BAB II
TINJAUAN PUSTAKA

2.1 Teori Dasar

Kemajuan teknologi digital telah mengubah pola belanja masyarakat. Salah satu inovasi dalam dunia perdagangan adalah hadirnya *e-commerce*. Untuk menciptakan aplikasi *e-commerce* yang berhasil, diperlukan pemahaman mendalam tentang prinsip dasar *e-commerce* dan teknologi pendukungnya. Bab ini akan mengupas konsep-konsep *fundamental* yang menjadi dasar penelitian dalam pengembangan aplikasi *e-commerce* berbasis Android menggunakan metode *Waterfall*. Pembahasan dalam bab ini mencakup *e-commerce*, aplikasi mobile, metode *Waterfall*, serta hubungan basis data dalam implementasi tersebut.

2.1.1 Rancang Bangun

Menurut JH & Prastowo (2021), rancang bangun merupakan proses merancang, menggambar, dan menyusun elemen-elemen yang sebelumnya terpisah menjadi satu kesatuan yang utuh dan berfungsi. Dengan kata lain, rancang bangun mengacu pada kegiatan mengolah hasil analisis menjadi perangkat lunak serta membangun atau memperbaiki sistem yang ada.

Sementara itu, menurut Novitasari et al. (2021), perancangan adalah tahap penting dalam pengembangan program. Tujuannya adalah memberikan pemahaman yang jelas dan menyeluruh kepada developer yang akan terlibat. Perancangan harus dibuat sesederhana mungkin, mudah dipahami, dan dapat digunakan secara efektif. Proses ini mencakup serangkaian langkah untuk menerjemahkan hasil analisis dan sistem ke dalam bahasa pemrograman serta menjelaskan secara rinci cara setiap komponen sistem dapat diterapkan. Dengan demikian, rancang bangun bukan sekadar proses pengembangan sistem, tetapi juga menyelaraskan perencanaan yang matang dengan implementasi yang terstruktur untuk menghasilkan sistem yang efisien, efektif, dan sesuai dengan kebutuhan pengguna.

2.1.2 Aplikasi

Menurut Dongoran et al. (2020), aplikasi merupakan sebuah perangkat lunak yang dirancang untuk memudahkan pengguna dalam menyelesaikan tugas-tugas tertentu. Pengembangannya melibatkan proses analisis terhadap kebutuhan pengguna, desain antarmuka yang interaktif, serta implementasi algoritma yang mendukung fungsi utama aplikasi tersebut. Aplikasi tidak hanya digunakan untuk tujuan individual, tetapi juga memiliki kontribusi besar dalam skala organisasi, seperti pengolahan data dan pengelolaan informasi.

Selain itu aplikasi merupakan elemen teknologi informasi yang berfungsi sebagai perantara antara pengguna dan sistem. Fungsi aplikasi tidak hanya terbatas pada eksekusi tugas, tetapi juga berperan dalam pengambilan keputusan melalui pengolahan data dan penyajian informasi yang relevan. Pengembangan aplikasi mencakup pemahaman mendalam tentang konteks pengguna dan penerapan prinsip-prinsip rekayasa perangkat lunak untuk memastikan efisiensi dan keandalan (Maulana & Susandi 2020)

Aplikasi dibagi menjadi 3 bagian yaitu:

1. **Aplikasi berbasis web**

   Aplikasi yang bisa diakses lewat browser internet, dirancang agar kompatibel dengan berbagai perangkat, dan biasanya tidak memerlukan instalasi langsung di perangkat pengguna. Contohnya adalah e-commerce dan sistem informasi berbasis web.

2. **Aplikasi berbasis desktop**

   Aplikasi ini dirancang untuk diinstal dan dijalankan pada komputer atau laptop. Biasanya memiliki fungsionalitas yang lebih kaya dan tidak bergantung pada koneksi internet secara penuh, seperti perangkat lunak pengolah data atau desain grafis.

3. **Aplikasi berbasis mobile**

   Aplikasi yang dibuatkan khusus untuk sebuah perangkat mobile seperti ponsel pintar dan tablet. Aplikasi ini umumnya menyediakan antarmuka yang lebih sederhana dan berfokus pada kebutuhan pengguna yang sering berpindah tempat. Contohnya meliputi aplikasi media sosial dan perbankan digital.

2.1.3 E-commerce

E-commerce merupakan bentuk perdagangan yang memanfaatkan teknologi digital untuk melakukan transaksi. Menurut Kalakota dan Andrew BI Whinston (1997), e-commerce mencakup pembelian, penjualan, dan pertukaran barang serta jasa melalui jaringan komputer, terutama internet. Laudon dan Traver (2013) menambahkan bahwa *e-commerce* tidak hanya melibatkan penjualan online, tetapi juga semua transaksi bisnis yang dilakukan secara elektronik. Turban et al. (2015) menyebutkan bahwa *e-commerce* merupakan proses yang mencakup berbagai aktivitas dari transaksi bisnis hingga transfer data secara online.

E-commerce dimulai pada tahun 1970-an dengan perkembangan *Electronic Data Interchange* (EDI) adalah sistem yang memungkinkan pertukaran data bisnis antar Perusahaan. Pada tahun 1990-an, dengan kemunculan *World Wide Web* dan browser pertama seperti Netscape, *e-commerce* mulai tumbuh pesat. Perusahaan seperti Amazon (didirikan pada tahun 1994) dan eBay (didirikan pada tahun 1995) menjadi pionir dalam industri ini.

Ada berbagai macam jenis e-commerce yg tersedia pada saat ini, diantaranya adalah:

- B2B (*Business to Business*): B2B merupakan sebuah *e-commerce* yang melibatkan transaksi antar perusahaan, seperti grosir yang menjual produk ke pengecer. Salah satu contoh *e-commerce* B2B adalah Alibaba yang merupakan sebuah platform *e-commerce* B2B terbesar.
- B2C (*Business to Consumer*): *E-commerce* B2C adalah jenis *e-commerce* yang melibatkan transaksi antara perusahaan dan konsumen akhir. Contoh *e-commerce* B2C yang sangat populer saat ini antara lain Amazon, Shopee, dan Tokopedia.
- C2C (*Consumer to Consumer*): *E-commerce* C2C melibatkan transaksi antara konsumen, biasanya difasilitasi oleh platform pihak ketiga seperti eBay atau OLX.

- C2B (*Consumer to Business*): Model ini melibatkan individu yang menjual produk atau jasa mereka kepada perusahaan. Contohnya adalah platform seperti *Upwork* di mana freelancer menawarkan layanan kepada bisnis
- B2G (*Business to Government*): Transaksi antara perusahaan dan pemerintah, misalnya penyediaan produk atau layanan untuk proyek pemerintah melalui tender online.

Didalam *e-commerce* terdapat beberapa aktivitas yg dilakukan dalam melakukan proses transaksi,diantaranya:

- Langkah awal yang dilakukan adalah mencari produk. Pengguna menelusuri produk atau layanan yang mereka butuhkan melalui situs *e-commerce* atau aplikasi mobile.
- Setelah produk yang diinginkan ditemukan, pengguna memasukkan produk tersebut ke dalam keranjang belanja.
- Pengguna menyelesaikan pembelian dengan memilih metode pembayaran yang tersedia, seperti kartu kredit, transfer bank, atau dompet digital (*e-wallet*).
- Setelah pembayaran berhasil dikonfirmasi, produk akan dikirimkan ke alamat yang telah ditentukan oleh pengguna.
- Kemudian tambahan layanan tambahan seperti pengembalian barang, klaim garansi, atau bantuan pelanggan untuk masalah teknis.

2.1.4 Android

Android merupakan sebuah sistem operasi yang berbasis Linux yang dibuat khusus untuk sebuah perangkat seluler. Sistem android ini mencakup sistem operasi, *middleware*, dan juga aplikasi. Serta menyediakan platform terbuka bagi pengembang untuk menciptakan aplikasi mereka. Pada awalnya, Android Inc., perusahaan yang fokus pada pengembangan perangkat lunak untuk ponsel pintar, diakuisisi oleh Google Inc. untuk mendukung pengembangan Android lebih lanjut, kemudian dibentuk Open Handset Alliance (OHA) (Maliki, 2021).

2.1.5 Metode Waterfall

Metode Waterfall merupakan sebuah motode yang paling umum dipakai dalam pengembangan perangkat lunak. Metode waterfall ini berfungsi secara

linear dan terstruktur, di mana setiap tahapan yang ada didalam metode waterfall harus diselesaikan dahulu sebelum melanjutkan ke tahap berikutnya. Model ini pertama kali diperkenalkan oleh Dr. Winston W. Royce pada tahun 1970. Metode ini menekankan pentingnya memiliki struktur yang terorganisir dengan baik serta dokumentasi yang lengkap.

Tahapan dalam metode Waterfall mengikuti siklus Software Development Life Cycle (SDLC), yaitu proses pengembangan yang digunakan untuk merancang dan membangun perangkat lunak atau aplikasi.

Dan berikut adalah detail tahapan-tahapan yang digunakan didalam metode waterfall:

- Perencanaan: Menetapkan tujuan proyek, kebutuhan pengguna, dan ruang lingkup proyek. Tahap ini melibatkan pembuatan dokumen persyaratan yang detail.
- Analisis: Mengidentifikasi dan mendokumentasikan kebutuhan perangkat lunak secara rinci, termasuk kebutuhan fungsional dan non-fungsional.
- Desain: Merancang arsitektur sistem, struktur basis data, antarmuka pengguna, dan elemen desain lainnya. Tahap ini menghasilkan dokumen perancangan yang berfungsi sebagai panduan dalam proses pengembangan.
- Implementasi: Tahap pengkodean di mana pengembang menulis kode sumber berdasarkan dokumen desain. Implementasi dilakukan secara modul atau komponen yang diintegrasikan secara bertahap.
- Pengujian: Pengujian merupakan sebuah tahapan implementasi Dimana tahapan ini dirancang untuk memastikan bahwa sistem beroperasi sesuai dengan spesifikasi yang telah ditetapkan. Proses pengujian mencakup pengujian unit, integrasi, sistem, serta uji penerimaan oleh pengguna.
- Pemeliharaan: Setelah sistem diluncurkan, diperlukan pemeliharaan untuk memperbaiki bug, meningkatkan performa, dan menambahkan fitur-fitur baru. Tahapan ini juga mencakup pembaruan keamanan serta penyesuaian terhadap perkembangan lingkungan teknologi.

2.1.6 React Native

React Native merupakan sebuah kerangka kerja *open-source* yang memungkinkan pengembangan aplikasi mobile menggunakan *JavaScript*. *Framework* ini dikembangkan oleh Facebook dan pertama kali dirilis pada tahun 2015. React Native memungkinkan pengembang untuk membuat aplikasi lintas platform (Android dan iOS) dengan satu basis kode.

Menurut Akther dan Azam (2021), React Native memanfaatkan pustaka React untuk membangun antarmuka pengguna dan menerjemahkan komponen tersebut ke dalam elemen native di perangkat.

Beberapa fitur utama pada React Native

- Pengembangan Lintas Platform (Dengan satu basis kode, pengembang dapat menciptakan aplikasi untuk Android dan iOS secara bersamaan)
- Pemuatan Ulang Langsung (Fitur ini memungkinkan pengembang untuk langsung melihat perubahan kode tanpa perlu membangun ulang aplikasi)
- Komponen Reusable (Komponen yang dibuat dapat digunakan kembali dalam berbagai proyek)
- Integrasi dengan Modul Native (React Native memungkinkan integrasi dengan modul native untuk memanfaatkan fitur perangkat keras, seperti kamera dan GPS.)

React Native menggunakan arsitektur bridge yang menghubungkan komponen React dengan API native. Proses ini memungkinkan komunikasi antara *JavaScript* dan elemen native melalui protokol asynchronous.

2.1.7 Node js

Node.js adalah lingkungan *runtime open-source* untuk *JavaScript* yang beroperasi di luar *browser*. Dikembangkan oleh Ryan Dahl pada tahun 2009,

Node.js memanfaatkan mesin *JavaScript* V8 dari Google untuk menjalankan kode *JavaScript* dengan performa tinggi. Teknologi ini memungkinkan pengembang membuat aplikasi sisi server yang cepat dan mudah untuk diskalakan.

Menurut Tilkov dan Vinoski (2010), Node.js dirancang untuk mengembangkan aplikasi jaringan yang ringan dengan pendekatan event-driven dan I/O non-blocking. Pendekatan ini menjadikannya sangat efisien dalam menangani aplikasi dengan banyak koneksi secara simultan.

.

Fitur Utama dalam Node Js:

- Non-Blocking I/O

  Node.js memproses operasi I/O secara asynchronous, memungkinkan aplikasi menangani ribuan permintaan sekaligus tanpa memblokir proses lainnya.

- Event-Driven Architecture

  Node.js menggunakan loop event yang terus berjalan untuk menangani tugas tanpa perlu membuat thread tambahan, sehingga meningkatkan efisiensi.

- NPM (Node Package Manager)

  Node.js dilengkapi dengan NPM, yang menyediakan akses ke lebih dari satu juta pustaka dan modul, memudahkan pengembang dalam menambahkan fitur tambahan ke aplikasi.

- Single-Threaded

Node.js menggunakan arsitektur single-threaded yang cocok untuk aplikasi real-time seperti obrolan, streaming data, dan API berbasis REST.

Node.js didasarkan pada event loop non-blocking yang berjalan dalam thread tunggal. Semua operasi *asynchronous* diproses melalui mekanisme *callback*, sementara tugas yang lebih berat diserahkan ke *thread pool*. Mesin V8 digunakan untuk mengeksekusi *JavaScript*, sedangkan libuv menangani operasi sistem tingkat rendah seperti I/O file dan jaringan.

2.1.8 UML (Unified Modeling Language)

Unified Modeling Language (UML) adalah sebuah permodelan yang paling umum untuk digunakan dalam mendefinisikan, memvisualisasikan, mengembangkan, dan mendokumentasikan artefak perangkat lunak. UML dirancang agar dapat membantu para developer dalam memahami, merancang, dan melakukan pengembanhan terhadap sistem perangkat lunak yang kompleks melalui berbagai jenis diagram yang telah distandarisasi.

Menurut Prasetya dkk. (2022), UML adalah bahasa pemodelan berorientasi objek yang digunakan untuk memvisualisasikan dan mendokumentasikan desain perangkat lunak. Dengan menyediakan representasi grafis yang terstruktur dan standar, UML mempermudah komunikasi antara tim pengembang, analis, dan pemangku kepentingan.

Ada beberapa fungsi utama UML, diantaranya:

- Memvisualisasikan Sistem

  UML membantu menggambarkan struktur, interaksi, dan perilaku sistem secara grafis, sehingga mudah dipahami oleh pengembang maupun non-teknis.

- Mendokumentasikan Sistem

UML menyediakan diagram yang dapat digunakan sebagai dokumentasi formal selama siklus hidup pengembangan perangkat lunak.

- Menganalisis dan Merancang Sistem

  UML digunakan untuk merancang arsitektur perangkat lunak dengan fokus pada pendekatan berorientasi objek.

Pada UML terdapat beberapa diagram, diantaranya adalah:

a. Use case Diagram

Use Case Diagram merupakan jenis diagram yang terdapat didalam UML yang biasa digunakan dalam memodelkan sebuah interaksi antara aktor (baik pengguna maupun sistem lain) dengan sistem. Diagram ini menggambarkan berbagai fungsi yang disediakan oleh sistem kepada penggunanya dalam bentuk "*use case."*

Menurut Aminah dan Budi (2020), "*Use Case* Diagram mempermudah pemahaman kebutuhan pengguna dengan menampilkan hubungan antara aktor dan fungsi sistem secara sederhana dan mudah dipahami."

Pada usecase diagram terdapat komponen utama seperti:

SIMBOL	NAMA	KETERANGAN
	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Gambar 2.1.8. 1 - komponen utama use case diagram

b. Activity Diagram

Activity Diagram adalah salah satu diagram UML yang digunakan untuk memvisualisasikan alur kerja (*workflow*) atau aktivitas dalam sebuah proses sistem, baik yang berlangsung secara paralel maupun berurutan.

Menurut Prasetyo (2021), "Activity Diagram digunakan untuk memodelkan proses bisnis atau alur logis sistem secara grafis, sehingga memudahkan analisis dan pengembangan."

Pada Activity diagram terdapat komponen utama seperti:

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.1.8. 2 - komponen utama activity diagram

c. Sequence Diagram

Sequence Diagram merupakan salah satu diagram UML yang digunakan untuk menunjukkan interaksi antar objek berdasarkan urutan waktu tertentu. Diagram ini menekankan bagaimana objek bekerja sama untuk menjalankan fungsionalitas dalam sistem.

Menurut Widodo (2021), "Sequence Diagram berfungsi untuk memvisualisasikan komunikasi antar objek dalam sistem, termasuk pesan yang dikirimkan serta tanggapannya."
Berikut adalah komponen utama pada sequence diagram:

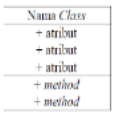
Gambar	Nama	Keterangan
	Entity Class	Gambaran sistem sebagai landasan dalam menyusun basis data
	Boundary Class	Menangani komunikasi antar lingkungan sistem
	Control Class	Bertanggung jawab terhadap kelas-kelas terhadap objek yang berisi logika
	Recursive	Pesan untuk dirinya
	Activation	Mewakili proses durasi aktivasi sebuah operasi
	Life Line	Komponen yang digambarkan garis putus terhubung dengan objek

Gambar 2.1.8. 3 - komponen utama Sequence diagram

d. Class Diagram

Class Diagram adalah salah satu diagram UML yang digunakan untuk memodelkan struktur statis sebuah sistem dengan menampilkan kelas, atribut, metode, serta hubungan antar kelas.

Menurut Nugroho (2020), "Class Diagram menyajikan representasi visual dari struktur sistem berorientasi objek, mencakup relasi antar kelas seperti *inheritance, association, dan aggregation*."
Berikut adalah komponen utama pada class diagram:

Simbol	Nama	Deskripsi
	Class	Himpunan objek-objek dari berbagai atribut yang me-miliki operasi yang sama.
	Association	Relasi antar kelas dengan makna umum dan biasanya disertai multiplicity.
	Directed Association	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain.
	Aggregation	Mengindikasikan kese-luruhan bagian rela-tionship disebut sebagai relasi.
	Composition	Relasi Composition terhadap class tempat dia bergantung.
	Dependency	Menunjukkan operasi pada suatu class yang menggunakan class yang lain.

Gambar 2.1.8. 4 - komponen utama class diagram

2.1.9 Basis Data

Basis data dapat dibagi menjadi dua jenis, yaitu Basis Data Relasional dan Non-relasional. Basis data relasional adalah jenis basis data yang menyusun data dalam tabel-tabel yang saling terhubung, di mana setiap tabel terdiri atas baris dan kolom. Basis data relasional selalu menggunakan SQl untuk mengelola dan memanipulasi data, contohnya adalah MySQL, PstgreSQL, dan Oracle. Sedangkan basis data Non-relasional atau juga dikenal sebagai NoSQL. Basis data ini tidak menggunakan tabel segai struktur utama dan sering kali digunakan untuk data yang tidak terstruktur atau semi-terstruktur. Contoh basis data Non-Relasional (NoSQL) adalah MongoDB (Basis data dokumen), Cassandra (Basis data kolom), dan Redis (Basis data *key-value*).

Struktur yang ada pada Basis Data terdiri dari Tabel, Record, dan juga Field:

- Tabel: Struktur utama dalam basis terdiri dari baris dan kolom. Setiap tabel menyimpan data tentang entitas tertentu, seperti pelanggan, produk, atau transaksi.
- Record: Satu baris dalam tabel yang mewakili entitas tunggal atau item data. Misalnya, satu record dalam tabel pelanggan dapat menyimpan data tentang satu pelanggan.
- Field: Kolom dalam tabel yang menyimpan atribut atau karakteristik dari entitas yang direpresentasikan oleh tabel. Misalnya, dalam tabel pelanggan, field bisa mencakup nama, alamat, dan nomor telepon.

Structured Query Language (SQL) adalah bahasa standar yang digunakan untuk pengelolaan dan manipulasi basis data relasional. Dengan SQL, pengguna dapat melakukan berbagai operasi, seperti menambahkan, membaca, mengubah, dan menghapus data dalam basis data.

Structured Query Language (SQL) mempunyai beberapa fungsi, diantaranya:

- DDL (*Data Definition Language*): Komponen SQL yang digunakan untuk menentukan struktur basis data, termasuk perintah seperti CREATE, ALTER, dan DROP.
- DML (*Data Manipulation Language*): Bagian dari SQL yang berfungsi untuk memanipulasi data dalam basis data menggunakan perintah seperti SELECT, INSERT, UPDATE, dan DELETE.
- DCL (*Data Control Language*): Bagian SQL yang bertujuan mengelola hak akses data dalam basis data, dengan perintah seperti GRANT dan REVOKE.

2.2 Tinjauan Studi

Tinjauan studi ini akan menguraikan berbagai penelitian yang telah dilakukan terkait pengembangan aplikasi *e-commerce*, penerapan metode pengembangan perangkat lunak Waterfall, dan studi kasus CV. PRATAMA TEKNIK. Tinjauan ini bertujuan untuk memberikan gambaran umum tentang perkembangan penelitian sebelumnya dan mengidentifikasi gap penelitian yang ada.

1. Penelitian Penelitian Terdahulu tentang Pengembangan Aplikasi *E-commerce*. Seiring dengan pertumbuhan teknologi mobile, penelitian mengenai pengembangan aplikasi *e-commerce* berbasis mobile semakin berkembang. Banyak studi yang telah dilakukan untuk mengevaluasi efektivitas aplikasi mobile dalam meningkatkan pengalaman pengguna dan mendorong penjualan. Penelitian-penelitian ini umumnya menekankan pentingnya desain antarmuka yang *user-friendly* dan navigasi yang intuitif untuk meningkatkan retensi pengguna dan kepuasan pelanggan.

   Android, sebagai platform mobile dengan pangsa pasar terbesar, sering menjadi pilihan utama untuk pengembangan aplikasi *e-commerce*. Penelitian yang berfokus pada pengembangan aplikasi *e-commerce* berbasis Android menunjukkan bahwa platform ini menawarkan *fleksibilitas* dan kemudahan akses bagi pengembang untuk menjangkau audiens yang lebih luas. Namun, tantangan seperti fragmentasi perangkat dan keamanan tetap menjadi perhatian utama dalam pengembangan aplikasi berbasis Android.

   Penelitian ini membahas pengembangan aplikasi *e-commerce* menggunakan metode Waterfall. Kurniawan, A., & Susanto, M. (2020) mengemukakan bahwa metode Waterfall memberikan struktur yang jelas dan terorganisir, yang sangat berguna dalam proyek dengan persyaratan yang sudah mapan. Namun, mereka juga mencatat keterbatasan metode ini dalam menanggapi perubahan yang terjadi setelah tahap analisis dan desain.

2. Penerapan Metode Waterfall dalam Pengembangan Aplikasi. Metode Waterfall merupakan model pengembangan perangkat lunak yang telah digunakan sejak lama dan tetap relevan hingga saat ini. Sommerville (2016) menyatakan bahwa metode ini ideal untuk proyek dengan spesifikasi yang sudah jelas dan stabil, di mana perubahan hanya sedikit terjadi setelah tahap analisis. Penelitian menunjukkan bahwa pendekatan linier Waterfall memungkinkan setiap tahap pengembangan direncanakan dan didokumentasikan secara rinci.

Dalam proyek berskala besar, metode Waterfall sering dianggap sebagai pendekatan yang andal karena memungkinkan pengawasan yang ketat pada setiap fase pengembangan. Pressman (2019) menekankan bahwa keberhasilan metode ini sangat dipengaruhi oleh efektivitas manajemen proyek dan kepatuhan tim terhadap jadwal yang telah ditentukan.

3. Studi Kasus CV. PRATAMA TEKNIK. CV. PRATAMA TEKNIK merupakan salah satu perusahaan yang beerfokus pada bidang penyediaan peralatan teknik dan industri. Perusahaan ini telah beroperasi selama kurang lebih 5 tahun dan memiliki jaringan distribusi yang cukup luas. Namun, dengan meningkatnya persaingan bisnis dan perubahan dalam cara konsumen melakukan pembelian, perusahaan ini perlu beradaptasi dengan teknologi digital untuk mempertahankan dan memperluas cangkupan pasarnya. Oleh sebab itu, perusahaan ini membutuhkan sebuah sistem/aplikasi yang dapat mendukung penjualan online, manajemen inventaris, serta transaksi yang efisien dan aman. Aplikasi *e-commerce* berbasis Android ini diharapkan dapat membantu perusahaan dalam menjangkau lebih banyak pelanggan, mengelola data produk secara efisien, dan menyediakan platform yang *user-friendly* bagi konsumen.
   Dalam mengadopsi teknologi e-commerce, CV. PRATAMA TEKNIK menghadapi beberapa tantangan, termasuk integrasi dengan sistem manajemen yang sudah ada, pelatihan staf, serta adaptasi pelanggan terhadap platform digital. Selain itu, menjaga keamanan data dan stabilitas sistem juga menjadi tantangan penting dalam implementasi aplikasi *e-commerce* ini.