I. PENDAHULUAN

1.1 Latar Belakang

Pertumbuhan pengguna internet di Indonesia saat ini mengalami peningkatan yang cukup pesat. Berdasarkan hasil survei Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) yang dilakukan pada tahun 2020 jumlah pengguna internet di Indonesia tahun 2020 mengalami peningkatan sebesar 8,9% atau setara 25,5 juta pengguna dari tahun 2018 dengan jumlah pengguna internet sekitar 64,8% menjadi 73,7% atau setara dengan 196,7 juta pengguna (Asosiasi Penyedia Jasa Internet Indonesia, 2020).

Internet *Service Provider* (ISP) merupakan perusahaan atau badan usaha yang menyediakan produk dan jasa yang berhubungan dengan internet. PT Lintas Data Multimedia sebagai salah satu ISP di Lampung yang menyediakan layanan teknologi berbasis internet untuk perusahaan yang bermitra. Dalam pengerjaan proyek pemasangan jaringan di PT Lintas Data Multimedia pembagian pekerjaan dibagi menjadi dua yaitu supervisor dan teknisi. Tugas supervisor yaitu mengelola alur kerja teknis dan melakukan pengecekan pemasangan jaringan yang dilakukan oleh teknisi. Sedangkan tugas teknisi melakukan instalasi pemasangan perangkat jaringan.

Mekanisme pengelolaan pemasangan jaringan yang saat ini sedang berjalan sudah cukup baik, namun memiliki beberapa kelemahan seperti pembagian jadwal pemasangan jaringan dilakukan di pagi hari sebelum melakukan pemasangan sehingga dapat membutuhkan waktu yang lebih lama untuk melakukan pemasangan. Setelah teknisi selesai melakukan pemasangan, teknisi mengirimkan bukti foto pemasangan dan foto pengujian jaringan melalui via *WhatsApp* dimana data tersebut belum tersimpan di database sehingga mempersulit dan memperlambat dalam proses pengelolaan data pemasangan. Supervisor juga harus datang ke lokasi pemasangan untuk melakukan pengecekan pemasangan, proses ini membutuhkan biaya dan memakan waktu. Formulir pemasangan masih dalam bentuk *print out* sehingga rentan kehilangan serta kerusakan dan membutuhkan

waktu yang lama dalam pencarian riwayat formulir pemasangan. Seluruh kelemahan pengelolaan data pemasangan jaringan dapat diperbaiki dengan membuat sebuah aplikasi pemasangan jaringan yang menggunakan arsitektur *microservices.*

Microservices merupakan suatu arsitektur dalam pengembangan aplikasi dimana sistem akan dipecah menjadi beberapa *service* kecil yang dapat berinteraksi antara *service* satu sama lain (Khoirunnisa', 2019). Sistem ini akan mempermudah teknisi dan supervisor dalam proses pengelolaan data pemasangan jaringan, sehingga teknisi dapat menerima jadwal pemasangan melalui aplikasi, proses pencarian data lebih mudah dan aman karena penyimpanan data sudah terkomputerisasi.

1.2 Tujuan

Tujuan dari pembuatan Tugas Akhir yang berjudul "Aplikasi *Backend Monitoring* Pemasangan Jaringan Pada PT Lintas Data Multimedia Menggunakan Arsitektur *Microservice*", yaitu menghasilkan aplikasi yang menerapkan teknologi *microservice* yang dapat digunakan oleh teknisi dan supervisor dalam melakukan *monitoring* pemasangan jaringan.

1.3 Kerangka Pemikiran

Dalam proses pengolahan data pemasangan jaringan pada PT. Lintas Data Multimedia saat ini masih kurang efisien, karena pengelolaan berkas formulir pemasangan masih menggunakan kertas *(hardcopy)* dan supervisor harus datang ke lokasi pemasangan untuk melakukan pengecekan, karena hal tersebut membutuhkan waktu yang lama serta membutuhkan biaya yang lebih besar.

Berdasarkan pemaparan tersebut, maka diperlukan solusi untuk mempermudah dalam melakukan pemasangan jaringan, yaitu dengan membuat "Aplikasi *Backend Monitoring* Pemasangan Jaringan Pada PT Lintas Data Multimedia Menggunakan Arsitektur *Microservice*". Tahapan kerangka pemikiran disajikan pada Gambar 1.

Permasalahan
Dalam menganalisis permasalahan menggunakan metode pieces *Performance* (Kinerja) : Proses pengecekan dan pencarian berkas formulir pemasangan yang memerlukan waktu yang lama karena masih berbentuk *print out.* *Information* (Informasi) : Jadwal pemasangan jaringan diberikan dalam bentuk *print out* serta tidak ada informasi jumlah alat yang tersedia dan alat yang telah terpasang. *Economic* (Ekonomi) : Pembuatan formulir pemasangan dalam bentuk *print out* sehingga membutuhkan biaya pencetakan dan membutuhkan biaya untuk supervisor melakukan pengecekan ke lokasi pemasangan. *Control* (Pengendalian) : Pengelolaan formulir pemasangan yang rentan mengalami kehilangan dan kerusakan. *Efficiency* (Efisiensi) : Membutuhkan waktu yang lama karena supervisor harus datang ke lokasi untuk melakukan proses pengecekan pemasangan. *Service* (Layanan) : Teknisi mengantri dalam menunggu jadwal pemasangan.
Solusi
Mengimplementasikan *microservice* pada *backend* aplikasi *monitoring* pemasangan jaringan pada PT lintas data multimedia.
Pengembangan sistem
Metode Pengembangan : *Rappid Application Development* (RAD) Bahasa Pemrograman : *Javascript* Framework : *NodeJs*
Hasil
Aplikasi *Backend Monitoring* Pemasangan Jaringan Pada PT Lintas Data Multimedia Menggunakan Arsitektur *Microservice*

Gambar 1. Kerangka Pemikiran

1.4 Kontribusi

Aplikasi *Backend Monitoring* Pemasangan Jaringan Pada PT Lintas Data Multimedia Menggunakan Arsitektur *Microservice* ini diharapkan dapat memberikan kontribusi kepada beberapa pihak antara lain :

1. Bagi Teknisi memberikan kemudahan dalam memperoleh jadwal pemasangan dan mempermudah dalam menginformasikan setelah selesai melakukan pemasangan.
2. Bagi Supervisor mempermudah dalam memberikan jadwal pemasangan dan mempermudah dalam mengelola data pemasangan jaringan serta mempermudah dalam melakukan pengecekan pekerjaan pemasangan.

II. TINJAUAN PUSTAKA

2.1 Aplikasi *Website* dan *Backend*

Aplikasi adalah sebuah program komputer yang dibuat untuk menolong dan mempermudah manusia dalam melaksanakan tugas tertentu (Nurcahyono, 2012). Sedangkan menurut Abdurahman & Riswaya (2014) aplikasi merupakan program siap pakai yang dapat digunakan untuk menjalankan perintah dari pengguna aplikasi tersebut dengan tujuan pembuatan aplikasi tersebut. Aplikasi menyiratkan pemecahan masalah menggunakan salah satu teknik pemrosesan data aplikasi, biasanya berdasarkan perhitungan yang diinginkan atau diharapkan dan pemrosesan data yang diharapkan.

Website adalah kumpulan halaman web yang dipublikasikan di Internet dengan domain yang dapat diakses oleh semua pengguna Internet dengan memasukkan alamat (Arief, 2011). Halaman *website* biasanya berupa dokumen yang ditulis dalam format HTML yang bisa diakses melalui HTTP atau HTTPS. Fungsi *website* secara umum yaitu sebagai alat komunikasi, media informasi, media hiburan dan sarana transaksi.

Backend adalah bagian belakang layar dari sebuah *website*. Setip *backend* situs web terdiri dari tiga bagian yaitu *server*, *database* dan aplikasi. Pengembang *backend* menulis kode yang memungkinkan ketiga komponen ini berinteraksi dan bekerja sama untuk menjalankan fungsinya dan menyampaikan informasi kepada pengguna akhir.

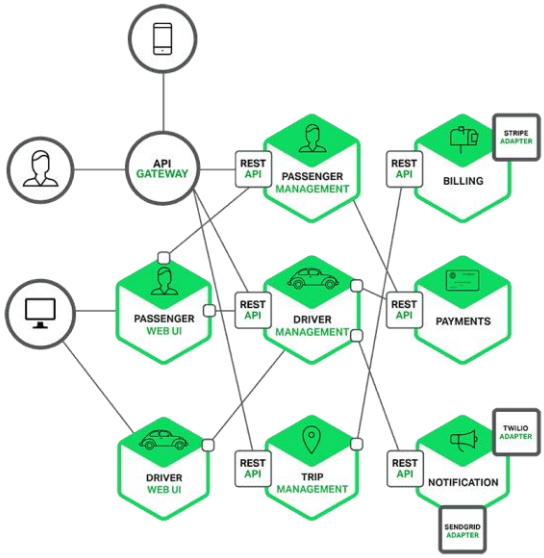
2.1.1 Monitoring

Monitoring adalah evaluasi yang berkesinambungan terhadap fungsi-fungsi kegiatan yang direncakan sesuai dengan rencana dan jadwal yang telah ditetapkan oleh kelompok (Rizan & Hamidah, 2016). Dengan dilakukan *monitoring* dapat diketahui program atau proyek yang dilaksanakan sesuai atau tidak sesuai dengan rencana. Bila ditemukan penyimpangan atau kelambanan maka segera dibenahi sehingga kegiatan dapat berjalan sesuai dengan rencana dan target.

Pada dasarnya, *monitoring* memiliki dua fungsi dasar yang berhubungan yaitu untuk memastikan proses sesuai dengan rencana dan untuk mengetahui perkembangan organisasi dalam pencapaian target yang diharapkan. Tujuan dilakukan *monitoring* yaitu :

1. Menjaga agar kebijakan yang sedang diimplementasikan sesuai dengan tujuan dan sasaran.
2. Menemukan kesalahan secepat mungkin sehingga mengurangi risiko yang lebih besar.
3. Melakukan tindakan modifikasi terhadap kebijakan apabila hasil monitoring mengharuskan untuk itu.

2.1.2 Microservices



Gambar 2. Microservices

Microservices merupakan suatu arsitektur dalam pengembangan aplikasi dimana sistem akan dipecah menjadi beberapa *service* kecil yang dapat berinteraksi antara *service* satu sama lain. *Microservices* muncul untuk mengatasi keterbatasan pengembangan tradisional dengan menggunakan arsitektur monolitik dimana membangun seluruh perangkat lunak dengan basis kode dan basis data yang sama sehingga dapat mempengaruhi kinerja sistem yang terus berkembang (Junilla, 2021). *Microservices* memecahkan masalah berorientasi layanan yang diselesaikan tanpa memengaruhi layanan lain. Artinya *programmer* tidak perlu merubah keseluruhan aplikasi hanya perlu menambahkan *service* baru kedalam aplikasi tersebut (Khoirunnisa', 2019).

Konsep *microservices* berbeda dengan pendahulunya, yaitu *system Oriented Architecture* (SOA). *Microservices* menggunakan protokol *REST* untuk komunikasi pesan yang lebih ringkas dibandingkan dengan SOA yang menggunakan XML SOAP, yang mengedepankan pembagian berdasarkan domain fungsionalitas pada organisasi (Thones, 2015).

Menurut Newman (2015) beberapa keuntungan dalam menggunakan arsitektur *microservices*, yaitu sebagai berikut:

1. *Technology heterogeneity,* pengembang dapat memlih teknologi yang sesuai dengan kebutuhan layanan. Selain itu, pengembang dapat mengadopsi teknologi baru dengan cepat dengan resiko yang minimal.
2. *Resilience,* dengan *microservices,* jika salah satu *service* mengalami kegagalan tidak mempengaruhi keseluruhan aplikasi.
3. *Scaling,* pengembang dapat melakukan pengukuran layanan yang memang perlu melakukan pengukuran tanpa mengukur seluruh layanan seperti arsitektur monolitik.
4. *Ease of deployment, microservices* memungkinkan untuk mengubah dan menjalankan kembali satu baris kode dalam suatu layanan tanpa melibatkan layanan lain yang tidak memerlukan perubahan, sehingga perubahan dapat dilakukan lebih cepat.
5. *Organizational alignment,* dapat meningkatkan kinerja tim menjadi lebih produktif karena dengan *microservices* memudahkan dalam pembagian jumlah orang yang bekerja untuk tiap layanan.
6. *Composability,* komposisi layanan dapat dengan mudah diimplementasikan karena hanya antarmuka antar layanan yang perlu dipertimbangkan saat memilih dan mengkompilasikan setiap layanan.
7. *Optimizing for replaceability,* karena layanan *microservices* kecil, biaya untuk mengganti layanan dengan implementasi yang lebih baik atau menghapusnya jauh lebih mudah dikelola.

Menurut Ghani, Wan-Kadir, Mustafa, & Babir (2019) dalam *microservice* ada beberapa jenis pengujian yang dapat dilakukan yaitu :

1. *Unit test*

   Untuk mengisolasi komponen terkecil yang dapat digunakan dari suatu aplikasi dan mengujinya satu per satu dilakukan untuk memastikan semuanya bekerja dengan sendirinya sebelum berintegrasi dengan aplikasi secara keseluruhan.

2. *Component test*

   Pengujian komponen untuk layanan mikro hanya dapat menjadi pengujian penerimaan untuk layanan dan pengujian perlu memvalidasi apakah layanan menyediakan fungsionalitas yang sesuai.

3. *Integration test*

   Tes integrasi, dari perspektif pengujian layanan mikro, menangani integrasi pengujian dan cacat antarmuka untuk komponen dalam layanan. Pengujian integrasi layanan mikro akan memastikan bahwa setiap komponen berperilaku sebagaimana mestinya ketika dihubungi.

4. *Contract test*

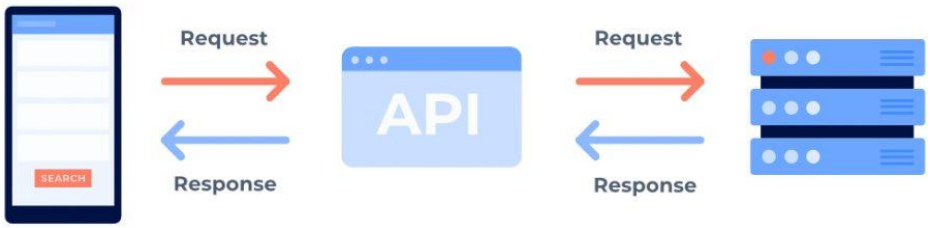
   Pengujian kontrak API layanan mikro dilakukan untuk melihat apakah API tersebut valid atau apakah reputasi layanan mikro adalah API.

5. *End-to-end test*

   untuk menguji fungsional keutuhan secara keseluruhan sistem.

2.1.3 API (Application Programming Interface)

API atau *Application Programming Interface* merupakan suatu dokumentasi yang mencakup antarmuka, fungsi, kelas, struktur, dan sebagainya untuk membangun perangkat lunak. Dengan API ini, *programmer* dapat dengan mudah membongkar perangkat lunak untuk pengembangan lebih lanjut atau integrasi dengan perangkat lunak lain. Kelebihan dari API ini adalah memungkinkan suatu aplikasi untuk terhubung dan berinteraksi dengan aplikasi lain (Amri, 2011). Cara kerja API dapat dilihat pada Gambar 3 :



Gambar 3. Cara kerja API

1. Tahap pertama Aplikasi mengakses API
2. API melakukan *request* ke *server*
3. *Server* memberi respon ke API
4. API menyampaikan respon ke aplikasi (Lawrence, 2020)

Menurut Rohman S. (2020) keuntungan menggunakan API yaitu :

1. Portabilitas

Pengembang yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana pun yang sudah terinstal API.

2. Lebih mudah dimengerti

API menggunakan bahasa yang lebih terstruktur dan lebih mudah dipahami daripada bahasa *system call. System call interface* berfungsi untuk sebagai penghubung antara API dan *system call* yang dimengerti oleh sistem operasi. *System call interface* ini akan menerjemahkan perintah ke dalam API dan kemudian memanggil *system call* yang diperlukan (Rohman S. , 2020).

2.1.4 JavaScript Object Notation (JSON)

JSON adalah format pertukaran data yang ringan, mudah dimengerti dan diimplementasikan oleh manusia serta memudahkan komputer dalam melakukan parsingnya (Hidayatullah, 2019). JSON merupakan format yang tidak tergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum yang digunakan *programmer* keluarga C termasuk C, C++, C#, Java, JavaScript, Perl dan Python. Kelebihan ini lah yang membuat JSON menjadi bahasa pertukaran data yang ideal. JSON terbuat dari dua struktur, yaitu :

1. Beberapa pasangan dari nama atau nilai *(object).*
2. Nilai-nilai yang tersusun secara terurut *(array).*

2.1.5 REST (REpresentation State Transfer)

REST adalah *web service* yang menerapkan konsep transisi *state*, dimana REST melalui tautan HTTP untuk melakukan operasi tertentu. *Response* dari *web service* REST dapat berupa XML atau JSON (Kurniawan, Oslan, & Kristanto, 2015). REST menggunakan *header* status HTTP untuk menunjukkan jenis operasi yang akan dilakukan. Dimulai dengan kode respon 200, 404 dan sebagainya. Kemudian memanfaatkan *method state* pada *request header* seperti GET, POST, PUT dan DELETE (Kusuma, 2021).

REST merupakan jenis web *service* yang memiliki arsitektur standar untuk pertukaran data antar aplikasi atau sistem berbasis web, yang menggunakan HTTP *(Hypertext Transfer Protocol)* sebagai protokol komunikasi dan memliki *resource* pada *server* dan *client* yang dapat diakses lewat *Uniform Resource Identifier* (URI) (Sugiharto & Painem, 2018). Kelebihan menggunakan REST menurut Samuel, Lina, & Arisandi (2020) yaitu :

1. Dapat digunakan dalam banyak bahasa pemrograman dan platform
2. Lebih simpel dibandingkan dengan SOAP
3. Mudah dipelajari

2.1.6 Javascript

Javascript pertama kali diciptakan oleh seorang karyawan Netscape yang bernama Brendan Eich pada tahun 1995. Pada awalnya bahasa pemrograman javascript bernama mocha yang digunakan untuk bahasa sederhana untuk browser Netscape Navigator 2. Nama mocha mengalami perubahan menjadi livescript kemudian diaplikasikan pada browser Netscape Navigator. Netscape kemudian bekerja sama dengan Sun (pengembang bahasa pemrograman Java) dan mengubah nama livescript menjadi javascript pada tanggal 4 desember 1995.

Javascript merupakan bahasa skrip yang disisipkan dalam kode HTML dan proses pada sisi klien, sehingga memperluas kemampuan dokumen HTML (Ripai, 2017). Menurut Rohman S. (2020) kelebihan menggunakan Javascript yaitu :

1. Lebih mudah dipelajari dibandingkan bahasa pemrograman lain

2. Web browser dapat menginterpretasikan javascript dengan HTML sehingga tidak membutuhkan *compiler*
3. Javascript dapat digunakan diberbagai browser dan platform
4. *Error* lebih mudah dicari dan diperbaiki
5. Javascript dapat digunakan untuk memvalidasi input dan mengurangi keinginan untuk mengecek data secara manual
6. Website menjadi lebih cepat, ringan dan interaktif dibandingkan bahasa pemrograman lainnya

2.2 Object-Oriented Programming (OOP)

Object-Oriented Programming (OOP) merupakan paradigma pemrograman yang menggunakan objek sebagai unit dasar dari suatu program dan merangkum program dan data di dalamnya untuk meningkatkan penggunaan kembali perangkat lunak. Objek berisi beberapa perintah standar sederhana yang dirakit menjadi modul dan disusun kedalam sebuah proyek (Lee, Chang, & Wang, 2020). Dalam *object oriented programming*, dikenal empat prinsip yang menjadi dasar penggunaannya. Keempat prinsip OOP tersebut adalah sebagai berikut :

1. *Encapsulation*

   Enkapsulasi adalah konsep mengikat data, atau menggabungkan metode yang berbeda untuk merenkapsulasi menjadi satu unit data. Proses enkapsulasi mempermudah untuk menggunakan sebuah objek dari suatu kelas karena kita tidak perlu mengetahui segala hal secara rinci.
2. *Abstraction*

   Prinsip ini memungkinkan seorang pengembang memerintahkan suatu fungsi, tanpa harus mengetahui bagaimana fungsi tersebut bekerja.
3. *Inheritance*

   Inheritance dalam konsep OOP adalah kemampuan untuk menurunkan fungsi atau membuat kelas baru yang mirip dengan fungsi yang ada.
4. *Polymorphism*

   Polymorphism merupakan konsep dimana suatu objek yang berbeda-beda dapat di akses melalui *interface* yang sama (Perdana, 2021).

2.2.1 Framework

Framework merupakan kumpulan instruksi-instruksi yang dikumpulkan dalam *class* dan *function-function* dengan fungsi yang sesuai untuk memudahkan pengembang memanggilnya tanpa harus menulis ulang *syntax* program yang sama berulang kali dan dapat menghemat waktu (Sallaby & Kanedi, 2020). Secara umum *framework* adalah kerangka yang menggunakan struktur MVC (*Model, View, Control*) yang merupakan media dalam pengelolaan suatu data sehingga menghasilkan sesuatu yang diinginkan (Firmansyah & Herman, 2021). Fungsi *framework* yaitu :

1. Membuat koding lebih mudah dan efisien
2. Meningkatkan keamanan
3. Mempercepat pembuatan *website*
4. Pemeliharaan dan perawatan *website* lebih mudah

2.2.2 Node.Js

Node.Js pertama kali dibuat oleh Ryan Dahl pada tahun 2009 dan hanya dapat dijalankan di dalam sistem operasi linux. Node.Js awalnya ditujukan untuk mengembangkan program-program yang berkaitan dengan permasalahan-permasalahan jaringan computer. Namun seiring dengan perkembangan sekarang Node.Js lebih banyak digunakan dalam proses pengembangan aplikasi web.

Node.Js adalah lingkungan atau *platform* untuk mengeksekusi kode-kode yang ditulis dalam Javascript, yang dikenal dengan sebutan Javascript *runtime environment.* Dalam melaksanakan tugasnya, Node.Js menggunakan V8, yaitu mesin Javascript yang diproduksi oleh Google. V8 bertugas untuk mengubah kode javascript ke dalam bentuk *bytecode* (Raharjo, 2019). Menurut Rohman S. (2020) ada beberapa keuntungan menggunakan Node.Js yaitu :

1. Pengembang hanya menggunakan satu bahasa untuk mengembangkan aplikasi klien dan server yang lengkap, sehingga mengurangi waktu yang diperlukan untuk mempelajari bahasa server lainnya.
2. *Code reuse*
3. Javascript secara *native* mendukung JSON, standar transmisi data yang banyak digunakan saat ini, sehingga penggunaan data dari pemrosesan pihak ketiga di Node.Js akan sangat mudah

4. Tidak perlu ada kekhawatiran bahwa browser tidak akan mendukung fitur-fitur di Node.js karena Node.js memakai V8 yang selalu mengikuti perkembangan standar ECMAScript

2.2.3 Express.js

Expess.js merupakan web *framework* untuk Node.js. Framework ini dikembangkan oleh Tj Holowaychuk yang bertujuan untuk membuat framework minimalis dengan fitur yang dapat ditambahkan melalui plugins (Rohman & Yatini, 2016).

Express.js digunakan untuk mengkompilasi perutean, yang merupakan mekanisme untuk mengakses permintaan HTTP dari klien ke *server* untuk merutekan *server* ke kode khusus yang menangani permintaan tersebut. Setiap rute yang dikompilasi memberitahukan server apa yang diminta klien, kemudian server merespon dengan tepat dan mengirim kembali ke klien (Widodo, 2020).

2.3 Perancangan dan Desain Sistem

Perancangan dan desain sistem merupakan kegiatan merancang, menggambarkan, merencanakan dan menentukan cara mengolah sistem dari hasil analisa sistem sehingga dapat memenuhi kebutuhan dari pengguna. Desain dalam pembangunan sistem merupakan upaya untuk mengonstruksi sebuah sistem yang memberikan spesifikasi kebutuhan fungsional, memenuhi target & kebutuhan secara signifikan dari segi performa maupun penggunaan sumber daya. Desain sistem berfokus pada proses bagaimana sistem dibangun untuk memenuhi kebutuhan pada fase analisis yang sudah dilakukan sehingga target pencapain organsiasi sesuai harapan.

2.3.1 Mapping Chart

Mapping chart merupakan diagram yang menyajikan aliran dokumen dalam suatu sistem informasi. *Mapping chart* berfungsi untuk mempermudah melakukan anallisa terhadap sistem dan sebagai sarana komunikasi antar pengguna dan analisis. Adapun simbol *mapping chart* dapat dilihat pada tabel 1.

Tabel 1. Simbol *mapping chart*

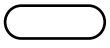
Simbol	Keterangan
	Terminal yang menunjukan sumber atau tujuan dokumen dan laporan
	Dokumen sumber atau laporan
	Operasi manual
	File untuk menyimpan dokumen sumber dan laporan
	Catatan akuntansi (jurnal, register, log, buku besar)
	Konektor intrahalaman
	Garis alir dokumen
	Tempat menyimpan data
	Input yang terkomputerisasi

Sumber: (Hall, 2007)

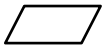
2.3.2 Flowchart

Menurut (Harwikarya dkk., 2017) *flowchart* adalah salah satu cara mempresentasikan langkah logis pemecahan masalah. Pengertian lain mengenai *flowchart* yaitu gambaran dari struktur logika berupa simbol-simbol yang digunakan untuk menyelesaikan masalah secara berurutan (Sitorus, 2015). Simbol-simbol dari *flowchart* dapat dilihat pada tabel 2.

Tabel 2. Simbol *Flowchart*

Simbol	**Nama**	**Fungsi**
(1)	**(2)**	**(3)**
	*Terminal*	Menyatakan permulaan atau akhir program.
	*Flow*	Menyatakan jalannya arus.

Tabel 3. (Lanjutan Simbol *Flowchart)*

Simbol	Nama	Fungsi
(1)	(2)	(3)
	*Decision*	Menyatakan kondisi yang menghasilkan 2 kemungkinan (iya atau tidak).
	*Connector*	Menyatakan sambungan dari proses ke proses dalam halaman yang sama
	*Punched Card*	Menyatakan sambungan dari proses ke proses dalam halaman yang sama
	*Punch Tape*	Input berasal dari kartu atau output ke kartu
	*Predefined Process*	Menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk harga awal
	*Process*	Menyatakan suatu proses dari komputer
	Dokumen	Dokumen sumber atau laporan
	*Input/Output*	Menyatakan input atau output tergantung jenis peralatannya
	*Offline Connector*	Menyatakan sambungan dari proses ke proses dalam halaman yang sama

Sumber: Sitorus (2015)

2.4 *Rapid Application Development* (RAD)

Rapid Application Development (RAD) merupakan metode alternatif yang dapat digunakan untuk mengembangkan sistem atau aplikasi. RAD dapat membuat sistem dengan waktu yang lebih cepat dan akurat untuk mencapai tujuan. Keuntungan dari metode ini adalah waktu pengembangan aplikasi hanya membutuhkan waktu 30-90 hari dibandingkan dengan metode pengembangan umum yang membutuhkan waktu setidaknya 180 hari (Tilley & Rosenblatt, 2017). Keuntungan lainnya menggunakan RAD yaitu :

1. Peningkatan fleksibilitas dan adaptasi, sebagai pengembang anda dapat melakukan pengaturan secara cepat selama proses pengembangan.
2. Iterasi yang cepat dapat mengurangi waktu pengembangan dan waktu pengiriman.
3. Dorongan untuk me-*reuse* kode,
4. yang mana akan lebih sedikit melakukan *coding* secara manual, kemungkinan *error* lebih kecil, dan waktu pengujian yang lebih singkat.
5. Meningkatkan kepuasan pelanggan dikarenakan level kolaborasi yang tinggi dan kordinasi antara pengembang, klien dan *end-user*.
6. *Risk management* yang lebih baik sebagai *stakeholder* dapat didiskusikan dan mengalamatkan kerentanan kode sementara menjaga proses pengembangan tetap berjalan.

Gambar 4. RAD

Sumber: (Kendall & Kendall, 2010).

RAD memliki 4 tahapan yaitu :

1. *Requirement Planning* (Merencanakan Kebutuhan)

   Pada tahap ini merupakan tahapan untuk mengetahui kebutuhan sistem diantaranya dengan mengidentifikasi kebutuhan informasi dan masalah yang dihadapi untuk menentukan tujuan, batasan sistem, kendala dan alternatif pemecahan masalah.

2. *Design Workshop* (Membuat Desain)

Tahap ini dilakukan identifikasi solusi alternatif dan memilih solusi yang terbaik. Setelah itu membuat desain proses bisnis dan desain pemrograman untuk data-data yang telah didapatkan dan domodelkan dalam arsitektur sistem informasi.

3. Implementation

Pada tahap ini dilakukan proses pengkodean aplikasi berdasarkan desain proses bisnis dan desain program yang telah dibuat ke dalam bahasa pemrograman.

4. *Cutover*

Cutover merupakan tahap terakhir yaitu melakukan proses pengujian secara keseluruhan dengan menggunakan metode pengujian. Untuk mengetahui apakah aplikasi yang dibangun sesuai dengan sistem yang diperlukan.

2.5 Pengujian

Pengujian dapat menjadi alat untuk mengukur kualitas perangkat lunak yang dibutuhkan sebelum perangkat lunak dirilis atau diluncurkan. Hal tersebut dilakukan untuk menyakinkan pengguna bahwa perangkat lunak dapat digunakan dan memenuhi kinerja yang sesuai dengan kebutuhan pengguna. Pada tugas akhir ini penulis menggunakan dua jenis pengujian yaitu *system usability testing.*

2.5.1 *System Usability Scale* (SUS)

System usability scale adalah metode pengujian yang melibatkan pengguna akhir dalam melakukan pengujiannya (Ependi, Kurniawan, & Panjaitan, 2019). *System usability testing* dikembangkan oleh Brooke pada tahun 1996 sebagai *quick and dirty* survei yang memungkinkan pengembang untuk memudahkan dalam menilai produk atau layanan tertentu. Kelebihan dalam menggunakan system usability testing menurut Bangor, Kortum, & Miller, (2008) yaitu :

1. Survei besrsifat agnostik teknologi sehingga cukup fleksibel untuk menilai secara luas berbagai teknologi antarmuka
2. Survei ini relatif cepat dan mudah digunakan oleh kedua studi peserta dan pengurus

3. survei memberikan skor tunggal pada skala yang mudah dipahami oleh berbagai orang
4. survei ini bersifat bukan hak milik, menjadikannya alat yang hemat biaya

2.6 Penelitian Terkait

Pembuatan proposal tugas akhir ini penulis menggunakan beberapa jurnal sebagai referensi dan data pendukung. Beberapa referensi jurnal yang penulis gunakan yaitu :

1. Mulyono, dkk., (2019) yang berjudul "Desain Dan Implementasi *Microservices* Studi Kasus Pada Layanan *Taking Order* (Aplikasi E-Commerce PT XYZ)" yang bertujuan untuk mengembangkan aplikasi *taking order* di PT XYZ menggunakan layanan *microservices* arsitektur untuk mengatasi masalah dalam aplikasi sebelumnya. Metode pengembangan sistem yang digunakan yaitu metode *Rapid Application Development* (RAD).
2. Utami, dkk., (2021), dalam jurnalnya yang berjudul "Implementasi Arsitektur *Microservice* Pada Sistem Informasi Data Alumni Jurusan Ekonomi Dan Bisnis Politeknik Negeri Lampung". Pada pengembangan sistem informasi data alumni saat ini masih menggunakan arsitektur *monolithic* dimana arsitektur ini masih memiliki banyak kekurangan yang dapat mempengaruhi aplikasi secara menyeluruh. Hasil dari penelitian ini yaitu menerapkan *web service* untuk mengelola data alumni pada aplikasi my ekbis sehingga mempermudah para user yang membutuhkan data alumni agar dapat ditransformasikan dan tidak harus mengakses *database* my ekbis secara langsung tetapi hanya melalui web service saja. Metode pengembangan sistem yang digunakan yaitu metode SDLC.
3. Saputra, dkk., (2019), dalam jurnalnya yang berjudul "Aplikasi *Monitoring* Berdasarkan Laporan Kegiatan Organisasi Perangkat Daerah (OPD) Kabupaten Pringsewu Studi Kasus pada Badan Perencanaan dan Pembangunan Daerah (Bappeda)". Aplikasi ini dibuat karena Bappeda mengalami kesulitan pada saat *monitoring* karena OPD hanya melakukan

pelaporan kegiatan di akhir tahun saja, sedangkan *monitoring* dilakukan guna melihat perkembangan status kegiatan yang OPD lakukan. Hasil dari penelitian ini yaitu dibuatnya Aplikasi Pelaporan Kegiatan OPD Kabupaten Pringsewu untuk membantu OPD dalam melakukan proses pelaporan dan membantu Bappeda dalam melakukan proses monitoring dari kegiatan tersebut. Metode yang digunakan dalam penelitian ini adalah metode *Rapid Application Development* (RAD).

4. Aliyah, dkk., (2017), dalam jurnalnya yang berjudul "Implementasi *Web Service* Dalam *Monitoring* Pendapatan Perusahaan Dari Penjualan Tiket Bus Di Perum Damri Kantor Cabang Bandar Lampung Berbasis Web". Penelitian ini bertujuan untuk mengimplementasikan teknologi *web service* dalam memonitoring pendapatan harian dari penjualan tiket bus yang diharapkan dapat membantu dalam monitoring UPP harian secara *realtime*, serta mempercepat proses penyajian UPP harian kepada *General Manager*. Metode yang digunakan yaitu metode *System Development Life Cycle* (SDLC).
5. Rilyani, dkk., (2018) dalam jurnalnya yang berjudul "Aplikasi Pelaporan Dan *Monitoring* Data Limbah B3 Pada Tempat Penyimpanan Sementara Berbasis Web Di PT. PLN (Persero) Sektor Pengendalian Pembangkitan Bandar Lampung". Staf K3L kantor sektor dan staf LK2 unit pembangkit mengalami kesulitan dalam melakukan proses pelaporan dan monitoring data limbah B3 yang dilakukan dengan cara mendatangi kantor sektor maupun unit pembangkit secara langsung karena jarak yang harus ditempuh jauh dan menghabiskan waktu yang lama. Penelitian ini dilakukan untuk menghasilkan aplikasi pelaporan dan *monitoring* data limbah B3 berbasis web untuk membantu proses pengolahan dan pelaporan data limbah B3 pada tempat penyimpanan sementara sehingga proses pengolahan data limbah B3 dapat terpantau. Metode yang digunakan dalam tugas akhir ini yaitu metode *Rapid Application Development* (RAD).