MEMBUAT APLIKASI TO-DO LIST DENGAN BERBASIS DOCKER

PROPOSAL PROYEK SISTEM OPERASI

Dosen Pengampu : Ferdi Chahyadi, S.Kom., M.Cs

Disusun Untuk Memenuhi Tugas Proyek

Mata Kuliah Sistem Operasi

DISUSUN OLEH :

1. **ADHIE MULIA SEMBIRING (2401020165)**
2. **MISYE KHALINA APRILIA MARBUN (2401020147)**
3. **RENDA KURNIA MANIK (2301020003)**
4. **YUDHA EKAPUTRA (2301020019)**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK DAN TEKNOLOGI KEMARITIMAN

UNIVERSITAS MARITIM RAJA ALI HAJI

2025

DAFTAR ISI

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam pengembangan aplikasi modern, sering terjadi ketidaksesuaian environment antara komputer developer dengan server production. Masalah ini dikenal sebagai "works on my machine" problem. Docker hadir sebagai solusi dengan teknologi containerization yang memungkinkan aplikasi berjalan dalam environment yang terisolasi dan konsisten.

Containerization menjadi teknologi yang semakin populer dalam industri software development karena efisiensi resource dan kemudahan deployment. Melalui proyek ini, kami ingin mempelajari dasar-dasar Docker, mulai dari pembuatan image, menjalankan container, hingga mengatur pembatasan resource menggunakan cgroups.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam proyek ini adalah:

1. Bagaimana mengimplementasikan containerization menggunakan Docker untuk aplikasi Python Flask?
2. Bagaimana cara menjalankan multiple container yang dapat saling berkomunikasi?
3. Bagaimana menerapkan pembatasan resource CPU dan memory menggunakan cgroups?
4. Bagaimana membangun aplikasi sederhana yang dapat mendemonstrasikan konsep containerization?

1.3 Tujuan Proyek

Adapun tujuan dari pelaksanaan proyek ini adalah:

1. Memahami konsep dan implementasi containerization dengan Docker
2. Membuat Dockerfile untuk aplikasi Python Flask
3. Menjalankan dan mengelola multiple container dengan Docker Compose
4. Menerapkan resource limits menggunakan cgroups untuk CPU dan memory
5. Membangun aplikasi to-do list sebagai studi kasus implementasi

1.4 Manfaat Proyek

Manfaat yang diharapkan dari proyek ini adalah:

1. Pengalaman praktis dalam menggunakan teknologi containerization
2. Pemahaman tentang virtualisasi ringan dan perbedaannya dengan virtual machine
3. Keterampilan dalam deployment aplikasi menggunakan Docker
4. Persiapan untuk karir di bidang DevOps dan software development

BAB 2

TINJAUAN PUSTAKA

2.1 Containerization

Containerization adalah metode virtualisasi ringan yang mengemas aplikasi beserta semua dependensinya dalam container yang terisolasi. Berbeda dengan virtual machine yang membutuhkan guest OS, container berbagi kernel dengan host system sehingga lebih efisien dalam penggunaan resource.

2.2 Docker

Docker adalah platform open-source untuk mengembangkan, mengirim, dan menjalankan aplikasi dalam container. Docker menyediakan tools untuk membuat image, menjalankan container, dan mengelola container lifecycle.

2.3 Cgroups (Control Groups)

Cgroups adalah fitur kernel Linux yang membatasi, mengaccounting, dan mengisolasi resource usage (CPU, memory, disk I/O, network) dari sekumpulan proses. Docker menggunakan cgroups untuk mengimplementasikan resource limits pada container.

2.4 Docker Compose

Docker Compose adalah tool untuk mendefinisikan dan menjalankan multi-container Docker applications. Dengan Compose, kita dapat mengkonfigurasi semua service aplikasi dalam satu file YAML.

BAB 3

METODOLOGI

3.1 Rancangan Sistem

Sistem yang akan dibangun terdiri dari dua container utama:

3.1.1 Todo-App Container

- Framework: Python Flask
- Port: 5000
- Fungsi: Manajemen task (Create, Read, Update, Delete)
- Endpoints: /todos, /add, /complete

3.1.2 Stats-App Container

- Framework: Python Flask
- Port: 5001
- Fungsi: Analisis dan statistik tasks
- Endpoints: /stats, /productivity

3.2 Teknologi yang Digunakan

1. Docker & Docker Compose
2. Python 3.9 dengan Flask
3. cgroups untuk resource limiting
4. Docker Hub untuk image repository
5. GitHub untuk version control dan source code management

3.3 Implementasi Resource Limits

Pembatasan resource akan diimplementasikan menggunakan:

1) CPU limits: 0.5 core untuk todo-app, 0.3 core untuk stats-app
2) Memory limits: 128MB untuk todo-app, 64MB untuk stats-app

3.4 Tahapan Pengerjaan

1) Environment setup dan instalasi Docker
2) Pengembangan aplikasi Python Flask
3) Pembuatan Dockerfile untuk masing-masing aplikasi
4) Konfigurasi Docker Compose dengan resource limits
5) Testing dan validasi sistem
6) Dokumentasi dan penyusunan laporan
7) Upload source code ke GitHub repository

BAB 4

HASIL YANG DIHARAPKAN

4.1 Dockerfile (Output A)

Akan dihasilkan dua Dockerfile yang berisi:

1) Konfigurasi base image Python 3.9-slim
2) Instalasi dependencies yang diperlukan
3) Konfigurasi environment aplikasi
4) Expose port dan startup comman

4.2 Container Berjalan (Output B)

Akan dijalankan dua container yang:

1) Berjalan secara bersamaan pada port 5000 dan 5001
2) Dapat saling berkomunikasi melalui Docker network
3) Mengimplementasikan fungsionalitas to-do list dan statistics

4.3 Resource Limit (Output C)

Akan diimplementasikan pembatasan resource:

1) CPU limits menggunakan cgroups
2) Memory limits menggunakan cgroups
3) Monitoring resource usage dengan docker stats

4.4 Laporan Lengkap (Output D)

Akan disusun laporan yang berisi:

1) Dokumentasi lengkap implementasi
2) Analisis hasil dan pembelajaran
3) Screenshot bukti implementasi
4) Source code aplikasi yang diupload ke GitHub

BAB 5

JADWAL PELAKSANAAN

5.1 Timeline Pengerjaan

Minggu 11: Penyusunan Proposal

- Studi literatur dan penelitian pendahuluan
- Penyusunan proposal proyek
- Perancangan arsitektur sistem

Minggu 12: Instalasi dan Setup Environment

- Instalasi Docker Desktop
- Setup development environment
- Verifikasi instalasi dan testing dasar
- Setup GitHub repository

Minggu 13: Pengembangan Aplikasi dan Dockerfile

- Pengembangan aplikasi todo-app dan stats-app
- Pembuatan Dockerfile untuk masing-masing aplikasi
- Build dan test Docker images
- Push initial code ke GitHub

Minggu 14: Implementasi Resource Limits

- Konfigurasi Docker Compose
- Implementasi resource limits dengan cgroups
- Testing multi-container setup
- Monitoring dan optimisasi
- Update GitHub repository dengan source code lengkap

Minggu 15: Finalisasi dan Demo

- Penyusunan laporan akhir
- Preparasi presentasi dan demo
- Testing final dan bug fixing
- Final push ke GitHub dan pengumpulan hasil proyek

BAB 6

PENUTUP

6.1 Kesimpulan

Proyek implementasi containerization dengan Docker ini akan memberikan pemahaman praktis tentang teknologi container dan resource management dalam sistem operasi. Dengan menyelesaikan proyek ini, kami berharap dapat menguasai dasar-dasar Docker yang sangat berguna dalam pengembangan software modern.

6.2 Harapan

Kami berharap proyek ini tidak hanya memenuhi requirements akademis, tetapi juga memberikan fondasi yang kuat untuk pengembangan karir di bidang IT, khususnya dalam area DevOps dan cloud computing. Source code proyek ini akan diupload ke GitHub sebagai portfolio dan referensi pembelajaran.