**BAB 2**

**LANDASAN TEORI**

**2.1 Toko Online**

Toko online adalah sebuah toko yang menjual barang-barang yang direalisasikan dalam tampilan sebuah *website* atau *mobile* yang dapat diakses saat terhubung dengan jaringan internet.

Menurut Ratri (2020:5), Toko online, seperti halnya toko fisik merupakan tempat bertransaksi dimana terjadi kegiatan pembelian dan penjualan. Yang diperjualbelikan dalam sebuah toko online bisa berupa produk atau jasa. Transaksi di toko online berlangsung melalui internet sehingga pembeli dan penjual tidak bertemu muka secara langsung. Komunikasi mata mengandalkan teknologi berupa fasilitas jejaring sosail maupun chatting.

Sedangkan menurut Hall (2021:9), evolusi e-commerce atau toko online membuat banyak konsep baru yang dapat memunculkan keahlian khusus yang diperlukan dalam menjalankan toko online yang sukses .

Menurut Hendriawan (2021:70), Toko online atau *e-commerce* merupakan salah satu konsep yang cukup berkembang dalam dunia internet. Pengguna sistem in sebenarnya dapat menguntungkan banyak pihak, baik pihak konsumen, produsen maupun penjual. Konsep *Online Shoping* menyediakan banyak kemudahan dan kelebihan jika dibandingkan dengan konsep belanja yang konvensional. Selain transaksi bisa jadi lebih cepat, di internet telah disediakan hampir semua barang yang biasanya dijual secara lengkap.

Dapat disimpulkan bahwa dengan toko online membantu dalam penjualan dan mendapatkan potensi pembeli jauh lebih luas serta meningkatkan penjualan pada toko.

**2.2 Framework**

Menurut Neuburg (2021:25), “Framework merupakan blok penyusan yang

digunakan untuk membuat antarmuka aplikasi.”

Menurut Laurina (2020:10), Framework merupakan kumpulan script (terutama function dan class) yang bisa membantu programmer atau development untuk mengatasi masalah-masalah dalam pemrograman, seperti variable, koneksi ke database, pemanggilan, file, dan lain-lain, sehingga pekerjaan developer lebih fokus dan lebih cepat dalam membangun aplikasi. Dalam pengaplikasian terhadap PHP, programmer cukup menempatkan library di dalam framework menggunakan fungsi-fungsi yang telah ditentukan sesuai ketentuan masing-masing. Aplikasi akan tersusun secara sistematis dan rapi karena framework telah menggunakan pattern stkitart, misalnya menggunakan framework yang terkenal yaitu Model View Controller (MVC).

Menurut Pressman (2020:293), “Framework merupakan arsitektur yang dapat digunakan kembali yang berfungsi sebagai fondasi dari pola desain yang dapat diterapkan yang menyediakan struktur dan perilaku generic.”

Jadi, Framework adalah sebuah kumpalan-kumpalan potongan program yang dipadukan menjadi satu kerja yang dapat digunakan untuk membatu dalam pembuatan sebuah aplikasi secara cepat.

**2.3 Framework Flutter**

Framework Flutter adalah kerangka kerja yang membangun user interface dengan menggunakan bahasa Dart untuk pengembangan lintas platform aplikasi seluler, web, dan desktop yang memakai satu basis kode. Framework Flutter berfungsi untuk membangun aplikasi yang lebih modern, dan cepat. Semua fitur yang dibutuhkan oleh programmer dalam membangun aplikasi disediakan oleh widget yang ada didalam Framework Flutter yang memfasilitasi proses pengembangan dengan peneysuaian penuh menghasilkan rendering yang sangat cepat dengan desain seluler yang ekspresif dan fleksibel.

Menurut Thornton (2020:93), “Flutter adalah kerangka kerja SDK/UI seluler Google yang memungkinkan pengembang membuat aplikasi untuk perangkat Android dan iOS dengan kode yang berfungsi di kedua platform dalam satu basis kode.”

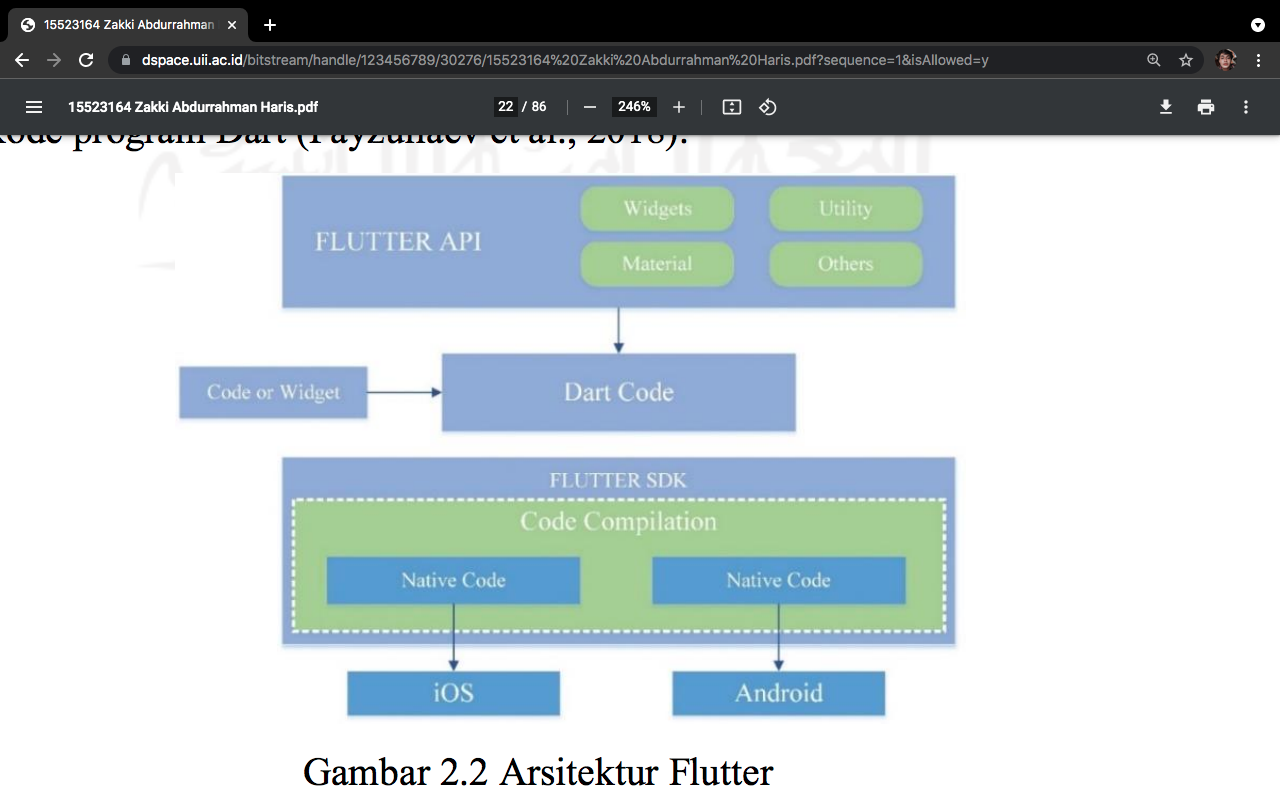
Menurut Ristimaki (2021:30), Framework Flutter juga memiliki fungsi utama yaitu Hot Reload yang merupakan alat pengembang khusus Framework Flutter yang merupakan fitur langka dari sekian banyaknya Framework. Hot Reload ini berfungsi membantu mengembangkan dan mengkompilasi ulang kode dengan cepat dalam pengembangan.

Menurut Tran (2020:2), “Flutter adalah toolkit antarmuka pengguna (UI) dengan Software Development Kit (SDK) yang dibuat untuk membangun aplikasi berkinerja asli yang dapat di perangkat iOS dan Andoid serta untuk aplikasi web dan desktop.”

Menurut Ernawati (2021:180), Flutter merupakan perangkat *UI* seluler dari Google untuk membuat antar muka yang cantik untuk aplikasi mobile yaitu Android dan iOS. Eksperimen yang dilakukan menggunakan flutter sangat cepat dan mudah seperti membuat *UI*, menambahkan fitur-fitur yang diperlukan dan memperbaiki *bug*.

**2.3.1 Arsitektur Flutter**

Menurut Haris (2021:7), Flutter menjalankan sebagian besar kerangka kerja dan kode aplikasi di dalam Dart virtual machine yang ringan. Kerangka kerja Flutter dibangun menggunakan Dart sedangakan rendering diimplementasikan dalam C++. Kode sumber dikompilasikan ke kode native menggunakan fitur kompilasi AoT (Ahead of Time) Dart seperti pada Gambar 2.2. Dalam Android kode program C/C++ dikompilasi oleh AoT ke dalam kode native dengan NDK Android, sedangakan iOS akan dikompilasi dengan LLVM (Low Level Virtual Machine), dan juga semua kode program Dart. Flutter memiliki arsitektur tersendiri, berikut adalah gambar arsitektur dari Framework Flutter.



**Sumber** : Haris, 2021

**Gambar 2.1** Arsitektur Framework Flutter

Komponen utama Flutter meliputi ((Thornton, 2020:100) :

1. Dart Platform

Fluter ditulis dalam Dart dan memanfaatkan fitur-fitur canggih dengan bahasa secara ekstensif. Flutter berjalan di mesin virtual Dart di Windows, macOS, dan Linux, yang memiliki mesin eksekusi tepat waktu. Flutter memanfaatkan kompilasi just-in-time (JIT) saat membangun dan men-debug aplikasi. Ini memberi pengembang kesempatan untuk "memuat ulang panas", memungkinkan perubahan pada file sumber untuk disuntikkan ke dalam aplikasi langsung. Flutter menambahkan dukungan untuk hot reload stateful, yang berarti bahwa perubahan pada kode sumber tercermin dalam aplikasi yang sedang berjalan tanpa memerlukan restart atau kehilangan status di sebagian besar keadaan. Versi rilis aplikasi Flutter untuk Android dan iOS dikompilasi dengan kompilasi sebelumnya (AOT) untuk meningkatkan efisiensi.

1. Flutter Engine

Flutter Engine adalah runtime portabel yang dapat digunakan untuk menghosting aplikasi Flutter. Ini termasuk animasi dan grafik, file dan jaringan I/O, dukungan aksesibilitas, arsitektur plugin, dan runtime Dart dan mengkompilasi toolchain, antara lain. Sebagian besar pengembang berinteraksi dengan Flutter melalui Flutter Framework, yang menyediakan kerangka kerja reaktif dan serangkaian widget platform, tata letak, dan fondasi.

1. Foundation Library

Kelas dan fungsi utilitas level terendah yang digunakan oleh semua level lain dari framework Flutter ditentukan di library ini. Pustaka Foundation, yang ditulis dalam Dart, menyediakan kelas dan fungsi dasar yang digunakan untuk membuat aplikasi menggunakan Flutter, seperti API untuk berkomunikasi dengan mesin.

1. Design Spesific Widgets

Framework Flutter memiliki dua set widget yang mematuhi gaya desain yang berbeda: widget Desain Material, yang menggunakan bahasa desain Desain Material Google, dan widget Cupertino, yang menggunakan prinsip-prinsip antarmuka manusia iOS Apple. Berikut ini adalah contoh widget Flutter yang tersedia :

1. *AlertDialog* : Alerts adalah interupsi singkat yang membutuhkan pengakuan dan menginformasikan pengguna tentang suatu masalah. Komponen ini diimplementasikan oleh widget AlertDialog.

2. *BottomNavigationBar* : Menggunakan bilah navigasi bawah, Anda dapat dengan cepat menjelajahi dan berpindah di antara tampilan tingkat atas. Komponen ini diimplementasikan oleh widget BottomNavigationBar.

3. *CheckBoxes* : memungkinkan pengguna untuk memilih dari kumpulan banyak kemungkinan. Komponen ini diimplementasikan oleh widget Kotak Centang.

4. Drawer: Panel Desain yang meluncur dari tepi Scaffold untuk menampilkan tautan navigasi dalam aplikasi.

5. *Expanded* : Ini adalah widget yang memperluas Baris, Kolom, atau anak Flex.

6. *Form* : Wadah yang dapat digunakan untuk mengabungkan beberapa widget

7. *Widget Gridview* : Daftar grid terdiri dari pola sel berulang yang disusun secara vertikal dan horizontal. Komponen ini diimplementasikan oleh widget GridView.

8. *IconButton* : Tombol ikon adalah gambar yang dicetak pada widget Material dan terisi warna saat disentuh (ink).

9. *Image Widget* : Widget yang menampilkan gambar dikenal sebagai widget gambar.

10.*ListBody* : Widget yang mengatur anak-anaknya dalam urutan tertentu di sepanjang satu sumbu, memaksa mereka ke dimensi yang sama dengan induknya di sumbu yang berlawanan.

11.*MaterialApp* : Ini adalah widget kenyamanan yang merangkum berbagai widget yang biasanya digunakan dalam aplikasi Desain Material

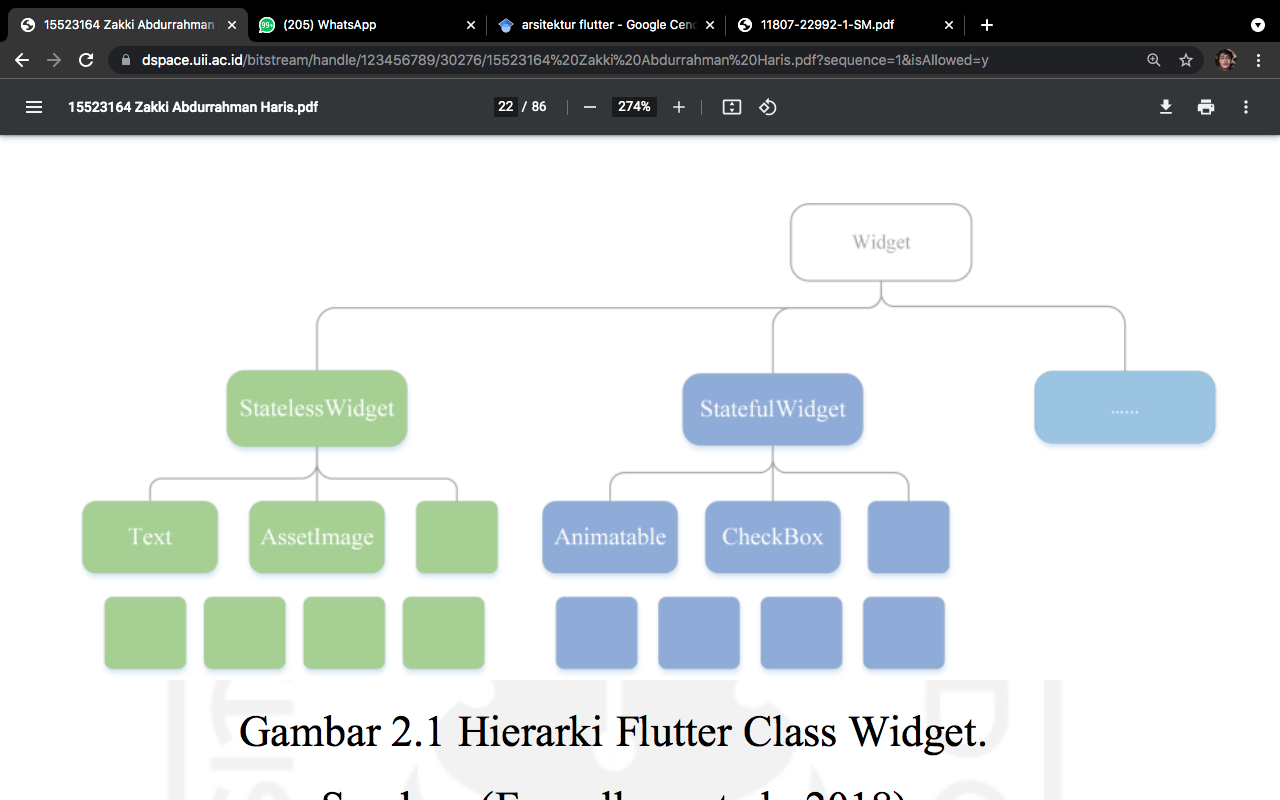
**2.3.2 Widget**

Menurut Haris (2021:6), Widget merupakan komponen penting dalam kerangka kerja Flutter untuk membangun tampilan. Flutter memiliki model objek yang konsisten dan terpadu, sehingga setiap widget memiliki deklarasi dalam pembangunan tampilan yang tidak dapat diubah. Flutter memiliki struktur widget yang dapat disesuaikan dan diperluas, serta yang paling utama Flutter tidak menggunakan komponen atau widget OEM tetapi menyediakan widget sendiri.”

Menurut Tran (2020:5), Widget meruapakan pendeskripsian elemen tertentu secara visual di UI berdasarkan konfigurasi dan statusnya. Disisi pengembngannya, widget menyediakan beberapa set antarmuka pemograman aplikasi (dikenal dengan API) dengan cara yang ekspresif bagi pengembangan untuk memvisualisasikan dan mengkonfigurasi widget dengan cara yang diharapkan.

**2.3.3 *Lifecycle Widget***

Menurut Haris (2021:7), Ada dua jenis widget utama yang ada dalam Flutter yaitu stateless widget dan stateful widget. Statefull widget memiliki state yang dapat berubah. State adalah informasi yang dapat dibaca secara bersama ketika sebuah widget dibuat dan dapat berubah selama pemakaian widget Sedangkan stateless widget tidak membutuhkan state, keduanya berada dalam level hierarki yang sama seperti terlihat pada gambar 2.2.

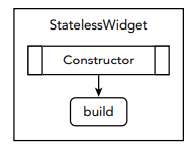


**Sumber** : Haris, 2021

**Gambar 2.2** Hierarki Flutter Class Widget

**2.3.4 Lifecycle *StatelessWidget***

Menurut Napoli (2020:6), Sebuah *StatelessWidget* dibangun berdasarkan konfigurasinya sendiri dan tidak berubah secara dinamsi. Misalnya, layar menampilkan gambar dengan deskripsi dan tidak akan berubah. Widget *stateless* dideklarasikan dengan satu kelas. Method *build*(bagian UI) dari widget *stateless* yang dipanggil dari 3 skenario yang berbeda. Itu bisa dipanggil saat pertama kali widget dibuat, saat parents widget berubah, dan saat Widget yang diwariskan telah berubah.



**Sumber** : Napoli, 2020

**Gambar 2.3** *Stateless Widget Lifecycle*

Berikut Contoh Struktur dasar kode *Stateless Widget*

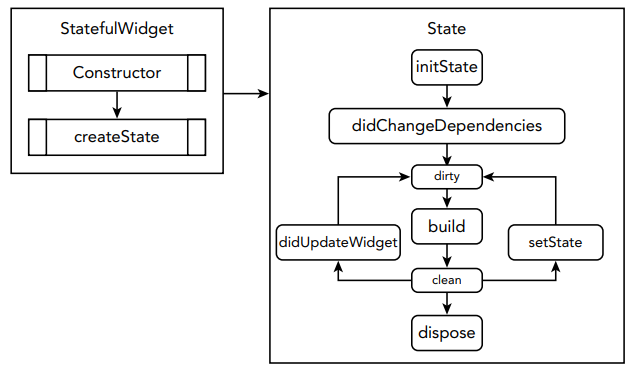
class JournalList extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Container();

}}



**Sumber** : Napoli, 2020

**Gambar 2.4** *Stateful Widget Lifecycle*

Menurut (Napoli, 2020:7), Pengembang dapat melakukan ovveride pada bagian berbeda dari *StatefulWidget* untuk menyesuaikan dan memanipulasi data di berbagai titik Lifecycle Widget. Gambar 2.4 menunjukkan beberapa pergantian utama widget stateful, dan kebanyakan pengembang akan menggunakan method *initState(), didChangeDependencies(),* dan *dispose().* Pengembang akan menggunakan method *build()* sepanjang waktu saat membangun user interface.

**2.3.5 Lifecycle StatefulWidget**

Menurut Napoli (2020:9), Sebuah *StatefullWidget* dibangun berdasarkan konfigurasinya sendiri tetapi dapat berubah secara dinamis. Misalnya, layar menampilkan ikon dengan deskripsi, tetapi nilainya dapat berubah berdasarkan interaksi pengguna, seperti memilih ikon atau deksripsi yang berbeda. Jenis widget ini memiliki state yang dapat berubah – ubah siring waktu. Widget *statefull* dideklarasikan dengan dua class, yaitu *StatefullWidget class* dan *State class. Class StatefullWidget* dibangun kembali ketika konfigurasi widget berubah, tetapi state class dapat bertahan, meningkatkan kinerja. Misalnya, ketika state berubah, widget dibangun kembali. Jika *StatefullWidget*  dihapus dari *tree* dan kemudian dimasukkan kembali ke dalam tree. State baru akan dibuat. Memanggil method *setState()* untuk memberi tahu framework bahwa objek ini memiliki perubahan, dan widget *build* method memanggil ini (dijadwalkan). Pengembang bisa mengatur nilai state baru di dalam *setState().* Contoh Struktur dasar *StatefullWidget.*

class JournalEdit extends StatefulWidget {

@override

\_ JournalEditState createState() => \_ JournalEditState();

}

class \_JournalEditState extends State<JournalEdit> {

@override

Widget build(BuildContext context) {

return Container();

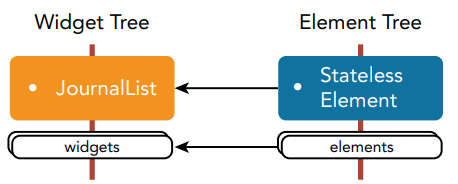
}}

Menurut Sinha (2020:807), Statusnya bisa berubah dan berubah dengan siklus hidup widget. Setiap kali status berubah, Flutter menggambar ulang widget. Contoh berikut akan membuat abstraksi lebih jelas dari sebelumnya. Widget TexField secara default adalah widget Stateful. Ini memiliki properti 'onChanged'; parameter bernama menunjuk ke fungsi anonim yang meneruskan nilai string yang diketik pengguna di layar ponsel. Setiap kali beberapa teks sedang diketik melalui 'TextField', teks tersebut tercermin di layar.

**2.3.6 Widget and Elements**

Menurut Napoli (2020:9), Widget adalah bagian konfigurasi UI dan bersama-sama mereka membuat widget tree, Flutter menggunakan widget sebagai konfigurasi untuk membangun setiap elemen,yang berarti elemen tersebut adalah widget yang dipasang (di-render) di layar. Elemen yang dipasang di layar membuat elemen tree.

Menurut Sinha (2020:921), Aturan pertama menggunakan widget adalah kita harus spesifik tentang jenis model yang ingin kita akses. Misalkan, kita menginginkan 'FirstModel', jadi kita menulis Consumer<FirstModel>. Jika tipe generik <FirstModel> tidak ditentukan, paket Penyedia tidak dapat membantu kami. Paket Penyedia didasarkan pada jenis. Oleh karena itu, kita harus menyebutkan jenisnya.Aturan terpenting kedua adalah kita harus menyediakan argumen 'pembangun' dari widget Konsumen. Ini adalah satu-satunya argumen yang diperlukan dari widget Konsumen. Setiap kali di kelas model ChangeNotifier berubah, argumen builder dipanggil. Mari kita coba memahami apa yang terjadi. Setiap kali ChangeNotifier berubah, metode notifyListeners() dipanggil, dan pada saat yang sama, semua metode builder dari semua widget Konsumen yang sesuai dipanggil. Pembangun'disebut dengan tiga argumen, yang pertama cukup akrab, 'konteks'; kami mendapatkannya di setiap metode build. Argumen 'nilai' kedua adalah turunan dari ChangeNotifier. Dengan menggunakan instance itu, kita dapat menentukan status aplikasi, dan bersamaan dengan itu, kita juga dapat menggunakan data dalam model sesuai dengan kebutuhan kita.



**Sumber** : Napoli, 2020

**Gambar 2.5** Widgets Tree dan Elemens Tree

Gambar 2.5 merupakan Widgets tree yang memiliki konfigurasi widgets dan Elements tree yang mewakili widgets yang dirender di layar. Saat aplikasi dijalankan fungsi main() memanggil metode runApp(), biasanya mengambil sebuah State-lessWidget sebagai argument, dan terpasang sebagai element root untuk aplikasi. Framework Flutter memproses semua widgets dan setiap elements yang terpasang.

Berikut ini adalah contoh kode saat memulai aplikasi Flutter, dan method runApp() membungus myApp StatelessWidget , artinya aplikasi utama itu sendiri adalah widget. Seperti yang terlihat, hampir semua hal di Flutter adalah widget.

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter App',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: MyHomePage(title: 'Home'),

);

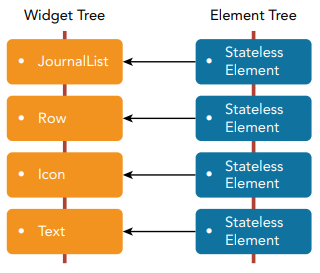
}}

Menurut Napoli (2020:9), Elemen memiliki referensi ke widget dan bertanggung jawab untuk membandingkan perbedaan widget. Jika widget bertanggung jawab untuk membuat chill widget, maka elemen dibuat untuk setiap chill widget. Ketika Pengembang menggunakan BuildContext, mereka adalah Elements objek. Untuk mencegah manipulasi langsung dari *elemen* objek, Interface *BuildContext* digunakan sebagai gantinya. Framework flutter menggunakan *BuildContext* objek untuk mencegah memanipulasi *elemen* objek. Dengan kata la, Pengembang akan menggunakan widget untuk membuat tata letak UI , tetapi ada baiknya mengetahui bagaimana Framework Flutter dirancang dan cara kerjanya dibalik layer.

Menurut Sinha (2020:207), Widget ada dua jenis. Satu terlihat, dan yang lainnya tidak terlihat. Di Flutter, kelas Text() meneruskan data String, yang terlihat. Untuk melewati data String, meskipun tidak terlihat bagus, namun terlihat. Dengan cara yang sama, menggunakan berbagai jenis tombol, yang juga terlihat. Dalam beberapa menit akan menggunakan Widget, 'RaisedButton()'; yang juga akan sangat terlihat. Kebetulan ada Widget yang tidak terlihat, seperti Column(), Row(), dan masih banyak lagi Widget lainnya yang akan digunakan di kemudian hari ketika akan membangun aplikasi. Dengan mengacu pada Widget yang tidak terlihat, Widget yang tidak terlihat ini sebenarnya membantu Widget yang terlihat untuk menggambar setiap piksel di layar ponsel. Oleh karena itu, mereka saling bergantung satu sama lain.

**2.3.7 StatelessWidget dan Elemen Tree**

Menurut (Napoli, 2020:9), sebuah *stateless* widget memiliki konfigurasi untuk membuat *statless* elemen. Setiap widget *stateless* memiliki elemen *stateless* yang sesuai. Framework flutter memanggil method *createElement, stateless* elemen dibuat dan dipasang ke dalam tree. Dengan kata lain , framework flutter membuat permintaan dari widget untuk membuat elemen dan kemudian memasang(menambahkan) elemen ke dalam tree. Proses ini menghasilkan dua tree: widget tree dan elemen tree.



**Sumber** : Napoli, 2020

**Gambar 2.6** *StatelessWidget Widget Tree* dan *Elemen Tree*

Gambar 2.6 menunjukan *StatelessWidget JournalList* yang memiliki Widget *Row, Icon* dan *Text* yang mewakili widget Tree. Framework flutter meminta setiap widget untuk membuat elemen, dan setiap elemen memiliki referensi kembali ke widget. Proses ini terjadi untuk setiap widget di bawah widget tree dan membuat elemen tree. Widget berisi instruksi untuk membangun elemen yang terpasang dilayar. Berikut contoh Syntax *StatelessWidget* dari gambar diatas (Napoli, 2020:10) :

class JournalList extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Row(

children: <Widget>[

Icon(),

Text(),

],

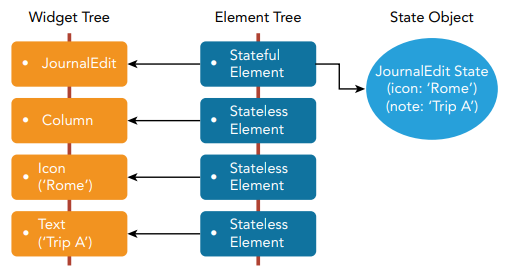
);

}

}

**2.3.8 StatefulWidget dan Elemen Tree**

Menurut Napoli (2020:10), *Stateful* widget memiliki konfigurasi untuk memuat elemen *stateful*. Framework flutter memanggil method *createElement* untuk membuat statefull element, dan sebuah *stateful* elemen akan dipasang ke element tree. Karena ini adalah *stateful* widget, *statefull* elemen akan meminta *request* widget membuat objek state dengan memanggil class *Statefull-widget* lalu method *createState.* Sekarang sebuah *stateful* elemen memiliki referensi ke objek state dan widget di lokasi yang akan diberikan di element tree. *Stateful* elemen memanggil objek state widget *build* method untuk mengecek child widget, dan setiap child widget membuat elemnt sendiri dan dipasangkan ke element tree. Proses ini mengasilkan dua tree: Widget tree dan element tree.

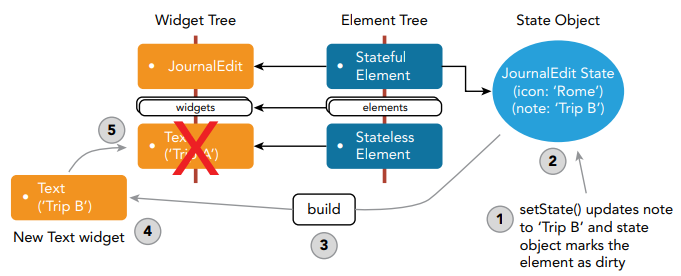


Sumber : Napoli, 2020

Gambar 2.7 Widget tree , Element Tree dan State Object

Menurut Napoli (2020:12), “Gambar 2.7 menunjukkan Widget tree , Element Tree dan State object. Element *stateful* memiliki referensi ke widget *Stateful* dan State Object. Untuk memperbaharui tampilan UI dengan data baru, pengembang bisa memanggil method *setState()*. Untuk data baru (Properties/Variables) value, memanggil method *setState()* untuk memperbaharui state object, dan state object menandai element sudah berubah dan menyebabkan UI diperbaharui.”

Sebagai Contoh, terdapat *StatefulWidget* dengan nama class *JournalEntry ,* dan didalam Object State class memanggil method *setState()* untuk mengubah Nilai Widget *Text* dari *‘Trip A’* menjadi *‘Trip B’* dengan melakukan setting nilai di variable *note* menjadi *‘Trip B’* maka widget *Text* baru akan dibuat dengan value *‘Trip B’,* dan widget *text* yang lama yang berisi value *‘Trip A’* akan dihapus pada gambar 2.7.”



Sumber : Napoli, 2020

Gambar 2.8 Proses Update State

Berikut contoh Syntax Proses *Stateful* Widget yang melakukan perubahan State:

class JournalEdit extends StatefulWidget {

@override

\_JournalEditState createState() => \_JournalEditState();

}

class \_JournalEditState extends State<JournalEdit> {

String note = 'Trip A';

void \_onPressed() {

setState(() {

note = 'Trip B';

});

}

@override

Widget build(BuildContext context) {

return Column(

children: <Widget>[

Icon(),

Text('$note'),

FlatButton(

onPressed: \_onPressed,

),

],

);

**2.3.9 Kinerja Flutter**

Menurut Tran (2020:3-4), Kinerja aplikasi yang dibangun dengan Flutter mengoptimalisasikan pengembangan yang baik dengan kecepatan yang mencapai 60 frame per detik (fps) atau 120 fps, selain itu aplikasi yang menggunakan Flutter memiliki UI yang mulus dan kecepatan yang stabil dengan tidak memiliki animasi yang junk.

Menurut Tashildar (2020:1), Kompilasi AOT (Kompilasi Ahead of Time) mengkompilasi bahasa pemograman tingkat tinggi seperti C atau C++, selain itu fitur utama flutter adalah Hot Reload yang membantu developer dalam bereksperimen dengan mudah untuk membangun UI, menambah fitur, dan memperbaiki bug dengan cepat.

Menurut Pamungkas (2020:2022), “Dalam perancangan aplikasi yang menuggunakan Framework Flutter, Flutter memiliki keunggulan dengan penggunaan memori yang lebih rendah dengan penggunaan CPU juga yang lebih rendah.”

Menurut Windmill (2020:29), “Bagi pengembang, hot reload berarti dapat mengembangkan dan mengkompilasi ulang kode dengan cepat dalam pengembangan tanpa mengorbankan kinerja non-asli dalam produksi”.

**2.3.10 *Foundation Library Flutter***

Menurut Thornton (2021:103), Kelas dan fungsi utilitas level terendah yang digunakan oleh semua level lain dari framework Flutter ditentukan di library ini. Pustaka Foundation, yang ditulis dalam Dart, menyediakan kelas dan fungsi dasar yang digunakan untuk membuat aplikasi menggunakan Flutter, seperti API untuk berkomunikasi dengan mesin. Itu dapat dengan mudah diakses di GitHub. Framework Flutter memiliki dua set widget yang mematuhi gaya desain yang berbeda: *widget* Desain Material, yang menggunakan bahasa desain Desain Material Google, dan widget Cupertino, yang menggunakan

Prinsip-prinsip user interface iOS yang tersedia di dalam *widget* Flutter (Thornton, 2021:103) :

a. *Alert Dialog* : *Alerts* adalah interupsi singkat yang membutuhkan pengakuan dan menginformasikan pengguna tentang suatu masalah. Komponen ini diimplementasikan oleh *widget* *Alert Dialog*.

b. *Bottom Navigation Bar* : Menggunakan bilah navigasi bawah, Anda dapat dengan cepat menjelajahi dan berpindah di antara tampilan tingkat atas. Komponen ini diimplementasikan oleh widget BottomNavigationBar.

Kotak centang memungkinkan pengguna untuk memilih dari kumpulan banyak kemungkinan. Komponen ini diimplementasikan oleh widget Kotak Centang.

c. *Drawer* : Panel Desain yang meluncur dari tepi Scaffold untuk menampilkan tautan navigasi dalam aplikasi.

d. *Expanded*: Ini adalah *widget* yang memperluas Baris, Kolom, atau anak Flex.

e. *Form*  : Wadah yang dapat digunakan untuk menggabungkan beberapa widget bidang fomulir misalnya widget *TextField*.

f. *Widget Gridview*: Sebuah daftar grid terdiri dari pola berulang sel yang diatur secara vertikal dan horizontal. Komponen ini diimplementasikan oleh widget GridView.

g. *IconButton*: Tombol ikon adalah gambar yang dicetak pada widget Material dan terisi warna saat disentuh (tinta).

h. Widget gambar: Widget yang menampilkan gambar dikenal sebagai widget gambar.

i. *List Body*: Sebuah widget yang mengatur anak-anaknya dalam urutan tertentu di sepanjang satu sumbu, memaksa mereka ke dimensi yang sama dengan induk pada sumbu yang berlawanan.

j. *Material App*: Ini adalah widget kenyamanan yang merangkum berbagai widget yang biasanya digunakan dalam aplikasi Desain Material.

Menurut Sinha (2020:406), Widget Flutter memiliki dasar yang kuat alat Flutter hadir dengan ratusan widget canggih. Beberapa widget dasar yang paling umum digunakan yang telah kita lihat. Widget teks membantu kita membuat banyak teks bergaya dalam aplikasi dan akan melihat implementasinya seiring kemajuan. Kemudian datang Baris dan Kolom. Keduanya memiliki fleksibilitas ekstrem untuk membuat tata letak Flutter. Saat ingin membangun tata letak dalam arah horizontal dan linier dapat menggunakan Row. Jika membutuhkan tata letak yang sejajar secara vertikal, dapat menggunakan widget Kolom. Widget kontainer adalah widget lain yang sangat penting yang akan digunakan dalam aplikasi. Ini menciptakan elemen visual persegi panjang.

**2.3.11 Debug Flutter**

Flutter menyediakan berbagai perangkat dan kemampuan untuk membantu proses debug aplikasi. Peralatan dan fasilitas yang tercantum di bawah ini (Thornton, 2021:357) :

a. *DevTools*: Alat masuk untuk men-debug aplikasi mungkin DevTools. Ini adalah seperangkat alat kinerja dan profil berbasis browser.

b. *Logging*: Inspektur widget tampilan Logging di DevTools, serta secara tidak langsung dari Android Studio dan IntelliJ IDEA. Representasi visual pohon widget dapat diperiksa menggunakan inspektur.

c. *Debug Flags*: Debug Flags memberi kita pilihan flag dan fungsi debug untuk membantu kita men-debug program kita pada saat yang berbeda. Anda harus mengkompilasi dalam mode debug, untuk menggunakan fungsi ini.

**2.4. Bahasa Pemograman Dart**

Menurut Haris (2021:6), Flutter menggunakan bahasa pemporgraman Dart sebagai bahasa utamanya. Dart adalah bahasa pemprograman yang dikembangkan dan dikelola oleh google pada tahun 2011. Google menggunakan dart untuk membuat aplikasi berkualitas tinggi untuk iOS, Android, dan web dengan fitur yang ditujukan untuk pengembangan sisi klien.

Dart merupakan bahasa pemograman yang dikembangkan oleh pihak google, dan kemudian dijadikan bahasa pemograman resmi untuk flutter, dart kemudian dikembangkan menjadi salah salah satu bahasa pemograman yang ada saat ini, dart adalah bahasa pemograman yang dioptimalkan untuk pengembangan *UI* secara cepat dan produktif pada banyak platform.

Menurut Suryana (2021:1), *Dart* adalah merupakan bahasa pemrograman yang dikembangkan oleh google untuk kebutuhan dalam membuat aplikasi Android atau *mobile, front-end, web, IoT, back-end* (CLI), dan *Game*. *Dart* menerapkan konsep pemrograman berorientasi objek (OOP) dimana struktur kode berada dalam class yang didalamnya berisi method maupun variable. Dart sendiri menggunakan *C-Style syntax* sehingga mekanisme dart mirip dengan bahasa pemrograman C, *java, javascript*, dan *Swift*.

**2.4.1 Cara Kerja Dart**

Tipe bahasa pemrograman dart berupa *Statically Typed* dimana tipe dari variabel diketahui ketika proses compile, ketika membangun aplikasi *mobile*, *dart* dicompile kedalam bahasa *native* agar dapat berjalan di mobile, lalu di transpiler ke *javascript* agar dapat berjalan di browser.

Transpiler bisa diartikan sebagai *source-to-source compiler* , yaitu sebuah compiler yang mengambil source code suatu program yang di tulis menggunakan suatu bahasa pemrograman sebagai input dan menghasilkan *source code* yang equivalent dalam bahasa pemrograman lainnya.

MenurutSuryana (2021:1), “*Dart* diterapkan pada *framework* flutter yang merupakan *framework* untuk membuat aplikasi *mobile* yang saat ini ramai diperbincangkan. Flutter adalah sebuah *Framework* yang dirancang khusus untuk membangun antarmuka (*UI*) aplikasi *mobile* yang lebih ekspresif dan menarik.”

**2.5 Android Studio**

Menurut Burton (2016:6), “Android menawarkan pendekatan yang menyeluruh dalam pengembangan aplikasi. Artinya satu aplikasi Android yang dibangun dapat berjalan di berbagai perangkat yang memungkinkan sistem operasi Android baik itu di smartphone, tablet, dan perangkat lainnya.”

Menurut Zografou (2021:18), Java Development Kit (JDK) menawarkan alat yang digunakan oleh IDE terintegrasi serta berbagai paket pengembangan perangkat lunak SDK, seperti kompiler Java. JDK memungkinkan program Java untuk berjalan normal pada sistem, karena termasuk lingkungan runtime Java (Java Lingkungan Waktu Proses - JRE). JRE atau lingkungan eksekusi pribadi termasuk mesin virtual (Java Virtual Machine) serta perpustakaan kelas yang sangat berguna bagi pengembang.

Menurut Vakaliuk (2021:122), SDK / *Software Development Kit* adalah seperangkat alat pengembangan yang memungkinkan para professional perangkat lunak untuk membuat aplikasi dengan paket perangkat lunak tertentu seperti alat pengembangan dasar perangkat lunak, perangkat keras platform, sistem computer, konsol game, sistem operasi, dan platform lainnya. SDK memanfaatkan setiap platform dan mengurangi waktu integrasi.

Menurut Burton (2016:6), “Untuk membangun suatu aplikasi Android, Terdapat beberapa peralatan yang perlu dipersiapkan agar dapat membuat suatu project pada IDE Android studio yaitu Java JDK atau Java JRE, Android SDK, Android Studio ADT dan Android Virtual Devices (AVD).”

**2.5.1 Komponen dan Arsitektur Sistem Operasi Android**

Didalam sebuah project Android terdapat komponen gradle dan modul aplikasi yang wajib ada berisi folder yaitu (Brunette, 2019:47) :

a. Manifest yang berisi file AndroidManifest.xml. File manifes menjelaskan informasi penting tentang aplikasi ke fitur build Android, sistem operasi Android. Manifest ini dapat berisikan script untuk permission. Salah satu contoh script permission di manifest adalah < uses – permission Android : name = “ Android. Permission.INTERNET”/> yang artinya aplikasi Android yang dibuat secara otomatis sudah memiliki permission-permission ini dan tidak akan menampilkan permintaan pengaksesan karena dianggap tidak membahayakan aplikasi.

1. Java yang berisi file kode java, di file java ini berisi kodingan untuk menjalankan xml yang sudahg dibuat. Contoh nya adalah namaVar= findviewbyid (R.id.idVar) yang berfungsi untuk mencari id yang akan diapnggil dan diberikan fungsi di java.
2. Res yang berisi sumua sumber daya non-kode, seperti layour XML, string UI dan gambar bitmap, yang dibagi ke dalam beberapa sub-direktorinya.

Sistem operasi Android memiliki lima (layer) yang merupakan sistem Android. Secara garis besar Arsitektur Android dapat diejaskan dan digambarkan sebagai berikut (Dimarzio, 2017:4):

1. Kernel Linux

Kernel Linux adalah kernel yang menjadi basis Android. Lapisan ini berisi semua driver perangkat tingkat rendah untuk berbagai komponen perangkat keras dari Android.

1. Libraries

Libraries ini berisi kode yang menyediakan fitur utama OS Android seperti perpustakaan SQLite yang menyediakan dukungan database sehingga aplikasi dapat menggunakannya untuk menyimppan data. Seperti juga Webkit Library yang menyediakan fungsional untuk penelusuran web.

1. Android Runtime

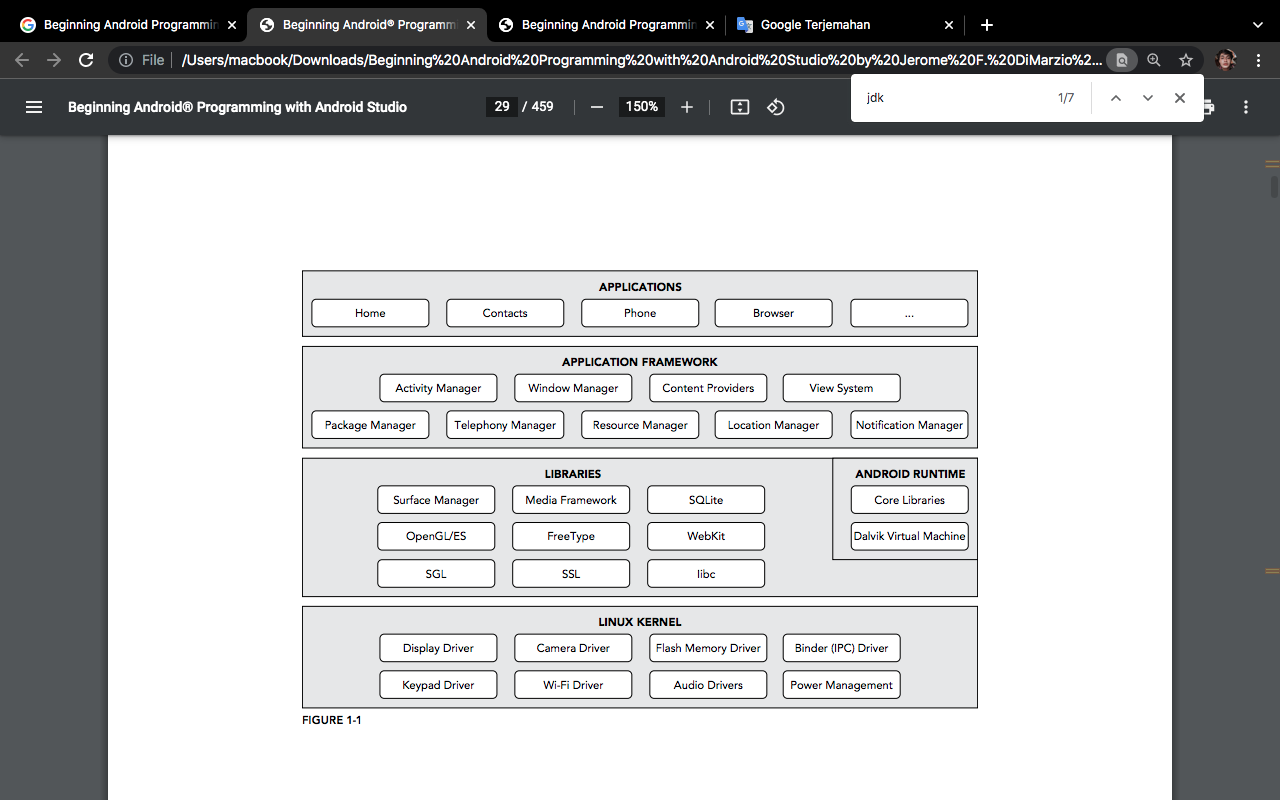
Android Runtime terletak di lapisan yang sama dengan perpustakaan dan menyediakan satu set pustaka inti yang memungkinkan pengembang untuk menulis aplikasi Android menggunakan Java bahasa pemrograman. Waktu proses Android juga mencakup mesin virtual Dalvik, yang memungkinkan setiap aplikasi Android berjalan dalam prosesnya sendiri, dengan instance-nya sendiri dari mesin virtual Dalvik. (Aplikasi Android dikompilasi ke dalam executable Dalvik). Dalvik adalah mesin virtual khusus yang dirancang khusus untuk Android dan dioptimalkan untuk perangkat seluler bertenaga baterai dengan memori dan daya CPU terbatas.

1. Application Framework

Application Framework memperlihatkan berbagai kemampuan dari OS Android kepada pengembang aplikasi sehingga mereka dapat menggunakannya dalam aplikasi.

1. Application

Appilcation pada lapisan atas ini adalah aplikasi yang dikirimkan bersama perangkat Android (seperti seperti Telepon, Kontak, Peramban, dan sebagainya), serta aplikasi yang Anda unduh dan instal dari Android Market. Aplikasi apa pun yang Anda tulis terletak di lapisan ini.



**Sumber** : Dimarzio, 2017

**Gambar 2.9** Arsitektur Android

**2.5.2 Keunggulan Android**

Beberapa keunggulan Android dari platform lain antara lain (Safaat, 2011:98):

1. Lengkap (*Complete Platform*)

Para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan *platform* Android. Android merupakan sistem operasi yang aman dan banyak menyediakan *tools* dalam membangun *software* dan untuk pengembangan aplikasi.

1. Terbuka (*Open Source Platform*)

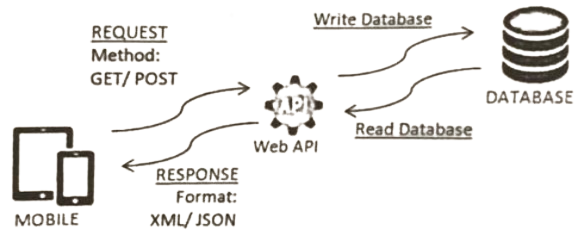
*Platform* Android disediakan melalui lisensi *open source*. Pengembang dapat dengan bebas untuk mengembangkan aplikasi. Android sendiri menggunakan *Linux Kernel* 2.6.

1. Bebas (*Free Platform*)

Android adalah *platform* atau aplikasi yang bebas untuk *developer*. Tidak ada lisensi untuk dikembangkan pada *platform* Android. Tidak ada biaya keanggotaan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Android dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

**2.5.3 Arsitektur Android dan Web Server**

Menurut Michale (2016:272), Untuk mengakses suatu database external perlu mempersiapkan suatu *Application Programming Interface*(API) yang dapat digunakan untuk menjembatani aplikasi Android dengan database server. API tersebut dapat dibuat berbasis web sehingga dapat diakses oleh aplikasi yang dibangun dan dapat diinstal di berbagai perangkat berbeda, juga menggunakan bahasa scripting PHP untuk membangun web API dan MySql sebagai database server yang diakses secara online.

**

Gambar 2.10 Arsitektur Framework untuk mengakses Database External

Menurut Sianipar (2015:36), “REST API mengirimkan perintah yang akan dikerjakan oleh server menggunakan metode – metode HTTP request mtheod yang disebut verb. Terdapat beberapa HTTP request method, yaitu GET, POST, PUT, DELETE. Namun yang umum digunakan dalam RESTful adalah GET dan POST.”

Pesan yang diterima dari server berupa kode HTTP berhasil atau gagal di dalam header dan isi pesan hasil pengolahan program itu sendiri. Berikut adalah kode – kode HTTP yang sering digunakan dalam penggunakan REST API (Sianipar, 2015) :

1. 200 OK Perintah yang dikirim ke server benar dan berhasil dijalankan.
2. 400 Bad Request Perintah yang dikirim ke server berisi isian yang salah
3. 401 Unauthorized Pengirim perintah mengirimkan kode kunci yang salah
4. 403 Forbidden Pengirim perintah tidak memiliki hak akses ke dalam resource yang dituju.
5. 404 Not Found Resource yang dituju tidak ditemukan dalam server
6. 429 Too Many Request pengirim perintah mengakses melebihi dari limit yang telah ditentukan dari batas waktu tertentu.
7. 500 Internal Server Error Server atau potongan program dalam resource mengakami kesalahan.

Menurut Sianipar (2015:76), “Salah satu pesan balikan / response yang dapat dan akan sering digunakan dalam sebuah REST API adalah JSON, karen JSON merupakan format pesan balik yang berukuran kecil yang mudah dibaca dan ditulis oleh manusial, dan juga mudah diatur.”

**2.6 Perancangan Antar Muka (Interface)**

Menurut Pressman (2020:175), Design interface menggambarkan abagaiman antarmuka perangkat lunak berkomunikasi dengan dirinya sendiri, dengan sistem yang bertinterporasi dengannya, dan dengan manusia yang mnggunakannya". Interface mengimplementasikan aliran informasi (misalnya data suatu control) dengan demikian data dan diagram alir kontrol memberikan informasi yang dibutuhkan bagian design interface.”

Sasaran-sasaran antarmuka pengguna untuk aplikasi-aplikasi web pada dasarnya (Pressman, 2020:56):

1. Menetapkan suatu jendela yang konsisten untuk meletakkan isi-isi dan fungsionalitas-fungsionalitas yang disediakan oleh antarmuka pengguna
2. Memandu para pengguna melalui serangkaian interaksi dengan aplikasi web yang dikembangkan
3. Mengorganisasi pilihan-pilihan navigasi dan isi-isi yang dapat dilihat oleh para pengguna.

Terdapat empat model yang berbeda pada saat sebuah interface manusia- komputer (human-computer interface) akan didesain, yaitu (Pressman, 2020:86):

1. Model desain dari keseluruhan sistem menggunakan data, arsitektur, interface, dan presentasi procedural dari perangkat lunak.
2. Model pemakai menggambarkan profil para pemakai akhir dari sistem.
3. Persepsi sistem (model pemakai) merupakan citra sistem yang ada dipakai seorang pemakai akhir.
4. Citra sistem merangkai manifestasi bagian luar dari sistem berbasis komputer (tampilan luar dan "rasa" interface) dengan semua informasi yang mendukung (buku-buku, manual, pita video) yang menggambarkan sintaksis dan semantic system.

Perancangan antarmuka (interface) ada 3 hal yang harus dijadikan focus perhatian, yaitu (Pressman, 2020:459):

1. Desain interface antar modul-modul perangkat lunak.
2. Desain antar perangkat lunak dan prosedur konsumen informasi bukan manusia lainnya (yakni entitas eksternal lainnya) dan
3. Desain interface antar seorang manusia (seperti pemakai) dan komputer.

Faktor-faktor yang perlu diperhatikan dalam merancang user interface antara lain (Scheiderman dan Plaisant, 2010:31):

1. *Time to learn* (waktu belajar)

Lamanya waktu yang diperlukan oleh user agar dapat mempelajari bagaimana cara menggunakan aksi-aksi yang relevan dalam mengerjakan suatu tugas.

1. *Speed of performance* (kecepatan kinerja)

Lamanya waktu yang dibutuhkan dalam mengerjakan suatu tugas.

1. *Rate of errors by users* (tingkat kesalahan)

Seberapa banyak dan tipe kesalahan apa saja yang dibuat oleh pengguna dalam mengerjakan suatu tugas.

1. *Retention over time* (daya ingat)

Seberapa baik pengguna mempertahankan pengetahuannya dalam jangkawaktu tertentu.

1. *Subjective satisfaction* (kepuasan subyektif)

Menyatakan besaran kepuasan user terhadap aspek-aspek yang ada di dalam interface pada saat penggunaan aplikasi.

Delapan aturan yang dapat digunakan sebagai petunjuk yang baik untuk merancang user interface yaitu (Shneiderman dan Plaisant, 2010:88) :

1. Konsistensi

Konsistensi dilakukan pada urutan tindakan, perintah, dan istilah. Konsistensi mencakup tata letak, menu, jenis font, bentuk prompt, dan warna tulisan yang digunakan.

1. Memenuhi kebutuhan universal

Rancangan layar harus mempertimbangkan perbedaan pengguna seperti pengalaman pengguna dengan sistem (novice-expert), rentang usia, disabilitas, dan variasi teknologi.

1. Memberikan umpan balik yang informatif

Untuk setiap aksi pengguna, sistem harus dapat memberikan umpan balik kepada pengguna. Untuk aksi kecil dan sering dilakukan, berikan umpan balik yang sederhana, sementara untuk aksi besar dan jarang dilakukan, berikan umpan balik yang lebih jelas.

1. Merancang dialog untuk menghasilkan keadaan akhir

Urutan aksi sebaiknya diorganisir dalam tiga bagian yaitu awal, tengah, dan akhir. Pada bagian akhir berikan umpan balik yang informatif. Hal ini berguna untuk memberikan kepuasan saat pekerjaan tersebut telah selesai, rasa lega, menurunkan kekhawatiran, dan tanda persiapan untuk aksi-aksi berikutnya.

1. Mencegah kesalahan

Usahakan agar sistem dapat mencegah pengguna melakukan kesalahan yang fatal. Jika pengguna melakukan kesalahan, sistem dapat mendeteksinya da memberikan instruksi yang mudah dipahami, spesifik, dan membangun untuk penanganan kesalahan.

1. Memungkinkan pembalikan aksi yang mudah

Sistem yang dibuat diharapkan memungkinkan pengguna untuk membatalkan aksi yang dilakukannya. Demikian, pengguna dapat merasa lebih rileks karena pen gguna mengetahui bahwa setiap kesalahan yang dilakukannya dapat dibatalkan dan menjadikan pengguna lebih berani untuk mencoba dengan fitur-fitur yang ada.

1. Mendukung pusat kendali internal

Siarkan pengguna yang mengkontrol antarmuka dan antarmuka tersebut merespon tindakan pengguna, bukan sebaliknya.

1. Mengurangi beban ingatan jangka pendek

Keterbatasan ingatan jangka pendek manusia membutuhkan tampilan yang sederhana, penggabungan beberapa halaman, perpindahan window yang minim, dan waktu yang cukup untuk mempelajari kode-kode, singkatan, serta urutan aksi.

Prinsip-prinsip perancangan user interface (Summerville, 2003:34):

1. Kebaikan user, interface harus menggunakan istilah dan konsep yang dimabil dari pengalam orang-orang yang akan paling sering menggunakan sistem tersebut.
2. Konsisten, interface harus konsisten, jika dimungkinkan, operasi- operasi yang hampir sama harus diaktifkan dengan cara yang sama.
3. Kejutan kinimal, user harus tidak dikejutkan oleh perilaku sistem.
4. Kemampuan pemulihan (recoverability), interface harus mencakup mekanisme untuk memungkinkan user pulih dari error.
5. Panduan user, interface harus menyediakan umpan bail yang berarti ketika terjadi kesalah dan menyedikan fasilitias bantuan yang context- sensitive.
6. Keragaman user, interface harus menyediakan fasilitas interaksi yang sesuai untuk berbagai user sistem.

Jadi, perancangan interface sangat penting sebab dalm hal ini menentukan keberhasilan suatu sistem. Suatu sistem yang user firendly akan lebih berguna daripada sistem yang lengkap tetapi sulit untuk dopekajari ataupun digunakan oleh user.

**2.7 Metode Pengujian Perangkat Lunak**

Menurut Hariyanto (2004:34), Pengujian adalah proses pemeriksaan atau evaluasi sistem atau komponen sistem secara manual atau otomatis untuk memverifikasi apakah sistem memenuhi kebutuhan-kebutuhan yang dispesifikasikan atau mengidentifikasi perbedaan-perbedaan antara hasil yang diharapkan dengan hasil yang terjadi.

Menurut Mustaqbal dkk (2015:31), “Pengujian adalah suatu proses pelaksanaan suatu program dengan tujuan menemukan suatu kesalahan. Suatu kasus test yang baik adalah apabila test tersebut mempunyai kemungkinan menemukan sebuah kesalahan yang tidak terungkap.”

Menurut Rosa & Shalahuddin (2013:272), Pengujian perangkat lunak adalah sebuah elemen sebuah topik yang memiliki cakupan luas dan sering kali dikaitkan dengan verifikasi (verification) dan validasi (validation) (V&V). Verifikasi mengacu pada sekumpulan aktivitas yang menjamin bahwa perangkat lunak mengimplementasikan dengan benar sebuah fungsi yang spesifik. Validasi mengacu pada sekumpulan aktivitas yang berbeda yang menjamin bahwa perangkat lunak yang dibangun dapat ditelusuri sesuai dengan kebutuhan pelanggan (customer).

Menurut Pressman (2020:596), “Pengujian sistem adalah “Sederetan pengujian yang berbeda yang tujuan utamanya adalah sepenuhnya menggunakan sistem berbasis komputer.”

Sebuah perangkat lunak harus dijaga untuk keperluan sebagai berikut (Rosa & Shalahuddin, 2018:271):

1. Agar dapat selamat bertahan hidup di dunia bisnis.
2. Dapat bersaing dengan perangkat lunak lain.
3. Penting untuk pemasaran global.
4. Mengefektifkan biaya agar tidak membuang perangkat lunak karena kegagalan pemasaran atau kegagalan produksi.
5. Mempertahankan pelanggan dan meningkatkan keuntungan.

Sejumlah aturan yang berfungsi sebagai sasaran pengujian adalah sebagai berikut (Pressman, 2020:132):

1. Pengujian merupakan proses eksekusi suatu program dengan tujuan untuk menemukan kesalahan-kesalahan yang ada di dalamnya.
2. Kasus pengujian yang baik adalah pengujian ulang memiliki kemungkinan yang tinggi untuk menemukan kesalahan-kesalahan yang tidak ditemukan sebelumnya.
3. Pengujian yang berhasil adalah pengujian yang mampu menyingkapkan kesalahan yang tidak ditemukan sebelumnya.

Atribut-atribut dari pengujian yang baik adalah sebagai berikut (Pressman, 2020):

1. Pengujian yang baik memiliki probabilitas yang tinggi untuk menemukan kesalahan.

Untuk mencapai tujuan ini, penguji harus memahami perangkat lunak dan mencoba mengembangkan sebuah gambaran mental bagaimana perangkat lunak dapat gagal.

1. Pengujian yang baik tidak berulang-ulang.

Waktu dan sumber daya terbatas. Tidak ada gunanya melakukan pengujian yang memiliki tujuan yang sama dengan pengujian lain.

1. Pengujian yang baik harus menjadi “bibit terbaik".

Dalam sebuah kelompok pengujian yang memiliki tujuan serupa, keterbatasan waktu dan sumber daya dapat mengurangi pelaksanaan bahkan hanya sebagian kecil dari pengujian ini.

1. Pengujian yang baik harus tidak terlalu sederhana atau terlalu rumit. Meskipun terkadang sangat memungkinkan bagi kita untuk menggabungkan serangkaian yang biasanya terjadi adalah banyak kesalahan ulang yang harus ditutupi. Secara umum setiap pengujian harus dilakukan secara terpisah.

**2.8 Perancangan *Mobile Apps***

Menurut Sutarman (2003:56), *Mobile apps* merupakan alamat (URL) yang berfungsi sebagai tempat penyimpanan data dan informasi dengan berdasarkan topik tertentu".Perancangan sebuah *Mobile apps* dimaksudkan untuk menghasilkan *Mobile Apps* yang dinamis dan interaktif baik dalam segi desain maupun isi. Perancangan *Mobile apps* harus sederhana dan mudah dimengerti, serta menjaga proses kerja tetap cepat ketika di akses. Dalam membuat *Mobile apps* ini juga harus memperhatikan tujuan pembuatan *Mobile spps*, fungsi dan fitur, tata letak halaman.

Menurut Widodo (2013:76), “*Mobile apps* atau biasa sering disingkat dengan web adalah halaman yang berisi tulisan-tulisan yang lazimnya dilengkapi dengan gambar-gambar maupun foto-foto.”

Perancangan *mobile apps* adalah istilah umum yang digunakan untuk mencakup bagaimana isi web konten ditampilkan, (biasanya berisi *hypertext* atau *hypermedia*) yang dikirimkan ke pengguna akhir melalui *world wide web*, dengan menggunakan sebuah browser web atau perangkat lunak berbasis web. Dalam merancang *mobile apps* harus memperhatikan tujuan pembuatan web, struktur *mobile apps*, desain *mobile apps*, fungsi dan fitur seperti komposisi multimedia.

Untuk mendapatkan antarmuka situs web yang baik, penulis menggunakan beberapa pedoman perancangan situs web yaitu (Kendall & Kendall, 2003:106):

1. Menggunakan perangkat-perangkat professional. Menggunakan perangkat lunak yang disebut web editor seperti Macromedia Dreamweaver.
2. Memperlajari *MobileApps*-*MobileApps* lainnya.
3. Gunakan sumber daya yang ditawarkan web.
4. Gunakan perangkat-perangkat yang telah dipelajari.

Panduan yang baik untuk merancang web (Kendall dan Kendall, 2003:108):

1. Gunakan perangkat desain yang professional perancangan harus menggunakan sebuah web editer seperti Microsoft Font Page atau Adobe PageMill untuk menunjang kreatifitas dan kecepatan dalam kerja.
2. Pelajari *mobile apps* yang sudah ada pengertian *mobile apps* lain yang kiranya berhubungan dengan apa yang hendak dikerjakan. Analisa elemen yang digunakan dan lihat bagaimana element berfungsi.
3. Perhatikan *mobile apps* yang dikerjakan oleh desainer professioanal dengan memperhatikan *mobile apps* pada professional ini kita bisa
4. mengetahui bagaimana sesungguhnya pengguna berinteraksi dengan *mobile apps*, misalnya cara kerja link ada email, form yang hendak diisi atau chartroom.
5. Baca buku-buku dengan membaca buku mengenai perancangan web, seseorang bisa menambah keahlian.
6. Gunakan sumber daya yang ditawarkan web.
7. Perhatikan web dengan perancangan yang buruk dengan memperhatikan rancangan yang gagal, kita bisa menghindari kesalahan yang sama.
8. Ciptakan template milik anda sendiri jika mengadopsi suatu halaman standar untuk sebagian besar halaman yang ada ciptakan, anda akan mendapatkan *mobile apps* ana berjalan cepat sekali serta terlihat baik.
9. Gunakan plug-in, menyenangkan memiliki fitur-fitur dengan halaman-halaman yang terlihat anda tidak memiliki setiap plug-in baru. Jangan mengecilkan hati audio dan video secukupnya saja memang pengunjung dengan halaman web anda.
10. Rencanakan kedepan tentang:
11. Struktur

Melakukan perancangan *mobile apps* adalah merupakan salah satu faktor terpenting yang harus selalu dilakukan dalam mengembangkan *mobile apps* professional. Setiap halaman yang dihasilkan harus memiliki suatu hubungan yang jelas dengan pesan yang akan disampaikan.

1. Isi

Setiap halaman harus dapat memberikan suatu jenis informasi tertentu, yang tentunya harus dapat memberikan daya tarik sehingga para netter menjadi tertarik dan ingin mengunjungi *mobile apps* tersebut.

1. Text

Penggunaan kata-kata dan kalimat yang disajikan harus dapat memberikan arti yang jelas dan mudah dimengerti.

1. Grafik

Penampilan dalam bentuk grafik harus memiliki latar belakang yang mendukung dan yang disertai dengan berbagai variasi tampilan, bentuk perintah dan kombinasi warna yang ditampilkan harus bisa memberikan makna jelas. Keluaran dalam bentuk grafik harus memperhatikan tujuan perancangan grafik jenis atau tipe data yang ditampilkan, siapa pemakainya, efek yang ditimbulkan bagi pemakai sehubungan dengan jenis grafik yang dihasilkan.

1. Presentasi

Bentuk penyajian informasi harus dibuat semenarik mungkin, agar halaman-halaman informasi yang disajikan dapat memberikan suatu image yang baik bagi para netter.

1. Navigasi

Berhubungan dengan teknik dan cara-cara yang dipergunakan untuk melakukan pergerakan terhadap halaman-halaman yang terdapat pada perancangan *mobile apps* yang dihasilkan.

1. Promosi

Bentuk sebuah perancangan *mobile apps* harus mengandung nilai-nilai jual, yang artinya dapat mempromosikan segala sesuatu yang akan disampaikan.

Sebelum mendesain web, sebaiknya pahami beberapa fungsi dari situs web agar desain yang kita buat sesuai dengan fungsi web tersebut. Terdapat enam hal yang perlu diperhatikan dalam merancang *mobile apps* (Sutarman, 2003:72):

1. Kegunaan dari *mobile apps* yang akan dibangun seperti apa, kita harus menentukan apa yang akan dicapai dalam *mobile apps*. Hal ini akan memberikan gambaran mengenai apa yang harus dikerjakan.
2. Tujuan-tujuan yang akan anda dapatkan dari adanya *mobile apps* apakah untuk memperoleh keuntungan finansial atau untuk berbagai informasi atau keduanya.
3. Target pengunjung, siapa saja yang diharapkan untuk mengunjungi *MobileApps*. Sesuai dengan tujuan di atas, dapat berdasarkan umur, lokasi, pendidikan, teknologi informasi atau semua segmen.
4. Content apa saja yang disajikan dalam *mobile apps* anda. Informasi, hiburan, e- commerce. Content yang harus dapat mendukung tujuan dari *mobile apps* anda serta pengunjung. Sehingga pahamilah apa yang diinginkan pengunjung.
5. Gaya yang digunakan dalam mendesain *mobile apps* juga sangat penting dan diselaraskan dengan hal-hal diatas.
6. Tentukan teknologi yang anda gunakan untuk membangun *mobile apps*.

Ada bermacam-macam langkah yang digunakan webmaster untuk membangun suatu home page. Berikut ini adalah proses ecara umum yang dilakukan kebanyakan webmaster di Indonesia untuk web (Sutarman, 2003:76):

1. Membuat skema desain
2. Membuat layout desain
3. Membagi gambar menjadi potongan-potongan kecil.
4. Membuat animasi
5. Membuat HTML
6. Programming dan Script
7. Evaluasi dan test
8. Upload HTML
9. Promosi Homepage

*Mobile apps* dapat dikategorikan menjadi 2 yaitu *mobile apps* statis dan *mobile apps* dinamis atau interaktif, web statis adalah web yang berisi atau menampilkan informasi-informasi yang sifatnya statis, sedangkan web dinamis adalah web yang menampilkan informasi serta dapat berinteraksi dengan user. Suatu situs web yang baik memiliki kriteria sebagai berikut (Suyanto, 2009:61-69):

1. *Usability*

*Usability* adalah sebagai suatu pengalaman pengguna dalam berinteraksi dengan aplikasi atau situs web sampai pengguna dapat mengoperasikannya dengan mudah dan cepat. Situs web harus memenuhi lima syarat untuk mencapai tingkat usability yang ideal, antara lain:

1. Mudah untuk dipelajari
2. Efisien dalam penggunaan
3. Mudah untuk diingat
4. Tingkat kesalahan rendah Kepuasan pengguna
5. Sistem Navigasi (Struktur)

Kemudahan bernavigasi dalam situs web melibatkan sistem navigasi situs web secara keseluruhan dan desain interface situs web tersebut. Navigasimembantu pengunjung untuk menemukan jalan yang mudah ketika menjelajahi situs web memberitahu dimana mereka berada, kemana mereka bisa pergi. Dengan demikian mereka dapat menemukan apa yang mereka cari dengan cepat dan mudah.

Syarat navigasi yang baik:

1. Mudah dipelajari
2. Tetap
3. Memungkinkan feedback
4. Muncul dalam konteks
5. Menawarkan alternatif lain
6. Memerlukan perhitungan waktu dan tindakan
7. Menyediakan pesan visual yang jelas
8. Menggunakan label yang jelas dan mudah dipahami
9. Mendukung tujuan dan perilaku user
10. *Graphic Design* (Desain Visual)

Desain yang baik sebaiknya memiliki komposisi warna yang baik dan konsisten, layout grafik yang konsisten, teks yang mudah dibaca, penggunaan grafik yang memperkuat isi teks, penggunaan animasi pada tempat yang tepat, isi animasi yang memperkuat isi teks, dan secara keseluruhan membentuk suatu pola yang harmonis.

1. *Contents*

Sebaik apapun situs web secara desain grafis, tanpa konten yang berguna dan bermanfaat maka akan kurang berarti. Konten yang baik akan menarik, relevan, dan sesuai untuk target audien situs web tersebut. Gaya penulisan dan bahasa yang diperlukan harus sesuai dengan web dan target audiens. Hindari kesalahan dalam penulisan, termasuk tata bahasa dan tanda baca, di tiap halaman, header, dan judulnya. Buat daftar penjelasan untuk istilah- istilah khusus.

1. *Compatibility*

Situs web harus kompatibel dengan berbagai perangkat tampilannya (browser), harus memberikan alternatif bagi browser yang tidak dapat melihat situsnya.

1. *Loading Time*

Sebuah penelitian yang dilakukan oleh Zona Research (April 1999) menyatakan bahwa 80% pengunjung akan menutup browser bila halaman web yang ia buka tidak tampil dalam 7-8 detik. Penelitian Jupiter Media Metrix (September 2001 pengunjung akan kembali mengunjungi situs web yang tampil lebih cepat.

1. *Functionality*

Seberapa baik sebuah situs web bekerja dari aspek teknologinya, dalam hal ini dapat melibatkan programmer dengan script-nya, misalnya HTML, PHP, ASP, ColdFusion, CGI, SSI, dan lain-lain. Konsisten Amerika Serikat) mengatakan bahwa 40%

1. *Accesibility*

Halaman web harus dapat dipakai oleh setiap orang, baik anak-anak, orang tua dan muda, termasuk orang cacat Ada berbagai hambatan yang ditemui dari sisi pengguna untuk bisa menikmati halaman web itu. Desainer setidaknya harus memeprtimbangkan masalah ini dan memberikan solusinya, terutama yang berhubungan dengan tugasnya sebagai desainer.

1. *Interactivity*

Interaktivitas adalah apa yang melībatkan pengguna situs web sebagai user experience dengan situs web itu sendiri. Dasar dari interaktivitas adalah hperlinks dan mekanisme feedback.

Hal-hal yang perlu diperhatikan dalam perancangan sebuah situs adalah (Sutarman, 2008 264-265):

1. Kegunaan

Fungsi dari *mobile apps* yang akan dibangun seperti apa, kita harus menentukan apa yang akan dicapai dalam membuat *mobile apps*. Hal ini akan memberikan gambaran mengenai apa yang harus dikerjakan. Misalnya Anda akan membangun *mobile apps* untuk :

1. Menjual produk
2. Menyediakan informasi
3. Promosi diri
4. Entertainment
5. Tujuan

Tujuan yang akan Anda dapatkan dari pembuatan *mobile apps*, apakah memperoleh keuntungan secara financial atau sekedar berbagi informasi, ataupun keduanya

1. Target Pengunjung

Siapa saja yang diharapkan untuk mengunjungi *mobile apps*, sesuai dengan tujuan di atas dapat dikelompokkan berdasarkan umur, lokasi, tingkat pendidikan, teknologi informasi, atau semua kalangan/segmen.

1. Content Apa saja yang dihasilkan dalam *mobile apps* Anda? Informasi, hiburan, e- commerce. Content harus dapat mendukung tujuan dari *mobile apps*. Anda serta target pengunjung, maka dari itu pahamilah apa yang diinginkan pengunjung
2. Gaya

Gaya yang digunakan dalam desain *mobile apps* juga sangat penting, dan diselaraskan dengan hal – hal di atas. Misalnya target pengunjung *MobileApps* adalah anak – anak, maka pengaturan warna disesuaikan dengan minat anak – anak, isinya juga menarik buat mereka. Untuk pengunjung remaja, gaya yang digunakan misalnya fungky, maka diperhitungkan dengan benar sasaran segmen pembaca *mobile apps* Anda beserta gaya *mobile apps* yang akan Anda pakai.

1. Teknologi Tentukan teknologi yang Anda akan gunakan untuk membangun *mobile apps*, apakah cukup dengan HTML saja perlu interaksi dengan bahasa pemprograman seperti Cold Fussion, PHP, PELS, JAVA, dan lainnya.

Panduan yang baik untuk merancangan sebuah *mobile apps* (Kendall dan Kendall, 2007:36-47):

1. Gunakan perangkat desain yang profesional

Perancangan harus menggunakan sebuah web editor seperti Microsoft FrontPage atau Adobe Page Mill untuk menunjang kreatifitas dan kecepatan kerja.

1. Pelajari *mobile apps* yang sudang ada

Perhatikan *mobile apps* lain yang kiranya berhubungan dengan apa yang hendak dikerjakan. Analisa elemen yang digunakan dan lihat bagaimana element itu berfungsi.

1. Perhatikan *mobile apps* yang dikerjakan oleh desainer profesional

Dengan memperhatikan *mobile apps* para profesional ini, bisa mengetahui bagaimana sesungguhnya user beriìnteraksi dengan *mobile apps*, misalnya cara kerja pada email, form yang hendak diisi atau chatroom.

1. Baca buku-buku

Dengan membaca buku mengenai perancangan web, seseorang bisa menambah keahliannya.

1. Perhatikan halaman web dengan perancangan yang buruk juga

Dengan memperhatikan rancangan yang gagal, bisa menghindari kesalahan yang sama.

1. Ciptakan template milik sendiri

Jika mengadopsi suatu halaman standar untuk sebagian besar halaman yang diciptakan, akan mendapatkan *mobile apps* berjalan dengan cepat sekali serta terlihat baik.

1. Gunakan plug-in, audio, dan video secukupnya

Memang menyenangkan memiliki fitur-fitur dengan halaman-halaman yang terlihat *mobile apps* tidak memiliki setiap plug-in baru. Jangan mengecilkan hati pengunjung dengan halaman web.

1. Rencanakan ke depan tentang:
2. Struktur

Melakukan perancangan *mobile apps* adalah merupakan salah satu faktor terpenting yang harus selalu dilakukan dalam mengembangkan *mobile apps* yang profesional. Setiap halaman yang dihasilkan harus memiliki suatu hubungan yang jelas dengan pesan yang akan disampaikan.

1. Isi

Setiap halaman harus dapat memberikan suatu jenis informasi tertentu, yang tentunya harus dapat memberikan daya tarik tertentu sehingga para user menjadi tertarik dan ingin mengunjungi *mobile apps* tersebut.

1. Teks

Penggunaan kata-kata dan kalimat yang disajikan harus dapat memberikan arti yang jelas dan mudah dimengerti.

1. Grafik

Penampilan dalam bentuk grafik harus memiliki latar belakang yang mendukung dan yang disertai dengan berbagai variasi tampilan, bentuk perintah dan kombinasi warna yang ditampilkan harus bisa memberikan makna jelas. Keluaran dalam bentuk grafik harus memperhatikan tujuan perancangan grafik jenis atau tipe data yang siapa pemakainya, efek yang ditimbulkan bagi pemakai

1. Presentasi

Bentuk halaman-halaman informasi yang disajikan dapat memberikan suatu image yang baik bagi para netter. Tata letak penulisan, perpaduan suara yang dihasilkan, dan disertai dengan berbagai animasi dapat menjadi informasi awal sebelum seseorang membaca informasi dari site tersebut lebih lanjut.

1. Navigasi

Adalah hubungan dengan teknik dan cara-cara yang dipergunakan untuk melakukan pergerakkan terhadap halaman-halaman yang terdapat pada perancangan *mobile apps* yang dihasilkan. Kemudahan melakukan pergerakkan mouse atau penggunaan toolbar yang disediakan merupakan nilai tambah lainnya.

1. Promosi

Bentuk sebuah perancangan *mobile apps* harus mengandung nilai-nilai jual, yang artinya dapat mempromosikan segala sesuatu yang akan disampaikan. Harus memiliki kemampuan untuk mengiring orang- orang dalam membuka sebuah *mobile apps*, dalam arti harus dapat menciptakan *video on demand*.

Menurut Hidayat (2010:2)*, Mobile apps* atau situs dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman.

Menurut Arief (2011:7), ”Web adalah salah satu aplikasi yang berisikan dokumen-dokumen multimedia (teks, gambar, suara, animasi, video) didalamnya yang menggunakan protocol HTTP (hypertext transfer protocol) dan untuk mengakses menggunakan perangkat lunak yang disebut browser.”

*Mobile apps* adalah sekumpulan halaman web yang bisa diakses melalui web browser. Satu halaman disebut Hompage, halaman ini adalah halaman yang pertama kali dilihat ketika seseorang mengunjungi *mobile apps*. Dari hompage pengunjung dapat mengklik teks atau gambar untuk berpindah kehalaman lain dalam *mobile apps* tersebut atau kehalaman lain di *mobile apps*. Homepage berada pada posisi teratas dengan halaman-halaman terkait berada dibawahnya.

**2.8.1 Arsitektur & Navigasi Pada Mobile Apps**

Navigasi pada sebuah *MobileApps* dimaksudkan untuk memberikan kemudahan kepada masyarakat untuk mengakses informasi yang telah disediakan, masyarakat yang membutuhkan informasi dapat mengikuti navigasi yang disediakan sehingga masyarakat akan langsung menuju kepada informasi yang dibuthkan tanpa perlu berputar-putar mencari informasi dari satu halaman ke halaman lain atau akhirnya tersesat dan tidak menemukan informasi yang diinginkan. Beberapa teknik navigasi yang dapat digunakan adalah (Lopuck, 2006:237):

1. Navigasi Global (Global Navigation)

Navigasi global adalah cara navigasi yang menyediakan navigation links agar saat masyarakat ingin kembali ke informasi sebelumnya, masyarakat tidak perlu terus meng-klik tombol kembali pada browser berkali-kali, cukup dengan meng-klik link yang disediakan untuk kembali pada informasi yang dibutuhkan. Menempatkan pola navigasi global pada tempat yang sama di setiap halaman membaut masyarakat terbiasa dengan *MobileApps* yang dibuat.

1. Navigasi Bagian (Section Navigation)

Navigasi bagian digunakan dengan asumsi bahwa setiap bagian navigasi memiliki beberapa tingkatan didalamnya. Dimana navigasi utama dirancang agar dapat dengan mudah ditemukan disetiap halaman *MobileApps*, dan agar saat masyarakat meng-klik tombol navigasi utama, akan muncul drop- down akan memunculkan pilihan navigasi dan informasi yang menu yang lebih spesifik lainnya. Terdapat tiga kelompok pada pola navigasi ini yaitu kelompok navigasi primer, sekunder, dan tersier. Navigasi Primer adalah navigasi yang dirancang dapat dengan mudah ditemukan di setiap halaman *MobileApps*, terletak dibagian depan-tengah setiap halaman, dan berukuran lebih besar, sedangkan bagian sekunder berukuran lebih kecil dan terletak di samping kiri halaman, dan bagian tersier ini biasanya beruapa search engine yang terletak di kanan atas setiap halaman *MobileApps*.

1. Bread Crumbs Navigation

Pola navigasi ini adalah pola navigasi yang merekam jejak setiap halaman yang dikunjungi. Dan jika masyarakat ingin kembali ke informasi sebelumnya, masyarakat tidak perlu menekan tombol Kembali satu per satu, cukup dengan menekan rekaman jejak yang telah tersedia dan akan langsung kembali ke informasi yang dituju.

Aspek-aspek yang harus dipertimbangkan dalam navigasi adalah (Simarmata, 2010:475):

1. Struktur situs (graf node dan link).
2. Navigasi horizontal dan vertical, navigasi campuran.
3. Bar navigasi.

Menurut Hakim (2004), Navigasi digunakan untuk menjelajah halaman demi halaman dalam suatu situs web, dalam navigasi biasanya disertai tombol-tombol yang akan mengantarkan pengunjung ke halaman yang diinginkan”. Navigasi termasuk struktur terpenting dalam pembuatan suatu multimedia. Peta navigasi merupakan rancangan hubungan dan rantai kerja dari beberapa area yang berbeda dan dapat membantu mengorganisasikan seluruh elemen multimedia dengan pemberian perintan dan pesan.

**2.9 Perancangan Database**

Data merupakan fakta mengenai suatu objek seperti manusia, benda, peristiwa, konsep, keadaan dan sebagainya yang dapat dicatat dan mempunyai arti. Dalam pengembangan perangkat lunak pasti ada yang basis data ini sendiri mempunyai peranan penting untuk menyimpan suatu data, dengan adanya basis data ini pengguna dapat mengakses data dengan mudah dan cepat.

Menurut Yanto (2016:11), “Basis data merupakan kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi), untuk memenuhi berbagai kebutuhan.”

Menurut Lubis (2016:3), “Basis data merupakan tempat berkumpulnya data yang saling berhubungan dalam suatu wadah (organisasi/perusahaan) bertujuan agar dapat mempermudah dan mempercepat untuk pemanggilan atau pemanfaatan data tersebut.”

Berdasarkan pendapat para ahli di atas, dapat disimpulkan bahwa basis data / database merupakan kumpulan data-data yang saling berhubungan dan tersimpan di perangkat keras dan diolah menggunakan perangkat lunak.

Menurut Jhontor (2012:36), Database yang baik adalah database yang dapat memberikan informasi yang sesuai dengan kehendak user (pengguna). Maka dari itu perlu adanya proses desain database yaitu proses memilah dan memilih data-data yang memang dibutuhkan oleh sistem, alhasil informasi yang diberikan sesuai dengan kehendak pengguna. Istilah desain database dapat dipergunakan untuk memvisualisasikan bagian-bagian yang berbeda dari keseluruhan database.

Dalam melakukan proses desain database terdapat beberapa yang harus dijalani, yaitu (Jhontor, 2012:54):

1. Analisis Persyaratan : memahami dan mengetahui data yang harus disimpan dalam database, aplikasi apa yang harus dibangun di atasnya, jenis operasi apa yang akan digunakan, dst.
2. Desain Data Base Konseptual : berdasarkan informasi yang diperoleh dari tahap analisis persyaratan inilah dipergunakan untuk mengembangkan deskripsi data menjadi lebih tinggi. Pada ini desainer sering membuat yang anmanya model E-R yang tujuan untuk membuat penyederhanaan gambaran tentang data yang sesuai dengan pemikiran user / pengguna.
3. Desain Data Base Logika : disinilah proses merubah model E-R skema database yang rasional. Dengan tujuan untuk mendapatkan skema konseptual pada data rasional sehingga sering disebut skema logika.
4. Perbaikan Skema : pengecekan sekumpulan relasi dalam skema sebelumnya (skema database rasional), supaya dapat menemukan problem yang ada dan memperbaikinya.
5. Desain Data Base Fisik : pembentukan indeks pada beberapa yang telah dibuatlah pada tahap ini dikerjakan. Mengelompokkan atau mendesain ulang yang bersifat substansial terhadap bagian dari skema database.
6. Desain Aplikasi dan Keamanan : Keseluruhan proyek software yang menggunakan DBMS sebaiknya memperhitungkan -faktor aplikasi yang ada diluar database, seperti enkripsi, digital, dll.

Menurut Connolly dan Begg, (2002:279), “Perancangan basis data merupakan proses menciptakan perancangan untuk basis data yang akan mendukung operasi dan tujuan perusahaan.”

Proses dalam metodelogi perancangan dibagi 3 (tiga) tahap (Connolly dan Beg, 2002:419):

1. Conseptual Database Design

Conceptual database design merupakan proses membangun suatu model berdasarkan infromasi yang digunakan oleh perusahaan atau organisasi, tanpa pertimbangkan perencanaan fisik.

1. Logical Database Design

Logical database design adalah proses pembuatan suatu model informasi yang digunakan pada perusahaan berdasarkan pada model data yang spesifik, tetapi tidak tergantung dari Database Management System(DBMS) yang khsuus dan pertimbangan fisik yang lain (Connolly dan Begg, 2002:441).

1. Physical Database Design

Physical database design adalah suatu proses untuk menghasilkan gambaran dari implementasi basis data pada tempat penyimpanan, menjelaskan dasar dari relasi, organisasi file dan indeks yang digunakan untuk efisiensi data yang menghubungkan beberapa integrity constraints dan keamanan (Connolly dan Begg, 2002:478).

Database Management System (DBMS) adalah perangkat lunak (software) yang berfungsi untuk mengolah database. Mulai dari membuat basis data itu sendiri, sampai dengan proses yang berlaku dalam basis data tersebut, baik berupa entry, edit, delete, query terhadap data, membuat laporan dan lain sebagainya secara efektif dan efisien.

Menurut Abdul Kadir dan Triwahyuni (2013:339), “DBMS (Database Management System) adalah sistem yang secara khusus dibuat untuk memudahkan pemakai dalam mengolah basis data. Sistem ini dibuat untuk mengatasi kelemahan sistem pemrosesan yang berbasis berkas.”

DBMS memiliki beberapa fasilitas yaitu (Connolly dan Begg, 2010:66) :

1. Memungkinkan pengguna untuk menentukan basis data, biasanya melalui Data Definition Language (DDL). DDL memungkinkan pengguna untuk menentukan jenis data, struktur, dan kendala pada data yang akan disimpan dalam basis data.
2. Memungkinkan menghapus, dan mengambil data dari basis data, biasanya melalui Data Manipulation Language(DML). DML memiliki fasilitas untuk data yang disebut query language.Bahasa query yang paling umum adalah Structured Query Languege(SQL) yang sekarang merupakan bahasa standar untuk DBMS relasional.
3. Menyediakan akses control ke basis data. Contohnya:
   1. Sistem keamanan (security system), mencegah pengguna yang tidak sah mengakses basis data.
   2. Sistem integritas (integrity system), yang mempertahankan data yang disimpan.
   3. Concurrency Control System, yang memungkinkan berbagi akses basis data
   4. Sistem control pemulihan (recovery control system), yang mengembalikan keadaan basis data ke keadaan semula yang konsisten.
   5. Sebuah catalog yang dapat diakses pengguna, yang berisi dari data dalam basis data.

Secara umum, suatu Sistem Manajemen Basis Data (DBMS) terdiri dari (Simarmata, 2007:15) :

* 1. Suatu koleksi modul, program, dan table-tabel.
  2. Suatu metode akses dan sebuah metodologi akses.
  3. Sekumpulan masukan data, manipulasi data, pelaporan, dan tool-tool retrieval

1. Ketentuan built-in untuk keamanan integritas data
2. Sekumpulan file, record, serta uraian-uraian elemen
3. Pengaturan tentang logika untuk mengkonstruksi file dan menangani data
4. Spesifikasi untuk menyimpan data fisik.

Penggunaan DBMS (Simarmata, 2007:16) :

* 1. Memudahkan pemindahan dari rutinitas manipulasi data eksternal dari aliran program (yaitu urutan, penggabungan, dan lain-lain)
  2. Menghapus kebutuhan untuk pemrosesan file secara penuh dalam pemilihan retrieval atau kondisi update
  3. Memudahkan pengguna file sesuai golongan oleh berbagai program (memperbaharui dan retrieval)
  4. Menyediakan pemulihan data setelah kegagalan
  5. Menyediakan akses data secara logika daripada secara fisik.

Komponen utama DBMS iyalah sebagai berikut (Kadir, 2005:18) :

1. Perangkat keras (hardware)

Perangkat keras berupa computer dan bagian-bagian didalamnya, seperti processor, memory, dan harddisk. Komponen inilah yang melakukan pemrosesan dan penyimpanan basis data

1. Data

Data didalam basis data mempunyai sidat saling berkaitan dan berbagi/bisa dipakai oleh sejumlah pengguna.

1. Perangkat lunak (software)

Perangkat lunak dalam hal ini DBMS, berkedudukan antara basis data (data yang disimpan dalam harddisk) dan pengguna. Perangkat lunak inilah yang berperan melayani permintaan pengguna.

1. Pengguna

Pemrograman aplikasi adalah orang yang membuat aplikasi yang menggunakan basis data. Program aplikasi tertentu saja yang sesuai dengan kebutuhan pengguna. Pengguna aplikasi adalah orang yang mengoperasikan program aplikasi yang dibuat oleh pemrograman aplikasi. Pengaksesan terhadap data didalam basis data dapat dilakukan dengan perintah SQL (Structured Query Language). Selain untuk mengambil data, SQL juga dapat digunakan untuk membangun, menghapus, mengganti, dan memanipulasi data, dan juga operasi-operasi lain.

### **2.9.1 Entity Relationship Diagram (ERD)**

Menurut Rosa & Shalahuddin (2015:53), ”ERD adalah bentuk paling awal dalam melakukan perancangan basis data relasional. Jika menggunakan OODMBS maka perancangan ERD tidak perlu dilakukan.”

Menurut Lubis (2016:31), “ERD menjadi salah satu pemodelan data konseptual yang paling sering digunakan dalam proses pengembangan basis data bertipe relasional.”

Berdasarkan pendapat para ahli dapat disimpulkan bahwa *entity relationship diagram* (ERD) merupakan tahapan perancangan basis yang menggunakan susunan data berupa gambar atau untuk mengambarkan hubungan yang terjadi antar entitas yang berada di basis data.

* 1. **Unifield Modeling Language (UML)**

Menurut Pressman dkk (2020:611), ”*Unified Modeling Language* (UML) adalah bahasa standar untuk menulis cetak biru perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan artefak dari sistem perangkat lunak yang intensif.”

Menurut Hendini (2016:104), “*Unified language* (UML) merupakan metodologi yang digunakan untuk mengembangkan sistem berorientasi objek dan juga sebagai alat bantu dalam pengembangan sistem.”

Menurut Rosa & Shalahuddin (2015:137), “UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.”

Berdasarkan pendapat beberapa ahli di atas, dapat disimpulkan bahwa *unified language* (UML) merupakan bahasa visual untuk pendokumentasian dan pemodelan rancangan sistem dimana hasil visualisasi ini dapat dijadikan sebagai alat bantu dalam pengembangan sistem (Hendini, 2016:119) :

* 1. Use Case Diagram

*Use case diagram* merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem informasi dan digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi serta siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

*Use case diagram* digunakan untuk memudahkan pengembang dalam memahami kebutuhan fungsional dari sistem yang akan dibangun.”

Berdasarkan pendapat para ahli di atas, maka dapat disimpulkan bahwa *use case diagram* merupakan diagram yang berfungsi untuk menjelaskan fungsi terhadap fungsionalitas dari sistem berdasarkan sudut pandang dari luar.

* 1. Activity Diagram

*Activity diagram* merupakan diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

*Activity diagram* merupakan diagram yang menerangkan tentang aktifitas-aktifitas yang dapat dilakukan oleh seorang atau pengguna terhadap aplikasi yang dirancang.” Berdasarkan kutipan dari para ahli di atas, dapat disimpulkan bahwa *activity diagram* merupakan diagram yang menjelaskan tentang *work flow* (aliran kerja) atau aktifitas terhadap aplikasi yang dirancang.

* 1. Sequence Diagram

*Sequence diagram* berfungsi sebagai alat untuk menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

*Sequence diagram* merupakan UML yang menggambarkan interaksi antar objek di dalam dan disekitar sistem, termasuk pengguna, *display*, dan sebagainya berupa *message* yang digambarkan terhadap waktu.

Berdasarkan pernyataan para ahli di atas, dapat disimpulkan bahwa *sequence diagram* merupakan diagram UML yang berfungsi sebagai alat untuk menggambarkan kelakukan objek pada *use case* dengan cara mendeskripsikan waktu hidup objek dan pesan yang terkirim dan diterima.

* 1. Class Diagram

*Class diagram* merupakan diagram yang menjelaskan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem.

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segini pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Berdasarkan definisi dari beberapa ahli, maka dapat disimpulkan bahwa *class diagram* merupakan diagram yang menjelaskan hubungan antar kelas, struktur definisi, operator dan modul yang digunakan untuk membangun aplikasi.

* 1. **Visual Studio Code**

*Visual Studio Code* adalah emulator *source code* yang dikembangkan oleh *Microsoft* untuk *Windows, Linux* dan *MacOS*. Ini termasuk dukungan untuk *debugging*, *GIT Control* yang disematkan, penyorotan sintaks, penyelesaiaan kode cerdas, cuplikan, dan kode *refactoring*. Hal ini juga dapat disesusaikan, sehingga pengguna dapat mengubah tema *editor*, *shortcut keyboard*, dan prefensi. *Visual Studio Code* gratis dan *open source*, meskipun sudah resmi berada di licensi proprietary.

Menurut Sole (2021:22), Visual Studio Code telah menjadi alat pengembangan lintas platform pertama dalam keluarga Microsoft Visual Studio yang berjalan di Windows, Linux, dan macOS. Ini gratis, open source (<https://github>.com/vscode), dan jelas merupakan alat yang berpusat pada kode, yang tidak hanya membuat pengeditan file kode dan sistem proyek berbasis folder menjadi lebih mudah tetapi juga memfasilitasi penulisan web lintas platform , seluler, dan aplikasi cloud melalui platform paling, seperti Node.js dan .NET 5 (termasuk versi .NET Core sebelumnya), dengan dukungan terintegrasi untuk sejumlah besar bahasa dan fitur pengeditan kaya seperti IntelliSense, menemukan referensi, dengan cepat mencapai definisi tipe, dan banyak lagi.

Menurut Lardinois (2015:58), *Visual Studio Code* didasarkan pada *electron*, kerangka kerja yang digunakan untuk menyebarkan aplikas *Node.js* untuk desktop yang berjalan pada Blinklayout. Meskipun menggunakan karangka, *Visual Studio Code* tidak menggunakan Atom dan menggunakan komponen editor yang sama (diberi kode nama “Monaco” yang diguankan dalam *Visual Studio Team Services* yang sebelumnya disebut *Visual Studio online*.

Menurut Sole (2021:23), Visual Studi Code merupakan kerangka kerjauntuk membuat aplikasi lintas platform dengan teknologi asli, dan menggabungkan kesederhanaan editor kode yang kuat dengan alat yang dibutuhkan pengembang untuk mendukunh aplikasi termasuk debugger dan integrasi control berdasarkan Git. Oleh karena itu Visual Studio Code adalah alat pengembang yang lengkap, daripada menjadi editor kode sederhana.

Visual Studio Code menyediakan fitur inti berikut (Sole, 2021:26) :

a. Dukungan bawaan untuk pengkodean dengan banyak bahasa, termasuk yang biasanya Anda gunakan dalam pengembangan lintas platform, seperti C# dan JavaScript, dengan fitur pengeditan lanjutan dan dukungan untuk bahasa tambahan melalui ekstensibilitas.

b. Debugger bawaan untuk Node.js, dengan dukungan untuk debugger tambahan (seperti .NET 5) melalui ekstensibilitas.

c. Kontrol versi berdasarkan sistem versi Git yang, yang memberikan pengalaman terintegrasi untuk komit dan cabang kode pendukung kolaborasi, dan itu adalah pilihan yang tepat untuk alat yang dimaksudkan untuk bekerja dengan kemungkinan bahasa apa pun.

**2.12 Web Services**

Menurut Ramadina (2021:1), *Web service* adalah teknologi website yang menyediakan informasi dan mengelola informasi yang bisa mengintegrasikan data untuk berkomunikasi antar mesin pada sistem informasi atau jaringan (biasa disebut komunikasi server-client). Web service memungkinkan dua aplikasi untuk berkomunikasi satu sama lain, meskipun aplikasi ini berada pada platform yang berbeda. Teknologi web service juga mendukung komunikasi dan interaksi antar mesin di jaringan. Oleh karena itu, pengguna hanya perlu memahami fungsi / metode web service yang disediakan oleh Web untuk memperoleh informasi dari situs web tanpa harus mengunjungi situs web tersebut.

Menurut Laffly (2020:193), *Web service* adalah sistem perangkat lunak yang diidentifikasi oleh URI, yang antarmuka dan ikatan publiknya ditentukan dan dijelaskan menggunakan XML. Definisinya dapat ditemukan oleh sistem perangkat lunak lain. Sistem ini kemudian dapat berinteraksi dengan layanan Web dengan cara yang ditentukan oleh definisinya, menggunakan pesan berbasis XML yang disampaikan oleh protokol Internet.

Menurut Novita (2021:1), *Web service* merupakan salah satu teknologi komputasi terdistribusi yang memungkinkan dilakukannya komputasi pada bnauyak mesin dan hasilnya dapat dimanfaatkan oleh banyak mesin. Tujuan dari *Web Service* adalah untuk memudahkan tukar menukar dan memperbarui informasi.

**2.12.1 Application Progamming Interface (API)**

MenurutDoglio (2018:101), API adalah antarmuka pemograman aplikasi yang digunakan untuk integrasi ke dalam sistem yang berfungsi untuk saling berbagi data antar aplikasi yang berbeda. Tujuan API yaitu mempercepat pengembangan aplikasi dengan menyediakan fungsi yang terpisah sehingga para pengembang tidak perlu lagi membuat fitur yang serupa.

Menurut Irian dkk (2021:23), *Application programming interface* (API) merupakan suatu dokumentasi yang terdiri dari interface, fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Dengan adanya API ini, maka memudahkan *programmer* untuk “membongkar” suatu software, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan programmer menggunakan sistem *function*. Proses ini dikelola melalui sistem operasi. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi.

Menurut Pangestu (2021:3), “API (*Application Programming Interface*) merupakan antarmuka program aplikasi, API ini berfungsi untuk menghubungkan antara perangkat Mikrotik *RouterBoard* dengan aplikasi berbasis web.”

Menurut Muqorobin (2021:78), “*Application Programming Interface* atau API merupakan cara untuk dua aplikasi komputer berbicara satu sama lain melalui jaringan menggunakan bahasa yang mereka berdua pahami.”

**2.13 Penelitian Terdahulu**

Merujuk pada penelitian sebelumnya yang relevan mengenai penerapan *Framework Flutter* pada aplikasi yang sudah banyak dikembangkan oleh para penelliti yang dijadikan refrensi dalam penulisan penelitian ini. Beberapa diantaranya dapat dilihat pada table 2.1 berikut ini :

**Tabel 2.1**

**Tinjauan Pustaka**

| No | Judul | Penulis | Publikasi | Ringkasan |
| --- | --- | --- | --- | --- |
| 1 | Penerapan Konsep *Marketplace* pada Bisnis *Laundry* dengan menggunakan *Framework Multiplatform Flutter* | Lauri, Satriany, Djoni Haryadi Setiabudi, dan Agustinus Noertjahyana | Jurnal Infra 8.2 (2020): 44-50. | Penelitian ini dilakukan untuk mengidentifikasi proses bisnis layanan jasa *laundry*, dan mengimplementasikan pada aplikasi berbasis mobile menggunakan *Framework Flutter*. Penelitian ini dimulai dengan mencari identifikasi masalah, studi literatur, pengumpulan data, lalu analisis kebutuhan, perancangan program hingga implementasi dan pengujian terhadap aplikasi yang dibuat. Hasil akhir dari penelitian adalah Aplikasi *e-commerce* Layanan Jasa Laundry Berbasis Mobile menggunakan *Framework Flutter* dapat digunakan oleh klien agar dapat melakukan proses bisnis layanan jasa *laundry* ini tanpa kendala ruang dan waktu. |
| 2 | Implementasi *Framework Flutter* untuk Aplikasi Penjualan Online Berbasis *Mobile* Menggunakan *Firebase* | Satriowibowo | Digital Library STMIK AKAKOM | Penelitian ini bertujuan untuk membuat Aplikasi *e-commerce* dengan menerapkan *Framewok Flutter* yang dapat membantu *user* untuk melihat barang dan melakukan pemesanan barang melalui aplikasi. Aplikasi ini juga dapat membantu penjual dalam melakukan pengolahan data pesanan. Hasil dari penelitian ini yaitu memudahkan dari segi waktu, pemasaran, dan segi kemudahan dalam berbelanja dengan *Framework Flutter* yang membangun *interface* lebih modern dan indah. |
| 3 | Pengembangan Aplikasi *E-Commerce* Di PT. Putra Sumber Abadi Menggunakan *Flutter* | Muhammad Hendriawan, dan T. Budiman | Jurnal Sistem Informasi, Informatika dan Komputasi 5.1 (2021): 68-88 | Penelitian ini bertujuan untuk membangun sistem *e-commerce* yang dapat mempermudah pemebeli untuk melakukan pemesanan barang tanpa harus datang langsung ke toko dan mempermudah dalam melakukan manajemen produk. Dengan penggunaan *Framework Flutter* aplikasi yang dihasilkan memiliki tampilan yang cukup menarik, dan dapat mempercepat pengolahan data. |
| 4 | Aplikasi Penjualan Tiket Kelas Pelatihan Berbasis *Mobile* menggunakan *Flutter* | FE Krisnada, R Tanone | Jurnal Teknik Informatika dan Sistem Informasi 5.3 (2019) | Penelitian ini bertujuan membangun aplikasi yang berguna untuk membantu dalam mencari dan mendaftar kelas pelatihan, dengan adanya aplikasi berbasis *mobile* dilengkapi dengan sistem yang dapat memudahlkan pengguna untuk memilih kelas yang menarik bagi pelanggan yaitu dengan sistem rekomendasi. |
| 5 | Penggunaan Teknologi *Flutter* dalam Aplikasi *Mobile* untuk Pengambangan Kedai Kopi | B Sudrajat | Jurnal Manajemen Informatika Komputer 6.1 (2021) | Penelitian ini bertujuan untuk membangun aplikasi mobile yang menjembatani antara pemilik kedai kopi dengan para penikmat kopi dalam transaksi dimasa pendemi. Aplikasi ini dapat membantu para pemilik kedai kopi utnuk memberikan informasi mengenai produk yang dijualnya juga promo serta diskon yang ada dan penikmat kopi tetap dapat menikmati kopi karena adanya kemudahan dalam mengkases aplikasi. |
| 6 | Membangun Aplikasi Mobile Berbasis *Flutter* Pada Startup Manawa Menggunakan Metode Waterfall | Aziz, Dendy Armandiaz, Rachmadita Andeswari, dan Soni Fajar Surya Gumilang | e-Proceeding of Engineering, Vol.7.2 | Penelitian ini bertujuan utnuk membanngun sebuah aplikasi mobile dengan *Framework Flutter* yang memiliki kelebihan yaitu kecepatan waktu dan pengembangan aplikasi pada Starup Manawa |

**2.18 Kerangka Pemikiran**

Kerangka berpikir sebagai model konseptual yang dimanfaatkan sebagai teori yang ada kaitannya dengan beberapa faktor yang diidentifikasi sebagai masalah penting. Konteks yang dimaksud untuk kerangka penelitian. Dalam menjalankan sebuah penelitian yang membutuhkan kerangka berpikir, alangkah lebih baiknya jika hal tersebut mampu menjelaskan secara teoritis. Sekaligus juga bisa menjelaskan hubungan antara variable yang diangkat. Jadi peneliti bisa menjelaskan hubungan antara variable independen & variable dependent.

**Penentuan permasalahan**

**Bagaimana menghasilkan sebuah *mobile apps* toko online yang dapat membantu mempermudah dalam penjualan produk pada Toko Highpom.**

**Pemecahan masalah**

**Menerapkan *Framework Flutter* sebagai alat bantu dalam merancang *mobile apps* toko online**

**Pengembangan sistem**

***Sofware contruction***

***Software* spesifikasi dan desain**

***UML, Black Box, Extreme Programing (XP)***

***MySQL, Framework Flutter***

**Menghasilkan sebuah *mobile apps* toko online Highpom dengan menerapkan *Framework Flutter* sebagai alat bantu dalam merancang *mobile apps*.**

**Gambar 2.11** Kerangka Pemikiran

Kerangka pemikiran yang penulis buat dimulai dari pemahaman terhadap permasalahan yang dialami ketika melakukan penelitian. Kemudian dari permasalahan yang didapat, penulis menjabarkan permasalah-permasalahan sesuai dengan metode yang tepat dalam penelitian dan pencapaian yang diinginkan. Adapun tahapan yang dilakukan yaitu merancang Toko Highpom menggunakan *Framework* Flutter dibantu dengan *library* pendukung. Pengujian perangkat lunak dilakukan dengan metode *Black-box* dengan perancangan menggunakan *Extreme Programming* (XP) dan hasil aplikasi yang dibuat mampu untuk memasarkan dan menjual produk berupa second brand pakaian serta meningkatkan penjualan toko harian, mingguan dan bulanan.