Pemanfaatan Flutter Pada Fitur Kenaikan Gaji Berkala Dalam Aplikasi mobile ASN Memayu (Studi Kasus CV. Atsoft Teknologi)

by Agung Wibowo

Submission date: 27-Nov-2021 12:45AM (UTC+0700)
Submission ID: 1712710508
File name: tanpa-nama-a4-18523230-paper.pdf (884.08K)
Word count: 2887
Character count: 18572

Pemanfaatan Flutter Pada Fitur Kenaikan Gaji Berkala Dalam Aplikasi *mobile* ASN Memayu

(Studi Kasus CV. Atsoft Teknologi)

***Abstract*—CV. Atsoft Teknologi (Atsoft) adalah perusahaan yang bergerak dalam bidang teknologi informasi (IT). Perusahaan ini mengembangkan teknologi informasi pada berbagai bidang, seperti: pendidikan, pemerintahap, industri, dan kesehatan. CV. Atsoft Teknologi menawarkan aplikasi ASN Memayu sebagai solusi kepada BKD DIY yang membutuhkan sebuah aplikasi untuk meningkatkan pelayanan serta pengelolaan manajemen kepegawaian BKD DIY. BKD DIY adalah lembaga kepegawaian yang berada di wilayah Daerah Istimewa Yogyakarta (DIY). Flutter dipilih sebagai *framework* yang akan mengembangkan aplikasi *mobile* untuk proyek ASN Memayu karena memiliki banyak manfaat bagi para developer. Flutter dapat mengembangkan aplikasi *mobile* android dan iOS hanya dengan satu basis kode saja. Flutter menggunakan Dart sebagai bahasa pemograman untuk mengembangkan aplikasinya. Tujuan dari penelitian ini adalah membantu meningkatkan pengalaman developer dalam mengembangkan aplikasi menggunakan Flutter.**

Keywords*—*Aplikasi ASN Memayu, Flutter

I. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi pada zaman sekarang sangatlah cepat dan pesat. Hal ini membuat manusia dapat menggunakan berbagai macam teknologi untuk membantu segala bentuk aktivitas sehari-harinya. Salah satu teknologinya adalah menggunakan teknologi *smartphone*. Dalam *smartphone* banyak sekali aplikasi yang dapat digunakan untuk membantu manusia. Adapun teknologinya meliputi berbagai macam aspek kehidupan yang salah satunya adalah bidang pendidikan. *Smartphone* dapat digunakan sebagai sumber belajar dan alat sarana komunikasi antara guru dengan muridnya sehingga memungkinkan untuk terjadinya tahap pembelajaran jarak jauh.

CV. Atsoft Teknologi (Atsoft) adalah perusahaan yang bergerak dalam bidang teknologi informasi (IT). Perusahaan ini mengembangkan teknologi informasi pada berbagai bidang, seperti: pendidikan, pemerintahap, industri, dan kesehatan. Perusahaan Atsoft memiliki 8 karyawan dimana 5 diantaranya bekerja sebagai *programmer* dan terdapat 3 karyawan lainnya yang bekerja sebagai pembicara maupun administrator. Setiap bulannya, perusahaan ini dapat mengerjakan hingga 10 proyek secara bersamaan.

ASN Memayu merupakan aplikasi yang telah ditawarkan oleh CV. Atsoft Teknologi kepada BKD DIY. BKD DIY adalah lembaga kepegawaian yang berada di wilayah Daerah Istimewa Yogyakarta (DIY). Lembaga kepegawaian ini merupakan lembaga yang berkembang dari lembaga kepegawaian yang telah ada. Lembaga ini telah diresmikan dan ditetapkan dengan Peraturan Daerah Nomor 11 Tahun 1960, mengenai susunan organisasi dan susunan pegawai instansi-instansi Pemerintah Provinsi DIY yang melalui lembaga Kantor Urusan Pegawai [1]. ASN Memayu adalah aplikasi yang dapat diakses melalui *website* dan *mobile*. Aplikasi ini bertujuan untuk meningkatkan pelayanan serta pengelolaan yang terintegrasi, akurat, *real time,* dan *user friendly* untuk manajemen kepegawaian BKD DIY. ASN Memayu diharapkan agar bermanfaat untuk dapat menjawab tuntutan dan tantangan perwujudan Aparatur Sipil Negara yang tidak hanya sekedar kerumunan pekerja kantoran, tetapi dapat menjadi insan peradaban yang sarat empati untuk melayani masyarakat [2].

Pengembangan pada aplikasi ASN Memayu berbasis *mobile* ini menjadi sangat kompleks. Developer harus dapat mengembangkan aplikasi *mobile* yang digunakan oleh pengguna dalam berbagai macam *platform*, seperti Android dan iOS. Developer juga dituntut harus bisa menyederhanakan aplikasi ASN Memayu menjadi bagian bagian kecil sehingga developer lain dapat mengembangkan aplikasi ini menjadi lebih mudah. Oleh karena itu, Developer dapat mengatasi permasalahan tersebut dengan cara memanfaatkan *framework* Flutter.

II. KAJIAN PUSTAKA

A. *Fitur Kenaikan Gaji Berkala*

Fitur Kenaikan Gaji Berkala adalah salah satu fitur yang ada pada Aplikasi *mobile* ASN Memayu. Fitur ini berfungsi untuk memudahkan pegawai dalam melihat data-data gajinya. Pada data-data tersebut, pegawai dapat mencari data-data gaji sesuai dengan kode golongannya. Fitur ini dibuat agar pegawai dapat mengecek data-data gajinya secara *real-time*.

B. *Flutter*

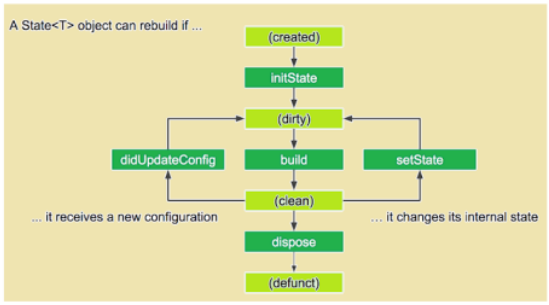
Flutter telah diresmikan sejak tahun 2015 oleh developer Dart yaitu Sky. Eric Seidel (Direktur untuk Flutter di Google). Flutter adalah *cross-platform framework* yang dibuat oleh Google untuk membangun berbagai aplikasi seperti android *mobile*, *iOS mobile,* web, dan *desktop*. *Framework* ini menggunakan *widgets* untuk membuat UI dan Dart sebagai bahasa pemograman untuk mengembangkan aplikasinya. *Widget* dapat diibaratkan dengan permainan Lego yang bisa menambahkan berbagai macam jenis bongkahan plastik kecil dan mengubah tampilan UI sesuai dengan yang diinginkan oleh developer. *Widget* adalah komponen UI yang membangun aplikasi Flutter. Setiap membuat UI dalam Flutter akan menciptakan berbagai macam pohon *widgets* [3]. Berikut beberapa informasi mengenai *widget*:

1. *Widget* digunakan untuk membuat UI, seperti: baris, kolom, *stack*, *card, form,* dan *padding*.

2. Pada Widget, developer dapat melakukan *styling*, seperti: tipe *font*, ukuran, berat, warna, batas, dan bayangan.
3. Widget dapat berupa kumpulan bentuk dan formulir.

Dalam *framework* Flutter terdapat fitur yang bernama *hot-reload*. Dengan fitur ini, developer membangun *User Interface (UI)*, menambahkan fitur, dan memperbaiki bug dapat menjadi lebih cepat dan mudah. *Hot-reload* bekerja dengan cara menanamkan kode *file* baru kedalam Dart Virtual Machine (VM). Setelah VM diperbaharui, kerangka kerja pada Flutter membuat ulang semua pohon *widget* secara otomatis. Hal ini membuat developer untuk melihat perubahan langsung pada UI aplikasi yang dikembangkannya [4].

Terdapat dua kombinasi *class* dalam mengembangkan aplikasi menggunakan *framework* Flutter, yaitu: *Stateless Widgets dan Statefull Widgets*. *Class* tersebut merupakan *widget* yang akan menggambarkan tampilan *User Interface* dengan membangun konstelasi *widget* lainnya agar menjadi lebih konkret. *Stateless Widget* hanya dapat menggambar satu kali saat *widget* dimuat sehingga tidak dapat menggunakan fitur *hot-reload* karena state tersebut sifatnya statis. Sebaliknya, *Statefull Widgets* dapat menggunakan *hot-reload* karena sifatnya yang dinamis. Dalam *Statefull Widgets* terdapat siklus hidup yang penting untuk diketahui oleh developer Flutter, dapat dilihat pada **Gambar 3.**



Gambar 3. Siklus Hidup *Statefull Widgets* (Sumber: https://www.developerlibs.com/2019/12/flutter-lifecycle-widgets.html)

Pada **Gambar 3** terdapat fungsi metode yang bisa dieksekusi ketika menggunakan *Statefull Widgets* [5].

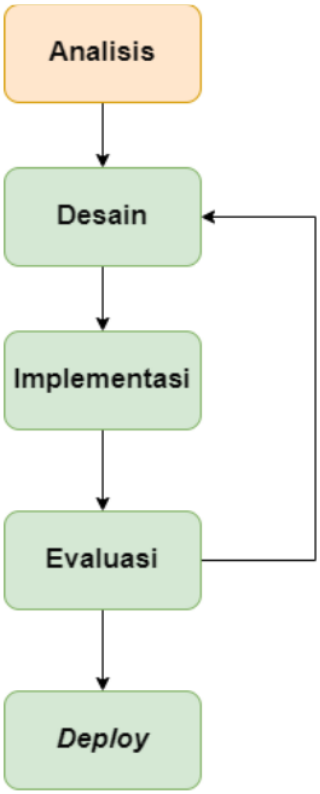
1. *createState:* Saat developer membuat *Statefull Widgets*, *framework* Flutter akan mengintruksikan metode *createState*. Metode ini mengembalikan sebuah *instance* yang dapat dilihat pada gambar sebelumnya.
2. *mounted(true/false):* Setelah membuat objek *state*, *framework* Flutter akan mengaitkannya dengan *Build Context* sebelum memanggil metode *initState*.
3. *initState:* metode yang pertama kali dipanggil ketika *Statefull Widgets* telah dibuat. Metode ini dipanggil hanya sekali. Developer dapat menginisialisasi data, properti, dan objek lainnya untuk mengubah data pada *widget* ini.
4. *didChangeDependencies:* Metode ini dipanggil setelah metode *initState* saat pertama kali *widgets* dibuat.
5. *build:* Metode ini menunjukan bagia dari *User Interface* yang diwakili oleh *widget*.
6. *didUpdateWidget:* Jika terdapat induk *widget* untuk mengubah konfigurasi dan harus membangun kembali *widget* yang ada. *Widget* tersebut akan dibangun bersamaan *runtimeType*. Setelah tahap ini, metode *didUpdateWidget* akan dipanggil untuk memperbaharui properti *widget* yang telah ada menjadi *widget* baru.
7. *setState:* Metode ini dipanggil dari *framework* oleh developer. Developer dapat mengubah status internal objek dan membuat perubahan dalam fungsi yang akan diteruskan ke metode ini. Metode *setState* akan memberitahu kerangka kerja bahwa terjadi perubahan yang akan mempengaruhi *User Interface*.
8. *deactive:* Metode ini dipanggil dari pohon *widget* yang mungkin dimasukkan kembali sebelum perubahan bingkai diselesaikan.
9. *dispose:* Metode ini dipanggil ketika objek *state* akan dihapus secara permanen.

C. *BLoC Pattern*

Business Logic Component design pattern (BloC Pattern) adalah *design pattern* yang membantu memisahkan antara *User Interface* dengan *business logic* sehingga komponen pada proyek dapat dibagi menjadi *presentational componen, BloC*, dan *backend*. Developer dapat fokus untuk mengkonversikan *event* menjadi *state* jika menggunakan *BloC Pattern*. Hal ini menjadikan developer lain dapat dengan mudah memahami kode yang telah dikembangkan sebelumnya. *BLoC Pattern* dapat membantu membuat aplikasi menjadi lebih kompleks yang tersusun dari komponen-komponen yang lebih kecil dan mudah di tes. Terdapat beberapa istilah yang penting jika developer melakukan penerapan *BloC* [6], yaitu:

1. *Event* merupakan *input* untuk *BLoC*.
2. *States* merupakan *output* dari *BloC*.
3. *Transitions* merupakan perubahan dari satu *state* ke *state* lain.
4. *Blocs* merupakan komponen yang akan mengonversi *Stream* dari *event* ke *Stream* yang akan mengubah *state*.

III. METODOLOGI



Gambar 4. Tahap Pembuatan Aplikasi

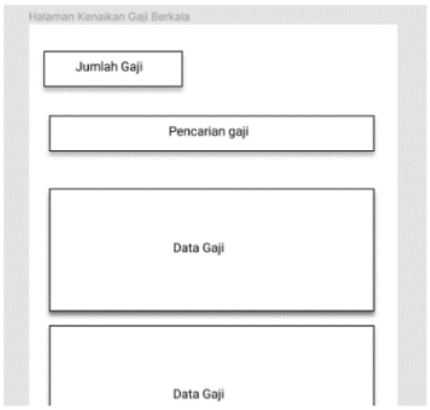
Pada **Gambar 4,** memperlihatkan metodologi yang digunakan pada makalah ini. Tahap metodologi tersebut dimulai dari Analisis hingga *Deploy*. Pada tahap Evaluasi, Tim Developer dapat kembali menuju tahap implementasi. Setelah semua sudah menyetujuinya, developer dapat melanjutkannya hingga tahap *Deploy*.

A. Analisis

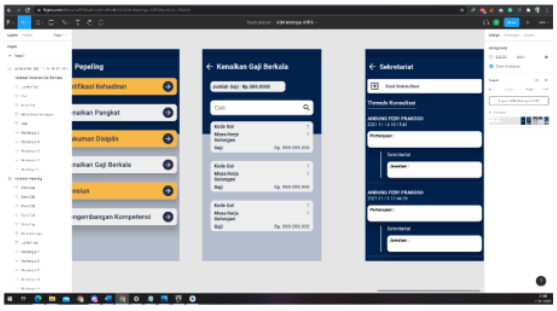
Analisis merupakan tahap yang dilakukan oleh Tim Developer yang berposisi sebagai *Project Manajer (PM)*. *PM* melakukan analisis untuk memenuhi semua kebutuhan klien. Setelah dilakukan analisis, *PM* tersebut dapat meminta kepada developer yang berposisi sebagai UI/UX untuk melanjutkan pengembangannya pada tahap *Desain*.

B. Desain

Desain merupakan tahap yang dibuat oleh Tim Developer yang berposisi sebagai UI/UX *Designer*. Pada tahap ini, *UI/UX Designer* melakukan pembuatan *wireframe*. *Wireframe* adalah gambaran kasar sebelum dijadikan desain. Terdapat contoh *wireframe* Halaman Kenaikan Gaji Berkala yang dapat dilihat pada **Gambar 5.** Dalam *wireframe* tersebut pegawai, terdapat Jumlah Gaji, Pencarian Gaji, dan Data Gaji. Dengan data tersebut, pegawai tidak perlu resah mengenai gaji-gaji selama bekerja di BKD DIY. Selanjutnya, UI/UX *Designer* membuat desain *mockup* yang akan diberikan kepada Tim Developer yang berposisi sebagai *FrontEnd Develoer*. UI/UX biasanya dalam membuat *mockup* menggunakan aplikasi Figma. Contoh *mockup* yang dibuat pada aplikasi Figma dapat dilihat pada **Gambar 6.**



Gambar 5. *Wireframe* Halaman Kenaikan Gaji Berkala



Gambar 6. *Mockup* ASN Memayu

C. Implementasi

Implementasi merupakan tahap yang dibuat oleh Tim Developer yang berposisi sebagai *FrontEnd Developer* dan *BackEnd Developer*. Pada tahap ini, *FrontEnd Developer* mengimplementasikan penggunaan *framework* Flutter untuk membuat aplikasi ASN Memayu berbasis *mobile*. Pada tahap ini juga, *BackEnd Developer* melakukan pembuatan REST API sebagai jembatan untuk komunikasi antara *FrontEnd* Developer dengan basis data pada *server*.

D. Evaluasi

Evaluasi merupakan tahapan yang akan dilakukan berkenaan dengan proses untuk menilai pekerjaan Tim Developer. Dalam tahapan Evaluasi, Tim Developer melakukan diskusi bersama klien. Jika semuanya sudah setuju dan dinilai bagus, Tim Developer dapat melanjutkan pekerjaannya ke tahapan *Deploy*. Sebaliknya, Tim Developer dapat kembali pada tahapan Desain jika ada yang perlu diperbaiki dan belum setuju. Pada tahapan ini juga, Tim Developer yang berposisi sebagai *FrontEnd* developer melakukan pengujian aplikasi secara manual dengan menggunakan *BlackBox testing*. Pengujian ini juga melibatkan pihak klien BKD DIY. *FrontEnd Developer* memberikan aplikasi dalam bentuk *file* yang dapat di *install* pada *smartphone* klien. Setelah pihak klien melakukan pengujian, pihak klien dapat memberikan masukkan kepada developer untuk kembali kepada tahap Desain. Setelah pihak

klien sudah puas dengan semua hasilnya, Tim Developer dapat melanjutkan ke tahap berikutnya, yaitu Deploy.

E. Deploy

Deploy merupakan tahapan untuk mempublikasikan sebuah aplikasi ke dalam aplikasi yang bernama *PlayStore* dan *AppStore*. Setelah aplikasi tersebut berhasil dipublikasikan, pengguna/pegawai dapat mengunduh aplikasi tersebut melalui aplikasi *PlayStore* dan *AppStore* yang ada di dalam *smartphone*-nya.

IV. PEMBAHASAN DAN HASIL

A. Kode Pembahasan Pembuatan Fitur Kenaikan Gaji Berkala menggunakan Flutter

1. Membuat Halaman Kenaikan Gaji Berkala.

Halaman Kenaikan Gaji Berkala dibuat menggunakan *class Statefull Widget*. *Statefull Widget* dapat membuat antar muka pengguna menjadi dinamis karena perlu memanggil token pengguna yang ada pada *database local*. **Gambar 7** adalah sebagian kode menggunakan *Statefull Widget*.

```
import 'package:shared_preferences/shared_preferences.dart';

class PageKenaikanGajiBerkala extends StatefulWidget {
  @override
  _PageKenaikanGajiBerkalaState createState() =>
      _PageKenaikanGajiBerkalaState();
}

class _PageKenaikanGajiBerkalaState extends State<PageKenaikanGajiBerkala> {
  String tokenLogged;
```

Gambar 7. Sebagian kode *Statefull Widget*.

2. Membuat *widgets* TopBar atau AppBar

Dalam *widgets* ini terdapat fitur yang berfungsi agar pegawai dapat mudah untuk kembali ke halaman sebelumnya. **Gambar 8** adalah kode untuk menampilkan *widget TopBar*.

```
child: Row(
  children: <Widget>[
    Container(
      child: Material(
        color: Colors.transparent,
        child: InkWell(
          onTap: () {
            Navigator.pop(context);
          },
          child: Icon(
            Icons.arrow_back,
            size: ResponsiveFlutter.of(context).fontSize(4),
            color: Colors.white,
          ),  // Icon
        ),  // InkWell
      ),  // Material
    ),  // Container
    Container(
        margin: EdgeInsets.only(
            left: ResponsiveFlutter.of(context).wp(2)),  // Edge
        child: Text(
          "Kenaikan Gaji Berkala",
          style: TextStyle(
              color: Colors.white,
              fontSize:
                  ResponsiveFlutter.of(context).fontSize(3),
              fontWeight: FontWeight.bold),  // TextStyle
        )),  // Text, Container
  ],  // <Widget>[]
),  // Row
```

Gambar 8. *Widget TopBar*

3. Membuat *widgets* jumlah gaji, pencarian, dan Kenaikan Gaji.

Widgets ini dapat dibuat terpisah dari kode Halaman Kenaikan Gaji Berkala. Setelah semua *widget* sudah dibuat, developer dapat memanggil *widgets* ini di Halaman Kenaikan Gaji Berkala. **Gambar 9** adalah kode pemanggilan *widget* Kenaikan Gaji Berkala. **Gambar 10** adalah sebagian kode *widget* Kenaikan Gaji Berkala. *Widget* Kenaikan Gaji berkala dibuat menggunakan *Stateless Widget* agar mengurangi pengunaan memori pada *smartphone*.

```
Container(
  child: WidgetKenaikanGaji(
    kenaikanGajiDataModel:
        kenaikanGajiBlocLoaded
                .listKenaikanGajiDataModel[
            index],
  ),  // WidgetKenaikanGaji
),  // Container
```

Gambar 9. Pemanggilan *Widget* Kenaikan Gaji

```
import 'package:flutter/material.dart';
import 'package:memayu/model/kenaikan_gaji_data_model.dart';
import 'package:pecahan_rupiah/pecahan_rupiah.dart';
import 'package:responsive_flutter/responsive_flutter.dart';

class WidgetKenaikanGaji extends StatelessWidget {
  KenaikanGajiDataModel kenaikanGajiDataModel;

  WidgetKenaikanGaji({this.kenaikanGajiDataModel});

  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
```

Gambar 10. *Widget* Kenaikan Gaji

4. Pembuatan fungsi untuk memanggil data Kenaikan Gaji dari *server*

Pengambilan data dilakukan menggunakan *library* HTTP Request. Pengambilan data ini merupakan fungsi yang bernama getKenaikanGaji. Terdapat dua parameter dalam memanggil fungsi ini, yaitu: Token dan Search. Pemanggilan fungsi tersebut akan mendapatkan kembalian data yang akan diteruskan kepada Model. Model merupakan salah satu contoh konsep dalam *Object Oriented Programming* (OOP). Dalam *mobile*, developer sering menggunakan konsep *Model View ViewModel* (MVVM). Pada Flutter ini, konsep MVVM diubah menjadi *Model View BloC pattern*. **Gambar 11** adalah fungsi yang dapat melakukan pemanggilan data dari *server*.

```dart
static Future<List<KenaikanGajiDataModel>> getKenaikanGaji(String token,
    {String search}) async {
  var response;

  if (search != null) {
    print("kodel gol tidak ada");
    response = await http.post(
        "${Constants.simpegApi}tbl_gaji_pp152019/get_list",
        body: jsonEncode(
            <String, String>{"token": "$token", "KODE_GOL": "$search"}));
  } else {
    response = await http.post(
        "${Constants.simpegApi}tbl_gaji_pp152019/get_list",
        body: jsonEncode(<String, String>{"token": "$token"}));
  }

  var jsonObject = json.decode(response.body);
  var data = (jsonObject as Map<String, dynamic>)['data'] as List;

  return data
      .map<KenaikanGajiDataModel>(
          (item) => KenaikanGajiDataModel.fromJson(item))
      .toList();
}
```

Gambar 11. Pemanggilan data dari *server*

5. Pembuatan *BLoC*

Dalam pembuatan *BloC*, developer perlu mengetahui konsep penerapan *BloC*. Proses pertama, developer harus membuat *class Event* yang dapat dilihat pada **Gambar 12.** Selanjutnya, developer membuat *classes States* yang dapat dilihat pada **Gambar 13.** Terakhir, developer perlu membuat *classes Bloc* yang dapat dilihat pada **Gambar 14.** Developer perlu mendaftarkan *BloC* ke dalam *widget MultiBlocProviders*. *Widget* ini merupakan *library* yang sudah disediakan oleh Flutter untuk melakukan inisialisasi *BloC* ke aplikasi. Aplikasi akan dapat mengirim ke berbagai antarmuka pengguna untuk memperbaharui *widgets*. Pemanggilan w*idget MultiBlocProviders* dapat dilihat pada **Gambar 15.**

```dart
class KenaikanGajiBlocEvent {
  String token;
  String search;

  KenaikanGajiBlocEvent({this.token, this.search});
}
```

Gambar 12. *Class Event*

```dart
class KenaikanGajiBlockState {}

class KenaikanGajiBlocLoading extends KenaikanGajiBlockState {}

class KenaikanGajiBlocError extends KenaikanGajiBlockState {}

class KenaikanGajiBlocUninitialized extends KenaikanGajiBlockState {}

class KenaikanGajiBlocLoaded extends KenaikanGajiBlockState {
  List<KenaikanGajiDataModel> listKenaikanGajiDataModel;

  KenaikanGajiBlocLoaded({this.listKenaikanGajiDataModel});

  KenaikanGajiBlocLoaded copyWith(
      {List<KenaikanGajiBlocLoaded> listKenaikanGajiDataModel}) {
    return KenaikanGajiBlocLoaded(
        listKenaikanGajiDataModel:
            listKenaikanGajiDataModel ?? this.listKenaikanGajiDataModel);
  }
}
```

Gambar 13. *classes States*

```dart
class KenaikanGajiBloc extends Bloc<KenaikanGajiBlocEvent, KenaikanGajiBlockState> {
  @override
  KenaikanGajiBlockState get initialState => KenaikanGajiBlocUninitialized();

  @override
  Stream<KenaikanGajiBlockState> mapEventToState(
      KenaikanGajiBlocEvent event) async* {
    List<KenaikanGajiDataModel> listKenaikanGajiDataModel;

    try {
      if (state is KenaikanGajiBlocUninitialized) {
        print("state uninit");
        listKenaikanGajiDataModel =
            await KenaikanGajiServices.getKenaikanGaji(event.token);
        yield KenaikanGajiBlocLoaded(
            listKenaikanGajiDataModel: listKenaikanGajiDataModel);
      }

      if (event.search != null) {
        print("state search");
        yield KenaikanGajiBlocLoading();
        listKenaikanGajiDataModel = await KenaikanGajiServices.getKenaikanGaji(
            event.token,
            search: event.search);
        yield KenaikanGajiBlocLoaded(
            listKenaikanGajiDataModel: listKenaikanGajiDataModel);
      }
    } catch (_) {
      print("gagal");
      yield KenaikanGajiBlocError();
    }
  }
}
```

Gambar 14. *class Blocs*

```dart
return MultiBlocProvider(
  providers: [
    BlocProvider<AuthCekBloc>(create: (context) => AuthCekBloc()),
```

Gambar 15. Pemanggilan *Widget MultiBlocProvider*

6. Pemanggilan *BLoC* di Halaman Kenaikan Gaji Berkala

```dart
  kenaikanGajiBloc = BlocProvider.of<KenaikanGajiBloc>(context);
```

Gambar 16. Pemanggilan *BLoC*

Gambar 16 adalah pemanggilan *BLoC* yang sudah didaftarkan pada *widget MultiBlocProviders*. Setelah itu, developer perlu memasukkan *input* ke dalam *class Event*, dapat dilihat pada **Gambar 17.**

```dart
kenaikanGajiBlocEvent = KenaikanGajiBlocEvent();
kenaikanGajiBlocEvent.token = tokenLogged;
kenaikanGajiBloc.add(kenaikanGajiBlocEvent);
```

Gambar 17. *input*-an ke dalam *Event*

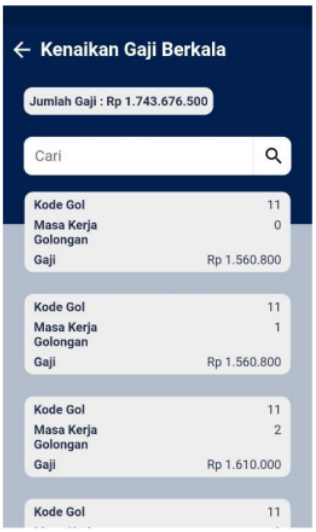
Setelah itu, developer perlu memanggil *widget BlocBuilder*. *Widget* ini berfungsi untul melakukan pengecekan jika terdapat *state* yang berubah. Contoh pemakaian atau penggunaan sebagian *widget BlocBuilder* dapat dilihat pada **Gambar 18.** Pada *widget* BlocBuilder, developer dapat membuat *logic* untuk luaran *User Interface (UI)* pengguna. Sebagai contoh, pada **Gambar 18** terdapat *logic state* KenaikanGajiBlocUninitialized maka *output*-nya adalah *widget* CircularProgressIndicator.

```
BlocBuilder<KenaikanGajiBloc, KenaikanGajiBlocState>(
  builder: (context, state) {
    if (state is KenaikanGajiBlocUninitialized) {
      print("1");
      return Expanded(
        child: Container(
          margin: EdgeInsets.only(
              bottom: ResponsiveFlutter.of(context).hp(30)),  // Edg
          child: Center(
            child: SizedBox(
              width: 30,
              height: 30,
              child: CircularProgressIndicator(),
            ),  // SizedBox
          ),  // Center
        ),  // Container
      );  // Expanded
    }
    if (state is KenaikanGajiBlocLoading) {
      print("2");
```

Gambar 18. *widget BlocBuilder*

B. Hasil Dari Kode Pembahasan Pada Pembuatan Fitur Kenaikan Gaji Berkala menggunakan Flutter

Fitur Kenaikan Gaji Berkala adalah salah satu fitur yang ada di Halaman Kenaikan Gaji Berkala pada aplikasi ASN Memayu, dapat dilihat pada **Gambar 19**.



Gambar 19. Halaman Kenaikan Gaji Berkala

Data yang ada pada halaman tersebut adalah Jumlah Gaji dan *list* data gaji. Jumlah gaji berfungsi untuk menjumlahkan semua gaji yang ada. Jumlah Gaji dapat dilihat pada **Gambar 20.**

Gambar 20. *User Interface Widget* Jumlah Gaji

Data gaji tersebut adalah Kode Golongan, Masa Kerja Golongan, dan Gaji. Pegawai dapat melakukan *vertical scrolling* pada data gaji tersebut. Data gaji dapat dilihat pada **Gambar 21.**

Gambar 21. *User Interface Widget* Data Gaji

Dengan fitur Kenaikan Gaji Berkala, pegawai dapat dengan mudah melihat informasi mengenai jumlah keseluruhan gaji dan gaji-gajinya. Pada Halaman Kenaikan Gaji Berkala terdapat 2 (dua) tombol, yaitu tombol pencarian dan tombol kembali.

1. Tombol pencarian. Tombol ini berfungsi sebagai tombol yang dapat mencari data-data gaji pegawai. Pegawai harus memasukkan data Kode Golongan kedalam *input* pencarian tersebut. Pencarian dapat dilihat pada **Gambar 22.** Setelah mengisi data tersebut, pegawai dapat menekan tombol pencarian agar data-data gaji pegawai berubah sesuai dengan Kode Golongannya. Tombol pencarian dapat dilihat pada **Gambar 23.**

Gambar 22. *User Interface Widget* Pencarian

Gambar 23. *User Interface Widget* Tombol Pencarian

2. Tombol kembali. Tombol ini berfungsi sebagai tombol yang akan mengarahkan pegawai ke halaman sebelumnya. Tombol tersebut dapat dilihat pada **Gambar 24.**

Gambar 24 *User Interface Widget* Tombol Kembali

C. Hasil Dari Pengujian Menggunakan BlackBox Testing

No.	Fungsional yang diuji	Hasil yang diharapkan	Hasil pengujian
1.	Login berhasil	Masuk ke Halaman Menu Utama ASN Memayu	Masuk ke Halaman Menu Utama ASN Memayu
2.	Login gagal	Menampilkan *snackbar "Username* dan *password* salah"	Menampilkan *snackbar "Username* dan *password* salah"
3.	Menampilkan data pribadi berhasil	Menampilkan data-data pribadi pada Halaman Data Pribadi	Menampilkan data-data pribadi pada Halaman Data Pribadi

4.	Menampilkan data pribadi gagal	Menampilkan data "Tidak ada data pribadi" pada Halaman Data Pribadi	Menampilkan "Tidak ada data pribadi" pada Halaman Data Pribadi
5.	Menampilkan data data gaji berhasil	Menampilkan data-data gaji pada Halaman Kenaikan Gaji Berkala	Menampilkan data-data gaji pada Halaman Kenaikan Gaji Berkala
6.	Menampilkan data data gaji gagal	Menampilkan data "Tidak ada data gaji" pada Halaman Kenaikan Gaji Berkala	Menampilkan data "Tidak ada data gaji" pada Halaman Kenaikan Gaji Berkala
7.	Mencari data-data gaji dengan menggunakan data kode golongan "12"	Hanya menampilkan data-data gaji yang kode golongannya 12 pada Halaman Kenaikan Gaji Berkala	Hanya menampilkan data-data gaji yang kode golongannya 12 pada Halaman Kenaikan Gaji Berkala
8.	Menampilkan data-data laporan yang salah dan belum diperbaiki oleh pihak admin ASN Memayu Berhasil	Data-data berhasil ditampilkan pada Halaman Ngecek Data	Data-data berhasil ditampilkan pada Halaman Ngecek Data
9	Menampilkan data-data laporan yang salah dan belum diperbaiki oleh pihak admin ASN Memayu gagal	Menampilkan data "Tidak ada data" pada Halaman Ngecek Data	Menampilkan data "Tidak ada data" pada Halaman Ngecek Data
10.	Menambahkan laporan data yang salah pada Halaman Tambah Koreksi dengan mengisi *input* "Data apa yang keliru" dan "Seharusnya" berhasil	Menampilkan *snackbar* "Berhasil" pada halaman Ngecek Data	Menampilkan *snackbar* "Berhasil" pada halaman Ngecek Data
11.	Menambahkan laporan data yang salah pada Halaman Tambah Koreksi dengan mengisi *input* "Data apa yang keliru" dan "Seharusnya" gagal	Menampilkan *snackbar* "Gagal" pada halaman Ngecek Data	Menampilkan *snackbar* "Gagal" pada halaman Ngecek Data

V. KESIMPULAN

Berdasarkan pengembangan aplikasi *mobile* ASN Memayu pada fitur Kenaikan Gaji Berkala dengan memanfaatkan *framework* Flutter, dapat disimpulkan bahwa:

1. Flutter dapat membuat *User Interface* pengguna menjadi interaktif.
2. *Widgets* dan *BLoC* pada Flutter dapat menyederhanakan pengembangan aplikasi menjadi bagian-bagian kecil dan lebih kompleks sehingga pemeliharaannya menjadi lebih mudah.
3. Fitur *hot-reload* pada Flutter dapat mempercepat pengembangan aplikasinya. Oleh karena itu, developer tidak perlu melakukan *run*/*debug* ulang sebuah aplikasi yang memungkinkan dapat melihat perubahaan *User Interface* secara langsung.
4. Flutter dapat membuat aplikasi pada banyak *platform* hanya dengan menggunakan satu *codebase* saja.

REFERENCES

[1] BKD D.I. Yogyakarta. Sejarah (2021), Visi, dan Misi BKD D.I. Yogyakarta. Diakses pada 22 November 2021, dari http://bkd.jogjaprov.go.id/profil/gambaran-umum-visi-dan-misi

[2] Humas Pemda DIY (2021). Gubernur DIY Luncurkan Logo dan Rebranding Simpeg "ASN Memayu". Diakses pada 22 November 2021, dari https://jogjaprov.go.id/berita/detail/9718-gubernur-diy-luncurkan-logo-dan-rebranding-simpeg-asn-memayu

[3] John Wiley & Sons, Inc (2020). INTRODUCING FLUTTER. Indianapolis, Indiana in Canada

[4] Flutter-dev@ (2021). Hot reload. Diakses pada 22 November 2021, dari https://docs.flutter.dev/development/tools/hot-reload

[5] Developer Libs (2019). Flutter – Lifecycle of Widgets. Diakses pada 22 November 2021, dari https://www.developerlibs.com/2019/12/flutter-lifecycle-widgets.html

[6] Nurul Faza (2019). BLoC Pattern pada Flutter. Diakses pada 22 November 2021, dari https://medium.com/learnfazz/bloc-pattern-pada-flutter-a13523af8484.

Pemanfaatan Flutter Pada Fitur Kenaikan Gaji Berkala Dalam Aplikasi mobile ASN Memayu (Studi Kasus CV. Atsoft Teknologi)

ORIGINALITY REPORT

7%	6%	1%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	medium.com Internet Source	2%
2	bkd.jogjaprov.go.id Internet Source	1%
3	www.developerlibs.com Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	eprints.ums.ac.id Internet Source	<1%
6	text-id.123dok.com Internet Source	<1%
7	repository.uib.ac.id Internet Source	<1%
8	Michael Agustav, Kathryn Widhiyanti, Edwin Meinardi Trianto. "Perancangan dan Pembuatan Aplikasi Pembelajaran Bahasa	<1%

Jepang Untuk Pemula Dengan Metode User Centered Design Berbasis Android", Teknika, 2017
Publication

9	core.ac.uk Internet Source	<1 %
10	repository.uin-suska.ac.id Internet Source	<1 %
11	www.rafsablog.id Internet Source	<1 %
12	Irwin Supriadi, Erwin Yulianto. "PEMBANGUNAN KIOS INFORMASI FASILITAS KEPEGAWAIAN BERBASIS MULTIMEDIA INTERAKTIF MENGGUNAKAN INDUSTRIAL PC", Jurnal Tekno Insentif, 2019 Publication	<1 %
13	doku.pub Internet Source	<1 %
14	Monica Widiasri, Lisana Lisana, Suwendi Liantara. "Motion Comic Dongeng Agama Buddha Untuk Anak Umur 4-6 Tahun", Teknika, 2018 Publication	<1 %

Exclude quotes On Exclude matches Off

Exclude bibliography On