

Nama: Calista Anindita

NIM: H!D022049

Responsi Pemrograman Mobile

1. Registrasi

- Mendaftarkan akun pasien
- Kode
 - Kode model untuk registrasi

```
class Registrasi {  
  int? code;  
  bool? status;  
  String? data;  
  Registrasi({this.code, this.status, this.data});  
  factory Registrasi.fromJson(Map<String, dynamic> obj) {  
    return Registrasi(  
      code: obj['code'], status: obj['status'], data:  
obj['data']);  
  }  
}
```

- Kode untuk bloc registrasi

```
import 'dart:convert';  
import 'package:manajemenkesehatan/helpers/api.dart';  
import 'package:manajemenkesehatan/helpers/api_url.dart';  
import 'package:manajemenkesehatan/model/registrasi.dart';  
  
class RegistrasiBloc {  
  static Future<Registrasi> registrasi(  
    {String? nama, String? email, String? password})  
  async {  
    String apiUrl = ApiUrl.registrasi;  
    var body = {"nama": nama, "email": email, "password":  
password};  
    var response = await Api().post(apiUrl, body);  
    var jsonObj = json.decode(response.body);  
    return Registrasi.fromJson(jsonObj);  
  }  
}
```

- Kode untuk ui registrasi

```
○ import 'package:flutter/material.dart';
○ import
  'package:manajemenkesehatan/bloc/registrasi_bloc.dart';
○ import
  'package:manajemenkesehatan/widget/success_dialog.dart';
○ import
  'package:manajemenkesehatan/widget/warning_dialog.dart';
○
○ class RegistrasiPage extends StatefulWidget {
○   const RegistrasiPage({super.key});
○
○   @override
○   _RegistrasiPageState createState() =>
    _RegistrasiPageState();
○ }
○
○ class _RegistrasiPageState extends State<RegistrasiPage> {
○   final _formKey = GlobalKey<FormState>();
○   bool _isLoading = false;
○
○   final _namaTextboxController = TextEditingController();
○   final _emailTextboxController = TextEditingController();
○   final _passwordTextboxController =
    TextEditingController();
○   final _konfirmasiPasswordTextboxController =
    TextEditingController();
○
○   @override
○   Widget build(BuildContext context) {
○     return Scaffold(
○       appBar: AppBar(
○         title: const Text(
○           "Registrasi",
○           style: TextStyle(
○             color: Colors.white,
○             fontSize: 22,
○             fontWeight: FontWeight.bold,
○             fontFamily: 'SansSerif',
○           ),
○         ),
○         backgroundColor: const Color.fromARGB(255, 7, 230,
92),
○       ),
○       backgroundColor: const Color.fromARGB(255, 253, 240,
117),
○       body: SingleChildScrollView(
```

```

○      child: Padding(
○        padding: const EdgeInsets.all(16.0),
○        child: Form(
○          key: _formKey,
○          child: Column(
○            mainAxisAlignment: MainAxisAlignment.center,
○            children: [
○              const SizedBox(height: 40),
○              const Text(
○                'Daftarkan Akun Baru',
○                style: TextStyle(
○                  fontSize: 24,
○                  fontWeight: FontWeight.bold,
○                  color: Color.fromARGB(255, 7, 230, 92),
○                  fontFamily: 'SansSerif',
○                ),
○              ),
○              const SizedBox(height: 30),
○              _namaTextField(),
○              const SizedBox(height: 20),
○              _emailTextField(),
○              const SizedBox(height: 20),
○              _passwordTextField(),
○              const SizedBox(height: 20),
○              _passwordKonfirmasiTextField(),
○              const SizedBox(height: 40),
○              _buttonRegistrasi(),
○            ],
○          ),
○        ),
○      ),
○    ),
○  );
○ }

○ //Membuat Textbox Nama
○ Widget _namaTextField() {
○   return TextFormField(
○     decoration: InputDecoration(
○       labelText: "Nama",
○       labelStyle: const TextStyle(fontSize: 18),
○       border: OutlineInputBorder(
○         borderRadius: BorderRadius.circular(12),
○       ),
○       filled: true,
○       fillColor: const Color.fromARGB(255, 252, 255,
○180),

```

```

o         contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
o     ),
o     keyboardType: TextInputType.text,
o     controller: _namaTextboxController,
o     validator: (value) {
o         if (value!.length < 3) {
o             return "Nama harus diisi minimal 3 karakter";
o         }
o         return null;
o     },
o );
o }

o //Membuat Textbox email
Widget _emailTextField() {
o     return TextFormField(
o         decoration: InputDecoration(
o             labelText: "Email",
o             labelStyle: const TextStyle(fontSize: 18),
o             border: OutlineInputBorder(
o                 borderRadius: BorderRadius.circular(12),
o             ),
o             filled: true,
o             fillColor: const Color.fromARGB(255, 252, 255,
180),
o             contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
o         ),
o         keyboardType: TextInputType.emailAddress,
o         controller: _emailTextboxController,
o         validator: (value) {
o             if (value!.isEmpty) {
o                 return "Email harus diisi";
o             }
o             Pattern pattern =
o                 r'^((^[<>()[]\.,;:\s@"]+(\.[^<>()[]\.,;:\s
@\""]+)*)|(\\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-
9]{1,3}\.[0-9]{1,3}\)|((\[a-zA-Z\-0-9]+\.)+[a-zA-
Z]{2,}))$';
o             RegExp regex = RegExp(pattern.toString());
o             if (!regex.hasMatch(value)) {
o                 return "Email tidak valid";
o             }
o             return null;
o         },
o     );
o }

```

```

○
○ //Membuat Textbox password
○ Widget _passwordTextField() {
○   return TextFormField(
○     decoration: InputDecoration(
○       labelText: "Password",
○       labelStyle: const TextStyle(fontSize: 18),
○       border: OutlineInputBorder(
○         borderRadius: BorderRadius.circular(12),
○       ),
○       filled: true,
○       fillColor: const Color.fromARGB(255, 252, 255,
180),
○     contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
○   ),
○   keyboardType: TextInputType.text,
○   obscureText: true,
○   controller: _passwordTextboxController,
○   validator: (value) {
○     if (value!.length < 6) {
○       return "Password harus diisi minimal 6 karakter";
○     }
○     return null;
○   },
○ );
○ }
○
○ //Membuat Textbox Konfirmasi Password
○ Widget _passwordKonfirmasiTextField() {
○   return TextFormField(
○     decoration: InputDecoration(
○       labelText: "Konfirmasi Password",
○       labelStyle: const TextStyle(fontSize: 18),
○       border: OutlineInputBorder(
○         borderRadius: BorderRadius.circular(12),
○       ),
○       filled: true,
○       fillColor: const Color.fromARGB(255, 252, 255,
180),
○     contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
○   ),
○   keyboardType: TextInputType.text,
○   obscureText: true,
○   controller: _konfirmasiPasswordTextboxController,
○   validator: (value) {
○     if (value != _passwordTextboxController.text) {

```

```

o         return "Konfirmasi Password tidak sama";
o     }
o     return null;
o },
o );
o }
o
o //Membuat Tombol Registrasi
o Widget _buttonRegistrasi() {
o     return SizedBox(
o         width: double.infinity,
o         child: ElevatedButton(
o             style: ElevatedButton.styleFrom(
o                 padding: const EdgeInsets.symmetric(vertical:
16),
o                 backgroundColor: const Color.fromARGB(255, 7,
230, 92),
o                 shape: RoundedRectangleBorder(
o                     borderRadius: BorderRadius.circular(12),
o                 ),
o             ),
o             child: _isLoading
o                 ? const CircularProgressIndicator(color:
Colors.white)
o                 : const Text(
o                     "Registrasi",
o                     style: TextStyle(
o                         fontSize: 18,
o                         fontWeight: FontWeight.bold,
o                         color: Colors.white,
o                         fontFamily: 'SansSerif',
o                     ),
o                 ),
o             onPressed: () {
o                 var validate = _formKey.currentState!.validate();
o                 if (validate) {
o                     if (!_isLoading) _submit();
o                 }
o             },
o         ),
o     );
o }
o
o // Fungsi untuk submit registrasi
o void _submit() {
o     _formKey.currentState!.save();
o     setState(() {
o         _isLoading = true;

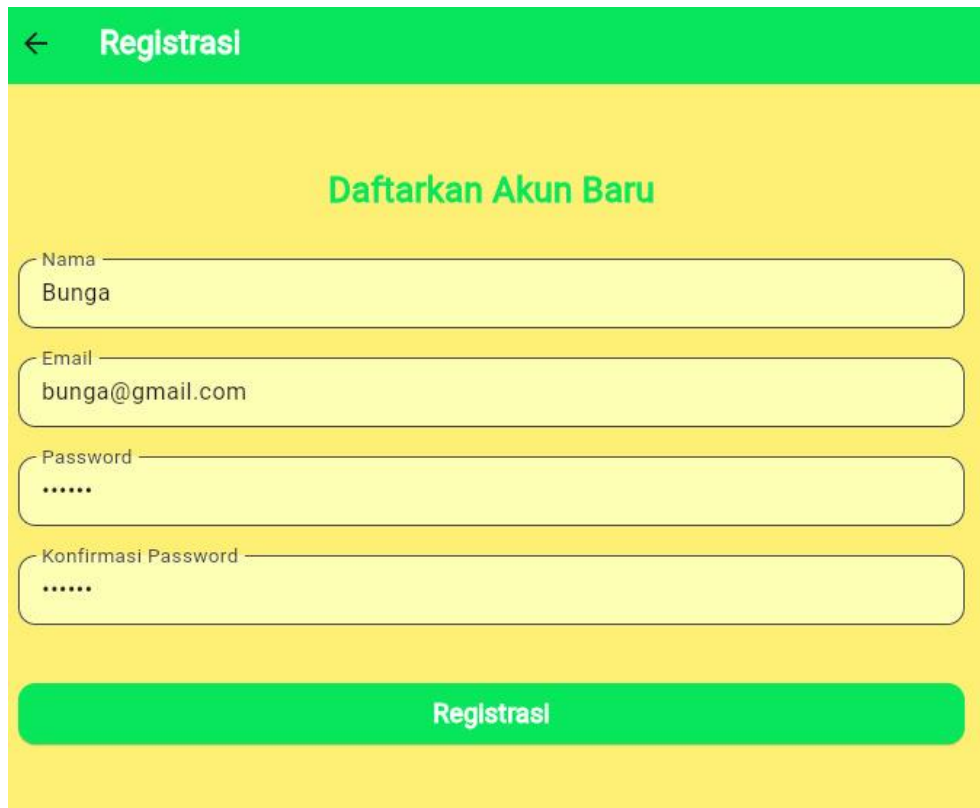
```

```

○   });
○   RegistrasiBloc.registrasi(
○       nama: _namaTextboxController.text,
○       email: _emailTextboxController.text,
○       password: _passwordTextboxController.text)
○   .then((value) {
○       showDialog(
○           context: context,
○           barrierDismissible: false,
○           builder: (BuildContext context) => SuccessDialog(
○               description: "Registrasi berhasil, silahkan
login",
○               okClick: () {
○                   Navigator.pop(context);
○               },
○           ));
○   }, onError: (error) {
○       showDialog(
○           context: context,
○           barrierDismissible: false,
○           builder: (BuildContext context) => const
WarningDialog(
○               description: "Registrasi gagal, silahkan
coba lagi",
○           ));
○   });
○   setState(() {
○       _isLoading = false;
○   });
○   }
○   }
○

```

- Screenshot Tampilan



A screenshot of a mobile application's registration screen. The header is a solid green bar with a white back arrow and the word 'Registrasi' in white. The main background is a solid yellow color. In the center, the text 'Daftarkan Akun Baru' is written in green. Below this, there are four white input fields with green borders. The first field is labeled 'Nama' and contains the text 'Bunga'. The second field is labeled 'Email' and contains 'bunga@gmail.com'. The third field is labeled 'Password' and contains six dots. The fourth field is labeled 'Konfirmasi Password' and also contains six dots. At the bottom of the form is a wide, rounded green button with the white text 'Registrasi'.

← Registrasi

Daftarkan Akun Baru

Nama
Bunga

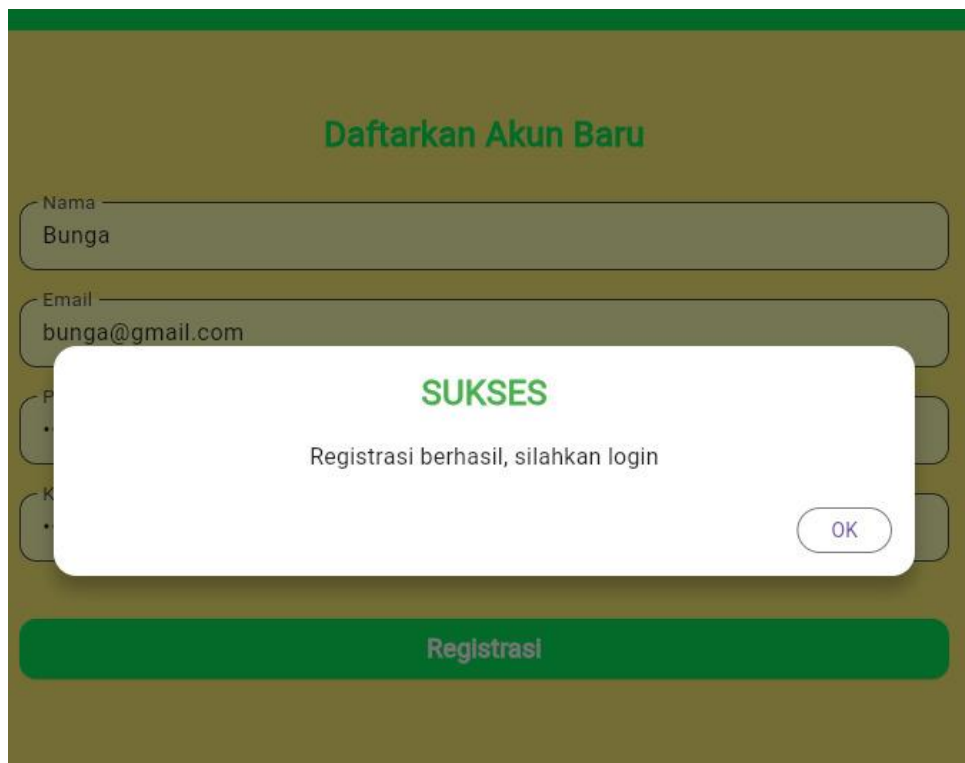
Email
bunga@gmail.com

Password
.....

Konfirmasi Password
.....

Registrasi

Pop Up Berhasil Registrasi



A screenshot of the same registration form as above, but with a success pop-up overlay. The background is dimmed. The pop-up is a white rounded rectangle with a green border. It contains the word 'SUKSES' in large green letters, followed by the text 'Registrasi berhasil, silahkan login' in black. In the bottom right corner of the pop-up is a small green button with the white text 'OK'.

Daftarkan Akun Baru

Nama
Bunga

Email
bunga@gmail.com

P
•

K
•

SUKSES

Registrasi berhasil, silahkan login

OK

Registrasi

2. Login

- Masuk ke halaman dengan akun pasien
- Kode

- Kode model untuk login

```
○ class Login {  
○   int? code;  
○   bool? status;  
○   String? token;  
○   int? userID;  
○   String? userEmail;  
○   Login({this.code, this.status, this.token, this.userID,  
this.userEmail});  
○   factory Login.fromJson(Map<String, dynamic> obj) {  
○     if (obj['code'] == 200) {  
○       return Login(  
○         code: obj['code'],  
○         status: obj['status'],  
○         token: obj['data']['token'],  
○         userID: obj['data']['user']['id'],  
○         userEmail: obj['data']['user']['email']);  
○     } else {  
○       return Login(  
○         code: obj['code'],  
○         status: obj['status'],  
○       );  
○     }  
○   }  
○ }  
○ }
```

- Kode untuk bloc login

```
○ import 'dart:convert';  
○ import 'package:manajemenkesehatan/helpers/api.dart';  
○ import 'package:manajemenkesehatan/helpers/api_url.dart';  
○ import 'package:manajemenkesehatan/model/login.dart';  
○  
○ class LoginBloc {  
○   static Future<Login> login({String? email, String?  
password}) async {  
○     String apiUrl = ApiUrl.login;  
○     var body = {"email": email, "password": password};  
○     var response = await Api().post(apiUrl, body);  
○     var jsonObj = json.decode(response.body);  
○     return Login.fromJson(jsonObj);  
○   }  
○ }
```

- Kode untuk ui login

```
○ import 'package:flutter/material.dart';  
○ import 'package:manajemenkesehatan/bloc/login_bloc.dart';
```

```

○ import 'package:manajemenkesehatan/helpers/user_info.dart';
○ import
  'package:manajemenkesehatan/ui/registrasi_page.dart';
○ import
  'package:manajemenkesehatan/widget/warning_dialog.dart';
○ import
  'package:manajemenkesehatan/widget/success_dialog.dart';
○
○ class LoginPage extends StatefulWidget {
○   const LoginPage({super.key});
○
○   @override
○   _LoginPageState createState() => _LoginPageState();
○ }
○
○ class _LoginPageState extends State<LoginPage> {
○   final _formKey = GlobalKey<FormState>();
○   bool _isLoading = false;
○   final _emailTextboxController = TextEditingController();
○   final _passwordTextboxController =
    TextEditingController();
○
○   @override
○   Widget build(BuildContext context) {
○     return Scaffold(
○       appBar: AppBar(
○         title: const Text(
○           'Login',
○           style: TextStyle(
○             color: Colors.white,
○             fontSize: 22,
○             fontWeight: FontWeight.bold,
○             fontFamily: 'SansSerif',
○           ),
○         ),
○       backgroundColor: const Color.fromARGB(255, 7, 230,
92),
○     ),
○     backgroundColor: const Color.fromARGB(255, 253, 240,
117),
○     body: SingleChildScrollView(
○       child: Padding(
○         padding: const EdgeInsets.all(16.0),
○         child: Form(
○           key: _formKey,
○           child: Column(
○             children: [
○               const SizedBox(height: 50),

```

```

o         const Text(
o             'Silakan Masukkan Data Anda!',
o             style: TextStyle(
o                 fontSize: 24,
o                 fontWeight: FontWeight.bold,
o                 color: Color.fromARGB(255, 7, 230, 92),
o                 fontFamily: 'SansSerif',
o             ),
o         ),
o         const SizedBox(height: 30),
o         _emailTextField(),
o         const SizedBox(height: 20),
o         _passwordTextField(),
o         const SizedBox(height: 40),
o         _buttonLogin(),
o         const SizedBox(
o             height: 30,
o         ),
o         _menuRegistrasi(),
o     ],
o ),
o ),
o ),
o ),
o );
o }

o //Membuat Textbox email
o Widget _emailTextField() {
o     return TextFormField(
o         decoration: InputDecoration(
o             labelText: "Email",
o             labelStyle: const TextStyle(fontSize: 18),
o             border: OutlineInputBorder(
o                 borderRadius: BorderRadius.circular(12),
o             ),
o             filled: true,
o             fillColor: const Color.fromARGB(255, 252, 255,
180),
o             contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
o         ),
o         keyboardType: TextInputType.emailAddress,
o         controller: _emailTextboxController,
o         validator: (value) {
o             //validasi harus diisi
o             if (value!.isEmpty) {
o                 return 'Email harus diisi';

```

```

    }
    return null;
  },
);
}

//Membuat Textbox password
Widget _passwordTextField() {
  return TextFormField(
    decoration: InputDecoration(
      labelText: "Password",
      labelStyle: const TextStyle(fontSize: 18),
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(12),
      ),
      filled: true,
      fillColor: const Color.fromARGB(255, 252, 255,
180),
      contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 12),
    ),
    obscureText: true,
    controller: _passwordTextboxController,
    validator: (value) {
      //validasi password harus diisi
      if (value!.isEmpty) {
        return "Password harus diisi";
      }
      return null;
    },
  );
}

//Membuat Tombol Login
Widget _buttonLogin() {
  return SizedBox(
    width: double.infinity,
    child: ElevatedButton(
      style: ElevatedButton.styleFrom(
        padding: const EdgeInsets.symmetric(vertical:
16),
        backgroundColor: const Color.fromARGB(255, 7,
230, 92),
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(12),
        ),
      ),
      child: _isLoading

```

```

○      ? const CircularProgressIndicator(color:
Colors.white)
○      : const Text(
○          "Login",
○          style: TextStyle(
○              fontSize: 18,
○              fontWeight: FontWeight.bold,
○              color: Colors.white,
○              fontFamily: 'SansSerif',
○          ),
○      ),
○      onPressed: () {
○          var validate = _formKey.currentState!.validate();
○          if (validate) {
○              if (!_isLoading) _submit();
○          }
○      },
○  ),
○ );
○ }

○
○ void _submit() {
○     _formKey.currentState!.save();
○     setState(() {
○         _isLoading = true;
○     });
○     LoginBloc.login(
○         email: _emailTextboxController.text,
○         password: _passwordTextboxController.text)
○         .then((value) async {
○             if (value.code == 200) {
○                 await UserInfo().setToken(value.token.toString());
○                 await
○ UserInfo().setUserID(int.parse(value.userID.toString()));
○                 showDialog(
○                     context: context,
○                     barrierDismissible: false,
○                     builder: (BuildContext context) => SuccessDialog(
○                         description: "Anda Berhasil Login!",
○                         onClick: () {
○                             Navigator.pop(context);
○                         },
○                     ),
○                 ));
○             } else {
○                 showDialog(
○                     context: context,
○                     barrierDismissible: false,

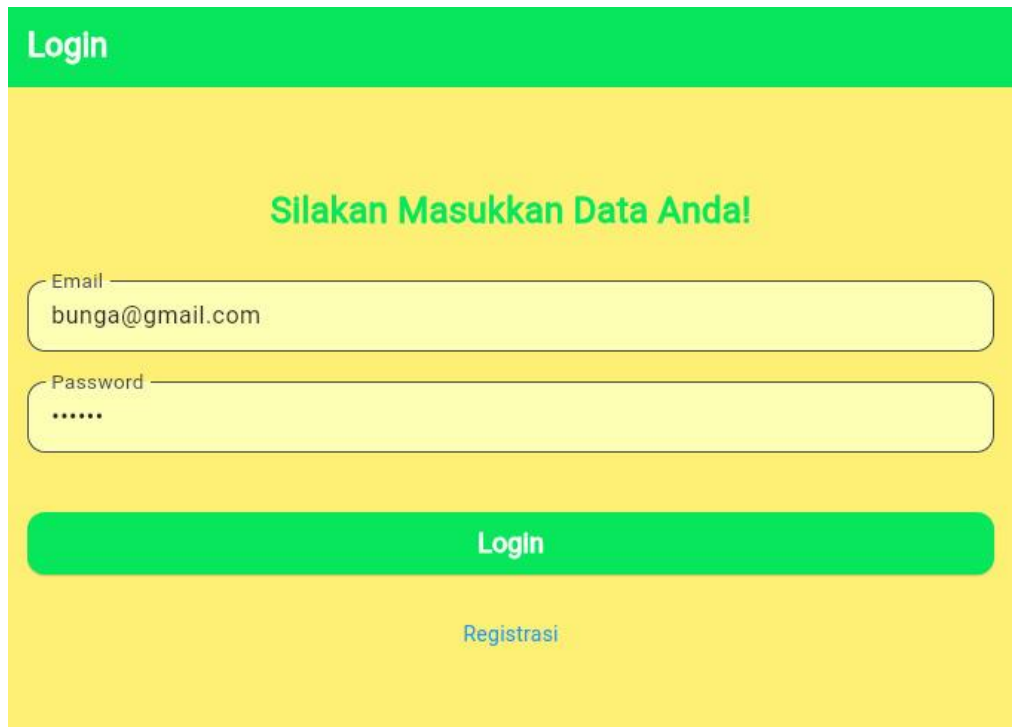
```

```

○         builder: (BuildContext context) => const
WarningDialog(
○             description: "Login gagal, silahkan coba
lagi",
○             ));
○     }
○ }, onError: (error) {
○     print(error);
○     showDialog(
○         context: context,
○         barrierDismissible: false,
○         builder: (BuildContext context) => const
WarningDialog(
○             description: "Login gagal, silahkan coba
lagi",
○             ));
○     });
○     setState(() {
○         _isLoading = false;
○     });
○ }
○
○ // Membuat menu untuk membuka halaman registrasi
Widget _menuRegistrasi() {
○     return Center(
○         child: InkWell(
○             child: const Text(
○                 "Registrasi",
○                 style: TextStyle(color: Colors.blue),
○             ),
○             onTap: () {
○                 Navigator.push(context,
○                     MaterialPageRoute(builder: (context) => const
RegistrasiPage()));
○             },
○         ),
○     );
○ }
○ }

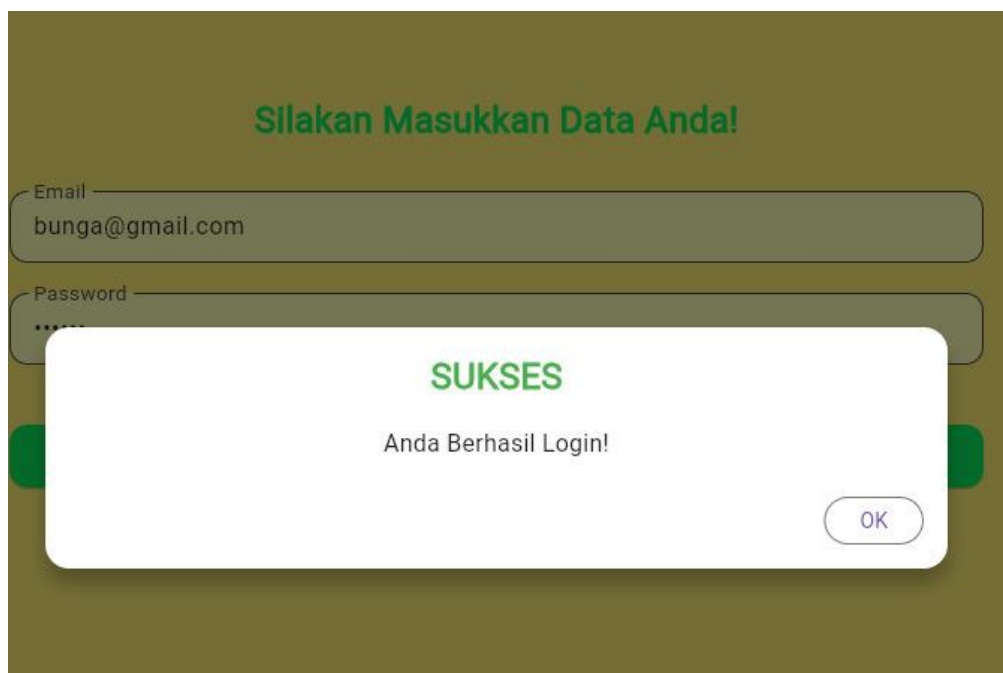
```

- Screenshot Tampilan



The image shows a login form with a green header labeled 'Login'. The main background is yellow. The text 'Silakan Masukkan Data Anda!' is centered. There are two input fields: 'Email' with the value 'bunga@gmail.com' and 'Password' with masked characters '.....'. Below the fields is a green 'Login' button and a blue 'Registrasi' link.

Pop Up Berhasil Login



The image shows the same login form as above, but with a white success pop-up overlay. The pop-up has a green title 'SUKSES' and the text 'Anda Berhasil Login!'. There is an 'OK' button in the bottom right corner of the pop-up.

3. Rekam Medis Page

- Menampilkan halaman yang berisi list rekam medis pasien
- Kode
 - Kode Model Rekam Medis Page

```
○ class RekamMedisPasien {  
○     String? id;
```

```

○ String? patientName;
○ String? symptom;
○ int? severity;
○ DateTime? createdAt;
○ DateTime? updatedAt;
○
○ RekamMedisPasien({
○   this.id,
○   this.patientName,
○   this.symptom,
○   this.severity,
○   this.createdAt,
○   this.updatedAt,
○ });
○
○ factory RekamMedisPasien.fromJson(Map<String, dynamic>
obj) {
○   return RekamMedisPasien(
○     id: obj['id']?.toString(),
○     patientName: obj['patient_name'],
○     symptom: obj['symptom'],
○     severity: obj['severity'] is int ? obj['severity'] :
int.parse(obj['severity']),
○     createdAt: obj['created_at'] != null ?
DateTime.parse(obj['created_at']) : null,
○     updatedAt: obj['updated_at'] != null ?
DateTime.parse(obj['updated_at']) : null,
○   );
○ }
○
○ Map<String, dynamic> toJson() {
○   return {
○     'id': id,
○     'patient_name': patientName,
○     'symptom': symptom,
○     'severity': severity.toString(),
○   };
○ }
○ }
○

```

○ Kode Bloc Rekam Medis Page

```

○ import 'dart:convert';
○ import 'package:manajemenkesehatan/helpers/api.dart';
○ import 'package:manajemenkesehatan/helpers/api_url.dart';
○ import
○   'package:manajemenkesehatan/model/rekam_medis_pasien.dart';
○

```



```

○ class RekamMedisPasienBloc {
○
○     static Future<List<RekamMedisPasien>>
getRekamMedisPasien() async {
○         String apiUrl = ApiUrl.listRekamMedisPasien;
○         var response = await Api().get(apiUrl);
○
○         if (response.statusCode == 200) {
○             var jsonObj = json.decode(response.body);
○             List<dynamic> listRekamMedisPasien = (jsonObj as
Map<String, dynamic>)['data'];
○             List<RekamMedisPasien> rekamMedisPasienList = [];
○
○             for (var item in listRekamMedisPasien) {
○                 rekamMedisPasienList.add(RekamMedisPasien.fromJson(
item));
○             }
○             return rekamMedisPasienList;
○         } else {
○             throw Exception('Failed to load rekam medis pasien');
○         }
○     }
○
○     static Future<bool> addRekamMedisPasien({required
RekamMedisPasien rekamMedisPasien}) async {
○         String apiUrl = ApiUrl.createRekamMedisPasien;
○         var body = {
○             "patient_name": rekamMedisPasien.patientName,
○             "symptom": rekamMedisPasien.symptom,
○             "severity": rekamMedisPasien.severity.toString()
○         };
○
○         var response = await Api().post(apiUrl, body);
○
○         if (response.statusCode == 200) {
○             var jsonObj = json.decode(response.body);
○             return jsonObj['status'];
○         } else {
○             throw Exception('Failed to add rekam medis pasien');
○         }
○     }
○
○     static Future<bool> updateRekamMedisPasien({required
RekamMedisPasien rekamMedisPasien}) async {
○         String apiUrl =
ApiUrl.updateRekamMedisPasien(int.parse(rekamMedisPasien.id
!));

```

```

○   var body = {
○       "patient_name": rekamMedisPasien.patientName,
○       "symptom": rekamMedisPasien.symptom,
○       "severity": rekamMedisPasien.severity.toString()
○   };
○
○   var response = await Api().put(apiUrl,
○   jsonEncode(body));
○
○
○   if (response.statusCode == 200) {
○       var jsonObj = json.decode(response.body);
○       return jsonObj['status'];
○   } else {
○       throw Exception('Failed to update rekam medis
○   pasien');
○   }
○   }
○
○
○   static Future<bool> deleteRekamMedisPasien({int? id})
○   async {
○       String apiUrl = ApiUrl.deleteRekamMedisPasien(id!);
○       var response = await Api().delete(apiUrl);
○       var jsonObj = json.decode(response.body);
○       return (jsonObj as Map<String, dynamic>)['data'];
○   }
○   }
○
○

```

○ Kode UI Rekam Medis Page

```

○   import 'package:flutter/material.dart';
○   import 'package:manajemenkesehatan/model/nutrisi.dart';
○   import 'package:manajemenkesehatan/ui/login_page.dart';
○   import 'package:manajemenkesehatan/bloc/logout_bloc.dart';
○   import 'package:manajemenkesehatan/ui/nutrisi_form.dart';
○   import 'package:manajemenkesehatan/bloc/NutrisiBloc.dart';
○   import 'package:manajemenkesehatan/ui/nutrisi_detail.dart';
○
○   class NutrisiPage extends StatefulWidget {
○       const NutrisiPage({Key? key}) : super(key: key);
○
○       @override
○       _NutrisiPageState createState() => _NutrisiPageState();
○   }
○
○   class _NutrisiPageState extends State<NutrisiPage> {
○       @override

```

```

o   Widget build(BuildContext context) {
o     return Scaffold(
o       appBar: AppBar(
o         title: const Text(
o           'Data Nutrisi',
o         style: TextStyle(
o           fontFamily: 'SansSerif',
o           fontSize: 22,
o           fontWeight: FontWeight.bold,
o           color: Colors.white,
o         ),
o       ),
o     ),
o     backgroundColor: const Color.fromARGB(255, 7, 230,
92),
o     actions: [
o       Padding(
o         padding: const EdgeInsets.only(right: 20.0),
o         child: GestureDetector(
o           child: const Icon(Icons.add, size: 26.0,
color: Colors.white),
o           onTap: () {
o             Navigator.push(
o               context,
o               MaterialPageRoute(builder: (context) =>
const NutrisiForm()),
o             );
o           },
o         ),
o       ),
o     ],
o   ),
o   drawer: Drawer(
o     child: ListView(
o       children: [
o         ListTile(
o           title: const Text(
o             'Logout',
o           style: TextStyle(
o             fontSize: 18,
o             fontFamily: 'SansSerif',
o           ),
o         ),
o         trailing: const Icon(Icons.logout, color:
Color.fromARGB(255, 7, 230, 92)),
o         onTap: () async {
o           await LogoutBloc.logout().then((value) {
o             Navigator.of(context).pushAndRemoveUntil(

```

```

o         MaterialPageRoute(builder: (context) =>
o const LoginPage()),
o         (route) => false,
o         );
o     });
o     },
o     ),
o     ],
o     ),
o     ),
o     backgroundColor: const Color.fromARGB(255, 253, 240,
o 117),
o     body: Column(
o     children: [
o         const SizedBox(height: 40),
o         Expanded(
o         child: FutureBuilder<List<Nutrisi>>(
o         future: NutrisiBloc.getNutrisi(),
o         builder: (context, snapshot) {
o             if (snapshot.hasError) {
o                 print(snapshot.error);
o             }
o             return snapshot.hasData
o                 ? ListNutrisi(list: snapshot.data)
o                 : const Center(
o                     child: CircularProgressIndicator(),
o                     );
o             },
o             ),
o             ),
o             ],
o             ),
o             );
o         }
o     }

o class ListNutrisi extends StatelessWidget {
o     final List<Nutrisi>? list;

o     const ListNutrisi({Key? key, this.list}) : super(key:
o key);

o     @override
o     Widget build(BuildContext context) {
o         return ListView.builder(
o             itemCount: list?.length ?? 0,
o             itemBuilder: (context, i) {

```

```

○         return ItemNutrisi(
○             nutrisi: list![i],
○         );
○     },
○ );
○ }
○ }

○ class ItemNutrisi extends StatelessWidget {
○     final Nutrisi nutrisi;

○     const ItemNutrisi({Key? key, required this.nutrisi}) :
super(key: key);

○     @override
○     Widget build(BuildContext context) {
○         return GestureDetector(
○             onTap: () {
○                 Navigator.push(
○                     context,
○                     MaterialPageRoute(
○                         builder: (context) => NutrisiDetail(nutrisi:
nutrisi),
○                     ),
○                 );
○             },
○             child: Card(
○                 color: const Color.fromARGB(255, 252, 255, 180),
○                 shape: RoundedRectangleBorder(
○                     borderRadius: BorderRadius.circular(12),
○                 ),
○                 elevation: 4,
○                 margin: const EdgeInsets.symmetric(horizontal: 16,
vertical: 8),
○                 child: ListTile(
○                     contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 10),
○                     title: Text(
○                         nutrisi.foodItem,
○                         style: const TextStyle(
○                             fontSize: 18,
○                             fontWeight: FontWeight.bold,
○                             color: Colors.black87,
○                         ),
○                     ),
○                     subtitle: Text(
○                         'Kalori: ${nutrisi.calories}, Lemak:
${nutrisi.fatContent} gram',

```

```

○         style: const TextStyle(fontSize: 16, color:
Colors.grey),
○     ),
○     trailing: const Icon(
○         Icons.arrow_forward_ios,
○         color: Color.fromARGB(255, 7, 230, 92),
○         size: 20,
○     ),
○ ),
○ ),
○ );
○ }
○ }
○

```

○ Kode Form Rekam Medis Page

```

○ import 'package:flutter/material.dart';
○ import
○     'package:manajemenkesehatan/bloc/RekamMedisPasienBloc.dart'
○     ;
○ import
○     'package:manajemenkesehatan/model/rekam_medis_pasien.dart';
○ import 'package:manajemenkesehatan/ui/pasien_page.dart';
○ import
○     'package:manajemenkesehatan/widget/warning_dialog.dart';
○
○ class RekamMedisPasienForm extends StatefulWidget {
○     final RekamMedisPasien? rekamMedisPasien;
○     const RekamMedisPasienForm({Key? key,
this.rekamMedisPasien}) : super(key: key);
○
○     @override
○     _RekamMedisPasienFormState createState() =>
_RekamMedisPasienFormState();
○ }
○
○ class _RekamMedisPasienFormState extends
State<RekamMedisPasienForm> {
○     final _formKey = GlobalKey<FormState>();
○     bool _isLoading = false;
○     String judul = "TAMBAH REKAM MEDIS PASIEN";
○     String tombolSubmit = "SIMPAN";
○
○     final _patientNameTextboxController =
TextEditingController();
○     final _symptomTextboxController =
TextEditingController();

```

```

○   final _severityTextboxController =
      TextEditingController();
○
○   @override
○   void initState() {
○       super.initState();
○       isUpdate();
○   }
○
○   void isUpdate() {
○       if (widget.rekamMedisPasien != null) {
○           setState(() {
○               judul = "UBAH REKAM MEDIS PASIEN";
○               tombolSubmit = "UBAH";
○               _patientNameTextboxController.text =
widget.rekamMedisPasien!.patientName!;
○               _symptomTextboxController.text =
widget.rekamMedisPasien!.symptom!;
○               _severityTextboxController.text =
widget.rekamMedisPasien!.severity.toString();
○           });
○       }
○   }
○
○   @override
○   Widget build(BuildContext context) {
○       return Scaffold(
○           appBar: AppBar(
○               title: Text(judul),
○               backgroundColor: Color.fromARGB(255, 7, 230, 92),
○           ),
○           backgroundColor: const Color.fromARGB(255, 253, 240,
117),
○           body: SingleChildScrollView(
○               child: Padding(
○                   padding: const EdgeInsets.all(16.0),
○                   child: Form(
○                       key: _formKey,
○                       child: Column(
○                           children: [
○                               const SizedBox(height: 50),
○                               _patientNameTextField(),
○                               const SizedBox(height: 16),
○                               _symptomTextField(),
○                               const SizedBox(height: 16),
○                               _severityTextField(),
○                               const SizedBox(height: 30),
○                               _buttonSubmit(),

```

```

○         ],
○     ),
○ ),
○ ),
○ ),
○ );
○ }
○
○ Widget _patientNameTextField() {
○     return TextFormField(
○         decoration: InputDecoration(
○             labelText: "Nama Pasien",
○             border: const OutlineInputBorder(),
○             labelStyle: TextStyle(color: Colors.black),
○         ),
○         controller: _patientNameTextboxController,
○         validator: (value) {
○             if (value!.isEmpty) {
○                 return "Nama Pasien harus diisi";
○             }
○             return null;
○         },
○     );
○ }
○
○ Widget _symptomTextField() {
○     return TextFormField(
○         decoration: InputDecoration(
○             labelText: "Gejala",
○             border: const OutlineInputBorder(),
○             labelStyle: TextStyle(color: Colors.black),
○         ),
○         controller: _symptomTextboxController,
○         validator: (value) {
○             if (value!.isEmpty) {
○                 return "Gejala harus diisi";
○             }
○             return null;
○         },
○     );
○ }
○
○ Widget _severityTextField() {
○     return TextFormField(
○         decoration: InputDecoration(
○             labelText: "Tingkat Keparahan (1-5)",
○             border: const OutlineInputBorder(),
○             labelStyle: TextStyle(color: Colors.black),

```



```

○      ),
○      keyboardType: TextInputType.number,
○      controller: _severityTextboxController,
○      validator: (value) {
○          if (value!.isEmpty) {
○              return "Tingkat Keparahan harus diisi";
○          }
○          if (int.tryParse(value) == null || int.parse(value)
< 1 || int.parse(value) > 5) {
○              return "Tingkat Keparahan harus antara 1 dan 5";
○          }
○          return null;
○      },
○  );
○  }
○
○  Widget _buttonSubmit() {
○      return ElevatedButton(
○          style: ElevatedButton.styleFrom(
○              padding: const EdgeInsets.symmetric(vertical: 16,
horizontal: 40),
○              shape: RoundedRectangleBorder(
○                  borderRadius: BorderRadius.circular(20),
○              ),
○              backgroundColor: Color.fromARGB(255, 7, 230, 92),
// Warna biru untuk tombol
○              foregroundColor: Colors.white, // Teks tombol
berwarna putih
○          ),
○          child: _isLoading
○              ? const CircularProgressIndicator(color:
Colors.white)
○              : Text(
○                  tombolSubmit,
○                  style: const TextStyle(fontSize: 18),
○              ),
○          onPressed: () {
○              if (_formKey.currentState!.validate()) {
○                  if (!_isLoading) {
○                      if (widget.rekamMedisPasien != null) {
○                          ubah();
○                      } else {
○                          simpan();
○                      }
○                  }
○              }
○          },
○      ),
○  );

```

```

○   }
○
○   void simpan() {
○       setState(() {
○           _isLoading = true;
○       });
○
○       RekamMedisPasien newRekamMedisPasien =
RekamMedisPasien(
○           id: null,
○           patientName: _patientNameTextboxController.text,
○           symptom: _symptomTextboxController.text,
○           severity: int.parse(_severityTextboxController.text),
○       );
○
○       RekamMedisPasienBloc.addRekamMedisPasien(rekamMedisPasi
en: newRekamMedisPasien).then(
○           (value) {
○               Navigator.of(context).pushReplacement(
○                   MaterialPageRoute(
○                       builder: (BuildContext context) => const
RekamMedisPasienPage(),
○                   ),
○               );
○           },
○           onError: (error) {
○               showDialog(
○                   context: context,
○                   builder: (BuildContext context) => const
WarningDialog(
○                       description: "Simpan gagal, silahkan coba
lagi",
○                   ),
○               );
○           },
○       ).whenComplete(() {
○           setState(() {
○               _isLoading = false;
○           });
○       });
○   }
○
○   void ubah() {
○       setState(() {
○           _isLoading = true;
○       });
○   }

```

```

○      RekamMedisPasien updatedRekamMedisPasien =
RekamMedisPasien(
○      id: widget.rekamMedisPasien!.id,
○      patientName: _patientNameTextboxController.text,
○      symptom: _symptomTextboxController.text,
○      severity: int.parse(_severityTextboxController.text),
○  );
○
○      RekamMedisPasienBloc.updateRekamMedisPasien(rekamMedisP
asien: updatedRekamMedisPasien).then(
○      (value) {
○          Navigator.of(context).pushReplacement(
○              MaterialPageRoute(
○                  builder: (BuildContext context) => const
RekamMedisPasienPage(),
○              ),
○          );
○      },
○      onError: (error) {
○          showDialog(
○              context: context,
○              builder: (BuildContext context) => const
WarningDialog(
○                  description: "Ubah data gagal, silahkan coba
lagi",
○              ),
○          );
○      },
○      ).whenComplete(() {
○          setState(() {
○              _isLoading = false;
○          });
○      });
○  }
○ }
○

```

○ Kode Detail Rekam Medis Page

```

○ import 'package:flutter/material.dart';
○ import
  'package:manajemenkesehatan/model/rekam_medis_pasien.dart';
○ import 'package:manajemenkesehatan/ui/pasien_page.dart';
○ import 'package:manajemenkesehatan/ui/pasien_form.dart';
○ import
  'package:manajemenkesehatan/bloc/RekamMedisPasienBloc.dart'
  ;
○ import
  'package:manajemenkesehatan/widget/warning_dialog.dart';

```

```

○
○ class RekamMedisPasienDetail extends StatefulWidget {
○   final RekamMedisPasien? rekamMedisPasien;
○
○   const RekamMedisPasienDetail({super.key,
○     this.rekamMedisPasien});
○
○   @override
○   _RekamMedisPasienDetailState createState() =>
○     _RekamMedisPasienDetailState();
○ }
○
○ class _RekamMedisPasienDetailState extends
○   State<RekamMedisPasienDetail> {
○   final Color primaryColor = const Color.fromARGB(255, 7,
○     230, 92);
○   final Color backgroundColor = const Color.fromARGB(255,
○     253, 240, 117);
○   final Color editButtonColor = Colors.blueAccent;
○   final Color deleteButtonColor = Colors.redAccent;
○
○   @override
○   Widget build(BuildContext context) {
○     return Scaffold(
○       appBar: AppBar(
○         title: const Text(
○           'Detail Rekam Medis Pasien',
○           style: TextStyle(
○             fontFamily: 'SansSerif',
○             fontSize: 22,
○             fontWeight: FontWeight.bold,
○             color: Colors.white,
○           ),
○         ),
○         backgroundColor: primaryColor,
○       ),
○       backgroundColor: backgroundColor,
○       body: Padding(
○         padding: const EdgeInsets.all(16.0),
○         child: Column(
○           crossAxisAlignment: CrossAxisAlignment.start,
○           children: [
○             const SizedBox(height: 40),
○             Align(
○               alignment: Alignment.center,
○               child: SizedBox(
○                 width: MediaQuery.of(context).size.width *
○                 0.85,

```

```

○         child: Card(
○             shape: RoundedRectangleBorder(
○                 borderRadius:
BorderRadius.circular(15),
○             ),
○             elevation: 8,
○             shadowColor:
primaryColor.withOpacity(0.5),
○             child: Padding(
○                 padding: const EdgeInsets.all(20),
○                 child: Column(
○                     mainAxisAlignment: MainAxisAlignment.min,
○                     children: [
○                         _buildDetailRow(Icons.badge, "ID
Rekam Medis: ${widget.rekamMedisPasien?.id}"),
○                         _buildDetailRow(Icons.person, "Nama
Pasien: ${widget.rekamMedisPasien?.patientName}"),
○                         _buildDetailRow(Icons.sick,
"Gejala: ${widget.rekamMedisPasien?.symptom}"),
○                         _buildDetailRow(Icons.warning,
"Tingkat Keparahan: ${widget.rekamMedisPasien?.severity}"),
○                     ],
○                 ),
○             ),
○         ),
○         const SizedBox(height: 30),
○         _tombo1HapusEdit(),
○     ],
○ ),
○ ),
○ );
○ }

○ Widget _buildDetailRow(IconData icon, String text) {
○     return Row(
○         children: [
○             Icon(icon, color: primaryColor),
○             const SizedBox(width: 10),
○             Expanded(
○                 child: Text(
○                     text,
○                     style: const TextStyle(
○                         fontFamily: 'SansSerif',
○                         fontSize: 18,
○                         color: Colors.black87,
○                     ),
○                 ),
○             ),
○         ],
○     );
○ }

```

```

o         ),
o     ),
o ],
o );
o }
o
o Widget _tombolHapusEdit() {
o     return Row(
o         mainAxisAlignment: MainAxisAlignment.center,
o         children: [
o             OutlinedButton(
o                 style: OutlinedButton.styleFrom(
o                     foregroundColor: Colors.white,
o                     backgroundColor: editButtonColor,
o                     side: const BorderSide(color:
Colors.blueAccent, width: 1.5),
o                 ),
o                 child: const Text("EDIT"),
o                 onPressed: () {
o                     Navigator.push(
o                         context,
o                         MaterialPageRoute(
o                             builder: (context) => RekamMedisPasienForm(
o                                 rekamMedisPasien:
widget.rekamMedisPasien!,
o                             ),
o                         ),
o                     );
o                 },
o             ),
o             const SizedBox(width: 20),
o             OutlinedButton(
o                 style: OutlinedButton.styleFrom(
o                     foregroundColor: Colors.white,
o                     backgroundColor: deleteButtonColor,
o                     side: const BorderSide(color: Colors.redAccent,
width: 1.5),
o                 ),
o                 child: const Text("DELETE"),
o                 onPressed: () => confirmHapus(),
o             ),
o         ],
o     );
o }
o
o void confirmHapus() {
o     AlertDialog alertDialog = AlertDialog(

```

```

○      content: const Text("Yakin ingin menghapus data
○      ini?"),
○      actions: [
○          // Tombol Ya
○          OutlinedButton(
○              style: OutlinedButton.styleFrom(
○                  foregroundColor: Colors.white,
○                  backgroundColor: Colors.redAccent,
○                  side: const BorderSide(color: Colors.redAccent,
○width: 1.5),
○              ),
○              child: const Text("Ya"),
○              onPressed: () {
○                  RekamMedisPasienBloc.deleteRekamMedisPasien(id:
○int.parse(widget.rekamMedisPasien!.id!)).then(
○                  (value) => {
○                      Navigator.of(context).pushReplacement(
○                          MaterialPageRoute(
○                              builder: (context) => const
○RekamMedisPasienPage(),
○                          ),
○                      )
○                  },
○                  onError: (error) {
○                      showDialog(
○                          context: context,
○                          builder: (BuildContext context) => const
○WarningDialog(
○                              description: "Hapus gagal, silahkan
○coba lagi",
○                              ),
○                      );
○                  },
○                  );
○              },
○          ),
○          // Tombol Batal
○          OutlinedButton(
○              style: OutlinedButton.styleFrom(
○                  foregroundColor: Colors.black,
○                  backgroundColor: Colors.grey[300],
○              ),
○              child: const Text("Batal"),
○              onPressed: () => Navigator.pop(context),
○          ),
○      ],
○  );

```

```

○      showDialog(context: context, builder: (context) =>
○        alertDialog);
○    }
○  }
○ }
○

```

- Screenshot Tampilan



Create

Detail Rekam Medis

← Detail Rekam Medis Pasien

Back

- ID Rekam Medis: 4
- Nama Pasien: Anna
- Gejala: Pusing
- Tingkat Keparahan: 3

EDIT DELETE

Update

← UBAH REKAM MEDIS PASIEN

Nama Pasien
Anna

Gejala
Pusing

Tingkat Keparahan (1-5)
3

UBAH

4. Jadwal Vaksinasi Page

- Menampilkan Data Vaksinasi oleh Pasien
- Kode
 - Kode Model Vaksinasi Page

```
class JadwalVaksinasi {
    String? id;
    String? patientName;
    String? vaccineType;
    int? age;
    DateTime? createdAt;
```

```

○   DateTime? updatedAt;
○
○   JadwalVaksinasi({
○     this.id,
○     this.patientName,
○     this.vaccineType,
○     this.age,
○     this.createdAt,
○     this.updatedAt,
○   });
○
○   factory JadwalVaksinasi.fromJson(Map<String, dynamic>
obj) {
○     return JadwalVaksinasi(
○       id: obj['id']?.toString(),
○       patientName: obj['person_name'],
○       vaccineType: obj['vaccine_type'],
○       age: obj['age'] is int ? obj['age'] :
int.tryParse(obj['age']?.toString() ?? '') ?? 0,
○       createdAt: DateTime.parse(obj['created_at']),
○       updatedAt: DateTime.parse(obj['updated_at']),
○     );
○   }
○ }
○

```

- Kode Bloc Vaksinasi Page
- Kode UI Vaksinasi Page
-

- Screenshot

5. Data Nutrisi

- Menampilkan Data Nutrisi
- Kode
 - Kode Model Data Nutrisi

```

○   class Nutrisi {
○     final int? id;
○     final String foodItem;
○     final int calories;
○     final double fatContent;
○
○     Nutrisi({
○       this.id,
○       required this.foodItem,
○       required this.calories,

```

```

o     required this.fatContent,
o   });
o
o   factory Nutrisi.fromJson(Map<String, dynamic> json) {
o     return Nutrisi(
o       id: json['id'],
o       foodItem: json['food_item'],
o       calories: json['calories'],
o       fatContent: json['fat_content'].toDouble(),
o     );
o   }
o
o   Map<String, dynamic> toJson() {
o     return {
o       'id': id,
o       'food_item': foodItem,
o       'calories': calories,
o       'fat_content': fatContent,
o     };
o   }
o }
o

```

o Kode Bloc Data Nutrisi

```

o import 'dart:convert';
o import 'package:manajemenkesehatan/helpers/api.dart';
o import 'package:manajemenkesehatan/helpers/api_url.dart';
o import 'package:manajemenkesehatan/model/nutrisi.dart';
o
o class NutrisiBloc {
o   static Future<List<Nutrisi>> getNutrisi() async {
o     String apiUrl = ApiUrl.listNutrisi;
o     var response = await Api().get(apiUrl);
o     if (response.statusCode == 200) {
o       var jsonObj = json.decode(response.body);
o       List<dynamic> listNutrisi = (jsonObj as Map<String,
dynamic>)[ 'data' ];
o       List<Nutrisi> nutrisilist = [];
o
o       for (var item in listNutrisi) {
o         nutrisilist.add(Nutrisi.fromJson(item));
o       }
o       return nutrisilist;
o     } else {
o       throw Exception('Failed to load nutrisilist');
o     }
o   }
o }
o

```

```

○   static Future<bool> addNutrisi({required Nutrisi
nutrisi}) async {
○     String apiUrl = ApiUrl.createNutrisi;
○     var body = {
○       "food_item": nutrisi.foodItem,
○       "calories": nutrisi.calories,
○       "fat_content": nutrisi.fatContent.toString()
○     };
○     var response = await Api().post(apiUrl, body);
○
○     if (response.statusCode == 200) {
○       var jsonObj = json.decode(response.body);
○       return jsonObj['status'];
○     } else {
○       throw Exception('Failed to add nutrisi');
○     }
○   }
○
○   static Future<bool> updateNutrisi({required Nutrisi
nutrisi}) async {
○     String apiUrl = ApiUrl.updateNutrisi(nutrisi.id!);
○     var body = {
○       "food_item": nutrisi.foodItem,
○       "calories": nutrisi.calories,
○       "fat_content": nutrisi.fatContent.toString()
○     };
○
○     var response = await Api().put(apiUrl,
jsonEncode(body));
○
○     if (response.statusCode == 200) {
○       var jsonObj = json.decode(response.body);
○       return jsonObj['status'];
○     } else {
○       throw Exception('Failed to update nutrisi');
○     }
○   }
○
○   static Future<bool> deleteNutrisi({int? id}) async {
○     String apiUrl = ApiUrl.deleteNutrisi(id!);
○     var response = await Api().delete(apiUrl);
○     var jsonObj = json.decode(response.body);
○     return (jsonObj as Map<String, dynamic>)['data'];
○   }
○ }
○

```

○ Kode UI Data Nutrisi

```

○ import 'package:flutter/material.dart';
○ import 'package:manajemenkesehatan/model/nutrisi.dart';
○ import 'package:manajemenkesehatan/ui/login_page.dart';
○ import 'package:manajemenkesehatan/bloc/logout_bloc.dart';
○ import 'package:manajemenkesehatan/ui/nutrisi_form.dart';
○ import 'package:manajemenkesehatan/bloc/NutrisiBloc.dart';
○ import 'package:manajemenkesehatan/ui/nutrisi_detail.dart';
○
○ class NutrisiPage extends StatefulWidget {
○   const NutrisiPage({Key? key}) : super(key: key);
○
○   @override
○   _NutrisiPageState createState() => _NutrisiPageState();
○ }
○
○ class _NutrisiPageState extends State<NutrisiPage> {
○   @override
○   Widget build(BuildContext context) {
○     return Scaffold(
○       appBar: AppBar(
○         title: const Text(
○           'Data Nutrisi',
○           style: TextStyle(
○             fontFamily: 'SansSerif',
○             fontSize: 22,
○             fontWeight: FontWeight.bold,
○             color: Colors.white,
○           ),
○         ),
○         backgroundColor: const Color.fromARGB(255, 7, 230,
92),
○         actions: [
○           Padding(
○             padding: const EdgeInsets.only(right: 20.0),
○             child: GestureDetector(
○               child: const Icon(Icons.add, size: 26.0,
color: Colors.white),
○               onTap: () {
○                 Navigator.push(
○                   context,
○                   MaterialPageRoute(builder: (context) =>
const NutrisiForm()),
○                 );
○               },
○             ),
○           ),
○         ],
○       ),
○     ),
○   ),
○ }

```

```

○      drawer: Drawer(
○        child: ListView(
○          children: [
○            ListTile(
○              title: const Text(
○                'Logout',
○                style: TextStyle(
○                  fontSize: 18,
○                  fontFamily: 'SansSerif',
○                ),
○            ),
○            trailing: const Icon(Icons.logout, color:
○Color.fromARGB(255, 7, 230, 92)),
○            onTap: () async {
○              await LogoutBloc.logout().then((value) {
○                Navigator.of(context).pushAndRemoveUntil(
○                  MaterialPageRoute(builder: (context) =>
○const LoginPage()),
○                (route) => false,
○              );
○            });
○          ],
○        ),
○        backgroundColor: const Color.fromARGB(255, 253, 240,
○117),
○        body: Column(
○          children: [
○            const SizedBox(height: 40),
○            Expanded(
○              child: FutureBuilder<List<Nutrisi>>(
○                future: NutrisiBloc.getNutrisi(),
○                builder: (context, snapshot) {
○                  if (snapshot.hasError) {
○                    print(snapshot.error);
○                  }
○                  return snapshot.hasData
○                    ? ListNutrisi(list: snapshot.data)
○                    : const Center(
○                      child: CircularProgressIndicator(),
○                    );
○                },
○              ),
○            ),
○          ],
○        ),
○      ],

```

```

○      ),
○    );
○  }
○ }
○
○ class ListNutrisi extends StatelessWidget {
○   final List<Nutrisi>? list;
○
○   const ListNutrisi({Key? key, this.list}) : super(key:
key);
○
○   @override
○   Widget build(BuildContext context) {
○     return ListView.builder(
○       itemCount: list?.length ?? 0,
○       itemBuilder: (context, i) {
○         return ItemNutrisi(
○           nutrisisi: list![i],
○         );
○       },
○     );
○   }
○ }
○
○ class ItemNutrisi extends StatelessWidget {
○   final Nutrisi nutrisisi;
○
○   const ItemNutrisi({Key? key, required this.nutrisisi}) :
super(key: key);
○
○   @override
○   Widget build(BuildContext context) {
○     return GestureDetector(
○       onTap: () {
○         Navigator.push(
○           context,
○           MaterialPageRoute(
○             builder: (context) => NutrisiDetail(nutrisisi:
nutrisisi),
○           ),
○       );
○     },
○     child: Card(
○       color: const Color.fromARGB(255, 252, 255, 180),
○       shape: RoundedRectangleBorder(
○         borderRadius: BorderRadius.circular(12),
○       ),
○       elevation: 4,

```

```

○      margin: const EdgeInsets.symmetric(horizontal: 16,
vertical: 8),
○      child: ListTile(
○        contentPadding: const
EdgeInsets.symmetric(horizontal: 16, vertical: 10),
○        title: Text(
○          nutrиси.foodItem,
○          style: const TextStyle(
○            fontSize: 18,
○            fontWeight: FontWeight.bold,
○            color: Colors.black87,
○          ),
○        ),
○        subtitle: Text(
○          'Kalori: ${nutrиси.calories}, Lemak:
${nutrиси.fatContent} gram',
○          style: const TextStyle(fontSize: 16, color:
Colors.grey),
○        ),
○        trailing: const Icon(
○          Icons.arrow_forward_ios,
○          color: Color.fromARGB(255, 7, 230, 92),
○          size: 20,
○        ),
○      ),
○    ),
○  );
○ }
○ }
○

```

○ Kode Form Data Nutrиси

```

○ import 'package:flutter/material.dart';
○ import 'package:manajemenkesehatan/bloc/NutrисиBloc.dart';
○ // Assuming you have a similar Bloc for Nutrиси
○ import 'package:manajemenkesehatan/model/nutrиси.dart';
○ import 'package:manajemenkesehatan/ui/nutrиси_page.dart';
○ import
○   'package:manajemenkesehatan/widget/warning_dialog.dart';
○
○ class NutrисиForm extends StatefulWidget {
○   final Nutrиси? nutrиси;
○   const NutrисиForm({Key? key, this.nutrиси}) : super(key:
key);
○
○   @override
○   _NutrисиFormState createState() => _NutrисиFormState();
○ }

```



```

○
○ class _NutrisiFormState extends State<NutrisiForm> {
○   final _formKey = GlobalKey<FormState>();
○   bool _isLoading = false;
○   String judul = "TAMBAH NUTRISI";
○   String tombolSubmit = "SIMPAN";
○
○   final _foodItemController = TextEditingController();
○   final _caloriesController = TextEditingController();
○   final _fatContentController = TextEditingController();
○
○   @override
○   void initState() {
○     super.initState();
○     isUpdate();
○   }
○
○   void isUpdate() {
○     if (widget.nutrisi != null) {
○       setState(() {
○         judul = "UBAH NUTRISI";
○         tombolSubmit = "UBAH";
○         _foodItemController.text =
widget.nutrisi!.foodItem;
○         _caloriesController.text =
widget.nutrisi!.calories.toString();
○         _fatContentController.text =
widget.nutrisi!.fatContent.toString();
○       });
○     }
○   }
○
○   @override
○   Widget build(BuildContext context) {
○     return Scaffold(
○       appBar: AppBar(
○         title: Text(judul),
○         backgroundColor: Color.fromARGB(255, 7, 230, 92),
○       ),
○       backgroundColor: const Color.fromARGB(255, 253, 240,
117),
○       body: SingleChildScrollView(
○         child: Padding(
○           padding: const EdgeInsets.all(16.0),
○           child: Form(
○             key: _formKey,
○             child: Column(
○               children: [

```

```

○         const SizedBox(height: 50),
○         _foodItemTextField(),
○         const SizedBox(height: 16),
○         _caloriesTextField(),
○         const SizedBox(height: 16),
○         _fatContentTextField(),
○         const SizedBox(height: 30),
○         _buttonSubmit(),
○     ],
○ ),
○ ),
○ ),
○ ),
○ );
○ }
○
○ Widget _foodItemTextField() {
○     return TextFormField(
○         decoration: InputDecoration(
○             labelText: "Nama Makanan",
○             border: const OutlineInputBorder(),
○             labelStyle: TextStyle(color: Colors.black),
○         ),
○         controller: _foodItemController,
○         validator: (value) {
○             if (value!.isEmpty) {
○                 return "Nama Makanan harus diisi";
○             }
○             return null;
○         },
○     );
○ }
○
○ Widget _caloriesTextField() {
○     return TextFormField(
○         decoration: InputDecoration(
○             labelText: "Kalori",
○             border: const OutlineInputBorder(),
○             labelStyle: TextStyle(color: Colors.black),
○         ),
○         keyboardType: TextInputType.number,
○         controller: _caloriesController,
○         validator: (value) {
○             if (value!.isEmpty) {
○                 return "Kalori harus diisi";
○             }
○             if (int.tryParse(value) == null || int.parse(value)
< 0) {

```

```

o         return "Kalori harus berupa angka positif";
o     }
o     return null;
o },
o );
o }
o
o Widget _fatContentTextField() {
o     return TextFormField(
o         decoration: InputDecoration(
o             labelText: "Konten Lemak (gram)",
o             border: const OutlineInputBorder(),
o             labelStyle: TextStyle(color: Colors.black),
o         ),
o         keyboardType: TextInputType.number,
o         controller: _fatContentController,
o         validator: (value) {
o             if (value!.isEmpty) {
o                 return "Konten Lemak harus diisi";
o             }
o             if (double.tryParse(value) == null ||
double.parse(value) < 0) {
o                 return "Konten Lemak harus berupa angka positif";
o             }
o             return null;
o         },
o     );
o }
o
o Widget _buttonSubmit() {
o     return ElevatedButton(
o         style: ElevatedButton.styleFrom(
o             padding: const EdgeInsets.symmetric(vertical: 16,
horizontal: 40),
o             shape: RoundedRectangleBorder(
o                 borderRadius: BorderRadius.circular(20),
o             ),
o             backgroundColor: Color.fromARGB(255, 7, 230, 92),
o             // Warna tombol
o             foregroundColor: Colors.white, // Teks tombol
berwarna putih
o         ),
o         child: _isLoading
o             ? const CircularProgressIndicator(color:
Colors.white)
o             : Text(
o                 tombolSubmit,
o                 style: const TextStyle(fontSize: 18),

```

```

○         ),
○         onPressed: () {
○             if (_formKey.currentState!.validate()) {
○                 if (!_isLoading) {
○                     if (widget.nutrisi != null) {
○                         ubah();
○                     } else {
○                         simpan();
○                     }
○                 }
○             }
○         },
○     );
○ }

○ void simpan() {
○     setState(() {
○         _isLoading = true;
○     });

○     Nutrisi newNutrisi = Nutrisi(
○         id: null,
○         foodItem: _foodItemController.text,
○         calories: int.parse(_caloriesController.text),
○         fatContent: double.parse(_fatContentController.text),
○     );

○     NutrisiBloc.addNutrisi(nutrisi: newNutrisi).then(
○         (value) {
○             Navigator.of(context).pushReplacement(
○                 MaterialPageRoute(
○                     builder: (BuildContext context) => const
○ NutrisiPage(),
○                 ),
○             );
○         },
○         onError: (error) {
○             showDialog(
○                 context: context,
○                 builder: (BuildContext context) => const
○ WarningDialog(
○                     description: "Simpan gagal, silahkan coba
lagi",
○                 ),
○             );
○         },
○     ).whenComplete(() {
○         setState(() {

```

```

○      _isLoading = false;
○    });
○  });
○ }
○
○ void ubah() {
○   setState(() {
○     _isLoading = true;
○   });
○
○   Nutrisi updatedNutrisi = Nutrisi(
○     id: widget.nutrisi!.id,
○     foodItem: _foodItemController.text,
○     calories: int.parse(_caloriesController.text),
○     fatContent: double.parse(_fatContentController.text),
○   );
○
○   NutrisiBloc.updateNutrisi(nutrisi:
updatedNutrisi).then(
○     (value) {
○       Navigator.of(context).pushReplacement(
○         MaterialPageRoute(
○           builder: (BuildContext context) => const
NutrisiPage(),
○         ),
○       );
○     },
○     onError: (error) {
○       showDialog(
○         context: context,
○         builder: (BuildContext context) => const
WarningDialog(
○           description: "Ubah data gagal, silahkan coba
lagi",
○         ),
○       );
○     },
○   ).whenComplete(() {
○     setState(() {
○       _isLoading = false;
○     });
○   });
○ }
○ }
○
○

```

○ Kode Detail Data Nutrisi

```

○ import 'package:flutter/material.dart';

```

```

o import 'package:manajemenkesehatan/model/nutrisi.dart';
o
o class NutrisiDetail extends StatelessWidget {
o   final Nutrisi nutrisi;
o
o   const NutrisiDetail({Key? key, required this.nutrisi}) :
super(key: key);
o
o   @override
o   Widget build(BuildContext context) {
o     return Scaffold(
o       appBar: AppBar(
o         title: Text(nutrisi.foodItem),
o         backgroundColor: const Color.fromARGB(255, 7, 230,
92),
o       ),
o       body: Padding(
o         padding: const EdgeInsets.all(16.0),
o         child: Column(
o           crossAxisAlignment: CrossAxisAlignment.start,
o           children: [
o             Text(
o               'Nama Makanan:',
o               style: const TextStyle(fontSize: 20,
fontWeight: FontWeight.bold),
o             ),
o             Text(
o               nutrisi.foodItem,
o               style: const TextStyle(fontSize: 18),
o             ),
o             const SizedBox(height: 16),
o             Text(
o               'Kalori:',
o               style: const TextStyle(fontSize: 20,
fontWeight: FontWeight.bold),
o             ),
o             Text(
o               '${nutrisi.calories} kcal',
o               style: const TextStyle(fontSize: 18),
o             ),
o             const SizedBox(height: 16),
o             Text(
o               'Konten Lemak:',
o               style: const TextStyle(fontSize: 20,
fontWeight: FontWeight.bold),
o             ),
o             Text(
o               '${nutrisi.fatContent} gram',

```

```

○         style: const TextStyle(fontSize: 18),
○     ),
○     const SizedBox(height: 16),
○     // You can add more nutritional information
here if available
○     Text(
○         'Informasi Nutrisi Lainnya:',
○         style: const TextStyle(fontSize: 20,
fontWeight: FontWeight.bold),
○     ),
○     const SizedBox(height: 16),
○     ElevatedButton(
○         onPressed: () {
○             // Implement any action for the button, if
needed
○         },
○         child: const Text('Kembali'),
○     ),
○ ],
○ ),
○ ),
○ );
○ }
○ }
○

```

- Screenshot

Tambah

← TAMBAH NUTRISI

Nama Makanan
Tahu

Kalori
26

Konten Lemak (gram)
78

SIMPAN