Architecture Design

Task 3.1: Sử dụng kiến trúc phân lớp để thiết kế hệ thống HCMUT-SPSS

Để thực hiện hệ thống HCMUT-SSPS, nhóm quyết định xây dựng hệ thống dựa theo hướng tiếp cận mô hình kiến trúc MVC (Model - View - Controller). Đây là mô hình kiến trúc phân lớp rất phổ biến, nhóm lựa chọn kiến trúc này với một số lý do sau:

• Mỗi tính năng trong hệ thống SSPS thường đòi hỏi nhiều giao diện (View) và liên quan đến nhiều thao tác với dữ liệu. Ví dụ, tính năng cung cấp thông tin in đòi hỏi hiển thị thông tin in, quản lý máy in, và quản lý cài đặt in trên nhiều giao diện khác nhau. Mỗi giao diện này thường phải xử lý các thao tác như xem, chỉnh sửa, cập nhật dữ liệu tương ứng. Sử dụng kiến trúc MVC giúp tách biệt rõ ràng giữa các thành phần chính: Model, View, và Controller. Model chịu trách nhiệm về dữ liệu, View đảm nhiệm hiển thị thông tin cho người dùng, và Controller quản lý logic xử lý dữ liệu. Điều này giúp làm cho mã nguồn trở nên rõ ràng và dễ bảo trì.

• Sự tách biệt giữa Model, View và Controller cũng giúp cho việc mở rộng hệ thống trở nên dễ dàng hơn. Bất kỳ yêu cầu nào liên quan đến tương tác với dữ liệu có thể được thêm vào hệ thống mà không ảnh hưởng đến giao diện người dùng và ngược lại. Điều này cho phép tích hợp và mở rộng các tính năng mới một cách thuận tiện và độc lập.

Tuy nhiên, kiến trúc MVC cũng đi kèm với nhược điểm của sự phức tạp. Để đảm bảo sự tương tác hiệu quả giữa Model, View và Controller, phải đảm bảo rằng các phần tử này hoạt động cùng nhau một cách hợp nhất. Điều này đòi hỏi kiến thức kỹ thuật và quản lý dự án chặt chẽ để đảm bảo tính ổn định và bảo trì của hệ thống.

Task 3.2: Mô tả cách trình bày giao diện người dùng

Trang web HCMUT-SSPS là một nơi cung cấp giao diện cho người sử dụng để có thể thao tác và sử dụng các dịch vụ của hệ thống HCMUT-SSPS từ xa cách tiện lợi và nhanh chóng nhất. Do đó, việc thiết kế giao diện hệ thống phải đảm bảo được các tiêu chí về thẩm mĩ, thân thiện với người dùng, trực quan và dễ thao tác để người dùng có thể sử dụng dịch vụ một cách hiệu quả:

• Đầu tiên khi truy cập vào trang web của hệ thống, người sử dụng sẽ được điều hướng về trang chủ. Trang chủ ở đây chỉ thể hiện hình ảnh và thông điệp chào đón, để có thể sử dụng các dịch vụ của hệ thống trên vai trò của sinh viên hay của SPSO, người sử dụng phải đăng nhập bằng cách bấm vào nút đăng nhập được đặt trên trang chủ.

• Sau khi bấm vào nút đăng nhập, người sử dụng sẽ được chuyển hướng sang một trang mới phục vụ quá trình đăng nhập. Khi này người sử dụng sẽ nhập thông tin tài khoản bao gồm tên đăng nhập và mật khẩu vào khu vực mà giao diện cung cấp; thông báo lỗi sẽ hiển thị khi người dùng đăng nhập không thành công, ngược lại, người dùng sẽ có khả năng truy cập đến với các dịch vụ của hệ thống.

• Khi đã đăng nhập thành công, người dùng sẽ được chuyển hướng đến trang hoạt động chính của hệ thống. Ở trang này, phần truy cập thông tin người dùng sẽ được đặt trên trang hoạt động chính và nổi bật ở chính giữa trang chính là các chức năng dịch vụ mà người sử dụng có thể chọn để có thể sử dụng bao gồm In, Xem lịch sử in nếu vai trò đăng nhập là sinh viên và In, Xem lịch sử in, Xem báo cáo định kỳ, Cài đặt in và Quản lý hệ thống nếu vai trò đăng nhập là SPSO. Bằng cách thể hiện các icon lớn đi đôi với tên các chức năng cố định, người dùng có thể dễ dàng nhấp vào bất cứ icon nào mà không lẫn lộn để thực hiện chức năng mà mình cần. Khi nhấp chọn chức năng nào đó, người dùng sẽ được chuyển hướng đến trang riêng phục vụ cho chức năng đó.

• Ở mỗi trang riêng phục vụ cho mỗi chức năng, các thông tin cần thiết đều được cung cấp đầy đủ, ở vị trí dễ nhìn và đảm bảo tính bao quát thông tin cho người dùng. Đối với các chức năng bắt phải nhập dữ liệu, các vị trí cần nhập các dữ liệu cần thiết đều được cung cấp đầy đủ ở các vị trí dễ nhìn cho người dùng nhập vào, bên canh đó giao diện còn cung cấp chức năng kiểm tra dữ liệu nhập vào của người dùng để đưa ra các thông báo lỗi kịp thời. Với mỗi hoạt động khi thực hiện thao tác xác nhận, các thông báo lỗi và thành công đều được cung cấp để người dùng có thể biết được tiến độ và tình trạng công việc mà mình thực hiện.

– Đối với trang Xem báo cáo định kỳ, giao diện người dùng cung cấp các bảng điều khiển thống kê và biểu đồ trực quan, giúp SPSO có cái nhìn tổng quan về hoạt động của hệ thống.

– Các chức năng quản lý, như cài đặt in, quản lý máy in được thiết kế một cách chi tiết và dễ sử dụng.

∗ Đối với chức năng Quản lý cài đặt in, SPSO có thể thay đổi các cài đặt in như số lượng giấy cấp định kỳ cho sinh viên, ngày cấp giấy định kỳ, các loại file được phép in.

∗ Đối với chức năng Quản lý máy in, SPSO sẽ được thấy danh sách các máy in đang có trong hệ thống cùng với các thông tin như ID, mẫu máy in, số lượng giấy còn lại, tình trạng kết nối. SPSO có thể quản lý máy in bằng cách nhấn vào nút Settings bên cạnh máy in cần quản lý. SPSO cũng có thể thêm máy in mới vào hệ thống bằng cách chọn chức năng "Add Printer".

– Đối với chức năng xem lịch sử in, để nâng cao sự thuận tiện và tương tác, giao diện người dùng của HCMUT-SSPS cũng tích hợp các tính năng tiên tiến như tìm kiếm thông minh và bộ lọc.

∗ Đối với sinh viên, chức năng bộ lọc giúp họ có thể giúp họ tìm kiếm lịch sử in của mình trong khoảng thời gian xác định từ ngày nào đến ngày nào.

∗ Đối với SPSO, ngoài chức năng bộ lọc như dành cho sinh viên thì còn có chức năng xem lịch sử in của sinh viên bằng cách tìm kiếm theo mã số sinh viên của sinh viên đó hoặc xem lịch sử in của máy in bằng cách tìm kiếm theo mã máy in.

• Các danh sách máy in cũng như danh sách lịch sử in được tối ưu để chỉ hiển thị 4 đến 5 dòng trên màn hình được sắp xếp thứ tự ưu tiên các máy in đang hoạt động đối với danh sách máy in và ưu tiên các lịch sử in gần đây nhất đối với danh sách lịch sử in.

• Để tối ưu hóa trải nghiệm người dùng, giao diện người dùng được thiết kế có thể tương tác trên nhiều thiết bị khác nhau, bao gồm cả máy tính, máy tính bảng, và điện thoại di động. Giao diện linh hoạt và tự động điều chỉnh kích thước để phản ánh đúng trên mọi loại màn hình, đảm bảo rằng người dùng có thể trải nghiệm mượt mà và thuận lợi mọi lúc, mọi nơi.

Task 3.3: Mô tả cách lưu trữ dữ liệu

Để lưu trữ dữ liệu cho hệ thống HCMUT-SSPS, cần sử dụng một hệ thống quản lý cơ sở dữ liệu mạnh mẽ và linh hoạt. Trong số các phần mềm quản lý cơ sở dữ liệu hiện nay, MySQL là một lựa chọn hợp lý để thực hiện vấn đề này. Các ưu điểm của MySQL có thể kể đến như:

• Hiệu suất cao: MySQL được tối ưu hóa để cung cấp hiệu suất cao cho các truy vấn và ghi dữ liệu. Điều này là quan trọng khi có nhu cầu xử lý lớn lượng dữ liệu từ các máy in và người dùng.

• Tính đồng nhất và Tính nhất quán: MySQL tuân thủ ACID (Atomicity, Consistency, Isolation, Durability), đảm bảo tính nhất quán và đồng nhất của dữ liệu trong mọi trạng thái.

• Hỗ trợ truy vấn phức tạp: MySQL hỗ trợ ngôn ngữ truy vấn SQL mạnh mẽ, cho phép thực hiện các truy vấn phức tạp để truy xuất, cập nhật và xử lý dữ liệu.

• Tính dễ mở rộng: MySQL có thể được mở rộng dễ dàng để đáp ứng nhu cầu tăng cường về dữ liệu.

• Tích hợp tốt với nhiều ngôn ngữ Lập trình: MySQL tương thích với nhiều ngôn ngữ lập trình phổ biến như PHP, Python, Java, giúp việc phát triển ứng dụng web trở nên thuận tiện.

• An toàn và bảo mật: MySQL cung cấp các tính năng bảo mật mạnh mẽ như kiểm soát quyền truy cập, mã hóa dữ liệu và theo dõi hoạt động.

• Hỗ trợ nhiều hệ điều hành: MySQL có sẵn cho nhiều hệ điều hành, từ Linux đến Windows, tạo điều kiện thuận lợi cho triển khai đa nền tảng.

• Tính linh hoạt và tùy chỉnh: MySQL cho phép tùy chỉnh cấu hình theo nhu cầu cụ thể của dự án, giúp tối ưu hóa hiệu suất hệ thống.

• Hỗ trợ đa luồng: MySQL cho phép các tác vụ đa luồng theo nhu cầu của người sử dụng. Ví dụ như cho phép một file in trên nhiều máy in hoặc một máy in có thể in nhiều file tuần tự.

Hệ thống cơ sở dữ liệu cho trang web in ấn cần bao gồm các bảng chính để lưu trữ thông tin về máy in, người dùng, và các tài liệu in. Dưới đây là một số bảng cơ sở dữ liệu quan trọng:

• Bảng Máy In (Printers): Mã máy in (Printer ID), tên máy in, thông tin kết nối (địa chỉ IP, cổng,...), số lượng giấy còn lại, số lần sử dụng in.

• Bảng Người Dùng (Users): Tên đăng nhập, mật khẩu, quyền truy cập, số lượng giấy còn lại của người dùng, các thông tin khác liên quan đến tài khoản người dùng.

• Bảng Tài Liệu In (Printed Documents): Mã tài liệu, tên tài liệu, người dùng đã in, máy in sử dụng, ngày in, số lần in, số lượng giấy đã in.

Task 3.4: Mô tả cách truy cập vào các dịch vụ/API bên ngoài

Về mặt kỹ thuật, API (Application Programming Interface) là viết tắt của Giao diện Lập trình Ứng dụng, đó là một cơ chế trung gian phần mềm cho phép hai ứng dụng khác nhau tương tác và trao đổi dữ liệu với nhau. API có thể được sử dụng trong nhiều ngữ cảnh, bao gồm các hệ thống web-based, hệ điều hành, hệ thống cơ sở dữ liệu, phần cứng máy tính, hoặc thư viện phần mềm. Ở dạng đơn giản nhất, API cung cấp một giao diện cho phép một ứng dụng tương tác với ứng dụng khác thông qua các yêu cầu và phản hồi, và cách mà dữ liệu được truy cập thông qua API có thể khác nhau tùy thuộc vào loại API, chẳng hạn như API SOAP hoặc REST.

REST (Representational State Transfer) là một kiểu kiến trúc được sử dụng để thiết kế các API. Nó sử dụng các phương thức HTTP đơn giản như GET, POST, DELETE, và PUT để tạo giao tiếp giữa các máy. Thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi yêu cầu HTTP đến một URL cụ thể để thực hiện các thao tác trên dữ liệu.

Cuộc sống của HTTP dựa vào quy trình luân phiên giữa Yêu cầu (Request) và Phản hồi (Response). Khách hàng gửi một yêu cầu, máy chủ phản hồi để xác định liệu máy chủ có thể thực hiện yêu cầu đó hay không. API được xây dựng dựa trên hai thành phần chính này: Yêu cầu và Phản hồi.

Yêu cầu (Request) đúng chuẩn cần bao gồm bốn phần chính:

• URL: Đây là địa chỉ duy nhất cho tài nguyên hoặc dịch vụ mà yêu cầu được gửi đến. Ví dụ, nó có thể là địa chỉ của một trang web, hình ảnh hoặc video. API mở rộng ý tưởng URL ban đầu để áp dụng cho các tài nguyên khác như danh sách khách hàng, sản phẩm, và những thứ tương tự. Các tài nguyên này thường được gọi là "resources" - nguồn lực.

• Method: Đây là hành động mà khách hàng muốn thực hiện trên tài nguyên hoặc dịch vụ. Thường, method thường là các động từ và bao gồm bốn loại phổ biến: GET (yêu cầu máy chủ trả về tài nguyên), POST (yêu cầu máy chủ tạo ra một tài nguyên mới), PUT (yêu cầu máy chủ sửa đổi hoặc bổ sung thông tin vào một tài nguyên đã tồn tại), DELETE (yêu cầu máy chủ xóa một tài nguyên).

• Headers: Đây là nơi chứa các thông tin cần thiết cho yêu cầu, mà end-users thường không cần biết. Ví dụ, nó có thể chứa độ dài của dữ liệu yêu cầu, thời gian gửi yêu cầu, loại thiết bị đang sử dụng, và định dạng phản hồi mà khách hàng có thể đọc.

• Body: Đây là nơi chứa dữ liệu mà khách hàng muốn gửi đến máy chủ. Ví dụ, nếu bạn đặt một chiếc bánh pizza, thì phần nội dung trong yêu cầu sẽ chứa thông tin như loại bánh pizza, kích cỡ, số lượng đặt. Phản hồi (Response) là phản hồi từ máy chủ sau khi xử lý yêu cầu từ khách hàng. Cấu trúc của một phản hồi tương tự như yêu cầu, bao gồm:

• Status code: Đây là mã trạng thái mô tả kết quả của yêu cầu. Mã này thể hiện liệu yêu cầu đã thành công, bị từ chối hoặc có lỗi gì đó.

• Headers: Chứa thông tin quan trọng liên quan đến phản hồi, giống như phần headers của yêu cầu.

• Body: Chứa dữ liệu hoặc thông tin mà máy chủ gửi lại cho khách hàng sau khi xử lý yêu cầu.

Việc quản lý APIs liên quan đến việc tạo, xuất bản và quản lý các kết nối API trong một doanh nghiệp hoặc môi trường đám mây. Nó không chỉ là cách để kết nối các API lại với nhau, mà còn là một nền tảng cho phép doanh nghiệp chia sẻ và quản lý cấu hình API trong khi vẫn kiểm soát quyền truy cập, thu thập và phân tích thống kê sử dụng, và thực hiện các chính sách bảo mật liên quan.

Có nhiều công cụ quản lý API khác nhau, trong đó một số ví dụ như JMeter, một công cụ mã nguồn mở ban đầu thiết kế để kiểm tra hiệu suất phần mềm, và các nền tảng khác nhau được sử dụng để quản lý và kiểm soát quyền truy cập vào các dịch vụ và tài nguyên qua API.

Ngoài ra, có bốn loại chính của web APIs là Open APIs, Partner APIs, Internal APIs và Composite APIs, trong đó Open APIs thường được sử dụng rộng rãi. Open APIs thường là các API mã nguồn mở có thể được truy cập thông qua giao thức HTTP. Chúng xác định API endpoints, định dạng yêu cầu và phản hồi, và được sử dụng để chia sẻ dữ liệu và chức năng từ các ứng dụng và dịch vụ khác.