BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam era digital yang semakin berkembang pesat, kebutuhan akan aplikasi lintas *platform* (*cross-platform*) telah menjadi semakin krusial. Menurut data Statista, jumlah pengguna smartphone secara global mencapai hampir 7 miliar pada tahun 2023 dan diprediksi akan melampaui 7,7 miliar pada tahun 2028 (Taylor, 2024). Pertumbuhan yang signifikan ini menunjukkan pentingnya pengembangan aplikasi *mobile* yang efisien dan dapat menjangkau berbagai *platform*. Berdasarkan data dari Statcounter Global Stats untuk periode September 2023 hingga September 2024, Android mendominasi dengan pangsa pasar sekitar 71.85%, sementara iOS memegang sekitar 27.6% pangsa pasar (Statcounter Global Stats, 2024). Dominasi ini menekankan pentingnya pengembangan aplikasi yang kompatibel dengan kedua *platform* untuk mencapai jangkauan pengguna yang luas, menciptakan tantangan bagi pengembang aplikasi untuk menjangkau pengguna yang luas secara efisien.

Di tengah tuntutan pasar yang semakin kompetitif, industri pengembangan aplikasi *mobile* terus mencari solusi yang lebih efisien dalam hal waktu, sumber daya, dan biaya. Survei pengembang pada tahun 2023 mengungkapkan bahwa sekitar sepertiga pengembang *mobile* telah beralih menggunakan teknologi lintas *platform*, dengan Flutter menjadi *framework* terpopuler yang digunakan oleh 46% pengembang (Vailshery, 2024). Perbedaan preferensi ini menunjukkan adanya *trade-off* dalam hal performa dan efisiensi yang mempengaruhi keputusan pengembangan. Pengembangan aplikasi *native* untuk setiap *platform* memerlukan waktu, sumber daya, dan biaya yang signifikan, sementara pendekatan pengembangan lintas *platform* menawarkan solusi yang lebih efisien dan ekonomis. Namun, mayoritas penelitian yang ada berfokus pada aspek subjektif seperti *developer experience* dan *learning curve*, sementara perbandingan performa teknis yang objektif dan terukur masih terbatas (Mozharovskii, 2024).

Pasar aplikasi *mobile* dibagi antara dua sistem operasi utama, yaitu Android dan iOS, menciptakan tantangan unik karena setiap *platform* membutuhkan pendekatan pengembangan yang berbeda. Untuk tetap kompetitif, perusahaan sering harus mengembangkan dua aplikasi terpisah untuk memenuhi kebutuhan bisnis mereka bagi semua pelanggan terlepas dari perangkat *mobile* yang mereka gunakan (Guśpiel, 2024). Tantangan ini semakin kompleks karena pengembang harus menangani berbagai masalah terkait kapasitas penyimpanan, spesifikasi perangkat *mobile*, mobilitas, pengalaman pengguna, keamanan, dan privasi dalam pengembangan aplikasi *mobile* (Meirelles et al., 2019).

Aplikasi *native* yang dikembangkan secara khusus untuk *platform* tertentu, seperti Swift untuk iOS dan Kotlin untuk Android, memang menawarkan performa tinggi dan integrasi mendalam dengan sistem operasi. Namun, pengembangan dan pemeliharaan dua basis kode terpisah sering kali menyebabkan peningkatan waktu dan biaya (Mozharovskii, 2024). Meskipun aplikasi *native* memberikan performa lebih baik karena interaksi langsung dengan sistem operasi perangkat dan pemanfaatan kemampuannya secara maksimal, pendekatan ini membutuhkan sumber daya yang lebih besar dalam pengembangannya (Mozharovskii, 2024). Native unggul dalam hal performa, keamanan, konsistensi pengalaman pengguna dan akses fitur penuh, namun memiliki kelemahan dalam biaya proyek, waktu pengembangan dan kurangnya kemampuan penggunaan ulang kode (Guśpiel, 2024).

Sebagai solusi atas permasalahan tersebut, berbagai *framework* lintas *platform* telah dikembangkan untuk memungkinkan pengembang menulis kode sekali dan menjalankannya di beberapa *platform*. Flutter hadir sebagai *toolkit* UI *open-source* dari Google yang dapat membuat aplikasi *cross-platform* dengan satu basis kode sambil tetap mempertahankan aspek tampilan *native* (Olsson, 2020). Sementara itu, Kotlin Multiplatform (KMP) yang dikembangkan oleh JetBrains memungkinkan pengembang untuk menggunakan kembali kode secara efisien di berbagai *platform* seperti Android, iOS, web, desktop, dan aplikasi *server-side*, sambil mempertahankan opsi untuk memanfaatkan kode *native* saat diperlukan (Guśpiel, 2024).

KMP yang dikembangkan oleh JetBrains telah mengalami perkembangan signifikan sejak pertama kali dirilis pada tahun 2018. Pada November 2023, KMP telah dinyatakan stable dan siap untuk digunakan dalam produksi, memungkinkan pengembang untuk dengan aman menggunakan teknologi ini di berbagai *platform* seperti Android, iOS, dan JVM (Petrova, 2023). KMP tidak hanya menawarkan kemampuan berbagi kode antar *platform*, tetapi juga menyediakan interoperabilitas penuh dengan Java, *null safety* melalui sistem tipe Kotlin, serta pendekatan yang kompak dan pragmatis dalam mengatasi masalah pengembangan. Adopsi KMP telah meluas di kalangan perusahaan besar, dengan Netflix, Philips, McDonald's, 9GAG, dan Baidu menggunakan teknologi ini untuk berbagi kode antar *platform* mereka (Petrova, 2023).

Meskipun penelitian sebelumnya telah mengeksplorasi berbagai aspek pengembangan *cross-platform*, sebagian besar fokus pada metrik subjektif seperti developer *productivity* dan *code sharing capabilities*. Penelitian yang menganalisis performa teknis objektif menggunakan metodologi yang terkontrol masih sangat terbatas (Miller et al., 2020). Padahal, performa aplikasi merupakan faktor kritis yang mempengaruhi *user experience* dan *adoption rate* dalam *production environment* (Skantz, 2023).

Evaluasi performa *cross-platform framework* memerlukan pengukuran objektif terhadap metrik teknis yang dapat direproduksi. Penelitian terdahulu menunjukkan variasi signifikan dalam CPU *usage*, *memory consumption*, dan *battery efficiency* antara *framework* yang berbeda (Mozharovskii, 2024). Namun, metodologi pengukuran yang digunakan seringkali tidak konsisten dan kurang kontrolnya terhadap variabel eksternal, sehingga hasil yang diperoleh sulit untuk dibandingkan atau direplikasi (Olsson, 2020). Diperlukan pendekatan *systematic benchmarking* dengan *controlled experimental conditions* untuk menghasilkan data performa yang valid dan reliable. Meskipun terdapat alternatif pendekatan lain seperti studi literatur komparatif atau simulasi performa menggunakan *tool profiling*, penelitian ini memilih pendekatan *systematic benchmarking* dengan *controlled experimental conditions* karena memberikan data yang lebih objektif, dapat diukur, dan dapat direproduksi.

Dalam konteks *software engineering research*, penggunaan *quasi-experimental design* menjadi pilihan metodologi yang tepat ketika *true experimental design* tidak feasible dilakukan (Miller et al., 2020). *Quasi-experimental design* memungkinkan peneliti untuk melakukan *controlled comparison* dengan mengontrol variabel-variabel eksternal sambil tetap mempertahankan internal validity yang tinggi (Maciejewski, 2020). Pendekatan ini telah terbukti efektif dalam implementation research dan *technology evaluation studies* (Gopalan et al., 2020).

Pengembangan aplikasi menggunakan KMP tidak tanpa tantangan. Menurut Stanic & Ćirković (2024), tantangan utama yang dihadapi pengembang dalam mengembangkan aplikasi adalah terbatasnya jumlah library yang dapat bekerja di semua *platform*. Maka dari itu, tugas utama sebagai pengembang adalah memanfaatkan basis kode bersama di semua *platform* dan library yang berfungsi pada setiap *platform*. Jika sebuah library tidak didukung pada salah satu *platform*, pengembang harus mencari library khusus *platform* dan mengimplementasikan fungsionalitasnya secara terpisah.

Namun, dengan perkembangan yang pesat, masalah ini menjadi semakin berkurang, dan tahun ini kita bahkan dapat melihat bahwa banyak library telah merilis versi beta plugin mereka yang bekerja pada aplikasi multi-*platform*. Tim Android Google juga secara aktif mendukung KMP dengan merilis versi eksperimental *multi-platform* dari *library* Jetpack. Sejauh ini, mereka telah membuat beberapa library, termasuk Collections, DataStore, Annotations, dan Paging, kompatibel dengan KMP (Stanic & Ćirković, 2024).

Dalam konteks pendidikan tinggi modern, kebutuhan akan sistem manajemen akademik yang efisien menjadi semakin mendesak. Penelitian menunjukkan bahwa pengembangan sistem berbasis perangkat *mobile* dapat memperkecil jarak komunikasi dan meningkatkan akses informasi secara signifikan. Hartono & Yektyastuti (2018) mengungkapkan bahwa 83,1% pengguna internet di Universitas Djuanda lebih memilih mengakses sistem informasi melalui smartphone, menunjukkan tren yang mengarah pada digitalisasi layanan akademik berbasis *mobile*. Selain itu, Ratnasari (2023) menjelaskan bahwa implementasi sistem berbasis *mobile* mendukung konsep

kampus cerdas (*smart campus*), yang mengintegrasikan teknologi informasi untuk meningkatkan efektivitas layanan akademik di perguruan tinggi. Sebagai contoh, Telkom University telah mengimplementasikan aplikasi My Tel-U, sebuah sistem manajemen akademik berupa *super-app* yang menyediakan fitur seperti presensi, pengecekan jadwal, akses nilai, dan media sosial mikro bagi seluruh civitas akademika (Telkom University, 2024).

Namun, pengembangan sistem informasi akademik berbasis *mobile* juga menghadapi berbagai tantangan, seperti performa aplikasi, keamanan data, dan konsistensi pengalaman pengguna di berbagai *platform*. Rahmawati et al. (2024) menekankan pentingnya evaluasi kebutuhan awal dan pelatihan berkelanjutan untuk memastikan implementasi yang sukses serta peningkatan efisiensi operasional. Dengan demikian, keberhasilan pengembangan aplikasi manajemen akademik tidak hanya bergantung pada teknologi tetapi juga pada integrasi sistem dan komitmen semua pihak terkait.

Berdasarkan permasalahan tersebut, penelitian ini akan menganalisis penggunaan KMP dalam pengembangan aplikasi *mobile* lintas *platform* dengan fokus pada pengukuran efisiensi dan performa. Analisis akan dilakukan dengan membandingkan KMP dengan Flutter, yang saat ini menjadi salah satu *framework cross-platform* paling populer dengan 46% pangsa pasar pengembang (Vailshery, 2024). Selain itu, penelitian ini menggunakan metode *Experimental Research* untuk menganalisis perbandingan kedua *framework* secara sistematis dan terkontrol. Metode ini memungkinkan manipulasi variabel independen, seperti *framework* pengembangan KMP dan Flutter, dan pengamatan langsung terhadap pengaruhnya pada variabel dependen, seperti performa teknis dan efisiensi pengembangan. Dengan pengendalian yang ketat terhadap variabel eksperimental, metode ini menjadi pendekatan yang akurat dalam menentukan hubungan sebab-akibat (Akbar et al., 2023). Studi kasus berupa aplikasi manajemen akademik mahasiswa akan digunakan untuk menguji fitur penting, seperti autentikasi, jadwal kuliah, informasi nilai, dan direktori akademik. Pemilihan Flutter sebagai pembanding didasarkan pada popularitasnya, performa yang mendekati *native*, serta dukungan komunitas yang besar.

Melalui penelitian ini, diharapkan dapat memberikan pemahaman yang lebih mendalam tentang kemampuan dan batasan KMP dibandingkan dengan Flutter dalam konteks pengembangan aplikasi *mobile* lintas *platform*. Perbandingan akan dilakukan dengan mengukur enam metrik performa teknis yang objektif dan terukur: CPU *usage*, *memory consumption*, *battery consumption*, *application size*, *startup performance*, dan UI *performance*. Pemilihan metrik ini didasarkan pada karakteristik yang dapat diukur secara konsisten dan *reproducible* menggunakan *standardized measurement tools* (Mozharovskii, 2024; Skantz, 2023).

Hasil penelitian ini dapat membantu pengembang dan organisasi dalam membuat keputusan yang lebih informasi mengenai pemilihan antara KMP dan Flutter untuk pengembangan aplikasi *mobile* mereka, dengan mempertimbangkan kebutuhan spesifik proyek dalam konteks pengembangan aplikasi manajemen akademik modern. Selain itu, penelitian ini juga akan memberikan kontribusi pada perkembangan pengetahuan dalam bidang pengembangan aplikasi *mobile* lintas *platform*, khususnya dalam konteks perbandingan antara pendekatan berbasis KMP dan Flutter. Penelitian ini mengadopsi *systematic benchmarking protocol* dengan *physical device testing* untuk memastikan validitas dan reliabilitas hasil pengukuran, mengikuti best practices dalam *mobile application performance evaluation*.

Dengan menggunakan *quasi-experimental research design* dan fokus pada metrik performa teknis yang objektif, penelitian ini diharapkan dapat memberikan kontribusi empiris yang signifikan pada *body of knowledge* mengenai *cross-platform mobile development*. Hasil penelitian ini akan menyediakan baseline *performance* data yang dapat digunakan sebagai referensi untuk decision making dalam pemilihan teknologi pengembangan aplikasi *mobile*, serta menjadi *foundation* untuk *future research* dalam bidang *mobile application performance benchmarking*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, penelitian ini berupaya untuk menjawab pertanyaan penelitian yang berfokus pada perbandingan performa teknis objektif antara Kotlin Multiplatform dan Flutter dalam

pengembangan aplikasi *mobile* lintas *platform*. Berikut adalah rumusan masalah yang akan dijawab dalam penelitian ini:

1. Bagaimana perbandingan performa Kotlin Multiplatform dan Flutter pada *platform* Android dan iOS dalam pengembangan aplikasi manajemen akademik mahasiswa?
2. Bagaimana pola performa Kotlin Multiplatform dan Flutter pada *platform* Android dan iOS berdasarkan hasil *benchmarking*?
3. Bagaimana implikasi hasil *benchmarking* terhadap pemilihan *framework* lintas *platform* dalam konteks pengembangan aplikasi manajemen akademik mahasiswa?

1.3 Batasan Masalah

Untuk memastikan penelitian ini tetap fokus dan dapat diselesaikan dalam waktu yang ditentukan, beberapa batasan masalah ditetapkan sebagai berikut:

1. Penelitian ini akan membandingkan performa teknis aplikasi lintas *platform* menggunakan Kotlin Multiplatform dan Flutter melalui quasi-experimental design dengan controlled testing conditions.
2. Aplikasi yang dikembangkan untuk pengujian adalah aplikasi manajemen akademik mahasiswa dengan modul-modul berikut:
   a. Modul Autentikasi (*login* dengan *session management*)
   b. Modul Jadwal Kuliah (*view* dan *filter schedule*)
   c. Modul Informasi Nilai (*grades* dan *transcript*)
   d. Modul Media Sosial Mikro (*post, like, comment functionality*)
   e. Modul Direktori Akademik (*contact list* dengan *search*)
3. Hasil penelitian terbatas pada konteks pengembangan aplikasi manajemen akademik dengan fitur-fitur yang telah ditentukan dan pengukuran performa dibatasi pada enam metrik teknis objektif yang dapat diukur secara konsisten:
   a. CPU *Usage* (persentase penggunaan *processor*)
   b. *Memory Consumption* (penggunaan RAM dalam MB)
   c. *Battery Consumption* (*drain rate* dalam %/*hour*)
   d. *Application Size* (ukuran APK/IPA dalam MB)
   e. *Startup Performance* (*cold* dan *warm startup time* dalam ms)

f. UI *Performance* (*frame rate* dan *responsiveness* dalam FPS)

4. Pengembangan kedua aplikasi dilakukan oleh peneliti yang sama dengan upaya untuk menyeimbangkan tingkat keahlian pada kedua *framework*.
5. Pengukuran dilakukan menggunakan standardized testing protocol dengan:
   a. *Physical device testing* (tidak menggunakan emulator/simulator)
   b. Controlled environment conditions
   c. Spesifikasi *hardware* yang identik untuk masing-masing *framework* dalam satu *platform*
6. Penelitian tidak mengukur aspek subjektif seperti *developer experience*, *learning curve*, atau *community support* untuk menghindari bias dan mempertahankan objektifitas *scientific measurement*.
7. Penelitian ini menggunakan perangkat uji dengan spesifikasi berbeda, yaitu Android pada *device* kelas menengah (Redmi 10) dan iOS pada *device flagship* (iPhone 14 Pro). Kondisi ini menjaga *fairness* internal karena setiap *framework* dibandingkan pada perangkat yang sama dalam *platform* yang sama, tetapi membatasi *fairness* lintas *platform* sehingga hasil Android dan iOS tidak dapat digeneralisasi secara absolut. Selain itu, sistem operasi yang digunakan, yaitu Android 13 dan iOS 18 hingga saat ini belum terdapat penelitian yang membuktikan keduanya dapat dianggap setara. Oleh karena itu, hasil pengukuran lintas *platform* dalam penelitian ini harus dipahami dalam keterbatasan tersebut.

1.4 Tujuan Penelitian

Penelitian ini dilakukan dengan beberapa tujuan yang sejalan dengan rumusan masalah yang telah ditetapkan. Tujuan-tujuan tersebut adalah sebagai berikut:

1. Menganalisis perbandingan performa Kotlin Multiplatform dan Flutter pada *platform* Android dan iOS dalam pengembangan aplikasi manajemen akademik mahasiswa.
2. Mengidentifikasi pola performa Kotlin Multiplatform dan Flutter pada Android dan iOS berdasarkan hasil *benchmarking*.

3. Mengevaluasi implikasi hasil *benchmarking* terhadap pemilihan *framework* lintas *platform* dalam konteks pengembangan aplikasi manajemen akademik mahasiswa.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan kontribusi signifikan, baik dari segi teoritis maupun praktikal, dalam bidang *mobile application performance engineering* dan *cross-platform development*. Berikut adalah manfaat yang diharapkan dari penelitian ini:

Manfaat Teoritis:

1. Menyediakan bukti empiris berdasarkan *quasi-experimental design* mengenai perbandingan kinerja Kotlin Multiplatform dan Flutter menggunakan metrik objektif, terukur, serta dapat diverifikasi.
2. Berkontribusi pada pengembangan metodologi *mobile application performance benchmarking* dengan menyediakan protokol pengujian yang terstandar dan dapat diadopsi dalam penelitian perbandingan *framework* selanjutnya.
3. Memperkaya literatur dalam experimental *software* engineering melalui studi kasus implementasi *quasi-experimental design* untuk evaluasi teknologi dalam lingkungan penelitian akademik yang terkontrol.
4. Memberikan data kinerja dasar yang dapat dijadikan rujukan untuk penelitian lanjutan terkait pengembangan dan optimalisasi *framework* lintas *platform*.

Manfaat Praktikal:

1. Menyediakan data perbandingan kinerja objektif yang dapat digunakan oleh *software architects* dan *technical decision makers* dalam memilih kerangka kerja lintas *platform* yang tepat berdasarkan kriteria kinerja yang dapat diukur daripada preferensi subjektif.
2. Memberikan dasar ilmiah untuk pemilihan teknologi berbasis kinerja dalam pengembangan aplikasi *mobile* perusahaan, khususnya untuk

aplikasi dengan karakteristik serupa seperti sistem informasi manajemen akademik mahasiswa.

3. Menyediakan protokol perbandingan performa yang dapat diadopsi oleh tim pengembang untuk melakukan evaluasi kerangka kerja internal dengan ketelitian metodologis dan validitas statistik.
4. Mengidentifikasi karakteristik kinerja spesifik dan *trade-off* antara Kotlin Multiplatform dan Flutter yang dapat menginformasikan strategi optimalisasi dan upaya penyetelan kinerja dalam lingkungan produksi.
5. Memberikan panduan berbasis bukti untuk institusi akademis dalam pengembangan kurikulum mata kuliah pengembangan aplikasi *mobile*, khususnya dalam mengajarkan praktik pengembangan lintas *platform* dengan mempertimbangkan kinerja.

1.6 Sistematika Penulisan

Sistematika penulisan skripsi ini disusun untuk memberikan gambaran yang sistematis dan komprehensif mengenai perbandingan Kotlin Multiplatform dan Flutter dalam pengembangan aplikasi manajemen akademik mahasiswa berbasis *mobile* lintas *platform*. Struktur penulisan ini dirancang agar pembahasan dapat tersaji secara terarah, konsisten, dan mendalam sesuai dengan tujuan penelitian. Penulisan skripsi ini dibagi menjadi lima bab dengan rincian sebagai berikut:

1. **BAB I PENDAHULUAN**

   Bab ini menguraikan latar belakang masalah yang mendasari pentingnya penelitian perbandingan *framework* pengembangan aplikasi lintas *platform*. Selain itu, bab ini memuat rumusan masalah yang akan dijawab, batasan masalah yang membatasi ruang lingkup penelitian, tujuan yang ingin dicapai, manfaat penelitian baik secara teoritis maupun praktikal, serta sistematika penulisan yang memberikan gambaran keseluruhan struktur skripsi.

2. **BAB II KAJIAN TEORI**

   Bab ini menyajikan peta literatur dan landasan teori yang relevan dengan penelitian. Kajian teori mencakup konsep pengembangan

aplikasi *mobile*, pengembangan *native* dan lintas *platform*, penjelasan mendalam tentang Kotlin Multiplatform dan Flutter, sistem informasi akademik, serta metodologi experimental research. Bab ini memberikan dasar teoritis yang kuat untuk memahami konteks dan pendekatan penelitian yang dilakukan.

3. **BAB III METODOLOGI PENELITIAN**

   Bab ini menjelaskan desain penelitian yang menggunakan pendekatan quasi-experimental design dalam kerangka experimental research. Metodologi penelitian mencakup alat dan bahan yang digunakan, prosedur penelitian yang dilaksanakan, protokol pengukuran yang terstandarisasi, serta metode analisis data yang diterapkan. Bab ini memberikan kerangka metodologis yang jelas untuk memahami bagaimana penelitian dilakukan secara sistematis dan objektif.

4. **BAB IV HASIL DAN PEMBAHASAN**

   Bab ini menyajikan hasil implementasi aplikasi manajemen akademik mahasiswa menggunakan kedua *framework* yang diteliti. Pembahasan meliputi proses pengembangan aplikasi, tantangan yang dihadapi selama implementasi, hasil pengukuran metrik teknis dan aspek pengembangan, serta analisis perbandingan yang komprehensif. Bab ini menampilkan temuan empiris dari penelitian yang telah dilakukan.

5. **BAB V KESIMPULAN DAN SARAN**

   Bab ini berisi kesimpulan dari seluruh penelitian yang merangkum jawaban atas rumusan masalah yang telah diajukan. Selain itu, bab ini juga memuat saran untuk penelitian selanjutnya yang dapat mengembangkan atau melengkapi temuan dari penelitian ini. Kesimpulan disajikan berdasarkan analisis objektif dari data yang dikumpulkan, sementara saran diberikan untuk memberikan arah pengembangan penelitian di masa mendatang.