Jurnal Teknologi Dan Sistem Informasi
Bisnis Vol. 8 No. 2 April 2026 Hal.137-147
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Analisis Pemanfaatan Platform GitHub Sebagai Alat Kontrol Versi dan Media Deployment Website

Miftah Faroq Santoso
Teknologi Informasi, Fakultas Teknik dan Informatika, Universitas Bina Sarana Informatika,
miftah.mfq@bsi.ac.id

Submitted: 26-03-2026 Reviewed: 31-03-2026 Accepted: 07-04-2026
https://doi.org/10.47233/jteksis.v8i1.2601

Abstract
Static HTML website development often faces serious challenges in managing source code files, which remains manual and unstructured. Conventional practices such as the use of FTP often cause crucial problems, including the risk of data loss due to file overwriting, the inability to track change history, and a lack of version documentation that hinders bug fixing. This research aims to analyze the effectiveness of the GitHub platform as a version control tool and integrated deployment medium by utilizing the GitHub Pages feature in static website projects. The research method applied is Research and Development (R&D) with a descriptive qualitative approach, where the product developed is a static portfolio website prototype based on HTML. Data analysis techniques were carried out through direct observation of the implementation of basic Git commands (init, add, commit, push, pull) as well as a comparative evaluation of the deployment workflow effectiveness against conventional methods. The results show that Git system integration has successfully created structured documentation of code change history through commit logs, thereby effectively eliminating the risk of overwriting and enabling rollback processes in the event of errors. The GitHub Pages feature is proven to provide a fast, free (zero-cost), and automatic deployment solution simply through the git push command without complicated server configuration. Comparative analysis confirms that this method is superior in terms of data security audit trails and operational cost efficiency compared to paid hosting. In conclusion, GitHub effectively functions as an integrated (all-in-one) solution capable of improving developer efficiency in managing and publishing static websites.
Keywords: GitHub, Version Control, Deployment, Static Website, Git.

Abstrak
Pengembangan website statis *HTML* sering menghadapi kendala serius dalam pengelolaan berkas kode sumber yang masih manual dan tidak terstruktur. Praktik konvensional seperti penggunaan *FTP* seringkali menimbulkan permasalahan krusial, meliputi risiko kehilangan data akibat penimpaan berkas *(overwriting)*, ketidakmampuan melacak riwayat perubahan, serta minimnya dokumentasi versi yang menghambat perbaikan bug. Penelitian ini bertujuan untuk menganalisis efektivitas *platform GitHub* sebagai alat kontrol versi sekaligus media *deployment* terintegrasi dengan memanfaatkan fitur *GitHub Pages* pada proyek *website* statis. Metode penelitian yang diterapkan adalah *Research and Development (R&D)* dengan pendekatan kualitatif deskriptif, di mana produk yang dikembangkan berupa *prototipe website* portofolio statis berbasis *HTML*. Teknik analisis data dilakukan melalui observasi langsung terhadap implementasi perintah dasar *Git (init, add, commit, push, pull)* serta evaluasi efektivitas alur kerja *deployment* secara komparatif terhadap metode konvensional. Hasil penelitian menunjukkan bahwa integrasi sistem *Git* berhasil menciptakan dokumentasi riwayat perubahan kode yang terstruktur melalui *log commit*, sehingga secara efektif mengeliminasi risiko *overwriting* dan memungkinkan proses *rollback* jika terjadi kesalahan. Fitur *GitHub Pages* terbukti menyediakan solusi *deployment* yang cepat, gratis *(zero-cost)*, dan otomatis hanya melalui perintah *git push* tanpa konfigurasi *server* rumit. Analisis komparatif mengonfirmasi bahwa metode ini lebih unggul dalam hal keamanan data jejak audit *(audit trail)* dan efisiensi biaya operasional dibandingkan hosting berbayar. Kesimpulannya, *GitHub* efektif berfungsi sebagai solusi terintegrasi *(all-in-one)* yang mampu meningkatkan efisiensi kerja pengembang dalam mengelola dan mempublikasikan *website* statis.
Keywords: GitHub, Kontrol Versi, Deployment, Website Statis, Git.

PENDAHULUAN

Perkembangan teknologi web saat ini telah mengubah paradigma pengembangan perangkat lunak dari pendekatan manual menuju otomatis yang terintegrasi. *Website* statis berbasis *HTML*, meskipun dianggap sebagai teknologi fundamental, kini kembali populer digunakan untuk berbagai kebutuhan seperti *landing page*, portofolio pribadi, dan dokumentasi proyek karena menawarkan kecepatan akses yang tinggi dan keamanan yang lebih baik dibandingkan *website* dinamis. Namun, tantangan utama yang sering dihadapi oleh pengembang, khususnya pemula dan mahasiswa, adalah pengelolaan berkas kode sumber yang masih belum terstruktur. Praktik konvensional seperti penggunaan *File Transfer Protocol (FTP)* atau penyalinan manual berkas ke *hosting* tradisional seringkali menimbulkan permasalahan

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN :2655-8238
P-ISSN : 2964-2132

serius, seperti risiko kehilangan data akibat penimpaan berkas *(overwriting)*, ketidakmampuan melacak riwayat perubahan kode, serta minimnya dokumentasi versi yang menghambat proses perbaikan *bug*. Kondisi ini menunjukkan urgensi kebutuhan akan sebuah sistem yang mampu mengintegrasikan pengelolaan kode dan publikasi *website* dalam satu alur kerja yang efisien.

Berbagai penelitian terkait pengembangan website telah banyak dilakukan dengan fokus pada fungsionalitas dan tampilan antarmuka. Penelitian yang berjudul *"Optimization of Delivery Management Through a Web-Based Monitoring Application Using PHP and Bootstrap"* mengembangkan aplikasi web untuk optimalisasi manajemen pengiriman menggunakan *PHP* dan *framework Bootstrap* [1]. Fokus utama penelitian tersebut adalah pada logika bisnis dan efisiensi proses monitor. Di sisi lain, penelitian tentang aspek *front-end* juga telah banyak dikaji, seperti penelitian berjudul "Teknik Responsive Web Design Bootstrap 4 Serta Penerapannya Dalam Rancang Bangun Layout Web" yang menekankan pentingnya tampilan adaptif di berbagai perangkat [2]. Selain itu, kompleksitas interaksi pengguna juga dibahas dalam penelitian berjudul "Teknik *Single Page Application (Spa) Layout Web* Dengan Menggunakan *React Js* dan *Bootstrap*" yang mengimplementasikan arsitektur *SPA* untuk meningkatkan responsivitas aplikasi [3]. Meskipun ketiga penelitian tersebut telah berhasil menghasilkan aplikasi yang fungsional, tidak ada yang membahas penerapan sistem kontrol versi dalam alur kerjanya. Proses *deployment* yang dilakukan masih bersifat konvensional, di mana kode sumber disalin langsung ke *server hosting* tanpa manajemen repositori, sehingga riwayat perubahan kode tidak dapat dilacak secara terstruktur.

Berdasarkan kesenjangan tersebut, penelitian ini mengusulkan pendekatan integratif dengan memanfaatkan *platform GitHub* sebagai solusi *all-in-one*. Cakupan penelitian tidak hanya terbatas pada *GitHub* sebagai alat kontrol versi, melainkan mengkaji secara spesifik efektivitas fitur *GitHub Pages* sebagai media *deployment* yang terintegrasi langsung dengan repositori kode. Kebaruan dari penelitian ini terletak pada analisis implementasi perintah dasar *Git (push* dan *pull)* dalam membangun alur kerja *(workflow)* publikasi *website* statis yang otomatis, serta analisis komparatif efisiensi biaya *(zero-cost deployment)* terhadap metode *hosting* konvensional. Pembatasan masalah difokuskan pada *website* statis *HTML* untuk menghasilkan model kerangka kerja yang ringan dan mudah diadopsi oleh pengembang pemula.

Harapan dari penelitian ini adalah dapat menghasilkan model alur kerja *(workflow)* yang terstandarisasi bagi pengembang pemula dalam mengelola proyek web. Secara teoritis, penelitian ini diharapkan dapat memperkaya khazanah literatur ilmu komputer, khususnya dalam mata kuliah Pemrograman Web atau Rekayasa Perangkat Lunak, mengenai penerapan praktik *DevOps* sederhana. Secara praktis, hasil penelitian ini diharapkan dapat menjadi panduan bagi akademisi dan praktisi dalam memanfaatkan *platform GitHub* secara optimal untuk menciptakan efisiensi pengembangan, meningkatkan keamanan data kode, serta mengurangi ketergantungan pada infrastruktur *hosting* berbayar.

METODE PENELITIAN

2.1. Jenis dan Pendekatan Penelitian

Penelitian ini menggunakan metode *Research and Development (R&D)* dengan pendekatan kualitatif deskriptif. Metode *R&D* dipilih karena penelitian tidak hanya menganalisis teori, tetapi juga menghasilkan produk berupa *prototipe* website statis yang dikelola menggunakan *platform GitHub*. Menurut Sugiyono[4] metode *R&D* digunakan untuk menghasilkan produk tertentu dan menguji keefektifan produk tersebut. Dalam konteks ini, produk yang dikembangkan adalah sebuah *website* portofolio statis, sementara analisis difokuskan pada proses pengelolaan versi dan efektivitas *deployment*.

2.2. Kajian Teori *(Literature Study)*

Pengembangan website statis merupakan salah satu fondasi utama dalam teknologi informasi yang hingga kini tetap relevan digunakan. Secara teknis, *website* statis dibangun menggunakan bahasa markup *HTML (Hypertext Markup Language)* yang berfungsi sebagai kerangka struktur konten, serta *CSS (Cascading Style Sheets)* yang bertanggung jawab atas aspek tampilan visual dan tata letak responsif [5], [6], [7]. Berbeda dengan website dinamis yang memerlukan pemrosesan di sisi *server*, *website* statis memiliki karakteristik ringan karena file dikirim langsung ke *browser* klien tanpa eksekusi skrip *server-side*, sehingga memberikan kecepatan akses yang lebih tinggi dan tingkat keamanan yang lebih baik dari serangan injeksi *database* [8], [9]. Namun, kompleksitas pengembangan web modern menuntut pengelolaan berkas kode yang tidak hanya fokus pada tampilan, tetapi juga pada manajemen revisi. Tanpa manajemen yang tepat, proses pengembangan website seringkali menghadapi masalah ketidaksinkronan data, terutama ketika melibatkan pengembang lebih dari satu orang [9], [10].

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN :2655-8238
P-ISSN : 2964-2132

Untuk mengatasi permasalahan pengelolaan kode tersebut, diperlukan penerapan Sistem Kontrol Versi *(Version Control System)*. *VCS* adalah sistem yang merekam perubahan pada berkas dari waktu ke waktu, memungkinkan pengguna untuk kembali ke keadaan sebelumnya jika terjadi kesalahan [11]. Di antara berbagai sistem yang ada, *Git* telah menjadi standar industri sebagai sistem kontrol versi terdistribusi *(Distributed VCS)*. Berbeda dengan sistem terpusat tradisional, *Git* menyimpan seluruh database proyek secara lokal di setiap komputer pengembang, sehingga memungkinkan pekerjaan tetap berjalan meskipun terputus dari jaringan internet [12], [13]. Fitur utama dalam *Git* seperti *commit* berfungsi sebagai penanda *snapshot* perubahan, sementara *branching* memungkinkan pengembangan fitur baru secara paralel tanpa mengganggu cabang utama *(main branch)*, yang secara signifikan meningkatkan efisiensi alur kerja pengembangan perangkat lunak [14], [15], [16].

Sebagai implementasi dari sistem *Git*, *Platform GitHub* hadir sebagai layanan hosting repositori berbasis *cloud* yang menyediakan antarmuka pengguna grafis dan fitur kolaborasi sosial untuk para pengembang [17]. *GitHub* tidak hanya berfungsi sebagai tempat penyimpanan kode, tetapi juga menyediakan fitur pelacakan masalah *(issue tracking)* dan *pull request* yang memfasilitasi review kode antar anggota tim [18], [10]. Lebih jauh lagi, dalam konteks publikasi web, *GitHub* menawarkan fitur *GitHub Pages*. Fitur ini memungkinkan repositori kode diubah langsung menjadi *website* statis yang dapat diakses publik melalui protokol HTTPS [8]. Pendekatan ini menawarkan paradigma baru dalam proses *deployment* yang lebih efisien dibandingkan metode konvensional menggunakan *File Transfer Protocol (FTP)*. Metode *FTP* tradisional seringkali rentan terhadap kesalahan manual saat mengunggah file dan tidak memiliki mekanisme otomatis untuk melakukan *rollback* jika terjadi kegagalan sistem, sedangkan integrasi *Git* dan *GitHub Pages* menciptakan alur *deployment* yang otomatis dan terdokumentasi rapi [19]. Dengan demikian, penggunaan platform ini menciptakan solusi terintegrasi yang menjawab tantangan biaya infrastruktur *(zero-cost hosting)* dan keamanan data pengembangan web[20].

2.3. Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan selama periode 2 (dua) bulan, dimulai dari tahap studi literatur hingga penyusunan laporan akhir. Proses pengembangan dan pengujian sistem dilakukan di Laboratorium Komputer, dengan memanfaatkan jaringan *internet* untuk mengakses layanan *cloud GitHub*.

2.4. Objek Penelitian

Objek dalam penelitian ini adalah *platform GitHub* sebagai penyedia layanan *hosting* repositori dan *deployment* website statis. Fokus utama penelitian meliputi:

a. Sistem kontrol versi *Git* yang terintegrasi pada *platform GitHub*.
b. Fitur *GitHub Pages* sebagai media publikasi *website* statis *HTML*.
c. Alur kerja *(workflow)* standar pengembangan web menggunakan perintah dasar *Git*.

2.5. Alat dan Bahan Penelitian

Untuk mendukung pelaksanaan penelitian, diperlukan seperangkat alat (instrumen) dan bahan yang relevan. Rincian spesifikasi alat dan bahan adalah sebagai berikut:

a. Alat (*Tools* dan *Software*)
Alat yang digunakan terdiri dari perangkat keras *(hardware)* dan perangkat lunak *(software)* yang terintegrasi dalam satu lingkungan pengembangan.

Tabel 1. Spesifikasi Alat Penelitian

Kategori	Spesifikasi Alat	
	Nama Alat	***Spesifikasi***
Perangkat Keras	Laptop / PC	Prosesor Intel Core i3 Gen 12, RAM 8 GB, SSD 120 GB, Internet
Sistem Operasi	Windows 10 / 11	Lingkungan kerja utama pengembang
Version Control	Git Bash (v2.40+)	Aplikasi CLI untuk menjalankan perintah Git lokal
Code Editor	Visual Studio Code	Editor teks
Web Browser	Google Chrome	Digunakan untuk akses platform GitHub dan pengujian tampilan website
Platform Cloud	GitHub.com	Layanan hosting repositori dan GitHub Pages

Sumber: Miftah Faroq

b. Bahan *(Materials)*

Bahan penelitian yang digunakan meliputi:

1. Kode Sumber *HTML* dan *CSS (Bootstrap* atau *Tailwind)*: Berupa script *markup* dan *styling* untuk membangun tampilan *website* portofolio statis.
2. Dokumentasi Resmi: Referensi teknis dari *GitHub Docs* mengenai penggunaan *GitHub Pages*.

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN :2655-8238
P-ISSN : 2964-2132

3. Literatur Terdahulu: Jurnal ilmiah terkait pengembangan web dan kontrol versi sebagai pembanding.

2.6. Metode Pengumpulan Data

Data dalam penelitian ini dikumpulkan melalui dua teknik utama:

a. Studi Literatur *(Library Research)*

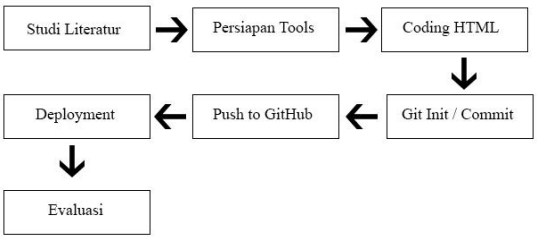
Teknik ini dilakukan dengan mengumpulkan data sekunder dari buku-buku teks, artikel jurnal ilmiah bereputasi (SINTA/Scopus), dan dokumentasi resmi. Studi literatur bertujuan untuk membangun landasan teori mengenai *Git*, sistem kontrol versi, dan arsitektur *website* statis. Data ini digunakan untuk membandingkan metode *deployment* konvensional dengan metode modern berbasis *GitHub*.

b. Observasi Partisipan

Peneliti bertindak langsung sebagai pengembang *(developer)* yang melakukan implementasi. Data primer diperoleh dari hasil pengamatan langsung terhadap output perintah *Git* di *terminal*, *log* aktivitas repositori di *GitHub*, serta keberhasilan akses domain *website* yang di-*deploy*. Observasi juga mencakup pencatatan waktu tempuh dari proses *coding* hingga website *live*.

2.7. Prosedur Penelitian

Prosedur penelitian dirancang secara sistematis untuk memastikan validitas hasil analisis. Tahapan penelitian diilustrasikan pada Gambar 1 dan penjelasan berikut:



Gambar 1. Alur dan Tahapan Prosedur Penelitian

a. Tahap 1: Studi Literatur dan Perencanaan

Tahap awal melibatkan pengumpulan informasi mendalam mengenai prinsip kerja *Version Control System (VCS)* dan fitur *static site hosting*. Peneliti merancang arsitektur *website* portofolio yang akan dibuat, memastikan website tersebut murni statis tanpa pemrosesan *server-side*.

b. Tahap 2: Persiapan Lingkungan Pengembangan

Pada tahap ini, perangkat lunak *Git* diinstal dan dikonfigurasi pada komputer lokal. Peneliti membuat akun *GitHub* dan menyiapkan repositori baru dengan penamaan standar (sesuai format *username.github.io* atau nama proyek). Konfigurasi SSH Key juga dilakukan untuk memastikan keamanan koneksi antara komputer lokal dan server GitHub.

c. Tahap 3: Pengembangan *Website* Statis HTML

Pembuatan website dilakukan dengan menulis kode struktur *(HTML)* dan gaya visual *(CSS)* pada *Visual Studio Code*. Proses ini menghasilkan berkas utama seperti *index.html*, *style.css*, dan aset gambar. Fokus pada tahap ini adalah fungsionalitas tampilan tanpa memikirkan kompleksitas *backend*.

d. Tahap 4: Implementasi Kontrol Versi *(Git Workflow)*

Ini adalah tahap kritis dalam penelitian. Proses pengelolaan kode dilakukan melalui serangkaian perintah *Git* di terminal:

1. Inisialisasi: Perintah ***git init*** untuk membuat repositori lokal.
2. *Staging*: Perintah ***git add .*** untuk menandai berkas yang akan disimpan.
3. *Commit*: Perintah ***git commit -m "pesan"*** untuk menyimpan versi secara virtual.
4. *Branching*: Membuat cabang percobaan untuk pengembangan fitur baru tanpa mengganggu cabang utama ***(main)***.

e. Tahap 5: *Deployment* dengan *GitHub Pages*

Setelah kode siap, berkas dikirim ke *platform cloud* menggunakan perintah *git push origin main*. Selanjutnya, fitur *GitHub Pages* diaktifkan secara manual pada *settings* repositori. Peneliti memilih sumber *deployment* dari cabang *main* untuk mempublikasikan *website*. Proses ini diamati untuk mengukur kecepatan propagasi hingga *website* dapat diakses publik.

f. Tahap 6: Evaluasi dan Analisis

Tahap terakhir adalah evaluasi hasil. Peneliti menganalisis *log commit* untuk memastikan riwayat perubahan tersimpan dengan baik, serta menguji fungsi *git pull* untuk menyinkronkan perubahan yang dilakukan di komputer lain. Evaluasi juga mencakup perbandingan efisiensi metode ini terhadap metode manual *(FTP)*.

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN :2655-8238
P-ISSN : 2964-2132

2.8. Teknik Analisis Data

Teknik analisis data yang digunakan adalah analisis deskriptif komparatif. Data yang diperoleh dari hasil observasi dan implementasi dibandingkan dengan teori dan penelitian terdahulu. Indikator keberhasilan analisis ditentukan sebagai berikut:

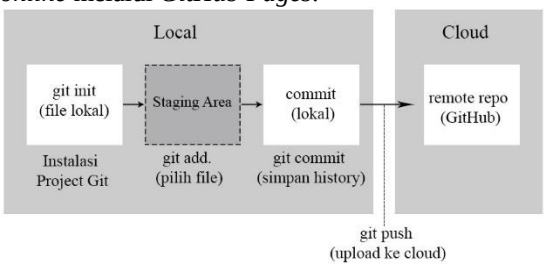
a. Keutuhan Riwayat: Apakah setiap perubahan kode terekam dalam *log Git*.
b. Ketersediaan Akses: Apakah *website* dapat diakses publik melalui *URL* yang disediakan *GitHub Pages*.
c. Efisiensi: Perbandingan waktu dan biaya antara metode *GitHub Pages* dan *hosting* konvensional.

HASIL DAN PEMBAHASAN

3.1. Perancangan Sistem dan Alur Kerja *Git*

Tahap awal dalam penelitian ini adalah merancang arsitektur sistem dan alur kerja *(workflow)* pengembangan website. Perancangan ini bertujuan untuk memastikan bahwa setiap tahapan pengembangan *website* statis *HTML* dapat terdokumentasi dengan baik menggunakan platform *GitHub*. Berbeda dengan pengembangan konvensional yang bersifat *linear*, perancangan ini mengadopsi pola iteratif yang didukung oleh sistem kontrol versi.

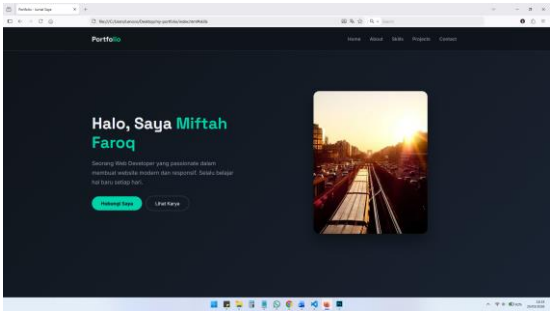
Proses pengelolaan kode sumber *website* portofolio dilakukan menggunakan sistem *version control Git*. *Workflow* dimulai dengan inisialisasi repositori lokal menggunakan perintah *git init*, kemudian *file-file website* ditambahkan ke *staging area* dengan *git add .* Setelah *file* siap, dilakukan *commit* untuk menyimpan *snapshot* perubahan menggunakan *git commit -m* "pesan". Tahap terakhir adalah menghubungkan repositori lokal ke *remote repository GitHub* menggunakan *git remote add origin* dan mengirim kode dengan *git push*. Dengan demikian, *website* dapat diakses secara *online* melalui *GitHub Pages*.



Gambar 2. Alur *(Workflow) Git*

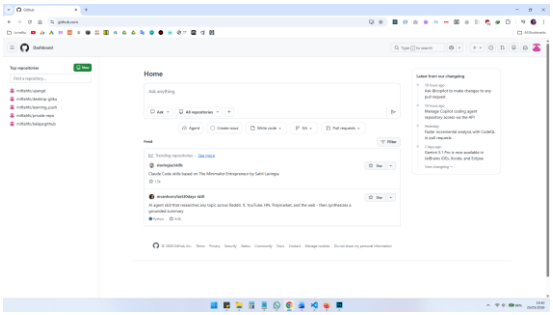
Pada Gambar 2, terlihat bahwa setiap perubahan kode di lingkungan lokal harus melalui proses *staging* dan *commit* sebelum dikirim ke repositori *remote*. Perancangan ini memastikan bahwa tidak ada perubahan kode yang tidak terdeteksi, sehingga meminimalisir risiko *bug* pada *website* yang telah dipublikasikan. Selain itu, struktur direktori website dirancang sederhana dengan berkas utama *index.html* sebagai *entry point* untuk mempermudah proses *deployment*.

Dalam perancangan *website* portofolio ini, akan digunakan *Visual Studio Code* sebagai *code editor* dengan implementasi teknologi *HTML* dan *CSS Bootstrap*. Struktur proyek dibuat sesederhana mungkin, hanya terdiri dari file *index.html* dan *style.css*, dengan tujuan untuk mempermudah pemahaman dan fokus pada pembelajaran perintah dasar *Git* seperti *git init, git add ., git commit, git push,* dan *git pull* tanpa kompleksitas struktur file yang berlebihan. Hasil implementasi website dapat dilihat pada Gambar 3 di bawah ini.



Gambar 3. Tampilan Web *Portfolio*

Sebelum melakukan *push* ke *remote repository*, pengguna wajib memiliki akun *GitHub* terlebih dahulu yang dapat didaftarkan secara gratis melalui situs *github.com*. Setelah akun tersedia, proses autentikasi dilakukan melalui *Git Credential Manager* atau *Personal Access Token* untuk mengamankan koneksi antara repositori lokal dengan *GitHub*. Selanjutnya, pengguna membuat repositori baru di *GitHub* sebagai tempat penyimpanan kode secara *online*. Repositori yang telah dibuat kemudian dihubungkan ke repositori lokal menggunakan perintah *git remote add origin* yang kemudian dilanjutkan dengan proses *git push* untuk mengirimkan kode. Dashboard halaman *GitHub* dapat dilihat pada gambar 4.



Gambar 4. *Dashboard GitHub*

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis

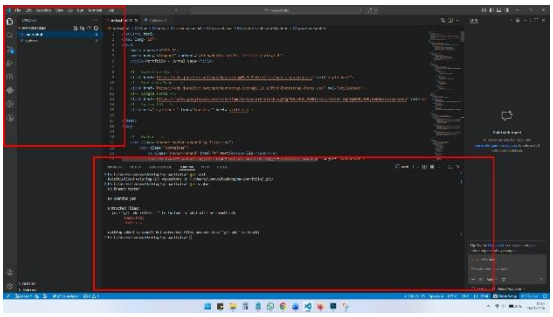
E-ISSN :**2655-8238**
P-ISSN : **2964-2132**

3.2. Implementasi dan Uji Coba

Tahap pertama dalam penelitian ini adalah mengimplementasikan sistem kontrol versi *Git* pada lingkungan pengembangan lokal. Proses ini bertujuan untuk menggantikan metode manual penyalinan *folder (Copy-Paste)* yang rawan kehilangan data.

Proses dimulai dengan inisialisasi repositori lokal, dapat dilihat pada gambar 5 hingga 10. Melalui terminal *Git Bash*, perintah *git init* dieksekusi. Hasil observasi menunjukkan bahwa *Git* berhasil membuat folder tersembunyi *.git* yang berfungsi sebagai database lokal untuk mencatat setiap perubahan kode. Selanjutnya, berkas *index.html* dimasukkan ke dalam area *staging* menggunakan perintah *git add .*, dan disimpan secara permanen menggunakan *git commit*.

Pencatatan versi ini terbukti efektif. Ketika peneliti melakukan kesalahan penulisan kode dan ingin kembali ke versi sebelumnya, perintah *git log* dapat menampilkan riwayat perubahan secara kronologis. Hal ini membuktikan bahwa pemanfaatan *Git* telah memecahkan masalah pengelolaan versi kode yang seringkali membingungkan bagi pengembang pemula.



Gambar 5. *Interface Visual Code Editor*

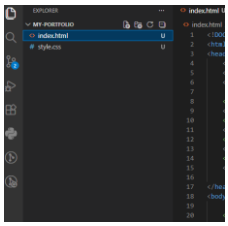
Tabel 2. Jenis dan perintah dasar git

Tahapan	Perintah	Implementasi	Fungsi
Init	git init	# Masuk ke folder project cd D:/portfolio-jurnal # Inisialisasi Git git init	Perintah *git init* digunakan untuk menginisialisasi *folder* lokal menjadi repositori *Git*. Setelah perintah ini dijalankan, *Git* akan membuat *folder* tersembunyi *.git* yang berisi semua metadata dan *history* perubahan.
Staging	git add.	# Melihat status file git status # Menambahkan file tertentu ke staging git add index.html git add style.css # Atau menambahkan semua file sekaligus git add .	*Staging* area adalah area persiapan sebelum *file* disimpan secara permanen. *File* yang berada di *staging area* sudah dipilih dan siap untuk di-*commit*, namun belum tersimpan dalam *history Git*.
Commit	git commit -m "pesan"	# Melakukan commit dengan pesan git commit -m "Initial commit: membuat website portfolio"	*Commit* merupakan proses menyimpan *snapshot* dari *staging area* ke *database Git* secara permanen. Setiap *commit* memiliki ID unik *(hash)* dan pesan yang menjelaskan perubahan yang dilakukan. Pesan *commit* yang baik mengikuti konvensi: Singkat dan jelas, menggunakan bahasa Inggris atau Indonesia secara konsisten, menjelaskan apa yang berubah, bukan bagaimana berubah
Remote	git remote add origin <url>	# Menambahkan remote repository git remote add origin https://github.com/username/portfolio-jurnal.git # Verifikasi remote git remote -v	*Remote repository* adalah tempat penyimpanan kode di *server (*seperti *GitHub, GitLab,* atau *Bitbucket)* yang memungkinkan kolaborasi dan *backup online*.
Push	git push -u origin main	# Mengirim commit lokal ke remote repository git push -u origin main	Perintah pada *Git* yang digunakan untuk mengirimkan *(upload) commit* dari repositori lokal ke repositori *remote* seperti *GitHub* atau *GitLab.*
Pull	git pull	# Mengambil file dari *Cloud* ke lokal repositori git pull origin main	Untuk mengambil *(download)* perubahan terbaru dari repositori *remote* dan menggabungkannya *(merge)* secara otomatis ke repositori lokal, sehingga kode lokal tetap sinkron dengan versi di *server*.

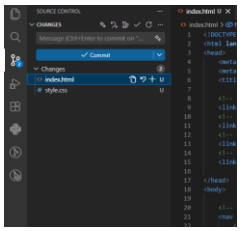
Sumber: Miftah Faroq

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis

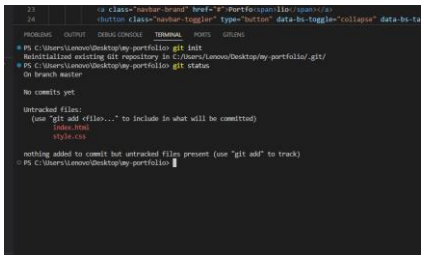
E-ISSN :**2655-8238**
P-ISSN : **2964-2132**



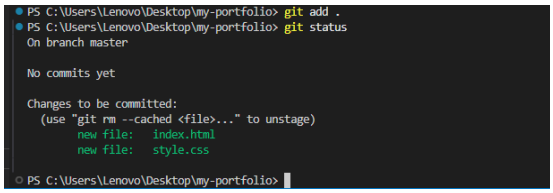
Gambar 6. Berkas *File*



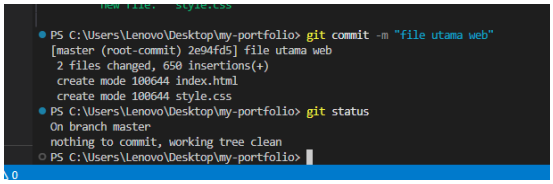
Gambar 7. *Source Control*



Gambar 8. *Terminal (Bash)*



Gambar 9. Perintah *terminal git add .*

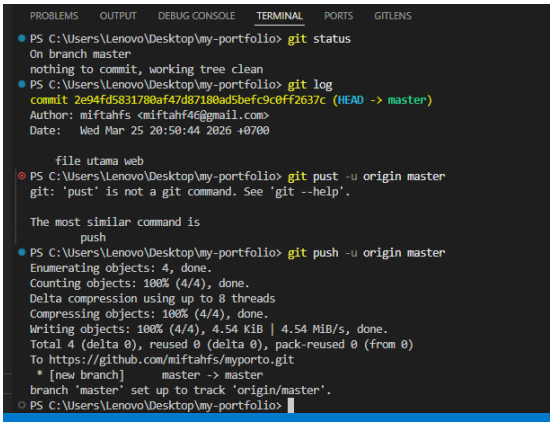


Gambar 10. Perintah *terminal git commit*

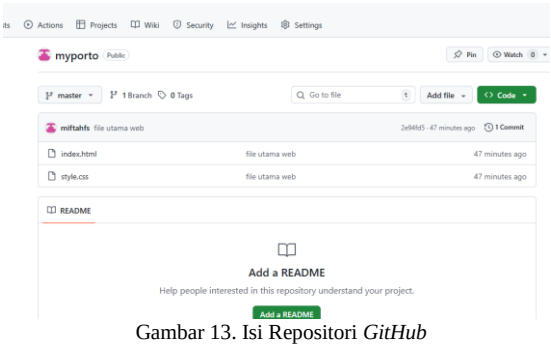
Langkah selanjutnya perintah *git push*. Inti dari penelitian ini adalah menganalisis efektivitas perintah *git push* sebagai mekanisme *deployment*. Berbeda dengan metode konvensional yang memerlukan klien *FTP* seperti *FileZilla*, proses *deployment* pada penelitian ini dilakukan dengan menghubungkan repositori lokal ke repositori *remote* di *GitHub*.

Uji coba dimulai dengan skenario pengiriman kode awal *(initial deployment)* ke *server GitHub*. Proses ini diawali dengan pembuatan repositori baru pada laman *github.com* untuk menghasilkan *remote URL (origin)*, yang diperlukan sebagai tujuan pengiriman data dari repositori lokal. Setelah konfigurasi *remote* berhasil, perintah *git push* dieksekusi untuk mentransfer kode. Berikut adalah hasil observasi dari proses *git push* yang telah dilakukan, dapat dilihat pada gambar 11 hingga 16.

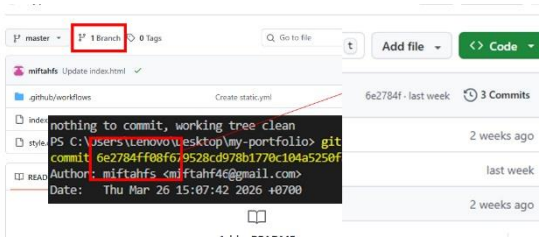
Gambar 11. *New Repository GitHub Cloud*



Gambar 12. Perintah *git push*



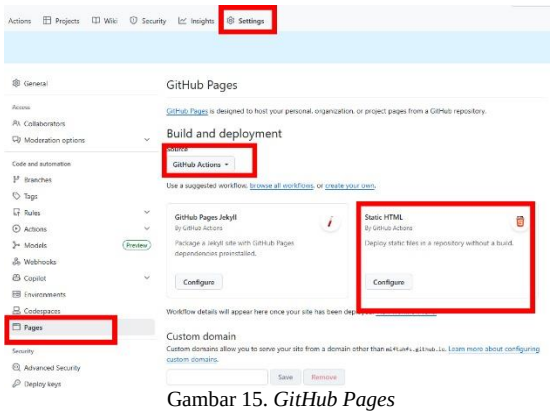
Gambar 13. Isi Repositori *GitHub*



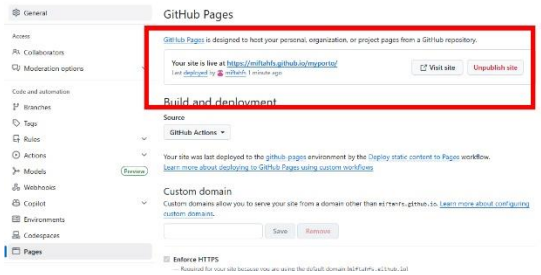
Gambar 14. File yang sudah di *commit*

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN :**2655-8238**
P-ISSN : **2964-2132**



Gambar 15. *GitHub Pages*



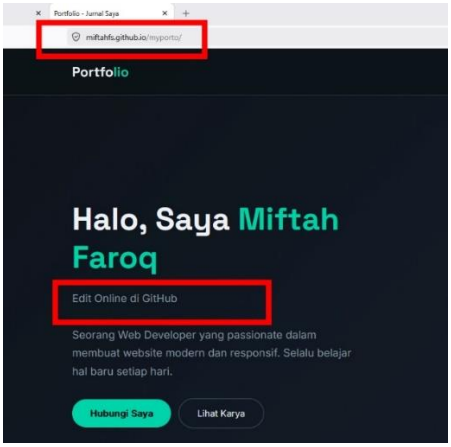
Gambar 16. *GitHub Pages Domain*

Setelah perintah *git push* berhasil dieksekusi, fitur *GitHub Pages* diaktifkan pada pengaturan repositori. Hasil yang didapatkan adalah *website* statis langsung *live* dan dapat diakses publik melalui URL https://miftahfs.github.io/myporto/ . Analisis menunjukkan bahwa *deployment* menggunakan *GitHub Pages* memiliki *latency* (waktu tunda) yang sangat rendah. Perubahan kode di lokal terlihat langsung di *website* publik hanya dalam hitungan detik setelah proses *push* selesai. Hal ini membuktikan bahwa *GitHub* efektif sebagai media *deployment* yang cepat dan otomatis.

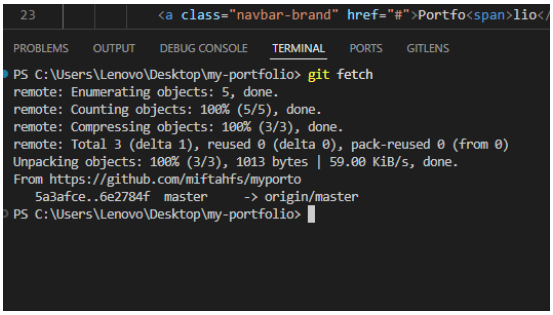
Aspek krusial lainnya dalam pengelolaan *website* adalah sinkronisasi data. Uji coba dilakukan untuk menguji kemampuan GitHub dalam menjaga konsistensi kode antara repositori *cloud* dan repositori lokal menggunakan perintah *git pull*. Skenario uji coba mensimulasikan adanya perubahan kode yang dilakukan di luar komputer lokal (misalnya perbaikan *bug minor* yang dilakukan langsung *via web browser*) dapat dilihat pada gambar 16 dan 17.

Setelah perubahan dilakukan di sisi remote, perintah *git pull* dieksekusi pada komputer lokal. Hasil uji coba menunjukkan bahwa Git secara otomatis mendeteksi adanya perbedaan versi dan mengunduh commit terbaru dari *server*, yang diperlihatkan pada gambar 21.

Gambar 16. *GitHub Editor*

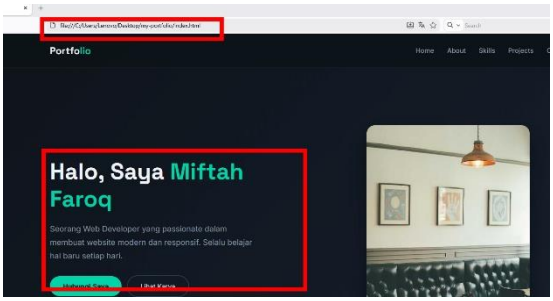


Gambar 17. Halaman web setelah *diedit online*



Gambar 18. Perintah *Git Fetch*

Gambar 18 proses *git fetch* untuk melakukan pemeriksaan *file* atau berkas apa saja yang terdapat perubahan, namun belum ditarik sepenuhnya ke *working directory* atau *folder* lokal repositori.



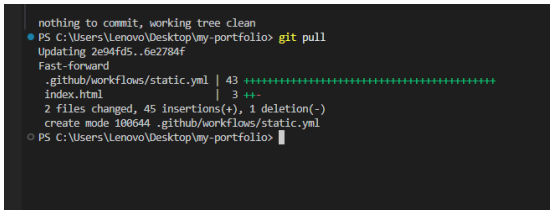
Gambar 19. Halaman Front End Lokal

Pada gambar 19 ditampilkan halaman *front-end*, namun dalam posisi lokal *host*, terlihat bahwa ada perbedaan dengan halaman yang ada

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN :2655-8238
P-ISSN : 2964-2132

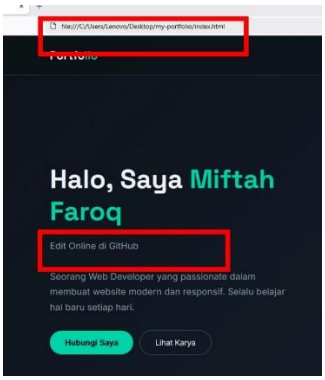
di *server*, yaitu halaman tidak dalam keadaan ter-*update*.

Gambar 20. *Git Status*



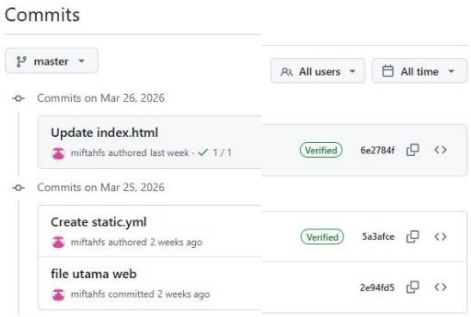
Gambar 21. *Git Pull*

Pada gambar 20 digunakan perintah git status, pesan tertulis bahwa *file* lokal tertinggal 2 *commit* terhadap *file remote* repositori, sehingga harus dilakukan perintah *get pull*, untuk mensinkronkan antara *file lokal* dan *file remote repositori* yang dapat dilihat pada gambar 21.



Gambar 22. Web lokal sudah terbaharui

Gambar 22 halaman antarmuka web sudah sama dengan yang ada di cloud *(GitHub Repository)*. Untuk history perubahan dapat dilihat pada folder GitHub, terdapat 3 *log commit* secara sederhana, file-file sebelumnya tetap terekam *(record)*, sehingga jika ingin kembali menggunakan *file* yang terdahulu, tinggal mengubah cabang *(branch)* untuk dapat digunakan sesuai dengan kebutuhan *development*, lihat gambar 23.



Gambar 23. Commit di GitHub Cloud

Tidak ditemukan konflik *(merge conflict)* selama proses ini, yang menandakan bahwa sinkronisasi berjalan mulus. Berkas *index.html* di komputer lokal secara otomatis ter-*update* sesuai dengan versi terbaru di *server*. Fitur ini memberikan keamanan data yang tinggi, di mana pengembang tidak perlu khawatir kehilangan progress pekerjaan atau menggunakan versi kode yang sudah usang.

3.3. Pembahasan

Berdasarkan hasil uji coba implementasi di atas, dapat dibahas efektivitas penggunaan platform GitHub dibandingkan dengan penelitian terdahulu dan metode konvensional.

Pertama, mengacu pada penelitian terdahulu mengenai pengembangan aplikasi web [3] dan rancang bangun *layout web* [2], fokus utama penelitian tersebut tertuju pada aspek fungsionalitas dan estetika tampilan *(front-end)*. Namun, proses *deployment* pada penelitian-penelitian tersebut umumnya masih dilakukan secara manual atau menggunakan layanan *hosting* terpisah yang memerlukan biaya sewa dan konfigurasi *server* yang rumit. Penelitian ini mengisi kesenjangan tersebut dengan menawarkan solusi *all-in-one* di mana *GitHub* berfungsi ganda sebagai penyimpan kode sekaligus *server* produksi.

Kedua, analisis biaya menunjukkan efisiensi signifikan. Tabel 3 menyajikan perbandingan biaya operasional antara metode konvensional dengan metode *GitHub*.

Tabel 3. Perbandingan Efisiensi Metode Deployment

Parameter	Metode Konvensional	Metode GitHub Pages
Biaya Hosting	Rp 200.000 – Rp 500.000/tahun	Rp 0 (Gratis)
Proses Deployment	*Manual Upload (FTP/File Manager)*	Otomatis (**git push**)
Kontrol Versi	Tidak tersedia (manual)	Tersedia (Git History)
Keamanan Data	Rentan overwrite tanpa jejak	Terlacak penuh (Commit Log)

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 5 No. 1 Januari 2023 Hal. 36-41
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN :**2655-8238**
P-ISSN : **2964-2132**

Sumber: Miftah Faroq

Tabel 3 menunjukkan bahwa metode *GitHub* menghasilkan *zero-cost deployment*. Hal ini sangat menguntungkan bagi pengembang pemula, mahasiswa, atau UMKM dalam membangun *online presence* tanpa beban biaya infrastruktur.

Ketiga, dari sisi keamanan, penggunaan perintah *git push* dan *git pull* menciptakan jejak audit *(audit trail)* berupa *log commit*. Berbeda dengan metode *FTP* tradisional di mana kesalahan penimpaan berkas bersifat permanen dan merusak, sistem *Git* memungkinkan *rollback* ke versi sebelumnya. Kesimpulan sementara dari pembahasan ini adalah bahwa pemanfaatan *platform GitHub* telah berhasil mengintegrasikan proses pengembangan dan publikasi *website* statis menjadi satu alur kerja yang efisien, aman, dan hemat biaya.

KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan mengenai analisis pemanfaatan platform *GitHub* pada pengembangan *website* statis *HTML*, dapat disimpulkan bahwa *platform* ini terbukti efektif berfungsi sebagai sistem kontrol versi sekaligus media *deployment*. Implementasi perintah dasar *Git (commit, push, pull)* berhasil menciptakan dokumentasi riwayat perubahan yang terstruktur, sehingga mengeliminasi risiko kehilangan data akibat penimpaan berkas *(overwriting)* yang sering terjadi pada metode manual. Fitur *GitHub Pages* yang terintegrasi langsung dengan repositori memberikan solusi *deployment* yang cepat dan otomatis, di mana proses publikasi *website* dapat dilakukan secara instan melalui perintah *git push* tanpa biaya *(zero-cost)* dan tanpa memerlukan konfigurasi *FTP* manual. Dengan demikian, penggunaan *GitHub* sebagai ekosistem *all-in-one* berhasil meningkatkan efisiensi kerja, menjadikan alur kerja pengembangan *website* statis lebih ringkas karena proses penyimpanan versi dan publikasi dapat dilakukan dalam satu *platform* yang sama.

Berdasarkan kesimpulan tersebut, penelitian ini memberikan beberapa saran untuk pengembangan dan penelitian selanjutnya. Penelitian selanjutnya disarankan untuk mengkaji perintah dan fungsi lain dari *Git*, mengingat penelitian ini terbatas pada perintah dasar, penelitian lanjutan perlu mengeksplorasi fitur manajemen repositori yang lebih kompleks, meliputi penggunaan *checkout* dan *switch* untuk navigasi cabang, *restore* dan *reset* untuk manajemen pembatalan perubahan, serta operasi *rename* dan *delete* untuk pengelolaan struktur proyek yang lebih dinamis.

UCAPAN TERIMAKASIH

Penulis mengucapkan puji syukur ke hadirat Tuhan Yang Maha Esa atas terselesaikannya artikel ini. Terima kasih disampaikan kepada Universitas Bina Sarana Informatika, khususnya Program Studi Teknologi Informasi, Fakultas Teknik dan Informatika, atas dukungan fasilitas selama penelitian. Terima kasih juga kepada dosen dan rekan sejawat yang telah memberikan masukan dalam penyusunan artikel ini. Semoga artikel ini bermanfaat bagi pengembangan ilmu pengetahuan di bidang teknologi informasi dan pengembangan web.

DAFTAR PUSTAKA

[1] M. F. Santoso, “MF Optimization of Delivery Management Through a Web-Based Monitoring Application Using PHP and Bootstrap,” *J. Teknol. Dan Sist. Inf. Bisnis*, vol. 7, no. 3, pp. 355–363, 2025, doi: 10.47233/jteksis.v7i3.1686.

[2] V. No, O. Hal, and M. F. Santoso, “Perbandingan Efektivitas Bootstrap dan Tailwind CSS dalam Pengembangan UI Web Responsif,” vol. 7, no. 4, pp. 489–497, 2025.

[3] M. Faroq, “TEKNIK SINGLE PAGE APPLICATION (SPA) LAYOUT WEB DENGAN MENGGUNAKAN REACT JS DAN BOOTSTRAP,” 2021.

[4] P. Pembelajaran, P. Di, K. Viiib, and S. M. P. Negeri, “PENGEMBANGAN E-MODUL CANVA TEMA 7 SUBTEMA 2 PADA MATA PELAJARAN IPA MATERI MACAM-MACAM GAYA UNTUK SISWA KELAS IV,” vol. 5, no. 2, pp. 344–350, 2025.

[5] T. J. Riasinir and Widyasari, “Pemanfaatan Framework Bootstrap,” *J. Enter*, vol. 2, no. 5, pp. 346–355, 2019.

[6] M. F. Santoso, “Jurnal Media Informatika [JUMIN] Implementasi Teknologi Frontend Modern pada Website Yellowweb: Kolaborasi Booostrap 5 Framework dan jQuery,” *J. Media Inform.*, vol. 6, no. 2, pp. 873–883, 2025, [Online]. Available: https://www.researchgate.net/publication/388654337_Implementasi_Teknologi_Frontend_Modern_pada_Website_Yellowweb_Kolaborasi_Boostrap_5_Framework_dan_jQuery

[7] Erlangga Fajar Ramadhan, Ardin Ariantana Putra, Moh. Teguh Purwanto, and Ardhi Feisal Wijayanto, “Evaluasi UI & UX Website Tulungagung.go.id Menggunakan Metode Heuristic,” vol. 9, pp. 1489–1498, 2025.

[8] Hendra Maulana *et al.*, “Perancangan Sistem Informasi Desa Berbasis Website di Desa Pandean Kecamatan Gondang Kabupaten Nganjuk,” *J. Penelit. Sist. Inf.*, vol. 1, no. 2, pp. 28–48, 2023, doi: 10.54066/jpsi.v1i2.472.

[9] M. Sudur, N. Azizah, and R. S. Razaqi, “Implementasi Dan Perancangan Sistem Informasi Akademik Berbasis Framework Sublime Text,” vol. 4, no. 2, pp. 976–989, 2025.

[10] E. F. Sari, “Penerapan Github Sebagai Media E-Learning Untuk Mengetahui Keefektifan Kolaborasi Project Pada Mata Pelajaran Pemrograman Web Dan Perangkat Bergerak Di Smk Negeri 2 Surabaya,” *J. IT-EDU*, vol. 06, no. 02, pp. 14–22, 2021.

[11] I. Maulana, R. Umar, and A. Yudhana, “Implementasi

E-ISSN :**2655-8238**
P-ISSN : **2964-2132**

Deployment Layanan Website Menggunakan Kubernetes Dengan Ci/Cd Jenkins," *J. SAINTIKOM (Jurnal Sains Manaj. Inform. dan Komputer)*, vol. 23, no. 2, pp. 290–296, 2024, doi: 10.53513/jis.v23i2.9992.

[12] Bagus Syafiq Faqihuddin, Acep Taryana, and Mulki Indana Zulfa, "Pengembangan Application Programming Interface (Api) Aplikasi Pendeteksi Penyakit Tanaman Menggunakan Metode Devops," *J. SINTA Sist. Inf. dan Teknol. Komputasi*, vol. 2, no. 2, pp. 66–75, 2025, doi: 10.61124/sinta.v2i2.42.

[13] R. Setiabudi, "Analisis Efektifitas Ci / Cd Dan Manual (Tradisional) Dalam Pengembangan Website Rakyatweb.Com," *J. Ismetek*, vol. 17, no. 2, pp. 29–36, 2024.

[14] M. Azmi, Y. Sonatha, I. Rahmayuni, M. R. Dewi, and S. O. Kirana, "Penerapan GitHub sebagai Media Kolaborasi untuk Meningkatkan Keterampilan Kerja Tim Siswa SMK," *Suluah Bendang J. Ilm. Pengabdi. Kpd. Masy.*, vol. 24, no. 3, p. 189, 2024, doi: 10.24036/sb.05950.

[15] I. Azimi, "Pengaruh Penggunaan Version Control System Terhadap Proses Belajar Pemrograman Mahasiswa," *J. Ilm. Edutic Pendidik. dan Inform.*, vol. 5, no. 2, pp. 81–87, 2019, [Online]. Available: https://journal.trunojoyo.ac.id/edutic/article/view/5100

[16] N. Fadilah, R. Hartono, and D. Syahrul Anwar, "Pengembangan Prototype Arsitektur Ci/Cd Pada Sistem Informasi Manajemen Terintegrasi Universitas Perjuangan Menggunakan Jenkins," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 9, no. 4, pp. 6873–6881, 2025, doi: 10.36040/jati.v9i4.13837.

[17] Fadira Az-zahra and Syafira Putri Yuliadi, "Analisis Perbandingan Kinerja Website Statis dan Dinamis dalam Optimalisasi Layanan Informasi Digital," *J. Sade. Publ. Ilmu Pendidikan, Pembelajaran dan Ilmu Sos.*, vol. 3, no. 4, pp. 91–100, 2025, doi: 10.61132/sadewa.v3i4.2430.

[18] M. Fauzan, A. Fiade, and F. E. M. A, "Infrastruktur Private Cloud Dengan Openstack," *J. Pseudocode, Vol. IV Nomor 2, Sept. 2017, ISSN 2355-5920*, pp. 180–189, 2017.

[19] L. D. Wati, M. Ariska, S. M. Siahaan, L. Marlina, and Y. Anwar, "DASHBOARD INTERAKTIF BERBASIS PLATFORM GITHUB KOMPUTASI," pp. 87–102.

[20] C. Widya *et al.*, "Analisis Risiko Keamanan Siber Website Peken Surabaya Menggunakan Standar ISO 27005 : 2019 dan OWASP ZAP Universitas Pembangunan Nasional Veteran Jawa Timur , Indonesia kerentanan teknis pada website , seperti Cross-Site Scripting (XSS), SQL Injection ," *J. Teknol. dan Sist. Inf.*, 2025.