

**PENERAPAN ARSITEKTUR CLIENT DATA LAYER  
MENGGUNAKAN TANSTACK QUERY PADA  
DASHBOARD SENTIMENT ANALYSIS  
DENGAN METODE FOUNTAIN**

**PROPOSAL TUGAS AKHIR**

Diajukan Sebagai Syarat Untuk Pengajuan Tugas Akhir Pada Program Studi Sarjana Terapan  
Teknologi Rekayasa Perangkat Lunak Jurusan Bisnis dan Informatika Politeknik Negeri  
Banyuwangi



**Oleh:**  
**NASRUL FAHMI ULUMUDDIN**  
**NIM 362258302204**

**PROGRAM STUDI SARJANA TERAPAN TEKNOLOGI REKAYASA  
PERANGKAT LUNAK  
JURUSAN BISNIS DAN INFORMATIKA  
POLITEKNIK NEGERI BANYUWANGI  
2026**

**PENERAPAN ARSITEKTUR CLIENT DATA LAYER  
MENGGUNAKAN TANSTACK QUERY PADA DASHBOARD  
SENTIMENT ANALYSIS  
DENGAN METODE FOUNTAIN**

**PROPOSAL TUGAS AKHIR**

Diajukan kepada Jurusan Bisnis dan Informatika Politeknik Negeri Banyuwangi Sebagai  
Syarat Untuk Pengajuan Tugas Akhir Pada Program Studi Sarjana Terapan Teknologi  
Rekayasa Perangkat Lunak

Oleh:

NASRUL FAHMI ULUMUDDIN

362258302204

Pembimbing:

Sepyan Purnama Kristanto, S.Kom., M.Kom

Arum Andary Ratri, S.Si., M.Si.

**PROGRAM STUDI SARJANA TERAPAN TEKNOLOGI REKAYASA  
PERANGKAT LUNAK  
JURUSAN BISNIS DAN INFORMATIKA  
POLITEKNIK NEGERI BANYUWANGI  
2026**

## **LEMBAR PERSETUJUAN**

Proposal Tugas Akhir dengan Judul

### **PENERAPAN ARSITEKTUR CLIENT DATA LAYER MENGGUNAKAN TANSTACK QUERY PADA DASHBOARD SENTIMENT ANALYSIS DENGAN METODE FOUNTAIN**

Disusun oleh:  
**NASRUL FAHMI ULUMUDDIN**  
**NIM 362258302204**

telah memenuhi syarat dan disetujui oleh Dosen Pembimbing untuk dilaksanakan Seminar  
Proposal Tugas Akhir bagi yang bersangkutan.

Banyuwangi, 7 Januari 2026

Mengetahui,  
Koordinator Program Studi,

Disetujui,  
Dosen Pembimbing 1,

Lutfi Hakim, S.Pd., M.T.  
NIP. 199203302019031012

Sepyan Purnama Kristanto, S.Kom., M.Kom  
NIP. 199009052019031024

Banyuwangi, 7 Januari 2026

Disetujui,  
Dosen Pembimbing 2,

Arum Andary Ratri, S.Si., M.Si.  
NIP. 199209212020122021

# **Penerapan Arsitektur Client Data Layer Menggunakan TanStack Query pada Dashboard Sentiment Analysis dengan metode Fountain**

Oleh  
NASRUL FAHMI ULUMUDDIN  
NIM: 362258302204

## **ABSTRAK**

Abstrak adalah sebuah ringkasan singkat yang menjelaskan secara umum tentang isi dari laporan tugas akhir. Abstrak ditulis dalam tiga (3) paragraf yang berisi beberapa kalimat yang menyatakan tujuan, metode, hasil, dan kesimpulan dari laporan tugas akhir. Paragraf pertama berisi latar belakang dan tujuan tugas akhir. Paragraf kedua berisi metode dan pembahasannya. Paragraf ketiga berisi hasil dan simpulan dari tugas akhir yang dikerjakan.

Abstrak harus menjelaskan secara jelas dan singkat apa yang dibahas dalam laporan tugas akhir, mengapa penelitian ini penting dan apa yang ditemukan dari penelitian tersebut. Abstrak harus ditulis dengan bahasa yang mudah dipahami dan harus mencakup informasi penting yang dibahas dalam laporan tugas akhir.

Abstrak harus mengandung kata-kata yang relevan dengan laporan tugas akhir dan ditulis dengan bahasa yang formal dan akademik. Abstrak merupakan bagian penting dari sebuah laporan tugas akhir karena merupakan bagian yang pertama kali dibaca oleh pembaca dan harus dapat memberikan gambaran yang jelas tentang isi dari laporan tugas akhir. Oleh karena itu, abstrak harus ditulis dengan baik dan sebaik mungkin agar dapat memberikan gambaran yang jelas tentang laporan tugas akhir yang ditulis. Panjang abstrak sebaiknya dicukupkan dalam satu halaman, termasuk kata kunci. Tiga kata kunci dipandang cukup, yang masing-masingnya memuat paduan kata utama, yang dapat merepresentasikan isi Abstrak.

Kata kunci: Konsep Abstrak, Komponen Abstrak, Kata Kunci.

# **Application of Client Data Layer Architecture Using TanStack Query on Sentiment Analysis Dashboard with the Fountain Method**

by:

NASRUL FAHMI ULUMUDDIN

NIM: 362258302204

## **ABSTRACT**

The abstract is a short summary that explains in general the contents of the final assignment report. The abstract is written in three (3) paragraphs containing several sentences stating the objectives, methods, results and conclusions of the final assignment report. The first paragraph contains the background and objectives of the final assignment. The second paragraph contains the method and discussion. The third paragraph contains the results and conclusions of the final assignment carried out.

The abstract must explain clearly and concisely what is discussed in the final project report, why this research is important and what was found from the research. The abstract must be written in language that is easy to understand and must include important information discussed in the final project report.

The abstract must contain words that are relevant to the final project report and be written in formal and academic language. The abstract is an important part of a final assignment report because it is the part that is first read by the reader and must be able to provide a clear picture of the contents of the final assignment report. Therefore, the abstract must be written well and as well as possible in order to provide a clear picture of the final project report being written. The length of the abstract should be limited to one page, including keywords. Three keywords are considered sufficient, each of which contains a combination of main words, which can represent the contents of the Abstract.

Key words: Abstract Concepts, Abstract Components, Key Words.

## DAFTAR ISI

<b>HALAMAN SAMPUL . . . . .</b>	<b>i</b>
<b>LEMBAR PERSETUJUAN PROPOSAL . . . . .</b>	<b>ii</b>
<b>ABSTRAK . . . . .</b>	<b>iii</b>
<b>ABSTRACT . . . . .</b>	<b>iv</b>
<b>DAFTAR ISI . . . . .</b>	<b>v</b>
<b>DAFTAR SINGKATAN . . . . .</b>	<b>vii</b>
<b>DAFTAR GAMBAR . . . . .</b>	<b>viii</b>
<b>DAFTAR TABEL . . . . .</b>	<b>ix</b>
<b>BAB 1 PENDAHULUAN . . . . .</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	3
1.3 Tujuan . . . . .	3
1.4 Manfaat . . . . .	3
1.5 Batasan . . . . .	4
<b>BAB 2 TINJAUAN PUSTAKA . . . . .</b>	<b>5</b>
2.1 Landasan Teori . . . . .	5
2.1.1 UMKM dan Digitalisasi . . . . .	5
2.1.2 Media Sosial sebagai Sumber Data . . . . .	6
2.1.3 Analisis Sentimen pada Media Sosial . . . . .	6
2.1.4 Dashboard Analitik dan Visualisasi Data . . . . .	7
2.1.5 Arsitektur Client Data Layer . . . . .	9
2.1.6 TanStack Query (React Query) . . . . .	12
2.1.7 REST API . . . . .	14
2.2 Penelitian Terkait . . . . .	15
2.3 Analisa Gap Penelitian . . . . .	15
2.3.1 Gap Penelitian 1 . . . . .	15
2.3.2 Gap Penelitian 2 . . . . .	15
2.3.3 Gap Penelitian 3 . . . . .	16
<b>BAB 3 METODOLOGI PENELITIAN . . . . .</b>	<b>17</b>
3.1 Waktu dan Jadwal penelitian . . . . .	17
3.1.1 Waktu Pelaksanaan Penelitian . . . . .	17
3.1.2 Jadwal Kegiatan Penelitian . . . . .	17
3.2 Methode Penelitian . . . . .	18
3.2.1 Metode Fountain . . . . .	18
3.2.2 Alasan Pemilihan Metode Fountain . . . . .	19
3.2.3 Alasan Pemilihan Metode Fountain . . . . .	19
3.3 Perancangan Sistem . . . . .	21
3.3.1 Gambaran Umum Sistem . . . . .	21
3.3.2 Perancangan Arsitektur Frontend . . . . .	21
3.3.3 Perancangan Client Data Layer . . . . .	23
3.3.4 Perancangan Penggunaan TanStack Query . . . . .	24
3.3.5 Perancangan Antarmuka . . . . .	25
3.4 Metode Pengujian dan Evaluasi Sistem . . . . .	27
3.4.1 Metode Pengujian . . . . .	27

3.4.2 Skenario dan Objek Pengujian . . . . .	27
3.4.3 Teknik Evaluasi . . . . .	27
<b>DAFTAR PUSTAKA . . . . .</b>	<b>29</b>

## **DAFTAR SINGKATAN**

FWHM	:	<i>Full width half maximum</i>
rms	:	<i>root mean square</i>
RFS	:	<i>Rotary forcespinning</i>
PVP	:	Polivinil pirolidon
SI	:	Satuan Internasional

## **DAFTAR GAMBAR**

2.1	Architecture Client Data Layer .....	10
3.1	Fountain SDLC Model .....	18
3.2	Diagram Arsitektur Frontend .....	22
3.3	Wireframe Landing Page .....	25
3.4	Wireframe Halaman Login .....	25
3.5	Wireframe Halaman Register .....	25
3.6	Wireframe Halaman Dashboard .....	26
3.7	Wireframe Halaman Sentiment .....	26
3.8	Wireframe Halaman Scraper .....	27
3.9	Wireframe Halaman Recomendation .....	27

## **DAFTAR TABEL**

2.1 Tabel Perbandingan Penelitian Terkait .....	15
3.1 Jadwal Kegiatan Penelitian .....	17

# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Usaha Mikro, Kecil, dan Menengah (UMKM) merupakan salah satu pilar utama dalam perekonomian Indonesia. Berdasarkan beberapa studi nasional, UMKM berkontribusi besar terhadap Produk Domestik Bruto (PDB) serta memiliki peran penting dalam penyerapan tenaga kerja. Namun, meskipun kontribusinya besar, tingkat literasi digital UMKM di Indonesia masih tergolong rendah, terutama dalam pemanfaatan teknologi informasi untuk analisis pasar dan pengambilan keputusan berbasis data. Penelitian oleh (Irianto et al., 2022) menyebutkan bahwa banyak UMKM belum mampu memanfaatkan teknologi digital secara optimal karena keterbatasan pengetahuan, akses, dan kemampuan mengelola data.

Di era digital saat ini, perilaku konsumen telah berubah secara signifikan. Media sosial seperti Instagram, TikTok, dan Facebook tidak hanya menjadi saluran pemasaran, tetapi juga menjadi ruang interaksi antara konsumen dan pelaku usaha. Studi oleh (Trulline, 2021) menunjukkan bahwa UMKM semakin bergantung pada media sosial untuk mempromosikan produk, membangun reputasi, dan memahami minat pasar. Komentar konsumen pada postingan media sosial mencerminkan persepsi publik terhadap produk atau layanan UMKM, sehingga dapat dimanfaatkan sebagai indikator kepuasan dan reputasi merek.

Namun, data komentar media sosial bersifat tidak terstruktur, jumlahnya besar, dan berubah secara dinamis, sehingga sulit dianalisis tanpa bantuan teknologi. Sejalan dengan (Joseph, 2024), Sentiment Analysis pada media sosial merupakan bidang penelitian yang berkembang pesat dan melibatkan penggunaan Natural Language Processing (NLP), text analysis, dan computational linguistics untuk mengidentifikasi serta mengekstraksi informasi subjektif dari data teks. Pendekatan NLP tersebut memungkinkan komentar tidak terstruktur diproses menjadi kategori sentimen yang dapat diinterpretasikan, baik positif, negatif, maupun netral. Dengan demikian, hasil analisis sentimen dari proses NLP dapat disajikan dalam bentuk dashboard interaktif agar UMKM dapat memahami tren sentimen, isu yang sering muncul, serta persepsi pelanggan secara lebih cepat dan intuitif.

Dalam pengembangan dashboard analitik berbasis web, tantangan tidak hanya terletak pada penyajian visualisasi data, tetapi juga pada pengelolaan alur data di sisi frontend. Dashboard analitik merupakan aplikasi yang bersifat data-driven, di mana berbagai komponen antarmuka seperti grafik, tabel, dan indikator bergantung pada data yang sama dan diperbarui

secara berkala melalui API. Jika pengambilan data dilakukan secara langsung pada setiap komponen tanpa arsitektur pengelolaan data yang terstruktur, maka dapat muncul berbagai permasalahan, seperti permintaan API berulang, inkonsistensi data antar-komponen, serta kesulitan dalam pemeliharaan kode aplikasi.

Pendekatan arsitektur Client Data Layer hadir sebagai solusi untuk mengelola data yang bersumber dari server secara terpusat di sisi klien. Dengan adanya Client Data Layer, proses pengambilan, penyimpanan sementara, dan sinkronisasi data dapat dilakukan secara terstruktur, sehingga komponen antarmuka tidak perlu berinteraksi langsung dengan API. Salah satu pustaka yang mendukung pendekatan ini adalah TanStack Query, yang dirancang untuk mengelola server state secara efisien melalui mekanisme caching, deduplikasi permintaan, serta sinkronisasi data antar-komponen.

Sejumlah penelitian terdahulu menunjukkan bahwa TanStack Query memiliki keunggulan dalam pengelolaan data asinkron pada aplikasi frontend dibandingkan pendekatan pengelolaan data konvensional, khususnya pada aplikasi yang bersifat data-driven dan menampilkan data dalam jumlah besar (Luz, 2025). Penelitian yang membandingkan TanStack Query dengan pustaka state management lain juga melaporkan adanya peningkatan efisiensi pengelolaan server state dan pengurangan permintaan data yang tidak diperlukan (Micheal, 2025). Namun demikian, sebagian besar penelitian tersebut masih berfokus pada aspek teknis pustaka atau perbandingan antar-library, dan belum secara spesifik mengkaji penerapan TanStack Query sebagai Client Data Layer pada studi kasus dashboard analitik.

Berdasarkan permasalahan tersebut, penelitian ini muncul dari hipotesis bahwa penerapan arsitektur Client Data Layer menggunakan TanStack Query pada aplikasi dashboard analitik mampu menghasilkan pengelolaan data frontend yang lebih terstruktur dan konsisten dibandingkan pendekatan pengelolaan data konvensional, yaitu pengambilan data API secara langsung pada komponen antarmuka tanpa mekanisme pengelolaan data terpusat. Perilaku tersebut diasumsikan tercermin melalui pemanfaatan mekanisme caching, pengendalian permintaan data dari API, serta konsistensi data yang ditampilkan pada berbagai komponen dashboard. Untuk menguji hipotesis tersebut, TanStack Query diterapkan pada studi kasus pengembangan Dashboard Analisis Sentimen UMKM dengan metode Fountain, dan hasilnya dievaluasi melalui observasi perilaku sistem menggunakan pendekatan blackbox testing berbasis log.

## **1.2 Rumusan Masalah**

1. Bagaimana menerapkan arsitektur Client Data Layer menggunakan TanStack Query pada pengembangan Dashboard Analisis Sentimen UMKM?
2. Bagaimana perilaku pengelolaan data frontend yang dihasilkan setelah penerapan TanStack Query ditinjau dari log permintaan data, mekanisme caching, dan konsistensi data antar-komponen dashboard?

## **1.3 Tujuan**

Penelitian ini bertujuan untuk menerapkan arsitektur Client Data Layer menggunakan TanStack Query pada studi kasus pengembangan Dashboard Analisis Sentimen UMKM. Penerapan arsitektur ini difokuskan pada pengelolaan data API di sisi frontend agar proses pengambilan dan penyajian data dapat dilakukan secara terstruktur.

Selain itu, penelitian ini bertujuan untuk mengevaluasi perilaku pengelolaan data frontend setelah penerapan TanStack Query melalui pendekatan blackbox testing. Evaluasi dilakukan dengan mengamati log sistem yang berkaitan dengan permintaan API, pemanfaatan mekanisme caching, serta sinkronisasi data antar-komponen dashboard. Melalui penerapan dan evaluasi tersebut, penelitian ini diharapkan dapat memberikan gambaran praktis mengenai implementasi arsitektur Client Data Layer pada aplikasi frontend berbasis React.

## **1.4 Manfaat**

### **Manfaat Teoritis**

1. Penelitian ini diharapkan dapat menambah literatur mengenai penerapan TanStack Query dalam arsitektur data layer pada studi kasus aplikasi dashboard.

### **Manfaat Praktis**

#### **1. bagi UMKM**

Penelitian ini menghasilkan sebuah dashboard analisis sentimen yang dapat membantu UMKM memahami persepsi konsumen berdasarkan data media sosial secara lebih terstruktur dan mudah dipahami.

#### **2. Manfaat bagi Pengembang**

Penelitian ini memberikan studi kasus penerapan arsitektur Client Data Layer menggunakan TanStack Query yang dapat dijadikan acuan dalam merancang pengelolaan data frontend pada aplikasi data-driven.

#### **3. Manfaat bagi Akademisi**

Penelitian ini dapat dijadikan referensi bagi penelitian sejenis yang membahas arsitektur frontend, pengelolaan server state, dan pengembangan dashboard

analitik.

## 1.5 Batasan

Batasan proyek ditetapkan agar penelitian tetap terfokus pada tujuan yang telah dirumuskan. Adapun batasan proyek dalam Tugas Akhir ini adalah sebagai berikut:

1. Studi kasus penelitian difokuskan pada Dashboard Analisis Sentimen UMKM dengan ruang lingkup penelitian terbatas pada sisi frontend aplikasi.
2. Penelitian hanya membahas penerapan arsitektur Client Data Layer menggunakan TanStack Query pada aplikasi frontend berbasis React.
3. Proses analisis sentimen, termasuk pengumpulan data media sosial dan pemrosesan teks, tidak dibahas dalam penelitian ini dan sepenuhnya dilakukan pada sisi backend.
4. Penelitian ini tidak melakukan perbandingan implementasi TanStack Query dengan pustaka atau framework state management lain
5. Evaluasi sistem dilakukan menggunakan pendekatan blackbox testing dengan mengamati log sistem, tanpa melakukan pengujian performa numerik atau benchmarking mendalam.
6. Hasil evaluasi disajikan dalam bentuk ringkasan log, tabel, dan visualisasi sederhana untuk menggambarkan perilaku sistem setelah penerapan TanStack Query.
7. Penelitian ini tidak membahas aspek optimasi backend, keamanan aplikasi, pengujian beban, maupun pengujian kegunaan secara mendalam.

## **BAB 2**

### **TINJAUAN PUSTAKA**

#### **2.1 Landasan Teori**

##### **2.1.1 UMKM dan Digitalisasi**

Usaha Mikro, Kecil, dan Menengah (UMKM) merupakan sektor usaha produktif yang memiliki peran penting dalam perekonomian, khususnya dalam menciptakan lapangan kerja dan mendorong pertumbuhan ekonomi lokal. UMKM umumnya memiliki karakteristik berupa skala usaha yang relatif kecil, keterbatasan modal, serta pengelolaan usaha yang masih sederhana. Dalam beberapa tahun terakhir, perkembangan teknologi digital telah mendorong UMKM untuk beradaptasi dengan perubahan lingkungan bisnis agar tetap mampu bersaing di tengah dinamika pasar yang semakin cepat.

Meskipun digitalisasi menawarkan berbagai peluang, UMKM masih menghadapi sejumlah tantangan dalam proses adopsinya. Tantangan tersebut meliputi rendahnya literasi digital, keterbatasan pemahaman dalam pemanfaatan teknologi informasi, serta kurangnya kemampuan dalam mengelola dan menganalisis data bisnis. Selain itu, salah satu kendala utama yang dihadapi UMKM adalah keterbatasan akses terhadap sumber daya yang dibutuhkan untuk mengembangkan usaha, seperti modal, informasi, dan teknologi (Alviani et al., 2025).

Permasalahan digitalisasi UMKM juga diperkuat oleh faktor demografis dan geografis. Menurut (Sofyan & Agusman, 2025), rendahnya literasi digital, khususnya pada pelaku UMKM usia lanjut dan yang berada di wilayah terpencil, menjadi hambatan dalam proses transformasi digital. Selain itu, tidak semua UMKM memiliki perangkat pendukung dan akses jaringan internet yang memadai, sehingga proses digitalisasi belum dapat diterapkan secara merata.

Dalam konteks pemasaran, pemanfaatan teknologi digital, khususnya media sosial, telah menjadi salah satu sarana utama bagi UMKM untuk mempromosikan produk dan menjangkau konsumen secara lebih luas. Aktivitas pemasaran digital tersebut menghasilkan data interaksi konsumen dalam jumlah besar yang berpotensi memberikan informasi berharga mengenai persepsi dan preferensi pasar. Oleh karena itu, UMKM membutuhkan sistem informasi yang mampu mengolah dan menyajikan data tersebut secara terstruktur dan informatif. Keberadaan sistem informasi berbasis dashboard analitik menjadi penting untuk mendukung pengambilan keputusan berbasis data serta meningkatkan efektivitas strategi pemasaran UMKM di era digital (Trulline, 2021).

### **2.1.2 Media Sosial sebagai Sumber Data**

Media sosial telah berkembang tidak hanya sebagai sarana komunikasi dan pemasaran, tetapi juga sebagai sumber data yang mencerminkan opini, persepsi, dan perilaku konsumen. Melalui media sosial, konsumen dapat mengekspresikan pengalaman serta keterlibatan mereka terhadap suatu merek melalui berbagai bentuk interaksi, seperti komentar, unggahan, tanda suka, dan aktivitas berbagi konten. Interaksi tersebut mencerminkan keterlibatan konsumen dan menghasilkan data yang bernalih untuk dianalisis lebih lanjut (Mardhatilah et al., 2024). Data yang dihasilkan dari media sosial umumnya bersifat tidak terstruktur dan terus bertambah secara dinamis, sehingga memerlukan sistem yang mampu mengelola dan menyajikan informasi tersebut secara terstruktur agar dapat dimanfaatkan secara optimal.

Data yang dihasilkan dari media sosial memiliki karakteristik bersifat tidak terstruktur, berjumlah besar, dan terus bertambah secara dinamis. Komentar dan ulasan konsumen umumnya berbentuk teks bebas yang mengandung opini subjektif, emosi, serta penilaian terhadap suatu produk atau layanan. Kondisi ini menyebabkan data media sosial sulit dianalisis secara manual. Oleh karena itu, diperlukan pendekatan sistematis untuk mengolah dan mengekstraksi informasi penting dari data tersebut agar dapat dimanfaatkan secara optimal.

Dalam konteks UMKM, data media sosial berpotensi memberikan wawasan penting terkait preferensi konsumen, tingkat kepuasan pelanggan, serta isu-isu yang sering muncul dalam interaksi publik. Informasi ini dapat dimanfaatkan sebagai dasar evaluasi strategi pemasaran dan pengambilan keputusan bisnis. Namun, tanpa dukungan sistem yang mampu mengelola dan menyajikan data secara terstruktur, potensi data media sosial tersebut sulit dimanfaatkan secara efektif.

Oleh karena itu, media sosial diposisikan sebagai salah satu sumber data utama dalam pengembangan sistem analitik bagi UMKM. Data yang diperoleh dari media sosial selanjutnya dapat diolah dan disajikan dalam bentuk informasi yang lebih ringkas dan mudah dipahami melalui dashboard analitik. Pendekatan ini memungkinkan UMKM untuk memantau persepsi konsumen dan dapat digunakan sebagai alat pengambilan keputusan.

### **2.1.3 Analisis Sentimen pada Media Sosial**

Analisis sentimen merupakan pendekatan analitik yang digunakan untuk mengidentifikasi dan mengklasifikasikan opini atau sikap pengguna terhadap suatu objek, seperti produk, layanan, atau merek, berdasarkan data teks. Dalam konteks media sosial, analisis sentimen memanfaatkan komentar, ulasan, dan berbagai bentuk interaksi pengguna untuk menentukan kecenderungan sentimen yang umumnya dikategorikan ke dalam sentimen

positif, negatif, atau netral. Pendekatan ini relevan karena media sosial menyediakan data opini konsumen yang bersifat spontan, terbuka, dan dihasilkan secara terus-menerus melalui interaksi pengguna.

Seiring dengan meningkatnya aktivitas pengguna di media sosial, volume data opini yang dihasilkan juga semakin besar dan bersifat dinamis. Data tersebut umumnya tidak terstruktur dan mengandung unsur subjektivitas, emosi, serta bahasa informal, sehingga sulit dianalisis secara manual. Oleh karena itu, analisis sentimen digunakan untuk menyederhanakan data teks yang kompleks menjadi informasi yang lebih ringkas dan terstruktur, sehingga dapat digunakan untuk memahami kecenderungan opini publik dan perilaku konsumen secara lebih sistematis.

Penelitian yang dilakukan oleh (Fajarini et al., 2025) menunjukkan bahwa analisis sentimen berbasis data media sosial merupakan metode yang inovatif dan efektif dalam memprediksi tren pasar serta memahami perilaku konsumen. Dengan memanfaatkan data berskala besar yang dihasilkan secara real time oleh pengguna media sosial, analisis sentimen mampu mengenali pola opini publik, preferensi konsumen, serta perubahan tren yang relevan di berbagai sektor industri. Hasil penelitian tersebut juga menegaskan bahwa informasi yang dihasilkan dari analisis sentimen dapat memberikan wawasan yang bernilai untuk mendukung pengambilan keputusan bisnis berbasis data.

Lebih lanjut, (Fajarini et al., 2025) menyatakan bahwa analisis sentimen berbasis media sosial menawarkan pendekatan yang lebih cepat dan efisien dibandingkan metode konvensional, seperti survei manual atau riset pasar tradisional, karena memungkinkan pemantauan opini publik secara berkelanjutan. Dengan demikian, analisis sentimen dipandang sebagai alat yang penting dalam mendukung strategi bisnis modern yang berbasis data dan responsif terhadap dinamika pasar.

Dalam penelitian ini, analisis sentimen diposisikan sebagai proses pengolahan data yang dilakukan pada sisi backend. Fokus penelitian tidak terletak pada metode atau algoritma analisis sentimen yang digunakan, melainkan pada pemanfaatan hasil analisis sentimen sebagai sumber data yang dikelola dan disajikan melalui dashboard analitik di sisi frontend. Hasil analisis sentimen tersebut digunakan sebagai input utama untuk mendukung penyajian informasi yang informatif, terstruktur, dan mudah dipahami oleh pengguna.

#### **2.1.4 Dashboard Analitik dan Visualisasi Data**

Dashboard analitik merupakan sistem informasi yang dirancang untuk menyajikan data dalam bentuk visual yang ringkas, terintegrasi, dan mudah dipahami oleh pengguna. Dashboard

ini umumnya memanfaatkan berbagai komponen visualisasi, seperti grafik, tabel, dan indikator kinerja, untuk menampilkan informasi penting yang mendukung proses pemantauan, analisis, dan pengambilan keputusan. Berbeda dengan laporan statis, dashboard analitik bersifat dinamis dan interaktif, sehingga memungkinkan pengguna untuk memperoleh gambaran kondisi secara cepat dan menyeluruh.

Visualisasi data memiliki peran penting dalam dashboard analitik karena mampu mengubah data mentah menjadi informasi yang lebih bermakna dan intuitif. Melalui visualisasi yang tepat, pengguna dapat dengan mudah mengidentifikasi pola, tren, serta perubahan yang terjadi pada data. Visualisasi data juga berfungsi sebagai sarana penyampaian informasi yang bersifat naratif, di mana kinerja dan kondisi bisnis dapat digambarkan secara komprehensif tanpa harus melalui proses analisis data yang kompleks.

Penelitian yang dilakukan oleh (Rathore et al., 2025) menunjukkan bahwa penerapan teknik visualisasi data yang efektif dapat membantu pengambil keputusan dalam menghasilkan keputusan yang lebih informatif, akurat, dan ringkas. Studi tersebut menegaskan bahwa dashboard, berbagai jenis grafik, serta pendekatan visualisasi yang terstruktur berperan penting dalam meningkatkan kualitas business intelligence dan mendukung perencanaan strategis. Selain itu, visualisasi data dinilai mampu meningkatkan efisiensi proses pengambilan keputusan dan mengurangi waktu yang dibutuhkan untuk menganalisis data, sehingga organisasi dapat merespons permasalahan dan dinamika bisnis secara lebih cepat.

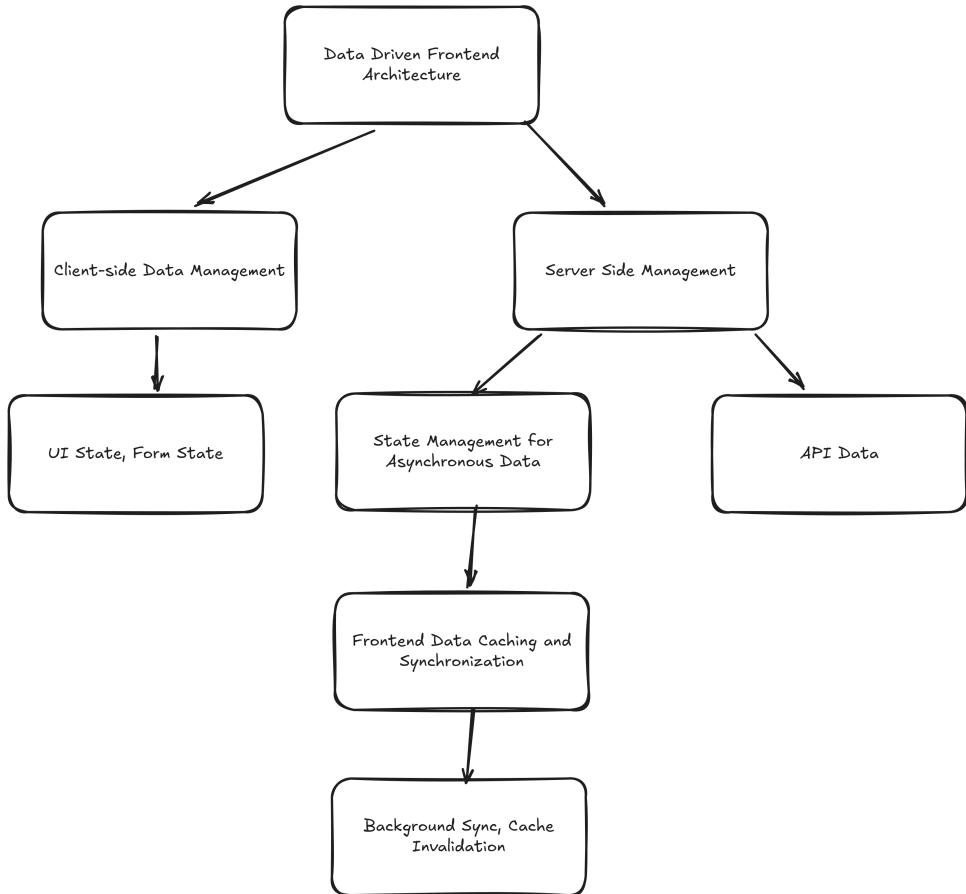
Dalam konteks aplikasi data-driven, dashboard analitik berfungsi sebagai jembatan antara data dan pengambilan keputusan. Dashboard memungkinkan penyajian data secara real time atau near real time, sehingga perubahan kondisi dan tren dapat dipantau secara berkelanjutan. Hal ini menjadikan dashboard analitik sebagai komponen penting dalam sistem yang memanfaatkan data berskala besar dan bersifat dinamis, termasuk data yang dihasilkan dari media sosial.

Bagi UMKM, keberadaan dashboard analitik menjadi sangat relevan karena dapat menyederhanakan informasi yang kompleks menjadi tampilan visual yang mudah dipahami. Informasi seperti kecenderungan sentimen konsumen, pola interaksi pengguna, dan ringkasan data pemasaran dapat disajikan secara visual, sehingga pelaku UMKM tidak perlu melakukan analisis data secara manual. Dengan demikian, dashboard analitik dapat mendukung UMKM dalam mengambil keputusan bisnis yang lebih cepat, tepat, dan berbasis data.

### **2.1.5 Arsitektur Client Data Layer**

Pada aplikasi frontend modern yang bersifat data-driven, pengelolaan data yang bersumber dari Application Programming Interface (API) menjadi salah satu aspek penting dalam arsitektur sistem. Aplikasi seperti dashboard analitik umumnya menampilkan berbagai komponen antarmuka, seperti grafik, tabel, dan indikator, yang bergantung pada data yang sama dan diperbarui secara berkala. Jika pengambilan dan pengolahan data dilakukan secara langsung di setiap komponen antarmuka, maka dapat menimbulkan berbagai permasalahan, seperti permintaan API yang berulang, inkonsistensi data antar-komponen, serta peningkatan beban render yang berdampak pada penurunan performa aplikasi.

Client Data Layer merupakan pendekatan arsitektural dalam pengembangan frontend modern yang bertujuan untuk mengelola data yang bersumber dari server secara terpusat di sisi klien. Pendekatan ini muncul sebagai respons terhadap meningkatnya kompleksitas aplikasi data-driven, dimana data bersifat asinkron, dinamis, dan digunakan oleh banyak komponen antarmuka secara bersamaan. Dengan adanya Client Data Layer, proses fetching, chaching, dan synchronize data dapat dilakukan secara lebih terstruktur, sehingga membantu menjaga konsistensi data dan sehingga membantu menjaga konsistensi data dan mendukung pengelolaan data frontend secara lebih terstruktur.



**Gambar 2.1** Architecture Client Data Layer

## 1. Server State Management

Server state management mengacu pada pengelolaan data yang bersumber dari sistem eksternal, seperti API atau layanan backend, yang bersifat asinkron dan dapat berubah di luar kendali langsung aplikasi klien. Data ini memiliki karakteristik dinamis karena dipengaruhi oleh kondisi jaringan, waktu respons server, serta pembaruan data di sisi backend. Oleh karena itu, server state memerlukan mekanisme khusus untuk menangani proses pengambilan data, status pemuatan, penanganan kesalahan, serta pembaruan dan sinkronisasi data agar informasi yang digunakan oleh berbagai komponen antarmuka tetap konsisten dan akurat.

## 2. Client-side Data Management

Client-side data management berfokus pada pengelolaan data di sisi klien setelah data tersebut diperoleh dari server, termasuk penyimpanan sementara, penggunaan ulang data, serta distribusi data ke berbagai komponen antarmuka. Pendekatan ini bertujuan untuk mengurangi ketergantungan terhadap permintaan data berulang ke server, sehingga dapat mengurangi ketergantungan terhadap permintaan data

berulang ke server dan menjaga konsistensi informasi yang ditampilkan. Dengan pengelolaan data yang terstruktur di sisi klien, aplikasi frontend dapat menyajikan data secara lebih responsif dan stabil, khususnya pada aplikasi yang bersifat data-driven seperti dashboard analitik.

### 3. Data-driven Frontend Architecture

Data-driven Frontend Architecture merupakan pendekatan pengembangan di mana seluruh antarmuka pengguna didorong oleh data sebagai sumber kebenaran utama. Dalam pola ini, UI dihasilkan sebagai fungsi dari state yang ada—artinya, setiap perubahan pada data akan secara otomatis memicu pembaruan pada tampilan aplikasi. Arsitektur ini sangat berguna untuk aplikasi yang menampilkan informasi dinamis seperti dashboard analitik, papan monitoring, atau aplikasi dengan data real-time. Keunggulan utamanya adalah konsistensi tampilan yang lebih terjamin, alur data yang mudah dilacak, dan pengembangan fitur baru yang lebih sistematis karena UI dan logika data terpisah dengan jelas.

### 4. State Management for Asynchronous Data

State Management for Asynchronous Data adalah pendekatan khusus untuk menangani data yang diperoleh melalui state, seperti panggilan API, operasi file, atau permintaan jaringan lainnya. Karena data tersebut tidak tersedia secara instan, sistem harus mampu mengelola berbagai state yang mungkin terjadi: mulai dari state (idle), state fetching atau onloading, state success, hingga state error. Tantangan utamanya adalah memastikan aplikasi tetap responsif dan memberikan umpan balik yang informatif kepada pengguna selama proses pengambilan data. Pendekatan ini juga mencakup strategi seperti pembatalan permintaan yang tidak diperlukan, pengulangan otomatis saat gagal, dan pembaruan data latar belakang untuk menjaga informasi tetap konsisten.

### 5. Frontend Data Caching and Synchronization

Data Caching and Synchronization adalah teknik untuk meningkatkan kinerja aplikasi dengan menyimpan salinan data dari server di memori klien. Dengan adanya cache, aplikasi dapat menampilkan informasi secara cepat tanpa perlu melakukan permintaan berulang ke server. Namun, teknik ini juga menimbulkan tantangan, yaitu data yang disimpan dapat menjadi kedaluwarsa jika terjadi perubahan di sisi server. Oleh karena itu, diperlukan mekanisme sinkronisasi yang cerdas, seperti pembaruan di latar belakang (background refresh) dan penandaan cache yang kedaluwarsa (cache invalidation). Dengan pendekatan ini, aplikasi

dapat menampilkan data secara lebih konsisten sekaligus menjaga keakuratan informasi yang ditampilkan.

Beberapa penelitian menunjukkan bahwa penerapan mekanisme caching pada aplikasi frontend dapat meningkatkan performa dan responsivitas sistem. Caching memungkinkan data hasil pemanggilan API disimpan sementara di sisi klien sehingga dapat digunakan kembali tanpa melakukan permintaan ulang ke server. Pendekatan ini terbukti mampu mengurangi duplikasi permintaan API dan mempercepat waktu respons aplikasi. Pemanfaatan pustaka seperti TanStack Query dalam mengelola caching dan prefetching data juga dinilai efektif dalam meningkatkan efisiensi pengelolaan data di sisi frontend serta pengalaman pengguna secara keseluruhan (Rahman & Prihanto, 2024). Dalam konteks penelitian ini, peningkatan performa dipahami sebagai perubahan perilaku pengelolaan data frontend yang diamati melalui mekanisme caching dan pengendalian permintaan data, tanpa dilakukan pengukuran performa numerik.

Penerapan Client Data Layer memberikan sejumlah manfaat dalam pengembangan aplikasi frontend. Salah satu manfaat utama adalah peningkatan konsistensi data, di mana beberapa komponen yang membutuhkan data yang sama dapat memperoleh informasi yang seragam tanpa harus melakukan permintaan data secara terpisah. Selain itu, Client Data Layer memungkinkan pengurangan jumlah permintaan API yang tidak diperlukan melalui mekanisme caching dan pengelolaan siklus data. Pendekatan ini juga mendukung mendukung penyajian data yang lebih stabil dan terkelola serta mempermudah pengelolaan data yang bersifat asinkron dan dinamis.

Dalam konteks dashboard analitik, keberadaan Client Data Layer menjadi semakin penting karena data yang ditampilkan umumnya bersifat besar, sering diperbarui, dan digunakan oleh banyak komponen secara bersamaan. Dengan memanfaatkan Client Data Layer, dashboard dapat menampilkan data secara lebih responsif dan stabil, sekaligus meminimalkan risiko inkonsistensi informasi yang ditampilkan kepada pengguna. Pendekatan ini mendukung terciptanya arsitektur frontend yang lebih terorganisasi, mudah dipelihara, dan skalabel.

### **2.1.6 TanStack Query (React Query)**

TanStack Query merupakan pustaka manajemen data pada sisi frontend yang dirancang untuk mengelola data yang bersumber dari server (server state) secara efisien. Dalam dokumentasi resminya, TanStack Query dijelaskan sebagai “the missing data-fetching layer for

“web applications” yang berfungsi untuk mempermudah proses pengambilan, penyimpanan sementara (caching), sinkronisasi, serta pembaruan data dari server (Tanstack LCC, 2025). Pendekatan ini ditujukan untuk menangani kompleksitas data asinkron yang tidak dapat dikelola secara optimal menggunakan mekanisme state management konvensional.

Dalam dokumentasi resminya, TanStack Query mendefinisikan server state sebagai data yang berasal dari sumber eksternal dan memiliki karakteristik asinkron, dapat berubah sewaktu-waktu, serta memerlukan mekanisme khusus untuk menjaga konsistensi data di sisi klien. Oleh karena itu, TanStack Query menyediakan pendekatan deklaratif dalam pengelolaan server state, di mana pengembang dapat mendefinisikan kebutuhan data tanpa harus menangani secara manual proses sinkronisasi dan pembaruan data di setiap komponen antarmuka (TanStack Documentation, 2024).

Salah satu fitur utama TanStack Query adalah mekanisme caching yang memungkinkan data hasil pemanggilan API disimpan sementara di sisi klien. Dengan adanya caching, data yang telah diperoleh dapat digunakan kembali oleh komponen lain tanpa perlu melakukan permintaan ulang ke server, selama data tersebut masih dianggap valid. Pendekatan ini berkontribusi dalam mengurangi jumlah permintaan API yang tidak diperlukan, meningkatkan efisiensi aplikasi, serta mempercepat waktu respons antarmuka pengguna.

Selain caching, TanStack Query juga menyediakan mekanisme sinkronisasi data yang mendukung pembaruan data secara otomatis. Melalui konsep seperti refetching dan invalidasi data, TanStack Query memastikan bahwa data yang ditampilkan tetap mutakhir ketika terjadi perubahan di sisi server. Mekanisme ini sangat relevan pada aplikasi data-driven, seperti dashboard analitik, yang menampilkan data secara dinamis dan digunakan oleh banyak komponen secara bersamaan.

Dalam konteks arsitektur frontend, TanStack Query dapat diposisikan sebagai implementasi konkret dari Client Data Layer. Pustaka ini berperan sebagai lapisan perantara antara backend API dan komponen antarmuka pengguna, sehingga komponen UI tidak berinteraksi langsung dengan API. Dengan demikian, TanStack Query membantu memisahkan logika pengelolaan data dari logika tampilan, meningkatkan keterbacaan kode, serta mempermudah pemeliharaan aplikasi dalam jangka panjang.

Pada penelitian ini, TanStack Query digunakan sebagai solusi untuk menerapkan arsitektur Client Data Layer pada pengembangan dashboard analitik. Fokus penggunaan TanStack Query diarahkan pada pengelolaan server state, caching data, serta sinkronisasi data antar-komponen, tanpa membahas aspek internal pustaka atau detail implementasi secara mendalam. Dengan pendekatan ini, TanStack Query berperan sebagai fondasi pengelolaan

data pada sisi frontend yang mendukung penyajian informasi sentimen secara konsisten, responsif, dan efisien.

### 2.1.7 REST API

Representational State Transfer Application Programming Interface (REST API) merupakan gaya arsitektur layanan web yang digunakan untuk memungkinkan komunikasi antara klien dan server melalui protokol HTTP secara terstandarisasi. REST API bersifat stateless, di mana setiap permintaan dari klien harus membawa seluruh informasi yang dibutuhkan untuk diproses oleh server, sehingga tidak bergantung pada status permintaan sebelumnya. Data yang dipertukarkan umumnya disajikan dalam format JSON karena bersifat ringan dan mudah diproses oleh aplikasi frontend, menjadikan REST API banyak digunakan pada aplikasi web modern yang bersifat data-driven.

Dalam konteks aplikasi frontend analitik, REST API berperan sebagai sumber utama data (server state) yang dikonsumsi oleh antarmuka pengguna. Data yang diperoleh melalui REST API bersifat asinkron dan dapat berubah sewaktu-waktu, sehingga memerlukan mekanisme pengelolaan data yang mampu menangani proses pengambilan, pembaruan, dan sinkronisasi data secara efisien. Penelitian terkini menunjukkan bahwa REST API memiliki keunggulan dalam penyajian data yang bersifat datar dan mudah di-cache, sehingga mampu meningkatkan efisiensi distribusi data serta memperbesar rasio cache hit pada lapisan jaringan. Pendekatan ini dinilai lebih optimal untuk kebutuhan aplikasi yang menampilkan data terstruktur secara berulang, seperti dashboard dan sistem pelaporan, dibandingkan pendekatan API lain yang lebih kompleks (Islam, 2025).

Lebih lanjut, penelitian tersebut menegaskan bahwa performa aplikasi web tidak ditentukan oleh satu teknologi tertentu, melainkan oleh keselarasan antara desain akses data, mekanisme caching, serta pengelolaan state pada sisi klien. REST API yang dirancang dengan kontrak data yang jelas dan cache-aware terbukti mendukung peningkatan performa dan skalabilitas aplikasi ketika dipadukan dengan lapisan pengelolaan data di frontend. Oleh karena itu, dalam pengembangan aplikasi frontend modern, REST API umumnya tidak diakses secara langsung oleh setiap komponen antarmuka, melainkan melalui lapisan pengelolaan data seperti Client Data Layer agar data dapat dikelola secara terpusat, konsisten, dan efisien.

Dalam penelitian ini, REST API diposisikan sebagai penyedia data hasil analisis sentimen yang diproses di sisi backend. Data tersebut selanjutnya dikelola pada sisi frontend melalui arsitektur Client Data Layer sebelum ditampilkan dalam bentuk visualisasi pada dashboard analitik. Dengan pemisahan peran ini, REST API berfungsi sebagai sumber data,

sementara pengelolaan performa, caching, dan sinkronisasi data dilakukan sepenuhnya di sisi frontend untuk mendukung penyajian informasi yang responsif dan konsisten.

## 2.2 Penelitian Terkait

Berikut adalah tabel perbandingan penelitian terkait yang relevan dengan pengembangan penerapan arsitektur Client Data Layer menggunakan TanStack Query pada dashboard sentiment analysis

**Tabel 2.1** Tabel Perbandingan Penelitian Terkait

No	Peneliti	Teknologi	Judul	Fitur
1	Fajarini, Sri Dwi; Kurniawati, Juliana; Yuliani, Fitria (2025)	NLP, Machine Learning (SVM, Random Forest, VADER)	<i>Social Media Sentiment Analysis as a New Tool for Predicting Market Trends and Consumer Behaviour</i>	Analisis sentimen media sosial untuk mengidentifikasi pola opini publik dan memprediksi perilaku konsumen serta tren pasar.
2	Shrutika Rathore; Rahul Nawkhare; Navin Sharma; Nitin Chaudhary; Saurabh Chakole; Bhaskar Vishwakrama (2025)	Dashboard analitik, visualisasi data, business intelligence	<i>Effective Data Visualization Techniques for Business Decision-Makers</i>	Penyajian data bisnis melalui dashboard analitik untuk meningkatkan efisiensi pengambilan keputusan dan perencanaan strategis.
3	Micheal, Author Dave (2024)	React, React Query (TanStack Query), lazy loading, caching	<i>React Query and Lazy Loading: Performance Optimization Best Practices</i>	Optimasi performa aplikasi frontend melalui pengelolaan data asinkron, caching, dan lazy loading untuk mengurangi permintaan API berulang.

## 2.3 Analisa Gap Penelitian

### 2.3.1 Gap Penelitian 1

Penelitian ini berfokus pada pemanfaatan analisis sentimen media sosial menggunakan pendekatan Natural Language Processing (NLP) dan machine learning untuk memprediksi perilaku konsumen dan tren pasar. Namun demikian, penelitian ini belum membahas bagaimana hasil analisis sentimen tersebut dikelola dan disajikan pada sisi frontend, khususnya dalam bentuk dashboard analitik. Aspek arsitektur pengelolaan data frontend, seperti mekanisme pengambilan data dari API, caching, serta sinkronisasi data antar-komponen, belum menjadi fokus dalam penelitian ini.

### 2.3.2 Gap Penelitian 2

Penelitian ini menekankan pada peran dashboard dan teknik visualisasi data dalam meningkatkan efektivitas pengambilan keputusan bisnis. Fokus utama penelitian berada pada desain visualisasi, jenis grafik, serta manfaat dashboard analitik bagi pengambil keputusan. Namun, penelitian ini belum mengkaji bagaimana data yang ditampilkan pada dashboard

dikelola di sisi frontend, khususnya terkait arsitektur pengambilan data, pengelolaan server state, serta mekanisme caching dan konsistensi data pada aplikasi frontend modern.

### **2.3.3 Gap Penelitian 3**

Penelitian ini membahas penggunaan React Query dalam pengelolaan data asinkron pada aplikasi React, serta pemanfaatan lazy loading dan caching untuk meningkatkan performa aplikasi frontend. Meskipun penelitian ini menunjukkan bahwa React Query efektif dalam mengurangi pemanggilan API berulang dan meningkatkan waktu respons aplikasi, konteks penerapannya masih bersifat umum. Penelitian ini belum membahas penerapan React Query sebagai bagian dari arsitektur Client Data Layer pada aplikasi dashboard analisis sentimen, serta belum mengaitkan pengelolaan server state dengan kebutuhan penyajian data analitik pada konteks UMKM.

## **BAB 3**

### **METODOLOGI PENELITIAN**

#### **3.1 Waktu dan Jadwal penelitian**

##### **3.1.1 Waktu Pelaksanaan Penelitian**

Waktu pelaksanaan penelitian ini direncanakan selama 5 bulan, yaitu dimulai pada bulan September 2025 dan berakhir pada bulan Januari 2026.

##### **3.1.2 Jadwal Kegiatan Penelitian**

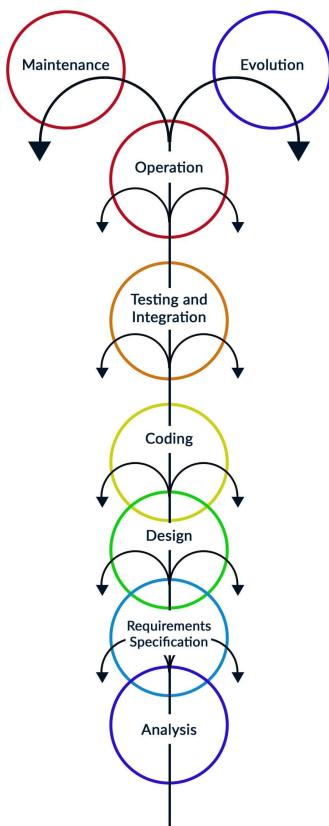
Berikut adalah serangkaian jadwal kegiatan yang dilakukan selama pelaksanaan penelitian ini, yang diuraikan pada Tabel 3.1.

**Tabel 3.1** Jadwal Kegiatan Penelitian

No	Nama Kegiatan	Bulan				
		September	Oktober	November	Desember	Januari
1	Analisis Kebutuhan Sistem					
2	Perancangan arsitektur frontend dan Client Data Layer					
3	Desain					
4	Implementasi frontend dan TanStack Query					
5	Analisis hasil pengujian dan penyusunan laporan					

### 3.2 Methode Penelitian

#### 3.2.1 Metode Fountain



**Gambar 3.1** Fountain SDLC Model

Metode Fountain merupakan model pengembangan perangkat lunak yang bersifat iteratif dan fleksibel, yang dirancang sebagai alternatif terhadap model pengembangan linear seperti Waterfall. Pada metode ini, tahapan pengembangan tidak harus diselesaikan secara kaku dan berurutan, melainkan dapat saling tumpang tindih dan dilakukan secara berulang sesuai kebutuhan. Pendekatan tersebut memungkinkan proses pengembangan sistem berjalan lebih adaptif terhadap perubahan kebutuhan yang muncul selama penelitian.

Karakteristik utama metode Fountain terletak pada kemampuannya untuk memungkinkan setiap tahapan kembali ke tahap sebelumnya tanpa harus mengulang keseluruhan proses pengembangan. Jika pada tahap implementasi ditemukan ketidaksesuaian dengan kebutuhan sistem, maka proses dapat kembali ke tahap analisis atau perancangan untuk dilakukan penyesuaian. Dengan mekanisme ini, metode Fountain mendukung pengembangan sistem yang membutuhkan fleksibilitas dan evaluasi berkelanjutan.

Metode Fountain banyak digunakan dalam pengembangan perangkat lunak dengan

kompleksitas menengah hingga tinggi, khususnya pada sistem yang kebutuhan fungsionalnya dapat berkembang dan berubah selama proses implementasi. Metode ini dinilai sesuai dalam konteks penelitian rekayasa perangkat lunak karena mampu mengakomodasi perubahan desain dan penyesuaian teknis yang sering terjadi tanpa mengganggu keseluruhan proses pengembangan sistem.

### **3.2.2 Alasan Pemilihan Metode Fountain**

Pemilihan metode Software Development Life Cycle (SDLC) yang tepat merupakan langkah strategis yang krusial untuk memastikan keberhasilan proyek serta mencegah pembengkakan biaya dan waktu pengembangan (Aniley et al., 2024). Berdasarkan pertimbangan tersebut, penelitian ini mengadopsi metode Fountain yang dinilai relevan karena memiliki karakteristik fleksibel namun tetap terstruktur. Efektivitas metode ini didukung oleh penelitian (Sastra & Sutawinata, 2023) pada perancangan sistem SIP-PTK, yang menunjukkan bahwa model Fountain mampu memandu tahapan analisis, desain, implementasi, hingga pengujian secara efektif pada sistem yang memiliki kebutuhan pengolahan data yang spesifik.

Relevansi metode Fountain juga sejalan dengan fokus penelitian ini, yaitu pengembangan aplikasi frontend yang bersifat data-driven dan memiliki ketergantungan tinggi terhadap interaksi antar-komponen. Dalam pengembangan frontend, tahapan perancangan arsitektur, implementasi, dan pengujian tidak berjalan secara linier, melainkan saling berkaitan dan sering memerlukan penyesuaian berdasarkan hasil evaluasi sistem. Penerapan metode Fountain memungkinkan tahapan pengembangan dan evaluasi berjalan secara paralel serta saling tumpang tindih (overlapping) tanpa menghilangkan struktur dan urutan penelitian yang jelas.

Melalui pendekatan ini, aktivitas analisis kebutuhan, perancangan arsitektur frontend, implementasi Client Data Layer, serta pengujian sistem dapat dilakukan secara berulang dan saling mempengaruhi. Hasil evaluasi pada satu tahapan dapat langsung digunakan sebagai dasar penyesuaian pada tahapan lainnya, seperti perubahan struktur data atau mekanisme sinkronisasi, tanpa harus menunggu seluruh siklus pengembangan selesai. Dengan demikian, proses pengembangan sistem menjadi lebih adaptif dan terkontrol, sehingga diharapkan mampu menghasilkan hasil penelitian yang sesuai dengan tujuan penelitian secara optimal.

### **3.2.3 Alasan Pemilihan Metode Fountain**

Tahapan penerapan metode fountain dalam penelitian ini terdiri atas beberapa tahap sebagai berikut:

1. Analysis

Tahap analisis dilakukan untuk memahami permasalahan dan kebutuhan sistem berdasarkan latar belakang dan rumusan masalah penelitian. Pada tahap ini dilakukan identifikasi kebutuhan dashboard analisis sentimen UMKM serta karakteristik data yang akan ditampilkan pada sisi frontend.

## 2. Requirement Specification

Tahap ini bertujuan untuk merumuskan kebutuhan sistem secara lebih terperinci, khususnya kebutuhan fungsional dan nonfungsional frontend. Spesifikasi kebutuhan difokuskan pada pengelolaan data API, konsistensi data antar-komponen, serta kebutuhan performa tampilan dashboard.

## 3. Design

Tahap perancangan mencakup perancangan arsitektur frontend dan Client Data Layer menggunakan TanStack Query, serta desain antarmuka pengguna. Perancangan ini bertujuan untuk menentukan struktur sistem sebelum proses implementasi dilakukan.

## 4. Coding (Implementation)

Tahap implementasi dilakukan dengan merealisasikan hasil perancangan ke dalam kode program frontend berbasis React. Pada tahap ini, TanStack Query diimplementasikan untuk mengelola server state, caching data, dan sinkronisasi data antar-komponen dashboard.

## 5. Testing

Tahap pengujian dilakukan untuk memastikan bahwa sistem berjalan sesuai dengan kebutuhan yang telah ditetapkan. Pengujian dilakukan menggunakan pendekatan blackbox testing dengan mengamati perilaku sistem melalui log permintaan data dan mekanisme caching.

## 6. Operation

Tahap operasi mencakup penggunaan sistem dalam lingkungan pengujian untuk memastikan aplikasi dapat berjalan dengan baik dan stabil setelah implementasi.

## 7. Maintenance

Tahap pemeliharaan dilakukan untuk memperbaiki kesalahan minor dan melakukan penyesuaian teknis apabila ditemukan kendala selama penggunaan sistem.

## 8. Evaluation

Tahap evaluasi dilakukan untuk menilai hasil penerapan arsitektur Client Data Layer menggunakan TanStack Query terhadap tujuan penelitian, khususnya terkait efisiensi pengelolaan data dan konsistensi data pada dashboard.

### **3.3 Perancangan Sistem**

#### **3.3.1 Gambaran Umum Sistem**

Sistem yang dikembangkan dalam penelitian ini berupa aplikasi frontend berbasis web yang berfungsi sebagai dashboard analisis sentimen UMKM. Dashboard ini dirancang untuk menyajikan hasil analisis sentimen yang diperoleh dari data media sosial dalam bentuk visualisasi yang informatif dan mudah dipahami oleh pengguna. Informasi yang ditampilkan meliputi ringkasan sentimen, distribusi sentimen, serta indikator lain yang relevan dengan kebutuhan pemantauan persepsi konsumen.

Sistem ini berfokus pada pengelolaan dan penyajian data di sisi frontend, sementara proses pengumpulan data media sosial dan analisis sentimen sepenuhnya dilakukan pada sisi backend. Data hasil analisis sentimen disediakan melalui REST API dan dikonsumsi oleh aplikasi frontend sebagai sumber data utama. Dengan pendekatan ini, frontend diposisikan sebagai lapisan presentasi yang bertanggung jawab dalam mengelola dan menampilkan data secara konsisten kepada pengguna.

Pengembangan sistem difokuskan pada penerapan arsitektur frontend yang mampu mengelola data secara terstruktur dan efisien. Sistem dirancang untuk menangani data yang bersifat dinamis dan digunakan oleh berbagai komponen antarmuka secara bersamaan. Oleh karena itu, diperlukan mekanisme pengelolaan data yang dapat menjaga konsistensi informasi, mengurangi permintaan data yang tidak diperlukan, serta mendukung pembaruan data secara terkontrol.

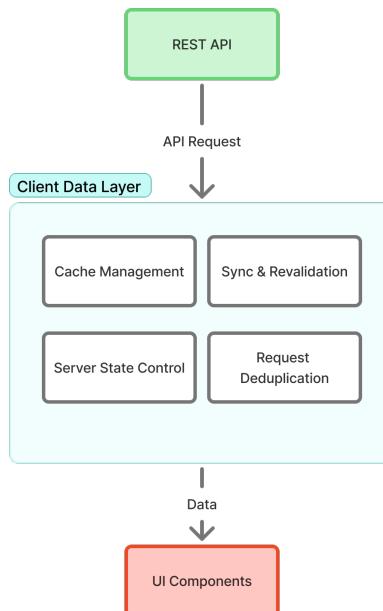
Pengguna sistem dalam penelitian ini adalah pengguna umum atau pelaku UMKM yang memanfaatkan dashboard sebagai alat pemantauan informasi sentimen. Interaksi pengguna dibatasi pada aktivitas pengamatan data dan eksplorasi informasi yang tersedia pada dashboard, tanpa melibatkan proses pengolahan data secara langsung. Dengan ruang lingkup tersebut, sistem difokuskan pada aspek penyajian data dan pengelolaan data frontend sesuai dengan tujuan penelitian.

#### **3.3.2 Perancangan Arsitektur Frontend**

Penelitian ini menerapkan prinsip component-based architecture, di mana antarmuka pengguna dibangun dari komponen-komponen modular yang memiliki tanggung jawab spesifik dan dapat digunakan kembali. Setiap komponen difokuskan pada penyajian data dan interaksi pengguna, sementara logika pengelolaan data dipisahkan ke dalam lapisan tersendiri agar struktur aplikasi lebih terorganisasi dan mudah dipelihara.

Arsitektur frontend dirancang dengan pendekatan data-driven, di mana tampilan

antarmuka sepenuhnya bergantung pada data yang dikelola oleh sistem. Untuk mendukung hal tersebut, prinsip separation of concerns diterapkan dengan memisahkan lapisan presentasi dan lapisan pengelolaan data, sehingga komponen antarmuka tidak berinteraksi langsung dengan REST API. Pendekatan ini mendukung pengembangan frontend yang lebih terstruktur dan fleksibel sesuai dengan metode Fountain yang digunakan dalam penelitian ini.



**Gambar 3.2** Diagram Arsitektur Frontend

Diagram arsitektur frontend pada Gambar 3.2 digunakan untuk menggambarkan pembagian lapisan sistem serta alur pengelolaan data pada sistem yang dikembangkan. Diagram ini menunjukkan bagaimana data dari REST API dikelola melalui Client Data Layer sebelum disajikan pada komponen antarmuka pengguna.

Lapisan REST API diposisikan sebagai sumber data eksternal yang menyediakan data hasil analisis sentimen. Seluruh proses pengolahan data, termasuk pengambilan data media sosial dan analisis sentimen, dilakukan pada sisi backend dan berada di luar ruang lingkup penelitian ini. Frontend berperan sebagai konsumen data yang mengakses informasi tersebut melalui antarmuka REST API.

Lapisan Client Data Layer berfungsi sebagai lapisan perantara antara REST API dan komponen antarmuka pengguna. Data yang diperoleh dari REST API dikelola dan dimodelkan secara terpusat pada Client Data Layer sebelum disajikan pada komponen antarmuka

pengguna. Lapisan ini bertanggung jawab dalam mengelola data yang bersumber dari server secara terpusat, mencakup proses pengambilan data, penyimpanan sementara (caching), serta sinkronisasi data antar-komponen. Dengan adanya Client Data Layer, komponen antarmuka tidak perlu berinteraksi langsung dengan REST API, sehingga pengelolaan data dapat dilakukan secara lebih terstruktur dan konsisten.

Lapisan UI Components merupakan lapisan presentasi yang bertugas menampilkan data kepada pengguna dan menangani interaksi pengguna dengan sistem. Komponen pada lapisan ini menerima data yang telah dikelola oleh Client Data Layer dan menyajikannya dalam bentuk visualisasi seperti grafik dan tabel. Pemisahan peran ini memastikan bahwa komponen antarmuka hanya berfokus pada aspek tampilan dan interaksi, tanpa menangani logika pengelolaan data.

### 3.3.3 Perancangan Client Data Layer

Perancangan Client Data Layer dilakukan untuk mengelola data yang bersumber dari backend secara terpusat pada sisi frontend sebelum disajikan pada komponen antarmuka pengguna. Pada aplikasi dashboard yang bersifat data-driven, data yang sama dapat digunakan oleh berbagai komponen secara bersamaan dan diperbarui secara dinamis. Oleh karena itu, diperlukan suatu lapisan pengelolaan data yang mampu mengatur alur data, menjaga konsistensi informasi, serta mengendalikan interaksi antara frontend dan REST API.

Client Data Layer diposisikan sebagai lapisan perantara antara REST API dan komponen antarmuka pengguna, sebagaimana ditunjukkan pada Gambar 3.2 diagram arsitektur frontend. Seluruh data yang diperoleh dari backend tidak langsung digunakan oleh komponen antarmuka, melainkan terlebih dahulu dikelola melalui Client Data Layer. Dengan pendekatan ini, komponen antarmuka tidak perlu mengetahui detail proses pengambilan data dari API, sehingga fokus komponen dapat diarahkan pada penyajian data dan interaksi pengguna.

Secara konseptual, Client Data Layer memiliki beberapa tanggung jawab utama dalam sistem frontend. Tanggung jawab tersebut meliputi proses pengambilan data dari REST API, pengelolaan server state, penyimpanan sementara data melalui mekanisme caching, serta sinkronisasi data antar-komponen antarmuka. Dengan pengelolaan data yang terpusat, permintaan data yang bersifat berulang dapat dikendalikan dan setiap komponen antarmuka memperoleh data yang konsisten sesuai dengan kondisi sistem.

Selain pengelolaan alur data, Client Data Layer juga dirancang untuk menangani pemodelan data sebelum digunakan oleh komponen antarmuka. Data yang diperoleh dari

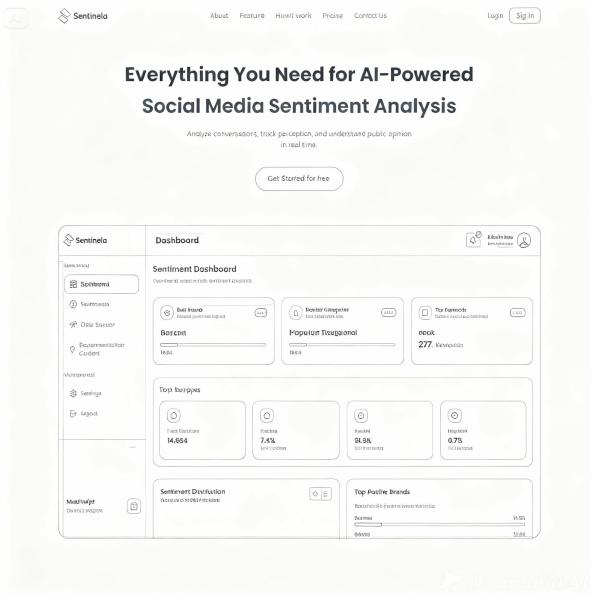
REST API dimodelkan secara terstruktur pada Client Data Layer agar memiliki bentuk dan konsistensi yang jelas. Pendekatan ini bertujuan untuk meminimalkan ketergantungan komponen antarmuka terhadap struktur data mentah dari backend serta mempermudah proses pengembangan dan pemeliharaan sistem frontend.

Dalam penelitian ini, Client Data Layer dirancang untuk diimplementasikan menggunakan TanStack Query sebagai pustaka pengelolaan server state pada frontend. Pemilihan TanStack Query didasarkan pada kemampuannya dalam menyediakan mekanisme pengelolaan data asinkron secara terpusat, termasuk caching, sinkronisasi data, dan pengendalian permintaan data. Dengan memanfaatkan pustaka tersebut, Client Data Layer diharapkan mampu mendukung pengelolaan data frontend yang lebih terstruktur, konsisten, dan efisien sesuai dengan kebutuhan dashboard analisis sentimen.

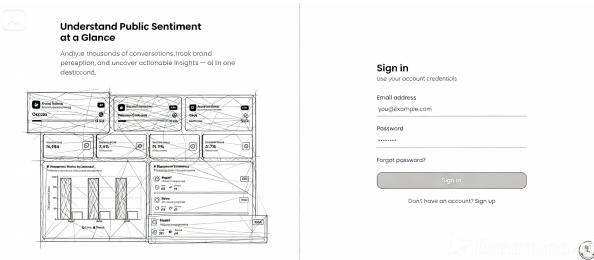
### **3.3.4 Perancangan Penggunaan TanStack Query**

Perancangan penggunaan TanStack Query pada penelitian ini mengacu pada dokumentasi resmi TanStack Query sebagai pustaka server state management untuk aplikasi frontend. Berdasarkan dokumentasi resmi TanStack Query (Tanstack LCC, 2025), pustaka ini dirancang untuk mengelola data asinkron yang bersumber dari API secara terpusat melalui mekanisme pengambilan data, penyimpanan sementara (caching), serta sinkronisasi data antar-komponen antarmuka. Pendekatan tersebut memungkinkan komponen frontend memperoleh data yang konsisten tanpa harus melakukan permintaan data secara langsung ke REST API, sehingga pemisahan tanggung jawab antara lapisan presentasi dan lapisan pengelolaan data dapat terjaga. Dengan karakteristik tersebut, TanStack Query dinilai sesuai untuk diimplementasikan sebagai Client Data Layer pada aplikasi frontend yang bersifat data-driven, khususnya dalam konteks dashboard analisis sentimen yang membutuhkan konsistensi data dan pembaruan informasi secara terkontrol.

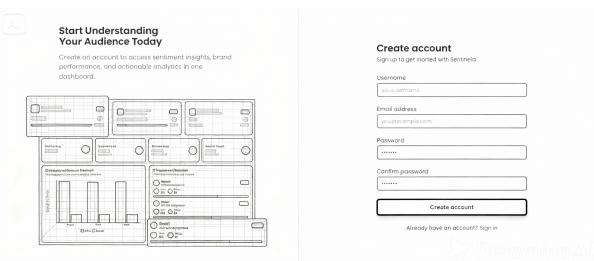
### 3.3.5 Perancangan Antarmuka



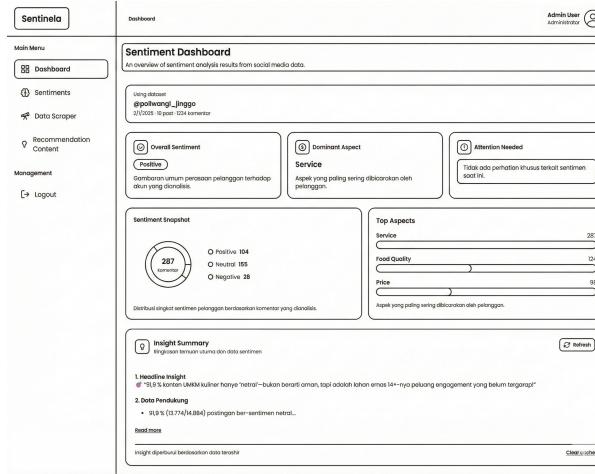
Gambar 3.3 Wireframe Landing Page



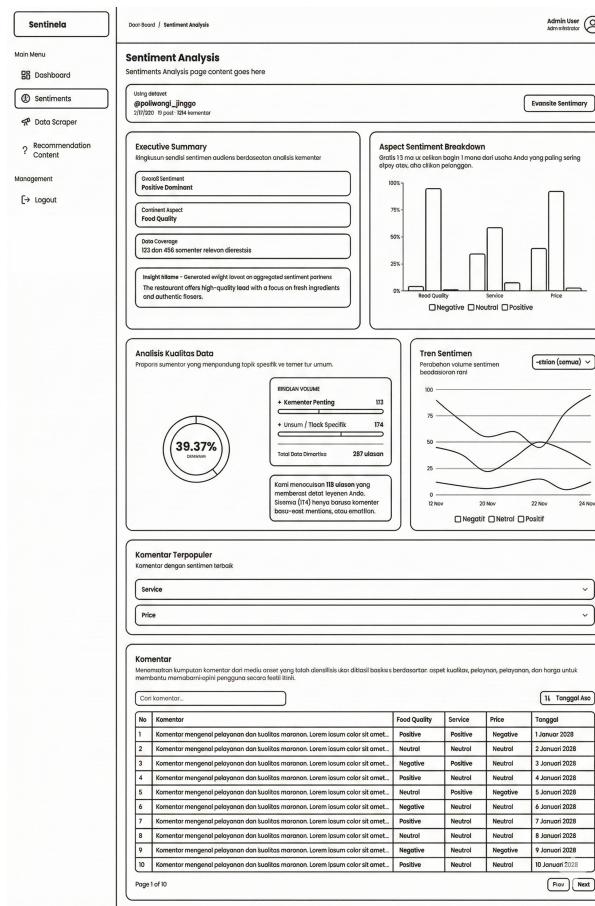
Gambar 3.4 Wireframe Halaman Login



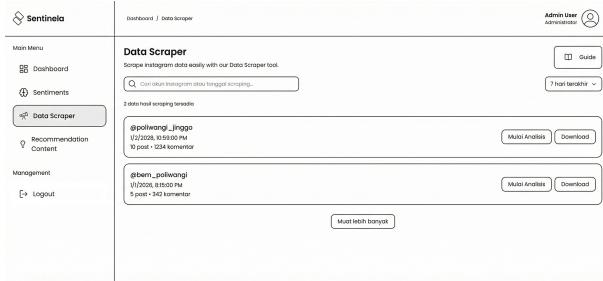
Gambar 3.5 Wireframe Halaman Register



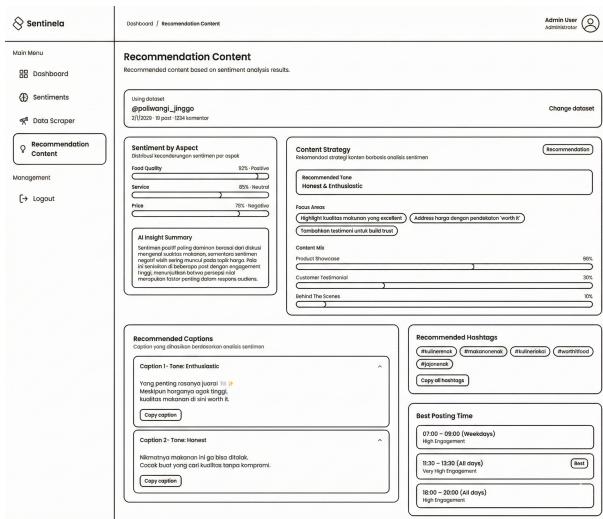
Gambar 3.6 Wireframe Halaman Dashboard



Gambar 3.7 Wireframe Halaman Sentiment



**Gambar 3.8** Wireframe Halaman Scraper



**Gambar 3.9** Wireframe Halaman Recomendation

## 3.4 Metode Pengujian dan Evaluasi Sistem

### 3.4.1 Metode Pengujian

### 3.4.2 Skenario dan Objek Pengujian

### 3.4.3 Teknik Evaluasi

Teknik evaluasi pada penelitian ini dilakukan untuk menilai kesesuaian perilaku sistem front-end terhadap rancangan arsitektur dan Client Data Layer yang telah ditetapkan. Evaluasi dilakukan dengan pendekatan kualitatif-deskriptif, yaitu melalui observasi terhadap perilaku sistem selama proses pengujian berlangsung tanpa melibatkan pengukuran numerik performa secara detail. Pendekatan ini dipilih karena fokus penelitian diarahkan pada perilaku pengelolaan data dan konsistensi tampilan sistem, bukan pada pengujian efisiensi algoritma atau kinerja back-end.

Sumber data evaluasi diperoleh dari hasil observasi terhadap tampilan dashboard serta perilaku pengelolaan data pada sisi front-end. Observasi dilakukan terhadap bagaimana data ditampilkan pada berbagai komponen antarmuka, bagaimana sistem merespons pembaruan data, serta bagaimana konsistensi data terjaga ketika komponen yang berbeda menggunakan

sumber data yang sama. Selain itu, evaluasi juga dilakukan dengan mengamati log permintaan data untuk memastikan bahwa mekanisme pengelolaan data berjalan sesuai dengan rancangan.

Sebagai alat bantu evaluasi, penelitian ini memanfaatkan fitur observasi yang disediakan oleh TanStack Query Devtools. Alat bantu ini digunakan untuk memantau status pengambilan data, mekanisme penyimpanan sementara (caching), serta proses sinkronisasi data antar-komponen selama skenario pengujian dijalankan. Penggunaan alat bantu ini bertujuan untuk mendukung proses observasi perilaku sistem secara lebih terstruktur, tanpa bergantung pada detail implementasi kode program.

Indikator evaluasi dalam penelitian ini meliputi beberapa aspek utama, yaitu konsistensi data antar-komponen antarmuka, perilaku mekanisme caching dalam mengendalikan permintaan data berulang, serta kemampuan sistem dalam menyinkronkan pembaruan data sesuai dengan kondisi yang terjadi pada sisi backend. Selain itu, evaluasi juga dilakukan untuk memastikan bahwa perilaku sistem frontend telah sesuai dengan perancangan Client Data Layer dan arsitektur frontend yang direncanakan pada tahap perancangan sistem.

Sistem frontend dinilai berhasil apabila seluruh indikator evaluasi tersebut terpenuhi dan tidak ditemukan ketidaksesuaian perilaku sistem selama skenario pengujian dijalankan.

## DAFTAR PUSTAKA

- Alviani, N. A., Studi, P., Fakultas, M., & Bangsa, U. B. (2025). Transformasi Digital pada UMKM dalam Meningkatkan Daya Saing Pasar. *Master Manajemen*, 3(1), 134–140.
- Aniley, D., Alemneh, E., & Abeba, G. (2024). Selection of software development life cycle models using machine learning approach. *International Journal of Computer Applications*, 186, 975–8887.
- Fajarini, S., Kurniawati, J., & Yuliani, F. (2025). Social media sentiment analysis as a new tool for predicting market trends and consumer behaviour. *Proceeding of International Conference on Social Science and Humanity*, 2, 899–909.
- Irianto, H., Viesta, A. D., Nugroho, A. T., Wahyuni, T., Prabowo, W. C., Hamid, I. N., Anufah, T. N., Permatasari, H. I., Salsabila, A., Sofyana, & Hardiyanti, F. Y. (2022). Digitalisasi UMKM sebagai Upaya Peningkatan Pemasaran dan Penjualan Online di Desa Tengklik. *Journal of Cooperative, Small and Medium Enterprise Development*, 1(2), 60–64.
- Islam, M. M. (2025). The impact of data-driven web frameworks on performance and scalability of u.s. enterprise applications. *International Journal of Business and Economics Insights*, 05, 523–558.
- Joseph, T. (2024). Natural Language Processing (NLP) for Sentiment Analysis in Social Media. *International Journal of Computing and Engineering*, 6(2), 35–48.
- Luz, H. (2025). Comparing Performance of Redux, MobX, and React Query. Preprint, ResearchGate.
- Mardhatilah, D., Omar, A., & Septiari, E. D. (2024). BUILDING CONSUMER ENGAGEMENT IN SOCIAL MEDIA: A SYSTEMATIC LITERATURE REVIEW. *JOURNAL OF BUSINESS MANAGEMENT AND ACCOUNTING*, 14(1), 1–35.
- Micheal, D. (2025). React Query and Lazy Loading: Performance Optimization Best Practices. Preprint, ResearchGate.
- Rahman, A. & Prihanto, A. (2024). Optimisasi Kinerja Aplikasi Fitness Berbasis Next.js Melalui Penerapan Metode Caching Pada PT. Anugerah Wijaya Raga. *Journal of Informatics and Computer Science (JINACS)*, 6(02), 333–340.
- Rathore, S., Nawkhare, R., Sharma, N., Chaudhary, N., Chakole, S., & Vishwakrama, B. (2025). Effective data visualization techniques for business decision-makers. *International Research Journal on Advanced Engineering and Management (IRJAEM)*, 3, 2029–2036.
- Sastraa, R. & Sutawinata, A. M. (2023). Perancangan Aplikasi SIP-PTK Sekolah Dasar Negeri Guntur 01 Menggunakan Model Fountain. *INSANtek*, 4(2), 63–68.
- Sofyan, S. & Agusman (2025). ANALISIS KESIAPAN UMKM MENGHADAPI EKONOMI DIGITAL. *Smart Jurnal Ilmiah*, IX(1), 1–4.
- Tanstack LCC (2025). Tanstack Query Official Documentation.
- Trulline, P. (2021). Pemasaran produk UMKM melalui media sosial dan e-commerce. *Jurnal Manajemen Komunikasi*, 5(2), 259.