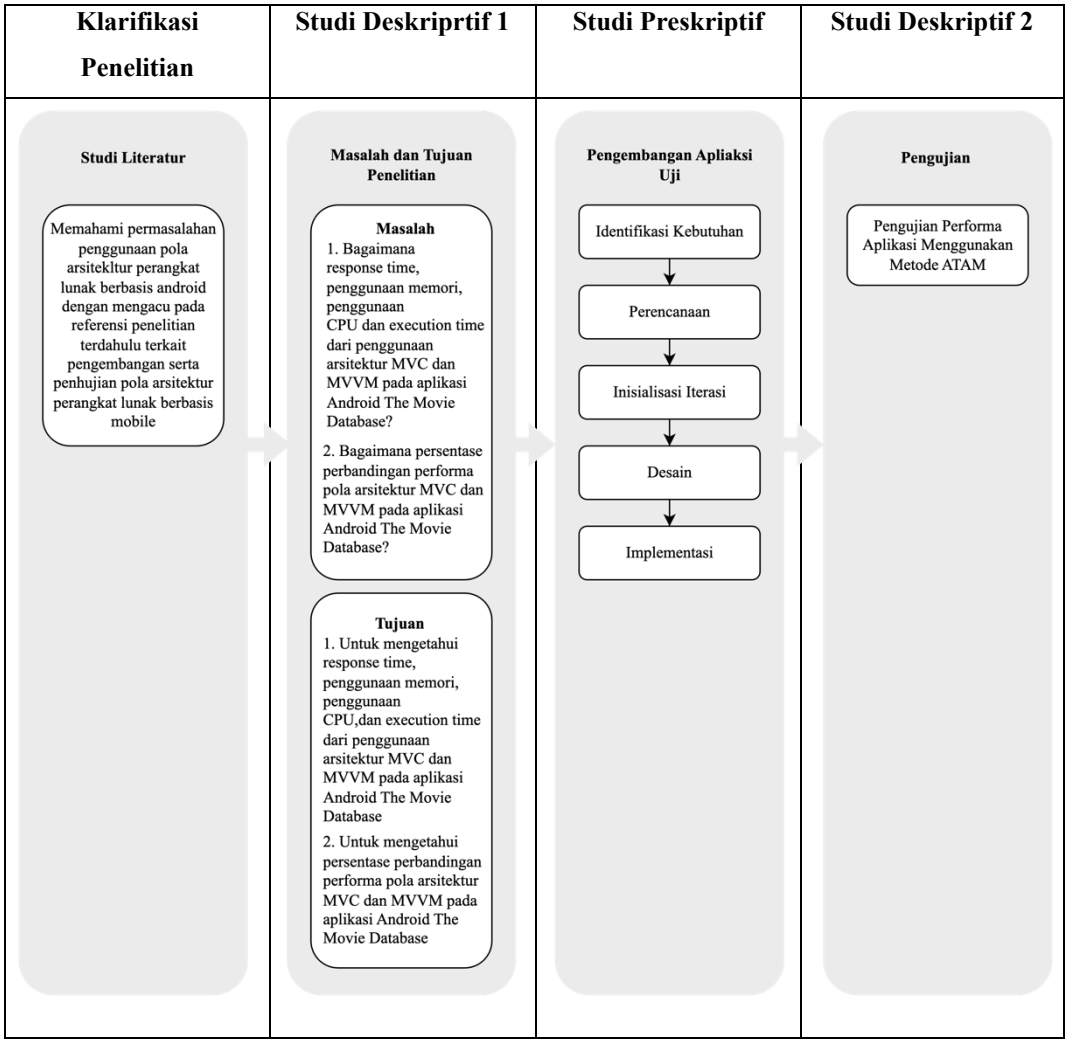
BAB III
METODOLOGI PENELITIAN

3.1. Desain Penelitian

Penelitian diawali dengan merancang aplikasi Android dengan menerapkan pola arsitektur MVVM dan MVC yang akan diuji performanya. Penelitian ini dirancang dengan menggunakan *Design Research Methodology* (DRM) yang memiliki 4 tahap dengan terdiri atas klarifikasi penelitian, studi deskriptif 1, studi preskriptif dan studi deskriptif 2. Metode ini digunakan sebagai kerangka penelitian untuk membantu dalam pengembangan dan pelaksanaan penelitian dengan lebih tepat (Lattanzio dkk., 2019). Berdasarkan metodologi tersebut, dirancang diagram atau alur penelitian seperti yang terlihat pada tabel 3.1 di bawah.

Tabel 3.1
Desain Penelitian



3.1.1. Klarifikasi Penelitian

Pada tahap pertama, Tinjauan literatur dilakukan untuk lebih memahami topik yang diangkat dalam penelitian. Tinjauan ini dilakukan dengan meninjau dan mengkaji referensi dari penelitian sebelumnya terkait dengan pengujian performa aplikasi dengan menggunakan pola arsitektur yang berbeda untuk mendapatkan hasil dari penggunaan pola arsitektur yang lebih baik dari segi performa. Pemahaman ini terdapat pada *State of The Art* yang membahas sepuluh penelitian yang berkaitan dengan penelitian yang dilakukan.

3.1.2. Studi Deskriptif 1

Pada tahap ini, dilakukan identifikasi masalah pada latar belakang yang telah ditetapkan dengan mengecu pada rumusan masalah sehingga diperoleh pokok masalah yang akan diteliti pada penelitian. Hasil dari identifikasi dan perumusan menciptakan atau menghasilkan tujuan dari penelitian yang akan dicapai sebagai hasil penelitian.

3.1.3. Studi Preskriptif

Pada tahap ini, dilakukan pengembangan aplikasi dengan tujuan untuk dijadikan aplikasi uji sebagai mana yang telah di tetapkan pada tujuan yaitu menguji performa aplikasi dengan mengimplementasikan pola arsitektur MVVM dan MVC. Tahapan Pengembangan perangkat lunak yang dihasilkan dari wawasan pada studi deskriptif 1, dimana wawasan yang diperoleh kemudian diselaraskan dengan tujuan penelitian sehingga menjadi suatu rancangan pengembangan yang tertata sendiri. Untuk mencapai tahap pengembangan ini, digunakan pendekatan *Personal Extreme Programming method* untuk mencapai pengembangan yang progresif yang dapat dilakukan oleh satu pengembang.

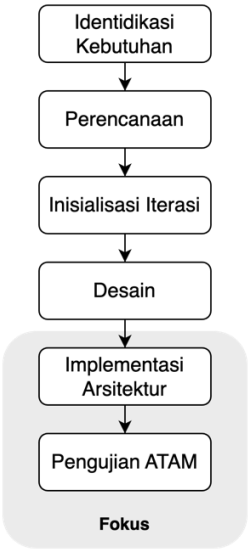
Pada penelitian ini, pengembangan aplikasi dirancang pada platform Android dengan implementasi pola arsitektur MVVM dan MVC. Memahami tujuan penelitian dan mempelajari literatur yang akan menguji performa dari MVVM dan MVC, serta membandingkan keseluruhan performa dari arsitektur tersebut. Hal yang paling membedakan dari pola arsitektur tersebut adalah pada bagian ViewModel pada MVVM dan *Controller* pada MVC. Perbedaan tersebutlah yang dapat memberikan hasil performa yang berbeda.

3.1.4. Studi Deskriptif 2

Pada langkah terakhir, dilakukan identifikasi terhadap perangkat lunak yang dibuat untuk tujuan utama dari penelitian. Identifikasi ini dilakukan dengan menguji performa terhadap arsitektur perangkat lunak. Pengujian utama dilakukan untuk menguji performa kedua aplikasi yang dibangun menggunakan arsitektur MVVM dan MVP dengan menganalisis penggunaan CPU, penggunaan memori, dan kecepatan *respons* aplikasi.

3.2. Desain Dan Tahap Pengembangan

Untuk menghasilkan perangkat lunak secara efisien sesuai rencana pengembangan, serta mempersiapkan perangkat lunak sebagai alat uji penelitian atau pengujian eksperimental, perlu diterapkan metode pengembangan yang sesuai dengan kebutuhan. Pada penelitian ini, pengembangan aplikasi akan menggunakan metode *Personal Extreme Programming* (PXP). Metode PXP dipilih karena merupakan metode pengembangan berbasis *agile* yang cocok untuk dikembangkan oleh satu individu. Metode ini memungkinkan pengembangan yang cepat, fleksibel, dan adaptif, dengan umpan balik yang terus menerus setelah setiap tahap, memungkinkan perubahan yang bertahap dilakukan (Lestari dkk., 2023). Pada penelitian ini, metode PXP dimodifikasi sedemikian rupa untuk disesuaikan dengan fokus utama pada penelitian ini yaitu pengujian dan analisis performa arsitektur perangkat lunak, sehingga urutan pengembangannya seperti pada diagram alur pada gambar 3.1 berikut



Gambar 3.1 Tahapan Metode PXP

Tahapan dalam pengembangan menggunakan metode PXP yang dimodifikasi mengikuti pola yang dijelaskan pada gambar 3.1. Proses dimulai dengan mengidentifikasi kebutuhan dan merencanakan pengembangan perangkat lunak. Selanjutnya, langkah berlanjut ke fase pengembangan yang mencakup beberapa tahapan utama, seperti inisialisasi iterasi, perancangan, implementasi arsitektur, dan pengujian ATAM (*Architecture Tradeoffs Analysis Method*).

3.2.1. Identifikasi Kebutuhan

Identifikasi kebutuhan merupakan langkah awal dalam proses pengembangan dengan metode PXP. Tujuan dari langkah ini adalah untuk mengidentifikasi kebutuhan utama sistem aplikasi yang akan dikembangkan, baik yang berkaitan dengan kebutuhan fungsional maupun non-fungsional. Dalam pengembangan aplikasi untuk penelitian ini, identifikasi kebutuhan secara fitur dilakukan berdasarkan kebutuhan aplikasi yang dapat memuat berbagai film dan dapat melihat data-data terkait untuk mendapatkan informasi yang bermanfaat.

3.2.2. Perencanaan

Perencanaan merupakan tahap pengebangan kedua dengan menggunakan metode PXP. Rencana pengembangan aplikasi ini akan disesuaikan dengan hasil identifikasi kebutuhan yang diperoleh. Perencanaan ini akan menghasilkan tahapan pengembangan aplikasi yang berfokus pada kebutuhan fungsional dan non-fungsional. Selain itu, pengembangan akan dimulai dengan menggunakan arsitektur MVVM dan MVC sesuai dengan tujuan penelitian ini. Hasil perencanaan juga akan menunjukkan bagaimana sistem yang akan dibuat, baik dari sudut pandang antarmuka aplikasi, interaksi data dalam aplikasi, rancangan logika sebagai model, dan sumber data yang akan menjadi dasar dari jalannya fitur aplikasi yang dikembangkan.

3.2.3. Inisialisasi Iterasi

Dalam metode PXP, langkah ketiga adalah inisialisasi iterasi. Memahami setiap proses yang akan dilakukan, apa saja yang menjadi prioritas utama dalam pengembangan, tingkat kesulitan, dan waktu yang diperlukan untuk menyelesaikan setiap tugas. Pada pengembangan untuk penelitian ini, alur pengerjaan akan dibagi berdasarkan fitur untuk menjadi prioritas utama yang akan dikerjakan. Untuk acuan pemberian skala atau prioritas fitur tidak ada acuan tetap seperti pada penelitian

yang dilakukan (Daoudi dkk., 2019; Epiloksa dkk., 2022; Kustino Muharram dkk., 2021a, 2021b; Lestari dkk., 2023; Maulana dkk., 2022), peneliti tersebut memberikan nilai skala berdasarkan studi kasus penelitian itu sendiri.

3.2.4. Desain

Dalam tahap keempat metode PXP, desain merupakan peranan penting. Dalam penelitian ini, desain aplikasi, baik antarmuka maupun UML (*Unified Modeling Language*) seperti *Use Case* diperlukan untuk menjadi referensi dari aplikasi yang biasanya digunakan untuk pendukung kerja ataupun pengembangan. Tujuan dari desain ini adalah supaya aplikasi dapat dikembangkan sesuai dengan kebutuhan dan siap untuk diuji serta disimulasikan.

3.2.5. Implementasi Arsitektur

Implementasi adalah tahap kelima metode PXP, dimana aplikasi dikembangkan sesuai dengan perencanaan, iterasi, dan desain yang telah dibuat pada tahap sebelumnya. Untuk memaksimalkan penelitian ini, kedua pola arsitektur yaitu MVVM dan MVC diimplementasikan pada pengembangan aplikasi untuk dijadikan aplikasi uji.

3.2.6. Pengujian ATAM

Pengujian ATAM dilakukan pada langkah keenam metode PXP. Dalam pengembangan ini, aplikasi diuji melalui pengujian ATAM untuk dilakukan analisis performa arsitekturnya. Pengujian ATAM akan membantu peneliti menentukan arsitektur perangkat lunak seperti apa yang baik untuk digunakan dengan menekankan pada performanya. Pengujian ini akan dilakukan pada skala kecil dengan hanya beberapa skenario yang sesuai dengan fitur yang telah dibuat dalam aplikasi.

3.3. Alat dan Bahan penelitian

Pengembangan dan pengujian aplikasi pada penelitian ini menggunakan perangkat komputer milik peneliti sebagai alat penelitiannya. Spesifikasi alat penelitian yang digunakan dapat dilihat pada tabel 3.2 berikut.

Tabel 3.2
Perangkat Uji (Komputer) dan Perangkat Lunak

Chip	Apple Silicone M1
RAM	16 GB

OS	macOS Sonoma 14.5
IDE	Android Studio Hedgehog 2023.1.1
Software Design	Figma
Smartphone	Samsung Galaxy A24 (Android 14)
Jaringan	Wifi (20 Mbps)

Selain alat berupa perangkat komputer, terdapat alat untuk menguji performa aplikasi Android, yaitu *profiling* menggunakan alat yang tersedia pada IDE Android Studio yang dapat menampilkan penggunaan CPU, penggunaan memori, penggunaan energi, dan waktu *startup*. Pengujian aplikasi dilakukan pada *smartphone* berbasis Android dengan beberapa *background process.* Perangkat pengujian aplikasi yang digunakan dapat dilihat pada tabel 3.3 berikut.

Tabel 3.3
Perangkat Uji (Smartphone)

Chip	MediaTek Helio G99
CPU	Octa-core (2x2.2 GHz Cortex-A76 & 6x2.0 GHz Cortex-A55)
GPU	Mali-G57 MC2
RAM	8 GB
OS	One UI 6.1 (Android 14)

Selain Alat pengujian, terdapat alat untuk analisis data numerik yang digunakan untuk menganalisis data yang didapatkan dari hasil pengujian. Alat untuk analisis data dapat dilihat pada tabel 3.4 berikut.

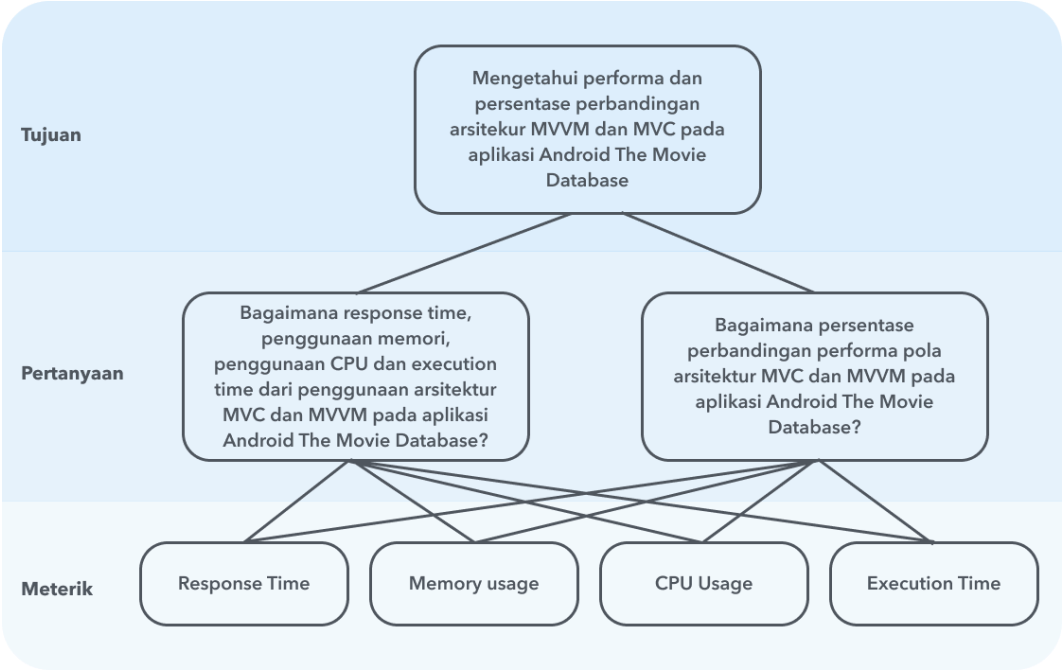
Tabel 3.4
Alat Analisis Data

Pengolah Angka 1	Microsoft Excel
Pengolah Angka 2	JASP

3.4. Instrumen Penelitian

Instrumen penelitian yang digunakan pada penelitian ini digambarkan pada Gambar *Goal Question Metric* (GQM). Pada gambar tersebut ditunjukkan bahwa metrik-metrik apa saja yang mewakili atau yang dapat mengukur permasalahan

yang ada pada pertanyaan rumusan masalah untuk mengantarkan penelitian kepada tujuan yang diteliti.



Gambar 3.2 Goal Question Metric

Pada gambar 3.2 di atas terdapat 4 metrik yang menjadi alat ukur pada penelitian ini berdasarkan pertimbangan dari penelitian sebelumnya yang dilakukan oleh (Daoudi dkk., 2019; Epiloksa dkk., 2022; Kustino Muharram dkk., 2021b; Maulana dkk., 2022), yaitu *response time* yang didapatkan ketika menerima *response* setelah *request* ke server, penggunaan memori dan CPU yang didapatkan dari suatu aktivitas aplikasi yang sedang berjalan sesuai skenario uji yang telah dibuat, serta *execution time* yang didapatkan dari waktu eksekusi sebuah tugas.

3.4.1. *Response Time*

Response time atau *response time* diukur pada saat aplikasi meminta *request* terhadap server, dan menerima *response*. *Response time* diukur dengan satuan *millisecond* (ms). Suatu pola arsitektur dikatakan lebih baik jika memiliki *response time* yang cepat. Menurut (Jakob Nielsen, 2020) *response time* yang baik berada pada kisaran 100 ms sampai dengan di bawah 1 detik.

3.4.2. *Memory Usage*

Memory Usage atau penggunaan memori diukur pada saat aplikasi melakukan serangkaian aktivitas yang telah dirancang pada skenario tes.

Penggunaan memori diukur dengan satuan *Mega Byte* (MB). Suatu pola arsitektur dikatakan lebih baik jika memiliki penggunaan memori yang lebih rendah. Menurut (Anastasia Finogenova, 2020) penggunaan memori yang efisien adalah antara 70 MB dan lebih rendah.

3.4.3. CPU *Usage*

CPU *Usage* atau penggunaan CPU diukur pada saat aplikasi melakukan serangkaian aktivitas yang telah dirancang pada skenario tes seperti halnya pengukuran penggunaan memori. Penggunaan CPU dikur dengan satuan persen (%). Suatu pola arsitektur dikatakan lebih baik jika memiliki penggunaan CPU yang lebih rendah. Menurut (Shelia Negron, 2023) penggunaan CPU yang ideal berkisar pada 12% dan lebih rendah.

3.4.4. *Execution Time*

Execution Time atau waktu eksekusi diukur pada saat aplikasi melakukan serangkaian aktivitas yang telah dirancang pada skenario tes. Waktu eksekusi dikur dengan satuan *millisecond* (ms). Suatu pola arsitektur dikatakan lebih baik jika memiliki waktu eksekusi pada serangkaina tugas yang lebih cepat. Menurut (Jakob Nielsen, 2020) *response time* yang baik berada pada kisaran 100 ms sampai dengan di bawah 1 detik.

3.4.5. Skenario Pengujian Performa

Adapun skenario pengujian performa secara umum untuk menunjang alur pengujian performa pada aplikasi yang dikembangkan dan hasil uji yang datanya akan dianalisis dan dibandingkan sebagai penelitian dari perbandingan performa pola arsitektur MVVM dan MVC serta dampak dari pada pola arsitektur pada performa aplikasi. Berikut tabel skenario uji yang dijadikan bahan penelitian.

Tabel 3.5
Skenario Pengujian Performa

No	ID Uji	Skenario	Tujuan
1	TC-01	*Home*	Mengetahui kecepatan, penggunaan CPU, penggunaan memori, dan waktu eksekusi ketika aplikasi berada di halaman *home* yang

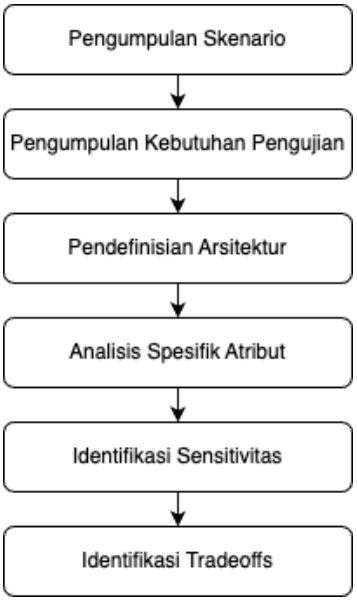
			menjalankan fitur "menampilkan *list* genre, menampilkan 5 data *upcoming movies*, dan menampilkan 5 data *popular movies*"
2	TC-02	*Movie List*	Mengetahui kecepatan, penggunaan CPU, penggunaan memori, dan waktu eksekusi ketika aplikasi berada di halaman *movie list* yang menjalankan fitur "menampilkan daftar film sesuai fitur yang dipilih pada halaman *home*"
3	TC-03	*Detail Movie* pada *Movie List*	Mengetahui kecepatan, penggunaan CPU, penggunaan memori, dan waktu eksekusi ketika aplikasi berada di halaman *movie detail* yang menjalankan fitur "menampilkan data detail dari film yang dipilih dari halaman *movie list*"
4	TC-04	*Detail Movie* pada *Home*	Mengetahui kecepatan, penggunaan CPU, penggunaan memori, dan waktu eksekusi ketika aplikasi berada di halaman *movie detail* yang menjalankan fitur "menampilkan data detail dari

film yang dipilih dari halaman *home*"

Tabel 3.5 di atas merupakan skenario dari uji performa aplikasi yang dikembangkan untuk pola arsitektur MVVM dan MVC. Skenario di atas akan dijalankan sesuai urutan dan data-datanya akan diambil berdasarkan *activity* yang berjalan.

3.5. Prosedur Penelitian

Untuk menunjang keberlangsungan dan keberhasilan penelitian ini, beberapa prosedur harus dilakukan. Pada penelitian ini menggunakan *Architecture Tradeoffs Analysis Method* (ATAM). ATAM adalah model spiral yang merupakan metode yang digunakan untuk menganalisis dan mengevaluasi sebuah desain arsitektur dengan berfokus pada penilaian dari beberapa atribut seperti kemudahan untuk melakukan modifikasi, performa dari penggunaan arsitektur yang digunakan, dan ketersediaan produk yang dibutuhkan dalam mengetahui apakah arsitektur tersebut telah sempurna sepenuhnya dalam memenuhi sebuah persyaratan tertentu (Axelsson dkk., 2024). Terdapat beberapa metode selain ATAM yang dapat digunakan, namun metode ATAM ini lebih cocok untuk digunakan pada penelitian ini karena pada tahapannya sangat berhubungan dengan pengujian yang akan dilakukan. Tahapan ATAM dapat dilihat pada gambar 3.3 berikut



Gambar 3.3 Tahapan ATAM

3.5.1. Pengumpulan Skenario

Pada tahap ini, skenario uji disusun sedemikian rupa sesuai dengan kasus yang akan diuji sesuai fitur yang didefinisikan pada analisis kebutuhan dalam pengembangan aplikasi, yaitu *home*, *movie list*, dan *movie detail*. Skenario yang digunakan telah didefinisikan pada instrumen penelitian di atas.

3.5.2. Pengumpulan Kebutuhan Pengujian

Pada tahap ini, setelah skenario pengujian dikumpulkan, selanjutnya mengumpulkan kebutuhan pengujian. Pengujian aplikasi pada penelitian ini berfokus pada pengujian performa, sehingga diperlukan perhatian khusus terhadap lingkungan pengujian dengan tujuan untuk menghilangkan variabel atau nilai lain yang dapat mengubah hasil pengujian.

3.5.3. Pendefinisian Arsitektur

Pada tahap ini, setelah kebutuhan pengujian dikumpulkan, selanjutnya pendefinisian arsitektur, membandingkan suatu pola arsitektur membutuhkan pendefinisian untuk memahami arsitektur yang lebih baik. Setiap pola arsitektur dideskripsikan mulai dari setiap komponen hingga cara kerja pola arsitektur tersebut. Pendefinisian Arsitektur telah dipaparkan pada Bab 2.

3.5.4. Analisis Spesifik Atribut

Pada tahap ini, setelah pendefinisian arsitektur, selanjutnya analisis spesifik atribut. Atribut khusus pada penelitian ini dijelaskan pada *instrument* penelitian terkait metrik-metrik penelitian yang digunakan untuk mengukur pertanyaan penelitian yang mengantarkan pada tujuan penelitian.

3.5.5. Identifikasi Sensitivitas

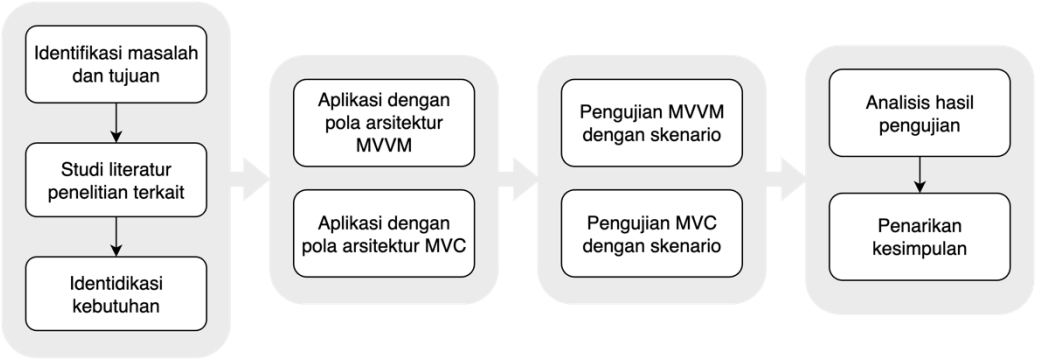
Pada tahap ini, dilakukan untuk bertujuan memahami hasil yang didapat dari tahap ke-4 dengan memahami bagai mana hasil pengujian menghasilkan nilai-nilai tertentu dengan implementasi rancangan pola arsitektur. Tahap ini menghasilkan pengetahuan atau pemahaman mengenai nilai kualitas dari pola arsitektur yang diimplementasikan.

3.5.6. Identifikasi *Tradeoffs*

Pada tahap ini, *tradeoffs* diidentifikasi untuk mengetahui kelebihan dan kekurangan dari arsitektur yang diterapkan. Proses identifikasi pada tahap ini akan memberikan wawasan terhadap peneliti maupun pengembang aplikasi Android

untuk memperkuat keputusan penggunaan pola arsitektur dalam aplikasi yang sedang dikembangkan.

Oleh karena itu, setelah memaparkan metode menggunakan ATAM dengan menunjukkan langkah-langkah metode untuk melakukan penelitian pengujian, prosedur penelitian digambarkan pada gambar 3.4 berikut.



Gambar 3.4 Prosedur Penelitian

3.5.7. Hipotesis Penelitian

Berikut merupakan hipotesis penelitian berdasarkan rumusan masalah yang sudah didefinisikan sebelumnya pada Bab 1 dimana terdiri dari hipotesis nol (H_0) dan hipotesis sementara (H_1). Hipotesis ini didefinisikan dengan bertujuan untuk menjawab rumusan masalah *point* 2. Rumusan masalah *point* 2 sendiri akan dipecah menjadi 4 bagian hipotesis karena pada penelitian ini terdapat 4 metrik yang menjadi acuan performa suatu pola arsitektur. Berikut hipotesis-hipotesis yang didefinisikan:

1) Perbedaan signifikan *response time*:
   - H_0= Tidak terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik *response time*
   - H_1 = Terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik *response time*
2) Perbedaan signifikan penggunaan memori:
   - H_0= Tidak terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik penggunaan memori
   - H_1 = Terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik penggunaan memori
3) Perbedaan signifikan penggunaan CPU:

- H_0= Tidak terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik penggunaan CPU
- H_1 = Terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik penggunaan CPU

4) Perbedaan signifikan *execution time*:

- H_0= Tidak terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik *execution time*
- H_1 = Terdapat perbedaan yang signifikan dari hasil penerapan masing-masing pola arsitektur terhadap metrik *execution time*

3.6. Analisis Data

Analisis data dilakukan menggunakan alat pemrosesan angka seperti Microsoft Excel untuk keperluan kalkulasi dan visualisasi data mentah dari setiap metrik yang digunakan untuk dikalkulasi menjadi nilai hasil dari penelitian. Selain menggunakan Microsoft Excel, menggunakan alat untuk analisis data numerik seperti JASP untuk kebutuhan analisis uji normalitas, uji Mann-Whitney dan uji t untuk menunjang keberhasilan penelitian. Terdapat beberapa ragam analisis data menurut (Candra Susanto dkk., 2024), pada penelitian ini terdapat 3 ragam analisis data yang digunakan, berikut analisis data yang digunakan pada penelitian ini.

1. Uji Normalitas

   Uji normalitas pada penelitian ini dilakukan untuk mengetahui apakah data yang dikumpulkan berdistribusi normal atau tidak. Uji normalitas ini juga bertujuan untuk menentukan langkah atau pengujian selanjutnya sebelum data akhir didapatkan.

2. Uji Mann-Whitney

   Uji Mann-Whitney dilakukan Ketika hasil dari uji normalitas, datanya tidak berdistribusi normal. Pada penelitian ini adalah mengetahui ada atau tidaknya perbedaan performa dari masing-masing aspek *response time*, penggunaan memori, penggunaan CPU, dan *execution time* pada pola arsitektur MVC dan MVVM.

3. Uji t

   Uji t dilakukan ketika hasil dari uji normalitas, datanya berdistribusi normal. Uji

ini seperti uji Mann-Whitney yaitu pada penelitian ini digunmakan untuk mengetahui ada atau tidaknya perbedaan performa dari masing-masing aspek *response time*, penggunaan memori, penggunaan CPU, dan *execution time* pada pola arsitektur MVC dan MVVM.

3.6.1. Pengumpulan Data

Proses ini akan mengikuti langkah-langkah yang diuraikan dalam poin 3.5.4, yang membahas analisis spesifik atribut. Komponen data yang akan dikumpulkan dalam penelitian ini akan sesuai dengan metrik pengujian yang disebutkan dalam poin 3.4. Data yang dimaksud untuk dikumpulkan adalah sebagai berikut:

- Rerata skor setiap metrik
- Rerata dari rerata skor setiap metrik
- Perbandingan rerata total skor pada setiap metrik antara pola arsitektur

Untuk mendapatkan data-data tersebut, dibutuhkan persamaan untuk mencari rerata maupun perbandingan dari rerata pada data. Berikut persamaan yang dapatkan dari penelitian (Epiloksa dkk., 2022) dan digunakan pada penelitian ini:

1. Rerata skor setiap metrik

$$\overline{x}_{metrik/activity} = \frac{\sum x_{data}}{n_{data}} \quad (1)$$

Dengan $\overline{x}_{metrik/activity}$ merupakan rerata skor pada suatu metrik di setiap *activity* yang diuji, x_{data} merupakan jumlah dari data yang diambil, dan n_{data} merupakan total banyaknya data atau iterasi yang didapatkan pada satu skenario atau *activity*.

2. Rerata dari rerata skor setiap metrik

$$\overline{x}_{metrik} = \frac{\sum \overline{x}_{metrik/activity}}{n_{activity}} \quad (2)$$

Dengan $\overline{x}_{metrik}$ merupakan rerata dari rerata skor setiap metrik yang diuji, $\overline{x}_{metrik/activity}$ merupakan jumlah dari rerata skor pada suatu metrik, dan $n_{activity}$ merupakan total banyaknya skenario atau *activity* yang diuji.

3. Persentase perbedaan hasil rerata total skor setiap metrik antara arsitektur

$$\%D_{activity} = \frac{|\overline{x}_{arc2} - \overline{x}_{arc1}|}{\overline{x}_{arc1}} \times 100 \quad (3)$$

Dengan $\%D_{activity}$ merupakan *percent difference* atau persentase perbedaan hasil rerata total skor setiap metrik antara arsitektur, $\overline{x}_{arc}$ merupakan rerata dari satu pola arsitektur dari semua skenario atau *activity*. Pada penelitian ini $\overline{x}_{arc2}$ merupakan rerata arsitektur yang lebih besar dari arsitektur lainnya dan $\overline{x}_{arc1}$ merupakan kebalikannya yaitu rerata arsitektur yang lebih kecil dari arsitektur lainnya serta penyebutan perbedaannya dapat dikatakan sebagai contoh, rerata arsitektur a lebih rendah berapa persen dari rerata arsitektur b.

Selain persamaan-persamaan di atas yang digunakan untuk mengumpulkan data, adapun persamaan yang digunakan untuk menentukan jumlah data sampel yang dibutuhkan untuk pengujian berdasarkan data penelitian terdahulu dengan *significance level* 5% dan *desired power* 80% menurut (Das dkk., 2016). Berikut persamaan yang digunakan

$$n = \left(\frac{(Z_{\frac{\alpha}{2}} + Z_{\beta})^2}{d^2} \right) \quad (4)$$

Dengan n merupakan banyak data, $Z_{a/2}$ merupakan *Z-Score* dari *significance level*, Z_{β} merupakan *Z-Score* dari *desired power*, dan d merupakan *effect size* (Cohen's d) dengan persamaan $d = \frac{arc_2 - arc_1}{SD_{pooled}}$ yang dimana arc_1 dan arc_2 merupakan arsitektur yang dibandingkan, dan SD_{pooled} merupakan *pooled standard deviation* yang dihitung menggunakan persamaan $SD_{pooled} = \sqrt{\frac{{SD_{arc1}}^2 + {SD_{arc2}}^2}{2}}$ dengan SD_{arc1} dan SD_{arc2} merupakan *standard deviation* dari arsitektur yang digunakan.

Data awal dari pengujian diperlukan untuk mendapatkan data-data tersebut. Alat yang tersedia dalam IDE Android Studio, Android Profiler dan logcat, memungkinkan peneliti untuk melihat penggunaan sumber daya saat aplikasi digunakan. Alat ini memungkinkan peneliti untuk melihat data untuk metrik *response time*, penggunaan CPU, memori, dan *execution time*. Data yang telah terkumpul akan dimuat dengan teratur menggunakan tabel. Tabel yang terdiri dari setiap komponen metrik yang diteliti akan digunakan untuk mengumpulkan data secara teratur dan setiap variabel akan diwakili dengan simbol yang disesuaikan,

seperti *response time* (Rt), penggunaan CPU (Cu), penggunaan memori (Mu), *execution time* (Xt).

Tabel 3.6
Tabel Data Pengujian

Activity	**Response Time (ms)**	**Penggunaan CPU (%)**	**Penggunaan Memori (MB)**	**Execution Time (ms)**
$Activity_1$	Rt_1	Cu_1	Mu_1	Xt_1
$Activity_2$	Rt_2	Cu_2	Mu_2	Xt_2
$Activity_{...}$	$Rt_{...}$	$Cu_{...}$	$Mu_{...}$	$Xt_{...}$
$Activity_n$	Rt_n	Cu_n	Mu_n	Xt_n