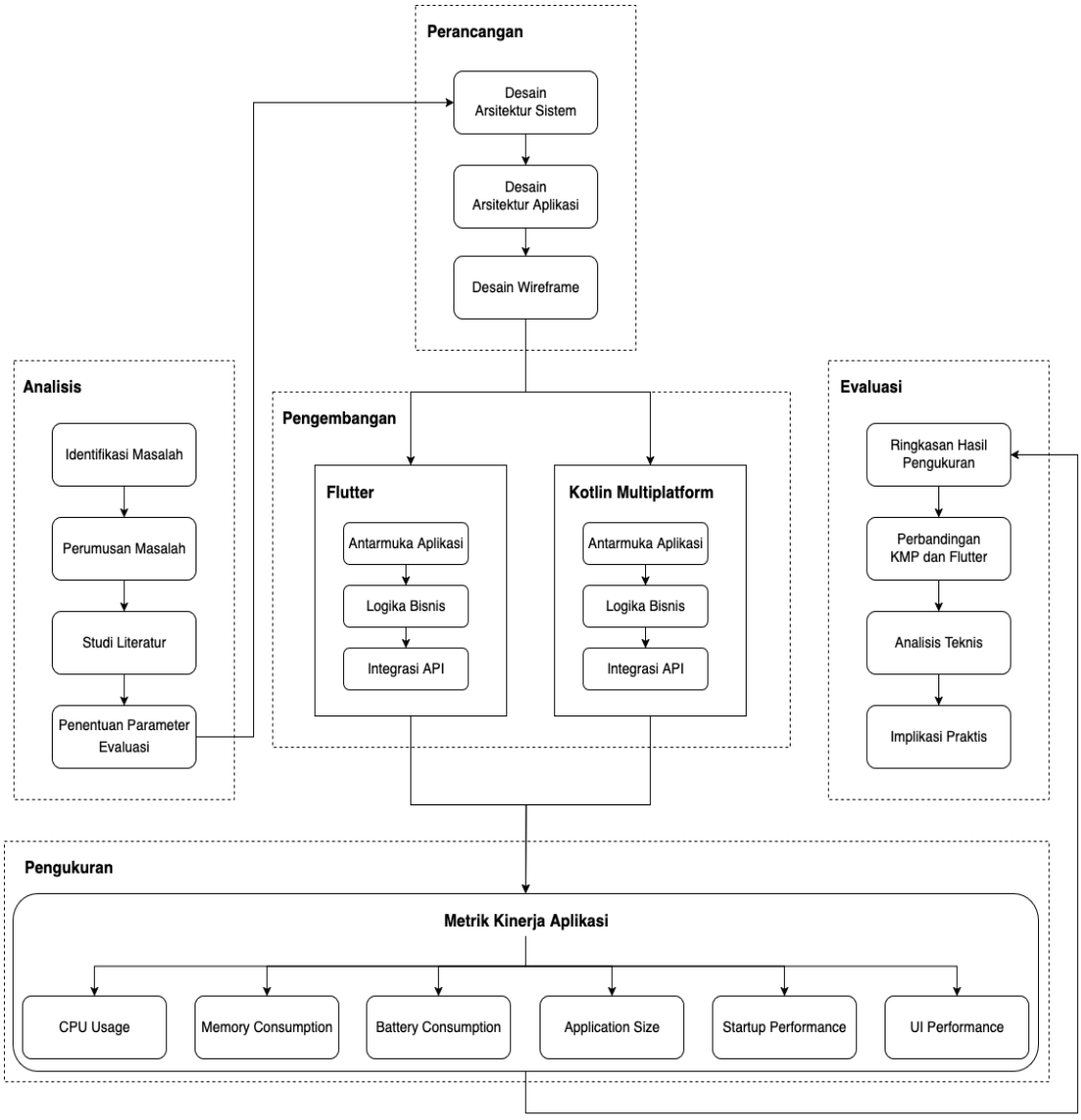
BAB III
METODOLOGI PENELITIAN

3.1 Desain Penelitian

Desain penelitian merupakan gambaran besar dari penelitian yang akan dilakukan. Tujuan dari adanya desain penelitian adalah untuk mendapatkan gambaran penelitian yang akan dilakukan dan berperan penting dalam menentukan proses yang diperlukan untuk mencapai tujuan dari penelitian. Pada Gambar 3.1 memperlihatkan urutan lengkap proses pengembangan aplikasi mulai dari tahap awal sampai tahap akhir.



Gambar 3.1 Desain Penelitian

Penelitian ini menggunakan *quasi-experimental design* untuk membandingkan kinerja teknis objektif antara Kotlin Multiplatform dan Flutter dalam pengembangan aplikasi *mobile* lintas *platform*. *Quasi-experimental design* merupakan desain eksperimen yang tidak sepenuhnya menerapkan randomisasi, namun tetap menjaga validitas hasil dengan mengontrol variabel-variabel penting (Miller et al., 2020). Pendekatan ini sesuai untuk *technology evaluation research* di mana *true randomization* tidak selalu dapat dilakukan, tetapi kondisi uji dapat dikendalikan sehingga perbandingan tetap adil dan bias dapat diminimalkan. Dalam konteks penelitian ini, *quasi-experimental design* berperan untuk memastikan bahwa kedua *framework* diuji dalam kondisi yang setara, misalnya melalui implementasi aplikasi dengan fitur identik, penggunaan perangkat fisik nyata, pengulangan pengukuran, serta protokol pengujian yang konsisten. Dengan cara ini, penelitian tetap mampu menghasilkan data *benchmarking* yang objektif dan reliabel meskipun tidak menggunakan *random assignment* sebagaimana dalam *true experimental design*. Pendekatan ini telah terbukti efektif dalam *software* engineering research untuk evaluasi teknologi dan banyak digunakan dalam literatur akademik (Maciejewski, 2020).

Dalam rancangan quasi-experimental ini, komponen penelitian dirancang secara sistematis untuk memastikan perbandingan terkendali yang valid dan hasil yang dapat dipertanggungjawabkan secara ilmiah. Penelitian ini menggunakan pendekatan perbandingan objektif dengan mengukur kinerja dua *framework* pada *platform* yang berbeda sambil mengendalikan faktor-faktor eksternal yang dapat mempengaruhi hasil. Penelitian ini fokus pada perbandingan kinerja *framework* dengan mengembangkan aplikasi yang identik menggunakan dua teknologi berbeda (Kotlin Multiplatform dan Flutter) pada dua *platform* berbeda (Android dan iOS). Untuk memastikan perbandingan yang objektif dan dapat diandalkan, penelitian ini mengukur enam metrik kinerja teknis yang relevan untuk aplikasi *mobile* sambil mengendalikan faktor-faktor eksternal yang dapat mempengaruhi hasil.

Penelitian ini membandingkan Kotlin Multiplatform dan Flutter sebagai dua pendekatan utama dalam pengembangan aplikasi lintas *platform*. Kedua *framework* dipilih karena memiliki arsitektur dan strategi kompilasi yang berbeda, sehingga memberikan karakteristik kinerja yang dapat dibandingkan secara objektif.

Perbandingan dilakukan pada *platform* Android dan iOS untuk mengidentifikasi bagaimana setiap *framework* berperforma pada ekosistem yang berbeda. Penelitian ini mengukur enam aspek kinerja aplikasi yang paling relevan untuk pengalaman pengguna dan efisiensi operasional. CPU Usage diukur untuk mengetahui efisiensi komputasi *framework* dalam menjalankan operasi aplikasi. Memory Consumption dianalisis untuk memahami bagaimana setiap *framework* mengelola sumber daya memori selama runtime. Battery Consumption dievaluasi karena efisiensi baterai merupakan faktor kritis dalam aplikasi *mobile*. Application Size diukur untuk mengetahui jejak penyimpanan yang dibutuhkan aplikasi. Startup Performance dianalisis untuk memahami responsivitas aplikasi saat pertama kali dibuka. UI Performance dievaluasi untuk mengukur kelancaran interaksi pengguna dengan antarmuka aplikasi.

Untuk memastikan validitas perbandingan, penelitian ini mengendalikan berbagai faktor yang dapat mempengaruhi hasil pengukuran. Implementasi kedua aplikasi dilakukan oleh developer yang sama untuk menghilangkan perbedaan keahlian coding. Penggunaan perangkat testing yang identik memastikan bahwa perbedaan kinerja benar-benar berasal dari karakteristik *framework*, bukan dari perbedaan *hardware*. Kedua aplikasi mengimplementasikan fitur yang persis sama dengan kompleksitas kode yang setara. Protokol pengukuran distandarisasi dengan prosedur yang identik untuk setiap metrik. Kondisi pengujian dijaga tetap konsisten, termasuk koneksi jaringan dan state perangkat. Integrasi backend menggunakan API endpoints dan struktur data yang sama untuk memastikan beban kerja jaringan yang identik.

Penelitian ini dilaksanakan melalui lima tahapan utama yang dirancang secara sistematis. Berikut tahapan utama dalam penilitian ini:

1) Analisis

Tahap analisis merupakan fondasi penelitian yang dimulai dengan identifikasi masalah penelitian melalui kajian mendalam terhadap kesenjangan riset dalam perbandingan *framework* pengembangan *mobile* lintas *platform*. Proses ini melibatkan analisis terhadap penelitian-penelitian sebelumnya untuk mengidentifikasi keterbatasan metodologi dan area yang belum dieksplorasi secara

komprehensif. Selanjutnya dilakukan perumusan masalah yang spesifik dan terukur berdasarkan analisis literature review dan kebutuhan praktis dalam pengembangan perangkat lunak. Studi literatur komprehensif dilakukan untuk mengidentifikasi praktik terbaik dalam *mobile* application *benchmarking*, metodologi penelitian yang terbukti efektif dalam perbandingan *framework*, dan standar pengukuran kinerja yang diterima dalam industri. Tahap ini diakhiri dengan penentuan parameter evaluasi yang mencakup pemilihan metrik kinerja yang paling relevan untuk aplikasi *mobile*, identifikasi alat pengukuran yang sesuai dan akurat, serta penetapan protokol pengujian yang dapat menghasilkan data yang reliable dan direproduksi.

2) Perancangan

Tahap perancangan bertujuan untuk menciptakan blueprint sistem yang akan dikembangkan dengan memastikan equivalensi implementasi antara kedua *framework*. Proses dimulai dengan desain arsitektur sistem yang mempertimbangkan karakteristik unik dari Kotlin Multiplatform dan Flutter, termasuk strategi kompilasi, manajemen memori, dan integrasi *platform native*. Arsitektur dirancang menggunakan pendekatan layered yang memisahkan tanggung jawab setiap komponen secara jelas untuk memungkinkan implementasi yang konsisten. Desain arsitektur aplikasi dikembangkan dengan mengikuti prinsip scalable dan maintainable architecture yang dapat diterapkan secara identik pada kedua *framework*. Selanjutnya dilakukan desain wireframe yang mendetailkan tata letak antarmuka pengguna, navigasi aplikasi, dan interaksi pengguna secara komprehensif. Wireframe dirancang dengan mempertimbangkan guideline *platform* Android dan iOS untuk memastikan pengalaman pengguna yang optimal dan konsisten. Tahap ini menghasilkan desain prototype yang berfungsi sebagai referensi visual dan fungsional untuk tahap implementasi, memastikan bahwa kedua aplikasi akan memiliki fitur, kompleksitas, dan user experience yang identik.

3) Pengembangan

Tahap pengembangan dilakukan secara paralel untuk kedua *framework* dengan mengikuti protokol yang ketat untuk memastikan kesetaraan implementasi. Implementasi Flutter dimulai dengan pengembangan antarmuka aplikasi

menggunakan sistem widget Flutter yang mengikuti wireframe yang telah dirancang, implementasi logika bisnis menggunakan bahasa pemrograman Dart dengan pola yang telah mapan, dan integrasi API menggunakan library HTTP client yang dioptimalkan untuk Flutter. Implementasi Kotlin Multiplatform dilakukan dengan mengembangkan logika bisnis bersama dan antarmuka bersama menggunakan Compose Multiplatform untuk memastikan konsistensi tampilan pada kedua *platform*, implementasi yang memanfaatkan berbagi kode maksimal antara Android dan iOS, dan integrasi API melalui lapisan jaringan bersama yang konsisten. Setiap tahap implementasi mengikuti daftar fitur yang identik yang memastikan bahwa kedua aplikasi memiliki fungsionalitas yang persis sama, kompleksitas kode yang setara, dan mekanisme penanganan kesalahan yang sama. Proses pengembangan juga melibatkan tinjauan kode sistematis untuk memastikan bahwa optimasi yang dilakukan pada satu *framework* juga diterapkan secara proporsional pada *framework* lainnya, sehingga perbandingan kinerja mencerminkan karakteristik asli dari masing-masing teknologi.

4) Pengukuran

Tahap pengukuran dilakukan menggunakan protokol sistematis dan ketat untuk menghasilkan data kinerja yang akurat dan dapat diulang. Seluruh pengukuran dilakukan menggunakan perangkat fisik yang identik untuk menghindari variasi yang dapat terjadi pada emulator atau simulator. Metrik kinerja aplikasi diukur secara menyeluruh mencakup CPU *usage* yang dipantau menggunakan tools selama berbagai skenario penggunaan, *memory consumption* yang dilacak melalui analisis heap dan pemantauan garbage collection, *battery consumption* yang diukur menggunakan battery profiler dengan sesi penggunaan yang terkendali, *application size* yang dihitung dari file distribusi final, *startup performance* yang diukur menggunakan alat pengatur waktu otomatis untuk skenario cold start dan warm start, dan UI *performance* yang dievaluasi melalui pemantauan frame rate dan pengukuran waktu respons sentuhan. Setiap metrik diukur minimum 30 kali untuk setiap kombinasi *framework-platform* untuk memastikan reliabilitas data secara statistik. Protokol pengukuran mencakup lingkungan pengujian yang terstandar dengan kondisi perangkat yang konsisten, proses latar belakang yang minimal, dan kondisi jaringan yang terkendali. Prosedur

validasi data diterapkan untuk mengidentifikasi dan menangani data pencilan atau kesalahan pengukuran, serta memastikan konsistensi pengukuran di berbagai sesi pengujian.

5) Evaluasi

Tahap evaluasi mengintegrasikan seluruh data yang telah dikumpulkan untuk menghasilkan wawasan yang menyeluruh tentang karakteristik kinerja kedua *framework*. Ringkasan hasil pengukuran dilakukan melalui analisis deskriptif yang menyajikan data empiris secara objektif dari enam metrik kinerja yang telah diukur. Perbandingan kinerja KMP dan Flutter dilakukan secara sistematis dengan mengidentifikasi perbedaan nilai pengukuran, menganalisis keunggulan relatif masing-masing *framework* pada setiap *platform*, dan mengeksplorasi pola konsistensi lintas berbagai metrik kinerja. Analisis teknis dilakukan melalui interpretasi mendalam terhadap faktor-faktor arsitektural yang berkontribusi pada karakteristik kinerja yang teramati, mencakup pendekatan kompilasi, arsitektur runtime, strategi manajemen memori, dan optimasi khusus *platform* yang mempengaruhi efisiensi operasional masing-masing *framework*. Proses evaluasi menghasilkan pemahaman komprehensif tentang kelebihan dan keterbatasan kedua *framework* dalam berbagai aspek kinerja aplikasi. Implikasi praktis dirumuskan berdasarkan temuan empiris untuk memberikan panduan berbasis bukti dalam pemilihan teknologi pengembangan aplikasi *mobile* lintas *platform*, dengan mempertimbangkan konteks implementasi nyata dan kebutuhan spesifik pengembangan aplikasi manajemen akademik mahasiswa.

3.2 Alat dan Bahan Penelitian

Penelitian ini memerlukan berbagai alat dan bahan untuk mendukung proses pengembangan, pengujian, dan analisis aplikasi *mobile* lintas *platform*. Pemilihan alat penelitian didasarkan pada requirement untuk physical device testing dan controlled experimental conditions sesuai dengan best practices dalam *mobile* application *performance* evaluation. Semua pengukuran dilakukan menggunakan perangkat fisik (bukan emulator atau simulator) untuk memastikan akurasi data *performance* yang representatif dengan kondisi real-world *usage* (Skantz, 2023). Berikut adalah rincian alat dan bahan yang digunakan dalam penelitian:

3.2.1 Alat Penelitian

1) Sistem komputer/laptop dengan spesifikasi sebagai berikut:
   a) Device : MacBook 3 Pro
   b) Processor : Apple M3 Pro
   c) RAM : 18 GB
   d) Storage : 512 GB
   e) Sistem Operasi : macOS 15.5 Sequoia
2) Perangkat android dengan spesifikasi sebagai berikut:
   a) Device : Xiaomi Redmi 10
   b) Processor : Mediatek Helio G88
   c) RAM : 6 GB
   d) Storage : 128 GB
   e) Android Version : Android 13
3) Perangkat iOS dengan spesifikasi sebagai berikut:
   a) Device : iPhone 14 pro
   b) Processor : A16 Bionic
   c) RAM : 6 GB
   d) Storage : 128 GB
   e) iOS Version : iOS 18
4) Tools Pengukuran Performa
   a) Android Studio Profiler untuk monitoring CPU *usage* dan *memory consumption*
   b) ADB (Android Debug Bridge) *platform* tools untuk automated measurement
   c) Xcode Instruments suite (Time Profiler, Allocations, Energy Impact tool)
   d) Custom automated testing scripts untuk consistent measurement procedures
   e) Statistical analysis software untuk data analysis
5) Controlled Testing Environment Setup
   a) Wi-Fi network dengan stable connection dan consistent bandwidth

b) Standardized device settings: brightness 50%, auto-lock disabled, airplane mode off
c) Clean testing environment dengan minimal background interference
d) Consistent power supply untuk charging devices antara testing sessions

6) Perangkat lunak untuk pengembangan aplikasi adalah sebagai berikut:
a) Android Studio
b) Xcode
c) Visual Studio Code
d) Claude Code
e) Git
f) Flutter SDK
g) Android SDK
h) Datagrip
i) Draw.io
j) Figma

3.2.2 Bahan Penelitian

Berikut merupakan daftar bahan penelitian yang diperlukan berdasarkan kerangka kerja penelitian yang telah dirancang:

1) Literatur dan Sumber Teori
a) Buku referensi tentang pengembangan aplikasi *mobile* dan cross-*platform development*
b) Artikel jurnal ilmiah terkait Kotlin Multiplatform, Flutter, dan pengembangan aplikasi lintas *platform*
c) Paper penelitian tentang metrik evaluasi dan pengukuran performa aplikasi *mobile*
d) Dokumentasi resmi dari Kotlin Multiplatform dan Flutter
e) Studi kasus dan best practices dalam pengembangan aplikasi manajemen akademik

2) Dokumentasi Teknis
a) Dokumentasi resmi Flutter
b) Dokumentasi resmi Koltin Multiplatform serta Compose Multiplatform
c) Dokumentasi library dan dependencies yang digunakan

3) Data Perangkat Lunak
   a) Kode program untuk implementasi kedua *framework*
   b) Desain rancangan sistem dan arsitektur aplikasi
   c) Dokumen spesifikasi sistem dan requirement
   d) Wireframe dan mockup desain antarmuka
   e) Schema database dan API specification
   f) Test cases dan testing documentation
4) Data Penelitian

   Penelitian ini menggunakan dataset simulasi yang merepresentasikan data akademik mahasiswa dengan rincian sebagai berikut:
   a) Sebanyak 16 user mahasiswa yang terdiri dari 8 laki-laki dan 8 perempuan. Setiap pasangan mahasiswa (1 laki-laki dan 1 perempuan) mewakili masing-masing program studi dari 8 fakultas.
   b) Setiap mahasiswa merupakan angkatan 2022 yang sedang berada pada semester 6. Oleh karena itu, masing-masing memiliki riwayat data nilai selama 5 semester sebelumnya, yang digunakan untuk fitur grades (perhitungan IPK, nilai per mata kuliah, dan transkrip akademik).
   c) Setiap mahasiswa memiliki data jadwal kuliah pada semester aktif dari hari Senin sampai Jumat, dengan variasi jumlah mata kuliah.
   d) Modul direktori akademik menggunakan 10 dengan berbagai macam kategori.
   e) Modul media sosial mikro menggunakan 10 data postingan, dengan masing-masing postingan memiliki 3 komentar.

Bahan-bahan penelitian ini dipilih dan disusun secara sistematis untuk mendukung setiap tahap penelitian, mulai dari pemahaman konsep dasar hingga implementasi dan evaluasi. Literatur dan sumber teori menjadi fondasi dalam membangun pemahaman mendalam tentang teknologi yang diteliti. Dokumentasi teknis memastikan implementasi yang sesuai dengan standar dan best practices. Sedangkan data perangkat lunak menjadi bahan utama dalam proses pengembangan dan analisis aplikasi.

3.3 Protokol Pengukuran

3.3.1 Persiapan Lingkungan Pengujian

Persiapan lingkungan pengujian dilakukan untuk memastikan kondisi terkendali dan mengurangi faktor eksternal yang dapat mempengaruhi hasil pengukuran. Prosedur persiapan meliputi penyiapan perangkat dengan restart kedua perangkat testing untuk memastikan kondisi yang bersih, instalasi aplikasi testing saja untuk mengurangi gangguan dari proses latar belakang, dan menonaktifkan automatic updates serta background sync *services*. Pengaturan perangkat distandarisasi dengan tingkat kecerahan 50%, auto-lock dinonaktifkan, koneksi Wi-Fi saja, dan tingkat baterai yang sama sebelum setiap sesi pengujian. Standardisasi lingkungan mencakup pengujian dalam jaringan Wi-Fi dengan bandwidth yang stabil dan gangguan minimal selama sesi pengukuran.

3.3.2 Prosedur Pengukuran per Metrik

Pengukuran dilakukan menggunakan protokol sistematis dengan testing perangkat fisik untuk memastikan hasil yang akurat dan dapat diulang. Setiap metrik diukur dengan minimum 30 pengulangan untuk memastikan keandalan data. Metrik performa runtime (CPU *usage*, *memory consumption*, *battery consumption*, *startup performance*, UI *performance*) diukur dengan 30 pengulangan per *framework-platform*, mengikuti prinsip Central Limit Theorem yang menyatakan bahwa distribusi *sampling means* akan mendekati normal distribution ketika $n \geq 30$, terlepas dari distribusi populasi aslinya (Turney, 2022). CPU Usage diukur selama 10 menit per tes dengan skenario idle, penggunaan normal, dan operasi berat menggunakan Android Studio Profiler dan Xcode Instruments. Memory Consumption dipantau selama 15 menit mencakup peluncuran aplikasi, operasi normal, dan fitur yang membutuhkan memori tinggi. Battery Consumption diukur selama 1 jam pengujian nirkabel dengan simulasi pola penggunaan realistis. Application Size dianalisis langsung dari file APK/IPA dan jejak instalasi. Startup Performance diukur dengan 30 peluncuran otomatis untuk skenario cold start dan warm start. UI Performance dievaluasi selama 10 menit dengan fokus pada scrolling, navigasi, dan kelancaran animasi.

3.3.3 Kontrol Kualitas Pengukuran

Kontrol kualitas dilakukan untuk memastikan keandalan dan akurasi data pengukuran. Prosedur quality assurance meliputi verifikasi kondisi perangkat sebelum pengujian, pemantauan kondisi selama pengujian, dan validasi data setelah pengujian. Prosedur deteksi anomali diterapkan untuk mengidentifikasi dan menangani data pencilan atau kesalahan pengukuran. Verifikasi konsistensi data dilakukan melalui validasi silang dengan pendekatan pengukuran *multiple* jika memungkinkan, dan validasi keandalan pengukuran di berbagai sesi yang berbeda.