BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Terdapat beberapa landasan teori yang digunakan dalam pembuatan penelitian ini, diantaranya adalah sebagai berikut:

2.1.1 *Klinik Kecantikan Mische*

Klinik kecantikan merupakan fasilitas pelayanan kesehatan yang menyelenggarakan pelayanan medis dasar dan/atau spesialistik yang berfokus pada estetika dan perawatan kulit. Menurut Permenkes No. 9 Tahun 2014, klinik adalah fasilitas pelayanan kesehatan perorangan yang dapat diselenggarakan oleh pemerintah, pemerintah daerah, atau swasta, dan menyediakan pelayanan promotif, preventif, kuratif, serta rehabilitatif. Klinik kecantikan biasanya memberikan layanan seperti perawatan wajah, pengobatan jerawat, terapi anti-aging, serta penjualan produk perawatan kulit. Dalam perkembangannya, klinik kecantikan membutuhkan sistem informasi yang dapat mendukung pelayanan tersebut agar berjalan dengan baik, profesional, dan mampu membangun kepercayaan pelanggan

Gambar 2. 1 Logo Klinik Kecantikan Mische

Dalam penelitian ini, penerapan dilakukan pada Klinik Kecantikan Mische, sebuah klinik kecantikan yang telah berdiri sejak tahun 2014 di Kota Pekanbaru, Riau, Indonesia, yang memiliki berbagai layanan estetika dan perawatan kulit yang profesional. Berdasarkan hasil wawancara yang terdapat pada **Lampiran C**, dalam menjalankan operasionalnya, klinik kecantikan Mische menyediakan berbagai produk *skincare* yang disesuaikan dengan kebutuhan perawatan kulit *customer*. Kemudian terdapat juga pelayanan treatment kulit yang dapat dilakukan reservasi, sehingga setiap customer dapat menyesuaikan dengan waktu yang dimilikinya. Selain itu, klinik ini juga sering membuat sebuah *event* di berbagai lokasi sebagai sarana untuk melakukan promosi dan edukasi serta dilegkapi dengan penawaran

promo baik kode voucher maupun diskon khusus yang menarik. Strategi ini bertujuan untuk menarik lebih banyak customer serta meningkatkan loyalitas customer terhadap klinik kecantikan mische.

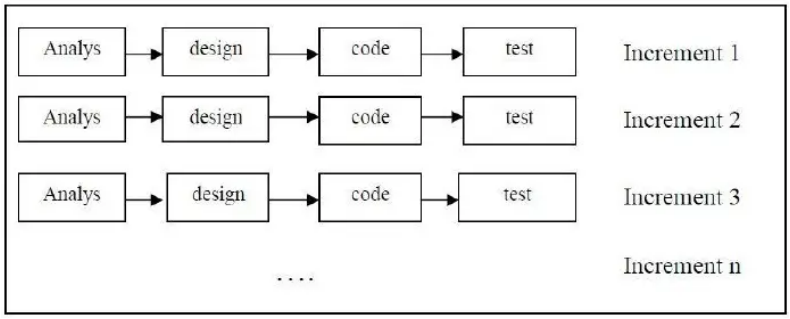
2.1.2 *Sistem Informasi*

Sistem informasi merupakan suatu sistem yang dirancang untuk mengumpulkan, menyimpan, mengelola, dan mendistribusikan informasi guna mendukung proses pengambilan keputusan dan aktivitas operasional dalam suatu organisasi. Menurut Andini et al. (2023), sistem informasi merupakan sebuah sistem yang melakukan sebuah pengumpulan, menyimpan, mengelola, memproses, dan peyebaran informasi dalam sebuah instansi atau organisasi. Sistem informasi juga berguna untuk meningkatkan efisiensi dalam melaksanakan operasional instansi atau organisasi serta memberikan dukungan strategi dalam melakukan pengambilan sebuah keputusan. Sehingga sistem informasi dapat berguna untuk memenuhi akan kebutuhan user dalam menanggulangi permasalahan yang di hadapi, contohnya memberikan kebijakan dan startegi bisnis yang dapat diterapkan selama melaksanakan operasional perusahaan, dan hal ini memberikan dampak dan keuntungan berupa meningkatkan efisiensi dalam sebuah instansi atau organisasi.

Menurut Oktaviyana (2023) sistem informasi mengandung tiga aktivitas dasar di dalamnya, yaitu terdapat aktivitas untuk melakukan sebuah masukkan (*input*), kemudian aktivitas untuk melakukan sebuah pemprosesan (*processing*), kemudian yang terakhir terdapat aktivitas keluaran (*output*). Kemudian dari ketiga aktivitas ini akan menghasilkan sebuah informasi yang dibutuhkan oleh organisasi untuk melakukan pengambilan keputusan, strategi pemasaran ataupun menciptakan produk baru . Masukkan (*input*) berperan untuk melakukan pengumpulan data mentah. Kemudian pemprosesan (*Processing*) merupakan tahapan pengolaahan data dari *input*-an menjadi sebuah informasi yang dapat digunakan pada organiasi. Dan yang terakhit keluaran (*output*) yaitu hasil dari proses yang dapat digunakan oleh perusahaan. Sehingga sistem informasi sangat penting dalam organisasi terutama dalam pengambilan keputusan organisasi untuk kedepannya.

2.1.3 *Metode Pengembangan Incremental*

Metode *Incremental* adalah salah satu metode pengembangan perangkat lunak (*Software Development*) yang dilakukan dengan pendekatan secara bertahap (*increment*) serta melibatkan iterasi yang dilakukan secara berulang selama pengemambangan sistem, sebagaimana dijelaskan oleh Sitepu et al. (2023). Motode ini juga melakukan penerapan dari elemen elemen yang ada pada metode *waterfall* dengan dibantu filosofi iteratif dari metode *prototyping*.Pada metode ini dibagi menjadibagian bagian kecil yang dimana disebut *increment*. Untuk disetiap *increment* dilakukan penambahan fitur atau fungsi yang sudah dapat dilakukan pengujian, setelah satu *increment* selesai dan sudah memenuhi akan kebutuhan dari *user* maka tahapan pengemabangan dilanjutkan ke *increment* yang berikutnya hingga sistem akan jadi sepenuhnya. Sehingga dengan metode ini fitur fitur yang sudah dibuat dan sesuai dengan kebutuhan pengguna nantinya bisa langsung digunakan oleh *user*. Selain itu, metode *incremental* juga memberikan kemudahan dalam melakukan penyesuaian baik itu perbaikan maupun penambahan kebutuhan dalam pengembangan sehingga dengan metode ini dapat langsung menyesuakan dengan kebutuhan dari *user*.



Gambar 2. 2 Tahapan Metode Incremental
Sumber: Habib & Kartika (2020)

Pada metode pengemabangan *Incremental* memiliki empat tahapan di dalamnya, seperti yang dijelaskan oleh Sutjiadi et al. (2022) sebagai berikut ini:

a) Analisis

Tahapan analisis pada metode *incremental* adalah dimana dilakukannya identifikasi terhadap masalah yang terjadi kepada calon pengguna sistem agar menemukan kebutuhan utama dan juga kebutuhan pendukung pada sistem tersebut.

b) Desain

Tahapan desain merupakan tahapan melakukan perancangan untuk sistem yang akan dikembangkan berdasarkan kebutuhan yang telah di dapatkan pada tahapan analisis.

c) Implementasi

Tahapan implementasi pada metode *incremental* yaitu tahapan dimana dilakukan nya implementasi ide desain ke dalam sebuah baris kode sistem yang akan dibangun.

d) Testing

Tahapan testing merupakan tahapan validasi dari hasil pembuatan pada tahapan sebelum sebelum nya apakah penerapan kode sudah sesuai dengan kebutuhan pengguna atau belum.

2.1.4 *Entity Relation Diagram (ERD)*

Menurut Akbar & Haryanti (2023) *Entity Relation Diagram* (*ERD*) adalah sebuah diagram yang dibentuk secara terstruktur yang digunakan untuk merancang sebuah basis data dari sistem yang sedang dibagun. Dengan kata lain, *ERD* menjadi suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. Selain itu *ERD* juga menjelaskan bagaimana ketentuan, batasan agar tetap konsisten dan valid selama diproses oleh sistem.

Dalam melakukan perancangan *ERD* perlu diperhatikan tiga konsep didalamnya, sebagai mana di jelaskan oleh Togatorop et al. (2021) *ERD* memiliki tiga konsep utama sebagai berikut ini:

a) Entitas

Entitas merupakan objek penting dalam sebuah organisasi seperti orang, tempat, benda ataupun kejadian yang harus dicatat pada basis data. *Entitas* juga terbagi menjadi dua jenis, yaitu *strong entity* (tidak bergantung pada entitas lain) dan *weak entity* (yang bergantung pada entitas lain).

b) Atribut

Pada *ERD* terdapat sebuah atribut yang dimana atribut adalah sebuah bagian yang mendeskripsikan *entitas*. Atribut terbagi atas beberapa jenis yaitu, simple

attributedan composite attribute, single valued attributedan multi value attribute, derived attribute, key attribute. Primary key merupakan sebuah nama dari atribut yang bersifat unik dan tidak memiliki kemiripan dengan *atribut* lain yang berguna untuk memberitahu lokasi seuatu data.

c) Relasi

Relasi adalah sebuah hubungan antara satu entitas dengan entitas lainnya yang saling berkaitan yang dilambangkan dengan bentuk belah ketupat (*diamond*). *Relasi* juga terbagi atas tiga jenis yaitu *unary, binary, ternary*.

Dari ketiga konsep tersebut yaitu *Entity*, *Antribut*, dan *Relasi* memiliki simbol dalam pembuatan nya seperti pada gambar 2.3 sebagai berikut ini:

Gambar 2. 3 Simbol Entity, Atribut, dan Relasi
Sumber: Togatorop et al. (2021)

2.1.5 *Database*

Database adalah sekumpulan data yang terorganisir secara sistematis untuk memudahkan penyimpanan, pengelolaan, dan pengambilan informasi. Menurut penelitian, *database* berfungsi sebagai alat yang memungkinkan pengguna untuk mengakses dan memanipulasi data secara efisien melalui sistem manajemen basis data (*DBMS*). Keunggulan utama *database* terletak pada kemampuannya untuk mengurangi redundansi data, meningkatkan integritas informasi, dan mendukung analisis data yang kompleks. Penelitian menunjukkan bahwa penerapan *database* dalam berbagai bidang, seperti bisnis dan pendidikan, dapat meningkatkan efisiensi operasional serta mendukung pengambilan keputusan yang berbasis data sebagaimana dijelaskan oleh Fahzirah & Nasution (2024).

Database juga memiliki berbagai macam jenis nya, menurut Daud Pujas et al. (2024) terdapat dua jenis database yang paling banyak digunakan oleh *software developert* dalam mengembangkan sebuah sistemnya, kedua jenis *database* tersebut adalah:

a) *Database SQL*

Database SQL (Structured Query Language) adalah *database* yang memiliki bentuk yang terstruktur dan saling berelasi satu tabel dengan tabel lain nya

berdasarkan Primary Key.

b) *Database NoSQL*

Database NoSQL (*No Structured Query Language*) adalah *database* memiliki konsep tidak berelasi (*non-relational*) yang dapat melakukan penyimpanan data tanpa memerlukan definisi struktur tabel terlebih dahulu. *NoSQL* dapat menyimpan berbagai macam bentuk data baik itu dalam bentuk dokumen, *graph*, *dan key-value*.

Kedua *database* tersebut memiliki kelebihan dan kekurangan nya masing masing, sehingga diperlukan analisis akan kebutuhan terlebih dahulu sebelum memilih dan menggunakannya.

2.1.6 *MySQL*

Menurut Hermiati et al. (2021)*MySQL* adalah salah satu sistem manajemen basis data relasional *RDBMS* (*Relational Database Management System*) yang cukup terkenal, yang dimana dapat melakukan pengolahan terhadap data yang terstruktur karena bahasa yang digunakan adalah bahas *SQL* (*Strukture Query Language*). *Database* relasional menyimpan data pada tabel-tabel yang terpisah, bukan menyimpan data dalam ruang Menurut Musliyana & Helinda (2022) *MySQL* merupakan sistem *database* yang banyak digunakan untuk pengembangan aplikasi *web* karena pengolahan datanya sederhana dan memiliki keamanan yang baik. Kemampuannya untuk menyimpan data dalam jumlah besar dan memberikan akses cepat terhadap data melalui *query SQL* menjadikannya pilihan utama dalam pengolahan data.

Gambar 2. 4 Logo MySQL

Kemudian *MySQL* bersifat *open source* dan juga memiliki kinerja yang cukup baik karena *MySQL* ini tidak memerlukan penggunaan *RAM* yang besar, serta juga mendukung dalam penggunaan *multiuser*, sebagaimana dikemukakan oleh Silalahi (2022). Keunggulan keunggulan ini menjadikan *MySQL* sebagai pilihan yang banyak digunakan oleh *developer* dalam melakukan pengembangan sistem yang

membutuhkan sebuah tampilan yang dinamis. Selain itu *MySQL* memiliki kompetabilitas akan bahasa yang cukup banyak dan juga terhadap *framework modern* sehingga mempermudah dalam melakukan proses integrasi terutama terhadap sisi *backend* yang langsung berkomunikasi dengan *database*. Kemampuan *MySQL* dalam menangani beban kerja yang tinggi secara efisien, serta kemudahan integrasinya dengan berbagai teknologi dan *framework*, semakin memperkuat posisinya sebagai pilihan utama bagi para developer dalam membangun aplikasi yang handal dan skalabel.

2.1.7 *Backend*

Backend merupakan sebuah kumpulan dari beberapa program dan *script* yang saling terintegrasi kemudian dijalankan pada server dan bekerja di bekalang dari sebuah sistem yang digunakan oleh pengguna. *Backend* dapat dikatakan juga sebagai sebuah penampung dari logika inti dan fungsional pada sebuah sistem, sebagaimana dikemukakan oleh Herdiyatmoko (2022). Sistem *backend* akan memastikan proses melakukan penampilan dan juga pengiriman data ke *frontend* dilakukan melalui metode terprogram. Kemudian *backend* terdiri dari beberapa komponen dimulai dari server, *database*, dan *API* (*Application Programming Interface*) yang membantu untuk melakukan interaksi dengan *frontend* secara aman dan terstruktur. Dengan kata lain *backend* memastikan dari semua interaksi yang dilakukan *user* seperti melakukan pemesanan produk, penjadwalan dan lain lainnya yang dilakukan pada sistem melalui *frontend* dapat berjalan dengan baik sesuai dengan logika yang sudah dirancang

2.1.8 *Laravel*

Laravel adalah *framework* merupakan salah satu *framework* yang berbasis bahasa pemrograman *PHP* yang dimana dapat digunakan untuk melakukan pengembangan sistem terutama sistem *backend*, sebagaimana dikemukakan oleh Melyani et al. (2023). Kemudian pada *framework laravel* juga memiliki berbagai macam fitur didalam yang dapat membantu *developer* dalam melakukan pengembangan sistem. Untuk fitur yang disediakan oleh *framework laravel* diantaranya adalah *Bundles*, *Eloquent ORM* (*Object-Relational-Mapping*), *Query Builder*, *Resource Controller*, *Blade*, *Migration*, *Middleware*, *dan*

Automatic Pagination. Sehingga hal ini menjadikan keunggulan bagi *framework laravel* dari segi penulisan kode yang lebih singkat, mudah untuk dipahami, dan ekspresif. Sehingga *framework laravel* sangat cocok digunakan untuk melakukan pengembangan sistem *backend* karena didukung penuh dengan fitur dari *framework laravel.*

Gambar 2. 5 Logo Laravel

Laravel adalah salah satu *framework* yang menerapkan konsep MVC (*Model-View-Controller*) di dalamnya, seperti yang tampak pada gambar 2.6. Menurut Salam et al. (2023) *MVC* membagi aplikasi menjadi tiga komponen utama yang terpisah, dengan pembagian fungsi sebagai berikut:

a) *Model*

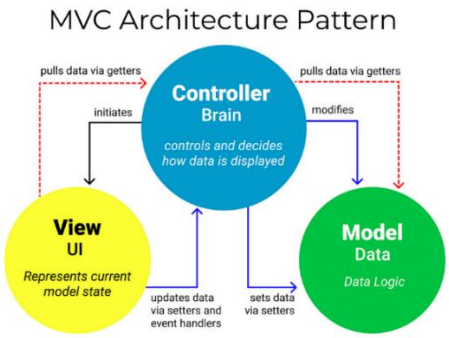
Model adalah kelas logika dari sistem yang berguna untuk melakukan penghubung antar sistem dengan *database*, serta digunakan juga untuk melakukan manipulasi data.

b) *View*

View adalah kelas yang mewakiki fungsi antarmuka suatu sistem dan menjadi penghubung interaksi antara sistem dengan pengguna.

c) *Controller*

Controller berfungsi sebagai penghubung antara *View* dan *Model*. *Controller* menerima *input* dari *View*, memproses data tersebut, kemudian mengirimkannya ke *Model* untuk disimpan ke dalam *database*, dan sebaliknya mengatur data dari *Model* agar dapat ditampilkan kembali ke *View.*



Gambar 2. 6 Arsitektur MVC
Sumber: Rahmawati & Sumarsono (2024)

2.1.9 *PHP*

PHP adalah singkatan dari *Hypertext Preprocessor* merupakan bahasa yang berbasis *server-side* yang diamana berarti dilakukan pengelohan dibagian server dan di desain untuk melakukan pengembangan *web*, sebagaimana dikemukakan oleh Noviana (2022). *PHP* merupakan bahasa skrip yang ditambahkan ke dalam *HTML* kemudian digunakan untuk membuat *website* yang *memiliki* tampilan yang dinamis dan dapat disesuaikan dengan kebutuhan *user*, sehingga informasi yang diterima oleh *clien*t selalu *up to date,* seperti dijelaskan oleh Apandi & Istini (2023). *PHP* dapat menjalankan kode secara langsung di server sebelum konten dikirimkan ke browser, sehingga memungkinkan interaksi yang lebih kompleks dengan pengguna. Bahasa ini juga mendukung berbagai protokol komunikasi yang penting dalam pengembangan aplikasi *web* modern.

PHP memiliki *sintaks* yang sederhana, fleksibel dan didukung dengan komunitas yang cukup besar. Sehingga bahasa pemrograman *PHP* dapat digunakan untuk melakukan pengembangan sistem *backend* karena mampu melakukan pengolahan bisnis serta dapat menjembatani pertukaran data antara server dengan klien dengan baik.

Gambar 2. 7 Logo PHP

PHP juga memiliki beberapa kelebihan seperti yang dipaparkan oleh Sitanggang et al. (2022) sebagai berikut ini:

a) Bahasa *multiplatform*

PHP merupakan bahasa yang dapat di jalankan di berbagai macam platform yang ada di dunia mulai dari *Linux*, *Unix*, *Macintos* dan juga *Windows*.

b) *Open source*

PHP bersifat *open source* yang berarti dapat digunakan oleh siapa saja dan gratis tanpa harus membayarkan royalti kepada pihak pengembang bahasa *PHP*.

c) Memiliki banyak dukungan *web server*

Ketersediaan *web server* yang mendukung *PHP* sangat banyak di dunia ini mulai dari apche, *IIS*, *Lighttpd*, *nginx*, hingga *Xitami* dengan konfigurasi yang cukup mudah dilakukan.

d) Mudah dalam pengembangan

Dari sisi pengembangan *PHP* sudah memiliki komunitas *developer* yang besar sehingga dapat saling bertukar solusi mengenai *PHP*.

e) Menduukung banyak *database*

PHP juga memiliki berbagai macam dukungan terhadap *database* yang ada mulai dari *MySQL*, *Oracle* dan *MS-SQL*.

2.1.10 *Laragon*

Menurut pendapat dari Fanani et al. (2023) *Laragon* merupakan perangkat yang digunakan oleh pihak *developer* dalam melakukan pengembangan sistem, yang dimana berguna untuk menyediakan lingkungan pengembangan secara lokal (*localhost*) bisa juga dikatakan sebagai penyedia *server virtual* yang dijalankan pada sistem operasi yang digunakan pihak pengembang. Hal ini sangat penting terutama dalam fase awal pembangunan aplikasi, karena *developer* dapat melakukan pengujian secara langsung tanpa harus bergantung pada server produksi. Dengan adanya *Laragon*, proses konfigurasi yang biasanya kompleks menjadi lebih sederhana dan otomatis, sehingga menghemat waktu dan meminimalkan kesalahan konfigurasi manual yang umum terjadi saat *setup server* secara manual.

Gambar 2. 8 Logo Laragon

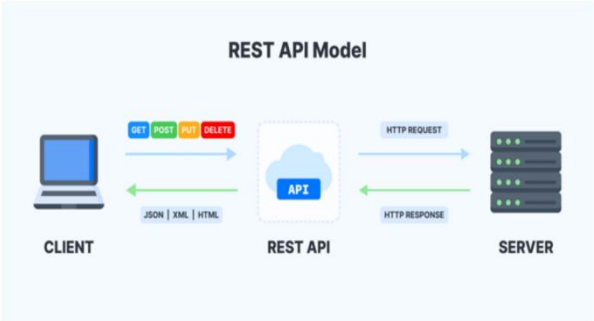
Kemudian laragon memiliki banyak fungsi seperti yang di sampaikan oleh Fanani et al. (2023) di dalam *Laragon* terdapat fungsi yang berguna untuk mendukung proses pengembangan sistem seperti apache, *PHP Server*, *PhpMyAdmin*, *MySQL*, *Composer*. Keberadaan komponen komponen tersebut dapat memudahkan pengembang tanpa melakukan pengaturan dan instalasi masing masing *tools* yang akan digunakan dalam pengembangan secara terpisah-pisah. Sehingga dengan fitur fitur yang disajikan tersebut menjadikan laragon populer dalam melakukan pengembangan sistem. Selain itu, *Laragon* juga mendukung penambahan modul atau stack tambahan seperti *Node.js*, *Python*, atau *database* lain secara fleksibel sesuai kebutuhan proyek. Keunggulan lainnya adalah antarmuka pengguna yang bersih dan intuitif, serta kemampuan *auto-start service* yang membuat *workflow* pengembangan menjadi lebih mudah dan stabil, bahkan saat sistem di-*reboot* sekalipun.

2.1.11 *Rest API*

Menurut Shofyan & Shita (2024) *Rest API* adalah sebuah *API* (*Application Programming Interface*) yang didalam nya menggunakan arsitektur *REST* untuk melakukan komunikasi antara *client* dengan *server* sistem. Sehingga membuat sistem menampilkan data, mengakses, mengolah menggunakan *HTTP* (*Hyper Text Transfer Protocol*) *request* dan dibantu dengan *URLs* dalam mengelola sumber data yang digunakan. Pendekatan ini dapat mempermudah bagi pengembang untuk melakukan pemeliharaan karena disetiap endpoint karena memiliki struktur yang konsisten. Selain itu, penggunaan *Rest API* juga bersifat stateless yang mana artinya setiap kali permintaan dari client harus dapat menyajikan semua informasi yang dibutuhkan.

Bila menggunakan *Rest API* terdapat beberapa permintaan *HTTP* yang umum digunakan seperti yang di sampaikan oleh Nurjaman et al. (2024) pada arsitektur *REST* seperti menggunakan *GET*, *POST*, *PUT*, dan *DELETE* untuk dapat melakukan operasi *CRUD* (*Create*, *Read*, *Update*, *Delete*) terhadap data yang disimpan pada *server*. Penggunaan *Rest API* menjadikan komunikasi antar *client* dan juga *server* menjadi lebih sederhana dan mudah, efisien dan dapat terintegrasi dengan sistem yang berbeda *platform*. Selain itu *Rest API* mempresentasikan data nya menggunakan format yang ringan untuk dibaca seperti *JSON* (*JavaScript*

Object Notation) dan juga *XML*. Hal ini menjadikan *Rest API* banyak digunakan pada web pada saat ini. Dukungan format data yang fleksibel ini sangat membantu integrasi antar sistem, khususnya ketika satu aplikasi membutuhkan data dari layanan lain tanpa harus melakukan parsing kompleks.



Gambar 2. 9 Arsitektur Rest API
Sumber: Mohidin (2022)

2.1.12 *Postman*

Menurut Raharja & Wijayanto (2023) Postman merupakan aplikasi yang digunakan oleh developer untuk membantu dalam proses melakukan pengujian, kemudian digunakan juga untuk melakukan dokumentasi, kolaborasi, dan juga pemantauan *API* yang dibuat secara *real-time*. *Developer* dapat menggunakan *postman* untuk melakukan pengecekan terhadap hasil *response* yang diberikan oleh *API*, yang dimana umumnya untuk *response* nya itu berbentuk *JSON*. Sehingga *Postman* sangat membantu dalam melakukan *debugging* terhadap *API* yang sedang dibuat oleh *developer* agar mengetahui *API* yang dibuat berjalan sesuai dengan fungsinya. Dalam pengembangan sistem terutama dibagian *backend* untuk menguji kestabilan dan keamanan *API* sebelum akan digunakan oleh *frontend.*

Gambar 2. 10 Logo Postman

Postman memiliki beberapa fitur unggulan yang dimiliki dan dapat membantu *developer* dalam proses melakukan pengujian dan didukung juga dengan pemaparan dari Herdiputra et al. (2021) fitur unggulan yang terdapat pada postman sebagai berikut ini:

a) *Collection*

Fitur ini berguna untuk melakukan pengaturan untuk *API* dilakukan

pengelompokkan atau secara dalam folder.

b) *Environment*

Fitur ini berguna untuk melakukan pengembangan dalam menyimpan attribute yang akan dimanfaatkan pada proses request *API*.

c) *Response*

Fitur ini digunakan untuk melakukan visualisasi terhadap *API* sebelum dilakukan implementasikan pada produk asli.

d) *Mock Server*

Fitur ini akan berguna untuk membuat *API* menjadi sebuah layanan yang telah di deploy pada *server*.

e) *Script Test*

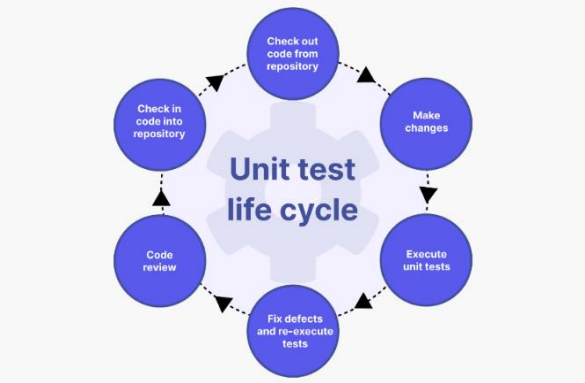
Fitur ini dapat dimanfaatkan untuk melakukan verifikasi respon dan test yang dibutuhkan pada pengembangan aplikasi.

f) *Automated Test*

Fitur *Automated Test* merupakan fitur yang dilakukan secara otomatis dengan script test yang digunakan untuk melakukan *request* dalam *collection*

2.1.13 *Unit Testing*

Menurut Hasibuan & Dirgahayu (2021) pengujian unit testing merupakan teknik pengujian untuk memastikan kode program perangkat lunak apakah sudah sesuai dengan dan bebas dari kesalahan. Pengujian ini dilakukan pada lingkup terkecil dari perangkat lunak, yaitu pada unit atau komponen terkecil yang dapat berdiri sendiri, seperti fungsi, prosedur, atau kelas. Hal ini sejalan dengan pendapat yang disampaikan oleh Abraham et al. (2021) juga memaparkan mengenai unit testing, yang dimana berguna untuk melakukan pengetesan dari setiap kode secara individu dimulai dari fungsi hingga methods. Penerapan unit testing juga memberikan manfaat dalam meningkatkan keandalan perangkat lunak, mempermudah proses perawatan (*maintenance*), serta mendukung praktik pengembangan modern, di mana kode diuji terlebih dahulu sebelum diintegrasikan dalam sistem yang lebih kompleks.

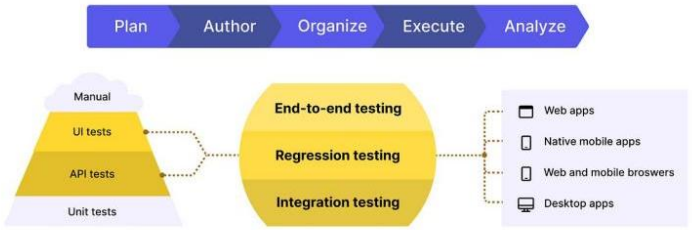


Gambar 2. 11 Siklus Unit Testing

Karakteristik unit test yang baik adalah cepat, sederhana, dan terisolasi. Pola *AAA* (*Arrange*, *Act*, *Assert*) digunakan untuk mengatur struktur unit test agar jelas dan mudah dibaca, di mana persiapan data, pemanggilan fungsi, dan verifikasi *output* dilakukan secara terstruktur. Implementasi unit testing dapat dilakukan dengan pendekatan *Test-First Developmen*t (*TFD*) atau siklus iterasi yang melibatkan penulisan kode uji, eksekusi pengujian (berhasil dan gagal), serta refactoring Rizkyana et al. (2021). Manfaat unit testing meliputi mempermudah perbaikan desain kode, mengurangi waktu *debugging*, memastikan kualitas komponen perangkat lunak, dan menyediakan pengujian menyeluruh untuk memastikan kode berkualitas baik. Dengan demikian, unit testing memainkan peran penting dalam memastikan keandalan dan efisiensi pengembangan perangkat lunak.

2.1.14 *Integration Testing*

Integration testing adalah salah satu testing yang biasa di gunakan untuk penggunaan pengembangan sistem. Menurut Setiawan & Risal (2024) *integration testing* merupakan pengujian yang berfokus pada bagian beberapa komponen sekaligus dan memastikan dapat berjalan dengan baik sesuai dengan kebutuhan pengguna. Pengujian ini penting walaupun kode program sudah melewati bagian unit testing, namun interaksi antar komponen belum dilakukan sehingga berkemungkinan dapat menimbulkan masalah baru yang tidak terlihat pada bagian pengujian sebelumnya. Hal ini menjadi sangat penting mengingat setiap komponen dalam sebuah sistem seringkali dikembangkan secara terpisah oleh tim yang berbeda dan memiliki dependensi data maupun fungsi yang saling terhubun.

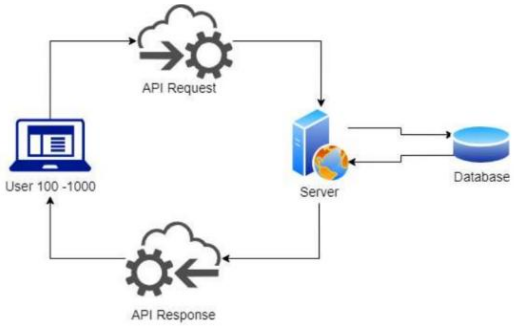


Gambar 2. 12 Integration Testing

Selain itu *Integration testing* memiliki tujuan khusus seperti yang di sampaikan oleh Aprila & Victor (2025) yaitu untuk mengetahui ketidak kesesuain antara komponen satu dengan komponen lain yang saling berinteraksi atau masalah antar dua komponen yang telah di gabungkan menjadi satu kesatuan. Melalui integration *testing*, masalah-masalah seperti kesalahan pada format data yang dipertukarkan, ketidakcocokan logika antar komponen, hingga kegagalan pada mekanisme komunikasi dapat diidentifikasi dan diperbaiki lebih awal sebelum sistem diimplementasikan secara penuh. Dengan demikian, pengujian ini berperan penting dalam memastikan integrasi komponen sistem yang handal, stabil, dan sesuai dengan kebutuhan pengguna. Dengan kata lain, *integration testing* membantu memastikan bahwa sistem tidak hanya berfungsi dengan baik pada masing-masing modul, tetapi juga mampu bekerja secara satu kesatuan yang utuh.

2.1.15 *Performance Testing*

Menurut Sianturi et al. (2021) *peformance testing* merupakan teknik pengujian perangkat lunak untuk mengetahui sejauh mana perangkat lunak dapat berjalan dibawah beban kerja yang di inginkan, atau bisa disebut katakan juga evaluasi kemampuan program untuk beroperasi dengan benar dipandang dari sisi acuan kebutuhan, misalnya ukuran pemakaian memori. Pada pengujian ini biasanya mencakup dari pengukuran waktu respon, tingkat pemanfaatan kemampuan *CPU*, *memory*, *bandwidth* jaringan, serta kestabilan dalam mengatasi beban lain. Kemudian testing ini dilakukan dengan kondisi ketika sudah *deploy* dari sistem agar mengetahui kinerja setelah di implementasi pada lingkungan produksi.



Gambar 2. 13 Alur Pengujian Peformance Testing
Sumber: Mulana et al. (2022a)

Tujuan dari *peformance testing* menurut Mulana et al. (2022b) menyatakan bahwa performance testing memiliki tujuan untuk menguji scalability, availability dan kinerja beban baik dari sisi perangkat keras ataupun dari sisi perangkat lunak. Sehingga dengan menggunakan performance testing ini pengembang dapat memastikan sistem yang dikembangkan tidak hanya memenuhi kebutuhan fungsional tetapi juga memberikan pengalaman yang stabil dan dapat di akses dalam kondisi beban yang cukup besar. Dampak lain dari penerapan pengujian ini juga berupa meningkatkan keandalan dan kepuasan pengguna karena sistem mampu menampilkan respon yang cepat dan konsisten dalam berbagai macam kondisi.

2.1.16 *Security Testing*

Securiy Testing menurut Siregar (2021) mengatakan bahwa *security testing* adalah pengujian keamanan untuk menilai sejauh mana sistem dapat dikatakan aman. Pengujian ini dilakukan untuk mengetahui sejauh mana sistem ini dapat menjaga keamanan data yang digunakan dan dikelolanya. Tujuan dari testing ini untuk memastikan dari mekanisme sistem dapat berjalan dengan baik dan lancar serta tidak mudah untuk di tembuh oleh pihak yang tidak berwenang. Dalam penggunaan *security testing* dapat membantu pengembang untuk mengetahui tingkat kemanan dari sistem yang dibuat dan di uji. Dengan demikian penggunaan security testing akan menjadi tindakan yang baik dalam proses pengembangan perangkat lunak yang menekankan pada aspek keamanan sebagai salah satu prioritas utama.

Menurut BPMPP UMA (2022) diantara nya sebagai berikut ini:

a) *Vulnerability Scanning*

Pengujian ini merupakan jenis pengujian yang mencari kelemahan pada sebuah sistem seperti *Cross* Site *Scripting* atau *SQL Injection*.

b) *Security scanning*

Jenis pengujian ini dilakukan dengan cara pemindaian pada sistem yang ingin kita testing, yang dimana akan berfokus dari beberapa area seperti *applications*, *ports*, *websites*, *services*, *networks*.

c) *Penetration Testing*

Penetration testing merupakan teknik pengujian perangkat lunak dengan menerapkan simulasi serangan *cyber* di dalamnya.

d) *Risk Assessment*

Identifikasi dan analisis risiko keamanan pada perangkat dan jaringan, lalu mengelompokkan risiko berdasarkan tingkat bahaya (tinggi, sedang, rendah). Biasanya dilakukan oleh tim IT internal.

e) *Security Auditing*

Merupakan testing yang dilakukan dengan cara melakukan evaluasi pada langkah langkah keamanan pada dalam perusahaan dan dilakukan audit secara rutin.

f) *Ethical hacking*

Ethical *hacking* ialah pengujian keamanan yg dilakukan menggunakan semua teknik peretasan dan teknik serangan personal komputer terkait lainnya.

g) *Posture Assessment*

Testing yang berjalan dengan tes keamanan untuk mengevaluasi dan meningkatkan kemampuan manajemen risiko di perusahaan

2.2 Penelitian Terdahulu

Penelitian pertama yang dilakukan oleh Zulkifli & Rahmalisa (2022) membahas mengenai permasalahan dalam melakukan pengelolaan data pasien dalam Klinik Kecantikan Qatrunnada yang masih menerapkan pembukuan secara manual. Hal ini menciptakan keterlambatan dalam melakukan pencatatan, kesalahan pencatatan, duplikasi data sehingga merugikan efektifitas klinik. Untuk

itu, dilakukan perancangan dan implementasi sistem informasi berbasis website dengan menggunakan *framework Laravel* yang berbasis bahasa pemrograman *PHP*, kemudian didukung dengan penggunaan *MySQL* sebagai *database*. Penelitian ini juga menggunakan metode pengembangan sistem waterfall untuk memastikan tahap perancangan sampai implementasi berjalan secara sistematis dan terstruktur. Hasil implementasi sistem informasi ini menunjukkan peningkatan efektifitas dalam pengolahan data dibandingkan dengan metode manual. Data tersimpan dengan aman, mudah diakses, dan meminimalisasi kehilangan data. Klinik juga dapat mengakses laporan kunjungan secara realtime dan mencetak laporan bulanan yang sebelumnya tidak tersedia.

Penelitian kedua dilakukan oleh Amrina et al. (2021) yang mengangkat masalah tentang pengelolaan data yang masih konvensional, sehingga menimbulkan ketidak efisiensi dalam menjalankan kegiatan operasional, semisal seorang dokter ketika mencari data pasien yang diinginkan namun membutuhkan waktu yang lama. Sehingga hal ini melandasi dalam melakukan pengembangan sistem informasi klinik kecantikan yang menerapkan *framework Codeigniter* dengan basis bahasa pemrogramannya *PHP* serta didukung dengan *MySQL* sebagai *database*-nya. Untuk metode pengembangan sistem yang digunakan adalah *Scrum* . Sistem ini di rancang untuk mampu mengelola berbagai aspek klinik kecantikan seperti pasien, rekam medis, antrian pasien, dan manajemen logistik. Hasil dari implementasi sistem informasi menghasilkan fitur fitur Fitur-fitur seperti pencatatan data pasien yang cepat, pencarian data yang mudah, pengelolaan antrian, hingga pembuatan laporan operasional, sehingga dengan fitur tersebut dapat memberikan efisiensi dan efektifitas operasional klinik.

Kemudian untuk penelitian ketiga adalah penelitian yang dilakukan oleh Agustin (2025) pada penelitian ini dilandasi dengan permasalahan dalam melakukan pengelolaan jadwal treatment pasien di Klinik Kecantikan *Ramdani Skincare* yang masih dilakukan secara manual dan konvensional, sehingga seringkali terjadi keterlambatan dan risiko kelupaan pengingat jadwal pasien. Sehingga untuk mengatasi permasalahan yang terjadi pada penelitian ini adalah dengan melakukan pengembangan sebuah sistem informasi yang dapat melakukan pengingatan secara otomatis terkait jadwal *treatment* berbasis *SMS Gateway*. Pada

implementasinya diterapkan beberapa teknologi didalamnya yang meliputi bahasa pemrograman *PHP* dengan *framework* Laravel yang mendukung pengembangan aplikasi *web* secara efisien dan *MySQL* sebagai basis data untuk penyimpanan informasi pasien dan jadwal treatment yang terstruktur, dan metode pengembangan yang diterapkan adalah *waterfall*. Dengan sistem yang dikembangkan ini dihasilkan sebuah fitur yang melakukan pengingatan secara otomatis kepada pasien yang terjadwal treatment melalui *SMS*, sehingga dapat mengurangi resiko kelupaan dan memastikan pasien mendapatkan layanan tepat waktu. Sehingga dengan penerapan sistem pengingat pada Klinik Kecantikan *Ramdani Skincare* menunjukkan peningkatan dalam manajemen jadwal pasien serta efektivitas pelayanan klinik secara keseluruhan.

Penelitian yang keempat dilakukan oleh Badri & Sari (2023) penelitian ini memiliki tujuan untuk melakukan pengembangan sistem informasi terutama berfokus pada bagian manajemen penjualan produk obat untuk apotek, yang dimana sebelumnya masih menggunakan metode pengelolaan secara tradisional dengan kendala dalam melakukan pengelolaan data stok dan transaksi. Sehingga untuk mengatasi permasalahan yang terjadi, peneliti melakukan perancangan sistem yang tidak hanya mampu melakukan pencatatan terhadap penjualan, namun juga mempermudah dalam melakukan pengelolaan inventaris obat dan memberikan laporan penjualan. Dalam penelitian ini digunakan metode incremental yang dimana pada pengembangannya dilakukan secara bertahap melalui beberapa iterasi didalamnya. Kemudian penelitian ini dilakukan sebanyak dua iterasi di dalamnya, pada iterasi pertama dilakukan pengembangan dibagian pengelola data master obat dan transaksi penjualan harian, untuk iterasi kedua untuk pembuatan laporan penjualan dan analisis data. Metode *incremental* dipilih karena memiliki fleksibilitas yang tinggi, dapat dilakukan penyesuain kembali dengan bantuan umpan balik dari pengguna, serta dapat melakukan pengembangan menghadirkan fitur yang siap pakai dibandingkan dengan metode lain seperti *waterfall*. Hasil dari implementasi metode ini menunjukkan bahwa sistem dapat dilakukan pengembangan secara efektif dan efisien, dengan dari setiap tahapan memberikan evaluasi dan penyempurnaan yang sesuai dengan kebutuhan dari pengguna.

Penelitian yang kelima adalah penelitian yang dilakukan oleh Filiana et al.

(2022) pada penelitian ini mengangkat dari permasalahan penting terkait dengan ketiadaannya sistem informasi terpusat yang menyediakan dan mengelola dari pasar tradisional di kota Yogyakarta. Sehingga permsalahan tersebut melatar belakangi dilakukan nya pengembangan *Rest API* untuk sistem informasi pasar tradisional kota Yogyakarta. Metode pengembangan yang diterapkan pada penelitian ini menggunakan *incremental* yang dimana terdiri dari tiga iterasi, dimulai dari iterasi pertama membuat pengelolaan data utama pasar, iterasi kedua pengelolaan deskripsi dan sejarah pasar, kemudian iterasi terakhir tentang pengelolaan fasilitas dan kekhasan pasar. Kemudian teknologi yang diaplikasikan pada peneliatian ini meliputi penggunaan *laravel* sebagai *framework* dengan basis bahasa pemrogramannya adalah *PHP*, dan didukung dengan penggunaan *MySQL* sebagai *database*-nya. Penggunaan metode *incremental* dipilih karena metode ini memudahkan pengembangan yang sesuai dengan kebutuhan pengguna, serta didukung oleh proses evaluasi dan masukan pada setiap tahapannya.

Berdasarkan penelitian-penelitian yang sudah dipaparkan sebelumnya, berikut merupakan tabel 2.1 perbandinganya sebagai berikut ini :

Tabel 2. 1 Tabel Penelitian Terdahulu

Nama Pembanding	Zulkifli & Rahmalisa (2022)	Amrina et al. (2021)	Agustin (2025)	Filiana et al. (2022)	Badri & Sari (2023)	Penelitian Saat Ini
Studi Kasus	Sistem informasi klinik	Sistem informasi klinik	Sistem informasi klinik kecantikan	Sistem akuntansi koperasi	Sistem informasi pasar	Sistem Informasi Klinik Kecantikan Mische
Metode	Waterfall	Scrum	Waterfall	Incremental	Incremental	Incremental
Teknologi	Laravel, PHP, MySQL	PHP, MySQL	Laravel, PHP, MySQL, SMS Gateway	PHP, MySQL	Laravel, PHP, MySQL	Laravel, PHP, MySQL, REST API
Fitur	Pengelolaan data pasien dan pembuatan laporan bulanan	Pencatatan pasien, antrian, pengelolaan produk, dan rekam medis	Pengingat jadwal treatment melalui SMS	Otomatisasi laporan rugi laba, neraca, dan rekalkulasi saldo	REST API untuk pengelolaan data pasar, fasilitas, deskripsi, dan sejarah pasar	Company profile, reservasi treatment, pengelolaan promo dan event, penjualan produk, autentikasi pengguna, dashboard admin, dan layanan

Nama Pembanding	Zulkifli & Rahmalisa (2022)	Amrina et al. (2021)	Agustin (2025)	Filiana et al. (2022)	Badri & Sari (2023)	Penelitian Saat Ini
						REST API
Pembeda	Berfokus pada digitalisasi data pasien, sedangkan penelitian ini mengembangkan backend berbasis REST API dengan cakupan fitur yang lebih luas untuk Sistem Informasi Klinik Kecantikan Mische.	Berfokus pada pengelolaan operasional klinik tanpa implementasi backend berbasis REST API sebagai layanan integrasi dengan frontend.	Berfokus pada pengingat jadwal treatment melalui SMS, sedangkan penelitian ini mengembangkan backend yang mendukung berbagai modul sistem secara terintegrasi.	Menerapkan metode Incremental, namun digunakan pada sistem akuntansi koperasi sehingga tidak membahas pengembangan backend sistem informasi klinik kecantikan berbasis REST API.	Menerapkan metode Incremental dan REST API, tetapi studi kasus berfokus pada sistem informasi pasar dengan fitur yang berbeda dari kebutuhan klinik kecantikan.	Mengembangkan backend Sistem Informasi Klinik Kecantikan Mische menggunakan Laravel dengan pendekatan REST API dan metode Incremental yang berfokus pada pengelolaan logika bisnis, keamanan, pengolahan data, serta integrasi dengan frontend.