BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam tinjauan pustaka ini, akan dibahas sejumlah penelitian terdahulu yang memiliki keterkaitan dengan penelitian yang akan dilakukan. Penelitianpenelitian tersebut memberikan wawasan dan informasi yang relevan dengan pengembangan penelitian ini. Berikut adalah beberapa penelitian terdahulu yang relevan:

Pertama Penelitian berjudul “Rancang Bangun Sistem Informasi Laporan Laba Rugi Usaha Mikro Kecil Dan Menengah pada PT ASM.”. yang dibuat oleh I Made Suarta, Putu Inten Citrawati Purna, dan I G. A. Astri Pramitari pada 2021, Sistem ini dikembangkan dengan pendekatan *Rapid Application Development (RAD)*, menawarkan menu utama yang mencakup modul Expenses untuk memasukkan dan memvisualisasikan pengeluaran dan penerimaan perusahaan yang bukan bagian dari operasi bisnis inti, serta modul Report untuk menampilkan faktur, laporan laba rugi perusahaan, persentase gaji supir van bagasi, dan persentase gaji supir mobil wisata. Sistem ini juga diuji terhadap antarmuka pengguna dengan teknik black box *testing* dan *output* sistem diuji untuk memverifikasi kecocokan antara perintah program dan hasil yang diharapkan. Hasil pengujian menunjukkan konsistensi dengan desain yang telah dirancang sebelumnya [11].

Kedua Penelitian yang berjudul "Rancang Bangun Aplikasi Laporan Laba Rugi pada CV. NURI" yang dilakukan oleh Muhammad Ilham dan Muhammad Ridwan Lubis membahas tentang pembuatan aplikasi laporan laba rugi pada CV. NURI. Penelitian ini menerapkan metode pengambilan sampel (*sampling)*. Pembuatan aplikasi ini menggunakan Microsoft Visual Studio 2010 sebagai platform perancangan sistem, *MySQL* sebagai penyimpanan basis data, dan *Crystal Report* untuk pembuatan laporan. Tujuan dari riset ini adalah

untuk memudahkan tugas admin CV. NURI dalam menyusun laporan laba rugi berdasarkan periode bulanan dan harian. Dengan menggunakan aplikasi ini, admin CV. NURI dapat menghasilkan laporan yang akurat, rinci, tepat waktu, dan dapat diandalkan.[12].

Ketiga Penelitian yang berjudul "Rancang Bangun Sistem Informasi Akuntansi Laporan Laba Rugi Berbasis *Website* pada PT. United Tractors Pontianak" oleh Nurmalasari Nurmalasari, Anna Anna, Riska Arissusandi pada 2019 membahas tentang pengembangan sistem informasi akuntansi berbasis *Website* untuk mempercepat proses pengolahan laporan laba rugi di PT. United Tractors Pontianak. Penelitian ini menggunakan metode pengembangan perangkat lunak *waterfall* dengan menggunakan bahasa pemrograman *PHP* dan *MySQL* sebagai database. Aplikasi ini dirancang untuk dapat melakukan pengolahan data penjualan dan pembuatan laporan keuangan. Hasil penelitian ini diharapkan dapat membantu admin dan *Branch Operation Head* (BOH) dalam mengolah laporan laba rugi lebih mudah, cepat, dan akurat [13].

Keempat Penelitian yang berjudul "Sistem Informasi *Point of sales* Berbasis *Website* Pada Toko Usaha Mandiri" oleh Lutfi Zaitunnisaa dan Rita Wahyuni Arifin pada 2021 membahas pengembangan aplikasi *Point of sales* (POS) berbasis *Website* untuk toko Usaha Mandiri. Penelitian ini menggunakan metode *Extreme programming* dalam pengembangan sistem informasi POS berbasis *Website*. Tujuan penelitian ini adalah untuk membuat sebuah aplikasi yang dapat membantu pencatatan transaksi berdasarkan pembelian. Hasil penelitian ini adalah pengembangan aplikasi berbasis *Website Point of sales* (POS) yang dapat memudahkan sistem perekapan data penjualan dan mempercepat proses pelayanan kepada pembeli. Aplikasi ini diharapkan dapat meningkatkan efisiensi dalam pengolahan data dan meminimalkan kesalahan dalam pengelolaan data [14].

Kelima Penelitian berjudul “Perancangan Sistem Informasi Penjualan pada Toko Stock Point Lily” oleh Nery Nestary pada 2020. Tujuan penelitian ini adalah mempermudah pemilik toko dalam mengelola, menginformasikan, dan mencari data terkait penjualan dan laporan persediaan barang. [15].

Tabel 2. 1 Ringkasan Penelitian Sebelumya

No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan
1	Rancang Bangun Sistem Informasi Laporan Laba Rugi Usaha Mikro Kecil Dan Menengah [11].	Made Suarta, Putu Inten Citrawati Purna, I G. A. Astri Pramitari (2021)	PT ASM	*Rapid application development (RAD)*	Hasil penelitian menunjukkan bahwa pengembangan sistem informasi akuntansi menggunakan metode RAD dapat meningkatkan efisiensi dan akurasi laporan keuangan pada PT ASM. Dalam penelitian ini, dilakukan identifikasi dan penentuan prioritas teknologi dan aplikasi yang memberikan manfaat terbaik bagi perusahaan, analisis mendalam terhadap proses bisnis yang akan dikembangkan, evaluasi terhadap alternatif solusi dan	Pada penelitian sebelumnya menggunakan Menggunakan metode penelitan RAD dan menggunakan Microsoft Visual Studio 2019, serta pembuatan report menggunakan Crystal Report 8.5. Sedangkan pada penelitian ini menggunakan Metodologi XP dan menggunakan teknologi *Express.js* dengan *MySQL* sebagai basis datanya.

No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan
					spesifikasi rinci dari solusi berbasis Komputer, serta implementasi sistem informasi. Uji coba sistem dilakukan dengan metode *blackbox testing* dan hasilnya menunjukkan bahwa sistem informasi yang dikembangkan dapat menghasilkan laporan laba rugi yang akurat, tepat waktu, dan dapat dipercaya, serta memenuhi ketentuan standar akuntansi yang berlaku untuk PT ASM.	

2	Rancang Bangun Aplikasi Laporan Laba Rugi Pada	Muhammad Ilham, Muhammad	CV. NURI Pematang siantar	Metode sampling	Hasil penelitian di atas adalah berhasilnya pembuatan aplikasi laporan laba rugi pada CV. NURI menggunakan Microsoft	Penelitian sebelumnya menghasilkan aplikasi desktop yang dibuat dengan C++ dan aplikasi Microsoft Visual Studio 2010

No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan
	CV. NURI Pematangsiantar[12]	Ridwan Lubis (2019)			Visual Studio 2010 dan *MySQL* Database. Aplikasi ini dapat mempermudah admin CV. NURI dalam membuat laporan laba rugi berdasarkan periode bulanan dan harian. Dengan menggunakan aplikasi ini, admin CV. NURI dapat menghasilkan laporan yang akurat, terperinci, tepat waktu, dan dapat dipercaya.	sedangkan penelitian ini akan menghasilkan aplikasi berbasis *Website* dengan menggunakan *JavaScript* dan Visual Studio Code.

3	Rancang Bangun Sistem Informasi Akuntansi Laporan Laba Rugi Berbasis *Website* Pada Pt.	Nurmalasari Nurmalasari, Anna Anna, Riska Arissusandi (2019)	Pt. United Tractors Pontianak	*Waterfall*	Hasil penelitian tersebut adalah rancangan dan pengembangan sistem informasi akuntansi laporan laba rugi berbasis *Website* pada PT. United Tractors Pontianak. Aplikasi ini	Penelitian sebelumnya menggunakan metode penelitian Waterfall sedangkan penelitian ini menggunakan Metode XP (*Extreme programming*).

No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan
	United Tractors Pontianak [13].				dirancang untuk mempercepat proses pengolahan data penjualan dan pembuatan laporan keuangan. Diharapkan dengan adanya aplikasi ini, admin *dan Branch Operation Head* (BOH) dapat lebih mudah, cepat, dan akurat dalam mengolah laporan laba rugi.	

4	Sistem Informasi *Point of sales* Berbasis *Website* Pada Toko Usaha Mandiri[14]	Lutfi Zaitunnisaa dan Rita Wahyuni Arifin (2021)	Toko Usaha Mandiri	*Extreme programming*	Pengembangan sebuah aplikasi berbasis *Website Point of sales* (POS) yang dapat membantu toko Usaha Mandiri dalam pencatatan transaksi penjualan berdasarkan pembelian. Dengan adanya sistem informasi POS berbasis *Website*, diharapkan	Penelitian sebelumnya membuat aplikasi *Point of sales* di studi kasus Toko Usaha Mandiri, sedangkan penelitian ini membuat aplikasi Penghitung keuangan laba rugi studi kasus CemantingArt Ecoprint Purbalingga.

No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan

					dapat meningkatkan efisiensi dalam pengolahan data dan mempercepat proses pelayanan kepada pembeli. Selain itu, penggunaan aplikasi ini juga diharapkan dapat meminimalkan kesalahan dalam pengelolaan data dan memudahkan dalam pengolahan informasi data serta penggunaan transaksi penjualan bagi toko Usaha	
5	Perancangan Sistem Informasi Penjualan pada Toko Stock Point Lily[15]	Nery Nestary (2020)	Toko Stock Point Lily	UML (*Unified Modeling Language*)	Dengan adanya sistem informasi penjualan pada Toko Stock Point Lily, dapat mengelola data-data penjualan, pembelian, dan stok yang ada. Sehingga mempermudah dalam pencarian	Penelitan sebelumnya menggunakan Bahasa pemprograman *PHP* dan basis data *MySQL*, sedangkan penelitian ini menggunakan Bahasa pemprograman *JavaScript*, dengan
No	Judul	Penulis	Studi kasus	Metode / Algoritma	Hasil	Perbedaan dengan penelitian yang dilakukan

					data yang diperlukan dan mempercepat proses pencariannya.	*Framework Express .js* dan basis data *MySQL*.

Pada penelitian sebelumnya yang dilakukan oleh I Made Suarta, Putu Inten Citrawati Purna, dan I G. A. Astri Pramitari pada tahun 2021, menunjukkan bahwa sistem informasi laba rugi pada PT ASM berhasil dikembangkan menggunakan pendekatan *Rapid Application Development* (RAD) [11]. Metode ini menawarkan fleksibilitas dan kecepatan dalam pengembangan, namun penelitian ini memilih *Extreme programming* (XP) sebagai metodologi utama. XP dipilih karena fokusnya pada kolaborasi tim yang intensif, adaptasi cepat terhadap perubahan kebutuhan, dan iterasi pengembangan yang lebih pendek, sehingga sesuai dengan kebutuhan dinamis UMKM. Pemilihan *framework* Express.js untuk pengembangan *backend* didasarkan pada kemampuannya dalam membangun aplikasi web yang cepat dan *scalable*. Express.js memungkinkan pembuatan server yang ringan dan efisien, serta mendukung arsitektur *RESTFUL API* yang memudahkan integrasi dengan berbagai *frontend* dan aplikasi pihak ketiga. Selain itu, *MySQL* dipilih sebagai sistem manajemen basis data karena kestabilannya, keandalan dalam menangani data dalam jumlah besar, dan dukungan komunitas yang luas. *MySQL* juga memiliki performa yang baik dalam operasi CRUD (*Create*, *Read*, *Update*, *Delete*), yang sangat penting untuk sistem informasi penghitung laba rugi dan POS.

2.2 Landasan Teori

2.2.1 Laporan Laba Rugi

Laporan Laba Rugi, yang juga dikenal sebagai *Income Statement*, merupakan dokumen keuangan krusial yang merinci kinerja finansial suatu entitas dalam suatu periode tertentu. Tujuan utama laporan ini adalah untuk memberikan pemahaman yang jelas tentang seberapa baik atau buruk kinerja keuangan perusahaan selama periode waktu tertentu.[16] Melalui laporan ini, perusahaan dapat mengidentifikasi sumber pendapatan utama dan memahami sejauh mana beban operasional mempengaruhi profitabilitas. Laporan Laba Rugi tidak hanya berfungsi sebagai gambaran keseluruhan performa finansial perusahaan, tetapi juga sebagai alat analisis yang sangat berharga dalam pengambilan keputusan bisnis dan perencanaan strategis untuk masa depan [17].

Unsur-unsur dalam Laporan Laba Rugi:

1. Pendapatan (*Revenue*): Merupakan jumlah total uang atau nilai yang diterima oleh perusahaan dari penjualan barang atau jasa. Pendapatan mencakup semua sumber penerimaan yang terkait dengan kegiatan operasional perusahaan .

2. Beban (*Expense*): Meliputi semua biaya yang dikeluarkan oleh perusahaan untuk menjalankan operasionalnya. Beban dapat dibagi menjadi beberapa kategori seperti beban operasional, beban bunga, beban pajak, dan lainnya.

3. Laba (*Profit*): Jika total pendapatan melebihi total beban, maka perusahaan akan mencatat laba. Laba adalah selisih positif antara pendapatan dan beban.

4. Rugi (*Loss*): Sebaliknya, jika total beban lebih besar dari total pendapatan, perusahaan akan mencatat rugi. Rugi adalah selisih negatif antara pendapatan dan beban.

Cara Menghitung Laba Rugi:

Rumus umum untuk menghitung laba rugi adalah:

Net Income = (Pendapatan) – (Beban + Rugi)

Pendapatan dapat mencakup penjualan produk atau jasa, bunga yang diterima, dan pendapatan lainnya. Beban melibatkan biaya operasional seperti gaji karyawan, biaya listrik, bahan baku, dan berbagai beban lainnya.Laporan Laba Rugi memberikan gambaran keuangan yang kritis bagi pemangku kepentingan, seperti investor, manajemen, dan kreditor, untuk mengukur kinerja perusahaan dan mengambil keputusan strategis. Analisis laba rugi membantu dalam mengevaluasi profitabilitas perusahaan dan mengidentifikasi area-area yang memerlukan perhatian lebih lanjut [18].

2.2.2 *Point of sales* (POS)

Point of sales (POS) merupakan sistem atau perangkat lunak yang digunakan dalam proses transaksi penjualan barang atau jasa di sebuah bisnis. POS biasanya terdiri dari perangkat keras (*hardware*) seperti Komputer, scanner barcode, dan printer, serta perangkat lunak (*software*) yang memungkinkan pengguna untuk melakukan berbagai tindakan seperti mencatat penjualan, mengelola stok barang, dan menghasilkan laporan keuangan.Fungsi utama dari POS adalah untuk mengotomatiskan dan mempermudah proses transaksi penjualan. Dengan menggunakan POS, penjual dapat mencatat dengan cepat dan akurat setiap transaksi yang terjadi, mencatat jumlah barang yang terjual, menghitung total pembayaran, dan mengeluarkan bukti pembelian kepada pelanggan [19].

Selain itu, POS juga memiliki fitur untuk mengelola stok barang. Dengan POS, penjual dapat memantau stok barang yang tersedia, melakukan pemesanan barang baru jika stok mulai menipis, dan mengatur penempatan barang di toko secara efisien. Selain fungsi dasar tersebut, POS juga sering

dilengkapi dengan fitur tambahan seperti manajemen pelanggan, pembuatan laporan penjualan, integrasi dengan sistem keuangan, dan lain sebagainya[20].

2.2.3 *Website*

Website adalah kumpulan halaman yang tergabung dalam suatu domain atau subdomain, yang artinya halaman-halaman tersebut tersusun secara terstruktur di bawah alamat *Website* yang unik. *Website* merupakan salah satu komponen penting dari *World Wide Website* (WWW). Sebuah *Website* berjalan di *Web* browser, yang memungkinkan pengguna untuk mengakses informasi, konten, atau layanan yang disediakan oleh pemilik *Website* [21]. *Website* dapat berisi beragam jenis informasi, mulai dari teks, gambar, video, hingga fitur interaktif. Dengan kata lain, *Website* adalah wadah virtual di mana pengguna dapat menjelajahi dan berinteraksi dengan berbagai konten yang telah disusun dengan rapi berdasarkan topik, tujuan, atau kepentingan tertentu. *Website* merupakan salah satu sarana yang sangat penting dalam era digital untuk menyebarkan informasi, berkomunikasi, dan bertransaksi secara online[22].

2.2.4 *Backend*

Backend merupakan bagian kunci dari sebuah aplikasi yang beroperasi di sisi server, dan tugas utamanya adalah mengelola data. Hal ini mencakup interaksi langsung dengan database, teru tama terkait manipulasi data seperti penyimpanan, pengambilan, pembaruan, dan penghapusan data. Meskipun perannya sangat vital dalam proses pengolahan data, *Backend* tidak berinteraksi langsung dengan pengguna secara langsung. Sebaliknya, *Backend* menyediakan antarmuka yang dikenal sebagai *API* (*Application Programming Interface*) yang digunakan untuk berkomunikasi dengan data [23]. Dengan cara ini, *Backend* berfungsi sebagai jembatan yang menghubungkan dengan database, memungkinkan akses yang terkontrol dan aman ke informasi. Selain itu, *Backend* juga menangani logika bisnis, mengatur autentikasi , otorisasi, dan berbagai tugas lain yang diperlukan untuk menjalankan aplikasi dengan baik. Sebagai komponen penting dalam arsitektur aplikasi, *Backend* memastikan bahwa data diatur dengan baik, aplikasi berjalan efisien, dan pengguna dapat

berinteraksi dengan aplikasi dengan lancar tanpa harus terlibat langsung dengan database[24].

2.2.5 *REST API*

API (*Application Programming Interface*) adalah sebuah antarmuka yang berfungsi sebagai penghubung antara komponen *Frontend* dan *Backend* dalam sebuah sistem atau aplikasi. *API* memiliki peran penting dalam meningkatkan efisiensi dan produktivitas pengembang, karena memungkinkan penggunaan fungsi-fungsi yang sudah ada dalam aplikasi tanpa perlu mengembangkannya dari awal. Dengan kata lain, *API* memungkinkan berbagai komponen perangkat lunak untuk saling berkomunikasi dan berbagi data, sehingga mempermudah integrasi sistem dan pengembangan aplikasi yang lebih cepat [25]. Salah satu arsitektur *Backend* yang populer dalam pengembangan *API* adalah *Representational State Transfer* (*REST*). *REST* merupakan pendekatan desain yang menggunakan prinsip-prinsip sederhana dan mudah dimengerti untuk mengelola sumber daya (*resources*) melalui operasi-operasi standar seperti *GET, POST, PUT*, dan *DELETE*. Dengan *REST*, pengembang dapat merancang *API* yang bersifat *stateless*, mudah diimplementasikan, dan dapat digunakan oleh berbagai jenis aplikasi, termasuk aplikasi *Website* dan *mobile*. *REST API* telah menjadi standar *de facto* dalam pengembangan *API Website* karena kemudahan dalam pemahamannya dan skalabilitas yang tinggi [24].

2.2.6 *Node.js Framework Express.js*

Express.js adalah sebuah *Framework* untuk *Node.js* yang dirancang untuk memudahkan pengembangan aplikasi *Backend Website*. *Framework* ini memiliki banyak fitur yang membuatnya populer di kalangan *developer*. *Express.js* memungkinkan pengembang untuk membuat aplikasi *Backend Website* dengan cepat dan efisien [26]. Salah satu keunggulan *Express.js* adalah kemudahan dalam mengelola rute (routing). Pengembang dapat dengan mudah menentukan rute-rute yang akan digunakan oleh aplikasi *Website*, sehingga permintaan dari pengguna dapat diarahkan dengan tepat ke bagian yang sesuai dalam aplikasi. Ini sangat berguna untuk mengatur bagaimana aplikasi akan

menangani berbagai jenis permintaan, seperti permintaan *GET* untuk mengambil data, permintaan *POST* untuk mengirim data, dan sebagainya [27].

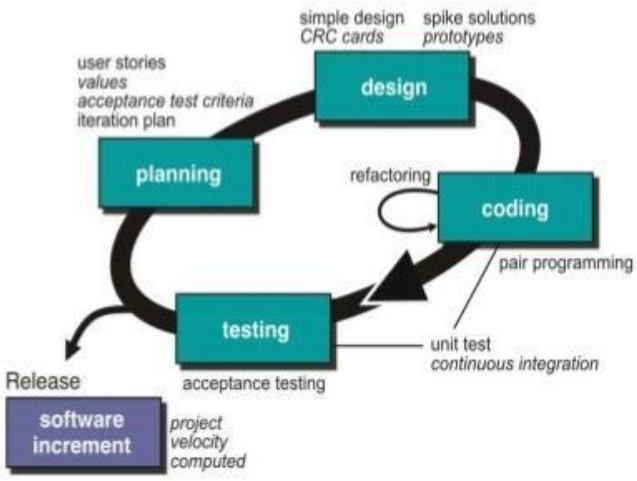
Express.js juga mendukung pengembangan aplikasi dengan pendekatan *middleware*, yang memungkinkan pengembang untuk menambahkan berbagai fungsi di antara permintaan masuk dan respon keluar. Hal ini memungkinkan implementasi otentikasi, otorisasi, pemantauan permintaan, dan banyak fungsi lainnya dengan mudah. Dengan bantuan *Express.js*, *developer* dapat membangun aplikasi *Website* yang responsif dan handal dengan cepat. *Framework* ini telah menjadi salah satu pilihan utama dalam pengembangan aplikasi *Website* menggunakan *Node.js*, dan terus menjadi favorit bagi pengembang yang menghargai produktivitas dan kemudahan dalam membuat aplikasi *Website* yang kuat [9].

2.2.7 Database *MySQL*

MySQL adalah sistem manajemen basis data (DBMS) relasional yang sangat populer dan sering digunakan dalam pengembangan perangkat lunak. Dibangun berdasarkan model relasional, *MySQL* menyimpan data dalam tabel yang terkait satu sama lain melalui kunci-kunci relasional. Sebagai bagian dari keluarga perangkat lunak *open-source*, *MySQL* menyediakan keandalan, kinerja, dan skalabilitas yang tinggi [10]. Kelebihan *MySQL* termasuk kemampuan untuk menangani volume data yang besar, dukungan transaksi ACID (*Atomicity, Consistency, Isolation, Durability*), dan kompatibilitas lintas *platform*. *MySQL* mendukung bahasa *SQL* (*Structured Query Language*) untuk mengelola dan memanipulasi data. Pengguna dapat melakukan operasi seperti menyimpan, mengambil, memperbarui, dan menghapus data dengan mudah menggunakan perintah *SQL*. Database *MySQL* digunakan dalam berbagai jenis aplikasi, termasuk situs *Website*, sistem manajemen konten, dan aplikasi bisnis. Dengan komunitas pengguna yang besar dan dukungan yang kuat dari komunitas open-source, *MySQL* tetap menjadi pilihan utama dalam pengembangan dan manajemen basis data relasional [28].

2.2.8 Metode *Extreme programming* (XP)

Metode *Extreme programming* (XP) adalah salah satu pendekatan dalam pengembangan perangkat lunak yang termasuk dalam kerangka kerja *Agile*. *Agile* adalah filosofi pengembangan perangkat lunak yang mendorong adaptabilitas dan responsivitas dalam menghadapi perubahan kebutuhan pelanggan. XP menjadi salah satu metode yang mengedepankan praktikpraktik teknis dan kerjasama tim [29]. XP mengorganisasi aktivitas pengembangan perangkat lunak menjadi empat tahap utama: perencanaan, perancangan, pengkodean, dan pengujian. Setiap tahap ini dilakukan secara berulang dalam iterasi pendek, yang disebut sebagai "siklus pengembangan." XP menerapkan beberapa praktik kunci, seperti "*pair Programming*" (pengembangan berpasangan) dan "*test-driven development*" (pengembangan berdasarkan pengujian), untuk memastikan kualitas perangkat lunak yang tinggi [30].



Gambar 2. 1 Tahap *Extreme programming* [30]

1. Perencanaan (*Planning*)

   Tahap perencanaan dimulai dengan analisis kebutuhan sistem. Dalam XP, perencanaan sering kali bersifat iteratif dan adaptif. Tim pengembangan dan pemangku kepentingan (stakeholders) bekerja bersama untuk mengidentifikasi dan merinci kebutuhan perangkat lunak. Perencanaan XP

cenderung lebih fleksibel daripada metode tradisional, dan rencana awal dapat berubah seiring berjalannya waktu [30].

2. Perancangan (*Design*)

Pada tahap perancangan, dilakukan pemodelan berdasarkan hasil analisis kebutuhan. Dalam XP, perancangan lebih bersifat sederhana dan berorientasi pada kebutuhan studi kasus. Tim pengembang berfokus pada merancang solusi yang sederhana, efisien, dan mudah diubah. Desain perangkat lunak dalam XP sering kali dilakukan secara bersama-sama dan berkolaborasi [30].

3. Pengkodean (*Coding*)

Pada tahap pengkodean, dimulai dengan mengimplementasikan perangkat lunak sesuai dengan desain yang telah dibuat. Dalam XP, praktikpraktik seperti "*pair Programming*" (program berpasangan) dan "*continuous integration*" (integrasi berkelanjutan) sangat penting. Pair *Programming* memungkinkan dua pengembang bekerja bersama-sama pada kode, yang meningkatkan kualitas dan transfer pen*get*ahuan. Continuous integration memastikan bahwa perubahan kode secara teratur diintegrasikan ke dalam versi utama perangkat lunak, sehingga masalah dapat dideteksi lebih awal [30].

4. Pengujian (*Testing*)

Tahap pengujian melibatkan pengujian *system* yang telah dikembangkan. Tujuannya adalah memastikan kesesuaian sistem yang telah dibangun dengan kebutuhan dan permintaan pengguna. Jika terdapat masalah atau ketidakcocokan dalam *system*, perbaikan akan dilakukan hingga sistem tersebut sesuai dengan permintaan pelanggan. *Blackbox* digunakan sebagai salah satu metodologi untuk mengukur sejauh mana sistem telah sesuai dengan kebutuhan pengguna [30].

2.2.9 *POSTMAN*

POSTMAN adalah sebuah aplikasi yang digunakan untuk menguji, mengelola, dan mengembangkan *API* (Application *Programming* Interface).

Aplikasi ini memiliki antarmuka pengguna yang intuitif dan memungkinkan pengguna untuk membuat, mengirim, dan mengelola permintaan *HTTP* seperti *GET*, *POST*, *PUT*, *DELETE*, dan lainnya kepada server yang menyediakan *API*. Dengan *POSTMAN*, pengguna dapat dengan mudah mengatur parameter permintaan, mengirim permintaan ke *Endpoint API*, dan melihat respon yang diterima dari server [31]. Aplikasi ini juga menyediakan fitur-fitur seperti autentikasi, pengelolaan koleksi permintaan, pengujian otomatis, dan dokumentasi *API*. *POSTMAN* membantu pengembang dalam pengujian dan debugging *API*, serta memudahkan penggunaan dan integrasi *API* dalam pengembangan perangkat lunak. Dengan fitur-fitur yang disediakan, *POSTMAN* menjadi alat yang sangat berguna dalam pengembangan dan manajemen *API* [32].

2.2.10 *Blackbox Testing*

Blackbox Testing adalah salah satu metode pengujian perangkat lunak yang fokus pada pengujian dari sudut pandang pengguna atau pengujian "kotak hitam". Dalam metode ini, pengujian dilakukan tanpa memperhatikan struktur internal kode program atau komponen perangkat lunak yang sedang diuji. Sebagai gantinya, pengujian berfokus pada masukan (input) yang diberikan dan keluaran (*output*) yang dihasilkan oleh perangkat lunak [33]. Tujuan utama dari *Blackbox Testing* adalah untuk memastikan bahwa perangkat lunak berperilaku sesuai dengan spesifikasi dan memenuhi kebutuhan pengguna. Selama *Blackbox Testing*, pengujian dilakukan dengan merancang berbagai skenario uji (*test cases*) yang mencakup berbagai situasi dan kondisi yang mungkin terjadi dalam penggunaan aplikasi. Pengujian ini akan menguji apakah aplikasi berfungsi dengan benar, mampu menangani kesalahan, dan memenuhi kriteria kinerja yang telah ditetapkan [34].

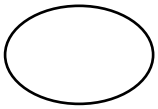
2.2.11 *Usecase* Diagram

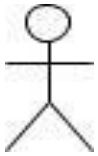
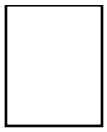
Usecase Diagram merupakan tipe diagram yang ada dalam *Unified Modelling Language* (UML) yang dirancang untuk menunjukkan interaksi antara sistem dan pengguna.Dalam *Usecase* Diagram, elemen utama yang digambarkan adalah "*Use Case*" atau kasus penggunaan. *Usecase* mewakili

fungsionalitas atau aktivitas spesifik yang dapat dilakukan oleh pengguna atau aktor dalam interaksi dengan sistem. Selain *Use Case*, diagram ini juga menyoroti aktor-aktor yang terlibat dalam interaksi tersebut.Aktors dalam *Usecase* Diagram adalah entitas luar yang berinteraksi dengan sistem. Mereka dapat berupa pengguna, sistem eksternal, atau entitas lain yang terlibat dalam menggunakan fungsionalitas sistem. Garis yang menghubungkan aktor dengan *Usecase* menunjukkan hubungan antara mereka [35].

Usecase Diagram memberikan pandangan tingkat tinggi tentang fungsionalitas sistem dan cara aktor-aktor berinteraksi dengan sistem tersebut. Hal ini membantu dalam pemahaman awal mengenai kebutuhan pengguna, fungsionalitas sistem, dan skenario interaksi yang mungkin terjadi.Dengan menggunakan *Usecase* Diagram, pengembang perangkat lunak dapat merancang sistem yang lebih sesuai dengan kebutuhan pengguna dan memastikan bahwa semua aspek fungsionalitas yang penting telah dipertimbangkan dalam perancangan. Diagram ini juga memfasilitasi komunikasi antara tim pengembang dan pemangku kepentingan dalam memahami kasus penggunaan yang harus diakomodasi oleh sistem [36]. Nama dan simbol beserta dengan fungsinya terdapat pada tabel berikut ini :

Tabel 3. 1 Tabel *Usecase* Diagram[35]

No.	**Gambar**	**Nama**	**Deskripsi**
1.		Use case	merepresentasikan fungsi atau tindakan yang dapat dilakukan oleh sistem.
No.	**Gambar**	**Nama**	**Deskripsi**
2.		Asosiasi	Asosiasi berguna sebagai menggambarkan hubungan antara aktor dan use case.

3.		Actors	merepresentasikan pengguna atau entitas eksternal lain yang berinteraksi dengan sistem.
4		System	Menampilkan Sistem yang sedang sedang berjalan
5.	<<include>> 	Include	*Relationship* antara *use case* yang memasukkan fungsionalitas dari *use case* lain.
6.	<<extend>> 	Extend	Hubungan antara *use case* yang menambahkan fungsionalitas ke *use case lain.*

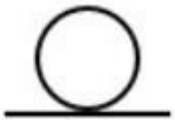
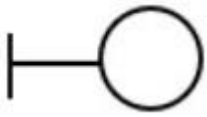
2.2.12 *Sequence* Diagram

Sequence Diagram adalah salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan interaksi antar objek dalam suatu sistem atau proses secara kronologis. Diagram ini memberikan pandangan visual tentang bagaimana objek-objek saling berkomunikasi dan berinteraksi satu sama lain dalam menjalankan suatu skenario atau urutan tindakan. Dalam *Sequence* Diagram, objek direpresentasikan sebagai kotak dengan nama objek di dalamnya, dan pesanpesan yang dikirim antar objek direpresentasikan sebagai garis-garis panah yang menghubungkan objek-objek tersebut. Garis-garis panah ini menunjukkan urutan pesan-pesan yang dikirim, serta arah aliran komunikasi antar objek [37].

Sequence Diagram membantu dalam memodelkan bagaimana objekobjek dalam sistem saling berinteraksi dan berkomunikasi satu sama lain dalam menjalankan suatu skenario atau proses tertentu. Ini memungkinkan

pengembang atau analis sistem untuk memahami bagaimana informasi atau pesan dikirim dan diterima antar objek, serta urutan eksekusi tindakan yang terjadi dalam suatu skenario. *Sequence* Diagram dapat menjadi alat yang efektif dalam analisis, perancangan, dan dokumentasi interaksi antar objek dalam suatu sistem, baik dalam konteks pengembangan perangkat lunak maupun pemodelan proses bisnis. Diagram ini membantu tim pengembang atau pemangku kepentingan dalam memahami secara visual bagaimana objek-objek saling berinteraksi dan berkomunikasi dalam menjalankan suatu proses atau skenario tertentu.[36]. Nama dan simbol beserta dengan fungsinya terdapat pada tabel berikut ini:

Tabel 3. 2 Tabel *Sequence* Diagram[37]

Simbol	**Nama**	**Keterangan**
	*Actor*	Menggambar objek yang berinteraksi dengan sebuah sistem
	Entity *Class*	Menggambarkan hubungan yang akan dilakukan
	*Boundary Class*	Menggambarkan sebuah gambaran dari sebuah foem
	*Control Class*	Menggambarkan Penghubung antara table dan boundary
Simbol	**Nama**	**Keterangan**

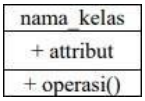
	*A focus of Control and A Life Line*	Menggambarkan dimulainya dan berakhirnya sebuah massage
	*A massage*	*Menggambarkan Pengiriman Pesan*

2.2.13 *Class* Diagram

Class Diagram adalah salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan struktur statis suatu sistem atau aplikasi, dengan fokus pada kelas-kelas dan hubungan antara kelaskelas tersebut. Dalam *Class* Diagram, kelas direpresentasikan oleh kotak dengan tiga bagian: bagian atas berisi nama kelas, bagian tengah berisi atribut atau properti kelas, dan bagian bawah berisi metode atau operasi yang dapat dilakukan oleh kelas tersebut. Hubungan antar kelas ditunjukkan dengan garis yang menghubungkan kelas-kelas, dan tanda panah menunjukkan arah hubungan [37].*Class* Diagram membantu dalam memahami struktur dan hierarki kelas dalam suatu sistem, serta hubungan antar kelas tersebut. Ini berguna dalam fase perancangan perangkat lunak dan membantu tim pengembang untuk memvisualisasikan struktur sistem secara jelas dan sistematis. *Class* Diagram juga mendukung komunikasi antara anggota tim pengembangan dan pemangku kepentingan untuk memahami organisasi struktural suatu aplikasi atau sistem [36]. Nama dan simbol beserta dengan fungsinya terdapat pada tabel berikut ini:

Tabel 3. 3 Tabel *Class* Diagram[37]

No.	Gambar	Nama	Deskripsi

1.		Kelas	Entitas yang mewakili objek dalam suatu sistem atau proses
2.	 **nama_interface**	*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
3.		Asosiasi	Hubungan antar kelas yang memiliki makna umum, sering kali disertai dengan *multiplicity*
4.		Asosiasi berarah	Hubungan antar kelas yang bermakna generalisasi spesialisasi (umum khusus)
5.		Generalisasi	Hubungan antar kelas asosiasi generalisasispesialisasi (umum khusus)
6.		*Dependecy*	Ketergantungan antar kelas