Deployment Website Portofolio Otomatis:
Infrastructure as Code dengan Terraform, GitLab CI/CD, dan AWS

Fitra Maulana

D4 Teknologi Rekayasa Internet · Sekolah Vokasi · Universitas Gadjah Mada · 2026

fitramaulana@mail.ugm.ac.id · gitlab.com/maulanafitra · fitramaulana.my.id

Abstrak

Dokumen ini mendeskripsikan desain, implementasi, dan arsitektur sistem *deployment* otomatis untuk *website* portofolio yang dibangun di atas prinsip *Infrastructure as Code* (IaC). Sistem memanfaatkan Terraform untuk *provisioning resource cloud*, GitLab CI/CD untuk *pipeline build* dan *deployment* otomatis, serta layanan AWS (S3, CloudFront, ACM, DynamoDB) untuk *hosting* situs statis yang terdistribusi secara global. Hasilnya adalah alur *deployment* yang sepenuhnya otomatis, *reproducible*, dan efisien dari sisi biaya — hanya dengan satu perintah `git push` ke *branch main*, seluruh siklus *build* hingga update produksi selesai dalam waktu sekitar dua menit.

Kata Kunci: *Infrastructure as Code, Terraform, GitLab CI/CD, AWS S3, CloudFront, Next.js, DevOps, Static Site Hosting*

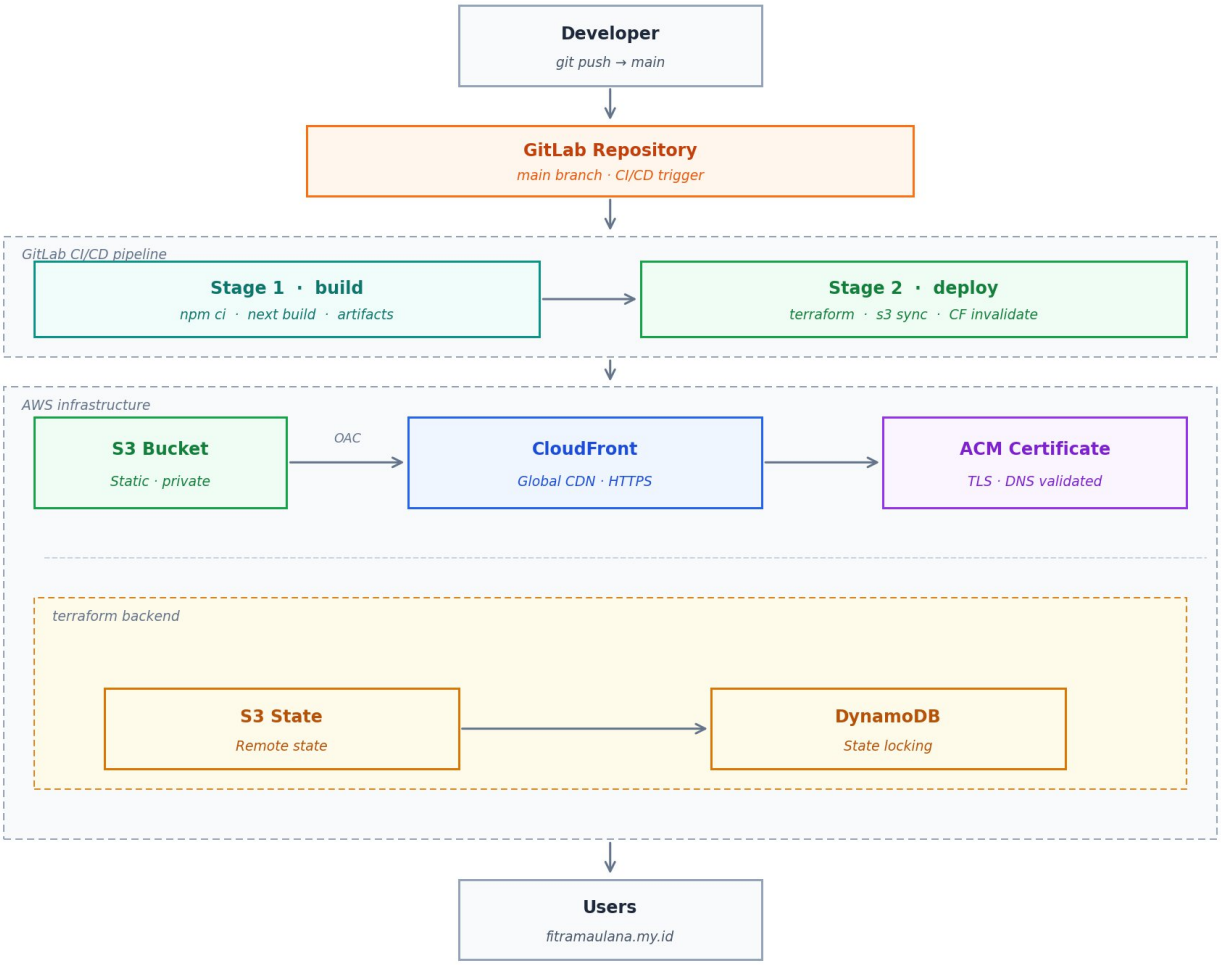
I. Pendahuluan

Rekayasa perangkat lunak modern semakin menuntut infrastruktur diperlakukan sebagai kode yang ter-versi dan dapat diaudit, bukan konfigurasi manual. *Infrastructure as Code* (IaC) memungkinkan tim untuk melakukan *provisioning*, modifikasi, dan penghapusan *resource cloud* secara deterministik, menghilangkan inkonsistensi lingkungan dan mengurangi risiko *deployment*.

Proyek ini menerapkan prinsip IaC untuk men-*deploy website* portofolio pribadi (fitramaulana.my.id), sekaligus dijadikan sebagai bukti nyata *(proof-of-work)* kemampuan *DevOps end-to-end*. Arsitektur ini membuktikan bahwa proyek personal sekalipun dapat memperoleh manfaat dari praktik otomasi tingkat produksi — mulai dari pengelolaan *remote state*, *state locking*, pengiriman konten via CDN, hingga *provisioning* sertifikat TLS secara otomatis.

II. Arsitektur Sistem

Arsitektur sistem terbagi menjadi dua layer utama: layer pipeline CI/CD yang dikelola oleh GitLab, dan layer infrastruktur *cloud* yang di-*provisioning* oleh Terraform di AWS. Diagram berikut menggambarkan alur lengkap dari `git push` hingga pengguna mengakses *website*.



Gambar 1. Arsitektur Sistem Infrastructure as Code — CI/CD Pipeline & AWS Infrastructure

A. Lapisan Aplikasi

Website portofolio dibangun menggunakan Next.js 16 (React 19, TypeScript) dengan Tailwind CSS v4. Situs diekspor sebagai *build* statis penuh (`output: 'export'`), menghasilkan direktori berisi aset HTML, CSS, dan JavaScript statis yang siap di-*host* pada *object storage* tanpa memerlukan *server*.

B. *Pipeline* CI/CD (GitLab)

Setiap *commit* yang di-*push* ke *branch main* memicu *pipeline* dua *stage* yang didefinisikan dalam file `.gitlab-ci.yml`. Berikut detail setiap *stage*:

Stage	Langkah	*Output*
`1 · build`	`npm ci → next build → next export`	Direktori `out/` (aset statis)
`2 · deploy`	`terraform init → terraform applyaws s3 sync out/ s3://bucketaws cloudfront create-invalidation`	*Update* produksi *live*

C. Infrastruktur AWS (Terraform)

Seluruh *resource* AWS didefinisikan dalam *file* konfigurasi Terraform HCL di direktori `infra/`. Komponen infrastruktur terdiri dari:

- **S3 Bucket:** *Object storage* privat untuk aset statis *website*. Akses publik diblokir penuh — konten hanya dapat diakses melalui CloudFront via Origin Access Control (OAC).
- **CloudFront Distribution:** CDN global dengan 200+ edge location. Dikonfigurasi dengan *redirect* HTTP ke HTTPS, *default* TTL 1 jam, dan kompresi diaktifkan. OAC membatasi akses S3 hanya dari CloudFront, menghilangkan celah akses langsung.
- **AWS Certificate Manager (ACM):** Sertifikat TLS/SSL untuk `fitramaulana.my.id` yang di-*provisioning* secara otomatis via DNS validation. Sertifikat dilampirkan pada distribusi CloudFront untuk HTTPS *end-to-end*.
- **Terraform Remote State:** File *state* disimpan di S3 bucket terpisah untuk ketahanan data dan kesiapan kolaborasi tim. Tabel DynamoDB menyediakan *state locking* untuk mencegah konflik terraform *apply* yang bersamaan — praktik terbaik bahkan untuk proyek individu.

III. Metrik dan Hasil

Berikut rangkuman metrik kinerja dan hasil implementasi sistem:

Metrik	Nilai	Keterangan
Waktu *deployment*	**~2 menit**	Dari `git push` hingga *live* di produksi
Jumlah *pipeline stage*	**2 *(build, deploy)***	*Sequential*, melewatkan *artifacts* antar *stage*
CloudFront TTL	**3600 detik**	Cache di-*invalidate* setiap *deployment*
HTTPS	**Wajib**	HTTP dialihkan ke HTTPS secara global
Akses publik S3	**Diblokir penuh**	Hanya dapat diakses via CloudFront (OAC)
Terraform *state*	**Remote (S3)**	DynamoDB *locking* diaktifkan
Biaya hosting	**~Rp0 / bulan**	Dalam batas AWS *Free Tier*
Infrastructure drift	**Nol**	Seluruh *resource* dideklarasikan dalam kode

IV. Stack Teknologi

Tabel berikut merangkum seluruh teknologi yang digunakan dalam proyek ini:

Layer	Teknologi	Fungsi
Frontend	Next.js 16 · TypeScript · Tailwind CSS v4	*Static site generation*
CI/CD	GitLab CI/CD · GitLab *Runner*	*Build* & *deployment* otomatis
IaC	Terraform (HCL)	*Provisioning cloud resource*
CDN & Hosting	AWS CloudFront · S3	*Hosting* statis global
TLS/HTTPS	AWS *Certificate Manager* (ACM)	Sertifikat HTTPS otomatis
State Backend	AWS S3 · DynamoDB	*Remote state & locking*
DNS	*Custom domain*	fitramaulana.my.id

V. Kesimpulan

Proyek ini mendemonstrasikan penerapan praktis *Infrastructure as Code* dan prinsip-prinsip *DevOps* dalam lingkungan produksi nyata. Dengan memperlakukan infrastruktur cloud sebagai kode yang ter-versi, sistem mencapai *reproducibility* penuh: menghancurkan dan membuat ulang seluruh *stack* infrastruktur hanya memerlukan satu perintah terraform *apply*.

Integrasi GitLab CI/CD memastikan setiap perubahan kode secara otomatis divalidasi, di-*build*, dan di-*deploy* — menghilangkan intervensi manual dan *human error* dari proses *deployment*. Postur keamanan diperkuat melalui akses S3 privat dengan OAC, HTTPS yang diwajibkan, dan permission IAM yang minimal.

Pengembangan selanjutnya dapat mencakup: integrasi Route 53 untuk otomasi DNS penuh, penambahan WAF *(Web Application Firewall)* untuk perlindungan DDoS, dan implementasi *staging environment* dengan *deployment* berbasis *branch*.

Source code: gitlab.com/maulanafitra/fitramaulana-portfolio · Website: fitramaulana.my.id