Jurnal Ismetek *ISSN 2406-9841*
Vol. 17 No.2 Juni Tahun 2024 *e-IISN 2986-2973*

ANALISIS EFEKTIFITAS *CI / CD* DAN MANUAL (TRADISIONAL) DALAM PENGEMBANGAN *WEBSITE* RAKYATWEB.COM

Rachmat Setiabudi
Teknik Informatika, Teknologi Insdustri, Institut Teknologi Budi Utomo
raffisetiabudi@gmail.com

Abstrak
Penelitian ini menganalisis efektivitas penggunaan Teknik CI / CD (*continuous integration / continuous deployment*) dan manual (tradisional) dalam pengembangan website rakyatweb.com. Dengan membandingkan dua pendekatan ini dalam hal efisiensi, kecepatan pengembangan dan kehandalan dari sebuah situs web. Analisis ini memberikan sebuah wawasan mengenai manfaat dan tantangan dari setiap pendekatan dan pengaruhnya terhadap pengembangan situs web secara keseluruhan. Kesimpulan ini memberikan gambaran berharga kepada pengembang dalam memilih pendekatan yang paling sesuai dengan kebutuhan proyek pengembangan situs web mereka.
Kata Kunci: *websites*, ci/cd, traditional, jenkins

1. PENDAHULUAN

Dengan munculnya teknologi informasi yang semakin canggih, situs web telah menjadi salah satu alat utama bagi organisasi dan individu untuk berkomunikasi, berinteraksi, dan berbisnis secara online bahkan sebagai media promosi. Seiring dengan pertumbuhan yang cepat di era digital ini serta persaingan bisnis di bidang digital yang semakin ketat, kebutuhan akan pengembangan berkelanjutan dari sebuah situs web yang efisien, cepat, dan handal semakin mendesak. Dalam hal ini, dua pendekatan utama telah muncul: *Continuous Integration/Continuous Deployment* (CI / CD) dan metode pengembangan manual.

Continuous Integration / Continuous Deployment (CI / CD) adalah suatu pendekatan dalam pengembangan perangkat lunak di mana perubahan kode secara terus-menerus diuji dan diintegrasikan ke dalam repositori bersama, kemudian di-deploy secara otomatis ke lingkungan produksi. Pendekatan ini menjanjikan perubahan yang lebih cepat, peningkatan kualitas perangkat lunak, dan efisiensi dalam pengembangan.

Namun, banyak organisasi dan pengembang masih menggunakan metode pengembangan manual dalam pembangunan situs web mereka. Metode ini melibatkan proses yang lebih statis, di mana pengembang melakukan semua langkah pengujian dan implementasi secara manual. Meskipun metode ini telah digunakan dalam beberapa tahun terakhir dan mungkin sesuai dengan kebutuhan tertentu, ada kebutuhan untuk mengevaluasi efektivitasnya terhadap pendekatan CI/CD dalam konteks pengembangan situs web.

Dalam konteks ini, evaluasi efektivitas CI/CD dibandingkan dengan metode manual dalam pengembangan situs web menjadi sangat penting. Penelitian ini akan memberikan pemahaman yang lebih baik tentang kelebihan dan kekurangan masing-masing pendekatan ini serta dampaknya terhadap pengembangan situs web, khususnya pada kasus studi yang akan diteliti, yaitu website rakyatweb.com.

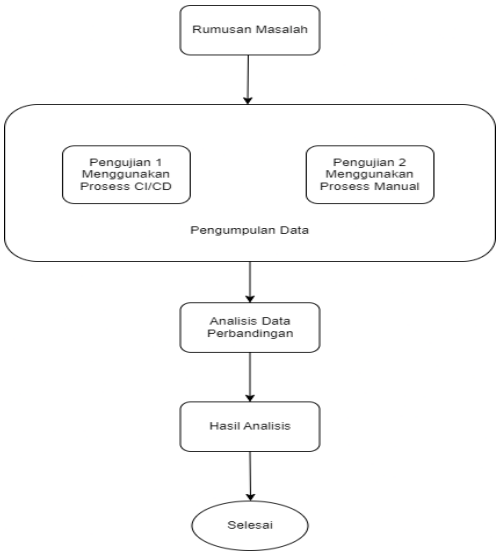
Dengan demikian, penelitian ini dapat memberikan wawasan yang berharga bagi pengembang dan organisasi yang berminat dalam meningkatkan efisiensi dan kualitas pengembangan situs web mereka.

2. METODOLOGI

2.1 Metode Penelitian

Penelitian ini menggunakan pendekatan komparatif untuk menganalisis efektivitas dua metode pengembangan website: CI/CD dan manual (tradisional). Penelitian ini merupakan jenis penelitian kuantitatif yang bertujuan untuk mengukur dan menganalisis data secara numerik untuk lebih memahami efektivitas kedua metode tersebut. Dasar pemikiran penelitian ini didasarkan pada

konsep bahwa penggunaan CI/CD dalam pengembangan *website* dapat meningkatkan efisiensi, kecepatan, dan kualitas pengembangan dibandingkan dengan metode manual (tradisional). Kerangka kerja ini juga mempertimbangkan faktor-faktor lain yang dapat mempengaruhi efektivitas kedua metode, seperti: Kompleksitas proyek, ukuran tim pengembangan, dan infrastruktur teknologi yang tersedia.



Gambar 1. Kerangka Pemikiran
Sumber: Penelitian Mandiri

Dasar pemikiran penelitian ini didasarkan pada konsep bahwa penggunaan CI/CD dalam pengembangan website dapat meningkatkan efisiensi, kecepatan, dan kualitas pengembangan dibandingkan dengan metode manual (tradisional). Kerangka kerja ini juga mempertimbangkan faktor-faktor lain yang dapat mempengaruhi efektivitas kedua metode, seperti: Kompleksitas proyek, ukuran tim pengembangan, dan infrastruktur teknologi yang tersedia. Agar kegiatan penelitian dapat berlangsung secara runut, kami jelaskan kerangka pemikiran dari gambar di atas sebagai berikut:

1. Rumusan masalah berisi identifikasi terkait permasalahan dan data apa yang sekiranya perlu di ambil sebagai bahan analisis.

2. Pengumpulan data dilakukan untuk membuat dua buah model dari proses CI/CD dan manual untuk mendapatkan data dari tiap-tiap proses yang dilakukan.

3. Kemudian di lakukan analisis perbandingan data antara Pengujian satu dan dua.

4. Dari analisis perbandingan kemudian dapat di ketahui hasil ataupun kesimpulan dari masalah yang ada.

2.1.1 CI/CD Pipeline

CI/CD atau *Continuous Integration* and *Continuous Deployment* merupakan suatu cara untuk mengautomatisasi proses testing dan *deployment* pada pengembangan sebuah aplikasi atau website dengan mengintegrasikan *software* atau layanan pihak ketiga sebagai *tools* untuk menguji dan merilis sebuah aplikasi.

CI/CD telah merevolusi cara perangkat lunak dikembangkan dengan mengotomatiskan pengujian dan penerapan. Namun ini lebih dari sekedar teknologi, ini mencerminkan perubahan budaya yang menekankan keterlibatan tim dan fokus pada kualitas. Memahami CI/CD sebagai filosofi pembangunan mendorong keterbukaan, kolaborasi, dan perubahan berkelanjutan.

2.1.2 *Continuous Integration*

Continuous Integration adalah suatu proses yang konsisten dimana pengembang atau developer aplikasi atau website diharuskan untuk menggabungkan hasil kode sumber ke dalam sebuah repository setiap kali menyelesaikan tugas atau pekerjaanya. Dari setiap penggabungan hasil kode sumber akan di lakukan verifikasi dan di uji secara otomatis sehingga kesalahan dapat di ketahui lebih dini.

Integrasi berkelanjutan (CI) adalah praktik umum dalam pengembangan perangkat lunak. Pengembang menggabungkan kode ke dalam repositori bersama beberapa kali sehari untuk mendapatkan masukan cepat. CI didukung oleh pembuatan dan pengujian otomatis, yang memungkinkan kolaborasi tim yang efisien. CI juga memungkinkan perusahaan perangkat lunak untuk merilis program lebih sering dan dalam siklus yang lebih pendek. Hal ini mengurangi kesulitan dan biaya integrasi. Tim pengembangan didorong untuk membangun perangkat lunak dalam iterasi singkat dan

mengintegrasikan kode fungsional dengan cepat.

2.1.3 *Continuous Deployment*

Continuous Deployment, Praktik untuk melakukan perilisan aplikasi secara otomatis ke dalam lingkungan produksi setelah kode sumber melewati proses CI, CD memberikan akses untuk aplikasi supaya tetap dalam kondisi yang update dan mempermudah perbaikan bug karena dapat dilakukan roolback ke versi sebelumnya.

Continuous Deployment (CD) adalah langkah selanjutnya dalam proses integrasi berkelanjutan (CI) di mana semua kode berhasil diintegrasikan. Selama fase ini, aplikasi dapat dibuat dan dirilis secara otomatis tanpa campur tangan manusia. Hal ini membuat perubahan kode yang diuji secara otomatis segera tersedia bagi pengguna akhir.

2.1.4 *Virtual Private Server*

VPS atau *Virtual Private Server* merupakan *server virtual* atau *virtual machine* yang sumber daya nya digunakan oleh satu user saja, virtual server yang di hasilkan merupakan hasil dari adanya teknologi virtualisasi pada *hardware* komputer.

VPS biasnaya menyediakan *FullRoot* akses pada level sistem Operasi Linux dan Administrator pada system Operasi Windows, pengguna bertanggung jawab penuh terhadap OS yang telah terinstall pada VPS.

2.1.5 Jenkins

Jenkins adalah sebuah *software open source* yang menjadi *tools* untuk membuild dan mendeploy sebuah projek dengan mudah dan cepat secara otomatis. Jenkin mampu di integrasikan dengan berbagai plugin yang dapat digunakan untuk meningkatkan optimalisasi dalam CI/CD. Jenkins sangat terkenal penggunaanya dalam CI/CD.

Jenkins juga merupakan web aplikasi yang dibuat menggunakan java yang dapat di install di sebuah *server* untuk dapat di gunakan dalam melakukan tugas-tugas dalam pengoperasian CI/CD.

2.2 Metode Pengumpulan Data

Metode kuantitatif digunakan untuk mengumpulkan data empiris yang dapat diukur secara numerik. Penelitian ini dapat mengumpulkan data melalui penelitian, observasi, dan analisis statistik metrik pengembangan situs web seperti waktu publikasi, tingkat keberhasilan implementasi, jumlah kesalahan yang ditemukan, dan kinerja situs web. Kami kemudian menganalisis data yang dikumpulkan secara statistik untuk mengidentifikasi perbedaan yang signifikan antara efektivitas CI/CD dan metode manual (tradisional) dalam mengembangkan situs Rakyatweb.com.

2.3 Analisis Data Kuantitatif

Dari data yang di kumpulkan akan di lakukan analisis secara terpisah dari masing-masing metode baik CI/CD atau manual. Hasil analisis akan di bentuk suatu perbandingan dengan pendekatan waktu pengembangan, kualitas code, dan efisiensi serta seberapa tinggi resiko human erorr dari kedua pendekatan.

3. HASIL DAN PEMBAHASAN

3.1 Hasil

Rakyatweb.com merupakan sebuah website yang menjembatani penjualan layanan hosting dan domain. Rakyatweb.com memberi layanan hosting yang terjangkau sehingga sangat cocok untuk mahasiswa, proyek pribadi dan portofolio online, serta menyediakan layanan web hosting berkualitas kepada UMKM.

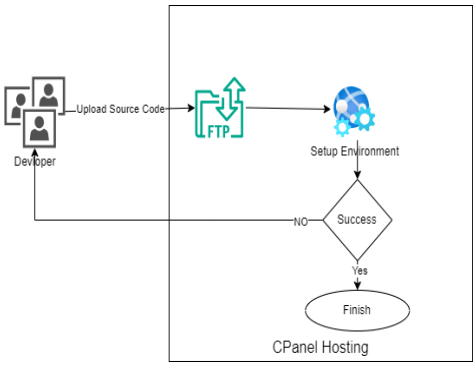
Proses Pengembangan website yang di lakukan di rakyatweb.com saat ini untuk production masih menggunakan metode traditional yaitu waterfall, sementara untuk metode pengembagan menggunakan CI/CD atau Agile Development masih dalam tahap testing.

3.1.1 Metode Traditional/Manual

1. Lingkungan Pengembangan
    a. Tim pengembang menggunakan editor local seperti VS Code dan *Sublime text* kemudian menyimpanya di local komputer masing-masing.
    b. Kemudian masing-masing pengembang menyatukan hasil codinganya untuk di integrasikan.
    c. Tim Pengembang menggunakan server local seperti xampp untuk melakukan pengujian.
2. Prosedur Implementasi/Deployment

ISSN 2406-9841
e-IISN 2986-2973

a. Tim pengembang melakukan pengujian pengkodean secara manual terhada perubahan fitur yang di buat.
b. *Source code* yang telah lulus *review* akan di kompres dalam format .zip
c. Tim Pengembang melakukan *backup configuration* di *hosting*.
d. Setelah itu tim pengembang mengunggah kode yang telah di compress dalam format .zip ke server produksi menggunakan layanan FTP atau melalui *upload* web ke *Server Hosting.*
e. *Source code* akan di ekstrak, kemudian kebutuhan lingkungan (*environment*) untuk *source code* akan di setup.
f. Tim pengembang melakukan verifikasi terhadap fitur baru dari *website*.

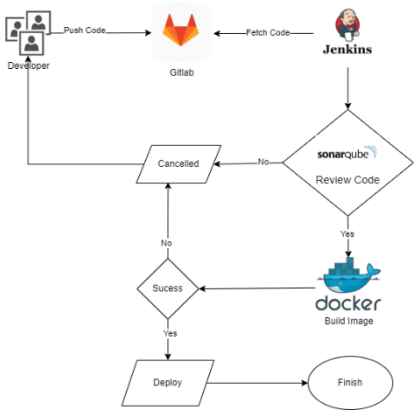


Gambar 2. *Flow Deployment Manual*
Sumber : Penelitian Mandiri

3.1.2 Metode CI/CD

1. Lingkungan Pengembang
    a. Tim pengembangan menggunakan sistem manajemen kode GitLab sebagai repositori kode.
    b. Pengembangan dilakukan secara lokal menggunakan editor kode yang sama dengan metode manual yang kemudian akan di kirim ke repositori kode.
    c. Sistem CI/CD terintegrasi dengan repositori kode untuk mengotomatiskan proses pengujian dan penerapan.
2. Prosedur Implementasi/Deployment
    a. Pengembang menulis atau memperbarui kode secara lokal dan mengirimkan perubahan ke repositori kode.
    b. Sistem CI/CD secara otomatis mendeteksi perubahan dan memicu proses integrasi seperti pengujian otomatis dan pembuatan image docker.
    c. Jika semua pengujian lulus, sistem CI/CD secara otomatis menerapkan perubahan pada lingkungan pengujian atau staging dan melanjutkan pengujian.
    d. Setelah semua pengujian selesai dan perubahan dianggap siap, sistem CI/CD secara otomatis menerapkan perubahan ke produksi tanpa campur tangan manusia.

ISSN 2406-9841
e-IISN 2986-2973



Gambar 3. *Flow Deployment* CI/CD
Sumber: Penelitian Mandiri

3.2 Pembahasan

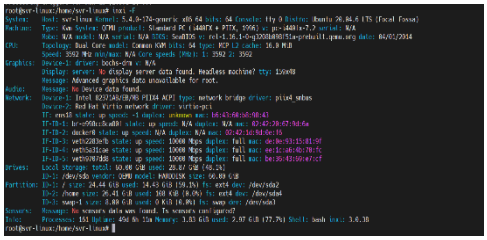
3.2.1 Identifikasi Teknis

1. Metode Manual

a. Hosting CPanel : penggunaan hosting CPanel sebagai lingkungan penyimpanan dan menjalankan website.
b. FTP (File Transfer Protokol) : Penggunaan protocol FTP untuk mengunggah dan memanagement file-file website.
c. Web Server : penggunaan webserver Apache yang di sediakan oleh hosting CPanel
d. Spesifikasi Hosting :
    a. 40 Storage
    b. Unlimited Bandwidth
    c. 40 Email
    d. 20 MySQL DB
    e. 20 Domains
    f. Yes SSH & Remote DB
    g. Free SSL (Let's Encrypt)
    h. 97% Uptime Server

2. Metode CI/CD

a. Jenkins : digunakan sebagai Server CI/CD untuk mengautomatisasi proses integrasi, pengujian/analysis code dan deployment.
    b. SonarQube : Tools untuk melakukan analisa statis kode dan kualitas kode yang di kirim oleh Jenkins
c. Docker : Penggunaan Docker untuk mengemas dan menjalankan aplikasi dalam kontainer.
d. Portainer : digunakan untuk memanajemen image dan container yang berbasi web.
e. Gitlab : digunakan sebagai repositori kode.
f. Spesifikasi VPS sebagai Server Jenkins, Docker dan Sonarqube



Gambar 4. Spesifikasi Server Ubuntu
Sumber : Penelitian Mandiri

3.2.2 Perbandingan CI/CD dengan Manual

1. Perbandingan Waktu

Penulis akan menjabarkan perbandingan waktu dari segi waktu pengembangan, waktu deployment dan waktu pemulihan jika terjadi kegagalan.

a. Waktu Pengembangan

Waktu pengembangan meliputi waktu dalam proses pengkodingan dan integrasi berbagai fitur website jika pengkodingan di lakukan oleh beberapa orang atau developer.

1. Metode Manual

Sesuai dengan hasil identifikasi sebelumnya, maka dapat di jelaskan bahwa setiap developer melakukan pengkodan sesuai tugas yang di berikan seperti berikut :

ISSN 2406-9841
e-IISN 2986-2973

Task	Developer	Week 1	Week 2	Week 3	Week 4	Week 5
Project Planning	Both	■				
Setup Development Environment	Dev 1	■				
Database Design	Dev 2	■	■			
Backend Development - Model	Dev 1		■	■		
Backend Development - Controller Development	Dev 2		■	■		
Frontend Development - View	Dev 1			■	■	
Backend Integration	Dev 2				■	■
Frontend Integration	Both					■
Testing - Unit Tests	Both					■
Testing - Integration Tests	Both					■
Final Review	Both					■
Project Start Date: 25-08-2023						
Project Leader: Rizky Febryan						

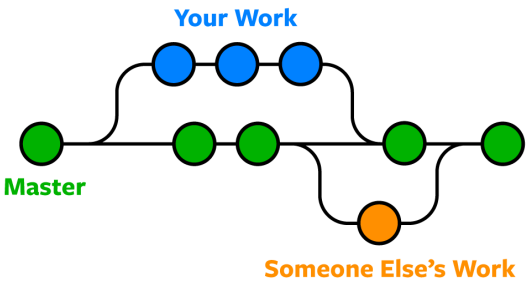
Gambar 5. *Timeline* Pengembangan
Sumber : Penelitian Mandiri

Dalam proses pengembangan dilakukan selama 4 minggu (28 hari) namun pada proses penggabungan codingan dan review/analisis code serta testing memakan waktu 7 hari.

2. Metode CI/CD

Pada Metode ini semua proses Integrasi Kode, review/analisis code di lakukan secara otomatis sehingga setiap sebuah fitur sudah selesai di kerjakan maka dapat langsung di lakukan deployment jika di inginkan. Integrasi *code* di lakukan dengan menggunakan gitlab sebagai *repository source code.*

Berikut flow ketika menggunakan gitlab sebagai *source code repository.*



Gambar 6. Sistem *Branch Git*
Sumber : Penelitian Mandiri

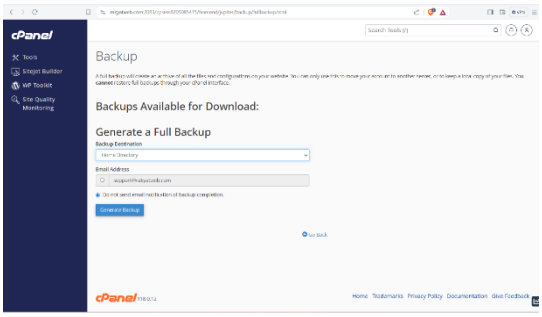
Di gambar ini menjelaskan ketika beberapa developer menggunakan gitlab maka mereka dapat membuat *branch* mereka sendiri sebagai *workspace* yang tidak akan mengganggu branch lain dan ketika sebuah fitur dari code yang di hasilkan dalam *branch* masing-masing *developer* sudah selesai mereka dapat mengirimnya ke *branch* master untuk di gabungkan. Setiap *branch* dapat mengupdate setiap perubahan yang terjadi di branch master. Dengan ini karena proses integrasi atau penggabungan code sudah di lakukan secara otomatis maka waktu penggabungan code menjadi nol.

Kemudian untuk review/analisis code, *source code* yang telah di satukan dalam branch main/master pada saat deploy akan secara otomatis di review/analisis terlebih dahulu dengan tools Sonarqube yang telah terintegrasi dengan Jenkis sebagai *Orchestrator* nya. Sehingga ketika pada stage analisis ini review code di anggap tidak lulus maka proses *deployment* tidak akan di lanjutkan.

Waktu Deployment di hitung ketika source code atau fitur baru yang kemudian di deploy ke linkungan server. Berikut perhitungan waktu deployment sesuai prosedur implementasi yang di jelaskan sebelumnya.Kemudian Downtime adalah dimana website tidak dapat di akses oleh penguna.

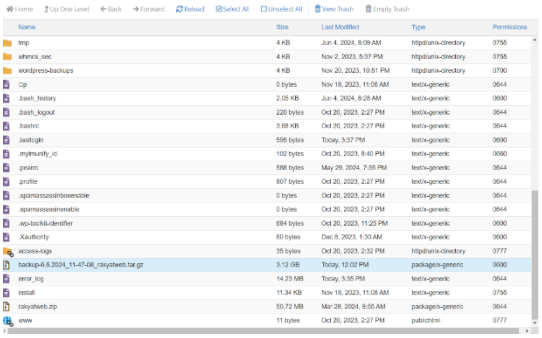
1. Metode Traditional

Karena pada metode traditional semua proses deployment di lakukan manual maka kita akan menghitung setiap Langkah pada proses deploymentnya mulai dari *backup configuration* sebelumnya, mempersiapkan *environment* nya yang meliputi *database server, configurasi '.env' file, set permissions, menyesuaikan root directory, configurasi .htaccess* jika di perlukan.

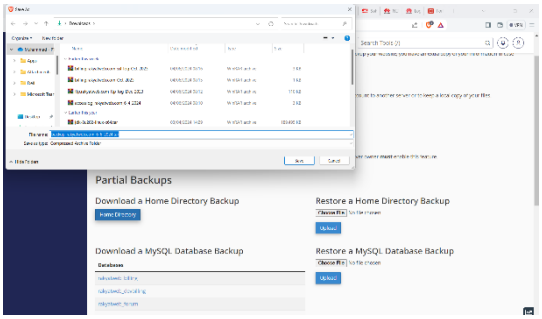


Gambar 7. *Backup Directory Hosting*
Sumber : Penelitian Mandiri

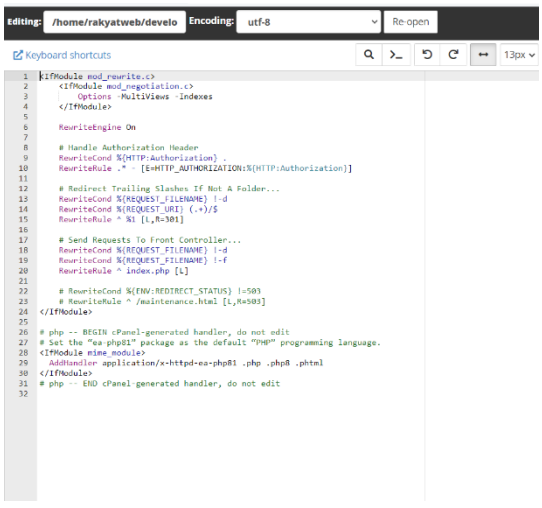
ISSN 2406-9841
e-IISN 2986-2973



Gambar 8. File Hasil Backup
Sumber : Penelitian Mandiri



Gambar 9. Proses Download File Backup
Sumber : Penelitian Mandiri



Gambar 10. Config File .htaccess
Sumber : Penelitian Mandiri

Dari setiap Proses yang penulis jabarkan maka dapat penulis rangkum keseluruhan waktu yang di perlukan dalam deployment ini. Kemudian penulis juga tulis apakah terjadi downtime atau tidak pada setiap Langkah. Berikut tabel perhitungan waktu dari setiap proses.

Tabel 1. Waktu Deployment

Langkah	Waktu	Downtime
Backup Web dan Konfigurasinya	47 Menit	Tidak ada downtime
Upload Aplikasi ke CPanel • Compress source code • Upload source code ke CPanel • Ekstrak source code	20 Detik 27 Deit 1 Menit	Tidak ada downtime
Konfigurasi Environment CPanel	5 menit	Ada downtime
Konfigurasi Database	10 Menit	Ada downtime
Set Permission	10 menit	Ada downtime
Set Root Directory ke Domain	5 menit	Ada Downtime
Konfigruasi file .httaccess	5 menit	Ada downtime
Total Waktu	1 Jam 23 Menit 47 Detik	

Sumber : Penelitian Mandiri

4. KESIMPULAN

Penelitian ini memberikan pemahaman mendalam mengenai efektivitas antara penggunaan continuous integrasi/continuous deployment (CI/CD). Dari pembahasan yang telah penulis deskripsikan maka dapat di ambil kesimpulan terkait hasil analisis dan seberapa efektif antara CI/CD dan manual sebagai berikut :

a. Dari segi waktu
   1. Waktu Pengembangan: Teknik CI/CD mengurangi waktu pengembangan sekitar 20%

dibandingkan dengan teknik manual. Hal ini menunjukkan bahwa CI/CD memungkinkan pengembangan yang lebih efektif melalui otomatisasi build dan pengujian.

2. Waktu Deployment: CI/CD juga mengurangi waktu penerapan sekitar 76%, memungkinkan Anda menerapkan perubahan dan pembaruan lebih sering dan cepat.
3. Waktu pemulihan kesalahan: CI/CD secara signifikan mengurangi waktu pemulihan kesalahan sekitar 90%. Hal ini menunjukkan bahwa otomatisasi dengan CI/CD memungkinkan deteksi dan penyelesaian masalah lebih cepat.

b. Tingkat Kompleksitas Integrasi/Kolaborasi:
   1. Metode Manual:
      a. Kompleksitas Integrasi: Proses integrasi manual sangat kompleks dan rawan kesalahan. Pengembang harus melakukan penyesuaian intensif dan menggabungkan perubahan kode secara manual, sehingga meningkatkan risiko konflik dan kesalahan.
      b. Kolaborasi: memerlukan komunikasi berkelanjutan dan sering kali tertunda karena penggabungan perubahan secara manual dan pengujian manual.

   2. CI/CD Metode:
      a. Kompleksitas Integrasi: CI/CD menyederhanakan integrasi kode dengan mengotomatiskan proses pembuatan dan pengujian. Setiap penerapan secara otomatis terintegrasi dan diuji, mengurangi risiko konflik dan memastikan kualitas kode yang lebih tinggi.
      b. Kolaborasi: meningkatkan kolaborasi dengan jalur pipa otomatis yang memberikan umpan balik cepat dan memungkinkan pengembangan berkelanjutan dan berulang. CI/CD mendukung integrasi berkelanjutan, yang memungkinkan pengembang menggabungkan perubahan lebih sering, sehingga secara signifikan mengurangi kompleksitas integrasi.

DAFTAR PUSTAKA

A. Farid dan I. Gita Anugrah, “Implementasi CI/CD Pipeline Pada Framework Androbase Menggunakan Jenkins (Studi Kasus: PT. Andromedia),” *Jurnal Nasional Komputasi dan Teknologi Informasi*, vol. 4, no. 6, 2021.

V. K. Thatikonda, “Beyond the Buzz: A Journey Through CI/CD Principles and Best Practices,” *European Journal of Theoretical and Applied Sciences*, vol. 1, no. 5, hlm. 334–340, Sep 2023, doi: 10.59324/ejtas.2023.1(5).24.

C. Gea, K. J. D. Lase, dan M. Syamsudin, “Implementasi Virtual Private Server untuk Mini Hosting,” *JURNAL SAINS DAN KOMPUTER*, vol. 7, no. 01, hlm. 5–9, Jan 2023, doi: 10.61179/jurnalinfact.v7i01.402.