các mốc thời gian 15 -20 phút…  
+ Kiểm tra những câu đã trả lời – chưa trả lời

+ Tự chấm điểm khi thời gian kết thúc & Hiển thị kết quả  
+ Người làm bài xem bài mình làm, thay đổi đáp án nếu mình thấy là sai  
+ Xem danh sách điểm những người làm bài có lưu điểm lại (các bài Test qua)  
+ Tìm kiếm câu hỏi

* + 1. Giới han phạm vi nghiêm cứu

Sử dụng ngôn ngữ Java, phần mền Android Studio, cơ sở dữ liệu SQLite, Firebase.

* + 1. Ý nghĩa khoa học thực tiễn

Phần mềm luyện thi trắc nghiệm giúp các bạn ôn lại những kiến thức tiết kiệm được thời gian ôn tập.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT & THIẾT KẾ HỆ THỐNG

* 1. Giới thiệu hệ điều hành Android 
     1. Android là gì?

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Tổng công ty Android, với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005. Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào tháng 10 năm2008.

Android là mã nguồn mở và Google phát hành mã nguồn theo Giấy phép Apache. Chính mã nguồn mở cùng với một giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên nhiệt huyết được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị, bằng một loại ngôn ngữ lập trình Java có sửa đổi. Vào tháng 10 năm 2012, có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play, cửa hàng ứng dụng chính của Android, ước tính khoảng 25 tỷ lượt.

Những yếu tố này đã giúp Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới, vượt qua Symbian vào quý 4 năm 2010, và được các công ty công nghệ lựa chọn khi họ cần một hệ điều hành không nặng nề, có khả năng tinh chỉnh, và giá rẻ chạy trên các thiết bị công nghệ cao thay vì tạo dựng từ đầu. Kết quả là mặc dù được thiết kế để chạy trên điện thoại và máy tính bảng, Android đã xuất hiện trên TV, máy chơi game và các thiết bị điện tử khác. Bản chất mở của Android cũng khích lệ một đội ngũ đông đảo lập trình viên và những người đam mê sử dụng mã nguồn mở để tạo ra những dự án do cộng đồng quản lý. Những dự án này bổ sung các tính năng cao cấp cho những người dùng thích tìm tòi hoặc đưa Android vào các thiết bị ban đầu chạy hệ điều hành khác.

Android chiếm 75% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 3 năm 2012, với tổng cộng 500 triệu thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Sự thành công của hệ điều hành cũng khiến nó trở thành mục tiêu trong các vụ kiện liên quan đến bằng phát minh.

* + 1. Lịch sử phát triển

Tổng công ty Android (Android, Inc.) được thành lập tại Palo Alto, California vào tháng 10 năm 2003 bởi Andy Rubin (đồng sáng lập công ty Danger), [20] Rich Miner (đồng sáng lập Tổng công ty Viễn thông Wildfire), Nick Sears (từng là Phó giám đốc T-Mobile), và Chris White (trưởng thiết kế và giao diện tại WebTV) để phát triển, theo lời của Rubin, "các thiết bị di động thông minh hơn có thể biết được vị trí và sở thích của người dùng". Dù những người thành lập và nhân viên đều là những người có tiếng tăm, Tổng công ty Android hoạt động một cách âm thầm, chỉ tiết lộ rằng họ đang làm phần mềm dành cho điện thoại di động. Trong năm đó, Rubin hết kinh phí. Steve Perlman, một người bạn thân của Rubin, mang cho ông 10.000 USD tiền mặt nhưng từ chối tham gia vào công ty.

Google mua lại Tổng công ty Android vào ngày 17 tháng 8 năm 2005, biến nó thành một bộ phận trực thuộc Google. Những nhân viên của chủ chốt của Tổng công ty Android, gồm Rubin, Miner và White, vẫn tiếp tục ở lại công ty làm việc sau thương vụ này. Vào thời điểm đó không có nhiều thông tin về Tổng công ty, nhưng nhiều người đồn đoán rằng Google dự tính tham gia thị trường điện thoại di động sau bước đi này. Tại Google, nhóm do Rubin đứng đầu đã phát triển một nền tảng thiết bị di động phát triển trên nền nhân Linux. Google quảng bá nền tảng này cho các nhà sản xuất điện thoại và các nhà mạng với lời hứa sẽ cung cấp một hệ thống uyển chuyển và có khả năng nâng cấp. Google đã liên hệ với hàng loạt hãng phần cứng cũng như đối tác phần mềm, bắn tin cho các nhà mạng rằng họ sẵn sàng hợp tác với các cấp độ khác nhau.

Ngày càng nhiều suy đoán rằng Google sẽ tham gia thị trường điện thoại di động xuất hiện trong tháng 12 năm 2006. Tin tức của BBC và Nhật báo phố Wall chú thích rằng Google muốn đưa công nghệ tìm kiếm và các ứng dụng của họ vào điện thoại di động và họ đang nỗ lực làm việc để thực hiện điều này. Các phương tiện truyền thông truyền thống lẫn online cũng viết về tin đồn rằng Google đang phát triển một thiết bị cầm tay mang thương hiệu Google. Một vài tờ báo còn nói rằng trong khi Google vẫn đang thực hiện những bản mô tả kỹ thuật chi tiết, họ đã trình diễn sản phẩm mẫu cho các nhà sản xuất điện thoại di động và nhà mạng. Tháng 9 năm 2007, InformationWeek đăng tải một nghiên cứu của Evalueserve cho biết Google đã nộp một số đơn xin cấp bằng sáng chế trong lĩnh vực điện thoại di động.

Ngày 5 tháng 11 năm 2007, Liên minh thiết bị cầm tay mở (Open Handset Alliance), một hiệp hội bao gồm nhiều công ty trong đó có Texas Instruments, Tập đoàn Broadcom, Google, HTC, Intel, LG, Tập đoàn Marvell Technology, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel và T-Mobile được thành lập với mục đích phát triển các tiêu chuẩn mở cho thiết bị di động. Cùng ngày, Android cũng được ra mắt với vai trò là sản phẩm đầu tiên của Liên minh, một nền tảng thiết bị di động được xây dựng trên nhân Linux phiên bản 2.6. Chiếc điện thoại chạy Android đầu tiên được bán ra là HTC Dream, phát hành ngày 22 tháng 10 năm 2008. Biểu trưng của hệ điều hành Android mới là một con rôbốt màu xanh lá cây do hãng thiết kế Irina Blok tại California vẽ.

Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng; ví dụ như phiên bản 1.5 Cupcake (bánh bông lan nhỏ có kem) tiếp nối bằng phiên bản 1.6 Donut (bánh vòng). Phiên bản mới nhất hiện nay là 5.0 Lollipop. Vào năm 2010, Google ra mắt loạt thiết bị Nexus-một dòng sản phẩm bao gồm điện thoại thông minh và máy tính bảng chạy hệ điều hành Android, do các đối tác phần cứng sản xuất. HTC đã hợp tác với Google trong chiếc điện thoại thông minh Nexus đầu tiên, Nexus One. Kể từ đó nhiều thiết bị mới hơn đã gia nhập vào dòng sản phẩm này, như điện thoại Nexus 4 và máy tính bảng Nexus 10, lần lượt do LG và Samsung sản xuất. Google xem điện thoại và máy tính bảng Nexus là những thiết bị Android chủ lực của mình, với những tính năng phần cứng và phần mềm mới nhất của Android.

Năm 2014, Google công báo Android Wear, hệ điều hành dành cho các thiết bị đeo được.

* + 1. Cộng đồng mã nguồn mở

Android có một cộng đồng các lập trình viên và những người đam mê rất năng động. Họ sử dụng mã nguồn Android để phát triển và phân phối những phiên bản chỉnh sửa của hệ điều hành. Các bản Android do cộng đồng phát triển thường đem những tính năng và cập nhật mới vào nhanh hơn các kênh chính thức của nhà sản xuất/nhà mạng, tuy không được kiểm thử kỹ lưỡng cũng như không có đảm bảo chất lượng; cung cấp sự hỗ trợ liên tục cho các thiết bị cũ không còn nhận được bản cập nhật chính thức; hoặc mang Android vào những thiết bị ban đầu chạy một hệ điều hành khác, như HP Touchpad. Các bản Android của cộng đồng thường được root sẵn và có những điều chỉnh không phù hợp với những người dùng không rành rẽ,

như khả năng ép xung hoặc tăng/giảm áp bộ xử lý của thiết bị. CyanogenMod là firmware của cộng đồng được sử dụng phổ biến nhất, và hoạt động như một tổ chức của số đông khác.

Trước đây, nhà sản xuất thiết bị và nhà mạng tỏ ra thiếu thiện chí với việc phát triển firmware của bên thứ ba. Những nhà sản xuất còn thể hiện lo ngại rằng các thiết bị chạy phần mềm không chính thức sẽ hoạt động không tốt và dẫn đến tốn tiền hỗ trợ. Hơn nữa, các firmware đã thay đổi như CyanogenMod đôi khi còn cung cấp những tính năng, như truyền tải mạng (tethering), mà người dùng bình thường phải trả tiền nhà mạng mới được sử dụng. Kết quả là nhiều thiết bị bắt đầu đặt ra hàng rào kỹ thuật như khóa bootloader hay hạn chế quyền truy cập root. Tuy nhiên, khi phần mềm do cộng đồng phát triển ngày càng trở nên phổ biến, và sau một thông cáo của Thư viện Quốc hội Hoa Kỳ cho phép "jailbreak" (vượt ngục) thiết bị di động, các nhà sản xuất và nhà mạng đã tỏ ra mềm mỏng hơn với các nhà phát triển thứ ba, thậm chí một số hãng như HTC, Motorola, Samsung và Sony, còn hỗ trợ và khuyến khích phát triển. Kết quả của việc này là dần dần nhu cầu tìm ra các hạn chế phần cứng để cài đặt được firmware không chính thức đã bớt đi do ngày càng nhiều thiết bị được phát hành với bootloader đã mở khóa sẵn hoặc có thể mở khóa, tương tự như điện thoại dòng Nexus, tuy rằng thông thường họ sẽ yêu cầu người dùng từ bỏ chế độ bảo hành nếu họ làm như vậy. Tuy nhiên, tuy được sự chấp thuận của nhà sản xuất, một số nhà mạng tại Mỹ vẫn bắt buộc điện thoại phải bị khóa.

Việc mở khóa và "hack" điện thoại thông minh và máy tính bảng vẫn còn là tác nhân gây căng thẳng giữa cộng đồng và công nghiệp. Cộng đồng luôn biện hộ rằng sự hỗ trợ không chính thức ngày càng trở nên quan trọng trước việc nền công nghiệp không cung cấp các bản cập nhật thường xuyên và/hoặc ngưng hỗ trợ cho chính các thiết bị của họ.

* + 1. Kiến trúc hệ điều hành Android

****

##### Hình 2.1: Kiến trúc hệ điều hành Android

**Linux Kernel**

Dưới cùng là lớp Linux – Linux 3.6 cùng với khoảng 115 bản vá. Lớp này cung cấp 1 cấp độ trừu tượng giữa phần cứng của thiết bị và các thành trình điều khiển phần cứng thiết yếu như máy ảnh, bàn phím, màn hình hiển thị… Đồng thời, hạt nhân (kernel) còn xử lý tất cả các thứ mà Linux có thể làm tốt như mạng kết nối và 1 chuỗi các trình điều khiển thiết bị, giúp cho giao tiếp với các thiết bị ngoại vi dễ dàng hơn.

**Các thư viện**

Ở trên lớp nhân Linux là tập các thư viện bao gồm WebKit – trình duyệt Web mã nguồn mở, được biết đến như thư viện libc, cơ sở dữ liệu SQLite – hữu dụng cho việc lưu trữ và chia sẻ dữ liệu ứng dụng, các thư viênj chơi và ghi âm audio, video, hay các thư viện SSL chiụ trách nhiệm bảo mật Internet…

**Các thư viện Android**

Đây là các thư viện dựa trên Java phục vụ cho việc phát triển Android. Ví dụ của các thư viện này bao gồm các thư viện ứng dụng dùng để xây dựng giao diện người dùng, vẽ đồ họa hay truy cập cơ sở dữ liệu. 1 số thư viện chính của Android:

* android.app – Cung cấp quyền truy cập vào ứng dụng và là nền tảng của tất cả ứng dụng Android.
* android.content – Cung cấp quyền truy cập nội dung (content), truyền tải thông điệp giữa các ứng dụng hay các thành phần của ứng dụng.
* android.database – Đựoc sử dụng để truy cập dữ liệu của content provider và cơ sở dữ liệu SQLite
* android.opengl – giao diện các phuơng thức Java để sử dụng OpenGL ES
* android.os – Cung cấp các ứng dụng với quyền truy cập vào các dịch vụ của hệ điều hành bao gồm thông điệp, các dịch vụ hệ thống và các giao tiếp nội tại
* android.text – Đựoc sử dụng để hiển thị và điều chỉnh chữ trên màn hình thiết bị
* android.view – Các thành phần cơ bản trong việc xây dựng giao diện người dùng của ứng dụng.
* android.widget – Tập các thành phần giao diện người dùng đã được xây dựng sẵn như các nút, các nhãn (label), list view,….
* android.webkit – Tập các lớp cho phép xây dựng khả năng duyệt web.

**Android Runtime**

Đây là phần thứ 3 của kiến trúc và nằm ở lớp thứ 2 từ dưới lên. Phần này cung cấp 1 bộ phận quan trọng là Dalvik Vỉtual Machine – là 1 loại Java Virtual Machine được thiết kế đặc biệt để tối ưu cho Android.

Dalvik VM sử dụng các đặc trưng của nhân Linux như quản lý bộ nhớ và đa luồng, những thứ mà đã có sẵn trong Java. Dalvik VM giúp mọ ứng dụng Android chạy trong tiến trình riêng của nó, với các thể hiện (instance) riêng của Dalvik virtual Machine.

Android Runtime cũng cung cấp 1 tập các thư viện chính giúp các nhà phát triển ứng dụng Android có thể viết ứng dụng Android bằng Java

**Application Framework**

Lớp Android Framework cung cấp các dịch vụ cấp độ cao hơn cho các ứng dụng dưới dạng các lớp Java. Các nhà phát triển ứng dụng được phép sử dụng các dịch vụ này trong ứng dụng của họ.

Android Framework bao gồm các dịch vụ chính sau:

* Activitty Manager – Kiểm soát tất cả khía cạnh của vòng đời ứng dụng và ngăn xếp các Activity.
* Content Providers – Cho phép các ứng dụng chia sẽ dữ liệu với các ứng dụng khác.
* Resource Manager – Cung cấp quyền truy cập vào các tài nguyên như các chuỗi, màu sắc, các layout giao diện người dùng…
* Notifications Manager – Cho phép các ứng dụng hiển thị cảnh báo và các thông báo cho người dùng.
* View System – Tập các thành phần giao diện (view) được sử dụng để tạo giao diện người dùng.

**Application**

Lớp trên cùng của kiến trúc là Application. Các ứng dụng bạn tạo ra sẽ được cài đặt trên lớp này. Ví dụ như: Danh bạ, nhắn tin, trò chơi…

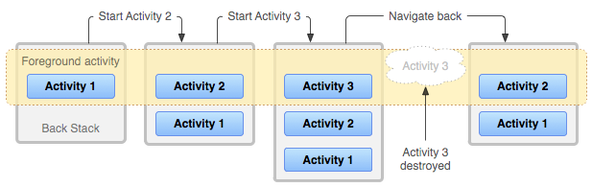
* 1. Các thành phần trong 1 ứng dụng android

Một ứng dụng Android được tạo thành bởi một trong các Thành phần (trong kỹ thuật ta gọi là Application Components). Mỗi thành phần cung cấp một cách thức khác nhau để hệ điều hành làm việc với ứng dụng. Mỗi thành phần tồn tại như một thực thể riêng và đóng một vai trò nhất định giúp xác định hành vi tổng thể của ứng dụng của bạn. Có 4 loại thành phần khác nhau trong Android:

* + Activities
  + Services
  + Content Providers
  + Broadcast Receivers

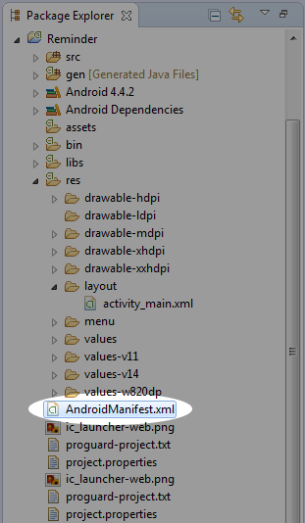
Ba trong bốn thành phần kể trên đây: Activities, Services và Broadcast Receiver được thực thi bởi một Intent. Intent là một thông điệp bất đồng bộ mà ta sẽ khảo sát ngay sau đây. Một file Manifest dùng để khai báo với hệ thống các thành phần có thể có và khai báo các yêu cầu của ứng dụng. Manifest cũng sẽ được nói cụ thể hơn ở phần sau. Cuối cùng, ứng dụng Android không thể thiếu Resources,  resources là phẩn tách biệt với source code của ứng dụng, cho phép ứng dụng sử dụng các hình ảnh, âm thanh, font chữ,… làm phong phú thêm các yêu cầu của ứng dụng. Chúng ta sẽ làm rõ từng phần kể trên ở các định nghĩa sau đây.

* + 1. Activities
* Activity đại diện cho một màn hình duy nhất với một giao diện người dùng mà ở đó người dùng có thể quan sát và tương tác. Ví dụ như một ứng dụng email có một Activity hiển thị danh sách các email, một Activity khác cung cấp một giao diện để soạn thảo email, và một Activity khác nữa cung cấp một giao diện để đọc email,…
* Mặc dù các Activity này liên kết chặt chẽ với nhau trong ứng dụng email như ví dụ trên nhằm đem lại trải nghiệm tốt nhất cho người dùng nhưng thực tế chúng hoàn toàn độc lập với nhau.
* Các ứng dụng khác (có thể không phải ứng dụng của ta xây dựng) hoàn toàn có thể khởi chạy các Activity mà chúng ta định nghĩa, dĩ nhiên trong trường hợp ứng dụng email của chúng ta cho phép.
* Mỗi một ứng dụng đều có một “main activity”, đây là Activity sẽ được hiển thị đầu tiên khi ứng dụng được khởi chạy, một ứng dụng sẽ có một main activity và thông thường sẽ phải có thêm nhiều Activity khác.
* Một Activity có thể khởi chạy các Activity khác. Mỗi lần một Activity mới được khởi chạy thì Activity trước đó bị stop và hệ thống sẽ lưu Activity stop này vào một back stack. Việc quay lại các Activity trước đó (khi người dùng nhấn vào nút back trên thiết bị chẳng hạn) thì hệ thống chỉ đơn giản lấy các Activity đã lưu trữ trong back stack, và Activity được lấy trong back stack này tiếp tục sẽ được hiển thị lên màn hình người dùng. Back stack kể trên được tổ chức theo đúng tiêu chí của một stack, đó là “vào sau, ra trước” (last in, first out), chúng ta có thể tìm hiểu thêm về stack bằng cách search trên Google hoặc ở nguồn này <http://en.wikipedia.org/wiki/Stack_(abstract_data_type)>.
* Để hình dung rõ ràng hơn về việc back stack làm việc với các Activity như thế nào, ta hãy xem sơ đồ sau.

[](https://gioilaptrinh.files.wordpress.com/2014/10/diagram_backstack.png)

##### Hình 2.2: Vòng đời hoạt động của Android

* Ở sơ đồ trên, khi ứng dụng được khởi chạy, một main activity sẽ được kích hoạt ban đầu, ví dụ như trên ta thấy main activity chính là Activity 1, Activity 1 sẽ được tự động chứa trong back stack và đang ở dạng foreground, tức là đang được hiển thị lên màn hình. Sau đó Activity 1 này sẽ start một Activity khác (do hành vi người dùng nhấn vào một button nào đó, hoặc Activity 1 tự động start Activity 2 theo timer được định sẵn theo kịch bản chẳng hạn), như hình ta thấy Activity 2 được hiển thị ở foreground, tức là lúc này Activity 2 sẽ thay Activity 1 hiển thị lên màn hình, Activity 1 sẽ không mất đi mà chỉ tạm stop lại rồi được lưu giữ ở back stack. Sau đó Activity 2 lại start Activity 3, và cũng như trên, Activity 3 sẽ giữ vai trò foreground, còn Activity 2 và Activity 1 thì tiếp tục được lưu giữa trong back stack. Đến lúc này nếu người dùng nhấn nút back trên thiết bị, thì Activity 3 sẽ bị lấy ra khỏi stack, chính lúc này Activity 3 mới bị destroy, tức là bị hủy luôn không thuộc diện quản lý của back stack, vai trò foreground hiển thị lên màn hình được nhường lại cho Activity kế đó bị đưa vào back stack lúc trước đó là Activity 2. Cứ như vậy nếu người dùng tiếp tục nhấn nút back thì Activity 1 lại tiếp tục được hiển thị ra foregroud còn Activity 2 bị hủy. Nếu người dùng lại tiếp tục nhấn back nữa thì Activity 1 cũng bị hủy khỏi back stack, lúc này ứng dụng xem như bị đóng lại (ngừng hoạt động). Để hiểu rõ hơn hoạt động của back stack, có thể xem thêm thông tin ở trang <http://developer.android.com/guide/components/tasks-and-back-stack.html>.
  + 1. Services
* Một Service là một thành phần chạy ngầm, để thực hiện các hoạt động lâu dài hoặc để thực hiện công việc cho các xử lý từ xa (remote processes).
* Service không cung cấp giao diện người dùng. Ví dụ một service có thể chơi nhạc trong khi người dùng đang sử dụng một ứng dụng khác.
* Một thành phần (application component) khác chẳng hạn như một activity có thể khởi động một service để nó chạy độc lập hoặc liên kết để tương tác với nó.
* Một service về cơ bản có 2 loại:
  + Started:
    - Khi được khởi động lên, thì service loại này sẽ hoạt động ở nền (background) vô thời hạn, dù cho thành phần khởi chạy nó đã kết thúc.
    - Thông thường một started service chỉ được giao cho một xử lý đơn lẻ và không trả kết quả về cho thành phần đã gọi nó.
  + Bound:
    - Một bound service cung cấp một giao diện client-server để cho phép các thành phần có thể tương tác được với nhau.
    - Thời gian sống của một bound service phụ thuộc vào thành phần liên kết (bound) với nó, tức là thành phần liên kết còn tồn tại thì bound service còn tồn tại và ngược lại.
    - Nhiều thành phần có thể liên kết với một bound service tại một thời điểm, nhưng khi tất cả các thành phần này đều ngưng liên kết (unbind) thì service loại này cũng bị hủy theo.
* Service sẽ được đề cập chi tiết hơn ở các bài học buổi khác.
  + 1. Content Providers
* Content Provider quản lý một tập hợp các dữ liệu được chia sẻ của ứng dụng. Thông qua Content Provider, các chương trình khác có thể truy xuất hoặc thậm chí thay đổi các dữ liệu của một ứng dụng (nếu Content Provider cho phép).
* Ví dụ hệ điều hành Android cung cấp một Content Provider quản lý danh bạ điện thoại. Thông qua Content Provider này mà các ứng dụng (có các quyền thích hợp) có thể đọc hoặc thêm thông tin vào danh bạ điện thoại này.
* Content Provider dĩ nhiên còn hữu ích cho cả việc đọc và ghi dữ liệu private của ứng dụng và không chia sẻ ra ngoài.
  + 1. Broadcast Receivers
* Broadcast Reveiver là một thành phần dùng để trả lời các thông điệp được phát ra từ hệ thống.
* Trong quá trình thiết bị di động vận hành có rất nhiều thông điệp khác nhau được hệ thống thông báo như: màn hình đã tắt, pin yếu, đã chụp xong một tấm hình,…
* Một Broadcast Receiver không tạo một giao diện người dùng nhưng nó có thể tạo một thông báo (notification) trên thanh trạng thái (status bar) để báo động người dùng khi một sự kiện nào đó xảy ra.
  + 1. Intent
* Ba trong số bốn loại thành phần của ứng dụng (activity, service, broadcast receiver) đều được khởi chạy bởi một lời gọi bất đồng bộ, ta gọi là Intent.
* Intent tạo nên một cơ sở gắn kết giữa các thành phần trong ứng dụng lại với nhau.
* Intent cũng sẽ được đề cập chi tiết hơn nữa ở bài sau.
  + 1. File Manifest
* File Manifest là nơi ứng dụng của bạn sẽ trình bày với hệ thống rằng “có những gì trong ứng dụng này”, như:
  + Đăng ký các user permission, tạm hiểu là các quyền hạn mà người dùng xin phép để sử dụng các tính năng của hệ thống, như là quyền được truy cập Interner, được đọc danh bạ (contacts), được ghi file ra thẻ nhớ ngoài,…
  + Chỉ định Minimum Required SDK, là hệ điều hành thấp nhất của Android mà ứng dụng của ta hỗ trợ, thông số này được tạo ra khi chúng ta tạo một ứng dụng mới ở Bài 2, khi đó chúng ta đã chỉ ra Minimum Requied SDK và thông số chúng ta chỉ định đó sẽ được lưu vào thuộc tính này của file Manifest.
  + Chỉ định các tính năng phần cứng và phần mềm mà ứng dụng của ta muốn sử dụng đến, như camera, bluetooth, hay ngay cả tính năng multitouch.
  + Một số các thư viện bên ngoài (bên ngoài ở đây có thể hiểu là các thư viện không có sẵn trong các API Framework của chúng ta) mà ứng dụng của ta có sử dụng đến, như thư viện Google Maps, Google Play,…
* Khi tạo mới một ứng dụng, ADT sẽ tự động tạo cho chúng ta một file AndroidManifest.xml, chúng ta có thể làm theo các bước sau để xem cụ thể một file Manifest sẽ trông như thế nào.
* Trong Eclipse, tìm đến cửa sổ Package Explorer, tìm đến project Reminder của chúng ta đã tạo trước đó, ta tìm file Manifest trong cây thư mục của project Reminder này như hình sau, click đúp vào file Manifest để mở nó lên (lưu ý là click đúp, vì một số bạn nếu đã lập trình iOS trước đó sẽ quen với việc click đơn để mở một file trong Xcode, trong Eclipse thì khác, khi muốn mở một file ta phải click đúp).

[](https://gioilaptrinh.files.wordpress.com/2014/10/manifest.png)

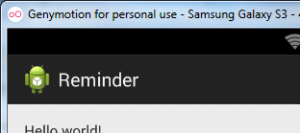
##### Hình 2.3: Thư mục Android Manifest

* Bất cứ một file xml nào, kể cả file Manifest.xml lẫn các file resource xml cũng đều mặc định ban đầu được mở lên với một màn hình có nhiều tab bên dưới, mục đích của màn hình này là sẽ cung cấp cho chúng ta có được nhiều tùy chọn hơn trong việc review và tùy chỉnh nội dung của file xml. Thường thì các tab đầu sẽ cung cấp một số các giao diện chỉnh sửa, còn tab cuối sẽ là các đoạn code xml cho chúng ta chỉnh sửa thủ công bằng tay. Trong các bài này và các bài sau chúng ta tập chung chủ yếu vào việc review và chỉnh sửa các file xml một cách thủ công bằng tay, vừa chính xác lại dễ quản lý sau này, hơn là các giao diện tự động sẽ làm rối file xml của chúng ta hơn nữa, bởi vì Eclipse không phải là một công cụ chuyên biệt cho Android, nên các giao diện tự động của Eclipse (có cả của ADT nhúng vào) sẽ không đảm bảo và dễ gây ra các lỗi ngớ ngẩn khó chịu sau này.
* Như đã nói ở trên, chúng ta nhìn vào dãy tab bên dưới và click vào tab AndroidManifest.xml để mở code xml ra review. Tôi khuyên bạn nên chọn lần lượt từng tab và có cái nhìn tổng quan trước mỗi tab như thế nào, vì tùy thuộc mỗi người sẽ chọn cho mình một phong cách làm việc sau này, đối với tôi thì tôi thích sửa code xml, nhưng bạn có thể sử dụng các cách thức khác hiệu quả hơn đối với bản thân bạn thì càng tốt.

[](https://gioilaptrinh.files.wordpress.com/2014/10/manifest2.png)

##### Hình 2.4: File AndroidManifest.xml

* Dưới đây sẽ giải thích một số ý nghĩa và cách sử dụng của các tag trong AndroidManifest.xml
  + <manifest> là thẻ gốc. Thuộc tính quan trọng mà bạn cần quan tâm trong thẻ này là package. Cũng như Minimum Required SDK, package xuất hiện sau khi bạn tạo một ứng dụng mới ở Bài 2, khi đó bạn đã khai báo một package name và cái tên chúng ta đã chỉ chỉ định khi đó sẽ được lưu vào trường này của file Manifest. Ngoài ý nghĩa của package name đã được giải thích ở Bài 2, package còn chỉ ra đây là tên gốc của ứng dụng, để đến khi bạn cần cung cấp một class mới ở bất cứ nơi nào khác trong file Manifest này, bạn chỉ việc đặt một dấu chấm (.) đằng trước thay cho cả đường dẫn của package gốc. Như ví dụ trên thay vì cung cấp tên đầy đù cho MainActivity là android:name=”com.example.reminder.MainActivity” thì ta chỉ cần ghi vắn tắt là android:name=”.MainActivity”.
  + <uses-sdk> xác định phiên bản hệ điều hành để chạy được chương trình. Cũng như các thuộc tính trên, thuộc tính của trường này cũng được tạo ra sau khi ta chỉ định các giá trị Minimum Requires SDK và Target SDK khi tạo project mới ở Bài 2.
  + <application> là thẻ quan trọng nhất trong file này. Nó chỉ định ra các thành phần có trong một ứng dụng.
    - android:icon là thuộc tính chỉ định hình ảnh dùng làm icon cho ứng dụng của chúng ta, bạn có thể thay đổi đường dẫn cho icon của ứng dụng bằng cách thay đổi tên hình trong trường này.
    - android:label là thuộc tính chỉ ra dòng text nào sẽ được dùng làm tên của ứng dụng, cả 2 thuộc tính kể trên đây đều tùy thuộc vào bạn tạo mới một ứng dụng ở Bài 2, khi bạn chỉ định một Application Name và một icon cho nó.
  + <activity> khai báo một Activity, nó bao gồm các thuộc tính cơ bản sau:
    - android:name khai báo tên và đường dẫn đến một Activity, tất nhiên đường dẫn này chính là package mà Activity đó nằm trong, và chúng ta có thể không cần khai báo full package name mà chỉ cần đại diện bằng dấu (.) nếu package name của Activity này trùng với trường package đã khai báo ở tag <manifest> như đã nói ở trên.
    - android:label chỉ định tên cho Activity, tên này sẽ xuất hiện trên title bar của ứng dụng khi Activity này hiển thị, như hình bên dưới.

[](https://gioilaptrinh.files.wordpress.com/2014/10/title.png)

##### Hình 2.5: Ví dụ về Label

* + <intent-filter> xác định xem trong điều kiện khi nào thì Activity mà thẻ này đang nằm trong đó được hiển thị. Như hình trên kia, thẻ này xác định Activity chứa nó là Activity chính của chương trình và sẽ là Activity xuất hiện đầu tiên khi người dùng start ứng dụng.
* Dĩ nhiên những gì đã đề cập trên đây chỉ là một phần của file Manifest, vì được dùng để định nghĩa tất cả những gì có trong ứng dụng nên file Manifest rất quan trọng, sẽ được đề cập nhiều hơn nữa ở bài sau.
  + 1. Resources
* Một ứng dụng không thể thiếu các resource, đó là các hình ảnh, font chữ, màu sắc, âm thanh… dùng để tăng thêm sự phong phú và tính thẩm mỹ cho ứng dụng. Ở bài tổng quan này chúng ta tìm hiểu một các sơ lược về resource, cụ thể như thế nào sẽ được làm rõ hơn trong các bài sau.
* Thông thường khái niệm resource trong Android được hiểu như là một cái gì đó tách biệt với source code, source code là nơi chứa đựng java code và các logic của ứng dụng, trong khi resource là những gì còn lại của ứng dụng không phải là source code.
* Resource được chứa đựng trong thư mục res/ (bạn có thể nhìn vào Package Explorer để thấy thư mục Resource này), trong thư mục này còn có rất nhiều thư mục khác giúp nhóm các resource theo loại và theo các tính năng mà resource đó hỗ trợ.
* Cụ thể từng loại và tính năng của resource được nói rõ hơn ở bài sau, ở đây ta chỉ nhắc đến 2 loại resource lớn.
  + Default resources: là tất cả các resource mà chúng ta không cần quan tâm đến việc hiển thị khác nhau như thế nào trong các thiết bị, resource này sẽ hiển thị default như những gì nó được cung cấp ở tất cả các thiết bị.
  + Alternative resources: là các resosurce sẽ phụ thuộc vào cụ thể thiết bị mà ta đã chỉ định, chẳng hạn ta muốn ứng dụng chạy trên thiết bị có ngôn ngữ mặc định là tiếng Việt sẽ phải load các text tiếng Việt, hình ảnh cờ Việt Nam, còn nếu ứng dụng chạy trên thiết bị có ngôn ngữ mặc định là tiếng Anh sẽ load các text tiếng Anh, hình ảnh cờ nước Anh,…
  1. Các thành phần giao diện trong Android
     1. View

**View**là gì? View đơn giản là một thành phần hiển thị trên màn hình.

* Một cái nút (Button) là một View.
* Một cái thanh kéo (SeekBar) cũng là một “View”: Vậy chúng ta có thể phán chắc nịch luôn: **Cái gì hiển thị trên màn hình thì nó là View**.

Trong Android có rất nhiều loại **View**, phù hợp nhiều nhu cầu. Bản thân lập trình viên cũng có thể tự tạo ra view của riêng mình bằng cách code các lớp Java *extends* từ lớp **View**. Tí nữa ở cuối bài biết đâu sẽ có ví dụ, hí hí. Một số loại **View**tiêu biểu là (mình lấy ví dụ ở bản Android 6.0 mới):

* **TextView**: Hiển thị chữ. Có thể thay đổi nội dung chữ, màu sắc, kích cỡ, kiểu đậm/gạch chân,…
* **Button**: Nút. Có thể thay đổi độ rộng, cao, màu nền nút, kiểu viền, đổ bóng,…
* **Switch**: Công tắc. Có 2 chế độ gạt qua gạt lại như chơi điện tử, nhưng mà chỉ trái-phải thôi.
* **EditText**: Trường nhập văn bản. Là một ô nhập văn bản. Văn bản ở EditText có thể là 1 hay nhiều dòng, có thể cài đặt đầu vào chỉ được nhập số, nhập chữ hay nhập dạng mật khẩu.
* **SeekBar**: Thanh kéo. Sử dụng khi người dùng muốn thao tác gì đó về biên độ / độ lớn bằng tay. Các View có những thuộc tính khác nhau, gọi là các attributes. Ví dụ như **EditText**thì có attributes là maxLength (số ký tự tối đa) và maxLines (số dòng tối đa). **TextView**thì có fontSize (cỡ chữ) và **inputType**(kiểu nhập: số / chữ / email,…). Tuy nhiên các View đều có một số attributes chung như:
* **Layout\_width**: Độ dài chiều ngang của View. Có thể là con số cụ thể hoặc là wrap\_content (Bao đủ nội dung bên trong thì thôi), hay match\_parent (bằng với chiều ngang của View chứa nó).
* **Layout\_height**: Độ dài chiều dọc của View. Có thể là con số cụ thể hoặc là wrap\_content (Bao đủ nội dung bên trong thì thôi), hay match\_parent (bằng với chiều dọc của View chứa nó).
* **ID**: Cái này rất quan trọng, mỗi một View trong một layout đều được gắn với một định danh (id) này để gắn chức năng với code Java ở trong. Sẽ được đề cập kỹ hơn ở phần sau.
  + 1. ViewGroup

ViewGroup là một lớp con của View và nó sẽ hoạt động như một lớp cơ sở cho **layouts**và **layouts parameters**. ViewGroup sẽ cung cấp một **bộ chứa (container)** vô hình để chứa các **View**hoặc **ViewGroup**khác và để xác định các thuộc tính layout.

**Ví dụ:** Linear Layout là ViewGroup chứa các UI control như button, textview, v.v. và các layout khác.

Sau đây là một số lớp con phổ biến của ViewGroup thường được sử dụng trong Android.

* Linear Layout
* Relative Layout
* Table Layout
* Frame Layout
* Web View
* List View
* Grid View

Lớp con của View và ViewGroup đều đóng vai trò quan trọng để tạo ra layout trong ứng dụng Android.

Về cơ bản các thành phần của View và View Group sẽ được trình bày trong series này mình đã liệt kê ở trên.

* + 1. Button

Button là view được sử dụng khá nhiều trong android, hầu như sử dụng ở mọi nơi cùng với EditText, TextView. Button có chức năng là làm nhiệm vụ nào đó khi mà người dùng click trong phương thức onClick.

Một số thuộc tính của Button

android:id: Xác định id cho Button

android:text: Xác định văn bản sẽ hiển thị lên Button

android:textColor: Xác định màu văn bản của Button.

android:background: Xác định màu nền của Button

Để thực hiện những khối lệnh khi chúng ta click vào button, chúng ta làm như sau:

Bước 1: Lấy về đối tượng Buton thông qua id được khai báo button ở file xml

Button btnClick = (Button)findViewById(R.id.btn\_click);

Bước 2: Set bộ lắng nghe sự kiện cho Button thông qua phương thức setOnClickListener và handle các công việc cần xữ lý trong phương thức onClick

btnClick.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

}

});

* + 1. ImageView

ImageView là một view sử dụng rất nhiều trong ứng dụng android, ImageView sử dụng để hiển thị hình ảnh.

Một số thuộc tính phổ biến của ImageView

android:id: Xác định id.

android:src: Xác định source hình ảnh hoặc drawable

* + 1. ListView

**ListView trong Android** là một view group, hiển thị các item theo một danh sách có thể cuộn được theo chiều thẳng đứng.

Một số thuộc tính phổ biến của ImageView

android:id: ID là duy nhất để nhận diện Layout

android:divider: Nó có thể vẽ hoặc tô màu giữa các item trong danh sách.

android:dividerHeight: Nó xác định chiều cao của dải phân cách. Điều này có thể là px, dp, sp, in hoặc mm.

android:entries: Chỉ định tham chiếu đến một tài nguyên mảng sẽ điền vào ListView.

android:footerDividersEnabled: Khi được đặt thành false, ListView sẽ không vẽ dải phân cách trước mỗi chế độ xem chân trang. Giá trị mặc định là true.

android:headerDividersEnabled: Khi được đặt thành false, ListView sẽ không vẽ dải phân cách sau mỗi chế độ xem tiêu đề. Giá trị mặc định là true.

* + 1. TextView

TextView là view sử dụng để hiển thị text màn hình. TextView được định nghĩa bởi

**android:textColor**: Xác định màu chử của TextView.

**android:textSize:** Xác định size chử của TextView

**android:textStyle**: Xác định style chữ TextView, có ba giá trị là normal, bold (in đậm), ilalic (nghiêng).

**android:background**: Xác định màu nền TextView.

**android:drawableLeft:** Xác định drawable (với những người mới bắt đầu thì có thể xem như là hình ảnh) nằm bên trái text.

Tương tự chúng ta có các thuộc tính **android:drawableRight,** **android:drawableTop**, **android:drawableBottom** nằm bên phải, trên, dưới text.

* + 1. EditText

**EditText** là view dùng để lấy giá trị từ người dùng nhập vào. EditText được định nghĩa bởi thẻ <EditText> trong xml.

Một số thuộc tính của EditText

**android:text**: Xác định text hiển thị lên EditText

**android:textColor**: Xác định màu của text.

**android:textSize**: Xác định kích thước của text.

**android:textStyle**: Xác định style của text gồm các giá trị italic (nghiêng), bold (in đậm), normal (kiểu thường).

**android:inputType:** Xác định phương thức nhập của Edittext, có các giá trị như sau: text, number, textPassword, phone, textUrl…

* + 1. CheckBox

Android:checked thiết lập trạng thái checked, unchecked với giá trị "true" hay "false" android:button để gán Drawable vào View (vẽ trạng thái cho CheckBox, RadioButton ...) android:buttonTint để gán màu Tint

* 1. Phân tích & thiết kế hệ thống
     1. Sơ sơ đồ phân cấp chức năng phía frontend

##### Hình 2.6: Sơ đồ phân cấp chức năng phía frontend

Chi tiết chức năng phía frontend

1. Xem thông tin: Người dùng có thể lựa chọn xem thông tin điểm của bản thân, xem công thức của tất cả môn học hay của từng môn ở mục công thức hay có thể xem thông tin tài khoản của bản thân khi bấm vào thanh menu trong App.
2. Môn học: Sauk hi mở App sẽ hiển thị danh sách các môn học, người dung có thể chọn các môn học để bắt đầu kiểm tra và chọn mã đề theo ý thích.
3. Người dùng có thể tìm kiếm công thức các môn học hay các loại lý thuyết ở mục công thức.
4. Quản lý tài khoản: Người dùng có thể cập nhật lại tên tài khoản, ảnh đại diện cũng như thay đổi mật khẩu.
   * 1. Sơ đồ phân cấp chức năng phía Backend

##### Hình 2.7: Sơ đồ phân cấp chức năng phía backend

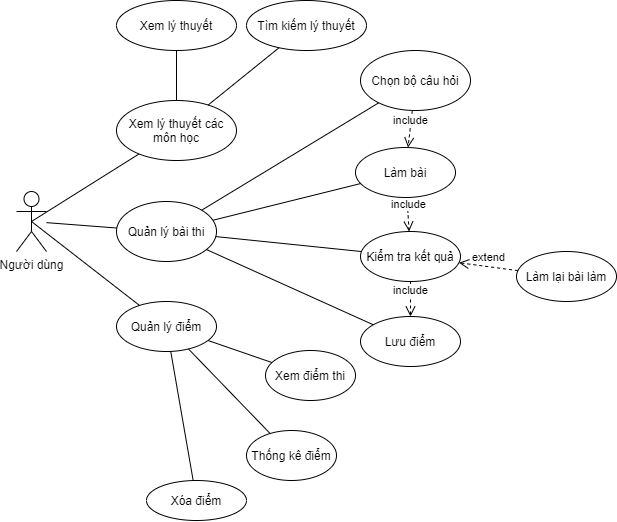
**Chi tiết chức năng phía Backend**

1. Quản trị hệ thống: Quản lý tài khoản của người dùng, cho phép đăng ký hoặc đăng nhập khi có tài khoản bằng email và password.
2. Quản lý danh mục:

* Môn học: Xem và chọn các môn học
* Điểm: Xem danh sách điểm đã lưu của bản thân sau những lần làm bài.
* Công thức: Xem công thức & lý thuyết theo từng môn, tìm kiếm công thức cần tìm.
* Tài khoản: Xem thông tin tài khoản

1. Quản lý nghiệp vụ:

* Chỉnh sửa thông tin: Người dùng chỉnh sửa tên, ảnh đại diện, email cũng như mật khẩu của tài khoản.
* Chọn môn thi: Chọn môn thi sau đó chọn đề thi và bắt đầu làm bài kiểm tra
* Lưu kết quả: Sau mỗi lần thi, người dùng có thể lựa chọn lưu kết quả của bản thân mình với điểm và các cấp độ đạt được sau mỗi bài kiểm tra.
  + 1. Biểu đồ Use – Case tổng quát



##### Hình 2.8: Sơ đồ UseCase

**Đặc tả UseCase**

|  |  |
| --- | --- |
| **Tác nhân** | Người dùng |
| **Mô tả:** tác nhân sử dụng UseCase này để thực hiện các chức năng của App như xem lý thuyết các môn học, quản lý bài thi và quản lý điểm | |
| **Dòng sự kiện chính:**   1. Dòng thứ 1: Tác nhân chọn chức năng xem lý thuyết các môn học để xem lý thuyết và công thức muốn xem 2. Dòng thứ 2: Tác nhân chọn bài thi để chọn gói câu hỏi và làm bài 3. Dòng thứ 3: Tác nhân chọn xem điểm để xem và quản lý điểm của bản thân sau các lần kiểm tra | |
| **Dòng sự kiện phụ:**   1. Dòng thứ 1: Tác nhân xem lý thuyết và tìm kiếm lý thuyết 2. Dòng thứ 2: Tác nhân có thể chọn làm lại bài, lưu điểm hoặc thoát sau khi kiểm tra xong 3. Dòng thứ 3: Tác nhân xem điểm thi, thông kê điểm thi hoặc xóa điểm thi | |
| **Các yêu cầu đặc biệt:** | Không có |
| * **Trạng thái hệ thống trước khi UseCase được sử dụng:** Tác nhân phải đăng nhập * **Trạng thái hệ thống sau khi UseCase được sử dụng:**   **Nếu thành công:** Tác nhân có thể sử dụng được các chức năng của hệ thống  **Nếu thất bại:** Hệ thống báo lỗi, đóng app | |
| **Điểm mở rộng** |  |
| **Tần suất sử dụng** | Thường xuyên |

* + 1. Biểu đồ lớp
       1. Danh sách lớp Subject

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | buttonFinish\_Click() | Chọn môn học |
| 2 | chooseNumExam() | Chọn đề thi |

##### Bảng 2.1: Danh sách các thuộc tính lớp Subject

* + - 1. Danh sách lớp ChooseNumExam

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | getCount() | Lấy danh sách mã đề |
| 2 | getItem() | Lấy môn học |
| 3 | getItemId() | Lấy mã đề |
| 4 | getView() | Lấy giao diện bài thi |

##### Bảng 2.2: Danh sách các phương thức lớp Subject

* + - 1. Danh sách lớp Exam

Danh sách các thuộc tính

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Ý nghĩa** |
| 1 | id | Id câu hỏi |
| 2 | question | Câu hỏi |
| 3 | ans-a | Đáp án a |
| 4 | ans-b | Đáp án b |
| 5 | ans-c | Đáp án c |
| 6 | ans-d | Đáp án d |
| 7 | Result | Kết quả câu hỏi |
| 8 | num-exam | Mã đề thi |
| 9 | subject | Môn học |
| 10 | image | ảnh trong câu hỏi |

##### Bảng 2.3: Danh sách các thuộc tính lớp Exam

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | getAnswer() | Lấy câu hỏi đã trả lời |
| 2 | setAnswer | Sửa câu hỏi đã trả lời |
| 3 | getLogo() | Lấy ảnh của câu hỏi |

##### Bảng 2.4: Danh sách các phương thức lớp Exam

* + - 1. Danh sách lớp Save points

Danh sách các thuộc tính

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Ý nghĩa** |
| 1 | id | Id điểm |
| 2 | name | Tên người làm bài |
| 3 | score | Số điểm |
| 4 | date | Ngày làm bài kiểm tra |

##### Bảng 2.5: Danh sách các thuộc tính lớp lớp Save points

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | addPoints() | Thêm điểm vào cơ sở dữ liệu |

##### Bảng 2.6: Danh sách các phương thức lớp lớp Save points

* + - 1. Danh sách lớp seeTestScores

Danh sách các thuộc tính

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Ý nghĩa** |
| 1 | id | Id điểm |
| 2 | name | Tên người làm bài |
| 3 | score | Số điểm |
| 4 | rum | Số điểm / tổng câu |

##### Bảng 2.7: Danh sách các thuộc tính lớp lớp seeTestScores

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | getId() | Lấy id điểm |
| 2 | getName() | Lấy tên người làm bài |
| 3 | setName() | Sửa tên người làm bài |
| 4 | getScore() | Lấy điểm bài thi |

##### Bảng 2.8: Danh sách các phương thức lớp seeTestScores

* + - 1. Danh sách lớp Formula

Danh sách các thuộc tính

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên thuộc tính** | **Ý nghĩa** |
| 1 | id | Id công thức |
| 2 | formula | Công thức |
| 3 | subject | Môn học muốn xem công thức |

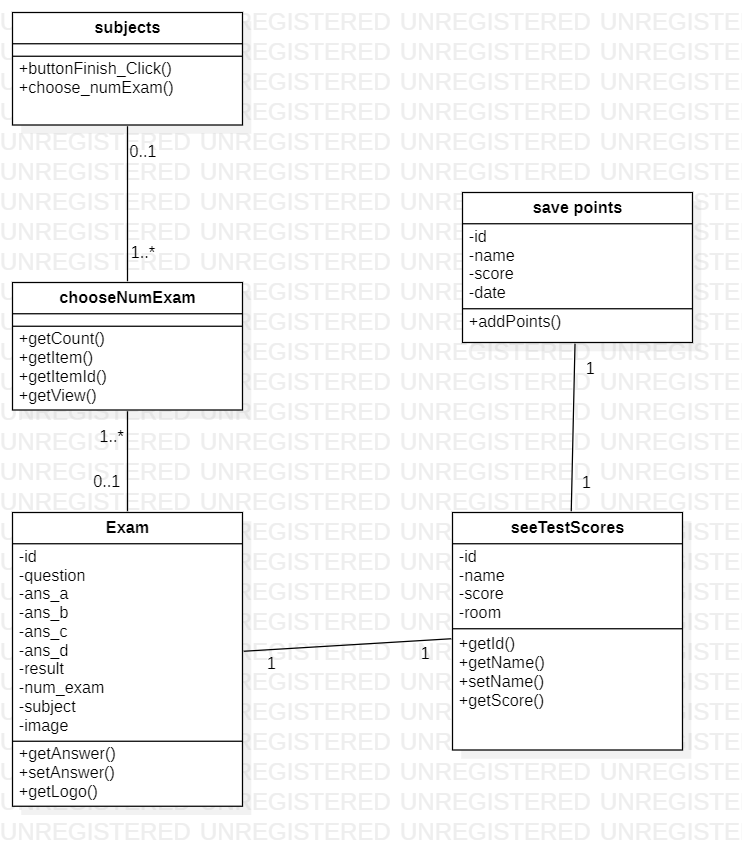
##### Bảng 2.9: Danh sách các thộc tính lớp Formula

Danh sách các phương thức

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Ý nghĩa** |
| 1 | getFormula() | Lấy danh sách công thức |

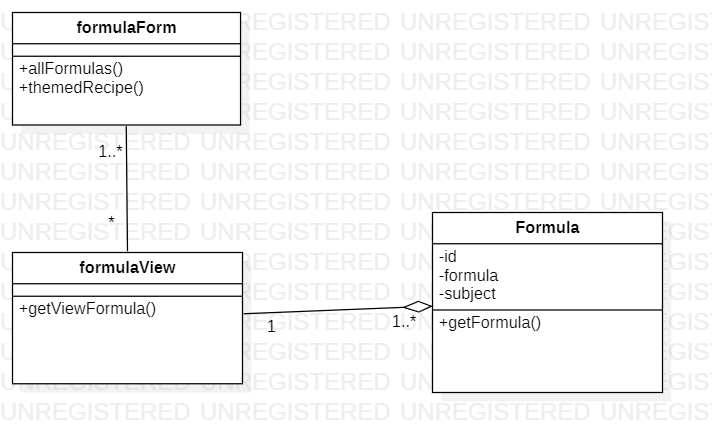
##### Bảng 2.10: Danh sách các phương thứ lớp Formula

* + - 1. Sơ đồ lớp của Exam



##### Hình 2.9: Sơ đồ lớp chức năng làm bài kiểm tra

* + - 1. Sơ đồ lớp của Formula

****

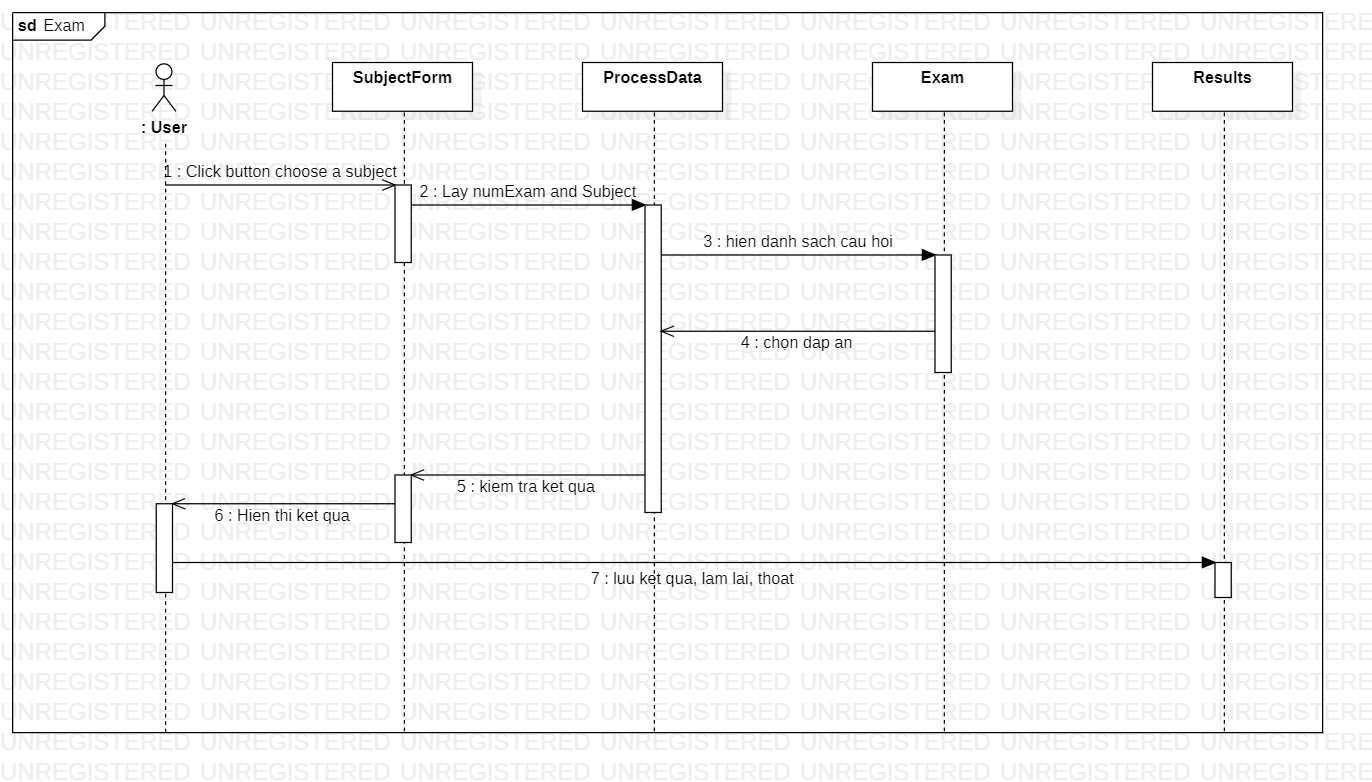
##### Hình 2.10: Sơ đồ lớp chức năng xem công thức

* + - 1. Sơ đồ trạng thái của Exam



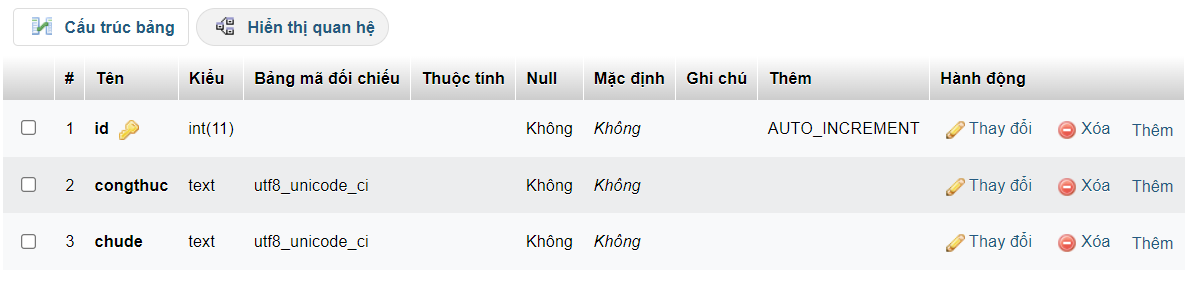
##### Hình 2.11: Sơ đồ trạng thái chức năng làm bài kiểm tra

* + - 1. Sơ đồ tuần tự của Exam



##### Hình 2.12: Sơ đồ tuần tự chức năng làm bài kiểm tra

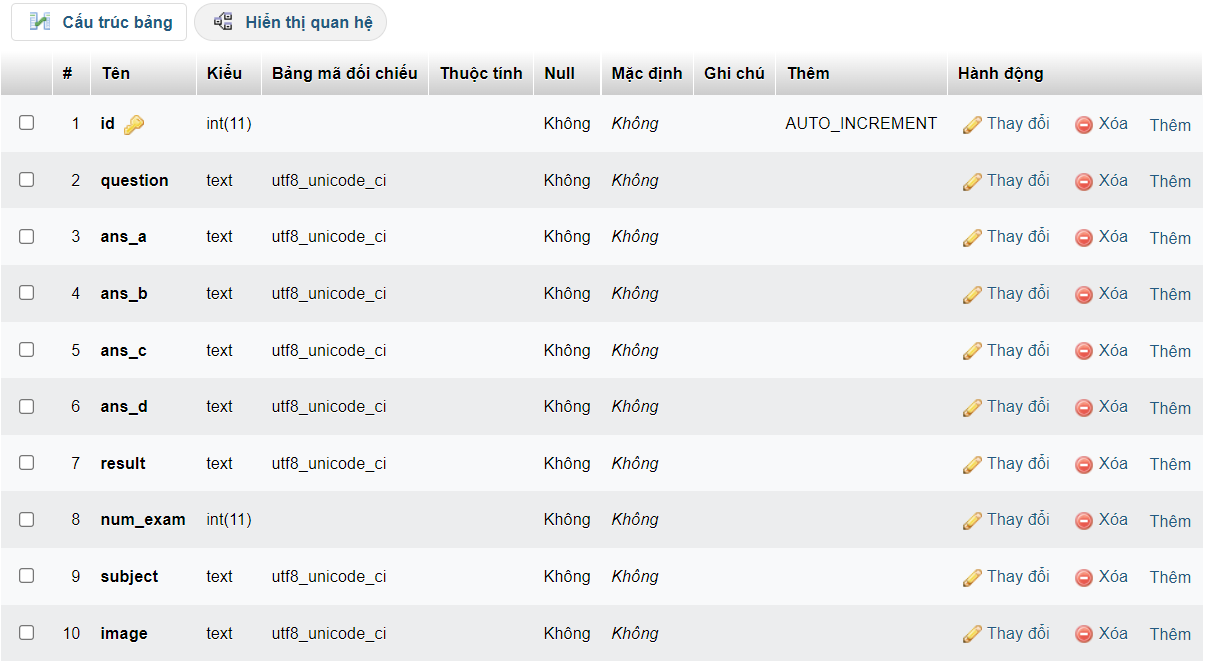
* + 1. Thiết kế cơ sở dữ liệu
       1. Bảng công thức



##### Hình 2.13: Bảng cơ sở dữ liệu công thức

Mã nguồn: [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) `dbtracnghiem`.`congthuc` (`id` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL AUTO\_INCREMENT , `congthuc` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `chude` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , PRIMARY KEY (`id`)) ENGINE = InnoDB;

* + - 1. Bảng trắc nghiệm



##### Hình 2.14: Bảng cơ sở dữ liệu trắc nghiệm

Mã nguồn: [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) `dbtracnghiem`.`tracnghiem` (`id` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL AUTO\_INCREMENT , `question` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `ans\_a` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `ans\_b` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `ans\_c` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `ans\_d` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `result` TEXT  [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `num\_exam` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `subject` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `image` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , PRIMARY KEY (`id`)) ENGINE = InnoDB;

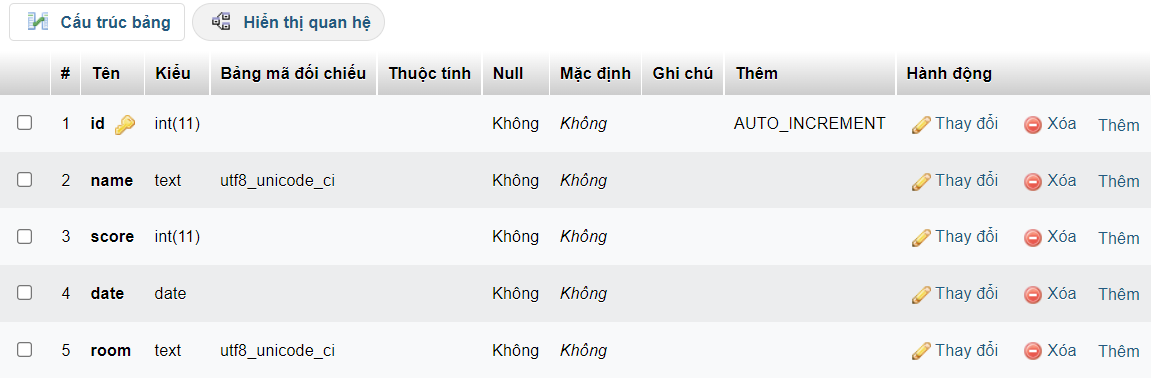
* + - 1. Bảng SQLite\_Sequence



##### Hình 2.15: Bảng cơ sở dữ liệu sqlite\_sequence

Mã nguồn: [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) `dbtracnghiem`.`sqlite\_sequence` (`name` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `seq` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL ) ENGINE = InnoDB;

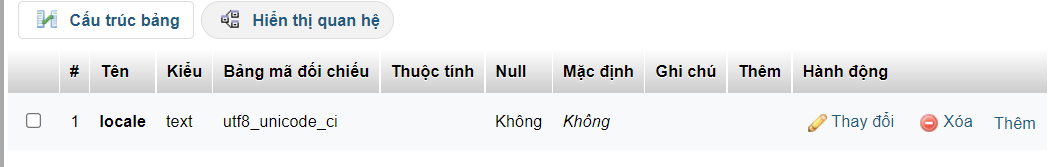
* + - 1. Bảng tbScore



##### Hình 2.16: Bảng cơ sở dữ liệu tbScore

Mã nguồn: [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) `dbtracnghiem`.`tbscore` (`id` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL AUTO\_INCREMENT , `name` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `score` INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `date` DATE [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , `room` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL , PRIMARY KEY (`id`)) ENGINE = InnoDB;

* + - 1. Bảng android\_metadata



##### Hình 2.17: Bảng cơ sở dữ liệu android\_metadata

Mã nguồn: [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) `dbtracnghiem`.`android\_metadata` (`locale` TEXT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL ) ENGINE = InnoDB;

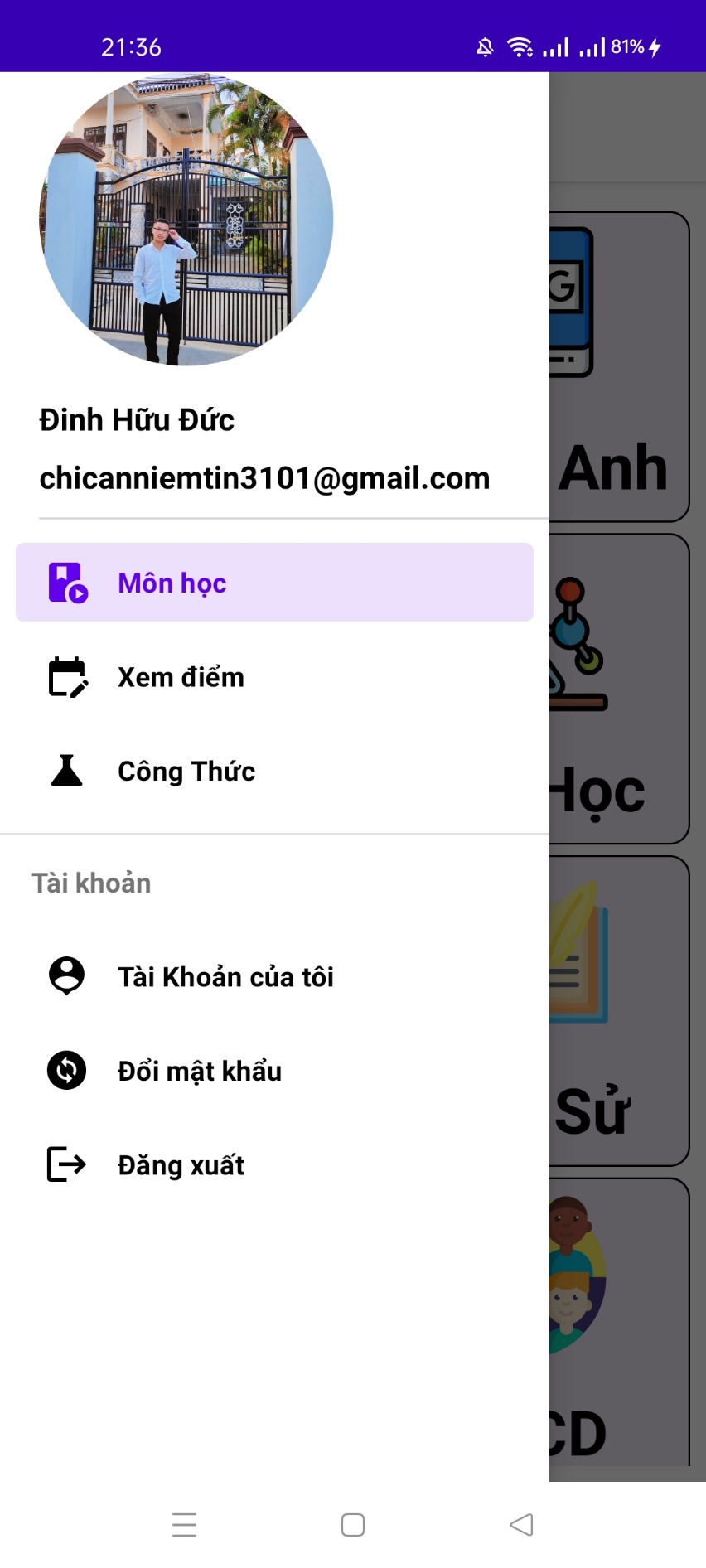
CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

* 1. Giao diện chương trình
     1. Giao diện màn hình chính:

**

##### Hình 3.1: Giao diện màn hình chính

* + 1. Giao diện menu:



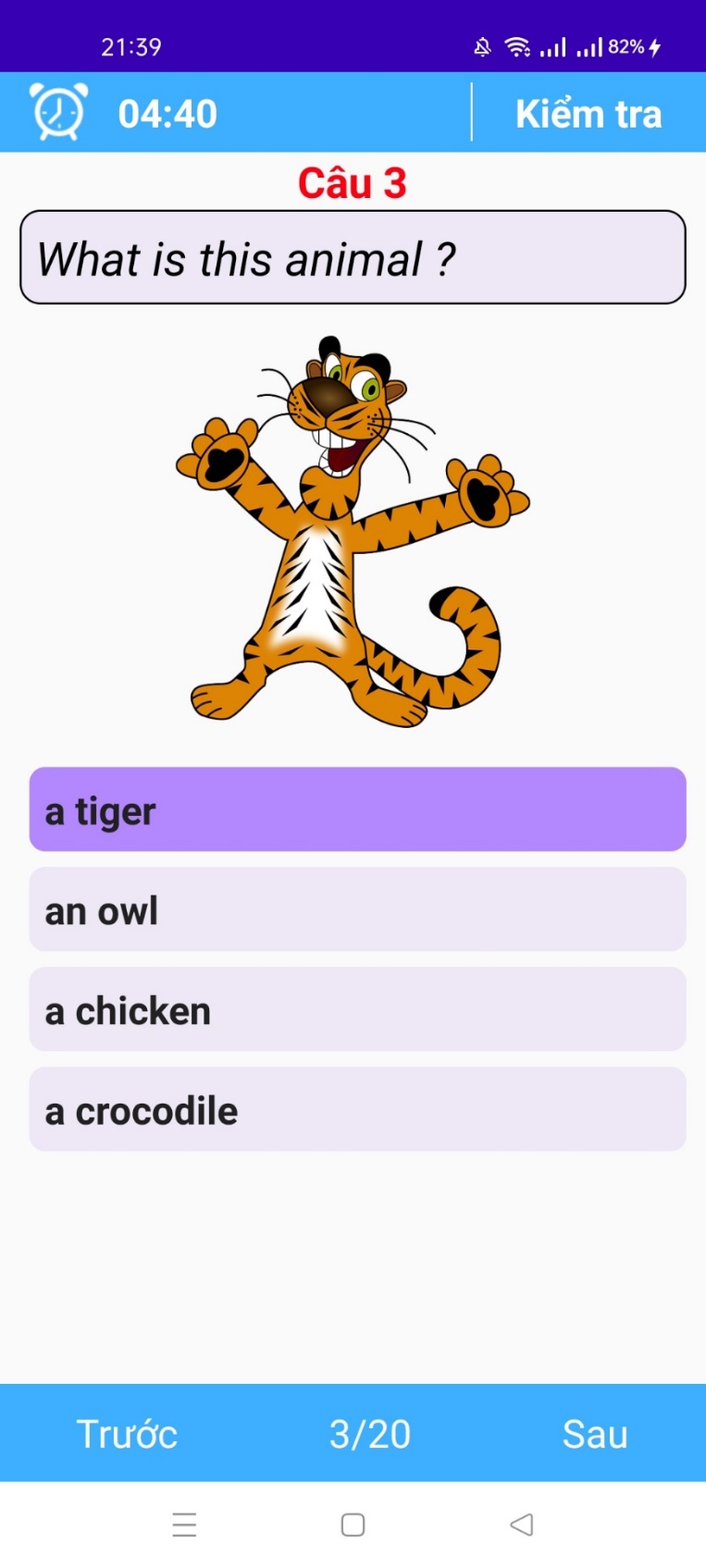
##### Hình 3.2: Giao diện Menu

* + 1. Giao diện đề:



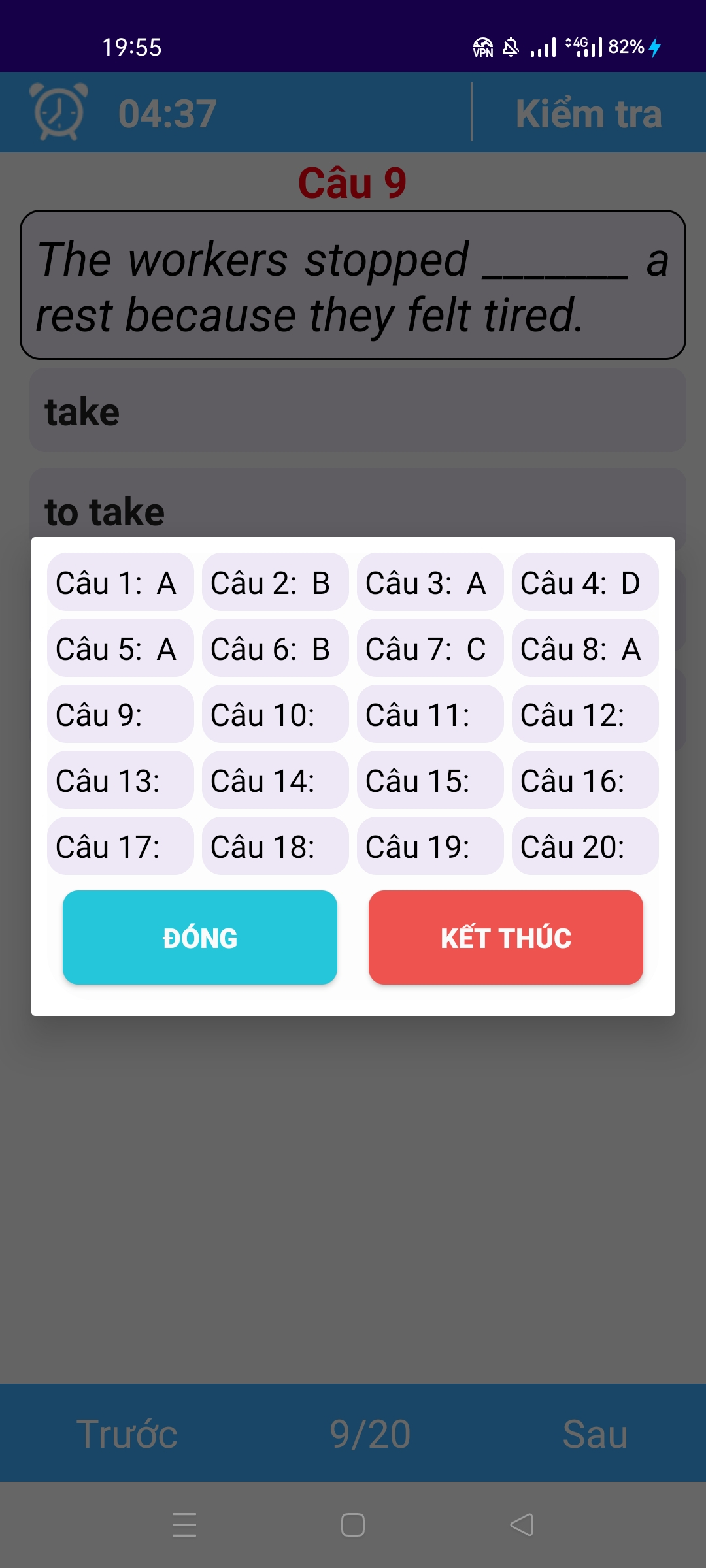
##### Hình 3.3: Giao diện chọn đề

* + 1. Giao diện trả lời:



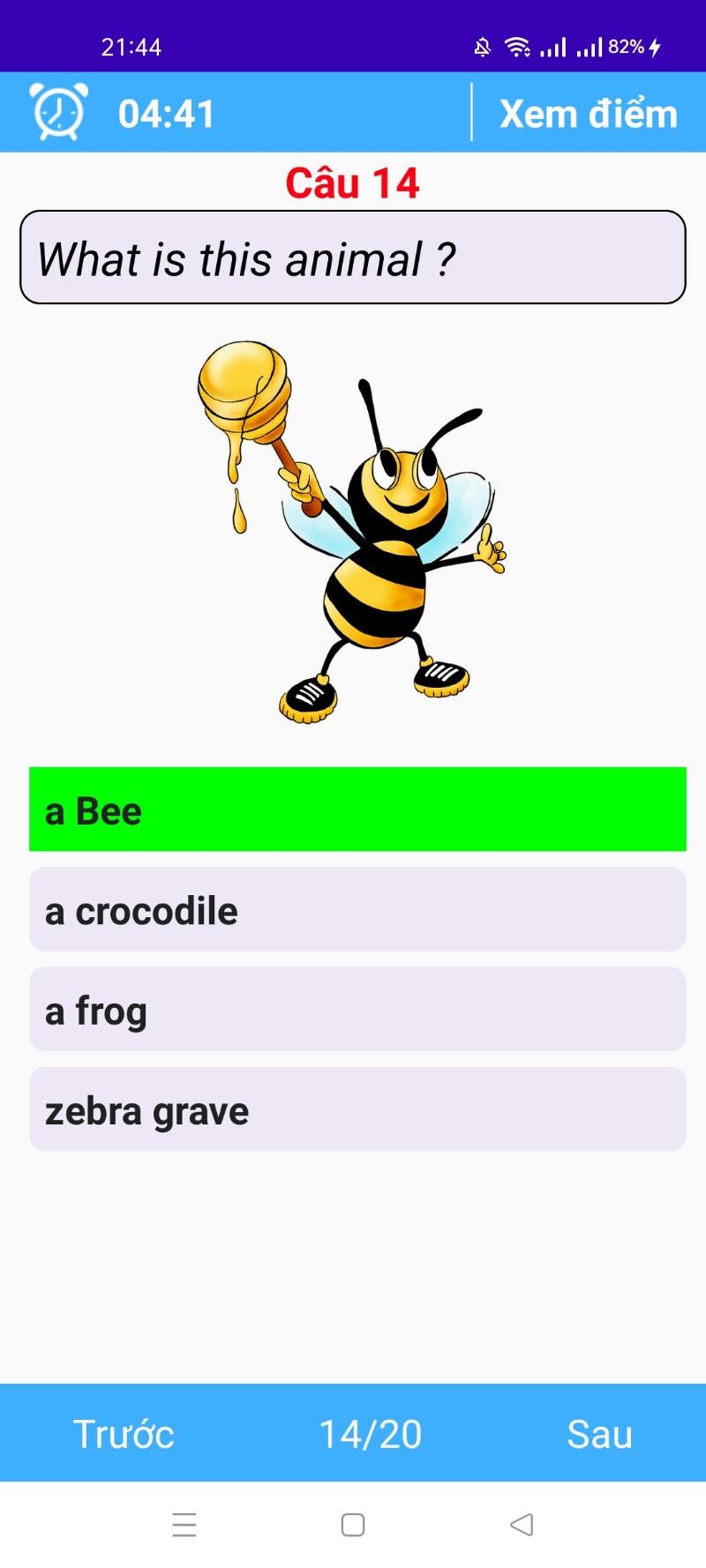
##### Hình 3.4: Giao diện chọn đáp án

* + 1. Giao diện tổng kết câu đã trả lời:

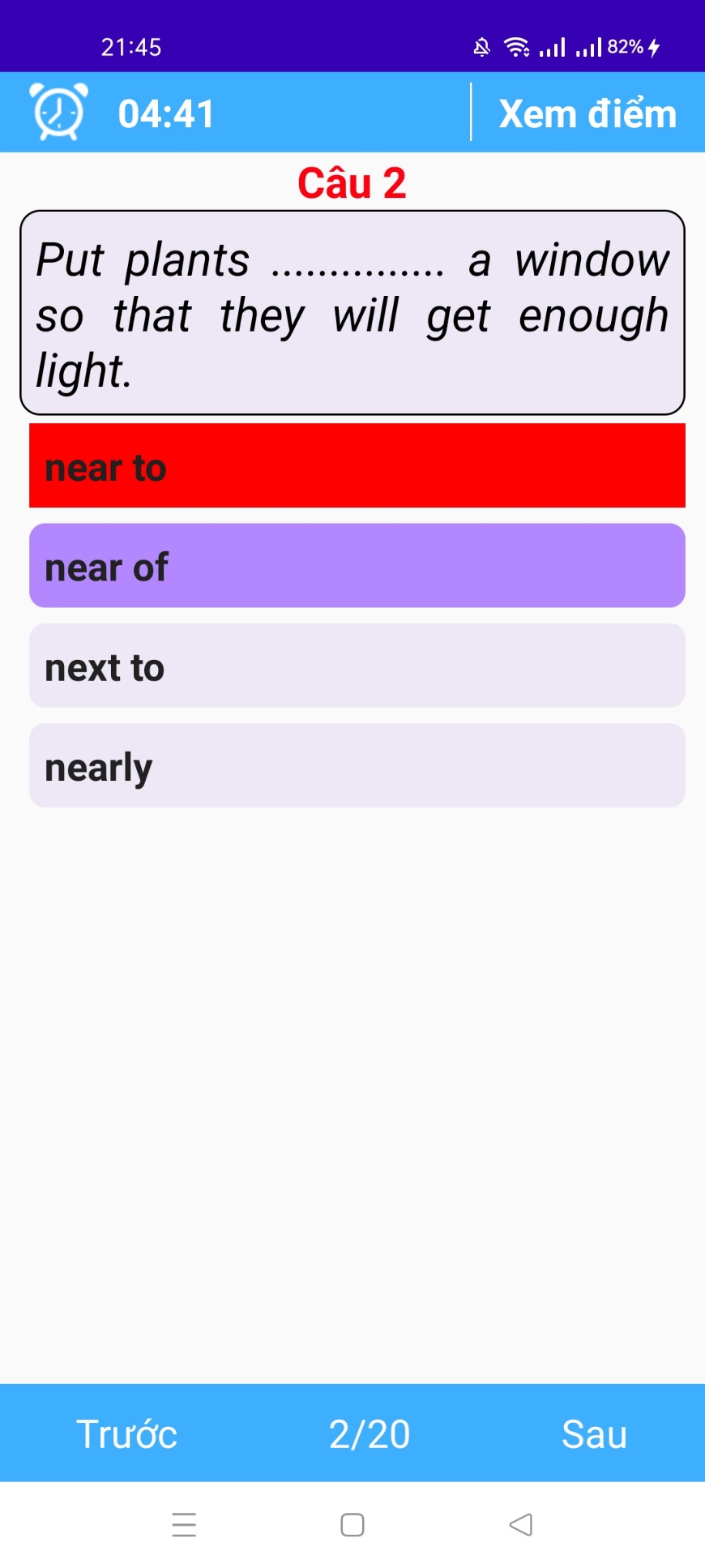


##### Hình 3.5: Giao diện xem các đáp án đã chọn

* + 1. Giao diện đáp án:

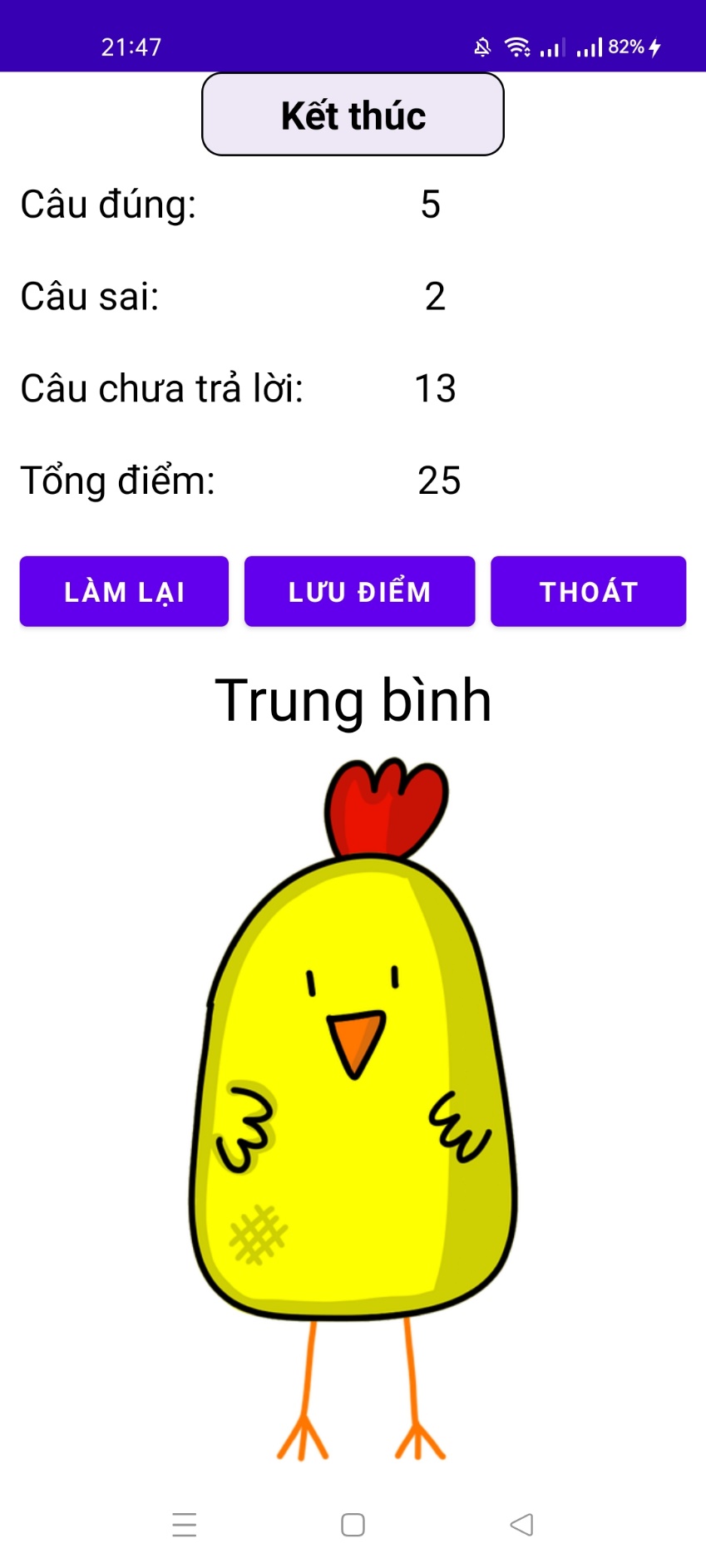


##### Hình 3.6: Giao diện đáp án đúng



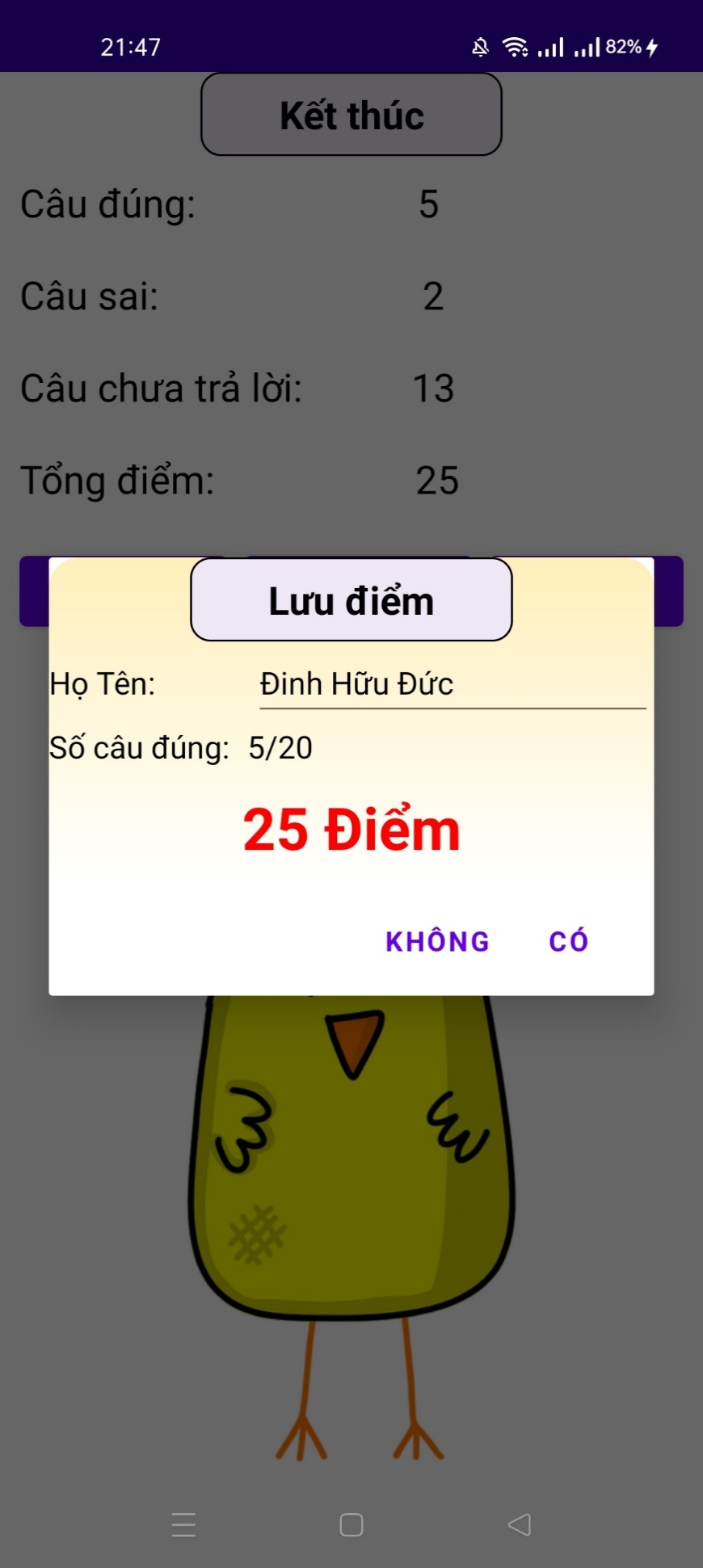
##### Hình 3.7: Giao diện đáp án sai

* + 1. Giao diện xem điểm:



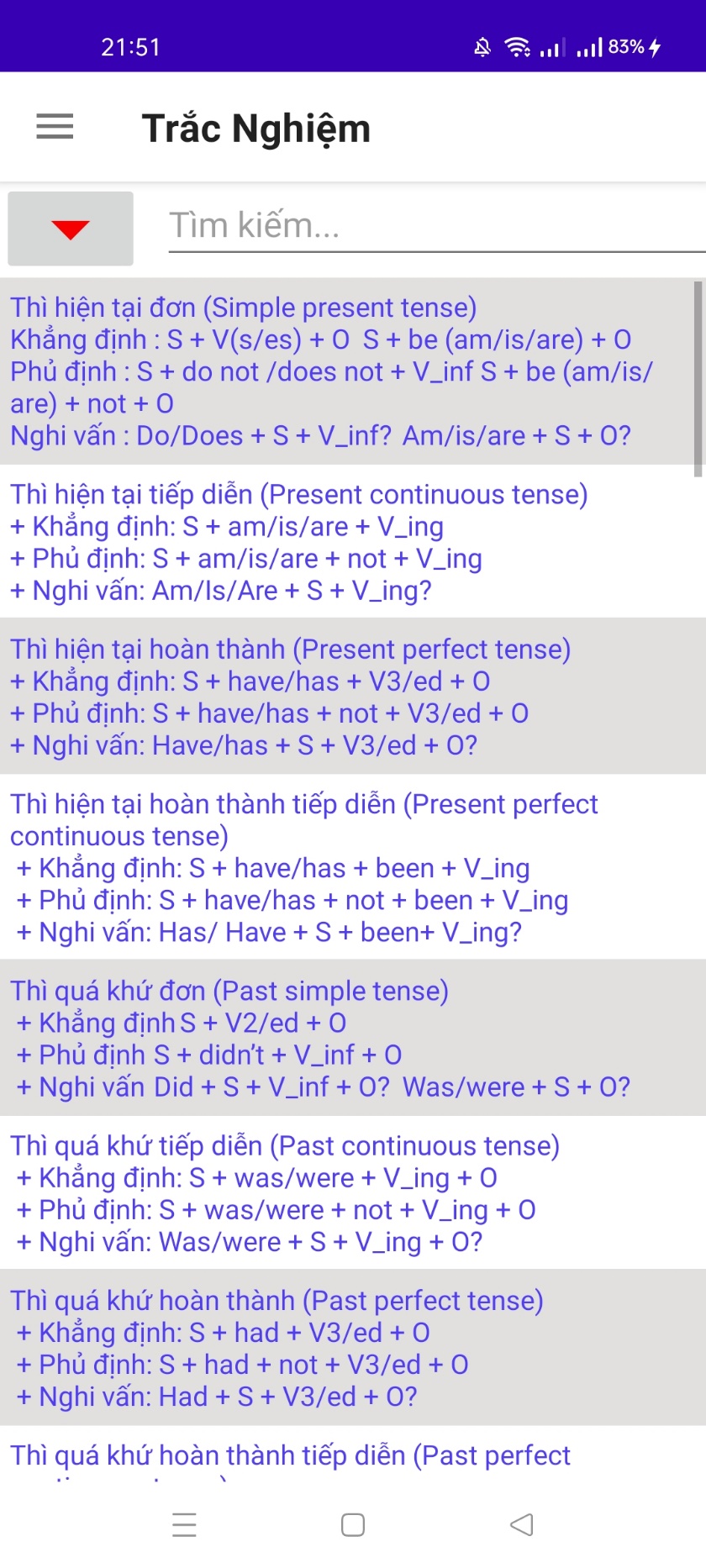
##### Hình 3.8: Giao diện xem điểm và sô câu đúng sai

* + 1. Giao diện lưu điểm:

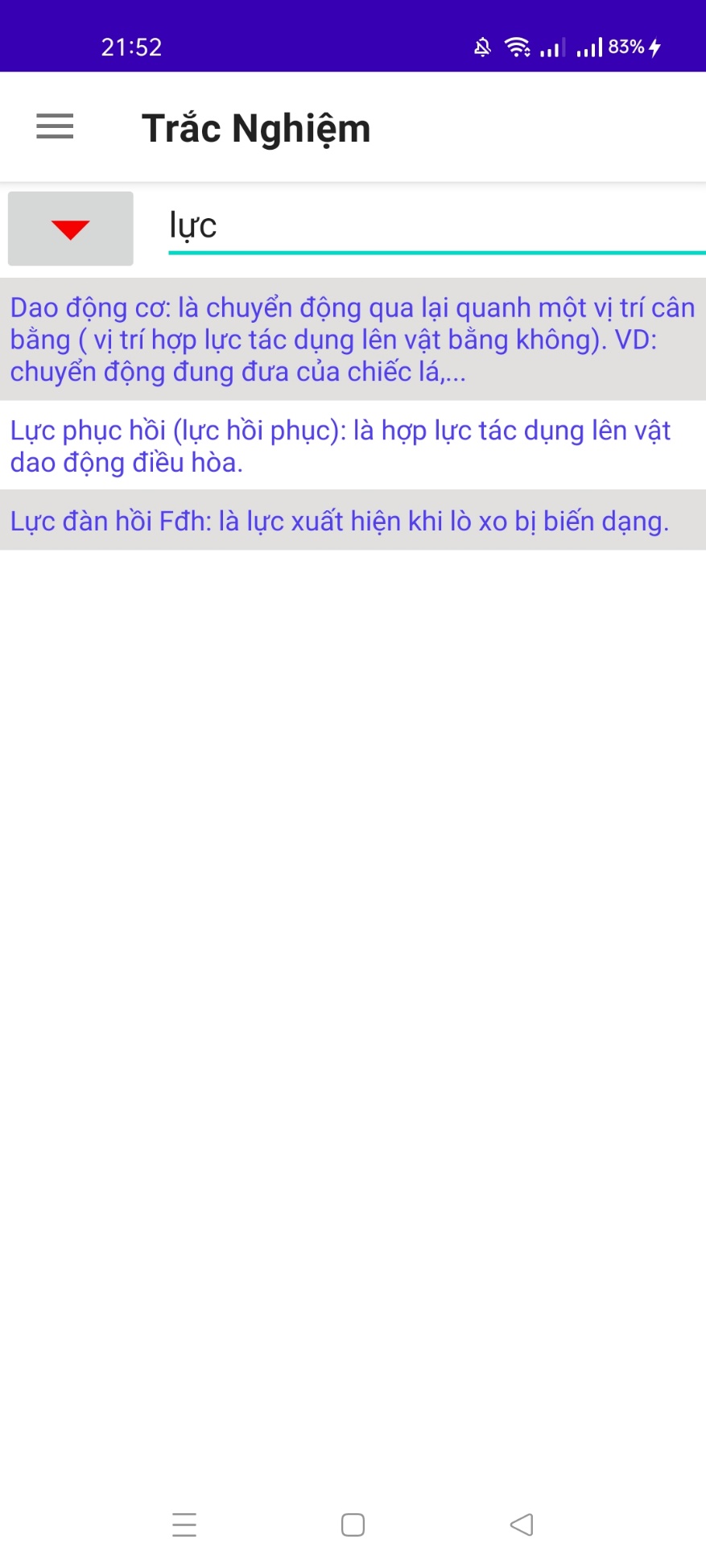
**

##### Hình 3.9: Giao diện lưu điểm

* + 1. Giao diện tìm kiếm công thức:



##### Hình 3.10: Giao diện tất cả công thức



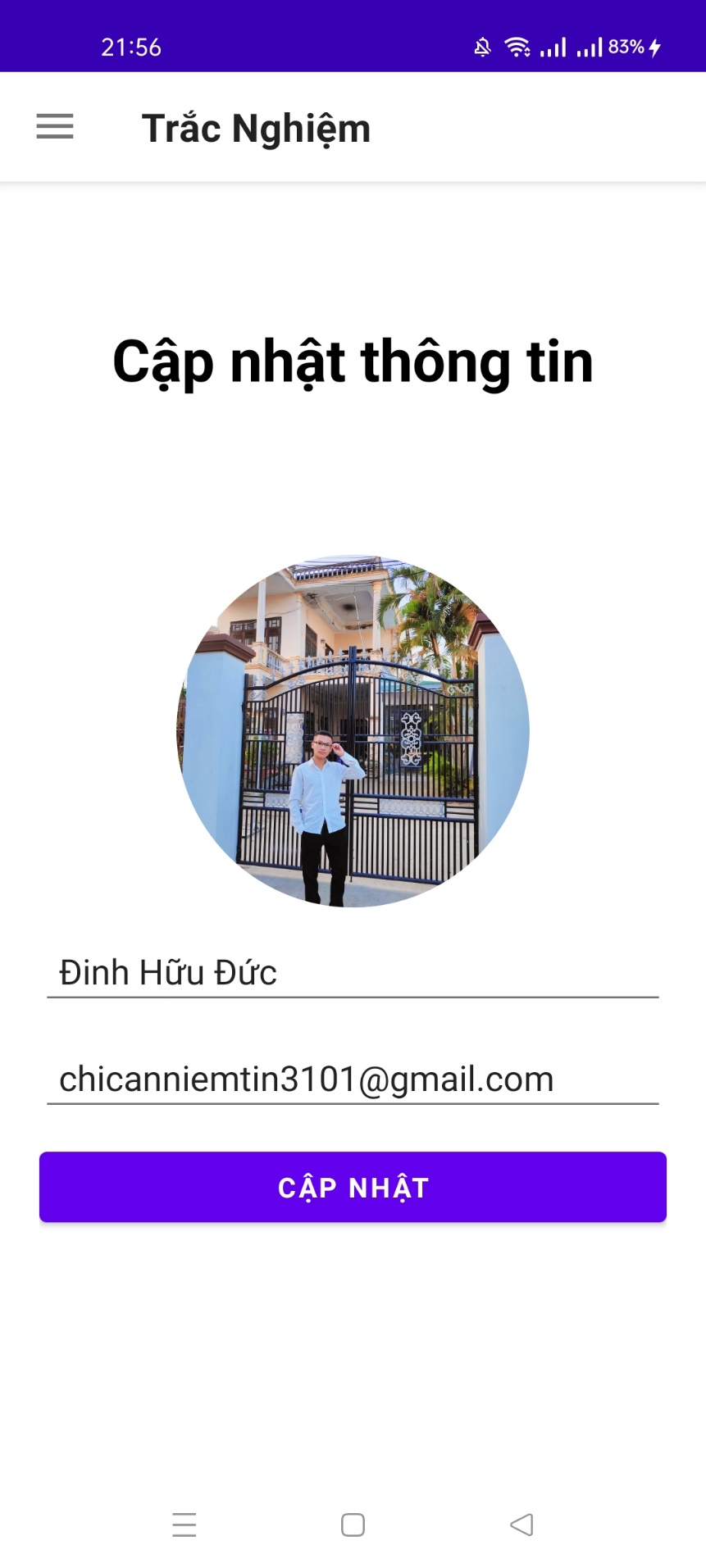
##### Hình 3.11: Giao diện công thức đã tìm

* + 1. Giao diện danh sách điểm:

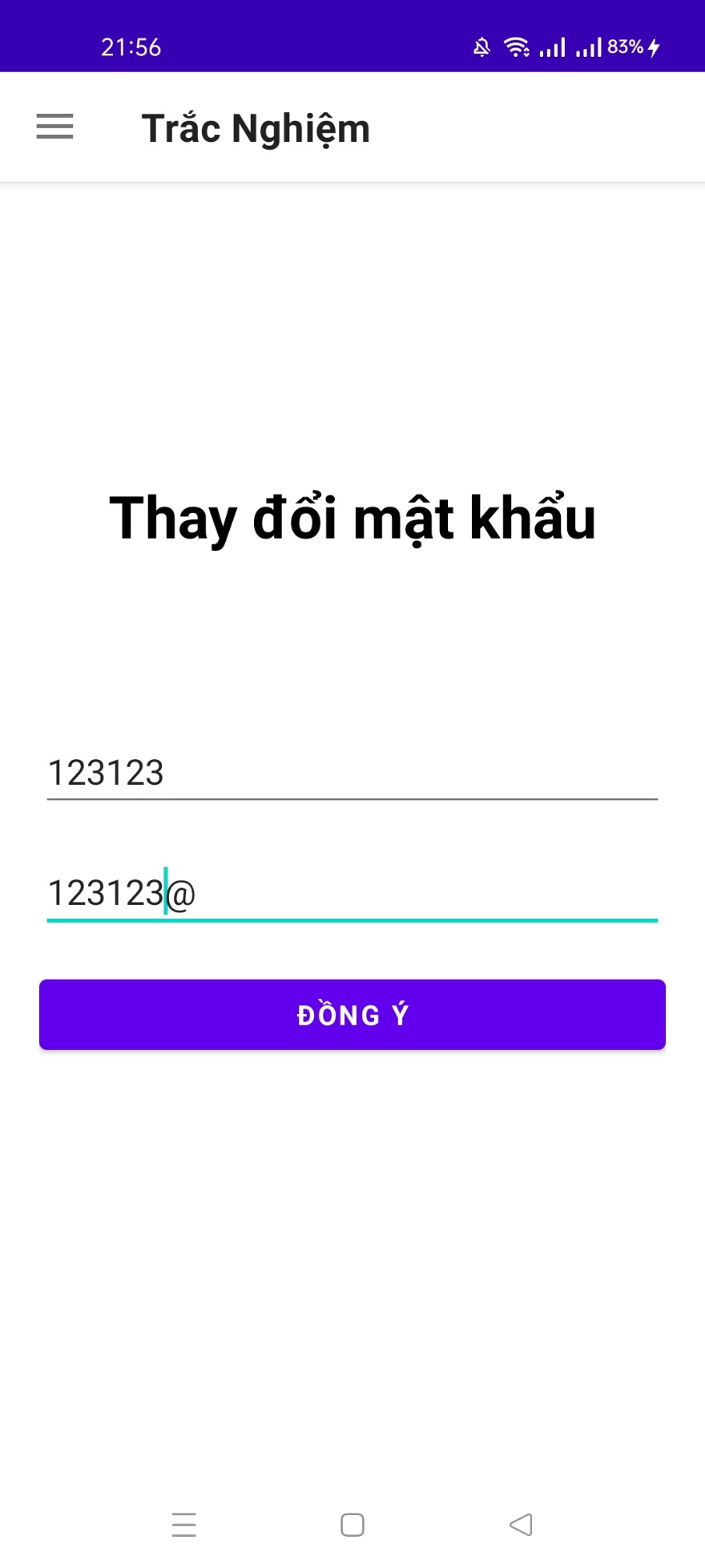


##### Hình 3.12: Giao diện danh sách điểm

* + 1. Giao diện thông tin tài khoản:



##### Hình 3.13: Giao diện cập nhật thông tin



##### Hình 3.13: Giao diện thay đổi mật khẩu

CHƯƠNG 4: KẾT LUẬN VÀ KIẾN NGHỊ

* 1. Kết quả đạt được
* Hoàn thiện được tính năng cơ bản của một app luyện thi trắc nghiệm.
* Giao diện được thiết kế đẹp mắt đơn giản và dễ sử dụng.
* Người dung đã có thể quản lý và cập nhật thông tin của bản thân
* Có thể tìm kiếm được lý thuyết học tập với nhiều lựa chọn cơ bản và chức năng tìm kiếm nâng cao.
  1. Hướng phát triển

Mặc dù đã cố gắng, tìm hiểu các kiến thức đã học, kết hợp với thực tế và tra cứu các tài liệu liên quan đến đề tài “**Xây dựng phần mềm luyện thi trắc nghiệm trên thiết bị di động**” do hạn chế về thời gian, khả năng và kinh nghiệm nên đề tài không tránh khỏi những sai sót. Sau đây là hướng phát triển của em trong tương lai:

* Tìm hiểu sâu hơn về ngôn ngữ java để có thể đáp ứng nhiều hơn nữa nhu cầu của người sử dụng, phát triển và tối ưu hóa hệ thống.
* Tìm hiểu thêm một số ngôn ngữ, các phần mềm ứng dụng để nâng cao giao diện đồ họa đẹp mắt, thân thiện hơn…
* Xây dựng ứng dụng quy mô lớn hơn với nhiều ứng dụng, chức năng…

**TÀI LIỆU THAM KHẢO**

Tài liệu tham khảo trực tuyến

[1]. https://khoapham.vn/khoa-hoc-lap-trinh-android.html

[2]. https://developer.android.com/training/animation/screen-slide