BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan implementasi dan pengujian yang telah dilakukan, dapat disimpulkan bahwa penerapan arsitektur *cloud-native* berbasis Kubernetes, didukung oleh Ansible untuk otomasi dan Prometheus-Grafana untuk monitoring, berhasil meningkatkan efisiensi dan pengelolaan laboratorium komputer. Konsolidasi PC dengan spesifikasi heterogen menjadi satu klaster Kubernetes yang terpadu memungkinkan pemanfaatan sumber daya komputasi secara optimal dan dinamis. Penggunaan Ansible secara efektif mengotomatisasi proses instalasi, konfigurasi, dan penambahan node baru, secara signifikan mengurangi waktu dan potensi kesalahan manusia.

Selain itu, sistem monitoring terpusat yang dibangun dengan Prometheus dan Grafana menyediakan visibilitas *real-time* yang komprehensif terhadap kinerja klaster, memungkinkan deteksi dini masalah, optimasi kinerja, dan perencanaan kapasitas yang lebih baik. Arsitektur ini juga menjamin konsistensi lingkungan praktik bagi mahasiswa melalui penggunaan kontainer Docker, mengatasi masalah umum dimana inkonsistensi hasil antara mesin satu dengan mesin lain. Secara keseluruhan, penelitian ini membuktikan bahwa teknologi *cloud-native* dapat diterapkan secara efektif di lingkungan *on-premise* untuk mengatasi tantangan manajemen sumber daya dan lingkungan praktik di laboratorium komputer.

5.2 Saran

Berdasarkan hasil implementasi sistem yang telah dilakukan, terdapat beberapa hal yang dapat dikembangkan lebih lanjut pada penelitian berikutnya agar sistem menjadi lebih optimal, efisien, dan aman. Adapun saran-saran tersebut adalah sebagai berikut:

- Penerapan *Persistent Storage*: Penelitian selanjutnya disarankan untuk mengeksplorasi dan mengimplementasikan solusi *persistent storage* seperti Ceph atau GlusterFS di dalam klaster. Penerapan ini akan

memungkinkan aplikasi *stateful* untuk menyimpan data secara persisten dan dapat diakses oleh seluruh *pod* dalam klaster. Fitur ini sangat penting untuk mendukung aplikasi basis data atau layanan lain yang membutuhkan penyimpanan data jangka panjang.

- Integrasi *CI/CD Pipeline*: Integrasi antara klaster Kubernetes dengan *Continuous Integration/Continuous Delivery (CI/CD)* pipeline, seperti Jenkins atau GitLab CI/CD, disarankan untuk dilakukan agar proses pembangunan, pengujian, dan *deployment* aplikasi dapat berlangsung secara otomatis. Dengan adanya integrasi ini, pengembang (misalnya mahasiswa atau staf pengajar) dapat mempercepat siklus pengembangan dan mengurangi potensi kesalahan manusia dalam proses penerapan aplikasi ke klaster.
- Implementasi *Security Best Practices*: Aspek keamanan merupakan hal yang krusial dalam infrastruktur *cloud-native*. Penelitian selanjutnya dapat berfokus pada penerapan praktik keamanan yang lebih mendalam, seperti konfigurasi *Network Policy* untuk mengisolasi lalu lintas antar *pod*, penggunaan sertifikat TLS pada seluruh layanan klaster, serta integrasi dengan solusi *container image scanning* guna mendeteksi kerentanan pada *image Docker*.
- Eksplorasi Fitur *Auto-Scaling*: Meskipun klaster telah menunjukkan kemampuan skalabilitas secara manual, penelitian berikutnya disarankan untuk mengeksplorasi fitur *Horizontal Pod Autoscaler (HPA)* dan *Cluster Autoscaler (CA)*. Penerapan fitur ini akan memungkinkan sistem untuk menyesuaikan jumlah *pod* dan *node* secara otomatis berdasarkan kebutuhan sumber daya, sehingga pemanfaatan klaster menjadi lebih dinamis dan efisien.
- Pemanfaatan *Cache Server* Lokal (Apt-Cacher NG): Untuk menghemat penggunaan bandwidth dan mempercepat proses instalasi perangkat lunak pada setiap *node*, disarankan untuk mengimplementasikan *Apt-Cacher NG* sebagai *local cache server*. Dengan adanya server cache lokal ini, setiap permintaan pembaruan

atau instalasi paket akan diarahkan terlebih dahulu ke cache server. Paket yang sudah diunduh akan disimpan sehingga dapat digunakan kembali oleh *node* lain tanpa perlu mengunduh ulang dari internet. Selain itu, mekanisme ini juga dapat berfungsi sebagai kontrol untuk memastikan bahwa hanya paket yang telah disetujui atau tersedia di cache server yang dapat diinstal, sehingga konsistensi lingkungan praktik dapat terjaga

- Evaluasi Kinerja dan Efisiensi Sistem: Selain pengembangan dari sisi implementasi, penelitian selanjutnya juga disarankan untuk melakukan evaluasi kinerja dan efisiensi sistem. Evaluasi ini dapat dilakukan dengan membandingkan berbagai konfigurasi klaster, metode orkestrasi, atau strategi penjadwalan *pod* guna menentukan kombinasi yang paling optimal bagi kebutuhan pembelajaran maupun pengembangan aplikasi di lingkungan tersebut.