BAB 1. PENDAHULUAN

1.1 Latar Belakang

Di dunia pengembangan aplikasi mobile, Flutter telah muncul sebagai platform yang menghadirkan kemudahan bagi pengembang untuk menciptakan aplikasi dengan desain yang menarik. Dengan menggunakan satu basis kode, Flutter memungkinkan pengembang untuk mengembangkan aplikasi yang dapat beroperasi diberbagai platform, seperti Android, iOS, dan desktop. Kemampuan deklaratifnya memperkuat konsep bahwa Flutter membangun antarmuka pengguna untuk mencerminkan kondisi aplikasi secara real-time. Artinya, ketika ada perubahan dalam aplikasi, baik itu interaksi pengguna maupun dari sistem, Flutter secara otomatis mengatur ulang antarmuka pengguna untuk mencerminkan perubahan tersebut (Flutter Docs, 2024).

Namun di balik kemudahan yang ditawarkan Flutter, kompleksitas pengelolaan *state* dalam pengembangan aplikasi tetap menjadi tantangan yang perlu diatasi. Itulah mengapa pemilihan pendekatan yang tepat dalam manajemen *state* sangat penting. Setiap metode memiliki keunggulan dan keterbatasannya sendiri yang perlu dipertimbangkan sesuai dengan kebutuhan aplikasi. Di tengah beragamnya opsi yang tersedia, solusi *state management* seperti Riverpod (Riverpod: Dart Package, 2024a) dan solusi *dependency injection* seperti GetIt (Get_it: Dart Package, 2024b) telah menonjol sebagai pilihan yang canggih dan fleksibel untuk pengembangan aplikasi Flutter. Riverpod, dengan keamanan waktu kompilasi, kemudahan pengguna, dan dukungan untuk status yang tidak dapat diubah, menjadi pilihan utama bagi banyak pengembang (Ahmed Taha ElElemy, 2023). Selain itu Riverpod juga sebagai solusi populer yang dibuat untuk melakukan *state management*, *dependency injection* dan *caching* data yang tidak disediakan oleh *framework* Flutter secara asli (Rafael Delos Santos, 2023). Di sisi lain, GetIt menawarkan solusi ekstra ringan dan kuat yang menggabungkan manajemen status berkinerja tinggi, *dependency injection*, dan manajemen rute dengan cepat dan praktis, yang memungkinkan pemisahan total tampilan, logika

presentasi, logika bisnis, *dependency injection*, dan navigasi (Alvaro Armijos, 2023).

Tidak hanya itu baik Riverpod maupun GetIt juga mendukung konsep *dependency injection* yang kuat, yang memudahkan pengembang dalam mengembangkan aplikasi yang meningkatkan kemampuan kontrol dan pengamatan kasus uji, meningkatkan ekstensibilitas dan reusabilitas perangkat lunak(Alejandro Ferrero, 2021). Dengan memanfaatkan fitur-fitur *dependency injection* yang disediakan oleh kedua solusi ini, pengembang dapat mengoptimalkan pengelolaan state dalam aplikasi *Flutter* mereka.

Dalam konteks penelitian sebelumnya M Fakhri Abdillah dkk. (2023), telah terdapat studi yang mengkaji performa berbagai *library state management* dalam pengembangan perangkat lunak berbasis Flutter untuk platform android. Penelitian sebelumnya telah memfokuskan pada perbandingan performa antara *state management* seperti GetX dan BLoC. Hasil dari penelitian tersebut memberikan wawasan bahwa GetX memiliki performa penggunaan *Central Processing Unit (CPU)* dan memori lebih efisien dari pada BLoC, dengan selisih penggunaan CPU sebesar 11% serta perbedaan memori sebesar 111.6333 MB. Sementara itu, GetX dan BLoC memiliki tingkat konsumsi energi yang sama persis, yaitu kategori ringan *(Light)*.

Namun, meskipun telah ada penelitian sebelumnya yang membandingkan beberapa *library state management*, belum ada penelitian yang secara khusus menganalisis performa *library* pada Flutter yang memiliki *dependency injection* yang kuat. Riverpod dan GetIt merupakan dua *library* yang populer dan sering digunakan oleh pengembang Flutter karena keduanya terkenal memiliki dukungan *dependency injection* yang kuat. Kedua *library* ini memungkinkan pengguna untuk dengan mudah menyediakan dan mengakses objek dari konteks aplikasi dengan struktur yang terstruktur dan kuat. Namun, perbandingan langsung dalam hal performa antara Riverpod dan GetIt memerlukan penelitian lebih mendalam.

Oleh karena itu, penelitian ini bertujuan untuk melengkapi kesenjangan pengetahuan yang ada khususnya dalam konteks *dependency injection* dengan melakukan analisis performa yang komprehensif terhadap Riverpod dan GetIt

dalam pengembangan perangkat lunak berbasis Flutter untuk platform Android, dengan studi kasus pengembangan Movie App. Aplikasi ini dipilih karena mewakili aplikasi modern dengan struktur data yang kompleks dan memerlukan pengelolaan state serta *dependency* yang optimal.

Untuk menunjang kinerja aplikasi, digunakan The Movie Database (TMDB) API sebagai sumber data eksternal. TMDB dipilih karena memiliki reputasi yang solid sebagai penyedia layanan informasi film dari pihak ketiga. API ini menyediakan berbagai jenis data yang lengkap, seperti judul film, genre, aktor, rating, hingga ulasan. Ketersediaan informasi yang luas ini memungkinkan sistem untuk menyajikan rekomendasi yang lebih personal kepada pengguna, berdasarkan preferensi genre yang diperoleh dari aktivitas pengguna di dalam aplikasi. Data preferensi tersebut menjadi komponen penting dalam menghasilkan rekomendasi yang relevan dan sesuai dengan ketertarikan pengguna (Toba Amiruddin Sitorus & Ledy Elsera Astrianty, 2024).

Melalui pengujian menggunakan Android Profiler, Snapdragon Profiler dan Flutter DevTools terhadap metrik performa. Dengan wawasan yang mendalam tentang kinerja kedua *library dependency injection* ini, diharapkan penelitian ini mampu memberikan manfaat yang berarti dalam pemilihan solusi yang optimal untuk aplikasi android yang responsif, mudah dipelihara, dan efisien.

1.2 Rumusan Masalah

Dalam pengembangan aplikasi Flutter, pemilihan *library* yang tepat menjadi krusial dalam memastikan kinerja aplikasi yang responsif dan efisien. Namun, masih terdapat kesenjangan dalam pemahaman tentang efektivitas penggunaan *library* tersebut, terutama dalam menghadapi kompleksitas aplikasi yang semakin meningkat. Dengan demikian, permasalahan yang dapat dirumuskan adalah:

1. Bagaimana karakteristik dan mekanisme kerja *dependency injection* pada Riverpod dan GetIt mempengaruhi pola pengelolaan state dalam pengembangan aplikasi Flutter berbasis Android?
2. Bagaimana perbandingan performa antara penggunaan Riverpod dan penggunaan GetIt yang dikombinasikan dengan BLoC pada pengembangan

perangkat lunak berbasis Flutter untuk platform Android, khususnya dalam hal *CPU usage*, *memory usage*, dan *energy consumption*?

3. Bagaimana perbedaan implementasi antara penggunaan Riverpod dan kombinasi GetIt dengan BLoC mempengaruhi kinerja aplikasi Android dalam hal skalabilitas dan kompleksitas?

1.3 Tujuan Penelitian

Tujuan penelitian yang akan dilaksanakan adalah :

1. Untuk menganalisis karakteristik dan mekanisme kerja *dependency injection* pada Riverpod dan GetIt serta bagaimana pengaruhnya terhadap pola pengelolaan state dalam pengembangan aplikasi Flutter berbasis Android.
2. Untuk menganalisis dan membandingkan performa Riverpod dan GetIt yang dikombinasikan dengan BLoC pada pengembangan perangkat lunak berbasis Flutter untuk platform Android, dengan fokus pada variabel *CPU usage*, *memory usage*, dan *energy consumption*.
3. Untuk mengevaluasi bagaimana perbedaan implementasi antara Riverpod dan kombinasi GetIt dengan BLoC mempengaruhi kinerja aplikasi Android dalam hal skalabilitas dan manajemen kompleksitas.

1.4 Batasan Masalah

Penelitian ini memiliki keterbatasan yang perlu dicermati, di antaranya:

1. Fokus dari penelitian ini hanya pada pengembangan aplikasi android menggunakan *framework* Flutter, tanpa membahas platform lain seperti iOS, web, atau desktop.
2. Penelitian ini dilakukan pada aplikasi yang berfokus pada penyedia informasi tentang film, yaitu aplikasi *Movie Apps*.
3. Aplikasi hanya menyediakan trailer yang memiliki durasi video pendek , tidak menyertakan film penuh atau konten video lainnya.

1.5 Manfaat Penelitian

Penelitian ini bertujuan untuk dapat memberikan berbagai manfaat sebagai berikut:

1. Memberi wawasan bagaimana mekanisme kerja dependency injection pada Riverpod dan GetIt, serta bagaimana keduanya mempengaruhi pola pengelolaan state dalam pengembangan aplikasi Flutter berbasis Android.
2. Membantu pengembang dalam memilih *library* yang tepat pada Flutter sesuai dengan kebutuhan aplikasi untuk meningkatkan efisiensi dan responsivitas aplikasi yang dikembangkan.
3. Menjadi referensi dalam optimalisasi penggunaan sumber daya perangkat saat menggunakan Riverpod atau kombinasi GetIt dengan BLoC dalam pengembangan perangkat lunak berbasis Flutter.
4. Memberikan pemahaman yang lebih baik mengenai dampak implementasi dependency injection terhadap skalabilitas dan manajemen kompleksitas, sehingga pengembang dapat mengambil keputusan yang lebih bijak dalam pemilihan pendekatan sesuai dengan kebutuhan aplikasi mereka.