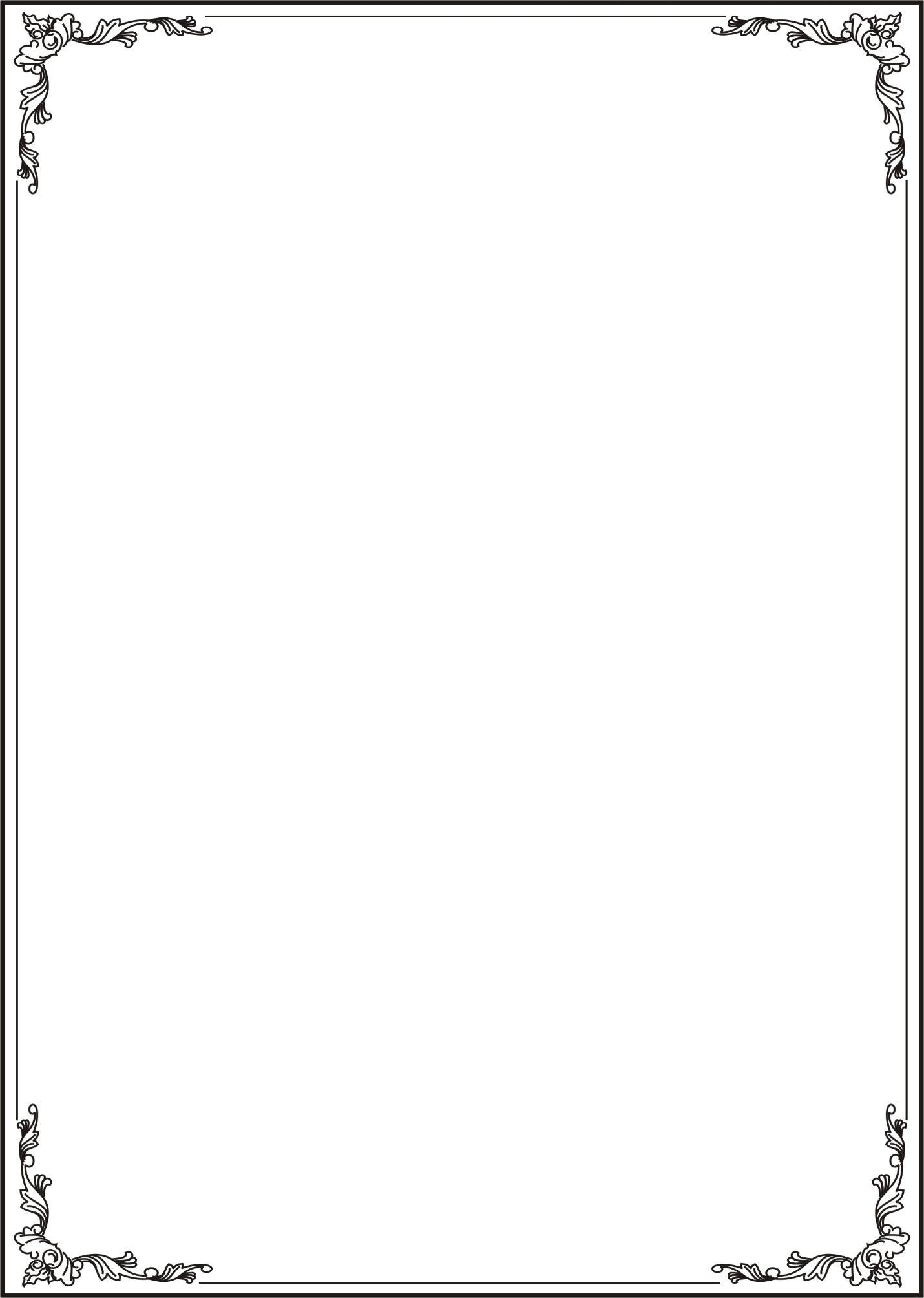
****

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

A blue logo with a black background

Description automatically generated with low confidence

**BÁO CÁO TIẾN ĐỘ**

**SO SÁNH HIỆU SUẤT GIỮA GIN (GO) VÀ SPRING BOOT (JAVA)**

**SE400.P11.PMCL**

**SEMINAR CÁC VẤN ĐỀ HIỆN ĐẠI CỦA**

**CÔNG NGHỆ PHẦN MỀM**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | **ThS. Đinh Nguyễn Anh Dũng** |
| **Thành viên:** | * **Nguyễn Tiến Vĩ - 21522788** * **Nguyễn Đình Khoa - 21520997** |

Thành phố Hồ Chí Minh, ngày 12 tháng 1, năm 2024

**DANH SÁCH THÀNH VIÊN**

|  |  |  |
| --- | --- | --- |
| **STT** | **Họ và tên** | **Mã số sinh viên** |
| **1** | **Nguyễn Tiến Vĩ** | **21522788** |
| **2** | **Nguyễn Đình Khoa** | **21520997** |

1. **Tổng quan**
   1. **Tổng quan về đề tài**

Đề tài tập trung vào việc so sánh hiệu suất giữa hai Framework phát triển API phổ biến hiện nay là Gin (một Framework của Go) và Spring Boot (một Framework lâu đời và phổ biến của hệ sinh thái Java). Việc so sánh hiệu suất giữa 2 Framework này sẽ giúp xác định các điểm mạnh và yếu của từng Framework trong ứng dụng xử lý API hiệu quả cao, từ đó hỗ trợ quyết định lựa chọn công nghệ phù hợp cho hệ thống.

* 1. **Lý do chọn đề tài**

- **Độ phổ biến của Framework:** Gin và Spring Boot là hai trong số những lựa chọn hàng đầu hiện nay khi triển khai các ứng dụng back-end. Gin nổi bật vì sự tối giản, trong khi vẫn đảm bảo tốc độ nhanh nhờ cách tiếp cận hướng hiệu suất của Go. Spring Boot thì mạnh mẽ nhờ các tính năng và số lượng thư viện phong phú của Java với lịch sử phát triển lâu đời. So sánh này giúp làm rõ các lợi ích cũng như hạn chế của mỗi Framework trong bối cảnh yêu cầu thực tế.

**- Khả năng tối ưu hóa hiệu suất:** Trong bối cảnh hiện đại ngày nay, phần lớn các hệ thống API đều cần xử lý số lượng yêu cầu lớn và đảm bảo tốc độ phản hồi nhanh. Gin và Spring Boot đều có khả năng xây dựng API hiệu quả, tuy nhiên mỗi Framework có cách quản lý tài nguyên, tối ưu hóa bộ nhớ và xử lý các tác vụ bất đồng bộ khác biệt. Việc kiểm tra hiệu suất sẽ cho góc nhìn rõ ràng hơn về hiệu quả và khả năng đáp ứng của mỗi Framework.

1. **Nội dung và phương pháp**
   1. **Nội dung**

Để đảm bảo kết quả so sánh toàn diện giữa hai framework, các trường hợp thử nghiệm sẽ bao gồm ba tình huống: **hệ thống ở trạng thái idle, request trả về dữ liệu JSON tĩnh và request trả về dữ liệu từ cơ sở dữ liệu.** Các tiêu chí đo lường trong quá trình thử nghiệm là: mức độ sử dụng tài nguyên hệ thống (CPU, bộ nhớ), độ trễ (latency), thông lượng (throughput)

* + 1. **Trường hợp hệ thống idle**

**- Mục tiêu:** Đo lường mức sử dụng tài nguyên hệ thống (CPU, bộ nhớ) khi hệ thống không xử lý yêu cầu để xác định mức tài nguyên cần thiết để duy trì hệ thống trong trạng thái sẵn sàng làm việc.

**- Cách thực hiện:** Khởi động ứng dụng Gin và Spring Boot và để chúng hoạt động ở trạng thái chờ trong một khoảng thời gian mà không có request nào

**- Kỳ vọng kết quả:** Vì cả hai hệ thống đều ở trạng thái không xử lý request nên mức tiêu thụ tài nguyên ở mức tối thiểu. Kết quả sẽ giúp so sánh mức tiêu thụ tài nguyên để duy trì trạng thái sẵn sàng giữa hai Framework.

* + 1. **Request trả về dữ liệu JSON tĩnh**

**- Mục tiêu:** Đánh giá hiệu suất khi hệ thống chỉ xử lý các tác vụ trả về dữ liệu JSON đơn giản mà không cần thực hiện truy vấn cơ sở dữ liệu.

**- Cách thực hiện:**

* Tạo một endpoint trả về một đối tượng JSON tĩnh, giả lập dữ liệu người dùng, ví dụ:{ "id": 1, "name": "Test User", "email": "user@example.com", "role": "admin" }
* Ứng dụng client gửi một loạt yêu cầu đến endpoint này với số lượng tăng dần (ví dụ: từ 100 đến 10000 yêu cầu/giây) để mô phỏng tải thực tế.

**- Kỳ vọng kết quả:** Vì đây là dữ liệu tĩnh nên thời gian xử lý chỉ phụ thuộc vào hiệu suất của Framework và cấu trúc JSON. Gin sẽ có lợi thế vì được tối ưu hóa cho tốc độ và các tác vụ xử lý đơn giản.

* + 1. **Request trả về dữ liệu JSON tĩnh**

**- Mục tiêu:** Đánh giá hiệu suất khi hệ thống phải thực hiện truy vấn cơ sở dữ liệu, nhằm đánh giá khả năng xử lý của từng Framework khi làm việc với cơ sở dữ liệu.

**- Cách thực hiện:**

* Thiết lập một cơ sở dữ liệu PostgreSQL chứa dữ liệu người dùng. Mỗi ứng dụng Gin và Spring Boot sẽ có một endpoint như /users, truy vấn danh sách người dùng từ cơ sở dữ liệu.
* Ứng dụng client gửi một loạt yêu cầu đến endpoint này với số lượng tăng dần (ví dụ: từ 100 đến 10000 yêu cầu/giây) để mô phỏng tải thực tế.

**- Kỳ vọng kết quả:** Spring Boot có thể tận dụng các tính năng quản lý kết nối của Java để duy trì hiệu suất ổn định trong khi Gin có thể vượt trội về tốc độ xử lý các truy vấn đơn giản nhờ khả năng tối ưu hóa của Golang. Sự khác biệt về tốc độ xử lý và mức tiêu thụ tài nguyên có thể thấy rõ khi tăng số lượng request hoặc khi truy vấn dữ liệu lớn.

1. f