

Implementasi Arsitektur MVVM dalam Pengembangan Aplikasi Rekomendasi Makanan Sehat Berbasis Mobile

Achmad Rionov Faddillah Ramadhan
School Of Electrical Engineering
Telkom University
Bandung, Indonesia
rionov@student.telkomuniversity.ac.id

Purba Daru Kusuma
School Of Electrical Engineering
Telkom University
Bandung, Indonesia
purbodaru@telkomuniversity.ac.id

Anggunmeka Luhur Prasasti
School Of Electrical Engineering
Telkom University
Bandung, Indonesia
anggunmeka@telkomuniversity.ac.id

Pola makan yang tidak sehat merupakan masalah kesehatan yang prevalen di Indonesia. Dengan konsumsi makanan yang tinggi kalori namun rendah nutrisi memperburuk kondisi penyakit kronis seperti diabetes, kolesterol dan hipertensi. Dengan banyaknya penyakit kronis yang tinggi akibat pola makan yang tidak sehat, aplikasi mobile yang memberikan rekomendasi makanan sehat bisa menjadi solusi yang strategis. Melalui aplikasi ini, pengguna dapat menjaga kesehatan mereka dengan memilih diet yang lebih sehat dan bertanggung jawab, dan didukung oleh fitur - fitur yang tersedia. Rekomendasi resep, pelacakan nutrisi, dan program diet menjadi sarana untuk mendukung pengguna dalam mencapai tujuan kesehatan mereka secara lebih efektif. Arsitektur Model-View-ViewModel (MVVM) digunakan dalam pengembangan aplikasi ini, yang barengi beserta penerapan *clean architecture* untuk memastikan modularitas dan kejelasan kode. Hasil pengujian menunjukkan bahwa aplikasi ini tidak hanya mendukung kebutuhan pengguna saat ini tetapi juga memudahkan skalabilitas dan pemeliharaan jangka panjang. Aplikasi ini berhasil meningkatkan pengalaman pengguna dengan performa yang stabil dan antarmuka pengguna yang efektif.

Kata kunci— *arsitektur MVVM, clean architecture, rekomendasi makanan sehat, aplikasi mobile, kesehatan masyarakat.*

I. PENDAHULUAN

Pola makan yang tidak sehat menjadi tantangan bagi kesehatan masyarakat Indonesia saat ini. Konsumsi makanan yang tidak sehat, kaya akan kalori namun kurang nutrisi, berkontribusi pada meningkatnya prevalensi penyakit tidak menular dan berpotensi kronis seperti diabetes, hipertensi, dan kolesterol. Hal ini menjadi lebih penting lagi bagi orang-orang yang menderita penyakit jangka panjang, di mana pilihan makanan yang tidak sehat dapat memperburuk kondisi mereka dan memperlambat pemulihan.

Penggunaan teknologi informasi, seperti aplikasi ponsel, menjadi solusi inovatif di tengah krisis kesehatan ini. Aplikasi seluler dapat meningkatkan kualitas hidup masyarakat dengan mendorong preferensi makanan yang lebih sehat. Berdasarkan hal-hal tersebut, membangun sebuah aplikasi untuk memberikan rekomendasi makanan sehat yang dapat disesuaikan dengan kebutuhan kesehatan

setiap orang menjadi sangat penting. Aplikasi ini memungkinkan pengguna untuk memilih resep harian maupun rencana makan mingguan yang dirancang berdasarkan usia, kondisi kesehatan, preferensi makanan, alergi, dan data kesehatan pengguna.

Berdasarkan pengamatan akan aplikasi lain yang tersedia di Play Store, dibuat Aplikasi Rekomendasi Makanan Sehat yang menangani masalah - masalah tersebut. Mulai dari keinginan untuk menurunkan dan menaikkan berat badan, ataupun menjaga pola makanan sehat, aplikasi ini memenuhi kebutuhan tersebut melalui fitur - fitur yang telah dikembangkan. Pelacakan nutrisi digunakan untuk memberikan informasi dan pelacakan konsumsi kalori, lemak, karbohidrat dan protein dan air yang telah dikonsumsi dalam sehari. Selain pelacakan, fitur ini menginformasikan pengguna akan kebutuhan nutrisi harian yang dibutuhkan berdasarkan data kesehatan mereka. Fitur resep digunakan untuk memberikan rekomendasi berdasarkan tinggi dan berat badan, jenis kelamin, usia, alergi, dan penyakit yang diderita pengguna. Fitur program khusus digunakan untuk membantu seluruh pengguna baik itu untuk penurunan, peningkatan, atau pemeliharaan berat badan mereka dengan melacak perkembangan berat badan, penyesuaian kebutuhan kalori, dan pelacakan seluruh nutrisi yang telah dikonsumsi selama periode program. Fitur Profile digunakan untuk personalisasi akun dan rekomendasi makanan dengan mengubah data kesehatan pengguna.

Dengan menerapkan arsitektur Model-View-ViewModel (MVVM), aplikasi ini juga diharapkan dapat memiliki performa yang baik supaya bisa meningkatkan pengalaman pengguna yang baik. Selain penerapan MVVM, perancangan aplikasi ini juga mengikuti prinsip Clean Architecture untuk memastikan bahwa struktur kode bersih, modular, dan mudah untuk dikelola.

II. KAJIAN TEORI

A. Arsitektur perangkat lunak

Arsitektur perangkat lunak adalah abstraksi tingkat tinggi yang mewakili struktur, perilaku, dan non-fungsional sifat-sifat sistem perangkat lunak (Goddard, 2005). Arsitektur perangkat lunak mencakup kerangka kerja terstruktur di mana semua komponen aplikasi - yang bertanggung jawab

untuk memberikan fungsionalitas pengguna - ditentukan, diterapkan, dan dijalankan. Arsitektur ini tidak hanya mendefinisikan infrastruktur tingkat tinggi, tetapi juga integrasi terperinci dari komponen fungsional dan teknis di berbagai tingkat perincian, memastikan sistem memenuhi persyaratan fungsional dan non-fungsional secara efektif. [1]

B. Kotlin

Kotlin merupakan bahasa pemrograman pragmatis dan dapat dijalankan pada *virtual machine* Java dan Android. Ini menggabungkan berorientasi objek dan fungsional dan fungsional, dan sepenuhnya dapat dioperasikan dengan Java. Oleh karena itu, dimungkinkan untuk menggabungkan Kotlin dan kode Java dalam aplikasi yang sama, untuk memanggil kode Kotlin dari kode Java dan sebaliknya (Góis Mateus & Martinez, 2019). Selain memiliki smart casting, higher-order functions, dan extension functions, Kotlin juga mendukung non-nullable type yang membuat bahasa ini lebih jarang menemui error Null Pointer Exception [2].

C. Makanan Sehat

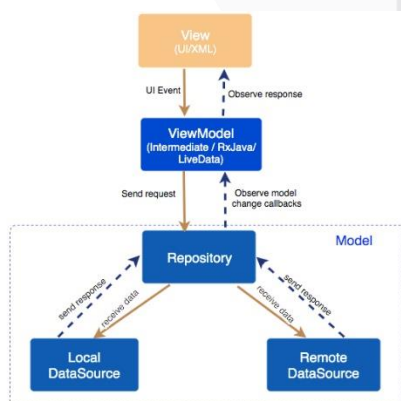
Mengonsumsi makanan yang kaya nutrisi memiliki manfaat yang besar dalam meningkatkan kekebalan supaya tubuh lebih tahan terhadap penyakit. Makanan bergizi seharusnya kaya akan serat, mineral, vitamin, dan nutrisi penting lainnya yang esensial bagi pertumbuhan dan kesehatan optimal tubuh. Beberapa contoh makanan yang bergizi meliputi daging, telur, sayuran hijau, buah-buahan, susu, ikan, dan lainnya. [3] Makanan yang sehat didefinisikan sebagai makanan yang dijamin ke higienisannya, tanpa pengawet atau bahan kimia berbahaya lainnya. (Abdullah & Ricardo, 2017)

D. Design Pattern

Design pattern adalah solusi yang umum digunakan untuk masalah yang kerap terjadi dalam mendesain perangkat lunak [4]. Pola ini bagaikan *blueprint* yang dapat disesuaikan dan sudah jadi untuk menyelesaikan masalah desain aplikasi yang bersifat berulang dalam kode. Konsep ini menyediakan metodologi standar untuk menyelesaikan masalah dengan cara yang dapat digunakan kembali di berbagai situasi tanpa memerlukan implementasi yang sama setiap kali. (Gamma, Helm, Johnson & Vlissides, 1994)

III. METODE

A. Arsitektur Aplikasi



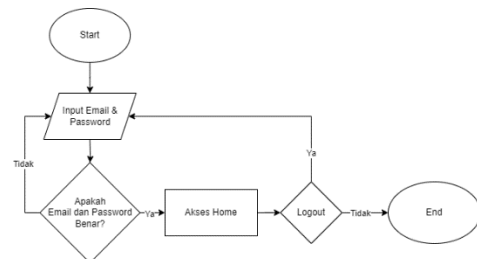
GAMBAR 1
(Diagram Arsitektur MVVM)

Aplikasi rekomendasi makanan sehat berbasis mobile menggunakan arsitektur atau pattern MVVM. Arsitektur

MVVM (Model-View-ViewModel) saat ini adalah salah satu desain arsitektur yang paling banyak digunakan saat mengembangkan aplikasi android. Model, ViewModel, dan View terdiri dari pola MVVM. View bertanggung jawab untuk membangun antarmuka, atau tampilan apa pun yang dilihat pengguna di layar. Tampilan ini dapat terdiri dari komponen elemen UI Android seperti TextView, Button, atau UI Jetpack Compose. ViewModel berfungsi sebagai penengah antara komponen Tampilan dan Model. Komponen ini independen, tidak bergantung pada Tampilan, dan menyimpan data bisnis atau status UI dari Model untuk ditransfer ke elemen UI.

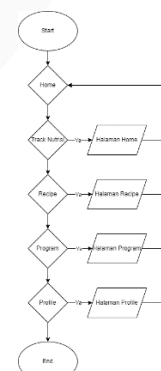
Lapisan Model mengenkapsulasi model data aplikasi, contohnya perhitungan kompleks dan logika bisnis. Layanan remote seperti Retrofit, basis data lokal, dan repositori biasanya digunakan bersamaan dengan kelas Model. Meskipun ada berbagai sumber data, repositori memastikan konsep *Single Source of Truth*. Penggunaan arsitektur MVVM untuk Android menguntungkan dalam beberapa hal, seperti *separation of concerns*, DataBinding, dan *lifecycle awareness*. Kelas View dalam MVVM dirancang untuk hanya menampung logika antarmuka/UI dan tidak memiliki data atau logika.

B. Flowchart



GAMBAR 2
(Flowchart Autentikasi)

Untuk mengakses fitur utama di aplikasi rekomendasi makanan sehat, pengguna harus *login* dengan memasukkan email dan password yang benar, di mana sistem akan memverifikasinya sebelum memberikan akses. Jika pengguna melakukan logout, mereka harus mengisi kembali email dan password untuk login ulang dan melanjutkan penggunaan aplikasi.



GAMBAR 3
(Flowchart Aplikasi)

Setelah masuk kedalam aplikasi, pengguna dapat mengakses fitur – fitur utama aplikasi seperti home, track nutrisi, resep, program, dan profile.

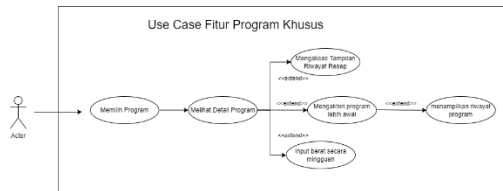
C. Class Diagram



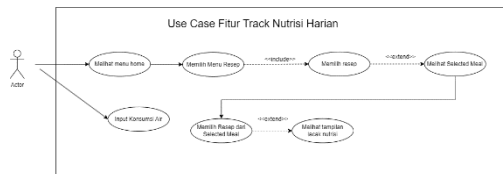
GAMBAR 4
(Class Diagram Aplikasi)

Class diagram ini merepresentasikan setiap kelas yang ada didalam aplikasi dan hubungannya satu sama lain secara *high-level*. Activity dan Fragment sebagai lapisan UI mewarisi dari AppCompatActivity dan Fragment mewarisi dari Fragment (androidx.fragment.app.Fragment). Sedangkan menu dialog kustom mewarisi dari Dialog Fragment.

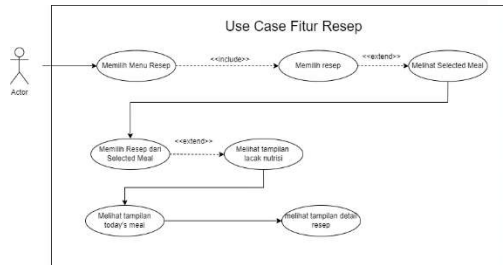
D. Diagram Use Case



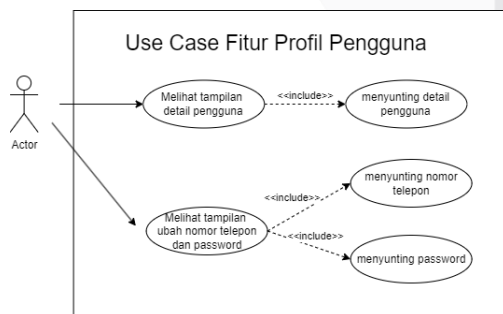
GAMBAR 4
(Diagram Use Case Fitur Program Khusus)



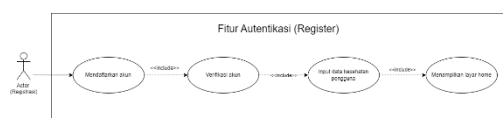
GAMBAR 5
(Diagram Use Case Fitur Track Nutrisi Hairan)



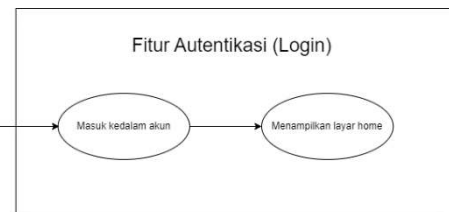
GAMBAR 6
(Diagram Use Case Fitur Resep)



GAMBAR 7
(Diagram Use Case Fitur Profile Pengguna)



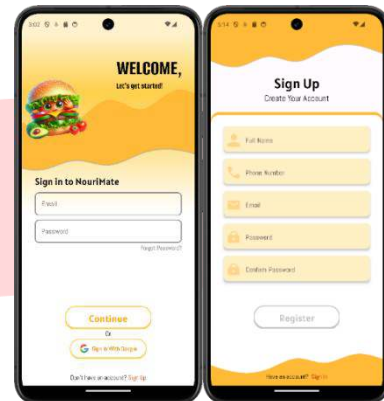
GAMBAR 8
(Diagram Use Case Fitur Register)



GAMBAR 9
(Diagram Use Case Fitur Program Login)

E. Tampilan Fitur

i. Autentikasi



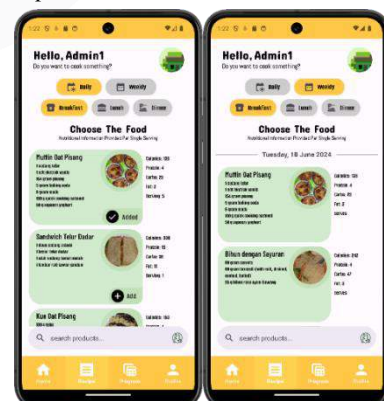
GAMBAR 10
(Tampilan UI Fitur Autentikasi)

ii. Track Nutrisi



GAMBAR 11
(Tampilan UI Fitur Track Nutrisi)

iii. Resep



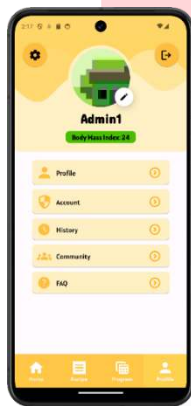
GAMBAR 12
(Tampilan UI Fitur Resep)

iv. Program Khusus



GAMBAR 13
(Tampilan UI Fitur Program Khusus)

v. Profile



GAMBAR 14
(Tampilan UI Fitur Profile)

F. Design Pattern

i. Adapter

Adapter adalah *structural design pattern* yang layaknya adapter biasa, menyediakan antarmuka untuk menghubungkan antara satu kode yang tidak cocok dengan kode lainnya.. Dalam aplikasi ini, Adapter pattern digunakan utamanya untuk menyesuaikan model data dari basis data lokal maupun remote untuk ditampilkan dalam RecyclerView. Kerap kali kedua sumber data ini memiliki struktur data yang berbeda atau cara interaksi yang tidak seragam, sehingga membutuhkan sebuah mekanisme untuk menyatukan kedua jenis sumber data tersebut agar bisa digunakan secara efektif.

ii. Singleton

Singleton adalah *creational design pattern* yang membuat sebuah kelas hanya memiliki tepat satu *instance*, sekaligus membuat *instance* ini dapat diakses secara global. Dalam konteks pengembangan Android, pattern ini sering diterapkan menggunakan companion object di Kotlin untuk mengatur pembuatan instance dari kelas seperti *repository*, database, atau datastore. Dalam contoh *repository* di aplikasi, penggunaan Singleton memastikan bahwa seluruh aplikasi berbagi satu objek *repository* yang sama.

iii. Repository

Repository Pattern merupakan design pattern yang digunakan untuk mengatur logika akses data dalam aplikasi. Dalam konteks pengembangan aplikasi Android, pattern ini mengisolasi lapisan data dari lapisan bisnis sehingga memudahkan pengujian dan pemeliharaan kode. Penggunaan pattern ini mendukung clean architecture dan single responsibility principle yang merupakan bagian dari SOLID paradigm. Suatu paradigma yang memastikan kode aplikasi tetap modular, mudah diubah serta dikembangkan.

iv. Factory

Factory adalah *creational design pattern* yang memungkinkan pembuatan objek dalam superclass, namun memperbolehkan subclass untuk memodifikasi tipe objek yang dibuat. Dalam konteks MVVM pada pengembangan aplikasi Android, Factory Method digunakan untuk memisahkan logika pembuatan instance dari ViewModel melalui ViewModelFactory, yang memungkinkan konfigurasi atau dependensi khusus disertakan saat pembuatan instance ViewModel.

v. Builder

Builder adalah *creational design pattern* yang digunakan untuk "membangun" objek kompleks secara bertahap menggunakan kode konstruksi yang sama. Dalam pengembangan aplikasi ini, Builder pattern digunakan dalam pembuatan AlertDialog dan notifikasi, di mana kedua komponen ini memerlukan konfigurasi yang kompleks sebelum ditampilkan. AlertDialog.Builder dan Notification.Builder menyediakan interface yang digunakan untuk mengatur properti objek tersebut seperti judul, pesan, ikon, dan aksi opsional sebelum objek tersebut akhirnya dibuat.

G. Android Jetpack

Android Jetpack adalah kumpulan pustaka dan alat yang membantu mempermudah proses pengembangan aplikasi Android. Dalam aplikasi ini, DataBinding digunakan untuk mengikat komponen UI secara langsung ke sumber data sehingga prosesnya lebih efisien. LiveData digunakan untuk membuat UI aplikasi reaktif apabila ada perubahan data yang perlu langsung ditampilkan. Room digunakan untuk abstraksi basis data, sementara Navigation dan WorkManager menyederhanakan navigasi antar halaman dan pengelolaan tugas yang ada di latar belakang seperti notifikasi.

IV. HASIL DAN PEMBAHASAN

Temuan akhir dari penelitian ini menunjukkan bahwa penggunaan arsitektur MVVM dalam pengembangan aplikasi rekomendasi makanan sehat memberikan pengalaman pengguna yang baik dari performa aplikasi yang baik pula. Hal ini dihasilkan dari integrasi antara arsitektur, komponen Android Jetpack, dan penggunaan *design pattern*. Proses pengembangan yang teratur menunjukkan bahwa penerapan Clean Architecture membantu membuat aplikasi dan kode yang modular dan *scalable*. Kedua faktor ini krusial untuk memudahkan perluasan fitur dan pemeliharaan

jangka panjang. Berdasarkan ini, MVVM tidak hanya mendukung kebutuhan pengguna saat ini tetapi juga memastikan aplikasi mudah dikembangkan skalanya dengan memastikan struktur kode mudah dipahami, diuji, dan modular.

Pengujian dilakukan dalam bentuk pengujian UI untuk memastikan seluruh fitur berfungsi dengan baik. Pengujian ini dilakukan secara otomatis dengan memanfaatkan library Espresso yang biasa digunakan untuk mengotomasi proses pengujian. Berikut adalah hasil pengujian UI:

TABEL 1
Hasil Pengujian UI

Halaman	Elemen UI yang Diuji	Hasil
Profile	Buttons: Change Profile Photo, Settings, Profile, Account, History, Faq Navigation, Logout	Pass
Home	Glass (Water), BMI Status, Nutrition Information	Pass
Recipe	Buttons: Breakfast, Lunch, Dinner, Daily, Weekly. SearchBar, SearchView	Pass
Program	Buttons: Choose Program, Restore Program	Pass
Register	Buttons: Submit Registration, Login Navigation. Text Inputs	Pass

V. KESIMPULAN

Penggunaan arsitektur MVVM dalam pengembangan aplikasi rekomendasi makanan sehat telah menunjukkan hasil yang signifikan dalam meningkatkan pengalaman pengguna melalui performa aplikasi yang optimal. Kombinasi antara MVVM, Clean Architecture dan design pattern menghasilkan kerangka aplikasi yang baik untuk pengembangan secara modular dan scalable, memastikan bahwa aplikasi tidak hanya memenuhi kebutuhan pengguna saat ini tetapi juga mudah disesuaikan untuk kebutuhan masa depan. Implementasi komponen Android Jetpack dalam aplikasi ini

telah memperkuat fungsionalitas dengan menyediakan solusi yang efisien untuk pengelolaan data dan UI yang reaktif, dimana hal ini sangat penting untuk aplikasi kesehatan yang membutuhkan tingkat keakuratan dan ketersediaan data yang tinggi. Dari hasil pengujian UI yang dilakukan, aplikasi ini terbukti fungsional dalam menyediakan layanan rekomendasi makanan sehat melalui fitur yang tersedia. Penelitian ini menyimpulkan bahwa penerapan teknologi informasi dalam bentuk aplikasi mobile untuk rekomendasi makanan sehat merupakan langkah strategis yang tidak hanya memperkaya kecanggihan teknologi tetapi juga mendukung upaya peningkatan kualitas kesehatan masyarakat.

REFERENSI

- [1] Solms, F. (2012) 'What is software architecture?', Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference [Preprint]. doi:10.1145/2389836.2389879.
- [2] Oliveira, V., Teixeira, L. and Ebert, F. (2020) 'On the adoption of Kotlin on Android Development: A triangulation study', 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER) [Preprint]. doi:10.1109/saner48275.2020.9054859.
- [3] F. Suraya, M. Suprayitno, R. A. Sari, F. Rifqiyah, dan D. Nafisah, "Penyuluhan Gizi Seimbang Melalui Posyandu Anak Sebagai Upaya Peningkatan Nutrisi Balita," ABDIMAS, vol. 25, no. 1, Juni 2021. [Online]. Tersedia: <https://journal.unnes.ac.id/nju/index.php/abdimas/>
- [4] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design patterns : elements of reusable object-oriented software. Boston: Addison-Wesley, 1994.
- [5] S. Zhang and S. Goddard, "xSADL: an architecture description language to specify component-based systems," Jan. 2005, doi: <https://doi.org/10.1109/itcc.2005.303>.
- [6] B. Góis Mateus and M. Martinez, "An empirical study on quality of Android applications written in Kotlin language," Empirical Software Engineering, vol. 24, no. 6, pp. 3356–3393, Jun. 2019, doi: <https://doi.org/10.1007/s10664-019-09727-4>.
- [7] A. A. Abdullah and R. Richardo, "Menumbuhkan Kemampuan Berpikir Kritis Siswa Dalam Memilih Makanan Sehat Dengan Pembelajaran Literasi Matematika Berbasis Konteks," JURNAL GANTANG, vol. II, no. 2, pp. 1-7, Sep. 2017. [Online]. Available: <http://ojs.umrah.ac.id/index.php/gantang/index>. ISSN: 2503-0671, e-ISSN: 2548-5547.