RANCANG BANGUN *FRONTEND* SISTEM *MONITORING CACHE* PADA *PROXY SERVER* BERBASIS *WEB* MENGGUNAKAN *REACTJS* (STUDI KASUS: PT. QUEEN NETWORK NUSANTARA)

(Skripsi)

Oleh

M. ALDI KURNIAWAN

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG

2024

RANCANG BANGUN FRONTEND SISTEM MONITORING CACHE PADA PROXY SERVER BERBASIS WEB MENGGUNAKAN REACTJS (STUDI KASUS: PT. QUEEN NETWORK NUSANTARA)

Oleh

M. ALDI KURNIAWAN

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA TEKNIK

Pada

Program Studi Teknik Informatika
Fakultas Teknik

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
2024

ABSTRAK

RANCANG BANGUN *FRONTEND* SISTEM *MONITORING CACHE* PADA *PROXY SERVER* BERBASIS *WEB* MENGGUNAKAN *REACTJS* (STUDI KASUS: PT. QUEEN NETWORK NUSANTARA)

Oleh

M. ALDI KURNIAWAN

Perkembangan teknologi digital mendorong kebutuhan akses data cepat dan efisien, termasuk melalui *proxy server* dengan fitur *caching*. PT Queen Network Nusantara (QNN) telah mengimplementasikan *cache* pada *proxy server*, namun sistem *monitoring* yang saat ini digunakan melalui terminal SSH masih memiliki keterbatasan, terutama dalam hal antarmuka pengguna yang menggunakan *Command Line Interface* (CLI). Penelitian ini bertujuan merancang sistem *monitoring cache* berbasis *web* menggunakan *ReactJS* untuk menyediakan antarmuka yang efisien dan ramah pengguna. Penelitian ini menggunakan metode *Rapid Application Development* (RAD) untuk pengembangan sistem yang cepat dan fleksibel. Tahap *requirement planning* dilakukan melalui studi literatur dan wawancara, *user design* menggunakan UML, *construction* dengan *ReactJS*, dan pengujian fungsionalitas menggunakan *Blackbox*. Pengujian non-fungsional dilakukan menggunakan *Google Lighthouse* untuk menguji kompatibilitas *browser*, serta *Zed Attack Proxy* untuk menguji keamanan terhadap *Cross-Site Scripting* (XSS). Hasil pengujian menunjukkan delapan fitur utama berhasil dikembangkan dengan 29 skenario pengujian yang sesuai harapan, serta kompatibilitas dan keamanan sistem terbukti baik. Pengujian *User Experience Questionnaire* (UEQ) juga menunjukkan peningkatan signifikan dalam pengalaman pengguna.

Kata kunci: *Cache, ReactJS, Website, Proxy Server*

ABSTRACT

DESIGN AND DEVELOPMENT OF A WEB-BASED CACHE MONITORING SYSTEM FOR PROXY SERVERS USING REACTJS (CASE STUDY: PT. QUEEN NETWORK NUSANTARA)

By

M. ALDI KURNIAWAN

The advancement of digital technology has driven the need for fast and efficient data access, including through proxy servers with caching features. PT Queen Network Nusantara (QNN) has implemented caching on its proxy server; however, the current monitoring system, which relies on SSH terminals, still has limitations, particularly in its user interface that utilizes a CLI. This research aims to design a web-based cache monitoring system using ReactJS to provide an efficient and user-friendly interface. The study employs the Rapid Application Development (RAD) methodology for fast and flexible system development. The requirement planning stage is conducted through literature studies and interviews, user design is carried out using UML, construction is implemented with ReactJS, and functionality testing is performed using Blackbox testing. Non-functional testing is conducted using Google Lighthouse to test browser compatibility and Zed Attack Proxy to assess security against Cross-Site Scripting (XSS). The test results show that eight main features have been successfully developed with 29 test scenarios meeting expectations, and the system's compatibility and security are proven to be robust. Additionally, testing with the User Experience Questionnaire (UEQ) indicates a significant improvement in user experience.

keywords: *Cache, ReactJS, Website, Proxy Server*

Judul Skripsi	:	**RANCANG BANGUN FRONTEND SISTEM MONITORING CACHE PADA PROXY SERVER BERBASIS WEB MENGGUNAKAN REACTJS (STUDI KASUS: PT. QUEEN NETWORK NUSANTARA)**
Nama Mahasiswa	:	**M. Aldi Kurniawan**
Nomor Pokok Mahasiswa	:	2015061071
Program Studi	:	Teknik Informatika
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Yessi Mulyani, S.T., M.T.	**Rio Ariestia Pradipta, S. Kom, M.T.I.**
NIP. 197312262000122001	NIP. 198603232019031013

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T., M.T.	**Yessi Mulyani, S.T., M.T.**
NIP. 197103141999032001	NIP. 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Yessi Mulyani, S.T., M.T.** .....

Sekretaris : **Rio Ariestia Pradipta, S. Kom, M.T.I.** .....

Penguji : **Wahyu Eko Sulistiono, S.T., M.Sc.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **19 Desember 2024**

SURAT PERNYATAAN

Saya yang bertandatangan di bawah ini, menyatakan bahwa skripsi saya dengan judul "Rancang Bangun *Frontend* Sistem *Monitoring Cache* Pada *Proxy Server* Berbasis *Web* Menggunakan *ReactJS* (Studi Kasus: PT. Queen Network Nusantara)" dibuat oleh saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 17 -02- 2025
Pembuat pernyataan,

M. Aldi Kurniawan
NPM. 2015061071

RIWAYAT HIDUP

Penulis dilahirkan di Bandar Lampung, pada tanggal 21 Februari 2002. Penulis merupakan anak pertama dari pasangan Bapak Sumono dan Ibu Sri Hastuti. Penulis menyelesaikan pendidikannya di SDN 3 Kemiling Permai pada tahun 2014, SMPN 2 Bandar Lampung pada tahun 2017, dan SMAN 9 Bandar Lampung pada tahun 2020. Pada tahun 2020 penulis terdaftar sebagai mahasiswa Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik Universitas Lampung melalui jalur SBMPTN. Selama menjadi mahasiswa, penulis melakukan beberapa kegiatan, antara lain:

1. Menjadi anggota biasa Himpunan Mahasiswa Teknik Elektro Universitas Lampung, Departemen Pendidikan dan Pengembangan Diri Divisi Minat dan Bakat pada tahun 2020.
2. Menjadi anggota biasa Himpunan Mahasiswa Teknik Elektro Universitas Lampung, Departemen Komunikasi dan Informasi Divisi Media Informasi pada tahun 2021.
3. Melaksanakan Kuliah Kerja Nyata di Desa Bandar Baru, Kecamatan Sukau, Kabupaten Lampung Barat, Provinsi Lampung pada bulan Januari 2023.
4. Mengikuti program *Studi Independen* Kampus Merdeka dari Kementerian Pendidikan dan Budaya dengan mengambil kelas Fullstack Web Developer di PT Nurul Fikri Cipta Inovasi pada tahun 2023.
5. Melakukan kerja praktik di Dinas Komunikasi Informatika dan Statistik Provinsi Lampung pada bulan Juni sampai Agustus tahun 2023 dengan membuat Website Sistem Rencana Pengembangan Kompetensi ASN.

MOTTO

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya."

(Q.S. Al-Baqarah : 286)

"Ilmu tanpa amal adalah kegilaan, dan amal tanpa ilmu adalah kesia-siaan."

(Imam Ghazali)

"Berhenti bercita-cita adalah tragedi terbesar dalam hidup manusia."

(Andrea Hirata)

"*Just Believe In Yourself And You Can Become A Hero*."

(Toshinori Yagi)

PERSEMBAHAN

Bismillaahirrohmaanirrahiim,
Dengan mengharapkan ridho dari Allah SWT,
Kupersembahkan karyaku ini untuk orang-orang yang kusayangi dengan setulus hati:
Orangtua tercinta,
Keluargaku,
Teman-Temanku,
Dan
Orang-orang yang telah membantu hidupku

Terimakasih untuk segalanya,
Kalian adalah hartaku yang paling berharga

SANWACANA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayat-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi/tugas akhir ini dengan judul "Rancang Bangun *Frontend* Sistem *Monitoring Cache* Pada *Proxy Server* Berbasis *Web* Menggunakan *ReactJS* (Studi Kasus: PT. Queen Network Nusantara)". Dalam pelaksanaan dan pembuatan skripsi/tugas akhir ini penulis menerima dukungan baik secara moril maupun materil yang sangat berharga dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada semua pihak yang telah membantu, khususnya kepada:

1. Kedua orangtuaku tercinta dan seluruh keluarga yang selalu memberikan doa, motivasi dan kasih sayang tiada terkira yang selalu mengingatkan penulis untuk menyelesaikan penelitian ini;
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung;
3. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung;
4. Ibu Yessi Mulyani, S.T., M.T., selaku Ketua Program Studi Teknik Informatika Universitas Lampung dan selaku Pembimbing Utama yang telah membantu proses kelancaran pengerjaan penelitian dan memberikan banyak saran serta masukan terhadap penelitian ini;
5. Bapak Rio Ariestia Pradipta, S. Kom., M.TI., selaku Pembimbing Pendamping yang selalu memberikan dukungan serta bimbingan agar menjadi lebih baik;
6. Bapak Wahyu Eko Sulistiono, S.T., M.Sc., selaku Penguji yang telah memberikan banyak saran dan masukan terhadap penelitian ini;

7. Bapak M. Komarudin, S.T., M.T., selaku Pembimbing Akademik yang selalu memberikan bimbingan dan dukungan kepada penulis selama menyelesaikan perkuliahan;
8. Mbak Rika selaku Admin Program Studi Teknik Informatika yang telah banyak membantu penulis dalam segala urusan administrasi selama perkuliahan;
9. Seluruh dosen dan staf Jurusan Teknik Informatika Unila yang memberi masukan dan mempermudah proses pembuatan skripsi/tugas akhir ini.
10. Adelia Rafika Putri yang selalu memberi dukungan dengan tulus sehingga penulis dapat menyelesaikan skripsi ini hingga tuntas.
11. Keluarga besar Teknik Elektro Angkatan 2020 yang telah menjadi teman seperjuangan sejak mahasiswa baru. Terimakasih telah mewarnai masa perkuliahan penulis.

Penulis berharap agar laporan ini dapat menjadi referensi bagi pengembangan keilmuan di bidang teknik informatika. Oleh karena itu, semoga penelitian ini bermanfaat bagi yang membacanya.

Bandar Lampung, 20 Februari 2025
Penulis,

M. Aldi Kurniawan

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

I. PENDAHULUAN

1.1 Latar Belakang

Dalam era digital yang semakin maju, kebutuhan akan akses data yang cepat dan efisien menjadi sangat krusial. Berbagai perusahaan, khususnya yang bergerak dalam bidang penyedia jasa layanan *internet*, sangat berupaya untuk memastikan bahwa pengguna mereka mendapatkan pengalaman terbaik saat mengakses layanan dan aplikasi *online* [1]. Salah satu teknologi yang sangat membantu mewujudkan hal ini adalah penggunaan *proxy server*, yang memanfaatkan fitur *caching* untuk meningkatkan efisiensi dan kecepatan akses *internet*. Dengan adanya *proxy server*, data yang sering diakses oleh pengguna dapat disimpan sementara dalam *cache*, sehingga ketika pengguna ingin mengakses informasi yang sama di kemudian hari, *proxy server* dapat menyajikannya dengan lebih cepat tanpa harus mengambil data dari sumber aslinya setiap kali permintaan dilakukan. Ini tidak hanya mengurangi waktu tunggu bagi pengguna, tetapi juga mengurangi beban pada *server* asal, yang pada gilirannya dapat meningkatkan performa keseluruhan jaringan [2]. Namun, untuk memastikan bahwa *proxy server* berfungsi secara optimal dan tidak menimbulkan masalah dalam jaringan, diperlukan sistem *monitoring* yang handal.

Sistem *monitoring* adalah suatu sistem yang digunakan untuk mengamati, menganalisis, dan mengelola kinerja serta kondisi dari perangkat atau jaringan. Sistem ini penting karena membantu memastikan bahwa semua komponen bekerja dengan optimal dan mendeteksi masalah secara cepat sebelum berkembang menjadi masalah yang lebih besar [3]. Sistem *monitoring cache* pada *proxy server* dapat mencakup pemantauan *timestamp*, *client address*, URL yang diminta, kode status HTTP, ukuran respons, dan lainnya. Dengan sistem ini,

perusahaan dapat memonitor dan menganalisis kinerja *cache* dengan lebih efektif dan mengambil tindakan yang diperlukan untuk mengoptimalkan kinerja *caching*.

PT. Queen Network Nusantara (QNN) telah berhasil mengimplementasikan *cache* pada *proxy server*, namun perusahaan masih memerlukan sistem untuk memonitor *event* yang terjadi pada *cache*. *Monitoring* terhadap *cache* penting agar segera bisa mengamati rasio *cache hit* (ketika data yang diminta tersedia di *cache*) dan *cache miss* (ketika data yang diminta tidak tersedia di *cache*) secara *real-tim*e. Hal ini membantu dalam meningkatkan optimalisasi kinerja sistem dan mengurangi *downtime* [4]. PT. Queen Network Nusantara, sebagai salah satu perusahaan penyedia layanan jaringan internet, telah lama menggunakan sistem *monitoring* melalui terminal SSH dan *Command Line Interface* (CLI) untuk memantau dan mengelola kinerja *cache proxy* mereka. Namun, metode ini memiliki beberapa kelemahan, seperti antarmuka yang kurang ramah pengguna dan kesulitan dalam memvisualisasikan data secara *real-time* [5]. Oleh karena itu, dibutuhkan solusi yang lebih efisien dan *user friendly* untuk memenuhi kebutuhan *monitoring cache* pada *proxy server* yang semakin kompleks.

Penggunaan *website* untuk sistem *monitoring cache* pada *proxy server* adalah solusi efektif untuk mendeteksi aktivitas yang tidak biasa atau mencurigakan, seperti lonjakan permintaan yang tidak wajar yang mungkin menunjukkan adanya serangan atau penyalahgunaan jaringan internet [6]. *Website* juga mendukung *multitasking* dengan membuka beberapa *tab* untuk *monitoring* secara *paralel*, sehingga memberikan solusi yang lebih *modern, efisien, dan user friendly* [7]. Oleh karena itu, perlu dilakukan penelitian mengenai perancangan sistem *monitoring cache* pada *proxy server* berbasis *web* menggunakan *ReactJS*. *ReactJS* dipilih karena kemampuannya dalam membangun antarmuka pengguna yang dinamis dan responsif, serta kemudahan integrasinya dengan berbagai teknologi *backend* [8]. Sistem *monitoring* ini diharapkan dapat menjadi solusi yang efektif dan efisien dalam meningkatkan *user experience* ketika memonitor *cache proxy* pada PT. Queen Network Nusantara.

1.2 Rumusan Masalah

Berdasarkan latar belakang, kajian masalah yang mendasari penelitian ini adalah sebagai berikut:

1. Bagaimana merancang dan membangun *frontend* sistem *monitoring cache* pada *proxy server* berbasis *web* untuk PT. Queen Network Nusantara (QNN) dengan *ReactJS* menggunakan metode *Rapid Application Development* (RAD)?
2. Apakah sistem yang dikembangkan dapat meningkatkan *user experience*?

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah merancang dan membangun *frontend* sistem *monitoring cache* pada *proxy server* berbasis *web* menggunakan *ReactJS* untuk PT. Queen Network Nusantara (QNN) yang dapat meningkatkan *user experience*.

1.4 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Sistem yang dikembangkan berfokus memonitor *cache* pada *proxy server* milik PT. Queen Network Nusantara (QNN).
2. Kebutuhan sistem berdasarkan informasi yang diberikan oleh PT. Queen Network Nusantara (QNN).

1.5 Sistematika Penulisan

Sistematika penulisan skripsi akhir ini terdiri dari 5 (lima) bab sebagai berikut:

BAB I : PENDAHULUAN

Bab ini menguraikan secara umum mengenai latar belakang penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Memaparkan teori-teori dasar yang digunakan sebagai referensi untuk memahami permasalahan terkait penelitian mengenai Rancang Bangun *Frontend* Sistem *Monitoring Cache* pada *Proxy Server* Berbasis *Web* Menggunakan *ReactJS*. Bab II berisi tentang pengertian *caching proxy*, perusahaan QNN, sistem *monitoring*,

website, *reactjs*, *google lighthouse*, *zed attack proxy*, *tailwind*, *vscode*, *unified modeling language*, *rapid application development*, *user experience questionnaire*, *blackbox testing*, penelitian terkait, dan *state of the art*.

BAB III : METODOLOGI PENELITIAN

Dalam bab ini membahas mengenai metode *Rapid Application Development* (RAD) yang terdiri dari *Requirements Planning*, *User Design*, *Construction*, dan *Cutover* yang digunakan dalam pengembangan sistem *monitoring* c*ache* pada *proxy server* di PT. Queen Network Nusantara (QNN).

BAB IV : HASIL DAN PEMBAHASAN

Dalam bab ini memuat hasil dan pembahasan yang diperoleh dalam penelitian.

BAB V : KESIMPULAN DAN SARAN

Dalam bab ini memuat kesimpulan dari hasil penelitian dan saran-saran mengenai perbaikan dan pengembangan lebih lanjut.

DAFTAR PUSTAKA

II. TINJAUAN PUSTAKA

2.1 *Caching Proxy*

Caching proxy adalah suatu mekanisme penyimpanan *object* dari suatu halaman *web* yang pernah diakses. Dengan adanya *caching proxy*, penggunaan *bandwidth* akan semakin efisien dan waktu untuk mengakses suatu halaman *web* pun semakin cepat. Berikut adalah beberapa fungsi dan manfaat dari *caching proxy*:

1. Mengurangi *Latency*

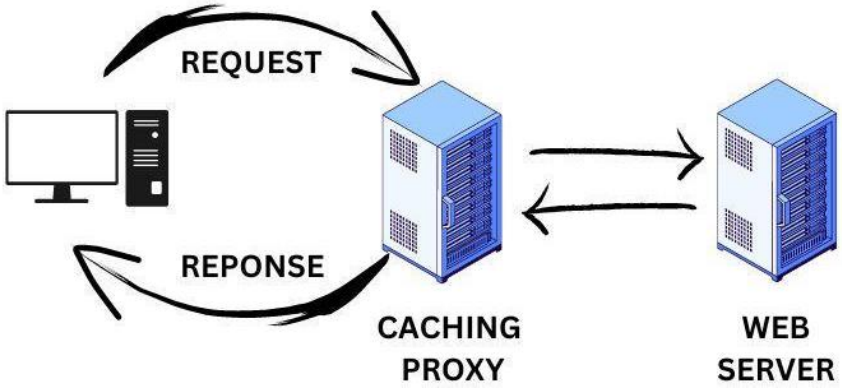
   Dengan menyimpan salinan konten yang sering diminta, *caching proxy* dapat mengurangi waktu yang dibutuhkan untuk memuat halaman web bagi pengguna.

2. Menghemat *Bandwidth*

   Caching proxy mengurangi jumlah data yang harus ditransfer dari *server* perusahaan ke pengguna dan menghemat penggunaan *bandwidth*.

3. Pengelolaan Konten

   Caching proxy juga dapat digunakan untuk memfilter dan mengelola konten yang diakses oleh pengguna, seperti memblokir situs *web* tertentu atau mengontrol akses ke konten tertentu berdasarkan kebijakan perusahaan [9].



Gambar 2.1 *Caching Proxy*

Dalam *caching proxy*, *log files* merupakan komponen penting untuk memantau dan menganalisis aktivitas yang terjadi pada server. Beberapa jenis *log files* yang bisa dimonitoring untuk memastikan kinerja caching adalah sebagai berikut:

1. *Access Log*

   Access log mencatat setiap permintaan yang diterima oleh *proxy server*. *Field* utama dalam *access log* meliputi *Timestamp*, yang menunjukkan waktu permintaan diterima. *Client IP Address* adalah alamat IP dari klien yang mengirimkan permintaan. *Request Method* menunjukkan metode HTTP yang digunakan dalam permintaan, seperti GET atau POST. *Requested* URL adalah alamat yang diminta oleh klien. HTTP *Status Code* mencatat kode status HTTP yang menunjukkan hasil dari permintaan, misalnya 200 untuk sukses atau 404 untuk tidak ditemukan.

2. *Store Log*

   Store log mencatat informasi terkait objek yang disimpan atau dikeluarkan dari *cache*. *Field* utama dalam *store log* meliputi *Timestamp*, yang menunjukkan waktu objek disimpan ke dalam *cache*. *Action* mengindikasikan tindakan pada objek *cache*. *Response Time* untuk mencatat waktu respons. *Cache Key* adalah ID unik untuk mengidentifikasi objek dalam cache. *Object Size* menunjukkan ukuran objek pada saat penyimpanan atau pelepasan, dengan nilai pertama sebagai ukuran yang tersimpan dan nilai kedua sebagai ukuran asli. URL menunjukkan alamat lengkap dari objek yang dimaksud. *HTTP Status* menunjukkan apakah objek berhasil disimpan dalam *cache* atau terjadi kegagalan saat penyimpanan. *Content Type* menunjukkan tipe konten objek, dan *Request Method* menandakan metode HTTP yang digunakan. Selain itu, *Last Modified* mencatat waktu terakhir objek dimodifikasi, memberikan informasi kapan konten terakhir diperbarui, dan *Expired* menandakan kapan objek tersebut akan kedaluwarsa dalam cache.

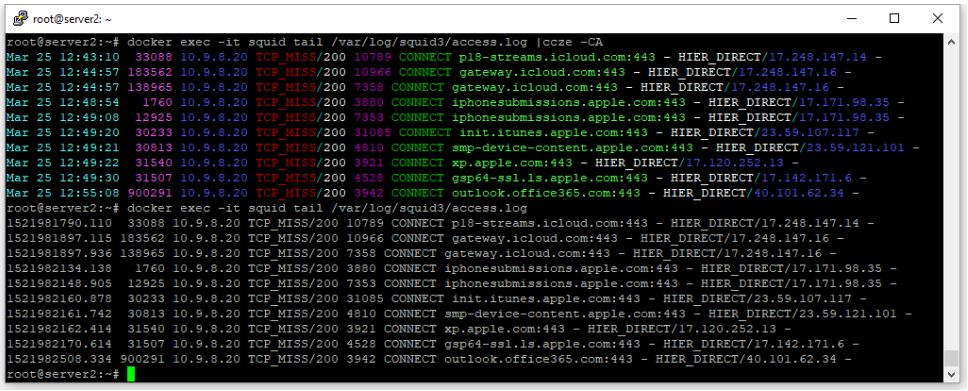
3. *User Agent Log*

   User agent log menyimpan informasi tentang perangkat dan aplikasi yang digunakan oleh klien saat mengakses *proxy server*. *Field* utama yang tercatat adalah *Timestamp*, yang menunjukkan waktu permintaan datang. *Client IP Address* mencatat alamat IP klien yang membuat permintaan. *User Agent*

adalah string yang mengidentifikasi perangkat dan *browser* yang digunakan oleh klien.

4. *Cache Log*

   Cache log di *Squid Proxy* mencatat informasi terkait operasi dan status *cache*, termasuk pemrosesan konfigurasi serta peringatan yang terjadi. *Log* ini membantu *administrator* untuk *memonitor* dan mengidentifikasi status serta potensi masalah yang terjadi selama operasi *proxy* dan *cache* di *Squid* [10].



Gambar 2.2 Contoh *Squid Proxy Log Files*

2.2 PT. Queen Network Nusantara

PT. Queen Network Nusantara merupakan perusahaan yang bergerak dalam jasa penyedia layanan internet, *backbone*, *maintenance* jaringan internet, multimedia, perangkat lunak, instalasi, dan konstruksi yang didirikan pada tahun 2016 oleh Bapak Supriyanto dan berpusat di Bandar Lampung. Saat ini PT. Queen Network Nusantara sudah memiliki wilayah cakupan di berbagai area di Provinsi Lampung, seperti Metro, Pesawaran, Pringsewu, Tanggamus, Gn. Sugih, dan Lampung Timur.

Gambar 2.3 Logo PT. Queen Network Nusantara

Salah satu produk yang dimiliki PT. Queen Network Nusantara adalah produk jaringan internet dari rumah ke rumah atau disebut FTTH (*Fiber to the Home*) yang sudah tersebar di beberapa daerah di kota Bandar Lampung, salah satunya adalah di Kecamatan Sukarame. Nilai-nilai perusahaan PT. Queen Network Nusantara adalah nilai-nilai yang berorientasi pada kemitraan. Visi dari PT. Queen Network Nusantara adalah "Kami memiliki komitmen yang besar untuk maju dan berkembang". Misi dari PT. Queen Network Nusantara adalah selalu menghargai dan menjaga kepercayaan mitra dalam setiap kerja sama yang dijalani, hubungan baik antar mitra akan selalu kami bina walaupun masa kerja sama telah berakhir, yang dijabarkan sebagai berikut:

1. Memberikan solusi yang efektif dan efisien pada setiap jasa yang kami berikan.
2. Mengkaji dengan seksama setiap masalah yang dihadapai guna memberikan solusi yang tepat guna.
3. Memberikan solusi guna meminimalisir resiko bisnis yang dihadapi tanpa mempengaruhi kegiatan usaha itu sendiri [11].

2.3 Sistem *Monitoring*

Monitoring adalah proses menghimpun data atau informasi dari banyaknya sumber yang umum dilakukan secara *real-time*. Sistem *monitoring* dapat berupa informasi ataupun data yang diambil secara langsung dan terus menerus dari sumbernya. *Monitoring* sendiri dilakukan untuk memastikan bahwa sistem atau jaringan berfungsi dengan baik, mendeteksi masalah atau anomali, dan memungkinkan respons cepat terhadap gangguan. Hasil informasi yang telah

diperoleh dari *monitoring* digunakan untuk mengevaluasi tindakan yang harus diambil selanjutnya sebagai bahan untuk menyampaikan suatu pertimbangan [12].

2.4 *Website*

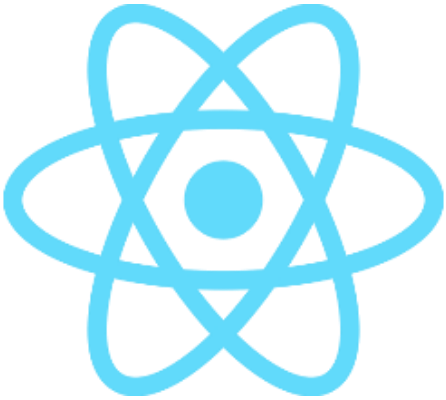
Website merupakan kumpulan halaman yang digunakan untuk menampilkan berbagai informasi seperti teks, gambar, animasi, dan suara, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian yang saling terhubung [13]. *Website* merupakan sekumpulan halaman yang berisi informasi dalam bentuk data digital baik berupa teks, gambar, video, audio, dan animasi lainnya yang disediakan melalui koneksi internet [7]. *Website* merupakan sejumlah halaman yang memiliki topik saling terkait pada tiap halaman, dan dapat disertakan gambar, video, animasi, atau jenis-jenis objek lainnya [14].

Gambar 2.4 Logo *Website*

2.5 *ReactJS*

ReactJS adalah *library JavaScript* yang dikembangkan oleh *Facebook* untuk memfasilitasi pembuatan antarmuka pengguna yang interaktif, stateful, dan mudah digunakan ulang. *ReactJS* sangat cocok untuk rendering antarmuka yang kompleks dengan performa tinggi. Sebagai *library* yang bersifat *composable user interface*, *React* memungkinkan kita membuat berbagai UI yang dapat dibagi menjadi beberapa komponen kecil. React mendukung konsep aplikasi *Single Page Application* (SPA), yaitu aplikasi yang hanya memiliki satu halaman. Berbeda dengan aplikasi *multipage*, SPA memungkinkan navigasi dilakukan tanpa perlu memuat ulang seluruh halaman. Hal ini menjadikan *React* efisien dalam

mengelola rendering. *ReactJS* mampu membuat antarmuka pengguna yang kompleks dengan menggunakan komponen-komponen kecil yang terisolasi. Komponen ini dapat digunakan kembali di berbagai bagian aplikasi, sehingga memudahkan proses pengembangan aplikasi berskala besar. Setiap komponen mendukung konsep *scalability* yang penting dalam pengembangan aplikasi modern. Dengan berbagai keunggulan seperti *open source*, *simplicity*, *scalability*, dan *reusable components*, ReactJS menjadi pilihan ideal untuk membangun sebuah website [8].



Gambar 2.5 Logo *React*

2.6 *Tailwind*

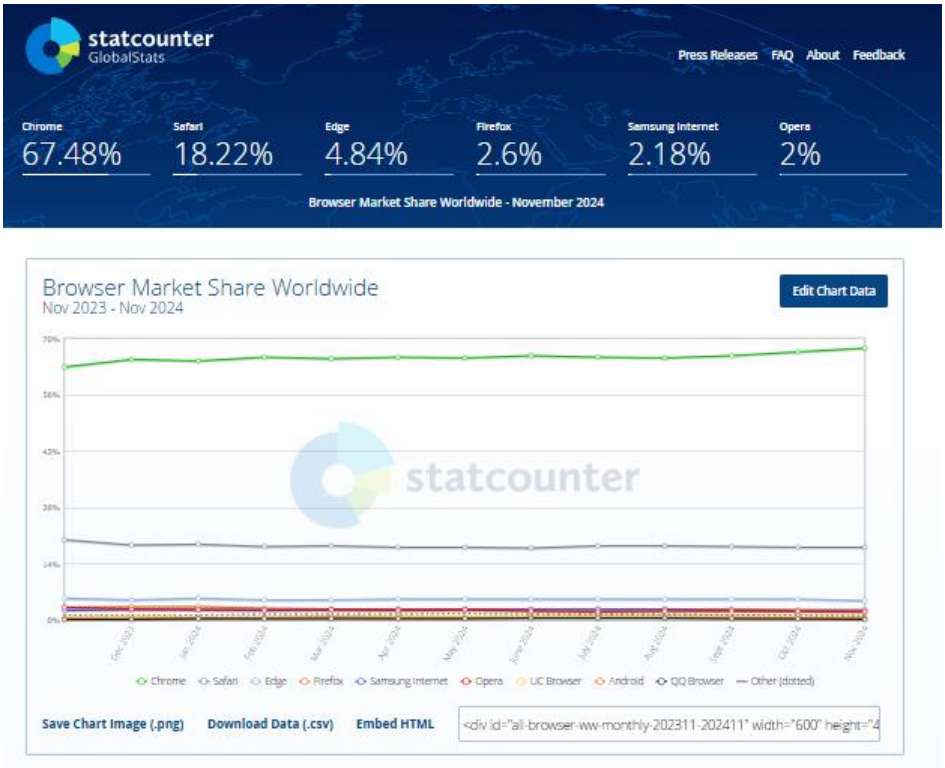
Tailwind CSS merupakan *framework* CSS yang bersifat *utility first* untuk membangun desain antarmuka khusus dengan cepat. *Tailwind* CSS berjalan diatas *JavaScript* yang bekerja diatas server *NodeJS* sehingga dalam prosesnya *File* CSS akan diterjemahkan (*Compile*) ke dalam Bahasa *JavaScript* dan dibaca oleh *File Config* yang dimiliki *Tailwind*, yang seterusnya dikembalikan menjadi sebuah *File* CSS yang lebih tersusun dan berukuran relatif lebih kecil karena melalui proses *Minify*, dan *Prefixing*. Menurut penelitian Fadli Rifandi berjudul *Website Gallery Development Using Tailwind* CSS *Framework* yang menggunakan *Tailwind* sebagai *framework* CSS di galeri *website*, dapat disimpulkan bahwa pengguna dapat mempersingkat waktu pengerjaan *style* CSS dan memberikan kebebasan untuk mendesain sesuai keinginan, seperti membuat tampilan yang *responsive* atau *mobile friendly* [15].

2.7 *JavaScript Object Notation* (JSON)

JSON merupakan format pertukaran data yang ringan dan dirancang untuk memudahkan komunikasi antar sistem. Format ini memiliki struktur yang sederhana, sehingga mudah dibaca dan ditulis oleh manusia, serta dapat dengan cepat diolah oleh komputer. JSON bersifat independen dari bahasa pemrograman tertentu, yang berarti format ini dapat digunakan untuk pertukaran data di antara berbagai platform dan teknologi yang berbeda. Fleksibilitas ini menjadikan JSON sangat populer dalam pengembangan aplikasi modern, terutama dalam integrasi API, komunikasi antar layanan, dan penyimpanan data. Berkat strukturnya yang berbasis *key value pairs*, JSON menjadi pilihan ideal untuk menyampaikan informasi yang terorganisir dengan baik dalam berbagai konteks pengembangan perangkat lunak [16].

2.8 *Google Lighthouse*

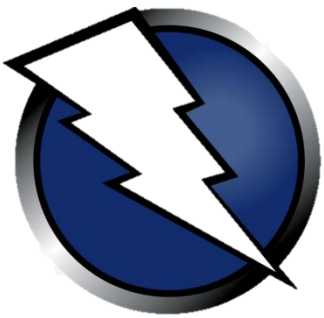
Lighthouse adalah alat audit *open source* otomatis untuk meningkatkan kualitas halaman web. Alat ini memberikan cara yang jelas untuk meningkatkan kualitas situs dengan memungkinkan *developer* menjalankan audit untuk performa, aksesibilitas, kompatibilitas progressive web app, dan lainnya. Kategori audit SEO dalam *Lighthouse* memungkinkan *developer* dan *webmaster* menjalankan *health check* SEO dasar untuk halaman web mana pun guna mengidentifikasi potensi area yang perlu ditingkatkan [17]. *Lighthouse* dijalankan secara lokal di *browser* seperti *google chrome, mozilla firefox,* dan *microsoft edge,* yang menurut data *StatCounter* adalah tiga *browser* paling populer di dunia [18]. Hal ini memungkinkan pengujian pada halaman dalam lingkungan staging, halaman aktif, halaman publik, dan halaman yang memerlukan autentikasi.



Gambar 2.6 *Browser Market Share Worldwide*

2.9 OWASP ZAP (*Zed Attack Proxy*)

OWASP ZAP (*Zed Attack Proxy*) merupakan sebuah aplikasi *open source* yang digunakan untuk *penetration testing* dalam menemukan *vulnerabilities* keamanan pada suatu aplikasi *website*. ZAP menyediakan scanner secara automatis. OWASP ZAP adalah sebuah *tools vulnerabilities scanner* yang dibuat oleh organisasi *Open Web Application Security Project* (OWASP) tools ini adalah suatu proyek dari OWASP yang paling aktif karena terus dikembangkan [19].



Gambar 2.7 Logo OWASP ZAP

2.10 *Visual Studio Code*

Visual Studio Code (VS Code) adalah *text editor* sumber terbuka yang dikembangkan oleh *Microsoft* dengan tujuan untuk memberikan pengalaman pengembangan yang efisien dan mudah digunakan. *Text editor* ini terkenal karena kemampuannya untuk menyesuaikan antarmuka pengguna, memfasilitasi kolaborasi tim, dan menghadirkan banyak fitur terbaru untuk menulis dan mengedit kode. *VS Code* tidak hanya mendukung bahasa pemrograman populer seperti *JavaScript*, *Python*, dan *Java*, tetapi juga menawarkan dukungan untuk bahasa lain seperti *Go, Rust*, dan R. *VS Code* menawarkan fitur *IntelliSense* yang memudahkan pengembang dalam menulis kode dengan cepat dan mudah. Fitur ini memungkinkan *editor* untuk memprediksi apa yang akan ditulis pengguna dan memberikan saran atau petunjuk yang sesuai secara otomatis. Selain itu, *VS Code* juga menyertakan *debugger* yang memungkinkan pengembang untuk melakukan *debug* pada kode dan menemukan kesalahan atau *bug* yang terjadi. *VS Code* juga menyediakan fitur seperti *Extension Marketplace* yang memungkinkan pengembang untuk mencari dan menginstal ekstensi atau *plugin* yang dapat meningkatkan kemampuan *editor*. Ekstensi ini memungkinkan pengembang untuk menyesuaikan dan memperluas kemampuan *editor*, mempercepat proses pengembangan, dan membuat pengembangan menjadi lebih efisien [20].

2.11 *Unified Modelling Language*

Unified Modeling Language (UML) merupakan sebuah bahasa standar yang bertujuan untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak [21]. UML dapat digunakan untuk semua metode pengembangan dan tahapan siklus hidup perangkat lunak. UML merupakan sekumpulan *diagram* untuk menggabungkan praktik terbaik dari perangkat lunak yang ingin dikembangkan dengan standar yang biasanya dilakukan. Dalam UML, *diagram* terbagi menjadi dua yaitu *Structural Diagram* dan *Behavioral Diagram* dan di dalamnya terdapat banyak *diagram* seperti *class, use case, sequence, activity* dan lainnya.

a. *Use Case Diagram*

Use Case Diagram adalah sebuah *diagram* yang menggambarkan fungsional dari sebuah sistem, serta hubungan antara aktor dengan *use case* di dalam sistem tersebut. *Use case* merupakan sebuah pekerjaan tertentu yang dapat terjadi dalam sebuah sistem dan penting untuk memvisualisasikan perilaku suatu objek [21]. Dengan adanya *Use Case Diagram*, penyajian gambaran tentang bagaimana sebuah objek dapat bekerja pada sistem perangkat lunak jadi lebih mudah. Tabel 2.1 di bawah ini menjelaskan komponen-komponen yang ada pada *Use Case Diagram*.

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri*(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Tabel 2.1 Komponen *Use Case Diagram*

b. *Activity Diagram*

Activity Diagram merupakan *diagram* yang memvisualisasikan aktivitas pada sistem, mulai dari awal sistem berjalan, pilihan yang mungkin akan terjadi, dan bagaimana akhir dari sistem tersebut. *Activity Diagram* adalah *state diagram* khusus, dimana banyak dari *state* nya merupakan aksi dengan munculnya sebuah transisi akibat *state* yang sebelumnya selesai [21]. Tabel 2.2 di bawah ini menjelaskan komponen-komponen yang ada pada *Activity Diagram*.

NO	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran
6		*Join Node*	Banyak aliran yang pada tahap tertentu berubah menjadi beberapa satu aliran
7		Decision	Pilihan untuk mengambil keputusan

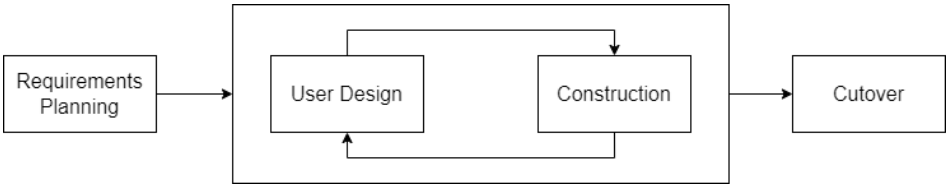
Tabel 2.2 Komponen *Actvity Diagram*

2.12 *Rapid Application Development* (RAD)

Rapid Application Development (RAD) adalah sebuah model proses perkembangan perangkat lunak sekuensial linier yang menekankan siklus perkembangan yang sangat pendek. Model RAD digunakan untuk proyek-proyek yang memerlukan pengembangan perangkat lunak yang cepat dan mempunyai

kebutuhan yang berubah-ubah [22]. Pemilihan metode RAD pada penelitian ini dikarenakan RAD memungkinkan keterlibatan langsung pengguna dalam proses pengembangan sistem. Jadi, pengguna dapat memberikan saran serta *feedback* secara langsung selama tahap-tahap pengembangan, sehingga aplikasi yang dibuat dapat sesuai dengan kebutuhan dan harapan pengguna.

Gambar 2.7 di bawah ini menggambarkan tahapan pada *Rapid Application Development* (RAD).



Gambar 2.8 Tahapan *Rapid Application Development*

Metode RAD terdiri dari empat tahapan, yaitu:

a. *Requirements Planning*: Tahapan awal dalam Model RAD adalah tahapan perencanaan kebutuhan. Tahapan ini dilakukan untuk mengidentifikasi dan memahami kebutuhan fungsional dan non fungsional pengguna.
b. *User Design*: Tahapan kedua adalah tahap perancangan arsitektur sistem. Tahapan ini melibatkan desain teknis tentang bagaimana komponen sistem bekerja satu sama lain dan bagaimana perangkat lunak bekerja.
c. *Construction*: Tahapan ketiga adalah tahap konstruksi atau membangun sistem. Dalam tahap ini, pengembang perangkat lunak membangun *prototype* atau model awal dari perangkat lunak. Model ini digunakan untuk mendapatkan umpan balik dari klien dan memperbaiki perangkat lunak sesuai dengan umpan balik tersebut.
d. *Cutover*: Tahapan terakhir dalam Model RAD adalah tahap implementasi. Tahapan ini merupakan peralihan sistem dari *prototype* ke perangkat lunak yang lebih baik. Perangkat lunak yang telah selesai dibangun, selanjutnya siap untuk diimplementasikan [23].

2.13 *User Experience Questionnaire (UEQ)*

User Experience Questionnaire adalah salah satu metode pengujian menggunakan kuesioner yang berguna untuk mengetahui tingkat *user experience* dari produk yang dikembangkan [24]. Terdapat 6 skala penilaian dari total 26 pertanyaan yang perlu dijawab pengguna dengan 7 pilihan jawaban. Adapun 6 skala penilaian tersebut, yaitu:

1. *Attractiveness* (Daya Tarik): Kesan secara keseluruhan apakah pengguna menyukai produk tersebut atau tidak?
2. *Perspicuity* (Kejelasan): Seberapa mudah produk tersebut untuk dipahami dan dipelajari oleh pengguna?
3. *Efficiency* (Efisiensi): Mengetahui apakah pengguna dapat menyelesaikan tugasnya dengan mudah dan tidak kesulitan atau membutuhkan bantuan?
4. *Dependability* (Ketepatan): Penilaian untuk mengetahui apakah pengguna dapat mengendalikan interaksi dengan produk tersebut atau pengguna dengan mudah terarah untuk melakukan interaksi dengan produk?
5. *Stimulation* (Stimulasi): Secara keseluruhan apakah produk tersebut menarik dan memotivasi untuk digunakan?
6. *Novelty* (Kebaruan): Seberapa inovatif dan kreatif dan apakah produk tersebut berhasil menarik minat pengguna?

2.14 *Blackbox Testing*

Blackbox Testing merupakan teknik pengujian perangkat lunak yang berfokus pada spesifikasi fungsional dari perangkat lunak, penguji tidak memiliki pengetahuan tentang desain internal dan tidak memiliki akses ke *source code*. Teknik ini digunakan untuk memastikan bahwa semua input yang dibutuhkan oleh sistem diterima dengan cara yang ditentukan dan memberikan *output* yang benar [25]. Keuntungan penggunaan metode *Blackbox Testing* adalah [26]:

1. Penguji tidak harus mengetahui tentang bahasa pemrograman tertentu.
2. Pengujian dilakukan dari perspektif pengguna untuk membantu mengidentifikasi ambiguitas atau inkonsistensi dalam spesifikasi persyaratan.
3. Programmer dan tester keduanya saling bergantung satu sama lain.

2.15 Penelitian Terkait

Terdapat beberapa penelitian terkait yang dijadikan sebagai perbandingan serta rujukan mengenai metode serta hasil yang dicapai pada penelitian ini. Berikut merupakan ulasan dari beberapa penelitian terkait.

2.15.1 Evaluation of User experience in Integrated Learning Information Systems Using User Experience Questionnaire (UEQ)

Penelitian dengan judul "*Evaluation of User experience in Integrated Learning Information Systems Using User Experience Questionnaire (UEQ)".* Diambil dari *Journal of Information Systems and Informatics*, diteliti oleh Arista Pratama, Asif Faroqi, dan Eka Prakarsa Mandyartha pada tahun 2022 di Universitas Pembangunan Nasional "Veteran" Jawa Timur yang membahas mengenai analisis pengalaman pengguna Sistem Informasi Pembelajaran Terpadu menggunakan metode *User Experience Questionnaire* (UEQ). Hasil evaluasi pengalaman pengguna pada sistem informasi pembelajaran menggunakan metode UEQ menyatakan bahwa daya tarik adalah 2.121, kemudahan dipahami adalah 2.152, efisiensi adalah 2.319, ketergantungan adalah 1.505, stimulasi adalah 1.716, dan kebaruan adalah 1.020. Hasil benchmark menunjukkan bahwa aspek daya tarik, kemudahan dipahami, efisiensi, dan stimulasi termasuk dalam kriteria sangat baik, sedangkan aspek ketergantungan termasuk dalam kriteria baik dan kebaruan termasuk dalam kriteria di atas rata-rata [27].

2.15.2 Preliminary Design of Website Application for Environmental and Radiation Monitoring Using React

Penelitian dengan judul "*Preliminary Design of Website Application for Environmental and Radiation Monitoring Using React".* Diambil dari *Nuclear Facility Engineering Center,* diteliti oleh Ismet Isnaini dan Muhammad Naufal Shidqi pada tahun 2020, yang membahas aplikasi pemantau lingkungan dan radiasi yaitu Radmon yang dirancang menggunakan paltform Grafana. Walaupun sangat mudah digunakan Grafana memiliki limitasi dalam hal kustomisasi. *React* merupakan salah satu kerangka *JavaScript* populer untuk membuat *website*,

dengan fitur JSX dapat membuat halaman dinamis dengan mudah dan karena berbasis komponen *React* dapat digunakan untuk membuat berbagai macam komponen dari paket npm sehingga memberikan banyak pilihan untuk mengkustomisasi fungsi serta tampilan dari aplikasi. Pembuatan aplikasi web pemantau menggunakan *React* memberikan aplikasi kemampuan kustomisasi yang sangat tinggi [28].

2.15.3 Blocking Prohibited Sites Using Proxy Server In Mikrotik And Squid Proxy Debian Server, At The Nurul Anwar Education Foundation, Tanjungbalai City

Penelitian dengan judul "*Blocking Prohibited Sites Using proxy server in Mikrotik and Squid Proxy Debian Server, at The Nurul Anwar Education Foundation, Tanjungbalai City"*, diteliti Zulham Sitorus, Mhd Arfan Sitorus, Didi Riswan, dan Heri Eko Rahmadi Putra pada tahun 2024 di Univeristas Pembangunan Panca Budi, yang membahas pentingnya penggunaan *proxy server* untuk menyaring akses internet dari situs-situs terlarang, seperti judi online, pornografi, dan konten kekerasan. Tanpa adanya pembatasan, situs-situs tersebut dapat diakses dengan mudah, terutama oleh anak-anak di bawah umur, yang dapat merusak moral mereka [29].

2.15.4 Web-Based Application Development for Training Data Management Using ReactJS

Penelitian dengan judul "*Web-Based Application Development for Training Data Management Using ReactJS",* diteliti oleh Hany Juliana Saragih Sitio, Irene Christovita, Rahma Kamila Ahmad, dan Yanto Setiawan pada tahun 2023 di Universitas Bina Nusantara, Jakarta yang membahas mengenai pengembangan sistem informasi berbasis web untuk manajemen data pelatihan guna meningkatkan program pelatihan dan peluang bisnis bagi perusahaan. Desain aplikasi berbasis web menggunakan *ReactJS* dan MySQL sebagai basis data, dengan bahasa pemrograman Java dan *JavaScript*. Pengembangan sistem informasi berbasis web ini bertujuan untuk mempermudah pengelolaan data pelatihan yang telah dilaksanakan sesuai jadwal dan timeline program pelatihan.

Dari hasil pengujian integrasi sistem (SIT) dan pengujian penerimaan pengguna (UAT), dapat disimpulkan bahwa sistem informasi yang dikembangkan meningkatkan efektivitas sebesar 90.4%, efisiensi sebesar 91.2%, dan kepuasan pengguna oleh karyawan dan admin sebesar 92.6%. Hasil evaluasi yang telah dilakukan menunjukkan bahwa aplikasi ini sangat layak digunakan, yaitu sebesar 91.5% [30].

2.15.5 Coastal and Marine Tourism Monitoring System Design using Rapid Application Development (RAD)

Penelitian dengan judul "*Coastal and Marine Tourism Monitoring System Design using Rapid Application Development (RAD)"*, diteliti oleh Yerik Afrianto Singgalen pada tahun 2023, yang membahas perancangan sistem informasi dan basis data untuk memantau kegiatan pariwisata pantai guna meminimalkan dampak lingkungan, terutama di Pantai Luari, Halmahera Utara. Metode yang digunakan adalah *Rapid Application Development* (RAD) dengan tahapan perencanaan kebutuhan, desain pengguna, konstruksi, dan *Cutover*. Hasil penelitian menunjukkan pentingnya memantau dan mengelola aktivitas pariwisata secara efektif sehingga dapat membantu pengelola destinasi dalam membuat kebijakan pengembangan infrastruktur yang sesuai [31].

2.15.6 **Rancang Bangun Sistem Informasi Monitoring Rencana Strategis Bisnis Bank X Menggunakan Metode RAD**

Penelitian dengan judul "Rancang Bangun Sistem Informasi Monitoring Rencana Strategis Bisnis Bank X Menggunakan Metode RAD*"*. Diambil dari Jurnal Ilmiah Informatika Komputer, diteliti oleh Dimas Nugeroho pada tahun 2019 di Depok, yang membahas masalah keterlambatan dalam penyampaian target bisnis kepada pemegang keputusan strategis akibat gangguan pada proses *End of Day* dan *End of Month* pada sistem perbankan inti. Keterlambatan ini berpotensi menyebabkan hilangnya pendapatan dan peluang bisnis. Solusi yang diusulkan adalah pengembangan sistem informasi monitoring rencana strategis yang menyediakan laporan keuangan dan pencapaian target bisnis secara *real-time*. Pengembangan sistem ini menggunakan metode *Rapid Application Development* (RAD), dan

hasil uji coba menunjukkan bahwa semua fungsi pada aplikasi berjalan dengan baik [32].

2.15.7 Model *Rapid Application Development* (RAD) Untuk Rancang Bangun Sistem Informasi Monitoring *Project* Pada Branch Business Process Re-Engineering (BBPR) Team

Penelitian dengan judul "Model *Rapid Application Development* (RAD) Untuk Rancang Bangun Sistem Informasi Monitoring *Project* Pada Branch Business Process Re-Engineering (BBPR) Team*"*. Diambil dari *Journal of Social Science Research (INNOVATIVE),* diteliti oleh Rika Astuti pada tahun 2023 di Universitas Siber Indonesia yang membahas mengenai pengembangan sistem informasi monitoring *Project* Pada *Branch Business Process Re-Engineering* (BBPR) Team. Dalam pengembangan sistem menggunakan model RAD (*Rapid Application Development*), dimana model ini merupakan pengembangan yang dapat dilakukan untuk proyek skala kecil dengan waktu yang lebih singkat. Hasil dari Model *Rapid Application Development* (RAD) untuk rancang bangun sistem informasi monitoring *Project* pada *branch business process re engineering* (BBPR) team yaitu sistem dapat menghasilkan informasi rincian *Project*, data *Project* dalam bentuk bar chart (*diagram* batang), yang melibatkan stream/divisi, nama inisiatif, target waktu *end state*, *persentase end state*, *status pace*, waktu *status pace*, *persentase end state*, dan total *persentase end state* yang telah dikerjakan secara tepat dan efisien [33].

2.15.8 Pengujian Aplikasi dengan Metode *Blackbox Testing* Boundary Value Analysis

Penelitian dengan judul "Pengujian Aplikasi dengan Metode *Blackbox Testing* Boundary Value Analysis*"*. Diambil dari Jurnal Pengembangan IT (JPIT), diteliti oleh Tri Snadhika Jaya pada tahun 2018 di Politeknik Negeri Lampung yang membahas mengenai pengujian perangkat lunak kantor digital di Politeknik Negeri Lampung. Proses pengujian dilakukan untuk mengetahui tingkat kesalahan yang terjadi pada perangkat lunak. Pengujian ini menggunakan *Blackbox Testing* boundary value analysis. Boundary Value Analysis merupakan jenis test case

dengan menentukan nilai normal, nilai minimal, dan nilai maksimal dari data yang akan diuji. Aplikasi mampu menangani data, baik data normal ataupun data tidak normal dengan persentase keberhasilan 91,67 % [26].

2.15.9 Penerapan Teknologi Cache Server Berbasis Iot Dengan Raspberry Pi3 Menggunakan Metode Forward Chainning (Studi Kasus SMK Binakarya Mandiri 2 Kota Bekasi)

Penelitian dengan judul "Penerapan Teknologi Cache Server Berbasis IOT Dengan Raspberry Pi3 Menggunakan Metode *Forward Chaining* (Studi Kasus SMK Bina Karya Mandiri 2 Kota Bekasi)*"*. Diambil dari JURNAL KILAT, diteliti oleh Subandri dan Sabar Hanadwiputra pada tahun 2018 di STMIK Bani Saleh yang membahas mengenai penggunaan *cache* pada *proxy server* untuk mempercepat waktu akses *internet* dan *filtering website*. Aplikasi yang digunakan untuk membuat *proxy server* ini adalah *squid* yang berjalan di *raspberry pi3* dengan sistem operasi *rasbian*. Aplikasi *squid* paling banyak digunakan di jaringan lokal maupun *internet* baik untuk menyimpan *cache web*, blokir beberapa *website* berbahaya, manajemen *bandwidth,* hingga ke pembatasan akses [5].

2.15.10 Rancang Bangun Sistem Informasi Pembayaran SPP Pada Sekolah Sepak Bola Tasbi Dengan Menggunakan *React* Dan NodeJS

Penelitian dengan judul "Rancang Bangun Sistem Informasi Pembayaran SPP Pada Sekolah Sepak Bola Dengan Menggunakan *React* Dan NodeJS*"*. Diambil dari Jurnal Teknologi dan Sains, diteliti oleh M Imam Santoso pada tahun 2023 di Universitas Panca Budi Medan yang membahas kendala yang dihadapi oleh Sekolah Sepak Bola Tasbi dalam pengelolaan data, terutama dalam pengolahan pembayaran iuran sekolah yang masih dilakukan secara manual. Untuk meningkatkan efektivitas dan efisiensi, diperlukan sistem informasi pembayaran yang terkomputerisasi dan berbasis web. Sistem ini dirancang menggunakan Unified Model Language (UML), dengan MySQL sebagai database, Visual Studio Code sebagai text editor, *React* sebagai *Frontend,* dan NodeJS sebagai backend. Hasilnya adalah sistem informasi pembayaran berbasis web yang

mempermudah proses pembayaran iuran dan pembuatan laporan keuangan bulanan di Sekolah Sepak Bola Tasbi [34].

2.16 *State of The Art*

State of The Art merupakan hasil analisa yang didapatkan dari penelitian sebelumnya yang memiliki keterkaitan dengan penelitian yang akan dilakukan. Hasil analisis ini kemudian dijadikan referensi perbaikan untuk menciptakan sistem yang lebih baik dari penelitian sebelumnya. Berdasarkan penelitian [30] dan [34], menyatakan bahwa penggunaan *ReactJS* menghasilkan render yang lebih cepat dan efisien sehingga cocok untuk membuat sistem *monitoring*. Berdasarkan penelitian [29] dan [5], menyatakan bahwa sistem *monitoring* Cache pada *proxy server* membantu mengidentifikasi di mana terjadi penundaan atau kemacetan dalam akses data sehingga dapat diambil tindakan untuk mengatasinya. Pada penelitian [28] dan [31], menyatakan bahwa sistem *monitoring* memerlukan iterasi cepat untuk menanggapi kebutuhan pengguna. Berdasarkan penelitian [32] dan [33] menyatakan bahwa *Rapid Application Development* adalah metode yang paling efektif dan efisien dalam mengembangkan perangkat lunak yang tujuan utamanya adalah untuk menghasilkan produk dengan cepat melalui *prototyping*, *feedback* berulang dari pengguna, dan iterasi. Oleh karena itu, pada penelitian ini yang memerlukan kemampuan *render* yang cepat, diputuskan untuk merancang sistem *monitoring cache* pada *proxy server* menggunakan *ReactJS* dan metodologi *Rapid Application Development* (RAD).

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat

Penelitian dilaksanakan mulai dari bulan Juli 2024 sampai dengan bulan Oktober 2024 yang bertempat di kantor PT. Queen Network Nusantara.

Tabel 3.1 Jadwal Penelitian

No	Aktivitas	Juli	Agustus	September	Oktober
1	*Requirements Planning*				
2	*User Design*				
3	*Construction* (Iterasi 1 – Iterasi 2)				
4	*Cutover*				
5	Pelaporan				

3.2 Alat dan Bahan Penelitian

3.2.1 Alat Penelitian

Adapun alat yang digunakan pada penelitian ini dapat dilihat pada Tabel 3.2 berikut:

Tabel 3.2 Alat Penelitian

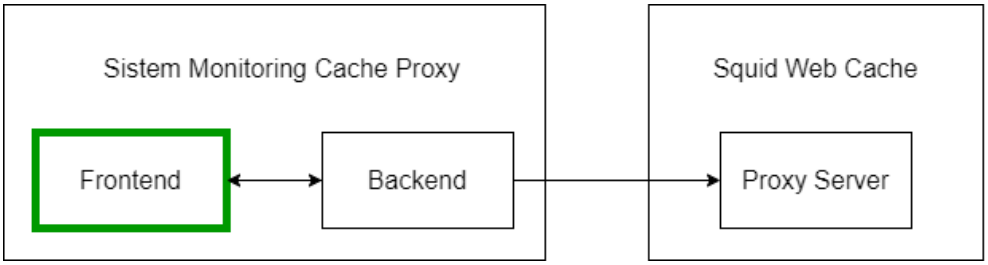
No.	**Nama Alat**	**Spesifikasi**	**Deskripsi**
1.	Laptop	Prosesor Intel Core i3-7020U, Memory 4GB, Nvidia GeForce MX 230, sistem operasi Windows	Perangkat keras untuk membangun aplikasi
2.	Visual Studio Code	Versi 1.78.2	Perangkat lunak untuk menuliskan baris kode aplikasi
3.	*ReactJS*	Versi 18.2.0	*Library* untuk merancang

			sistem *monitoring*
7	*Draw.io*	Versi 20.8.16	Perangkat lunak untuk merancang *UML (Unified Modelling Language)*
8	*Figma*	Versi 28.4.12.3	Perangkat lunak untuk merancang antarmuka *sistem*
9	ZAP	Versi 2.15.0	Perangkat lunak untuk menguji *vulnerabilities* pada *website*

3.2.2 Bahan Penelitian

Bahan yang digunakan dalam penelitian adalah *proxy server* yang disediakan oleh PT. Queen Network Nusantara.

3.3 *Capstone Project*



Gambar 3.1 *Capstone Project*

Gambar 3.1 merupakan gambar *capstone project* sistem *monitoring cache* pada *proxy server*. Pengembangan sistem *monitoring cache* pada *proxy server* dimulai dengan implementasi dan konfigurasi pada *proxy server*, berlanjut ke pengembangan *backend*, dan diakhiri dengan integrasi *frontend*.

- Pertama, tim *proxy server* bertanggung jawab untuk menginstal dan mengonfigurasi *Squid Web Cache* di sistem operasi *FreeBSD*. *Squid* digunakan untuk menangani *cache hit* dan *cache miss* dari objek yang sering diakses oleh pengguna. *Squid* dikonfigurasi untuk menghasilkan berbagai log, seperti *access log, store log, user agent log, dan cache log* yang berfungsi untuk mencatat aktivitas proxy. Setelah konfigurasi selesai, dilakukan pengujian untuk memastikan bahwa sistem mampu menangani beban lalu lintas jaringan dengan optimal dan menyimpan log dengan benar. Penanggung jawab: Pegawai QNN.

- Selanjutnya, tim *backend* bertugas mengembangkan *endpoint* API yang berfungsi sebagai penghubung antara *proxy server* dan *frontend*. *endpoint* API ini dikembangkan menggunakan *Django Rest Framework* dan terhubung dengan *proxy server* melalui SSH. Melalui koneksi ini, *endpoint* API mengambil data log dari Squid sesuai permintaan. Data yang diperoleh, lalu diolah dan dikonversi menjadi format JSON (*JavaScript Object Notation*), yang kemudian disajikan kepada *frontend*. *Backend* juga memastikan bahwa data log dapat difilter berdasarkan kriteria tertentu seperti *timestamp*, status cache (hit/miss), dan alamat IP pengguna. Penanggung jawab: M. Nawwir Albi.
- Setelah *endpoint* API siap, tim *frontend* mengembangkan antarmuka menggunakan *ReactJS*, dengan tujuan menampilkan data *real-time* dari *endpoint* API dalam bentuk grafik dan tabel. *Frontend* mengintegrasikan *endpoint* API untuk menampilkan data log secara dinamis, dengan *WebSockets* untuk memperbarui data secara *real-time*. Antarmuka dirancang agar intuitif dan mudah digunakan, menyediakan fitur filter bagi pengguna untuk melihat data log berdasarkan parameter spesifik. *Dashboard* juga menyajikan *chart* untuk menampilkan data secara keseluruhan. Penanggung jawab: M. Aldi Kurniawan.

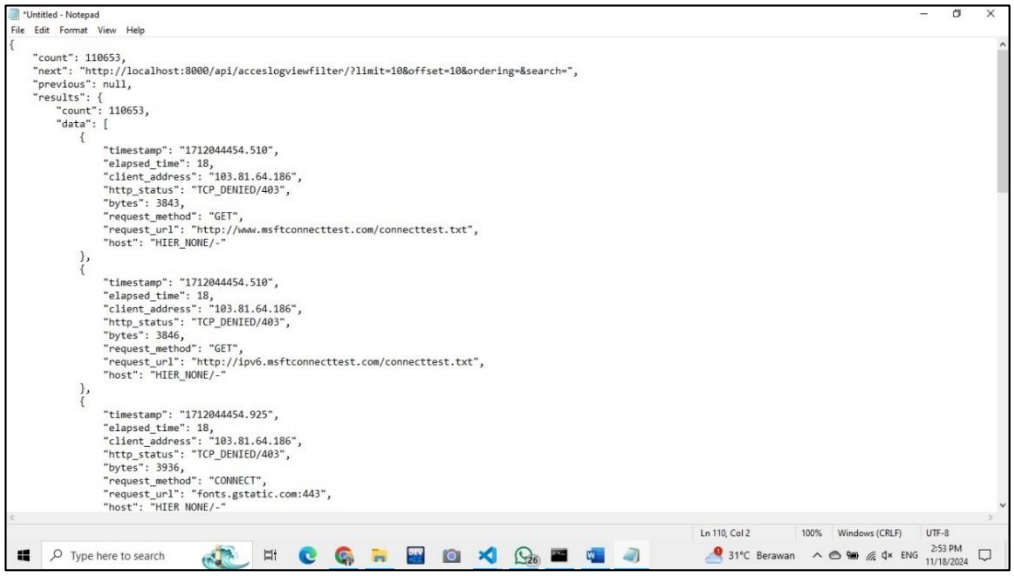
3.3.1 JSON API

Tim *backend* menyediakan *endpoint* dalam format JSON untuk mendukung komunikasi antara klien dan *server*. *Endpoint* ini dirancang untuk mengembalikan data yang relevan dengan format yang terstruktur, sehingga memudahkan integrasi dengan antarmuka pengguna atau aplikasi lain yang memanfaatkan API tersebut.

3.3.1.1 Data JSON *Access Log*

Endpoint yang tersedia untuk *access log* memungkinkan pengguna untuk mendapatkan data *log* secara terperinci. Data ini meliputi informasi seperti waktu akses, alamat klien, status HTTP, URL yang diminta, dan *byte* data yang digunakan. Struktur utama JSON terdiri dari beberapa elemen penting, *count* menunjukkan total jumlah data yang tersedia, dengan nilai 116053, *next* adalah

URL untuk mengambil data halaman berikutnya menggunakan *pagination*, sedangkan *previous* bernilai *null*, menandakan bahwa ini adalah halaman pertama. Elemen *results* adalah array yang berisi objek-objek log. Setiap objek dalam array memiliki properti data, yang menyimpan informasi log spesifik. Berikut adalah penjabaran rinci mengenai data access log berdasarkan dokumentasi Squid yang telah dijelaskan pada tinjauan pustaka.



Gambar 3.2 Data JSON *Access Log*

1. *Time* (Waktu Permintaan)

Merupakan waktu terjadinya permintaan yang dicatat dalam format *Unix timestamp. Unix timestamp* adalah jumlah detik yang telah berlalu sejak 1 Januari 1970 pukul 00:00:00 UTC, dengan akurasi hingga milidetik. Format ini umum digunakan dalam sistem komputasi karena sifatnya yang linier dan mudah diproses. Berikut adalah cara konversi *Unix timestamp*.

- Ambil jumlah detik dan bagi dengan 60 untuk mendapatkan menit.
- Kemudian bagi hasilnya lagi dengan 60 untuk mendapatkan jam.
- Bagi lagi dengan 24 untuk mendapatkan hari (dan tahun jika diperlukan).
- Dengan *timestamp* 1722911319, hasil konversi adalah 23 Juni 2024 pukul 16:21:59 UTC.

2. *Duration* (Durasi Pemrosesan)

Menunjukkan waktu total (dalam milidetik) yang dibutuhkan oleh *Squid* untuk memproses permintaan. Durasi dihitung sejak permintaan diterima hingga respons selesai dikirimkan ke klien. Contoh: Jika nilai 18**,** berarti permintaan hanya membutuhkan waktu 18 milidetik untuk diproses, menunjukkan efisiensi tinggi dari *Squid Proxy*.

3. *Client Address* (Alamat Klien)

Merupakan alamat IP atau nama host dari klien (perangkat pengguna) yang mengirimkan permintaan ke *Squid Proxy*. Contoh: Alamat IP seperti 103.81.64.186 menunjukkan perangkat pengguna yang terhubung ke jaringan dan menggunakan layanan *proxy*.

4. *Result Codes* (Kode Hasil)

Kode status ini memberikan informasi tentang bagaimana *Squid* memproses permintaan, diikuti oleh kode status HTTP untuk respons. Contoh:

- TCP_HIT/200: Data ditemukan di *cache Squid* dan respons diberikan dengan status HTTP 200 OK.
- TCP_MISS/200: Data tidak ditemukan di *cache*, diteruskan ke server asal, dan respons berhasil (200 OK).
- TCP_DENIED/403: Permintaan diblokir berdasarkan aturan akses, menghasilkan status 403 *Forbidden*.
- TCP_REFRESH_HIT/304: Data di cache diperbarui dari server asal karena sudah kedaluwarsa, menghasilkan status 304 *Not Modified*.

5. *Bytes* (Ukuran Data)

Jumlah *byte* data yang ditransfer dari server ke klien dalam satu permintaan, termasuk header HTTP dan konten. *Bytes* dapat dikonversi ke unit yang lebih besar (seperti KB, MB, atau GB) menggunakan sistem konversi *biner*, dimana 1 *kilobyte* (KB) = 1024 *bytes* dan 1 *megabyte* (MB) = 1024 *kilobytes* (KB)**.** Contoh: Jika nilai 346, berarti total data sebesar 346/1024 = 0.34 *kilobytes* (KB) dikirimkan.

6. *Request Method* (Metode Permintaan)

Menunjukkan metode HTTP yang digunakan klien untuk mengirimkan permintaan ke server melalui Squid.

- GET: Digunakan untuk meminta data atau sumber daya.
- POST: Digunakan untuk mengirim data ke server, misalnya formulir atau unggahan file.
- CONNECT: Membuka koneksi tunneling, biasanya untuk koneksi aman (HTTPS).

7. URL (Alamat Permintaan)

URL lengkap yang diminta oleh klien, mencakup protokol (HTTP/HTTPS), nama host, dan path sumber daya. Contoh:

- *http://www.msftconnecttest.com/connecttest.txt*: Meminta file uji koneksi dari *Microsoft*.
- *https://example.com/resource.pdf*: Meminta file PDF.

8. *Hierarchy Code* (Kode Hierarki)

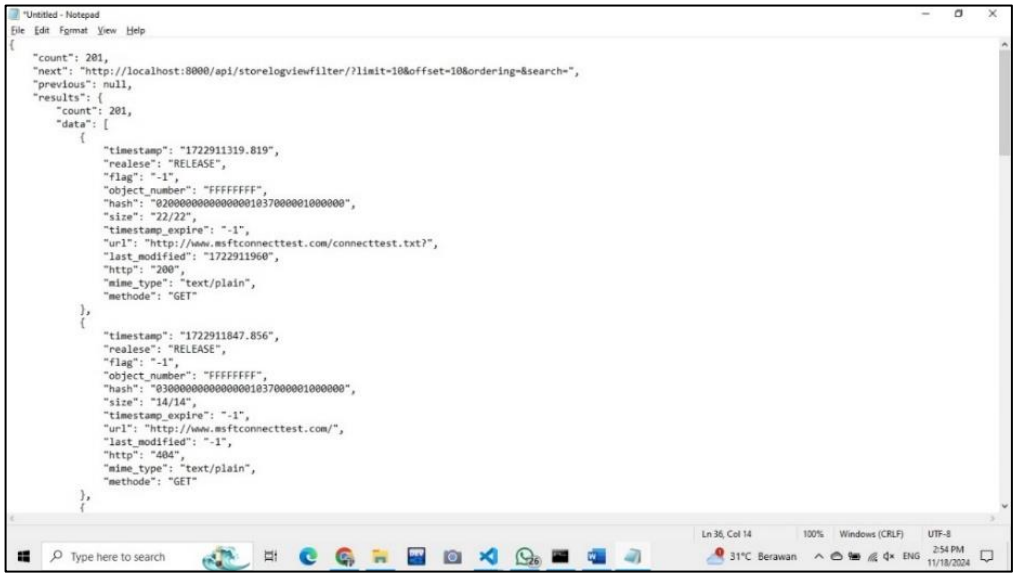
Kode ini memberikan informasi tentang sumber data permintaan atau cara permintaan ditangani oleh hierarki *proxy*.

- DIRECT: Permintaan diteruskan langsung ke server asal tanpa melalui *proxy* lain.
- NONE: Tidak ada informasi hierarki yang relevan.
- PARENT_HIT: Permintaan diteruskan ke *proxy* induk yang memiliki *cache* data. Data JSON *access log* ditunjukkan pada Gambar 3.2.

3.3.1.2 Data JSON *Store Log*

Endpoint yang tersedia untuk *store log* memungkinkan pengguna untuk mendapatkan data *log* secara terperinci terkait aktivitas penyimpanan dan pelepasan objek dalam sistem cache. Data ini meliputi informasi waktu dan status pemrosesan (timestamp, release, flags), detail file atau URL yang diminta (url, mime_type, http), ukuran data yang ditransfer (size), dan informasi tambahan seperti metode HTTP dan waktu kedaluwarsa cache. Struktur utama JSON terdiri dari beberapa elemen penting, *count* menunjukkan total jumlah data yang tersedia,

dengan nilai 201, *next* adalah URL untuk mengambil data halaman berikutnya menggunakan *pagination*, sedangkan *previous* bernilai *null*, menandakan bahwa ini adalah halaman pertama. Elemen *results* adalah array yang berisi objek-objek log. Setiap objek dalam array memiliki properti data, yang menyimpan informasi log spesifik. Berikut adalah penjabaran rinci mengenai data *store log* berdasarkan dokumentasi Squid yang telah dijelaskan pada tinjauan pustaka.



Gambar 3.3 Data JSON *Store Log*

1. *Time* (Waktu Permintaan)

Merupakan waktu terjadinya permintaan yang dicatat dalam format *Unix timestamp. Unix timestamp* adalah jumlah detik yang telah berlalu sejak 1 Januari 1970 pukul 00:00:00 UTC, dengan akurasi hingga milidetik. Format ini umum digunakan dalam sistem komputasi karena sifatnya yang linier dan mudah diproses. Berikut adalah cara konversi *Unix timestamp*.

- Ambil jumlah detik dan bagi dengan 60 untuk mendapatkan menit.
- Kemudian bagi hasilnya lagi dengan 60 untuk mendapatkan jam.
- Bagi lagi dengan 24 untuk mendapatkan hari (dan tahun jika diperlukan).
- Dengan timestamp 1722911319, hasil konversi adalah 23 Juni 2024 pukul 16:21:59 UTC.

2. Action

Menjelaskan status log permintaan yang tercatat di *Squid Proxy*. Setiap log mencatat tindakan atau status yang terjadi pada data dalam *cache*. Contoh:

- CREATE: Data disimpan ke dalam *cache* saat permintaan pertama kali diterima.
- RELEASE: Data dihapus dari *cache* setelah digunakan, biasanya karena sudah tidak diperlukan atau untuk mengosongkan ruang.
- SWAPOUT: Data yang diterima dari server asal dipindahkan ke penyimpanan lain untuk memberikan ruang bagi konten baru.
- SWAPIN: Data diambil dari penyimpanan dan memasukkannya kembali ke dalam *cache* untuk digunakan kembali.

3. Dir Number

Dir Number adalah nomor direktori tempat file cache disimpan pada sistem file *Squid. Squid* mengorganisasi cache dalam bentuk direktori dan subdirektori untuk efisiensi pengelolaan data. Contoh:

- *Dir Number* dengan nilai 00: 00 menunjukkan lokasi fisik file dalam struktur direktori *cache Squid*. Misalnya, jika direktori *cache Squid* adalah */var/cache/squid/*, maka file *cache* bisa berada di path */var/cache/squid/00/XX/*. Dimana 00 adalah *Dir Number* dan XX adalah nama file yang dihasilkan oleh *Squid* (berdasarkan hash).
- *Dir Number* dengan nilai -1: -1 menunjukkan bahwa file tidak lagi ada di disk *cache* atau hanya berada sementara di RAM. Nilai -1 sering kali digunakan sebagai *placeholder* untuk menunjukkan bahwa data tersebut sudah tidak lagi tersedia di direktori *cache disk* atau hanya berada sementara di memori.

4. File Number

Nomor file unik yang digunakan untuk mengidentifikasi data di dalam direktori cache. Nilai ini biasanya berupa bilangan heksadesimal yang digunakan untuk mengatur file secara efisien di dalam direktori cache. Contoh:

- *File Number* dengan nilai FFFFFFFF: Menandakan file tidak valid, *placeholder*, atau telah dilepaskan dari sistem *cache Squid*. Nilai FFFFFFFF biasanya berarti *Squid* tidak memiliki data fisik atau lokasi yang valid untuk

file tersebut. Objek tersebut mungkin hanya ada sebagai metadata atau sudah dihapus dari *cache*.

- *File Number* dengan nilai 00000001: Menandakan file yang valid dan disimpan di *cache*. File 00000001 adalah file pertama yang disimpan di direktori *cache*.

5. Hash

Hash merujuk pada nilai yang digunakan untuk mengidentifikasi dan mengakses objek *cache* di dalam sistem. *Hash* ini biasanya dihasilkan dengan menggunakan algoritma *hash* (misalnya MD5, SHA-1, atau lainnya) berdasarkan URL atau konten objek yang di-*cache*. Nilai *hash* digunakan untuk menentukan di mana objek *cache* disimpan di direktori *cache*. Contoh:

- *Hash* dengan nilai 02000000000000001037000001000000: *Hash* yang digunakan untuk mengidentifikasi objek *cache*, dengan struktur direktori *cache* lebih panjang dan lebih terstruktur.
- Hash dengan nilai 60F58D90BBDCE4D397FC5290AF44BDF3: *Hash* yang lebih pendek dan sering dijumpai dalam algoritma MD5 atau SHA-1, digunakan dalam struktur direktori *cache Squid*. Nilai-nilai *hash* ini penting untuk mengakses dan mengelola file *cache* di *Squid Proxy*, dan mereka sangat bergantung pada algoritma *hash* yang digunakan dalam implementasi *Squid*.

6. Size

Size merujuk pada dua ukuran berbeda terkait objek yang di-*cache*, yang biasanya disajikan dalam format dua angka. Ini mengacu pada informasi yang disimpan dalam *log* atau metadata terkait objek *cache*. Contoh:

Jika log *Squid* menunjukkan entri *Size*: 14/14. Ini berarti.

- Ukuran yang Tersimpan di Cache adalah 14 *bytes*.
- Ukuran asli adalah 14 *bytes* sebelum dikompresi atau dimodifikasi.

Ini bisa berarti bahwa objek tersebut tidak mengalami perubahan besar baik dari segi kompresi atau modifikasi oleh *Squid*. Namun, dalam beberapa kasus di mana kompresi atau perubahan dilakukan oleh Squid, ukuran yang tersimpan di cache mungkin lebih kecil dari ukuran asli objek.

7. URL

Kolom ini mencatat URL lengkap dari permintaan yang di-*cache* oleh *Squid*. URL mencakup protokol (HTTP/HTTPS), nama host, dan path file atau sumber daya yang diminta. URL digunakan untuk melacak sumber permintaan cache dan menganalisis pola lalu lintas data di jaringan.

8. Lastmod (Last Modified)

Waktu terakhir file atau data diperbarui di *server* asal, dicatat dalam format *Unix timestamp*. Informasi ini penting untuk memastikan data di *cache* selalu up-to-date. *Lastmod* digunakan untuk menentukan apakah data di *cache* masih valid atau perlu diperbarui dan untuk menghindari pengambilan data usang dari *cache*.

9. Expires

Waktu kedaluwarsa *cache* dalam format *Unix timestamp*. *Squid* menggunakan nilai ini untuk menentukan kapan data di *cache* harus diperbarui atau dihapus. Contoh:

- Nilai -1: *Cache* tidak memiliki waktu kedaluwarsa dan akan tetap tersimpan sampai dihapus secara manual.
- Nilai seperti 1723000000: Menunjukkan waktu spesifik kapan *cache* dianggap tidak lagi valid.

10. Status

Kode status HTTP yang dihasilkan oleh permintaan, menunjukkan hasil pemrosesan data oleh *Squid*. Contoh:

- 200 OK: Permintaan berhasil.
- 304 Not Modified: Data tidak berubah di server asal, sehingga cache digunakan.
- 403 Forbidden: Permintaan ditolak karena alasan akses.
- 404 Not Found: File tidak ditemukan di server asal.

11. Type

Tipe data atau MIME *type* dari konten yang diminta. MIME *type* menjelaskan format data yang ditransfer. Digunakan untuk menganalisis jenis data yang paling sering diminta oleh klien. Contoh:

- text/plain: File teks biasa.
- application/pdf: File PDF.
- image/jpeg: File gambar dalam format JPEG.

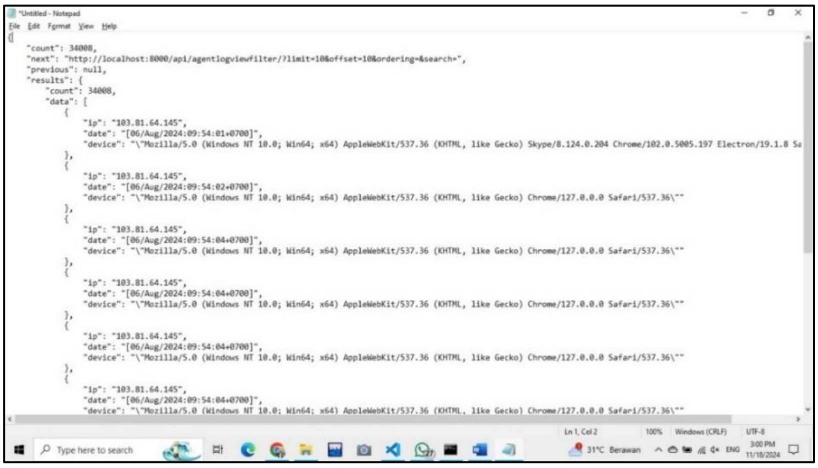
12. Method

Metode HTTP yang digunakan untuk permintaan. *Squid* mencatat metode ini untuk memahami bagaimana data diminta atau dikirimkan. Contoh:

- GET: Meminta data dari server.
- POST: Mengirimkan data ke server (misalnya formulir atau upload).
- CONNECT: Membuka koneksi tunneling untuk permintaan HTTPS. Data JSON *store log* ditunjukkan pada Gambar 3.3.

3.3.1.3 Data JSON *User Agent Log*

Endpoint yang tersedia untuk *user agent log* memungkinkan pengguna untuk mendapatkan data *log* secara terperinci terkait informasi perangkat dan aktivitas pengguna yang terhubung ke sistem. Data ini mencakup informasi seperti alamat klien (*Client Address*), waktu akses (*Date*) dalam format yang terstandardisasi, serta informasi lengkap mengenai perangkat dan aplikasi yang digunakan, yang tercantum dalam atribut device. Atribut device mencakup *string* identifikasi *user agent* yang berisi detail sistem operasi, *browser*, versi perangkat lunak, dan komponen lainnya (contoh: "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.0.0 Safari/537.36"). *Endpoint* ini mendukung fitur *pagination*, pencarian, dan pengurutan untuk membantu pengguna mengelola dan menganalisis data dalam jumlah besar. Informasi dari *user agent log* bermanfaat untuk memahami pola penggunaan perangkat, mendeteksi anomali, serta meningkatkan pengalaman pengguna dan keamanan sistem. Data JSON *user agent log* ditunjukkan pada Gambar 3.4.

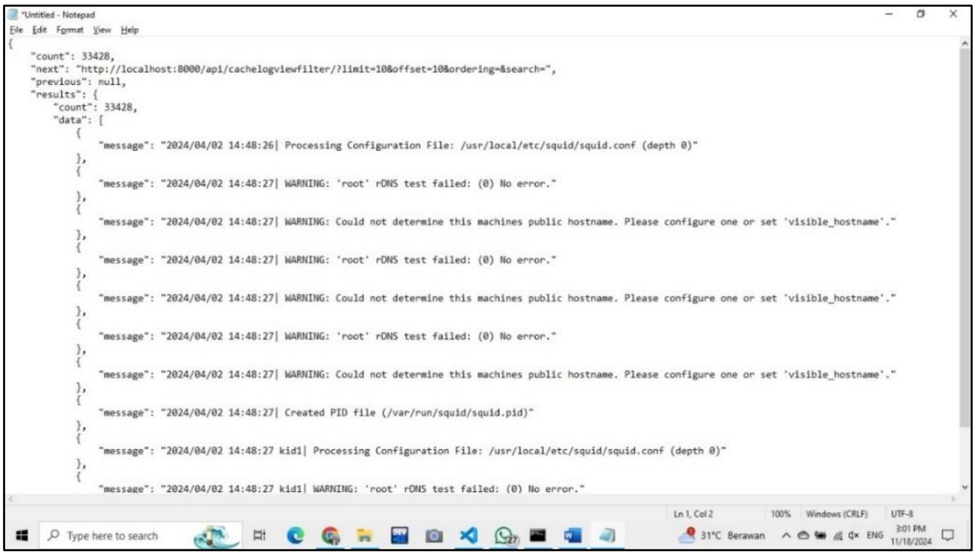



```
{
    "count": 34008,
    "next": "http://localhost:8000/api/agentlogviewfilter/?limit=10&offset=10&ordering=&search=",
    "previous": null,
    "results": {
        "count": 34008,
        "data": [
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:01+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Skype/8.124.0.204 Chrome/102.0.5005.197 Electron/19.1.8 Sa
            },
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:02+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36\""
            },
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:04+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36\""
            },
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:04+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36\""
            },
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:04+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36\""
            },
            {
                "ip": "103.81.64.145",
                "date": "[06/Aug/2024:09:54:04+0700]",
                "device": "\"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36\""
```


Gambar 3.4 Data JSON *User Agent Log*

3.3.1.4 Data JSON *Cache Log*

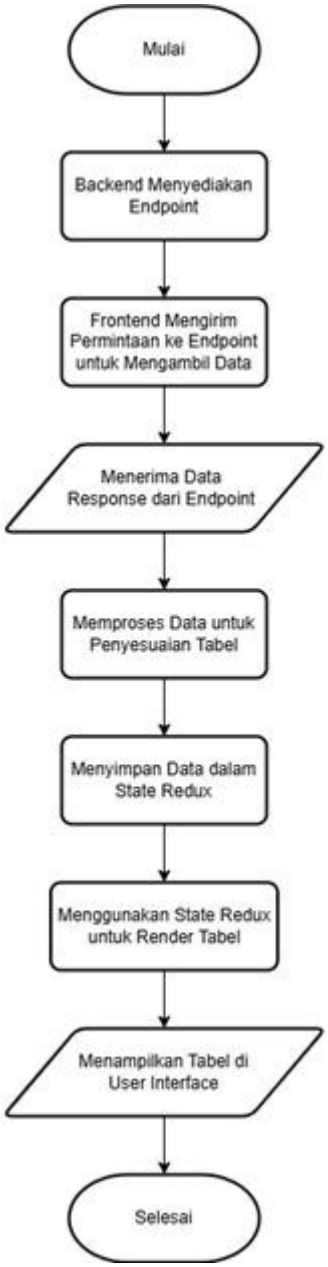
Endpoint yang tersedia untuk *cache log* memungkinkan pengguna untuk mendapatkan data *log* secara terperinci terkait aktivitas dan status operasi *cache* pada *proxy server*. Data *log* ini mencakup atribut *message* yang berisi informasi lengkap mengenai peristiwa yang terjadi, seperti pemrosesan file konfigurasi (*Processing Configuration File*), peringatan terkait pengaturan *hostname* publik yang tidak terdeteksi (*Could not determine this machine's public hostname*), serta log diagnostik lainnya seperti kegagalan tes DNS (*rDNS test failed*) dan pembuatan file PID (*Created PID File*). Log ini ditandai dengan *timestamp* yang mencatat waktu kejadian secara akurat, membantu pengguna melacak aktivitas sistem secara kronologis. Fitur *pagination*, pencarian, dan pengurutan mempermudah pengguna untuk mengelola dan meninjau data dalam jumlah besar. Informasi yang tersedia di cache log bermanfaat untuk memantau kesehatan sistem, mengidentifikasi kesalahan konfigurasi, dan memastikan performa *cache* berjalan optimal sesuai kebutuhan sistem. Data JSON *cache log* ditunjukkan pada Gambar 3.5.



Gambar 3.5 Data JSON *Cache Log*

3.3.2 Alur Komunikasi Data

Alur komunikasi data dalam pembuatan sistem monitoring *cache proxy* milik PT. Queen Network Nusantara dimulai dengan backend menyediakan endpoint untuk mengakses data yang diperlukan. Frontend kemudian mengirim permintaan GET ke endpoint tersebut dan menerima respons data dalam format JSON. Data yang diterima diproses agar sesuai dengan struktur tabel yang akan ditampilkan di antarmuka pengguna. Data yang telah diproses disimpan ke dalam *state Redux* untuk mempermudah pengelolaan data secara *global*. *State Redux* tersebut kemudian digunakan untuk merender tabel secara dinamis di antarmuka pengguna, sehingga tabel selalu diperbarui dengan data terbaru. Akhirnya, tabel ditampilkan kepada pengguna, menandai proses selesai. Gambar 3.6 di bawah ini menggambarkan *flowchart* alur komunikasi data.

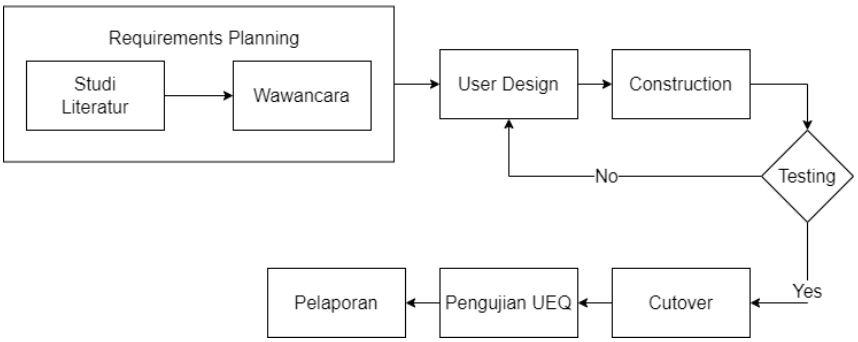


Gambar 3.6 *Flowchart* Alur Komunikasi Data

3.4 Tahapan Penelitian

Tahapan yang dilakukan pada penelitian ini mengikuti model pengembangan perangkat lunak *Rapid Application Development* (RAD). Metode ini memungkinkan perubahan dan penyesuaian yang cepat selama proses pengembangan. Metode ini juga melibatkan kolaborasi yang erat antara pengembang dan pengguna, dimana melibatkan *user* sebagai pengguna sistem. Hal ini untuk memastikan bahwa kebutuhan pengguna terpenuhi sehingga sistem

yang dibangun dapat diadaptasi dengan baik. Gambar 3.7 di bawah ini menggambarkan tahapan penelitian.



Gambar 3.7 Tahapan Penelitian

Berdasarkan Gambar 3.7, penjabaran metode yang digunakan berdasarkan tahapan penelitian adalah sebagai berikut:

3.4.1 *Requirements Planning*

Requirements Planning atau perencanaan kebutuhan adalah tahap yang bertujuan untuk mengidentifikasi tujuan pembuatan sistem dan kebutuhan pengguna. Tahap ini dilakukan untuk memahami permasalahan yang ada serta mengumpulkan data. Dalam penelitian ini, proses *Requirements Planning* dilakukan melalui studi literatur dan wawancara dengan rincian sebagai berikut.

3.4.1.1 Studi Literatur

Pada tahap studi literatur, dilakukan analisis terhadap beberapa penelitian mengenai sistem *monitoring cache* pada *proxy server* di berbagai perusahaan ISP. Tahap ini bertujuan menjadikan sistem perusahaan lain sebagai acuan untuk membangun sistem yang paling efektif dan dapat diterapkan di PT. Queen Network Nusantara.

3.4.1.2 Wawancara

Tahap wawancara dilakukan melalui proses tanya jawab secara lisan untuk mengumpulkan informasi langsung dari pengguna. Tujuannya adalah memahami kebutuhan, permasalahan yang dihadapi terkait monitoring cache pada proxy server, serta mendapatkan saran untuk perancangan sistem.

Tabel 3.3 Jumlah Informan

No	Informan	Jumlah Informan
1	Pegawai PT. Queen Network Nusantara	6

Pertanyaan wawancara ditentukan dengan mengikuti pendekatan yang terstruktur. Pertama, melakukan studi literatur untuk memahami permasalahan yang diteliti dan kebutuhan pengguna. Selanjutnya, mengidentifikasi area utama yang perlu dijelajahi dalam konteks *monitoring cache*. Kemudian, merumuskan pertanyaan-pertanyaan yang relevan dan spesifik, yang dapat memberikan informasi mendalam tentang kebutuhan, harapan, masalah serta saran terkait pembuatan aplikasi. Adapun beberapa pertanyaan yang ditanyakan adalah sebagai berikut.

1. Bagaimana cara Bapak/Ibu memonitor event yang terjadi pada *cache*?
2. Apa saja kendala yang ditemui oleh Bapak/Ibu dalam *monitoring cache*?
3. Apa saja yang menjadi kebutuhan utama Bapak/Ibu dalam *monitoring cache*?
4. Apa fitur dan fungsionalitas utama yang harus ada dalam sistem *monitoring cache*?
5. Apa aspek yang sangat penting untuk diperhatikan dalam proses pengembangan sistem *monitoring cache*?
6. Apakah Bapak/Ibu memiliki saran dan masukan mengenai fitur-fitur yang dapat meningkatkan sistem ini?
7. Bagaimana pandangan Bapak/Ibu terhadap pembuatan sistem ini?

Dari hasil wawancara yang dilakukan, didapatkan kesimpulan bahwa :

1. Hasil wawancara dengan keenam pegawai menunjukkan kesamaan dalam memonitor *event* yang terjadi pada *cache*. Saat ini, *monitoring cache* dilakukan di terminal SSH menggunakan *Command Line Interface* (CLI) dengan antarmuka berbasis teks yang memerlukan perintah baris. Penggunaan CLI ini memungkinkan akses langsung dan kontrol penuh terhadap sistem, tetapi metode ini dapat menjadi sulit dan tidak intuitif bagi pengguna yang tidak terbiasa dengan *command line*. Akibatnya, proses *monitoring* dapat memperlambat respons terhadap masalah, terutama bagi pengguna yang kurang familiar dengan perintah-perintah yang tersedia.

2. Hasil rangkuman dari seluruh wawancara menunjukkan beberapa kendala yang dihadapi oleh pegawai dalam memonitor *cache*, yaitu *user interface* dan pengalaman pengguna yang terbatas serta kurangnya *real-time monitoring*.
3. Berdasarkan rangkuman seluruh wawancara, didapatkan kebutuhan pegawai yaitu suatu sistem yang dapat mendukung *real-time monitoring*, memiliki *user interface* yang interaktif, dan pengalaman pengguna yang lebih baik.
4. Berdasarkan hasil wawancara, fitur dan fungsionalitas utama yang diperlukan pada sistem mencakup *otentikasi*, *dashboard*, *monitoring access log*, *monitoring store log*, *monitoring user agent log*, dan *monitoring cache log*.
5. Berdasarkan rangkuman dari hasil wawancara bersama enam pegawai menunjukkan bahwa aspek-aspek yang perlu diperhatikan dalam proses *monitoring cache* pada *proxy server* yaitu sistem yang intuitif, mudah digunakan, memiliki navigasi yang jelas, dan bahasa yang konsisten.
6. Bedasarkan hasil wawancara dengan keenam pegawai, didapatkan saran untuk menambahkan fitur untuk bisa mengelola *server*.
7. Berdasarkan hasil wawancara, semua pegawai berpandangan bahwa pembuatan aplikasi ini penting dilakukan untuk menjawab kendala dan kebutuhan yang dihadapi oleh para pegawai dalam *monitoring cache*.

3.4.1.3 Kebutuhan Fungsional dan Non Fungsional

Berdasarkan hasil wawancara, diperoleh kebutuhan pengguna yang terdiri dari kebutuhan fungsional dan non fungsional. Adapun rincian kebutuhan fungsional dan non fungsional dalam penelitian ini adalah sebagai berikut.

Tabel 3.4 Kebutuhan Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1	KF- 01	Autentikasi	Pengguna dapat melakukan otentikasi *sistem*.
2	KF- 02	Monitoring Access Log	Pengguna dapat memantau *access log* pada *proxy server* secara *real time*.
3	KF- 03	Monitoring Store Log	Pengguna dapat memantau *store log* pada *proxy server* secara *real time*.
4	KF- 04	Monitoring User Agent Log	Pengguna dapat memantau *user agent log* pada *proxy server* secara *real time*.
5	KF- 05	Monitoring Cache Log	Pengguna dapat memantau *cache log* pada *proxy server* secara *real time*.
6	KF- 06	Dashboard	Pengguna dapat memantau *chart* pada *dashboard*.

Tabel 3.5 Kebutuhan Non Fungsional

No.	Kode Fungsi	Nama Fungsi	Deskripsi
1	NF- 01	*Usability*	Sistem mudah digunakan, intuitif, dan menyediakan navigasi yang jelas bagi pengguna.
2	NF- 02	*Performance*	Sistem mampu menampilkan konten lebih cepat dan efisien.
3	NF- 03	*Compatibility*	Sistem sudah kompatibel dengan browser modern seperti *Chrome*, *Firefox*, dan *Edge* versi terbaru.
4	NF- 04	*Security*	Sistem aman dari ancaman seperti *Cross Site Scripting (XSS)* untuk melindungi data pengguna.

3.4.2 *User Design*

Pada tahap ini, dilakukan penggambaran proses rancangan aplikasi serta perbaikan apabila terdapat ketidaksesuaian desain antara pengguna dan pengembang. Setelah kebutuhan pengguna diidentifikasi, langkah selanjutnya adalah membuat *use case diagram* dan *activity diagram* untuk sistem *monitoring cache* pada *proxy server*. Beberapa *diagram* yang digunakan selama pengembangan adalah sebagai berikut.

3.4.2.1 *Use Case Diagram*

Use case diagram digunakan untuk menggambarkan interaksi antara pengguna yang terlibat dengan sistem, serta menggambarkan berbagai fitur fungsionalitas yang diperlukan oleh pengguna. *Use case diagram* sistem *monitoring cache* pada *proxy server* dibuat berdasarkan kebutuhan fungsional dan non fungsional yang sebelumnya telah ditentukan.

3.4.2.2 *Activity Diagram*

Activity diagram digunakan untuk menggambarkan alur dari suatu aktivitas pada sistem. *use case diagram* yang sebelumnya sudah ditentukan kemudian dibuat proses aktivitasnya secara keseluruhan. Kemudian proses aktivitasnya digambarkan ke dalam *activity diagram* sehingga mudah dimengerti.

3.4.2.3 Perancangan Antarmuka

Perancangan antarmuka adalah desain tampilan sistem yang akan dibangun, antarmuka sistem merupakan sebuah perantara antara pengguna dengan sistem yang ada. Perancangan antarmuka dibuat berdasarkan *activity diagram* yang sebelumnya telah ditentukan.

3.4.3 *Construction*

Pada tahap c*onstruction*, sistem yang sudah dirancang direalisasikan melalui penulisan kode, sehingga menghasilkan sebuah sistem yang dapat digunakan oleh pengguna. Dalam penelitian ini, sistem dikembangkan pada *visual studio code* dengan berbasis *website* dan *ReactJS*.

3.4.3.1 *Testing*

Tahap *testing* dilakukan menggunakan metode *blackbox* untuk memastikan bahwa sistem berjalan sesuai harapan. Jika sistem belum mencapai hasil yang diinginkan, maka akan segera dilakukan perbaikan. Dalam penelitian ini, pengujian dilakukan pada sistem yang sudah dirancang dan diuji secara langsung, termasuk oleh pengguna. Pengujian didasarkan pada *use case* yang telah dibuat, serta mengacu pada kebutuhan fungsional dan non fungsional. Hasil akhir dari tahap testing adalah sistem yang berfungsi sesuai dengan harapan pengguna dan siap digunakan.

3.4.4 *Cutover*

Pada tahap *cutover* atau implementasi, sistem yang telah selesai dibangun siap diintegrasikan dengan *backend*. Selanjutnya, sistem dihosting menggunakan *Vercel* dan dilakukan uji coba sebelum dapat dioperasikan secara penuh. Uji coba ini mencakup pengujian kebutuhan non fungsional dan *usability testing* menggunakan *User Experience Questionnaire* (UEQ) untuk mengukur tingkat pengalaman pengguna terhadap sistem yang telah dikembangkan.

3.4.5 Pengujian *User Experience Questionnaire (UEQ)*

Penggunaan *User Experience Questionnaire* (UEQ) pada penelitian ini bertujuan untuk mengetahui kepuasan pengguna terhadap sistem yang sudah dibuat, responden dalam pengujian ini adalah administrator jaringan PT. Queen Network Nusantara yang sudah familiar dengan penggunaan terminal untuk monitoring *cache proxy*. Hasil akhir dari tahap pengujian *User Experience Questionnaire* (UEQ) adalah sebuah sistem yang dapat meningkatkan pengalaman pengguna.

3.4.6 Pelaporan

Tahap pelaporan menjadi tahap akhir dari penelitian ini, yaitu pelaporan hasil dan temuan dari penelitian Rancang Bangun *Frontend* Sistem *Monitoring Cache* Pada *Proxy Server* Berbasis *Web* Menggunakan *ReactJS*. Semua data yang telah diperoleh dan telah dianalisis akan dilakukan pengambilan kesimpulan dan saran dan digunakan sebagai skripsi pada Universitas Lampung.

V. SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan hasil dari penelitian yang telah disampaikan, didapatkan kesimpulan sebagai berikut:

1. Berhasil dikembangkannya sebuah sistem *monitoring cache proxy* berbasis *web* menggunakan *ReactJS*. Sistem yang dikembangkan menggunakan metode *Rapid Application Development* (RAD) ini berhasil diselesaikan dalam waktu 70 hari dengan 2 iterasi dan dilengkapi dengan berbagai fitur seperti autentikasi, *dashboard*, *monitoring cache*, pengelolaan profil, serta manajemen *server*, yang dirancang untuk memenuhi kebutuhan administrator jaringan di PT. Queen Network Nusantara (QNN).
2. Berdasarkan pengujian *blackbox* dengan 29 skenario, sistem yang dikembangkan telah memenuhi seluruh kebutuhan yang ditetapkan. Semua fungsionalitas, termasuk autentikasi, *dashboard*, *monitoring cache*, pengelolaan profil, dan manajemen server, berfungsi dengan baik dan sesuai dengan spesifikasi. Hal ini menunjukkan bahwa sistem siap digunakan dalam lingkungan operasional untuk mendukung aktivitas monitoring dan manajemen server secara *real-time*.
3. Berdasarkan hasil pengujian kebutuhan non fungsional, uji usability menggunakan *User Experience Questionnaire (UEQ)* menunjukkan bahwa sistem memiliki tingkat kemudahan penggunaan dan pengalaman pengguna yang sangat baik, sesuai dengan ekspektasi pengguna. Uji kompatibilitas dengan *Google Lighthouse* menghasilkan skor tinggi dalam aspek *performance*, *accessibility*, dan *best practices*, yang menunjukkan bahwa sistem dapat berjalan dengan cepat dan stabil pada berbagai *browser*. Uji keamanan menggunakan *Zed Attack Proxy* (ZAP) berhasil mengidentifikasi dan menutup potensi kerentanan, termasuk terhadap serangan *Cross-Site*

Scripting (XSS), sehingga memastikan sistem memiliki tingkat keamanan yang tinggi untuk melindungi data dan aktivitas pengguna.

4. Berdasarkan hasil evaluasi dan implementasi, sistem *monitoring cache* berbasis *web* yang dikembangkan menggunakan *ReactJS* berhasil meningkatkan pengalaman pengguna secara signifikan dibandingkan sistem lama, hal ini ditunjukkan oleh peningkatan skor pada semua dimensi *User Experience Questionnaire* (UEQ). Pada dimensi daya tarik, skor meningkat dari -2,22 (25% terburuk) menjadi 2,26 (10% terbaik), pada dimensi kejelasan dari -2,01 menjadi 2,29 (10% terbaik), serta pada efisiensi dari -2,06 menjadi 2,04. Hal ini menunjukkan bahwa perubahan desain dan teknologi yang diterapkan telah berhasil memberikan antarmuka yang lebih intuitif dan meningkatkan pengalaman pengguna.

5.2 Saran

Adapun saran untuk pengembangan selanjutnya adalah sebagai berikut:

1. Melakukan pengembangan sistem lebih lanjut dengan menambahkan fitur laporan berkala untuk mencatat performa *cache* dari waktu ke waktu melalui laporan yang dapat diekspor dalam format PDF.
2. Menambahkan sistem otorisasi berbasis peran (*role-based access control*) untuk meningkatkan keamanan sistem *monitoring cache* dengan membatasi akses pengguna sesuai peran mereka.
3. Menambahkan fitur pemantauan alamat MAC (*Media Access Control*) milik klien untuk mengidentifikasi perangkat yang terhubung.
4. Menambahkan fitur notifikasi untuk memberi tahu administrator jika ada lonjakan *cache miss* yang tinggi, atau kondisi tidak biasa lainnya.
5. Menambahkan fitur pembatasan beban (*load limiting*) atau pengelolaan antrian data untuk menangani volume *traffic* yang sangat tinggi.

DAFTAR PUSTAKA

[1] Nindy, “Apa Itu Internet Service Provider (ISP) dan Bagaimana Cara Kerjanya?,” PT Telekomunikasi Selular, [Online]. Available: https://www.byu.id. [Diakses 23 Oktober 2024].

[2] A. Rachman, “Rancang Bangun Proxy Server Dan Analisis Pemakaian Internet Dengan Menggunakan SARG (Studi Kasus Di BMKG Juanda Surabaya),” *Jurnal IPTEK,* vol. 17, no. 1, pp. 1-3, 2013.

[3] Hapsari, “Monitoring Dan Evaluasi Kebijakan Pendidikan,” *Indonesian Journal of Teaching and Learning,* vol. 1, no. 1, pp. 77-88, 2022.

[4] Lumenta, “Organisasi Cache Memory,” *Jurnal Teknik Elektro dan Komputer,* vol. 2, no. 5, pp. 3-4, 2013.

[5] Subandri, “Penerapan Teknologi Cache Server Berbasis Iot Dengan Raspberry Pi3 Menggunakan Metode Forward Chainning (Studi Kasus SMK Binakarya Mandiri 2 Kota Bekasi),” *JURNAL KILAT,* vol. 7, no. 2, pp. 169-177, 2018.

[6] F. M. Sarimole, “Implementasi Sistem Monitoring Security Berbasis Web di Komplek Bulak Jakarta Timur,” *Jurnal Pendidikan Tambusai,* vol. 6, no. 1, pp. 3445-3456, 2022.

[7] R. Abdulloh, Web Programming is Easy, Jakarta: PT Elex Media Komputindo, 2015.

[8] A. Wibowo, “Pemanfaatan ReactJS dan Protokol MQTT untuk Visualisasi Sinyal Lampu dan Notifikasi secara Waktu Nyata pada Sistem Pemonitor APILL di Kota Pekanbaru,” *Jurnal Komputer Terapan ,* vol. 7, no. 2, pp. 314 - 328, 2021.

[9] M. S. Raunak, “Implications of Proxy Caching for Provisioning Networks and Servers,” *ACM SIGMETRICS Performance Evaluation Review*, vol. 28, no. 1, pp. 66-77, 2000.

[10] A. Dawson, “Squid: Optimising Web Delivery,” Squid Web Cache, [Online]. Available: https://wiki.squid-cache.org/SquidFaq/SquidLogs. [Diakses 14 November 2024].

[11] G. Arisman, “Queen Network Nusantara,” PT. Queen Network Nusantara, [Online]. Available: https://qnn.co.id/. [Diakses 5 Agustus 2024].

[12] D. Setiawan, “Perancangan Sistem Monitoring Kendaraan Listrik,” *Jurnal Teknik*, vol. 16, no. 1, pp. 96-102, 2022.

[13] A. Susanto, “Rancang Bangun Sistem Informasi Penjualan Aksesoris Hanphone Berbasis Web Pada Dazzle Cellular Semarang,” *Dinamika Informatika Jurnal Ilmiah Teknologi Informasi*, vol. 5, no. 1, pp. 1-6, 2013.

[14] Y. Susilowati, Modul E-Commerce - Teaching Factory For Students, Mutiara Publisher, 2019.

[15] F. Rifandi, “Website Gallery Development Using Tailwind CSS Framework,” *Jurnal E-Komtek*, vol. 6, no. 2, pp. 205-214 , 2022.

[16] R. Sahrial, “PEMANFAATAN JSON UNTUK MENAMPILKAN DATA REALTIME COVID-19 DENGAN MODEL VIEW PRESENTER,” *Jurnal TEKNOINFO*, vol. 16, no. 1, pp. 144-145, 2022.

[17] Valentyn, “Peluncuran kategori Audit SEO di ekstensi Chrome Lighthouse,” Google, [Online]. Available: https://developers.google.com/search/blog/2018/02/seo-audit-category-in-lighthouse?hl=id. [Diakses 3 December 2024].

[18] StatCounter, “Statcounter GlobalStats,” StatCounter, [Online]. Available: https://gs.statcounter.com/browser-market-share. [Diakses 3 December 2024].

[19] A. F. Hasibuan, “Analisis Keretanan Website Dengan Aplikasi Owasp Zap,” *Jurnal Ilmu Komputer dan Sistem Informasi (JIRSI)*, vol. 2, no. 2, pp. 257-270, 2023.

[20] Microsoft, “Visual Studio Code - Code Editing,” [Online]. Available: https://code.visualstudio.com/. [Diakses 5 Agustus 2024].

[21] S. Dharwiyanti, Pengantar Unified Modeling, IlmuKomputer.Com, 2003.

[22] J. Martin, Rapid Application Development, New York: Macmillan Group, 1991.

[23] A. Zein, Konsep Dasar Rekayasa Perangkat Lunak, Cendikia Mulia Mandiri, 2023.

[24] M. Schrepp, User Experience Questionnaire Handbook Version 11, Germany: Amazon Digital Services, 2023.

[25] S. R. Jan, “An Innovative Approach to Investigate Various Software Testing Techniques and Strategies,” *International Journal of Scientific Research and Engineering Trends,* vol. 2, no. 2, p. 682–689, 2016.

[26] T. S. Jaya, “Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus: Kantor Digital Politeknik Negeri Lampung),” *Jurnal Informatika: Jurnal Pengembangan IT (JPIT),* vol. 3, no. 2, pp. 45-48, 2018.

[27] A. Pratama, “Evaluation of User Experience in Integrated Learning Information Systems Using User Experience Questionnaire (UEQ),” *Journal of Information Systems and Informatics,* vol. 4, no. 4, pp. 1020-1028, 2022.

[28] I. Isnaini, “Preliminary Design of Website Application for Environmental and Radiation Monitoring Using React,” *International Nuclear Information System,* vol. 53, no. 2, pp. 449-453, 2020.

[29] Z. Sitorus, “Blocking Prohibited Sites Using Proxy Server in Mikrotik and Squid Proxy Debian Server, at The Nurul Anwar Education Foundation, Tanjungbalai City,” *Journal of Information Technology, computer science and Electrical Engineering (JITCSE),* vol. 1, no. 2, pp. 10-19, 2024.

[30] H. J. S. Sitio, “Web-Based Application Development for Training Data Management Using REACTJS,” *Indonesian Journal of Multidisciplinary Science,* vol. 2, no. 6, pp. 2573-2587, 2023.

[31] Y. A. Singgalen, “Coastal and Marine Tourism Monitoring System Design

using Rapid Application Development (RAD)," *Journal of Information System Research (JOSH),* vol. 5, no. 2, pp. 458-476, 2023.

[32] D. Nugeroho, "Rancang Bangun Sistem Informasi Monitoring Rencana Strategis Bisnis Bank X Menggunakan Metode RAD," *Jurnal Ilmiah Informatika Komputer,* vol. 24, no. 1, pp. 14-22, 2019.

[33] R. Astuti, "Model Rapid Application Development (RAD) Untuk Rancang Bangun Sistem Informasi Monitoring Project Pada Branch Business Process Re-Engineering (BBPR) Team," *INNOVATIVE: Journal Of Social Science Research,* vol. 3, no. 6, pp. 6683-6696, 2023.

[34] M. I. Santoso, "Rancang Bangun Sistem Informasi Pembayaran SPP Pada Sekolah Sepak Bola Tasbi Dengan Menggunakan React Dan NodeJS," *Jurnal Teknologi dan Sains,* vol. 1, no. 1, pp. 10-14, 2023.