Implementasi Sistem Manajemen Pengguna dengan Integrasi Next.js dan ASP.NET

Christopher Hermawan[#1], Adelia[*2]

[#]*Program Studi Sistem Informasi Fakultas Teknologi Informasi, Universitas Kristen Maranatha*
Jl. Surya Sumantri No.65, Bandung 40164
[1]2073003@maranatha.ac.id
[2]adelia@it.maranatha.edu

Abstract — ***This report revolves around the project focused on developing a user management web page during the internship at biro-f, PT XYZ. The report covers various topics, including the introduction, information about the institute and job description, theoretical foundations, project outcomes, and the conclusion with suggestions. The primary task involved the creation of a user management web page integrated into the Content Management System (CMS) website. The user management web page serves the purpose of controlling user access within the CMS website. Key features of the user management web page include viewing and searching for specific user data, creating new user profiles, updating user information, deleting user accounts, and deactivating user accounts. The development of the user management web page utilized the Next.js framework for front-end development, while the required APIs were created and developed using ASP.NET MVC 4 as the backend framework. The end result of this project is a seamlessly integrated User Management web page that effectively interacts with the underlying API.***

Keywords— ***ASP.NET, CMS, Framework, Next.js, web page***

I. PENDAHULUAN

A. Latar Belakang

PT XYZ adalah bank swasta terbesar di Indonesia. Bank XYZ didirikan pada 21 Februari 1957. Bank XYZ yang bergerak dibidang perbankan dan sudah dipercaya oleh masyarakat Indonesia dan dapat berkembang lebih dari setengah abad. Bank XYZ pada saat ini di tahun 2023, terus melakukan inovasi dengan membuat *website* atau aplikasi yang dapat memudahkan nasabah untuk dapat mengakses dan melakukan transaksi atau sekadar mencari informasi seputar keuangan, promosi dan layanan finansial lainnya yang ditawarkan oleh XYZ.

Bank XYZ memiliki departemen IT sendiri. Pengembangan *website* XYZ juga dilakukan di salah satu biro yaitu, biro F. Pengembangan *website* menggunakan berbagai macam *framework*, seperti salah satunya adalah ASP.NET yang memanfaatkan bahasa pemrograman C# atau dikenali juga dengan C-Sharp. Tetapi tidak hanya ASP.NET saja, karena pada biro F sering melakukan eksplorasi untuk teknologi baru atau yang belum pernah dipakai pada proyek. Teknologi yang dimaksud adalah seperti *framework back-end* seperti Spring Boot dengan bahasa pemrograman Java ataupun pengembangan dengan *framework front-end* Next.js dan juga Angular sebagai *framework front-end*.

Pengembangan *website* XYZ juga sudah memanfaatkan Google Cloud. Sudah ada 1 *website* yang di deploy biro F pada Google Cloud, yaitu sebuah *website* yang berisikan informasi promo untuk merayakan ulang tahun XYZ dan ada juga promo lebaran dan sebagainya.

Berhubungan dengan *website* promo ini, sedang dikembangkan juga sebuah CMS atau *content management system* internal dengan tujuan untuk mempermudah mengatur isi atau konten dari *website* promo tersebut yang *user* atau penggunannya adalah dari pihak internal XYZ. Sehingga dengan adanya CMS ini dapat membuat perubahan pada *website* promo tidak lagi harus dilakukan oleh *programmer* tetapi bisa dengan mudah dilakukan oleh *user* atau pengguna dari divisi terkait atau lebih efisien. Tentunya pada *website* CMS yang sedang dikembangkan ini terdapat sebuah halaman yang dapat digunakan untuk menambahkan dan mengubah *user* berserta dengan *role* dari *user* tersebut supaya dapat akses untuk apa yang user tersebut dapat lakukan pada *website* CMS tersebut.

B. Rumusan Masalah

Berikut merupakan rumusan masalah pada pembahasan, yaitu:

1. Bagaimana pengembangan tampilan halaman web *user management* sesuai dengan permintaan dari instansi?
2. Bagaimana pengembangan API sesuai dengan permintaan dari instansi?
3. Bagaimana mengintegrasikan API dengan aplikasi web *user management*?

C. Tujuan Pembahasan

Berikut merupakan tujuan pembahasan laporan, yaitu:

1. Dapat mengembangkan tampilan halaman web *user management* sesuai dengan permintaan intansi dengan menggunakan *framework* Next.js.
2. Dapat mengembangkan API dengan menggunakan *framework* ASP.NET dan SQL Server.
3. Dapat mengintegrasikan API yang dikembangkan dengan ASP.NET dan SQL Server, dengan web *page user management* yang dikembangkan dengan Next.js.

D. Ruang Lingkup

Ruang lingkup pada pembahasan laporan, yaitu:

1. Bentuk aplikasi merupakan *website*.
2. Pengguna aplikasi atau user adalah tim Markom atau marketing komunikasi.
3. Range masa kerja adalah 1 tahun (10 Juli 2023 – 9 Juli 2024).
4. Keterlibatan pengembangan aplikasi (Sampai sidang tugas akhir dilakukan di tanggal 18 Januari 2024), dari tahapan *requirement, development, code review, revision,* dan *testing*.
5. Fungsionalitas: Menambahkan data *user* (*Create*), mendapatkan data user (*Read*), mengubah data user (Update), menghapus data user (*Delete*).
6. ASP.NET: Pembuatan proyek ASP.NET 5 dan konfigurasi, koneksi database dengan SQL Server, pengembangan API dengan ASP.NET.
7. Next.js: Pengembangan proyek dengan Next.js dan konfigurasi, pemanfaatan library CSS MUI pada Next.js
8. SQL Server: Stored Procedure, User-Defined Table Type, Database.

II. DESKRIPSI PEKERJAAN DAN TANGGUNG JAWAB

Pada tahapan pertama adalah pengembangan user interface atau UI yang terlebih dahulu yang dikerjakan dengan menggunakan framework front-end Next.js. Bahasa pemrograman yang digunakan adalah JavaScript dan juga menggunakan framework CSS MUI. Pada tahapan ini yang dilakukan adalah mengembangkan tampilan dari model tampilan yang telah dibuat pada Figma. Proses yang dilakukan adalah membuat tampilan yang identik dengan model tampilan tersebut dengan framework Next.js.

Pada tahapan kedua adalah melanjutkan dengan pengembangan API yang dilakukan dengan framework ASP.NET. Bahasa pemrograman yang digunakan adalah C# dan juga dengan koneksi RDMBS SQL Server. Pada tahapan ini yang dilakukan adalah mengembangkan API untuk kebutuhan proses dari client ke server. API yang dikembangkan berupa CRUD yang dikembangkan juga dengan stored procedure dari SQL Server.

III. LANDASAN TEORI

A. Website

Situs web atau *website* merupakan salah satu bentuk aplikasi yang dapat diakses secara lokal ataupun daring pada *hardware* seperti komputer ataupun *smartphone* yang masih banyak dipakai oleh banyak individu maupun organisasi untuk berbagai macam tujuan, seperti media sosial, melakukan promosi, penginputan data, ataupun media permainan daring atau *online*. Kemudian kesimpulan yang didapatkan dari Niagahoster mengenai apa itu situs web atau *website* adalah serangkaian halaman web yang berisi informasi yang terhubung satu sama lain yang dapat diakses dengan koneksi internet [1]. Pada *website* sendiri terdapat beberapa hal lainnya yang sebenarnya publik dapat lihat, seperti URL dan juga *cookies* yang mungkin sering ditemui ketika mengakses beberapa web yang dapat dilihat juga dengan akses *inspect* pada halaman web.

1. Uniform Resource Locator (URL)

Uniform Resource Locator atau yang lebih dikenal luas publik sebagai URL adalah alamat digunakan untuk mengakses web pada *browser*. Pengertian lainnya yang didapatkan dari halaman web resmi developer.mozilla, URL adalah mekanisme yang digunakan oleh *browser* untuk mendapatkan sumber daya web yang telah di *publish* [2].

2. Cookies

Cookie atau *cookies* memiliki beberapa pengertian dari berbagai sumber, seperti dari *official support* google yang menyatakan *cookie* adalah *file* yang dibuat oleh situs yang *user* buka dan dapat menyimpan informasi kunjungan *user* yang digunakan untuk meningkatkan pengalaman *user* pada *website* yang dikunjungi [3].

B. Internet Information Services (IIS)

Pada pengembangan aplikasi *website* diperlukan web server yang menjadi penghubung antara server dan client. Web server yang dimaksud disini adalah *Internet Information Services* atau singkatnya IIS adalah web server satu-satunya yang dapat

digunakan pada user sistem operasi Windows yang diciptakan oleh Microsoft. Berdasarkan deskripsi dari IIS, *Internet Information Services* (IIS) adalah web server terkelola, flexibel, dan aman untuk hosting apapun pada web [4].

C. Back-end

Back-end dalam konteks pengembangan aplikasi, sesuai dengan namanya, yaitu tempat atau proses aplikasi berjalan pada bagian belakang atau tidak berhubungan secara langsung dengan *user* atau penggunannya. Sesuai seperti artikel dari Universitas Esa Unggul yang menyimpulkan bahwa *back-end* berurusan dengan data dan proses di balik layar, bertanggung jawab dalam pengaturan data dan berkomunikasi dengan *front-end* untuk mengirim informasi ke tampilan web dan juga menerima informasi dari sisi client [5].

D. Front-end

Bersumber dari Jenius co.create menyatakan *front-end developer* adalah seseorang yang membangun sebuah *website* dengan fokus pada tampilan depan *website* yang dilihat oleh user atau semua hal yang berhubungan dengan *client side* [6]. Dapat disimpulkan dalam konteks pengembangan aplikasi bahwa front-end adalah proses pengembangan aplikasi yang berhubungan secara langsung dengan *user* atau dapat berinteraksi dan dilihat secara langsung oleh *user*.

E. Application Programming Interface (API)

Application Programming interface atau yang lebih sering disebut juga API, berdasarkan pengertian dari Subramanian, Siddharth, API adalah mekanisme abstrak kuat yang memberikan akses fungsionalitas kompleks, sering melibatkan pemanggilan melalui kelas dan metode sederhana. API memungkinkan pengembang untuk menggunakan fungsi-fungsi canggih tanpa perlu memodifikasi atau memahami detail implementasi yang mendasarinya [7].

F. Framework

Pengertian *framework* sesuai dengan namanya, yaitu kerangka kerja dan pada konteks pengembangan aplikasi, *framework* digunakan untuk membantu *developer* untuk menuliskan program dengan struktur sesuai dengan *framework* yang digunakan.

Kemudian fungsi *framework* menurut Kasih Purwantini menyatakan fungsi framework untuk developer adalah membuat kode program lebih terstruktur, membantu kinerja dari developer, meningkatkan keamanan aplikasi, pemeliharan dan dokumentasi lebih mudah, dan mempercepat proses pembuatan aplikasi [8].

1. Dotnet (.NET)

Bersumber dari Microsoft menyatakan .NET adalah *platform open-source* yang gratis, lintas *platform* yang dapat membangun banyak jenis aplikasi. Dengan .NET, terdapat beberapa pilihan bahasa program, *editor*, *library* untuk menciptakan program web, *desktop*, *mobile*, *games*, IoT, dan lainnya [9].

2. ASP.NET

ASP.NET bagian dari kumpulan teknologi dari *framework* .NET untuk pengembangan web dengan konsep OOP (*Object – Oriented Programming*) *multiplatform* dan terdapat konfigurasi untuk *cloud*.

ASP.NET merupakan *framework open source* dan memiliki komunitas yang besar dan masih sangat populer, sehingga masih terus berkembang dan menerima pembaharuan dan *support* dari komunitasnya. Berikut merupakan beberapa keunggulan ASP.NET seperti:

- Kode open source
- Multi-platform
- Kemampuan untuk hosting dengan IIS dengan mudah
- Ekstensi dalam paket NuGet
- Terdapat konfigurasi untuk dukungan pengembangan web cloud

3. Next.js

Next.js adalah *framework* React fleksibel yang memberikan *developer* kebebasan dalam mengembangkan blok bangunan untuk membuat aplikasi web yang cepat.

Pengembangan *front-end* dengan *framework* Next.js menurut Ronny Jubhari Phie Joarno et al. (2022) memiliki beberapa keunggulan yang dapat dipertimbangkan untuk membuat aplikasi web, seperti *server-side rendering* atau me-*render* halaman di

sisi server sebelum mengirimkannya ke *browser* pengguna, meningkatkan kecepatan pemuatan halaman dengan itu membuat performa *website* lebih baik dan juga *SEO friendly*.

G. Material UI (MUI)

Material UI adalah komponen dari *library* React *open-source* yang mengimplementasi Google Material Design. Material UI memiliki keunggulan pada komponen siap pakai dengan beberapa kustomisasi dan juga dokumentasinya yang kuat

H. SQL Server

Pengertian yang didapatkan langsung dari Microsoft menyatakan SQL Server adalah *relational database management system* (RDBMS) yang dikembangkan oleh Microsoft. SQL Server memiliki GUI yang membuatnya relatif lebih mudah dioperasikan dan mudah diintegrasikan [10].

I. Postman

Postman menurut Rececca Hyams dan Tamara Pilko adalah aplikasi yang diciptakan sebagai alat untuk membantu proses *testing* API. Postman memiliki fitur yang mendukung untuk pengembangan API dan penggunaan API [11].

Postman memiliki tampilan yang dapat membantu penggunanya untuk melihat, mengirim, dan berinteraksi dengan *request* API. Kemudian dengan Postman *developer* dapat dengan mudah melihat apakah API yang telah dikembangkan atau yang dipanggil dapat bekerja atau tidak dan melihat *response* yang dikembalikan pada tampilan Postman.

J. JSON Web Token (JWT)

JSON Web Token dipakai saat pengembangan *website* dengan fungsi untuk otentikasi dan otorisasi pengguna *website* dan merupakan salah satu aspek keamanan yang dapat dipakai saat pengembangan *website*. Pengertian JSON Web Token menurut Jones M, et al. 2015 adalah sarana yang ringkas dan aman untuk merepresentasikan klaim untuk ditransfer antar 2 pihak. Klaim pada JWT dijadikan objek JSON. Klaim dapat berupa data-data yang akan dijadikan klaim, seperti contohnya iduser, email, nama lengkap yang kemudian dijadikan objek JSON [12].

K. Server-Side Rendering (SSR)

Server-side rendering adalah sebuah metode dalam men-*handle request user* pada server, kemudian *request* akan diproses pada server dan server kirim kembali ke *client* [13].

Metode ini pada penggunaannya memiliki kelebihan, seperti dapat memuat halaman awal lebih cepat karena mengirimkan *pre-rendered* HTML dengan konten yang mengurangi waktu yang dibutuhkan untuk me-*render* konten di sisi *client*. Namun dengan begitu metode SSR juga membuat beban server lebih berat karena harus menghasilkan HTML setiap *request* terjadi.

L. Client-Side Rendering (CSR)

Menurut Herman dan Geovanny, *client-side rendering* adalah metode me-*render* situs web pada *browser* menggunakan JavaScript, dan *browser* hanya mendapatkan konten dari dokumen HTML dan *file* JS yang akan di-*render* [14].

IV. HASIL PEKERJAAN & PEMBAHASAN

Proyek atau tugas yang dikerjakan adalah untuk membuat halaman web *user management* dengan *framework front-end* Next.js dan *framework back-end* ASP.NET. Berikut merupakan urutan tahapan implementasi *project* pembuatan halaman web *user management*.

A. Tahapan Persiapan

Tahapan persiapan adalah hal-hal yang perlu dilakukan pertama kali untuk dapat memulai proses pengembangan halaman *user management*. Tahapan persiapan meliputi instalasi *software* yang dibutuhkan untuk keseluruhan proses *development* berlangsung. Kemudian persiapan juga berupa eksplorasi *framework* yang dimana seluruh proses pengembangan terjadi.

1. User Requirement

Kebutuhan yang diperlukan untuk halaman web *user management* adalah sebagai halaman yang bisa digunakan untuk pengelolaan pengguna CMS. Dapat memberikan akses berdasarkan *role* atau peran secara mudah dan cepat. Berikut beberapa fungsi yang dibutuhkan ada pada halaman web *user management*, seperti:

- Sistem menyediakan fungsi untuk penambahan pengguna CMS (*Create*).
- Sistem dapat menampilkan data pengguna CMS yang sudah terdaftar, serta terdapat fitur pencarian pengguna CMS (*Read*).
- Dapat melakukan pengelolaan data pengguna CMS, seperti perubahan nama, email, udomain, password, serta melakukan penghapusan data user dengan *soft delete* (*Update*).

- Sistem dapat memberikan hak akses kepada user CMS yang hak aksesnya bisa disesuaikan berdasarkan *role* pengguna.
- Memiliki UI (*User Interface*) yang ramah pengguna, intuitif dan responsif.
- Terdapat pencatatan aktivitas (log) yang mencatat seluruh aktivitas yang terjadi pada halaman web *user management*, seperti *login, logout, update, create*, dan *get data*.
- Hanya *super admin* yang dapat mengakses API *user management* dan mengakses halaman *user management*.

2. Instalasi Software

Terdapat beberapa kebutuhan untuk melakukan instalasi beberapa *software* yang dibutuhkan, seperti:

- Visual Studio untuk pengerjaan *back-end* dengan ASP.NET MVC 4
- Visual Studio Code untuk pengerjaan *front-end* dengan Next.js
- Node.js untuk *download* next.js, MUI, dan *run project* dengan npm (*Node Package Manage*r) atau sistem management paket.
- *Internet information service* (IIS) sebagai web server
- Postman untuk *testing* API.
- *Relational database management system* (RDBMS) dengan SQL Server 2019 Kemudian juga instalasi paket di dalam *software*, seperti:
- Instalasi ASP.NET pada Visual Studio
- Instalasi Next.js pada terminal atau CMD dengan npm
- Instalasi MUI (*library* React untuk UI) pada terminal Visual Studio Code

3. Pengaturan IIS

Pada tahapan persiapan ini diperlukan juga untuk melakukan pengaturan ISS untuk halaman web *user management*. Pada saat *add website* akan terdapat pilihan untuk mengatur lokasi *folder* yang akan dipakai untuk menyimpan kode *back-end* dari ASP.NET yang di *publish* dan kemudian terdapat juga pengaturan port serta *identity setting* pada *application pools* untuk pengaturan administrator.

B. Tahapan Pengembangan

Tahapan pengembangan adalah proses yang meliputi seluruh pengembangan pada sisi front-end dan sisi back-end. Kemudian ada proses integrasi API yang berperan penting menghubungkan *client* dan server. Selanjutnya pada tahapan implementasi ini, perlu juga dilakukan uji coba atau testing untuk fungsionalitas dari API dan juga validasi pada *front-end* dan *back-end*. Kemudian setelah atau selama testing dilakukan, ada juga proses adjustment atau penyesuaian dengan tujuan memperbaiki *error*, *bug*, dan penyesuaian variabel sebelum proses penyatuan kode atau merging. Proses *merging* dilakukan setelah *adjustment* selesai dilakukan, sehingga meminimalisir *error* atau ketidaksesuaian saat penyatuan *project* yang berisikan *folder* dengan *file* dan kode.

1. Pengerjaan Front-end

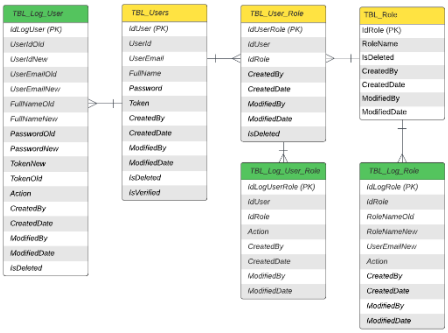
Pengerjaan dimulai dari sisi tampilan terlebih dahulu dengan Next.js. Pengerjaan dimulai dengan membuat komponen-komponen yang diperlukan untuk dapat membuat tampilan dengan model yang telah diberikan untuk ditiru pada Figma.

Kemudian pembuatan komponen pada Next.js dituliskan dalam kode HTML yang mirip dengan JSX dan dengan library CSS MUI. Telah diberikan juga source code project existing CMS dari tim yang bisa digunakan sebagai referensi kode untuk membantu proses pengerjaan.

2. Pengerjaan Back-end

Pengerjaan dilanjutkan dengan pembuatan API dengan ASP.NET dan SQL Server. Pengerjaan menggunakan struktur dari *source code* yang diberikan tim, sehingga struktur yang dipakai sama dan tidak perlu melakukan banyak konfigurasi lagi. Sehingga pengembangan API *user management* hanya butuh 3 *folder* penting, yaitu *model*, *repository*, dan *controller*. Terdapat tahapan yang yang dipakai pada pengembangan API untuk proyek ini, seperti:

- Tahap pertama adalah untuk membuat *folder controller*, *model* dan *repositories* kalau belum tersedia.
- Tahap kedua adalah membuat *stored procedure* dan transaksi *query* yang akan dipanggil pada kelas *repository*.
- Tahap ketiga adalah dengan membuat *class* model pada *folder* model dan isi dari class model ini diisi sesuai dengan data yang dibutuhkan dan sesuai juga dengan tabel dari *database* seperti yang dapat dilihat pada gambar.



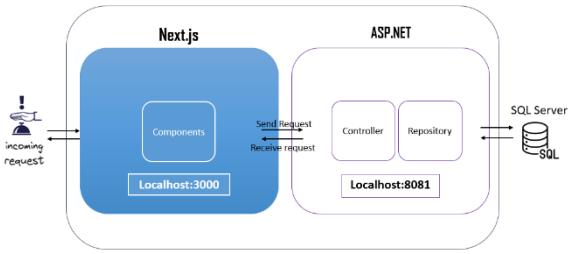
Gambar 1. ERD

Berikut merupakan gambar ERD (Entity Relationship Diagram). Terdapat total 6 tabel untuk *project user management* yang terdiri dari 3 tabel utama, yaitu tabel user, tabel user_role dan tabel *role* dan 3 tabel log untuk masing-masing tabel utama.

- Tahap keempat adalah membuat kelas *repository*. Pada kelas ini diisi prosedur pemanggilan *stored procedure* dan transaksi yang dibutuhkan pada kelas *controller*. Tentunya diperlukan koneksi sql yang tepat. Koneksi sql dapat diatur pada *file* web.config.
- Tahap kelima adalah pembuatan transaksi *query* pada *stored procedure* yang kemudian dipanggil pada *repository*.
- Tahapan keenam adalah membuat *controller* sesuai dari *stored procedure* yang ada pada *repository*.

1. Integrasi API

Pengerjaan proyek ini dilakukan secara pararel atau tidak dilakukan secara terpisah, dikarenakan saat proses integrasi biasanya perlu dilakukan lagi beberapa penyesuaian pada sisi *front-end* atau *back-end* nya. Integrasi API yang akan dilakukan adalah supaya Next.js dapat terhubung dengan API yang dibentuk pada ASP.NET. Sehingga sisi *front-end* dapat mengirimkan data ke *back-end* dan juga menerima data dari *back-end* ke *front-end*.

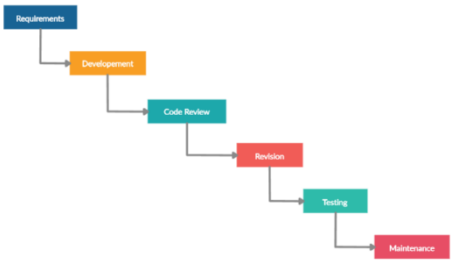


Gambar 2. Flow Architecture

Berikut merupakan gambaran dari alur yang akan terjadi. Proses dimulai saat ada interaksi dari *user* dari sisi *front-end* yang akan melakukan *request* API ke sisi server.

C. Tahapan Peninjauan

Tahap ini akan meninjau kembali proses pengembangan proyek halaman web *user management* dengan berbasis SDLC waterfall.



Gambar 3. SDLC Waterfall

Berikut adalah SDLC (*System Development Life Cycle*) pada pengembangan halaman web *user management*.

Sesuai dengan SDLC pengerjaan proyek diawali dengan *meeting* atau pertemuan yang meliputi pembahasan *user requirement.* Kemudian meliputi juga diskusi bersama tim untuk membahas adanya perubahan fitur, alur, tampilan.

Setelahnya dilanjutkan dengan proses *development. Development* yang dilakukan mengacu pada *requirements* yang sebelumnya sudah diproses. Pengembangan yang dilakukan meliputi pengembangan *front-end* dengan Next.js dan pengembangan *back-end* dengan ASP.NET dan SQL Server.

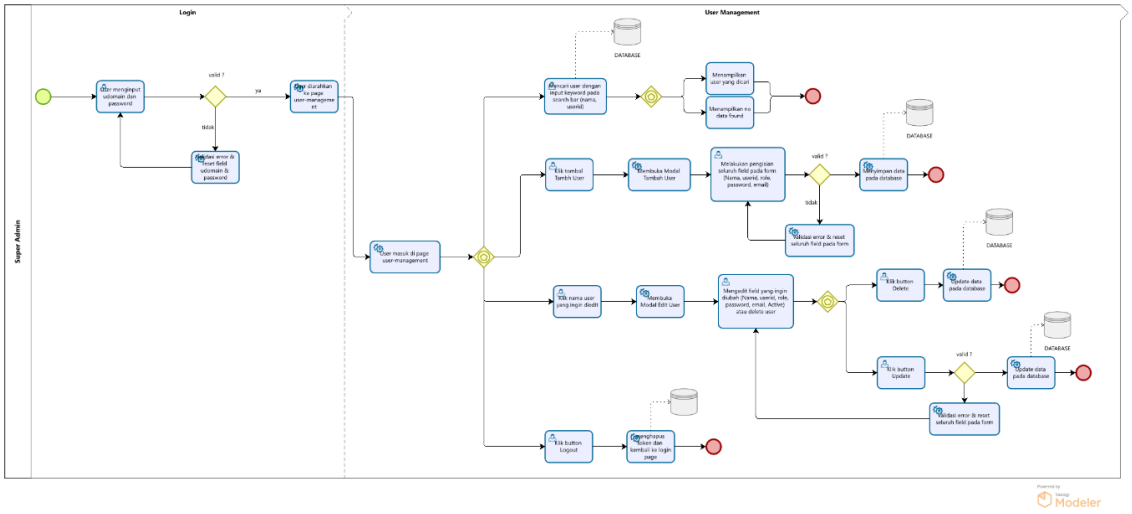
Kemudian terdapat *Code review*. Tujuan dari *code review* ini adalah untuk meningkatan kualitas kode program. Kode program yang berkualitas disini adalah kode program bebas dari malfungsi atau *bug*, dapat diproses dengan cepat dan tepat, dan juga rapih. Pada tahapan ini biasanya akan mendapatkan beberapa revisi dari PIC bersangkutan. PIC bersangkutan yang dimaksud adalah PIC bagian *back-end*, PIC bagian *front-end*, dan PIC bagian *code reviewer back-end*.

Code review dilakukan setelah API yang dibutuhkan selesai dikembangkan dan berfungsi saat dilakukan *testing* dari sisi *back-end* dan telah diintegrasikan dan berfungsi pada sisi *front-end*. Terdapat 3 PIC yang ikut serta dalam pengerjaan proyek dan mendapatkan kode terbaru, seperti PIC *code reviewer front-end* dan 2 PIC *code reviewer* PIC bagian *back-end*.

Kemudian hasil dari *code review* adalah revisi yang bisa berupa peningkatan kode, penyesuaian kode redundan, *bug fixing*, penambahan atau penyesuaian *requirement.* Pada tahap revisi terjadi banyak komunikasi atau diskusi yang dilakukan untuk memastikan proyek memiliki hasil akhir dengan kode yang baik dan sesuai dengan *requirement*.

Testing atau pengujian yang dimaksudkan adalah untuk menguji fungsionalitas, performa, integrasi baik itu dari sisi *back-end* dan juga *front-end*. Terdapat 2 pengujian yaitu SIT (*System Integration Testing*) dan UAT (*User Acceptance Testing*).

Maintenance atau pemeliharan dapat mencakup revisi ke tahap-tahap sebelumnya sesuai dengan kebutuhan proyek, contohnya seperti adanya *enhancement* yang akan dilakukan ataupun menemukan kesalahan yang tidak ditemukan sebelumnya.



Gambar 4. Proses Bisnis User Management

Pada gambar 4 adalah proses bisnis user management. Proses bisnis ini menggambarkan proses yang dilalui super admin dan jenis interaksi yang dilakukan pada web *page login* dan web *page user management*.

Berikut beberapa hasil pengerjaan proyek pengembangan halaman web *user management* dan juga akan dijelaskan juga hasil yang didapatkan berdasarkan pembahasan sebelumnya. Pembahasan hasil akan dibahas secara berurutan dari hasil yang didapatkan dari tahapan persiapan. Kemudian dilanjutkan dengan hasil implementasi.

A. Hasil Persiapan

Hasil persiapan berupa beberapa *software* yang dipakai untuk pengembangan halaman web *user management*. Berikut merupakan beberapa *software* yang dipakai, seperti:

1. *Instalasi Software*

Berikut merupakan hasil dari beberapa *software* yang perlu diinstal untuk pengembangan halaman web *user management*, seperti:

- Node.js (*latest version*) → *download* node.js LTS (v21.1.0)
- Next.js (*Framework front-end*) (v12.3.4) → buka terminal atau cmd → ketik npx create-next-app@latest → sesuaikan nama *project* dan konfigurasi sesuai kebutuhan
- Visual Studio Code (*Code editor* untuk *project* Next.js) → download visual studio code sesuai dengan operating system yang digunakan.
- Visual Studio (*IDE Back-end*) → Visual studio Installer → *Modify* → ASP.NET and web development → *Create a new project* → cari dan pilih ASP.NET Application (.NET Framework)

  Pada saat melakukan instalasi visual studio terdapat beberapa pilihan pada *workloads* dan salah satunya adalah ASP.NET and web development yang perlu dicentang untuk kebutuhan pengembangan API halaman web *user management.*
- Internet Information Services (Windows server *hosting*) → ketik pada pencarian / *search windows* "*Turn Windows features on or off*" → Disesuaikan
- Postman (*testing software)* → *download* postman *app*

2. *Hasil Pengaturan IIS*

Berikut merupakan hasil dari pengaturan IIS yang diperlukan untuk pengembangan BE *user management.*Pada tahapan pertama buka aplikasi IIS, kemudian pada sites klik kanan dan akan terdapat pilihan "*Add Website*" dan kemudian di klik.

Berikutnya dapat mengisikan *site name* dan memilih lokasi dari *folder site name* akan ditempatkan pada *physical path*. Setelahnya dapat mengganti port sesuai yang diinginkan atau yang tersedia. Kemudian pilih ok dan IIS pengaturan sudah selesai dilakukan.

B. Hasil Pengembangan

Berikut merupakan beberapa implementasi yang telah dilakukan pada proyek pengembangan halaman web *user management*. Beberapa implementasi yang akan dibahas disini merupakan salah satu bagian paling penting yang diperlukan pada halaman web *user management*.

1. *Hasil Pengerjaan Front-end*

Terdapat beberapa komponen yang dikembangkan, seperti *navbar* dan *sidebar*, *page header*, *table*, *pagination*, *modal*, dan *pop up dialog*. Kemudian dibuat juga fungsi-fungsi untuk web yang interaktif dengan JavaScript.

2. *Hasil Pengerjaan Back-end*

Pada pengembangan halaman web *user management* ini terdapat 2 *services* atau layanan, yaitu *user management services* dan *library services*. Layanan ini dibentuk untuk memenuhi kebutuhan fungsionalitas halaman *user management*.

- *User Management Services*

  Terdapat 6 API yang dibentuk oleh *controller*, *model*, *repository* pada *project user management* yang dipakai pada halaman web *user management*. API tersebut dibuat dengan ASP.NET MVC 4 dengan kelas *model*, *controller*, dan *repository*.

  Pada *controller class* terdapat 6 *methods*, seperti Get User, Get Role Name, Create User, Update User, Delete User, Offset. Tentunya *method* ini terhubung dengan *stored procedure* pada SQL Server yang dipanggil pada kelas *repository*. Fungsi dari masing-masing metode adalah sesuai dengan nama dari metodenya, seperti:
  - Get User – Mendapatkan data dari 1 id user yang dipilih.
  - Get Role Name – Mendapatkan data *role* yang akan dipakai untuk *value* dari beberapa pilihan di *checkbox.*
  - Create User – Menciptakan data baru dari inputan data yang *user* isi dari *modal*_tambahUser.
  - Update User – Memperbarui data yang telah diedit oleh *user* dari *modal*_editUser.
  - Delete User – Memberikan *value* IsDeleted = 1 untuk data yang di delete dari halaman *modal*_editUser.

Offset– Mendapatkan *list* data-data yang akan ditampilkan pada tabel, serta membawa fungsi untuk pencarian data berdasarkan parameter *keyword*, dan fungsi untuk perpindahan halaman data tabel dengan parameter *skip* dan *limit*.

- *JWT Services*

  Kemudian terdapat 7 API yang dibentuk oleh *controller*, *model*, *repository* pada *project Library* yang dipakai pada halaman web *user management* dan halaman web lainnya pada keseluruhan *project* CMS. API tersebut dibuat dengan ASP.NET MVC 4 dengan kelas *model*, *controller*, dan *repository*.

  Pada *controller class* terdapat 4 *methods*, seperti JWT, Login, Logout, dan Get User Session. Seluruh *method* ini terhubung dengan *stored procedure* pada SQL Server yang dipanggil pada kelas *repository*. Fungsi dari masing-masing metode adalah sesuai dengan penamaan dari metodenya, seperti:

  - JWT – *Regenerate* token JWT dengan refresh token.
  - Login – Melakukan validasi saat *user login*. Terdapat 2 *SP* berbeda pada *repository* yang dipanggil pada *method login*, yaitu Get Data User untuk mendapatkan *user* yang sedang *login* dan validasi *login* untuk melakukan validasi *user* yang sedang *login*. Jika validasi *login* berhasil maka *user* akan memiliki token JWT dan dapat masuk ke *page* sesuai dengan *role* yang dimiliki.
  - Logout – Melakukan validasi saat *user logout*. Saat *user* sudah *logout* hal yang terjadi adalah token jwt user tersebut di *delete* atau dibuat menjadi *null*.
  - Get User Session – Mendapatkan data dari *user* yang sedang melakukan *login*.

3. Hasil Integrasi

Proses Integrasi API pada saat pengerjaannya terjadi 2 kali perubahan. Integrasi yang pertama kali keseluruhannya dilakukan pada *client-side*. API yang berada pada *client-side* adalah API yang interaktif karena akan berhubungan langsung dengan interaksi dari pengguna, seperti API *create user, edit user, delete user, get user*. Pemanggilan API dilakukan secara langsung pada *client-side* dengan mengakses *file* serviceUserManagement pada *folder* API Next.js.

Kemudian pada integrasi yang kedua mulai dipisahkan API yang harusnya berada pada *client-side* dan API yang berada pada *server-side*. Pada Next.js pemanggilan API secara *server-side* dapat dilakukan dengan getServerSideProps. Menggunakan getServerSideProps adalah metode *server-side rendering* (SSR) pada Next.js. Dengan begitu data yang dibutuhkan akan diambil dan di *generate* pada sisi server setiap terjadi *request* API.

B. Hasil Peninjauan

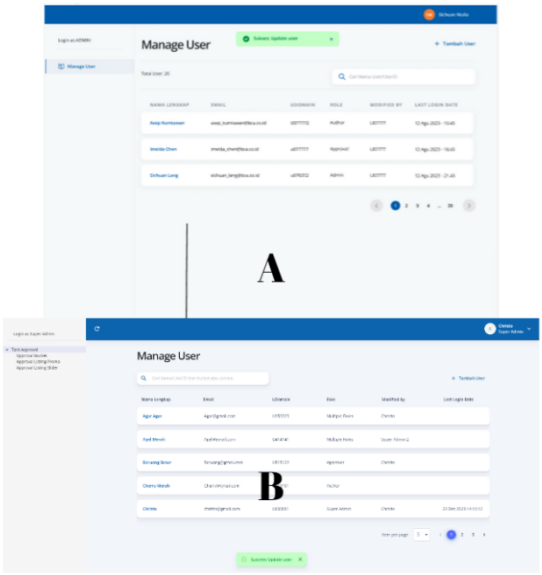
Berikut merupakan hasil dari tahapan peninjauan yang dilakukan selama pengembangan halaman web *user management* yang merupakan bagian dari proyek CMS.

TABEL I
HASIL EVALUASI

Versi Awal	Versi Terbaru (11 Desember 2023)	Tujuan
(FE) *Role* → *radio button*	(FE) *Role* → *checkbox* (*multiple roles*)	Dapat memilih lebih dari 1 roles
(FE) Menggunakan Navbar dan sidebar yang identik dengan desain dari Figma	(FE) Mengganti navbar dan sidebar dengan me-*reuse* komponen *layout* navbar & sidebar dari PIC FE	Supaya navbar & sidebar page user-management sama dengan page lainnya
(FE) PageHeader adalah komponen terpisah (*title, button* tambah user, *search bar, total user*)	(FE) *PageHeader* adalah 1 komponen denga nisi, seperti (*title, button* tambah *user, search bar, total user*)	Menggabung komponen lainnya menjadi 1 segmen dan rapih
(FE) Tidak ada pilihan *limit user per page* (Pada desain Figma tidak ada)	(FE) Membuat komponen untuk *user per page* sebagai bagian dari *pagination*	Tampilan data bisa diatur sesuai keinginan user
(FE) Penjagaan pada *handle Delete* pada kasus hanya 1 data pada *page* dan *limit* tertentu maka *router* akan error	(FE) Fixed penjagaan *handle Delete* untuk kasus *router error*. Jika terkena kasus sebelumnya maka *router* akan kembali ke *page* 1 & *limit* 3 (*default*)	Fixing handleDelete yang sebelumnya error
(FE) Super admin masih bisa mengedit super admin lainnya	(FE) Memberikan *handle* Super Admin tidak dapat mengedit super admin lainnya	Menambahkah tahapan keamanan data
(FE) Pesan *pop up* ada di sisi atas halaman	(FE) Pesan *pop up* ada di sisi bawah halaman	Disamakan dengan pop up page lain yang ada di sisi bawah
(FE) Pengisian data email pada *modal edit* dan tambah menggunakan *textfield*	(FE) *Textfield* digantikan dengan komponen *Input Text*	Memanfaatkan component yang sudah ada dan mengefisiensikan kode
(FE) Belum ada *button refresh page*	(FE) Menambahkan *button refresh* page pada navbar	Terdapat opsi untuk refresh page dengan button
(BE) Tipe data *role* string	(BE) Tipe data *role* list<string>	karena role jadi multiple roles, maka data jadi list
(BE) Role → tabel_user → 1 data	(BE) *Role* → tabel_user_role → *multiple role*	Pemisahan kolum role dari table_user ke table user role
(BE) userid & email bisa identik dengan data lainnya pada tabel di *database*	(BE) userid & email tidak boleh identik dengan data lainnya pada tabel di *database*	Supaya tidak ada userid dan email yang identik
(BE, FE) API offset punya parameter *skip, limit, keyword*	(BE, FE) API offset punya parameter *page, limit, keyword*	Parameter page lebih mudah dipahami user
(BE, FE) Tidak ada date format	(BE, FE) Terdapat *Date Format* (dd/MM/yyyy)	Supaya format sama dengan page lainnya
(BE, FE) *value* pada *role* masih *hardcode* pada FE	(BE, FE) *value* pada *role* sudah dari API getrolename	Jika terjadi perubahan rolename maka di FE juga mengikuti
(BE, FE) *value* dari *role* adalah dari kolom rolename yang berupa string	(BE, FE) *value* dari *role* adalah dari kolom roleId yang berupa GUID	Jika terjadi perubahan rolename maka di FE juga mengikuti
(BE) pada *database* belum ada *user-defined table types*	(BE) pada *database* ada *user-defined table types* yaitu @userRoleDatatype	Datatype dibutuhkan pada query yang membutuhkan join
(BE) sp *insert* untuk *create new user* tanpa @userRoleDatatype karena masih *insert* 1 *role* saja dan tanpa penjagaan userid & email (masih bisa identik)	(BE) sp *insert* untuk *create new user* sudah dengan @userRoleDatatype karena memungkinkan untuk *insert multiple roles* dan ada penjagaan *user id* & *email* supaya tidak bisa identik	Memungkinkan untuk dapat insert lebih dari 1 roles untuk 1 user
(BE) sp *update user* tanpa @userRoleDatatype karena masih menggunakan *radio button* / *single data* dan tanpa penjagaan *user id* & *email* (masih bisa identik). Masih terdapat *subquery*.	(BE) sp *update user* sudah dengan @userRoleDatatype karena sudah menggunakan *checkbox* dan sudah dengan penjagaan *user id* & email supaya tidak bisa identik. *Subquery* dihapus & diganti @userRoleDatatype	Sebagai pengganti dari subquery pada store procedure
(BE) sp *get data by id* masih menggunakan *subquery* untuk mendapatkan *multiple* data dari *role*	(BE) sp get data by id menggunakan JOIN untuk mendapatkan *multiple data* dari *role* dan *subquery* dihapus	Penghapusan seluruh subquery untuk peningkatan performa
(BE) sp offset masih menggunakan *subquery* untuk mendapatkan *multiple data* dari role	(BE) sp *get data by id* menggunakan JOIN untuk mendapatkan *multiple data* dari *role* dan *subquery* dihapus	Penghapusan seluruh subquery untuk peningkatan performa

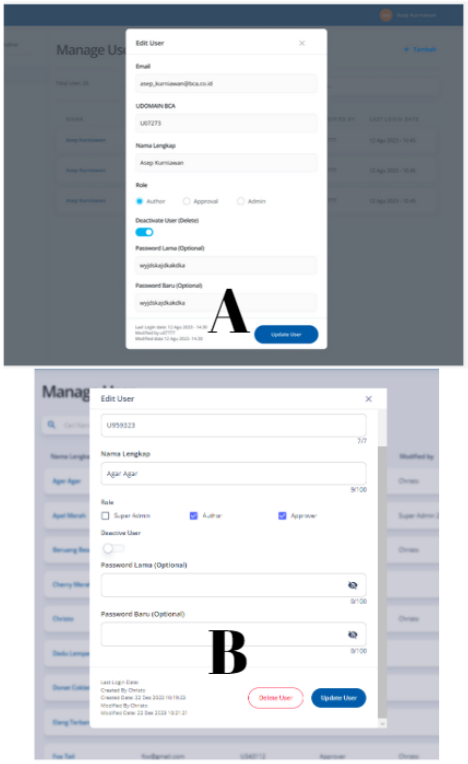
(BE) *Claim* JWT terdapat *Id, Full name, Role*	(BE) *Claim* JWT menjadi *Id, Full Name* dan *Role Id*	Menjadi Role Id supaya perubahan name tidak masalah
(BE) Belum ada penjagaan *role* untuk *request* API tertentu	(BE) Terdapat penjagaan *role* untuk dapat melakukan *request* API apa saja	Penjagaan supaya seluruh API user-management hanya diakses role tertentu
(BE) Belum ada penjagaan suatu *role* untuk bisa masuk *page* mana saja	(BE) Terdapat penjagaan suatu *role* hanya bisa masuk ke *page* tertentu sesuai *role*	Penjagaan tambahan di BE supaya role tertentu bisa masuk ke page mana saja

Berikut adalah *test case* yang dipakai selama pengembangan halaman web *user management*. *Test case* ini dilakukan pengujian berkali-kali atau setiap adanya perubahan baik dari sisi *back-end* ataupun *front-end* untuk memastikan API dan tampilan tidak ada *error* atau ketidaksesuaian.



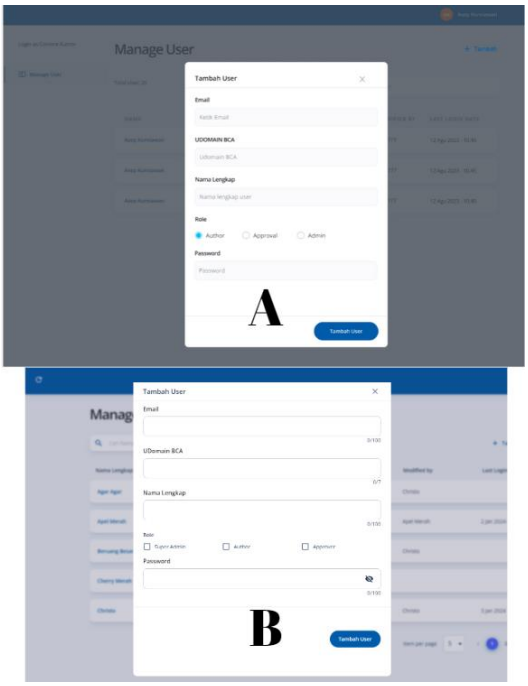
Gambar 5. Perbandingan Parent Page User Management

Berikut adalah perbedaan dari versi awal dan akhir dari *parent page* halaman *user management*. Terdapat huruf A dan B pada gambar tersebut untuk menunjukan A adalah versi awal sesuai dengan desain pada Figma dan B adalah versi terbaru atau *latest version* (11 Desember 2023). Terdapat beberapa perubahan yang dilakukan dari versi awal hingga menjadi versi terbaru, seperti:



Gambar 6. Perbandingan Modal Edit User

Berikut merupakan gambar dari *modal edit user*. Terdapat huruf A dan B pada gambar untuk menunjukan bahwa gambar dengan huruf A adalah versi awal sesuai dengan desain dari Figma dan B adalah versi terbaru dari *modal edit user* (11 Desember 2023). Perbedaan dari versi A dan B adalah pada versi B atau terbaru, yaitu:



Gambar 7. Perbandingan Modal Tambah User

Berikut merupakan gambar *modal* tambah user. Terdapat huruf A dan B pada gambar untuk menunjukan bahwa gambar dengan huruf A adalah versi awal sesuai dengan desain pada Figma dan huruf B adalah versi terbaru. Pada *modal* tambah *user* tidak banyak perubahan yang terjadi. Perubahan yang terjadi hanya terjadi pada pilihan *role* yang sebelumnya dengan *radio button* dan kemudian pada versi terbaru diganti dengan *checkbox* dengan tujuan dapat memilih lebih dari 1 *roles.* Perubahan lainnya adalah kotak *modal* dibuat lebih besar sehingga tampilan tidak begitu sempit.

C. Evaluasi Hasil Kerja

Pengerjaan proyek halaman web *user management* telah dikembangkan sesuai dengan kebutuhan user dan juga sesuai dengan permintaan instansi. Proyek diselesaikan pada tanggal 22 Desember 2023. Seluruh *folder*, *file*, ataupun kode versi terbaru halaman web *user management* sudah diterima PIC *front-end*, *back-end*, dan *code reviewer*. Halaman *web user management* telah digabungkan menjadi bagian dari *website* CMS Promo dan sedang dalam tahap SIT (*System Integration Testing*) di bulan Januari 2024 dan setelahnya akan masuk pada tahap UAT (*User Acceptance Testing*).

V. KESIMPULAN

Simpulan dari pengerjaan proyek pengembangan halaman web *user management* dengan *framework* Next.js dan ASP.NET adalah seperti berikut:

Telah dikembangkan tampilan halaman web *user management* dengan menggunakan *framework* Next.js dengan tampilan berserta fungsi yang telah sesuai dengan permintaan dari instansi.

Telah dikembangkan API yang dikembangkan dengan memanfaatkan *framework* ASP.NET dan RDBMS SQL Server dan telah sesuai dengan permintaan dari instansi.

Telah berhasil mengintegrasikan API yang dikembangkan dengan ASP.NET dan SQL Server, dengan halaman web *user management* yang dikembangkan dengan Next.js.

UCAPAN TERIMA KASIH

Penyusunan laporan magang sebagai tugas akhir ini dapat diselesaikan tentunya dengan adanya bantuan dan bimbingan dari Ibu Adelia, S. Kom., M.T. selaku dosen pembimbing dari Fakultas Teknologi Informasi Universitas Kristen Maranatha. Oleh karena itu saya mengucapkan terima kasih sebesar-besarnya kepada semua pihak yang telah membantu saya dalam penyusunan laporan tugas akhir. Saya mengucapkan terima kasih sebesar-besarnya khusunya kepada:

1. Bapak Ir.Teddy Marcus Zakaria, M.T. selaku Dekan Fakultas Teknologi Informasi, Universitas Kristen Maranatha.
2. Bapak Sendy Ferdian Sujadi, S.Kom., M.T. selaku ketua program studi Sistem Informasi S1, Fakultas Teknologi Informasi, Universitas Kristen Maranatha. vi
3. Ibu Adelia, S.Kom., M.T. selaku koordinator mata kuliah program pengayaan, Sistem Informasi S1, Fakultas Teknologi Informasi, Universitas Kristen Maranatha dan juga selaku dosen pembimbing.
4. Kak Aryo Wibisono selaku team leader Digital Experience Team.
5. Kak Andri Hadi Wijaya selaku mentor atau pembimbing lapangan selama kegiatan magang di PT XYZ.
6. Biro F, biro seluruh kegiatan magang dan bimbingan berlangsung selama magang di PT XYZ

DAFTAR PUSTAKA

[1] Ariffud Muhammad, “Apa Itu Website? Pengertian, Fungsi, Sejarah, Unsur, Jenisnya,” Niagahoster. Accessed: Sep. 27, 2023. [Online]. Available: https://www.niagahoster.co.id/blog/pengertian-website/?gclid=Cj0KCQjwpc-oBhCGARIsAH6ote-RcXECKtUuNUaeiKqKoDtMgHjPN3Clb9gv5iasxWm2q_I6ZSQbtBwaAnYFEALw_wcB

[2] MDN contributors, “What is a URL?,” developer.mozilla.org. Accessed: Sep. 27, 2023. [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Web_mechanics/What_is_a_URL

[3] Google Support, “Menghapus, mengizinkan, & mengelola cookie di Chrome,” support.google.com. Accessed: Sep. 29, 2023. [Online]. Available: https://support.google.com/chrome/answer/95647?hl=id&co=GENIE.Platform%3DAndroid

[4] “A flexible & easy-to-manage web server...,” www.iis.net.

[5] “Pengertian Front End dan Back End Developer, Apa Bedanya?,” Universitas Esa Unggul. Accessed: Sep. 29, 2023. [Online]. Available: https://fasilkom.esaunggul.ac.id/pengertian-front-end-dan-back-end-developer-apa-bedanya/

[6] Jenius co.create, “Front-End dan Back-End Developer: Apa Perbedaannya?,” cocraete.id.

[7] S. Subramanian, “Live API Documentation,” Ontario, 2014.

[8] K. PURWANTINI, “Pengenalan Apa Itu Framework,” Universitas Sains & Teknologi Komputer. Accessed: Sep. 29, 2023. [Online]. Available: https://komputerisasi-akuntansi-d3.stekom.ac.id/informasi/baca/Pengenalan-Apa-itu-Framework/1c2051fbd87002b38b87cda4b5f36dda08206977

[9] “What is .NET?,” dotnet.microsoft.com.

[10] Randolph West, “What is SQL Server?,” learn.microsoft.com. Accessed: Oct. 02, 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16

[11] R. Hyams and T. Pilko, "'You could use the API!': A Crash Course in Working with the Alma APIs using Postman," *code4lib*, no. 54, Aug. 2022, Accessed: Oct. 18, 2023. [Online]. Available: https://journal.code4lib.org/articles/16597
[12] M. Jones, J. Bradley, and N. Sakimura, "Internet Engineering Task Force (IETF)," 2015, [Online]. Available: http://www.rfc-editor.org/info/rfc7519.
[13] T. Fadhilah Iskandar, M. Lubis, T. Fabrianti Kusumasari, and A. Ridho Lubis, "Comparison between client-side and server-side rendering in the web development," *IOP Conf Ser Mater Sci Eng*, vol. 801, no. 1, pp. 2–2, May 2020, doi: 10.1088/1757-899X/801/1/012136.
[14] Herman and A. Geovanny, "ANALISIS RENDERING PERFORMA ANTARA SERVER SIDE DAN CLIENT SIDE PADA WEB APPLICATION," *Jurnal Ilmiah Betrik*, vol. 13, no. 3, pp. 312–312, Dec. 2022.