

## BAB II


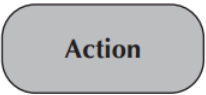


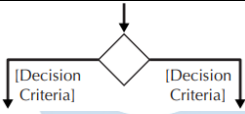

### LANDASAN TEORI

#### 2.1 Tinjauan Teori

##### 2.1.1 *Activity Diagram*

*Activity diagram* merupakan sebuah visualisasi berbentuk rancangan struktur aliran aktivitas pada sebuah sistem yang akan dijalankan. Fungsinya yakni memperlihatkan urutan aktivitas pada sistem, dan membantu memahami proses secara keseluruhan [8]. Tabel 2.1 merupakan simbol yang digunakan dalam *activity diagram*:





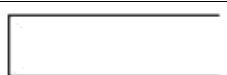


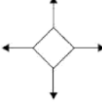


Tabel 2.1 Simbol *activity diagram*

Simbol	Nama	Keterangan
	<i>Activity</i>	Objek yang memperlihatkan masing-masing kelas antar muka saling berinteraksi.
	<i>Action</i>	Objek yang memperlihatkan pemrosesan bahwa sistem melakukan suatu aksi.
	<i>Start</i>	Objek mengawali aktivitas.
	<i>End</i>	Objek mengakhiri aktivitas.
	<i>Decision</i>	Objek yang digunakan untuk menggambarkan suatu validasi atau pilihan (Ya/Tidak).
	<i>Line connector</i>	Objek yang digunakan untuk menghubungkan simbol satu dengan simbol lainnya.

##### 2.1.2 *Flowchart*

*Flowchart* merupakan penggambaran langkah-langkah dari algoritma dalam bentuk diagram [9]. Tabel 2.2 merupakan simbol standar yang digunakan dalam *flowchart*:

Tabel 2.2 Simbol *flowchart*

Simbol	Nama	Keterangan
	<i>Terminal</i>	Digunakan untuk menandakan awal dan akhir dari serangkaian proses.
	<i>Input/output</i>	Digunakan untuk menandakan operasi <i>input/output</i> .
	<i>Computer processing</i>	Digunakan untuk menandakan proses komputasi.
	<i>Predefined processing</i>	Digunakan untuk menandakan proses yang tidak didefinisikan secara khusus dalam <i>flowchart</i> .
	<i>Comment</i>	Digunakan untuk menulis penjelasan untuk menjelaskan sesuatu.
	<i>Flow line</i>	Digunakan untuk menghubungkan simbol
	<i>Document Input/Output</i>	Digunakan ketika <i>input/output</i> berasal dari dokumen.
	<i>Decision</i>	Digunakan ketika terdapat kondisi dimana keputusan harus diambil untuk menentukan aksi selanjutnya.
	<i>On-page connector</i>	Digunakan untuk menghubungkan bagian proses yang dilanjutkan pada halaman yang sama.
	<i>Off-page connector</i>	Digunakan untuk menghubungkan bagian proses yang dilanjutkan pada halaman yang berbeda.

### 2.1.3 Bahasa Pemrograman

Bahasa pemrograman adalah sebuah instruksi standar untuk memerintah komputer agar menjalankan fungsi tertentu [10]. Bahasa pemrograman memungkinkan seorang *programmer* mendefinisikan rangkaian perintah yang dijalankan komputer untuk melakukan sebuah tugas yang spesifik. Berdasarkan tingkat kedekatannya dengan mesin komputer, bahasa pemrograman terdiri dari:

1. Bahasa mesin, yaitu perintah yang diberikan kepada komputer dengan menggunakan kode biner, sebagai contoh: 100011101.

2. Bahasa tingkat rendah, yaitu perintah yang diberikan kepada komputer dengan memakai kode-kode singkat (kode mnemonic), sebagai contoh: SUB, CMP, JMP, JGE, JL, LOOP, dsb.
3. Bahasa tingkat menengah, yaitu bahasa komputer yang memakai campuran instruksi dalam kata-kata bahasa tingkat tinggi dan instruksi yang bersifat simbolik, sebagai contoh: {, }, ?, <<, >>, &&, ||, dsb.
4. Bahasa tingkat tinggi, yaitu bahasa komputer yang berasal dari unsur-unsur bahasa manusia, sebagai contoh: begin, end, if, for, while, and, or, dsb.

#### **2.1.4 Java**

Bahasa pemrograman Java dikembangkan oleh James Gosling yang dibantu oleh rekan-rekannya di suatu perusahaan perangkat lunak yang bernama *Sun Microsystems*, pada tahun 1991. Bahasa pemrograman ini mula-mula bernama “Oak”, namun pada tahun 1995 “Oak” berganti nama menjadi “Java”.

Java adalah sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *stand-alone* ataupun pada lingkungan jaringan [11].

Java adalah bahasa pemrograman berorientasi objek tingkat tinggi, dan program Java terdiri dari bagian-bagian yang disebut kelas. Kelas terdiri dari metode yang bekerja dan mengembalikan informasi setelah melakukan tugasnya [12].

Berdasarkan definisi di atas, dapat disimpulkan bahwa Java merupakan bahasa pemrograman tingkat tinggi yang digunakan untuk membuat dan menjalankan perangkat lunak pada komputer *stand-alone* maupun pada lingkungan jaringan.

#### **2.1.5 Multithreading**

*Multithreading* merupakan kemampuan CPU untuk menyediakan banyak ketersediaan eksekusi sekaligus. Proses *multithreading* dapat membuat beberapa *thread* yang menggunakan memori virtual yang sama, masing-masing *thread*

dapat mengeksekusi kode pada *core* nya masing-masing. *Multithreading* adalah antarmuka yang disediakan untuk memungkinkan eksekusi paralel [13].

Multithreading menerapkan suatu proses eksekusi sebuah kerja program komputer yang dikerjakan secara bersamaan sehingga mampu mempercepat proses pada suatu program pada komputer [14].

### **2.1.6 Basis Data**

Dalam buku yang berjudul “Teori Basis Data”, Basis Data adalah kumpulan data terintegrasi yang diatur untuk memenuhi kebutuhan pengguna di sebuah organisasi [15].

*Database* adalah salah satu komponen kunci dari sistem informasi, karena berfungsi sebagai dasar untuk menyediakan informasi kepada pengguna. Penggunaan *database* dalam sistem informasi disebut dengan *database* sistem [16].

Berdasarkan pengertian di atas, basis data merupakan data yang terintegrasi yang berfungsi sebagai basis penyedia informasi bagi para pemakai di dalam suatu organisasi.

### **2.1.7 Basis Data NoSQL**

NoSQL (*Not Only SQL*), merupakan basis data non-relasional yang menjadi alternatif dari *database* SQL. Perbedaan antara kedua tipe basis data tersebut adalah pada format skema. SQL memiliki skema yang kaku, sedangkan basis data NoSQL memiliki bentuk skema yang lebih fleksibel dan mudah diubah tanpa mengganggu sistem yang sedang berjalan [17].

Penerapan basis data NoSQL memungkinkan para pengembang aplikasi untuk mengembangkan skema tanpa harus merubah struktur memori ke struktur relasional. Basis data NoSQL dibuat untuk menangani konsep “Big Data” dengan memanfaatkan mesin-mesin server yang terdistribusi. [17].

### **2.1.8 Stored Procedure**

*Stored procedure* adalah sekelompok pernyataan SQL yang sebelumnya telah dikompilasi ke dalam satu rencana eksekusi, dibuat dan disimpan dalam

*database* sehingga dapat digunakan kembali saat dibutuhkan [18]. Dengan menggunakan *stored procedure*, *programmer* tidak perlu menulis ulang baris kode SQL saat perintah SQL yang sama dibutuhkan. Sebagian besar produk basis data yang dipakai di industri sudah mendukung penggunaan *stored procedure* sebagai contoh: Oracle, Microsoft SQL Server, IBM DB2, dan lain sebagainya.

### **2.1.9 Extract Transform Load (ETL)**

ETL merupakan singkatan dari *extract*, *transform*, *load* secara sederhana didefinisikan sebagai set proses untuk mendapatkan data dari OLTP (*online transaction processing*) masuk ke *data warehouse* [19].

ETL (*Extraction, Transformation, Load*) merupakan proses mengambil dan mengirim data dari data sumber ke data warehouse [20].

Berdasarkan definisi di atas, dapat disimpulkan bahwa proses ETL merupakan serangkaian proses mendapatkan data dari sumber dan memuatnya ke dalam *data warehouse*.

### **2.1.10 Application Programming Interface (API)**

*Application Programming Interface* atau API adalah integrasi dari dua bagian dari sistem aplikasi. API terdiri dari elemen *function*, *protocols*, dan *tools* lainnya digunakan pengembang untuk membangun sebuah aplikasi [21].

*Application programming interface* (API) adalah dokumen yang terdiri dari antarmuka, fungsi, kelas, struktur, dll. untuk membuat perangkat lunak. [22].

Berdasarkan pengertian di atas, dapat disimpulkan bahwa *Application Programming Interface* (API) adalah integrasi dari bagian sistem aplikasi yang terdiri dari elemen fungsi, interface, kelas, struktur, dan lain sebagainya.

### **2.1.11 Web Service**

*Web Service* adalah komunikasi yang didefinisikan antara sistem komputer yang berbeda. Tanpa *web service*, komunikasi *peer-to-peer custom* menjadi kompleks [23].

*Web service* merupakan kumpulan suatu layanan berbasis web dengan menggunakan jaringan protokol HTTP, layanan tersebut dapat diakses dan dimanfaatkan pengguna bahasa pemrograman, arsitektur dan sistem operasi yang berbeda (*interoperability*) [24].

Berdasarkan definisi di atas, dapat disimpulkan bahwa *web service* merupakan layanan yang berbasis web dengan menggunakan protokol HTTP yang memungkinkan komunikasi antara sistem komputer yang berbeda. Saat ini terdapat dua konsep yang sering digunakan dalam implementasi *web service* yaitu *Representational State Transfer* (REST) dan *Simple Object Access Protocol* (SOAP).

#### **2.1.10.1 REST**

Istilah *Representational State Transfer* (REST) diperkenalkan oleh Roy Fielding pada tahun 2000. Arsitektural REST adalah arsitektur client-server di mana klien mengirimkan permintaan ke server, dan server kemudian memproses permintaan dan mengembalikan respons [25]. REST dapat mengembalikan *response* dengan format *plain text*, JSON, XML, dsb. Protokol komunikasi yang umum digunakan dalam gaya arsitektur REST adalah *Hyper-Text Transfer Protocol* (HTTP). Berikut merupakan metode HTTP yang umumnya digunakan dalam arsitektur REST:

1. GET, untuk mengambil data
2. POST, untuk menambah data
3. PUT, untuk mengubah (*update*) data
4. DELETE, untuk menghapus data

#### **2.1.10.2 SOAP**

SOAP (Simple Object Access Protocol) adalah bahasa markup berbasis XML untuk bertukar pesan antar aplikasi. SOAP berfungsi sebagai amplop yang digunakan untuk bertukar objek data di jaringan [26].

Cara kerja SOAP ialah, aplikasi klien mengirim *request* berbentuk XML kepada *provider web service*. *Web service* menerima *request* tersebut, menjalankan *service*, kemudian mengirimkan *response* ke aplikasi klien

dalam bentuk XML. Baik request maupun response keduanya menggunakan protokol SOAP.

#### **2.1.12 Red Hat OpenShift**

OpenShift merupakan *platform container* yang dioptimalkan untuk aplikasi web, memungkinkan membangun, menguji, dan menggunakan aplikasi web tanpa menyediakan dan memelihara server khusus untuk setiap aplikasi [27]. OpenShift dikembangkan oleh perusahaan perangkat lunak Amerika Red Hat, yang merupakan anak perusahaan dari IBM. OpenShift berperan sebagai *cloud platform as a service (PaaS)* yang dibangun diatas sistem operasi Red Hat Enterprise Linux yang dikelola oleh Kubernetes. Dengan penggunaan *platform* Red Hat OpenShift dapat memangkas biaya pemeliharaan, biaya hardware, dan juga meningkatkan kemampuan untuk skalabilitas.

#### **2.1.13 Spring Boot framework**

*Framework* Spring Boot adalah framework open source berbasis java yang menyediakan infrastruktur yang komprehensif dalam mengembangkan aplikasi java dengan mudah dan cepat. Spring pertama kali ditulis dan dirilis oleh Rod Johnson dengan lisensi Apache 2.0 pada bulan Juni 2003.

Konsep yang digunakan pada *framework* Spring Boot ini adalah MVC (Model View Controller), MVC merupakan sebuah konsep yang paling populer untuk membangun sebuah aplikasi berbasis web, konsep pada MVC ini adalah memisahkan model (*database*) dan view (*user interface*) [6].

#### **2.1.14 JavaScript Object Notation (JSON)**

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah untuk dibaca dan ditulis serta mudah dikonversi oleh mesin [28]. JSON merupakan format teks yang tidak tergantung pada bahasa pemrograman apapun. Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data dengan format JSON. Oleh karena itu, JSON merupakan format pertukaran data yang ideal dan banyak digunakan.

```

{
  "book": [
    {
      "id": "01",
      "language": "Java",
      "edition": "third",
      "author": "Herbert Schildt"
    },
    {
      "id": "07",
      "language": "C++",
      "edition": "second",
      "author": "E. Balagurusamy"
    }
  ]
}

```

Gambar 2.1 Format Data JSON

### 2.1.15 *eXtensible Markup Language (XML)*

XML merupakan meta-language seperti tag HTML (Hypertext Markup Language) yang digunakan untuk mendeskripsikan data. XML didesain untuk mampu menyimpan data secara ringkas dan mudah diatur [29]. XML merupakan kelanjutan dari HTML (*HyperText Markup Language*) yang merupakan bahasa standar untuk melacak Internet.

```

<books>
  <book>
    <id>01</id>
    <language>Java</language>
    <edition>third</edition>
    <author>Herbert Schildt</author>
  </book>
  <book>
    <id>07</id>
    <language>C++</language>
    <edition>second</edition>
    <author>E. Balagurusamy</author>
  </book>
</books>

```

Gambar 2.2 Format Data XML

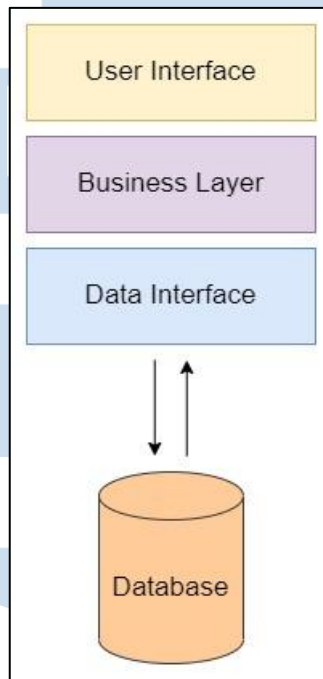
### 2.1.16 *Monolithic Architecture*

Arsitektur *monolithic* adalah sebuah aplikasi dengan basis kode tunggal yang mencakup beberapa *service* [30].



Arsitektur *monolithic* adalah suatu arsitektur yang menggambarkan sebuah aplikasi yang menjalankan semua logika dalam satu server aplikasi [31].

Berdasarkan deskripsi di atas, dapat disimpulkan bahwa arsitektur *monolithic* adalah suatu arsitektur dengan basis kode tunggal yang menjalankan semua logika dalam satu server aplikasi.



Gambar 2.3 Arsitektur *Monolithic*  
Sumber: [31]

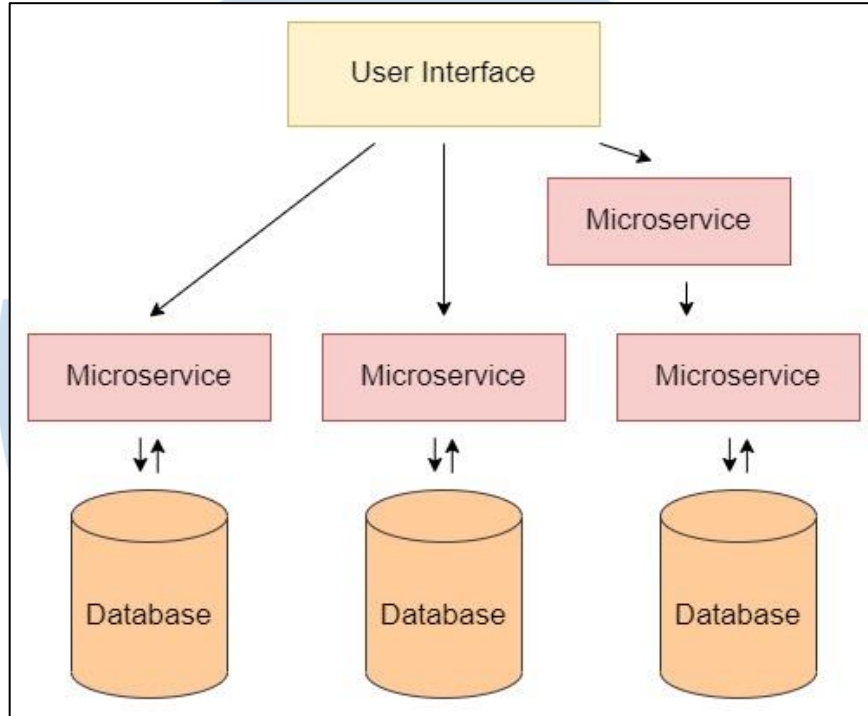
### 2.1.17 *Microservices Architectures*

Arsitektur *microservices* adalah sebuah arsitektur perangkat lunak berupa sistem besar yang dibagi menjadi satu atau lebih komponen kecil yang terfokus pada tugas tertentu dan bekerja sama untuk melakukan tugas dengan tingkat yang lebih tinggi [32].

Arsitektur *microservices* adalah arsitektur yang terdesentralisasi yang terdiri dari beberapa *microservice* serupa. Masing-masing *microservice* merupakan proses individu yang didedikasikan fungsi aplikasi tertentu [33].

Berdasarkan penjabaran di atas, dapat disimpulkan bahwa arsitektur *microservices* adalah arsitektur perangkat lunak yang terdiri dari beberapa

microservice yang berfokus untuk fungsi tertentu dari aplikasi, dan bekerjasama untuk mencapai tugas tingkat tinggi.



Gambar 2.4 Arsitektur *Microservices*  
Sumber: [33]

### 2.1.18 GraphQL

Pada tahun 2015, Facebook meluncurkan sebuah *query language* yang menjadi metode baru dalam mengatur pendistribusian data [7]. *Query language* tersebut dikenal dengan GraphQL. GraphQL memungkinkan *service API* untuk dapat mengembalikan *response API* dengan data sesuai dengan yang dibutuhkan.

## 2.2 Penelitian Terdahulu

Tabel 2.3 Penelitian Terdahulu

Penulis	Judul	Nama Jurnal	Rumusan Masalah	Kesimpulan
Agus Dudu Abdulah Rohmana, Husni Mubarak, Rohmat Gunawan	Pengukuran Kinerja <i>Stored Procedure</i> pada <i>Database</i> Relasional	Jurnal Siliwangi, Volume 5, No.2, 2019	Pada sistem basis data relasional, terdapat salah satu fitur yang bernama <i>stored procedure</i> .	Penelitian ini menghasilkan pembuktian bahwa penggunaan <i>stored procedure</i>

Penulis	Judul	Nama Jurnal	Rumusan Masalah	Kesimpulan
			Penelitian ini melakukan pengukuran waktu eksekusi <i>query</i> menggunakan <i>stored procedure</i> serta perbandingannya dengan <i>native query</i> .	memberikan performa kecepatan yang lebih baik dibandingkan <i>native query</i> .
Anshu Soni, Virender Ranga	<i>API Features Individualizing of Web Services: REST and SOAP</i>	<i>International Journal of Innovative Technology and Exploring Engineering (IJITEE)</i> ISSN: 2278-3075, Volume-8, Issue-9S, July 2019	<i>REST</i> dan <i>SOAP</i> merupakan gaya arsitektur yang sering digunakan dalam pengembangan API. Kedua arsitektur tersebut memiliki kelebihan dan kekurangannya masing-masing. Penelitian ini membandingkan kedua gaya arsitektur tersebut dari berbagai aspek.	Penelitian ini menghasilkan pembuktian bahwa gaya arsitektur <i>REST</i> lebih baik dibandingkan <i>SOAP</i> ketika digunakan dalam proyek yang berfokus pada kesederhanaan dan performa kecepatan.
Jansen Wiratama, Ririn Ikana Desanti	<i>Analysis and Design of Web-Based Information System for Church Congregations Case Study: Church BNKP Pewarta</i>	<i>Ultima Infosys: Jurnal Ilmu Sistem Informasi</i> , Vol 12, No 2, 2021	Selama pandemi COVID-19, kegiatan ibadah di gereja dibatasi. Kurangnya persiapan dan pendataan jemaat yang berkunjung untuk beribadah secara rutin setiap minggu meningkatkan risiko penularan. Hal tersebut dapat diminimalisir dengan pendataan kondisi kesehatan jemaat menggunakan aplikasi berbasis web.	Penelitian ini menggunakan metode <i>black box testing</i> dalam pengujian fitur perangkat lunak. <i>Black box testing</i> berfokus kepada <i>input</i> dan <i>output</i> aplikasi berdasarkan persyaratan dan spesifikasi perangkat lunak.

Penulis	Judul	Nama Jurnal	Rumusan Masalah	Kesimpulan
Anca-Raluca Breje, Robert Györödi, Cornelia Györödi, Doina Zmaranda, George Pecherle	<i>Comparative Study of Data Sending Methods for XML and JSON Models</i>	(IJACSA) <i>International Journal of Advanced Computer Science and Applications</i> , Vol. 9, No. 12, 2018	XML dan JSON merupakan format yang sering digunakan sebagai media pertukaran data. Penelitian ini memberikan komparasi terhadap kedua format pertukaran data tersebut untuk menentukan format yang tepat untuk setiap penggunaan.	Penelitian ini menghasilkan pembuktian format JSON lebih efektif dalam hal ukuran data dan kecepatan waktu respons <i>web API</i> . Penelitian ini membuktikan format data JSON memberikan performa yang lebih cepat 30 hingga 40% dibandingkan XML.
Konrad Gos, Wojciech Zabierowski	<i>The Comparison of Microservice and Monolithic Architecture.</i>	2020 IEEE XVith International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH), April 2020	<i>Microservice</i> dan <i>monolithic</i> merupakan dua arsitektur sistem perangkat lunak. Penelitian ini memberikan komparasi terhadap kedua arsitektur tersebut untuk mengetahui kelebihan dan kekurangan dari kedua arsitektur tersebut.	Penelitian ini menghasilkan pembuktian bahwa arsitektur <i>microservice</i> memiliki kecepatan dan keamanan yang lebih baik dibandingkan <i>monolithic</i> dalam menangani <i>request</i> HTTP dengan jumlah yang besar.

Penelitian terdahulu dengan judul Pengukuran Kinerja *Stored Procedure* pada *Database* Relasional memberikan masukan pada penelitian ini untuk menggunakan *stored procedure* dalam basis data karena memberikan performa kecepatan yang lebih baik dibandingkan *native query* [18]. Jurnal dengan judul *API Features Individualizing of Web Services: REST and SOAP* memberikan masukan pada penelitian ini untuk menggunakan arsitektur REST dibanding SOAP karena memberikan performa kecepatan yang lebih baik [34]. Jurnal berjudul *Analysis and Design of Web-Based Information System for Church Congregations Case Study: Church BNKP Pewarta* memberikan masukan pada penelitian ini untuk menggunakan metode *black box testing* dalam pengujian perangkat lunak karena

metode tersebut berfokus kepada *input* dan *output* berdasarkan persyaratan dan spesifikasi perangkat lunak [35]. Jurnal berjudul *Comparative Study of Data Sending Methods for XML and JSON Models* memberikan masukan pada penelitian ini untuk menggunakan format JSON sebagai format pertukaran data karena dapat memberikan ukuran data yang lebih kecil dan kecepatan yang lebih baik [36]. Jurnal berjudul *The Comparison of Microservice and Monolithic Architecture* memberikan masukan pada penelitian ini untuk menerapkan konsep arsitektur *microservices* karena dapat memberikan performa yang lebih baik dalam menangani *request* HTTP dengan jumlah yang besar [37].

Perbedaan penelitian ini dengan penelitian terdahulu yaitu penelitian ini menghasilkan produk akhir berupa *web service API* yang dinamis untuk pertukaran data yang dimiliki PT XYZ kepada anak perusahaannya. Komponen yang digunakan dalam penelitian merupakan hasil pertimbangan berdasarkan kesimpulan penelitian terdahulu.

