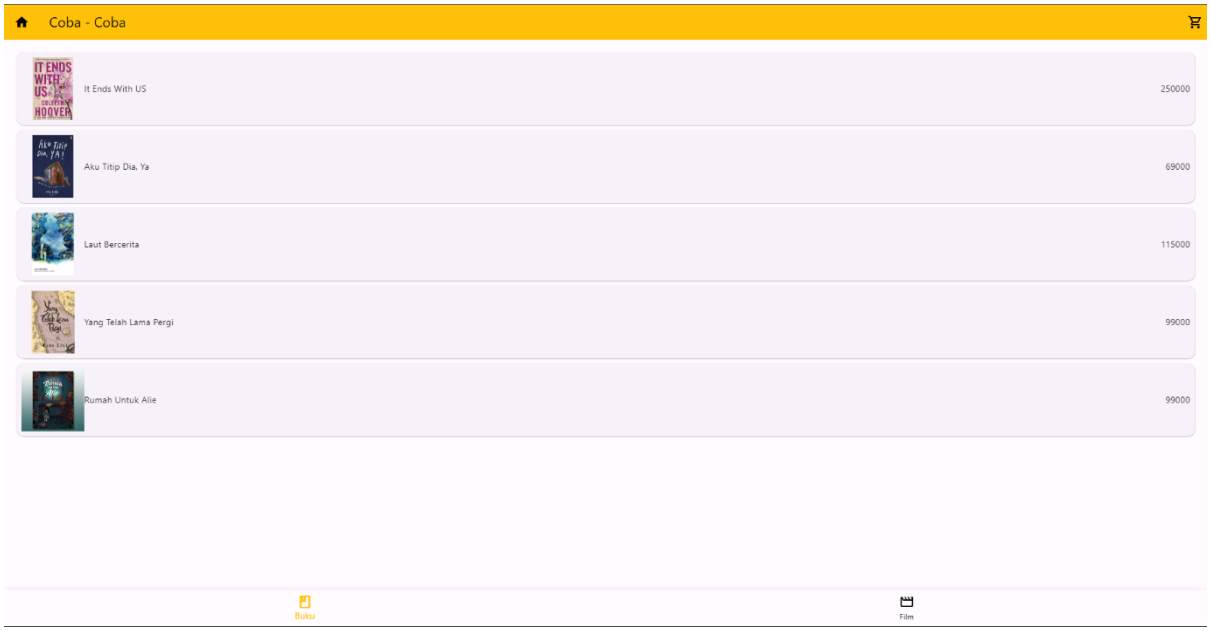
Nama : Ibni Andarta

Absen : 13s

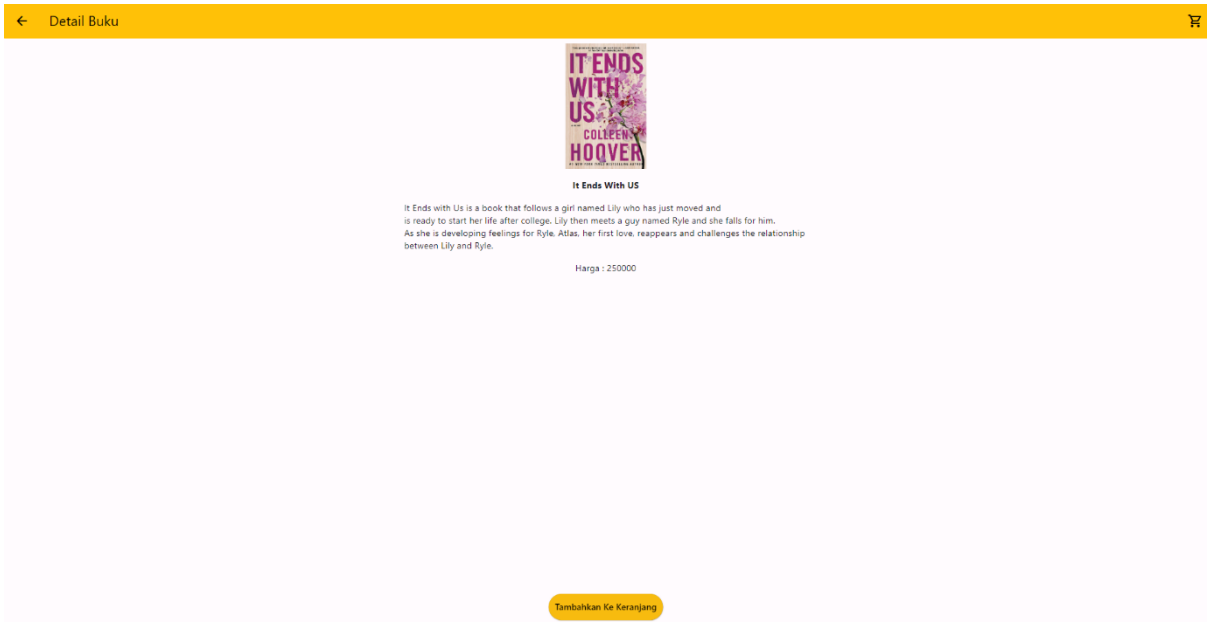
Kelas : XI RPL 4

Screenshoot hasil

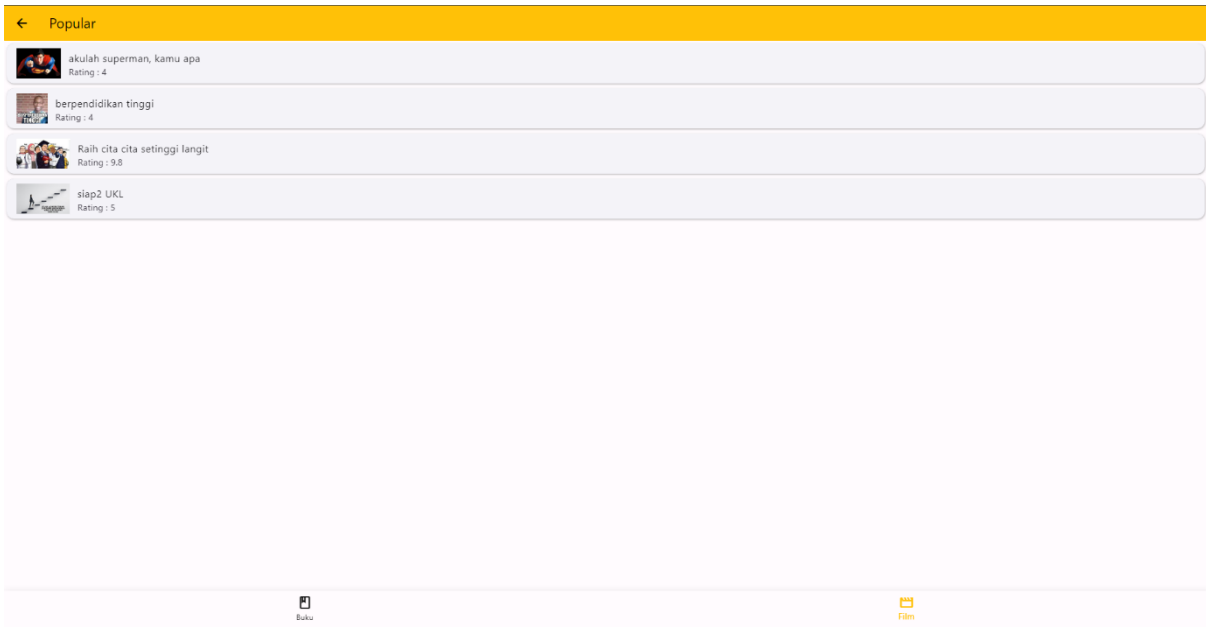
Layar pertama :



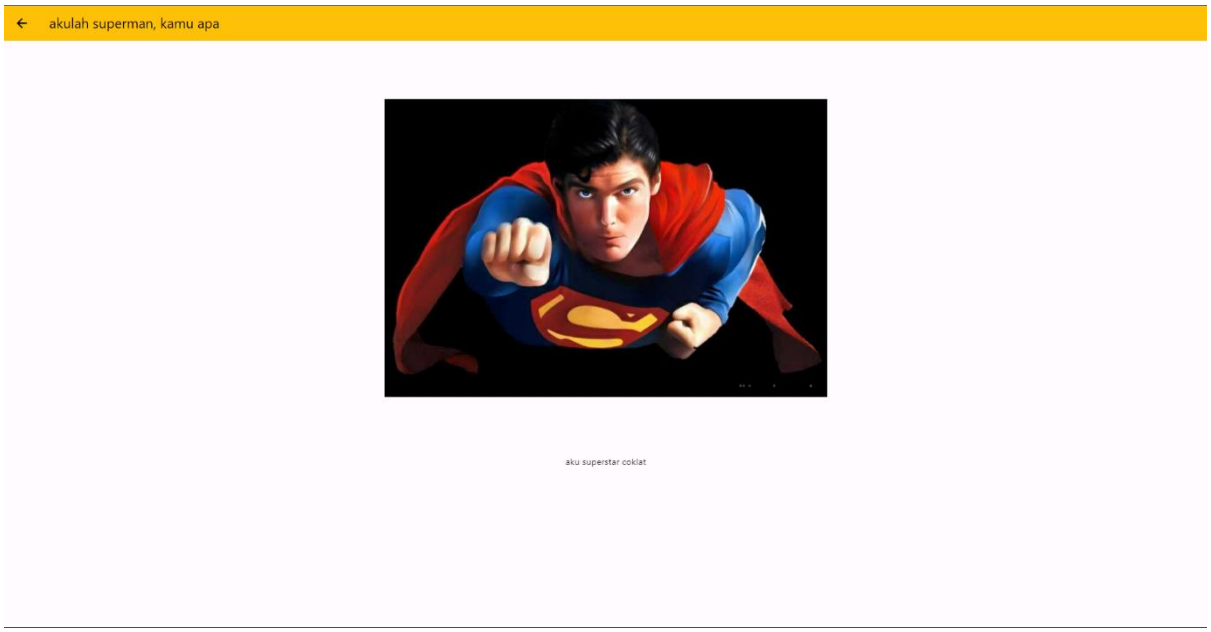
Layar kedua :



Layar ketiga :



Layar keempat :



Penjelasan code :

```
class Item {
  String name;
  String desc;
  String foto;
  int harga;
  int id;

  Item({required this.name, required this.desc, required this.foto, required this.harga, require

}
```

Code ini berisi class yang mendefinisikan nama, deskripsi, foto, harga, dan id pada layar pertama. Class ini akan di panggil Class Homepage (Layar pertama) nantinya untuk menyimpan data data dari layar pertama.

```
class _HomePageState extends State<HomePage> {
  final List<Item> items = [
    Item( name: 'It Ends With US',
          harga: 250000,
          desc: """It Ends with Us is a book that follows a girl named Lily who has just moved a
is ready to start her life after college. Lily then meets a guy named Ryle and she falls for him
As she is developing feelings for Ryle, Atlas, her first love, reappears and challenges the rela
between Lily and Ryle.""",
          foto: "https://cdn.gramedia.com/uploads/picture_meta/2023/5/26/zicekhdrq9sheivpqogtc3.
          id: 0
          ),
```

Ini adalah salah satu contoh penggunaan class Item di dalam class Homepage

```
return Scaffold(
  appBar: AppBar(
    title: Text("Coba - Coba"),
    backgroundColor: Colors.amber,
    foregroundColor: Colors.black,
    leading: IconButton(onPressed: (){
      Navigator.pushNamed(context, '/home');
    },
    icon: Icon(Icons.home)
    ), // IconButton
    actions: [
      IconButton(
        onPressed: (){
          Navigator.pushNamed(context, '/cart');
        },
        icon: Icon(Icons.shopping_cart_outlined)
      ) // IconButton
    ],
  ), // AppBar
    body: Container(
      margin: EdgeInsets.all(8),
      child: ListView.builder(
        padding: EdgeInsets.all(8),
        itemCount: items.length,
        itemBuilder: (context, index) {
          final item = items[index];
          return Card(
            child: InkWell(
              onTap: (){
                Navigator.pushNamed(context, "/item", arguments: item);
              },
              child: Container(
                margin: EdgeInsets.all(8),
                child: Row(
                  children: [
                    Image(image: NetworkImage(item.foto), height: 100, width: 100,),
                    Expanded(child: Text(item.name)),
                    Expanded(
                      child: Text(
                        item.harga.toString(),
                        textAlign: TextAlign.end,
                      ), // Text
                    ) // Expanded
                  ],
                ), // Row
```

Secara keseluruhan, kode ini mendefinisikan tampilan halaman dengan app bar, daftar item yang dapat scroll (seperti recyclerview dalam kotlin) , dan navigasi bar. Ketika ditekan, kita akan diarahkan ke halaman detail dari item tersebut.

```
body: Padding(
  padding: EdgeInsets.all(8),
  child: Column(
    children: [
      Image.network(itemArgs.foto, height: 200, width: 200),
      Padding(padding: EdgeInsets.all(8)),
      Text(
        itemArgs.name,
        style: TextStyle(fontWeight: FontWeight.bold),
      ), // Text
      Padding(padding: EdgeInsets.all(8)),
      Text(itemArgs.desc),
      Padding(padding: EdgeInsets.all(8)),
      Text("Harga : " + itemArgs.harga.toString()),
      Expanded(child: Container()),
      ElevatedButton(
        onPressed: () {
          cartProvider.addToCart(itemArgs); // Using the correct variable name
          ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
              content: Text("Item Berhasil Ditambahkan"),
            ), // SnackBar
          );
        },
        child: Text("Tambahkan Ke Keranjang"),
        style: ButtonStyle(
          backgroundColor: MaterialStateProperty.all(Colors.amber),
          foregroundColor: MaterialStateProperty.all(Colors.black),
          minimumSize: MaterialStateProperty.all(Size(150, 50)),
          padding: MaterialStateProperty.all(EdgeInsets.all(10)),
        ), // ButtonStyle
      ) // ElevatedButton
    ],
  ), // Column
), // Padding
```

Secara keseluruhan, kode ini menampilkan detail item dengan gambar, nama, deskripsi, dan harga dari class Item yang sudah di inisialisasikan pada class HomePage. ada tombol untuk menambahkan item ke keranjang belanja. Ketika tombol ini ditekan, item akan ditambahkan ke keranjang belanja dan aka ada pop up konfirmasi.

```
body: ListView.builder(
  itemCount: cartProvider.cartItems.length,
  itemBuilder: (context, index) {
    final cartItem = cartProvider.cartItems[index];
    return ListTile(
      title: Text(cartItem.barang.name),
      subtitle: Text(
        'Price: \RP. ${cartItem.barang.harga} - Quantity: ${cartItem.jumlah}',
      ), // Text
      trailing: IconButton(
        icon: Icon(Icons.remove_shopping_cart),
        onPressed: () {
          cartProvider.removeFromCart(cartItem);
        },
      ), // IconButton
    ); // ListTile
  },
), // ListView.builder
bottomNavigationBar: BottomAppBar(
  child: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Text(
      'Total: \RP. ${cartProvider.getTotal().toStringAsFixed(2)}',
      style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
    ), // Text
  ), // Padding
), // BottomAppBar
```

Secara keseluruhan, kode ini menampilkan total harga dari setiap item yang ditambahkan ke dalam keranjang.

```
class CartProvider extends ChangeNotifier {
  List<CartItem> _cartItems = [];
  List<CartItem> get cartItems => _cartItems;

  void addToCart(Item barang) {
    for (CartItem item in _cartItems) {
      if (item.barang.id == barang.id) {
        item.jumlah++;
        notifyListeners();
        return;
      }
    }
    _cartItems.add(CartItem(barang: barang));
    notifyListeners();
  }

  void removeFromCart(CartItem item) {
    _cartItems.remove(item);
    notifyListeners();
  }

  double getTotal() {
    return _cartItems.fold(0, (total, item) => total + item.barang.harga * item.jumlah);
  }
}
```

Secara keseluruhan code dalam class addTocart ini berisi tentang perhitungan harga dari semua barang yang ditambahkan ke dalam keranjang.

```dart
class Film{
  int? id;
  String? title;
  double? voteAverage;
  String? overview;
  String? posterPath;

  Film(this.id, this.title, this.voteAverage, this.overview, this.posterPath);

  Film.fromJson(Map<String, dynamic> parsedJson){
    id = parsedJson['id'];
    title = parsedJson['title'];
    voteAverage = parsedJson['voteaverage']*1.0;
    overview = parsedJson['overview'];
    posterPath = parsedJson['posterpath'];
  }
}
```

Secara keseluruhan, kelas Film ini dapat digunakan untuk merepresentasikan sebuah film dengan properti id, title, voteAverage, overview, dan posterPath. Kelas ini juga menyediakan fungsi untuk membuat instance Film dari data JSON.

```dart
  Future initialize() async{
    film = [];
    film = (await service?.getPopularMovies());
    setState(() {
      moviesCount = film?.length;
      film = film;
    });
  }

  @override
  void initState(){
    service = HttpService();
    initialize();
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    service!.getPopularMovies().then((value) => {
      setState(() {
        result =(value != null) as String;
      })
    });
```

kode ini mendefinisikan widget yang menampilkan daftar film populer, memperbarui state berdasarkan data yang diambil dari layanan HTTP, dan menavigasi ke detail film ketika salah satu film di daftar diklik.

```dart
class HomePageFilm extends StatelessWidget {
  const HomePageFilm({super.key});

  @override
  Widget build(BuildContext context) {
    return const FilmList();
  }
}
```

Code diatas berisi tentang pengembalian class FilmList, jadi Class FlimList aja yang akan ditampilkan.

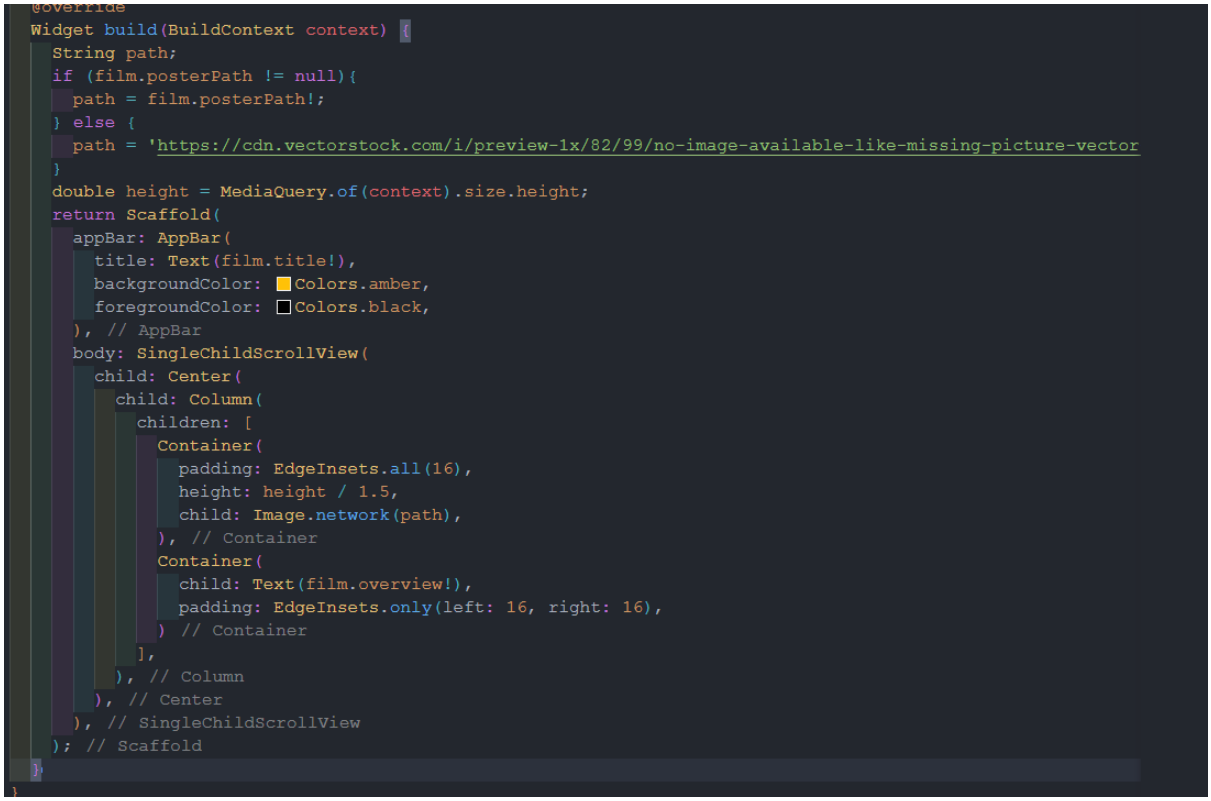
```dart
class HttpService {
  final String baseUrl = 'https://movie.tukanginyuk.com/api/getmovie';

  Future<List?> getPopularMovies() async {
    final String uri = baseUrl;

    http.Response result = await http.get(Uri.parse(uri));

    if(result.statusCode == HttpStatus.ok){
      print("Connected");
      final jsonResponse = json.decode(result.body);
      final moviesMap = jsonResponse['data'];
      List movie = moviesMap.map((i) => Film.fromJson(i)).toList();
      return movie;
    }else{
      print("Not Connected");
      return null;
    }

  }
}
```

Class HttpService: Class ini mendefinisikan sebuah layanan HTTP untuk berinteraksi dengan API.

baseUrl: Variabel ini menyimpan URL dasar API yang akan digunakan untuk permintaan HTTP.

getPopularMovies(): Metode asynchronous ini melakukan permintaan GET ke API dan mengembalikan daftar film jika berhasil. Jika permintaan gagal, akan mengembalikan null.

Secara keseluruhan, kelas ini digunakan untuk mengambil data film populer dari API dan mengubahnya menjadi daftar objek Film yang dapat digunakan dalam aplikasi.



```
@override
Widget build(BuildContext context) {
  String path;
  if (film.posterPath != null){
    path = film.posterPath!;
  } else {
    path = 'https://cdn.vectorstock.com/i/preview-1x/82/99/no-image-available-like-missing-picture-vector
  }
  double height = MediaQuery.of(context).size.height;
  return Scaffold(
    appBar: AppBar(
      title: Text(film.title!),
      backgroundColor: Colors.amber,
      foregroundColor: Colors.black,
    ), // AppBar
    body: SingleChildScrollView(
      child: Center(
        child: Column(
          children: [
            Container(
              padding: EdgeInsets.all(16),
              height: height / 1.5,
              child: Image.network(path),
            ), // Container
            Container(
              child: Text(film.overview!),
              padding: EdgeInsets.only(left: 16, right: 16),
            ) // Container
          ],
        ), // Column
      ), // Center
    ), // SingleChildScrollView
  ); // Scaffold
}
}
```

metode build ini mendefinisikan tampilan detail film dengan gambar dan sinopsis. Jika gambar film tidak tersedia, gambar “tidak tersedia” akan ditampilkan.

```dart
class _NavBarState extends State<NavBar> {
  int _currentIndex = 0;

  void changeSelectedNavBar(int index){
    setState(() {
      _currentIndex = index;
    });
    if (index == 0){
      Navigator.pushNamed(context, '/home');
    }
    else if (index == 1){
      Navigator.pushNamed(context, '/film');
    }
  }

  @override
  Widget build(BuildContext context) {
    return BottomNavigationBar(
      items: const <BottomNavigationBarItem> [
        BottomNavigationBarItem(
          icon: Icon(Icons.book_outlined),
          label: 'Buku'
        ), // BottomNavigationBarItem
        BottomNavigationBarItem(
          icon: Icon(Icons.movie_outlined),
          label: 'Film'
        ) // BottomNavigationBarItem
      ], // <BottomNavigationBarItem>[]
      selectedItemColor: Colors.amber,
      unselectedItemColor: Colors.black,
      onTap: changeSelectedNavBar,
      currentIndex: widget.selectedItem,
    ); // BottomNavigationBar
  }
}
```

widget NavBar ini menampilkan navigasi bar di bagian bawah dengan dua item: Buku dan Film. Ketika item ditekan, aplikasi akan navigasi ke halaman yang sesuai.

```dart
void main() {
  runApp(
    MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (context) => CartProvider()),
      ],
      child: MaterialApp(
        debugShowCheckedModeBanner: false,
        initialRoute: '/home',
        routes: {
          '/home': (context) => HomePage(),
          '/item': (context) => ItemPage(),
          '/cart': (context) => CartWidget(),
          '/film':(context) => FilmList()
        },
      ), // MaterialApp
    ), // MultiProvider
  );
}
```

Secara keseluruhan, kode ini mendefinisikan titik masuk aplikasi Flutter yang menggunakan CartProvider sebagai penyedia state, dan MaterialApp sebagai root dari pohon widget. Aplikasi ini memiliki empat rute: '/home', '/item', '/cart', dan '/film', masing-masing dikaitkan dengan widget HomePage, ItemPage, CartWidget, dan FilmList. Ketika aplikasi pertama kali dijalankan, rute '/home' akan dipilih dan HomePage akan ditampilkan.