BAB II
TINJAUAN PUSTAKA

2.1 Teori Dasar

2.1.1 Rancang Bangun

Rancang bangun adalah istilah yang digunakan untuk mendesain atau membuat suatu objek dari awal hingga selesai. Proses ini melibatkan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan berbagai elemen yang terpisah menjadi satu kesatuan yang utuh dan berfungsi. Oleh karena itu, rancang bangun dapat diartikan sebagai kegiatan menerjemahkan hasil analisis menjadi bentuk perangkat lunak, lalu menciptakan atau memperbaiki sistem yang sudah ada (Wulandari, S., et al. 2021).

2.1.2 Sistem

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling terkait dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau mencapai tujuan tertentu. Sistem juga dapat diartikan sebagai serangkaian data atau komponen yang berinteraksi untuk mencapai suatu tujuan. Sistem merupakan kesatuan data yang terhubung dan diorganisir secara prosedural (Achmad Fikri Sallaby, Indra Kanedi, 2020).

2.1.3 Informasi

Informasi adalah data yang telah dikelola dan diproses untuk memberikan makna dan memperbaiki proses pengambilan keputusan. Informasi juga dapat diartikan sebagai data yang telah diproses sedemikian rupa sehingga memberikan makna dan membantu dalam pengambilan keputusan yang lebih baik. Berdasarkan uraian di atas, dapat disimpulkan bahwa informasi adalah sekumpulan data yang diproses sehingga menghasilkan informasi yang bermanfaat dan dapat diterima dengan baik oleh penerima informasi (Achmad Fikri Sallaby, Indra Kanedi, 2020).

2.1.4 Sistem Informasi

Sistem informasi adalah suatu sistem dalam sebuah organisasi yang memenuhi kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat

manajerial, dan kegiatan strategis organisasi, serta menyediakan laporan-laporan yang dibutuhkan pihak luar tertentu. Sistem informasi terdiri dari kumpulan individu yang bekerja sama untuk mencapai tujuan tertentu. Berdasarkan penjelasan tersebut, dapat disimpulkan bahwa sistem informasi menyediakan informasi dalam bentuk laporan-laporan kepada pihak-pihak yang memerlukan informasi yang bersifat manajerial (Dewi, R. K., Ardian, Q. J., Sulistiani, H., & Isnaini, F., 2021).

Berikut adalah beberapa contoh sistem informasi yang digunakan dalam berbagai bidang (Sahusilawane, W., dkk., 2023):

1. **Sistem Informasi Manajemen (MIS)**

   Sistem ini digunakan untuk mengelola informasi yang dibutuhkan manajemen untuk membuat keputusan operasional, taktis, dan strategis. Contoh: Sistem ERP (*Enterprise Resource Planning*) yang mengintegrasikan berbagai fungsi bisnis seperti keuangan, sumber daya manusia, dan inventaris dalam satu platform.

2. **Sistem Informasi Akuntansi (AIS)**

   Sistem ini mengumpulkan, menyimpan, dan memproses data akuntansi untuk menghasilkan laporan keuangan yang berguna bagi manajemen dan pihak eksternal. Contoh: QuickBooks atau SAP FinancialAccounting.

3. **Sistem Informasi Sumber Daya Manusia (HRIS)**

   Sistem ini mengelola data karyawan, penggajian, rekrutmen, pelatihan, dan evaluasi kinerja. Contoh: Workday, SAP SuccessFactors, atau BambooHR.

4. **Sistem Informasi Geografis (GIS)**

   Sistem ini mengumpulkan, menganalisis, dan menampilkan data yang terkait dengan lokasi geografis. Contoh: ArcGIS, Google Earth, atau QGIS.

5. **Sistem Informasi Pemasaran (MKIS)**

   Sistem ini mengelola data pemasaran dan pelanggan untuk mendukung keputusan pemasaran. Contoh: Salesforce, HubSpot, atau Marketo.

6. **Sistem Pendukung Keputusan (DSS)**

   Sistem ini membantu manajemen dalam membuat keputusan dengan menyediakan informasi analitis yang relevan. Contoh: IBM Cognos

Analytics, Microsoft Power BI, atau Tableau.

7. **Sistem Informasi Eksekutif (EIS)**

   Sistem ini menyediakan informasi penting kepada eksekutif untuk membantu dalam pengambilan keputusan strategis. Contoh: SAP BusinessObjects, Oracle Hyperion, atau IBM Watson Analytics.

8. **Sistem Informasi Medis (MIS)**

   Sistem ini mengelola data medis pasien dan membantu dalam administrasi rumah sakit. Contoh: Electronic Health Records (EHR) seperti Epic, Cerner, atau Allscripts.

9. **Sistem Manajemen Rantai Pasokan (SCM)**

   Sistem ini mengelola aliran barang, informasi, dan uang di seluruh rantai pasokan dari pemasok hingga pelanggan akhir. Contoh: SAP SCM, Oracle SCM, atau JDA Software.

10. **Sistem Manajemen Hubungan Pelanggan (CRM)**

    Sistem ini mengelola interaksi dengan pelanggan saat ini dan potensial, dengan tujuan meningkatkan hubungan dan mempertahankan pelanggan. Contoh: Salesforce, Zoho CRM, atau Microsoft Dynamics CRM.

2.1.5 Manajemen Keuangan

Manajemen keuangan adalah suatu disiplin yang berfokus pada perencanaan, pengelolaan, dan pengendalian sumber daya keuangan suatu organisasi atau individu. Dalam konteks UMKM, manajemen keuangan yang efektif sangat penting untuk memastikan kelangsungan usaha dan pertumbuhan jangka panjang. Teori ini mencakup beberapa konsep dasar yang menjadi landasan dalam pengelolaan keuangan, yaitu pencatatan transaksi, pengelolaan kas, analisis laporan keuangan, dan pelaporan keuangan. Masing-masing konsep ini saling terkait dan membentuk kerangka kerja yang membantu UMKM dalam mencapai stabilitas dan efisiensi keuangan.

1. **Pencatatan Transaksi:**

   Pencatatan transaksi merupakan langkah awal dalam manajemen keuangan, di mana setiap kegiatan ekonomi yang melibatkan uang dicatat secara

sistematis dan akurat. Bagi UMKM, pencatatan transaksi yang rapi dan terorganisir sangat penting karena membantu dalam memantau arus kas masuk dan keluar, serta memberikan gambaran yang jelas tentang kondisi keuangan usaha. Kesalahan dalam pencatatan transaksi dapat mengakibatkan laporan keuangan yang tidak akurat, yang pada gilirannya dapat menyebabkan pengambilan keputusan yang salah. Aplikasi keuangan yang dirancang khusus untuk UMKM dapat membantu menyederhanakan proses pencatatan ini, dengan menyediakan antarmuka yang mudah digunakan dan otomatisasi yang meminimalkan kesalahan manusia.

2. **Pengelolaan Transaksi:**

   Pengelolaan transaksi melibatkan pemantauan dan pengaturan atas semua aktivitas keuangan yang terjadi dalam bisnis, termasuk pemasukan dan pengeluaran. Bagi UMKM, pemasukan dapat berasal dari berbagai sumber, seperti penjualan produk atau jasa, investasi, dan penerimaan lainnya. Sementara itu, pengeluaran mencakup biaya operasional, pembelian bahan baku, gaji karyawan, dan pengeluaran lain yang diperlukan untuk menjalankan bisnis. Penting bagi UMKM untuk mengelola pemasukan dan pengeluaran ini dengan cermat agar dapat menjaga aliran kas yang sehat dan memastikan bahwa saldo keuangan selalu memadai untuk memenuhi kewajiban finansial.

3. **Pelaporan Keuangan:**

   Pelaporan keuangan adalah proses penyusunan dan penyajian informasi keuangan kepada pemangku kepentingan, seperti investor, kreditor, dan otoritas pajak. Bagi UMKM, pelaporan yang akurat dan tepat waktu sangat penting, baik untuk tujuan internal, seperti pengambilan keputusan strategis, maupun untuk memenuhi kewajiban eksternal, seperti pelaporan pajak. Aplikasi keuangan yang dirancang untuk UMKM dapat menyederhanakan proses ini dengan menyediakan template laporan yang sesuai dengan standar akuntansi, serta kemampuan untuk menghasilkan laporan dalam format yang dapat dicetak atau dibagikan secara digital.

2.1.6 Software Development Life Cycle (SDLC)

SDLC adalah suatu pendekatan klasik yang digunakan dalam pengembangan, pemeliharaan, dan penggunaan sistem informasi. Metode ini melibatkan serangkaian tahapan yang digunakan dalam proses pengembangan sistem. SDLC dapat dianggap sebagai kerangka kerja formal tertua dalam metodologi untuk membangun sistem informasi. Konsep utama dari SDLC adalah untuk mengembangkan sistem informasi secara terstruktur dan terorganisir, dengan memperhatikan tahapan siklus hidup dari konsepsi hingga implementasi (Rizki, W., Rayuwati, R., & Gemasih, H., 2022).

SDLC memiliki tahapan-tahapan atau serangkaian langkah atau fase yang diikuti dalam pengembangan sistem informasi. Meskipun berbagai model SDLC mungkin memiliki varian dalam nama dan jumlah tahapan, umumnya SDLC terdiri dari tahapan- tahapan berikut (Jamal, S., & Kusnadi, K., 2022):

1. **Perencanaan (Planning)**

Tahap di mana rencana proyek disusun. Ini termasuk mengidentifikasi tujuan proyek, menyusun jadwal, menentukan anggaran, dan menetapkan sumber daya yang dibutuhkan.

2. **Analisis (Analysis)**

Tahap di mana kebutuhan bisnis dan fungsional sistem diidentifikasi dan didokumentasikan secara rinci. Ini melibatkan wawancara pemangku kepentingan, pemetaan proses bisnis, dan menghasilkan dokumen kebutuhan.

3. **Desain (Design)**

Tahap di mana desain teknis sistem direncanakan. Ini mencakup merancang arsitektur sistem, mengembangkan spesifikasi teknis, serta merancang antarmuka pengguna dan basis data.

4. **Implementasi (Implementation)**

Tahap di mana sistem aktual dikembangkan berdasarkan desain yang telah disetujui. Ini melibatkan menulis kode, mengintegrasikan komponen, dan melakukan pengujian unit.

5. Pengujian (Testing)

Tahap di mana sistem diuji untuk memastikan bahwa ia memenuhi spesifikasi dan bekerja sesuai yang diharapkan. Ini melibatkan pengujian fungsionalitas, kinerja, keamanan, dan keandalan sistem.

6. Pemeliharaan (Maintenance)

Tahap yang terjadi setelah penerapan di mana sistem dipelihara dan diperbaiki sesuai kebutuhan. Ini mencakup perbaikan bug, peningkatan fungsionalitas, dan penyesuaian dengan perubahan kebutuhan bisnis. Tahapan-tahapan ini sering kali diikuti secara berurutan, tetapi dalam praktiknya, beberapa tahapan mungkin tumpang tindih atau berulang tergantung pada metode pengembangan yang digunakan dan kompleksitas proyek.

2.1.7 Analisis dan Perancangan

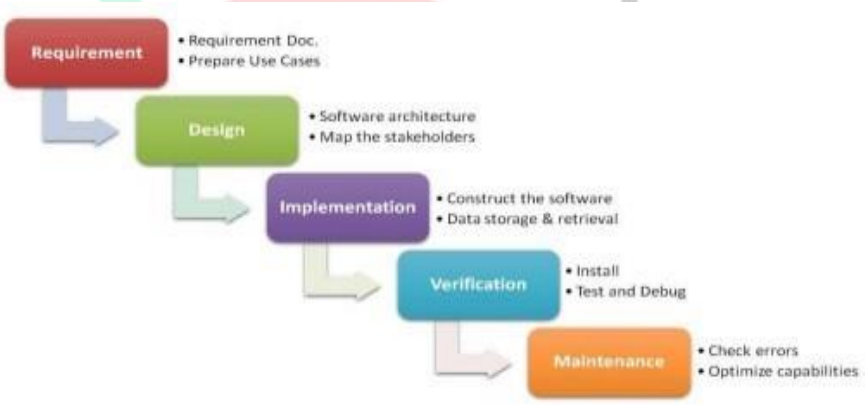
Analisis dan perancangan merupakan istilah yang menggambarkan tahap-tahap awal dalam pengembangan sistem secara keseluruhan. Proses analisis dan perancangan penting untuk pemahaman sistem yang akan dibangun dan merancang sistem informasi yang berbasis komputer. Tujuan utama dari proses ini adalah menciptakan solusi sistem baru yang efektif untuk menangani masalah yang dihadapi oleh perusahaan, dengan mengeksplorasi dan mengembangkan opsi sistem yang optimal (Indyah Hartami, 2020).

OOAD (Object-Oriented Analysis and Design) adalah salah satu pendekatan dalam analisis dan perancangan sistem. OOAD (Object-Oriented Analysis and Design) adalah pendekatan yang lebih modern dalam analisis dan perancangan sistem yang berfokus pada penggunaan objek dan kelas dalam sistem. Metode ini mengidentifikasi objek-objek dalam sistem, hubungan antara objek, dan perilaku objek melalui diagram UML (Unified Modeling Language). OOAD berorientasi pada objek, menggunakan konsep seperti

pewarisan, enkapsulasi, dan polimorfisme, menghasilkan desain yang modular dan fleksibel (Apandi, A., 2023).

2.1.8 Waterfall

Metode Waterfall adalah pendekatan dalam pengembangan perangkat lunak yang mengikuti urutan langkah-langkah secara berurutan atau sekuensial. Dalam metode ini, setiap tahap pengembangan, seperti analisis, desain, implementasi, pengujian, dan pemeliharaan, dilakukan secara berturut-turut. Artinya, tahapan berikutnya dimulai setelah tahapan sebelumnya selesai sepenuhnya. Analoginya seperti air terjun, di mana air mengalir ke bawah secara berurutan dan tidak ada mundur ke tahapan sebelumnya. Metode Waterfall menekankan perencanaan yang cermat di awal proyek dan setiap tahap memiliki deliverables (hasil kerja) yang jelas sebelum memulai tahap berikutnya (Badrul, M., 2021).



Metode Waterfall cocok untuk proyek-proyek yang memiliki persyaratan yang stabil dan terdefinisi dengan jelas di awal proyek, serta memiliki sedikit perubahan yang diharapkan selama siklus pengembangan. Berikut adalah situasi-situasi di mana Metode Waterfall cocok digunakan (Pricillia, T., 2021):

1. Proyek dengan persyaratan stabil dan jelas. Ketika persyaratan proyek telah ditetapkan dengan jelas dan tidak banyak berubah selama siklus pengembangan, Metode Waterfall dapat menjadi pilihan yang baik karena fokus pada perencanaan awal dan eksekusi bertahap.

2. Proyek dengan batas waktu yang tidak singkat. Metode Waterfall memungkinkan perencanaan yang cermat dari awal, yang dapat membantu dalam menetapkan jadwal yang ketat dan memastikan pengiriman tepat waktu.
3. Proyek dengan kebutuhan risiko yang rendah. Metode Waterfall cocok untuk proyek-proyek di mana risiko perubahan atau kegagalan sistem rendah, karena mengharuskan semua persyaratan dan desain ditetapkan di awal.

Meskipun cocok untuk situasi-situasi tersebut, penting untuk diingat bahwa setiap pendekatan pengembangan memiliki kelebihan dan kelemahan tersendiri. Oleh karena itu, penting untuk mempertimbangkan faktor-faktor tersebut dan memilih metodologi yang paling sesuai dengan karakteristik dan kebutuhan proyek tertentu.

2.1.8 *Unified Modelling Language* (UML)

UML adalah singkatan dari Unified Modeling Language yang merupakan bahasa pemodelan yang dikembangkan oleh Object Management Group (OMG). UML digunakan untuk menggambarkan dan merepresentasikan berbagai aspek dalam sebuah sistem, seperti struktur kelas, diagram use case, activity diagram, dan lain-lain. Bahasa pemodelan visual ini digunakan oleh pengembang perangkat lunak untuk membantu dalam proses analisis, desain, dan dokumentasi sistem. UML bisa dianggap sebagai bahasa pemodelan visual yang digunakan untuk merepresentasikan konsep, struktur, dan perilaku sistem yang kompleks. Sebagai standar industri, UML dapat digunakan untuk menggambarkan berbagai jenis sistem, mulai dari sistem perangkat lunak hingga sistem bisnis. Fleksibilitas dan kesederhanaan UML memungkinkannya digunakan dalam berbagai konteks dan dapat disesuaikan dengan kebutuhan pengguna (Fitrani, L. D., & Puspitaningrum, A. C., 2023).

Selain itu, UML juga dapat digunakan oleh berbagai pihak terkait dalam pengembangan sistem, termasuk pengembang, analis, manajer proyek, dan pengguna akhir. Terdapat berbagai jenis diagram yang terdapat dalam UML, seperti berikut:

1. ***Use Case Diagram***

Use Case Diagram adalah salah satu jenis diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan interaksi antara sistem dan aktor-aktor eksternal yang terlibat dalam sistem (Hartiwi, Y., & Nurhayati, N., 2022). Diagram ini memberikan gambaran visual tentang fungsionalitas sistem dari sudut pandang pengguna atau aktor, dan membantu dalam memahami kebutuhan pengguna serta interaksi antara sistem dan lingkungan eksternalnya. Simbol dari komponen-komponen utama dalam Use Case Diagram dapat dilihat dalam Tabel 2.1.

Tabel 2. 1 Simbol Use Case Diagram

Simbol	Nama	Keterangan
	*Actor*	Menentukan kumpulan peran yang pengguna gunakan saat berinteraksi dengan *use case*.
	Dependency	Hubungan di mana perubahan pada elemen independen akan memengaruhi elemen yang bergantung padanya, yang bersifat tidak independen.
	*Generalization*	Hubungan di mana objek turunan (*descendent*) mewarisi perilaku dan struktur dari objek induk (*ancestor*)
	Include	Menspesifikasikan bahwa *use case* berasal dari sumber secara eksplisit.
	Extend	Menentukan bahwa target *use case* memperluas perilaku dari sumber.
	Association	Penghubung antara satu objek dengan objek lainnya.
	*System*	Menggambarkan paket yang menampilkan sistem.
	*Use Case*	Deskripsi dari aktivitas-aktivitas yang ditampilkan sistem yang dapat dilihat oleh aktor.

	*Collaboration*	Interaksi aturan-aturan dan elemen- elemen yang terhubung untuk aktivitas yang lebih besar dari jumlah dan sinergi elemen-elemen.
	*Notes*	Elemen fisik yang nyata saat sistem berjalan.

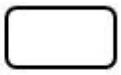
(Sumber: Jacobson, L., & Booch, J. R. G., 2021)

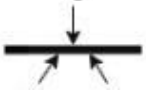
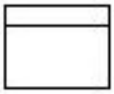
Activity Diagram membantu dalam memodelkan aliran kerja secara visual dan memahami bagaimana aktivitas-aktivitas saling terkait dan berinteraksi satu sama lain dalam suatu proses. Diagram ini dapat digunakan untuk mengidentifikasi masalah dalam aliran kerja, mengoptimalkan proses, dan memvalidasi desain sistem sebelum implementasi.

2. Activity Diagram

Activity Diagram adalah salah satu jenis diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan aliran kerja atau aktivitas dalam suatu proses atau fungsi (Hartiwi, Y., & Nurhayati, N., 2022). Diagram ini membantu dalam memodelkan aktivitas, keputusan, dan aliran kontrol antara aktivitas-aktivitas tersebut. Simbol dari komponen-komponen utama dalam Activity Diagram dapat dilihat dalam Tabel 2.2.

Tabel 2. 2 Simbol Activity Diagram

Simbol	Nama	Keterangan
	*Start*	Status awal dari sebuah aktivitas sistem digambarkan oleh sebuah diagram aktivitas sebagai status permulaan.
	*Activity*	Aktivitas yang dilakukan oleh sistem biasanya dimulai dengan kata kerja.

	*Decision*	Asosiasi atau percabangan adalah kondisi di mana terdapat lebih darisatu pilihan aktivitas.
	Join	Kondisi di mana lebih dari satu aktivitas digabungkan menjadi satu.
	*End*	Status akhir yang dicapai oleh sistem, setiap sistem harus memiliki satu status akhir.
	*Fork*	Menunjukkan kegiatan yang dilakukan secara paralel atau bersamaan.
	*Swimlane*	Memisahkan aktor-aktor yang bertanggung jawab terhadap aktivitas-aktivitas yang terjadi.

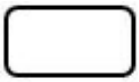
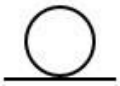
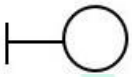
(Sumber: Jacobson, L., & Booch, J. R. G., 2021)

Activity Diagram membantu dalam memodelkan aliran kerja secara visual dan memahami bagaimana aktivitas-aktivitas saling terkait dan berinteraksi satu sama lain dalam suatu proses. Diagram ini juga dapat digunakan untuk mengidentifikasi masalah dalam aliran kerja, mengoptimalkan proses, dan memvalidasi desain sistem sebelum implementasi.

3. Sequence Diagram

Sequence Diagram adalah salah satu jenis diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan interaksi antara objek dalam suatu sistem dalam urutan waktu tertentu (Syarif, M., & Nugraha, W., 2020). Diagram ini menunjukkan pesan-pesan yang dikirim antara objek-objek dan urutan waktu mereka dalam berinteraksi satu sama lain. Simbol dari komponen-komponen utama dalam Sequence Diagram dapat dilihat dalam Tabel 2.3.

Tabel 2. 3 Simbol Sequence Diagram

Simbol	Nama	Keterangan
	*Actor*	Menggambarkan siapa yang sedang berinteraksi dengan sistem.
	*Object*	Objek yang berinteraksi atau mengirimkan pesan juga dapat berupa *instance* dari kelas dengan nama objek di dalamnya yang diawali dengan titik dua.
	*Entitiy Class*	Menggambarkan hubungan yang akan dilakukan.
	*Boundary Class*	Menggambarkan sebuah gambaran dari *foem.*
	*Control Class*	Menggambarkan penghubung antara *boundary* dan tabel.
	*Focus of Control & Life Line*	Menunjukkan tempat *start* dan *end* suatu objek yang berinteraksi dengan pesan.
	*Message*	Menggambarkan komunikasi atau pengiriman pesan antar objek.

(Sumber: Jacobson, L., & Booch, J. R. G., 2021)

4. Class Diagram

Class Diagram adalah salah satu jenis diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan struktur statis dari sistem atau aplikasi yang sedang dikembangkan (Hartiwi, Y., & Nurhayati, N., 2022). Diagram ini memodelkan kelas-kelas, atribut, metode, dan hubungan antara kelas-kelas tersebut. Simbol dari komponen- komponen utama dalam Class Diagram dapat dilihat dalam Tabel 2.4.

Tabel 2. 4 Simbol Class Diagram

Simbol	Nama	Keterangan
	Generalization	Hubungan di mana objek turunan (*descendent*) mewarisi perilaku dan struktur dari objek induk (*ancestor*).
	*n-ary Association*	Simbol untuk asosiasi dengan lebih dari 2 objek.
	*Class*	Kumpulan dari objek-objek yang memiliki atribut dan operasi yang sama.
	*Collaboration*	Interaksi aturan-aturan dan elemen-elemen yang terhubung untuk aktivitas yang lebih besar dari jumlah dan sinergi elemen-elemen.
	Realization	Operasi yang secara nyata dilakukan oleh sutau objek.
	Dependency	Hubungan di mana perubahan pada elemen independen akan memengaruhi elemen yang bergantung padanya, yang bersifat tidak independen.
	Association	Penghubung antara satu objek dengan objek lainnya.

(Sumber: Jacobson, L., & Booch, J. R. G., 2021)

Class Diagram membantu dalam memodelkan struktur statis dari sistem, termasuk kelas-kelas yang ada, atribut dan metode yang dimiliki oleh kelas-kelas tersebut, serta hubungan antara kelas-kelas. Diagram ini memungkinkan

pengembang untuk memahami struktur logika dari sistem, mengidentifikasi hubungan antar kelas, dan memvalidasi desain sistem sebelum implementasi. Class Diagram juga merupakan alat komunikasi yang efektif antara pengembang, analis, dan pemangku kepentingan lainnya dalam proyek pengembangan perangkat lunak.

2.1.9 *Website*

Website adalah kumpulan halaman yang terdiri dari informasi dalam berbagai bentuk seperti teks, gambar, video, audio, dan animasi, yang tersedia melalui koneksi internet. Dengan kata lain, *website* adalah halaman-halaman yang mengandung informasi yang dapat diakses melalui *browser* dan memberikan manfaat bagi pengaksesnya. Berdasarkan sifatnya, *website* dapat dibedakan menjadi dua jenis (Muhammad Ibnu Sa`ad, 2020) :

1. **Web Statis**

   Web Statis adalah jenis *website* yang berisi konten yang tidak berubah-ubah. Artinya, konten dari halaman-halaman tersebut tidak dapat diubah dengan cepat dan mudah. Hal ini disebabkan oleh teknologi yang digunakan untuk membuat dokumen web tersebut, yang tidak memungkinkan untuk melakukan perubahan isi atau data dengan mudah. Teknologi yang digunakan untuk web statis biasanya adalah jenis *client-side scripting* seperti HTML, *Cascading Style Sheet* (CSS). Perubahan isi atau data pada halaman web statis hanya dapat dilakukan dengan mengubah langsung isinya pada file mentah tersebut atau mengubah *script*.

2. **Web Dinamis**

   Web Dinamis adalah jenis website yang kontennya dapat diubah setiap waktu dan harus diubah dalam file atau dengan istilah bongkar script. Sebuah website yang menampilkan tampilan dinamis melalui halaman admin tanpa flash belum tentu termasuk web dinamis. Web dinamis biasanya dibuat dengan menyimpan data pada database seperti MySQL.

   Aplikasi web adalah jenis aplikasi yang menggunakan arsitektur client-server. Pada jenis arsitektur ini, sebuah program client terhubung pada sebuah server

untuk mendapatkan informasi yang dibutuhkan guna melengkapi tugas-tugas yang telah ditentukan oleh pengguna. Dalam pengembangan web, terdapat istilah Server-side dan Client-side Scripting. Server-Side Scripting adalah bahasa pemrograman web yang pengolahannya dilakukan oleh komputer server/penyedia. Setiap kali sebuah web dikunjungi, server akan mengirimkan data-data yang diminta dari database yang kemudian akan ditampilkan di web. Kode yang ada dalam *server-side* tidak bisa dilihat karena sifatnya yang rahasia untuk *client*. Contoh dari *Server-side Script* adalah PHP, ASP, ASP.Net dan Java. *Client-Side Scripting* adalah bahasa pemrograman web yang pengolahannya dilakukan oleh komputer client atau pengguna. Ketika seseorang berkunjung ke sebuah web, komputernya akan mengunduh *script* yang bersifat client-side di web tersebut. Contoh dari *Client-side Script* adalah HTML, CSS, JavaScript dan XML (Geovanny, A., 2022).

2.1.9 Database

Database atau basis data adalah kumpulan data atau informasi yang disimpan secara terstruktur dan terorganisir dalam suatu sistem komputer atau perangkat lunak tertentu. Tujuan utama penggunaan database adalah untuk menyimpan, mengelola, dan mengakses data dengan efisien dan efektif. *Database* dapat berisi berbagai jenis data yang berkaitan dengan operasi bisnis atau organisasi tertentu. Setiap data dalam *database* biasanya memiliki atribut atau kolom yang merepresentasikan karakteristik atau properti tertentu dari data tersebut (Elmasri, R., & Navathe, S. B., 2020).

Database terdiri dari sekumpulan data yang saling berhubungan secara logis dan terorganisir dengan baik. Kumpulan data dalam *database* dapat berbentuk *file*, tabel, atau arsip yang saling terhubung dan tersimpan dalam media penyimpanan elektronis. Basis data merupakan salah satu komponen utama dalam pendukung program yang terkomputerisasi. Hampir semua program aplikasi yang melibatkan pengelolaan data dapat dipastikan menggunakan basis data sebagai tempat penyimpanan datanya (Kadarsih, 2022).

Untuk memasukkan data ke dalam *database*, atau sebaliknya, mengambil data dari database, diperlukan perangkat lunak yang disebut *Database Management*

System (DBMS).

Dengan DBMS, pengguna dapat mengontrol, memelihara, dan mengakses data secara praktis dan efisien. DBMS berfungsi sebagai perantara antara pengguna dengan *database*, sehingga interaksi antara pengguna dengan *database* memerlukan command atau sintaks tertentu (Tri Rachmadi, 2020). Terdapat dua jenis command dalam DBMS, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML) (Setyawati, E., Wijoyo, H., & Soeharmoko, N., 2020).

Salah satu jenis *database* adalah *relational database*, yang menyimpan data dalam bentuk tabel yang terhubung menggunakan kunci yang memiliki nilai serupa pada kolom yang ada di masing- masing tabel. Contoh dari database jenis ini adalah MySQL, Oracle, dan SQL Server. Dalam pembuatan sebuah *relational database*, dilakukan normalisasi untuk menghilangkan kemungkinan adanya anomali yang terjadi pada saat pengelolaan data di dalam *database*. Normalisasi merupakan pendekatan dalam membangun desain logika basis data relasional yang tidak secara langsung berkaitan dengan model data, namun dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal. Bentuk-bentuk normalisasi antara lain Bentuk Tidak Normal, 1 *Normal Form* (1NF), 2 *Normal Form* (2NF), dan 3 *Normal Form* (3NF) (Setyawati, E., Wijoyo, H., & Soeharmoko, N., 2020).

2.2 Tinjauan Studi

Penulis melakukan tinjauan studi dengan tujuan untuk mendukung penyusunan penelitian ini berdasarkan penelitian-penelitian terdahulu sebagai referensi. Tinjauan studi dilakukan dengan mengidentifikasi lima penelitian sebagai referensi utama yang dipilih karena merupakan penelitian terkini dan relevan terhadap rumusan masalah yang diajukan. Berikut adalah pembahasan terkait lima penelitian tersebut:

1. Penelitian dengan judul "**Pengembangan Aplikasi Manajemen Keuangan Digital Berbasis Web untuk UMKM di Kota Yogyakarta**" yang ditulis oleh Setyawan, A., & Wicaksono, R. pada tahun 2021 dalam Jurnal Teknologi Informasi dan Bisnis. Penelitian ini berfokus pada pengembangan aplikasi manajemen keuangan berbasis web yang dirancang khusus untuk UMKM di Kota Yogyakarta. Aplikasi ini

menyediakan fitur pencatatan transaksi, pengelolaan kas, dan pelaporan keuangan yang sederhana dan mudah digunakan oleh pelaku UMKM. Tujuan dari penelitian ini adalah untuk meningkatkan efisiensi pengelolaan keuangan di kalangan UMKM. Pengujian sistem dilakukan menggunakan metode black box testing, dan hasilnya menunjukkan bahwa aplikasi ini mampu meningkatkan akurasi dan efisiensi dalam pencatatan keuangan, serta mempermudah pembuatan laporan keuangan untuk kepentingan internal maupun eksternal.

2. Penelitian dengan judul "**Aplikasi Keuangan Digital untuk UMKM: Studi Kasus di Kabupaten Bandung**" yang ditulis oleh Prasetyo, B. A., & Ramadhani, A. R. pada tahun 2020 dalam Jurnal Sistem Informasi dan Teknologi. Penelitian ini mengembangkan aplikasi keuangan digital berbasis web yang dirancang untuk membantu UMKM di Kabupaten Bandung dalam mencatat dan mengelola transaksi keuangan mereka. Aplikasi ini dilengkapi dengan fitur dashboard yang menampilkan visualisasi saldo, grafik pemasukan dan pengeluaran, serta laporan keuangan yang dapat diunduh dalam format PDF. Penggunaan aplikasi ini diharapkan dapat memberikan solusi praktis bagi UMKM dalam mengelola keuangan mereka secara lebih efisien dan akurat, serta memudahkan mereka dalam membuat keputusan keuangan berdasarkan data yang tersedia secara real-time.
3. Penelitian dengan judul "**Implementasi Aplikasi Manajemen Keuangan Digital Berbasis Web untuk Meningkatkan Daya Saing UMKM**" yang ditulis oleh Surya, H. P., & Nurhadi, M. pada tahun 2022 dalam Jurnal Manajemen dan Teknologi Informasi. Penelitian ini membahas pengembangan aplikasi manajemen keuangan berbasis web yang dirancang untuk meningkatkan daya saing UMKM di era digital. Aplikasi ini menawarkan fitur kategori transaksi yang memungkinkan pengguna untuk mengelompokkan pemasukan dan pengeluaran sesuai dengan jenisnya. Selain itu, aplikasi ini dilengkapi dengan pelacakan pemasukan dan pengeluaran, serta kemampuan untuk mencetak laporan keuangan secara otomatis dalam format PDF. Pengembangan aplikasi ini dilakukan dengan menggunakan metodologi Agile, dan hasil pengujian menunjukkan bahwa aplikasi ini mampu meningkatkan efisiensi operasional UMKM dalam mengelola keuangan mereka.
4. Penelitian dengan judul "**Rancang Bangun Aplikasi Keuangan Digital Berbasis Web untuk Pengelolaan Keuangan UMKM di Surabaya**" yang ditulis oleh Widodo,

A., & Kurniawati, L. pada tahun 2023 dalam Jurnal Rekayasa Perangkat Lunak. Penelitian ini mengembangkan aplikasi keuangan digital berbasis web dengan fokus pada UMKM di Surabaya. Aplikasi ini mencakup fitur-fitur seperti pencatatan transaksi, pengelolaan saldo, dan dashboard grafik keuangan yang menampilkan data secara harian, bulanan, dan tahunan. Sistem ini dirancang dengan metode pengembangan waterfall, dan diuji menggunakan black box testing. Hasil penelitian menunjukkan bahwa aplikasi ini mampu memberikan solusi praktis bagi UMKM dalam mengelola keuangan mereka, meningkatkan transparansi keuangan, serta memudahkan pembuatan laporan keuangan.

5. Penelitian dengan judul "**Pengembangan Aplikasi Keuangan UMKM Berbasis Web dengan Fitur User Role dan Pelaporan Keuangan Otomatis**" yang ditulis oleh Hartono, E., & Lestari, D. pada tahun 2023 dalam Jurnal Sistem dan Teknologi Informasi. Penelitian ini mengembangkan aplikasi keuangan digital untuk UMKM dengan fokus pada keamanan dan manajemen user role. Aplikasi ini memungkinkan pelaku UMKM untuk mengatur akses pengguna berdasarkan peran mereka, memastikan bahwa hanya pengguna dengan otorisasi yang tepat yang dapat mengakses data keuangan sensitif. Selain itu, aplikasi ini juga menyediakan fitur pelaporan keuangan yang dapat dicetak dalam format PDF, memungkinkan UMKM untuk menghasilkan laporan keuangan secara otomatis dan mengurangi kesalahan manual. Pengujian aplikasi dilakukan dengan metode black box testing, dan hasilnya menunjukkan bahwa sistem ini aman dan efektif dalam mendukung pengelolaan keuangan UMKM.