**Lampiran Listing Program**

1. main.dart

|  |
| --- |
| import 'package:firebase\_core/firebase\_core.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_boc/flutter\_bloc.dart';  import 'package:healio/blocs/article/article\_bloc.dart';  import 'package:healio/blocs/auth/auth\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:healio/blocs/heart\_disease/heart\_disease\_bloc.dart';  import 'package:healio/blocs/page/page\_cubit.dart';  import 'package:healio/blocs/user/user\_bloc.dart';  import 'package:healio/pages/all\_blood\_oxygen\_page.dart';  import 'package:healio/pages/all\_heart\_rate\_page.dart';  import 'package:healio/pages/article\_page.dart';  import 'package:healio/pages/blood\_oxygen\_information\_page.dart';  import 'package:healio/pages/blood\_oxygen\_page.dart';  import 'package:healio/pages/heart\_disease\_check\_page.dart';  import 'package:healio/pages/heart\_rate\_information\_page.dart';  import 'package:healio/pages/heart\_rate\_page.dart';  import 'package:healio/pages/main\_page.dart';  import 'package:healio/pages/sign\_in\_page.dart';  import 'package:healio/pages/sign\_up\_page.dart';  import 'package:healio/pages/splash\_page.dart';  void main() async {    WidgetsFlutterBinding.ensureInitialized();    await Firebase.initializeApp();    runApp(const MyApp());  }  class MyApp extends StatelessWidget {    const MyApp({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      return MultiBlocProvider(        providers: [          BlocProvider(            create: (context) => PageCubit(),          ),          BlocProvider(            create: (context) => HealthBloc(),          ),          BlocProvider(            create: (context) => ArticleBloc(),          ),          BlocProvider(            create: (context) => AuthBloc(),          ),          BlocProvider(            create: (context) => UserBloc(),          ),          BlocProvider(            create: (context) => HeartDiseaseBloc(),          ),        ],        child: MaterialApp(          debugShowCheckedModeBanner: false,          routes: {            '/': (context) => const SplashPage(),            '/main': (context) => const MainPage(),            '/sign-in': (context) => const SignInPage(),            '/sign-up': (context) => const SignUpPage(),            '/article': (context) => const ArticlePage(),            '/heart-rate': (context) => const HeartRatePage(),            '/all-heart-rate': (context) => const AllHeartRatePage(),            '/all-blood-oxygen': (context) => const AllBloodOxygenPage(),            '/heart-rate-information': (context) =>                const HeartRateInformationPage(),            '/blood-oxygen': (context) => const BloodOxygenPage(),            '/blood-oxygen-information': (context) =>                const BloodOxygenInformationPage(),            '/heart-disease-check': (context) => const HeartDiseaseCheckPage(),          },        ),      );    }  } |

1. all\_blood\_oxygen\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import 'package:healio/widget/health\_data\_item.dart';  import 'package:health/health.dart';  class AllBloodOxygenPage extends StatefulWidget {    const AllBloodOxygenPage({Key? key}) : super(key: key);    @override    State<AllBloodOxygenPage> createState() => \_AllBloodOxygenPageState();  }  class \_AllBloodOxygenPageState extends State<AllBloodOxygenPage> {    @override    void initState() {      super.initState();      context.read<HealthBloc>().add(HealthGetAllBloodOxygen());    }    getValueFromString(HealthValue healthValue) {      List<String> value = healthValue.toString().split('.');      return int.parse(value[0]);    }    @override    Widget build(BuildContext context) {      Widget header() {        return const CustomAppBar(          title: 'Saturasi Oksigen',        );      }      Widget body() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthAllBloodOxygenSuccess) {              return Container(                margin: EdgeInsets.only(                  left: defaultMargin,                  right: defaultMargin,                  bottom: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Data Semua Saturasi Oksigen',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    Column(                      children: state.allBloodOxygen                          .map((health) => HealthDataItem(                                health,                                isAll: true,                                category: 'blood\_oxygen',                              ))                          .toList(),                    ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. all\_heart\_rate\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import 'package:healio/widget/health\_data\_item.dart';  import 'package:health/health.dart';  class AllHeartRatePage extends StatefulWidget {    const AllHeartRatePage({Key? key}) : super(key: key);    @override    State<AllHeartRatePage> createState() => \_AllHeartRatePageState();  }  class \_AllHeartRatePageState extends State<AllHeartRatePage> {    @override    void initState() {      super.initState();      context.read<HealthBloc>().add(HealthGetHeartRate());      context.read<HealthBloc>().add(HealthGetAllHeartRate());    }    getValueFromString(HealthValue healthValue) {      List<String> value = healthValue.toString().split('.');      return int.parse(value[0]);    }    @override    Widget build(BuildContext context) {      Widget header() {        return const CustomAppBar(          title: 'Denyut Jantung',        );      }      Widget body() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthAllHeartRateSuccess) {              return Container(                margin: EdgeInsets.only(                  left: defaultMargin,                  right: defaultMargin,                  bottom: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Data Semua Denyut Jantung',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    Column(                      children: state.allHeartRate                          .map((health) => HealthDataItem(                                health,                                isAll: true,                              ))                          .toList(),                    ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. article\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/models/article\_model.dart';  import 'package:healio/pages/webview\_page.dart';  import 'package:healio/widget/article\_card.dart';  import '../blocs/article/article\_bloc.dart';  import '../shared/theme.dart';  class ArticlePage extends StatelessWidget {    const ArticlePage({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          padding: const EdgeInsets.all(24),          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(8),                  decoration: BoxDecoration(                    color: kLightGrayColor,                    borderRadius: BorderRadius.circular(                      12,                    ),                  ),                  margin: const EdgeInsets.only(right: 10),                  child: const Icon(                    Icons.chevron\_left,                    size: 24,                  ),                ),              ),              Text(                'Artikel',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),            ],          ),        );      }      Widget article() {        return Container(          margin: const EdgeInsets.only(            left: 24,            right: 24,            bottom: 24,          ),          child: Column(            children: listArticle.map((article) {              return ArticleCard(article);            }).toList(),          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            article(),          ],        ),      );    }  } |

1. blood\_oxygen\_information\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import '../shared/theme.dart';  class BloodOxygenInformationPage extends StatelessWidget {    const BloodOxygenInformationPage({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      Widget header() {        return const CustomAppBar(          title: 'Tentang Saturasi Oksigen',        );      }      Widget content() {        return Container(          margin: EdgeInsets.only(            right: defaultMargin,            left: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              ClipRRect(                borderRadius: BorderRadius.circular(12),                child: Image.asset(                  'assets/bo\_thumbnail.jpg',                ),              ),              SizedBox(                height: defaultMargin,              ),              Text(                'Saturasi Oksigen',                style: blackTextStyle.copyWith(                  fontSize: 18,                  fontWeight: semiBold,                ),              ),              const SizedBox(                height: 10,              ),              Text(                'Saturasi oksigen merupakan nilai yang menunjukkan kadar oksigen di dalam darah. Nilai ini sangat berpengaruh terhadap berbagai fungsi organ dan jaringan tubuh. Pengukuran nilai saturasi oksigen dapat dilakukan dengan 2 cara, yakni dengan analisis gas darah (AGD) atau menggunakan alat oximeter.\n\nHasil pengukuran saturasi oksigen yang dilakukan dengan analisis gas darah ditunjukkan dengan istilah PaO2 (tekanan parsial oksigen). Sementara itu, hasil pengukuran saturasi oksigen dengan menggunakan oximeter ditunjukkan dengan istilah SpO2.',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              SizedBox(                height: defaultMargin,              ),              Text(                'Saturasi Oksigen Normal',                style: blackTextStyle.copyWith(                  fontSize: 18,                  fontWeight: bold,                ),              ),              const SizedBox(                height: 10,              ),              Text(                'Berikut adalah nilai saturasi oksigen normal pada orang dengan kondisi paru-paru yang sehat atau tidak memiliki kondisi medis tertentu:',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              Text(                '     •  Analisis gas darah (PaO2): 80–100 mmHg\n     •  Oximeter (SpO2): 95–100%',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              SizedBox(height: defaultMargin),              Text(                'Saturasi Oksigen Rendah',                style: blackTextStyle.copyWith(                  fontSize: 18,                  fontWeight: bold,                ),              ),              const SizedBox(                height: 10,              ),              Text(                'Berikut adalah kriteria nilai saturasi oksigen rendah atau di bawah normal:',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              Text(                '     •  Analisis gas darah (PaO2): dibawah 80 mmHg\n     •  Oximeter (SpO2): dibawah 95%\n\nOrang yang memiliki saturasi oksigen rendah atau hipoksemia bisa merasakan berbagai gejala, seperti nyeri dada, sesak napas, batuk, sakit kepala, detak jantung cepat, kebingungan, dan kulit membiru.\n\nKendati demikian, orang yang mengalami hipoksemia juga bisa tidak merasakan gejala apa pun. Kondisi ini yang disebut dengan happy hypoxia ini bisa terjadi pasien COVID-19.\n\nHipoksemia, baik yang menimbulkan gejala maupun tidak, bisa menganggu kerja organ dan jaringan tubuh. Bila dibiarkan, hal ini dapat menyebabkan kerusakan pada organ vital, seperti jantung, otak, dan ginjal, dan berisiko menyebabkan komplikasi yang berbahaya.',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              SizedBox(                height: defaultMargin,              ),              const Divider(),            ],          ),        );      }      Widget disclaimer() {        return Container(          margin: EdgeInsets.all(defaultMargin),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Row(                crossAxisAlignment: CrossAxisAlignment.center,                children: [                  Icon(                    Icons.warning,                    size: 16,                    color: kRedColor,                  ),                  const SizedBox(                    width: 5,                  ),                  Text(                    'Disclaimer',                    style: blackTextStyle.copyWith(                      fontSize: 18,                      fontWeight: bold,                    ),                  ),                ],              ),              const SizedBox(                height: 10,              ),              Text(                'Harap diingat bahwa pemantauan saturasi oksigen hanya untuk kebugaran dan kesehatan, bukan untuk diagnosis atau perawatan kondisi medis apapun. Jika Anda memiliki kekhawatiran tentang jantung Anda, pastikan untuk berkonsultasi dengan profesional medis.',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            content(),            disclaimer(),          ],        ),      );    }  } |

1. blood\_oxygen\_page.dart

|  |
| --- |
| import 'dart:developer';  import 'dart:math' as math;  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import 'package:healio/widget/health\_data\_item.dart';  import 'package:healio/widget/more\_button.dart';  import 'package:health/health.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../models/health\_model.dart';  import '../services/health\_service.dart';  import '../services/push\_notification\_service.dart';  import '../shared/shared\_methods.dart';  class BloodOxygenPage extends StatefulWidget {    const BloodOxygenPage({Key? key}) : super(key: key);    @override    State<BloodOxygenPage> createState() => \_BloodOxygenPageState();  }  late HealthModel boFromFirebase;  List<HealthDataPoint> dataBloodOxygen = [];  int age = 0;  String category = '';  class \_BloodOxygenPageState extends State<BloodOxygenPage> {    @override    void initState() {      super.initState();      getDataFromFirebase();      fetchData();      getAge();    }    void fetchData() {      context.read<HealthBloc>().add(HealthGetBloodOxygen());      final healthState = context.read<HealthBloc>().state;      if (healthState is HealthBloodOxygenSuccess) {        dataBloodOxygen = healthState.bloodOxygen;      }    }    void getAge() {      final authState = context.read<AuthBloc>().state;      if (authState is AuthSuccess) {        age = calculateAge(DateTime.parse(authState.user.dateOfBirth));      }    }    void getDataFromFirebase() async {      boFromFirebase = await HealthService().getBloodOxygenFromFirebase();    }    void updateBloodOxygen() async {      if (boFromFirebase.value ==              getValueFromHealthValue(dataBloodOxygen.first.value) &&          boFromFirebase.dateFrom == dataBloodOxygen.first.dateFrom.toString()) {        log('Sama');        log(age.toString());      } else {        log('beda');        await HealthService().updateBloodOxygenFromFirebase(            getValueFromHealthValue(dataBloodOxygen.first.value),            dataBloodOxygen.first.dateFrom.toString());        category = setCategory(          getValueFromHealthValue(dataBloodOxygen.first.value),          'blood\_oxygen',          age,        );        log(category);        if (category == 'rendah') {          customShowSnackBar(context, 'Saturasi oksigen kamu rendah');          await PushNotificationService().pushNotification(            'Saturasi Oksigen ${getValueFromHealthValue(dataBloodOxygen.first.value)}',            'Saturasi Oksigen kamu dibawah nilai normal nih :(.',          );        }      }    }    syncData() async {      fetchData();      getDataFromFirebase();      updateBloodOxygen();    }    @override    Widget build(BuildContext context) {      Widget header() {        return CustomAppBar(          title: 'Saturasi Oksigen',          action: GestureDetector(            onTap: () {              syncData();            },            child: Container(              padding: const EdgeInsets.all(10),              decoration: BoxDecoration(                color: kLightGrayColor,                borderRadius: BorderRadius.circular(                  defaultRadius,                ),              ),              child: Image.asset(                'assets/fi\_refresh-cw.png',                height: 20,              ),            ),          ),        );      }      Widget report() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthBloodOxygenSuccess) {              List<int> listValue = state.bloodOxygen                  .map((e) => getValueFromHealthValue(e.value))                  .toList();              return Container(                margin: EdgeInsets.symmetric(                  horizontal: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Ringkasan',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    const SizedBox(height: 10),                    Container(                      padding: const EdgeInsets.all(16),                      decoration: BoxDecoration(                        color: kLightGrayColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: Column(                        crossAxisAlignment: CrossAxisAlignment.start,                        children: [                          Text(                            'Rabu, 12 Juli',                            style: grayTextStyle,                          ),                          const SizedBox(height: 24),                          Column(                            children: [                              Container(                                height: 56,                                width: 56,                                decoration: BoxDecoration(                                  color: kWhiteColor,                                  shape: BoxShape.circle,                                ),                                padding: const EdgeInsets.all(16),                                child: Image.asset(                                  'assets/fi\_droplet.png',                                  color: kPrimaryColor,                                ),                              ),                              const SizedBox(height: 24),                              Row(                                children: [                                  Expanded(                                    child: Container(                                      decoration: BoxDecoration(                                        border: Border(                                          right: BorderSide(                                            width: 1,                                            color: kGrayColor.withOpacity(0.3),                                          ),                                        ),                                      ),                                      child: Column(                                        crossAxisAlignment:                                            CrossAxisAlignment.center,                                        children: [                                          Text(                                            'Minimum:',                                            style: grayTextStyle.copyWith(                                              fontSize: 12,                                            ),                                          ),                                          Wrap(                                            crossAxisAlignment:                                                WrapCrossAlignment.end,                                            children: [                                              Text(                                                '${state.bloodOxygen.isEmpty ? 0 : listValue.reduce(math.min)}',                                                style: blackTextStyle.copyWith(                                                  fontSize: 20,                                                  fontWeight: bold,                                                ),                                              ),                                              Text(                                                '%',                                                style: grayTextStyle.copyWith(                                                  fontSize: 20,                                                ),                                              ),                                            ],                                          ),                                        ],                                      ),                                    ),                                  ),                                  Expanded(                                    child: Container(                                      decoration: BoxDecoration(                                        border: Border(                                          right: BorderSide(                                            width: 1,                                            color: kGrayColor.withOpacity(0.3),                                          ),                                        ),                                      ),                                      child: Column(                                        crossAxisAlignment:                                            CrossAxisAlignment.center,                                        children: [                                          Text(                                            'Terbaru:',                                            style: grayTextStyle.copyWith(                                              fontSize: 12,                                            ),                                          ),                                          Wrap(                                            crossAxisAlignment:                                                WrapCrossAlignment.end,                                            children: [                                              Text(                                                '${state.bloodOxygen.isEmpty ? 0 : getValueFromHealthValue(state.bloodOxygen.first.value)}',                                                style: blackTextStyle.copyWith(                                                  fontSize: 20,                                                  fontWeight: bold,                                                ),                                              ),                                              Text(                                                '%',                                                style: grayTextStyle.copyWith(                                                  fontSize: 20,                                                ),                                              ),                                            ],                                          ),                                        ],                                      ),                                    ),                                  ),                                  Expanded(                                    child: Column(                                      crossAxisAlignment:                                          CrossAxisAlignment.center,                                      children: [                                        Text(                                          'Maksimum:',                                          style: grayTextStyle.copyWith(                                            fontSize: 12,                                          ),                                        ),                                        Wrap(                                          crossAxisAlignment:                                              WrapCrossAlignment.end,                                          children: [                                            Text(                                              '${state.bloodOxygen.isEmpty ? 0 : listValue.reduce(math.max)}',                                              style: blackTextStyle.copyWith(                                                fontSize: 20,                                                fontWeight: bold,                                              ),                                            ),                                            Text(                                              '%',                                              style: grayTextStyle.copyWith(                                                fontSize: 20,                                              ),                                            ),                                          ],                                        ),                                      ],                                    ),                                  ),                                ],                              ),                            ],                          ),                        ],                      ),                    ),                  ],                ),              );            }            return Container();          },        );      }      Widget more() {        return const MoreButton(          text: 'Tentang Saturasi Oksigen',          route: '/blood-oxygen-information',        );      }      Widget history() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthBloodOxygenSuccess) {              return Container(                margin: EdgeInsets.all(                  defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Row(                      children: [                        Text(                          'Data',                          style: blackTextStyle.copyWith(                            fontSize: 16,                            fontWeight: bold,                          ),                        ),                        const Spacer(),                        GestureDetector(                          onTap: () {                            Navigator.pushNamed(context, '/all-blood-oxygen');                          },                          child: Text(                            'Semua',                            style: grayTextStyle.copyWith(                              fontSize: 14,                              fontWeight: bold,                            ),                          ),                        ),                      ],                    ),                    state.bloodOxygen.isEmpty                        ? Center(                            child: Container(                              margin: const EdgeInsets.only(                                top: 56,                              ),                              padding: const EdgeInsets.symmetric(horizontal: 36),                              child: Column(                                children: [                                  Image.asset(                                    'assets/document.png',                                    height: 100,                                  ),                                  const SizedBox(height: 10),                                  Text(                                    'Data Kosong',                                    style: blackTextStyle.copyWith(                                      fontSize: 20,                                      fontWeight: bold,                                    ),                                  ),                                  const SizedBox(height: 5),                                  Text(                                    'Data saturasi oksigen kamu hari ini masih kosong nih, yuk lakukan pengecekan pada smartwatch kamu.',                                    style: grayTextStyle.copyWith(),                                    textAlign: TextAlign.center,                                  ),                                ],                              ),                            ),                          )                        : Column(                            children: state.bloodOxygen                                .map(                                  (health) => HealthDataItem(                                    health,                                    category: 'blood\_oxygen',                                  ),                                )                                .toList(),                          ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        body: ListView(          children: [            header(),            report(),            more(),            history(),          ],        ),      );    }  } |

1. edit\_date\_of\_birth\_page.dart

|  |
| --- |
| import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/shared/shared\_methods.dart';  import 'package:healio/shared/theme.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../blocs/user/user\_bloc.dart';  class EditDateOfBirthPage extends StatefulWidget {    const EditDateOfBirthPage({required this.date, Key? key}) : super(key: key);    final String date;    @override    State<EditDateOfBirthPage> createState() => \_EditDateOfBirthPageState();  }  late DateTime initialDate;  late String selectedDate;  class \_EditDateOfBirthPageState extends State<EditDateOfBirthPage> {    User? user = FirebaseAuth.instance.currentUser;    bool validate() {      if (selectedDate.isEmpty) {        return false;      }      return true;    }    @override    void initState() {      super.initState();      initialDate = DateTime.parse(widget.date);      selectedDate = widget.date;    }    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(10),                  decoration: BoxDecoration(                    color: kRedColor,                    borderRadius: BorderRadius.circular(                      defaultRadius,                    ),                  ),                  child: Icon(                    Icons.close,                    size: 20,                    color: kWhiteColor,                  ),                ),              ),              const Spacer(),              BlocConsumer<UserBloc, UserState>(                listener: (context, state) {                  if (state is UserSuccess) {                    context.read<AuthBloc>().add(AuthGetCurrentUser(user!.uid));                    Navigator.pushNamed(context, '/main');                  }                },                builder: (context, state) {                  if (state is UserLoading) {                    return Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: SizedBox(                        height: 20,                        width: 20,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return GestureDetector(                    onTap: () {                      if (validate()) {                        context.read<UserBloc>().add(                            UserUpdate('date\_of\_birth', selectedDate, user!.uid));                      } else {                        customShowSnackBar(                            context, 'Field nama tidak boleh kosong');                      }                    },                    child: Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: Icon(                        Icons.check,                        size: 20,                        color: kWhiteColor,                      ),                    ),                  );                },              ),            ],          ),        );      }      Widget body() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Tanggal Lahir',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              GestureDetector(                onTap: () {                  showDatePicker(                    context: context,                    initialDate: initialDate,                    firstDate: DateTime(1950),                    lastDate: DateTime(2050),                  ).then((value) {                    if (value != null) {                      setState(() {                        selectedDate = value.toString();                      });                    }                  });                },                child: Container(                  decoration: BoxDecoration(                    color: kLightGrayColor,                    borderRadius: BorderRadius.circular(                      defaultRadius,                    ),                  ),                  padding: const EdgeInsets.symmetric(                    horizontal: 16,                    vertical: 10,                  ),                  height: 57,                  width: double.infinity,                  child: Row(                    children: [                      Text(                        selectedDate,                        style: grayTextStyle,                      ),                    ],                  ),                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. edit\_gender\_page.dart

|  |
| --- |
| import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/shared/theme.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../blocs/user/user\_bloc.dart';  class EditGenderPage extends StatefulWidget {    const EditGenderPage({Key? key}) : super(key: key);    @override    State<EditGenderPage> createState() => \_EditGenderPageState();  }  class \_EditGenderPageState extends State<EditGenderPage> {    String gender = 'male';    User? user = FirebaseAuth.instance.currentUser;    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(10),                  decoration: BoxDecoration(                    color: kRedColor,                    borderRadius: BorderRadius.circular(                      defaultRadius,                    ),                  ),                  child: Icon(                    Icons.close,                    size: 20,                    color: kWhiteColor,                  ),                ),              ),              const Spacer(),              BlocConsumer<UserBloc, UserState>(                listener: (context, state) {                  if (state is UserSuccess) {                    context.read<AuthBloc>().add(AuthGetCurrentUser(user!.uid));                    Navigator.pushNamed(context, '/main');                  }                },                builder: (context, state) {                  if (state is UserLoading) {                    return Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: SizedBox(                        height: 20,                        width: 20,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return GestureDetector(                    onTap: () {                      context                          .read<UserBloc>()                          .add(UserUpdate('gender', gender, user!.uid));                    },                    child: Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: Icon(                        Icons.check,                        size: 20,                        color: kWhiteColor,                      ),                    ),                  );                },              ),            ],          ),        );      }      Widget body() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Jenis Kelamin',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              Row(                children: [                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'male';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'male' ? kPrimaryColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.male,                              size: 20,                              color: gender == 'male' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Laki-laki',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'male' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                  const SizedBox(width: 16),                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'female';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'female' ? kRedColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.female,                              size: 20,                              color:                                  gender == 'female' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Perempuan',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'female' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                ],              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. edit\_name\_page.dart

|  |
| --- |
| import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/user/user\_bloc.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../shared/shared\_methods.dart';  import '../shared/theme.dart';  class EditNamePage extends StatelessWidget {    const EditNamePage({required this.name, Key? key}) : super(key: key);    final String name;    @override    Widget build(BuildContext context) {      final nameController = TextEditingController(text: name);      User? user = FirebaseAuth.instance.currentUser;      validate() {        if (nameController.text.isEmpty) {          return false;        }        return true;      }      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(10),                  decoration: BoxDecoration(                    color: kRedColor,                    borderRadius: BorderRadius.circular(                      defaultRadius,                    ),                  ),                  child: Icon(                    Icons.close,                    size: 20,                    color: kWhiteColor,                  ),                ),              ),              const Spacer(),              BlocConsumer<UserBloc, UserState>(                listener: (context, state) {                  if (state is UserSuccess) {                    context.read<AuthBloc>().add(AuthGetCurrentUser(user!.uid));                    Navigator.pushNamed(context, '/main');                  }                },                builder: (context, state) {                  if (state is UserLoading) {                    return Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: SizedBox(                        height: 20,                        width: 20,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return GestureDetector(                    onTap: () {                      if (validate()) {                        context.read<UserBloc>().add(                            UserUpdate('name', nameController.text, user!.uid));                      } else {                        customShowSnackBar(                            context, 'Field nama tidak boleh kosong');                      }                    },                    child: Container(                      padding: const EdgeInsets.all(10),                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: Icon(                        Icons.check,                        size: 20,                        color: kWhiteColor,                      ),                    ),                  );                },              ),            ],          ),        );      }      Widget body() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Nama',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              Container(                decoration: BoxDecoration(                  color: kLightGrayColor,                  borderRadius: BorderRadius.circular(                    defaultRadius,                  ),                ),                padding: const EdgeInsets.symmetric(                  horizontal: 16,                  vertical: 10,                ),                height: 57,                child: Center(                  child: TextFormField(                    controller: nameController,                    style: blackTextStyle,                    decoration: InputDecoration.collapsed(                      hintText: 'Nama',                      hintStyle: grayTextStyle,                    ),                  ),                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. health\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/health\_category\_tile.dart';  class HealthPage extends StatelessWidget {    const HealthPage({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(defaultMargin),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Kesehatan',                style: blackTextStyle.copyWith(                  fontSize: 26,                  fontWeight: bold,                ),              ),              const SizedBox(height: 5),              Text(                'Meliputi denyut jantung dan saturasi oksigen',                style: grayTextStyle,              ),            ],          ),        );      }      Widget body() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Kategori',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              const HealthCategoryTile(                healthCategory: HealthCategory.heartrate,              ),              const SizedBox(height: 10),              const HealthCategoryTile(                healthCategory: HealthCategory.bloodoxygen,              ),              const SizedBox(height: 24),              Text(                'Lainnya',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              GestureDetector(                onTap: () {                  Navigator.pushNamed(context, '/heart-disease-check');                },                child: Container(                  decoration: BoxDecoration(                    color: kLightGrayColor,                    borderRadius: BorderRadius.circular(12),                  ),                  margin: const EdgeInsets.only(top: 10),                  padding: const EdgeInsets.all(16),                  height: 57,                  child: Row(                    children: [                      Text(                        'Prediksi Penyakit Jantung',                        style: blackTextStyle,                      ),                    ],                  ),                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. heart\_disease\_check\_page.dart

|  |
| --- |
| import 'dart:developer';  import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/models/heart\_disease\_form\_model.dart';  import 'package:healio/shared/shared\_methods.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import 'package:healio/widget/custom\_button.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../blocs/health/health\_bloc.dart';  import '../blocs/heart\_disease/heart\_disease\_bloc.dart';  import '../widget/custom\_text\_form\_field.dart';  import 'heart\_disease\_result\_page.dart';  class HeartDiseaseCheckPage extends StatefulWidget {    const HeartDiseaseCheckPage({Key? key}) : super(key: key);    @override    State<HeartDiseaseCheckPage> createState() => \_HeartDiseaseCheckPageState();  }  final ageController = TextEditingController(text: '');  final heartRateController = TextEditingController(text: '');  final cigarettesController = TextEditingController(text: '');  final glucoseController = TextEditingController(text: '');  final diabetesController = TextEditingController(text: '');  final cholesterolController = TextEditingController(text: '');  final systolicController = TextEditingController(text: '');  final diastolicController = TextEditingController(text: '');  String gender = 'male';  class \_HeartDiseaseCheckPageState extends State<HeartDiseaseCheckPage> {    @override    void initState() {      super.initState();      fetchData();      final healthState = context.read<HealthBloc>().state;      if (healthState is HealthHeartRateSuccess) {        heartRateController.text =            getValueFromHealthValue(healthState.heartRate.first.value).toString();      }      final authState = context.read<AuthBloc>().state;      if (authState is AuthSuccess) {        ageController.text =            calculateAge(DateTime.parse(authState.user.dateOfBirth)).toString();      }    }    fetchData() {      context.read<HealthBloc>().add(HealthGetHeartRate());    }    @override    Widget build(BuildContext context) {      bool validate() {        if (ageController.text.isEmpty ||            heartRateController.text.isEmpty ||            cigarettesController.text.isEmpty ||            glucoseController.text.isEmpty ||            diabetesController.text.isEmpty ||            cholesterolController.text.isEmpty ||            systolicController.text.isEmpty ||            diastolicController.text.isEmpty) {          return false;        }        return true;      }      Widget appBar() {        return const CustomAppBar(title: 'Prediksi Penyakit Jantung');      }      Widget body() {        return Container(          margin: EdgeInsets.only(            left: defaultMargin,            right: defaultMargin,            bottom: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              CustomTextFormField(                label: 'Umur',                controller: ageController,              ),              const SizedBox(height: 16),              Text(                'Jenis Kelamin',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              Row(                children: [                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'male';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'male' ? kPrimaryColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.male,                              size: 20,                              color: gender == 'male' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Laki-laki',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'male' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                  const SizedBox(width: 16),                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'female';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'female' ? kRedColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.female,                              size: 20,                              color:                                  gender == 'female' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Perempuan',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'female' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                ],              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Konsumsi Rokok per Hari',                controller: cigarettesController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Denyut Jantung',                controller: heartRateController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Gula Darah',                controller: glucoseController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Diabetes',                controller: diabetesController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Kolesterol',                controller: cholesterolController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Sistolik',                controller: systolicController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Diastolik',                controller: diastolicController,              ),              const SizedBox(height: 16),              BlocConsumer<HeartDiseaseBloc, HeartDiseaseState>(                listener: (context, state) {                  if (state is HeartDiseaseSuccess) {                    Navigator.push(                      context,                      MaterialPageRoute(                        builder: (context) => HeartDiseaseResultPage(                          data: HeartDiseaseFormModel(                            age: int.parse(ageController.text),                            cigs: int.parse(cigarettesController.text),                            hRate: int.parse(heartRateController.text),                            gluc: int.parse(glucoseController.text),                            dia: int.parse(diabetesController.text),                            chol: int.parse(cholesterolController.text),                            sBP: int.parse(systolicController.text),                            dBP: int.parse(diastolicController.text),                            sex: gender == 'male' ? 1 : 0,                          ),                        ),                      ),                    );                  } else if (state is HeartDiseaseFailed) {                    customShowSnackBar(context, state.e);                  }                },                builder: (context, state) {                  if (state is HeartDiseaseLoading) {                    return CustomButton(                      onPressed: () {},                      child: SizedBox(                        height: 21,                        width: 21,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return CustomButton(                    onPressed: () {                      if (validate()) {                        context.read<HeartDiseaseBloc>().add(                              HeartDiseasePrediction(                                HeartDiseaseFormModel(                                  age: int.parse(ageController.text),                                  cigs: int.parse(cigarettesController.text),                                  hRate: int.parse(heartRateController.text),                                  gluc: int.parse(glucoseController.text),                                  dia: int.parse(diabetesController.text),                                  chol: int.parse(cholesterolController.text),                                  sBP: int.parse(systolicController.text),                                  dBP: int.parse(diastolicController.text),                                  sex: gender == 'male' ? 1 : 0,                                ),                              ),                            );                      } else {                        customShowSnackBar(context, 'Semua field harus diisi.');                      }                    },                    child: Text(                      'Cek',                      style: whiteTextStyle.copyWith(                        fontWeight: bold,                      ),                    ),                  );                },              ),            ],          ),        );      }      return Scaffold(        backgroundColor: kWhiteColor,        body: ListView(          children: [            appBar(),            body(),          ],        ),      );    }  } |

1. heart\_disease\_result\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/models/heart\_disease\_form\_model.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import '../blocs/heart\_disease/heart\_disease\_bloc.dart';  class HeartDiseaseResultPage extends StatelessWidget {    const HeartDiseaseResultPage({required this.data, Key? key})        : super(key: key);    final HeartDiseaseFormModel data;    @override    Widget build(BuildContext context) {      Widget appBar() {        return const CustomAppBar(title: 'Hasil Prediksi');      }      Widget body() {        return BlocBuilder<HeartDiseaseBloc, HeartDiseaseState>(          builder: (context, state) {            if (state is HeartDiseaseSuccess) {              return Container(                margin: const EdgeInsets.only(                  left: 24,                  right: 24,                  bottom: 24,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    SizedBox(                      width: double.infinity,                      child: Column(                        children: [                          Image.asset(                            'assets/health-checkup-4034059-3337536.png',                            height: 100,                          ),                          const SizedBox(height: 24),                          Text(                            'Probabilitas',                            style: blackTextStyle.copyWith(                              fontSize: 16,                              fontWeight: bold,                            ),                          ),                          const SizedBox(height: 10),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Tidak memiliki: ',                                style: blackTextStyle,                              ),                              Text(                                state.data['probability'][0][0]                                    .toStringAsFixed(3),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Memiliki: ',                                style: blackTextStyle,                              ),                              Text(                                state.data['probability'][0][1]                                    .toStringAsFixed(3),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                        ],                      ),                    ),                    Container(                      margin: EdgeInsets.only(top: defaultMargin),                      child: Column(                        crossAxisAlignment: CrossAxisAlignment.start,                        children: [                          Text(                            'Data Masukan',                            style: blackTextStyle.copyWith(                              fontSize: 16,                              fontWeight: bold,                            ),                          ),                          const SizedBox(height: 10),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Umur: ',                                style: blackTextStyle,                              ),                              Text(                                data.age.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Konsumsi Rokor per Hari: ',                                style: blackTextStyle,                              ),                              Text(                                data.cigs.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Jenis Kelamin: ',                                style: blackTextStyle,                              ),                              Text(                                data.sex.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Denyut Jantung: ',                                style: blackTextStyle,                              ),                              Text(                                data.hRate.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Gula Darah: ',                                style: blackTextStyle,                              ),                              Text(                                data.gluc.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Diabetes: ',                                style: blackTextStyle,                              ),                              Text(                                data.dia.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Kolesterol: ',                                style: blackTextStyle,                              ),                              Text(                                data.chol.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Sistolik: ',                                style: blackTextStyle,                              ),                              Text(                                data.sBP.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                          Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Text(                                'Diastolik: ',                                style: blackTextStyle,                              ),                              Text(                                data.dBP.toString(),                                style: blackTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                        ],                      ),                    ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        backgroundColor: kWhiteColor,        body: ListView(          children: [            appBar(),            body(),          ],        ),      );    }  } |

1. heart\_rate\_high\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class HeartRateHighPage extends StatelessWidget {    const HeartRateHighPage({required this.heartRate, Key? key})        : super(key: key);    final int heartRate;    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(defaultMargin),          width: double.infinity,          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(7),                  decoration: BoxDecoration(                    color: const Color(0xffefefef),                    borderRadius: BorderRadius.circular(12),                  ),                  child: const Icon(                    Icons.chevron\_left,                    size: 21,                  ),                ),              ),            ],          ),        );      }      Widget content() {        return Container(          margin: EdgeInsets.only(            left: defaultMargin,            right: defaultMargin,            bottom: defaultMargin,          ),          child: Column(            children: [              Text(                'Denyut Jantung Tinggi',                style: blackTextStyle.copyWith(                  fontSize: 20,                  fontWeight: semiBold,                ),              ),              const SizedBox(height: 10),              Image.asset(                'assets/health-checkup-4034059-3337536.png',                height: 100,              ),              const SizedBox(height: 10),              Text(                '$heartRate bpm',                style: blackTextStyle.copyWith(                  fontSize: 20,                  fontWeight: semiBold,                ),              ),              SizedBox(height: defaultMargin),              Column(                crossAxisAlignment: CrossAxisAlignment.start,                children: [                  Text(                    'Denyut jantung kamu termasuk ke dalam kategori tinggi, atau biasa disebut dengan istilah takikardia. Takikardia terjadi jika denyut jantung melebihi 100 kali per menit atau bpm. Kondisi ini terjadi karena adanya gangguan listrik di jantung yangn berperan dalam mengontrol irama denyut jantung.\nTakikardia bisa muncul tanpa menimbulkan komplikasi. Namun, jika tidak ditangani, denyut jantung yang tinggi bisa menyebabkan komplikasi serius, seperti stroke, gagal jantung, hentin jantung, dan bahkan kematian.\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Text(                    'Penanganan Denyut Jantung Tinggi',                    style: blackTextStyle.copyWith(                      fontSize: 20,                      fontWeight: semiBold,                    ),                  ),                  const SizedBox(height: 10),                  Text(                    'Denyut jantung tinggi yang terjadi bukan karena penyakit, umumnya tidak membutuhkan pengobatan karena dapat membaik dengan sendirinya. Jika disebabkan oleh kondisi medis tertentu, penanganan denyut jantung tinggi akan disesuaikan dengan faktor penyebabnya.\nPenanganan yang dilakukan bertujuan untuk memperlambat denyut jantung tinggi hingga kembali de dalam batas normal, mencegah denyut jantung tinggi kembali, menekan risiko komplikasi, dan mengobati penyakit mendasar yang dapat menyebabkan takikardia.\n\nPenanganan untuk takikardia meliputi:\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Text(                    'Tindakan',                    style: blackTextStyle.copyWith(                      fontSize: 20,                      fontWeight: semiBold,                    ),                  ),                  const SizedBox(height: 10),                  Text(                    'Untuk meredakan denyut jantung tinggi, kamu bisa melakukan tindakan Manuver Vagal, yaitu melakukan gerakan seperti batuk, mengejan sebagaimana tengah buang air besar, dan menaruh es pada wajah untuk mempengaruhi saraf vagus yang mengatur denyut jantung.\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Text(                    'Obat-obatan',                    style: blackTextStyle.copyWith(                      fontSize: 20,                      fontWeight: semiBold,                    ),                  ),                  const SizedBox(height: 10),                  Text(                    'Untuk menormalkan denyut jantung tinggi atau cepat, dokter kemungkinan akan memberikan obat-obatan lain, seperti:\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Obat aniaritmia, untuk mengembalikan denyut jantung normal. Obat ini diberikan jika Menuver Vagal tidak berhasil meredakan denyut jantung tinggi.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Penghambat saluran kalsium dan penghambat beta, untuk mencegah denyut jantung tinggi teradi lagi.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Obat pengencer darah, untuk membantu menurunkan risiko penggumpalan darah.\n',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Text(                    'Menerapkan Gaya Hidup Sehat',                    style: blackTextStyle.copyWith(                      fontSize: 20,                      fontWeight: semiBold,                    ),                  ),                  const SizedBox(height: 10),                  Text(                    'Menerapkan gaya hidup sehat, seperti menjaga berat badan ideal, tetap aktif bergerak, mengurangi stress, dan menghindari rokok, merupakan beberapa cara agar jantung tetap sehat dan terhindar dari denyut jantung tinggi.\n\nJika denyut jantung tinggi terjadi secara berkelanjutan dan tidak diketahui dengan pasti apa penyebabnya, konsultasikan dengan dokter untuk mendapatkan pemeriksaan lebih lanjut.',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                ],              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            content(),          ],        ),      );    }  } |

1. heart\_rate\_information\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import '../shared/theme.dart';  class HeartRateInformationPage extends StatelessWidget {    const HeartRateInformationPage({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      Widget header() {        return const CustomAppBar(          title: 'Tentang Denyut Jantung',        );      }      Widget columnItem(String value, bool isTitle) {        return Column(          children: [            Container(              padding: const EdgeInsets.all(10),              child: Text(                value,                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: isTitle ? bold : reguler,                ),              ),            ),          ],        );      }      Widget content() {        return Container(          margin: EdgeInsets.only(            right: defaultMargin,            left: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              ClipRRect(                borderRadius: BorderRadius.circular(12),                child: Image.asset(                  'assets/hr\_thumbnail.jpg',                ),              ),              SizedBox(                height: defaultMargin,              ),              Text(                'Detak Jantung',                style: blackTextStyle.copyWith(                  fontSize: 18,                  fontWeight: bold,                ),              ),              const SizedBox(                height: 10,              ),              Text(                'Jantung merupakan organ vital pada tubuh manusia. Fungsi jantung adalah memompa darah ke seluruh tubuh, sehingga berbagai organ dan sistem tubuh Anda bisa bekerja sebagaimana mestinya.\n\nSelain tekanan darah, salah satu indikator penting dari kesehatan jantung adalah detak jantung. Detak jantung adalah berapa kali jantung Anda berdenyut dalam satu menit. Adapun detak jantung seseorang dipengaruhi oleh berbagai faktor, seperti usia, ukuran tubuh, kondisi jantung, cuaca atau temperatur udara, aktivitas fisik, emosi, dan obat-obatan tertentu.',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),              SizedBox(                height: defaultMargin,              ),              Text(                'Tabel Detak Jantung Berdasarkan Usia',                style: blackTextStyle.copyWith(                  fontSize: 18,                  fontWeight: bold,                ),              ),              const SizedBox(                height: 10,              ),              Center(                child: Table(                  border: TableBorder.all(                    color: Colors.black,                    style: BorderStyle.solid,                    width: 1,                  ),                  children: [                    TableRow(                      children: [                        columnItem('Usia', true),                        columnItem('Normal (bpm)', true),                      ],                    ),                    TableRow(                      children: [                        columnItem('Bayi', false),                        columnItem('70 - 130', false),                      ],                    ),                    TableRow(                      children: [                        columnItem('Anak - anak', false),                        columnItem('80 - 110', false),                      ],                    ),                    TableRow(                      children: [                        columnItem('Dewasa', false),                        columnItem('60 - 100', false),                      ],                    ),                  ],                ),              ),              SizedBox(                height: defaultMargin,              ),              const Divider(),            ],          ),        );      }      Widget disclaimer() {        return Container(          margin: EdgeInsets.all(defaultMargin),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Row(                crossAxisAlignment: CrossAxisAlignment.center,                children: [                  Icon(                    Icons.warning,                    size: 16,                    color: kRedColor,                  ),                  const SizedBox(                    width: 5,                  ),                  Text(                    'Disclaimer',                    style: blackTextStyle.copyWith(                      fontSize: 18,                      fontWeight: bold,                    ),                  ),                ],              ),              const SizedBox(                height: 10,              ),              Text(                'Harap diingat bahwa pemantauan detak jantung hanya untuk kebugaran dan kesehatan, bukan untuk diagnosis atau perawatan kondisi medis apapun. Jika Anda memiliki kekhawatiran tentang jantung Anda, pastikan untuk berkonsultasi dengan profesional medis.',                style: blackTextStyle.copyWith(                  fontSize: 16,                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            content(),            disclaimer(),          ],        ),      );    }  } |

1. heart\_rate\_low\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:health/health.dart';  import '../shared/theme.dart';  class HeartRateLowPage extends StatelessWidget {    const HeartRateLowPage({required this.heartRate, Key? key}) : super(key: key);    final int heartRate;    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(defaultMargin),          width: double.infinity,          child: Row(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  padding: const EdgeInsets.all(7),                  decoration: BoxDecoration(                    color: const Color(0xffefefef),                    borderRadius: BorderRadius.circular(12),                  ),                  child: const Icon(                    Icons.chevron\_left,                    size: 21,                  ),                ),              ),            ],          ),        );      }      Widget content() {        return Container(          margin: EdgeInsets.only(            left: defaultMargin,            right: defaultMargin,            bottom: defaultMargin,          ),          child: Column(            children: [              Text(                'Denyut Jantung Rendah',                style: blackTextStyle.copyWith(                  fontSize: 20,                  fontWeight: semiBold,                ),              ),              const SizedBox(height: 10),              Image.asset(                'assets/health-checkup-4034059-3337536.png',                height: 100,              ),              const SizedBox(height: 10),              Text(                '$heartRate bpm',                style: blackTextStyle.copyWith(                  fontSize: 20,                  fontWeight: semiBold,                ),              ),              SizedBox(height: defaultMargin),              Column(                crossAxisAlignment: CrossAxisAlignment.start,                children: [                  Text(                    'Denyut jantung kamu termasuk ke dalam kategori rendah, atau biasa disebut dengan istilah bradikardia. Bradikardia adalah kondisi ketika jantung berdetak lebih lambat dari biasanya.\n\nMelambatnya denyut jantung umumnya merupakan hal yang normal. Kondisi tersebut dapat terjadi pada orang yang sedang tidur, remaja, atau atlet. Namun, jika disertai dengan gejala pusing atau sesak nafas, denyut jantung yang melambat bisa menjadi tand adanya gangguan pada aktivitas listrik jantung.\n\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Text(                    'Pencegahan Bradikardia',                    style: blackTextStyle.copyWith(                      fontSize: 20,                      fontWeight: semiBold,                    ),                  ),                  const SizedBox(height: 10),                  Text(                    'Bradikardia dapat dicegah dengan menghindari faktor-faktor yang dapat meningkatkan risiko terjadinya kondisi ini. Caranya adalah dengan mengubah gaya hidup agar lebih sehat, yaitu dengan melakukan langkah sederhana berikut ini:\n',                    style: blackTextStyle.copyWith(                      fontSize: 16,                    ),                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Menghindari kebiasaan merokok.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Menghindari penggunaan NAPZA.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Membatasi konsumsi alkohol.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Menghindari stres.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Menjaga berat badan ideal.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Berolahraga secara rutin.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      Text(                        '     •  ',                        style: blackTextStyle.copyWith(                          fontSize: 16,                        ),                      ),                      Expanded(                        child: Text(                          'Mengonsumsi makanan bergizi seimbang dan rendah garam.',                          style: blackTextStyle.copyWith(                            fontSize: 16,                          ),                        ),                      ),                    ],                  ),                ],              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            content(),          ],        ),      );    }  } |

1. heart\_rate\_page.dart

|  |
| --- |
| import 'dart:developer';  import 'dart:math' as math;  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/models/health\_model.dart';  import 'package:healio/services/health\_service.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_app\_bar.dart';  import 'package:healio/widget/health\_data\_item.dart';  import 'package:healio/widget/more\_button.dart';  import 'package:health/health.dart';  import 'package:intl/intl.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../blocs/health/health\_bloc.dart';  import '../services/push\_notification\_service.dart';  import '../shared/shared\_methods.dart';  class HeartRatePage extends StatefulWidget {    const HeartRatePage({Key? key}) : super(key: key);    @override    State<HeartRatePage> createState() => \_HeartRatePageState();  }  late HealthModel hrFromFirebase;  List<HealthDataPoint> dataHeartRate = [];  int age = 0;  String category = '';  class \_HeartRatePageState extends State<HeartRatePage> {    @override    void initState() {      super.initState();      getDataFromFirebase();      fetchData();      getAge();    }    void fetchData() {      context.read<HealthBloc>().add(HealthGetHeartRate());      final healthState = context.read<HealthBloc>().state;      if (healthState is HealthHeartRateSuccess) {        dataHeartRate = healthState.heartRate;      }    }    void getDataFromFirebase() async {      hrFromFirebase = await HealthService().getHeartRateFromFirebase();    }    void getAge() {      final authState = context.read<AuthBloc>().state;      if (authState is AuthSuccess) {        age = calculateAge(DateTime.parse(authState.user.dateOfBirth));      }    }    void updateHeartRate() async {      if (hrFromFirebase.value ==              getValueFromHealthValue(dataHeartRate.first.value) &&          hrFromFirebase.dateFrom == dataHeartRate.first.dateFrom.toString()) {        log('Sama');        log(age.toString());      } else {        log('beda');        await HealthService().updateHeartRateFromFirebase(            getValueFromHealthValue(dataHeartRate.first.value),            dataHeartRate.first.dateFrom.toString());        category = setCategory(getValueFromHealthValue(dataHeartRate.first.value),            'heart\_rate', age);        log(category);        if (category == 'rendah') {          customShowSnackBar(context, 'Denyut jantung kamu rendah');          await PushNotificationService().pushNotification(            'Denyut Jantung ${getValueFromHealthValue(dataHeartRate.first.value)}',            'Denyut jantung kamu dibawah nilai normal nih :(.',          );        } else if (category == 'tinggi') {          customShowSnackBar(context, 'Denyut jantung kamu tinggi');          await PushNotificationService().pushNotification(            'Denyut Jantung ${getValueFromHealthValue(dataHeartRate.first.value)}',            'Denyut jantung kamu tinggi nih :(.',          );        }      }    }    syncData() async {      fetchData();      getDataFromFirebase();      updateHeartRate();    }    @override    Widget build(BuildContext context) {      String formattedDate = DateFormat("EEEE, d MMMM").format(DateTime.now());      Widget header() {        return CustomAppBar(          title: 'Denyut Jantung',          action: GestureDetector(            onTap: () {              Navigator.pop(context);            },            child: GestureDetector(              onTap: () {                syncData();              },              child: Container(                padding: const EdgeInsets.all(10),                decoration: BoxDecoration(                  color: kLightGrayColor,                  borderRadius: BorderRadius.circular(                    defaultRadius,                  ),                ),                child: Image.asset(                  'assets/fi\_refresh-cw.png',                  height: 20,                ),              ),            ),          ),        );      }      Widget report() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthHeartRateSuccess) {              List<int> listValue = state.heartRate                  .map((e) => getValueFromHealthValue(e.value))                  .toList();              return Container(                margin: EdgeInsets.symmetric(                  horizontal: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Ringkasan',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    const SizedBox(height: 10),                    Container(                      padding: const EdgeInsets.all(16),                      decoration: BoxDecoration(                        color: kLightGrayColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      child: Column(                        crossAxisAlignment: CrossAxisAlignment.start,                        children: [                          Text(                            formattedDate,                            style: grayTextStyle,                          ),                          const SizedBox(height: 24),                          Column(                            children: [                              Container(                                height: 56,                                width: 56,                                decoration: BoxDecoration(                                  color: kWhiteColor,                                  shape: BoxShape.circle,                                ),                                padding: const EdgeInsets.all(16),                                child: Image.asset(                                  'assets/fi\_heart.png',                                  color: kRedColor,                                ),                              ),                              const SizedBox(height: 24),                              Row(                                children: [                                  Expanded(                                    child: Container(                                      decoration: BoxDecoration(                                        border: Border(                                          right: BorderSide(                                            width: 1,                                            color: kGrayColor.withOpacity(0.3),                                          ),                                        ),                                      ),                                      child: Column(                                        crossAxisAlignment:                                            CrossAxisAlignment.center,                                        children: [                                          Text(                                            'Minimum:',                                            style: grayTextStyle.copyWith(                                              fontSize: 12,                                            ),                                          ),                                          Wrap(                                            crossAxisAlignment:                                                WrapCrossAlignment.end,                                            children: [                                              Text(                                                '${state.heartRate.isEmpty ? 0 : listValue.reduce(math.min)}',                                                // '',                                                style: blackTextStyle.copyWith(                                                  fontSize: 20,                                                  fontWeight: bold,                                                ),                                              ),                                              Text(                                                'BPM',                                                style: grayTextStyle.copyWith(),                                              ),                                            ],                                          ),                                        ],                                      ),                                    ),                                  ),                                  Expanded(                                    child: Container(                                      decoration: BoxDecoration(                                        border: Border(                                          right: BorderSide(                                            width: 1,                                            color: kGrayColor.withOpacity(0.3),                                          ),                                        ),                                      ),                                      child: Column(                                        crossAxisAlignment:                                            CrossAxisAlignment.center,                                        children: [                                          Text(                                            'Terbaru:',                                            style: grayTextStyle.copyWith(                                              fontSize: 12,                                            ),                                          ),                                          Wrap(                                            crossAxisAlignment:                                                WrapCrossAlignment.end,                                            children: [                                              Text(                                                '${state.heartRate.isEmpty ? 0 : getValueFromHealthValue(state.heartRate.first.value)}',                                                // '',                                                style: blackTextStyle.copyWith(                                                  fontSize: 20,                                                  fontWeight: bold,                                                ),                                              ),                                              Text(                                                'BPM',                                                style: grayTextStyle.copyWith(),                                              ),                                            ],                                          ),                                        ],                                      ),                                    ),                                  ),                                  Expanded(                                    child: Column(                                      crossAxisAlignment:                                          CrossAxisAlignment.center,                                      children: [                                        Text(                                          'Maksimum:',                                          style: grayTextStyle.copyWith(                                            fontSize: 12,                                          ),                                        ),                                        Wrap(                                          crossAxisAlignment:                                              WrapCrossAlignment.end,                                          children: [                                            Text(                                              '${state.heartRate.isEmpty ? 0 : listValue.reduce(math.max)}',                                              // '',                                              style: blackTextStyle.copyWith(                                                fontSize: 20,                                                fontWeight: bold,                                              ),                                            ),                                            Text(                                              'BPM',                                              style: grayTextStyle.copyWith(),                                            ),                                          ],                                        ),                                      ],                                    ),                                  ),                                ],                              ),                            ],                          ),                        ],                      ),                    ),                  ],                ),              );            }            return Container();          },        );      }      Widget more() {        return const MoreButton(          text: 'Tentang Denyut Jantung',          route: '/heart-rate-information',        );      }      Widget history() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthHeartRateSuccess) {              return Container(                margin: EdgeInsets.all(                  defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Row(                      children: [                        Text(                          'Data',                          style: blackTextStyle.copyWith(                            fontSize: 16,                            fontWeight: bold,                          ),                        ),                        const Spacer(),                        GestureDetector(                          onTap: () {                            Navigator.pushNamed(context, '/all-heart-rate');                          },                          child: Text(                            'Semua',                            style: grayTextStyle.copyWith(                              fontSize: 14,                              fontWeight: bold,                            ),                          ),                        ),                      ],                    ),                    state.heartRate.isEmpty                        ? Center(                            child: Container(                              margin: const EdgeInsets.only(                                top: 56,                              ),                              padding: const EdgeInsets.symmetric(horizontal: 36),                              child: Column(                                children: [                                  Image.asset(                                    'assets/document.png',                                    height: 100,                                  ),                                  const SizedBox(height: 10),                                  Text(                                    'Data Kosong',                                    style: blackTextStyle.copyWith(                                      fontSize: 20,                                      fontWeight: bold,                                    ),                                  ),                                  const SizedBox(height: 5),                                  Text(                                    'Data denyut jantung kamu hari ini masih kosong nih, yuk lakukan pengecekan pada smartwatch kamu.',                                    style: grayTextStyle.copyWith(),                                    textAlign: TextAlign.center,                                  ),                                ],                              ),                            ),                          )                        : Column(                            children: state.heartRate                                .map((health) => HealthDataItem(health))                                .toList(),                          ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        body: ListView(          children: [            header(),            report(),            more(),            history(),          ],        ),      );    }  } |

1. home\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/models/article\_model.dart';  import 'package:healio/pages/blood\_oxygen\_low\_handling\_page.dart';  import 'package:healio/pages/heart\_rate\_high\_page.dart';  import 'package:healio/pages/heart\_rate\_low\_page.dart';  import 'package:healio/shared/shared\_methods.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/article\_card.dart';  import '../blocs/article/article\_bloc.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../blocs/health/health\_bloc.dart';  class HomePage extends StatefulWidget {    const HomePage({Key? key}) : super(key: key);    @override    State<HomePage> createState() => \_HomePageState();  }  int age = 0;  class \_HomePageState extends State<HomePage> {    @override    void initState() {      super.initState();      context.read<HealthBloc>().add(HealthGetLandingPage());      getAge();    }    void getAge() {      final authState = context.read<AuthBloc>().state;      if (authState is AuthSuccess) {        age = calculateAge(DateTime.parse(authState.user.dateOfBirth));      }    }    @override    Widget build(BuildContext context) {      Widget header() {        return BlocBuilder<AuthBloc, AuthState>(          builder: (context, state) {            if (state is AuthSuccess) {              age = calculateAge(DateTime.parse(state.user.dateOfBirth));              return Container(                margin: EdgeInsets.all(                  defaultMargin,                ),                child: Row(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Column(                      crossAxisAlignment: CrossAxisAlignment.start,                      children: [                        Text(                          'Halo,\n${state.user.name}',                          style: blackTextStyle.copyWith(                            fontSize: 26,                            fontWeight: bold,                          ),                        ),                        const SizedBox(height: 10),                        Text(                          'Bagaimana kesehatanmu hari ini?',                          style: grayTextStyle.copyWith(),                        ),                      ],                    ),                    const Spacer(),                    Container(                      height: 64,                      width: 64,                      decoration: const BoxDecoration(                        image: DecorationImage(                          image: AssetImage(                            'assets/avatar.png',                          ),                        ),                        shape: BoxShape.circle,                      ),                    ),                  ],                ),              );            }            return Container();          },        );      }      Widget today() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthLandingPageSuccess) {              return Container(                margin: EdgeInsets.symmetric(                  horizontal: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Hari Ini',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    const SizedBox(height: 10),                    Row(                      crossAxisAlignment: CrossAxisAlignment.start,                      children: [                        Expanded(                          child: Container(                            padding: const EdgeInsets.all(16),                            decoration: BoxDecoration(                              color: kLightGrayColor,                              borderRadius: BorderRadius.circular(                                12,                              ),                            ),                            child: Column(                              crossAxisAlignment: CrossAxisAlignment.start,                              children: [                                Container(                                  height: 42,                                  width: 42,                                  decoration: BoxDecoration(                                    color: kWhiteColor,                                    shape: BoxShape.circle,                                  ),                                  padding: const EdgeInsets.all(12),                                  margin: const EdgeInsets.only(                                    bottom: 10,                                  ),                                  child: Image.asset(                                    'assets/fi\_heart.png',                                    color: kRedColor,                                  ),                                ),                                Text(                                  'Denyut Jantung',                                  style: grayTextStyle.copyWith(                                    fontSize: 14,                                  ),                                ),                                const SizedBox(height: 5),                                Wrap(                                  crossAxisAlignment: WrapCrossAlignment.end,                                  children: [                                    Text(                                      // '$newestHeartRate ',                                      '${state.heartRate.isEmpty ? 0 : getValueFromHealthValue(state.heartRate.first.value)}',                                      style: blackTextStyle.copyWith(                                        fontSize: 20,                                        fontWeight: bold,                                      ),                                    ),                                    Text(                                      'BPM',                                      style: grayTextStyle.copyWith(                                          fontWeight: bold),                                    ),                                  ],                                ),                              ],                            ),                          ),                        ),                        const SizedBox(width: 16),                        Expanded(                          child: Container(                            padding: const EdgeInsets.all(16),                            decoration: BoxDecoration(                              color: kLightGrayColor,                              borderRadius: BorderRadius.circular(                                defaultRadius,                              ),                            ),                            child: Column(                              crossAxisAlignment: CrossAxisAlignment.start,                              children: [                                Container(                                  height: 42,                                  width: 42,                                  decoration: BoxDecoration(                                    color: kWhiteColor,                                    shape: BoxShape.circle,                                  ),                                  padding: const EdgeInsets.all(12),                                  margin: const EdgeInsets.only(                                    bottom: 10,                                  ),                                  child: Image.asset(                                    'assets/fi\_droplet.png',                                    color: kPrimaryColor,                                  ),                                ),                                Text(                                  'Saturasi Oksigen',                                  style: grayTextStyle.copyWith(                                    fontSize: 14,                                  ),                                ),                                const SizedBox(height: 5),                                Wrap(                                  crossAxisAlignment: WrapCrossAlignment.center,                                  children: [                                    Text(                                      '${state.bloodOxygen.isEmpty ? 0 : getValueFromHealthValue(state.bloodOxygen.first.value)}',                                      style: blackTextStyle.copyWith(                                        fontSize: 20,                                        fontWeight: bold,                                      ),                                    ),                                    Text(                                      '%',                                      style: grayTextStyle.copyWith(                                        fontSize: 20,                                      ),                                    ),                                  ],                                ),                              ],                            ),                          ),                        ),                      ],                    ),                  ],                ),              );            }            return Container();          },        );      }      Widget announce() {        return BlocBuilder<HealthBloc, HealthState>(          builder: (context, state) {            if (state is HealthLandingPageSuccess) {              String hrCategory = state.heartRate.isNotEmpty                  ? setCategory(                      getValueFromHealthValue(state.heartRate.first.value),                      'heart\_rate',                      age,                    )                  : 'empty';              String boCategory = state.bloodOxygen.isNotEmpty                  ? setCategory(                      getValueFromHealthValue(state.bloodOxygen.first.value),                      'blood\_oxygen',                      age,                    )                  : 'empty';              int hrValue = getValueFromHealthValue(state.heartRate.first.value);              int boValue =                  getValueFromHealthValue(state.bloodOxygen.first.value);              return Container(                margin: EdgeInsets.all(defaultMargin),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Text(                      'Notifikasi',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    const SizedBox(height: 10),                    Container(                      width: double.infinity,                      decoration: BoxDecoration(                        color: kRedColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      padding: const EdgeInsets.all(16),                      child: Row(                        children: [                          Image.asset(                            'assets/heart-rate-monitor.png',                            height: 75,                          ),                          const SizedBox(width: 10),                          Expanded(                            child: Column(                              crossAxisAlignment: CrossAxisAlignment.start,                              children: [                                Text(                                  hrCategory == 'tinggi'                                      ? 'Denyut Jantung Tinggi'                                      : hrCategory == 'rendah'                                          ? 'Denyut Jantung Rendah'                                          : hrCategory == 'normal'                                              ? 'Denyut Jantung Normal'                                              : 'Data Denyut Jantung Kosong',                                  style: whiteTextStyle.copyWith(                                    fontSize: 16,                                    fontWeight: bold,                                  ),                                ),                                const SizedBox(height: 10),                                Text(                                  hrCategory == 'tinggi'                                      ? 'Wah denyut jantung kamu tinggi 😟 Yuk baca tips berikut sebagai penanganan pertama pada kondisi kamu.'                                      : hrCategory == 'rendah'                                          ? 'Wah denyut jantung kamu rendah 😟 Yuk baca tips berikut sebagai penanganan pertama pada kondisi kamu.'                                          : hrCategory == 'normal'                                              ? 'Denyut Jantung kamu normal nih, tetap pertahankan ya.'                                              : 'Data Denyut jantung masih kosong nih, yuk lakukan pengecekan pada smartwatch kamu.',                                  style: whiteTextStyle,                                ),                                const SizedBox(height: 10),                                hrCategory == 'rendah' || hrCategory == 'tinggi'                                    ? Row(                                        mainAxisAlignment: MainAxisAlignment.end,                                        children: [                                          Container(                                            decoration: BoxDecoration(                                              color: kWhiteColor,                                              borderRadius: BorderRadius.circular(                                                defaultRadius,                                              ),                                            ),                                            padding: const EdgeInsets.symmetric(                                              horizontal: 16,                                              vertical: 10,                                            ),                                            child: GestureDetector(                                              onTap: () {                                                Navigator.push(                                                  context,                                                  MaterialPageRoute(                                                    builder: (content) =>                                                        hrCategory == 'rendah'                                                            ? HeartRateLowPage(                                                                heartRate:                                                                    hrValue,                                                              )                                                            : HeartRateHighPage(                                                                heartRate:                                                                    hrValue,                                                              ),                                                  ),                                                );                                              },                                              child: Text(                                                'Lihat',                                                style: blackTextStyle.copyWith(                                                  fontWeight: bold,                                                ),                                              ),                                            ),                                          ),                                        ],                                      )                                    : const SizedBox(),                              ],                            ),                          ),                        ],                      ),                    ),                    const SizedBox(height: 10),                    Container(                      width: double.infinity,                      decoration: BoxDecoration(                        color: kPrimaryColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      padding: const EdgeInsets.all(16),                      child: Row(                        children: [                          Expanded(                            child: Column(                              crossAxisAlignment: CrossAxisAlignment.start,                              children: [                                Text(                                  boCategory == 'rendah'                                      ? 'Saturasi Oksigen Rendah'                                      : boCategory == 'normal'                                          ? 'Saturasi Oksigen Normal'                                          : 'Data Saturasi Oksigen Kosong',                                  style: whiteTextStyle.copyWith(                                    fontSize: 16,                                    fontWeight: bold,                                  ),                                ),                                const SizedBox(height: 10),                                Text(                                  boCategory == 'rendah'                                      ? 'Wah Saturasi Oksigen kamu rendah 😟 Yuk baca tips berikut sebagai penanganan pertama pada kondisi kamu.'                                      : boCategory == 'normal'                                          ? 'Saturasi Oksigen kamu normal nih, tetap pertahankan ya.'                                          : 'Data Saturasi Oksigen masih kosong nih, yuk lakukan pengecekan pada smartwatch kamu.',                                  style: whiteTextStyle,                                ),                                boCategory == 'rendah'                                    ? GestureDetector(                                        onTap: () {                                          Navigator.push(                                            context,                                            MaterialPageRoute(                                              builder: (context) =>                                                  BloodOxygenLowHandllingPage(                                                bloodOxygen: boValue,                                              ),                                            ),                                          );                                        },                                        child: Container(                                          decoration: BoxDecoration(                                            color: kWhiteColor,                                            borderRadius: BorderRadius.circular(                                              defaultRadius,                                            ),                                          ),                                          margin: const EdgeInsets.only(top: 10),                                          padding: const EdgeInsets.symmetric(                                            horizontal: 16,                                            vertical: 10,                                          ),                                          child: Text(                                            'Lihat',                                            style: blackTextStyle.copyWith(                                              fontWeight: bold,                                            ),                                          ),                                        ),                                      )                                    : const SizedBox(),                              ],                            ),                          ),                          Image.asset(                            'assets/announce2\_image.png',                            height: 75,                          ),                        ],                      ),                    ),                  ],                ),              );            }            return Container();          },        );      }      Widget article() {        return Container(          margin: EdgeInsets.only(            left: defaultMargin,            right: defaultMargin,            bottom: defaultMargin,          ),          child: Column(            children: [              Row(                mainAxisAlignment: MainAxisAlignment.spaceBetween,                children: [                  Text(                    'Artikel',                    style: blackTextStyle.copyWith(                      fontSize: 16,                      fontWeight: bold,                    ),                  ),                  GestureDetector(                    onTap: () {                      Navigator.pushNamed(context, '/article');                    },                    child: Text(                      'Semua',                      style: grayTextStyle.copyWith(                        fontWeight: bold,                      ),                    ),                  ),                ],              ),              Column(                children: List.generate(                  5,                  (index) => ArticleCard(                    listArticle[index],                  ),                ),              ),            ],          ),        );      }      return Scaffold(        backgroundColor: kWhiteColor,        body: ListView(          children: [            header(),            today(),            announce(),            article(),          ],        ),      );    }  } |

1. main\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/blocs/article/article\_bloc.dart';  import 'package:healio/blocs/health/health\_bloc.dart';  import 'package:healio/pages/health\_page.dart';  import 'package:healio/pages/home\_page.dart';  import 'package:healio/pages/user\_page.dart';  import 'package:healio/shared/theme.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import '../blocs/page/page\_cubit.dart';  class MainPage extends StatefulWidget {    const MainPage({Key? key}) : super(key: key);    @override    State<MainPage> createState() => \_MainPageState();  }  class \_MainPageState extends State<MainPage> {    @override    void initState() {      super.initState();      fetchData();    }    fetchData() {      context.read<HealthBloc>().add(HealthGetLandingPage());      context.read<HealthBloc>().add(HealthGetHeartRate());      context.read<HealthBloc>().add(HealthGetBloodOxygen());      context.read<ArticleBloc>().add(const ArticleGet('heart rate'));    }    @override    Widget build(BuildContext context) {      List pages = const [        HomePage(),        HealthPage(),        UserPage(),      ];      return BlocBuilder<PageCubit, int>(        builder: (context, currentIndex) {          return RefreshIndicator(            onRefresh: \_pullToRefresh,            child: Scaffold(              body: pages[currentIndex],              bottomNavigationBar: BottomNavigationBar(                type: BottomNavigationBarType.fixed,                backgroundColor: Colors.white,                onTap: (value) {                  context.read<PageCubit>().setPage(value);                },                currentIndex: currentIndex,                showSelectedLabels: false,                showUnselectedLabels: false,                items: [                  BottomNavigationBarItem(                    label: 'Home',                    icon: Column(                      children: [                        Image.asset(                          currentIndex == 0                              ? 'assets/icon\_home\_active.png'                              : 'assets/icon\_home.png',                          color: currentIndex == 0 ? kPrimaryColor : kGrayColor,                          width: 20,                          height: 20,                        ),                        const SizedBox(                          height: 5,                        ),                        Text(                          'Home',                          style: blackTextStyle.copyWith(                            fontSize: 12,                            fontWeight: currentIndex == 0 ? bold : reguler,                            color: currentIndex == 0 ? kPrimaryColor : kGrayColor,                          ),                        ),                      ],                    ),                  ),                  BottomNavigationBarItem(                    label: 'Kesehatan',                    icon: Column(                      children: [                        Image.asset(                          currentIndex == 1                              ? 'assets/icon\_heart\_active.png'                              : 'assets/icon\_heart.png',                          color: currentIndex == 1 ? kPrimaryColor : kGrayColor,                          height: 20,                          width: 20,                        ),                        const SizedBox(                          height: 5,                        ),                        Text(                          'Kesehatan',                          style: blackTextStyle.copyWith(                            fontSize: 12,                            fontWeight: currentIndex == 1 ? bold : reguler,                            color: currentIndex == 1 ? kPrimaryColor : kGrayColor,                          ),                        ),                      ],                    ),                  ),                  BottomNavigationBarItem(                    label: 'User',                    icon: Column(                      children: [                        Image.asset(                          currentIndex == 2                              ? 'assets/icon\_user\_active.png'                              : 'assets/icon\_user.png',                          color: currentIndex == 2 ? kPrimaryColor : kGrayColor,                          width: 20,                          height: 20,                        ),                        const SizedBox(                          height: 5,                        ),                        Text(                          'Akun',                          style: blackTextStyle.copyWith(                            fontSize: 12,                            fontWeight: currentIndex == 2 ? bold : reguler,                            color: currentIndex == 2 ? kPrimaryColor : kGrayColor,                          ),                        ),                      ],                    ),                  ),                ],              ),            ),          );        },      );    }    Future<void> \_pullToRefresh() async {      context.read<HealthBloc>().add(HealthGetLandingPage());    }  } |

1. sign\_in\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_button.dart';  import 'package:healio/widget/custom\_text\_form\_field.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../shared/shared\_methods.dart';  class SignInPage extends StatefulWidget {    const SignInPage({Key? key}) : super(key: key);    @override    State<SignInPage> createState() => \_SignInPageState();  }  final emailController = TextEditingController(text: 'julianreza@gmail.com');  final passwordController = TextEditingController(text: 'julianreza');  class \_SignInPageState extends State<SignInPage> {    @override    void initState() {      super.initState();    }    @override    Widget build(BuildContext context) {      bool validate() {        if (emailController.text.isEmpty || passwordController.text.isEmpty) {          return false;        }        return true;      }      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Masuk',                style: blackTextStyle.copyWith(                  fontSize: 24,                  fontWeight: bold,                ),              ),              Text(                'Masuk untuk memantau kondisi kesehatanmu.',                style: grayTextStyle,              ),            ],          ),        );      }      Widget form() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            children: [              CustomTextFormField(                label: 'Email',                icon: 'assets/fi\_mail.png',                controller: emailController,              ),              const SizedBox(height: 10),              CustomTextFormField(                label: 'Password',                icon: 'assets/fi\_lock.png',                isPassword: true,                obscureText: true,                controller: passwordController,              ),              const SizedBox(height: 24),              BlocConsumer<AuthBloc, AuthState>(                listener: (context, state) {                  if (state is AuthSuccess) {                    Navigator.pushNamedAndRemoveUntil(                        context, '/main', (route) => false);                  } else if (state is AuthFailed) {                    customShowSnackBar(context, state.e);                  }                },                builder: (context, state) {                  if (state is AuthLoading) {                    return CustomButton(                      onPressed: () {},                      child: SizedBox(                        height: 21,                        width: 21,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return CustomButton(                    onPressed: () {                      if (validate()) {                        context.read<AuthBloc>().add(                              AuthSignIn(                                emailController.text,                                passwordController.text,                              ),                            );                      } else {                        customShowSnackBar(context, 'Semua field harus diisi');                      }                    },                    child: Text(                      'Masuk',                      style: whiteTextStyle.copyWith(                        fontWeight: bold,                      ),                    ),                  );                },              ),              const SizedBox(height: 24),              Wrap(                crossAxisAlignment: WrapCrossAlignment.center,                children: [                  Text(                    'Belum punya akun? ',                    style: grayTextStyle,                  ),                  GestureDetector(                    onTap: () {                      Navigator.pushNamed(context, '/sign-up');                    },                    child: Text(                      'Daftar',                      style: primaryTextStyle,                    ),                  ),                ],              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            form(),          ],        ),      );    }  } |

1. sign\_up\_page.dart

|  |
| --- |
| import 'dart:developer';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_button.dart';  import 'package:healio/widget/custom\_text\_form\_field.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../shared/shared\_methods.dart';  class SignUpPage extends StatefulWidget {    const SignUpPage({Key? key}) : super(key: key);    @override    State<SignUpPage> createState() => \_SignUpPageState();  }  String gender = 'male';  final nameController = TextEditingController(text: '');  final emailController = TextEditingController(text: '');  final passwordController = TextEditingController(text: '');  String selectedDate = '';  class \_SignUpPageState extends State<SignUpPage> {    DateTime initialDate = DateTime.now();    @override    Widget build(BuildContext context) {      bool validate() {        if (emailController.text.isEmpty ||            passwordController.text.isEmpty ||            nameController.text.isEmpty ||            selectedDate == '' ||            gender == '') {          return false;        }        return true;      }      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Daftar',                style: blackTextStyle.copyWith(                  fontSize: 24,                  fontWeight: bold,                ),              ),              Text(                'Daftar untuk memantau kondisi kesehatanmu.',                style: grayTextStyle,              ),            ],          ),        );      }      Widget form() {        return Container(          margin: EdgeInsets.symmetric(            horizontal: defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              CustomTextFormField(                label: 'Nama',                icon: 'assets/fi\_user.png',                controller: nameController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Email',                icon: 'assets/fi\_mail.png',                controller: emailController,              ),              const SizedBox(height: 16),              CustomTextFormField(                label: 'Password',                icon: 'assets/fi\_lock.png',                isPassword: true,                obscureText: true,                controller: passwordController,              ),              const SizedBox(height: 16),              Text(                'Jenis Kelamin',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              Row(                children: [                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'male';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'male' ? kPrimaryColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.male,                              size: 20,                              color: gender == 'male' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Laki-laki',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'male' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                  const SizedBox(width: 16),                  Expanded(                    child: SizedBox(                      height: 46,                      child: ElevatedButton(                        onPressed: () {                          setState(() {                            gender = 'female';                          });                        },                        style: ElevatedButton.styleFrom(                          elevation: 0,                          primary:                              gender == 'female' ? kRedColor : kLightGrayColor,                          shape: RoundedRectangleBorder(                            borderRadius: BorderRadius.circular(                              defaultRadius,                            ),                          ),                        ),                        child: Wrap(                          crossAxisAlignment: WrapCrossAlignment.center,                          children: [                            Icon(                              Icons.female,                              size: 20,                              color:                                  gender == 'female' ? kWhiteColor : kGrayColor,                            ),                            const SizedBox(                              width: 10,                            ),                            Text(                              'Perempuan',                              style: whiteTextStyle.copyWith(                                color:                                    gender == 'female' ? kWhiteColor : kGrayColor,                              ),                            ),                          ],                        ),                      ),                    ),                  ),                ],              ),              const SizedBox(height: 16),              Text(                'Tanggal Lahir',                style: blackTextStyle.copyWith(                  fontSize: 16,                  fontWeight: bold,                ),              ),              const SizedBox(height: 10),              GestureDetector(                onTap: () {                  showDatePicker(                    context: context,                    initialDate: initialDate,                    firstDate: DateTime(1950),                    lastDate: DateTime(2050),                  ).then((value) {                    if (value != null) {                      setState(() {                        selectedDate = value.toString();                      });                    }                  });                },                child: Container(                  height: 57,                  decoration: BoxDecoration(                    color: kLightGrayColor,                    borderRadius: BorderRadius.circular(                      defaultRadius,                    ),                  ),                  padding: const EdgeInsets.symmetric(                    horizontal: 16,                  ),                  child: Row(                    children: [                      Image.asset(                        'assets/fi\_trello.png',                        height: 16,                        color: kGrayColor,                      ),                      const SizedBox(width: 10),                      Text(                        selectedDate,                        style: grayTextStyle,                      ),                    ],                  ),                ),              ),              const SizedBox(height: 16),              BlocConsumer<AuthBloc, AuthState>(                listener: (context, state) {                  if (state is AuthSuccess) {                    Navigator.pushNamedAndRemoveUntil(                        context, '/main', (route) => false);                  } else if (state is AuthFailed) {                    customShowSnackBar(context, state.e);                  }                },                builder: (context, state) {                  if (state is AuthLoading) {                    return CustomButton(                      onPressed: () {},                      child: SizedBox(                        height: 21,                        width: 21,                        child: CircularProgressIndicator(                          color: kWhiteColor,                        ),                      ),                    );                  }                  return CustomButton(                    onPressed: () {                      if (validate()) {                        context.read<AuthBloc>().add(                              AuthSignUp(                                emailController.text,                                passwordController.text,                                nameController.text,                                gender,                                selectedDate,                              ),                            );                      } else {                        customShowSnackBar(context, 'Semua field harus diisi');                      }                    },                    child: Text(                      'Daftar',                      style: whiteTextStyle.copyWith(                        fontWeight: bold,                      ),                    ),                  );                },              ),              const SizedBox(height: 24),              Center(                child: Wrap(                  crossAxisAlignment: WrapCrossAlignment.center,                  children: [                    Text(                      'Sudah punya akun? ',                      style: grayTextStyle,                    ),                    GestureDetector(                      onTap: () {                        Navigator.pushNamed(context, '/sign-in');                      },                      child: Text(                        'Masuk',                        style: primaryTextStyle,                      ),                    ),                  ],                ),              ),            ],          ),        );      }      return Scaffold(        body: ListView(          children: [            header(),            form(),          ],        ),      );    }  } |

1. splash\_page.dart

|  |
| --- |
| import 'dart:async';  import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/shared/theme.dart';  import '../blocs/auth/auth\_bloc.dart';  class SplashPage extends StatefulWidget {    const SplashPage({Key? key}) : super(key: key);    @override    State<SplashPage> createState() => \_SplashPageState();  }  class \_SplashPageState extends State<SplashPage> {    @override    void initState() {      Timer(const Duration(seconds: 2), () {        User? user = FirebaseAuth.instance.currentUser;        if (user == null) {          Navigator.pushNamedAndRemoveUntil(              context, '/sign-in', (route) => false);        } else {          context.read<AuthBloc>().add(AuthGetCurrentUser(user.uid));          Navigator.pushNamedAndRemoveUntil(context, '/main', (route) => false);        }      });      super.initState();    }    @override    Widget build(BuildContext context) {      return Scaffold(        body: Center(          child: Text(            'healio',            style: primaryTextStyle.copyWith(              fontSize: 56,              fontWeight: bold,            ),          ),        ),      );    }  } |

1. user\_page.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/blocs/page/page\_cubit.dart';  import 'package:healio/pages/edit\_date\_of\_birth\_page.dart';  import 'package:healio/pages/edit\_gender\_page.dart';  import 'package:healio/pages/edit\_name\_page.dart';  import 'package:healio/shared/shared\_methods.dart';  import 'package:healio/shared/theme.dart';  import 'package:healio/widget/custom\_button.dart';  import 'package:healio/widget/user\_about\_item.dart';  import 'package:health/health.dart';  import 'package:intl/intl.dart';  import '../blocs/auth/auth\_bloc.dart';  class UserPage extends StatelessWidget {    const UserPage({Key? key}) : super(key: key);    @override    Widget build(BuildContext context) {      Widget header() {        return Container(          margin: EdgeInsets.all(            defaultMargin,          ),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text(                'Akun',                style: blackTextStyle.copyWith(                  fontSize: 26,                  fontWeight: bold,                ),              ),              const SizedBox(height: 5),              Text(                'Informasi terkait akun kamu',                style: grayTextStyle,              ),            ],          ),        );      }      Widget body() {        return BlocBuilder<AuthBloc, AuthState>(          builder: (context, state) {            if (state is AuthSuccess) {              final date = DateTime.parse(state.user.dateOfBirth);              String formattedDate = DateFormat('y-MM-d').format(date);              return Container(                margin: EdgeInsets.symmetric(                  horizontal: defaultMargin,                ),                child: Column(                  crossAxisAlignment: CrossAxisAlignment.start,                  children: [                    Center(                      child: Column(                        children: [                          Container(                            height: 84,                            width: 84,                            margin: const EdgeInsets.only(bottom: 10),                            decoration: const BoxDecoration(                              image: DecorationImage(                                image: AssetImage(                                  'assets/avatar.png',                                ),                                fit: BoxFit.cover,                              ),                              shape: BoxShape.circle,                            ),                          ),                          Column(                            crossAxisAlignment: CrossAxisAlignment.center,                            children: [                              Text(                                state.user.name,                                style: blackTextStyle.copyWith(                                  fontSize: 24,                                  fontWeight: bold,                                ),                              ),                              Text(                                state.user.email,                                style: grayTextStyle,                              ),                            ],                          ),                        ],                      ),                    ),                    SizedBox(                      height: defaultMargin,                    ),                    Text(                      'Tentang',                      style: blackTextStyle.copyWith(                        fontSize: 16,                        fontWeight: bold,                      ),                    ),                    const SizedBox(height: 10),                    Container(                      padding: const EdgeInsets.symmetric(vertical: 10),                      decoration: BoxDecoration(                        color: kLightGrayColor,                        borderRadius: BorderRadius.circular(                          defaultRadius,                        ),                      ),                      margin: EdgeInsets.only(                        bottom: defaultMargin,                      ),                      child: Column(                        children: [                          UserAboutItem(                            label: 'Nama',                            value: state.user.name,                            page: EditNamePage(                              name: state.user.name,                            ),                          ),                          UserAboutItem(                            label: 'Jenis Kelamin',                            value: state.user.gender == 'male'                                ? 'Laki-laki'                                : 'Perempuan',                            page: const EditGenderPage(),                          ),                          UserAboutItem(                            label: 'Tanggal Lahir',                            value: formattedDate,                            page: EditDateOfBirthPage(                              date: state.user.dateOfBirth,                            ),                          ),                        ],                      ),                    ),                    const SizedBox(                      height: 10,                    ),                    BlocConsumer<AuthBloc, AuthState>(                      listener: (context, state) {                        if (state is AuthFailed) {                          customShowSnackBar(context, state.e);                        } else if (state is AuthInitial) {                          Navigator.pushNamedAndRemoveUntil(                              context, '/sign-in', (route) => false);                        }                      },                      builder: (context, state) {                        if (state is AuthLoading) {                          return CustomButton(                            onPressed: () {},                            child: SizedBox(                              height: 21,                              width: 21,                              child: CircularProgressIndicator(                                color: kWhiteColor,                              ),                            ),                          );                        }                        return CustomButton(                          onPressed: () {                            context.read<AuthBloc>().add(AuthSignOut());                            context.read<PageCubit>().setPage(0);                            Navigator.pushNamedAndRemoveUntil(                                context, '/sign-in', (route) => false);                          },                          child: Wrap(                            crossAxisAlignment: WrapCrossAlignment.center,                            children: [                              Image.asset(                                'assets/fi\_log-out.png',                                height: 16,                                color: kWhiteColor,                              ),                              const SizedBox(width: 10),                              Text(                                'Logout',                                style: whiteTextStyle.copyWith(                                  fontWeight: bold,                                ),                              ),                            ],                          ),                        );                      },                    ),                  ],                ),              );            }            return Container();          },        );      }      return Scaffold(        body: ListView(          children: [            header(),            body(),          ],        ),      );    }  } |

1. webview\_page.dart

|  |
| --- |
| import 'dart:io';  import 'package:flutter/material.dart';  import 'package:webview\_flutter/webview\_flutter.dart';  import '../shared/theme.dart';  class WebviewPage extends StatefulWidget {    const WebviewPage({required this.url, Key? key}) : super(key: key);    final String url;    @override    State<WebviewPage> createState() => \_WebviewPageState();  }  class \_WebviewPageState extends State<WebviewPage> {    @override    void initState() {      super.initState();      if (Platform.isAndroid) WebView.platform = AndroidWebView();    }    @override    Widget build(BuildContext context) {      return Scaffold(        body: SafeArea(          child: Column(            children: [              GestureDetector(                onTap: () {                  Navigator.pop(context);                },                child: Container(                  width: double.infinity,                  padding: const EdgeInsets.all(16),                  color: kPrimaryColor,                  child: Row(                    children: [                      Icon(                        Icons.chevron\_left,                        size: 24,                        color: kWhiteColor,                      ),                      const SizedBox(                        width: 10,                      ),                      Text(                        'Kembali ke aplikasi',                        style: whiteTextStyle.copyWith(                          fontSize: 14,                          fontWeight: bold,                        ),                      ),                    ],                  ),                ),              ),              Expanded(                child: WebView(                  initialUrl: widget.url,                ),              ),            ],          ),        ),      );    }  } |

1. article\_card.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/models/article\_model.dart';  import '../pages/webview\_page.dart';  import '../shared/theme.dart';  class ArticleCard extends StatelessWidget {    const ArticleCard(      this.article, {      Key? key,    }) : super(key: key);    final ArticleModel article;    @override    Widget build(BuildContext context) {      return Container(        margin: const EdgeInsets.only(top: 10),        decoration: BoxDecoration(          boxShadow: [            BoxShadow(              color: kGrayColor.withOpacity(0.1),              offset: const Offset(3, 3),              blurRadius: 10,            ),          ],        ),        width: double.infinity,        child: ClipRRect(          borderRadius: BorderRadius.circular(12),          child: GestureDetector(            onTap: () {              Navigator.push(                context,                MaterialPageRoute(                  builder: (context) => WebviewPage(                    url: article.link.toString(),                  ),                ),              );            },            child: Container(              decoration: BoxDecoration(                color: kWhiteColor,                border: Border.all(                  color: kLightGrayColor,                ),              ),              child: Row(                crossAxisAlignment: CrossAxisAlignment.start,                children: [                  Container(                    height: 96,                    width: 144,                    decoration: BoxDecoration(                      image: DecorationImage(                        image: NetworkImage(                          article.thumbnail.toString(),                        ),                        fit: BoxFit.cover,                      ),                    ),                  ),                  Expanded(                    child: Container(                      padding: const EdgeInsets.all(10),                      child: Column(                        crossAxisAlignment: CrossAxisAlignment.start,                        mainAxisAlignment: MainAxisAlignment.start,                        children: [                          Text(                            article.title.toString(),                            style: blackTextStyle.copyWith(                              fontWeight: bold,                            ),                          ),                          const SizedBox(                            height: 5,                          ),                          Text(                            article.description!,                            style: grayTextStyle.copyWith(                              fontSize: 14,                            ),                            maxLines: 1,                            overflow: TextOverflow.ellipsis,                          ),                        ],                      ),                    ),                  ),                ],              ),            ),          ),        ),      );    }  } |

1. custom\_app\_bar.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class CustomAppBar extends StatelessWidget {    const CustomAppBar({      required this.title,      this.action = const SizedBox(),      Key? key,    }) : super(key: key);    final String title;    final Widget? action;    @override    Widget build(BuildContext context) {      return Container(        margin: EdgeInsets.all(          defaultMargin,        ),        child: Row(          children: [            Row(              children: [                GestureDetector(                  onTap: () {                    Navigator.pop(context);                  },                  child: Container(                    padding: const EdgeInsets.all(8),                    decoration: BoxDecoration(                      color: kLightGrayColor,                      borderRadius: BorderRadius.circular(                        defaultRadius,                      ),                    ),                    child: const Icon(                      Icons.chevron\_left,                      size: 24,                    ),                  ),                ),                const SizedBox(width: 10),                Text(                  title,                  style: blackTextStyle.copyWith(                    fontSize: 16,                    fontWeight: bold,                  ),                )              ],            ),            const Spacer(),            SizedBox(              child: action,            ),          ],        ),      );    }  } |

1. custom\_button.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class CustomButton extends StatelessWidget {    const CustomButton({      required this.onPressed,      required this.child,      Key? key,    }) : super(key: key);    final Widget child;    final Function() onPressed;    @override    Widget build(BuildContext context) {      return SizedBox(        height: 57,        width: double.infinity,        child: ElevatedButton(          onPressed: () {            onPressed();          },          style: ElevatedButton.styleFrom(            primary: kPrimaryColor,            shape: RoundedRectangleBorder(              borderRadius: BorderRadius.circular(                defaultRadius,              ),            ),          ),          child: child,        ),      );    }  } |

1. custom\_text\_form\_field.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class CustomTextFormField extends StatefulWidget {    const CustomTextFormField({      required this.label,      this.icon,      this.isPassword = false,      this.obscureText = false,      required this.controller,      this.hint,      Key? key,    }) : super(key: key);    final String label;    final bool isPassword;    final bool obscureText;    final String? icon;    final TextEditingController controller;    final String? hint;    @override    State<CustomTextFormField> createState() => \_CustomTextFormFieldState();  }  bool isShow = false;  class \_CustomTextFormFieldState extends State<CustomTextFormField> {    @override    void initState() {      super.initState();      isShow = widget.obscureText;    }    @override    Widget build(BuildContext context) {      return Column(        crossAxisAlignment: CrossAxisAlignment.start,        children: [          Text(            widget.label,            style: blackTextStyle.copyWith(              fontSize: 16,              fontWeight: bold,            ),          ),          const SizedBox(            height: 10,          ),          Container(            height: 57,            decoration: BoxDecoration(              color: kLightGrayColor,              borderRadius: BorderRadius.circular(                defaultRadius,              ),            ),            padding: const EdgeInsets.symmetric(              horizontal: 16,              vertical: 10,            ),            child: Row(              children: [                widget.icon != null                    ? Row(                        children: [                          Image.asset(                            widget.icon!,                            height: 16,                            color: kGrayColor,                          ),                          const SizedBox(width: 10),                        ],                      )                    : const SizedBox(),                Expanded(                  child: TextFormField(                    controller: widget.controller,                    decoration: InputDecoration.collapsed(                      hintText: widget.hint ?? 'Masukan ${widget.label}',                      hintStyle: grayTextStyle,                    ),                    obscureText: isShow ? true : false,                  ),                ),                widget.isPassword                    ? GestureDetector(                        onTap: () {                          setState(() {                            isShow = !isShow;                          });                        },                        child: Container(                          margin: const EdgeInsets.only(left: 10),                          child: Image.asset(                            isShow                                ? 'assets/fi\_eye.png'                                : 'assets/fi\_eye-off.png',                            height: 16,                            color: kGrayColor,                          ),                        ),                      )                    : const SizedBox(),              ],            ),          ),        ],      );    }  } |

1. health\_category\_tile.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:healio/shared/theme.dart';  enum HealthCategory {    heartrate,    bloodoxygen,  }  class HealthCategoryTile extends StatelessWidget {    const HealthCategoryTile({      required this.healthCategory,      Key? key,    }) : super(key: key);    final HealthCategory healthCategory;    @override    Widget build(BuildContext context) {      return GestureDetector(        onTap: () {          Navigator.pushNamed(            context,            healthCategory == HealthCategory.heartrate                ? '/heart-rate'                : '/blood-oxygen',          );        },        child: Container(          padding: const EdgeInsets.symmetric(            horizontal: 16,            vertical: 16,          ),          decoration: BoxDecoration(            color: kLightGrayColor,            borderRadius: BorderRadius.circular(              12,            ),          ),          child: Row(            crossAxisAlignment: CrossAxisAlignment.center,            children: [              Text(                healthCategory == HealthCategory.heartrate                    ? 'Denyut Jantung'                    : 'Saturasi Oksigen',                style: blackTextStyle,              ),              const Spacer(),              Container(                height: 50,                width: 50,                decoration: BoxDecoration(                  color: kWhiteColor,                  shape: BoxShape.circle,                ),                padding: const EdgeInsets.all(13),                child: Image.asset(                  healthCategory == HealthCategory.heartrate                      ? 'assets/fi\_heart.png'                      : 'assets/fi\_droplet.png',                  color: healthCategory == HealthCategory.heartrate                      ? kRedColor                      : kPrimaryColor,                ),              ),            ],          ),        ),      );    }  } |

1. health\_data\_item.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:flutter\_bloc/flutter\_bloc.dart';  import 'package:healio/pages/blood\_oxygen\_page.dart';  import 'package:health/health.dart';  import 'package:intl/intl.dart';  import 'package:healio/shared/shared\_methods.dart';  import '../blocs/auth/auth\_bloc.dart';  import '../shared/theme.dart';  int age = 0;  class HealthDataItem extends StatefulWidget {    const HealthDataItem(      this.healthDataPoint, {      this.isAll = false,      this.category = 'heart\_rate',      Key? key,    }) : super(key: key);    final HealthDataPoint healthDataPoint;    final bool isAll;    final String category;    @override    State<HealthDataItem> createState() => \_HealthDataItemState();  }  class \_HealthDataItemState extends State<HealthDataItem> {    @override    void initState() {      super.initState();      getAge();    }    void getAge() {      final authState = context.read<AuthBloc>().state;      if (authState is AuthSuccess) {        age = calculateAge(DateTime.parse(authState.user.dateOfBirth));      }    }    @override    Widget build(BuildContext context) {      String getTodayDate =          DateFormat("Hm").format(widget.healthDataPoint.dateFrom);      String getAllDate =          DateFormat("d MMMM y h:m aaa").format(widget.healthDataPoint.dateFrom);      String labelCategory = setCategory(        getValueFromHealthValue(widget.healthDataPoint.value),        widget.category,        age,      );      return Container(        padding: const EdgeInsets.symmetric(          horizontal: 16,          vertical: 16,        ),        decoration: BoxDecoration(          color: kLightGrayColor,          borderRadius: BorderRadius.circular(            defaultRadius,          ),          boxShadow: [            BoxShadow(              color: kLightGrayColor,              offset: const Offset(3, 3),              blurRadius: 10,            ),          ],        ),        margin: const EdgeInsets.only(          top: 10,        ),        child: Row(          crossAxisAlignment: CrossAxisAlignment.center,          children: [            Column(              crossAxisAlignment: CrossAxisAlignment.start,              children: [                Wrap(                  crossAxisAlignment: WrapCrossAlignment.end,                  children: [                    Text(                      '${getValueFromHealthValue(widget.healthDataPoint.value)}',                      style: blackTextStyle.copyWith(                        fontSize: 20,                        fontWeight: bold,                      ),                    ),                    Text(                      widget.category == 'heart\_rate' ? 'BPM' : '%',                      style: grayTextStyle.copyWith(                        fontSize: widget.category == 'heart\_rate' ? 14 : 20,                      ),                    ),                  ],                ),                const SizedBox(height: 5),                Container(                  padding: const EdgeInsets.symmetric(                    horizontal: 8,                    vertical: 3,                  ),                  decoration: BoxDecoration(                    color: (labelCategory == 'normal' ? kGreenColor : kRedColor)                        .withOpacity(0.1),                    borderRadius: BorderRadius.circular(5),                  ),                  child: Text(                    labelCategory,                    style: greenTextStyle.copyWith(                      fontSize: 12,                      color: labelCategory == 'normal' ? kGreenColor : kRedColor,                    ),                  ),                )              ],            ),            const Spacer(),            Text(              widget.isAll ? getAllDate : getTodayDate,              style: grayTextStyle,            ),          ],        ),      );    }  } |

1. more\_button.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class MoreButton extends StatelessWidget {    const MoreButton({      required this.text,      required this.route,      Key? key,    }) : super(key: key);    final String text;    final String route;    @override    Widget build(BuildContext context) {      return GestureDetector(        onTap: () {          Navigator.pushNamed(context, route);        },        child: Container(          margin: EdgeInsets.only(            left: defaultMargin,            right: defaultMargin,            top: 10,          ),          padding: const EdgeInsets.all(16),          decoration: BoxDecoration(            color: kLightGrayColor,            borderRadius: BorderRadius.circular(              defaultRadius,            ),          ),          child: Row(            children: [              Text(                text,                style: blackTextStyle.copyWith(                  fontSize: 14,                ),              ),              const Spacer(),              const Icon(                Icons.chevron\_right,              ),            ],          ),        ),      );    }  } |

1. user\_about\_item.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import '../shared/theme.dart';  class UserAboutItem extends StatelessWidget {    const UserAboutItem({      required this.label,      required this.value,      required this.page,      Key? key,    }) : super(key: key);    final String label;    final String value;    final Widget page;    @override    Widget build(BuildContext context) {      return Container(        padding: const EdgeInsets.symmetric(          vertical: 10,          horizontal: 16,        ),        child: Row(          children: [            Text(              label,              style: blackTextStyle,            ),            const Spacer(),            GestureDetector(              onTap: () {                Navigator.push(                  context,                  MaterialPageRoute(                    builder: (context) => page,                  ),                );              },              child: Wrap(                crossAxisAlignment: WrapCrossAlignment.center,                children: [                  Text(                    value,                    style: grayTextStyle,                  ),                  const SizedBox(width: 5),                  Icon(                    Icons.chevron\_right,                    color: kGrayColor,                  ),                ],              ),            )          ],        ),      );    }  } |

1. shared\_methods.dart

|  |
| --- |
| import 'package:another\_flushbar/flushbar.dart';  import 'package:flutter/cupertino.dart';  import 'package:healio/shared/theme.dart';  import 'package:health/health.dart';  int getValueFromHealthValue(HealthValue healthValue) {    List<String> value = healthValue.toString().split('.');    return int.parse(value[0]);  }  void customShowSnackBar(BuildContext context, String message) {    Flushbar(      flushbarPosition: FlushbarPosition.TOP,      message: message,      backgroundColor: kRedColor,      duration: const Duration(seconds: 2),    ).show(context);  }  String setCategory(    int value,    String category,    int age,  ) {    String label = "";    if (category == 'heart\_rate') {      if (age <= 1) {        if (value < 70) {          label = 'rendah';        } else if (value >= 70 && value < 190) {          label = 'normal';        } else {          label = 'tinggi';        }      } else if (age > 1 && age <= 10) {        if (value < 80) {          label = 'rendah';        } else if (value >= 80 && value < 110) {          label = 'normal';        } else {          label = 'tinggi';        }      } else if (age > 10) {        if (value < 60) {          label = 'rendah';        } else if (value >= 60 && value < 100) {          label = 'normal';        } else {          label = 'tinggi';        }      }    } else if (category == 'blood\_oxygen') {      if (value < 95) {        label = 'rendah';      } else {        label = 'normal';      }    }    return label;  }  calculateAge(DateTime dateOfBirth) {    DateTime currentDate = DateTime.now();    int age = currentDate.year - dateOfBirth.year;    int month1 = currentDate.month;    int month2 = dateOfBirth.month;    if (month2 > month1) {      age--;    } else if (month1 == month2) {      int day1 = currentDate.day;      int day2 = dateOfBirth.day;      if (day2 > day1) {        age--;      }    }    return age;  } |

1. theme.dart

|  |
| --- |
| import 'package:flutter/material.dart';  import 'package:google\_fonts/google\_fonts.dart';  double defaultMargin = 24.0;  double defaultRadius = 12;  Color kGrayColor = const Color(0xff9B9B9B);  Color kBlackColor = const Color(0xff252525);  Color kPrimaryColor = const Color(0xff0572f1);  Color kLightGrayColor = const Color(0xfff3f4f8);  Color kSecondaryColor = const Color(0xffB6C1EA);  Color kWhiteColor = const Color(0xfffdfdfd);  Color kTransparentColor = Colors.transparent;  Color kRedColor = const Color(0xffFF2566);  Color kGreenColor = const Color(0xff2ecc71);  TextStyle blackTextStyle = GoogleFonts.nunito(    color: kBlackColor,  );  TextStyle primaryTextStyle = GoogleFonts.nunito(    color: kPrimaryColor,  );  TextStyle grayTextStyle = GoogleFonts.nunito(    color: kGrayColor,  );  TextStyle whiteTextStyle = GoogleFonts.nunito(    color: kWhiteColor,  );  TextStyle lightGrayTextStyle = GoogleFonts.nunito(    color: kLightGrayColor,  );  TextStyle redTextStyle = GoogleFonts.nunito(    color: kRedColor,  );  TextStyle greenTextStyle = GoogleFonts.nunito(    color: kGreenColor,  );  FontWeight light = FontWeight.w300;  FontWeight reguler = FontWeight.w400;  FontWeight medium = FontWeight.w500;  FontWeight semiBold = FontWeight.w600;  FontWeight bold = FontWeight.w700;  FontWeight extraBold = FontWeight.w800;  FontWeight black = FontWeight.w900; |

1. article\_model.dart

|  |
| --- |
| class ArticleModel {    String? title;    String? description;    String? link;    String? thumbnail;    ArticleModel({      this.title = '',      this.description = '',      this.link = '',      this.thumbnail = '',    });    factory ArticleModel.fromJson(Map<String, dynamic> json) => ArticleModel(          title: json['title'],          description: json['description'],          link: json['link'],          thumbnail: json['thumbnail'],        );  } |

1. health\_model.dart

|  |
| --- |
| class HealthModel {    final String type;    final int value;    final String unit;    final String dateFrom;    HealthModel({      required this.type,      required this.value,      required this.unit,      required this.dateFrom,    });    factory HealthModel.fromJson(Map<String, dynamic> json) => HealthModel(          type: json['type'],          value: json['value'],          unit: json['unit'],          dateFrom: json['dateFrom'],        );    Map<String, dynamic> toJson() => {          'type': type,          'value': value,          'unit': unit,          'dateFrom': dateFrom,        };  } |

1. heart\_disease\_form\_model.dart

|  |
| --- |
| class HeartDiseaseFormModel {    final int? age;    final int? sex;    final int? cigs;    final int? chol;    final int? sBP;    final int? dia;    final int? dBP;    final int? gluc;    final int? hRate;    HeartDiseaseFormModel({      this.age,      this.sex,      this.cigs,      this.chol,      this.sBP,      this.dia,      this.dBP,      this.gluc,      this.hRate,    });    Map<String, dynamic> toJson() => {          'age': age,          'sex': sex,          'cigs': cigs,          'chol': chol,          'sBP': sBP,          'dia': dia,          'dBP': dBP,          'gluc': gluc,          'hRate': hRate,        };  } |

1. user\_model.dart

|  |
| --- |
| class UserModel {    final String id;    final String email;    final String name;    final String gender;    final String dateOfBirth;    UserModel(        {required this.id,        required this.email,        required this.name,        required this.gender,        required this.dateOfBirth});    factory UserModel.fromJson(String id, Map<String, dynamic> json) => UserModel(          id: id,          email: json['email'],          name: json['name'],          gender: json['gender'],          dateOfBirth: json['date\_of\_birth'],        );    Map<String, dynamic> toJson() => {          'id': id,          'email': email,          'name': name,          'gender': gender,          'date\_of\_birth': dateOfBirth,        };  } |

1. auth\_service.dart

|  |
| --- |
| import 'package:firebase\_auth/firebase\_auth.dart';  import 'package:healio/services/user\_service.dart';  import '../models/user\_model.dart';  class AuthService {    final FirebaseAuth \_auth = FirebaseAuth.instance;    Future<UserModel> signIn({      required String email,      required String password,    }) async {      try {        UserCredential userCredential = await \_auth.signInWithEmailAndPassword(          email: email,          password: password,        );        UserModel user =            await UserService().getUserById(userCredential.user!.uid);        return user;      } catch (e) {        rethrow;      }    }    Future<UserModel> signUp({      required String email,      required String password,      required String name,      required String gender,      required String dateOfBirth,    }) async {      try {        UserCredential userCredential = await \_auth            .createUserWithEmailAndPassword(email: email, password: password);        UserModel user = UserModel(          id: userCredential.user!.uid,          email: email,          name: name,          gender: gender,          dateOfBirth: dateOfBirth,        );        await UserService().setUser(user);        return user;      } catch (e) {        rethrow;      }    }    Future<void> signOut() async {      try {        await \_auth.signOut();      } catch (e) {        rethrow;      }    }  } |

1. health\_service.dart

|  |
| --- |
| import 'dart:developer';  import 'package:cloud\_firestore/cloud\_firestore.dart';  import 'package:health/health.dart';  import 'package:permission\_handler/permission\_handler.dart';  import '../models/health\_model.dart';  class HealthService {    final CollectionReference \_healthReference =        FirebaseFirestore.instance.collection('health');    Future<List<HealthDataPoint>> fetchData(      DateTime now,      DateTime yesterday,      List<HealthDataType> types,    ) async {      List<HealthDataPoint> healthDataList = [];      HealthFactory health = HealthFactory();      final permissions = [        HealthDataAccess.READ,      ];      bool requested =          await health.requestAuthorization(types, permissions: permissions);      await Permission.activityRecognition.request();      await Permission.location.request();      if (requested) {        try {          List<HealthDataPoint> healthData =              await health.getHealthDataFromTypes(yesterday, now, types);          healthDataList.addAll((healthData.length < 100)              ? healthData              : healthData.sublist(0, 100));        } catch (error) {          throw Exception("Exception in getHealthDataFromTypes: $error");        }        healthDataList = HealthFactory.removeDuplicates(healthDataList);        healthDataList.sort((a, b) {          var aDate = a.dateFrom;          var bDate = b.dateFrom;          return bDate.compareTo(aDate);        });        return healthDataList;      } else {        throw Exception("Authorization not granted");      }    }    Future<HealthModel> getHeartRateFromFirebase() async {      try {        DocumentSnapshot snapshot =            await \_healthReference.doc('heart\_rate').get();        return HealthModel(          type: snapshot['type'],          value: snapshot['value'],          unit: snapshot['unit'],          dateFrom: snapshot['dateFrom'],        );      } catch (e) {        throw Exception(e);      }    }    Future<void> updateHeartRateFromFirebase(      int value,      String dateFrom,    ) async {      try {        \_healthReference.doc('heart\_rate').update({          'value': value,          'dateFrom': dateFrom,        });      } catch (e) {        throw Exception(e);      }    }    Future<HealthModel> getBloodOxygenFromFirebase() async {      try {        DocumentSnapshot snapshot =            await \_healthReference.doc('blood\_oxygen').get();        return HealthModel(          type: snapshot['type'],          value: snapshot['value'],          unit: snapshot['unit'],          dateFrom: snapshot['dateFrom'],        );      } catch (e) {        throw Exception(e);      }    }    Future<void> updateBloodOxygenFromFirebase(      int value,      String dateFrom,    ) async {      try {        \_healthReference.doc('blood\_oxygen').update({          'value': value,          'dateFrom': dateFrom,        });      } catch (e) {        throw Exception(e);      }    }  } |

1. heart\_disease\_service.dart

|  |
| --- |
| import 'dart:convert';  import 'dart:developer';  import 'package:healio/models/heart\_disease\_form\_model.dart';  import 'package:http/http.dart' as http;  class HeartDiseaseService {    String baseUrl = 'https://heartapi.herokuapp.com';    Future<Map<String, dynamic>> prediction(HeartDiseaseFormModel data) async {      var url = Uri.parse(          '$baseUrl/predict?age=${data.age}&sex=${data.sex}&cigs=${data.cigs}&chol=${data.chol}&sBP=${data.sBP}&dia=${data.dia}&dBP=${data.dBP}&gluc=${data.gluc}&hRate=${data.hRate}');      var res = await http.get(url);      if (res.statusCode == 200) {        var result = jsonDecode(res.body);        return result;      } else {        throw Exception('Gagal memprediksi');      }    }  } |

1. push\_notification\_service.dart

|  |
| --- |
| import 'dart:convert';  import 'package:http/http.dart' as http;  class PushNotificationService {    String baseUrl = "https://fcm.googleapis.com/fcm/send";    Future<bool> pushNotification(      String title,      String desc,    ) async {      var url = Uri.parse(baseUrl);      var headers = {        'Content-Type': 'application/json',        'Authorization':            'key=AAAAstAiflM:APA91bHcn\_64JkornQ-u6KRSZtE-C\_n2-dHgV7PzqQj7kj83AVAK43AfXGM659FxEjqxH2bQXwE6MbNhzOatFTOAq9kLGjwhMbDtgy5AA2RapIdNhRiMMJHNBr9vTM-6NHSS-c6Vc23x',      };      var body = jsonEncode({        "to": "/topics/health",        "collapse\_key": "type\_a",        "notification": {          "title": title,          "body": desc,        }      });      var response = await http.post(        url,        headers: headers,        body: body,      );      if (response.statusCode == 200) {        return true;      } else {        return false;      }    }  } |

1. user\_service.dart

|  |
| --- |
| import 'package:cloud\_firestore/cloud\_firestore.dart';  import '../models/user\_model.dart';  class UserService {    final CollectionReference \_userReference =        FirebaseFirestore.instance.collection('users');    Future<void> setUser(UserModel user) async {      try {        \_userReference.doc(user.id).set({          'email': user.email,          'name': user.name,          'gender': user.gender,          'date\_of\_birth': user.dateOfBirth,        });      } catch (e) {        throw Exception(e);      }    }    Future<UserModel> getUserById(String id) async {      try {        DocumentSnapshot snapshot = await \_userReference.doc(id).get();        return UserModel(          id: id,          email: snapshot['email'],          name: snapshot['name'],          gender: snapshot['gender'],          dateOfBirth: snapshot['date\_of\_birth'],        );      } catch (e) {        throw Exception(e);      }    }    Future<UserModel> updateUser(        {required String label,        required String value,        required String userId}) async {      try {        \_userReference.doc(userId).update({          label: value,        });        UserModel user = await getUserById(userId);        return user;      } catch (e) {        throw Exception(e);      }    }  } |

1. auth\_bloc.dart

|  |
| --- |
| import 'package:bloc/bloc.dart';  import 'package:equatable/equatable.dart';  import 'package:healio/services/user\_service.dart';  import '../../models/user\_model.dart';  import '../../services/auth\_service.dart';  part 'auth\_event.dart';  part 'auth\_state.dart';  class AuthBloc extends Bloc<AuthEvent, AuthState> {    AuthBloc() : super(AuthInitial()) {      on<AuthEvent>((event, emit) async {        if (event is AuthSignUp) {          try {            emit(AuthLoading());            UserModel user = await AuthService().signUp(              email: event.email,              password: event.password,              name: event.name,              gender: event.gender,              dateOfBirth: event.dateOfBirth,            );            emit(AuthSuccess(user));          } catch (e) {            emit(AuthFailed(e.toString()));          }        }        if (event is AuthSignIn) {          try {            emit(AuthLoading());            UserModel user = await AuthService().signIn(              email: event.email,              password: event.password,            );            emit(AuthSuccess(user));          } catch (e) {            emit(AuthFailed(e.toString()));          }        }        if (event is AuthSignOut) {          try {            emit(AuthLoading());            await AuthService().signOut();            emit(AuthInitial());          } catch (e) {            emit(AuthFailed(e.toString()));          }        }        if (event is AuthGetCurrentUser) {          try {            emit(AuthLoading());            UserModel user = await UserService().getUserById(event.id);            emit(AuthSuccess(user));          } catch (e) {            emit(AuthFailed(e.toString()));          }        }      });    }  } |

1. auth\_event.dart

|  |
| --- |
| part of 'auth\_bloc.dart';  abstract class AuthEvent extends Equatable {    const AuthEvent();    @override    List<Object> get props => [];  }  class AuthSignUp extends AuthEvent {    final String email;    final String password;    final String name;    final String gender;    final String dateOfBirth;    const AuthSignUp(        this.email, this.password, this.name, this.gender, this.dateOfBirth);    @override    List<Object> get props => [email, password, name, gender, dateOfBirth];  }  class AuthSignIn extends AuthEvent {    final String email;    final String password;    const AuthSignIn(this.email, this.password);    @override    List<Object> get props => [email, password];  }  class AuthSignOut extends AuthEvent {}  class AuthGetCurrentUser extends AuthEvent {    final String id;    const AuthGetCurrentUser(this.id);    @override    List<Object> get props => [id];  } |

1. auth\_state.dart

|  |
| --- |
| part of 'auth\_bloc.dart';  abstract class AuthState extends Equatable {    const AuthState();    @override    List<Object> get props => [];  }  class AuthInitial extends AuthState {}  class AuthLoading extends AuthState {}  class AuthFailed extends AuthState {    final String e;    const AuthFailed(this.e);    @override    List<Object> get props => [e];  }  class AuthSuccess extends AuthState {    final UserModel user;    const AuthSuccess(this.user);    @override    List<Object> get props => [user];  } |

1. health\_bloc.dart

|  |
| --- |
| import 'package:bloc/bloc.dart';  import 'package:equatable/equatable.dart';  import 'package:health/health.dart';  import '../../services/health\_service.dart';  part 'health\_event.dart';  part 'health\_state.dart';  class HealthBloc extends Bloc<HealthEvent, HealthState> {    HealthBloc() : super(HealthInitial()) {      on<HealthEvent>((event, emit) async {        if (event is HealthGetHeartRate) {          try {            DateTime now = DateTime.now();            DateTime yesterday = now.subtract(const Duration(days: 1));            final types = [              HealthDataType.HEART\_RATE,            ];            emit(HealthLoading());            final heartRate =                await HealthService().fetchData(now, yesterday, types);            emit(HealthHeartRateSuccess(heartRate));          } catch (e) {            emit(HealthFailed(e.toString()));          }        }        if (event is HealthGetAllHeartRate) {          try {            DateTime now = DateTime.now();            DateTime yesterday = now.subtract(const Duration(days: 30));            final types = [              HealthDataType.HEART\_RATE,            ];            emit(HealthLoading());            final allHeartRate =                await HealthService().fetchData(now, yesterday, types);            emit(HealthAllHeartRateSuccess(allHeartRate));          } catch (e) {            emit(HealthFailed(e.toString()));          }        }        if (event is HealthGetBloodOxygen) {          try {            DateTime now = DateTime.now();            DateTime yesterday = now.subtract(const Duration(days: 1));            final types = [              HealthDataType.BLOOD\_OXYGEN,            ];            emit(HealthLoading());            final bloodOxygen =                await HealthService().fetchData(now, yesterday, types);            emit(HealthBloodOxygenSuccess(bloodOxygen));          } catch (e) {            emit(HealthFailed(e.toString()));          }        }        if (event is HealthGetAllBloodOxygen) {          try {            DateTime now = DateTime.now();            DateTime yesterday = now.subtract(const Duration(days: 30));            final types = [              HealthDataType.BLOOD\_OXYGEN,            ];            emit(HealthLoading());            final bloodOxygen =                await HealthService().fetchData(now, yesterday, types);            emit(HealthAllBloodOxygenSuccess(bloodOxygen));          } catch (e) {            emit(HealthFailed(e.toString()));          }        }        if (event is HealthGetLandingPage) {          try {            DateTime now = DateTime.now();            DateTime yesterday = now.subtract(const Duration(days: 1));            final heartRateTypes = [              HealthDataType.HEART\_RATE,            ];            final bloodOxygenTypes = [              HealthDataType.BLOOD\_OXYGEN,            ];            emit(HealthLoading());            final heartRate =                await HealthService().fetchData(now, yesterday, heartRateTypes);            final bloodOxygen =                await HealthService().fetchData(now, yesterday, bloodOxygenTypes);            emit(HealthLandingPageSuccess(heartRate, bloodOxygen));          } catch (e) {            emit(HealthFailed(e.toString()));          }        }      });    }  } |

1. health\_event.dart

|  |
| --- |
| part of 'health\_bloc.dart';  abstract class HealthEvent extends Equatable {    const HealthEvent();    @override    List<Object> get props => [];  }  class HealthGetHeartRate extends HealthEvent {}  class HealthGetAllHeartRate extends HealthEvent {}  class HealthGetBloodOxygen extends HealthEvent {}  class HealthGetAllBloodOxygen extends HealthEvent {}  class HealthGetLandingPage extends HealthEvent {} |

1. health\_state.dart

|  |
| --- |
| part of 'health\_bloc.dart';  abstract class HealthState extends Equatable {    const HealthState();    @override    List<Object> get props => [];  }  class HealthInitial extends HealthState {}  class HealthLoading extends HealthState {}  class HealthFailed extends HealthState {    final String e;    const HealthFailed(this.e);    @override    List<Object> get props => [e];  }  class HealthHeartRateSuccess extends HealthState {    final List<HealthDataPoint> heartRate;    const HealthHeartRateSuccess(this.heartRate);    @override    List<Object> get props => [heartRate];  }  class HealthAllHeartRateSuccess extends HealthState {    final List<HealthDataPoint> allHeartRate;    const HealthAllHeartRateSuccess(this.allHeartRate);    @override    List<Object> get props => [allHeartRate];  }  class HealthBloodOxygenSuccess extends HealthState {    final List<HealthDataPoint> bloodOxygen;    const HealthBloodOxygenSuccess(this.bloodOxygen);    @override    List<Object> get props => [bloodOxygen];  }  class HealthAllBloodOxygenSuccess extends HealthState {    final List<HealthDataPoint> allBloodOxygen;    const HealthAllBloodOxygenSuccess(this.allBloodOxygen);    @override    List<Object> get props => [allBloodOxygen];  }  class HealthLandingPageSuccess extends HealthState {    final List<HealthDataPoint> heartRate;    final List<HealthDataPoint> bloodOxygen;    const HealthLandingPageSuccess(this.heartRate, this.bloodOxygen);    @override    List<Object> get props => [heartRate, bloodOxygen];  } |

1. heart\_disease\_bloc.dart

|  |
| --- |
| import 'package:bloc/bloc.dart';  import 'package:equatable/equatable.dart';  import 'package:healio/models/heart\_disease\_form\_model.dart';  import 'package:healio/services/heart\_disease\_service.dart';  part 'heart\_disease\_event.dart';  part 'heart\_disease\_state.dart';  class HeartDiseaseBloc extends Bloc<HeartDiseaseEvent, HeartDiseaseState> {    HeartDiseaseBloc() : super(HeartDiseaseInitial()) {      on<HeartDiseaseEvent>((event, emit) async {        if (event is HeartDiseasePrediction) {          try {            emit(HeartDiseaseLoading());            final res = await HeartDiseaseService().prediction(event.data);            emit(HeartDiseaseSuccess(res));          } catch (e) {            emit(HeartDiseaseFailed(e.toString()));          }        }      });    }  } |

1. heart\_disease\_event.dart

|  |
| --- |
| part of 'heart\_disease\_bloc.dart';  abstract class HeartDiseaseEvent extends Equatable {    const HeartDiseaseEvent();    @override    List<Object> get props => [];  }  class HeartDiseasePrediction extends HeartDiseaseEvent {    final HeartDiseaseFormModel data;    const HeartDiseasePrediction(this.data);    @override    List<Object> get props => [data];  } |

1. heart\_disease\_state.dart

|  |
| --- |
| part of 'heart\_disease\_bloc.dart';  abstract class HeartDiseaseState extends Equatable {    const HeartDiseaseState();    @override    List<Object> get props => [];  }  class HeartDiseaseInitial extends HeartDiseaseState {}  class HeartDiseaseLoading extends HeartDiseaseState {}  class HeartDiseaseFailed extends HeartDiseaseState {    final String e;    const HeartDiseaseFailed(this.e);    @override    List<Object> get props => [e];  }  class HeartDiseaseSuccess extends HeartDiseaseState {    final Map<String, dynamic> data;    const HeartDiseaseSuccess(this.data);    @override    List<Object> get props => [data];  } |

1. page\_cubit.dart

|  |
| --- |
| import 'package:bloc/bloc.dart';  class PageCubit extends Cubit<int> {    PageCubit() : super(0);    void setPage(int newPage) {      emit(newPage);    }  } |

1. user\_bloc.dart

|  |
| --- |
| import 'package:bloc/bloc.dart';  import 'package:equatable/equatable.dart';  import '../../models/user\_model.dart';  import '../../services/user\_service.dart';  part 'user\_event.dart';  part 'user\_state.dart';  class UserBloc extends Bloc<UserEvent, UserState> {    UserBloc() : super(UserInitial()) {      on<UserEvent>((event, emit) async {        if (event is UserUpdate) {          try {            emit(UserLoading());            UserModel user = await UserService().updateUser(                label: event.label, value: event.value, userId: event.userId);            emit(UserSuccess(user));          } catch (e) {            emit(UserFailed(e.toString()));          }        }      });    }  } |

1. user\_event.dart

|  |
| --- |
| part of 'user\_bloc.dart';  abstract class UserEvent extends Equatable {    const UserEvent();    @override    List<Object> get props => [];  }  class UserUpdate extends UserEvent {    final String label;    final String value;    final String userId;    const UserUpdate(this.label, this.value, this.userId);    @override    List<Object> get props => [label, value, userId];  } |

1. user\_state.dart

|  |
| --- |
| part of 'user\_bloc.dart';  abstract class UserState extends Equatable {    const UserState();    @override    List<Object> get props => [];  }  class UserInitial extends UserState {}  class UserLoading extends UserState {}  class UserFailed extends UserState {    final String e;    const UserFailed(this.e);    @override    List<Object> get props => [e];  }  class UserSuccess extends UserState {    final UserModel user;    const UserSuccess(this.user);    @override    List<Object> get props => [user];  } |