BAB 2 LANDASAN TEORI

2.1 Android

Android merupakan sistem operasi *mobile* yang dikembangkan oleh Google dan dibangun di atas *kernel* Linux. Sistem ini menyediakan kerangka kerja (*framework*) yang kaya dengan API sehingga pengembang dapat membangun aplikasi lintas perangkat dengan mudah. Selain itu, distribusi aplikasi dilakukan melalui Google Play Store, yang menjadikan Android salah satu ekosistem terbesar dalam mendukung aplikasi produktivitas maupun hiburan. [2]

2.2 Firebase

Firebase adalah layanan berbasis *cloud* yang berfungsi sebagai *Backend-as-a-Service* (BaaS) [3]. Platform ini menyediakan berbagai komponen penting untuk mendukung pengembangan aplikasi *mobile*, di antaranya:

1. *Realtime Database*: basis data NoSQL yang memungkinkan sinkronisasi data secara langsung antar pengguna.
2. *Authentication*: sistem autentikasi yang mendukung *login* menggunakan email, nomor telepon, maupun layanan pihak ketiga.
3. *Cloud Messaging* (FCM): layanan notifikasi *push* yang memudahkan pengiriman pengingat atau pesan ke perangkat pengguna.

Dengan memanfaatkan Firebase, pengembang tidak perlu mengelola server secara manual, sehingga proses pengembangan menjadi lebih efisien dan skalabel.

2.3 Aplikasi Reminder dan To-do List

Aplikasi pengingat membantu pengguna mendapatkan notifikasi pada waktu-waktu tertentu sehingga mereka tidak melewatkan aktivitas penting. Di sisi lain, daftar tugas adalah kumpulan tugas yang dapat ditandai sebagai selesai atau

belum selesai. Bersama-sama, kedua fitur ini membantu pengguna mengatur aktivitas harian mereka dengan cara yang lebih terstruktur, sehingga meningkatkan produktivitas. [10, 11, 12]

Penelitian terlebih dahulu terkait dengan aplikasi *reminder* telah banyak dilakukan diantaranya adalah sebagai berikut:

1. Penelitian A. D. Irawan dan W. S. Utami, tahun 2023 dengan judul Aplikasi *reminder* jadwal kuliah dan tugas mahasiswa berbasis Android [10]
   Kelebihan utama:
   1. Menggunakan *Firebase Realtime Database* sehingga data tersinkronisasi secara *real-time*.
   2. Memberikan notifikasi otomatis sebelum jadwal kuliah dimulai atau sebelum deadline tugas.
   3. Dibangun dengan Android Studio + Kotlin, memanfaatkan ekosistem modern Android.

   Nilai tambah: Fokus pada kebutuhan mahasiswa yang sering lupa jadwal/tugas, sehingga meningkatkan disiplin akademik.

2. Penelitian G. P. Edde dan K. Budayawan, tahun 2021 dengan judul Pembuatan aplikasi *reminder* jadwal perkuliahan di Jurusan Teknik Elektronika berbasis Android [11]
   Kelebihan utama:
   1. Menyediakan informasi penggantian jadwal kuliah secara real-time, yang tidak tersedia di portal akademik tradisional.
   2. Memanfaatkan *Firebase Realtime Database* (FRD) untuk pencatatan perubahan jadwal.
   3. Menggunakan UML untuk pemodelan sistem, sehingga desain aplikasi lebih terstruktur.

Nilai tambah: Sangat membantu dosen dan mahasiswa dalam mengatur jadwal pengganti, diuji dengan black box testing dan dinyatakan layak digunakan.

3. Penelitian V. Ilhadi, D. Ardiansyah, dan Muthmainnah, tahun 2022 dengan judul Aplikasi *reminder* jadwal kegiatan berbasis *mobile* [12]
   Kelebihan utama:
   1. Memiliki fitur *Pop-Up Reminder* yang otomatis membuka perangkat untuk memberi pengingat.
   2. Menggunakan *SQLite* lokal sehingga aplikasi tetap berfungsi tanpa koneksi internet.
   3. Menyediakan progres mingguan yang membuat aplikasi lebih menarik dan memotivasi pengguna.

   Nilai tambah: Tidak hanya untuk kegiatan kampus, tetapi juga bisa digunakan untuk pekerjaan rumah, kantor, atau aktivitas sehari-hari.

2.4 User Interface (UI) dan User Experience (UX)

Di dunia digital yang berkembang pesat saat ini, desain *User Experience* (UX) dan *User Interface* (UI) merupakan dua elemen kunci yang saling bersinergi dalam pengembangan produk digital. Meskipun sering kali diterapkan secara bersamaan, keduanya memiliki peran dan tanggung jawab masing-masing dalam menciptakan pengalaman pengguna yang utuh. [14]

Perbedaan UI dan UX:

1. *User Interface* (UI)**:** aspek visual aplikasi, meliputi tata letak, warna, ikon, dan navigasi. UI yang baik harus konsisten dan mudah dipahami.
2. *User Experience* (UX)**:** pengalaman keseluruhan pengguna saat berinteraksi dengan aplikasi. UX mencakup kemudahan penggunaan, efisiensi, serta kepuasan pengguna.

Kedua aspek ini saling melengkapi. UI yang menarik tanpa UX yang baik akan membuat aplikasi sulit dipakai, sedangkan UX yang bagus tanpa UI yang jelas dapat mengurangi kenyamanan visual.

.

2.5 Evaluasi UI/UX dengan System Usability Scale (SUS)

System Usability Scale (SUS) [2] adalah metode evaluasi kegunaan yang menggunakan kuesioner berisi 10 pertanyaan standar [9] yang merupakan turunan dari *Eight Golden Rules* [8]. Adapun 10 pertanyaan standar yang digunakan untuk menguji SUS adalah sebagai berikut:

1. Saya akan sering menggunakan sistem ini.
2. Sistem ini terasa rumit/tidak perlu rumit.
3. Sistem ini mudah digunakan.
4. Saya butuh bantuan teknis untuk menggunakan sistem ini.
5. Fitur sistem ini terintegrasi dengan baik.
6. Ada terlalu banyak inkonsistensi dalam sistem ini.
7. Sebagian besar orang akan cepat belajar menggunakan sistem ini.
8. Sistem ini terasa membingungkan.
9. Saya merasa percaya diri menggunakan sistem ini.
10. Saya perlu belajar banyak sebelum bisa menggunakan sistem ini.

Setiap jawaban diberi skor 1 s/d 5, kemudian dihitung dengan mengubah setiap skor, untuk menghasilkan nilai antara 0 s/d 100. [13]

Rumus 2.1 System Usability Scale

$$\bar{x} = \frac{\sum x}{n}$$

$\bar{x}$ = skor rata-rata

$\sum x$ = jumlah skor SUS

n = jumlah responden

Setelah itu semua skor dirata-ratakan dan diinterpretasikan sebagai berikut:

1. $\geq 70 \rightarrow$ aplikasi dianggap mudah digunakan (*usable*).
2. $< 70 \rightarrow$ aplikasi perlu perbaikan signifikan.

Metode SUS dipilih karena sederhana, cepat, dan telah terbukti valid dalam menilai tingkat kegunaan aplikasi.