ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KĨ THUẬT MÁY TÍNH

PHÁT TRIỂN ỨNG DỤNG TRÊN THIẾT BỊ DI ĐỘNG (CO3043)

ASSIGNMENT 3 REPORT

MOBILE APP PUBLISHING AND MAINTENANCE

Giảng viên hướng dẫn: Hoàng Lê Hải Thanh
Nhóm: Harder4Better
Sinh viên: Đặng Quốc Phong - 2212548
Trần Chính Bách - 2210187
Hoàng Mạnh Đức - 2210787
Huỳnh Đức Nguyên - 2252542

TP. HỒ CHÍ MINH, THÁNG 5/2025

Mục lục

1 Giới thiệu

1.1 Tuyên bố vấn đề

Việc tổ chức sự kiện, dù là một buổi workshop nhỏ, hay các sự kiện cộng đồng tự phát, luôn đòi hỏi một quy trình nhất định: gửi lời mời, theo dõi người tham dự, điểm danh, và cuối cùng là tổng hợp, thống kê. Tuy nhiên, phần lớn những người tổ chức không chuyên - sinh viên, giáo viên, nhóm dự án, người làm cộng đồng, hoặc bất kỳ ai không thuê dịch vụ chuyên nghiệp - thường phải "xoay sở" với một loạt công cụ rời rạc như Google Forms, Google Sheets, gửi email thủ công, nhóm Zalo hoặc Facebook, và tự theo dõi trạng thái tham dự.

Với quy trình tổ chức thủ công, rời rạc, dễ sai sót khi số lượng người tham dự tăng lên. Thêm vào đó, người tổ chức mất nhiều thời gian cho các thao tác kỹ thuật phụ trợ thay vì tập trung vào nội dung và chất lượng sự kiện.

1.2 Mục tiêu và mục đích

Mini Event được xây dựng để giải quyết vấn đề trên: một ứng dụng trực quan, đủ mạnh để giúp người dùng tổ chức và quản lý sự kiện nhỏ đến vừa mà không cần dùng đến hàng loạt công cụ hỗ trợ thủ công khác.

Mục tiêu của ứng dụng không phải là tạo ra một hệ thống nặng nề hay chuyên biệt hóa theo mô hình doanh nghiệp, mà là tạo điều kiện cho bất kỳ ai - chỉ với một chiếc điện thoại - cũng có thể tự tổ chức sự kiện cho cộng đồng của mình một cách mạch lạc và kiểm soát được toàn bộ quy trình.

Cụ thể, ứng dụng tập trung vào ba khía cạnh chính:

- Cung cấp một nền tảng di động dễ sử dụng, nơi người dùng có thể tạo, chỉnh sửa và quản lý sự kiện một cách trực tiếp mà không cần máy tính hay kỹ năng kỹ thuật phức tạp.
- Hỗ trợ toàn bộ vòng đời sự kiện: từ gửi lời mời đến theo dõi danh sách người tham gia, điểm danh và xuất thống kê - tất cả gói gọn trong một ứng dụng duy nhất.
- Giảm thiểu sự phụ thuộc vào các công cụ bên ngoài như Google Form, Sheet, hoặc email thủ công, giúp tiết kiệm thời gian và đơn giản hóa quá trình tổ chức.

Mini Event không chỉ là một công cụ hỗ trợ, mà là một giải pháp tập trung vào người tổ chức không chuyên, giúp họ cảm thấy tự tin và chủ động hơn trong việc điều phối sự kiện – dù lớn hay nhỏ.

2 Cách tiếp cận & Phương pháp

2.1 Cách tiếp cận

Thay vì tập trung phát triển hệ thống theo chiều rộng ngay từ đầu, chúng em chọn lối tiếp cận theo chiều sâu: giải quyết trước một cách triệt để các chức năng thiết yếu (MVP), đồng thời tạo nền móng vững chắc để mở rộng trong tương lai. Cụ thể, các chức năng, giải pháp được thiết kế xoay quanh ba trụ cột chính:

- **Khả năng tiếp cận:** Người dùng có thể tham gia sự kiện chỉ với một vài thao tác đơn giản qua mobile app, đồng thời có thể check-in bằng nhiều phương thức (QR code, định vị GPS).

- **Kiểm soát và bảo mật:** Dữ liệu khách mời và trạng thái check-in được xử lý tập trung qua hệ thống backend, hạn chế rò rỉ và gian lận thông tin.
- **Trải nghiệm realtime & tương tác tốt:** Với việc tận dụng khả năng realtime của Firebase và giao diện mobile trực quan.

2.2 Phương pháp phát triển

Quy trình phát triển ứng dụng Mini Event được tổ chức theo phương pháp Agile Scrum, giúp nhóm linh hoạt thích ứng với thay đổi yêu cầu, kiểm soát tiến độ và cải tiến liên tục qua từng Sprint. Mỗi Sprint kéo dài 1–2 tuần, với các hoạt động tiêu chuẩn như Sprint Planning, Daily Standup và Review.

Về mặt kỹ thuật, hệ thống lựa chọn các công nghệ sau:

Backend:

- Sử dụng **Spring Boot** để xây dựng các RESTful API, xử lý logic nghiệp vụ, xác thực người dùng và tích hợp các dịch vụ bên ngoài (email, lưu trữ đám mây).
- Kết nối **Firebase** (Authentication, Firestore, Realtime Database) để lưu trữ dữ liệu thời gian thực, giảm thiểu độ trễ và tối ưu hiệu năng đọc/ghi theo thời gian thực.

Frontend:

- Ứng dụng mobile được phát triển bằng React Native, tận dụng khả năng phát triển đa nền tảng với chi phí thấp và thời gian triển khai nhanh.
- Giao diện được thiết kế theo hướng component-based, dễ mở rộng và tái sử dụng.

Monitoring & Debugging:

- Tích hợp **Sentry**để theo dõi hiệu suất, thu thập lỗi runtime và tạo cảnh báo sớm cho đội phát triển.

3 Tính năng & Chức năng của ứng dụng

3.1 Tính năng chính

Ứng dụng **Mini Event** được thiết kế với mục tiêu phục vụ cả hai nhóm đối tượng chính: người tổ chức sự kiện (organizer) và người tham dự (attendee). Các chức năng cốt lõi trong phiên bản MVP được phân loại như sau:

3.1.1 Tạo và quản lý sự kiện

Người dùng sau khi đăng nhập có thể dễ dàng khởi tạo sự kiện mới với các thông tin cần thiết như:

- Tên sự kiện, mô tả chi tiết
- Thời gian, địa điểm tổ chức
- Chế độ riêng tư (Private/Public)

- Giới hạn số lượng người tham gia

Các sự kiện đã tạo được lưu trữ tập trung và có thể chỉnh sửa, cập nhật hoặc xoá khi cần thiết.

3.1.2 Quản lý danh sách khách mời

Đối với các sự kiện riêng tư, organizer có thể:

- Thêm khách mời theo email hoặc số điện thoại
- Tự động gửi thư mời qua email bằng hệ thống SMTP tích hợp

Danh sách khách mời được đồng bộ với hệ thống backend và hiển thị theo thời gian thực trên dashboard quản trị.

3.1.3 Check-in thông minh

Ứng dụng hỗ trợ đa phương thức check-in:

- **Mã QR động:** người tham dự dùng app để tạo mã QR cá nhân cho từng sự kiện.
- **GPS (tùy chọn):** sử dụng kết hợp với vị trí để xác minh người check-in thực sự đang có mặt tại khu vực tổ chức.

Thông tin check-in sẽ được ghi nhận ngay lập tức và cập nhật lên dashboard để người tổ chức theo dõi theo thời gian thực.

3.1.4 Kết nối với sự kiện công khai (Public Events)

Bên cạnh các sự kiện riêng tư, người dùng có thể tìm kiếm và duyệt các sự kiện công khai. Những sự kiện này được hiển thị trên newsfeed, có thể đăng ký tham gia ngay trong ứng dụng và được lưu vào lịch cá nhân.

3.1.5 Định hướng cải tiến

Nhằm mở rộng phạm vi ứng dụng và cải thiện trải nghiệm người dùng, nhóm đề xuất các cải tiến sau:

- **Hybrid check-in:** kết hợp xác thực mã QR với định danh trên trình duyệt web để phù hợp với người dùng không cài app.
- **Tuỳ chọn hình thức check-in cho từng sự kiện:** người tổ chức có thể cấu hình giữa các chế độ QR-only, GPS-only hoặc kết hợp (Hybrid).
- **Chế độ offline fallback:** ghi tạm dữ liệu check-in trên thiết bị local nếu mất kết nối internet, đồng bộ lại khi online.
- **Tích hợp thanh toán:** bổ sung phương thức thanh toán online cho sự kiện trả phí thông qua cổng ví điện tử hoặc thẻ ngân hàng.

4 Chi tiết triển khai

4.1 Công nghệ & công cụ sử dụng

Để đáp ứng được yêu cầu trong giai đoạn MVP mà còn đặt nền tảng tốt để mở rộng hệ thống lớn hơn về sau, chúng em sử dụng các công nghệ:

Thành phần	Công nghệ sử dụng	Vai trò
Frontend (Mobile)	React Native	Phát triển ứng dụng Android, hướng component-based, tối ưu hiệu năng và dễ bảo trì
Backend (API server)	Spring Boot	Xử lý nghiệp vụ, xây dựng RESTful API, tích hợp các dịch vụ phụ trợ
Database	Firebase Firestore + Realtime Database	Lưu trữ dữ liệu thời gian thực, giảm độ trễ trong các thao tác check-in và cập nhật dashboard
Authentication	Firebase Authentication	Đăng nhập, xác thực người dùng bằng email/password hoặc OAuth
Theo dõi lỗi	Sentry	Ghi nhận lỗi runtime và hiệu suất (crash-free rate, active session, trace stack)
Thiết kế UI/UX	Figma	Thiết kế và prototyping toàn bộ flow người dùng trước khi phát triển
Quản lý mã nguồn	Git + GitHub	Lưu trữ và phân phối mã nguồn, hỗ trợ pull request và code review
CI/CD & triển khai	GitHub Actions + Azure App Service	Tự động hoá quy trình build, deploy sau mỗi lần push lên nhánh chính

4.2 Lý do lựa chọn công nghệ

React Native (Frontend

- Cho phép phát triển ứng dụng mobile nhanh chóng với chi phí thấp nhờ khả năng tái sử dụng code giữa Android và iOS.
- Hệ sinh thái lớn, hỗ trợ thư viện mạnh mẽ (navigation, camera, barcode scanning, maps...).
- Hướng component-based giúp chia tách logic rõ ràng, dễ bảo trì và mở rộng.

Spring boot (Backend)

- Cung cấp nền tảng mạnh mẽ để xây dựng RESTful API theo kiến trúc phân lớp (controller – service – repository).
- Tích hợp tốt với các dịch vụ như Firebase, RabbitMQ, hệ thống gửi email, giúp xử lý nghiệp vụ đồng bộ và bất đồng bộ linh hoạt.
- Dễ dàng cấu hình bảo mật (JWT, CORS), hỗ trợ Swagger/OpenAPI để tự động hóa tài liệu.

Firebase (Data layer + Auth)

- Firestore giúp lưu trữ dữ liệu theo mô hình NoSQL phù hợp với các luồng realtime như check-in.
- Realtime Database cho phép lắng nghe và cập nhật dữ liệu thời gian thực, tối ưu trải nghiệm dashboard và tracking.
- Firebase Authentication đơn giản hoá đăng nhập, hỗ trợ cả custom claims nếu muốn phân quyền nâng cao.

CI/CD & Triển khai

- GitHub Actions giúp tích hợp liên tục và triển khai tự động chỉ với một file cấu hình YAML duy nhất.
- Azure App Service cung cấp môi trường host ổn định, dễ tích hợp, hỗ trợ auto-scaling, và theo dõi logs qua Application Insights.

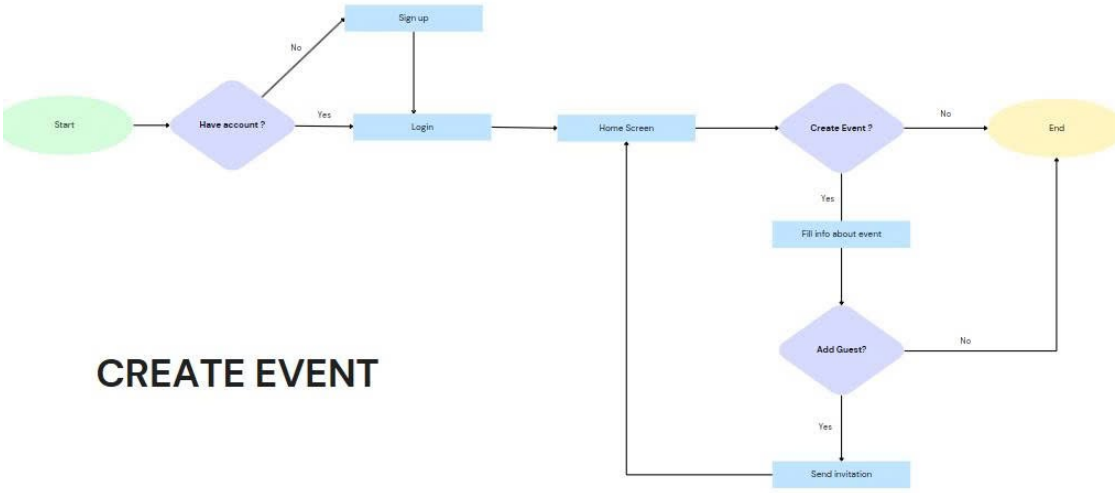
4.3 Đánh giá tổng quan

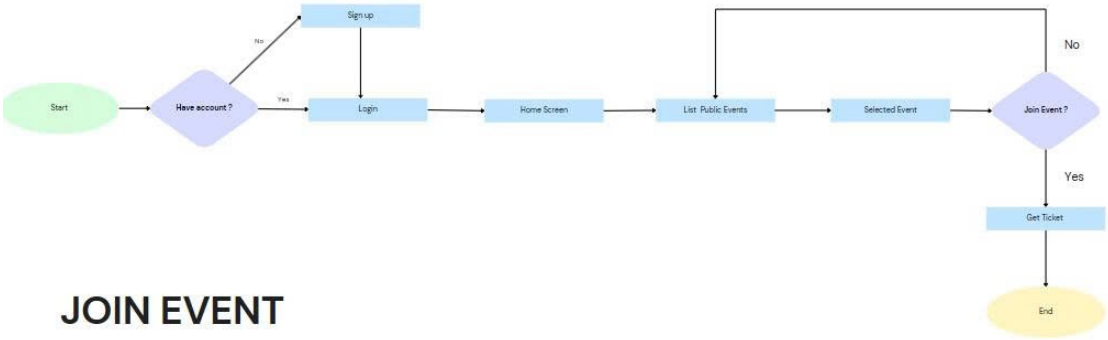
Việc kết hợp giữa React Native + Spring Boot + Firebase mang lại: Hiệu suất phát triển nhanh (Fast Iteration)

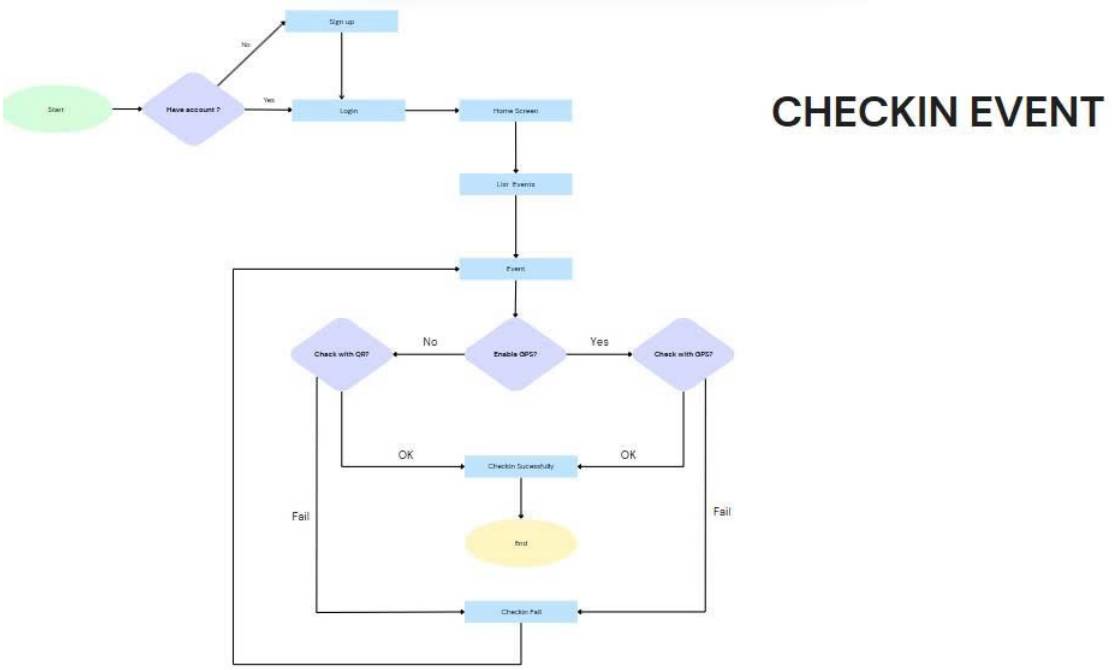
- Hiệu suất phát triển nhanh (Fast Iteration)
- Trải nghiệm người dùng tốt (Responsive + Realtime)
- Hạ tầng ổn định và dễ mở rộng (Scalable Infrastructure)

5 Trải nghiệm người dùng (User Experience)

5.1 Luồng (user flow) chi tiết cho MVP







6 Kiến trúc hệ thống

6.1 Kiến trúc tổng quan: Client–Server

Ứng dụng Mini Event được thiết kế theo mô hình Client–Server, là kiến trúc phổ biến cho các ứng dụng hiện đại. Trong mô hình này, vai trò giữa frontend (client) và backend (server) được phân tách rõ ràng:

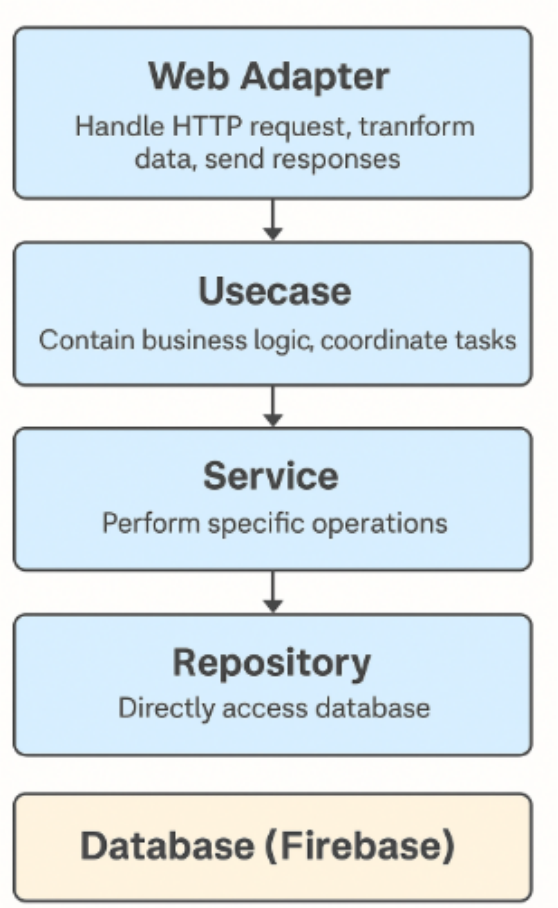
Client (ứng dụng mobile):

- Được phát triển bằng React Native.
- Chịu trách nhiệm giao diện người dùng, thu thập input và hiển thị kết quả từ server.
- Gửi yêu cầu HTTP đến backend thông qua các API RESTful và xử lý phản hồi.

Server (backend Spring Boot):

- Xử lý logic nghiệp vụ, xác thực, phân quyền và tương tác với cơ sở dữ liệu (Firebase).
- Đóng vai trò trung gian giữa frontend và hệ thống lưu trữ dữ liệu.
- Đồng thời cung cấp các API bảo mật, hiệu quả và mở rộng được.

Mô hình này giúp hệ thống dễ bảo trì, mở rộng theo chiều ngang, đồng thời tách biệt rõ ràng giữa luồng xử lý giao diện và luồng nghiệp vụ.



6.2 Kiến trúc backend

Kiến trúc backend của Mini Event áp dụng mô hình phân lớp Layered Architecture, kết hợp cùng nguyên lý Clean Architecture, nhằm đảm bảo tính tách biệt giữa các thành phần, giảm thiểu sự phụ thuộc lẫn nhau và tăng khả năng kiểm thử cũng như tái sử dụng.

- **Web Adapter (controller, dto, exception, response):** Tiếp nhận HTTP request, chuyển dữ liệu tới Usecase và phản hồi lại client.
- **Usecase:** Chứa các logic nghiệp vụ chính, đóng vai trò trung gian giữa adapter và service.
- **Service:** Thực hiện các tác vụ cụ thể như xác thực, gửi mail, tạo mã QR.
- **Repository:** Giao tiếp trực tiếp với Firebase để thao tác dữ liệu.

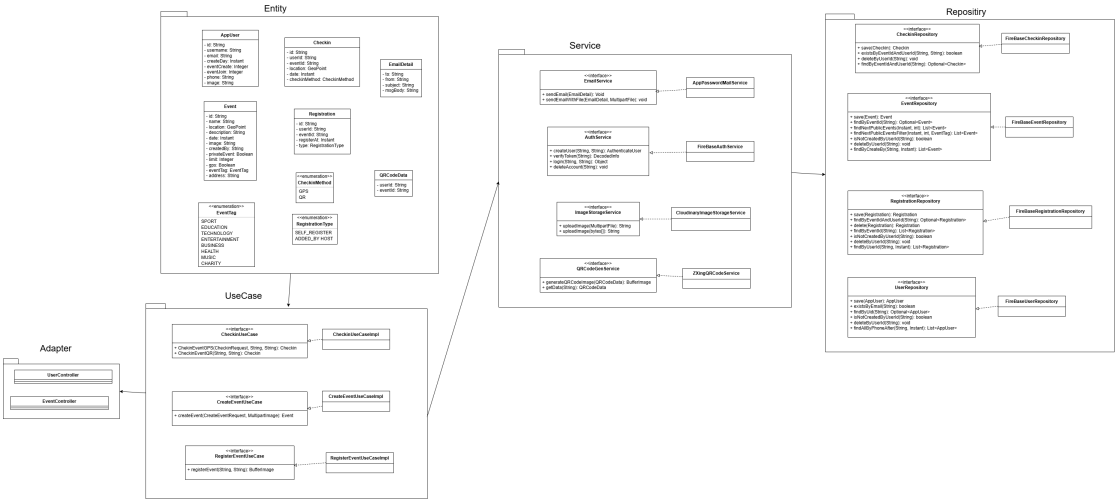
6.3 Kiến trúc frontend:

Ứng dụng mobile được xây dựng theo mô hình **component-based architecture**, với các nguyên tắc sau:

- Component độc lập: Mỗi màn hình được tổ chức dưới dạng component, giúp tái sử dụng và bảo trì dễ dàng.

- Quản lý trạng thái: Sử dụng Context API và các hook để quản lý trạng thái giữa các thành phần.
- API-driven UI: Giao tiếp với backend qua RESTful API để nhận và hiển thị dữ liệu một cách linh hoạt.

7 Class diagram



Xem chi tiết diagram tại đây

8 Thiết kế Web Service

Kiến trúc dịch vụ web (Web Service Design) trong hệ thống Mini Event được xây dựng theo mô hình RESTful API, cho phép client giao tiếp với server thông qua các phương thức HTTP tiêu chuẩn. Việc thiết kế tuân thủ các nguyên tắc REST không chỉ giúp đảm bảo tính dễ hiểu, nhất quán và khả năng mở rộng, mà còn thuận tiện cho việc tích hợp.

8.1 Nguyên tắc thiết kế

Tuân thủ chuẩn REST: tài nguyên (resource) rõ ràng, sử dụng HTTP method (GET, POST, PUT, DELETE) phù hợp.

Versioning: đặt đường dẫn API theo phiên bản, ví dụ /api/v1/events để dễ bảo trì và nâng cấp.

Stateless: mỗi request chứa đủ thông tin, không lưu trạng thái phiên trên server.

Bảo mật: sử dụng JWT Bearer Token trong header Authorization, kết hợp HTTPS.

8.2 Thiết kế tài nguyên và endpoints

Các API được tổ chức xung quanh các tài nguyên chính của hệ thống như sau:
Tài nguyên sự kiện (Event):

- GET /api/v1/events: Lấy danh sách sự kiện công khai hoặc theo bộ lọc.
- GET /api/v1/events/eventId: Lấy chi tiết sự kiện cụ thể.
- POST /api/v1/events: Tạo mới một sự kiện.
- PUT /api/v1/events/eventId: Cập nhật thông tin một sự kiện.
- DELETE /api/v1/events/eventId: Xóa một sự kiện.

Tài nguyên đăng ký (Registration) và điểm danh (Check-in):

- POST /api/v1/events/eventId/registrations: Đăng ký tham gia một sự kiện.
- GET /api/v1/events/eventId/registrations: Lấy danh sách khách mời và trạng thái điểm danh.
- POST /api/v1/events/eventId/checkin: Gửi yêu cầu check-in (bằng mã QR hoặc GPS).

Tài nguyên người dùng (User):

- POST /api/v1/users/register: Đăng ký người dùng mới.
- POST /api/v1/users/login: Đăng nhập và nhận access token.
- GET /api/v1/users/userId: Lấy thông tin người dùng.

Tất cả các endpoint đều yêu cầu xác thực JWT trừ các hành vi công khai như truy cập danh sách sự kiện công khai. Những API nhạy cảm như tạo sự kiện, check-in hoặc xem dashboard đều yêu cầu phân quyền chính xác tương ứng với vai trò người dùng.

8.3 Định dạng dữ liệu và phản hồi

Dữ liệu trao đổi giữa client và server sử dụng định dạng JSON. Mỗi phản hồi từ server được chuẩn hóa theo cấu trúc chung để frontend dễ dàng xử lý:

```
{
"status": 201,
"data":  ... ,
"message": "Event created successfully"
}
```

Trong trường hợp lỗi, phản hồi bao gồm mã lỗi rõ ràng, cùng thông điệp thân thiện với người dùng:

```
{
"status": 404,
"message": "Event with id 123 not found"
}
```

8.4 Tài liệu hóa và kiểm thử

Để đảm bảo khả năng tích hợp tốt giữa frontend và backend, chúng em sử dụng Swagger/OpenAPI để tự động sinh tài liệu API từ các annotation trong mã nguồn backend. Tài liệu được xuất ra tại endpoint /swagger-ui/index.html và được cập nhật liên tục khi có thay đổi logic.

Song song đó, nhóm cũng sử dụng Postman để xây dựng collection kiểm thử thủ công, đặc biệt hữu ích trong giai đoạn đầu phát triển MVP. Mỗi endpoint chính đều được viết unit test và integration test với dữ liệu giả lập để đảm bảo độ tin cậy khi triển khai thực tế.

9 Chiến lược triển khai

9.1 Tiêu chí lựa chọn nền tảng

Việc lựa chọn nền tảng triển khai backend cho ứng dụng Mini Event được cân nhắc dựa trên các yếu tố cốt lõi gồm khả năng tích hợp liên tục (CI/CD), chi phí vận hành thấp trong giai đoạn phát triển, tính ổn định khi vận hành, và khả năng mở rộng cho các giai đoạn tương lai.

Sau khi đánh giá các lựa chọn phổ biến như Heroku, AWS Elastic Beanstalk, DigitalOcean và Vercel, nhóm phát triển quyết định lựa chọn **Azure App Service**. Nền tảng này cung cấp khả năng tích hợp CI/CD chặt chẽ với GitHub, hỗ trợ triển khai ứng dụng Java theo dạng .jar đơn giản, đồng thời đảm bảo các tiêu chuẩn bảo mật và độ tin cậy của hạ tầng đám mây Microsoft.

9.2 Quy trình triển khai với GitHub Actions

Hệ thống sử dụng GitHub Actions làm công cụ chính cho quá trình tích hợp và triển khai liên tục. Mỗi khi có thay đổi được đẩy lên nhánh main, GitHub Actions sẽ tự động kích hoạt một workflow gồm hai giai đoạn: build và deploy.

Trong giai đoạn build, mã nguồn backend được tải về, biên dịch bằng Maven, và tạo ra file .jar. Artifact này sau đó được lưu trữ tạm thời trong pipeline để phục vụ cho giai đoạn triển khai.

Ở giai đoạn deploy, pipeline sử dụng Azure CLI để xác thực với tài khoản Azure thông qua thông tin client ID, tenant ID và subscription ID đã được cấu hình trong GitHub Secrets. Sau khi xác thực thành công, workflow tiếp tục upload file .jar và triển khai lên Web App định danh là MiniEvent với môi trường Production.

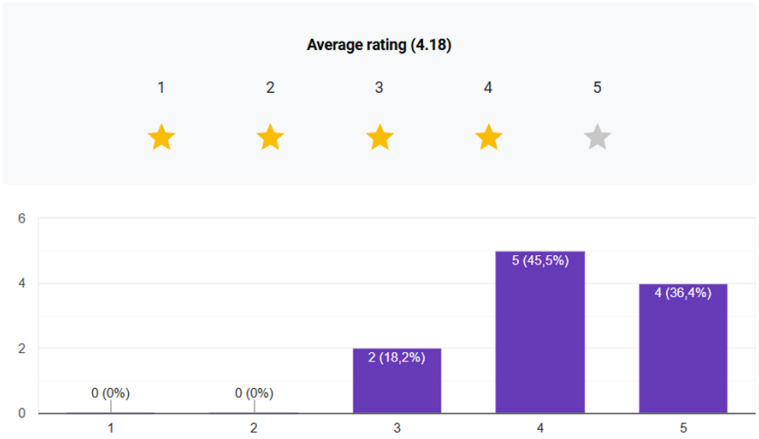
Việc sử dụng CI/CD mang lại nhiều lợi ích đáng kể, bao gồm giảm thiểu lỗi triển khai thủ công, đảm bảo tính đồng nhất giữa các môi trường, và cho phép phát hiện lỗi sớm ngay trong quá trình build. Đồng thời, với khả năng tự động hoá, nhóm phát triển có thể duy trì tốc độ triển khai nhanh và liên tục.

10 Phân tích khảo sát mức độ hài lòng của người dùng

Bạn thấy ứng dụng dễ sử dụng như thế nào?

Sao chép biểu đồ

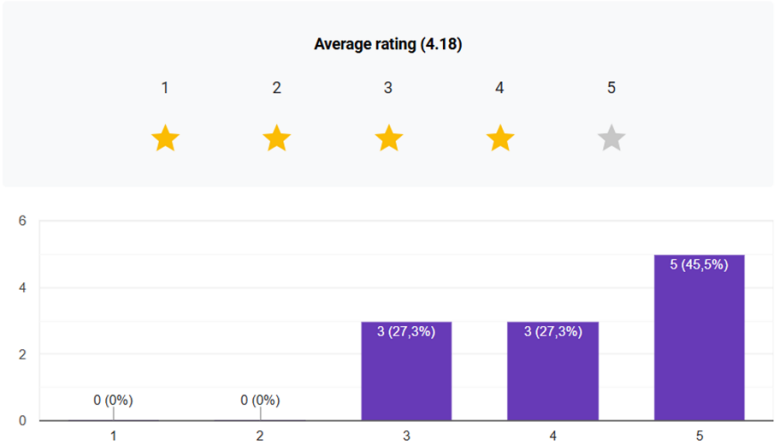
11 câu trả lời



Bạn đánh giá như thế nào về màu sắc và bố cục của ứng dụng?

Sao chép biểu đồ

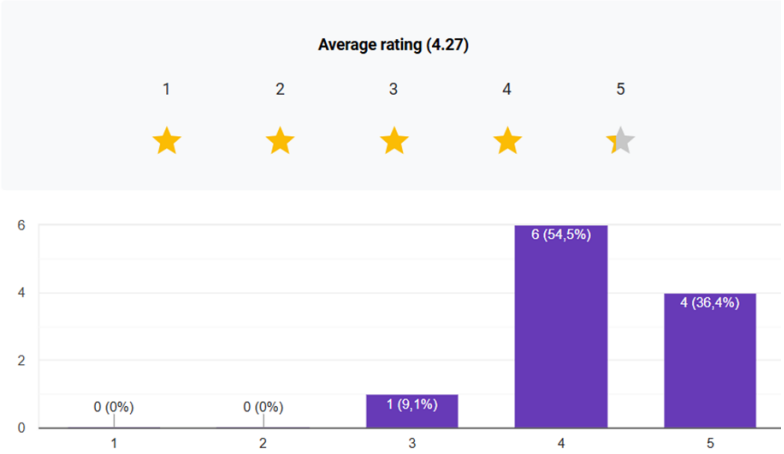
11 câu trả lời



Các biểu tượng (icon) và hình ảnh (logo) của ứng dụng rõ ràng, dễ hiểu không?

Sao chép biểu đồ

11 câu trả lời

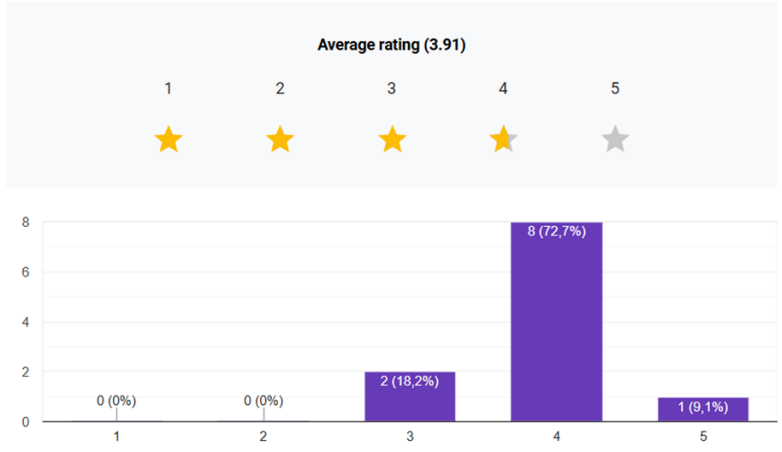


Tính năng tạo & quản lý sự kiện

Việc tạo mới một sự kiện dễ dàng và nhanh chóng như thế nào?

Sao chép biểu đồ

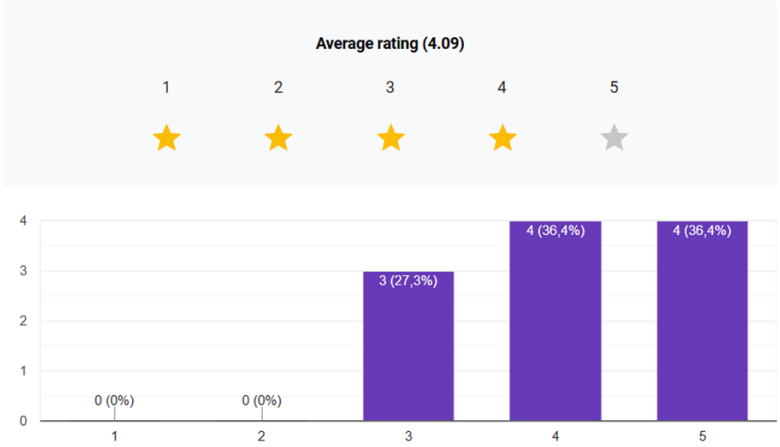
11 câu trả lời



Việc quản lý danh sách khách mời (thêm, xóa, sửa) thuận tiện ra sao?

Sao chép biểu đồ

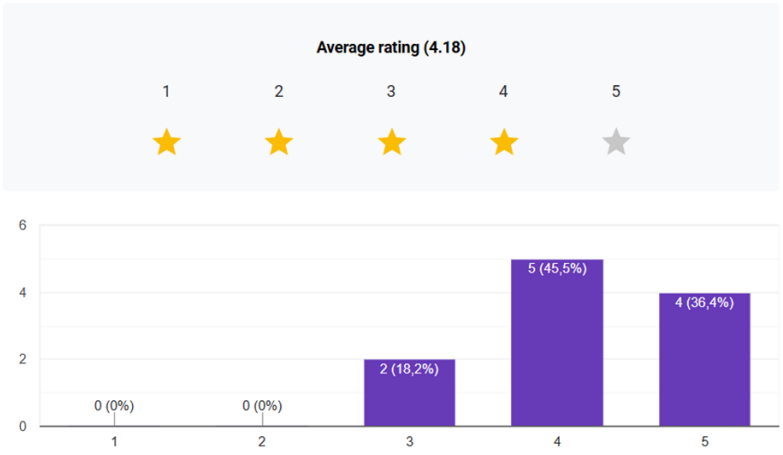
11 câu trả lời

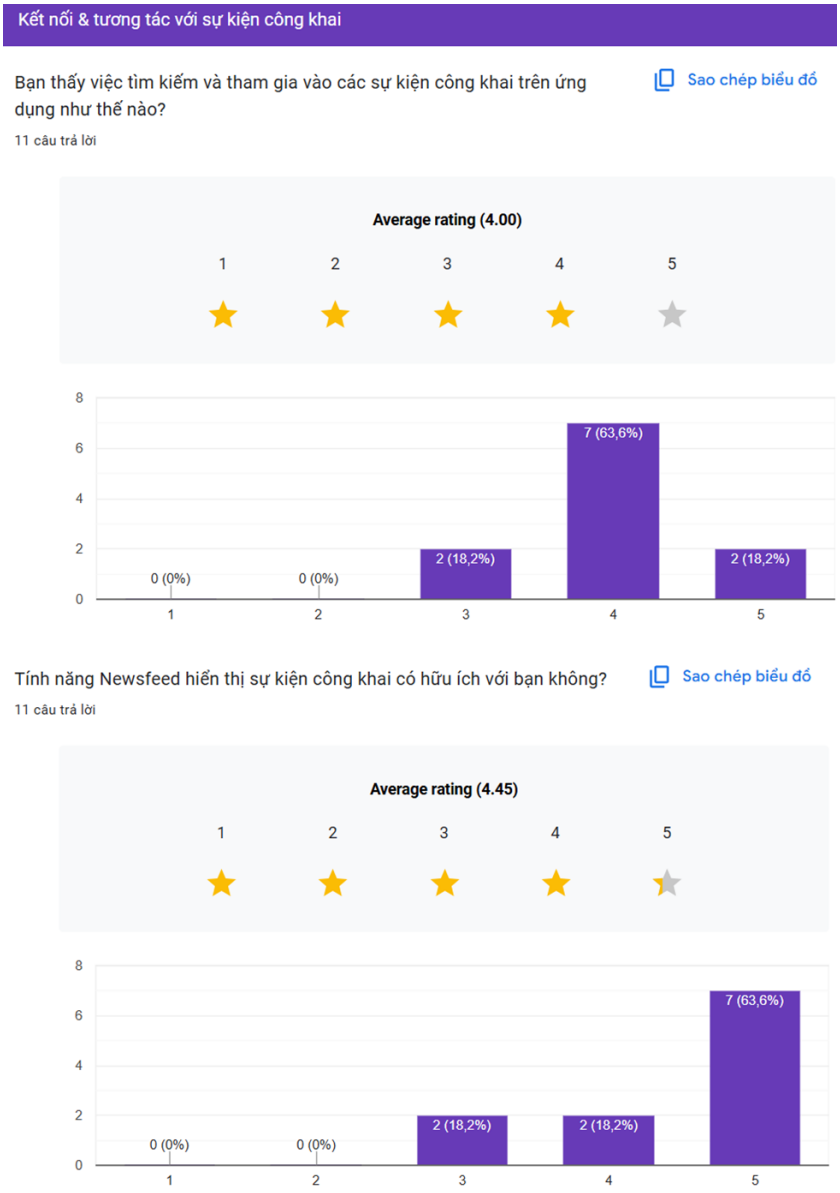


Bạn đánh giá tính năng Check-in bằng mã QR như thế nào?

Sao chép biểu đồ

11 câu trả lời





Dựa trên dữ liệu khảo sát từ tài liệu, mức độ hài lòng của người dùng đối với các tính năng của ứng dụng được đánh giá thông qua năm câu hỏi, mỗi câu hỏi có 11 câu trả lời với thang điểm từ 1 đến 6 (trừ câu hỏi về Newsfeed sử dụng thang điểm từ 1 đến 8). Dưới đây là phân tích chi tiết:

1. Trải nghiệm sử dụng ứng dụng (Đánh giá trung bình: 4.18/6): Người dùng đánh giá trải nghiệm tổng thể của ứng dụng ở mức khá, với phần lớn điểm số tập trung ở mức 4 và 5. Tuy nhiên, sự xuất hiện của các điểm số thấp (2 và 3) cho thấy một số người dùng có thể gặp khó khăn hoặc không hài lòng với giao diện hoặc hiệu suất của ứng dụng.

2. Tính rõ ràng của biểu tượng và hình ảnh (Đánh giá trung bình: 4.27/6): Các biểu

tượng và hình ảnh của ứng dụng được đánh giá cao hơn một chút so với trải nghiệm tổng thể, với điểm trung bình 4.27. Điểm số 4 và 5 chiếm ưu thế, cho thấy phần lớn người dùng nhận thấy thiết kế trực quan và dễ hiểu, mặc dù vẫn có một số đánh giá thấp (1 và 2), có thể liên quan đến vấn đề về độ rõ nét hoặc sự nhất quán trong thiết kế.

3. Tính năng tạo và quản lý sự kiện (Đánh giá trung bình: 3.91/6): Đây là tính năng nhận được mức đánh giá thấp nhất trong khảo sát, với trung bình 3.91. Sự phân bố điểm số cho thấy một số người dùng gặp khó khăn trong việc tạo hoặc quản lý sự kiện, với nhiều điểm số từ 1 đến 3. Điều này chỉ ra rằng tính năng này có thể cần cải thiện để đáp ứng tốt hơn nhu cầu người dùng.

4. Tính năng Check-in bằng mã QR (Đánh giá trung bình: 4.18/6): Tính năng Check-in bằng mã QR nhận được đánh giá tương đương với trải nghiệm tổng thể (4.18). Điểm số chủ yếu nằm ở mức 4 và 5, cho thấy người dùng đánh giá cao tính tiện lợi của tính năng này, nhưng vẫn có một số điểm thấp (2 và 3), có thể liên quan đến vấn đề kỹ thuật hoặc trải nghiệm không đồng đều.

5. Tính năng Newsfeed hiển thị sự kiện công khai (Đánh giá trung bình: 4.45/8): Tính năng Newsfeed nhận được đánh giá cao nhất, với 63.6% người dùng chấm 7/8 điểm. Điều này cho thấy tính năng hiển thị sự kiện công khai được đánh giá là hữu ích và đáp ứng tốt nhu cầu của người dùng, với ít đánh giá tiêu cực hơn so với các tính năng khác.

11 Đánh giá dự án và hướng cải thiện

11.1 Đánh giá dự án

Dự án Mini Event là một ứng dụng di động được phát triển với mục tiêu rõ ràng: hỗ trợ những người tổ chức sự kiện không chuyên quản lý toàn bộ quy trình tổ chức – từ gửi lời mời, theo dõi khách tham dự đến điểm danh và thống kê – chỉ qua một nền tảng duy nhất. Báo cáo thể hiện sự đầu tư nghiêm túc cả về kỹ thuật lẫn định hướng sản phẩm.

Về mặt kỹ thuật, nhóm sử dụng React Native cho frontend, Spring Boot cho backend và tích hợp Firebase để xử lý realtime, xác thực và lưu trữ. Cách tiếp cận theo mô hình MVP và phát triển theo phương pháp Agile Scrum là lựa chọn phù hợp, giúp nhóm linh hoạt và kiểm soát tốt tiến độ.

Các tính năng cốt lõi như tạo sự kiện, quản lý khách mời, check-in thông minh (QR, GPS), newsfeed sự kiện công khai đều được mô tả rõ ràng và nhất quán. Ngoài ra, nhóm đã thể hiện tầm nhìn sản phẩm dài hạn khi đề xuất các định hướng cải tiến như hybrid check-in, chế độ offline fallback, và tích hợp thanh toán.

11.2 Hướng cải thiện

1. Mở rộng trải nghiệm người dùng:

Hiện phần mô tả về User Experience còn sơ sài (trong báo cáo chỉ đề cập đến user flow). Nên bổ sung mô tả các giao diện chính, ví dụ bằng hình ảnh mockup hoặc phản hồi người dùng thử nghiệm để đánh giá tính thân thiện và hiệu quả UX.

2. Kiểm thử tự động & Bảo mật:

Báo cáo có đề cập đến kiểm thử bằng Postman và unit test, nhưng chưa rõ mức độ bao phủ. Nhóm có thể bổ sung CI kiểm thử tự động với coverage report, và đánh giá bảo mật API kỹ hơn (ví dụ kiểm thử xâm nhập hoặc xử lý rate-limit, SQL injection nếu có phần mở rộng).

3. Khả năng mở rộng và hiệu suất:

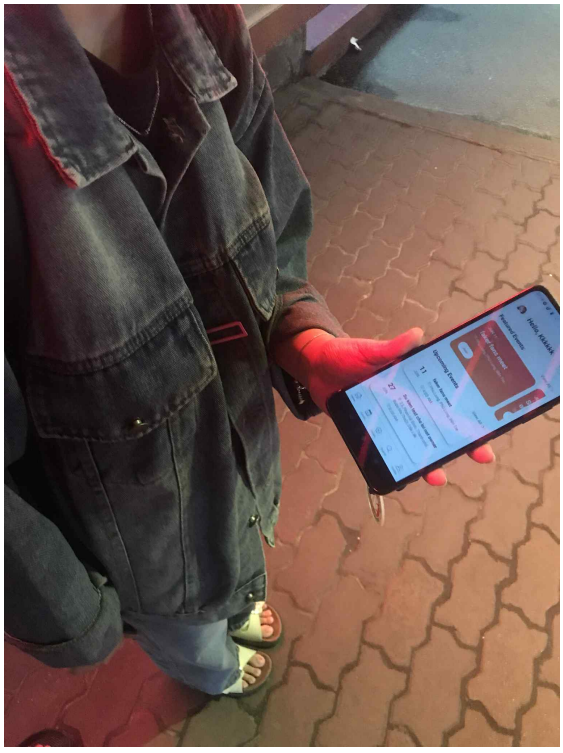
Firebase là lựa chọn tốt cho MVP, nhưng cần phân tích thêm về giới hạn khi mở rộng quy mô (ví dụ: số lượng kết nối realtime, chi phí Firebase khi có hàng ngàn người dùng). Nên có phần đánh giá hiệu suất backend dưới tải.

4. Truy cập đa nền tảng:

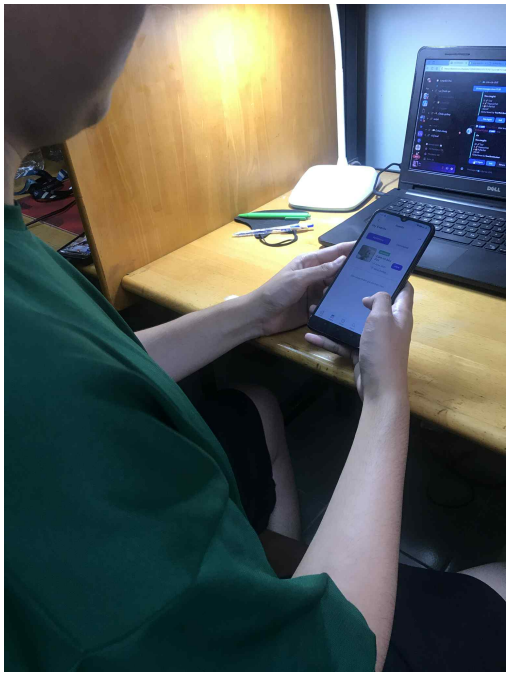
Một cải tiến đáng cân nhắc là xây dựng bản web nhẹ để người không dùng app vẫn có thể nhận lời mời, check-in hoặc xem thông tin sự kiện – điều này rất phù hợp với các sự kiện cộng đồng.

5. Chức năng analytics nâng cao:
Hiện dashboard chỉ hiển thị theo thời gian thực. Có thể bổ sung phân tích sau sự kiện như: thời gian trung bình check-in, tỉ lệ tham gia theo nhóm tuổi/khu vực nếu có, v.v.

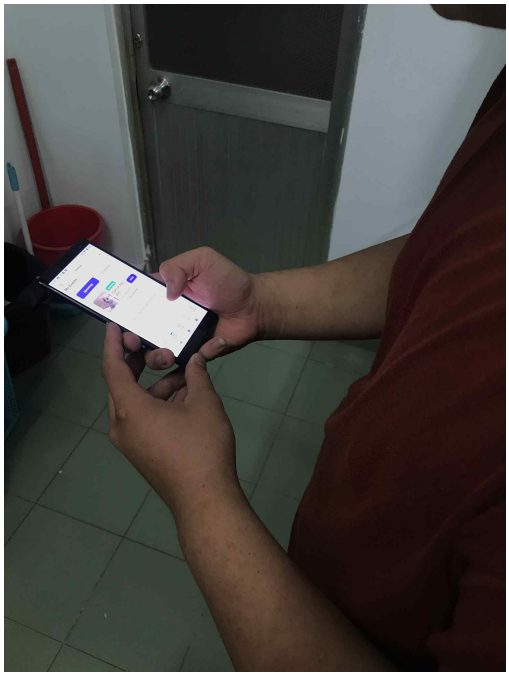
12 Ảnh minh chứng người sử dụng app



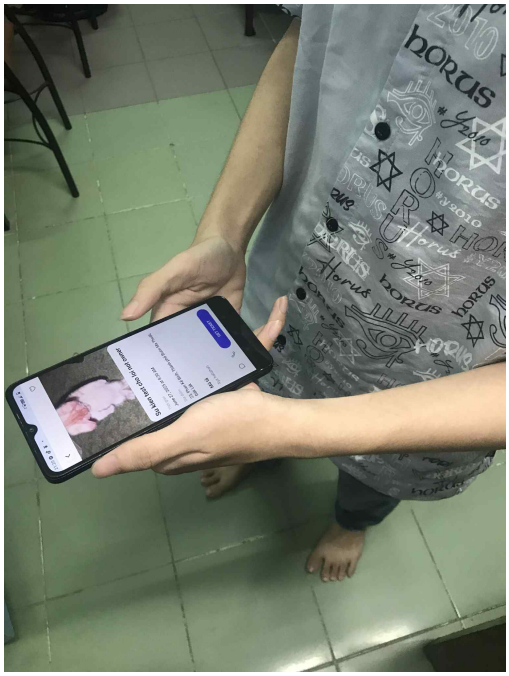
Hình 1: Người dùng 1



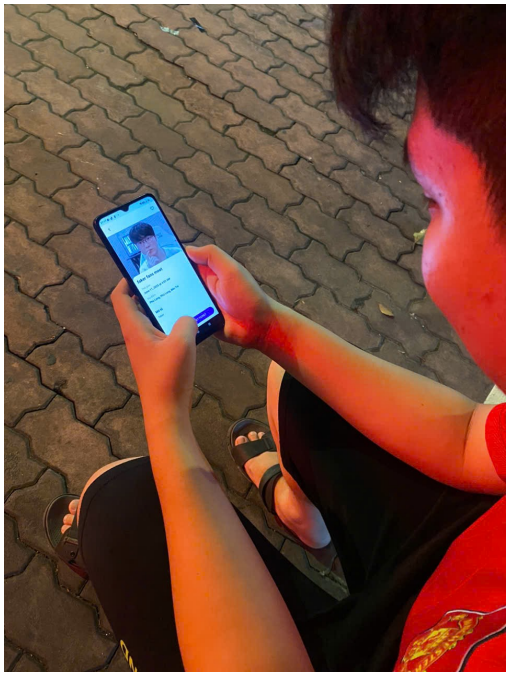
Hình 2: Người dùng 2



Hình 3: Người dùng 3



Hình 4: Người dùng 4



Hình 5: Người dùng 5

13 Link backend - frontend, link tải app - QR tải app và link video demo

Link Backend: ⟶ **Truy cập tại đây**

Link Frontend: ⟶ **truy cập tại đây**

QR tải file apk của app:

Link drive chứa file apk của app: ⟶ **truy cập tại đây**

Link drive video demo: ⟶ **truy cập tại đây**

14 Poster của app

Link drive chứa video demo: ⟶ **truy cập tại đây**