BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Bab ini berisi tentang penelitian terdahulu/tinjauan umum, serta menjelaskan mengenai konsep dasar umum yang meliputi pengertian sistem, karakteristik sistem dan peralatan pendukungnya (*tools system*).

Table 2.1 Tabel Tinjauan Pustaka

Penulis	Object	Metode/ Teknologi	Bahasa Pemrograman	Interface
Setiawan Eka (2018)	Apotek dan Klinik di Kabupaten Bantul	Google Maps API	Kotlin	Aplikasi Android
Lou Tian (2016)	Komparasi Arsitektur Android Native	MVC,MVP, MVVM	Java	Aplikasi Android
Reza Abdillah (2017)	Informasi Lowongan Pekerjaan di DIY	Android, Firebase Cloud Messaging	Java	Aplikasi Android
Resta Bayu Setiawan (2017)	UKM Informatika & Komputer (UKM IK)	Android, Firebase Auth	Java	Aplikasi Android
Aghnia Fi'la Urfan (2017)	Kegiatan Seni Kontemporer wilayah DIY	Android, Integrasi Google Map & Google Calendar	Kotlin	Aplikasi Android

Eka Setiawan (2018) pada penelitiannya menekankan pada pemetaan dengan menggunakan Google Maps API. Peta yang ditampilkan diambil dari server

Google Maps. Pada penelitiannya tidak hanya menampilkan peta informasi Apotek dan Klinik yang ada di Kabupaten Bantul, tetapi dapat mengetahui berapa jarak lokasi apotek dari pengguna berada serta dapat mengetahui informasi klinik untuk mendapatkan informasi pelayanan dari klinik. Selain itu terdapat fungsi pencarian terhadap apotek yang terkait.

Tian Lou (2016) pada penelitiannya menekankan pada Komparasi Arsitektur Android Native dengan mengunci pembuatan aplikasi android sederhana menggunakan beberapa *pattern* yaitu MVC,MVP,MVVM, dari hasil penelitiannya didapat bahwa MVVM lebih baik dalam hal pemeliharaan memori dan penambahan fitur baru pada aplikasi.

Reza Abdillah melakukan penelitian dengan topik Implementasi Push Notification pada Aplikasi Lowongan Kerja. Objek dari penelitian tersebut adalah Informasi Lowongan Pekerjaan di Daerah Istimewa Yogyakarta. Dari penelitian tersebut dihasilkan aplikasi android Loker Jogja. Aplikasi tersebut dibangun Menggunakan Android Studio dengan library Parse, Parse NoSQL database, dan

Resta Bayu Setiawan (2017) melakukan penelitian yang berfokus pada objek penelitian & teknologi yang digunakan untuk membuat aplikasi ini. Dimana objek yang digunakan untuk penelitian ini adalah organisasi mahasiswa UKM Informatika & Komputer STMIK AKAKOM YOGYAKARTA.

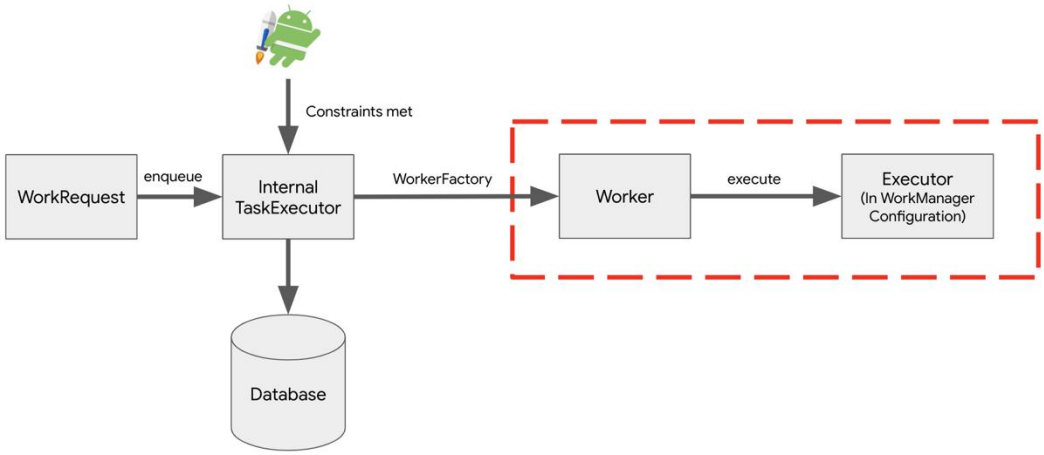
Pada tahun 2017 Aghnia Fi'la Urfan melakukan penelitian dengan topik Aplikasi Kalender Event Seni Kontemporer, dimana objek dari penelitian tersebut adalah kegiatan Seni Kontemporer yang ada di wilayah Daerah Istimewa Yogyakarta. Dari penelitian tersebut dihasilkan aplikasi android Jogja Festivals.

Aplikasi tersebut dibangun menggunakan Android Studio IDE, dimana bahasa pemrograman yang digunakan adalah java, database nya menggunakan MySQL, serta Google Map API sebagai library untuk fitur penunjuk arah. Fitur dari aplikasi ini pengguna dapat melihat daftar acara yang ada, menambahkan pengingat acara ke Google Calendar, serta melakukan tracking penunjuk arah tempat berlangsungnya acara dengan Google Maps.

2.2 Dasar Teori

2.2.1 WorkManager

WorkManager adalah pustaka yang digunakan untuk mengantrekan pekerjaan yang dapat ditangguhkan yang dijamin akan dijalankan beberapa saat setelah pekerjaan Constraints itu terpenuhi. WorkManager memungkinkan observasi status pekerjaan dan kemampuan untuk membuat rantai kerja yang kompleks.



Gambar 2.1 Alur Kerja WorkManager

1. Internal TaskExecutor

Internal TaskExecutor adalah sebuah utas tunggal Executor yang menangani semua permintaan untuk melakukan pekerjaan.

2. Database WorkManager

Database WorkManager merupakan database lokal yang melacak semua informasi dan status semua pekerjaan. Ini termasuk hal-hal seperti keadaan pekerjaan saat ini, input dan output ke dan dari pekerjaan dan segala kendala pada pekerjaan. Basis data ini adalah yang memungkinkan WorkManager untuk menjamin pekerjaan akan selesai - jika perangkat pengguna restart dan pekerjaan terganggu, semua detail pekerjaan dapat ditarik dari database dan pekerjaan itu dapat dimulai kembali ketika perangkat dinyalakan kembali.

3. Constraints

Spesifikasi persyaratan yang harus dipenuhi sebelum WorkRequest dapat dijalankan. Secara default, WorkRequests tidak memiliki persyaratan apapun dan dapat segera dijalankan. Dengan menambahkan persyaratan, Anda dapat memastikan bahwa pekerjaan hanya berjalan dalam situasi tertentu - misalnya, saat Anda memiliki jaringan tanpa meteran dan sedang mengisi daya.

WorkManager menggunakan layanan pengiriman pekerjaan yang mendasari jika tersedia berdasarkan kriteria berikut:

- Menggunakan JobScheduler untuk API 23+

- Menggunakan implementasi AlarmManager + BroadcastReceiver khusus untuk API 14-22

Semua pekerjaan harus dilakukan di ListenableWorker kelas. Penerapan sederhana, Worker direkomendasikan sebagai titik awal bagi sebagian besar pengembang. Dengan dependensi opsional, Anda juga dapat menggunakan CoroutineWorker atau RxWorker. Semua pekerjaan latar belakang diberikan waktu maksimal sepuluh menit untuk menyelesaikan pelaksanaannya. Setelah waktu ini habis, pekerja akan diberi tanda untuk berhenti.

Ada dua jenis pekerjaan yang didukung oleh WorkManager: OneTimeWorkRequest dan PeriodicWorkRequest. OneTimeWorkRequests dapat digabungkan menjadi grafik asiklik. Pekerjaan memenuhi syarat untuk dieksekusi ketika semua prasyaratnya selesai. Jika salah satu prasyaratnya gagal atau dibatalkan, pekerjaan tidak akan pernah berjalan.

WorkRequests dapat menerima Constraints, memasukkan (lihat Data), dan kriteria backoff. WorkRequests dapat diberi tag dengan Strings yang dapat dibaca manusia (lihat addTag(String)), dan rantai kerja dapat diberi nama yang dapat diidentifikasi secara unik dengan kebijakan konflik. *

Secara default, WorkManager diinisialisasi menggunakan ContentProvider dengan default Configuration. ContentProviders dibuat dan dijalankan sebelum Application objek, sehingga hal ini memungkinkan penyiapan tunggal WorkManager sebelum kode Anda dapat dijalankan dalam banyak kasus.

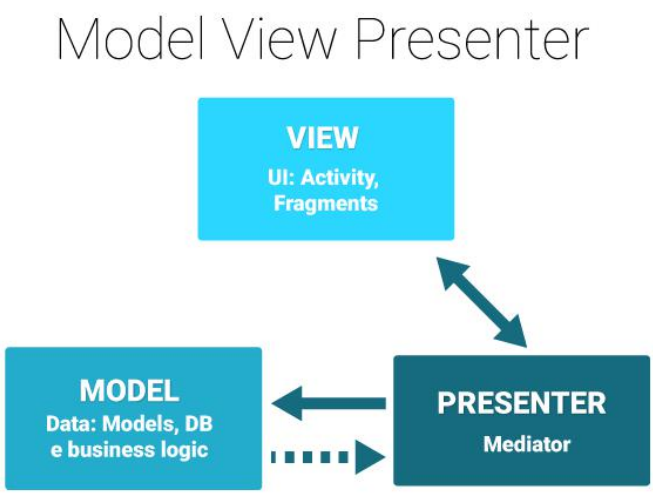
Ini cocok untuk sebagian besar pengembang. Namun, Anda dapat memberikan kebiasaan Configuration dengan menggunakan Configuration.Provider atau initialize(android.content.Context, androidx.work.Configuration).

WorkManager BroadcastReceiverakan memantau Constraints perangkat sebelum API 23. BroadcastReceivers dinonaktifkan pada API 23 dan yang lebih baru. Secara khusus, WorkManager mendengarkan berikut ini Intent:

- android.intent.action.ACTION_POWER_CONNECTED
- android.intent.action.ACTION_POWER_DISCONNECTED
- android.intent.action.BATTERY_OKAY
- android.intent.action.BATTERY_LOW
- android.intent.action.DEVICE_STORAGE_LOW
- android.intent.action.DEVICE_STORAGE_OK
- android.net.conn.CONNECTIVITY_CHANGE (Developer android, 2020).

2.2.2 MVP

Model View Presenter atau yang biasa disingkat menjadi MVP adalah sebuah konsep arsitektur pengembangan aplikasi yang memisahkan antara tampilan aplikasi dengan proses bisnis yang bekerja pada aplikasi. Arsitektur ini akan membuat pengembangan aplikasi menjadi lebih terstruktur, mudah di-*test* dan juga mudah di-*maintain.*



Gambar 2.2 Alur Kerja MVP

1. View

View merupakan layer untuk menampilkan data dan interaksi ke *user*. View biasanya berupa *Activity, Fragment* atau *Dialog* di Android. View ini juga yang langsung berkomunikasi dengan *user*.

2. Model

Model merupakan layer yang menunjuk kepada objek dan data yang ada pada aplikasi.

3. Presenter

Presenter merupakan layer yang menghubungkan komunikasi antara Model dan View. Setiap interaksi yang dilakukan oleh *user* akan memanggil Presenter untuk memprosesnya dan mengakses Model lalu mengembalikan responnya kembali kepada View (Developer android, 2019).

2.2.3 Firebase

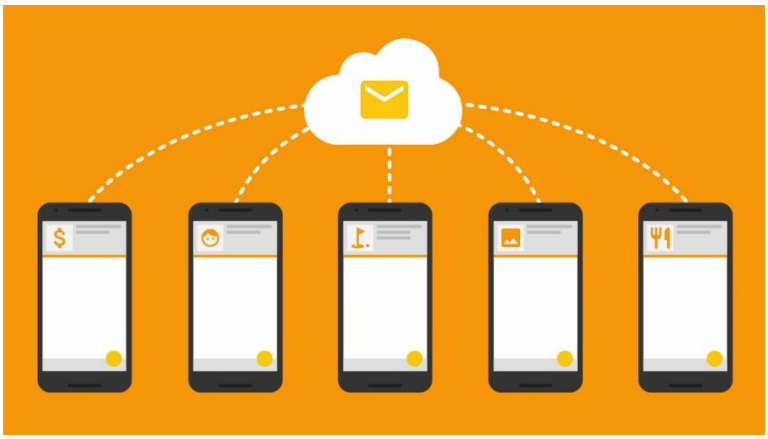
Firebase adalah platform pengembangan aplikasi seluler dan web, yang didukung oleh Google, untuk membantu pengembang menghadirkan pengalaman aplikasi yang lebih kaya. Firebase mengelola infrastrukturnya sendiri dengan seperangkat alat yang bagus untuk menyederhanakan alur kerja pengembang dengan menyediakan kit pengembangan dan dasbor online kepada mereka. Toolkit ini saling berhubungan, dapat diskalakan, dan dapat diintegrasikan dengan perangkat lunak pihak ketiga untuk mengatasi tantangan kompleks dengan blok bangunan standar.

Platform ini terdiri dari seperangkat alat pengembangan yang hebat. Realtime database dan Cloud Firestore dapat menyimpan data terstruktur dokumen dan menyinkronkan aplikasi terkait dalam milidetik setiap kali terjadi transformasi data. Ini berarti bahwa aplikasi dan database-nya mendengarkan satu sama lain, memberikan pengalaman aplikasi yang reaktif kepada pengguna. Dan Firebase Cloud Functions bahkan dapat memperluas fungsionalitas ini. Fungsi ini memungkinkan pengembang untuk menulis kode backend untuk menanggapi peristiwa yang terjadi di platform Firebase tanpa harus berurusan dengan server apa pun.

1. Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) adalah solusi perpesanan lintas platform yang memungkinkan Anda mengirim pesan dengan andal tanpa biaya. Firebase Cloud Messaging (FCM) menawarkan berbagai opsi dan kemampuan

pengiriman pesan. Informasi di halaman ini dimaksudkan untuk membantu Anda memahami berbagai jenis pesan FCM dan apa yang dapat Anda lakukan dengannya.



Gambar 2.3 Alur Kerja Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) memungkinkan Anda mengirim notifikasi push langsung dari Firebase console atau dengan server aplikasi atau beberapa lingkungan terpercaya lainnya tempat logika server dijalankan. Namun, Anda mungkin ingin mengirim pemberitahuan push antar perangkat, seperti di kebanyakan aplikasi obrolan, tetapi tanpa menulis kode sisi server apa pun. Tentu saja sangat mungkin. Tutorial ini menjelaskan dengan tepat bagaimana Anda bisa mencapainya dalam empat langkah mudah.

2. Firebase Realtime Database

Firebase Realtime Database adalah database yang di-host di cloud. Data disimpan sebagai JSON dan disinkronkan secara realtime ke setiap klien yang terhubung. Ketika Anda mem-build aplikasi lintas platform dengan SDK iOS,

Android, dan JavaScript, semua klien akan berbagi sebuah instance Realtime Database dan menerima update data terbaru secara otomatis.

Realtime Database menyediakan bahasa aturan berbasis ekspresi yang fleksibel, atau disebut juga Aturan Keamanan Firebase Realtime Database, untuk menentukan metode strukturisasi data dan kapan data dapat dibaca atau ditulis. Ketika diintegrasikan dengan Firebase Authentication, developer dapat menentukan siapa yang memiliki akses ke data tertentu dan bagaimana mereka dapat mengaksesnya.

Realtime Database adalah database NoSQL, sehingga memiliki pengoptimalan dan fungsionalitas yang berbeda dengan database terkait. API Realtime Database dirancang agar hanya mengizinkan operasi yang dapat dijalankan dengan cepat. Hal ini memungkinkan Anda untuk mem-build pengalaman realtime yang luar biasa dan dapat melayani jutaan pengguna tanpa mengorbankan kemampuan respons. Oleh karena itu, perlu dipikirkan bagaimana pengguna mengakses data, kemudian buat struktur data sesuai dengan kebutuhan tersebut (Developer android, 2020).