TRƯỜNG ĐẠI HỌC AN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

THỰC TẬP CUỐI KHÓA NGÀNH KỸ THUẬT PHẦN MỀM

XÂY DỰNG ỨNG DỤNG THI TRẮC NGHIỆM TRÊN NỀN TẢNG DI ĐỘNG

TRUNG TÂM TIN HỌC – TRƯỜNG ĐẠI HỌC AN GIANG

TRỊNH THANH TÙNG

AN GIANG, 12 THÁNG 04 NĂM 2021

TRƯỜNG ĐẠI HỌC AN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

THỰC TẬP CUỐI KHÓA NGÀNH KỸ THUẬT PHẦN MỀM

XÂY DỰNG ỨNG DỤNG THI TRẮC NGHIỆM TRÊN NỀN TẢNG DI ĐỘNG

TRỊNH THANH TÙNG
DPM175086

GIẢNG VIÊN HƯỚNG DẪN: ThS. LÊ TRUNG THƯ

AN GIANG, 12 THÁNG 04 NĂM 2021

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

Nội dung nhận xét:

- **Đồng ý** hay **không đồng ý** cho sinh viên báo cáo TTCK; Nếu không đồng ý cần ghi rõ lý do.
- Kết quả đạt được so với yêu cầu;
- Ý kiến khác (nếu có)

LỜI CẢM ƠN

Trước tiên, em xin gửi lời cảm ơn chân thành, sâu sắc đến ban Giám Hiệu trường Đại học An Giang cùng các thầy, cô trong khoa Công Nghệ Thông Tin đã tận tình dạy dỗ, tuyền đạt cho em nhiều kiến thức quý báo và bổ ích trên giảng đường đại học, làm hành trang vững chắc cho chặng đường sau này trong tương lai.

Để hoàn thành được đề tài luận văn này với kết quả tốt nhất, bên cạnh sự nổ lực học hỏi của bản thân, vận dụng những kiến thức đã học được ở trường và tìm hiểu ngoài thực tế, em xin bài tỏ lòng biết ơn chân thành, sâu sắc nhất tới thầy Lê Trung Thư đã trực tiếp hướng dẫn, quan tâm và tận tình giúp đỡ em trong quá trình thực hiện đề tài.

Xin chân thành cảm ơn các anh, chị khóa trên, các bạn trong khoa Công Nghệ Thông Tin nói chung và tập thể lớp DH18PM nói riêng đã giúp đỡ, động viên tôi rất nhiều trong quá trình học tập và nghiên cứu tại trường.

Mặc dù đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép nhưng với kinh nghiệm còn hạn chế của một sinh viên chắc chắn thiếu sót là điều không thể tránh khỏi. Em kính mong nhận được sự cảm thông, chỉ bảo tận tình của các thầy, cô để nâng cao khả năng tìm hiểu và nghiên cứu cho công việc thực tế sau này.

Em xin chân thành cảm ơn!

Long Xuyên, ngày 19 tháng 04 năm 2021

Sinh viên thực tập

Trịnh Thanh Tùng

TÓM TẮT

Khi cuộc sống ngày càng hiện đại, công nghệ thông tin đã và đang dần được áp dụng vào từng lĩnh vực dù là nhỏ nhất trong cuộc sống. Do đó, các công ti và doanh nghiệp nhà nước hay tư nhân đều có yêu cầu cơ bản về tin học khi tuyển dụng hay trong cuộc sống hàng ngày, áp dụng tin học có dễ giúp bản thân mỗi người giải quyết vấn đề nhanh gọn và cô bản hơn. Để đáp ứng nhu cầu đó, Trung tâm Tin học thường xuyên tổ chức đào tạo và kiểm tra cấp chứng chỉ tin học trình độ A, B quốc gia và các lớp chuyên đề. Giúp cho học viên bắt kịp với yêu cầu cơ bản về tin học hiện nay. Tuy nhiên, khi học viên muốn kiểm tra hay trau dồi thêm kiến thức cho bản thân trong thời gian rảnh thì gặp bất tiện khi chỉ có thể làm các bài thi thử trong các buổi học tai trung tâm.

Để giải quyết vấn đề đó, tôi quyết định tìm hiểu về các công nghê đa nền tảng, và xây dựng nên ứng dụng kiểm tra trắc nghiệm trên nền tảng di động giúp các học viên tham gia tại trung tâm có thể trau dồi rèn luyện bên cho trình độ bản thân mà không cần trực tiếp lên trung tâm.

Đề tài này bao gồm 3 chương:

Chương I: Giới thiệu về Trung tâm Tin học và vấn đề.

Chương II: Cơ sở lí thuyết được sử dụng để giải quyết vấn đề, một số khái niệm.

Chương III: Bắt tay vào xây dựng hệ thống ứng dụng từ các yêu cầu.

MỤC LỤC

DANH SÁCH HÌNH ẢNH

DANH SÁCH BẢNG BIỂU

DANH SÁCH TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	Ý NGHĨA
1	SDK	Software Development Kit
2	HTML	Hypertext Markup Language
3	CSS	Style Sheet Language
4	UX	User Experience (tối ưu trải nghiệm người dùng)
5	UI	User Interface (giao diện người dùng)
6	RESTful API	Những API được thiết kế theo cấu trúc REST
7	HTTP	Hypertext Transfer Protocol
8	ECMA	European Computer Manufacturers Association
9	ARM	Advanced RISC Machine
10	UWP	Universal Windows Platform
11	IDE	Integrated development environment (Môi Trường Phát Triển Tích Hợp)
12	SSL	Secure Sockets Layer (tiêu chuẩn an ninh công nghệ toàn cầu)
13	CDN	Content Delivery Network (Mạng phân phối nội dung)
14	Json	JavaScript Object Notation (kiểu dữ liệu mở trong JavaScript)
15	noSQL	NonStructured Query Language ()
16	SMS	Short Message Services
17	SQL	Structured Query Language (Ngôn ngữ truy vấn mang tính cấu trúc)
18	RTDB	Real time database
20	PHP	Hypertext Preprocessor

CHƯƠNG I
GIỚI THIỆU VỀ TRUNG TÂM TIN HỌC VÀ ĐỀ TÀI

1.1.Trung tâm tin học.

Trung tâm Tin học Trường Đại học An Giang (sau đây gọi là Trung tâm) là cơ sở giáo dục thường xuyên của hệ thống giáo dục quốc dân, là đơn vị sự nghiệp công lập trực thuộc Trường Đại học An Giang, có tư cách pháp nhân, có con dấu và tài khoản riêng để hoạt động theo quy định của pháp luật.

1.2. Nhiệm vụ.

Xây dựng và tổ chức thực hiện quy hoạch, kế hoạch phát triển Trung tâm phù hợp với quy hoạch, kế hoạch phát triển giáo dục của cả nước, địa phương và của Trường Đại học An Giang.

Tư vấn và quản lý việc triển khai và ứng dụng công nghệ thông tin (CNTT) cho Nhà trường, đảm bảo việc khai thác, sử dụng và phát triển các nguồn tài nguyên thông tin của Trường một cách hiệu quả và đúng theo qui định của pháp luật; quản trị toàn bộ hệ thống mạng thông tin của Trường Đại học An Giang

1.3. Chức năng kinh doanh.

Tổ chức thực hiện hoạt động đào tạo dịch vụ, hợp tác hoặc liên kết đào tạo dịch vụ trong lĩnh vực CNTT như: tin học ứng dụng, các chương trình giáo dục thường xuyên ứng dụng CNTT - truyền thông, các chương trình bồi dưỡng kiến thức tin học khác như: lập trình phát triển phần mềm, quản trị dự án phần mềm, triển khai và quản trị mạng, an toàn và an ninh thông tin, .v.v...

Tổ chức kiểm tra, cấp chứng nhận, chứng chỉ tin học trình độ quốc gia cho các thí sinh thi đạt tại các kỳ thi của Trung tâm theo quy định của Bộ Giáo dục và Đào tạo.

Cung cấp hoạt động dịch vụ về CNTT theo nhu cầu của xã hội, điển hình như: hợp đồng giảng dạy chuyên đề tin học theo nhu cầu; Phát triển phần mềm theo hợp đồng; Cung cấp thiết bị CNTT - Truyền thông; Hợp đồng bảo hành, bảo trì, cho thuê thiết bị; Hợp đồng tư vấn và đảm bảo an toàn, an ninh cho hệ thống CNTT (mạng, website, máy trạm, …) và các hoạt động dịch vụ khác có liên quan đến CNTT.

Tư vấn cho các gói thầu mua sắm hàng hóa: lập báo cáo kinh tế kỹ thuật, lập hồ sơ thiết kế, quản lý dự án, xây dựng hồ sơ mời thầu, đánh giá hồ sơ dự thầu, giám sát triển khai hệ thống ứng dụng CNTT, …

1.4. Giới thiệu đề tài.

Ngày nay, công nghệ thông tin ngày càng phát triển, mỗi người đều sở hữu cho riêng mình một chiếc điện thoại thông minh, thế nhưng thời gian học ngoài giờ của mỗi người lại bị hạn chế đi nhiều, việc xây dựng một ứng dụng giúp mọi người tiếp cận, ôn tập và trau dồi kiến thức mọi lúc mọi nơi trở thành nhu cầu cần thiết.

Với một trong những chức năng chính là tổ chức kiểm tra, cấp chứng nhận, chứng chỉ tin học trình độ quốc gia cho các thí sinh thi đạt tại các kỳ thi của Trung tâm theo quy định của Bộ Giáo dục và Đào tạo. Trung tâm có các khoá học, các buổi ôn luyện thi cho học viên luyện thi, nhưng lại gặp khó khăn khi muốn kiểm tra trình độ của bản thân, học viên phải tới các tiết học mới có thể tham gia vào các buổi thi thử với thời gian có hạn.

Đề tài xây dựng ứng dụng thi trắc nghiệm trên nền tảng di động với mong muốn cung cấp các bài kiểm tra trắc nghiệm, thi thử một cách nhanh và thuận tiện nhất đến các học viên của Trung tâm tin học.

CHƯƠNG II
CƠ SỞ LÝ THUYẾT

2.1. Đặt vấn đề

Ngày nay việc ứng dụng công nghệ thông tin vào khoa học, đời sống đã trở nên phổ biến. Cùng với sự phát triển không ngừng về kỹ thuật máy tính, mạng internet và các nền tảng công nghệ đã trở thành một trong những sản phẩm có giá trị lớn lao và ngày càng trở thành những công cụ không thể thiếu, là nền tảng rút ngắn thời gian của một công việc, giúp mọi thứ trở nên dễ dàng và thuận tiện hơn trong đời sống xã hội hiện nay.

Việc xây dựng một ứng dụng điện thoại đem lại rất nhiều mặt hữu ích cho người dùng lẫn doanh nghiệp đặc biệt tăng tương tác giữa người dùng với các dịch vụ và sản phẩm của doanh nghiệp. Một ứng dụng chuyên nghiệp còn giúp doanh nghiệp tạo được uy tín cũng như xây dựng được thương hiệu của mình. Một ứng dụng với giao diện phù hợp, hấp dẫn còn thu hút người dùng bằng tính tò mò, hiếu thị.

Do đó tôi chọn đề tài "Xây dựng ứng dụng thi trắc nghiệm trên nền tảng di động" cho Trung tâm Tin học làm đề tài thực tập cuối khoá của mình nhằm giúp

trung tâm tang sự tương tác với học viên và giúp học viên thuận thiện hơn trong việc kiểm tra khả năng của bản thân.

2.2. Cross Platform.

Cross Platform hay còn được gọi là Multi Plaform là thuật ngữ để chỉ những ứng dụng đa nền tảng. Trong khi các ứng dụng gốc tốn quá nhiều phí để xây dựng trọn bộ ứng dụng trên tất cả các nền tảng thì với Cross Platform, mọi thứ đều có thể giải quyết. Lập trình viên chỉ cần lập trình một lần và biên dịch hoặc phiên dịch ra thành nhiều bản Native App tương ứng với từng nền tảng khác nhau.

Công cụ quan trọng nhất để thực hiện các dự án ứng dụng đa nền tảng (Cross Platform) chính là Frameworks đa nền tảng. Có rất nhiều Framework đa nền tảng. Mỗi loại sẽ có những điểm mạnh và điểm yếu khác nhau. Tùy vào mục tiêu xây dựng App mà lập trình viên sẽ lựa chọn Framework nào cho phù hợp.

Nổi tiếng và phổ biến nhất là Framework Ximarin. Ngôn ngữ lập trình chủ đạo trong Ximarin là C#, ngoài ra còn có Objective-C, Swift và Java. Ngoài ra, còn một số cái tên mà bạn nên lưu tâm là Sencha Touch, react (Facebook), Monocross, Corona SDK, Flutter (Google), …

Vậy là hẳn nhiên bạn có thể thấy được lợi thế lớn nhất của Cross Platform là những tối ưu về thời gian và chi phí thực hiện. Điều này cũng góp phần đáp ứng nhu cầu sử dụng các ứng dụng đa nền tảng của người dùng hiện nay.

Và tất nhiên, công nghệ này vẫn còn rất nhiều hạn chế, cũng là lý do giải thích vì sao loại ứng dụng này chưa thể chiếm thị trường lớn. Những hạn chế đó tồn tại trong chính các Framework của loại công nghệ này. Kể cả Ximarin vốn có ưu điểm là có độ tương thích cao thì cũng chính ưu điểm này hạn chế khả năng tận dụng vô số thư viện nguồn mở trên cả Android và IOS. Sencha Touch là Framework ứng dụng công nghệ hàng đầu vậy mà nhiều lập trình viên vẫn từ chối sử dụng vì cho rằng “quá trình cấp phép thương mại” của Sencha Touch có phần khó hiểu.

2.3. Hybird App vs Native App.

2.3.1. Native App.

Native app hay còn được gọi là ứng dụng gốc. Vốn dĩ nó được gọi như vậy là bởi vì nó được viết bằng chính các ngôn ngữ lập trình gốc tối ưu nhất dành riêng cho từng nền tảng cụ thể. Hai nền tảng di động phổ biến nhất hiện nay là Android và IOS (Windows Phone thì đã bị khai tử vào tháng 10/ 2017). Từ đó, các ngôn ngữ lập trình tương ứng được chính các công ty mẹ tạo ra phù

hợp với từng nền tảng. Chẳng hạn như Apple đã có Swift, Objecive-C được dành cho lập trình ứng dụng trên nền tảng IOS. Lập trình trên Android thì dùng Java, mặc dù đây không phải ngôn ngữ do Google tạo ra.

Phần lớn ứng dụng ngày nay như phần mềm quản lý nhà trọ được lập trình dựa trên công nghệ này mặc dù chi phí để thực hiện chúng tương đối cao hơn những loại công nghệ khác.

Điểm mạnh của lập trình ứng dụng gốc đó là ứng dụng được lập trình dành riêng cho một hệ điều hành duy nhất. Được sự hỗ trợ của các SDK từ các nền tảng nên ứng dụng có thể tận dụng được gần như là tất cả tính năng trên hệ điều hành. Tỉ lệ hoàn thiện của ứng dụng cũng cao hơn rất nhiều so với ứng dụng khác, ít khi mắc lỗi lặt vặt.

Tuy vậy, dù phổ biến nhất, công nghệ Native App cũng không tránh khỏi hạn chế của mình, điều làm nó phải nhường phần còn lại thị trường cho những công nghệ khác. Khi phát triển ứng dụng gốc, lập trình viên bị hạn chế khả năng phát triển trên nhiều nền tảng khác nhau, vì mỗi ứng dụng chỉ phù hợp với duy nhất một nền tảng mà thôi. Nếu muốn phát triển ứng dụng trên 2 nền tảng lớn nhất hiện nay, lập trình viên phải làm 2 ứng dụng y hệt nhau nhưng bằng hai loại ngôn ngữ khác nhau. Đó là lý do vì sao chi phí thực hiện một ứng dụng gốc trên di động lại tốn nhiều chi phí hơn so với nhiều loại khác.

Facebook là một trong những kiểu ứng dụng gốc (Native App) và có cả hai phiên bản dành cho Android và IOS. Facebook dành cho chút xíu thị trường của Windows Phone vẫn còn đang hoạt động, tuy nhiên đã không còn đươc hỗ trợ nữa.

2.3.2. Hybird App.

Hybrid App hay còn được gọi là các ứng dụng đa nền tảng. Ứng dụng đa nền tảng được tạo ra bằng ba loại công nghệ Front End quan trọng là HTML, CSS và JavaScript. Đó thực chất là một cái web rỗng và được đặt vào bên trong một Native Container. Nhờ đó mà lập trình viên vẫn có thể đưa chúng lên AppStore và hoạt động như một cái app thông thường.

Đóng vai trò chính trong công nghệ của Hybrid App là các Framework như PhoneGap, Sencha Touch, ...

Ưu điểm được các lập trình viên xác nhận của các ứng dụng lai là tận dụng được nhiều điểm mạnh của Native App và Mobile App, giúp tăng trải nghiệm của người dùng trên ứng dụng. Thời gian và chi phí thực hiện của dạng ứng dụng này thấp hơn rất nhiều so với Native App. Thậm chí chi phí bảo hành,

bảo trì cũng thấp hơn rất nhiều, vì chỉ cần điều chỉnh trên một app mà có hiệu quả trên tất cả hệ điều hành.

Mặc nhiên, kiểu lập trình này vẫn còn tồn tại một vài hạn chế, nhất là độ mượt mà không bằng Native App và lập trình viên sẽ rất khó để tùy biến các tính năng UX/ UI trên ứng dụng.

2.4. API.

API được ví như "cánh tay phải" đắc lực không thể thiếu đối với lập trình website và thiết kế các phần mềm ứng dụng di động. Nó có tác dụng liên kết các tính năng của web, app với các cơ sở dữ liệu, tăng tương tác tối đa giữa người dùng và ứng dụng. Bên cạnh đó, công cụ này còn có rất nhiều điều hấp dẫn khác.

2.4.1. API là gì.

API được viết tắt bởi cụm từ tiếng Anh Application Programming Interface, tức giao diện lập trình ứng dụng. Đây là phương tiện cho hai hoặc nhiều ứng dụng trao đổi, tương tác với nhau, tạo ra tương tác giữa người dùng với ứng dụng hiệu quả và tiện lợi hơn.

Với API, các lập trình viên có thể tiếp cận, truy xuất dữ liệu từ máy chủ thể hiện chúng trên ứng dụng phần mềm hoặc website của mình một cách dễ dàng hơn.

Bạn có thể hình dung như thế này: Giả sử bạn đang viết một phần mềm cho Windows 10 và bạn muốn phần mềm của mình có thể dễ dàng tương tác được với các tính năng hay dịch vụ của Windows như thay đổi hình nền, điều khiển con lăn, các thao tác liên quan tới giao diện người dùng hay thậm chí tính năng/ứng dụng cao cấp hơn. Tất cả điều đó đều có thể thực hiện được qua Windows API miễn phí dành cho Windows.

Tính tới nay, API đã phát triển với nhiều loại ứng dụng và phần mềm khác nhau. Thế hệ mới nhất của web/ app API có thể ứng dụng được ở mọi hệ thống từ cơ sở dữ liệu, hệ điều hành, hệ thống nền web, thư viện hay thậm chí là phần cứng máy tính.

2.4.2. Xu hướng phát triển của API.

API hiện có 2 chính sách bảo vệ cơ bản gồm:

- Các công ty bảo vệ API cusack chính mình.
- Các công ty chuyên cung cấp API miễn phí.

Với loại thứ nhất, ở đây các công ty tường sẽ chủ trương bảo vệ API và thu lời từ các nhà phát triển phần mềm thứ 3 đăng ký, xin phép họ. Họ thường là những công ty sản xuất thiết bị, game, công nghệ, ví dụ như Sony cùng hệ thống playstation.

Với loại thứ 2, họ là các công ty chuyên cung cấp các sản phẩm API miễn phí. Người dùng có thể thoải mái sử dụng API này để viết lên phần mềm bên thứ 3. Tuy nhiên, người dùng vẫn cần mua thêm phần mềm để sử dụng. Đây cũng chính là nguồn lợi của các nhà cung cấp API miễn phí, đồng thời vừa có thể đem tới hiệu quả lan tỏa và marketing mạnh mẽ hơn.

Điển hình cho loại thứ 2 đó là hệ sinh thái của Microsoft, Google hay Apple. Họ hầu như đều cung cấp các API miễn phí, đổi lại là các lập trình viên sẽ viết phần mềm hệ điều hành, người dùng phải mua của Window, Google để có thể sử dụng phần mềm đó. Đặc biệt trong đó, API của Google chính là ứng dụng trong hệ sinh thái giúp họ ngày càng phổ biến và dễ dàng tiếp cận mọi người khắp thế giới hơn.

Hiện nay, xu hướng người dùng chủ yếu thiên về loại thứ 2. Điều này khiến cho cá API miễn phí ngày càng trở nên phổ biến hơn, vận hành và có mặt ở mọi nơi trên internet và cuộc sống.

Mức độ phổ biến của công cụ này ngày càng tăng khi chúng được nâng cao về các tiêu chuẩn về sự thân thiện, đơn giản, dễ sử dụng. Không những thế, tiêu chuẩn về bảo mật cũng được nâng cấp, giúp giảm thiểu các rủi ro hoặc khai thác các lỗ hổng bảo mật tốt hơn. Mọi hoạt động của API đều được giám sát nên hiệu suất của chúng cũng được cải thiện tốt hơn.

Với sự phát triển của các thiết bị di động và ứng dụng đi kèm, API giờ đây cũng được nâng cấp nhằm thích nghi tốt hơn với các thiết bị di động. Thực tế cho thấy, hầu hết các ứng dụng nền web và ứng dụng di động tận dụng nhiều API hơn, tạo động lực tăng trưởng API vượt bậc.

Trang Programmableweb cũng từng đánh giá, một trong những kho API miễn phí lớn nhất thế giới hiện nay có tới hơn 21.000 API được chia thành 450 mục khác nhau, bao phủ toàn bộ các lĩnh vực trong đời sống con người. Có thể thấy, API trở thành công cụ quan trọng giúp cho hỏa động vận hành hệ thống tốt hơn, trở thành chìa khóa thông minh, đồng bộ và tự động hóa đặc biệt của con người.

2.4.3. Rest – RestFul API.

REST được viết tắt bởi Representational State Transfer, là cấu trúc mẫu quy định các ứng dụng giao tiếp và tương tác với nhau. Nó bao gồm 3 bộ phận

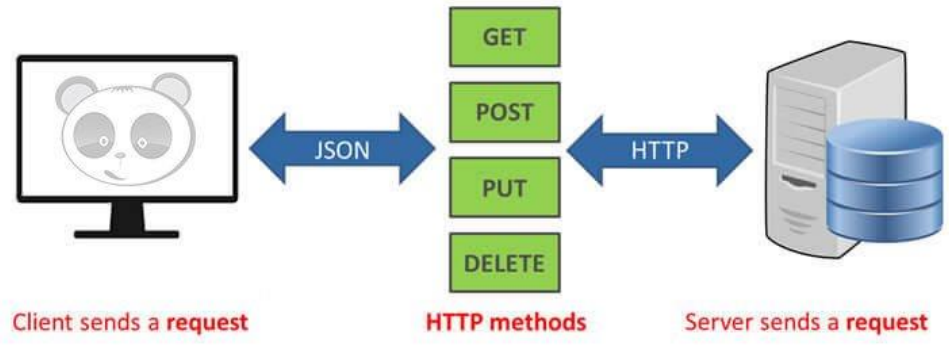
cơ bản như: bộ máy chủ ngoài chứa dữ liệu (external server), máy chủ API và máy chủ khách (client). Trong đó, máy khách có thể là bất cứ thứ gì, ứng dụng trên nền web, thư viện hoặc thậm chí là các phần mềm khác nhau của một phần mềm máy tính.

REST cho phép các máy khách truy cập máy chủ API và thực hiện các lệnh lấy về, chỉnh sửa hay xóa dữ liệu từ external server. Các lập trình viên có thể thoải mái truy xuất, chỉnh sửa dữ liệu từ máy chủ mà không cần biết hệ thống hoạt động như thế nào.

Giao thức chính của REST sử dụng là HTTP, một giao thức phổ biến với hầu hết các ứng dụng hay dịch vụ web hiện nay. Nó đem tới hiệu quả nhanh chóng trong bối cảnh đường truyền mạnh mẽ và khiến cho REST kiến trúc tốc độ nhanh hơn.

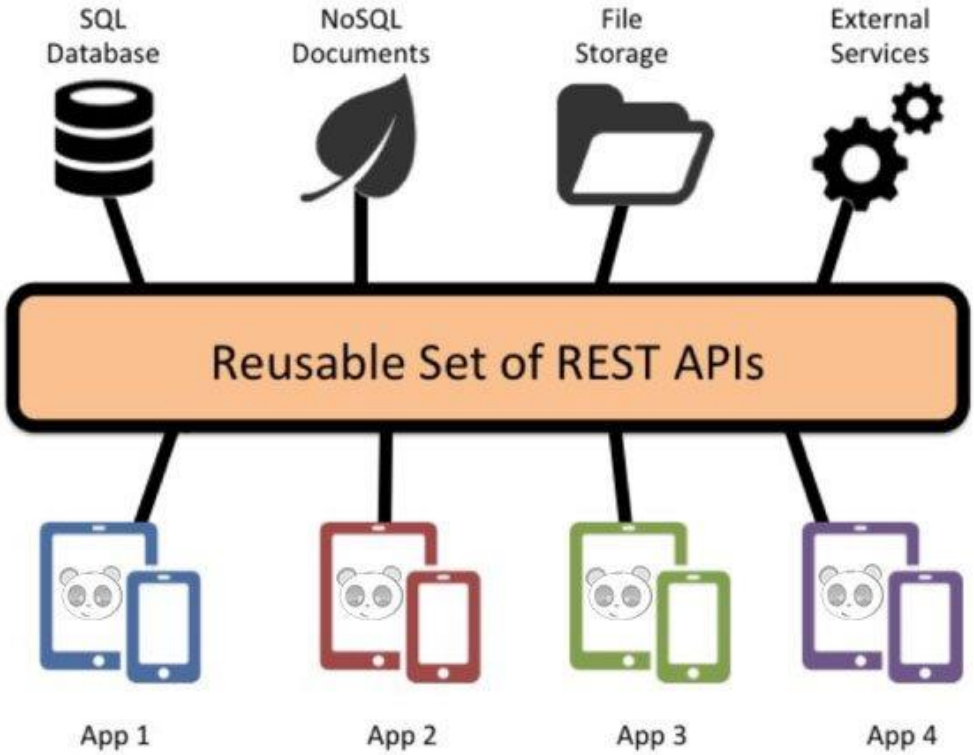
Ngoài ra, REST cũng có ưu điểm khi sử dụng giao thức stateless (không trạng thái). Hệ thống này không sử dụng session, cookie, không cần biết những thông tin đó trong mỗi lần request đến máy chủ ngoài. Điều này giúp REST giảm tải cho máy chủ ngoài, nâng cao hiệu suất làm việc.

REST ban đầu được thiết kế để sử dụng cho các dịch vụ web. Tuy nhiên, bất cứ phần mềm nào cũng có thể ứng dụng REST làm cầu nối giao tiếp với các ứng dụng khác tốt và hiệu quả hơn. Đó cũng là lý do giúp cho REST trở thành tiêu chuẩn mặc định cho hầu hết các giao thức. Và những API được thiết kế theo cấu trúc REST được gọi là RESTful API



Hình 1: Tìm hiểu về Rest API – Hình mẫu quy định cách ứng dụng giao tiếp.

4 lệnh crud cơ bản:



Hình 2: 4 lệnh cơ bản của REST API.

CRUD bao gồm: Create, Read, Update, Delete. Đây là bốn chức năng cơ bản của bất kỳ cơ sở dữ liệu nào. REST và RESTful cũng không ngoại lệ. Cá kiến trúc này đều hỗ trợ đầy đủ bốn lệnh, giúp bạn có thể thao tác với dữ liệu lấy từ máy chủ dễ dàng hơn.

Trong REST, bốn lệnh có tên gọi hơi khác một chút:

- **Post**: Có chức năng tạo dữ liệu, thông tin mới
- **Get**: Lệnh đọc/lấy một dữ liệu, thông tin mới
- **Put**: Cập nhật thông tin và dữ liệu đã có
- **Delete**: Xóa thông tin và dữ liệu đã có

Trong danh sách, Get được đánh giá là lệnh phổ biến nhất, được cho phép bởi hầu hết các Restful API hiện nay. Ba lệnh còn lại ít nhiều làm thay đổi các dữ liệu gốc máy chủ, có nguy cơ dẫn tới tai nạn ngoài ý muốn. Cho nên, chỉ những API được thiết kế riêng hay được cấp thẩm quyền như API đối tác, Admin mới được thực hiện các lệnh đó.

Tổng kết:

Từ những giao thức tách biệt, API như một cầu nối gắn kết chúng, giúp người dùng có thể tương tác giữ ứng dụng và website, cải thiện và mở rộng vận hành trên mọi thiết bị và phần cứng dù trực tuyến hay ngoại tuyến.

API giúp giảm tải phần lớn các công việc, nâng cao sự chính xác trong quá trình thiết kế website hay ứng dụng, tạo sự chuyên nghiệp, nhanh chóng và thuận tiện hơn cho các lập trình viên. Tất cả những ưu điểm đó giúp cho API trở thành những chiếc chìa khóa vạn năng, cải thiện cuộc sống tiện lợi, thông minh và hiện đại hơn.

2.5. Flutter.

2.5.1. Giới thiệu chung.

- Flutter là một framework giúp chúng ta tạo một ứng dụng chạy trên nhiều nền tảng như Web, android, IOS và tương lai còn có cả desktop như Window, MacOS và Linux (đang phát triển).

- Sử dụng ngôn ngữ lập trình Dart.

- Được giới thiệu năm 2018.

- Được phát triển và hỗ trợ bởi Google.

- Là Fast Development vì một code mà dùng được nhiều nền tảng: web app, android app, IOS app, ...

- Performance tốt hơn các tiền bối đã ra mắt sớm hơn như: Xamarin (2011 by Microsoft), React Native (2015 by Facebook), ...

2.5.2. Dart.

Dart là ngôn ngữ lập trình đa mục đích ban đầu được phát triển bởi Google và sau đó được Ecma (ECMA-408) phê chuẩn làm tiêu chuẩn. Nó được sử dụng để xây dựng các ứng dụng web, server, máy tính để bàn và thiết bị di động. Dart là một ngôn ngữ hướng đối tượng, được xác định theo lớp, với cơ chế garbage-collected, sử dụng cú pháp kiểu C để dịch mã tùy ý sang JavaScript. Nó hỗ trợ interface, mixin, abstract, generic, static typing và sound type (static typing và sound type có thể hiểu là type-safe). Dart là ngôn ngữ mã nguồn mở và miễn phí, được phát triển trên GitHub.

Ưu điểm của Dart:

- Năng suất: Cú pháp Dart rõ ràng và súc tích, công cụ của nó đơn giản nhưng mạnh mẽ. Type-safe giúp bạn xác định sớm các lỗi tinh tế. Dart có các thư viện cốt lõi và một hệ sinh thái gồm hàng ngàn package.

- **Nhanh:** Dart cung cấp tối ưu hóa việc biên dịch trước thời hạn để có được dự đoán hiệu suất cao và khởi động nhanh trên các thiết bị di động và web.

- **Di động:** Dart biên dịch thành mã ARM và x86, để các ứng dụng di động của Dart có thể chạy tự nhiên trên IOS, Android và hơn thế nữa. Đối với các ứng dụng web, chuyển mã từ Dart sang JavaScript.

- **Dễ gần:** Dart quen thuộc với nhiều nhà phát triển hiện có, nhờ vào cú pháp và định hướng đối tượng không gây ngạc nhiên của nó. Nếu bạn đã biết C ++, C # hoặc Java, bạn có thể làm việc hiệu quả với Dart chỉ sau vài ngày.

2.5.3. Widget.

Flutter là thế giới của những Widget. Từ một Container đơn giản cho đến một Button hay bất kỳ thành phần nào tạo ra ban đầu đều là Widget. Widget là những khối cơ bản dùng để xây dựng và tái sử dụng.

Tương tự như bộ môn xếp hình LEGO, một con khủng long to lớn dược xếp thành từ các khối LEGO nhỏ có đủ các hình dạng. Flutter chính là con khủng long và các khối LEGO chính là các Widget.

Data của Witget:

Bản thân mỗi Widget, nó sẽ mang trong mình những thông tin để có thể hiển thị trên UI. Và trong lập trình thì thông tin sẽ được hình thành từ data mà data thì được lưu trữ trong các biến (variable) hoặc hằng (constant). Ví dụ widget Text thì có những thông tin nào nhỉ, tất nhiên là một biến text kiểu String để lưu trữ cái text rồi, bên cạnh đó chắc chắn cũng sẽ có những biến kiểu như textColor lưu trữ thông tin về màu sắc của text, fontSize thể hiện cho kích thước font chữ, ...

Widget đơn giản là những class. Ví dụ: class Text. Như vậy, nếu liên kết những yếu tố này lại ta có thể hình dung một class Widget có những property lưu trữ và thể hiện những thông tin của Widget như thế này:

```
class Text { // Widget chỉ là class
    String text; // các property lưu trữ thông tin của widget
    Color textColor; // thông tin về màu sắc của text
    int fontSize; // thông tin về kích cỡ của font
}
```

Hình 3: Ví dụ minh hoạ về Widget.

Hàm Build:

Tương tự công thức toán học quen thuộc y = f(x). Đó là một hàm số, khi ta có giá trị của x, dựa vào một hàm f ta sẽ tính được giá trị y. Bất cứ khi

nào x thay đổi thì cũng cho ta một giá trị mới của y. Flutter cũng tương tự như vậy, nó sử dụng công thức là:

UI = f(Data)

Khi Data của Widget thay đổi, UI sẽ được update theo công thức hàm f.

```
lass CounterTextWidget extends StatelessWidget {
 int counter; // Data của widget được lưu trữ trong property có kiểu int

 @override
 Widget build(BuildContext context) { // function f chính là hàm build này
   // hàm build là một công thức, nhận Data từ biến counter
   // mỗi lần biến counter thay đổi từ 0 lên 1, 2, 3, 4,... thì ta sẽ hiển thị 1 UI khác
   return Text('Tui là widget Text. Data của tui hiện tại là $counter');
 }
```

Hình 4: Ví dụ minh hoạ về Widget.

Như vậy, chúng ta đã hình dung được Data của Widget và hàm build và mối quan hệ UI = Build(Data). Đây chính là nguyên tắc hoạt động của Widget trong Flutter.

Có 2 loại Widget chính:

- Một là: StatefulWidget, bản thân Widget này sẽ chủ động update UI.

- Hai là: StatelessWidget, bản thân Widget này sẽ thụ động update UI, hay nói cách khác là bị Widget khác ép phải update UI.

2.5.4. Flutter vs React

2.5.4.1. Điểm tương đồng.

Cả hai đều được thiết kế để phát triển cross-platform app:

- React Native và Flutter đều cung cấp hiệu suất ứng dụng vượt trội, nhanh chóng và chất lượng cao trên nhiều platform bao gồm Android, IOS và UWP.

- Cả hai framework được hỗ trợ bởi các cộng đồng công nghệ khổng lồ:

React Native được cung cấp bởi Facebook còn Flutter là SDK nguồn mở của Google để phát triển mobile app. Cả hai đều có cộng đồng các developer mạnh mẽ

Nguồn mở, miễn phí và nhanh chóng:

- React Native và Flutter là các SDK nguồn mở và miễn phí cho phép các developer tạo ra các app tuyệt vời một cách nhanh chóng dựa vào native performance của chúng.

Documentation chi tiết và luôn được cập nhật:

- Các cộng đồng của cả hai framework này luôn nỗ lực hết sức để liên tục cập nhập documentation mới nhất với các tài liệu tham khảo API và tài nguyên toàn diện.

Hỗ trợ UI và Native Experience tuyệt vời

- React Native sử dụng các yếu tố xây dựng UI cơ bản trong Android / IOS để mang lại native experience. Flutter cũng sử dụng các widget để cung cấp native experience đáng chú ý trên platform Android và IOS.

Hot reloading và sửa đổi nhanh

- React Native hỗ trợ "Hot reloading" cho phép đồng thời chạy code mới và giữ trạng thái ứng dụng, thay vì phải recompile. Tính năng này làm cho sự phát triển nhanh hơn, tức thì và hiệu quả.

- "Stateful Hot Reloading" của Flutter hỗ trợ phản ánh các thay đổi ngay lập tức mà không cần restart hoặc trong trường hợp mất trạng thái ứng dụng.

2.5.4.2. Khám phá sự khác biệt.

Điểm khác biệt đầu tiên chính là ngôn ngữ lập trình:

- Flutter sử dụng Dart. React Native cho phép bạn phát triển một app bằng JavaScript.

Tính ổn định và tính linh hoạt:

- Xét về tính ổn định, không có sự khác biệt đáng kể nào vì cả hai đều được quản lý bởi các cộng đồng công nghệ mạnh mẽ.

- Xét về tính linh hoạt và tùy biến, Flutter cung cấp một set widget tùy chỉnh phong phú để build những trải nghiệm hấp dẫn. Mặt khác, React Native cung cấp trải nghiệm người dùng liền mạch thông qua giao tiếp trực tiếp với các native platform.

- Flutter đang cố gắng để có chỗ đứng trên thị trường do phát hành bản ổn định gần đây, còn React Native vẫn đang tận hưởng vị trí đầu tiên trong danh sách các tool phát triển ứng dụng cross-platform trên thị trường.

Thời gian phát triển của một app:

- Các công ty tận dụng cross-platform để tiết kiệm thời gian code app bằng nhiều ngôn ngữ.

- Bên cạnh đó, cả React Native và Flutter đều cam kết cung cấp time-to-market cho app nhanh hơn. Bạn có thể sử dụng library của bên thứ ba và các component sẵn sàng để sử dụng để build một app. Một loạt các widget tùy chỉnh và chất lượng cao giúp build một app trong thời gian ngắn hơn so với phát triển native app.

Hiệu suất:

- Flutter chắc chắn chiến thắng ở phần này nhờ vào sự đơn giản của nó.

- Tuy nhiên, React Native lại nổi tiếng với việc truyền tải trải nghiệm người dùng tuyệt vời trên cả hai platform. Flutter có thêm lợi thế là sử dụng lại code trong khi React Native ít phù hợp hơn do kiến trúc của nó. Một lợi thế khác của Flutter khi so với layer JavaScript là giúp giao tiếp với các native component dễ dàng hơn.

- Cả hai đều là sự lựa chọn đáng tin cậy nhưng Flutter có lợi thế cạnh tranh hơn vì tính đơn giản và Dart platform của nó.

IDE và dễ code:

- Flutter còn khá mới và được hỗ trợ bởi Android Studio / IntelliJ, Visual Studio Code. Còn React Native đã có nhiều năm tuổi hơn kể từ khi ra mắt, nó được hỗ trợ tốt bởi hầu hết các IDE hàng đầu.

- React Native cung cấp curve dễ học do sử dụng JavaScript. Flutter thì đòi hỏi phải học Dart để tạo ra một app. Tuy nhiên, Dart là một điểm cộng cho các C ++ và Java developer.

2.5.4.3. Thêm một số ưu điểm và hạn chế của Flutter.

Ưu điểm:

- Documentation sạch sẽ, rõ ràng và trơn tru.
- Sự hỗ trợ từ Google và một cộng đồng mạnh mẽ.
- Tốc độ phát triển nhanh hơn.
- Native app performance tuyệt vời.
- Một loạt các UI component dễ biểu đạt và linh hoạt.
- Stateful Hot Reloading cho sự thực hiện thay đổi nhanh hơn.
- Khả năng truy cập native SDK và các feature của chúng.
- Cách tiếp cận coding hiện đại, nâng cao và linh hoạt.
- API hỗ trợ các hiệu ứng, animation và gesture 2D.

Nhược điểm:

- Mặc dù Dart là một ngôn ngữ lập trình dễ sử dụng nhưng vẫn còn khá mới, và Flutter yêu cầu phải học Dart.

- Khó khăn trong việc styling cho các component.
- Quản lý vòng đời phức tạp hơn so với React Native.
- Vì được ra mắt trễ hơn, nó không trưởng thành như React Native.

2.6. FireBase.

2.6.1. Database.

Cơ sở dữ liệu (Database) là một tập hợp các dữ liệu có tổ chức, thường được lưu trữ và truy cập điện tử từ hệ thống máy tính. Khi cơ sở dữ liệu phức tạp hơn, chúng thường được phát triển bằng cách sử dụng các kỹ thuật thiết kế và mô hình hóa chính thức.

2.6.2. Firebase.

Firebase là một nền tảng do Google phát triển để tạo ra các ứng dụng web và di động, bao gồm các API đơn giản và mạnh mẽ mà không cần backend hay server. Ban đầu Firebase là một công ty độc lập được thành lập vào năm 2011. Vào năm 2014, Google đã mua lại nền tảng này và hiện nó là sản phẩm chủ lực của họ để phát triển ứng dụng.

Firebase giúp các lập trình viên rút ngắn thời gian triển khai và mở rộng quy mô của ứng dụng mà họ đang phát triển.

Firebase là dịch vụ cơ sở dữ liệu hoạt động trên nền tảng đám mây – cloud. Kèm theo đó là hệ thống máy chủ cực kỳ mạnh mẽ của Google. Chức năng chính là giúp người dùng lập trình ứng dụng bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu.

Lịch sử phát triển:

Gần một thập niên trước, Firebase ra đời với tiền thân là Envolve. Đây là một nền tảng đơn giản chuyên cung cấp những API cần thiết để tích hợp tính năng chat vào trang web. Bên cạnh ứng dụng nhắn tin trực tuyến, Envolve còn được người dùng sử dụng để truyền và đồng bộ hóa dữ liệu cho những ứng dụng khác như các trò chơi trực tuyến, … Do đó, các nhà sáng lập đã tách biệt hệ thống nhắn tin trực tuyến và đồng bộ dữ liệu thời gian thực thành hai phần riêng biệt.

Trên cơ sở đó, năm 2012, Firebase ra đời với sản phẩm cung cấp là dịch vụ Backend-as-a-Service. Tiếp đến, vào năm 2014, Google mua lại Firebase và phát triển nó thành một dịch vụ đa chức năng được hàng triệu người sử dụng cho đến hiện nay.

Sau khi Google mua lại và phát triển, Firebase hiện nay bao gồm các hoạt động như:

- Realtime Database.
- Authentication.
- Firebase Hosting.

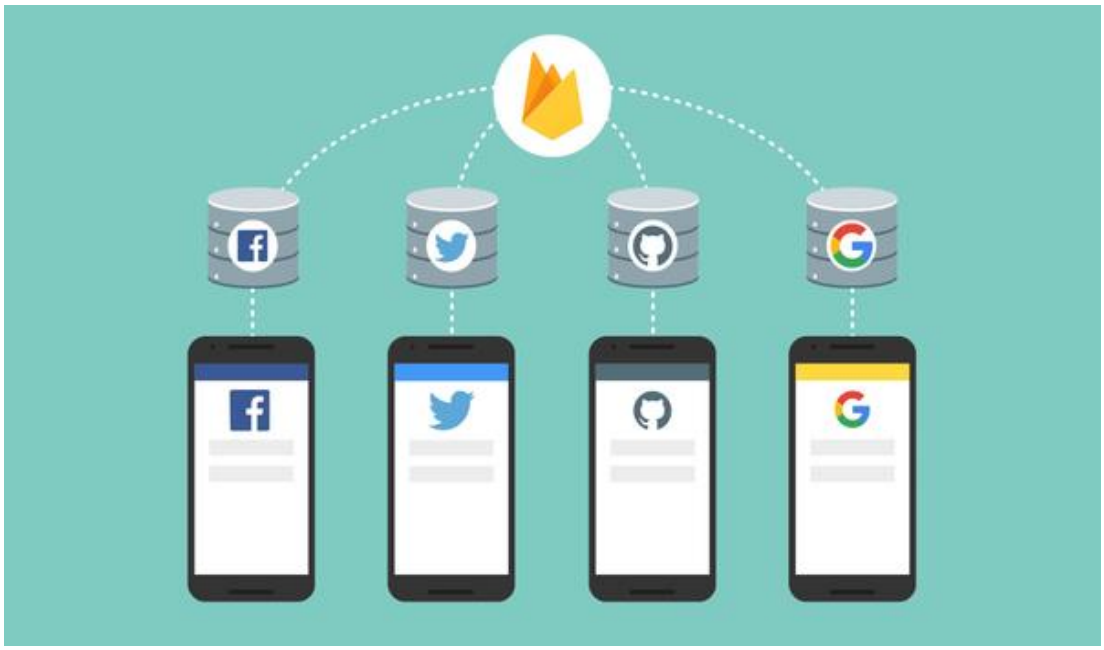
2.6.3. Realtime DataBase.

Khi đăng ký một tài khoản trên Firebase để tạo ứng dụng, bạn đã có một cơ sở dữ liệu thời gian thực. Dữ liệu bạn nhận được dưới dạng JSON. Đồng thời nó cũng luôn được đồng bộ thời gian thực đến mọi kết nối client.

Đối với các ứng dụng đa nền tảng, tất cả các client đều sử dụng cùng một cơ sở dữ liệu. Nó được tự động cập nhật dữ liệu mới nhất bất cứ khi nào các lập trình viên phát triển ứng dụng. Cuối cùng, tất cả các dữ liệu này được truyền qua kết nối an toàn SSL có bảo mật với chứng nhận 2048 bit.

Trong trường hợp bị mất mạng, dữ liệu được lưu lại ở local. Vì thế khi có mọi sự thay đổi nào đều được tự động cập nhật lên Server của Firebase. Bên cạnh đó, đối với các dữ liệu ở local cũ hơn với Server thì cũng tự động cập nhật để được dữ liệu mới nhất.

2.6.4. Authentication.



Hình 5: Minh hoạ Firebase Authentication.

Hoạt động nổi bật của Firebase là xây dựng các bước xác thực người dùng bằng Email, Facebook, Twitter, GitHub, Google. Đồng thời cũng xác thực nặc danh cho các ứng dụng. Hoạt động xác thực có thể giúp thông tin cá nhân của người sử dụng được an toàn và đảm bảo không bị đánh cắp tài khoản.

2.6.5. Firebase Hosting.

Hình 6: Minh hoạ Firebase Hosting.

Cách thức hoạt động cuối cùng của Firebase là cung cấp các hosting. Hosting được phân phối qua tiêu chuẩn công nghệ bảo mật SSL từ mạng CDN.

2.6.6. Một số dịch vụ của Firebase.

Nhóm công cụ phát triển và kiểm thử các ứng dụng được thiết kế và phát triển bao gồm các dịch vụ nổi bật sau:

- **Realtime Database**: là dịch vụ lưu trữ và đồng bộ dữ liệu người dùng thời gian thực. Có hỗ trợ cho Android, IOS, Web, C++, Unity và Xamarin. Người dùng có thể lưu trữ và lấy dữ liệu từ máy chủ rất dễ dàng.

- **Crashlytics**: là hệ thống theo dõi và lưu trữ thông tin lỗi của ứng dụng. Các thông tin lỗi sẽ được thu thập triệt để và trình bày hợp lý. Từ mỗi chu trình hoạt động đến khi xảy ra lỗi.

- **Cloud Firestore**: là dịch vụ lưu trữ và đồng bộ dữ liệu giữa người dùng và thiết bị quy mô toàn cầu. Dịch vụ sử dụng NoSQL được lưu trữ trên hạ tầng cloud.

- **Authentication**: là dịch vụ quản lý người dùng đơn giản và an toàn. Authentication cung cấp nhiều phương pháp xác thực email và mật khẩu Google, Facebook.

- **Cloud Functions**: là dịch vụ mở rộng ứng dụng bằng mã phụ trợ tùy chỉnh mà không cần quản lý và quy mô các máy chủ riêng.

- **Cloud Storage**: là dịch vụ có khả năng lưu trữ và chia sẻ nội dung do người dùng tạo ra như hình ảnh, âm thanh và video với bộ nhớ mạnh, đơn giản và tiết kiệm chi phí được xây dựng cho quy mô của Google.

- **Hosting**: Dịch vụ thuê hosting giúp đơn giản hóa lưu trữ web với các công cụ thực hiện cụ thể có tính năng cao dành cho các trang web hiện đại.

- **Test Lab for Android**: là công cụ tự động chạy thử và tùy chỉnh cho ứng dụng trên các thiết bị ảo và vật lý của Google cung cấp.

- **Performance Monitoring**: là dịch vụ có khả năng chẩn đoán các vấn đề xảy ra với hiệu suất ứng dụng.

Nhóm công cụ Firebase Grow & Engage Your Audience của Firebase bao gồm:

- **Google Analytics** sẽ phân tích thuộc tính và hành vi của người sử dụng trong bảng điều khiển. Cuối cùng sẽ đưa ra quyết định về lộ trình xây dựng ứng dụng. Ngoài ra nó còn nhận thông tin về thời gian thực từ báo cáo cho đến việc xuất dữ liệu sự kiện thô tới Google BigQuery để phân tích tùy chỉnh.

- **Cloud Messaging** là dịch vụ gửi tin, thông báo miễn phí đến user trên nền tảng Android, IOS và Web. Bạn có thể gửi tin nhắn đến các nhóm thiết bị, chủ đề hoặc phân đoạn người dùng cụ thể. Đây là công cụ gửi hàng tỷ thư mỗi ngày trên ứng dụng lớn nhỏ.

- **Predictions** là công cụ tạo ra nhóm người dùng năng động dựa trên hành vi dự đoán. Họ là đối tượng nhắm mục tiêu trong các sản phẩm khác như thông báo, cấu hình từ xa…

- **Firebase Dynamic Links** được áp dụng trên Android, IOS và Web, dịch vụ này sử dụng liên kết động để cung cấp trải nghiệm người dùng, hỗ trợ web di động thúc đẩy chuyển đổi ứng dụng gốc, người dùng chia sẻ người dùng, các chiến dịch xã hội và tiếp thị…

- **Remote Config** là dịch vụ tùy chỉnh cách ứng dụng khi hiển thị cho mỗi người dùng. Nó được dùng để thay đổi giao diện, triển khai dần dần các tính năng, chạy thử nghiệm A/B, cung cấp nội dung tùy chỉnh cho người dùng hoặc thực hiện các cập nhật khác mà không cần triển khai phiên bản mới nhất từ bảng điều khiển.

- Invites là công cụ cho phép người dùng chia sẻ mã giới thiệu, nội dung yêu thích trên ứng dụng qua email và SMS. Dịch vụ này kết hợp với Google Analytics for Firebase để bạn biết thời điểm người dùng mở hay cài đặt ứng dụng qua lời mời.

- App Indexing là dịch vụ tích hợp với công cụ tìm kiếm Google giúp thu hút người dùng đã sử dụng các ứng dụng trước.

- AdMob là công cụ hiển thị quảng cáo hấp dẫn giúp lập trình viên kiếm tiền hiệu quả. Ngoài ra, dịch vụ này có thể thực hiện các chiến lược kiếm tiền trên lớp bậc nhất và tối đa hóa doanh thu do mỗi người dùng tạo ra.

- AdWords giúp có được thông tin và thu hút người dùng. Điều này thực hiện bằng cách chạy các quảng cáo trên tìm kiếm, hiển thị và video. Ngoài ra, bạn có thể cải thiện nhắm mục tiêu quảng cáo, tối ưu hiệu suất chiến dịch của bạn.

2.6.7. Ưu điểm và hạn chế.

Ưu điểm:

- Tạo tài khoản và sử dụng dễ dàng

Firebase cho phép người dùng đăng nhập bằng tài khoản Google đơn giản. Gói Spark của Firebase miễn phí và cung cấp nhiều tính năng để giúp các nhà phát triển bắt đầu sử dụng. Khi yêu cầu ngày càng tăng, người dùng có thể chọn gói Blaze có trả phí với nhiều tính năng nâng cao hơn.

- Tốc độ phát triển nhanh

Firebase là một tùy chọn phát triển ứng dụng phù hợp có thể giúp các nhà phát triển tiết kiệm thời gian và giảm thời gian tiếp thị ứng dụng.

Thông thường, mọi Developer cần có quyền truy cập vào Server và Host để tạo, bảo trì cơ sở dữ liệu và dịch vụ phụ trợ. Do đó, cần có một Backend Developer và một Frontend Developer để xây dựng các ứng dụng. Tuy nhiên, việc này thường có thể dẫn đến lỗi và các vấn đề có thể gây ra sự cố ứng dụng và làm tăng chi phí phát triển.

Thông qua việc sử dụng Firebase và Firestore, Frontend Developer có thể quản lý, giảm thời gian cần thiết để hoàn thành tất cả công việc.

- Nhiều dịch vụ trong một nền tảng

Firebase cũng cung cấp danh sách đầy đủ các sản phẩm để hỗ trợ các Developer trong quá trình phát triển.

Hai tùy chọn cơ sở dữ liệu là Firestore và Realtime Database của Firebase. Tương tự như vậy, Firebase cho phép bạn thực hiện lưu trữ Cloud Media dễ dàng. Nó cũng cho phép phát triển ứng dụng không cần máy chủ thông qua việc tích hợp Cloud Functions.

Firebase bao gồm toàn bộ chu trình phát triển ứng dụng. Nền tảng này chứa các tính năng để xây dựng, phát hành và giám sát các ứng dụng. Ngoài ra, là bước cuối cùng của chu trình phát triển ứng dụng, nó cung cấp các công cụ để thu hút người dùng và giữ họ sử dụng nó. Đây cũng là ưu điểm của Firebase mà tôi đánh giá khá cao.

- Được cung cấp bởi Google

Firebase được cung cấp bởi Google, một trong những tên tuổi nổi bật và đáng tin cậy nhất trong thế giới công nghệ. Theo tôi được biết, kể từ khi được mua lại, Firebase đã trải qua một loạt thay đổi, phát triển và trở thành nền tảng đáng tin cậy như ngày nay. Nó khai thác sức mạnh của Google Cloud và nhiều dịch vụ của Google.

Firebase hiện là một phần của Google Cloud Platform. Nó hoạt động tốt với các dịch vụ Google Cloud khác và tích hợp với nhiều dịch vụ của bên thứ ba.

- Tập trung vào phát triển giao diện người dùng

Một ưu điểm khác khiến tôi yêu thích Firebase đó là nó cho phép Developer tập trung vào việc tạo mã Frontend cho các ứng dụng di động. Nền tảng này giúp cho việc phát triển ứng dụng trở nên thuận tiện và giúp giảm chi phí đáng kể.

Sử dụng Firebase cũng cho phép các công ty, Developer chuẩn hóa môi trường Backend theo một công nghệ duy nhất và dễ học. Mẫu Backend làm giảm số lượng đào tạo cần thiết để hỗ trợ nó và cho phép Developer tập trung vào phát triển giao diện người dùng.

- Firebase không có máy chủ

Việc mở rộng hoặc giảm quy mô máy chủ không phải là một nhiệm vụ dễ dàng. Đặc biệt, việc mở rộng một cụm cơ sở dữ liệu là một thách thức và việc tối ưu hóa hiệu suất cho khối lượng công việc khổng lồ đòi hỏi các chuyên gia có kinh nghiệm.

Firebase giải quyết vấn đề này và cung cấp một môi trường hoàn toàn không có máy chủ. Firebase đi kèm với kiến trúc không máy chủ (Serverless Architecture). Do đó, người dùng sẽ không cần lo lắng về cơ sở hạ tầng máy chủ.

- Học máy (Machine Learning)

Theo Gartner, 30% doanh nghiệp sẽ sử dụng Machine Learning (ML) trong một phần quy trình của họ. Firebase cũng có lợi vì nó cung cấp cho các nhà phát triển tùy chọn để dựa vào Machine Learning.

Firebase đi kèm với bộ ML với các API sẵn có cho các tính năng khác nhau của nền tảng di động như nhận dạng văn bản, nhận diện khuôn mặt, ghi nhãn hình ảnh, quét mã vạch, …

- Tạo lưu lượng truy cập

Firebase hỗ trợ lập chỉ mục ứng dụng để cho phép người dùng thu hút lại những khách tìm kiếm trên Google. Cụ thể, nó cung cấp liên kết ứng dụng trên mục tìm kiếm của Google.

Xếp hạng ứng dụng cũng có thể được nâng cao trên Tìm kiếm một lần bằng cách lập chỉ mục ứng dụng. Điều này giúp ứng dụng của bạn tiếp xúc với những người dùng mới có thể cài đặt ứng dụng đó.

- Theo dõi lỗi

Tiếp theo là tính năng Crashlytics của Firebase. Đây là một công cụ tuyệt vời để tìm và khắc phục sự cố một cách nhanh chóng. Firebase có thể theo dõi cả lỗi không nghiêm trọng và lỗi nghiêm trọng. Đồng thời, báo cáo được tạo dựa trên mức độ ảnh hưởng của lỗi đến trải nghiệm của người dùng.

- Sao lưu

Firebase đảm bảo tính bảo mật tối ưu và tính sẵn có của dữ liệu nhờ các bản sao lưu thường xuyên. Các ứng dụng được bảo vệ khỏi mọi khả năng mất dữ liệu bằng cách dựa vào tính năng sao lưu tự động của nền tảng này.

Khi sử dụng gói Blaze, có thể dễ dàng định cấu hình Cơ sở dữ liệu thời gian thực của Firebase để thực hiện các bản sao lưu tự động.

Hạn chế:

- Không phải là mã nguồn mở

Firebase không phải là một tùy chọn mã nguồn mở để phát triển ứng dụng di động. Điều này khiến nó trở thành một lựa chọn không tối ưu cho nhiều nhà phát triển. Người dùng không thể sửa đổi mã nguồn Firebase. Theo tôi thì đây là hạn chế lớn nhất của Firebase và ngăn cộng đồng cải tiến sản phẩm.

- Người dùng không có quyền truy cập mã nguồn

Đây là một hạn chế lớn khác của Firebase cho người dùng. Việc không có quyền truy cập vào mã nguồn có thể khá khó khăn đối với một số nhà phát triển. Đặc biệt, đối với các ứng dụng lớn, việc chuyển sang các nhà cung cấp khác không phải là một nhiệm vụ dễ dàng và để thực hiện việc này, toàn bộ Backend sẽ cần được xây dựng lại từ đầu.

- Firebase không hoạt động ở nhiều quốc gia

Firebase là một Subdomain của Google. Trang Web chính thức của nó là https://firebase.google.com và bị chặn ở nhiều quốc gia.

- Chỉ hoạt động với Cơ sở dữ liệu NoSQL

Cả Firestore và Firebase Realtime Database đều cung cấp cấu trúc NoSQL. Chúng không có tùy chọn nào để sử dụng cơ sở dữ liệu quan hệ.

Mặc dù Firebase đã thực hiện những cải tiến đáng kể trên Cloud Firestore so với RTDB, nhưng việc chạy các truy vấn phức tạp vẫn là một thách thức đối với một số người dùng nhất định.

Với Firebase, người dùng không thể xử lý nhanh chóng việc di chuyển dữ liệu tương tự như cơ sở dữ liệu SQL đơn giản. Firebase sử dụng JSON và hầu như không có tính năng SQL nào. Vì vậy, việc di chuyển từ cơ sở dữ liệu sẽ không dễ dàng.

- Truy vấn chậm

Không phải tất cả cơ sở dữ liệu đều lý tưởng cho mọi trường hợp sử dụng và Cloud Firestore cũng không ngoại lệ. Mặc dù đó là một giải pháp tuyệt vời để mở rộng ứng dụng của bạn, cung cấp hỗ trợ truy vấn ngoại tuyến và cập nhật thời gian thực, nhưng nó cũng có những hạn chế như:

- Kích thước yêu cầu API tối đa 10 MiB.
- Không có truy vấn tổng hợp gốc.
- Giới hạn kích thước tài liệu là 1 MiB.
- Kết nối đồng thời tối đa 1M.
- Truy vấn hiệu suất chậm.

- Không phải tất cả các dịch vụ Firebase đều miễn phí

Các chức năng đám mây chỉ khả dụng trên gói Blaze và người dùng không thể dùng thử dịch vụ trong gói Spark.

API Cloud Vision trong tính năng học máy cũng không khả dụng trong gói Spark miễn phí của nền tảng này … Dịch vụ này chỉ có sẵn cho người dùng của gói Blaze và tính phí 1,5 USD/ k API.

- Firebase khá đắt và giá không ổn định

Firebase là một nền tảng phát triển ứng dụng di động có nhiều tính năng nhưng cũng là một nền tảng đắt tiền đối với một số người. Nó cung cấp một gói miễn phí nhưng có hạn chế nhất định.

Firebase cung cấp nhiều tính năng hữu ích hơn nhưng giá cũng sẽ cao hơn so với một nhà cung cấp IaaS thuần túy. Nói chung, tôi cho rằng việc tự lưu trữ một ứng dụng sẽ rẻ hơn so với sử dụng Firebase.

Trước khi sử dụng Firebase bạn nên so sánh chính xác chi phí trả trước của việc thiết lập một cụm tại nhà cung cấp IaaS và chi phí đang thực hiện để duy trì máy chủ so với chạy ứng dụng trực tiếp trên Firebase.

- Chỉ chạy trên Google Cloud

Firebase hiện là một phần của Google và cơ sở hạ tầng của nó chạy hoàn toàn trên Google Cloud. Bạn sẽ không có tùy chọn để chạy Firebase trên các nhà cung cấp đám mây khác như AWS, Azure hoặc Digital Ocean.

- Thiếu Dedicated Servers và hợp đồng doanh nghiệp

Firebase không có tùy chọn Dedicated Servers hoặc hợp đồng doanh nghiệp. Cách duy nhất để sử dụng Firebase là sử dụng cấu trúc Serverless ít tính linh hoạt hơn.

Một hạn chế mà tôi thấy khá bất tiện khi dùng Firebase là việc không cung cấp các gói định giá, hợp đồng hoặc hỗ trợ doanh nghiệp. Tuy nhiên, gói Blaze có thể được coi là một phương án chấp nhận được. Nhưng tôi vẫn hy vọng có thêm nhiều lựa chọn và sự hỗ trợ hơn.

- Không cung cấp các API GraphQL

Firebase không cung cấp API GraphQL như một phần của thiết lập tiêu chuẩn. Mặc dù có những giải pháp thay thế cho việc triển khai GraphQL với Firebase, REST vẫn là tùy chọn mặc định của nền tảng.

2.7. NodeJS

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

2.7.1. Giới thiệu chung.

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

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

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

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

Các đặc tính của NodeJS:

- **Không đồng bộ:** None blocking – không đồng bộ có trên tất cả các API của NodeJS nó chủ yếu dựa trên server giúp trả dữ liệu về. Việc di chuyển trên máy chủ của các API theo sau khi cơ chế thông báo các sự kiện của NodeJS Server và chờ đợi server trả dữ liệu về. Việc di chuyển máy chủ đến các API sau khi gọi và cơ chế thông báo các sự kiện của NodeJS giúp máy chủ có thể phản ứng từ các cuộc gọi API trước (thời gian thực)

- **Chạy rất nhanh:** Node được phát triển vững chắc dựa trên nền tảng Java Script Engine do vậy việc thực thi chương trình diễn ra nhanh chóng.

- **Đơn luồng tuy nhiên khả năng mở rộng cao:** NodeJS mang đến một mô hình luồng dữ liệu duy nhật với sự kiện lặp với cơ chế tổ chức sự kiện giúp các máy chủ đáp ứng một cách ngăn chặn và làm cho máy chủ có khả năng mở rộng với các máy chủ truyền thống và tạo đề hạn chế để xử lý yêu cầu.

NodeJS sử dụng một chương trình đơn luồng và các chương trình tương tự giúp thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu tại các máy chủ truyền thống như Apache HTTP Server.

- **Không đệm:** NodeJS không đệm bất kỳ một dữ liệu thông qua các ứng dụng này chủ yếu là đầu ra của dữ liệu.

2.7.2. Ứng dụng.

Tuy nhiên NodeJS không phải là vạn năng, ví dụ như một ứng dụng cần tính ổn định cao, logic phức tạp thì các ngôn ngữ PHP hay Ruby… vẫn là sự lựa chọn tốt hơn. Còn dưới đây là những ứng dụng có thể và nên viết bằng NodeJS:

- Websocket server: Các máy chủ web socket như là Online Chat, Game Server…

- Fast File Upload Client: là các chương trình upload file tốc độ cao.

- Ad Server: Các máy chủ quảng cáo.

- Cloud Services: Các dịch vụ đám mây.

- RESTful API: đây là những ứng dụng mà được sử dụng cho các ứng dụng khác thông qua API.

- Any Real-time Data Application: bất kỳ một ứng dụng nào có yêu cầu về tốc độ thời gian thực. Micro Services: Ý tưởng của micro services là chia nhỏ một ứng dụng lớn thành các dịch vụ nhỏ và kết nối chúng lại với nhau. NodeJS có thể làm tốt điều này.

2.7.3. Ưu điểm và hạn chế.

Ưu điểm:

NodeJS có tốc độ rất nhanh: Nếu bạn đang là một startup thì nó có vai trò cực quan trọng bởi bạn đang cố gắng tạo ra một sản phẩm lớn và bạn muốn đảm bảo rằng nó có thể mở rộng nhanh chóng, và đáp ứng đầy đủ số lượng lớn người dùng khi trang web của bạn phát triển lên.

NodeJS có tốc độ xử lý nhiều kết nối trong một khoảng thời gian đồng thời trong khi PHP sẽ chỉ có nước sụp đổ. Do vậy ngoài các lợi ích về tốc độ thực thi và khả năng mở rộng thì bạn cũng có thể biết một chút về JavaScript, do vậy bạn sẽ dễ dàng tìm hiểu và học thêm về một ngôn ngữ lập trình hoàn toàn mới như PHP?

Hạn chế:

Cũng giống như đa phần các công nghệ mới, quá trình triển khai NodeJS trên host không phải là điều dễ dàng. Nếu bạn đang sử dụng một web hosting xài chung, bạn không thể sử dụng đơn giản bằng việc tải lên một ứng dụng NodeJS và mong chờ nó hoạt động tốt. VPS và dedicated

server được xem là một sự lựa chọn tốt hơn – bạn có thể cài đặt NodeJS trên chúng.

Việc bạn có thể dễ dàng cài đặt NodeJS chạy toàn bộ trên máy tính của bạn và sử dụng các hệ điều hành như Windows, Mac hoặc Linux và bạn sẽ bắt đầu sử dụng và phát triển ứng dụng này ngay lập tức – chỉ việc tải phiên bản NodeJS tương ứng. Một điều quan trọng bạn cần lưu ý là NodeJS không đơn giản là một sự lựa chọn thay thế cho Apache – các ứng dụng web đã và đang tồn tại sẽ không có khả năng tương thích, và bạn sẽ làm việc hiệu quả với những ứng dụng được phát triển từ đầu (mặc dù có rất nhiều framework ngoài kia có thể giúp đỡ bạn với nhiều đặc trưng phổ biến).

CHƯƠNG III.
PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1. Tổng quan hệ thống.

3.1.1. Mô tả chung.

Ứng dụng thi trắc nghiệm đảm bảo tính tiện dụng đối với học viên và quản trị viên. Ứng dụng giúp học viên có thể kiểm tra khả năng hiện tại của mình qua các bài trắc nghiệm, thi thử trên ứng dụng. Đồng thời giúp quản trị viên quản lí dễ dàng hơn hệ thống câu hỏi.

3.1.2. Mô tả chức năng.

3.1.2.1. Đăng nhập.

Mỗi học viên sẽ được cung cấp tài khoản theo thông tin mà bản thân đã đăng ký trong khoá học tại trung tâm, tài khoản quản trị viên sẽ có phân quyền để quản lí các thông tin hiển thị trên ứng dụng.

3.1.2.2. Thông tin.

Mỗi tài khoản đều có thể quản lí thông tin cá nhân tài khoản, Quản trị viên có thể quản lí thông tin các tài khoản khác.

3.1.2.3. Kiểm tra trắc nghiệm.

- Các học viên có thể tham gia các bài kiểm tra trắc nghiệm.
- Các câu hỏi trắc nghiệm sẽ được quản lí bởi quản trị viên.

3.1.2.4. Giới thiệu.

- Giới thiệu chung: hiển thị các thông tin về Trung tâm tin học.

- Các thông tin khác: hiển thị thông báo về về sự kiện hay các khoá học của trung tâm.
- Các thông tin có thể liên hệ tới Trung tâm để biết thêm về thông tin khoá học, cách thức đăng ký hay các vấn đề khác.

3.2. Phân tích yêu cầu.

3.2.1. Yêu cầu chức năng.

- Các bài thi trắc nghiệm kiểm tra khả năng và đưa ra hướng đi.
- Quản lý thông tin được ghi trong phần giới thiệu, quản lí câu hỏi, các bài kiểm tra trắc nghiệm.
- Quản lý tài khoản người dùng và phân quyền người dùng.

3.2.2. Yêu cầu phi chức năng.

- Xây dựng ứng dụng thân thiện, tiện lợi, dễ sử dụng.
- Yêu cầu bảo mật hệ thống: có chứng nhận quyền hạn người sử dụng.

3.3. Phân tích, thiết kế hệ thống.

3.3.1. Sơ đồ UseCase.

3.3.1.1. Danh sách và vai trò các actor.

Bảng 1: Danh sách và vai trò của actor.

STT	ACTOR	MÔ TẢ
1	Manager	Manager được phép thực hiện tất cả các chức năng trong hệ thống: đăng nhập đăng xuất, làm trắc nghiệm, các chức năng quản lí thông tin cá nhân nói riêng và các chức năng quản lí ứng dụng nói chung.
2	User	User bị giới hạn chức năng quản lí so với Manager khi chỉ có thể đăng nhập, đăng xuất, quản lí thông tin cá nhân và làm các bài trắc nghiệm.

3.3.1.2. Chức năng của từng usecase.

- Đăng nhập: đăng nhập vào hệ thống.

- Đăng xuất: đăng xuất khỏi hệ thống.

- Làm trắc nghiệm: làm các bài kiểm tra trắc nghiệm để kiểm tra trình độ của bản thân trước khi bước vào thi chính thức.

- Quản lí tài khoản (thông tin cá nhân): cập thật thông tin tài khoản, thêm và xoá tài khoản khi cần thiết.

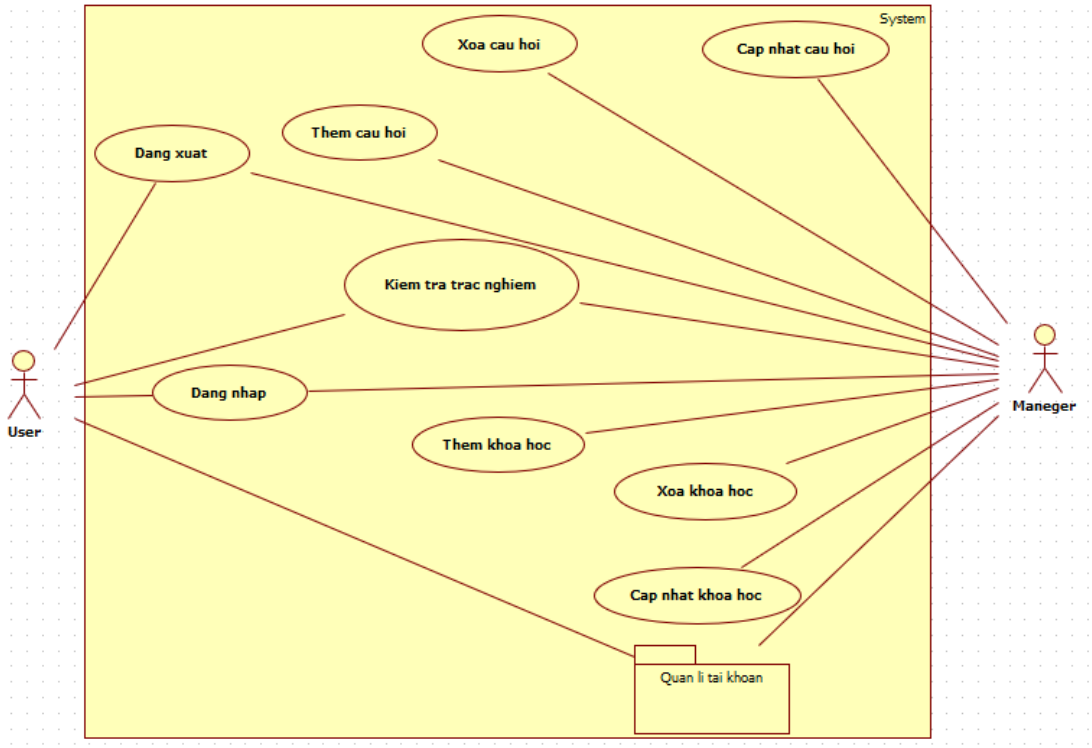
- Quản lí khoá học: thêm, xoá, sửa khoá học khi cần.

- Quản lí câu hỏi: thêm, xoá, sửa câu hỏi khi cần.

3.3.2. Mô tả Usecase.

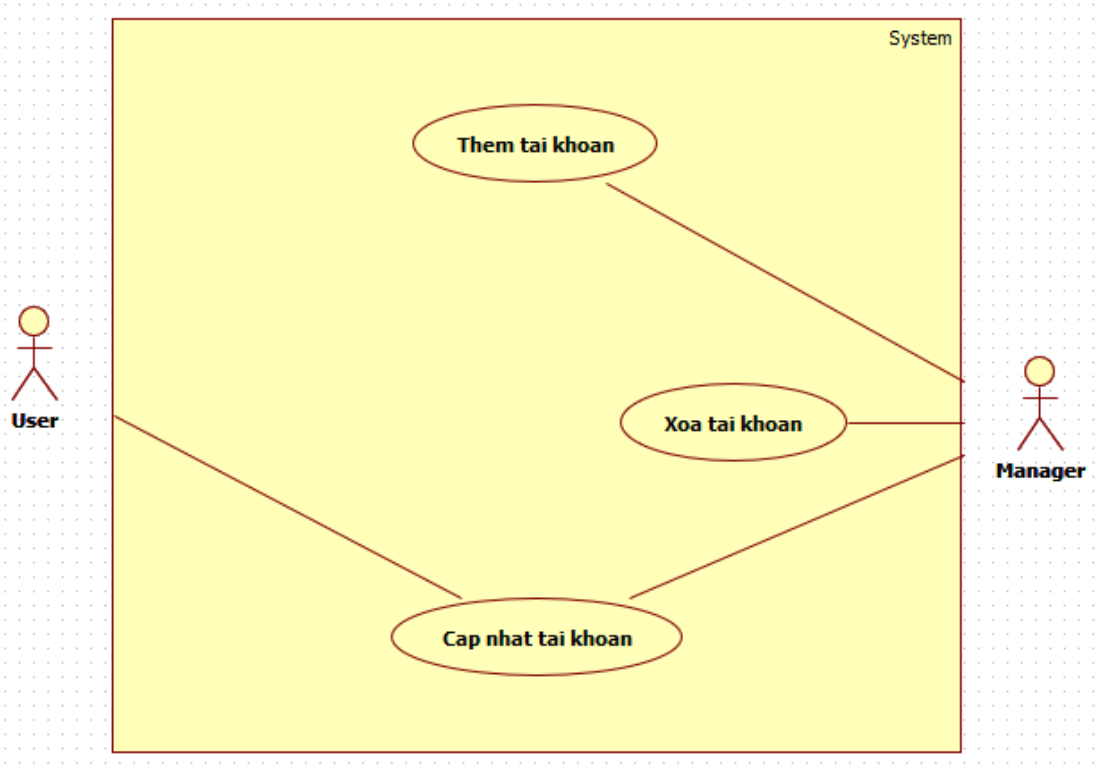
3.3.2.1. Sơ đồ Usecase.

-Mức 0:



Hình 7: Sơ đồ Usecase tổng quát.

+ Phân rã quản lí tài khoản:



Hình 8: Phân rã quản lí tài khoản.

3.3.2.2. Mô tả Usecase.

Bảng 2: Mô tả Usecase Đăng nhập.

Tên: Đăng nhập	ID: UC01
Tác nhận chính: User, Maneger.	
Mô tả ngắn gọn: Người dùng (User/Manager) đăng nhập vào hệ thống.	
Phạm vi: Người dùng đăng nhập vào hệ thống bằng ứng dụng.	
Điều kiện tiên quyết: Hệ thông hoạt động bình thường và người dùng có nhu cầu sử dụng hệ thống.	
Sự kiện kích hoạt: Người dùng chọn chức năng đăng nhập.	
Điều kiện thực hiện: Người dùng đã có sẵn tài khoản trên hệ thống.	
Luồng sự kiện chính: 1. Người dùng chọn chức năng đăng nhập. 2. Người dùng nhập thông tin.	

3. Hệ thống kiểm tra thông tin, nếu thông tin không hợp lệ thì thực hiện luồng thay thế. 4. Thông tin được xác nhận hợp lệ. 5. Người dùng được đưa tới trang chủ với giao diện tương ứng với phân quyền.
Luồng thay thế: 1. Hệ thống thông báo thông tin cả người dùng nhập sai. 2. Hệ thống cho phép người dùng nhập lại. 3. Hệ thống kiểm tra thông tin, thông tin không hợp lệ, khởi tạo lại luồng thay thế. 4. Thông tin được xác nhận hợp lệ.
Luồng ngoại lệ:
Các yêu cầu khác: - Hệ thống trả về kết quả nhanh chóng với độ trễ không quá 10 giây. - Không cho phép người dùng vào hệ thông khác với phân quyền.

Bảng 3: Mô tả Usecase Làm trắc nghiệm.

Tên: Làm trắc nghiệm	ID: UC02
Tác nhận chính: User, Manager.	
Mô tả ngắn gọn: User làm trắc nghiệm kiểm tra khả năng bản thân, Maneger kiểm tra thử hệ thống.	
Phạm vi: Người dùng sử dụng chức năng trắc nghiệm trong hệ thống.	
Điều kiện tiên quyết: Hệ thống đang hoạt động bình thường và người dùng có nhu cầu làm trắc nghiệm.	
Sự kiện kích hoạt: Người dùng chọn chức năng làm trắc nghiệm.	
Điều kiện thực hiện: Người dùng có tài khoản và đã đăng nhập hệ thống.	
Luồng sự kiện chính: 1. Người dùng chọn chức năng làm kiểm tra theo khoá học. 2. Hệ thông đưa người dùng tới giao diện làm trắc nghiệm.	

3. Người dùng làm trắc nghiệm. 4. Hệ thông duy trì trang thái xử lí câu hỏi và lưu câu hỏi đồng thời. 5. Người dùng kết thúc quá trình làm trắc nghiệm. 6. Hệ thống đưa người dùng tới trang kết quả. 7. Nếu trong quá trình làm trắc nghiệm, người dùng chọn thoát ra ngoài, hệ thống chạy luồng thay thế.
Luồng thay thế: 1. Hệ thống huỷ kết quả đang làm của người dùng. 2. Hệ thống trang thái xử lí câu hỏi khởi động lại. 3. Hệ thống đưa người dùng về lại trang chủ.
Luồng ngoại lệ:
Các yêu cầu khác: - Hệ thống trả về kết quả nhanh chóng. - Giao diện thân thiện, dễ sử dụng. - Ngăn không cho người dùng thoát giữa chừng bằng hay sử dụng cách nhớ kết quả.

Bảng 4: Mô tả Usecase Quản lí tài khoản.

Tên: Quản lí tài khoản	ID: UC03
Tác nhận chính: Manager.	
Mô tả ngắn gọn: Maneger quản lí hệ thống tài khoản.	
Phạm vi: Manager sử dụng chức năng quản lí tài khoản trong hệ thông.	
Điều kiện tiên quyết: Hệ thống đang hoạt động bình thường và Manager cần quản lí hệ thống tài khoản.	
Sự kiện kích hoạt: Manager chọn chức năng quản lí tài khoản.	
Điều kiện thực hiện: Manager có tài khoản và đã đăng nhập hệ thống.	
Luồng sự kiện chính: 1. Manager chọn chức năng quản lí tài khoản (thêm, xoá, sửa). 2. Manager nhập thông tin tài khoản cần quản lí.	

3. Hệ thông kiểm tra thông tin xem có hợp lệ hay không, nếu không, chuyển qua luồng thay thế. 4. Thông tin hợp lệ, hệ thống lưu lại thông tin. 5. Giao diện quản lí đưa về trang thái ban đầu.
Luồng thay thế: 1. Hệ thống báo cho Manager về thông tin không hợp lệ. 2. Hệ thông yêu cầu nhập lại. 3. Thông tin tiếp tục không hợp lệ, khởi tạo lại luồng thay thế. 4. Thông tin hợp lệ.
Luồng ngoại lệ:
Các yêu cầu khác: - Giao diện quản lí đơn giản, dễ sử dụng - Hệ thống trả về kết quả nhanh chóng.

Bảng 5: Mô tả Usecase Quản lí khoá học.

Tên: Quản lí khoá học	ID: UC04
Tác nhận chính: Manager.	
Mô tả ngắn gọn: Maneger quản lí hệ thống khoá học.	
Phạm vi: Manager sử dụng chức năng quản lí khoá học trong hệ thông.	
Điều kiện tiên quyết: Hệ thống đang hoạt động bình thường và Manager cần quản lí hệ thống khoá học.	
Sự kiện kích hoạt: Manager chọn chức năng quản lí khoá học.	
Điều kiện thực hiện:	
Luồng sự kiện chính: 1. Manager chọn chức năng quản lí khoá học (thêm, xoá, sửa). 2. Manager nhập thông tin khoá học cần quản lí. 3. Hệ thông kiểm tra thông tin xem có hợp lệ hay không, nếu không, chuyển qua luồng thay thế. 4. Thông tin hợp lệ, hệ thống lưu lại thông tin.	

5. Giao diện quản lí đưa về trang thái ban đầu.
Luồng thay thế: 1. Hệ thống báo cho Manager về thông tin không hợp lệ. 2. Hệ thông yêu cầu nhập lại. 3. Thông tin tiếp tục không hợp lệ, khởi tạo lại luồng thay thế. 4. Thông tin hợp lệ.
Luồng ngoại lệ:
Các yêu cầu khác: - Giao diện quản lí đơn giản, dễ sử dụng. - Hệ thống trả về kết quả nhanh chóng.

Bảng 6: Mô tả Usecase Quản lí câu hỏi.

Tên: Quản lí câu hỏi	ID: UC05
Tác nhận chính: Manager.	
Mô tả ngắn gọn: Maneger quản lí hệ thống câu hỏi.	
Phạm vi: Manager sử dụng chức năng quản lí câu hỏi trong hệ thông.	
Điều kiện tiên quyết: Hệ thống đang hoạt động bình thường và Manager cần quản lí hệ thống câu hỏi.	
Sự kiện kích hoạt: Manager chọn chức năng quản lí câu hỏi.	
Điều kiện thực hiện: Manager có tài khoản và đã đăng nhập hệ thống.	
Luồng sự kiện chính: 1. Manager chọn chức năng quản lí câu hỏi (thêm, xoá, sửa). 2. Manager nhập thông tin câu hỏi cần quản lí. 3. Hệ thông kiểm tra thông tin xem có hợp lệ hay không, nếu không, chuyển qua luồng thay thế. 4. Thông tin hợp lệ, hệ thống lưu lại thông tin. 5. Giao diện quản lí đưa về trang thái ban đầu.	
Luồng thay thế:	

1. Hệ thống báo cho Manager về thông tin không hợp lệ. 2. Hệ thông yêu cầu nhập lại. 3. Thông tin tiếp tục không hợp lệ, khởi tạo lại luồng thay thế. 4. Thông tin hợp lệ.
Luồng ngoại lệ:
Các yêu cầu khác: - Giao diện quản lí đơn giản, dễ sử dụng. - Hệ thống trả về kết quả nhanh chóng.

Bảng 7: Mô tả Usecase Cập nhật thông tin cá nhân.

Tên: Cập nhật thông tin cá nhân	ID: UC06
Tác nhận chính: User.	
Mô tả ngắn gọn: User, Maneger quản lí thông tin cá nhân.	
Phạm vi: Người dùng sử dụng chức năng cập nhật thông tin cá nhân trong hệ thông.	
Điều kiện tiên quyết: Hệ thống đang hoạt động bình thường và người dùng cần cập nhật thông tin cá nhân.	
Sự kiện kích hoạt: Người dùng chọn chức năng cập nhật thông tin cá nhân.	
Điều kiện thực hiện: Người dùng có tài khoản và đã đăng nhập hệ thống.	
Luồng sự kiện chính: 1. Người dùng chọn chức năng cập nhật lại thông tin cá nhân trong giới hạn cho phép. 2. Người dùng nhập thông tin thay thế và xác nhận thông tin. 3. Hệ thông kiểm tra thông tin, nếu không hợp lệ, chuyển qua luồn thay thế. 4. Thông tin hợp lệ, hệ thống lưu lại thông tin. 5. Giao diện thông tin cá nhân được cập nhật lại thông tin mới.	
Luồng thay thế: 1. Hệ thống báo cho người dùng về thông tin không hợp lệ.	

2. Hệ thông yêu cầu nhập lại. 3. Thông tin tiếp tục không hợp lệ, khởi tạo lại luồng thay thế. 4. Thông tin hợp lệ.
Luồng ngoại lệ:
Các yêu cầu khác: - Giao diện cập nhật đơn giản, dễ sử dụng. - Hệ thống trả về kết quả nhanh chóng.

3.3.3. Sơ đồ lớp.

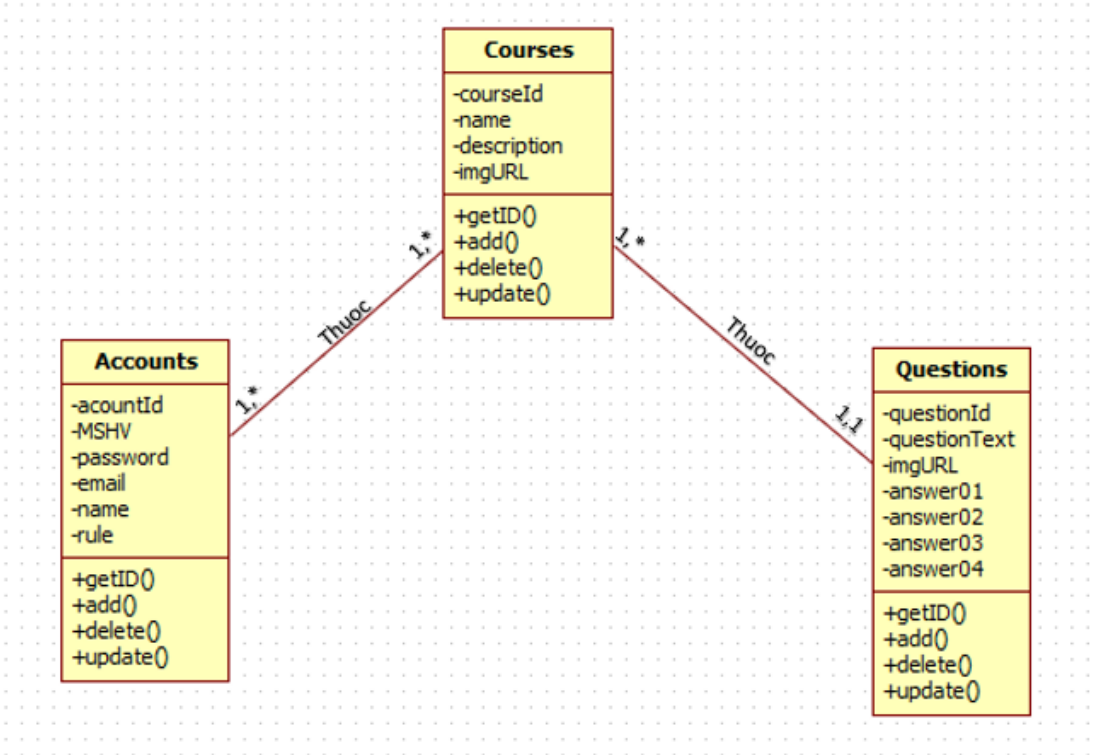
3.3.3.1. Các đối tượng trong hệ thống.

- Tài khoản: accountID, courseId, mã số học viên, mật khẩu, email, tên người dùng, phân quyền.

- Khoá học: courseID, tên khoá học, lời giới thiệu, đường dẫn ảnh.

- Câu hỏi: quesionID, courseId, nội dung câu hỏi, đường dẫn ảnh, câu trả lời 1, câu trả lời 2, câu trả lời 3, câu trả lời 4.

3.3.3.2. Sơ đồ lớp.



Hình 9: Sơ đồ lớp.

3.4. Thiết kế cơ sở dữ liệu.

Bảng 8: Courses.

STT	Tên thuộc tính	Kiểu dữ liệu	Null	Diễn giải
1	ID	String		Mã khoá học
2	Name	String		Tên khoá học
3	Description	String		Giới thiệu khoá học
4	imgURL	String		Đường dẫn ảnh của khoá học

Bảng 9: Accounts.

STT	Tên thuộc tính	Kiểu dữ liệu	Null	Diễn giải
1	ID	String		Mã tài khoản
2	MSHV	String		Mã số học viên
3	password	String		Mật khẩu tài khoản
4	Email	String	v	Email
5	Name	String		Tên người dùng
6	rule	String		Phân quyền

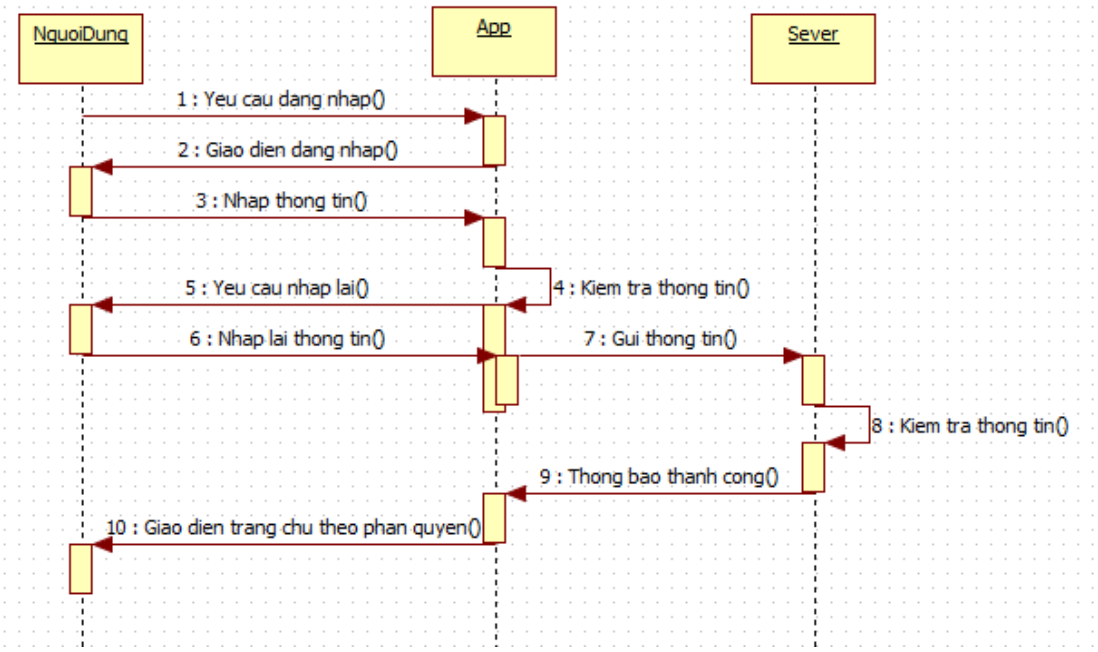
Bảng 10: Questions.

STT	Tên thuộc tính	Kiểu dữ liệu	Null	Diễn giải
1	ID	String		Mã câu hỏi
2	questionText	String		Nội dung câu hỏi
3	imgURL	String	v	Đường dẫn ảnh của câu hỏi
4	Answer01	String		Trả lời 1 (câu đúng)
5	Answer02	String		Trả lời 2
6	Answer03	String		Trả lời 3

7	Answer04	String		Trả lời 4

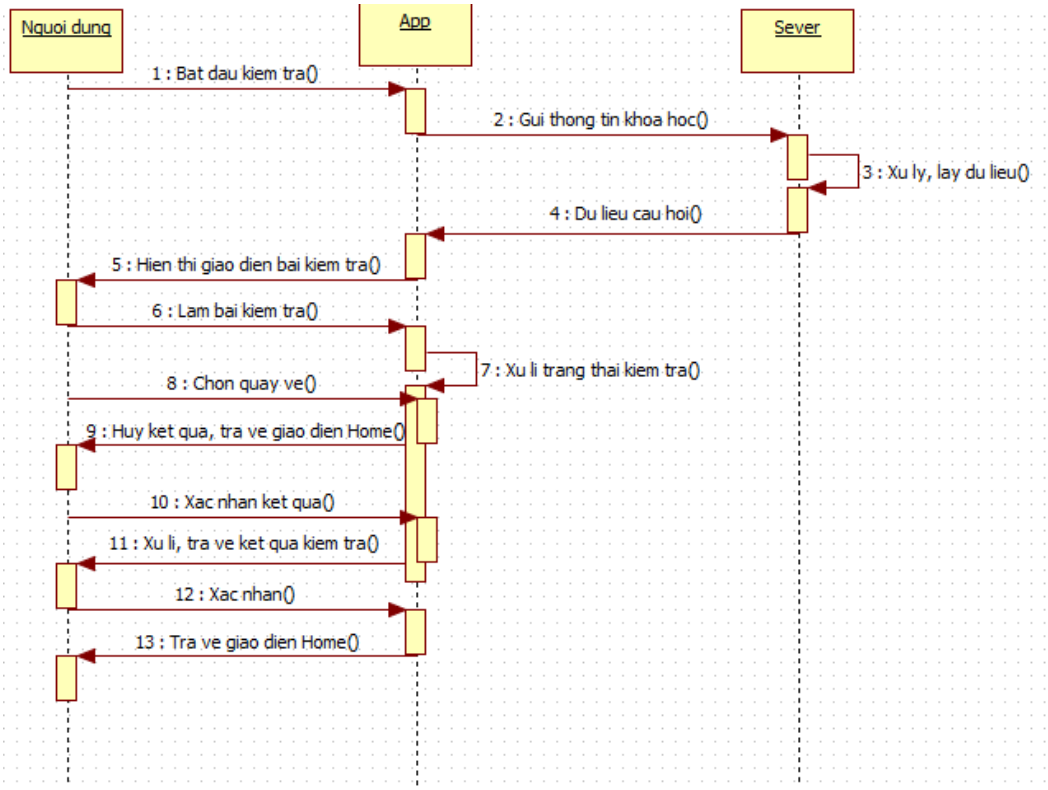
3.3.4. Sơ đồ tuần tự.

- Đăng nhập:



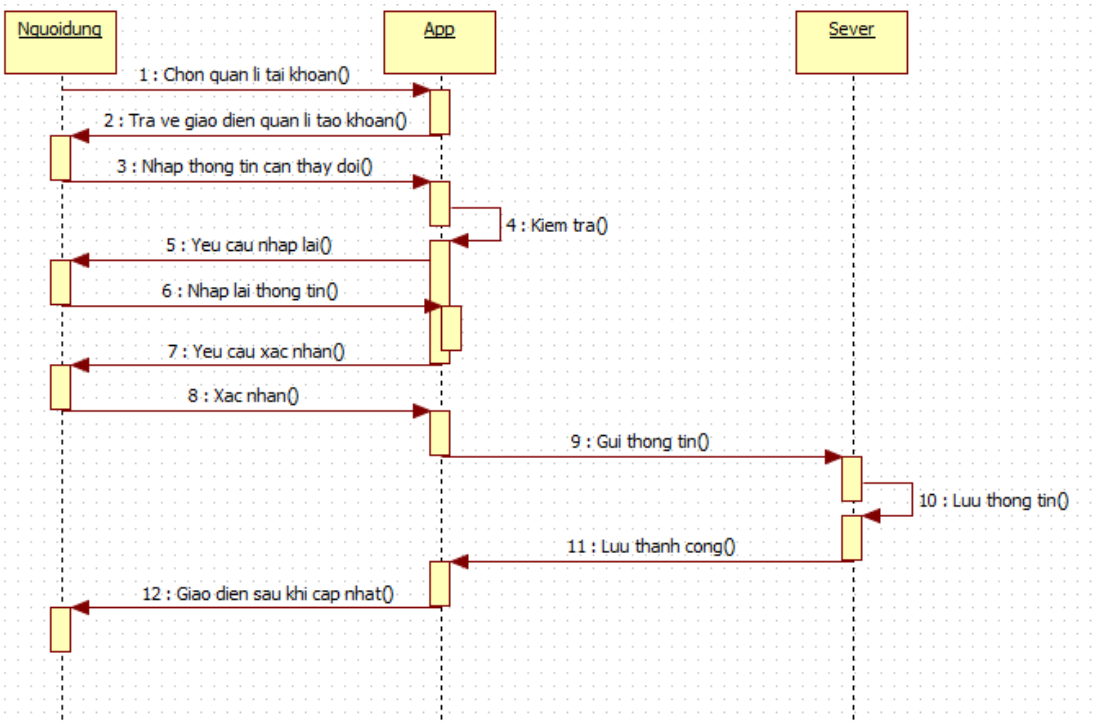
Hình 10: Sơ đồ tuần tự - Đăng nhập.

- Làm trắc nghiệm:



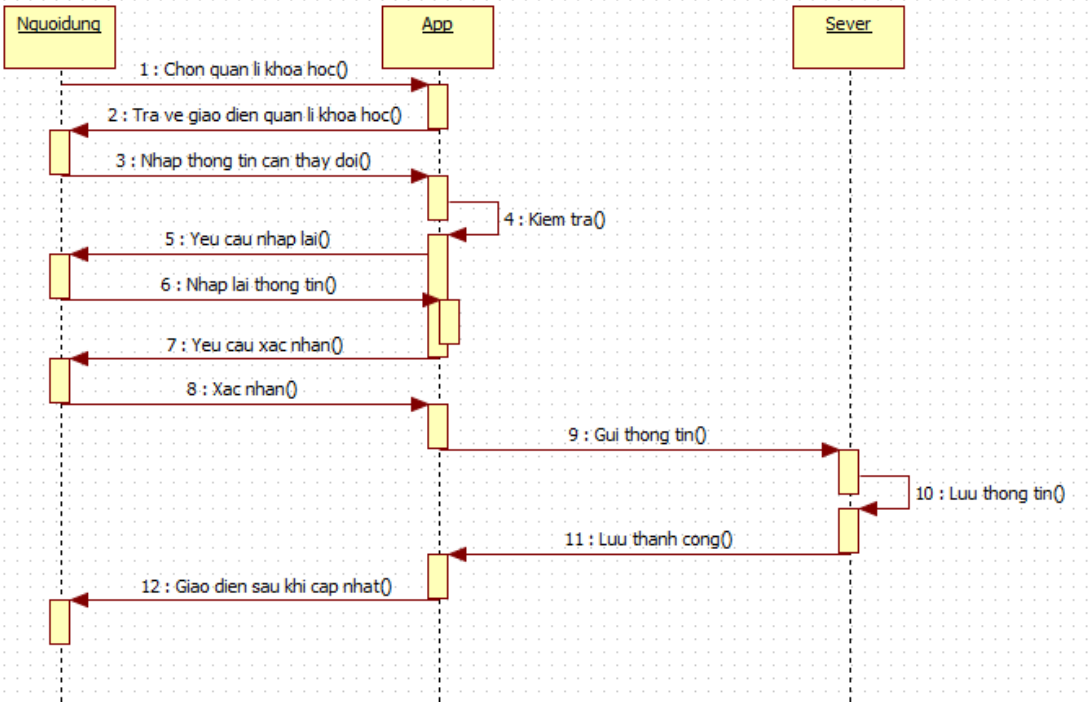
Hình 11: Sơ đồ tuần tự - Làm trắc nghiệm.

- Quản lí tài khoản:



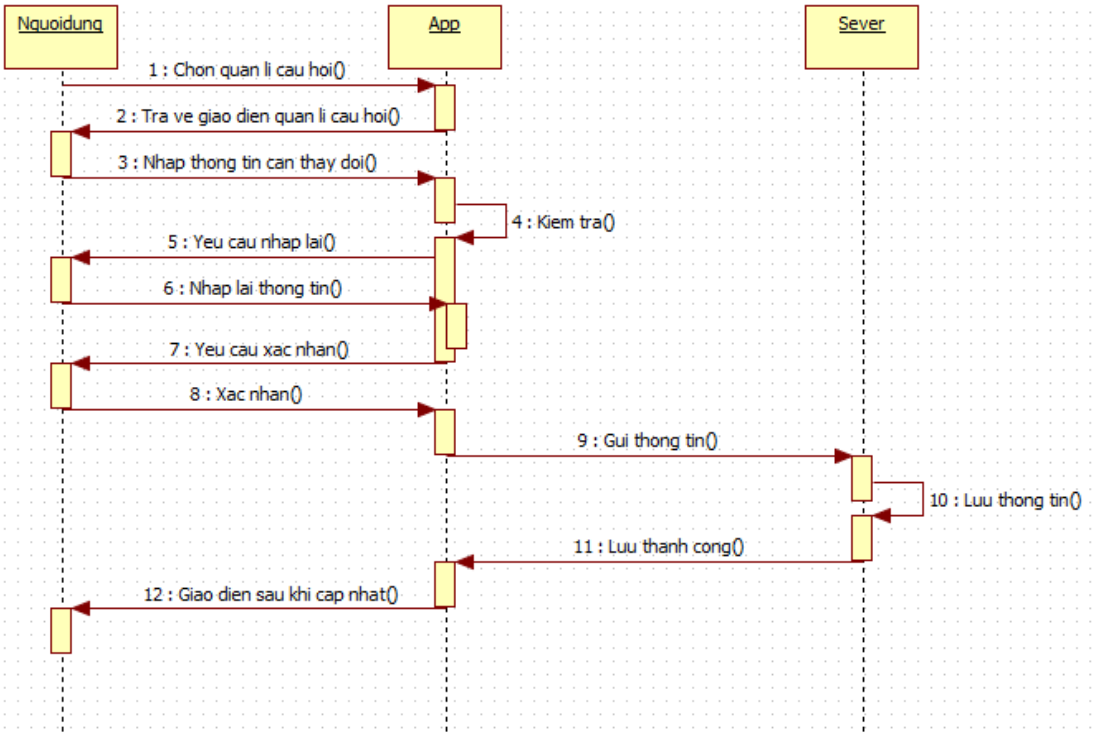
Hình 12: Sơ đồ tuần tự - Quản lí tài khoản.

- Quản lí khoá học:



Hình 13: Sơ đồ tuần tự - Quản lí khoá học.

- Quản lí câu hỏi:



Hình 14: Sơ đồ tuần tự - Quản lí câu hỏi.

3.5. Thiết kế giao diện.

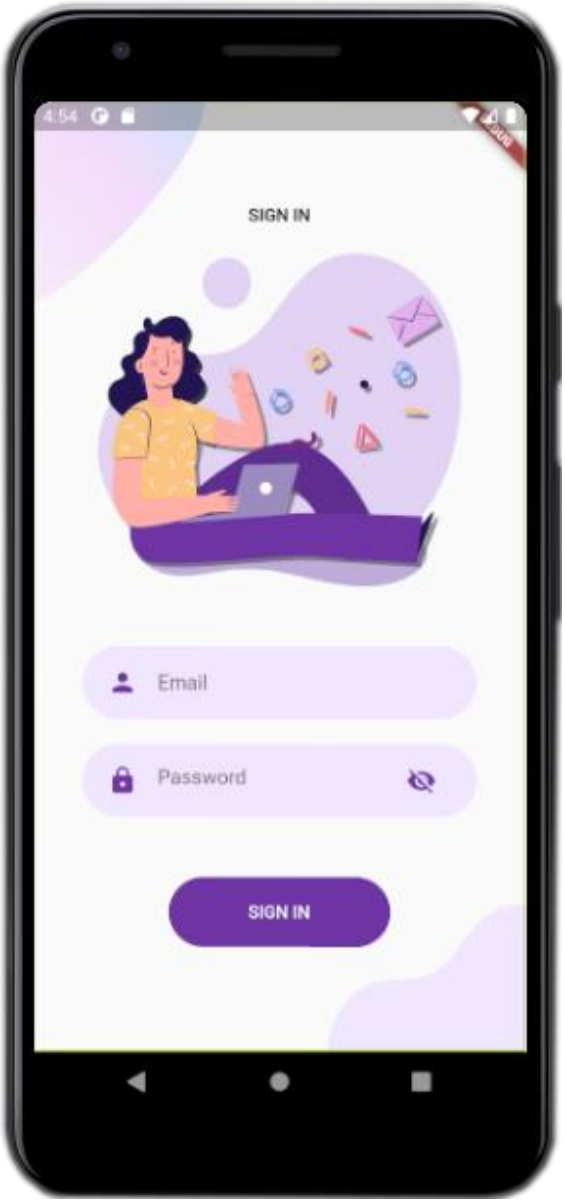
3.5.1. Giao diện Welcome.

Hình 15: Giao diện Welcome.

3.5.2. Giao diện giới thiệu.

Hình 16: Giao diện giới thiệu

3.5.3. Giao diện đăng nhập.

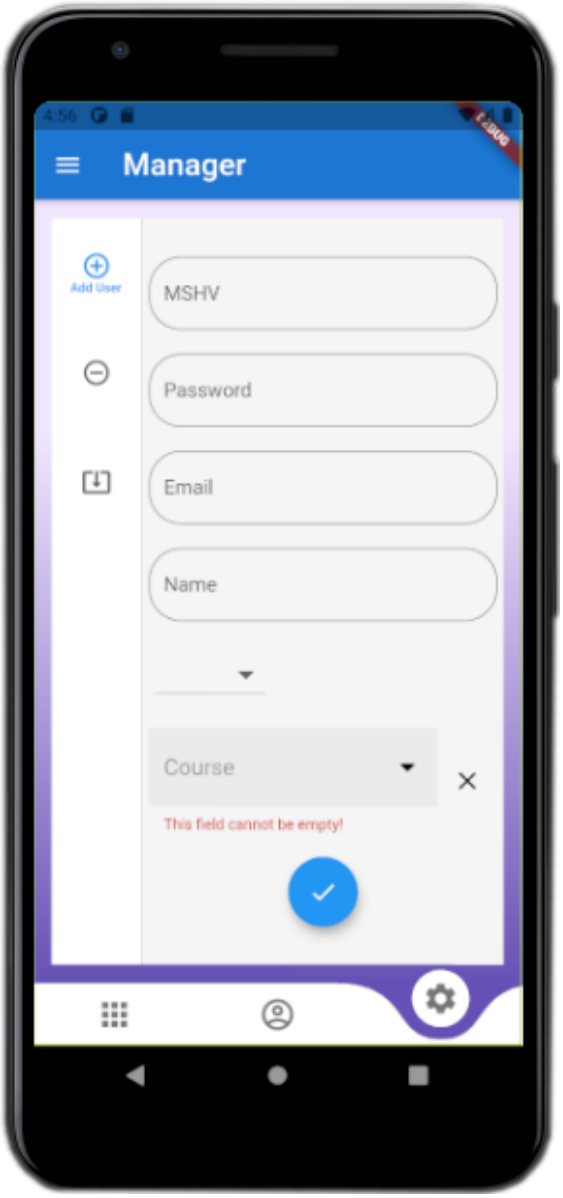


Hình 17: Giao diện đăng nhập.

3.5.4. Giao diện trang chủ.

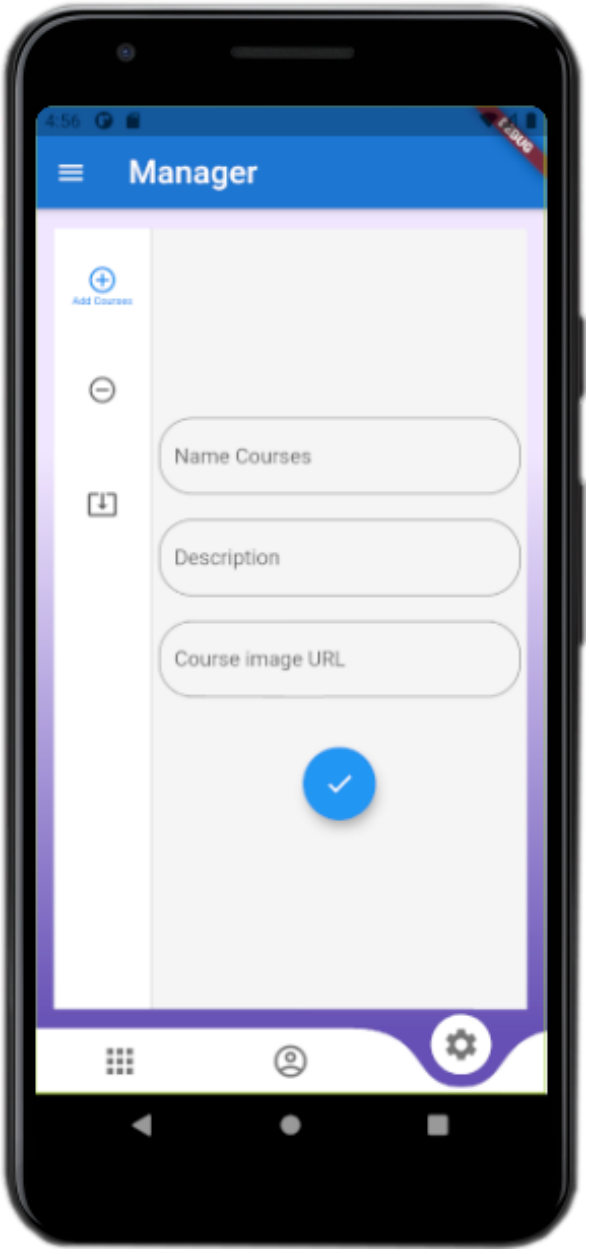
Hình 18: Giao diện trang chủ.

3.5.5. Giao diện quản lí người dùng.



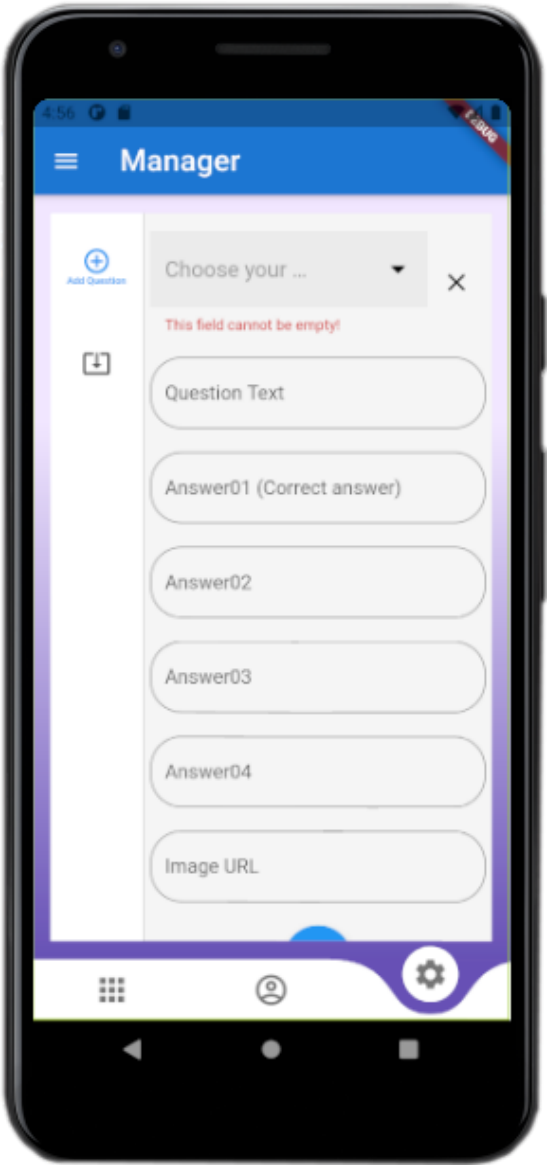
Hình 19: Giao diện quản lí người dùng.

3.5.6. Giao diện quản lí khoá học.



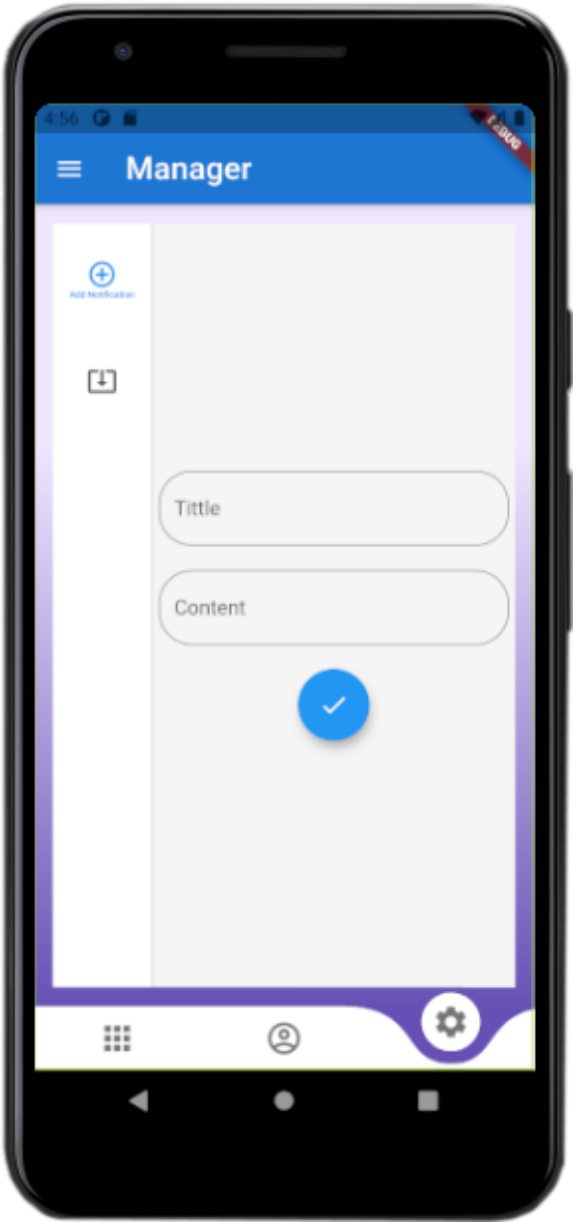
Hình 20: Giao diện quản lí khoá học.

3.5.7. Giao diện quản lí câu hỏi.



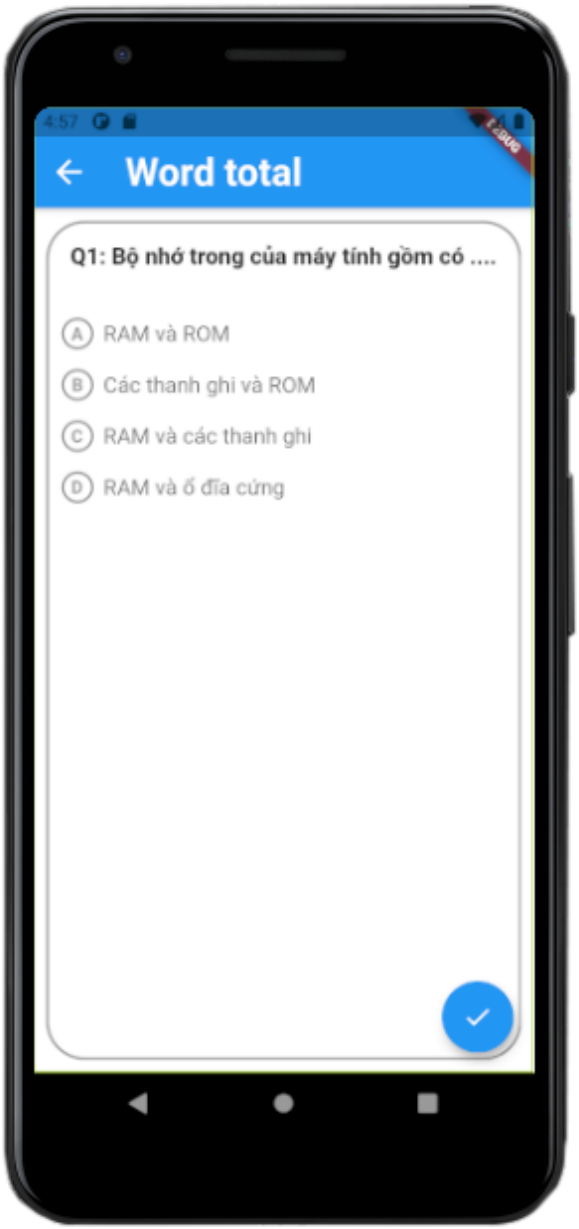
Hình 21: Giao diện quản lí câu hỏi.

3.5.8. Giao diện quản lí thông báo.



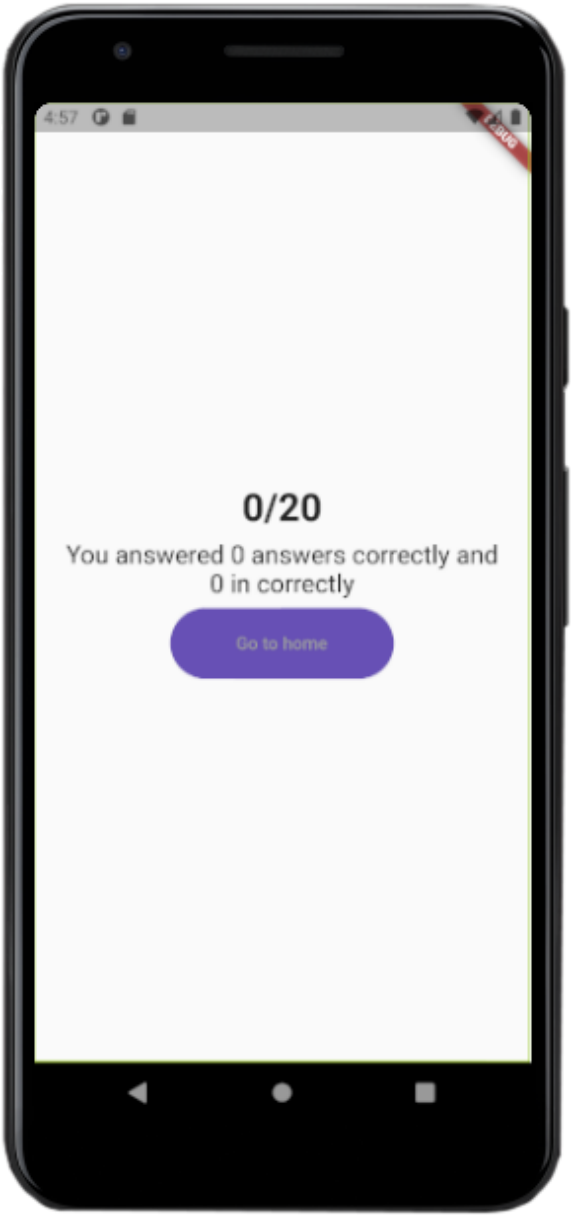
Hình 22: Giao diện quản lí thông báo.

3.5.9. Giao diện làm trắc nghiệm.



Hình 23: Giao diện làm trắc nghiệm.

3.5.10. Giao diện kết quả.



Hình 24: Giao diện kết quả.

3.6. Cài đặt chương trình.

- Yêu cầu hệ thống phát triển chương trình (máy tính):

+ Windows10 pro:

- RAM: 8GB.
- System type: x64bit.
- IDE: VSCode hoặc Android Studio.
- JDK 08.
- Flutter vs Dart plugin.

- Flutter vs Dart SDK.
- Android toolchain.
- Chrome.

+ Ubuntu:

- RAM: 8GB.
- System type: x64bit.
- IDE: VSCode và Android Studio.
- JDK 08.
- Flutter vs Dart plugin.
- Flutter vs Dart SDK.
- Android toolchain.
- Chrome.

- Yêu cầu hệ thống cài đặt chương trình (điện thoại):

- Android version 05 trở lên.
- Ram tối thiểu 2GB.
- MediaTek Helio P60 4 nhân.
- Speed CPU tối thiểu 2.0 GHz.
- Bộ nhớ trống tối thiểu 5 GB.

3.7. Kết luận và hướng phát triển.

3.7.1. Kết luận.

- Kết quả đạt được:

+ Hiểu rõ về cách hoạt động của một ứng dụng đa nền tảng.

+ Tìm hiểu về công nghệ mới, nắm bắt, áp dụng thực tiễn Flutter và Firebase để phát triển ứng dụng.

+ Mở rộng kiến thức về cơ sở dữ liệu phi quan hệ (noSQL), về ứng dụng thuần và ứng dụng đa nền tảng.

+ Xây dựng được một ứng dụng kiểm tra trắc nghiệm cơ bản với giao diện thân thiện, dễ sử dụng.

+ Vận dụng được kiến thức trên trường vào thực tiễn.

+ Tiền đề thực tiễn để định hướng công việc, có then kinh nghiệm cho công việc sau khi ra trường.

- Hạn chế:

+ Một số tính năng chưa được hoàn thiện.

+ Dữ liệu hiện thị trên giao diện chưa được đồng bộ trong phần quản lí.

+ Giao diện phần quản lí chưa được tối ưu cho cả web lẫn android.

3.7.2. Hướng phát triển.

- Tối ưu hoá hiệu suất cho ứng dụng khi xây dựng giao diện.

- Tối ưu hoá việc truy xuất khi liên kết với cơ sở dữ liệu.

- Hoàn thiện phần quản lí cùng một số chức năng chưa hoàn thiện và Xây dựng các chức năng mới phù hợp với hệ thống cùng Trung Tâm.

- Áp dụng thêm các công nghệ mới của các ông lớn đảm bảo hệ thống phát triển lâu dài, bền vững theo thời gian.

TÀI LIỆU THAM KHẢO

Danh Mục Tài Liệu Tham Khảo

1. Đỗ Đức Đình Đạt, 25/02/2021, Firebase là gì? Giải pháp lập trình không cần Backend từ Google, pp: https://wiki.matbao.net/firebase-la-gi-giai-phap-lap-trinh-khong-can-backend-tu-google/.
2. Mon, 09/03/2017, phân biệt giữa native app, cross platform và hybrid, pp: https://mona.solutions/phan-biet-giua-native-crossplatform-va-hybrid/.
3. Mon, 09/03/2018, API là gì, Những điểm nổi bật của web API trong lập trình website, pp: https://monamedia.co/api-la-gi-diem-noi-bat-cua-web-api/.
4. Nguyễn Thành Minh, 06/11/2020, Học Flutter từ cơ bản đến nâng cao. Phần 1: Làm quen cô nàng Flutter, pp: https://viblo.asia/p/hoc-flutter-tu-co-ban-den-nang-cao-phan-1-lam-quen-co-nang-flutter-4dbZNJOvZYM.
5. Trung Quân, 23/04/2018, Một cái nhìn tổng quan nhất về Nodejs, pp: https://viblo.asia/p/mot-cai-nhin-tong-quan-nhat-ve-nodejs-Ljy5VeJ3lra.