BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang didapat dari penelitian ini adalah bahwa pembangunan layanan cloud computing di Google Cloud Platform (*serverless dan IaaS*) menggunakan Terraform sebagai tool *Infrastructure as Code (IaC)* dan penggunaan *container* untuk aplikasi yang digunakan dalam penelitian dan diikuti dengan integrasi *CI/CD Pipeline* menggunakan metode pembangunan perangkat lunak DevOps dapat meningkatkan efisiensi, kecepatan, konsistensi dan efektivitas pembangunan layanan cloud computing maupun aplikasi.

5.2 Saran

Jika penelitian ini digunakan dan dikembangkan, sebagai saran guna memenuhi prinsip *least of privilege* (pemberian akses maupun izin sekecil mungkin) cara yang bisa dilakukan adalah (jika menggunakan google cloud platform) membuat *service account* dengan role yang lebih spesifik berdasarkan spesifikasi kebutuhan layanan yang digunakan.

Selain itu, saran yang bisa di gunakan ketika dalam proses pembangunan aplikasi maupun layanan cloud menggunakan *Infrastructure as Code* dan *CI/CD Pipeline* adalah penggunaan *environment* terpisah, misal untuk *environment development* untuk mengerjakan aplikasi berikut layanan cloud untuk aplikasi yang masih dalam tahap pengembangan, *environment staging* dan *environment production* sebagai *source code* utama dari aplikasi yang di deliver (dikirimkan) ke pengguna berikut konfigurasi layanan cloud untuk aplikasi yang di deliver ke pengguna dengan konfigurasi *trigger CI/CD Pipeline*, file konfigurasi *Infrastructure as Code* (misal terraform) dan file konfigurasi *CI/CD Pipeline* (misal cloudbuild.yaml) yang ikut disesuaikan dengan kebutuhan. Karena pada penelitian ini, peneliti menggunakan satu *environment* langsung yaitu *production* karena proses push dilakukan pada *branch* master pada layanan *Cloud Source Repositories* (milik Google Cloud Platform, namun sudah tidak tersedia untuk pengguna baru https://cloud.google.com/source-repositories/docs) dan untuk mengimplementasikan saran yang disebut memerlukan keterampilan lebih dalam penggunaan layanan *VCS (Version Control System)* seperti *github, gitlab* dan lainnya.

Saran lainnya guna mencegah hal yang tidak diinginkan ketika menangani informasi sensitif seperti informasi database ataupun informasi sensitif lainnya ketika membangun aplikasi dengan mengimplementasikan *Infrastructure as Code* dan *CI/CD Pipeline* saran yang perlu untuk diimplementasikan nantinya adalah penggunaan layanan Secret Manager (pada Google Cloud) atau layanan manajemen rahasia lainnya (*secret management*) agar tidak menampilkan informasi sensitif secara langsung didalam file aplikasi ataupun file konfigurasi *Infrastructure as Code* (misal Terraform) maupun di *version control system* seperti *github, gitlab* atau yang lainnya guna meningkatkan keamanan dalam proses *CI/CD Pipeline* itu sendiri.

Lalu penggunaan Cloud SQL Proxy dapat menjadi saran sekaligus praktik yang lebih baik (jika menggunakan database SQL, dengan layanan Cloud SQL di Google Cloud Platform) untuk memberikan akses IP address yang lebih spesifik untuk menghubungkan Database Instance dengan aplikasi yang akan digunakan nantinya agar koneksi aplikasi dengan database menjadi lebih aman karena pada penelitian ini, peneliti secara langsung menghubungkan database instance didalam file aplikasi tanpa menggunakan SQL Proxy.

Sebagai saran juga, konfigurasi monitoring pada penelitian ini dapat dilakukan penyesuaian untuk komponen apa saja yang akan dilakukan monitoring serta dapat dikostumisasi visualisasi dashboardnya dengan layanan "monitoring infrastruktur" pihak ketiga seperti *Grafana* guna mendapatkan visualisasi yang lebih baik. Karena pada penelitian ini konfigurasi monitoring berupa dashboard yang dibuat di console Google Cloud Platform bersifat sederhana dan hanya untuk keperluan demonstrasi saja.

Guna membangun layanan *cloud computing* yang bersifat HA (*High Availability*) sekaligus mitigasi jika terjadi sesuatu yang tidak diinginkan terdapat beberapa saran yang bisa diterapkan antara lain :

1. Memanfaatkan fitur *replication* (replikasi atau membuat salinan) pada Cloud SQL Instance. Salah satu contohnya adalah *Cross-region read replicas* (membuat salinan baca didalam *region* yang berbeda dari lokasi SQL instance berada) guna meningkatkan kecepatan proses baca database yang ada didalam cloud SQL Instance serta memberikan kemampuan pemulihan dari bencana (*disaster recovery*) untuk menangani masalah kegagalan di setiap *region*. Terdapat dokumentasi terkait bagaimana pemanfaatan fitur *replication* untuk Cloud SQL yang dapat diakses pada tautan https://cloud.google.com/sql/docs/mysql/replication dan *disaster recovery* yang dapat

diakses pada tautan https://cloud.google.com/sql/docs/mysql/intro-to-cloud-sql-disaster-recovery

2. Menggunakan mode *dual-region* atau *multi-region* dalam membangun layanan Cloud Storage Bucket untuk mendukung *Data Availability and Durability* agar setiap objek dalam *bucket* yang tersimpan akan dibuat redundansinya setidaknya pada dua daerah dengan geografis yang berbeda. Terdapat dokumentasi terkait *Data Availability and Durability* untuk Cloud Storage Bucket yang dapat diakses pada tautan https://cloud.google.com/storage/docs/availability-durability
3. Menggunakan mode *multi-region* dalam membangun layanan Artifact Registry agar setiap *artifact* (misal docker images) tersedia pada dua *region* yang sama secara geografis.

Untuk dokumentasi lengkap yang berisi tentang cara membangun layanan Google Cloud Platform yang HA (*High Availability*) serta mitigasi jika terjadi sesuatu yang tidak diinginkan dapat diakses pada tautan https://cloud.google.com/architecture/ dan *Disaster Recovery* yang dapat diakses pada tautan https://cloud.google.com/architecture/dr-scenarios-planning-guide