

Nama : Rizqullah Abiyyu Hade
NIM : H1D022091
Shift : A

Responsi Praktikum Pemrograman Mobile – Paket Pariwisata

Segmen Satu : Aplikasi Manajemen Pariwisata
Segmen Dua : 1. Nama Tabel: ulasan
2. Kolom 1: id (int, PK, increment)
3. Kolom 2: reviewer (String)
4. Kolom 3: rating (Integer)
5. Kolom 4: comments (String)
Segmen Tiga : Tema Terang Merah, Font Arial

1. API Service

a. Source: lib/service/api_service.dart

```
import 'dart:convert';
import 'package:http/http.dart' as http;
import '../models/UlasanModel.dart';
import '../models/RegisterModel.dart';
import '../models/LoginModel.dart';

class ApiService {
  final String baseUrl = 'http://responsi.webwizards.my.id/api';

  Future<bool> register(RegisterModel registerModel) async {
    try {
      final url = Uri.parse('$baseUrl/registrasi');
      final response = await http.post(
        url,
        headers: {'Content-Type': 'application/json'},
        body: jsonEncode(registerModel.toJson()),
      );

      if (response.statusCode == 200) {
        print('Response Body: ${response.body}');
        final data = jsonDecode(response.body);
        return data['status'] == true;
      } else {
        final data = jsonDecode(response.body);
        print(
          'Error: ${response.statusCode}, ${data['message']} ?? 'Unknown error'');
        return false;
      }
    } catch (e) {
      print('Error: $e');
    }
  }
}
```

```

        return false;
    }
}

Future<void> login(LoginModel loginModel) async {
    final response = await http.post(
        Uri.parse('$baseUrl/login'),
        headers: {'Content-Type': 'application/json'},
        body: jsonEncode(loginModel.toJson()),
    );

    if (response.statusCode == 200) {
        final Map<String, dynamic> responseData = json.decode(response.body);
        if (responseData['status'] == true) {
            return;
        } else {
            throw Exception('Login gagal: ${responseData['message']}');
        }
    } else {
        throw Exception('Failed to login: ${response.body}');
    }
}

Future<List<UlasanModel>> getAllUlasan() async {
    final url = Uri.parse('$baseUrl/pariwisata/ulasan');
    final response = await http.get(url);

    if (response.statusCode == 200) {
        final List<dynamic> data = jsonDecode(response.body)['data'];
        return data.map((item) => UlasanModel.fromJson(item)).toList();
    } else {
        throw Exception('Failed to load ulasan');
    }
}

Future<bool> createUlasan(UlasanModel ulasanModel) async {
    final url = Uri.parse('$baseUrl/pariwisata/ulasan');
    final response = await http.post(
        url,
        headers: {'Content-Type': 'application/json'},
        body: jsonEncode(ulasanModel.toJson()),
    );

    return response.statusCode == 200;
}

```

```

Future<bool> updateUlasan(UlasanModel ulasanModel) async {
  final url =
    Uri.parse('$baseUrl/pariwisata/ulasan/${ulasanModel.id}/update');
  final response = await http.put(
    url,
    headers: {'Content-Type': 'application/json'},
    body: jsonEncode(ulasanModel.toJson()),
  );

  print('Update Ulasan Response: ${response.statusCode}, ${response.body}');

  return response.statusCode == 200;
}

Future<bool> deleteUlasan(int id) async {
  final url = Uri.parse('$baseUrl/pariwisata/ulasan/$id/delete');
  print('Delete URL: $url');
  final response = await http.delete(url);

  print('Delete Ulasan Response: ${response.statusCode}, ${response.body}');

  if (response.statusCode == 200) {
    final responseData = jsonDecode(response.body);
    if (responseData['code'] == 200 && responseData['status'] == true) {
      return true;
    } else {
      return false;
    }
  }
  return false;
}
}

```

Source code ini berisi class ApiService yang bertanggung jawab untuk mengelola permintaan HTTP terkait registrasi, login, dan manajemen ulasan pada aplikasi manajemen pariwisata. Kelas ini berinteraksi dengan API menggunakan URL dasar yang disimpan dalam variabel baseUrl. Terdapat beberapa metode di dalamnya:

1. register untuk mengirim data pendaftaran pengguna ke endpoint registrasi dan memeriksa apakah registrasi berhasil berdasarkan status dari respons API.
2. login untuk melakukan autentikasi pengguna dengan mengirim data login dan menangani respons yang diterima.
3. getAllUlasan untuk mengambil semua ulasan yang tersedia dari API dan mengonversi data JSON ke dalam bentuk objek UlasanModel.
4. createUlasan untuk menambahkan ulasan baru ke dalam sistem dengan mengirimkan data ulasan ke API.
5. updateUlasan untuk memperbarui ulasan yang ada dengan ID tertentu, mengirim data ulasan yang telah diubah ke API.

6. deleteUlasan untuk menghapus ulasan berdasarkan ID dan memeriksa status penghapusan berdasarkan respons API.

2. Register

a. Source: lib/pages/RegisterPage.dart

```
import 'package:flutter/material.dart';
import 'package:manajemen_pariwisata/services/api_service.dart';
import 'package:manajemen_pariwisata/models/RegisterModel.dart';
import 'package:manajemen_pariwisata/pages/LoginPage.dart';

class RegisterPage extends StatefulWidget {
  @override
  _RegisterPageState createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  final ApiService apiService = ApiService();
  final _formKey = GlobalKey<FormState>();
  String nama = "";
  String email = "";
  String password = "";

  void _register() async {
    if (_formKey.currentState!.validate()) {
      RegisterModel registerModel = RegisterModel(
        nama: nama,
        email: email,
        password: password,
      );

      bool success = await apiService.register(registerModel);

      if (success) {
        _showDialog('Registrasi berhasil!', true);
      } else {
        _showDialog('Registrasi gagal!', false);
      }
    }
  }

  void _showDialog(String message, bool isSuccess) {
    showDialog(
      context: context,
      builder: (context) {
        return AlertDialog(
          title: Text(isSuccess ? 'Sukses' : 'Gagal'),
```

```

        content: Text(message),
        actions: [
          TextButton(
            onPressed: () {
              Navigator.of(context).pop();
              if (isSuccess) {
                Navigator.pushReplacement(
                  context,
                  MaterialPageRoute(builder: (context) => LoginPage()),
                );
              }
            },
            child: Text('OK'),
          ),
        ],
      );
    },
  );
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Registrasi', style: TextStyle(fontFamily: 'Arial'))),
    body: Container(
      decoration: BoxDecoration(
        gradient: LinearGradient(
          colors: [Colors.redAccent, Colors.white],
          begin: Alignment.topLeft,
          end: Alignment.bottomRight,
        ),
      ),
      child: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Form(
          key: _formKey,
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              TextFormField(
                decoration: InputDecoration(
                  labelText: 'Nama',
                  border: OutlineInputBorder(),
                  prefixIcon: Icon(Icons.person),
                ),
              ),
            ],
          ),
        ),
      ),
    ),
  );
}

```

```

onChanged: (value) => nama = value,
validator: (value) =>
    value!.isEmpty ? 'Nama tidak boleh kosong' : null,
style: TextStyle(fontFamily: 'Arial'),
),
SizedBox(height: 16),
TextFormField(
    decoration: InputDecoration(
        labelText: 'Email',
        border: OutlineInputBorder(),
        prefixIcon: Icon(Icons.email),
    ),
    onChanged: (value) => email = value,
    validator: (value) =>
        value!.isEmpty ? 'Email tidak boleh kosong' : null,
    style: TextStyle(fontFamily: 'Arial'),
),
SizedBox(height: 16),
TextFormField(
    decoration: InputDecoration(
        labelText: 'Password',
        border: OutlineInputBorder(),
        prefixIcon: Icon(Icons.lock),
    ),
    obscureText: true,
    onChanged: (value) => password = value,
    validator: (value) =>
        value!.isEmpty ? 'Password tidak boleh kosong' : null,
    style: TextStyle(fontFamily: 'Arial'),
),
SizedBox(height: 20),
ElevatedButton(
    onPressed: _register,
    style: ElevatedButton.styleFrom(
        padding: EdgeInsets.symmetric(vertical: 12),
        shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(8),
        ),
    ),
    child: Text('Daftar', style: TextStyle(fontFamily: 'Arial')),
),
),
],

```

```
);
}
}
```

Halaman ini menggunakan StatefulWidget karena memerlukan pembaruan data secara dinamis berdasarkan input dari pengguna. Formulir pendaftaran ini dikendalikan oleh GlobalKey<FormState> yang memungkinkan validasi terhadap input yang diberikan pengguna. Tiga TextFormField disediakan untuk mengisi nama, email, dan password, masing-masing dengan validasi yang memastikan bahwa tidak ada kolom yang kosong.

Ketika tombol "Daftar" ditekan, metode _register() akan dipanggil. Metode ini membuat objek RegisterModel yang berisi data pengguna, kemudian mengirimkannya ke API melalui metode register() dari ApiService. Apabila registrasi berhasil, pengguna akan diberi konfirmasi melalui dialog sukses, dan diarahkan ke halaman login. Sebaliknya, jika registrasi gagal, dialog error akan ditampilkan.

Tampilan halaman ini menggunakan LinearGradient sebagai latar belakang dengan gradasi warna merah dan putih, memberikan tampilan yang lebih menarik. Semua elemen dalam formulir diatur secara vertikal menggunakan Column, dengan spasi yang diatur menggunakan SizedBox. Setiap TextFormField dilengkapi dengan ikon untuk memudahkan pengguna dalam memahami input yang diminta, dan tombol "Daftar" dilengkapi dengan gaya yang sederhana namun fungsional.

b. Source: lib/modes/RegisterModel.dart

```
class RegisterModel {
  String nama;
  String email;
  String password;

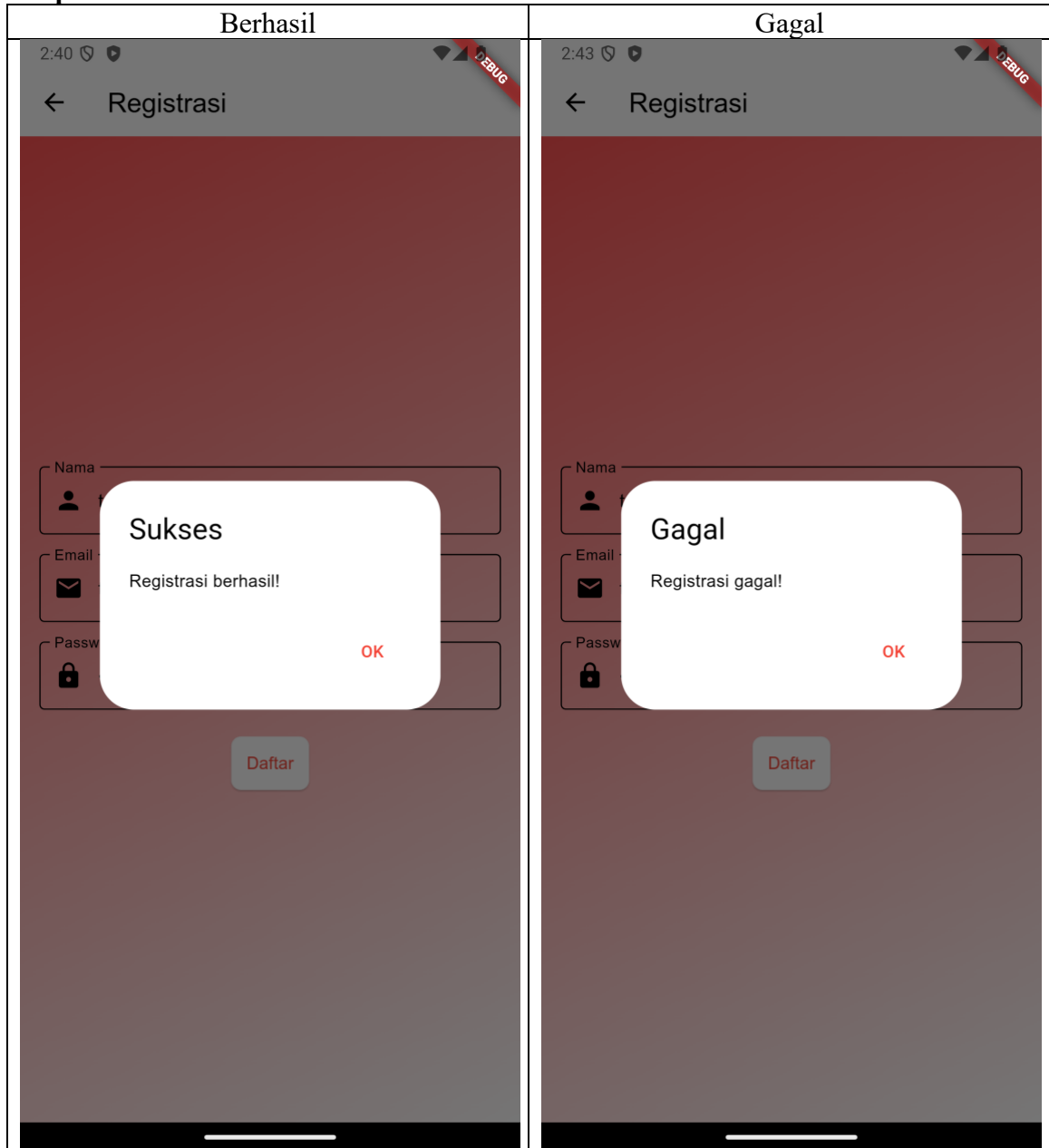
  RegisterModel({
    required this.nama,
    required this.email,
    required this.password,
  });

  Map<String, dynamic> toJson() {
    return {
      'nama': nama,
      'email': email,
      'password': password,
    };
  }
}
```

Source code ini mendefinisikan kelas RegisterModel, yang digunakan untuk merepresentasikan data pendaftaran pengguna, yaitu nama, email, dan password. Kelas ini memiliki tiga atribut: nama, email, dan password, yang semuanya diharuskan (required). Selain itu, terdapat metode toJson() yang mengubah objek RegisterModel

menjadi format Map<String, dynamic>, yang kemudian digunakan untuk mengirim data dalam format JSON saat melakukan pendaftaran pengguna ke API. Metode ini memastikan bahwa atribut-atribut kelas tersebut dapat dengan mudah dikonversi menjadi key-value pair yang dapat dikirimkan dalam permintaan HTTP.

c. Output



3. Login

a. Source: lib/pages/LoginPage.dart

```
import 'package:flutter/material.dart';  
import 'package:manajemen_pariwisata/services/api_service.dart';  
import 'package:manajemen_pariwisata/models/LoginModel.dart';
```



```

import 'package:manajemen_pariwisata/pages/RegisterPage.dart';
import 'package:manajemen_pariwisata/pages/UlasanPage.dart';

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final ApiService apiService = ApiService();
  final _formKey = GlobalKey<FormState>();
  String email = "";
  String password = "";

  void _login() async {
    if (_formKey.currentState!.validate()) {
      LoginModel loginModel = LoginModel(email: email, password: password);
      try {
        await ApiService().login(loginModel);

        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(content: Text('Login berhasil!')),
        );

        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) => UlasanPage()),
        );
      } catch (e) {
        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(content: Text('Login gagal!')),
        );
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            colors: [Colors.redAccent, Colors.white],
            begin: Alignment.topLeft,
            end: Alignment.bottomRight,
          ),
        ),
      ),
    );
  }
}

```

```

),
child: Center(
  child: SingleChildScrollView(
    padding: const EdgeInsets.all(16.0),
    child: Card(
      elevation: 8,
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(12),
      ),
    ),
    child: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Form(
        key: _formKey,
        child: Column(
          mainAxisAlignment: MainAxisAlignment.min,
          children: [
            Text(
              'Login',
              style: TextStyle(
                fontSize: 24,
                fontWeight: FontWeight.bold,
                fontFamily: 'Arial',
                color: Colors.redAccent,
              ),
            ),
            SizedBox(height: 20),
            TextFormField(
              decoration: InputDecoration(
                labelText: 'Email',
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.email),
              ),
              onChanged: (value) => email = value,
              validator: (value) =>
                value!.isEmpty ? 'Email tidak boleh kosong' : null,
              style: TextStyle(fontFamily: 'Arial'),
            ),
            SizedBox(height: 16),
            TextFormField(
              decoration: InputDecoration(
                labelText: 'Password',
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.lock),
              ),
              obscureText: true,
              onChanged: (value) => password = value,

```

```

        validator: (value) => value!.isEmpty
          ? 'Password tidak boleh kosong'
          : null,
        style: TextStyle(fontFamily: 'Arial'),
      ),
      SizedBox(height: 20),
      ElevatedButton(
        onPressed: _login,
        style: ElevatedButton.styleFrom(
          padding: EdgeInsets.symmetric(vertical: 12),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(8),
          ),
        ),
        child: Text('Login',
          style: TextStyle(fontFamily: 'Arial')),
      ),
      SizedBox(height: 20),
      TextButton(
        onPressed: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => RegisterPage(),
            ),
          );
        },
        child: Text(
          'Belum punya akun? Daftar di sini.',
          style: TextStyle(fontFamily: 'Arial'),
        ),
      ),
    ],
  ),
),
),
),
),
),
),
),
),
),
);
}
}

```

Halaman ini menggunakan StatefulWidget untuk memungkinkan interaksi pengguna. Terdapat formulir login dengan validasi menggunakan _formKey, di mana pengguna mengisi email dan password. Jika formulir valid, metode _login dipanggil untuk mengirim data ke API melalui kelas ApiService. Jika login berhasil, aplikasi akan

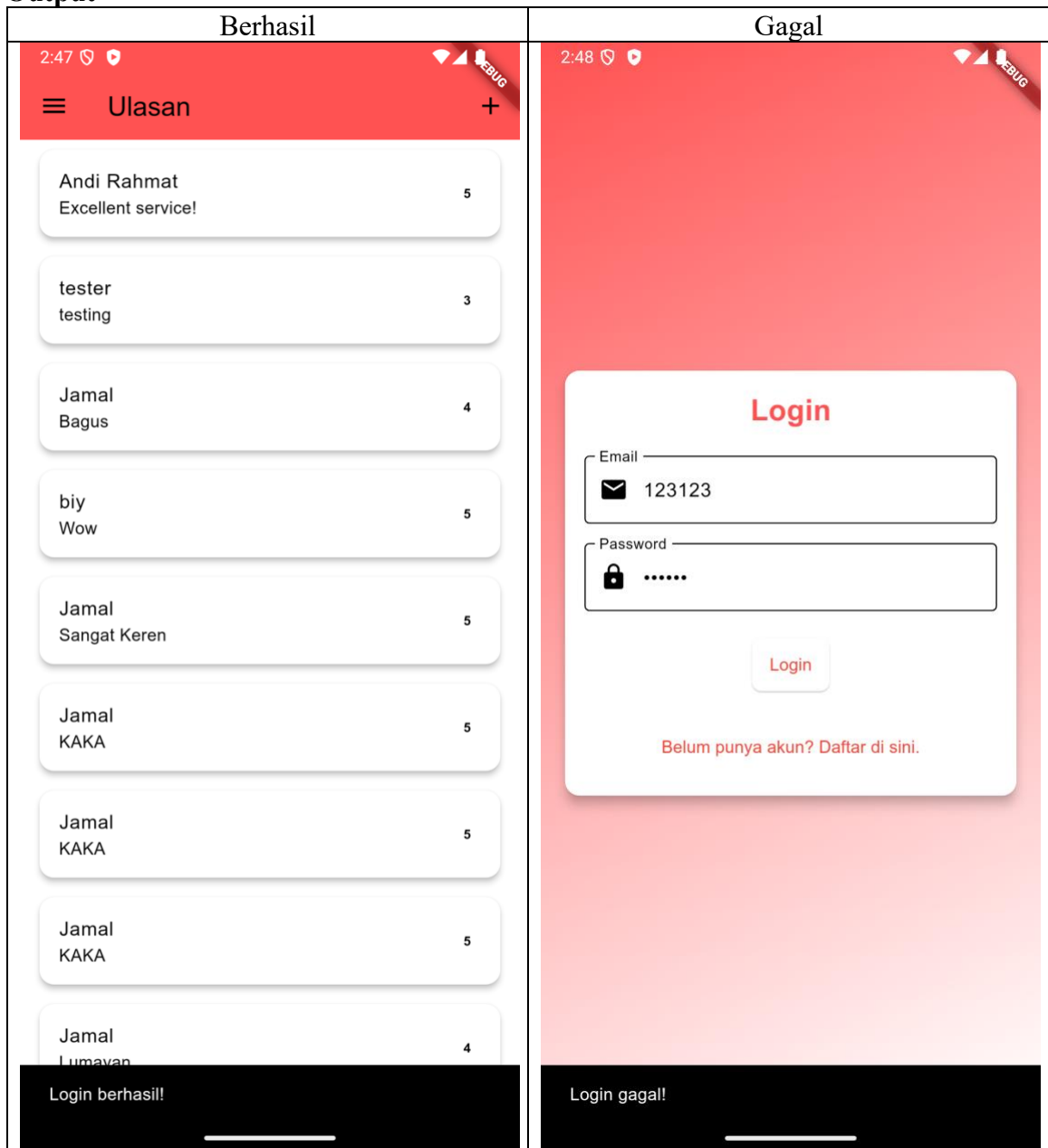
menampilkan pesan keberhasilan menggunakan ScaffoldMessenger dan kemudian mengarahkan pengguna ke halaman ulasan (UlasanPage). Jika gagal, aplikasi akan menampilkan pesan kesalahan. Selain itu, pengguna juga bisa mengakses halaman pendaftaran (RegisterPage) jika belum memiliki akun. Layout halaman ini didesain dengan kombinasi elemen-elemen UI seperti Card, TextFormField, dan ElevatedButton, serta dihiasi dengan gradien warna.

b. Source: lib/models/LoginModel.dart

```
class LoginModel {  
  String email;  
  String password;  
  
  LoginModel({  
    required this.email,  
    required this.password,  
  });  
  
  Map<String, dynamic> toJson() {  
    return {  
      'email': email,  
      'password': password,  
    };  
  }  
}
```

Model ini memiliki dua properti, yaitu email dan password, yang diperlukan untuk login. Keduanya diinisialisasi melalui konstruktor dengan parameter yang wajib diisi (required). Selain itu, terdapat metode toJson() yang mengonversi data LoginModel menjadi bentuk map (Map<String, dynamic>), agar data tersebut dapat di-encode menjadi format JSON untuk dikirim melalui API dalam proses login.

c. Output



4. Ulasan

a. Source: lib/pages/UlasanPage.dart

```
import 'package:flutter/material.dart';
import 'package:manajemen_pariwisata/services/api_service.dart';
import 'package:manajemen_pariwisata/models/UlasanModel.dart';
import 'package:manajemen_pariwisata/pages/LoginPage.dart';
import 'package:manajemen_pariwisata/pages/CreateUlasanPage.dart';
import 'package:manajemen_pariwisata/pages/DetailUlasanPage.dart';

class UlasanPage extends StatefulWidget {
```

```

@override
_UlasanPageState createState() => _UlasanPageState();
}

class _UlasanPageState extends State<UlasanPage> {
  final ApiService apiService = ApiService();
  List<UlasanModel> ulasanList = [];

  @override
  void initState() {
    super.initState();
    _getAllUlasan();
  }

  void _getAllUlasan() async {
    ulasanList = await apiService.getAllUlasan();
    setState(() {});
  }

  void _createUlasan() async {
    final result = await Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => CreateUlasanPage()),
    );
    if (result == true) {
      _getAllUlasan();
    }
  }

  void _editUlasan(UlasanModel ulasan) async {
    final updatedUlasan = await Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) => CreateUlasanPage(ulasan: ulasan),
      ),
    );
    if (updatedUlasan != null) {
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(
          builder: (context) => DetailUlasanPage(ulasan: updatedUlasan),
        ),
      );
    }
  }
}

```

```

void _deleteUlasan(int id) async {
  bool success = await apiService.deleteUlasan(id);
  String message =
    success ? 'Ulasan berhasil dihapus!' : 'Gagal menghapus ulasan!';

  ScaffoldMessenger.of(context).showSnackBar(
    SnackBar(
      content: Text(
        message,
        style: TextStyle(fontFamily: 'Arial'),
      ),
      backgroundColor: success ? Colors.green : Colors.redAccent,
      duration: Duration(seconds: 2),
    ),
  );

  if (success) {
    _getAllUlasan();
  }
}

void _logout() {
  Navigator.pushReplacement(
    context,
    MaterialPageRoute(builder: (context) => LoginPage()),
  );
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Ulasan', style: TextStyle(fontFamily: 'Arial')),
      backgroundColor: Colors.redAccent,
      actions: [
        IconButton(
          icon: Icon(Icons.add),
          onPressed: _createUlasan,
        ),
      ],
    ),
    drawer: Drawer(
      child: ListView(
        padding: EdgeInsets.zero,
        children: <Widget>[
          DrawerHeader(

```

```

        decoration: BoxDecoration(
          color: Colors.redAccent,
        ),
        child: Text(
          'Menu',
          style: TextStyle(
            color: Colors.white,
            fontSize: 24,
            fontWeight: FontWeight.bold,
            fontFamily: 'Arial',
          ),
        ),
      ),
    ),
    ListTile(
      title: Text('Logout', style: TextStyle(fontFamily: 'Arial')),
      onTap: () {
        Navigator.pop(context);
        _logout();
      },
    ),
  ],
),
body: ulasanList.isEmpty
    ? Center(
      child: Text(
        'Belum ada ulasan.',
        style: TextStyle(fontFamily: 'Arial', fontSize: 18),
      ),
    )
    : ListView.builder(
      itemCount: ulasanList.length,
      itemBuilder: (context, index) {
        return Card(
          margin: EdgeInsets.symmetric(vertical: 8, horizontal: 16),
          elevation: 4,
          child: ListTile(
            title: Text(ulasanList[index].reviewer,
              style: TextStyle(fontFamily: 'Arial')),
            subtitle: Text(ulasanList[index].comments,
              style: TextStyle(fontFamily: 'Arial')),
            trailing: Text(
              ulasanList[index].rating.toString(),
              style: TextStyle(
                fontFamily: 'Arial', fontWeight: FontWeight.bold),
            ),
          ),
        );
      },
    ),
  ),
),
)

```



```

        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => DetailUlasanPage(
                ulasan: ulasanList[index],
              ),
            ),
          );
        },
        onLongPress: () => _deleteUlasan(ulasanList[index].id!),
      ),
    );
  },
),
);
}
}
}

```

Halaman ini menampilkan daftar ulasan pariwisata yang diambil dari API melalui kelas ApiService. Pada inisialisasi, metode `_getAllUlasan()` dipanggil untuk mengambil data ulasan dan ditampilkan dalam bentuk daftar menggunakan widget `ListView.builder`. Pengguna dapat menambahkan ulasan baru dengan tombol "+" pada app bar, yang membuka halaman `CreateUlasanPage`. Selain itu, pengguna juga dapat mengedit ulasan dengan menekan lama pada item ulasan, atau menghapus ulasan menggunakan metode `_deleteUlasan()`. Ada juga opsi untuk logout melalui menu di drawer, yang membawa pengguna kembali ke halaman login.

b. Source: lib/model/UlasanModel.dart

```

class UlasanModel {
  int? id;
  String reviewer;
  int rating;
  String comments;
  DateTime? createdAt;
  DateTime? updatedAt;

  UlasanModel({
    this.id,
    required this.reviewer,
    required this.rating,
    required this.comments,
    this.createdAt,
    this.updatedAt,
  });

  factory UlasanModel.fromJson(Map<String, dynamic> json) {

```

```

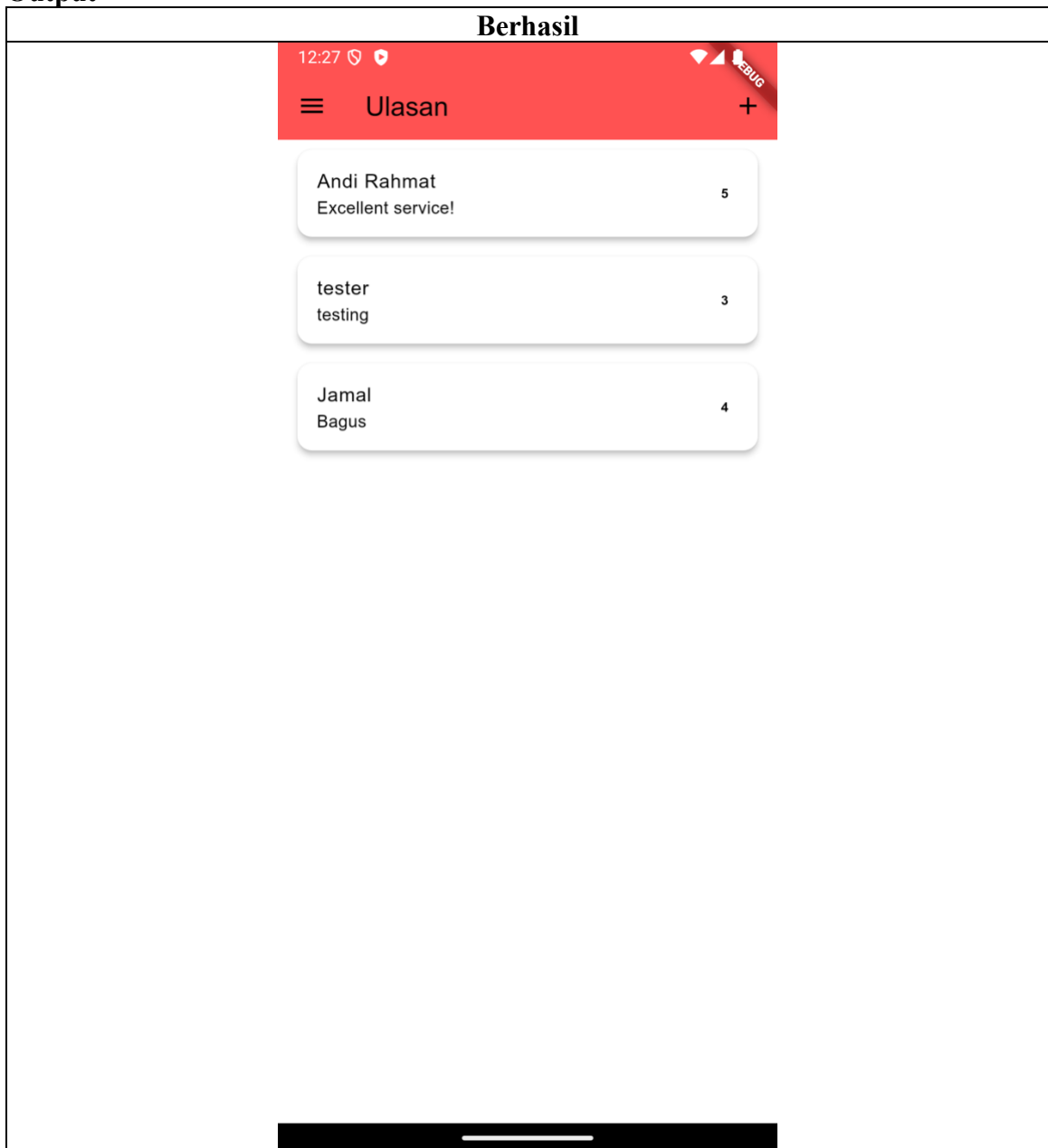
return UlasanModel(
    id: json['id'] is int ? json['id'] : int.tryParse(json['id'].toString()),
    reviewer: json['reviewer'],
    rating: json['rating'] is int
        ? json['rating']
        : int.tryParse(json['rating'].toString()),
    comments: json['comments'],
    createdAt: json['created_at'] != null
        ? DateTime.parse(json['created_at'])
        : null,
    updatedAt: json['updated_at'] != null
        ? DateTime.parse(json['updated_at'])
        : null,
);
}

Map<String, dynamic> toJson() {
    return {
        'id': id,
        'reviewer': reviewer,
        'rating': rating,
        'comments': comments,
        'created_at': createdAt?.toIso8601String(),
        'updated_at': updatedAt?.toIso8601String(),
    };
}
}

```

Model ini memiliki beberapa atribut seperti id, reviewer, rating, comments, createdAt, dan updatedAt. Pada bagian fromJson, data JSON yang diterima dari API dikonversi menjadi objek UlasanModel, dengan penanganan tipe data yang lebih fleksibel, seperti memastikan bahwa id dan rating dikonversi ke tipe integer, meskipun dalam beberapa kasus API mungkin mengirimnya sebagai string. Metode toJson digunakan untuk mengonversi objek UlasanModel kembali menjadi format JSON, misalnya ketika akan dikirimkan ke API. Tanggal createdAt dan updatedAt dikonversi ke format ISO 8601 ketika di-serialisasi.

c. Output



5. Ulasan (*Create dan Edit*)

a. Source: `lib/pages/CreateUlasanPage.dart`

```
import 'package:flutter/material.dart';
import 'package:manajemen_pariwisata/services/api_service.dart';
import 'package:manajemen_pariwisata/models/UlasanModel.dart';

class CreateUlasanPage extends StatefulWidget {
  final UlasanModel? ulasan;
```

```

CreateUlasanPage({this.ulasan});

@override
_CreateUlasanPageState createState() => _CreateUlasanPageState();
}

class _CreateUlasanPageState extends State<CreateUlasanPage> {
  final ApiService apiService = ApiService();
  final _formKey = GlobalKey<FormState>();
  String reviewer = "";
  int rating = 0;
  String comments = "";

  @override
  void initState() {
    super.initState();
    if (widget.ulasan != null) {
      reviewer = widget.ulasan!.reviewer;
      rating = widget.ulasan!.rating;
      comments = widget.ulasan!.comments;
    }
  }

  void _saveUlasan() async {
    if (_formKey.currentState!.validate()) {
      UlasanModel ulasanModel = UlasanModel(
        id: widget.ulasan?.id,
        reviewer: reviewer,
        rating: rating,
        comments: comments,
      );

      bool success;
      if (widget.ulasan == null) {
        success = await apiService.createUlasan(ulasanModel);
        if (success) {
          Navigator.pop(context, ulasanModel);
        }
      } else {
        success = await apiService.updateUlasan(ulasanModel);
        if (success) {
          Navigator.pop(context, ulasanModel);
        }
      }

      if (!success) {

```

```

        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(content: Text('Gagal menyimpan ulasan!')),
        );
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.ulasan == null ? 'Buat Ulasan' : 'Edit Ulasan',
          style: TextStyle(fontFamily: 'Arial')),
        backgroundColor: Colors.redAccent,
      ),
      body: Container(
        color: Colors.red[50],
        padding: const EdgeInsets.all(16.0),
        child: Form(
          key: _formKey,
          child: ListView(
            children: [
              TextFormField(
                decoration: InputDecoration(
                  labelText: 'Reviewer',
                  border: OutlineInputBorder(),
                ),
                initialValue: reviewer,
                onChanged: (value) => reviewer = value,
                validator: (value) =>
                  value!.isEmpty ? 'Reviewer tidak boleh kosong' : null,
              ),
              SizedBox(height: 16),
              TextFormField(
                decoration: InputDecoration(
                  labelText: 'Rating (1-5)',
                  border: OutlineInputBorder(),
                ),
                initialValue: rating.toString(),
                keyboardType: TextInputType.number,
                onChanged: (value) {
                  if (value.isNotEmpty) {
                    rating = int.parse(value);
                  }
                },
                validator: (value) {

```



```

        if (value!.isEmpty) return 'Rating tidak boleh kosong';
        int? ratingValue = int.tryParse(value);
        if (ratingValue == null ||
            ratingValue < 1 ||
            ratingValue > 5) {
            return 'Rating harus antara 1-5';
        }
        return null;
    },
),
 SizedBox(height: 16),
 TextFormField(
    decoration: InputDecoration(
        labelText: 'Comments',
        border: OutlineInputBorder(),
    ),
    initialValue: comments,
    onChanged: (value) => comments = value,
    validator: (value) =>
        value!.isEmpty ? 'Comments tidak boleh kosong' : null,
),
 SizedBox(height: 20),
 ElevatedButton(
    onPressed: _saveUlasan,
    style: ElevatedButton.styleFrom(
        backgroundColor: const Color.fromARGB(255, 254, 254, 254),
    ),
    child: Text(
        widget.ulasan == null ? 'Buat Ulasan' : 'Simpan Ulasan',
        style: TextStyle(fontFamily: 'Arial')),
),
],
),
),
),
);
}
}

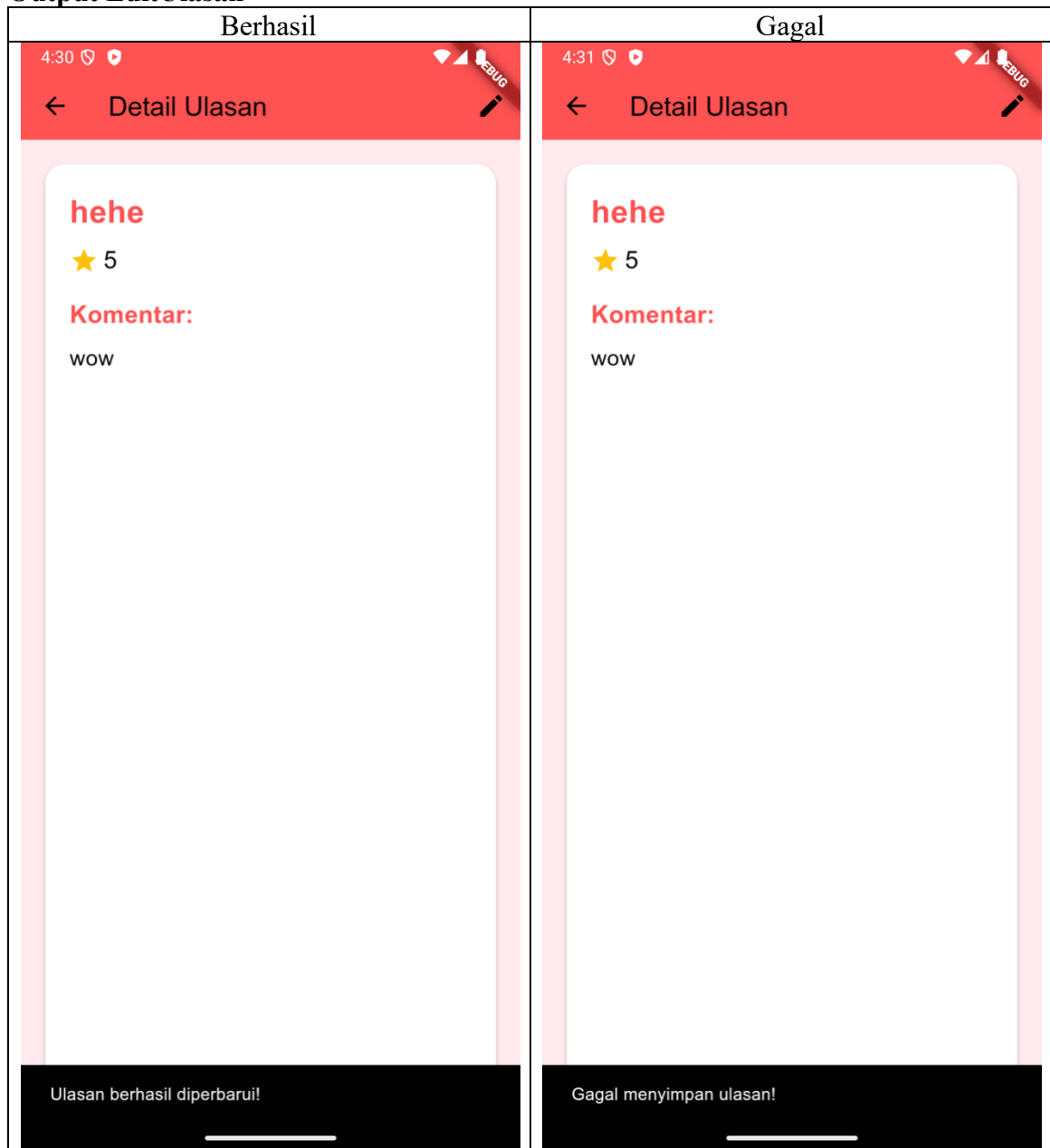
```

Halaman ini terdiri dari sebuah form yang memungkinkan pengguna memasukkan nama reviewer, rating, dan komentar. Jika halaman ini dibuka untuk mengedit ulasan, data ulasan yang ada akan diisi pada form, sedangkan jika membuat ulasan baru, form akan kosong. Fungsi `_saveUlasan` bertanggung jawab untuk memvalidasi form, lalu memutuskan apakah akan memanggil API untuk membuat ulasan baru atau memperbarui ulasan yang sudah ada melalui `ApiService`. Jika operasi berhasil, halaman akan menutup dan mengembalikan data ulasan yang telah disimpan. Jika gagal, pesan kesalahan akan ditampilkan menggunakan `ScaffoldMessenger`.

b. Output CreateUlasan

Berhasil	Gagal
	

c. **Output EditUlasan**



6. **Ulasan (DetailUlasan)**

a. **Source: lib/pages/DetailUlasan.dart**

```
import 'package:flutter/material.dart';
import 'package:manajemen_pariwisata/models/UlasanModel.dart';
import 'package:manajemen_pariwisata/pages/CreateUlasanPage.dart';

class DetailUlasanPage extends StatelessWidget {
  final UlasanModel ulasan;
```



```

DetailUlasanPage({required this.ulasan});

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Detail Ulasan', style: TextStyle(fontFamily: 'Arial')),
      backgroundColor: Colors.redAccent,
      actions: [
        IconButton(
          icon: Icon(Icons.edit),
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => CreateUlasanPage(ulasan: ulasan),
              ),
            );
          },
        ),
      ],
    ),
    body: Container(
      color: Colors.red[50],
      padding: const EdgeInsets.all(16.0),
      child: Card(
        elevation: 4,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(16),
        ),
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Text(
                ulasan.viewer,
                style: TextStyle(
                  fontSize: 26,
                  fontWeight: FontWeight.bold,
                  fontFamily: 'Arial',
                  color: Colors.redAccent,
                ),
              ),
              SizedBox(height: 8),
              Row(

```

```

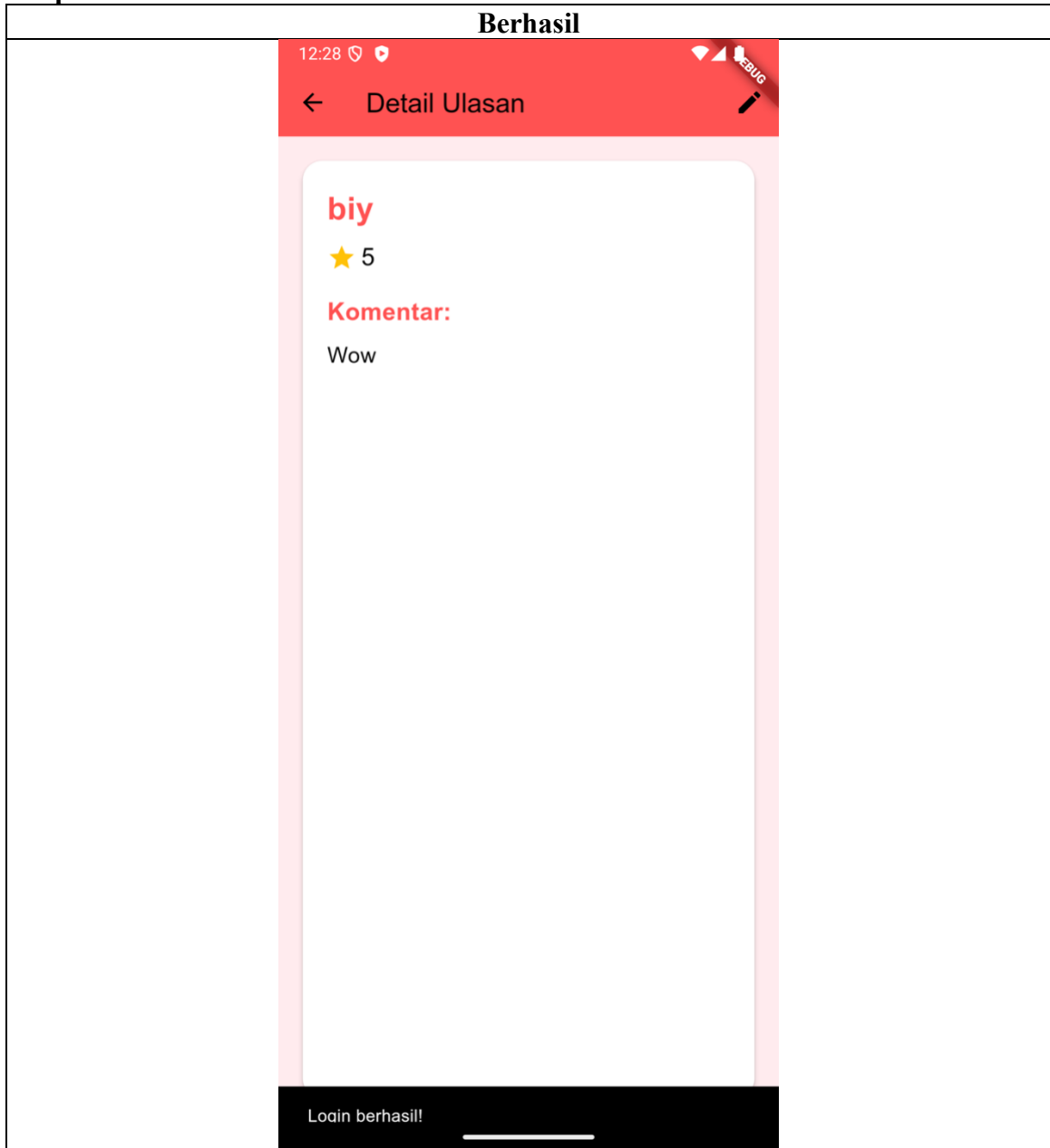
        children: [
          Icon(Icons.star, color: Colors.amber),
          SizedBox(width: 4),
          Text(
            '${ulasan.rating}',
            style: TextStyle(
              fontSize: 20,
              fontFamily: 'Arial',
              color: Colors.black,
            ),
          ),
        ],
      ),
      SizedBox(height: 16),
      Text(
        'Komentar:',
        style: TextStyle(
          fontSize: 20,
          fontWeight: FontWeight.bold,
          fontFamily: 'Arial',
          color: Colors.redAccent,
        ),
      ),
      SizedBox(height: 8),
      Text(
        ulasan.comments,
        style: TextStyle(
          fontSize: 18,
          fontFamily: 'Arial',
          color: Colors.black,
        ),
      ),
    ],
  ),
),
);
}

```

Kelas ini mengambil objek UlasanModel sebagai parameter dan menggunakan widget StatelessWidget. Dalam metode build, halaman ini terdiri dari AppBar yang berjudul "Detail Ulasan" dengan tombol edit di sebelah kanan, yang memungkinkan pengguna untuk mengedit ulasan yang ditampilkan. Bagian tubuh halaman berisi kontainer dengan latar belakang merah muda, yang menampilkan informasi ulasan dalam sebuah kartu.


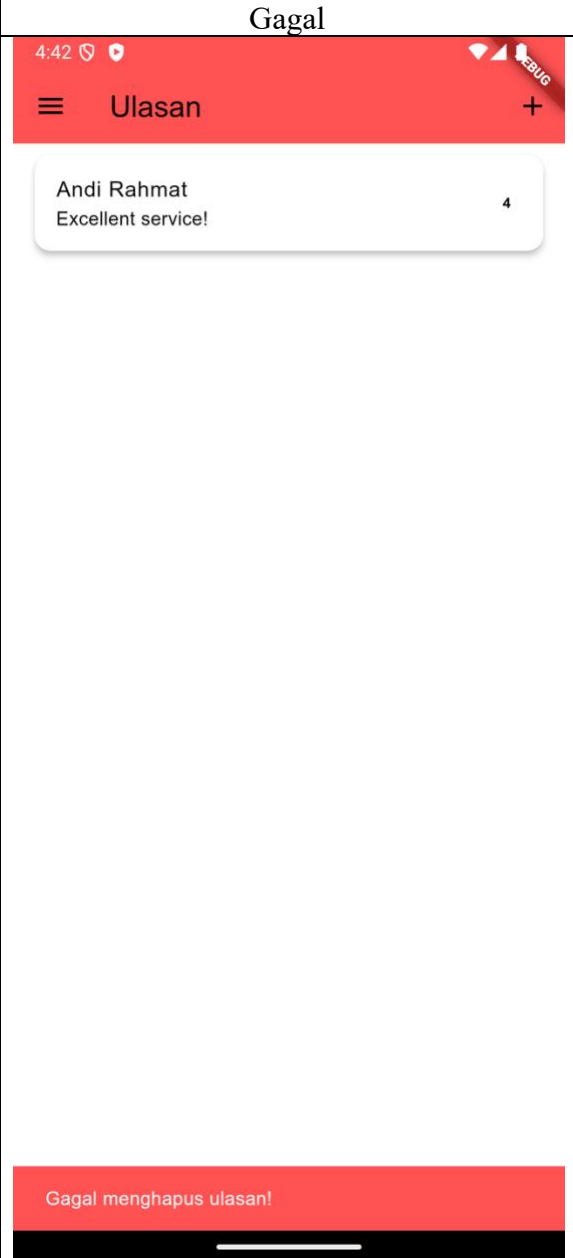
Informasi tersebut mencakup nama penulis ulasan (reviewer) dengan ukuran teks yang besar dan tebal, rating dalam bentuk bintang dan angka, serta komentar ulasan. Desain halaman menggunakan padding, ukuran, dan warna teks yang konsisten untuk menciptakan tampilan yang menarik dan mudah dibaca.

b. Output



7. Ulasan (Delete) – Untuk menghapusnya dengan cara menekan agak lama pada container ulasan.

a. Output

Berhasil	Gagal
 <p>The screenshot shows the 'Ulasan' screen with a red header bar. The status bar at the top shows the time 4:40. A review card for 'Andi Rahmat' with the text 'Excellent service!' and a rating of 4 is visible. At the bottom, a green toast message reads 'Ulasan berhasil dihapus!'.</p>	 <p>The screenshot shows the 'Ulasan' screen with a red header bar. The status bar at the top shows the time 4:42. A review card for 'Andi Rahmat' with the text 'Excellent service!' and a rating of 4 is visible. At the bottom, a red toast message reads 'Gagal menghapus ulasan!'.</p>