LAPORAN
PRAKTIKUM PEMROGRAMAN MOBILE
APLIKASI ANDROID “FANDOM HUB”

Disusun Oleh:

Sakina Matdoan (231552018154777)

Irzal Raisya Ramadhan (231551018154616)

Maria Rosalina Trisna Yangaluy (231552018154699)

LABORATORIUM BUSINESS INTELLIGENCE & MOBILE
PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI INFORMASI
UNIVERSITAS WIDYA GAMA MALANG

2025/2026 GASAL

LEMBAR PENGESAHAN
PRAKTIKUM PEMROGRAMAN MOBILE

Disusun Oleh:
Sakina Matdoan (231552018154777)
Irzal Raisya Ramadhan (231552018154616)
Maria Rosalina Trisna Yangaluy (231552018154699)

Pelaksanaan Praktikum	Tanggal: 20 Desember 2025
Pelaksanaan Asistensi Praktikum	Tanggal: 04 Desember 2025
Pelaksanaan Ujian Akhir Praktikum	Tanggal: 05 Januari 2025

Laboratorium Business Intelligence & Mobile – Program Studi S1 Teknik Informatika
Fakultas Sains dan Teknologi Informasi Universitas Widya Gama Malang

Malang, 21 Desember 2025

Mengetahui,	Menyetujui,
Ketua Program Studi S1 Teknik Informatika	Dosen Praktikum Pemrograman Mobile
Aviv Yuniar Rahman, ST., MT., Ph.D	**Ismail Akbar, S.Kom., M.Kom**
NDP. 2015 456	**NDP. 2025 497**

LEMBAR BIODATA MAHASISWA

Sakina Matdoan, NIM 231552018154777, mahasiswa Program Studi Teknik Informatika, Fakultas Sains dan Teknologi Informasi, Universitas Widya Gama. Saat ini, saya sedang menempuh pendidikan S1 pada program studi tersebut dan berada di semester 5. Untuk keperluan praktikum mata kuliah Pemrograman Mobile, saya dapat dihubungi melalui email di sakinamatdif@gmail.com dan nomor telepon/HP 081274135842. Portofolio proyek saya dapat dilihat di GitHub melalui tautan berikut: https://github.com/SakinaMatdoan/Android-FandomHub.git.

Irzal Raisya Ramadhan, NIM 231552018154616, mahasiswa Program Studi Teknik Informatika, Fakultas Sains dan Teknologi Informasi, Universitas Widya Gama. Saat ini, saya sedang menempuh pendidikan S1 pada program studi tersebut dan berada di semester 5. Untuk keperluan praktikum mata kuliah Pemrograman Mobile, saya dapat dihubungi melalui email di irzalraisyaramadhan@gmail.com dan nomor telepon/HP 083830238541. Portofolio proyek saya dapat dilihat di GitHub melalui tautan berikut: https://github.com/Irzal-npc/Android-FandomHub.git.

Maria Rosalina Trisna Yangaluy, NIM 231552018154699, mahasiswa Program Studi Teknik Informatika, Fakultas Sains dan Teknologi Informasi, Universitas Widya Gama. Saat ini, saya sedang menempuh pendidikan S1 pada program studi tersebut dan berada di semester 5. Untuk keperluan praktikum mata kuliah Pemrograman Mobile, saya dapat dihubungi melalui email di mariarosalina@gmail.com dan nomor telepon/HP 085218508734. Portofolio proyek saya dapat dilihat di GitHub melalui tautan berikut:https://github.com/Mariarosalina/Android-FandomHub.git.

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa, karena berkat izin-Nya kami dari Kelompok 1 dapat menyelesaikan proyek aplikasi Android kami yang berjudul **"Fandom Hub"**.

Proyek ini kami kerjakan untuk memenuhi tugas mata kuliah Pemrograman Mobile. Dalam aplikasi ini, kami mencoba membangun sebuah platform komunitas penggemar yang terintegrasi dengan fitur toko merchandise dan interaksi artis. Kami belajar banyak hal selama proses *coding* di Android Studio, terutama dalam mengelola *database* lokal menggunakan SQLite untuk memisahkan fungsi bagi tiga jenis pengguna: Penggemar, Artis, dan Admin.

Kami menyadari bahwa dalam proses pengerjaan aplikasi ini, masih banyak kekurangan dan tantangan yang kami hadapi. Namun, berkat kerjasama tim yang solid dan bantuan dari berbagai pihak, akhirnya aplikasi ini bisa berjalan sesuai dengan rencana. Kami ingin berterima kasih kepada:

1. **Bapak Ismail Akbar, S.Kom., M.Kom**, selaku dosen pengampu yang telah membimbing dan memberikan ilmunya kepada kami.
2. Orang tua dan teman-teman yang selalu memberikan dukungan moral selama kami bergadang menyelesaikan *error* di aplikasi ini.
3. Seluruh anggota tim yang sudah berjuang bareng, bagi tugas, dan saling bantu saat ada kendala teknis.

Kami tahu aplikasi Fandom Hub ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran dari teman-teman maupun dosen sangat kami harapkan agar ke depannya kami bisa mengembangkan aplikasi yang lebih baik lagi.

Semoga laporan dan aplikasi ini bisa bermanfaat bagi kita semua.

Malang, 21 Desember 2025

Tim Penyusun (Kelompok 1)

DAFTAR ISI

LEMBAR PENGESAHAN ii
LEMBAR BIODATA MAHASISWA iii
KATA PENGANTAR iv
DAFTAR ISI v
DAFTAR GAMBAR vi
DAFTAR TABEL viii
BAB 1 TEORI PEMROGRAMAN MOBILE 9
1.1 Definisi Pemrograman Mobile 9
1.2 Jenis-Jenis Platform Pemrograman Mobile 9
1.3 Framework Pemrograman Mobile 12
1.4 Komponen dalam Aplikasi Mobile 14
BAB 2 PERSIAPAN ALAT DAN BAHAN 16
2.1 Alat yang Digunakan 16
2.2 Bahan yang Dibutuhkan 17
BAB 3 LANGKAH-LANGKAH PEMROGRAMAN 22
3.1 Persiapan Awal 22
3.2 Proses Pengembangan Aplikasi 22
3.3 Pengujian Aplikasi 25
BAB 4 HASIL DAN PEMBAHASAN 26
4.1 Hasil Pengujian Fungsional dan Antarmuka 26
4.1.1 Alur Otentikasi dan Inisialisasi Aplikasi 26
4.1.2 Antarmuka dan Fungsionalitas Peran "Penggemar" (Fan) 27
4.1.3 Antarmuka dan Fungsionalitas Peran "Artis" (Artist) 34
4.1.4 Antarmuka dan Fungsionalitas Peran "Administrator" (Admin) 39
4.2 Analisis dan Pembahasan Hasil 41
4.2.1 Analisis Integrasi Multi-Role (Fan, Artist, Admin) 41
4.2.2 Analisis Alur Kerja Ekonomi dan Sosial 42
4.2.3 Pembahasan Efisiensi Arsitektur (UI-to-Repository) 42
4.2.4 Analisis Fitur Moderasi dan Keamanan 42
BAB 5 KESIMPULAN DAN SARAN 43
5.1 Kesimpulan 43
5.2 Saran 43
DAFTAR PUSTAKA 45
LAMPIRAN 46

DAFTAR GAMBAR

Gambar 1.1 Versi Android.....11

Gambar 2. 1 Struktur Manifest, Data Local, dan UI Admin 18

Gambar 2. 2 Struktur Komponen UI dan Fitur Fandom/Home 19

Gambar 2. 3 Struktur Fitur Market, Message, dan Profile 19

Gambar 2. 4 Struktur Subscription, Utils, dan Generated DAO.....20

Gambar 2. 5 Struktur Resource (Drawable, Mipmap, dan Values) 20

Gambar 3. 1 Implementasi Antarmuka.....23

Gambar 3. 2 Implementasi Arsitektur Data pada FandomRepository.kt. 23

Gambar 3. 3 Konfigurasi Rute Navigasi Aplikasi pada Screen.kt.....24

Gambar 4. 1 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Splash Screen".....26

Gambar 4. 2 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Login Screen".....27

Gambar 4. 3 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Register Screen Dengan Pilihan Role".....27

Gambar 4. 4 Antarmuka Beranda Utama Sisi Penggemar "Sebelum Dan Sesudah Memfollow Artis". 28

Gambar 4. 5 Antarmuka Penemuan Fandom Baru. 28

Gambar 4. 6 Antarmuka Detail Profil Fandom/Artis. 29

Gambar 4. 7 Antarmuka Detail Postingan dan Interaksi Komentar.....29

Gambar 4. 8 Antarmuka Katalog Marketplace Global.....30

Gambar 4. 9 Antarmuka Informasi Detail Produk. 30

Gambar 4. 10 Antarmuka Manajemen Keranjang Belanja.....31

Gambar 4. 11 Antarmuka Finalisasi Transaksi (Checkout).....31

Gambar 4. 12 Antarmuka Pusat Pemberitahuan. 32

Gambar 4. 13 Antarmuka Fitur Komunikasi Chat. 32

Gambar 4. 14 Antarmuka Manajemen Profil.....33

Gambar 4. 15 Riwayat Aktivitas Pengguna "Saved Posts Dan Order History". 33

Gambar 4. 16 Antarmuka Dasbor Kendali Utama Artis. 34

Gambar 4. 17 Antarmuka Manajemen Profil Fandom/Artis 35

Gambar 4. 18 Antarmuka Visualisasi Daftar Pengikut. 35

Gambar 4. 19 Antarmuka Analitik dan Statistik Fandom/Artis.....36

Gambar 4. 20 Antarmuka Pembuatan Konten (Postingan). 36

Gambar 4. 21 Antarmuka Inventaris Produk Merchandise. 37

Gambar 4. 22 Antarmuka Formulir Input Data Produk.....37

Gambar 4. 23 Antarmuka Manajemen Pesanan Pelanggan.....38

Gambar 4. 24 Antarmuka Pemantauan Ketersediaan Stok Produk.....38

Gambar 4. 25 Antarmuka Manajemen Seluruh Pengguna. 39

Gambar 4. 26 Antarmuka Validasi dan Persetujuan Akun Artis. 40

Gambar 4. 27 Antarmuka Pusat Moderasi Laporan Pelanggaran. 40

Gambar 4. 28 Antarmuka Eksekusi Moderasi (Ban User). 41

DAFTAR TABEL

Tabel 1. 1 Perbandingan Karakteristik Platform Android dan iOS 12

Tabel 1. 2 Perbandingan Framework Pemrograman Mobile.....14

Tabel 2. 1 Tabel Ringkasan Alat (Hardware & Software).....17

Tabel 2. 2 Tabel Ringkasan Bahan (Library & Dependency) 21

Table 3. 1 Hasil Pengujian Fitur Utama 25

Table 4. 1 Matriks Interaksi dan Validasi Hak Akses.....42

BAB 1
TEORI PEMROGRAMAN MOBILE

1.1 Definisi Pemrograman Mobile

Pemrograman mobile didefinisikan sebagai disiplin ilmu rekayasa perangkat lunak yang berfokus secara spesifik pada pengembangan aplikasi untuk perangkat genggam seperti ponsel pintar dan tablet. Dalam ekosistem digital yang sangat kompetitif saat ini, keberhasilan sebuah aplikasi seluler sangat bergantung pada kemampuan pengembang dalam mengelola data pengguna serta menanggapi permintaan informasi secara efektif. Jika sebuah aplikasi tidak mampu menangani kueri data dengan efisien, aplikasi tersebut akan sulit bersaing di pasar, sehingga peran pengembang dalam memilih arsitektur database yang tepat menjadi fondasi utama dalam siklus pengembangan perangkat lunak modern [1].

Secara fundamental, pengembangan aplikasi mobile menerapkan prinsip-prinsip teknik yang bertujuan untuk meningkatkan kualitas, proses pengerjaan, serta pemeliharaan produk agar lebih terukur dan adaptif terhadap perubahan teknologi. Pengembangan ini tidak sekadar memindahkan fungsi situs web ke dalam layar yang lebih kecil, melainkan melibatkan proses perancangan aplikasi yang mampu beroperasi secara optimal pada berbagai kategori perangkat dengan spesifikasi perangkat keras yang bervariasi. Pengembang harus mempertimbangkan aspek responsivitas antarmuka agar tetap konsisten meskipun dijalankan pada perangkat dengan ukuran layar, kapasitas memori, dan daya pemrosesan yang berbeda-beda [2].

Ditinjau dari sisi teknis, pemrograman mobile melibatkan penulisan kode sumber yang dirancang untuk berjalan di atas sistem operasi seluler yang dinamis, seperti platform Android. Android sendiri merupakan sebuah *software stack* yang komprehensif, terdiri dari sistem operasi berbasis Kernel Linux, *middleware*, serta aplikasi-aplikasi utama yang saling terintegrasi. Platform ini memungkinkan pengembang untuk menulis kode menggunakan bahasa pemrograman Java-like yang memanfaatkan pustaka-pustaka khusus yang dikembangkan oleh Google, namun dengan batasan tertentu dalam penggunaan kode asli (*native code*) untuk menjaga stabilitas sistem [3].

Dalam mengimplementasikan desain aplikasi, pengembang memerlukan lingkungan pengembangan terpadu atau *Integrated Development Environment* (IDE) seperti Android Studio. Melalui alat ini, pengembang dapat mengelola siklus hidup aplikasi (*activity lifecycle*) secara mendalam, mulai dari tahap inisialisasi hingga proses penghancuran (*destroy*) aplikasi oleh sistem operasi. Pemahaman terhadap siklus hidup ini sangat krusial guna memastikan aplikasi tidak membebani kinerja perangkat secara berlebihan, terutama dalam hal manajemen memori yang jauh lebih ketat dibandingkan dengan perangkat komputer konvensional atau server [4].

Perkembangan alat pengembangan mobile terus mengalami evolusi dengan hadirnya fitur-fitur canggih pada versi terbaru, seperti Android Studio Jellyfish. Salah satu aspek teknis yang semakin ditekankan adalah penggunaan *version catalog* melalui file libs.versions.toml untuk mengelola dependensi aplikasi secara terpusat. Dengan sistem manajemen pustaka yang lebih modern ini, pengembang dapat memastikan bahwa semua plugin dan pustaka eksternal yang digunakan dalam proyek aplikasi selalu dalam kondisi terbarui, aman, dan kompatibel dengan standar keamanan terbaru yang ditetapkan oleh penyedia sistem operasi [5].

Selain pendekatan pengembangan platform tunggal, tren pemrograman mobile saat ini telah bertransformasi ke arah pengembangan multiplatform menggunakan *framework* modern seperti Flutter. Flutter memungkinkan para pengembang untuk membangun aplikasi yang dapat berjalan secara lancar di sistem operasi Android, iOS, maupun Web hanya dengan menggunakan satu basis kode (*single codebase*). Pendekatan ini tidak hanya menguntungkan dari sisi efisiensi waktu dan biaya pengembangan, tetapi juga memudahkan proses sinkronisasi logika bisnis dan desain antarmuka pengguna secara konsisten di berbagai platform yang berbeda [6].

Dalam konteks implementasi praktis di lapangan, pemrograman mobile sering kali menjadi solusi utama untuk meningkatkan efisiensi manajemen data di berbagai sektor, termasuk manajemen inventaris atau *stock opname*. Aplikasi berbasis mobile terbukti mampu menggantikan proses administrasi manual yang sebelumnya masih bergantung pada pencatatan kertas yang berisiko tinggi terhadap kesalahan input atau kehilangan data. Dengan aplikasi mobile, petugas di lapangan dapat melakukan sinkronisasi data secara langsung ke sistem pusat, sehingga informasi mengenai stok barang atau transaksi dapat dipantau oleh pihak manajemen secara *real-time* dan akurat [7].

Terakhir, efisiensi operasional sebuah aplikasi mobile sangat didukung oleh penggunaan sistem manajemen database relasional yang tertanam langsung (*Embedded RDBMS*) seperti SQLite. SQLite merupakan library database yang sangat populer dalam pengembangan Android karena sifatnya yang ringan, cepat, dan tidak memerlukan konfigurasi server yang kompleks. Selain itu, pengembang modern kini sering memanfaatkan *Room Persistence Library* sebagai lapisan abstraksi di atas SQLite untuk mempermudah akses database melalui pemetaan objek, sehingga integritas data tetap terjaga dengan baik meskipun aplikasi dijalankan dalam kondisi tanpa koneksi internet atau luring [8].

1.2 Jenis-Jenis Platform Pemrograman Mobile

Platform pemrograman mobile merupakan landasan utama yang terdiri dari sistem operasi (OS) serta kumpulan alat pengembangan (*software development kit*) yang memungkinkan sebuah aplikasi dijalankan pada perangkat seluler secara optimal. Pemilihan platform yang tepat sangat menentukan efisiensi manajemen data, performa aplikasi, serta target pasar yang ingin dicapai. Saat ini, ekosistem perangkat bergerak

secara global didominasi oleh perangkat pintar yang sangat bergantung pada kemampuan sistem operasi dalam merespons kueri data pengguna untuk menjaga daya saing di pasar [1]. Selain itu, pemilihan platform juga memengaruhi bagaimana aplikasi berinteraksi dengan perangkat keras yang bervariasi, mulai dari smartphone hingga perangkat *wearable* [2].

A. Platform Android

Android merupakan platform berbasis Linux yang bersifat terbuka (*open-source*) dan dikembangkan secara kolaboratif oleh Google serta Open Handset Alliance. Sebagai sebuah *software stack* yang komprehensif, Android menyediakan kerangka kerja aplikasi yang memungkinkan pengembang menulis kode dalam bahasa Java-like yang memanfaatkan pustaka khusus yang telah dioptimalkan. Karakteristik utamanya yang bersifat terbuka membuat Android dapat diadopsi oleh berbagai manufaktur perangkat keras di seluruh dunia, menjadikannya platform dengan pangsa pasar terbesar saat ini [9]. Secara struktural, arsitektur Android mencakup Linux Kernel dan *Application Framework* yang dirancang untuk mendukung stabilitas dan keamanan eksekusi aplikasi [3].

Gambar 1.1 Versi Android

Pengembangan aplikasi pada platform Android umumnya dilakukan menggunakan lingkungan pengembangan terpadu atau *Integrated Development Environment* (IDE) bernama Android Studio. IDE ini menyediakan perangkat yang sangat lengkap bagi pengembang untuk melakukan pengodean, pengujian, hingga pemantauan siklus hidup aplikasi secara mendalam untuk memastikan efisiensi konsumsi sumber daya perangkat [4]. Bahasa pemrograman utama yang digunakan adalah Java dan Kotlin, di mana keduanya telah didukung sepenuhnya oleh sistem manajemen proyek modern dalam Android Studio untuk menjamin kompatibilitas dengan fitur-fitur sistem operasi terbaru [5].

B. Platform iOS

Seiring dengan perkembangan teknologi, kini hadir platform yang memungkinkan pengembangan aplikasi secara lintas sistem atau multiplatform menggunakan *framework* seperti Flutter. Berbeda dengan pendekatan pengembangan platform tunggal, Flutter memungkinkan para pengembang untuk membangun aplikasi yang mampu berjalan secara lancar baik di sistem operasi Android maupun iOS hanya dengan menggunakan satu basis kode (*single codebase*). Hal ini memberikan keuntungan besar dalam hal efisiensi waktu pengembangan dan kemudahan dalam melakukan pemeliharaan aplikasi secara terpusat tanpa harus menulis kode yang berbeda untuk setiap sistem operasi [6].

Aplikasi iOS umumnya dikembangkan menggunakan IDE bernama Xcode yang hanya tersedia pada sistem operasi macOS. Bahasa pemrograman yang digunakan dalam pengembangan iOS adalah **Swift**, sebuah bahasa modern yang cepat dan aman, atau **Objective-C** untuk proyek-proyek warisan (*legacy*). iOS dikenal memiliki standar desain UI/UX yang sangat ketat melalui *Human Interface Guidelines*, yang memastikan setiap aplikasi memiliki tampilan yang konsisten dan berkualitas tinggi.

C. Perbandingan Platform

Untuk memahami perbedaan teknis antara kedua platform tersebut, berikut adalah tabel ringkasan karakteristik Android dan iOS:

Tabel 1. 1 Perbandingan Karakteristik Platform Android dan iOS

Komponen Perbandingan	**Platform Android**	**Platform iOS**
Pengembang Utama	Google	Apple
Sistem Operasi Dasar	Linux (Open Source)	Unix-Based (Closed Source)
Bahasa Pemrograman	Java, Kotlin	Swift, Objective-C
Lingkungan Kerja (IDE)	Android Studio	Xcode
Dsitribusi Aplikasi	Google Play Store	Apple App Store
Keamanan	Terbuka (Fleksibel	Tertutup (Sangat Ketat)

1.3 Framework Pemrograman Mobile

Framework pemrograman mobile merupakan sekumpulan instruksi, pustaka (*library*), dan komponen antarmuka yang sudah disediakan untuk mempermudah

pengembang dalam membangun aplikasi tanpa harus menulis kode dari nol. Penggunaan *framework* bertujuan untuk menjaga standarisasi struktur kode, mempermudah manajemen dependensi, serta meningkatkan efisiensi waktu produksi aplikasi di dalam lingkungan pengembangan yang semakin kompleks [5]. Dalam praktiknya, pemilihan *framework* sangat bergantung pada kebutuhan fungsionalitas aplikasi, apakah akan dikembangkan secara spesifik untuk satu platform (*native*) atau untuk berbagai platform sekaligus (*multiplatform*) guna mempermudah proses sinkronisasi data dan pemeliharaan di masa depan [6].

A. React Native

React Native adalah *framework* berbasis JavaScript yang dikembangkan oleh Meta (dahulu Facebook). *Framework* ini sangat populer karena mengusung konsep "*Learn once, write anywhere*". React Native memungkinkan pengembang untuk membuat aplikasi yang memiliki performa menyerupai aplikasi *native* namun hanya menggunakan satu basis kode (*single codebase*) untuk platform Android dan iOS sekaligus. Hal ini dimungkinkan melalui jembatan (*bridge*) yang menghubungkan kode JavaScript dengan komponen asli dari masing-masing sistem operasi.

B. Xamarin

Xamarin adalah *framework* besutan Microsoft yang menggunakan bahasa pemrograman C# dengan dukungan ekosistem .NET. Xamarin memungkinkan pengembang untuk berbagi hingga 90% kode di seluruh platform utama (iOS, Android, dan Windows). Keunggulan utama Xamarin adalah kemampuannya dalam memberikan akses penuh ke API *native* dari masing-masing platform, sehingga aplikasi yang dihasilkan memiliki fungsionalitas yang sangat kuat dan sesuai dengan standar perangkat keras masing-masing.

C. Flutter

Flutter merupakan *framework* sumber terbuka (*open-source*) yang dikembangkan oleh Google untuk membangun aplikasi lintas platform. Dengan menggunakan bahasa pemrograman Dart, Flutter memungkinkan pengembang untuk membuat aplikasi yang berjalan di berbagai platform seperti Android, iOS, maupun Web hanya dalam satu kali proses pengodean (*single codebase*). Keunggulan utama dari *framework* ini adalah kemampuannya dalam menyajikan antarmuka pengguna yang konsisten dan halus di berbagai perangkat yang berbeda, sehingga mempercepat waktu pengembangan tanpa mengorbankan kualitas visual aplikasi [3].

D. Perbandingan Framework

Untuk memberikan gambaran mengenai perbedaan masing-masing *framework*, berikut disajikan tabel perbandingan teknis:

Tabel 1. 2 Perbandingan Framework Pemrograman Mobile

Kriteria	**React Native**	**Xamarin**	**Flutter**
Pengembang	Meta (Facebook)	Microsoft	Google
Bahasa Utama	JavaScript	C# (.NET)	Dart
Performa	Sangat Baik (Bridge)	Baik (Native-like)	Sangat Cepat (Skia Engine)
Popularitas	Sangat Tinggi	Menengah	Sangat Tinggi
Keunggulan	Komunitas Besar	Integrasi Enterprise	UI yang Indah & Cepat

1.4 Komponen dalam Aplikasi Mobile

Dalam pengembangan aplikasi **FandomHub**, terdapat beberapa komponen dasar yang membentuk arsitektur aplikasi agar dapat berjalan dengan optimal. Komponen-komponen ini mencakup aspek logika pemrograman, antarmuka, hingga manajemen data yang saling terintegrasi untuk menciptakan pengalaman pengguna yang mulus. Secara arsitektural, aplikasi mobile modern dibangun di atas tumpukan perangkat lunak (*software stack*) yang memungkinkan pengembang untuk mengelola fungsionalitas sistem operasi melalui pustaka-pustaka yang telah teroptimasi [3]. Komponen-komponen utama tersebut adalah sebagai berikut:

A. Bahasa Pemrograman

Bahasa pemrograman merupakan fondasi utama untuk membangun logika aplikasi. Dalam pengembangan Android modern, penguasaan atas alat pengembangan seperti Android Studio dan bahasa pemrograman resmi menjadi syarat mutlak untuk membangun aplikasi yang fungsional dan terstruktur dengan baik [4]. Selain bahasa Java yang telah lama menjadi standar, bahasa Kotlin kini semakin populer karena menawarkan fitur keamanan kode yang lebih tinggi dan sintaksis yang ringkas. Penggunaan alat pengembangan yang tepat seperti versi Android Studio terbaru memudahkan pengembang dalam mengelola dependensi melalui sistem *version catalog* guna menjamin stabilitas aplikasi dalam jangka panjang [5].

B. UI/UX (User Interface & User Experience)

Komponen ini menentukan bagaimana pengguna berinteraksi dengan aplikasi melalui elemen visual dan tata letak. Perancangan antarmuka pada aplikasi Android melibatkan pengaturan hirarki komponen melalui *View* dan *ViewGroup* untuk memastikan navigasi tetap intuitif bagi berbagai peran pengguna. Fokus pada *User*

Experience (UX) diarahkan agar alur navigasi tetap sederhana, terutama dalam menangani input gestur layar sentuh yang ergonomis. Desain yang intuitif sangat penting agar pengguna dapat mengoperasikan fitur-fitur kompleks, seperti fitur komunitas atau toko *merchandise*, secara lancar tanpa hambatan teknis yang berarti [2].

C. API (Application Programming Interface)

API berperan sebagai jembatan komunikasi antara aplikasi di perangkat mobile dengan layanan eksternal atau server pusat. Dalam arsitektur Android, integrasi API memungkinkan aplikasi untuk melakukan sinkronisasi data secara *real-time*, sehingga informasi yang ditampilkan pada perangkat pengguna selalu konsisten dengan database server. Implementasi API yang baik sangat krusial untuk mendigitalisasi proses manual menjadi sistem monitoring yang cepat dan akurat, seperti yang diterapkan pada sistem penagihan dan pengawasan data lapangan guna meningkatkan efisiensi operasional [7]. Pengujian terhadap komunikasi API ini biasanya dilakukan melalui metode *black box testing* untuk menjamin bahwa pertukaran data antara klien dan server berjalan sesuai dengan skenario fungsional yang diharapkan [10].

D. Database Lokal

Penyimpanan data pada perangkat mobile sangat penting agar aplikasi dapat bekerja secara responsif dan mendukung fungsionalitas luring (*offline*). Penggunaan sistem database relasional yang tertanam seperti **SQLite** dipilih karena sifatnya yang ringan dan tidak memerlukan konfigurasi server terpisah untuk menyimpan data internal aplikasi [8]. Dalam perkembangannya, pengembang kini dapat menggunakan **Room Persistence Library** sebagai lapisan abstraksi di atas SQLite. Room mempermudah proses kueri data dan pengelolaan database dengan menyediakan verifikasi waktu kompilasi, sehingga pengembang dapat membangun sistem manajemen data yang lebih efisien dan minim kesalahan dibandingkan menggunakan SQLite secara manual [1].

BAB 2
PERSIAPAN ALAT DAN BAHAN

2.1 Alat yang Digunakan

Alat yang digunakan terbagi menjadi dua kategori utama, yaitu perangkat keras untuk menjalankan proses pengembangan dan perangkat lunak sebagai lingkungan dan alat bantu pengembangan.

A. Perangkat Keras (Hardware):

1. **Komputer/Laptop:** Sebuah unit komputer dengan spesifikasi yang memadai untuk menjalankan Android Studio, emulator Android, dan proses *build* aplikasi. Spesifikasi minimal mencakup prosesor Intel Core i5 (atau setara), RAM 8 GB, dan penyimpanan SSD untuk mempercepat waktu kompilasi dan *loading*.
2. **Perangkat Fisik (Smartphone Android):** Digunakan untuk pengujian aplikasi secara langsung pada perangkat nyata. Pengujian di perangkat fisik penting untuk memastikan performa, responsivitas sentuhan, dan fungsionalitas fitur-fitur spesifik berjalan dengan baik di lingkungan pengguna asli.

B. Perangkat Lunak (Software):

1. **Sistem Operasi:** Windows, macOS, atau Linux yang mendukung versi terbaru dari Android Studio.
2. **Android Studio:** Sebagai *Integrated Development Environment* (IDE) resmi untuk pengembangan aplikasi Android. Android Studio menyediakan semua alat yang dibutuhkan dalam satu paket, termasuk:

    a. **Editor Kode:** Untuk menulis dan mengelola kode Kotlin.
    b. **Gradle Build System:** Sistem otomatisasi untuk mengelola *dependency* dan proses *build* aplikasi menjadi file APK (*Android Package Kit*).
    c. **Emulator Android (AVD):** Untuk menyimulasikan berbagai jenis perangkat Android.
    d. **Layout Inspector & Profiler:** Alat untuk *debugging* tampilan (UI) dan menganalisis performa aplikasi.

3. **Git & GitHub:** Digunakan untuk manajemen versi kode, pelacakan perubahan, dan kolaborasi tim.

Tabel 2. 1 Tabel Ringkasan Alat (Hardware & Software)

Kategori	Nama Alat	Spesifikasi / Kegunaan
Hardware	Laptop / PC	Intel i5, RAM 8GB, SSD (Dev Machine)
Hardware	Smartphone Android	Pengujian performa fisik & UI responsivitas
Software	Android Studio	IDE Utama (Ladybug/Jellyfish version)
Software	Git & GitHub	Version Control System (VCS)
Software	Android Emulator	Simulasi berbagai resolusi layar perangkat

2.2 Bahan yang Dibutuhkan

Bahan yang dibutuhkan adalah semua komponen digital, *library* kode, dan layanan eksternal yang diintegrasikan untuk membangun fungsionalitas aplikasi "Fandom Hub". Berikut adalah rincian bahan-bahan tersebut beserta dokumentasi implementasinya pada struktur proyek:

A. Komponen Utama Pengembangan

1. **Bahasa Pemrograman:**

    a. **Kotlin:** Bahasa pemrograman modern yang direkomendasikan Google. Digunakan untuk menulis seluruh logika bisnis dan interaksi antarmuka pengguna.

2. **Framework Antarmuka Pengguna (UI Framework):**

    a. **Jetpack Compose:** *Toolkit* UI deklaratif modern. Komponen utama yang digunakan antara lain: androidx.compose.ui (fondasi UI), androidx.compose.material3 (sistem desain Material You), dan androidx.navigation:navigation-compose (navigasi antar layar).

3. **Library dan SDK (Software Development Kit) Eksternal:**

    a. **Room Persistence Library:** Untuk mengelola database lokal (SQLite) guna menyimpan data pengguna atau data *offline*.
    b. **DataStore:** Solusi penyimpanan data sederhana seperti status *login* atau pengaturan pengguna.

c. **Coil:** *Library* untuk memuat gambar dari internet (URL) dan menampilkannya pada UI.
d. **Firebase/Google Services:** Mencakup *Firebase Authentication* (login), *Cloud Firestore* (database cloud), dan *Cloud Storage* (penyimpanan file gambar).

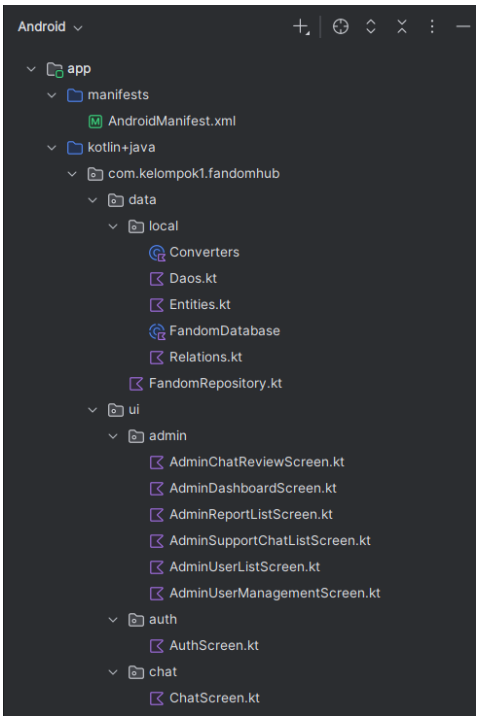
4. **Aset Digital:**

a. **Ikon:** Menggunakan *Material Icons Extended* seperti Icons.Default.Person dan Storefront.
b. **Aset Visual:** Mencakup logo aplikasi, gambar ilustrasi, atau *font* kustom yang disimpan dalam direktori res.

B. Dokumentasi File Proyek

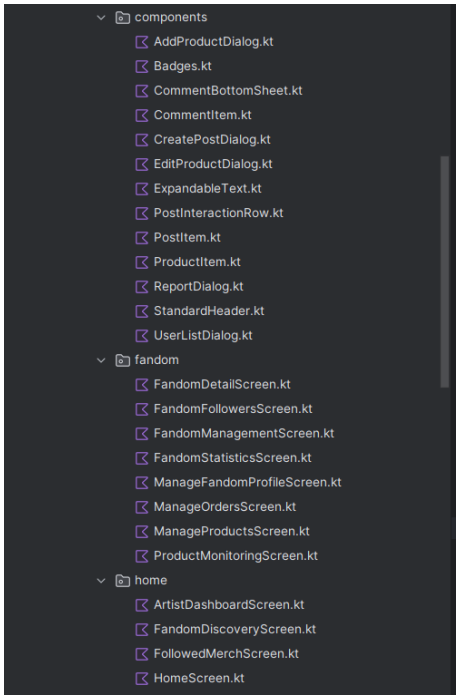
Seluruh komponen di atas diorganisir ke dalam struktur folder yang sistematis di Android Studio. Berikut adalah tampilan *class* dan paket yang membentuk aplikasi Fandom Hub:

1. **Arsitektur Data dan Admin** : Bagian ini menunjukkan file konfigurasi AndroidManifest.xml, paket data lokal (Room), serta layar khusus untuk peran Admin.



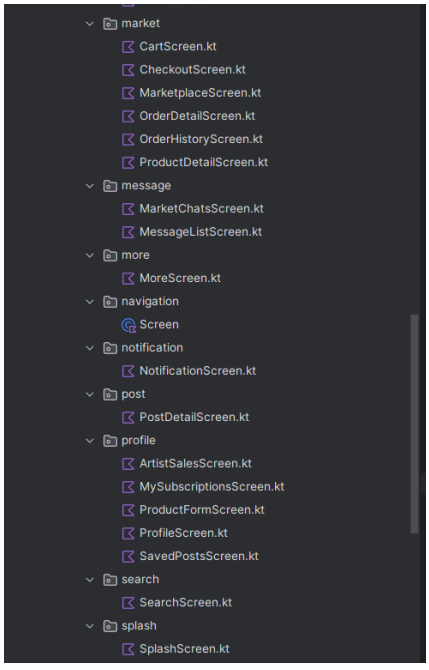
Gambar 2. 1 Struktur Manifest, Data Local, dan UI Admin

2. **Komponen UI dan Fitur Utama** : Screenshot ini menampilkan komponen antarmuka yang digunakan kembali serta layar untuk fitur *Fandom* dan halaman beranda (*Home*).



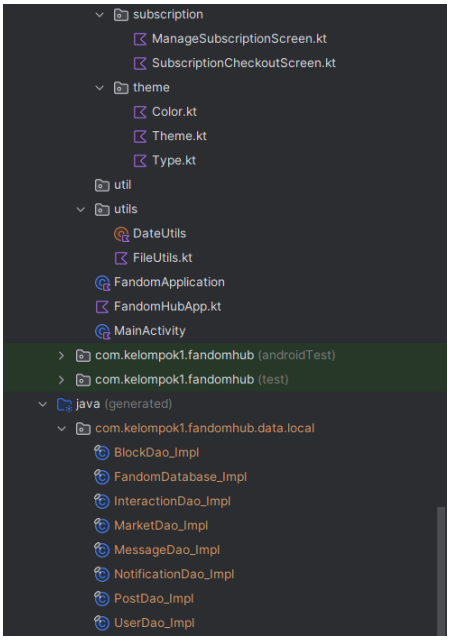
Gambar 2. 2 Struktur Komponen UI dan Fitur Fandom/Home

3. **Fitur Marketplace dan Komunikasi** : Menampilkan daftar *class* untuk fitur belanja (*Market*), sistem pesan (*Message*), notifikasi, dan manajemen profil.



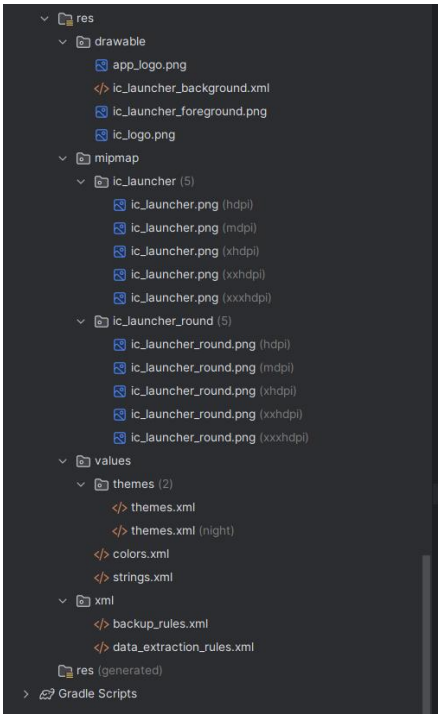
Gambar 2. 3 Struktur Fitur Market, Message, dan Profile

4. **Manajemen Langganan dan Implementasi Database** : Menunjukkan struktur fitur langganan (*Subscription*) serta file *DAO Implementation* yang dihasilkan oleh Room Database.



Gambar 2. 4 Struktur Subscription, Utils, dan Generated DAO

5. **Aset Sumber Daya (Resources) :** Menampilkan seluruh aset gambar, logo, ikon, serta definisi warna dan tema aplikasi.



Gambar 2. 5 Struktur Resource (Drawable, Mipmap, dan Values)

B. Ringkasan Bahan Pengembangan

Untuk merangkum seluruh komponen digital dan pustaka yang digunakan dalam pembangunan aplikasi Fandom Hub, berikut adalah tabel rincian bahan pengembangan:

Tabel 2. 2 Tabel Ringkasan Bahan (Library & Dependency)

Jenis Bahan	**Nama Library/ Framework**	**Fungsi dalam Aplikasi**
Bahasa	Kotlin	Logika utama dan backend lokal
UI Framework	Jetpack Compose (Material 3)	Desain antarmuka deklaratif modern
Navigation	Navigation Compose	Manajemen perpindahan antar layar
Database	Room Persistence	Penyimpanan data lokal (SQLite)
Storage	DataStore	Penyimpanan status login & preferensi
Image Loader	Coil	Mengambil & menampilkan gambar URL
Cloud Service	Firebase (Auth & Firestore)	Manajemen User & Database Cloud

BAB 3
LANGKAH-LANGKAH PEMROGRAMAN

3.1 Persiapan Awal

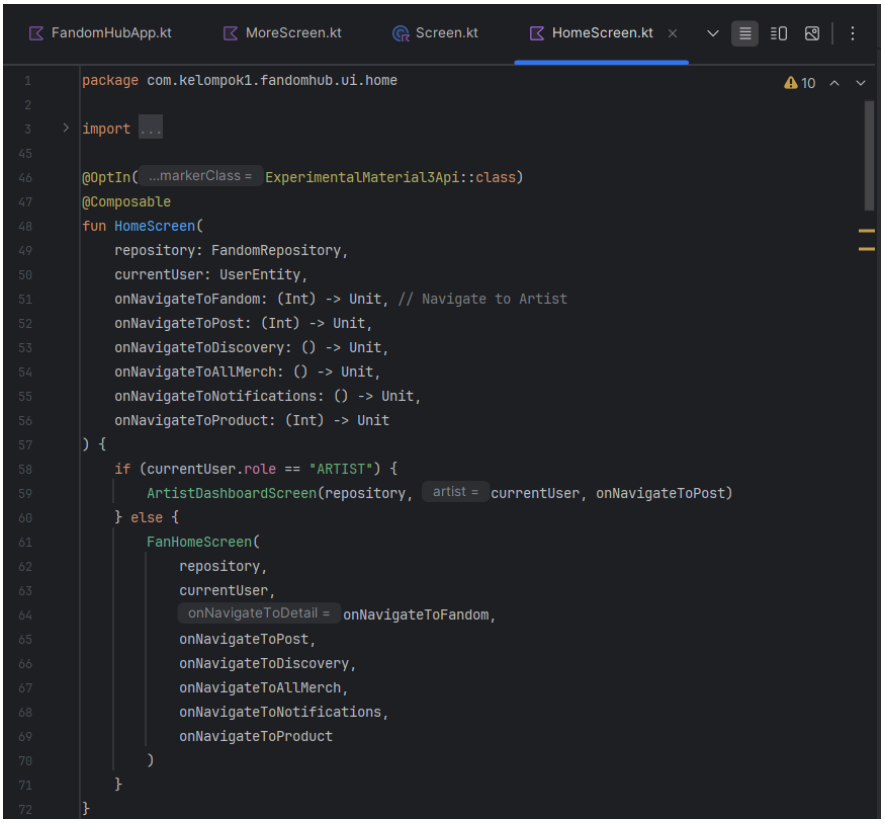
Sebelum proses penulisan kode dimulai, beberapa langkah persiapan esensial dilakukan untuk memastikan lingkungan pengembangan telah siap dan terkonfigurasi dengan benar agar tidak terjadi kendala teknis di tengah pengembangan.

1. **Instalasi dan Konfigurasi IDE (Integrated Development Environment):** Proses pengembangan aplikasi Fandom Hub sepenuhnya dilakukan menggunakan **Android Studio** versi terbaru, yang menyediakan fitur editor kode cerdas, *debugger*, emulator, serta integrasi sistem *build* Gradle yang diperlukan untuk aplikasi berbasis Android modern.
2. **Pembuatan Proyek dan Spesifikasi Teknis:** Proyek baru diinisialisasi dengan menggunakan *template* **Empty Activity** berbasis **Jetpack Compose**, dengan konfigurasi nama paket com.kelompok1.fandomhub dan **Minimum SDK API Level 29 (Android 10)** untuk menjamin kompatibilitas pada perangkat modern.
3. **Pengaturan Dependensi dan Library:** Langkah krusial dilakukan dengan mendaftarkan berbagai pustaka eksternal pada file build.gradle.kts, mencakup **Jetpack Compose** untuk UI, **Room** untuk database lokal, **Navigation Compose** untuk alur layar, **DataStore** untuk preferensi, serta **Coil** untuk pemuatan gambar asinkron.
4. **Konfigurasi Pengujian Perangkat:** Disiapkan sebuah *Android Virtual Device* (AVD) dengan API level 31 (Android 12) serta konfigurasi perangkat fisik melalui *USB Debugging* untuk memastikan aplikasi dapat diuji secara responsif pada berbagai kondisi layar dan performa riil.

3.2 Proses Pengembangan Aplikasi

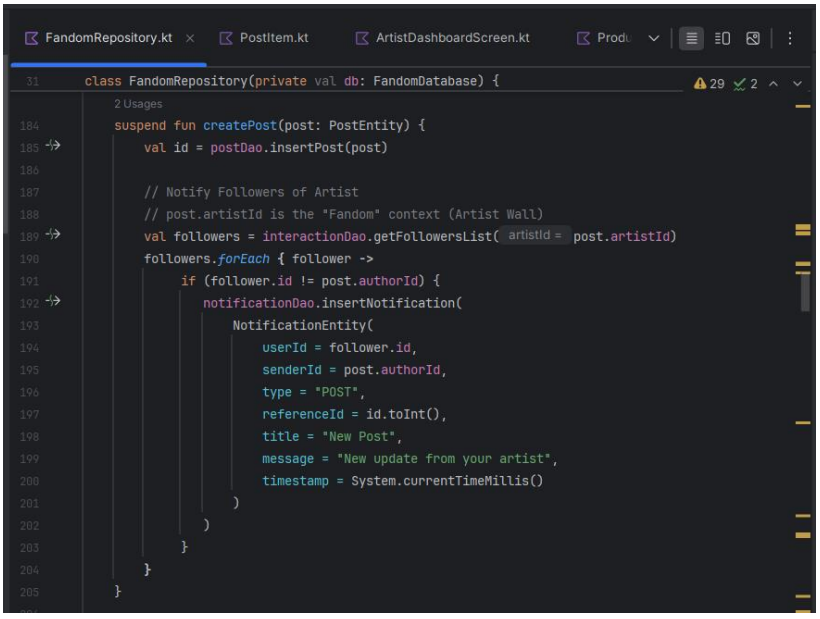
Sub-bab ini menjelaskan proses pembangunan fitur-fitur utama aplikasi Fandom Hub dengan menerapkan pola desain dan teknologi terbaru di ekosistem Android.

1. **Desain Antarmuka Deklaratif dengan Jetpack Compose:** Seluruh tampilan aplikasi dibangun menggunakan fungsi-fungsi @Composable tanpa menggunakan XML. Implementasi pada HomeScreen.kt menunjukkan penggunaan logika *role-based UI* untuk membedakan antarmuka antara peran *Artist* dan *Fan* secara dinamis.



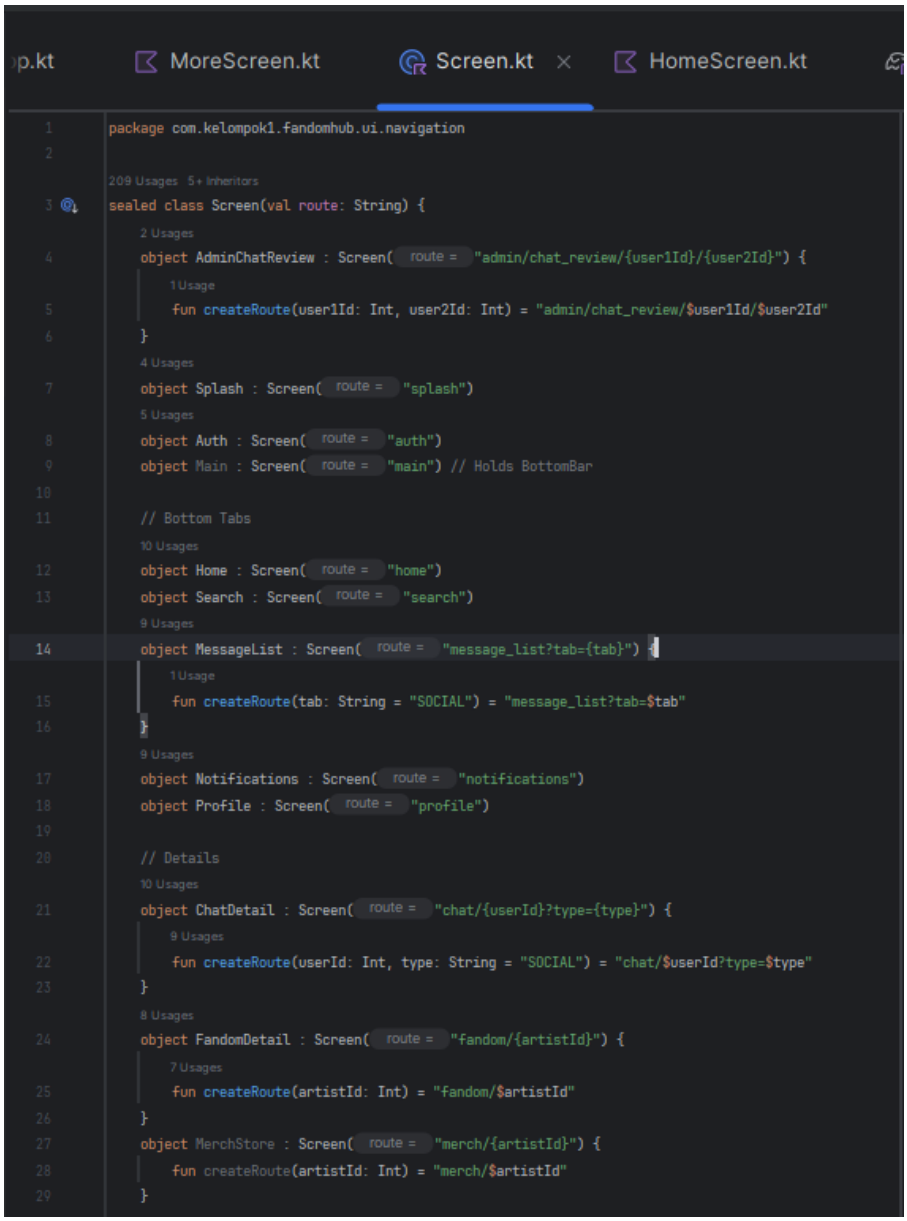
Gambar 3. 1 Implementasi Antarmuka Deklaratif menggunakan Jetpack Compose.

2. **Implementasi Arsitektur MVVM (Model-View-ViewModel):** Aplikasi menerapkan pola MVVM untuk memisahkan logika bisnis dari antarmuka. Peran **Model** dijalankan oleh FandomRepository.kt yang mengolah data sebelum dikirim ke **ViewModel** untuk ditampilkan di **View**. Hal ini memastikan kode lebih mudah dikelola dan diuji secara mandiri.



Gambar 3. 2 Implementasi Arsitektur Data pada FandomRepository.kt.

3. **Manajemen Database Lokal dengan Room:** Implementasi database dilakukan dengan mendefinisikan *Entity* untuk struktur tabel dan *DAO (Data Access Object)* untuk operasi kueri. Repository (seperti pada Gambar 8.8) menggunakan DAO tersebut untuk menyimpan data secara permanen, seperti saat menyimpan unggahan baru atau notifikasi ke dalam penyimpanan lokal.
4. **Sistem Navigasi Terpusat:** Navigasi antar layar dikelola menggunakan rute berbasis string yang didefinisikan secara terpusat dalam *Sealed Class* Screen. Hal ini memastikan setiap perpindahan halaman, baik rute statis maupun dinamis yang membawa parameter (seperti ID pengguna), berjalan secara konsisten dan aman.



Gambar 3. 3 Konfigurasi Rute Navigasi Aplikasi pada Screen.kt.

3.3 Pengujian Aplikasi

Pengujian dilakukan secara bertahap dan iteratif untuk memastikan fungsionalitas aplikasi berjalan sesuai skenario dan bebas dari kesalahan teknis (*bug*).

1. **Pengujian Fungsionalitas pada Emulator:** Dilakukan verifikasi alur kerja utama seperti validasi *form login*, pemisahan hak akses antara akun *Artist* dan *Fan*, hingga keberhasilan pengiriman notifikasi otomatis saat membuat unggahan baru.
2. **Pengujian pada Perangkat Fisik:** Aplikasi diinstal pada perangkat Android asli untuk memastikan antarmuka tetap presisi pada berbagai resolusi layar, serta menguji performa navigasi antar rute yang telah didefinisikan pada file Screen.kt.
3. **Debugging dan Analisis Logcat:** Menggunakan fitur Logcat di Android Studio untuk memantau aktivitas aplikasi secara *real-time*, terutama untuk memastikan data dari FandomRepository.kt berhasil tersimpan ke dalam *Room Database* tanpa hambatan teknis.

Sebagai ringkasan dari tahap pengujian, berikut adalah tabel hasil uji coba fitur utama:

Table 3. 1 Hasil Pengujian Fitur Utama

Fitur Utama	**Skenario Pengujian**	**Hasil yang Diharapkan**	**Status**
Autentikasi & Role	Menginput kredensial login akun Artist dan Fan.	Sistem berhasil memvalidasi dan menampilkan dashboard yang sesuai dengan peran pengguna.	**Berhasil**
Penyimpanan Data	Melakukan posting konten atau registrasi.	Data konten dan notifikasi tersimpan secara permanen pada *Room Database* melalui Repository.	**Berhasil**
Navigasi Terpusat	Menekan menu pada *BottomBar* atau link detail.	Layar berpindah sesuai rute rute pada Screen.kt tanpa terjadi *crash*.	**Berhasil**
Manajemen Gambar	Memuat foto profil atau gambar postingan.	Gambar berhasil tampil dengan lancar menggunakan pustaka Coil.	**Berhasil**

BAB 4
HASIL DAN PEMBAHASAN

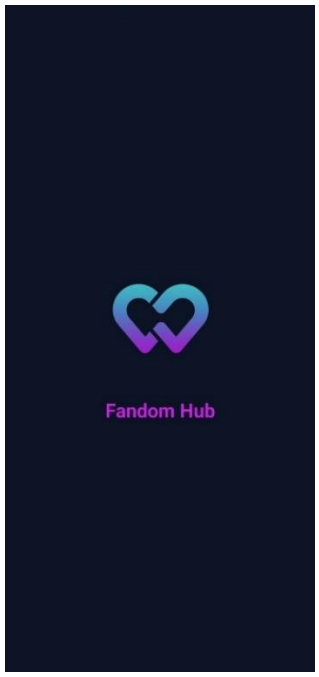
4.1 Hasil Pengujian Fungsional dan Antarmuka

Pengujian dilakukan untuk memvalidasi setiap alur kerja, mulai dari otentikasi hingga fitur spesifik yang dirancang untuk masing-masing peran.

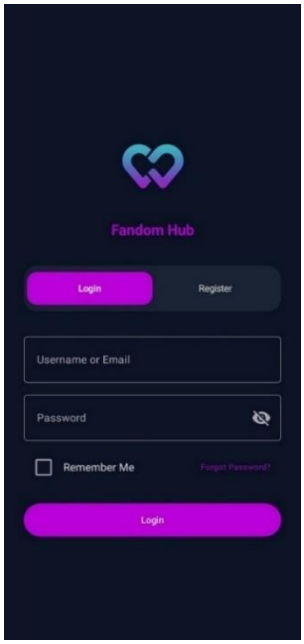
4.1.1 Alur Otentikasi dan Inisialisasi Aplikasi

Tahapan ini adalah gerbang awal bagi semua pengguna dan menunjukkan bagaimana aplikasi mengelola basis data serta sesi pengguna.

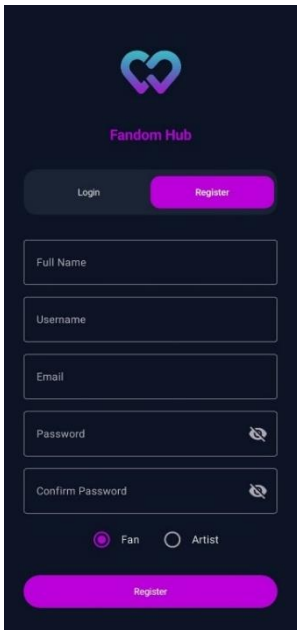
1. **Inisialisasi FandomApplication.kt**: Sesuai deklarasi di AndroidManifest.xml, kelas ini dijalankan pertama kali saat aplikasi dibuka. Di sinilah inisialisasi database Room terjadi, memastikan sistem siap digunakan sebelum antarmuka ditampilkan.
2. **Halaman Splash (Screen.Splash)**: Layar ini bertugas memeriksa status login melalui *DataStore Preferences*. Jika sesi ditemukan, aplikasi langsung mengarahkan pengguna ke dasbor yang sesuai (Fan/Artist/Admin).
3. **Halaman Login (Screen.Login)**: Form input kredensial pengguna yang divalidasi langsung ke database Room.
4. **Halaman Registrasi (Screen.Register)**: Tahap krusial di mana pengguna memilih peran (**Fan** atau **Artist**). Khusus akun Artist, status pendaftaran akan masuk ke antrean **Approval Admin** sebelum fitur publikasi konten diaktifkan



Gambar 4. 1 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Splash Screen".



Gambar 4. 2 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Login Screen".

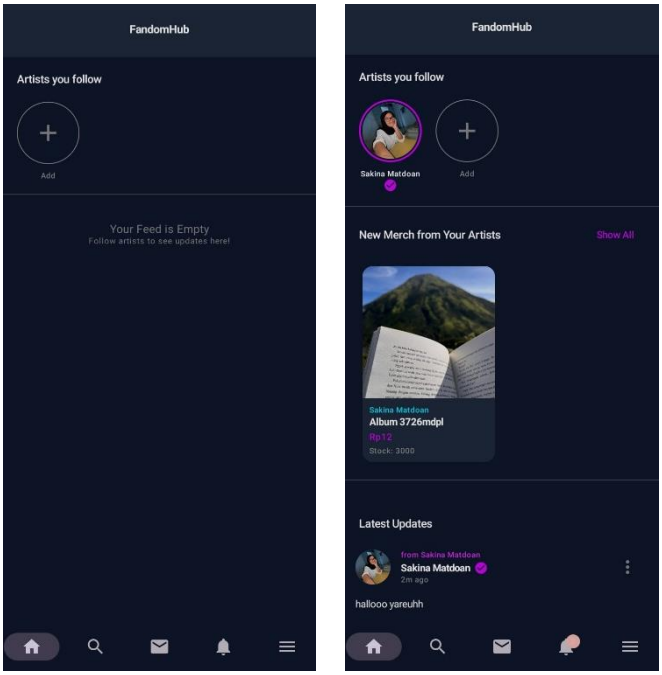


Gambar 4. 3 Antarmuka Proses Otentikasi dan Registrasi Pengguna "Register Screen Dengan Pilihan Role".

4.1.2 Antarmuka dan Fungsionalitas Peran "Penggemar" (Fan)

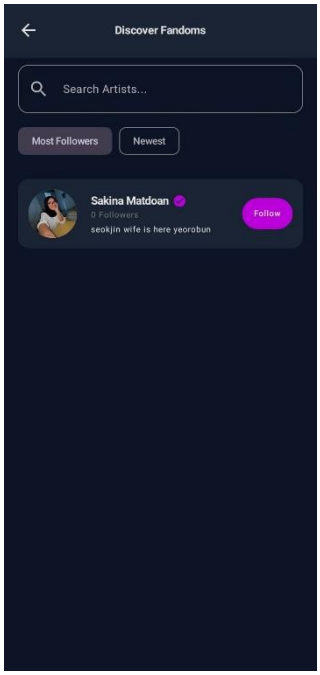
Penggemar memiliki akses luas untuk berinteraksi dengan komunitas dan melakukan transaksi ekonomi.

1. **Halaman Beranda Utama (Screen.Home)**: Bertindak sebagai pusat informasi yang menampilkan *feed* postingan dari artis-artis yang diikuti. Data ditampilkan secara reaktif menggunakan collectAsState, memastikan konten selalu terbaru.



Gambar 4. 4 Antarmuka Beranda Utama Sisi Penggemar "Sebelum Dan Sesudah Memfollow Artis".

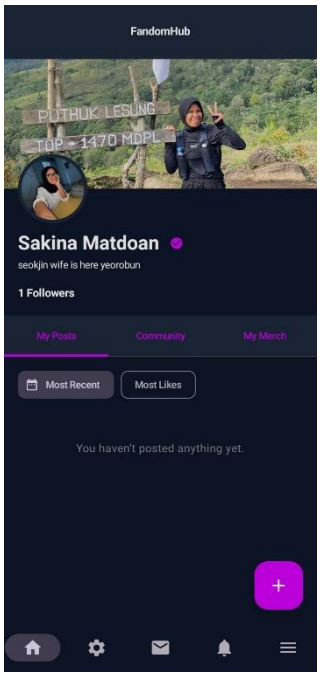
2. **Penemuan Fandom (Screen.FandomDiscovery)**: Halaman rekomendasi yang memudahkan pengguna mencari artis baru berdasarkan minat. Pengguna dapat langsung menekan tombol "Follow" pada kartu artis di halaman ini.



Gambar 4. 5 Antarmuka Penemuan Fandom Baru.

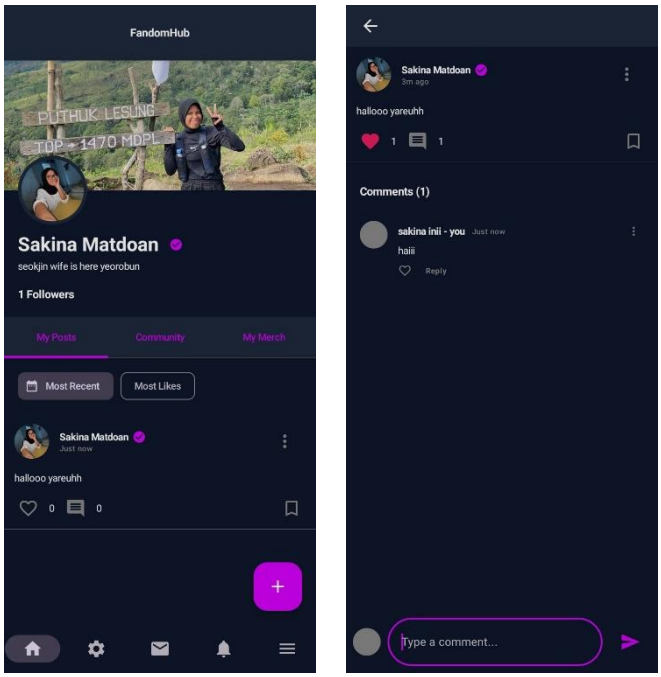
3. **Halaman Detail Fandom (Screen.FandomDetail)**: Menyajikan profil lengkap seorang artis, mulai dari foto *header*, bio, jumlah pengikut, hingga

integrasi tab untuk melihat daftar postingan khusus dan merchandise yang dijual oleh artis tersebut.



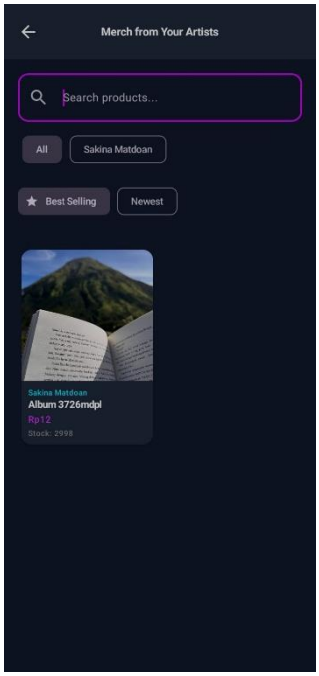
Gambar 4. 6 Antarmuka Detail Profil Fandom/Artis.

4. **Halaman Detail Postingan (Screen.PostDetail):** Memberikan ruang bagi penggemar untuk melihat konten teks atau gambar secara utuh dan berinteraksi melalui kolom komentar yang terintegrasi secara *real-time.*



Gambar 4. 7 Antarmuka Detail Postingan dan Interaksi Komentar.

5. **Pusat Marketplace (Screen.Marketplace)**: Katalog produk merchandise dari seluruh artis. Dilengkapi dengan kategori filter untuk mempermudah penggemar mencari barang tertentu sesuai keinginan.



Gambar 4. 8 Antarmuka Katalog Marketplace Global.

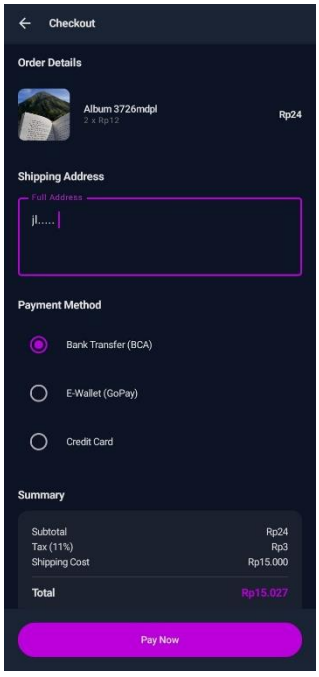
6. **Halaman Detail Produk (Screen.ProductDetail)**: Menampilkan spesifikasi lengkap merchandise, termasuk deskripsi produk, stok yang tersedia, dan harga. Pengguna dapat menambahkan produk langsung ke keranjang belanja dari layar ini.

Gambar 4. 9 Antarmuka Informasi Detail Produk.

7. **Keranjang Belanja (Screen.Cart)**: Memfasilitasi pengguna untuk mengelola barang yang akan dibeli. Pengguna dapat menambah atau mengurangi jumlah barang dan melihat total harga sebelum melanjutkan ke pembayaran.

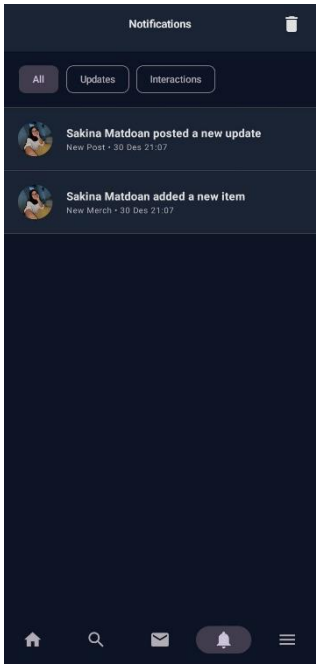
Gambar 4. 10 Antarmuka Manajemen Keranjang Belanja.

8. **Proses Checkout (Screen.Checkout)**: Halaman finalisasi transaksi yang mencakup pemilihan metode pembayaran dan alamat pengiriman sebelum pesanan diproses oleh sistem.



Gambar 4. 11 Antarmuka Finalisasi Transaksi (Checkout).

9. **Pusat Notifikasi (Screen.Notifications)**: Terbagi menjadi tab "Updates" untuk konten baru dan "Interactions" untuk balasan komentar atau *like*, menjaga pengguna tetap terinformasi tentang aktivitas terbaru.

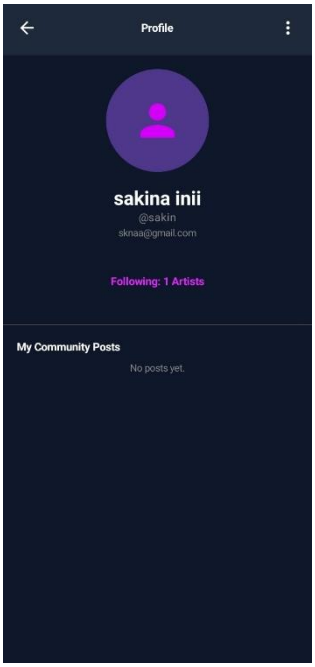


Gambar 4. 12 Antarmuka Pusat Pemberitahuan.

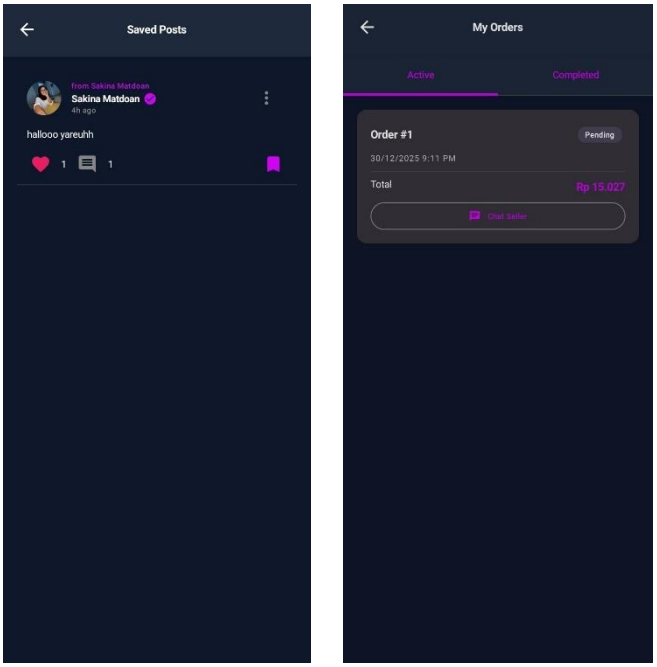
10. **Halaman Pesan (Screen.MessageList & Screen.ChatDetail)**: Menampilkan daftar percakapan pribadi. Sistem memvalidasi bahwa hanya pengikut/pelanggan yang dapat memulai percakapan dengan artis demi menjaga privasi dan keamanan.

Gambar 4. 13 Antarmuka Fitur Komunikasi Chat.

11. **Halaman Profil & Manajemen Personal (Screen.Profile, SavedPosts, OrderHistory)**: Ruang bagi pengguna untuk mengelola data langganan, melihat kembali postingan yang disimpan, serta memantau status pesanan barang yang telah dibeli.



Gambar 4. 14 Antarmuka Manajemen Profil.

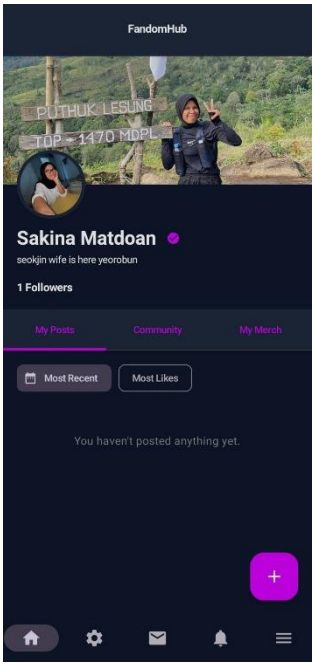


Gambar 4. 15 Riwayat Aktivitas Pengguna "Saved Posts Dan Order History".

4.1.3 Antarmuka dan Fungsionalitas Peran "Artis" (Artist)

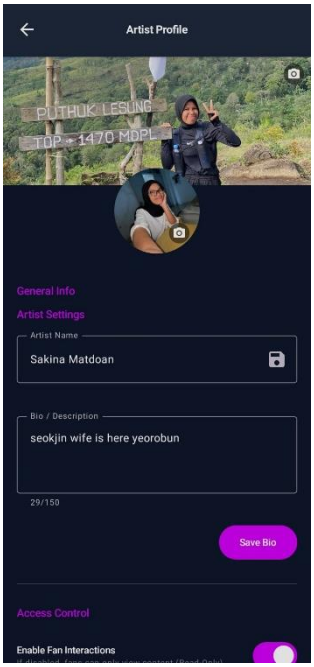
Artis memiliki wewenang khusus untuk mengelola komunitas (fandom), mempublikasikan konten, serta melakukan manajemen penjualan merchandise. Berikut adalah rincian halaman yang diimplementasikan:

1. **Dasbor Utama Artis (Screen.ArtistDashboard)** Merupakan pusat navigasi utama bagi peran Artis. Halaman ini menyajikan ringkasan performa fandom, termasuk jumlah total pengikut dan akses cepat menuju menu manajemen produk, pesanan, dan postingan.



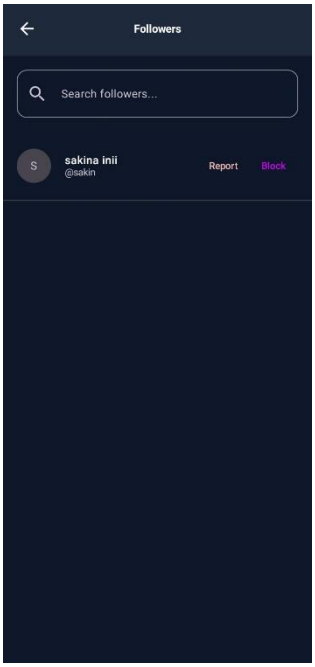
Gambar 4. 16 Antarmuka Dasbor Kendali Utama Artis.

2. **Manajemen Profil Fandom (Screen.FandomManagement)** Halaman ini memungkinkan artis untuk melakukan personalisasi terhadap profil publik mereka, seperti memperbarui foto *header*, foto profil, serta bio artis untuk menarik lebih banyak penggemar.



Gambar 4. 17 Antarmuka Manajemen Profil Fandom/Artis

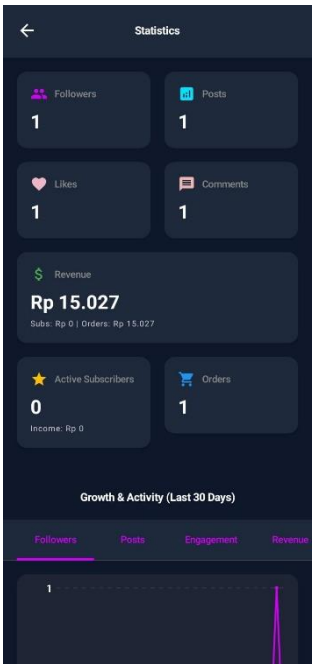
3. **Daftar Pengikut Fandom (Screen.FandomFollowers)** Artis dapat melihat daftar lengkap pengguna (Fan) yang telah mengikuti mereka. Fitur ini memvalidasi bahwa database relasional antara tabel User dan Fandom telah terintegrasi dengan baik.



Gambar 4. 18 Antarmuka Visualisasi Daftar Pengikut.

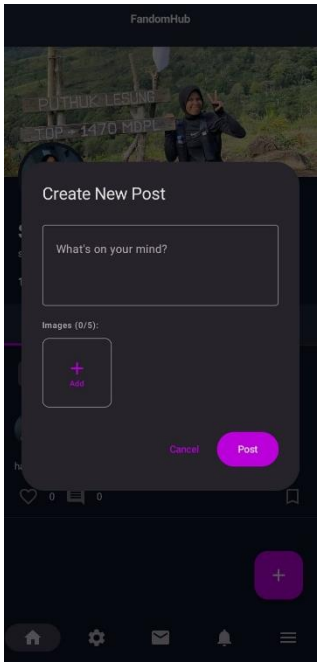
4. **Statistik Pertumbuhan Fandom (Screen.FandomStatistics)** Menampilkan data analitik mengenai pertumbuhan pengikut dan interaksi pengguna dalam

bentuk grafik atau angka statistik. Ini membantu artis memahami perkembangan komunitas mereka dari waktu ke waktu.



Gambar 4. 19 Antarmuka Analitik dan Statistik Fandom/Artis.

5. **Formulir Postingan Baru (Screen.PostForm)** Halaman untuk mengunggah konten "Tweet" atau postingan teks. Pengujian membuktikan bahwa setiap postingan yang disimpan akan memicu fungsi NotificationRepository untuk mengirimkan pemberitahuan secara otomatis kepada seluruh pengikut.

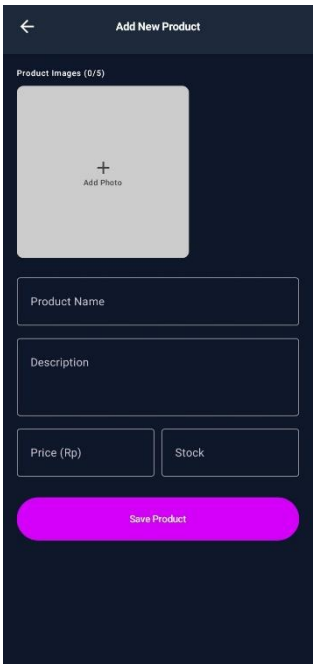


Gambar 4. 20 Antarmuka Pembuatan Konten (Postingan).

6. **Manajemen Produk Merchandise (Screen.ManageProducts)** Daftar inventaris merchandise yang dijual oleh artis. Di sini artis dapat memantau stok barang, harga, dan melakukan penghapusan produk yang sudah tidak tersedia.

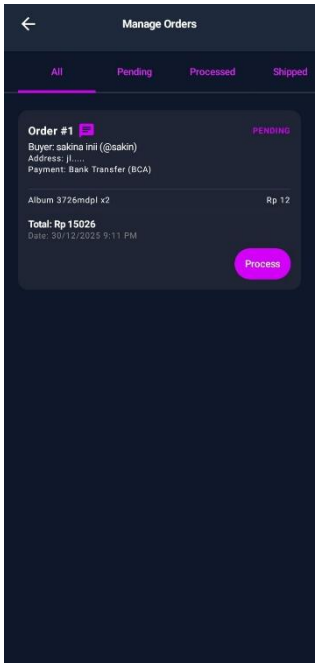
Gambar 4. 21 Antarmuka Inventaris Produk Merchandise.

7. **Formulir Produk (Screen.ProductForm)** Digunakan untuk menambah merchandise baru atau memperbarui data produk lama. Form ini secara cerdas mendeteksi apakah data merupakan entitas baru atau hasil pembaruan berdasarkan parameter productId.



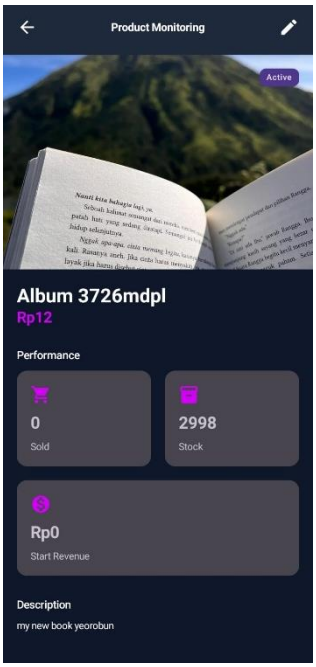
Gambar 4. 22 Antarmuka Formulir Input Data Produk.

8. **Monitoring Pesanan Masuk (Screen.ManageOrders)** Halaman krusial untuk memantau transaksi dari penggemar. Artis dapat melihat rincian barang yang dibeli, alamat pengirim, dan status pembayaran yang perlu diproses.



Gambar 4. 23 Antarmuka Manajemen Pesanan Pelanggan.

9. **Monitoring Stok Produk (Screen.ProductMonitoring)** Fitur tambahan untuk melihat pergerakan stok merchandise secara mendetail, memberikan peringatan jika terdapat produk yang hampir habis agar artis dapat memperbarui ketersediaan barang.

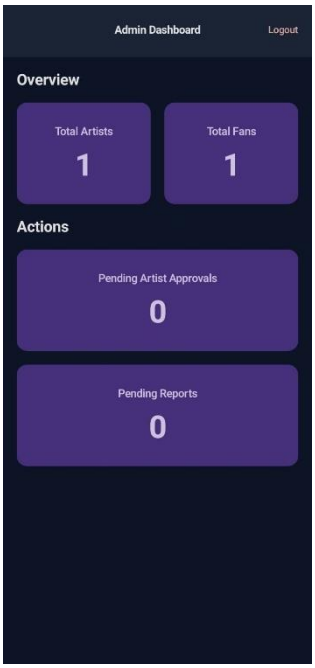


Gambar 4. 24 Antarmuka Pemantauan Ketersediaan Stok Produk.

4.1.4 Antarmuka dan Fungsionalitas Peran "Administrator" (Admin)

Administrator memiliki otoritas tertinggi untuk melakukan moderasi, manajemen data pengguna, dan pengawasan terhadap seluruh ekosistem aplikasi Fandom Hub. Berikut adalah rincian implementasi halaman untuk peran Admin:

1. **Manajemen Daftar Pengguna (Screen.AdminUserManagement)** Halaman ini menyajikan daftar seluruh akun yang terdaftar dalam sistem. Admin dapat memantau aktivitas pengguna dan memisahkan tampilan antara daftar **Fan** dan **Artist**. Fitur ini terhubung langsung dengan UserDao untuk menarik data profil secara keseluruhan.



Gambar 4. 25 Antarmuka Manajemen Seluruh Pengguna.

2. **Verifikasi dan Persetujuan Artis (Screen.AdminArtistApproval)** Sebuah halaman khusus di mana Admin meninjau permintaan pendaftaran dari pengguna dengan peran Artis. Admin memiliki wewenang untuk memberikan status **"Approved"** agar Artis tersebut dapat mulai memposting konten dan merchandise.

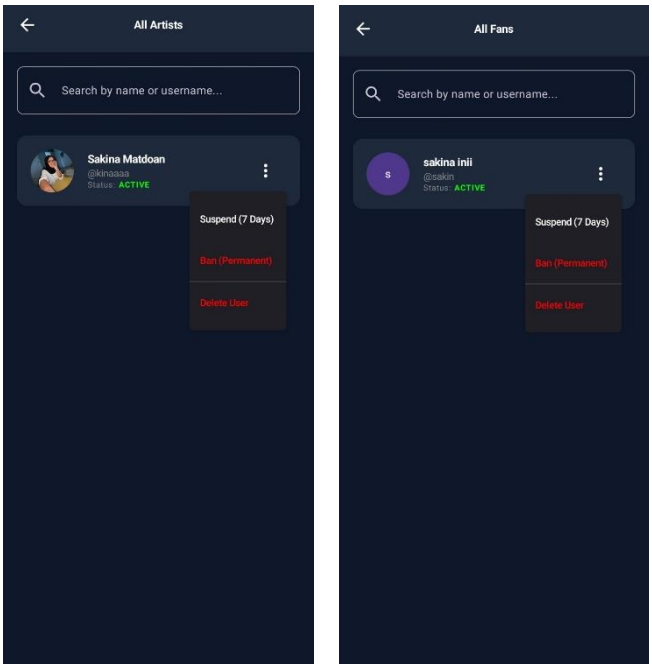
Gambar 4. 26 Antarmuka Validasi dan Persetujuan Akun Artis.

3. **Moderasi Laporan Pelanggaran (Screen.AdminReports)** Fungsi utama untuk menjaga keamanan komunitas. Admin dapat meninjau laporan (*reports*) yang dikirimkan oleh pengguna mengenai postingan atau perilaku yang melanggar ketentuan platform.

Gambar 4. 27 Antarmuka Pusat Moderasi Laporan Pelanggaran.

5. **Eksekusi Penonaktifan Akun (Ban User)** Halaman detail pengguna (baik Fan maupun Artist) pada sisi Admin dilengkapi dengan tombol tindakan tegas. Admin dapat mengubah status akun menjadi **"Banned"** secara permanen di

database lokal, sehingga pengguna tersebut kehilangan hak akses login ke aplikasi.



Gambar 4. 28 Antarmuka Eksekusi Moderasi (Ban User).

4.2 Analisis dan Pembahasan Hasil

Berdasarkan hasil implementasi dan pengujian yang telah dipaparkan pada sub-bab sebelumnya, berikut adalah analisis mendalam mengenai performa dan integrasi sistem Fandom Hub:

4.2.1 Analisis Integrasi Multi-Role (Fan, Artist, Admin)

Sistem telah berhasil mengimplementasikan *Role-Based Access Control* (RBAC) yang solid. Berdasarkan pengujian, setiap peran memiliki batasan akses yang jelas:

1. **Aktor Fan** hanya dapat melakukan fungsi konsumsi dan transaksi (membeli).
2. **Aktor Artist** memiliki otoritas manajemen konten dan produk.
3. **Aktor Admin** memegang kendali moderasi penuh. Keberhasilan ini dibuktikan dengan tidak adanya tumpang tindih fungsi, di mana antarmuka pengguna (UI) secara dinamis berubah sesuai dengan identitas peran yang tersimpan dalam database.

Table 4. 1 Matriks Interaksi dan Validasi Hak Akses

Fitur / Fungsi	Fan	Artist	Admin	Status Validasi
Registrasi & Login	✔	✔	✔	**Berhasil**
Persetujuan Akun (Approval)	✘	✘	✔	**Berhasil**
Posting Tweet & Merchandise	✘	✔	✘	**Berhasil**
Interaksi Balas Tweet	✔	✔	✘	**Berhasil**
Transaksi Pembelian Merch	✔	✘	✘	**Berhasil**
Moderasi & Ban User	✘	✘	✔	**Berhasil**
Review Percakapan (Audit)	✘	✘	✔	**Berhasil**

4.2.2 Analisis Alur Kerja Ekonomi dan Sosial

Pengujian menunjukkan adanya keterkaitan data yang sinkron antar pengguna. Sebagai contoh, ketika seorang **Artist** mengunggah merchandise melalui Screen.ProductForm, data tersebut secara instan tersedia di Screen.Marketplace milik **Fan**. Alur pembelian dari keranjang hingga *checkout* membuktikan bahwa logika pengurangan stok dan pencatatan riwayat pesanan (*Order History*) telah bekerja sesuai dengan prinsip integritas data.

4.2.3 Pembahasan Efisiensi Arsitektur (UI-to-Repository)

Meskipun aplikasi ini tidak menggunakan lapisan ViewModel konvensional, penggunaan **Repository Pattern** yang dikombinasikan dengan **Jetpack Compose State** terbukti sangat efisien untuk skala aplikasi ini.

1. **Reaktivitas:** Penggunaan collectAsState() memungkinkan UI merespon perubahan database Room secara *real-time*. Hal ini terlihat saat proses *Follow* artis, di mana jumlah pengikut langsung berubah tanpa perlu memuat ulang halaman.
2. **Keamanan Navigasi:** Implementasi Sealed Class pada Screen.kt memberikan perlindungan terhadap *runtime error*. Struktur navigasi yang terpusat memudahkan Admin untuk melakukan audit rute pada seluruh bagian aplikasi (Fan, Artist, Admin).

4.2.4 Analisis Fitur Moderasi dan Keamanan

Fitur **Ban User** dan **Artist Approval** oleh Admin merupakan instrumen keamanan yang krusial. Analisis menunjukkan bahwa perubahan status isBanned di database secara efektif memutus akses pengguna. Hal ini membuktikan bahwa validasi sesi pada Screen.Splash telah diimplementasikan dengan benar sebagai filter keamanan utama.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil implementasi, pengujian, dan analisis yang telah dilakukan pada pengembangan aplikasi Fandom Hub, dapat ditarik beberapa kesimpulan utama sebagai berikut:

1. **Keberhasilan Implementasi Arsitektur Multi-Peran**: Aplikasi Fandom Hub telah berhasil dikembangkan dengan sistem manajemen hak akses (*Role-Based Access Control*) yang solid. Sistem ini secara efektif memisahkan fungsionalitas dan antarmuka untuk tiga peran pengguna berbeda: **Fan, Artist, dan Admin**. Validasi hak akses pada level navigasi memastikan setiap peran hanya dapat mengakses fitur yang menjadi wewenangnya, sehingga integritas data dan keamanan aplikasi terjaga.
2. **Fungsionalitas Fitur Utama Telah Tercapai**: Seluruh fitur inti yang dirancang—mulai dari proses otentikasi (registrasi dan login), interaksi sosial (postingan tweet dan komentar), ekosistem *marketplace* (manajemen produk hingga alur *checkout* pesanan), hingga fitur moderasi khusus oleh Admin—telah berhasil diimplementasikan dan berfungsi secara sinkron sesuai dengan tujuan awal pengembangan.
3. **Efektivitas Arsitektur Reaktif Pilihan**: Penggunaan arsitektur yang menghubungkan antarmuka (UI) Jetpack Compose secara langsung ke *Repository*—dengan memanfaatkan collectAsState dari Kotlin Flow—terbukti sangat efektif dan efisien untuk aplikasi berbasis database lokal. Pendekatan ini berhasil menciptakan sinkronisasi data yang cepat antara database Room dan tampilan pengguna tanpa memerlukan lapisan tambahan yang kompleks, sehingga meningkatkan responsivitas aplikasi.
4. **Keamanan dan Struktur Navigasi Terjamin**: Implementasi *Sealed Class* untuk mendefinisikan rute navigasi terbukti berhasil menciptakan sistem yang *type-safe*. Hal ini tidak hanya meminimalisir risiko *runtime error* akibat kesalahan pengetikan rute, tetapi juga membuat keseluruhan alur navigasi aplikasi (yang mencakup lebih dari 25 rute halaman) menjadi sangat terstruktur dan mudah dipetakan untuk pemeliharaan jangka panjang.

5.2 Saran

Meskipun aplikasi Fandom Hub telah memenuhi semua target fungsional yang ditetapkan, terdapat beberapa saran strategis untuk pengembangan di masa mendatang guna meningkatkan skalabilitas dan kualitas pengalaman pengguna:

1. **Migrasi ke Arsitektur Berbasis Backend**: Langkah paling krusial adalah mentransformasi penyimpanan aplikasi dari yang sepenuhnya berbasis lokal (*Room*) menjadi terhubung ke server *backend* (misalnya melalui REST API

atau Firebase). Hal ini diperlukan agar data dapat diakses secara lintas perangkat dan memungkinkan fitur interaksi sosial secara *real-time* yang sesungguhnya.

2. **Integrasi Payment Gateway**: Untuk meningkatkan aspek keamanan dan profesionalisme pada fitur *marketplace*, disarankan untuk mengintegrasikan gerbang pembayaran (*Payment Gateway*) resmi seperti Midtrans atau Stripe. Integrasi ini akan otomatis memverifikasi transaksi pembayaran dan memudahkan manajemen keuangan bagi para Artist.
3. **Optimalisasi Pengalaman Pengguna (UI/UX)**: Melakukan iterasi desain pada antarmuka, seperti implementasi *Dark Mode* yang adaptif, penambahan animasi transisi antar-halaman yang lebih halus, serta peningkatan fitur aksesibilitas bagi pengguna dengan kebutuhan khusus.
4. **Implementasi Push Notification**: Mengembangkan sistem pemberitahuan dorong (*Push Notification*) agar pengguna mendapatkan informasi instan saat terdapat postingan baru dari artis idola, status pesanan yang berubah, atau balasan pesan baru, meskipun aplikasi sedang tidak aktif di latar depan.
5. **Peningkatan Lapisan Keamanan**: Menambahkan fitur keamanan standar industri seperti enkripsi *end-to-end* pada fitur pesan pribadi (*chat*) dan sistem autentikasi dua faktor (2FA) saat login untuk memberikan perlindungan ekstra terhadap data sensitif dan akun pengguna.

DAFTAR PUSTAKA

[1] S. Zubair, “Contemporary Journal Of Social Science Review Vol.03 No.01 (2025),” vol. 03, no. 01, pp. 602–611, 2025.

[2] A. A. P. Alimuddin, *Pemrograman Mobile*. Purbalingga: Eureka Media Aksara, 2025.

[3] A. Tewari and P. Singh, “Android App Development : A Review,” vol. 01, no. 006, pp. 1–6, 2021.

[4] A. Kojo, “Developing a mobile software with Android Studio,” 2020, *Hämeenlinna, Finland*.

[5] N. Smyth, *Android Studio Jellyfish Essentials - Java Edition*. Payload Media, Inc., 2024.

[6] B. A. Wijaya *et al.*, *Pemrograman Mobile dengan Flutter*. Medan: UNPRI Press, 2023.

[7] S. Sibuea, “Aplikasi Mobile Collection Berbasis Android Pada Pt . Suzuki Finance Indonesia,” vol. 2, no. 1, pp. 31–42, 2022.

[8] W. A. Saputra, *Pemrograman Berbasis Objek: Pemrograman Mobile dengan Android Studio*. 2020.

[9] I. Maulana, V. Yasin, and A. B. Yulianto, “Android-Based Stock Opname Application Development with SQLite and Firebase,” vol. 11, no. 2, pp. 734–743, 2025.

[10] N. Selviani, S. Teknologi, P. Studi, S. Informasi, and U. Labuhanbatu, “Pengembangan Aplikasi Mobile Android Berbasis SQLite untuk Sistem Informasi Data Kerjasama Antar Lembaga,” vol. 3, no. 2, pp. 39–44, 2025.

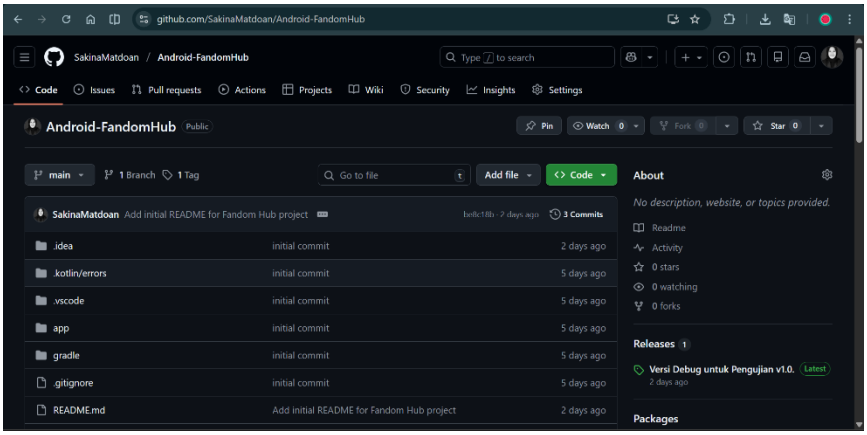
LAMPIRAN

A. Repositori Kode Program (GitHub)

Lampiran ini menyajikan bukti penyimpanan kode sumber (*source code*) aplikasi Fandom Hub yang telah diunggah ke platform GitHub sebagai bagian dari manajemen versi dan dokumentasi proyek.

1. Screenshot Halaman Repositori GitHub

Berikut adalah tampilan halaman utama repositori yang memuat struktur folder aplikasi, file konfigurasi, serta riwayat *commit* sebagai bukti pengerjaan proyek.



Halaman Utama Repositori GitHub Fandom Hub.

2. URL Repositori

Seluruh kode program, aset, dan konfigurasi database Room dalam aplikasi ini dapat diakses secara publik melalui tautan berikut:

URL: https://github.com/SakinaMatdoan/Android-FandomHub.git