Fund Management App Proposal

Hội đồng duyệt dự án: Huỳnh Hoàng Kha
Nguyễn Tấn Đạt
Người thực hiện: Nguyễn Phúc Vinh
Cù Đỗ Thanh Nhân
Trần Hà Tuấn Kiệt

TP. Hồ Chí Minh, ngày 15 tháng 01 năm 2022

Contents

1 Tổng quan dự án **2**
1.1 Tên dự án . . . 2
1.2 Mô tả . . . 2
1.3 Các đối tượng liên quan của dự án . . . 2
1.4 Ý nghĩa . . . 2

2 Thiết kế dự án **4**
2.1 Vấn đề đặt ra . . . 4
2.2 Giải pháp đề xuất . . . 4
2.3 Yêu cầu sản phẩm . . . 4
2.3.1 Yêu cầu chức năng . . . 5
2.3.2 Yêu cầu phi chức năng . . . 6
2.3.3 Yêu cầu dữ liệu . . . 6

3 Kiến trúc phần mềm **7**
3.1 Kiến trúc C4 . . . 7
3.2 Sơ đồ ngữ cảnh hệ thống (System Context) . . . 7
3.3 Sơ đồ vùng chứa (Container) . . . 8

4 Cơ sở dữ liệu **10**
4.1 Phân tích . . . 10
4.2 Bài toán phân quyền ứng dụng . . . 11
4.3 Thiết kế ý niệm . . . 12
4.3.1 Dữ liệu có thể thay đổi (nhóm 1) . . . 12
4.3.2 Dữ liệu không thay đổi sau khi lưu (Nhóm 2) . . . 14
4.4 Điều khiển truy cập của DBMS . . . 15

5 Sao lưu dữ liệu và Bảo mật **16**
5.1 Kế hoạch sao lưu dữ liệu . . . 16
5.2 Bảo mật phía Cơ sở dữ liệu (CSDL) . . . 16
5.3 Bảo mật phía Ứng dụng (application server) . . . 16
5.4 Giải thích các thuật ngữ được sử dụng trong phần bảo mật . . . 18

6 Deployment **18**
6.1 Deploy thủ công . . . 18
6.2 Sử dụng dịch vụ hosting từ bên thứ ba . . . 19

7 Kế hoạch **20**
7.1 Các cột mốc quan trọng . . . 20
7.2 Chi phí . . . 20

1 Tổng quan dự án

1.1 Tên dự án

TickFund - TickLab Fund Management Application
(Phần mềm quản lý quỹ cho TickLab)

1.2 Mô tả

TickLab là một phòng thí nghiệm khoa học kĩ thuật và công nghệ được thành lập bởi một nhóm sinh viên đến từ Đại học Bách khoa - Đại học Quốc gia thành phố Hồ Chí Minh từ năm 2017. TickLab hoạt động với mục tiêu tạo lập và duy trì một môi trường học tập, thực hành, thí nghiệm năng động và chuyên nghiệp trong lĩnh vực khoa học kỹ thuật, định hướng phát triển con người và các giải pháp hiệu quả cho đời sống, góp phần xây dựng và đổi mới đất nước. Hiện tại, TickLab đang quản lý các nguồn thu chi chưa được hiệu quả. Cụ thể, một Thủ Quỹ quản lí các khoản thu chi của TickLab một cách thủ công và lưu trữ thu chi trên Google Sheet. Việc tổ chức quản lý thu chi mang tính chất tự phát như trên không tránh khỏi sai sót đến từ thao tác thủ công, thiếu sự minh bạch và khó khăn trong việc thống kê quỹ. Ban Quản Trị khó tiếp cận được thông tin hiện tại của quỹ và đưa ra những chính sách quỹ hợp lý.
Giải pháp do tổ Phát triển dự án của TickLab đề xuất sẽ phát triển một phần mềm Web tích hợp hệ cơ sở dữ liệu mang tên TickFund - TickLab Fund Management Application - phục vụ việc lưu trữ và cung cấp một giao diện chung để thao tác. Mục tiêu của dự án là xây dựng được một nền tảng chung để việc quản lý quỹ trở nên dễ dàng, minh bạch và hiệu quả hơn so với cách quản lý trước đây.

1.3 Các đối tượng liên quan của dự án

Các đối tượng liên quan của dự án TickLab Fund Management App bao gồm:

1. Phòng thí nghiệm TickLab: là bên sở hữu kết quả của dự án.
2. Ban Quản Trị TickLab: là những người chịu trách nhiệm quản lý và giám sát hệ thống, đảm báo hệ thống được vận hành đúng quy trình.
3. Thủ quỹ: là một hoặc nhiều thành viên của TickLab đảm nhiệm vai trò ghi lại toàn bộ giao dịch và dự trù của Lab.
4. Toàn bộ thành viên TickLab: là những người mong muốn được theo dõi tình hình quỹ Lab và các khoản thu chi của Lab.
5. Bộ phận bảo trì: là một nhóm một hoặc nhiều người trong bộ phận IT của Lab chịu trách nhiệm bảo trì và phát triển hệ thống.

1.4 Ý nghĩa

Đối với TickLab, dự án giải quyết được vấn đề quản lý tài chính trong tương lai.

Đối với cá nhân các thành viên trong nhóm, kết quả của dự án có ý nghĩa như sau:

- Là một case study rõ ràng, bao quát được những khía cạnh cần thiết khi phát triển phần mềm. Toàn bộ dự án sẽ được document chi tiết để làm tài liệu tham khảo cho các khoá sau.

- Các thành viên, sau khi hiện thực dự án thành công, có được cái nhìn đối với phát triển phần mềm ở mức high-level, từ đó xác định được hướng đi phù hợp trong tương lai.
- Với các thành viên là sinh viên năm 3, dự án là một điểm cộng để ghi vào CV.

2 Thiết kế dự án

2.1 Vấn đề đặt ra

Vào những năm đầu thành lập, TickLab có rất ít thành viên và cơ cấu tổ chức cũng như mô hình hoạt động còn đơn giản nên phương pháp, cách thức quản lý quỹ chung của tập thể còn mang tính tự phát, nhằm giải quyết vấn đề một cách nhanh nhất. Cụ thể, một người thành viên trong Lab sẽ được phân công làm Thủ Quỹ và có trách nhiệm ghi lại toàn bộ các khoản thu chi của TickLab trong một file Google Sheet. Các khoản thu của TickLab thời đó chủ yếu phụ thuộc vào tiền đóng góp hàng tháng của các thành viên nên việc ghi lại khá đơn giản. Trong khi đó, các khoản thu xuất phát từ nhiều nguồn khác nhau mà tên của các nguồn thì không được thống nhất, không được định dạng rõ ràng gây khó khăn trong việc thống kê. Ngoài ra, việc tạo ra các sheet mới cho từng tháng khiến file Spreadsheet trở nên cồng kềnh và khó quản lý.
Với sự phát triển của TickLab như hiện nay, số lượng thành viên ngày càng tăng lên, các khoản thu không còn giới hạn trong tiền đóng góp hàng tháng của các thành viên nữa mà có thể từ rất nhiều nguồn khác như dự án, nghiên cứu khoa học,...Không chỉ thế, sự phát triển của Lab đòi hỏi một quy trình quản lý quỹ một cách hệ thống, hiệu quả và minh bạch hơn. Tính hệ thống và hiệu quả ở đây muốn nói là sự chuẩn hóa trong việc ghi lại các khoản thu, chi cũng như lưu trữ những thông tin đó ở nơi phù hợp và chuyên dụng hơn. Và, tính minh bạch nghĩa là thông tin quỹ Lab nên được theo dõi và giám sát của Ban Quản Trị và mọi thành viên trong Lab thay vì một người Thủ Quỹ. Một vấn đề nữa mà phương pháp quản lý cũ chưa giải quyết được là chưa có công cụ thống kê cần thiết để hỗ trợ Ban Quản Trị trong việc điều chỉnh kế hoạch quỹ và chính sách quỹ trong tương lai.

2.2 Giải pháp đề xuất

Sau khi tìm hiểu, nhóm đưa ra một số đặc điểm hệ thống như sau:

- Với số lượng thành viên TickLab hiện tại là 26, lượng user của hệ thống được dự đoán không nhiều hơn 100 - 200 user
- Lượng request lên hệ thống cùng lúc là không lớn
- Yêu cầu về độ chính xác của một giao dịch là tương đối
- Dữ liệu hệ thống tăng rất chậm theo thời gian

Nhóm đề xuất xây dựng web app để giúp quản lí việc thu chi của Lab. Với quy mô ứng dụng nhỏ, nên việc chọn các công nghệ không quá ảnh hưởng tới hoạt động của hệ thống. Nhóm tham khảo technical stack là MERN, tuy nhiên thay vì sử dụng Mongo, nhóm sử dụng MySQL.

2.3 Yêu cầu sản phẩm

Trước hết, ta xác định thứ tự ưu tiên giảm dần của các yêu cầu như sau:

- 1 - Phải có: Những yêu cầu ở mức tối thiểu bắt buộc phải đáp ứng. Ứng dụng không thực hiện được những yêu cầu này sẽ không có giá trị về mặt sử dụng (MVP)
- 2 - Quan trọng tương tự mức 1, nhưng nằm sau số 1
- 3 & 4 - Những yêu cầu nâng cao hoặc đi kèm, có thể có hoặc không

Những yêu cầu có độ ưu tiên ≤ 2 được xác định là MVP

2.3.1 Yêu cầu chức năng

1. Yêu cầu đến từ thành viên

#	Yêu cầu chức năng	Luồng dữ liệu	Input	Output	Phạm vi	Độ ưu tiên
1	Mở tài khoản	Thành viên → Hệ thống	Yêu cầu đăng kí tài khoản và thông tin của Thành viên	Thông tin của Thành viên được ghi nhận	Trong hệ thống	4
2	Xác nhận tài khoản	Hệ thống → Thành viên và Thành viên → Hệ thống	Email chứa đường dẫn xác nhận tài khoản trong thời hạn nhất định	Trạng thái xác minh của tài khoản	Trong hệ thống	4
3	Thay đổi thông tin cá nhân (trừ tên đăng nhập)	Thành viên → Hệ thống	Thông tin cá nhân thay đổi và xác nhận của thành viên	Thông tin sửa đổi của Thành viên được ghi nhận	Trong hệ thống	3
4	Xem lịch sử giao dịch của TickLab và riêng tài khoản	Thành viên → Hệ thống	Yêu cầu xem lịch sử giao dịch của tài khoản	Lịch sử giao dịch của TickLab và riêng tài khoản	Trong hệ thống	1
5	Chỉnh sửa ghi chú về các giao dịch liên quan đến tài khoản	Thành viên → Hệ thống	Ghi chú mới của thành viên	Ghi chú mới của thành viên đã được ghi nhận	Trong hệ thống	2
6	Tạo yêu cầu giải ngân /đóng góp quỹ	Thành viên → Hệ thống	Yêu cầu giải ngân/ đóng góp quỹ	Ghi nhận yêu cầu giải ngân/ đóng góp quỹ	Trong hệ thống	1

2. Yêu cầu đến từ Thủ Quỹ

#	Yêu cầu chức năng	Luồng dữ liệu	Input	Output	Phạm vi	Độ ưu tiên
1	Tạo các giao dịch	Thủ quỹ → Hệ thống	Yêu cầu giao dịch và minh chứng số tiền đã thanh toán	Giao dịch đã xác nhận thực hiện thành công hoặc hệ thống hủy bỏ giao dịch	Trong hệ thống	1
2	Hủy bỏ các giao dịch	Thủ quỹ → Hệ thống	Yêu cầu hủy bỏ giao dịch và minh chứng số tiền đã hoàn trả	Hủy bỏ giao dịch hợp lệ hay không hợp lệ	Trong hệ thống	1
3	Tạo/vô hiệu hoá/ kích hoạt/sửa tên các loại giao dịch	Thủ quỹ → Hệ thống	Thao tác với các loại giao dịch	Danh mục các loại giao dịch đã cập nhật	Trong hệ thống	1
4	Xem lịch sử giao dịch của TickLab và giao dịch liên quan đến tài khoản	Thủ quỹ → Hệ thống	Yêu cầu xem lịch sử giao dịch của tài khoản	Lịch sử giao dịch của TickLab và riêng tài khoản	Trong hệ thống	1
5	Chỉnh sửa ghi chú các giao dịch	Thủ quỹ → Hệ thống	Ghi chú mới của thành viên	Ghi chú mới của thành viên đã được ghi nhận	Trong hệ thống	1
6	Duyệt yêu cầu giải ngân/ đóng góp quỹ	Hệ thống → Thủ quỹ	Danh mục các yêu cầu giải ngân/đóng góp	Giao dịch hoặc dự trù	Trong hệ thống	1
7	Thêm/xoá/chỉnh sửa các khoản dự trù thu/chi	Thủ quỹ → Hệ thống	Thao tác với các khoản dự trù	Danh mục các loại dự trù đã cập nhật	Trong hệ thống	1
8	Thống kê quỹ	Thủ quỹ → Hệ thống	Yêu cầu thống kê quỹ thu chi/ số dư	Báo cáo tổng quỹ thu chi/ số dư, biểu diễn theo thông số và biểu đồ	Trong hệ thống	2
9	Thống kê dự trù	Thủ quỹ → Hệ thống	Yêu cầu thống kê dự trù	Thống kê dự trù theo loại giao dịch, thời điểm, số dư trước và sau dự trù	Trong hệ thống	2

3. Yêu cầu đến từ Ban Quản Trị

#	Yêu cầu chức năng	Luồng dữ liệu	Input	Output	Phạm vi	Độ ưu tiên
1	Tạo tài khoản	Quản lí → Hệ thống	Yêu cầu tạo tài khoản và thông tin của Thành viên	Tài khoản được ghi nhận ở hệ thống với quyền đã được chỉ định	Trong hệ thống	1
2	Đổi mật khẩu, vô hiệu hóa, thay đổi nhóm quyền đối với tài khoản	Quản lí → Hệ thống	Thao tác với tài khoản	Tài khoản đã thay đổi quyền và thuộc tính	Trong hệ thống	1
3	Tạo quyền	Quản lí → Hệ thống	Tên nhóm quyền và lựa chọn thuộc tính kèm hành vi hiện có	Quyền gắn liền với 1 thuộc tính	Trong hệ thống	3
4	Ghi lại tiến trình lịch sử	Hệ thống	Sự kiện diễn ra	Sự kiện đã được ghi nhận.	Trong hệ thống	3
6	Tạo yêu cầu giải ngân /đóng góp quỹ	Thành viên → Hệ thống	Yêu cầu giải ngân/ đóng góp quỹ	Ghi nhận yêu cầu giải ngân/đóng góp quỹ	Trong hệ thống	1

2.3.2 Yêu cầu phi chức năng

#	Yêu cầu	Độ ưu tiêu
1	Ứng dụng có độ chính xác và tin cậy cao	1
2	Các thành viên của phòng thí nghiệm TickLab dễ dàng sử dụng hệ thống sau 1 hoặc 2 lần sử dụng	1
3	Hệ thống dựa trên nền tảng web giúp người dùng thuận tiện truy cập mà không cần tải app.	1
4	Hệ thống hoạt động ổn định khi nhiều nhất 200 người dùng cùng lúc	2
5	1 câu truy vấn yêu cầu I/O time <2s	2
6	Bảo mật thông tin thành viên của TickLab	2
7	Dung lượng dữ liệu tăng/giảm có kiểm soát	3
8	Hệ thống có khả năng mở rộng	3

2.3.3 Yêu cầu dữ liệu

Yêu cầu dữ liệu được phân tích dựa trên những yêu cầu chức năng và phi chức năng. Các yêu cầu dữ liệu của ứng dụng được phân tích và trình bày như sau:

1. Thông tin về thành viên bao gồm họ tên, ngày sinh, số điện thoại, email, hình ảnh đại diện, tài khoản, mật khẩu do người dùng cung cấp, mã số thành viên và loại tài khoản do quản lí và hệ thống khởi tạo. Độ ưu tiên: 1

2. Thông tin của một dự trù thu chi bao gồm mã định danh, loại giao dịch, dạng thu/chi, lượng giao dịch, dấu thời gian, chu kì giao dịch nếu có, ghi chú. Độ ưu tiên: 1

3. Thông tin của một giao dịch bao gồm mã định danh, loại giao dịch, dạng thu/chi, lượng giao dịch, dấu thời gian, ghi chú, mã số tài khoản tạo. Độ ưu tiên: 2

4. Thông tin của một sự kiện được lưu lại lịch sử bao gồm dấu thời gian, chủ thể/khách thể, hành động. Độ ưu tiên: 3

3 Kiến trúc phần mềm

3.1 Kiến trúc C4

Để mô tả kiến trúc phần mềm trong dự án này, nhóm đã dùng kiến trúc C4 để mô hình hóa chúng. Mô hình C4 mô tả kiến trúc dự án từ ngoài vào trong, tức đi mức độ từ tổng quan bên ngoài đến cụ thể bên trong. Kiến trúc C4 bao gồm 4 mức độ là:

1. Mức độ ngữ cảnh (context Level): Ở mức độ này, chúng ta sẽ mô tả bối cảnh của hệ thống.
2. Mức độ vùng chứa (container level): Ở mức độ này, chúng ta sẽ mô tả bức tranh tổng quát thể hiện các thành phần chính trong hệ thống.
3. Mức độ thành phần (component level): Ở mức độ này, chúng ta sẽ làm rõ hơn các modules trong từng container.
4. Mức độ code (code level): Ở mức độ này, chúng ta sẽ thể hiện các lớp (classes) và các hàm (functions) trong từng module.

Trong bản proposal, nhóm sẽ phân tích kiến trúc ở hai mức độ Context và Container.

3.2 Sơ đồ ngữ cảnh hệ thống (System Context)

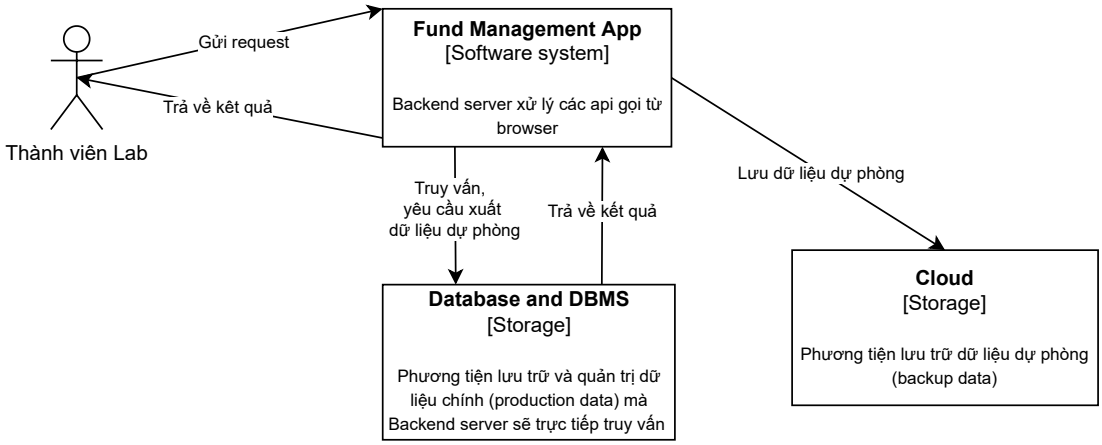

Figure 1: Sơ đồ ngữ cảnh hệ thống

3.3 Sơ đồ vùng chứa (Container)

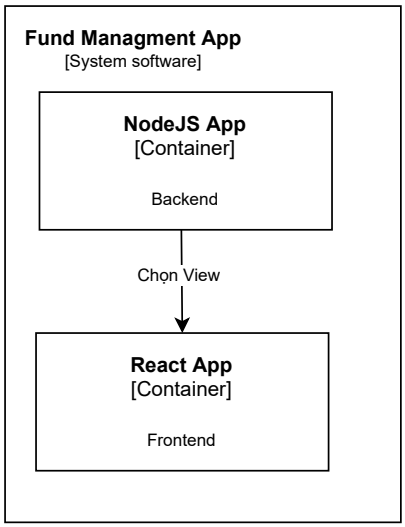

Figure 2: Sơ đồ vùng chứa của system software Fund Manament App

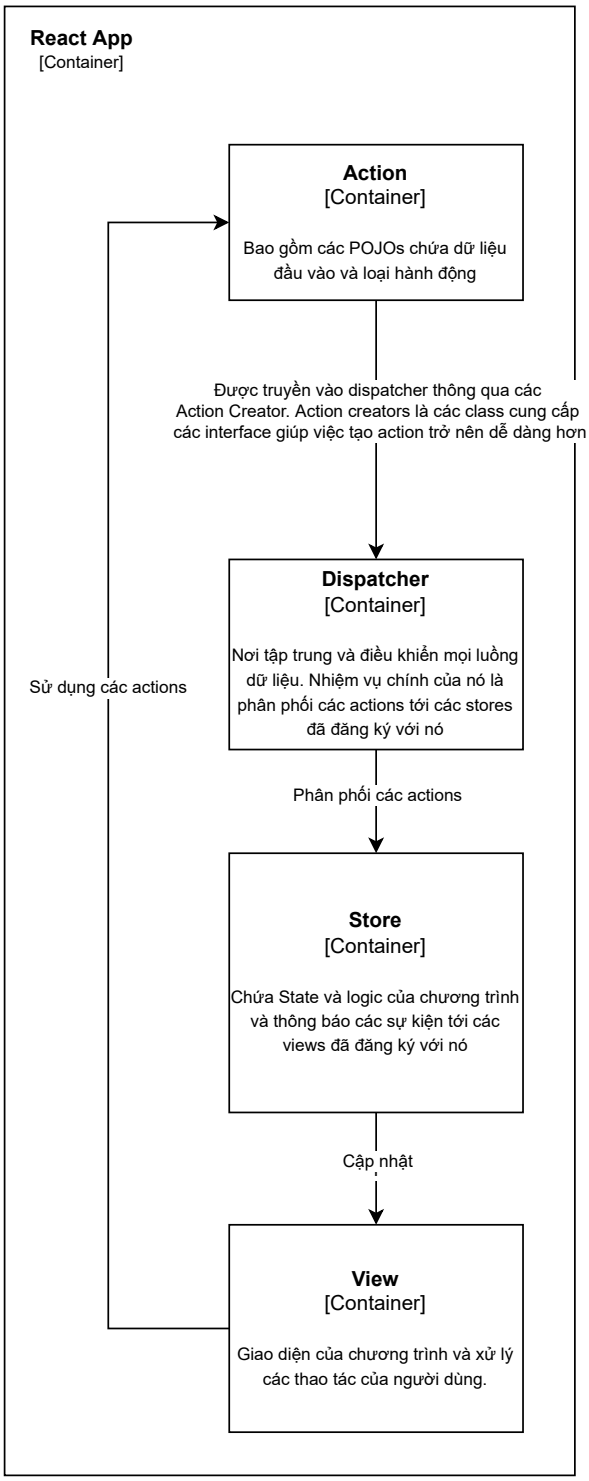

Figure 3: Các vùng chứa con của vùng chứa React App

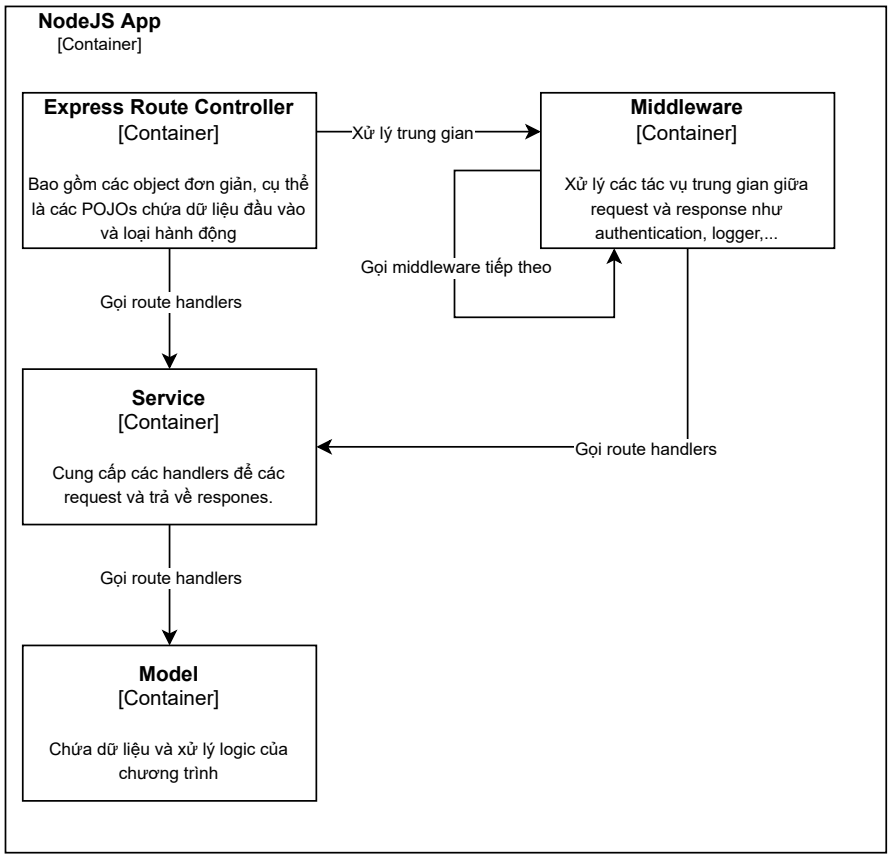

Figure 4: Các vùng chứa con của vùng chứa NodeJS App

4 Cơ sở dữ liệu

4.1 Phân tích

Dựa trên mô tả trong phần data requirements, dữ liệu của ứng dụng có thể chia làm hai nhóm:

- Có thể thay đổi (Nhóm 1): Những dữ liệu liên quan đến tài khoản user như email, sđt, ngày sinh ...
- Không thay đổi sau khi lưu trữ (Nhóm 2): Thông tin liên quan đến giao dịch, log người dùng, log hệ thống.

Dữ liệu thuộc nhóm 1 có mối quan hệ ràng buộc chặt chẽ với nhau, được định nghĩa rõ ràng và không mở rộng theo chiều ngang, vì vậy nhóm sử dụng CSDL quan hệ cho nhóm 1 với mức chuẩn hoá 3NF.

Dữ liệu thuộc nhóm 2 chỉ có thể INSERT hoặc SELECT, do đó lượng dữ liệu lớn dần theo thời gian. Ngoài ra, dữ liệu thuộc nhóm này phục vụ cho mục đích thống kê, truy xuất số lượng lớn record nên cần được đảm bảo về mặt hiệu suất. Mức chuẩn hoá của dữ liệu thuộc nhóm 2 tối đa là 1NF. Tuỳ vào dữ liệu sinh ra trong quá trình hiện thực ứng dụng thuộc dạng cấu trúc hay phi cấu trúc, ta sẽ sử dụng SQL hoặc NoSQL cho phù hợp.

4.2 Bài toán phân quyền ứng dụng

Ứng dụng Quản lý thu chi (FMA) ban đầu chia ra 3 role cơ bản: Admin, Accountant và Member. Mỗi role sẽ có các quyền truy cập hệ thống khác nhau khi sử dụng ứng dụng. Như vậy, ta có thể phân quyền theo cấp bậc đơn giản bằng cách thiết kế DB như sau:

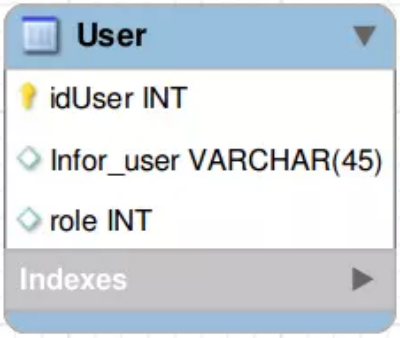

Figure 5: Naive Access Control

Kiểu phân quyền trên dễ dàng với người mới bắt đầu, nhưng lại có những khuyết điểm rất lớn như sau:

- Rất khó có thể mở rộng dự án
- Thực tế không phải lúc nào cũng có 3 role và các role được định nghĩa rõ ràng từ đầu. Các requirements thay đổi có thể phát sinh role kỳ dị.
- Gặp khó khăn trong phân quyền chi tiết

Nhóm đề xuất mô hình **Role-based Access Control** (RBAC) để giải quyết nhu cầu phân quyền ứng dụng. Trong RBAC, có ba khái niệm chính:

- Role
- Action
- Resource

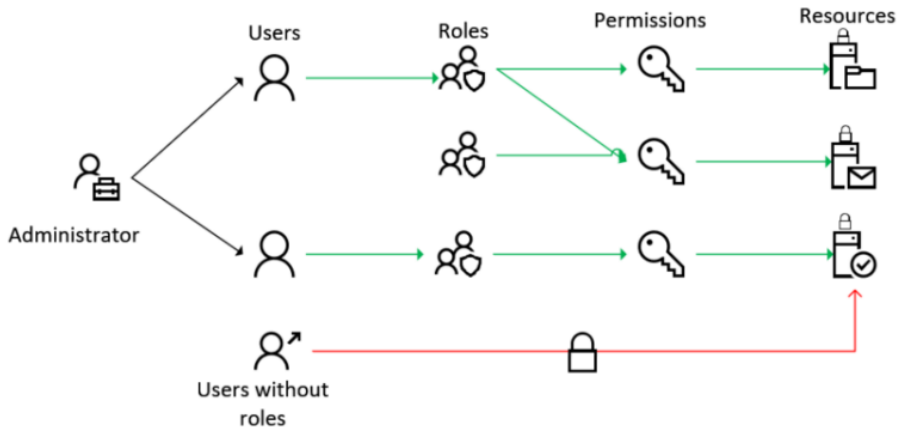

Figure 6: RBAC Overview

Mỗi **Role** được gắn với một tập hợp gồm các cặp **Resource** và **Action** phân biệt. Action thể hiện các hành động như create, retrieve, update, delete ... và Resource là tài nguyên của hệ thống. Để cấp quyền cho một User, ta chỉ việc gắn Role phù hợp cho User đó.
Thiết kế mô hình RBAC được biểu diễn bởi lược đồ ERD như sau:

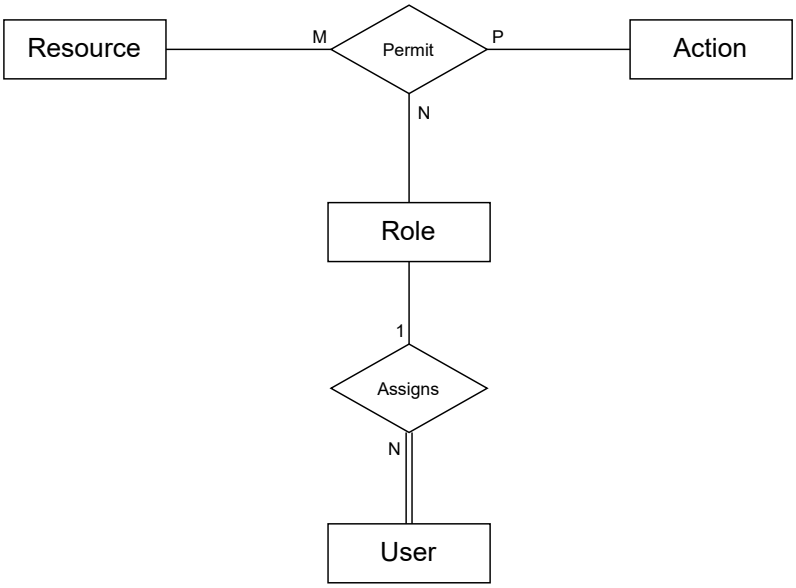

Figure 7: RBAC Model Design

Với mô hình RBAC, ta có thể gắn nhiều hơn một Role cho một User. Tuy nhiên theo yêu cầu chức năng số 2 từ Ban quản trị, một User chỉ có một Role tại cùng một thời điểm nên mối quan hệ giữa Role và User là 1-N.

Lưu ý rằng, mô hình RBAC được thiết kế như trên dùng để phân quyền tại lớp **ứng dụng**, cụ thể sẽ sử dụng để bảo mật API cho backend. Ta cần phân biệt phân quyền tại lớp ứng dụng và phân quyền tại lớp **Database** (được trình bày trong mục 4.4)

4.3 Thiết kế ý niệm

Tuỳ vào phân nhóm dữ liệu trong mục 4.1, ta sẽ có thiết kế khác nhau

4.3.1 Dữ liệu có thể thay đổi (nhóm 1)

Các thực thể chính thuộc nhóm 1 gồm: **Người dùng**, **Dự trù** và **Danh mục**. Người dùng (đã được cấp quyền) có thể tạo ra Dự trù và Danh mục. Một Dự trù bắt buộc phải thuộc một Danh mục nào đó. Ta có thiết kế ERD như sau:

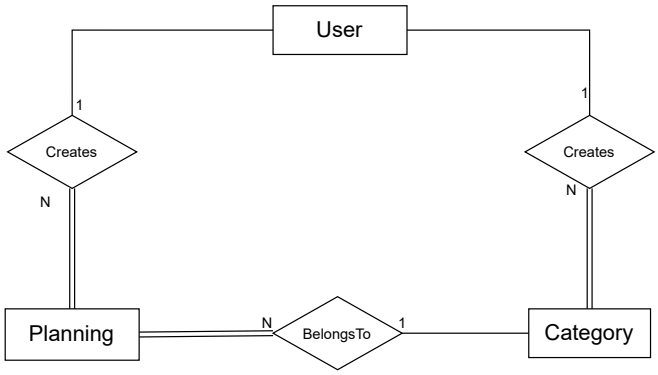

Figure 8: ERD cho nhóm 1

Kết hợp với mô hình phân quyền RBAC đã trình bày trong mục 3.2 và chỉ rõ các thuộc tính của từng thực thể, ta có lược đồ ERD hoàn chỉnh cho dữ liệu thuộc nhóm 1:

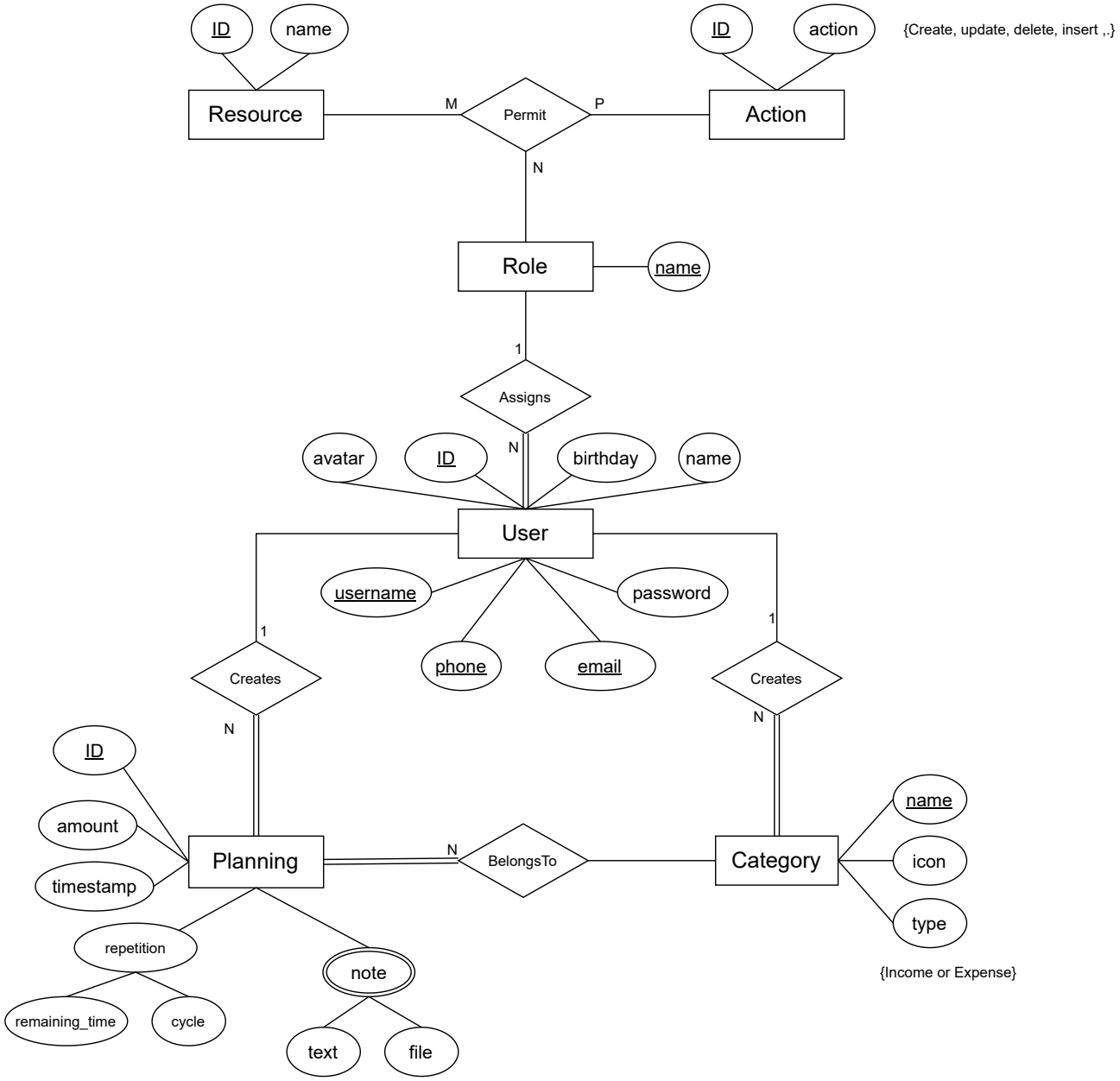

Figure 9: ERD hoàn chỉnh cho nhóm 1

Mức chuẩn hoá *tối đa* cho dữ liệu thuộc nhóm 1 là 3NF.

4.3.2 Dữ liệu không thay đổi sau khi lưu (Nhóm 2)

Các thực thể chính thuộc nhóm 2 gồm Transaction và Log. Vì tính chất đặc trưng chỉ cho phép các thao tác create và retrieve, để lưu trữ thông tin ta chỉ cần lưu trữ dữ liệu dưới dạng bản ghi thuần, với mức chuẩn hoá *tối đa* là 1NF.

Ngoài ra, các thực thể thuộc nhóm này không nên có mối liên kết với nhau, và không có liên kết với các thực thể thuộc nhóm 1. Để giải thích lý do, ta lấy một yêu cầu chức năng làm ví dụ điển hình: Các giao dịch nên được phân theo danh mục cụ thể, như tiền nhà, tiền điện nước ... Khi đó ta có lược đồ ERD như

sau:

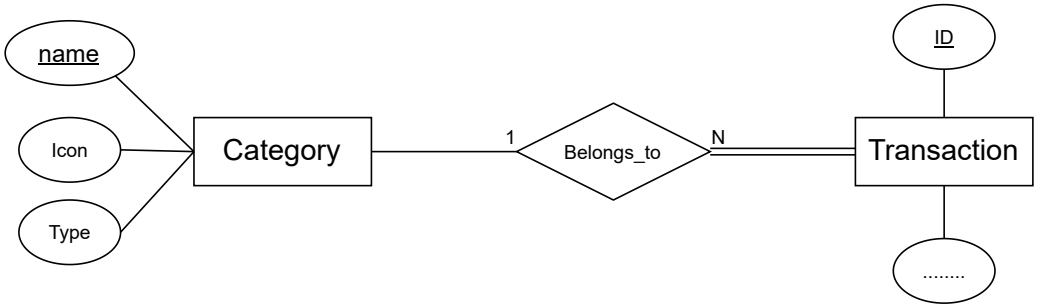

Figure 10: Ví dụ về mâu thuẫn giữa dữ liệu nhóm 1 và dữ liệu nhóm 2

Khi thực hiện mapping từ ERD sang database schema, bảng (hoặc quan hệ) Transaction sẽ chứa key của bảng Category với vai trò là foreign key. Như vậy, khi truy xuất thông tin của một giao dịch và danh mục tương ứng, ta sẽ sử dụng câu lệnh JOIN hai bảng và lấy thông tin cần thiết. Tuy nhiên, giá trị của các tuple trong bảng Category có thể thay đổi, dẫn tới câu truy vấn thay đổi kết quả theo. Điều này vi phạm quy tắc "Giao dịch không thể thay đổi". Vấn đề tương tự xảy ra với bất kỳ thực thể thuộc nhóm 2 có mối liên kết với thực thể thuộc nhóm 1.

Có nhiều cách giải quyết vấn đề này, chẳng hạn ta có thể copy hết tất cả những attribute có thể thay đổi giá trị và gắn vào relationship giữa hai thực thể. Trong ứng dụng này, nhóm chỉ lưu lại các giao dịch và log như một bản ghi. Tuỳ vào cấu trúc dữ liệu của giao dịch và log có thay đổi nhiều hay không, nhóm sẽ lựa chọn SQL hoặc NoSQL cho nhóm dữ liệu này.

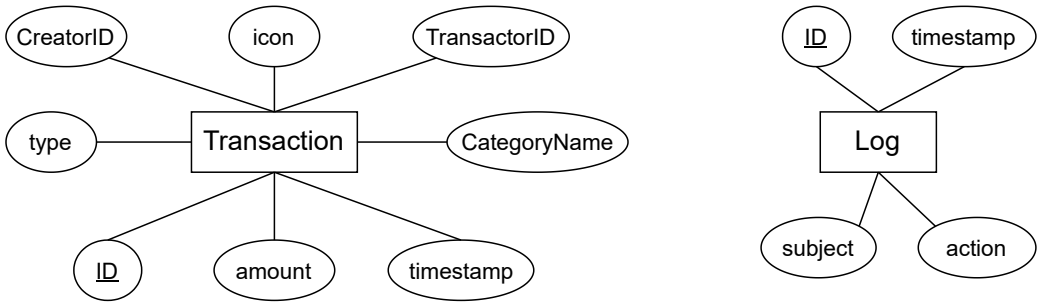

Figure 11: Mô hình dữ liệu thuộc nhóm 2

4.4 Điều khiển truy cập của DBMS

DBMS cần cung cấp một tài khoản cho việc kết nối với ứng dụng. Tài khoản này được cấp quyền ở mức tối thiểu, đáp ứng vừa đủ nhu cầu của ứng dụng. Hiện tại, chỉ có một ứng dụng quản lý thu chi sử dụng database được thiết kế, vì vậy nhóm sử dụng các lệnh GRANT, REVOKE để phân quyền cho tài khoản kết nối. Cách phân quyền này gọi là Discretionary Access Control (DAC).

Tài khoản không được có các quyền thực hiện các câu lệnh DDL như CREATE, ALTER, DROP (trong SQL). Tài khoản có một hoặc nhiều trong bốn quyền CRUD (Create, Retrieve, Update, Delete) trên mỗi bảng tuỳ vào logic ứng dụng. Ví dụ, với những bảng như Category, tài khoản chỉ có quyền Create và Retrieve.

5 Sao lưu dữ liệu và Bảo mật

5.1 Kế hoạch sao lưu dữ liệu

Nói về việc sao lưu dữ liệu, người ta thường đề cập tới quy tắc sao lưu 3-2-1. Quy tắc này chỉ ra rằng hệ thống phải có 3 bản sao dữ liệu trên ít nhất 2 phương tiện khác nhau với 1 bản sao được cất giữ ở nơi khác bên ngoài tổ chức. Nhóm dữ án đã chỉnh sửa lại quy tắc này một chút để phù hợp với tình hình thực tiễn.

Cụ thể dữ liệu của hệ thống sẽ có 2 bản sao thay vì 3 bản sao vì khả năng mất hoặc hỏng dữ liệu với môi trường hiện tại ở Lab là rất thấp nên không cần thiết phải có bản sao thứ 3. Trong đó, bản 1 là bản chính được lưu trong ổ cứng của server phục vụ cho việc vận hành. Bản 2 sẽ được lưu trên cloud của một bên thứ 3. Việc sao lưu dữ liệu vào bản 1 và 2 sẽ được hiện thực một cách tự động bằng phần mềm. Nhóm vẫn dữ lại nguyên tắc có 2 phương tiện sao lưu (sao lưu ở ổ cứng và sao lưu trên cloud) và nguyên tắc 1 bản sao được cất ở bên ngoài tổ chức (cloud của bên thứ 3).

Dữ liệu được sao lưu ở bản 1 và bản 2 bao gồm full copy và differential copy. Dữ liệu được sao lưu ở bản 3 chỉ chứa full copy. Ngoài ra bản 1 và bản 2 còn sao lưu thêm transaction logs phục vụ cho việc khôi phục dữ liệu tại một thời điểm nhất định.

Kế hoạch về thời gian sao lưu dữ liệu:

Chu kỳ	Loại sao lưu	Sao lưu vào
1 ngày	Differential backup	Bản 1, 2.
1 tuần	Full backup	Bản 1, bản 2
2 tuần	Full backup	Bản 3.

Table 1: Kế hoạch về thời gian sao lưu dữ liệu

5.2 Bảo mật phía Cơ sở dữ liệu (CSDL)

Để đảm bảo được bảo mật, CSDL của dự án phải đảm bảo được những yêu cầu sau đây:

1. Tạo ra các databse users khác nhau cho các ứng dụng khác nhau sử dụng chung một CSDL. Điều này có nghĩa là dự án Fund Management App (FMA) và các dự án khác trong tương lai sẽ được cấp các tài khoản database khác nhau và có các quyền (privileges) khác nhau tùy thuộc vào đặc thù của từng dự án.

2. Tài khoản root của database không được cấp cho bất cứ ứng dụng nào.

3. Các thông tin nhạy cảm và bí mật phải được mã hóa trong CSDL. Các thông tin này bao gồm nhưng không giới hạn mật khẩu của các tài khoản, thông tin cá nhân của các thành viên, thông tin dự án,...

4. CSDL phải hiện thực các phương pháp của Statistical Database security nhằm ngăn cản việc truy vấn các thông tin nhạy cảm một cách đơn lẻ.

5.3 Bảo mật phía Ứng dụng (application server)

Application server của ứng dụng phải đảm bảo được những yêu cầu sau đây:

1. Ngăn chặn được các tấn công SQL Injection tiềm năng. Các phương pháp ngăn chặn bao gồm (nhưng không giới hạn) sử dụng các Parameterized Statements, ORM frameworks hay Escaping Inputs.
2. Ngặn chặn được các tấn công Command Injection tiềm năng. Các phương pháp ngăn chặn bao gồm (nhưng không giới hạn) không gọi trực tiếp các lệnh command lines, sử dụng escape inputs hay giới hạn các quyền đối với chương trình.
3. Ngăn chặn được các tấn công Cross Site Scripting (XSS) tiềm năng. Các phương pháp ngăn chặn bao gồm (nhưng không giới hạn) sử dụng Whitelist values, hiện thực Content-Security Policy hay Santilize HTML
4. Ngăn chặn được các tấn công Cross Site Request Forgery (CSRF) tiềm năng. Các phương pháp ngăn chặn bao gồm (nhưng khôn giới hạn) tuân thủ nguyên tắc REST trong việc thiết kế, sử dụng Anti-CSRF cookies.
5. Các thông báo lỗi hiển thị phía người dùng không được gián tiếp hỗ trợ việc tấn công trở nên dễ dàng hơn. Ví dụ khi người dùng nhập đúng tên tài khoản nhưng sai mật khẩu thì phải thông báo là "Tài khoản hoặc mật khẩu không chính xác"
6. Địa chỉ các URLs phải sạch sẽ, không tiết lộ nhiều thông tin của server. Ví dụ không thêm các file suffixs như .php, .txt vào URLs.
7. Thông tin của cookies phải được hash và không có quy luật.
8. Thông tin file Javascript khi deploy phải được che dấu bằng kỹ thuật obfuscation.
9. Query strings không được chứa các thông tin nhạy cảm như cookie, mật khẩu,...
10. Các thông tin trao đổi giữa client và server phải được mã hóa bằng SSL hoặc TLS. Chứng chỉ SSL hoặc TLS có thể mua hoặc tự ký.
11. Các biến môi trường như thông tin kết nối database phải được lưu trữ riêng và không được ghi thằng trực tiếp vào source code.

5.4 Giải thích các thuật ngữ được sử dụng trong phần bảo mật

Thuật ngữ	Định nghĩa
SQL Injection	Là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn SQL để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database tấn công bằng cách chèn thêm một đoạn SQL để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database
Command Injection	Là một kỹ thuật tấn công lợi dụng lỗ hổng cho phép kẻ tấn công thực thi các lệnh bất kì của hệ điều hành trên server.
Cross Site Scripting	Là một kỹ thuật tấn công cho phép kẻ tấn công chèn những đoạn script độc hại vào phía browser của người dùng và chạy các đoạn script độc hại đó nhằm đánh cắp các dữ liệu quan trọng như cookie.
Cross Site Request Forgery	Là một kỹ thuật tấn công bằng cách sử dụng quyền chứng thực của người dùng đối với một website. CSRF là kỹ thuật tấn công vào người dùng, dựa vào đó hacker có thể thực thi những thao tác phải yêu cầu sự chứng thực.

Table 2: Bảng các thuật ngữ được sử dụng trong mục Bảo mật

6 Deployment

Như mô tả trong sơ đồ kiến trúc, ứng dụng được chia làm ba lớp: Frontend phía người dùng, Backend xử lý nghiệp vụ hệ thống và Database lưu trữ dữ liệu. Có hai hướng để deploy ứng dụng: deploy thủ công hoặc dựa vào dịch vụ hosting thứ ba. Ở cả hai cách, ta đều sử dụng Docker để đóng gói ứng dụng. Cách sử dụng theo các bước như sau:

- Ở mỗi lớp, ta chạy cách Dockerfile để đóng gói thành cách *image*. Các *image* này sẽ được chạy tạo ra các *container* tương ứng.
- Để định nghĩa và chạy multi-container cho ứng dụng, ta sử dụng Docker Compose. Tại đây, ta cần config docker networking để kết nối Backend và DBMS.

Lưu ý rằng, mỗi container là một process riêng biệt. Khi container bị xoá, dữ liệu trong container sẽ mất. Do đó, để giữ cho database đồng nhất giữa các version của ứng dụng, ta phải sử dụng Docker Volume.

Sau khi đóng gói thành công ứng dụng, ta có hai lựa chọn sẽ trình bày cụ thể sau đây.

6.1 Deploy thủ công

Với lựa chọn này, ta cần một máy chủ để chạy và quản lý ứng dụng. Có hai bước để ứng dụng có thể sử dụng trên internet:

- Chạy ứng dụng trên máy chủ
- Cấu hình mạng để máy chủ kết nối với thiết bị trong mạng nội bộ và với internet bên ngoài

Ta có sơ đồ mạng TickLab như trong Fig7. Các địa chỉ IP mang tính tượng trưng, thiết bị switch trong TickLab có chức năng tương tự như L3 Switch nên nhóm sử dụng L3 Switch trong sơ đồ này.

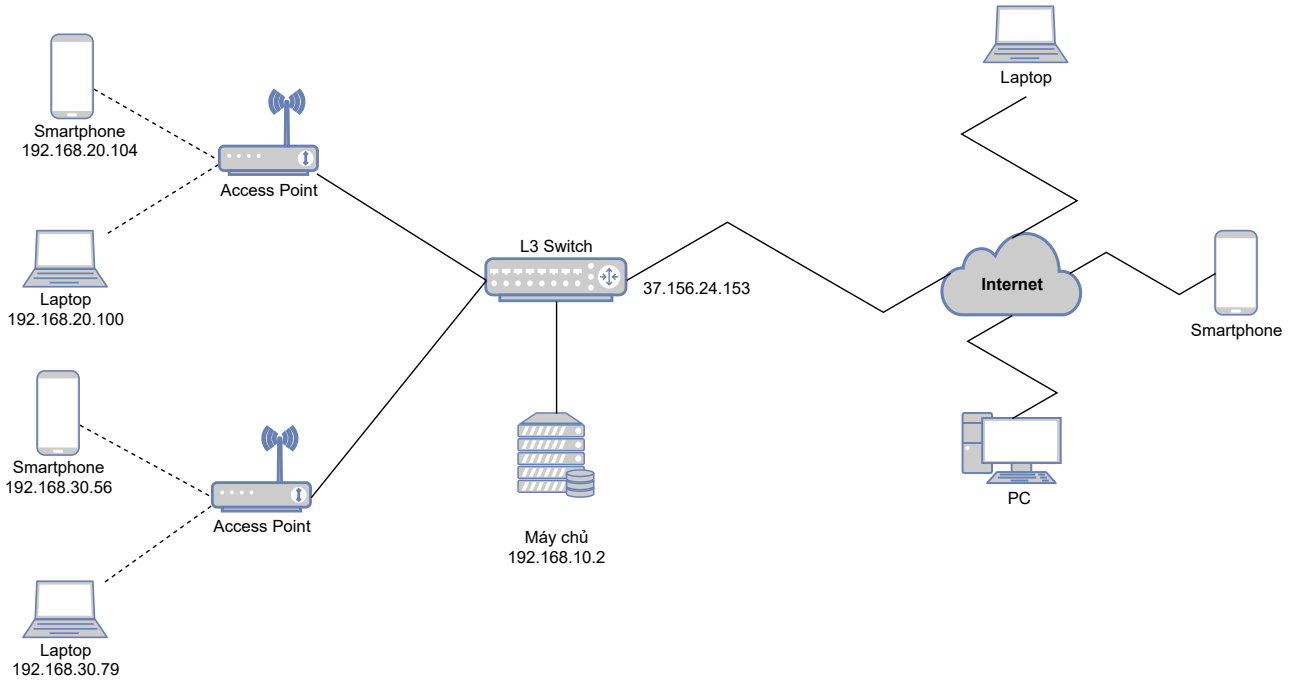

Figure 12: Sơ đồ mạng

Máy chủ sẽ khác VLAN với các thiết bị trong hệ thống mạng Ticklab. Ta cần config thiết L3 Switch sao cho các máy khác VLAN có thể kết nối được với nhau. Như vậy sau khi chạy ứng dụng tại máy chủ, các thiết bị trong cùng LAN có thể truy cập ứng dụng bằng cách nhập 192.168.10.2:3000 trên thanh URL của browser bất kỳ, với 192.168.10.2 là địa chỉ IP của máy chủ trong mạng LAN và 3000 là port ứng dụng đang chạy.

Để thiết bị bên ngoài internet có thể truy cập được ứng dụng, ta cần cài đặt Network Address Translation (NAT) trên thiết bị L3 Switch. Các gói tin gửi tới máy chủ sẽ được gửi tới địa chỉ IP của cổng kết nối L3 Switch với internet (địa chỉ này phải là public IP), sau đó dựa vào bảng NAT, L3 Switch sẽ forward gói tin tới máy chủ.

Với lựa chọn deploy thủ công, ta chỉ tốn chi phí thuê domain name cho địa chỉ IP của cổng kết nối L3 Switch với internet (trong Fig7 địa chỉ này là 37.156.24.153). Tuy nhiên, hiện tại nhóm không nắm rõ địa chỉ kết nối với internet của thiết bị Switch của TickLab có phải là public IP hay không. Nếu không phải là public IP, Switch phải nối với 1 router khác có public IP, thì nhóm có quyền config trên router đó hay không.

Tuỳ vào hoàn cảnh, nhóm sẽ có lựa chọn phù hợp.

6.2 Sử dụng dịch vụ hosting từ bên thứ ba

Với lựa chọn này, ta sẽ tìm một dịch vụ hosting phù hợp với technical stack của ứng dụng. Việc cần làm duy nhất là tạo một repository trên Docker Hub, push các image đã đóng gói và file *docker-compose.yml*, sau đó bàn giao cho bên cung cấp dịch vụ. Lựa chọn này sẽ có chi phí cao hơn so với deploy thủ công.

7 Kế hoạch

7.1 Các cột mốc quan trọng

#	Tiêu đề	Bắt đầu	Kết thúc	Nội dung	Kết quả
1	Thiết kế và viết đặc tả dự án	10/02/2022	31/03/2022	Viết các functional requirement và nonfunctional requirement, các diagram mô tả chức năng hệ thống, thiết kế cơ sở dữ liệu và kiến trúc phần mềm	Bản Software Specification
2	Hiện thực các MVP	10/02/2022	05/03/2022	Xây dựng database & Hiện thực được các MVP (gồm cả back-end lẫn front-end)	Release bản demo
3	Hiện thực ứng dụng hoàn chỉnh	06/03/2022	20/03/2021	Hoàn thiện các chức năng còn thiếu	Ứng dụng được deploy local
4	Kiểm thử ứng dụng	21/03/2022	31/03/2021		Testing hệ thống và chính thức đưa vào sử dụng
5	Thu hoạch	01/04/2022	07/04/2020	Đánh giá tổng kết dự án, viết document cho hệ thống	Có được bản software specification và tài liệu hướng dẫn sử dụng

7.2 Chi phí

Công việc thiết kế và hiện thực dự án đều do thành viên trong lab đảm nhận, vì vậy phần chi phí quy ra tiền mặt chỉ gồm chi phí cho việc deploy ứng dụng. Nếu chỉ mua domain name, chi phí khoảng 400k-500k VNĐ/năm (~33k-41k) chưa tính tiền điện. Nếu thuê dịch vụ hosting, chi phí vào khoảng 900k trở lên (> 75k/tháng).

=> Nhóm quyết định chọn cách deploy local.