BAB I PENDAHULUAN

1.1 Latar Belakang

Transformasi digital telah menjadi sebuah keharusan bagi perusahaan yang ingin mempertahankan relevansi dan keunggulan kompetitif di era modern. Salah satu pilar utama dari transformasi ini adalah pengembangan aplikasi internal yang dirancang khusus untuk mengoptimalkan proses bisnis. Dalam lingkungan bisnis yang dinamis, efisiensi dan efektivitas operasional bukan lagi sekadar tujuan, melainkan prasyarat untuk bertahan dan berkembang [1]. Aplikasi internal berperan penting dalam mendukung operasional harian perusahaan, mulai dari manajemen sumber daya hingga pelaporan keuangan. Data yang dihasilkan dari aplikasi ini membantu perusahaan membuat keputusan yang lebih tepat karena didasarkan pada informasi yang akurat dan terukur [2].

Meskipun demikian, perjalanan pengembangan aplikasi internal tidak selamanya mulus. Prosesnya sering kali dihadapkan pada serangkaian tantangan yang kompleks dan berulang. Salah satu tantangan terbesar adalah kebutuhan untuk membangun setiap aplikasi dari awal, yang sering kali mengarah pada duplikasi pekerjaan yang tidak perlu, inkonsistensi antarmuka antar aplikasi, dan membengkaknya waktu serta biaya pengembangan. Tanpa adanya standar yang terpadu, setiap tim pengembang mungkin akan menciptakan solusi dengan gaya dan kualitas yang berbeda, yang pada akhirnya mempersulit proses pemeliharaan dan integrasi di masa depan [3].

Untuk menjawab tantangan ini, pendekatan pengembangan modern mengandalkan penggunaan *boilerplate*. Secara konseptual, *boilerplate* dapat diibaratkan sebagai sebuah cetak biru atau fondasi bangunan yang telah disiapkan sebelumnya. Ia menyediakan kerangka kerja dasar yang kokoh, lengkap dengan struktur proyek yang terorganisir, komponen-komponen esensial, dan konfigurasi standar yang telah teruji [3, 4]. Dengan menggunakan *boilerplate*, tim pengembang tidak perlu lagi memulai dari nol. Mereka dapat langsung fokus pada pembangunan fitur-fitur unik yang memberikan nilai tambah bagi bisnis, sementara aspek-aspek fundamental seperti arsitektur, keamanan, dan konsistensi desain telah ditangani oleh *boilerplate* itu sendiri.

Dalam konteks pengembangan *boilerplate* ini, pemilihan teknologi menjadi faktor penentu keberhasilan. Vue 3 dipilih karena merupakan *framework* JavaScript yang sangat progresif dan berkinerja tinggi. Fitur andalannya, *Composition API*, secara radikal mengubah cara pengembang mengorganisir logika aplikasi. Ia

memungkinkan pengelompokan logika berdasarkan fitur (*feature-based*) ketimbang berdasarkan opsi (*option-based*), sehingga menghasilkan kode yang jauh lebih modular, mudah dibaca, dan dapat digunakan kembali di berbagai bagian aplikasi [5, 6]. Di sisi lain, TypeScript hadir untuk melengkapi Vue 3 dengan sistem tipe data statis yang kuat. TypeScript berperan sebagai "penjaga kualitas" yang bekerja pada tahap pengembangan, mendeteksi potensi kesalahan tipe data sebelum kode dijalankan [7]. Hal ini tidak hanya meminimalkan *bug* saat aplikasi digunakan oleh pengguna (*runtime errors*), tetapi juga meningkatkan produktivitas pengembang melalui fitur seperti *autocompletion* yang cerdas dan dokumentasi kode yang implisit [8].

Dengan demikian, pengembangan *boilerplate* yang mengintegrasikan kekuatan Vue 3 dan TypeScript merupakan sebuah langkah strategis yang sangat relevan. Solusi ini secara langsung mengatasi masalah efisiensi dan konsistensi, memungkinkan perusahaan untuk mengembangkan aplikasi internal berkualitas tinggi dengan lebih cepat dan andal. Pada akhirnya, inisiatif ini tidak hanya mendukung percepatan transformasi digital perusahaan, tetapi juga membangun fondasi teknologi yang solid untuk inovasi di masa mendatang [3, 9].

1.2 Rumusan Masalah dan Solusi

1. Bagaimana mengembangkan boilerplate yang dapat mempercepat dan mempermudah proses pengembangan aplikasi internal perusahaan tanpa mengorbankan kualitas dan konsistensi kode?

2. Bagaimana memanfaatkan fitur-fitur dari Vue 3 dan TypeScript untuk menghasilkan boilerplate yang modular, mudah diimplementasikan, dan dapat disesuaikan dengan kebutuhan spesifik setiap aplikasi?

3. Bagaimana menciptakan boilerplate yang dapat mendukung pengembangan aplikasi secara berkelanjutan dan meminimalkan risiko kesalahan dalam penulisan kode?

Solusi:

1. Mengembangkan boilerplate berbasis Vue 3 dan TypeScript yang berisi struktur dasar dan komponen-komponen umum untuk aplikasi internal perusahaan. Boilerplate ini dirancang dengan modularitas tinggi agar dapat dikembangkan dan disesuaikan sesuai kebutuhan aplikasi yang berbeda.

2. Memanfaatkan fitur-fitur Vue 3, seperti Composition API, dan dukungan tipe dari TypeScript untuk meningkatkan kualitas kode dan memungkinkan pengembang membangun aplikasi dengan keamanan yang lebih baik.

3. Menyediakan dokumentasi lengkap dan panduan implementasi untuk boilerplate ini, sehingga memudahkan tim pengembang dalam memahami cara penggunaan, penerapan, dan pengembangan lebih lanjut. Dengan boilerplate ini, pengembang dapat fokus pada fitur utama tanpa perlu membuat ulang komponen dasar, meningkatkan efisiensi serta konsistensi aplikasi.

1.3 Tujuan

Berdasarkan rumusan masalah yang ada, tujuan yang akan dicapai adalah:

1. Menghasilkan boilerplate yang efisien dan konsisten:

   Mengembangkan boilerplate yang mampu mempercepat dan mempermudah proses pengembangan aplikasi internal perusahaan, tanpa mengorbankan kualitas dan konsistensi kode.

2. Memanfaatkan keunggulan teknologi Vue 3 C TypeScript:

   Memaksimalkan fitur-fitur dari Vue 3 (seperti Composition API) dan keamanan tipe dari TypeScript untuk menciptakan boilerplate yang modular, mudah diimplementasikan, serta bisa disesuaikan dengan kebutuhan spesifik setiap aplikasi.

3. Mendukung pengembangan berkelanjutan dan meminimalisir kesalahan:

   Membuat boilerplate yang mendukung proses pengembangan aplikasi secara berkelanjutan dan dapat meminimalisir risiko kesalahan dalam penulisan kode dengan menyediakan dokumentasi serta panduan implementasi yang jelas.

1.4 Batasan Masalah

Batasan masalah dalam pembuatan sistem ini adalah:

1. Fokus pada boilerplate aplikasi internal berbasis web: Pengembangan hanya mencakup boilerplate front-end untuk aplikasi internal perusahaan, yang terdiri dari struktur proyek, komponen UI dasar (seperti tombol, form, dan navigasi), serta pengelolaan state dan layout. Proyek ini tidak membahas fitur backend, integrasi dengan API eksternal, maupun pengembangan aplikasi mobile atau desktop.

2. Teknologi terbatas pada Vue 3, Vuetify, dan TypeScript:

   Boilerplate dikembangkan menggunakan teknologi utama Vue 3, Vuetify, dan TypeScript, dengan penerapan fitur-fitur seperti Composition API dan type checking. Tidak dilakukan integrasi dengan framework lain, teknologi backend, atau library di luar ekosistem tersebut.

3. Dokumentasi hanya untuk penggunaan dasar dan pengembangan internal:

   Dokumentasi dan panduan implementasi yang disusun dibatasi pada instalasi, struktur proyek, dan contoh implementasi boilerplate untuk aplikasi internal. Proyek ini tidak membahas dokumentasi lanjutan untuk studi kasus eksternal, pengujian lanjutan (unit/E2E), atau skenario implementasi di luar lingkungan internal perusahaan.

1.5 Penjadwalan Kerja

Sub bab ini berisi hal-hal apa yang akan dilakukan untuk mencapai solusi yang telah ditawarkan pada poin sebelunya. Jadwal Pelaksanaan ini akan menjadi pedoman baik bagi mahasiswa maupun para pembimbing akan target-target pencapaian sesuai timeline yang disepakati.
Satuan waktu yang digunakan sebagai timeline dapat disesuaikan dengan kebutuhan masing-masing pekerjaan yang dilakukan. Satuan waktu yang sebaiknya digunakan adalah satuan jam, hari, atau minggu.

Detail pelaksanaan aktivitas kerja selama masa magang ditampilkan pada tabel berikut.

Tabel 1 1 Penjadwalan Kerja

No	Deskripsi Kerja	Agustus 2024	September2024				Oktober2024				November 2024				Desember2024				Januari2025				Februari2025				Maret2025				April2025				Mei2025				Juni2025			
		4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Orientasi Perusahaan																																									
2	Diskusi dan Perencanaan Awal																																									
3	Pemahaman Teknologi\																																									
4	Identifikasi Kebutuhan																																									
5	Diskusi & Koordinasi																																									
6	Perencanaan Teknis																																									
7	Pengembangan Frontend Framework																																									
8	Pengujian Implementasi ke Project 1																																									
9	Pengujian Implementasi ke Project 2																																									
10	Evaluasi dan Optimalisasi																																									
11	Membuat Dokumentasi																																									
12	Launching Framework UI																																									