**MÔ TẢ BÀI LUẬN**

**LẬP TRÌNH WEB VỚI NODEJS - 502070**  
**HỌC KỲ 1 – NĂM HỌC 2024 – 2025**

**Giảng viên: Mai Văn Mạnh**

**Tổng quan và Mục đích**

Mục đích của bài luận này là giúp sinh viên hiểu toàn diện về Docker, một nền tảng mạnh mẽ để phát triển, triển khai, và chạy các ứng dụng bên trong container. Qua bài luận này, sinh viên sẽ khám phá các nền tảng lý thuyết của Docker và thể hiện kỹ năng thực hành bằng cách sử dụng Docker hóa một ứng dụng web Node.js. Mục tiêu là giúp sinh viên học cách sử dụng Docker để triển khai dịch vụ và ứng dụng web theo cách có thể mở rộng và duy trì.

**Kết quả mong đợi:**  
Cuối cùng của bài tập này, sinh viên sẽ có thể:

* Hiểu các khái niệm cốt lõi về Docker và containerization.
* Sử dụng Docker để triển khai một ứng dụng web đa dịch vụ.
* Mở rộng ứng dụng bằng Docker Compose hoặc các công cụ điều phối container khác.
* Áp dụng các khái niệm nâng cao như cân bằng tải, phân tách dịch vụ, và điều phối.

**Mô tả Nhiệm vụ**

Sinh viên sẽ làm việc theo nhóm (2-3 thành viên) để:

* Nghiên cứu và hiểu các khía cạnh lý thuyết của Docker, bao gồm kiến trúc, lợi ích, và các trường hợp sử dụng trong phát triển phần mềm hiện đại.
* Thực hiện một bài thực hành (lab) sử dụng Docker và Docker Compose. Các nhóm nâng cao được khuyến khích khám phá các công cụ điều phối container như Docker Swarm, Kubernetes hoặc ECS.

**Cấu trúc Báo cáo**

Báo cáo phải tuân theo định dạng tiêu chuẩn của khoa và bao gồm các phần sau:

1. **Giới thiệu về Docker**:
   * Định nghĩa Docker và giải thích tại sao nó quan trọng trong phát triển web hiện đại.
   * Giới thiệu các khái niệm chính như containers, images, Dockerfile, Docker Compose, và container orchestration.
2. **Khảo sát Lý thuyết**:
   * Giải thích kiến trúc của Docker và công nghệ nền tảng của nó (ví dụ: namespaces, cgroups, union file systems).
   * Thảo luận về cách Docker đơn giản hóa phát triển và triển khai các ứng dụng Node.js.
   * Đề cập đến lợi ích và thách thức của containerization trong phát triển web (như mở rộng, cô lập, và tự động hóa triển khai).
3. **Tổng quan Dự án**:
   * Giới thiệu dự án cụ thể mà nhóm của bạn chọn để trình diễn.
   * Giải thích bối cảnh của dự án và lý do sử dụng Docker trong tình huống này.
   * Mô tả các dịch vụ tham gia (ví dụ: front-end, back-end, cơ sở dữ liệu) và cách chúng tương tác với nhau.
4. **Kiến trúc và Triển khai**:
   * Cung cấp giải thích chi tiết về kiến trúc mà nhóm của bạn đã thiết kế, tập trung vào cách Docker và Docker Compose được sử dụng.
   * Bao gồm mô tả rõ ràng về các dịch vụ được triển khai và các tương tác của chúng, kèm theo Docker Compose file (nếu sử dụng) và Dockerfiles.
   * Đối với các nhóm nâng cao, mô tả cách bạn triển khai mở rộng, cân bằng tải, và phân tách dịch vụ.
   * Nếu các công cụ điều phối (ví dụ: Docker Swarm hoặc Kubernetes) được sử dụng, giải thích vai trò của chúng trong dự án và cách chúng cải thiện việc triển khai.
5. **Kết quả và Thảo luận**:
   * Trình bày kết quả của dự án Docker hóa của bạn.
   * Thảo luận về hiệu suất, khả năng mở rộng, và độ tin cậy của giải pháp của bạn.
   * Làm nổi bật bất kỳ thách thức nào đã gặp phải và cách bạn vượt qua chúng.
6. **Kết luận**:
   * Tóm tắt những bài học chính từ dự án này.
   * Suy ngẫm về cách Docker có thể nâng cao quá trình phát triển và triển khai ứng dụng web hiện đại.

**Mức độ Khó**

Mỗi nhóm phải chọn một mức độ thử thách để thể hiện sự hiểu biết về Docker:

* **Mức 1:**
  + Triển khai một hệ thống Docker đơn giản bằng Docker Compose.
  + Hệ thống phải bao gồm ít nhất 3 dịch vụ: front-end (ví dụ: HTML, React, Angular), back-end (Node.js), và cơ sở dữ liệu (ví dụ: MySQL, MongoDB).
  + Đảm bảo các dịch vụ này nằm trong cùng một mạng Docker và có thể giao tiếp với nhau.
* **Mức 2:**
  + Bao gồm tất cả các yêu cầu của Mức 1.
  + Triển khai một hệ thống có khả năng mở rộng bằng cách nhân bản một dịch vụ (ví dụ: mở rộng dịch vụ back-end thành nhiều container) và triển khai cân bằng tải giữa chúng.
  + Áp dụng phân tách giữa các dịch vụ bằng cách sử dụng các dịch vụ trung gian như RabbitMQ hoặc Redis để truyền tin nhắn hoặc lưu trữ bộ đệm.
* **Mức 3:**
  + Bao gồm tất cả các yêu cầu của Mức 2.
  + Trình diễn cách dự án có thể được triển khai trên một nền tảng điều phối container như Docker Swarm hoặc Kubernetes.
  + Giải thích cách điều phối cải thiện khả năng mở rộng, tính khả dụng, và quản lý dịch vụ.

**Sản phẩm bàn giao**

Mỗi nhóm phải nộp các sản phẩm bàn giao sau:

1. **Báo cáo Hoàn chỉnh (Word hoặc PDF)**
   * Đảm bảo báo cáo được cấu trúc và định dạng theo hướng dẫn của khoa.
   * Bao gồm tất cả các phần được đề cập ở trên, bao gồm khảo sát lý thuyết, mô tả dự án, kiến trúc và kết quả.
2. **Mã nguồn và các Phụ thuộc**
   * Nộp tất cả mã nguồn cần thiết cho ứng dụng web Node.js.
   * Bao gồm các Dockerfile, Docker Compose file, và các file cấu hình khác.
   * Đảm bảo rằng tất cả các phụ thuộc (ví dụ: package.json) đều được cung cấp và dự án có thể được xây dựng và triển khai bằng Docker.
3. **Video Trình bày**
   * Tạo một video giải thích về dự án của bạn và trình diễn ứng dụng Docker hóa.
   * Trình bày cách giải pháp của bạn được xây dựng, triển khai và mở rộng.
   * Video nên bao gồm tất cả các tính năng chính của dự án và cung cấp những cái nhìn sâu sắc về cách bạn đã vượt qua các thách thức.
4. **ReadMe.txt**
   * Cung cấp một file ReadMe.txt rõ ràng và ngắn gọn bao gồm hướng dẫn chạy dự án trên máy cục bộ.
   * Bao gồm chi tiết về cách xây dựng các image, chạy các container, và kiểm thử ứng dụng.

**Tiêu chí** | **Điểm** | **0%** | **25%** | **50% - 75%** | **Điểm tối đa**

**Báo cáo (6.0 điểm)**

1. **Hiểu biết lý thuyết (2.0 điểm)**  
   Không có sự hiểu biết về Docker hoặc thiếu phần lý thuyết. (0%)  
   Hiểu biết cơ bản, bao quát vài khái niệm Docker nhưng thiếu chi tiết hoặc rõ ràng. (25%)  
   Bao quát các khái niệm chính nhưng thiếu chiều sâu kỹ thuật ở một vài chỗ. (50% - 75%)  
   Hiểu biết toàn diện về Docker, bao gồm việc phân tích chi tiết các khái niệm quan trọng như image, container, Docker Compose, và orchestrator. (100%)
2. **Tổng quan dự án và bối cảnh (1.0 điểm)**  
   Không có hoặc giải thích mơ hồ. (0%)  
   Mô tả cơ bản nhưng chưa rõ ràng về các dịch vụ hoặc vai trò của Docker. (25%)  
   Mô tả dự án, có đề cập nhưng thiếu chi tiết về các dịch vụ và tương tác của chúng. (50% - 75%)  
   Giải thích rõ ràng, mô tả đầy đủ bối cảnh dự án, lý do sử dụng Docker và các dịch vụ liên quan. (100%)
3. **Kiến trúc và triển khai (1.0 điểm)**  
   Không có kiến trúc hoặc triển khai sai. (0%)  
   Kiến trúc cơ bản, có định nghĩa vài dịch vụ nhưng thiếu các cấu hình quan trọng. (25%)  
   Kiến trúc và triển khai phần nào đầy đủ, nhưng thiếu chi tiết về Dockerfile hoặc kết nối mạng giữa các container. (50% - 75%)  
   Kiến trúc đầy đủ và chính xác, các dịch vụ được định nghĩa rõ ràng, với Dockerfile, Docker Compose và cấu hình mạng đầy đủ. (100%)
4. **Trình bày báo cáo (2.0 điểm)**  
   Không có hoặc báo cáo thiếu sót nặng. (0%)  
   Báo cáo cơ bản, thiếu cấu trúc hoặc có lỗi định dạng. (25%)  
   Hoàn thiện phần lớn nhưng còn lỗi định dạng hoặc thiếu chi tiết ở một số phần. (50% - 75%)  
   Báo cáo được trình bày chuyên nghiệp, rõ ràng, theo cấu trúc yêu cầu, và có logic xuyên suốt. (100%)

**Trình diễn (4.0 điểm)**  
Không có video trình diễn, không chấm phần này.

**Trình diễn hoạt động (Cấp độ 1)**  
Không có hoặc triển khai Docker chưa đầy đủ. (0%)  
Demo chạy nhưng các dịch vụ chưa kết nối hoặc hoạt động hoàn chỉnh. (25%)  
Demo phần lớn hoạt động, nhưng có lỗi nhỏ về tương tác hoặc quy mô. (50% - 75%)  
Ứng dụng Docker hoàn chỉnh với các dịch vụ (front-end, back-end, cơ sở dữ liệu) hoạt động và kết nối giữa chúng chính xác. (100%)

**Tính năng nâng cao (Cấp độ 2)**  
Không có tính năng nâng cao như mở rộng, cân bằng tải. (0%)  
Có thực hiện cơ bản nhưng thiếu tích hợp với Redis hoặc RabbitMQ. (25%)  
Thực hiện tính năng nâng cao một phần, nhưng chưa hoàn chỉnh. (50% - 75%)  
Thực hiện đầy đủ các tính năng như mở rộng, cân bằng tải, phân tách dịch vụ với Redis hoặc RabbitMQ. (100%)

**Tính năng nâng cao (Cấp độ 3)**  
Không sử dụng các công cụ orchestrator. (0%)  
Triển khai đầy đủ trên Docker Swarm, Kubernetes, hoặc ECS, với khả năng mở rộng và quản lý dịch vụ. (100%)

*Các lỗi thường gặp khi viết báo cáo (danh sách không đầy đủ):*

* Trang bìa sai thông tin, dùng logo chất lượng kém, thông tin giảng viên hoặc thành viên nhóm sai.
* Thiếu các phần bắt buộc như khảo sát lý thuyết, phân tích yêu cầu, thiết kế hệ thống, kết quả và so sánh.
* Lỗi chính tả, font chữ không đồng nhất, khoảng cách dòng không đúng, quá lạm dụng dấu đầu dòng.
* Thiếu hình ảnh, biểu đồ để minh họa.
* Sử dụng quá nhiều mã nguồn mà thiếu mô tả, khiến khó hiểu.

**Trừ điểm có thể xảy ra trong các trường hợp sau:**

* Chất lượng video hoặc âm thanh kém, khó nghe hoặc xem.
* Phân chia công việc không đều giữa các thành viên.
* Nộp muộn hoặc qua email cá nhân.
* Không đúng định dạng hoặc thiếu thông tin cần thiết.