BAB II
STUDI LITERATUR

2.1. APA ITU ANDROID

a. Definisi

Android adalah perangkat lunak untuk perangkat mobile yang mencakup sistem operasi, middleware, dan aplikasi inti. Pengembangan aplikasi pada platform ini dilakukan menggunakan bahasa pemrograman Java. Android menyediakan serangkaian aplikasi inti, seperti klien email, program SMS, kalender, peta, browser, kontak, dan lainnya. Dengan platform pengembangan yang terbuka, Android memungkinkan pengembang untuk membangun aplikasi yang kaya fitur dan inovatif. Pengembang dapat memanfaatkan berbagai fitur, seperti akses ke perangkat keras, informasi lokasi, layanan latar belakang (background services), pengaturan alarm, serta menambahkan notifikasi ke status bar.

Android mengandalkan kernel Linux versi 2.6 untuk layanan sistem inti, termasuk keamanan, manajemen memori, manajemen proses, network stack, dan model driver. Kernel ini juga berfungsi sebagai lapisan abstraksi antara perangkat keras dan seluruh lapisan perangkat lunak. .

b. Sejarah Android

Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat bergerak seperti ponsel pintar dan tablet. Platform ini menyediakan lingkungan terbuka bagi pengembang untuk menciptakan aplikasi yang dapat digunakan pada berbagai perangkat. Awalnya, Android dikembangkan oleh Android Inc., yang kemudian diakuisisi oleh Google pada tahun 2005. Untuk mempercepat pengembangan, Google membentuk Open Handset Alliance (OHA) pada tahun 2007, sebuah konsorsium yang terdiri dari berbagai perusahaan teknologi terkemuka.

Pada 5 November 2007, Android bersama OHA secara resmi mengumumkan dukungannya terhadap pengembangan standar terbuka untuk perangkat seluler. Google merilis kode sumber Android di bawah lisensi Apache,

memungkinkan pengembang untuk memodifikasi dan mendistribusikan Android secara bebas. Saat ini, terdapat dua jenis distribusi Android: yang mendapatkan dukungan penuh dari Google, dikenal sebagai Google Mobile Services (GMS), dan yang didistribusikan secara bebas tanpa dukungan langsung dari Google, dikenal sebagai Open Handset Distribution (OHD).

Perangkat pertama yang menggunakan Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Sejak itu, Android terus berkembang dengan merilis berbagai versi baru yang memperkenalkan fitur-fitur inovatif. Pada September 2024, Android 15 dirilis, menandakan komitmen berkelanjutan Google dalam mengembangkan sistem operasi ini.

Dengan pertumbuhan ekosistem aplikasi yang pesat dan dukungan dari komunitas pengembang yang luas, Android telah menjadi salah satu sistem operasi paling dominan di dunia, digunakan oleh berbagai perangkat dari berbagai produsen.

c. Arsitektur Sistem Operasi Android

Arsitektur Sistem Operasi Android terdiri dari beberapa lapisan yang bekerja sama untuk menyediakan platform yang stabil dan efisien bagi aplikasi mobile. Berikut adalah gambaran umum arsitektur Android:

1. Kernel Linux

Di dasar arsitektur Android, terdapat kernel Linux yang bertanggung jawab untuk mengelola perangkat keras seperti CPU, memori, dan perangkat I/O. Kernel ini juga menangani aspek penting lainnya seperti keamanan, manajemen memori, dan pengelolaan proses. Selain itu, kernel juga bertindak sebagai lapisan abstraksi antara perangkat keras dan perangkat lunak yang lebih tinggi.

2. Libraries (Perpustakaan)

Di atas kernel, terdapat berbagai pustaka perangkat lunak yang menyediakan fungsi dasar bagi aplikasi Android. Beberapa pustaka utama termasuk:

- WebKit: Mesin rendering untuk browser web.
- SQLite: Database ringan untuk penyimpanan data aplikasi.
- OpenGL ES: API untuk rendering grafik 2D dan 3D.
- Libc: Library C standar untuk operasi sistem dasar.

3. Android Runtime (ART) dan Dalvik Virtual Machine (DVM)

Android sebelumnya menggunakan Dalvik Virtual Machine (DVM) untuk menjalankan aplikasi, namun sejak Android 5.0 (Lollipop), Android beralih ke Android Runtime (ART). ART meningkatkan performa dengan menjalankan aplikasi dalam mode kompilasi just-in-time (JIT) dan ahead-of-time (AOT), memungkinkan eksekusi aplikasi yang lebih cepat dan efisien.

4. Application Framework

Lapisan ini menyediakan berbagai API yang digunakan oleh aplikasi untuk mengakses fitur sistem. Beberapa komponen penting dari lapisan ini termasuk:

- Activity Manager: Mengelola siklus hidup aplikasi dan aktivitas.
- Window Manager: Mengelola tampilan dan interaksi pengguna dengan aplikasi.
- Content Providers: Mengelola akses ke data aplikasi lain.
- Resource Manager: Menyediakan akses ke berbagai sumber daya seperti gambar, string, dan layout.

5. Aplikasi

Di lapisan atas arsitektur, terdapat aplikasi Android yang dikembangkan oleh pengembang. Aplikasi ini berinteraksi dengan lapisan-lapisan di bawahnya melalui API yang disediakan oleh Application Framework. Android menyediakan aplikasi inti seperti telepon, pesan teks, browser web, dan email, serta memungkinkan pengembang untuk membangun aplikasi mereka sendiri menggunakan bahasa pemrograman seperti Java atau Kotlin.

2.2. PENGENALAN DART

a. Apa itu dart

Dart adalah bahasa pemrograman yang bersifat open-source dan general-purpose, yang dikembangkan oleh Google dengan tujuan untuk membangun aplikasi multiplatform yang dapat dijalankan di berbagai platform, seperti mobile, desktop, web, dan server. Dart pertama kali diperkenalkan pada tahun 2011, dan sejak itu terus berkembang untuk mendukung berbagai kebutuhan pengembangan perangkat lunak yang efisien dan modern.

Salah satu kegunaan utama Dart adalah untuk pengembangan aplikasi menggunakan Flutter, sebuah framework antarmuka pengguna (UI) open-source yang juga dikembangkan oleh Google. Flutter memungkinkan pengembang untuk membuat aplikasi mobile yang memiliki tampilan dan performa seperti aplikasi native, baik untuk Android maupun iOS. Dengan menggunakan Dart, Flutter memungkinkan pengembang untuk menulis kode yang bisa dijalankan di kedua platform tersebut dengan basis kode yang sama, yang sangat menghemat waktu dan usaha dalam pengembangan.

Dart menawarkan sejumlah fitur yang membuatnya cocok untuk pengembangan aplikasi modern. Salah satunya adalah kemampuannya untuk melakukan Just-In-Time (JIT) Compilation saat pengembangan, yang memungkinkan pengembang untuk melihat perubahan kode secara langsung tanpa harus melakukan kompilasi ulang seluruh aplikasi. Selain itu, Dart juga mendukung Ahead-Of-Time (AOT) Compilation untuk menghasilkan aplikasi yang lebih cepat dan lebih efisien saat dipublikasikan.

Selain digunakan dalam Flutter, Dart juga dapat digunakan untuk mengembangkan aplikasi web menggunakan Dart2js, yang memungkinkan kode Dart dikompilasi menjadi JavaScript, memungkinkan aplikasi Dart berjalan di semua browser modern. Untuk aplikasi server-side, Dart menyediakan dart:io, sebuah pustaka yang mendukung pengembangan server dan aplikasi berbasis backend dengan kemampuan jaringan, I/O, dan file system.

Dart memiliki sintaksis yang mirip dengan bahasa pemrograman lainnya seperti Java, C++, dan JavaScript, sehingga memudahkan pengembang yang sudah berpengalaman dengan bahasa tersebut untuk memulai dengan Dart. Beberapa fitur unggulan Dart termasuk null safety, yang membantu pengembang untuk menulis kode yang lebih aman dengan meminimalkan kemungkinan kesalahan terkait referensi null, serta kemampuan untuk membuat aplikasi asynchronous yang efisien, menggunakan fitur seperti Future dan Stream.

Secara keseluruhan, Dart adalah bahasa yang sangat fleksibel dan efisien untuk pengembangan aplikasi modern di berbagai platform. Dengan dukungan kuat dari Google dan ekosistem yang berkembang pesat, Dart dan Flutter semakin menjadi pilihan utama bagi pengembang yang ingin membangun aplikasi multiplatform dengan performa tinggi dan tampilan menarik.

b. Kenapa perlu belajar dart

Terdapat sejumlah alasan yang mendasari pentingnya mempelajari bahasa pemrograman Dart, terutama bagi individu yang tertarik untuk menjadi pengembang aplikasi Flutter atau yang ingin mendalami lebih lanjut tentang bahasa ini. Berikut adalah beberapa alasan mengapa mempelajari Dart sangat bermanfaat:

- Pengembangan Multiplatform: Dart dirancang untuk mendukung pengembangan aplikasi multiplatform, termasuk aplikasi untuk perangkat mobile, desktop, web, dan server. Dengan menggunakan Dart, pengembang dapat menulis kode yang dapat dijalankan di berbagai platform dengan basis kode yang sama, yang meningkatkan efisiensi pengembangan aplikasi.

- Proyek Open Source: Dart adalah proyek open source, yang berarti pengembang dapat berkontribusi dalam pengembangan bahasa ini atau memanfaatkan sumber daya yang dibagikan oleh komunitas. Terlibat dalam proyek open source memberikan peluang untuk meningkatkan keterampilan pengembangan serta memperluas jaringan profesional.

- Performa yang Optimal: Dart menggunakan Dart VM (Virtual Machine) sebagai mesin eksekusi untuk menjalankan kode. Mesin ini dirancang untuk memberikan performa yang optimal, terutama dalam pengembangan aplikasi menggunakan Flutter, sebuah framework UI yang dikembangkan oleh Google.

- Bahasa yang Mudah Dipelajari: Dart memiliki sintaksis yang sederhana dan mudah dipahami, terutama bagi mereka yang sudah berpengalaman dalam menggunakan bahasa pemrograman lain. Struktur sintaksnya mirip dengan bahasa pemrograman modern seperti JavaScript, Java, dan C++, yang memudahkan transisi bagi pengembang dari berbagai latar belakang.

- Dukungan Alat dan Dokumentasi: Dart didukung oleh berbagai alat pengembangan populer seperti Visual Studio Code, IntelliJ IDEA, dan Android Studio. Selain itu, Dart memiliki dokumentasi yang komprehensif dan komunitas yang aktif, sehingga memudahkan pengembang untuk memperoleh bantuan serta sumber belajar yang berkualitas.

- Dart Native dan Flutter Web: Selain digunakan dalam pengembangan aplikasi mobile menggunakan Flutter, Dart juga dapat digunakan untuk mengembangkan aplikasi Native melalui Dart Native, serta aplikasi web menggunakan Flutter Web. Hal ini menjadikan Dart sebagai pilihan yang fleksibel dalam pengembangan aplikasi multiplatform.

- Null Safety: Dart mendukung fitur null safety, yang secara signifikan mengurangi risiko kesalahan terkait dengan pointer null dalam kode. Fitur ini membantu meningkatkan keamanan dan stabilitas aplikasi dengan memastikan bahwa referensi objek tidak bernilai null tanpa penanganan yang tepat.

2.3. PENGENALAN FLUTTER

Flutter adalah sebuah framework open-source yang dikembangkan oleh Google untuk membangun aplikasi nativ e dengan tampilan dan performa yang tinggi, yang dapat dijalankan di berbagai platform, termasuk Android, iOS, Web, dan Desktop. Dengan menggunakan Flutter, pengembang dapat menulis satu basis kode yang dapat dijalankan di beberapa platform, yang sangat mengurangi waktu dan usaha yang diperlukan dalam pengembangan aplikasi untuk berbagai perangkat.

Flutter menggunakan Dart sebagai bahasa pemrograman utama. Salah satu keunggulan utama Flutter adalah kemampuannya untuk memberikan UI (User Interface) yang sangat responsif dan kaya, dengan kontrol penuh terhadap setiap piksel pada layar. Flutter menyediakan berbagai widget bawaan yang memungkinkan pengembang untuk membuat antarmuka pengguna yang menarik dan fungsional, yang dapat disesuaikan dengan berbagai gaya dan tema desain. Selain itu, Flutter menawarkan beberapa fitur penting yang membuatnya menonjol:

a. Hot Reload: Salah satu fitur utama Flutter adalah Hot Reload, yang memungkinkan pengembang untuk melihat perubahan kode secara langsung tanpa perlu memulai ulang aplikasi. Fitur ini sangat membantu dalam mempercepat proses pengembangan dan pengujian.

b. Performa Tinggi: Flutter mengkompilasi kode ke dalam bentuk kode mesin yang dijalankan secara langsung oleh perangkat, memberikan performa yang hampir setara dengan aplikasi native. Hal ini memungkinkan aplikasi Flutter berjalan dengan lancar dan cepat, bahkan untuk aplikasi yang kompleks.

c. Customizable Widgets: Flutter menyediakan serangkaian widget yang dapat disesuaikan sepenuhnya. Pengembang memiliki fleksibilitas penuh untuk mendesain antarmuka pengguna sesuai dengan kebutuhan aplikasi mereka, tanpa terbatas oleh tampilan standar sistem operasi.

d. Komunitas yang Aktif: Sebagai framework open-source, Flutter didukung oleh komunitas yang aktif dan terus berkembang. Pengembang dapat memanfaatkan berbagai plugin dan pustaka yang tersedia untuk memperluas fungsionalitas aplikasi mereka.

e. Aplikasi Multiplatform: Dengan Flutter, pengembang dapat membangun aplikasi yang dapat dijalankan di beberapa platform dari satu basis kode. Hal ini sangat menghemat waktu dan usaha dalam pengembangan aplikasi untuk Android, iOS, Web, dan Desktop.

2.4. INTEGRASI TEKNOLOGI

Flutter menyediakan berbagai cara untuk mengintegrasikan teknologi dan fitur perangkat keras, seperti GPS, kamera, sensor, dan penggunaan API eksternal. Berikut adalah beberapa contoh integrasi teknologi yang dapat dilakukan dalam aplikasi Flutter:

a. Integrasi GPS (Geolocation)

Flutter memungkinkan pengembang untuk mengakses lokasi perangkat secara real-time. Dengan menggunakan teknologi geolokasi, aplikasi dapat memperoleh koordinat geografis perangkat, seperti latitude dan longitude, yang berguna untuk aplikasi yang memerlukan informasi lokasi pengguna, seperti aplikasi peta atau layanan berbasis lokasi.

b. Penggunaan Kamera

Aplikasi Flutter dapat mengakses kamera perangkat untuk mengambil foto atau merekam video. Pengembang dapat mengontrol berbagai fitur kamera, seperti pengambilan gambar dalam mode lanskap atau potret, serta mengatur kualitas dan resolusi gambar. Ini berguna untuk aplikasi yang memerlukan pemindaian kode QR, pengambilan gambar profil, atau fitur video call.

c. Integrasi API Eksternal (Web API)

Aplikasi Flutter dapat berkomunikasi dengan server eksternal untuk mengambil data atau mengirim informasi menggunakan permintaan HTTP. Ini memungkinkan aplikasi untuk terhubung dengan berbagai layanan web, seperti pengambilan data dari database, integrasi dengan sistem pembayaran, atau mengambil konten dinamis dari server.

d. Sensor Perangkat (Accelerometer, Gyroscope, etc.)

Flutter dapat mengakses berbagai sensor yang ada pada perangkat, seperti akselerometer untuk mendeteksi gerakan atau giroskop untuk mengukur orientasi perangkat. Fitur ini berguna dalam aplikasi yang memerlukan deteksi pergerakan, seperti aplikasi fitness atau game yang memanfaatkan gerakan perangkat.

e. Integrasi Layanan Pihak Ketiga

Aplikasi Flutter dapat dengan mudah diintegrasikan dengan layanan pihak ketiga, seperti Google Maps untuk menampilkan peta interaktif atau layanan pembayaran untuk memfasilitasi transaksi online. Teknologi ini memungkinkan aplikasi untuk memperluas fungsionalitasnya dengan layanan yang sudah ada.