Model Arsitektur Lingkungan Pengembangan Portabel Menggunakan Docker Compose untuk Aplikasi Web Monolitik (Studi Kasus: Sistem E-Voting e-Pilketos)

Akhmad Syaifudin, Teknik Informatika, Universitas Islam Sultan Agung
akhmadsyaifudin505@gmail.com

Abstrak

Pengembangan dan pemeliharaan aplikasi web monolitik, seperti sistem berbasis PHP-MySQL, sering kali dihadapkan pada tantangan signifikan terkait konsistensi lingkungan pengembangan. Perbedaan konfigurasi antar mesin pengembang sering kali menyebabkan masalah "*it works on my machine*" yang menghambat produktivitas dan kolaborasi. Penelitian ini mengusulkan sebuah model arsitektur untuk mengatasi masalah tersebut melalui penerapan teknologi kontainerisasi. Dengan menggunakan Docker dan Docker Compose, kami merancang sebuah lingkungan pengembangan yang terisolasi dan portabel untuk aplikasi studi kasus, e-Pilketos. Model ini memisahkan layanan aplikasi (Apache/PHP) dan database (MySQL) ke dalam kontainer yang terpisah namun terhubung dalam satu jaringan virtual. Implementasi model ini dilakukan melalui definisi Dockerfile untuk image aplikasi dan docker-compose.yml untuk orkestrasi layanan. Hasil pengujian fungsional menunjukkan bahwa aplikasi berjalan sepenuhnya sesuai harapan di dalam lingkungan kontainer. Kontribusi utama dari penelitian ini adalah sebuah model yang dapat direplikasi untuk standarisasi lingkungan pengembangan aplikasi web monolitik, yang secara drastis mengurangi waktu penyiapan (setup), menjamin konsistensi, dan menyederhanakan proses deployment.

Kata Kunci: Kontainerisasi, Docker, Docker Compose, Arsitektur Monolitik, PHP, Lingkungan Pengembangan, Portabilitas.

I. PENDAHULUAN

Perkembangan aplikasi web telah menjadi tulang punggung transformasi digital di berbagai sektor. Meskipun tren saat ini mengarah pada arsitektur microservices, banyak sistem yang masih beroperasi atau dikembangkan dengan arsitektur monolitik, terutama yang dibangun dengan tumpukan teknologi seperti LAMP (Linux, Apache, MySQL, PHP). Salah satu tantangan persisten dalam pengembangan aplikasi monolitik adalah manajemen lingkungan pengembangan. Proses penyiapan manual yang melibatkan instalasi server web, database, runtime bahasa, dan berbagai ekstensi sering kali kompleks, memakan waktu, dan rawan kesalahan (*error-prone*). Lebih jauh lagi, perbedaan kecil dalam versi perangkat lunak atau konfigurasi sistem operasi antar pengembang dapat menimbulkan masalah dependensi yang dikenal sebagai *dependency hell*, yang berujung pada fenomena "di komputer saya jalan" (*it works on my machine*).

Teknologi kontainerisasi, yang dipopulerkan oleh Docker, muncul sebagai solusi state-of-the-art untuk mengatasi tantangan ini. Docker memungkinkan pengembang untuk membungkus aplikasi beserta seluruh dependensinya ke dalam sebuah unit standar yang ringan dan portabel yang disebut kontainer (Ekaputra & Affandi, 2023).

Penelitian ini bertujuan untuk merancang, mengimplementasikan, dan mengevaluasi sebuah model arsitektur lingkungan pengembangan yang portabel untuk aplikasi web monolitik. Sebagai studi kasus, kami menggunakan "e-Pilketos", sebuah sistem e-voting berbasis PHP dan MySQL. Kontribusi dari makalah ini adalah menyajikan sebuah templat (template) yang praktis dan dapat direplikasi untuk modernisasi alur kerja pengembangan aplikasi sejenis, sehingga meningkatkan efisiensi, kolaborasi tim, dan keandalan proses deployment.

II. TINJAUAN PUSTAKA

2.1 Arsitektur Monolitik dan Tantangannya

Aplikasi monolitik adalah aplikasi yang dibangun sebagai satu unit tunggal yang utuh. Seluruh komponen, mulai dari antarmuka pengguna, logika bisnis, hingga lapisan akses data, saling terikat erat. Meskipun sederhana untuk dikembangkan pada tahap awal, arsitektur ini menjadi sulit untuk dikelola dan diskalakan seiring bertambahnya kompleksitas. Salah satu tantangan utamanya adalah lingkungan deployment yang kaku dan sulit direplikasi (Pradana et al., 2024).

2.2 Kontainerisasi dengan Docker dan Docker Compose

Docker adalah platform yang memungkinkan pembuatan dan pengelolaan kontainer. Dengan mendefinisikan lingkungan dalam sebuah Dockerfile, Docker menjamin bahwa aplikasi akan selalu berjalan dalam kondisi yang sama, terlepas dari mesin host yang mendasarinya. Untuk aplikasi yang memerlukan beberapa komponen seperti aplikasi web dan database, Docker Compose hadir sebagai alat orkestrasi. Docker Compose menggunakan file YAML untuk mendefinisikan dan menjalankan aplikasi multi-kontainer, menyederhanakan pengelolaan jaringan dan volume data antar layanan (Bik, 2017)

2.3 Sistem E-Voting Berbasis Web

Sistem e-voting merupakan salah satu implementasi teknologi informasi dalam proses demokrasi. Banyak penelitian telah dilakukan untuk merancang sistem e-voting yang aman dan andal untuk lingkup organisasi, termasuk pemilihan di lingkungan akademis (Mukti & Airlangga, 2025). Sistem seperti e-Pilketos menjadi studi kasus yang ideal karena mewakili aplikasi web monolitik pada umumnya yang bergantung pada tumpukan teknologi spesifik .

III. METODOLOGI PENELITIAN

Metodologi yang digunakan dalam penelitian ini terdiri dari tiga tahap utama: desain model, implementasi teknis, dan pengujian.

3.1 Desain Model Arsitektur

Model yang diusulkan memisahkan aplikasi e-Pilketos menjadi dua layanan utama yang dikontainerisasi:

- Layanan `app`: Berisi server web Apache dan runtime PHP 8.1. Kontainer ini bertanggung jawab untuk mengeksekusi kode aplikasi e-Pilketos.
- Layanan `db`: Berisi server database MySQL 8.0. Kontainer ini bertanggung jawab untuk penyimpanan data persisten.

Kedua layanan ini didefinisikan untuk berjalan di dalam jaringan virtual yang sama yang dibuat oleh Docker Compose, memungkinkan komunikasi antara layanan `app` dan `db` melalui nama layanan sebagai hostname.

3.2 Implementasi Teknis

Implementasi model ini diterjemahkan ke dalam dua artefak utama:

- `Dockerfile`: Sebuah file definisi untuk membangun image layanan `app`, dimulai dari image dasar `php:8.1-apache` dan menambahkan ekstensi `mysqli`.
- `docker-compose.yml`: File orkestrasi yang mendefinisikan kedua layanan (`app` dan `db`), mengatur pemetaan port, konfigurasi volume (untuk persistensi kode dan data), dan variabel lingkungan untuk koneksi database.

3.3 Metode Pengujian

Pengujian dilakukan untuk memvalidasi fungsionalitas dan portabilitas model:

- Pengujian Fungsional: Melakukan pengujian black-box pada seluruh alur kerja aplikasi, termasuk registrasi, login admin dan siswa, proses voting, dan penampilan hasil.

Tujuannya adalah untuk memastikan aplikasi berfungsi 100% benar di dalam lingkungan kontainer.

- Pengujian Reproducibility: Mengkloning repositori proyek pada mesin yang berbeda (dengan Docker terinstal) dan menjalankan lingkungan menggunakan perintah `docker-compose up` untuk memverifikasi bahwa lingkungan dapat direplikasi secara identik tanpa penyiapan manual tambahan.

IV. HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi dan Pengujian

Implementasi model yang diusulkan berhasil dieksekusi, menghasilkan sebuah lingkungan aplikasi yang fungsional dan portabel. Seluruh artefak teknis, termasuk kode sumber dan file konfigurasi Docker yang digunakan untuk menghasilkan temuan ini, tersedia secara publik untuk verifikasi di repositori GitHub: https://github.com/SYFDNNN/e_Pilketos.git

Perintah `docker-compose up -d --build` mampu membangun image aplikasi dan menjalankan seluruh layanan kontainer tanpa error. Hasil pengujian fungsional menunjukkan bahwa seluruh alur kerja aplikasi—mencakup otentikasi pengguna, manajemen data oleh admin, dan proses voting oleh siswa—beroperasi sesuai dengan spesifikasi fungsionalnya di dalam lingkungan terkontainerisasi. Lebih lanjut, pengujian reproduktifitas yang dilakukan dengan mengkloning repositori pada sistem host yang berbeda mengkonfirmasi bahwa lingkungan pengembangan dapat direplikasi secara identik dan konsisten, yang secara langsung membuktikan portabilitas dari model yang dirancang.

4.2 Pembahasan

Penerapan model kontainerisasi pada aplikasi e-Pilketos memberikan beberapa keuntungan signifikan yang menjawab masalah penelitian.

- Peningkatan Efisiensi dan Reduksi Waktu Setup: Dibandingkan dengan proses manual yang bisa memakan waktu berjam-jam, penyiapan lingkungan dengan model ini hanya memerlukan satu perintah dan selesai dalam beberapa menit. Ini secara drastis meningkatkan efisiensi bagi pengembang.
- Portabilitas dan Konsistensi Lingkungan: Dengan mendefinisikan seluruh infrastruktur sebagai kode (*Infrastructure as Code*), masalah "*it works on my machine*" dapat dieliminasi. Setiap pengembang di dalam tim dijamin bekerja pada lingkungan yang identik, yang didefinisikan secara ketat oleh `Dockerfile` dan `docker-compose.yml`.
- Isolasi Proyek: Setiap tumpukan teknologi proyek (Apache, PHP, MySQL) berjalan dalam kontainer terisolasi, sehingga tidak akan menimbulkan konflik dengan proyek lain atau perangkat lunak lain yang terinstal di mesin host.

Meskipun demikian, model ini memiliki keterbatasan. Konfigurasi yang disajikan dioptimalkan untuk lingkungan pengembangan. Untuk lingkungan produksi, diperlukan pertimbangan lebih lanjut seperti pengerasan keamanan (security hardening), manajemen log yang terpusat, dan strategi backup database yang lebih kuat.

V. KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan sebuah model arsitektur lingkungan pengembangan yang portabel untuk aplikasi web monolitik menggunakan Docker dan Docker Compose. Melalui studi kasus aplikasi e-Pilketos, model ini terbukti efektif dalam menyederhanakan proses penyiapan, menjamin konsistensi lingkungan antar pengembang, dan

meningkatkan efisiensi alur kerja secara keseluruhan. Model ini dapat diadopsi sebagai praktik standar untuk modernisasi pengembangan aplikasi berbasis PHP-MySQL atau tumpukan teknologi sejenis.

Untuk penelitian di masa depan, disarankan untuk melakukan analisis kinerja kuantitatif yang membandingkan performa aplikasi di dalam kontainer dengan instalasi bare-metal, serta mengembangkan model ini menjadi arsitektur yang siap produksi menggunakan alat orkestrasi yang lebih canggih seperti Kubernetes.

DAFTAR PUSTAKA

Bik, M. F. R. (2017). Implementasi Docker untuk pengelolaan banyak aplikasi web (Studi kasus: Jurusan Teknik Informatika UNESA). *Jurnal Manajemen Informatika*, *7*(2).

Ekaputra, A. R., & Affandi, A. S. (2023). Pemanfaatan layanan cloud computing dan docker container untuk meningkatkan kinerja aplikasi web. *Journal of Information System and Application Development*, *1*(2), 138–147.

Mukti, R. I., & Airlangga, P. (2025). Website-Based Electronic Voting (E-Voting) System in the Bem Unwaha General Election. *NEWTON: Networking and Information Technology*, *4*(3), 141–147.

Pradana, S. A., Andika, R., Wibowo, M. A. P., Hutagalung, M. R. S., Sipahutar, H. K., & Rizal, C. (2024). Perancangan Sistem Informasi E-Voting Berbasis Web Untuk Pemilihan Ketua Himpunan Di UIN Sumatera Utara Medan. *Jurnal Komputer Teknologi Informasi Dan Sistem Informasi (JUKTISI)*, *3*(2), 782–793.