NAMA : AURA DEVANY SALSABILA BACHTIAR

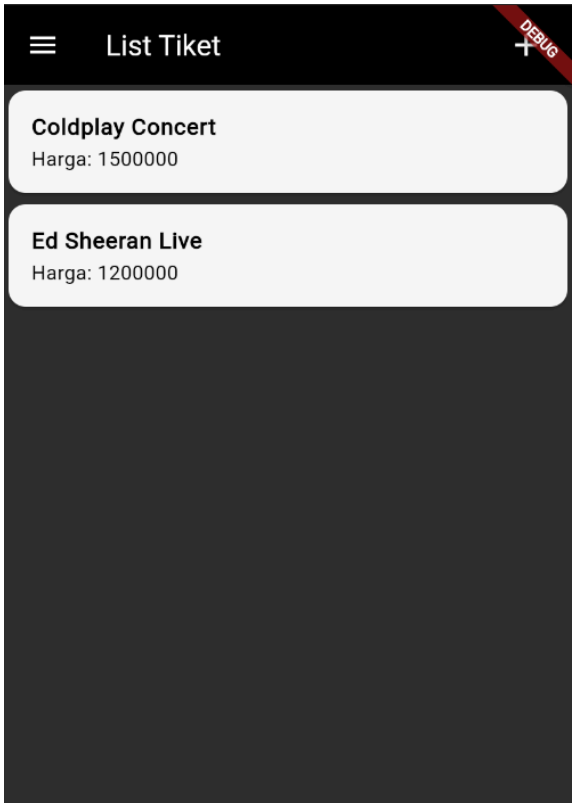
NIM : H1D022015

MATKUL : PRAKTIKUM PEMROGRAMAN MOBILE

RESPONSI 1

CRUD

1. Tampil Data List



a. Impor Package dan Kelas yang Dibutuhkan:

```
import 'package:flutter/material.dart';
import 'package:pariwisata/bloc/logout_bloc.dart';
import 'package:pariwisata/bloc/tiket_bloc.dart';
import 'package:pariwisata/model/hargatiket.dart';
import 'package:pariwisata/ui/login_page.dart';
import 'package:pariwisata/ui/hargatiket_detail.dart';
import 'package:pariwisata/ui/hargatiket_form.dart';
```

- Mengimpor package Material untuk menggunakan komponen UI Flutter.

- Mengimpor berbagai kelas dari aplikasi yang mencakup BLoC untuk logout (logout_bloc.dart) dan tiket (tiket_bloc.dart), model data HargaTiket, dan beberapa UI seperti halaman login, detail harga tiket, serta form tiket.

b. *Kelas HargaTiketPage dan State Management:*

```
class HargaTiketPage extends StatefulWidget {
  const HargaTiketPage({Key? key}) : super(key: key);

  @override
  _HargaTiketPageState createState() => _HargaTiketPageState();
}
```

- Kelas ini mendefinisikan halaman utama untuk menampilkan daftar harga tiket, dengan StatefulWidget agar dapat menangani perubahan UI.
- createState() digunakan untuk menciptakan state dari halaman ini.

c. *Tema Aplikasi dengan Warna dan Font:*

```
class _HargaTiketPageState extends State<HargaTiketPage> {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      theme: ThemeData(
        brightness: Brightness.light,
        primaryColor: Colors.black,
        colorScheme: ColorScheme.light(
              primary: Colors.black, secondary: Colors.grey), //
Use colorScheme
        scaffoldBackgroundColor: Colors.grey[850],
        fontFamily: 'Georgia',
        appBarTheme: const AppBarTheme(
          color: Colors.black,
          iconTheme: IconThemeData(color: Colors.white),
          titleTextStyle: TextStyle(
            // Use titleTextStyle instead of textTheme
            color: Colors.white,
            fontSize: 20,
            fontFamily: 'Georgia',
          ),
        ),
        textTheme: const TextTheme(
          bodyLarge: TextStyle(
                color: Colors.black, fontFamily: 'Georgia'), //
Update as needed
```

```
          bodyMedium: TextStyle(
                  color: Colors.black, fontFamily: 'Georgia'), //
Update as needed
        ),
        cardColor: Colors.white,
        dividerColor: Colors.grey, // Update to correct property
      ),
```

- ThemeData mengatur tema aplikasi. Warna dasar aplikasi adalah hitam (primaryColor: Colors.black) dan skema warna sekunder adalah abu-abu.
- Font yang digunakan di seluruh aplikasi adalah 'Georgia', termasuk pada teks di AppBar dan body teks.
- Background halaman (scaffoldBackgroundColor) diatur menjadi putih.

d. *AppBar dan Tombol Tambah Tiket:*

```
home: Scaffold(
        appBar: AppBar(
          title: const Text('List Tiket'),
          actions: [
            Padding(
              padding: const EdgeInsets.only(right: 20.0),
              child: GestureDetector(
                child: const Icon(Icons.add, size: 26.0),
                onTap: () async {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                        builder: (context) =>
                            const HargaTiketForm()), // Use const
                  );
                },
              ),
            )
          ],
        ),
```

- AppBar menampilkan teks 'List Tiket'.
- Ada tombol add di pojok kanan atas yang ketika ditekan, akan membuka halaman form untuk menambah tiket baru (HargaTiketForm).

e. *Drawer dan Logout:*

```
drawer: Drawer(
```

```
          child: ListView(
            children: [
              ListTile(
                title:
                     const Text('Logout', style: TextStyle(color:
Colors.black)),
                       trailing: const Icon(Icons.logout, color:
Colors.black),
                onTap: () async {
                  await LogoutBloc.logout().then((value) {
                    Navigator.of(context).pushAndRemoveUntil(
                      MaterialPageRoute(
                                      builder: (context) => const
LoginPage()), // Use const
                        (route) => false,
                    );
                  });
                },
              )
            ],
          ),
        ),
```

- Drawer menyediakan opsi untuk logout, menggunakan LogoutBloc untuk melakukan proses logout.
- Setelah logout, pengguna diarahkan ke halaman login (LoginPage).

f. Menampilkan Daftar Harga Tiket Menggunakan FutureBuilder:

```
body: FutureBuilder<List<HargaTiket>>(
          future: HargaTiketBloc.getHargaTiket(),
          builder: (context, snapshot) {
            if (snapshot.hasError) {
              print(snapshot.error);
            }
            return snapshot.hasData
                ? ListHargaTiket(
                    list: snapshot.data,
                  )
                : const Center(
                    child: CircularProgressIndicator(),
                  );
          },
        ),
      ),
```

```
    );
  }
}
```

- FutureBuilder digunakan untuk mengambil data harga tiket dari BLoC (HargaTiketBloc.getHargaTiket()).
- Jika data sudah ada (snapshot.hasData), data ditampilkan menggunakan widget ListHargaTiket.
- Jika data belum tersedia, akan muncul loading indicator berupa CircularProgressIndicator.

g. *Menampilkan List Harga Tiket dengan ListHargaTiket:*

```
class ListHargaTiket extends StatelessWidget {
  final List<HargaTiket>? list;

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

  @override
  Widget build(BuildContext context) {
    return ListView.builder(
      itemCount: list?.length ?? 0,
      itemBuilder: (context, i) {
        return ItemHargaTiket(
          hargaTiket: list![i],
        );
      },
    );
  }
}
```

- ListView.builder digunakan untuk membuat list harga tiket secara dinamis sesuai dengan jumlah data yang diambil dari API.
- ItemHargaTiket dipanggil untuk setiap item dalam list untuk menampilkan setiap detail harga tiket.

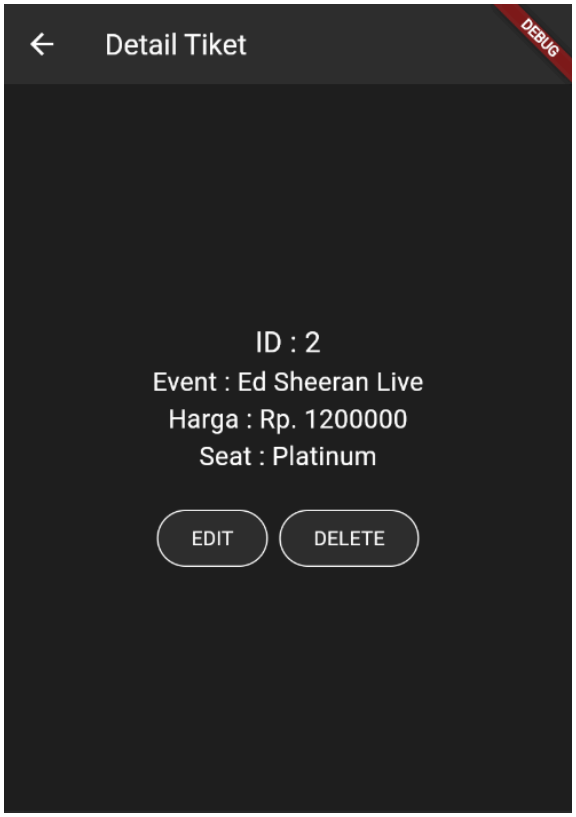
h. *Item Harga Tiket dengan Gesture untuk Navigasi Detail:*

```
class ItemHargaTiket extends StatelessWidget {
  final HargaTiket hargaTiket;

   const ItemHargaTiket({Key? key, required this.hargaTiket}) :
super(key: key);
```

```
  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => HargaTiketDetail(
              hargaTiket: hargaTiket,
            ),
          ),
        );
      },
      child: Card(
        color: Colors.grey[100],
        elevation: 2,
        child: ListTile(
          title: Text(
            hargaTiket.event!,
            style: const TextStyle(
                                fontFamily:  'Georgia',  fontWeight:
FontWeight.bold),
          ),
          subtitle: Text(
            'Harga: ${hargaTiket.price.toString()}',
            style: const TextStyle(fontFamily: 'Georgia'),
          ),
        ),
      ),
    );
  }
}
```

- ItemHargaTiket menampilkan setiap tiket dalam sebuah Card berwarna abu-abu terang (Colors.grey[100]).
- Ketika pengguna menekan item tersebut, akan diarahkan ke halaman detail tiket (HargaTiketDetail).
- Teks untuk event tiket menggunakan font Georgia dengan bold untuk judul event dan teks harga ditampilkan di subtitle.

2. Tampil Detail



a. Impor yang Dibutuhkan:

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

import 'package:pariwisata/bloc/tiket_bloc.dart';

import 'package:pariwisata/model/hargatiket.dart';

import 'package:pariwisata/ui/hargatiket_form.dart';

import 'package:pariwisata/ui/hargatiket_page.dart';

import 'package:pariwisata/widget/warning_dialog.dart';
```

- Impor berbagai paket yang dibutuhkan, termasuk bloc, model, form, dan widget lainnya seperti warning_dialog.

b. Deklarasi Widget HargaTiketDetail:

```
class HargaTiketDetail extends StatefulWidget {

  final HargaTiket? hargaTiket;



     const   HargaTiketDetail({Key?   key,   this.hargaTiket})   :
super(key: key);
```

```
  @override
              _HargaTiketDetailState       createState()       =>
_HargaTiketDetailState();
}
```

- Widget ini merupakan StatefulWidget yang digunakan untuk menampilkan detail harga tiket. Nilai hargaTiket diterima sebagai parameter.

c. *State dari Widget HargaTiketDetail:*

```
class _HargaTiketDetailState extends State<HargaTiketDetail> {
  @override
  Widget build(BuildContext context) {
```

- State digunakan untuk menangani perubahan dalam tampilan detail tiket.

d. *Scaffold & AppBar:*

```
 return Scaffold(
        backgroundColor: Colors.grey[900], // Latar belakang abu
gelap
      appBar: AppBar(
        title: const Text(
          'Detail Tiket',
                style: TextStyle(fontFamily: 'Georgia'), // Font
Georgia
        ),
        backgroundColor: Colors.grey[850], // AppBar abu gelap
      ),
```

- Scaffold adalah struktur dasar layout.
- AppBar memiliki warna latar belakang abu gelap (Colors.grey[850]), dan teks diatur menggunakan font Georgia, memberikan tampilan klasik dan profesional.

e. *Menampilkan Detail Tiket:*

```
body: Center(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text(
                    "ID : ${widget.hargaTiket?.id ?? 'N/A'}", //
Tampilkan ID
                style: const TextStyle(
                  fontSize: 20.0,
                  color: Colors.white,
                  fontFamily: 'Georgia',
                ),
              ),
              Text(
                  "Event : ${widget.hargaTiket?.event ?? 'N/A'}",
// Tampilkan nama event
                style: const TextStyle(
                  fontSize: 18.0,
                  color: Colors.white,
                  fontFamily: 'Georgia',
                ),
              ),
              Text(
                                                  "Harga   :   Rp.
${widget.hargaTiket?.price.toString()   ??   '0'}",   //   Tampilkan
harga
                style: const TextStyle(
```

```
                  fontSize: 18.0,
                  color: Colors.white,
                  fontFamily: 'Georgia',
                ),
              ),
              Text(
                "Seat : ${widget.hargaTiket?.seat ?? 'N/A'}", // Tampilkan seat
                style: const TextStyle(
                  fontSize: 18.0,
                  color: Colors.white,
                  fontFamily: 'Georgia',
                ),
              ),
              const SizedBox(height: 20.0), // Spasi antar elemen
              _buildEditDeleteButtons(), // Panggil widget tombol edit/hapus
            ],
          ),
        ),
      ),
    );
  }
```

- Data tiket seperti ID, nama event, harga, dan seat ditampilkan menggunakan widget Text.
- Setiap teks menggunakan warna putih (Colors.white) dan font Georgia. Jika data tidak tersedia (null), akan ditampilkan 'N/A'.
- SizedBox digunakan untuk memberi jarak antar elemen vertikal sebesar 20.0 pixel.

- Panggil widget _buildEditDeleteButtons() untuk menampilkan tombol Edit dan Hapus.

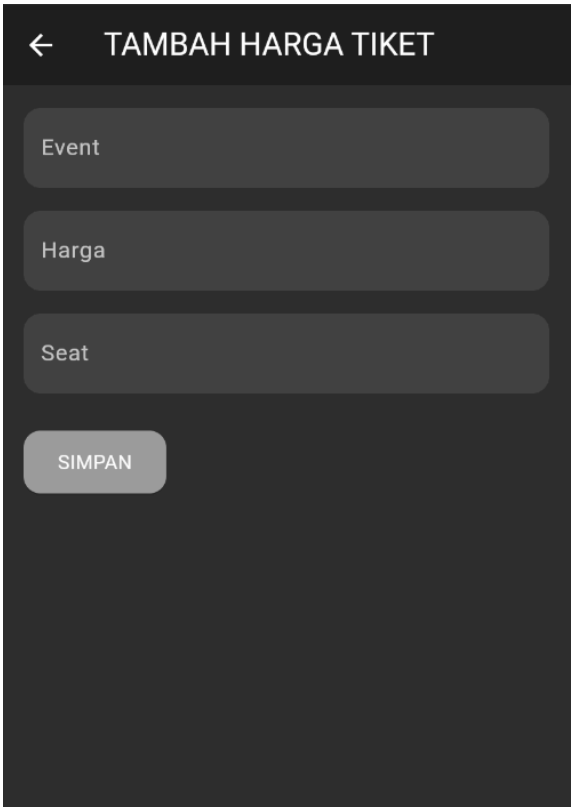
f. Tampilan Tombol Edit & Hapus:

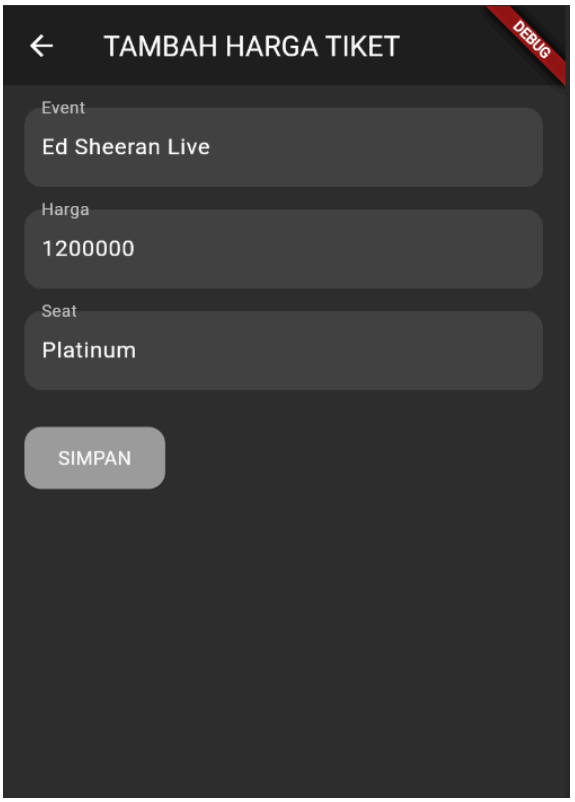
```
Widget _buildEditDeleteButtons() {
    return Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        // Tombol Edit
        OutlinedButton(
          style: OutlinedButton.styleFrom(
              backgroundColor: Colors.grey[850], // Warna tombol abu gelap
            side: const BorderSide(color: Colors.white), // Warna border putih
          ),
          child: const Text(
            "EDIT",
            style: TextStyle(
                color: Colors.white, fontFamily: 'Georgia'), // Font dan warna
          ),
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => HargaTiketForm(
                  hargaTiket: widget.hargaTiket!,
                ),
              ),
```

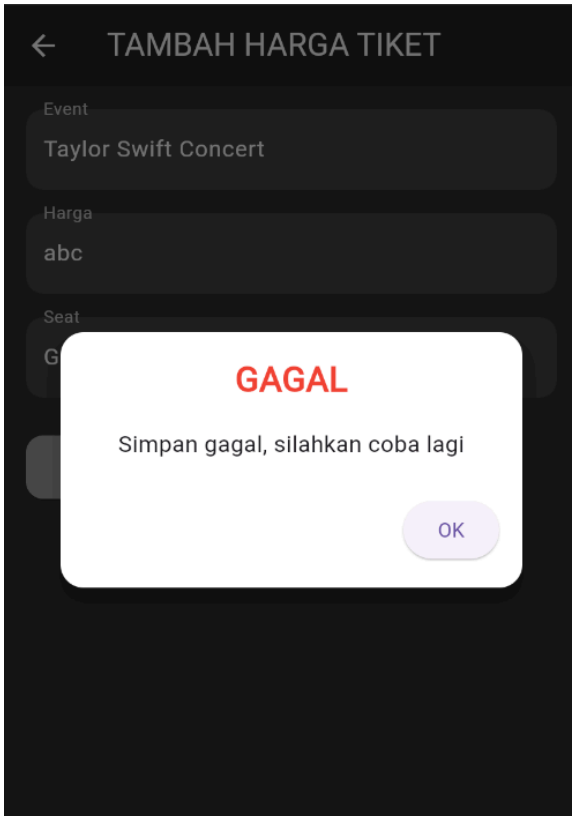
```
            );
          },
        ),
        const SizedBox(width: 8.0), // Spasi antar tombol
        // Tombol Hapus
        OutlinedButton(
          style: OutlinedButton.styleFrom(
              backgroundColor: Colors.grey[850], // Warna tombol
abu gelap
            side: const BorderSide(color: Colors.white), // Warna
border putih
          ),
          child: const Text(
            "DELETE",
            style: TextStyle(
                color: Colors.white, fontFamily: 'Georgia'), //
Font dan warna
          ),
          onPressed: confirmDelete,
        ),
      ],
    );
  }
```

- Tombol EDIT dan DELETE ditampilkan sebagai OutlinedButton dengan latar belakang abu gelap (Colors.grey[850]) dan border putih. Font yang digunakan tetap Georgia, dan teks tombol berwarna putih.

3. Tampil Tambah







a. Import dan Deklarasi Kelas

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

import 'package:pariwisata/bloc/tiket_bloc.dart';
```

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

import 'package:pariwisata/bloc/tiket_bloc.dart';

import 'package:pariwisata/model/hargatiket.dart';

import 'package:pariwisata/ui/hargatiket_page.dart';

import 'package:pariwisata/widget/warning_dialog.dart';


class HargaTiketForm extends StatefulWidget {

  final HargaTiket? hargaTiket;


   const HargaTiketForm({Key? key, this.hargaTiket}) : super(key:
key);


  @override

  _HargaTiketFormState createState() => _HargaTiketFormState();

}
```

- Import: Mengimpor paket-paket yang diperlukan untuk membangun antarmuka pengguna, termasuk material design, blok logika bisnis (BLoC) untuk manajemen data tiket, model data tiket, dan widget dialog peringatan.
- HargaTiketForm: Kelas ini merupakan widget yang memiliki status (stateful) untuk menampilkan form input harga tiket. Menerima parameter opsional hargaTiket untuk menampilkan informasi tiket yang ada.
- Mengembalikan instance dari _HargaTiketFormState, yang bertanggung jawab untuk mengelola logika dan state dari widget.

b. Inisialisasi State

```dart
class _HargaTiketFormState extends State<HargaTiketForm> {

  final _formKey = GlobalKey<FormState>();

  bool _isLoading = false;

  late String judul;

  late String tombolSubmit;
```

```
  final _eventTextboxController = TextEditingController();

  final _hargaTextboxController = TextEditingController();

  final _seatTextboxController = TextEditingController();



  @override

  void initState() {

    super.initState();

    _initializeForm();

  }
```

- State Variables: Mencakup kunci form, status loading, judul form, label tombol submit, dan controller untuk mengelola input pengguna.
- initState: Metode ini dipanggil saat widget diinisialisasi, memanggil _initializeForm() untuk menyiapkan form berdasarkan apakah hargaTiket ada atau tidak.

c. *Inisialisasi Form*

```
void _initializeForm() {

    if (widget.hargaTiket != null) {

      setState(() {

        judul = "UBAH HARGA TIKET";

        tombolSubmit = "UBAH";

          _eventTextboxController.text = widget.hargaTiket!.event
?? '';

        _hargaTextboxController.text =

            widget.hargaTiket!.price?.toString() ?? '';

         _seatTextboxController.text = widget.hargaTiket!.seat ??
'';

      });

    } else {

      judul = "TAMBAH HARGA TIKET";
```

```
      tombolSubmit = "SIMPAN";
    }
  }
```

- _initializeForm: Mengatur judul dan label tombol sesuai dengan status form (ubah atau tambah) dan mengisi controller dengan data yang ada jika hargaTiket tidak null.

d. Build Method

```
@override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[850],
      appBar: AppBar(
        leading: IconButton(
          icon: Icon(Icons.arrow_back,
                  color: Colors.white), // Ikon leading berwarna
putih
          onPressed: () {
                  Navigator.pop(context); // Kembali ke halaman
sebelumnya
          },
        ),
        title: Text(
          judul,
          style: TextStyle(
                  color: Colors.white, fontFamily: 'Georgia'), //
Apply Georgia font
        ),
        backgroundColor:
              Colors.grey[900], // Using a vibrant color for the
AppBar
```

```
      ),
      body: Container(
        // Dark background for the entire body
        padding: const EdgeInsets.all(16.0),
        child: SingleChildScrollView(
          child: Form(
            key: _formKey,
            child: Column(
              crossAxisAlignment:
                    CrossAxisAlignment.start, // Align form items
to the start
              children: [
                _eventTextField(),
                const SizedBox(height: 16), // Added spacing
                _hargaTextField(),
                const SizedBox(height: 16), // Added spacing
                _seatTextField(),
                const SizedBox(height: 24), // Added spacing
                _buttonSubmit(),
              ],
            ),
          ),
        ),
      ),
    );
  }
```

- build: Metode yang membangun antarmuka pengguna. Menggunakan Scaffold untuk menyediakan struktur dasar, dengan AppBar yang

menampilkan judul dengan font Georgia dan latar belakang berwarna abu-abu gelap.

- Container: Menyediakan latar belakang gelap untuk seluruh tubuh halaman dengan padding.
- SingleChildScrollView: Memungkinkan scrolling ketika konten melebihi ukuran layar.
- Form: Digunakan untuk mengelola input pengguna dan validasi.
- Column: Menyusun widget secara vertikal dengan ruang tambahan menggunakan SizedBox.

e. Field Input

```
Widget _eventTextField() {
    return TextFormField(
      decoration: InputDecoration(
        labelText: "Event",
        labelStyle: TextStyle(
            color: Colors.white70,
            fontFamily: 'Georgia'), // Lighter label color
        fillColor: Colors.grey[800], // Input background color
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none, // No border
        ),
      ),
      style:
          TextStyle(color: Colors.white, fontFamily: 'Georgia'),
// Text color
      keyboardType: TextInputType.text,
      controller: _eventTextboxController,
```

```
      validator: (value) => value!.isEmpty ? "Event harus diisi"
: null,
    );
  }

  Widget _hargaTextField() {
    return TextFormField(
      decoration: InputDecoration(
        labelText: "Harga",
         labelStyle: TextStyle(color: Colors.white70, fontFamily:
'Georgia'),
        fillColor: Colors.grey[800],
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none,
        ),
      ),
            style:  TextStyle(color:  Colors.white,  fontFamily:
'Georgia'),
      keyboardType: TextInputType.number,
      controller: _hargaTextboxController,
       validator: (value) => value!.isEmpty ? "Harga harus diisi"
: null,
    );
  }

  Widget _seatTextField() {
    return TextFormField(
```

```
      decoration: InputDecoration(
        labelText: "Seat",
         labelStyle: TextStyle(color: Colors.white70, fontFamily:
'Georgia'),
        fillColor: Colors.grey[800],
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none,
        ),
      ),
            style:  TextStyle(color:  Colors.white,  fontFamily:
'Georgia'),
      keyboardType: TextInputType.text,
      controller: _seatTextboxController,
      validator: (value) => value!.isEmpty ? "Seat harus diisi" :
null,
    );
  }
```

Event Test Field

- TextFormField: Widget untuk input teks dengan pengaturan dekorasi.
- InputDecoration: Mengatur label, warna latar belakang, dan batas.
- Label Style: Menggunakan warna putih pudar dan font Georgia untuk label.
- Controller: Menghubungkan field dengan _eventTextboxController untuk mengelola input.
- Validator: Memastikan input tidak kosong.

Harga Test Field

- Mirip dengan _eventTextField tetapi untuk input harga, mengatur tipe keyboard menjadi angka dan menggunakan validator yang sama.

Seat Test Field

- Fungsi yang sama dengan perbedaan pada label dan controller untuk seat.

f. *Build*

```
Widget _buttonSubmit() {
    return ElevatedButton(
      child: Text(
        tombolSubmit,
        style: TextStyle(
            color: Colors.white,
            fontFamily: 'Georgia'), // Text color for button
      ),
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.grey, // Button background color
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
        ),
        padding: const EdgeInsets.symmetric(
                vertical: 12.0, horizontal: 24.0), // Increased
button padding
      ),
      onPressed: _isLoading ? null : _onSubmit,
    );
  }
```

- ElevatedButton: Tombol yang ditampilkan untuk mengirim form.
- Text Style: Menggunakan warna putih dan font Georgia untuk teks tombol.
- Style: Mengatur warna latar belakang tombol dan bentuk dengan radius sudut yang ditingkatkan.

- OnPressed: Memanggil _onSubmit ketika tombol ditekan, kecuali saat loading.

g. *Submit Logic*

```
void _onSubmit() {
    if (_formKey.currentState!.validate()) {
      _isLoading = true;
      setState(() {});

      if (widget.hargaTiket != null) {
        _ubah();
      } else {
        _simpan();
      }
    }
  }

  void _simpan() {
    final price = int.tryParse(_hargaTextboxController.text) ??
        0; // Safely parse price as int
    final createHargaTiket = HargaTiket(
      id: null,
      event: _eventTextboxController.text,
      price: price, // Make sure this is an int
      seat: _seatTextboxController.text,
    );

                        HargaTiketBloc.addHargaTiket(hargaTiket:
createHargaTiket).then(
```

```
      (value) {
        _navigateToHargaTiketPage();
      },
      onError: (error) {
        _showWarningDialog("Simpan gagal, silahkan coba lagi");
      },
    ).whenComplete(() {
      _isLoading = false;
      setState(() {});
    });
  }
void _navigateToHargaTiketPage() {
    Navigator.of(context).pushReplacement(
      MaterialPageRoute(
                    builder: (BuildContext context) => const
HargaTiketPage()),
    );
  }
```

- _onSubmit: Memeriksa validasi form dan menjalankan fungsi simpan atau ubah berdasarkan ada tidaknya hargaTiket.
- _simpan: Mengumpulkan data dari field input, membuat instance baru HargaTiket, dan menambahkan data baru ke BLoC.

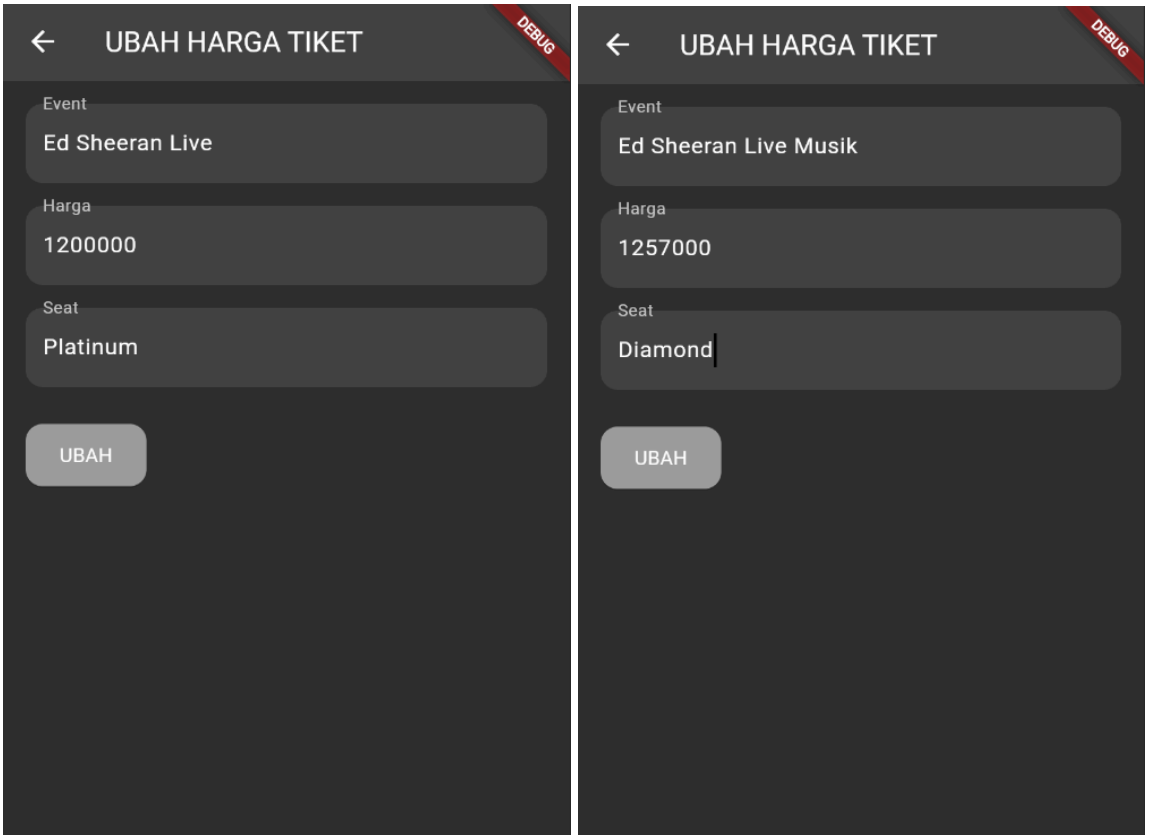
h. Dialog Peringatan

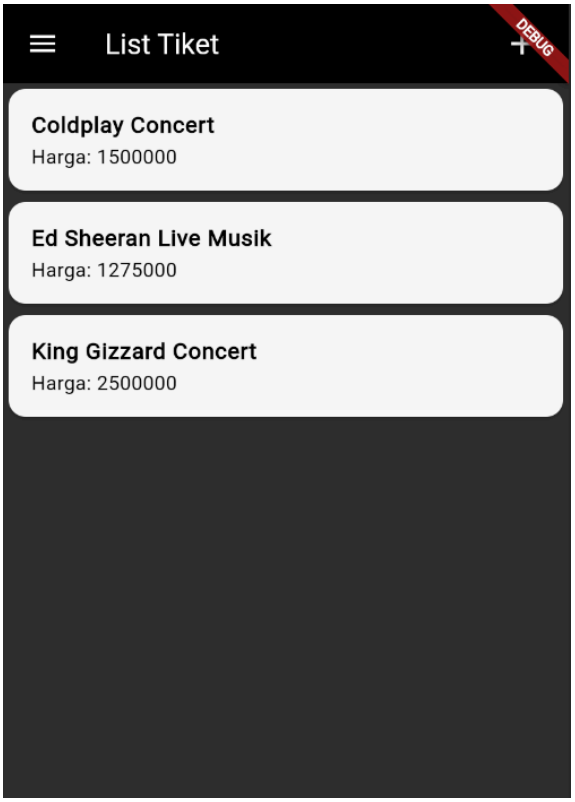
```
void _showWarningDialog(String message) {
    showDialog(
      context: context,
                      builder: (BuildContext context) =>
WarningDialog(description: message),
    );
```

```
  }

}
```

- _showWarningDialog: Menampilkan dialog peringatan setelah menyimpan atau mengubah data tiket.

4. Tampil Ubah





a. _initializeForm Method

```
void _ubah() {

    final price =
```

```
            int.tryParse(_hargaTextboxController.text) ?? 0; // Parsing price safely
    final updateHargaTiket = HargaTiket(
      id: widget.hargaTiket!.id!,
      event: _eventTextboxController.text,
      price: price,
      seat: _seatTextboxController.text,
    );

                    HargaTiketBloc.updateHargaTiket(hargaTiket: updateHargaTiket).then(
      (value) {
        _navigateToHargaTiketPage();
      },
      onError: (error) {
         _showWarningDialog("Permintaan ubah data gagal, silahkan coba lagi");
      },
    ).whenComplete(() {
      _isLoading = false;
      setState(() {});
    });
  }

  void _navigateToHargaTiketPage() {
    Navigator.of(context).pushReplacement(
      MaterialPageRoute(
                    builder: (BuildContext context) => const HargaTiketPage()),
```

```
    );
  }


  void _showWarningDialog(String message) {
    showDialog(
      context: context,
                      builder:    (BuildContext    context)    =>
WarningDialog(description: message),
    );
  }
}
```

- _ubah: Sama dengan _simpan, tetapi mengupdate tiket yang ada dengan ID yang diterima dari hargaTiket.

b. *Field Input Methods*

```
Widget _eventTextField() {
    return TextFormField(
      decoration: InputDecoration(
        labelText: "Event",
        labelStyle: TextStyle(
            color: Colors.white70,
            fontFamily: 'Georgia'), // Lighter label color
        fillColor: Colors.grey[800], // Input background color
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none, // No border
        ),
      ),
```

```
      style:
          TextStyle(color: Colors.white, fontFamily: 'Georgia'),
// Text color
      keyboardType: TextInputType.text,
      controller: _eventTextboxController,
       validator: (value) => value!.isEmpty ? "Event harus diisi"
: null,
    );
  }

  Widget _hargaTextField() {
    return TextFormField(
      decoration: InputDecoration(
        labelText: "Harga",
         labelStyle: TextStyle(color: Colors.white70, fontFamily:
'Georgia'),
        fillColor: Colors.grey[800],
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none,
        ),
      ),
            style:  TextStyle(color:  Colors.white,  fontFamily:
'Georgia'),
      keyboardType: TextInputType.number,
      controller: _hargaTextboxController,
       validator: (value) => value!.isEmpty ? "Harga harus diisi"
: null,
```

```
    );
  }


  Widget _seatTextField() {
    return TextFormField(
      decoration: InputDecoration(
        labelText: "Seat",
         labelStyle: TextStyle(color: Colors.white70, fontFamily:
'Georgia'),
        fillColor: Colors.grey[800],
        filled: true,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12.0), // Increased
border radius
          borderSide: BorderSide.none,
        ),
      ),
            style:  TextStyle(color:  Colors.white,  fontFamily:
'Georgia'),
      keyboardType: TextInputType.text,
      controller: _seatTextboxController,
      validator: (value) => value!.isEmpty ? "Seat harus diisi" :
null,
    );
  }
```

Event Test Field

- TextFormField: Widget untuk input teks dengan pengaturan dekorasi.
- InputDecoration: Mengatur label, warna latar belakang, dan batas.
- Label Style: Menggunakan warna putih pudar dan font Georgia untuk label.

- Controller: Menghubungkan field dengan _eventTextboxController untuk mengelola input.
- Validator: Memastikan input tidak kosong.

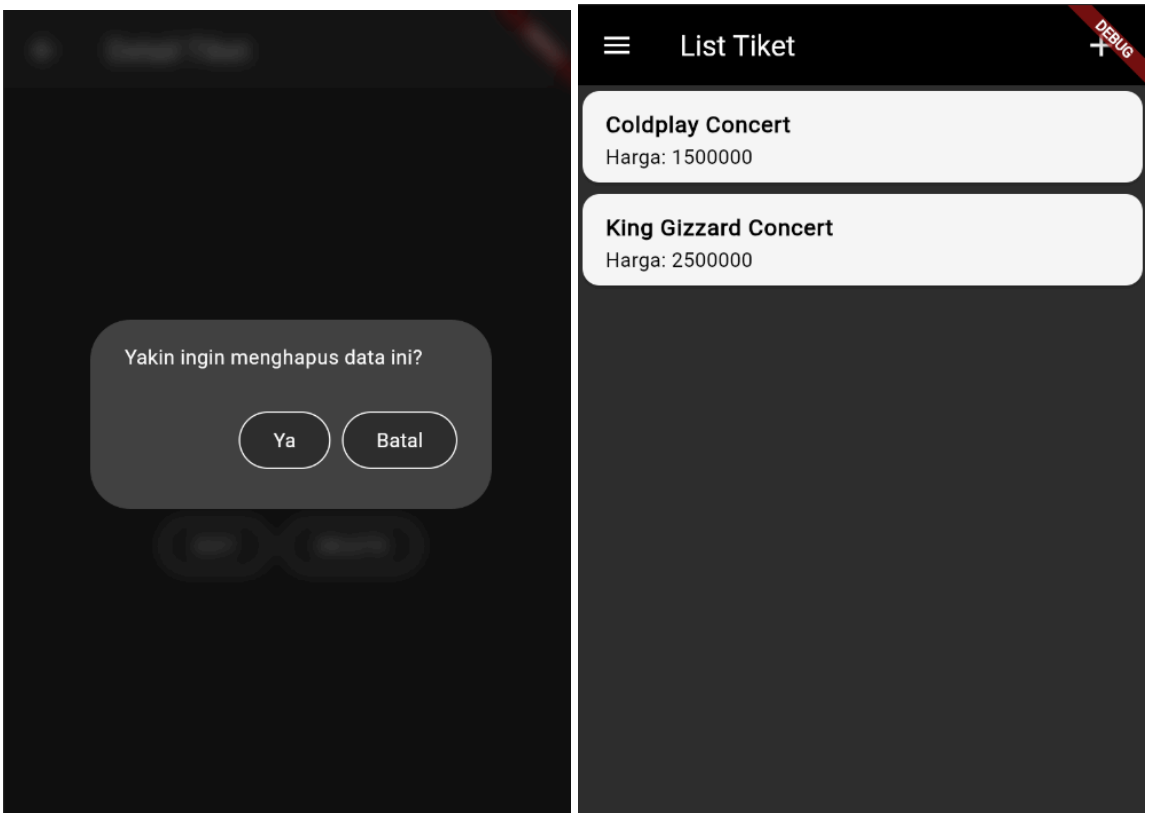
Harga Test Field

- Mirip dengan _eventTextField tetapi untuk input harga, mengatur tipe keyboard menjadi angka dan menggunakan validator yang sama.

Seat Test Field

- Fungsi yang sama dengan perbedaan pada label dan controller untuk seat.

5. Hapus Data



a. Dialog Konfirmasi Hapus:

```
void confirmDelete() {
    AlertDialog alertDialog = AlertDialog(
       backgroundColor: Colors.grey[800], // Warna latar belakang
dialog abu
      content: const Text(
```

```
        "Yakin ingin menghapus data ini?", // Pesan konfirmasi
        style: TextStyle(
            color: Colors.white, fontFamily: 'Georgia'), // Font
dan warna
      ),
      actions: [
        // Tombol Konfirmasi Hapus
        OutlinedButton(
          style: OutlinedButton.styleFrom(
              backgroundColor: Colors.grey[850], // Warna tombol
abu
                side: const BorderSide(color: Colors.white), //
Border putih
          ),
          child: const Text(
            "Ya",
            style: TextStyle(
                color: Colors.white, fontFamily: 'Georgia'), //
Font dan warna
          ),
          onPressed: () {
            final id = widget.hargaTiket?.id;
            if (id != null) {
              HargaTiketBloc.deleteHargaTiket(id: id).then(
                (value) {
                  Navigator.of(context).pushReplacement(
                    MaterialPageRoute(
                                builder: (context) => const
HargaTiketPage(),
                    ),
```

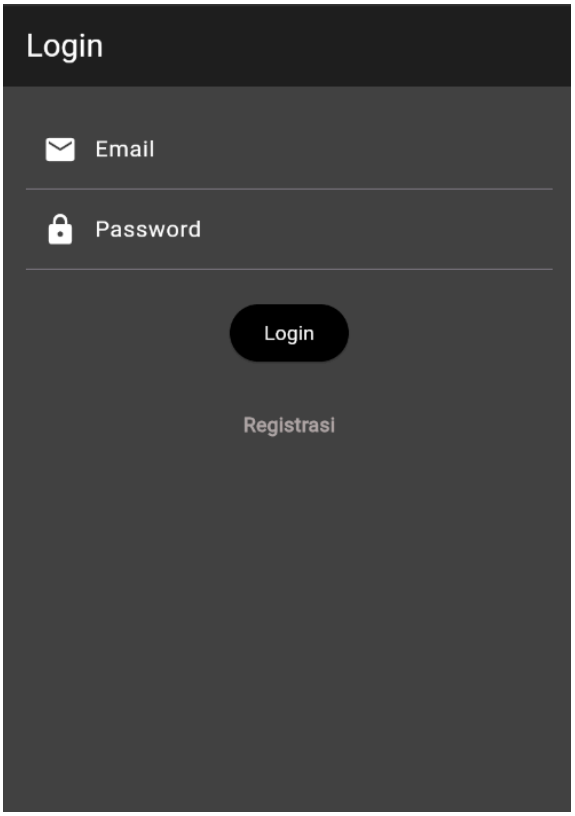
```
                  );
                },
                onError: (error) {
                  showDialog(
                    context: context,
                    builder: (context) => const WarningDialog(
                         description: "Hapus gagal, silahkan coba
lagi",
                    ),
                  );
                },
              );
            } else {
              showDialog(
                context: context,
                builder: (context) => const WarningDialog(
                  description: "ID tidak valid.",
                ),
              );
            }
          },
        ),
        // Tombol Batal
        OutlinedButton(
          style: OutlinedButton.styleFrom(
               backgroundColor: Colors.grey[850], // Warna tombol
abu
                side: const BorderSide(color: Colors.white), //
Border putih
```

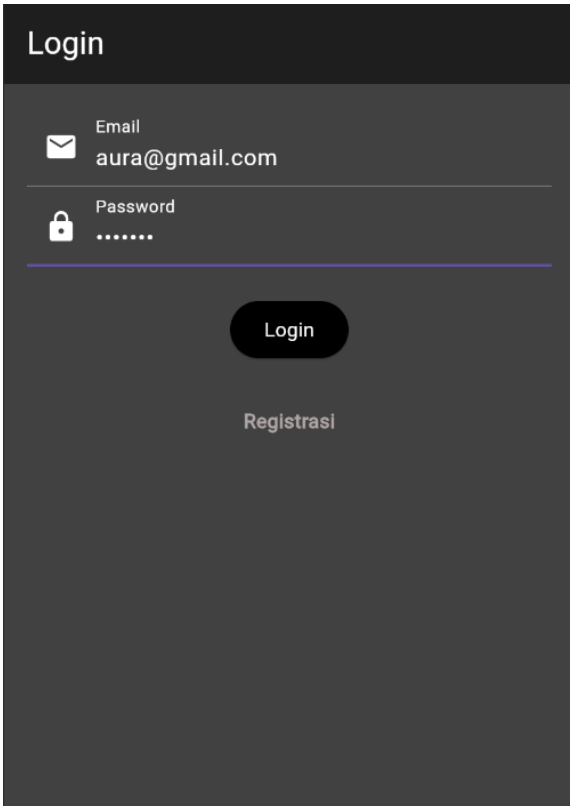
```
          ),
          child: const Text(
            "Batal",
            style: TextStyle(
                  color: Colors.white, fontFamily: 'Georgia'), //
Font dan warna
          ),
          onPressed: () => Navigator.pop(context),
        ),
      ],
    );

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

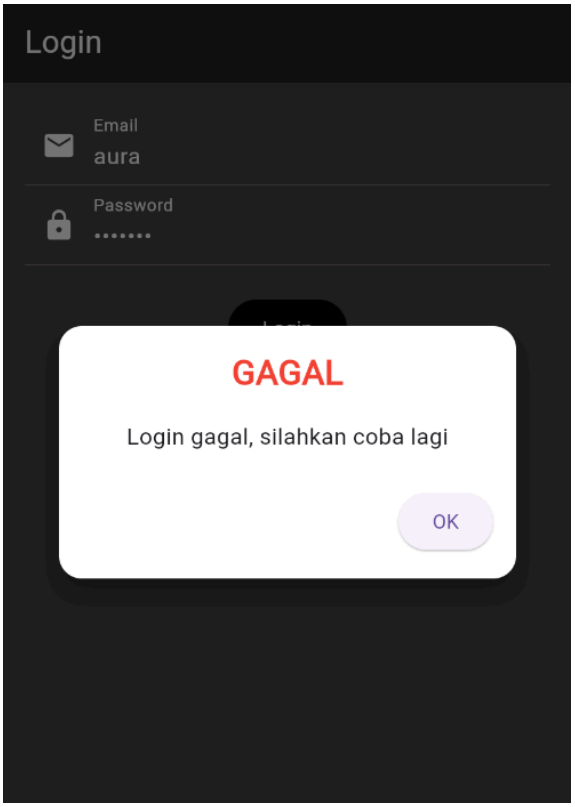
- Dialog konfirmasi menggunakan AlertDialog dengan latar belakang abu gelap (Colors.grey[800]) dan pesan menggunakan font Georgia berwarna putih.
- Jika pengguna menekan tombol "Ya", data akan dihapus menggunakan HargaTiketBloc.deleteHargaTiket. Jika terjadi error, dialog peringatan akan ditampilkan.

Login & Registrasi

1. Login







a. Import Libraries

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

import 'package:pariwisata/bloc/login_bloc.dart';

import 'package:pariwisata/helpers/user_info.dart';

import 'package:pariwisata/ui/hargatiket_page.dart';

import 'package:pariwisata/ui/registrasi_page.dart';

import 'package:pariwisata/widget/warning_dialog.dart';
```

- flutter/material.dart: Mengimpor library Flutter Material yang digunakan untuk membangun UI.
- pariwisata/bloc/login_bloc.dart: Mengimpor file yang mengelola logika login.
- pariwisata/helpers/user_info.dart: Mengimpor file yang berfungsi untuk menyimpan dan mengambil informasi pengguna.
- pariwisata/ui/hargatiket_page.dart: Mengimpor halaman untuk harga tiket.
- pariwisata/ui/registrasi_page.dart: Mengimpor halaman registrasi untuk pengguna baru.
- pariwisata/widget/warning_dialog.dart: Mengimpor widget dialog untuk menampilkan pesan peringatan.

b. *Kelas LoginPage*

```dart
class LoginPage extends StatefulWidget {

  const LoginPage({Key? key}) : super(key: key);



  @override

  _LoginPageState createState() => _LoginPageState();

}
```

- LoginPage: Kelas utama untuk halaman login, diturunkan dari StatefulWidget yang memungkinkan perubahan state.
- _LoginPageState: Kelas state untuk mengelola logika dan UI halaman login.

c. *State _LoginPageState*

```dart
class _LoginPageState extends State<LoginPage> {

  final _formKey = GlobalKey<FormState>();
```

```
  bool _isLoading = false;


  final _emailTextboxController = TextEditingController();
  final _passwordTextboxController = TextEditingController();
```

- _formKey: Kunci global untuk mengelola dan memvalidasi form.
- _isLoading: Variabel boolean untuk menampilkan status loading saat proses login.
- _emailTextboxController dan _passwordTextboxController: Kontroler untuk mengambil input dari pengguna.

d. Build Method

```
@override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[800],
      appBar: AppBar(
        title: const Text(
          'Login',
          style: TextStyle(
              color: Colors.white, fontFamily: 'Georgia'), // Use
Georgia font
        ),
        backgroundColor: Colors.grey[900],
      ),
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Form(
            key: _formKey,
            child: Column(
```

```
            children: [
              _emailTextField(),
              _passwordTextField(),
              const SizedBox(height: 20),
              _buttonLogin(),
              const SizedBox(height: 30),
              _menuRegistrasi(),
            ],
          ),
        ),
      ),
    ),
  );
}
```

- Scaffold: Struktur dasar untuk halaman, mencakup AppBar, body, dan tampilan umum.
- backgroundColor: Colors.grey[800]: Warna latar belakang untuk halaman login menggunakan warna abu-abu gelap.
- AppBar: Menampilkan judul halaman login dengan teks berwarna putih di atas latar belakang abu-abu lebih gelap.
- SingleChildScrollView: Memungkinkan tampilan untuk menggulir jika isi melebihi tinggi layar.
- Padding: Memberikan ruang di sekeliling form dengan padding 16.0 pixel.
- Form: Widget untuk menangani validasi input.
- Column: Menyusun widget dalam kolom, mencakup text field untuk email dan password, tombol login, dan menu registrasi.

e. *Text Field untuk Email*

```
Widget _emailTextField() {
  return TextFormField(
    decoration: const InputDecoration(
```

```
        labelText: "Email",
        labelStyle: TextStyle(
             color: Colors.white, fontFamily: 'Georgia'), // Use
Georgia font
        prefixIcon: Icon(Icons.email, color: Colors.white),
      ),
      keyboardType: TextInputType.emailAddress,
      controller: _emailTextboxController,
      style: const TextStyle(
            color: Colors.white, fontFamily: 'Georgia'), // Use
Georgia font
      validator: (value) {
        if (value!.isEmpty) {
          return 'Email harus diisi';
        }
        return null;
      },
    );
  }
```

- TextFormField: Widget input untuk email dengan fitur validasi.
- decoration: Menyediakan gaya untuk text field, termasuk label dan ikon.
- labelText: Teks label yang muncul di atas text field.
- labelStyle: Gaya label dengan teks berwarna putih.
- prefixIcon: Ikon email di dalam text field berwarna putih.
- keyboardType: Mengatur jenis keyboard untuk input email.
- controller: Menghubungkan kontroler untuk mengambil input pengguna.
- style: Menentukan warna teks input (putih).
- validator: Fungsi validasi untuk memastikan email tidak kosong.

f. Text Field untuk Password

```
Widget _passwordTextField() {
```

```
    return TextFormField(
      decoration: const InputDecoration(
        labelText: "Password",
        labelStyle: TextStyle(
             color: Colors.white, fontFamily: 'Georgia'), // Use
Georgia font
        prefixIcon: Icon(Icons.lock, color: Colors.white),
      ),
      keyboardType: TextInputType.text,
      obscureText: true,
      controller: _passwordTextboxController,
      style: const TextStyle(
            color: Colors.white, fontFamily: 'Georgia'), // Use
Georgia font
      validator: (value) {
        if (value!.isEmpty) {
          return "Password harus diisi";
        }
        return null;
      },
    );
  }
```

- Sama seperti _emailTextField, tetapi untuk password.
- obscureText: true: Menyembunyikan teks yang diketik di text field password.

g. *Tombol Login*

```
Widget _buttonLogin() {
    return ElevatedButton(
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.black,
```

```
        foregroundColor: Colors.white,
      ),
      child: const Text("Login",
                style: TextStyle(fontFamily: 'Georgia')), // Use
Georgia font
      onPressed: () {
        var validate = _formKey.currentState!.validate();
        if (validate) {
          if (!_isLoading) _submit();
        }
      },
    );
  }
```

- ElevatedButton: Tombol untuk mengirim form.
- backgroundColor: Warna latar belakang tombol (hitam).
- foregroundColor: Warna teks tombol (putih).
- onPressed: Fungsi yang dipanggil ketika tombol ditekan, melakukan validasi dan memanggil fungsi _submit.

h. *Fungsi _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()));
        Navigator.pushReplacement(context,
                  MaterialPageRoute(builder: (context) => const
HargaTiketPage()));
      } else {
        showDialog(
            context: context,
            barrierDismissible: false,
                      builder: (BuildContext context) => const
WarningDialog(
                  description: "Login gagal, silahkan coba lagi",
                ));
      }
    }, onError: (error) {
      showDialog(
          context: context,
          barrierDismissible: false,
          builder: (BuildContext context) => const WarningDialog(
                description: "Login gagal, silahkan coba lagi",
              ));
    });
    setState(() {
      _isLoading = false;
    });
  }
```

- _submit: Mengelola proses login.
- _formKey.currentState!.save(): Menyimpan state form.

- setState: Mengubah status loading saat login sedang diproses.
- LoginBloc.login(...): Memanggil metode login dengan email dan password dari kontroler.
- then: Mengelola hasil login, jika berhasil mengatur token dan user ID, kemudian menavigasi ke halaman HargaTiketPage.
- showDialog: Menampilkan dialog peringatan jika login gagal.

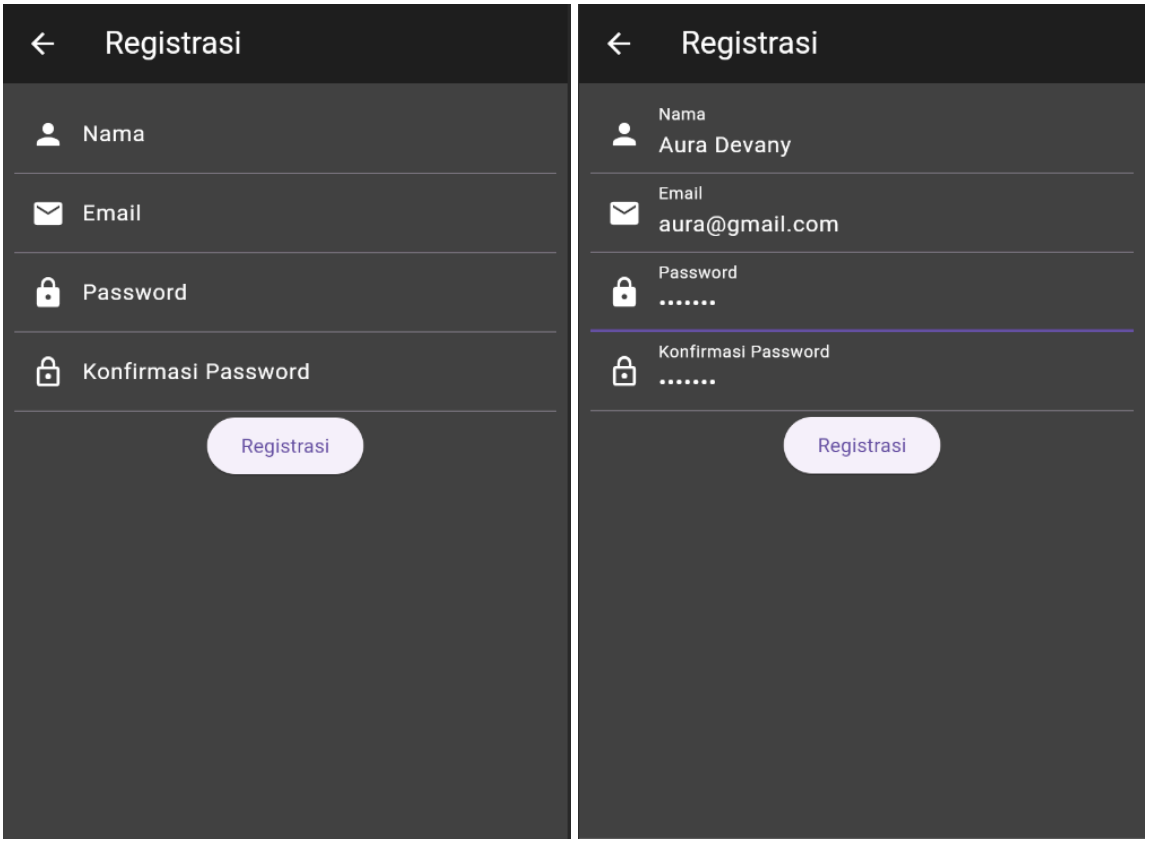
i. Menu Registrasi

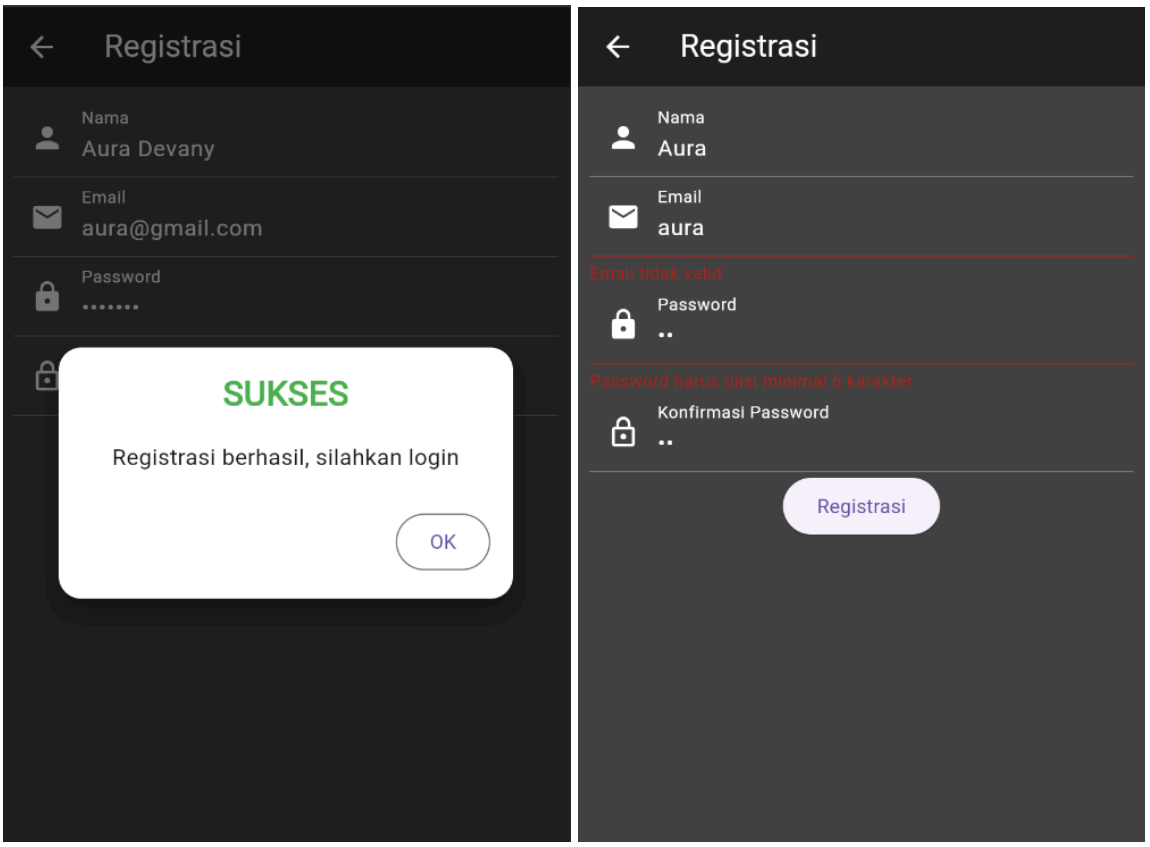
```
Widget _menuRegistrasi() {
    return Center(
      child: InkWell(
        child: const Text(
          "Registrasi",
          style: TextStyle(
              color: Color.fromARGB(255, 175, 167, 167),
              fontWeight: FontWeight.bold,
              fontFamily: 'Georgia'), // Use Georgia font
        ),
        onTap: () {
          Navigator.push(context,
                  MaterialPageRoute(builder: (context) => const
RegistrasiPage()));
        },
      ),
    );
  }
}
```

- Center: Mengatur posisi widget di tengah.
- InkWell: Membuat area yang dapat diketuk untuk navigasi ke halaman registrasi.
- Text: Teks untuk menu registrasi dengan gaya khusus.

- onTap: Menavigasi ke halaman registrasi saat teks diketuk.

2. Registrasi





a. Import Statements

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

```dart
import 'package:pariwisata/bloc/registrasi_bloc.dart';

import 'package:pariwisata/widget/success_dialog.dart';

import 'package:pariwisata/widget/warning_dialog.dart';
```

- import: Mengimpor paket yang diperlukan untuk aplikasi Flutter, termasuk widget dasar dari Flutter, BLoC untuk registrasi, dan dialog untuk menampilkan pesan sukses dan peringatan.

b. RegistrasiPage Class

```dart
class RegistrasiPage extends StatefulWidget {

  const RegistrasiPage({Key? key}) : super(key: key);



  @override

  _RegistrasiPageState createState() => _RegistrasiPageState();

}
```

- RegistrasiPage: Sebuah widget stateful yang memungkinkan pengguna untuk melakukan registrasi.
- createState: Mengembalikan instance dari _RegistrasiPageState yang mengelola state dari widget ini.

c. _RegistrasiPageState Class

```dart
class _RegistrasiPageState extends State<RegistrasiPage> {

  final _formKey = GlobalKey<FormState>();

  bool _isLoading = false;



  final _namaTextboxController = TextEditingController();

  final _emailTextboxController = TextEditingController();

  final _passwordTextboxController = TextEditingController();
```

- _formKey: Kunci global untuk mengidentifikasi form dan mengelola validasinya.
- _isLoading: Boolean untuk mengatur status loading saat registrasi.

- TextEditingController: Digunakan untuk mengelola input dari field teks untuk nama, email, dan password.

d. Build Method

```
@override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[800],
      appBar: AppBar(
        title: const Text(
          "Registrasi",
                      style:  TextStyle(fontFamily:  'Georgia',  color:
Colors.white),
        ),
        backgroundColor: Colors.grey[900],
        leading: IconButton(
          icon: const Icon(Icons.arrow_back,
              color: Colors.white), // Ikon panah kiri
          onPressed: () {
            Navigator.pop(context); // Kembali ke halaman sebelumnya
          },
        ),
      ),
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(8.0),
          child: Form(
            key: _formKey,
            child: Column(
```

```
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              _namaTextField(),
              _emailTextField(),
              _passwordTextField(),
              _passwordKonfirmasiTextField(),
              _buttonRegistrasi()
            ],
          ),
        ),
      ),
    ),
  );
 }
```

- Scaffold: Menyediakan struktur dasar untuk layout, termasuk AppBar dan body.
- AppBar: Menampilkan judul "Registrasi" dengan latar belakang hitam.
- SingleChildScrollView: Memungkinkan konten untuk di-scroll jika melebihi tinggi layar.
- Padding: Memberikan jarak sekitar form.
- Form: Mengelompokkan field input untuk registrasi dan menggunakan _formKey untuk validasi.
- Column: Mengatur field input dan tombol registrasi secara vertikal di tengah.

e. Field Input

```
//Membuat Textbox Nama
 Widget _namaTextField() {
   return TextFormField(
     decoration: const InputDecoration(
       labelText: "Nama",
```

```
        labelStyle: TextStyle(
            fontFamily: 'Georgia',
            color: Colors.white), // Warna label menjadi putih
         prefixIcon: Icon(Icons.person, color: Colors.white), // White
icon
      ),
      keyboardType: TextInputType.text,
      controller: _namaTextboxController,
      style: const TextStyle(
          fontFamily: 'Georgia',
          color: Colors.white), // Warna teks menjadi putih
      validator: (value) {
        if (value!.length < 3) {
          return "Nama harus diisi minimal 3 karakter";
        }
        return null;
      },
    );
  }


  //Membuat Textbox email
  Widget _emailTextField() {
    return TextFormField(
      decoration: const InputDecoration(
        labelText: "Email",
        labelStyle: TextStyle(
            fontFamily: 'Georgia',
            color: Colors.white), // Warna label menjadi putih
```

```
          prefixIcon: Icon(Icons.email, color: Colors.white), // White
icon

      ),

      keyboardType: TextInputType.emailAddress,

      controller: _emailTextboxController,

      style: const TextStyle(

          fontFamily: 'Georgia',

          color: Colors.white), // Warna teks menjadi putih

      validator: (value) {

        if (value!.isEmpty) {

          return "Email harus diisi";

        }

        Pattern pattern =


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,}))$';

        RegExp regex = RegExp(pattern.toString());

        if (!regex.hasMatch(value)) {

          return "Email tidak valid";

        }

        return null;

      },

    );

  }


  //Membuat Textbox password

  Widget _passwordTextField() {

    return TextFormField(
```

```
      decoration: const InputDecoration(
        labelText: "Password",
        labelStyle: TextStyle(
            fontFamily: 'Georgia',
            color: Colors.white), // Warna label menjadi putih
          prefixIcon: Icon(Icons.lock, color: Colors.white), // White
icon
      ),
      keyboardType: TextInputType.text,
      obscureText: true,
      controller: _passwordTextboxController,
      style: const TextStyle(
          fontFamily: 'Georgia',
          color: Colors.white), // Warna teks menjadi putih
      validator: (value) {
        if (value!.length < 6) {
          return "Password harus diisi minimal 6 karakter";
        }
        return null;
      },
    );
  }

  //Membuat Textbox Konfirmasi Password
  Widget _passwordKonfirmasiTextField() {
    return TextFormField(
      decoration: const InputDecoration(
        labelText: "Konfirmasi Password",
```

```
        labelStyle: TextStyle(
            fontFamily: 'Georgia',
            color: Colors.white), // Warna label menjadi putih
         prefixIcon: Icon(Icons.lock_outline, color: Colors.white), //
White icon
      ),
      keyboardType: TextInputType.text,
      obscureText: true,
      controller: _passwordTextboxController,
      style: const TextStyle(
          fontFamily: 'Georgia',
          color: Colors.white), // Warna teks menjadi putih
      validator: (value) {
        if (value != _passwordTextboxController.text) {
          return "Konfirmasi Password tidak sama";
        }
        return null;
      },
    );
  }
```

_namaTextField Method

- TextFormField: Widget untuk input teks dengan validasi.
- decoration: Menentukan label dan gaya label untuk field.
- labelStyle: Mengatur warna label menjadi Colors.white70, warna abu-abu terang.
- validator: Fungsi untuk memvalidasi input; jika nama kurang dari 3 karakter, akan memberikan pesan kesalahan.
- style: Mengatur warna teks menjadi putih dan font menjadi Georgia.

_emailTextField Method

- keyboardType: Menentukan jenis keyboard yang ditampilkan untuk email.
- validator: Memastikan email tidak kosong dan memenuhi format email yang valid dengan menggunakan ekspresi reguler.
- style: Sama dengan sebelumnya, menggunakan font Georgia untuk teks.

_passwordTextField Method

- obscureText: Mengatur agar karakter yang dimasukkan tidak terlihat (disembunyikan).
- validator: Memastikan password memiliki minimal 6 karakter.

_passwordKonfirmasiTextField Method

- validator: Memastikan konfirmasi password sama dengan password yang telah dimasukkan sebelumnya.

f. Button Registrasi

```
//Membuat Tombol Registrasi
  Widget _buttonRegistrasi() {
    return ElevatedButton(
      child: const Text("Registrasi"),
      onPressed: () {
        var validate = _formKey.currentState!.validate();
        if (validate) {
          if (!_isLoading) _submit();
        }
      },
    );
  }
```

- ElevatedButton: Tombol yang akan meng-trigger registrasi.
- onPressed: Fungsi yang dijalankan saat tombol ditekan. Melakukan validasi dan memanggil fungsi _submit() jika validasi berhasil.
- style: Mengatur warna latar belakang tombol menjadi hitam dan teks menjadi putih.

g. *Submit Method*

```
void _submit() {
    _formKey.currentState!.save();
    setState(() {
      _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;
    });
  }
}
```

- _submit(): Fungsi untuk memproses data registrasi.
- save(): Menyimpan state dari form.
- setState: Memperbarui UI menjadi loading saat proses registrasi berlangsung.
- RegistrasiBloc.registrasi: Memanggil metode registrasi dari BLoC dengan data dari field input.
- showDialog: Menampilkan dialog sukses atau peringatan berdasarkan hasil registrasi.
- setState: Mengatur status loading kembali ke false setelah proses selesai.