# PERANCANGAN INFRASTRUKTUR SERVER VCS (VERSION CONTROL SYSTEM) DENGAN GITLAB BERBASIS GIT

[1]Henni Endah Wahanani, [2]Wahyu S.J. Saputra, [3]Bari Hade Variant Wahono
[123]Program Studi Teknik Informatika Fakultas Ilmu Komputer
Universitas Pembangunan Nasional "Veteran" Jawa Timur
Email: [1]henniendah.if@upnjatim.ac.id, [2]wahyu.s.j.saputra.if@upnjatim.ac.id,
[3]barihade@gmail.com

**Abstrak.** *Penggunaan server VCS (Version Control System) dalam dunia pemrograman merupakan faktor penting dalam aktivitas dokumentasi dan pengubahan kode. Tidak digunakannya teknologi VCS dalam kegiatan perkuliahan programming yang semakin pesat ini termasuk hambatan dalam sektor infrstruktur dan server yang semakin bergantung dengan teknologi tersebut. Dengan demikian, diperlukan infrastruktur server VCS (Version Control System) yang dapat membantu mahasiswa sehingga dapat dimudahkan dalam melakukan aktivitas dokumentasi tugas kuliahnya. Penelitian dilakukan dengan merancang infrastruktur server VCS dengan menerapkan teknologi Gitlab yang berbasis Git. Peran teknologi Gitlab dalam infrastruktur server VCS ini merupakan salah satu bentuk layanan Git yang bersifat open source, sehingga layanan tersebut bebas untuk dipasang tanpa memikirkan biaya untuk lisensinya. Infrastruktur server VCS yang dihasilkan dari penelitian ini dapat menghasilkan fasilitas yang dapat melakukan dokumentasi dan version control terhadap file yang diinginkan dalam bentuk repository. Untuk hasil akhirnya, Gitlab berbasis Git dapat digunakan sebagai infrastruktur server VCS layanan alternatif sebagai pengganti layanan dokumentasi lama yang bersifat manual. Dimana, penggunaan layanan dokumentasi diharapkan dapat mendokumentasikan tugas mahasiswa yang bersifat aplikasi dan website.*

***Kata Kunci**: VCS (Version Control System), Git, Gitlab*

Beberapa teknologi dari pengembangan industri 4.0 yang paling berpengaruh, dalam dunia pemrograman salah satunya adalah teknologi VCS (*Version Control System*) dan Git. *Version control* adalah sebuah sistem yang mencatat setiap perubahan terhadap sebuah berkas atau kumpulan berkas sehingga pada suatu saat dapat kembali kepada salah satu versi dari berkas tersebut. Sebagai contoh dalam penelitian ini akan menggunakan kode sumber perangkat lunak sebagai berkas yang akan dilakukan *version controlling*, meskipun pada kenyataannya dapat melakukan ini pada hampir semua tipe berkas di komputer [1]. Version Control System (VCS) adalah sistem yang mengelola pengembangan objek yang berkembang. Dengan kata lain, ini adalah sistem yang merekam setiap perubahan yang dilakukan oleh pengembang perangkat lunak. Ada banyak kegunaan untuk VCS dalam pengembangan perangkat lunak yang membuat proses pengembangan lebih mudah dan lebih cepat [2].

Dalam proses pengembangan perangkat lunak, wajar bagi pengembang perangkat lunak untuk terus membuat perubahan dalam potongan kode dan file lain yang melibatkan penambahan dan penghapusan fitur. Disadari bahwa beberapa revisi akan dilakukan sebelum menghasilkan versi final. Sulit untuk mengelola dan mengatur kode dan file karena jumlah revisi bertambah besar karena sistem yang lebih besar dan kompleks. Karenanya, keberadaan VCS sangat membantu pengembang perangkat lunak untuk mempercepat dan menyederhanakan proses pengembangan [3].

Tanpa VCS, pengembang perangkat lunak tertarik untuk menyimpan berbagai duplikat kode di komputer mereka. Ini berisiko karena mudah untuk mengubah atau menghapus dokumen atau file dalam salinan kode yang salah, mungkin menyebabkan kehilangan pekerjaan. Sistem kontrol versi akan menangani masalah ini dengan mengelola semua versi kode. Adopsi VCS secara progresif diamanatkan untuk memberdayakan semua pengembang perangkat lunak yang bekerja pada proyek yang sama untuk bekerja bersama efektif menuju proyek *milestones*.

Dalam perkembangannya lahir banyak macam VCS salah satunya VCS yang berbasis

Git. VCS yang berbasis Git terhitung sangat populer, dan banyak dipakai di dunia jaringan komputer dan *system administrator*. Dalam implementasinya, *server* VCS ini dibangun untuk beragam tujuan dan fungsi [4]. VCS dengan Gitlab berbasis Git dimana Gitlab adalah sebuah manajer repositori Git berbasis web dengan fitur wiki dan pelacakan masalah menggunakan lisensi open source yang dikembangkan oleh GitLab Inc. Perangkat lunak ini di tulis oleh Dimitriy Zaporozhets dan Valery Sizov dari Ukraina, kode yang digunakan adalah Ruby. Kemudian beberapa bagian telah di tulis ulang di Go [5][6]. GitLab adalah sistem hosting kode *open source* untuk manajemen repositori. Ini memungkinkan untuk melacak masalah untuk repositori Git, melakukan tinjauan kode, dan membuat dokumentasi proyek tambahan pada halaman wiki — dengan kata lain, itu hampir sama dengan GitHub dan Bitbucket. Keunggulan unik GitLab adalah bahwa sebagai produk *open source*, dapat menginstal perangkat lunak di mana pun, tanpa membayar biaya lisensi; dan dipersilakan untuk memperluas perangkat lunak secara langsung, alih-alih dibatasi membuat *add ons* melalui API [7].

Kegiatan perkuliahan adalah kegiatan belajar mengajar yang berfungsi untuk mengajarkan ilmu kepada mahasiswa agar dapat diaplikasikan kepada masyarakat. Selain teori yang diberikan di kelas di butuhkan adanya modul sebagai buku panduan untuk melaksanakan kegiatan praktikum, sedangkan untuk menilai keberhasilan praktikum di akhir semester diadakan kegiatan *final project*.

Masalah yang terjadi adalah mengenai belum adanya media yang berfungsi sebagai media dokumentasi tugas mahasiswa tersebut yang berupa aplikasi dan *website*. Dengan adanya permasalahan tersebut, peneliti ingin memberikan pandangan mengenai perancangan infrastruktur *server* VCS sebagai media dokumentasi tugas mahasiswa tersebut. Dengan adanya *server* VCS, mahasiswa memiliki media untuk dokumentasi tugas yang berupa aplikasi dan *website*.

## I. Metodologi

Metode penelitian dibuat alur kerja yang sistematis sehingga hasil dan kesimpulan dapat disesuaikan dengan rancangan dan topologi. Berikut adalah penjelasan dan alur kerja penelitian yang dilakukan pada gambar 1.:

1. Tahap studi literatur dilakukan untuk mengumpulkan berbagai literatur maupun dokumentasi terkait dengan perancangan infrastuktur *server* VCS (*Version Control System*) yang menggunakan Gitlab sebagai basis pengembangannya, mulai dari jurnal, situs penyedia dokumentasi, serta buku pegangan.
2. Tahap analisis kebutuhan sistem sangat penting untuk menunjang kelancaran penelitian, kebutuhan sistem ini sama dengan kebutuhan sumber daya baik sumber daya perangkat keras maupun perangkat lunak.
   a. Server Fisik
      *Server* fisik menggunakan PC milik Laboratorium Jaringan Komputer Teknik Informatika UPN "Veteran" Jawa Timur untuk melakukan pengujian maupun implementasi. Adapun spesifikasinya dipaparkan pada tabel 1 :

Tabel 1. Spesifikasi Server Fisik

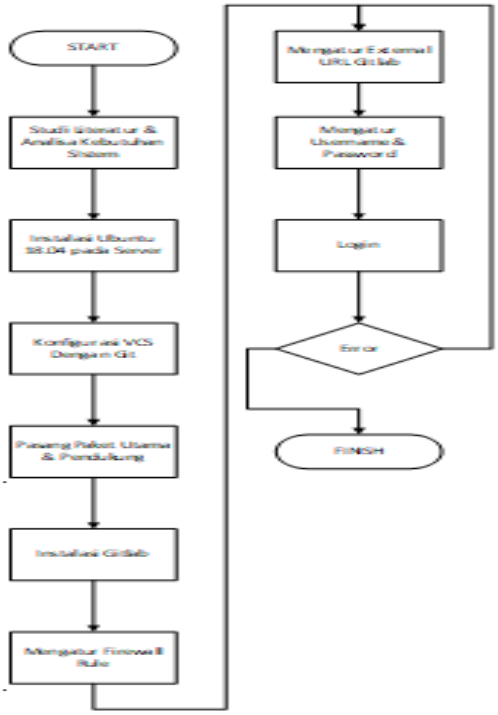
| Prosesor | RAM | Storage | OS |
|---|---|---|---|
| Inte Core i5 6400T | 4GB | HDD 500GB | Ubuntu Server 18.04 |

   b. VCS
      Adapun spesifikasi dari sistem kontainer yang digunakan dijelaskan pada tabel 2:

Tabel 2. Spesifikasi VCS

| Image | Aplikasi | Exposed Port |
|---|---|---|
| Ubuntu Server 18.04 | ca-certificates, curl, openssh-server, postfix, gitlab-ce | 25, 465, 587, 110, 143 993, 995, 80, 443, 8080, 8443, 7071. |

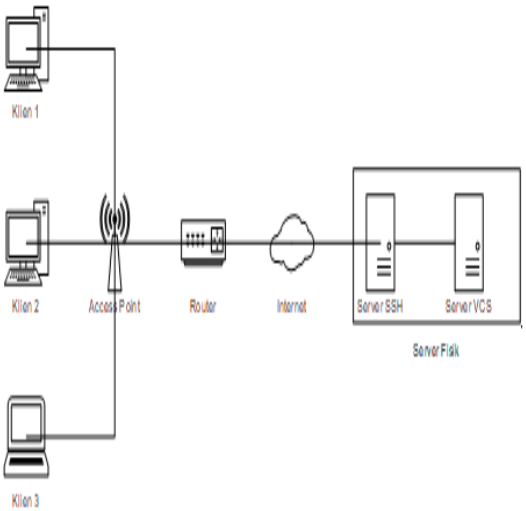
Berikut ini diagram alur penelitian yang di gambarkan pada gambar 1.




Gambar 1. Diagram Alur Penelitian

**Topologi Jaringan**

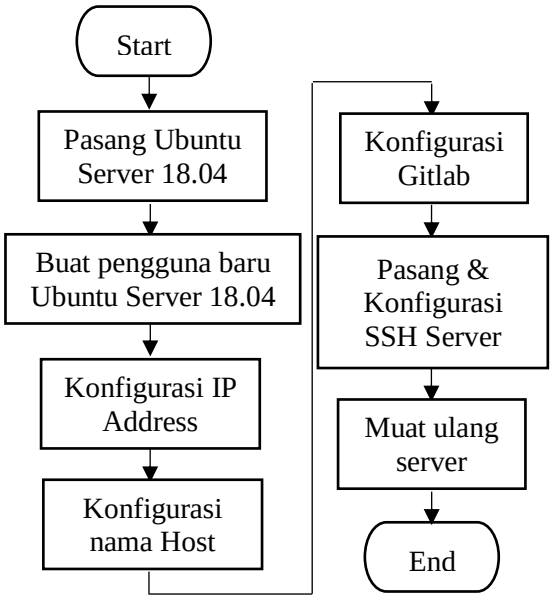
Berikut ini adaah rancangan topologi jaringan yang akan dipakai pada perancangan infrastruktur *server cloud computing* PaaS (*platform as a service*) berbasis *container*. Adapun untuk topologi jaringannya pada gambar 2:




Gambar 2. Topologi Jaringan

**Konfigurasi Server VCS**

Konfigurasi *server* VCS (*Version Control System*) ini dilakukan untuk membuat lingkungan *server* utama yang juga menjadi tempat dijalankannya sistem VCS (*Version Control System*). Adapun proses konfigurasi pada *server* dijelaskan pada gambar 3, yaitu:




Gambar 3. Konfigurasi Server VCS

**Pasang Paket Utama dan Pendukung**

Implementasi penelitian ini membutuhkan beberapa paket utama dan pendukungnya (*dependency*) agar berjalan sesuai harapan. Adapun paket-paket yang dipasang dibagi menjadi dua, yaitu :

a. Paket Utama
   Paket-paket pada *server* digunakan untuk menjalankan git dan gitlab beserta *repository*nya
b. Paket Pendukung
   Paket-paket yang ada pada *server* digunakan untuk menjalankan aplikasi pendukung. Adapun paket-paket ini bersifat *dependency*

## II. Hasil dan Pembahasan

Berikut ini hasil dan pembahasan perancangan infrastruktur server VCS dengan GitLab berbasis Git

1. Instal Paket Dependency
   Sebelum dapat menginstal Gitlab, penting untuk menginstal beberapa paket perangkat lunak yang dimanfaatkan selama instalasi maupun proses kedepannya. Semua paket pendukung yang diperlukan dapat diinstal dengan mudah dari *repository default* Ubuntu.

Pertama-tama update repository dengan perintah:

```
$ sudo apt update
```

Lalu ketikkan perintah berikut untuk proses selanjutnya :

```
$ sudo apt install ca-sertificate curl openssh-server-postfix
```

2. Instal GitLab
   Setelah semua paket pendukung telah terpasang, Gitlab sudah siap untuk diinstal. Dimana proses instalasi ini cukup mudah untuk dilakukan. Pindah ke direktori "/tmp" dan *download* skrip instalasi dengan perintah :

   ```
   $ cd /tmp
   $ curl –LO https://pckages.gitlab.com/install/repositories/gitlab/gitlab-ce/script.deb.sh
   ```
   Setelah skrip instalasi ter*download*, jalankan *installer* dengan perintah:

   ```
   $ sudo bash /tmp/script.deb.sh
   ```

   Skrip akan mengatur *server* untuk menggunakan *repository* yang dikelola GitLab. Ini memungkinkan mengelola GitLab dengan alat manajemen paket yang sama dengan yang digunakan untuk paket sistem lainnya. Setelah ini selesai, dapat menginstal aplikasi GitLab yang sebenarnya dengan perintah :

   ```
   $ sudo apt install gitlab-ce
   ```

3. Mengatur Firewall Rule
   Sebelum mengkonfigurasi GitLab, pastikan bahwa *firewall rule* cukup permisif untuk memungkinkan lalu lintas web. Jika mengikuti panduan yang ditautkan dalam prasyarat, ufw *firewall* harus diaktifkan. Secara *default*, *firewall rule* Ubuntu 18.04 saat ini memungkinkan lalu lintas SSH melaluinya, tetapi akses ke layanan lain dibatasi. GitLab adalah aplikasi web, maka harus mengizinkan akses HTTP. Karena akan mengambil keuntungan dari kemampuan GitLab untuk meminta dan mengaktifkan sertifikat TLS / SSL gratis dari Let's Encrypt, izinkan juga akses HTTPS. Protokol *port mapping* untuk HTTP dan HTTPS tersedia di *file* "/etc/services", sehingga dapat mengizinkan *traffic* berdasarkan nama. Jika belum mengaktifkan *traffic* OpenSSH, dapat diaktifkan dengan perintah :

   ```
   $ sudo ufw allow http
   $ sudo ufw allow https
   $ sudo ufw allow OpenSSH
   ```

   Setelah itu cek status ufw dengan perintah:

   ```
   $ sudo ufw status
   ```

4. Mengatur External URL
   Sebelum dapat menggunakan Gitlab, pertama-tama *update file* konfigurasi dan jalankan *command* rekonfigurasi. Pertama-tama buka *file* konfigurasi Gitlab dengan perintah :

   ```
   $ sudo nano /etc/gitlab/gitlab.rb
   ```

   Cari baris "external_url 'http://example.com'". Untuk example.com dapat diganti dengan IP Public / Domain yang tersedia pada *server*. Apabila terdapat domain yang tersedia, disarankan untuk mengganti http menjadi https, sehingga Gitlab secara otomatis melakukan *redirect* yang dilindungi oleh Let's Encrypt. Selanjutnya, cari baris "letsencrypt['contact_emails']". Baris ini menetapkan alamat email yang dapat digunakan Let's Encrypt untuk menghubungi apabila terjadi masalah pada domain. Dapat diisi seperti berikut :

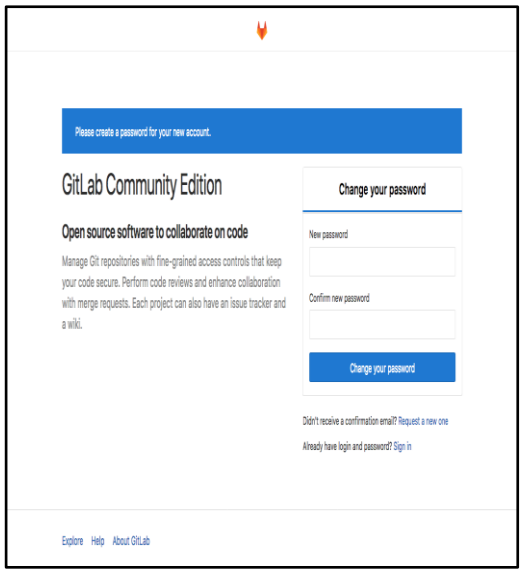
   ```
   Letsencrypt['contact emails'] = ['sammy@example.com']
   ```

   Simpan dan tutup file. Lakukan rekonfigurasi GitLab dengan perintah:

   ```
   $ sudo gitlab-ctl reconfigure
   ```

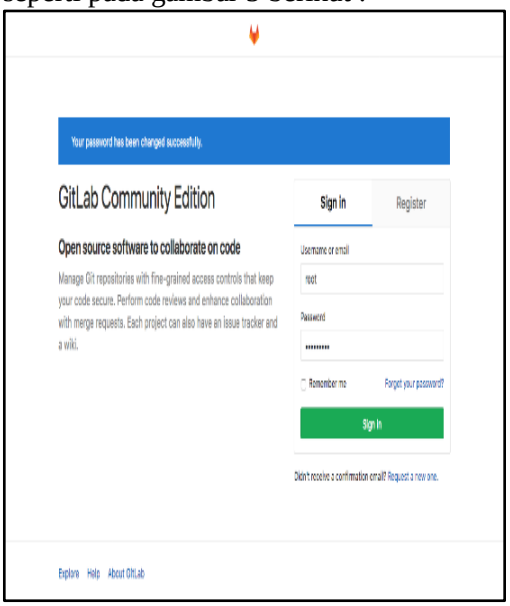
5. Login
   Dengan GitLab berjalan dan akses diizinkan, dapat dilakukan beberapa

konfigurasi awal aplikasi melalui *interface* web. Pertama-tama ketikkan IP Public/Domain yang tersedia di *server* pada *web browser*. Setelah itu, akan ditampilkan halaman web untuk mengatur *password* seperti pada gambar 5 :




Gambar 4. Mengatur *Password*

Pada 2 *form* yang tersedia isi *password* yang diinginkan. Lalu klik *button* "Change your password" ketika selesai. Setelah itu, akan diarahkan pada halaman web *login* seperti pada gambar 5 berikut :

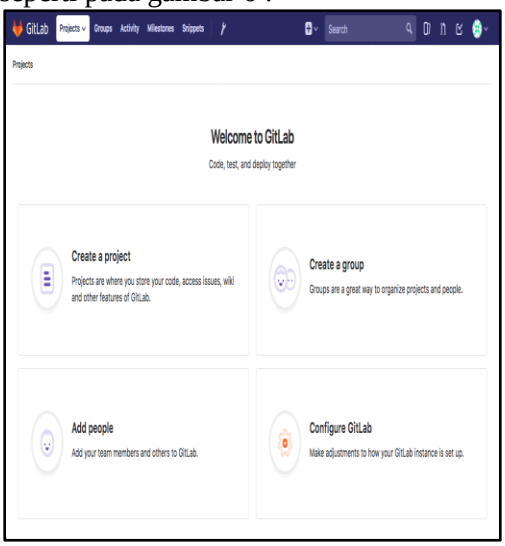



Gambar 5. Halaman Login

Setelah mengeset *password* pada halaman web sebelumnya, dapat login dengan menggunakan *username* berikut :

- Username: root
- Password : [Password yang sudah diset]

Masukkan *value* yang diinginkan pada *form* yang tersedia dan klik *button* "Sign in". Setelah itu akan masuk aplikasi dan diarahkan pada halaman Home Gitlab seperti pada gambar 6 :




Gambar 6. Halaman Home Gitlab

## III. Simpulan

Kesimpulan yang dapat diambil terkait penelitian perancangan infrstruktur *server* VCS (*Version Control System*) dengan Gitlab berbasis Git adalah sebagai berikut :

a. Layanan Gitlab dapat digunakan sebagai infrastruktur layanan dokumentasi alternatif pengganti layanan dokumentasi lama yang bersifat manual.
b. Layanan Gitlab dapat digunakan sebagai tempat untuk mengemas konfigurasi, kode, *library*, dan *runtime*, mengemasnya dan mengaksesnya dimana saja selama ada akses internet.
c. Layanan Gitlab dapat digunakan untuk membangun infrastruktur *server* VCS (*Version Control System*) berbasis Git.

## IV. Daftar Pustaka

[1] Team, G. D. 2018. *Memulai Git - Sejarah Singkat Git*. Retrieved from Git: https://git-scm.com/book/id/v1/Memulai-Git-Sejarah-Singkat-Git - Diakses 22 Juni 2019.

[2] Otte S. 2009. Version control systems. Computer Systems and Telematics.

[3] Nazatul Nurlisa Zolkifli, Amir Ngah, Aziz Deraman. 2018. Version Control System:A Review. International Conference on Computer Science and Computational Intelligence.

[4] Fajar, R. 2014. *10 Version Control System yang Harus Kamu Kenal*. Retrieved from Codepolitan: https://www.codepolitan.com/10-version-control-system-yang-harus-kamu-kenal - Diakses 22 Juni 2019

[5] Team, G. D. 2018. *What is GitLab?* Retrieved from Gitlab: https://about.gitlab.com/what-is-gitlab/ - Diakses 22 Juni 2019

[6] Team, G. D. 2018. *Memulai Git - Tentang Version Control*. Retrieved from Git: https://git-scm.com/book/id/v1/Memulai-Git-Tentang-Version-Control - Diakses 22 Juni 2019.

[7] Emma Jane Hogbin Westby. 2015. Git For Teams. O'Reilly.

[8] Jakub Narębski. 2016. Mastering Git. Packt Publishing Ltd