Câu 1: Kiến trúc hướng quy trình, hướng sự kiện là gì ?

- **Kiến trúc hướng quy trình**: là một mô hình thiết kế hệ thống trong đó các quy trình kinh doanh là trung tâm của thiết kế và triển khai hệ thống. Trong mô hình này, các quy trình kinh doanh được định nghĩa, quản lý và tự động hóa để tối ưu hóa hiệu quả và năng suất.
- **Các đặc điểm chính**:
    + **Tập trung vào quy trình:** Tất cả các hoạt động và dịch vụ trong hệ thống được xây dựng xung quanh các quy trình kinh doanh.
    + **Tự động hóa:** Quy trình kinh doanh được tự động hóa để giảm thiểu sự can thiệp của con người và tối ưu hóa hiệu suất.
    + **Quản lý quy trình:** Các quy trình kinh doanh có thể được theo dõi, quản lý và cải tiến liên tục.
    + **Tích hợp:** Dễ dàng tích hợp với các hệ thống khác thông qua các dịch vụ hoặc API.
- **Ưu điểm:**
    + Tăng cường hiệu quả và năng suất.
    + Cải thiện sự phối hợp giữa các bộ phận trong tổ chức.
    + Tạo điều kiện cho việc cải tiến liên tục.
- **Nhược điểm:**
    + Có thể phức tạp và tốn kém khi triển khai ban đầu.
    + Yêu cầu sự thay đổi về văn hóa và quy trình làm việc.
- **Kiến trúc hướng sự kiện**: là một mô hình thiết kế hệ thống trong đó các sự kiện là trung tâm của thiết kế và triển khai hệ thống. Trong mô hình này, các thành phần của hệ thống giao tiếp với nhau thông qua việc phát sinh và xử lý các sự kiện.
- **Các đặc điểm chính:**
    + **Phát sinh sự kiện:** Các hành động trong hệ thống tạo ra các sự kiện.
    + **Xử lý sự kiện:** Các thành phần khác của hệ thống lắng nghe và phản ứng với các sự kiện này.
    + **Không đồng bộ:** Xử lý sự kiện thường không đồng bộ, nghĩa là các thành phần không cần chờ đợi phản hồi trực tiếp từ các thành phần khác.
    + **Tính mở rộng:** Dễ dàng mở rộng và linh hoạt, phù hợp với các hệ thống phân tán và có độ phức tạp cao.
- **Ưu điểm:**
    + Tăng cường tính linh hoạt và khả năng mở rộng.
    + Giảm sự phụ thuộc giữa các thành phần trong hệ thống.
    + Tăng cường khả năng phản ứng nhanh với các thay đổi và sự kiện.
- **Nhược điểm:**
    + Có thể khó khăn trong việc theo dõi và gỡ lỗi.
    + Cần có cơ chế đảm bảo tính nhất quán và độ tin cậy của sự kiện.

Câu 2: Các chiến lược và công cụ nào có thể sử dụng để tăng cường hiệu suất của web server

1. **Hạn chế HTTP requests:** mỗi requests cần nhiều băng thông, bộ nhớ và tài nguyên CPU ở cả hai phía, có thể gây ra độ trễ và tắc nghẽn. Để hạn chế HTTP requests, có thể sử dụng các kĩ thuật như sau:
    - Kết hợp nhiều file CSS và JavaScript thành 1
    -Sử dụng CSS Image Sprites để kết hợp nhiều hình ảnh
    -Sử dụng Lazy Loading để hạn chế tải các hình ảnh và các element chưa sử dụng đến.
2. **Tối ưu cấu hình web server:**

- Thay đổi và nâng cấp hệ thống lưu trữ đóng vai trò qua quan trọng trong việc duy trì tốc độ website khi dự án phát triển.

- Đầu tư vào server chuyên dụng sẽ đáp ứng 1 lượng truy cập lớn mà không gây chậm trễ hay xảy ra sự cố, đặc biệt trong thời gian cao điểm như Black Friday.

3. **Quan sát và phân tích hiệu suất web server:**
    - Để đảm bảo web server được tối ưu, cần phải có sự đo lường và phân tích các thông số và các chỉ số phản ánh độ hiệu quả của nó. Bao gồm tốc độ phản hồi và thông lượng, chỉ số lỗi, tính có sẵn và mức độ sử dụng tài nguyên. Để quan sát và phân tích hiệu quả, có thể sử dụng các file logs, các trang web cung cấp các tool phân tích như Google Analytics và Matomo, các trang web test hiệu năng như Apache JMeter hoặc LoadRunner, và các trang web cho phép quan sát như New Relic hoặc Datadog.
4. **Tối ưu hóa code và truy vấn database:**

- Có thể sẽ có những thuật toán và data không sử dụng hay không cần thiết xuất hiện trong mã nguồn của trang web, và chúng có thể có tác động rất lớn đến với hiệu suất của web server. Để đảm bảo hiệu năng của web server, áp dụng nguyên tắc DRY, tránh viết lặp bất kì đoạn mã nào, các vòng lặp hay các tính toán không cần thiết, sử dụng các thuật toán và cấu trúc dữ liệu phù hợp.

Câu 3: Sử dụng kết hợp cả 2 server service

Sử dụng cả 2 web server là Apache và Nginx cùng nhau sẽ giúp tối ưu được điểm mạnh của từng web server. Thông thường, Nginx sẽ được đặt trước Apache như reverse proxy. Điều này sẽ cho phép Nginx quản lý các request từ client, giúp tận dụng tốc độ xử lý nhanh và khả năng xử lý số lượng lớn các kết nối đồng thời.

Nginx vượt trội đối với các nội dung tĩnh, các files sẽ được xử lý nhanh gọn và được gửi trực tiếp tới client. Còn với các nội dung động, ví dụ như các file PHP, Nginx sẽ ủy quyền cho Apache, Apache sẽ xử lý và trả về kết quả trang đã được rendered, sau đó Nginx sẽ truyền kết quả nhận được đến cho client.

Câu 4 : Sử dụng Nginx làm Web Server và Proxy

Ưu điểm:

1. Đơn giản hóa cấu trúc hệ thống: Giảm bớt số lượng thành phần cần quản lý, cấu hình, và bảo trì.
2. Hiệu suất tốt: Nginx có hiệu suất cao trong việc xử lý các yêu cầu HTTP và có thể hoạt động hiệu quả như cả web server và reverse proxy.

3. Tiết kiệm tài nguyên: Một tiến trình duy nhất có thể giảm thiểu việc sử dụng tài nguyên hệ thống so với việc chạy nhiều dịch vụ.
4. Dễ dàng cấu hình và quản lý: Một tệp cấu hình Nginx duy nhất có thể kiểm soát cả hai chức năng, giúp quản lý dễ dàng hơn.

Nhược điểm:

1. Giới hạn khả năng mở rộng: Nếu ứng dụng phát triển lớn hơn, việc gộp chung có thể trở nên khó quản lý và mở rộng.
2. Rủi ro bảo mật: Nếu Nginx bị tấn công, cả web server và proxy của bạn đều có thể bị ảnh hưởng.

Sử dụng riêng từng chức năng

Ưu điểm:

1. Tính mô-đun và linh hoạt: Dễ dàng mở rộng và thay thế từng phần của hệ thống mà không ảnh hưởng đến các phần khác.
2. Tối ưu hóa cho từng nhiệm vụ: Có thể tùy chỉnh từng thành phần để phù hợp nhất với công việc của nó (ví dụ: sử dụng Nginx làm proxy và một web server khác như Apache hoặc một server ứng dụng chuyên biệt).
3. Bảo mật tốt hơn: Tách biệt các chức năng giúp hạn chế phạm vi của các lỗ hổng bảo mật.

Nhược điểm:

1. Phức tạp hơn trong quản lý: Cần phải quản lý và cấu hình nhiều dịch vụ hơn.
2. Tiêu tốn nhiều tài nguyên hơn: Chạy nhiều tiến trình có thể yêu cầu nhiều tài nguyên hệ thống hơn.

Kết luận:

- Nếu ta có một **hệ thống nhỏ đến trung bình và muốn đơn giản hóa việc quản lý**: Sử dụng Nginx làm cả Web Server và Proxy là lựa chọn tốt.
- Nếu ta có một **hệ thống lớn hoặc phức tạp, yêu cầu mở rộng linh hoạt và bảo mật cao hơn**: Tách biệt các chức năng là lựa chọn hợp lý hơn.

Câu 5: Apache có thể xử lý những loại tập tin nào ? Apache có những biện pháp bảo mật nào cho máy chủ ?

A. Apache có thể xử lý tập lệnh CGI và các tệp được phân tích từ máy chủ, cụ thể là các tệp có phần mở rộng là **.cgi, .pl, .plx, .ppl, .perl, .shtml.**

B. Máy chủ Apache HTTP cung cấp một số phương pháp bảo mật cho máy chủ như sau:
   1. **Kiểm soát truy cập:**
      a. Tệp .htaccess: Tệp cấu hình cho phép kiểm soát truy cập cấp thư mục.
      b. Kiểm soát truy cập dựa trên IP: Cho phép hoặc từ chối quyền truy cập vào các địa chỉ hoặc phạm vi IP cụ thể bằng cách sử dụng các lệnh như Require, Allow, và Deny.

    c. Xác thực và ủy quyền người dùng: Sử dụng mod_auth_basic và mod_auth_digest để xác thực cơ bản và thông báo.
2. **Mã hóa:**
    a. SSL/TLS: Sử dụng mod_ssl, Apache hỗ trợ SSL và TLS để mã hóa dữ liệu được truyền giữa máy chủ và máy khách.
    b. HTTPS: Định cấu hình Apache để sử dụng HTTPS đảm bảo liên lạc an toàn.
3. **Ghi nhật ký và giám sát:**
    a. Access and Error Logs: Giữ nhật ký chi tiết để theo dõi và xem xét các lỗi và quyền truy cập.
    b. ModSecurity: Tường lửa ứng dụng web cung cấp khả năng bảo vệ chống lại các cuộc tấn công dựa trên web khác nhau.
4. **Bảo mật cấu hình:**
    a. Server Tokens: Định cấu hình ServerTokens và ServerSignature để ẩn thông tin phiên bản Apache.
    b. Disabling Unnecessary Modules: Chỉ kích hoạt các mô-đun được yêu cầu để giảm bề mặt tấn công.
    c. File Permissions: Đặt quyền chính xác cho tệp và thư mục để hạn chế quyền truy cập.
5. **Yêu cầu lọc và giới hạn:**
    a. mod_evasive: Bảo vệ chống lại các cuộc tấn công từ chối dịch vụ (DoS) và từ chối dịch vụ phân tán (DDoS).
    b. mod_reqtimeout: Đặt thời gian chờ và tốc độ dữ liệu tối thiểu để nhận yêu cầu.
    c. mod_security: Cung cấp khả năng tường lửa ứng dụng web toàn diện.
6. **Xác thực đầu vào:**
    a. mod_rewrite: Có thể được sử dụng để tạo quy tắc viết lại URL nhằm xác thực và vệ sinh đầu vào.
    b. mod_proxy: Định cấu hình proxy an toàn để ngăn chặn các lỗ hổng proxy mở.

7. **Bảo mật máy chủ ảo:** Đảm bảo máy chủ ảo được cấu hình an toàn để ngăn chặn rò rỉ thông tin giữa các trang web.
8. **Hạn chế phương thức HTTP:** Giới hạn các phương thức HTTP (ví dụ: vô hiệu hóa các phương thức TRACE, OPTIONS, PUT và DELETE nếu không cần thiết) bằng cách sử dụng các lệnh Limit và LimitExcept.
9. **Bảo mật nội dung:**
    a. Content Security Policy (CSP): Triển khai các tiêu đề CSP để ngăn chặn các cuộc tấn công XSS.
    b. Cross-Origin Resource Sharing (CORS): Định cấu hình tiêu đề CORS để kiểm soát việc chia sẻ tài nguyên trên các nguồn gốc khác nhau.

Cuối cùng, hãy thường xuyên cập nhật Apache và tất cả các mô-đun liên quan lên phiên bản mới nhất để vá các lỗ hổng đã biết.

Câu 6: Tại sao NGINX có khả năng mở rộng và hiệu suất cao hơn Apache ?

NGINX có khả năng mở rộng và đem lại hiệu suất cao hơn Apache. Điều này xuất phát từ cách hoạt động của cả hai máy chủ web:

1. **Apache:**
    - Linh hoạt và phạm vi module rộng hơn: Apache là một máy chủ web linh hoạt và được sử dụng rộng rãi. Nó cung cấp tính linh hoạt và có thể được tùy chỉnh thông qua việc bổ sung các mô đun vào Web Server.
    - Cấu trúc mô hình “one-server-does-all”: Apache triển khai mô hình này, trong đó một máy chủ duy nhất xử lý tất cả các yêu cầu. Tuy nhiên, khi lưu lượng truy cập tăng hoặc trang web phức tạp hơn, hiệu suất của Apache có thể bị ảnh hưởng.
2. **NGINX:**
    - Hiệu suất và khả năng mở rộng tốt hơn: NGINX hoạt động dựa trên cấu trúc Event-Driven và có khả năng mở rộng tốt hơn. Mỗi Worker Process của NGINX có thể đồng thời xử lý hàng nghìn kết nối HTTP, giúp xử lý các quá trình tải dữ liệu lớn dễ dàng hơn. Đồng thời, NGINX dự đoán về mức sử dụng RAM, CPU và độ trễ chính xác hơn.

Tóm lại, NGINX là lựa chọn tốt cho các ứng dụng web lớn và có lượng truy cập cao, trong khi Apache phù hợp với các ứng dụng web phức tạp và có tính đa dạng hơn.