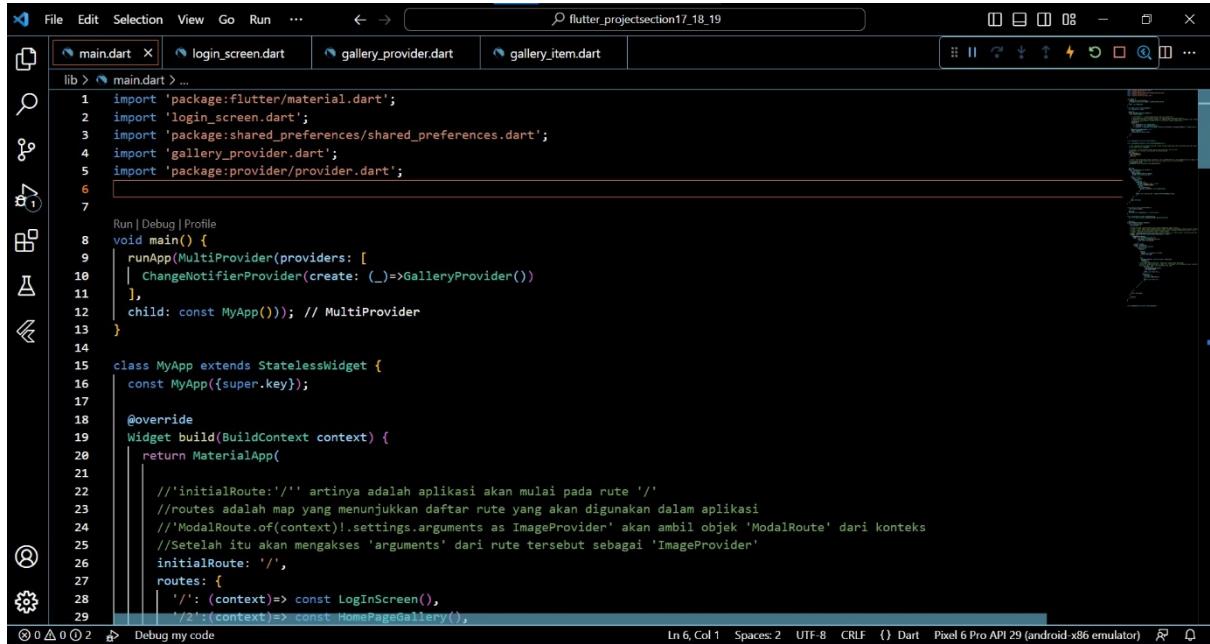


'initialRoute: '/' artinya adalah aplikasi akan mulai pada rute '/'

routes adalah map yang menunjukkan daftar rute yang akan digunakan dalam aplikasi

'ModalRoute.of(context)!.settings.arguments as ImageProvider' akan ambil objek 'ModalRoute' dari konteks

Setelah itu akan mengakses 'arguments' dari rute tersebut sebagai 'ImageProvider'

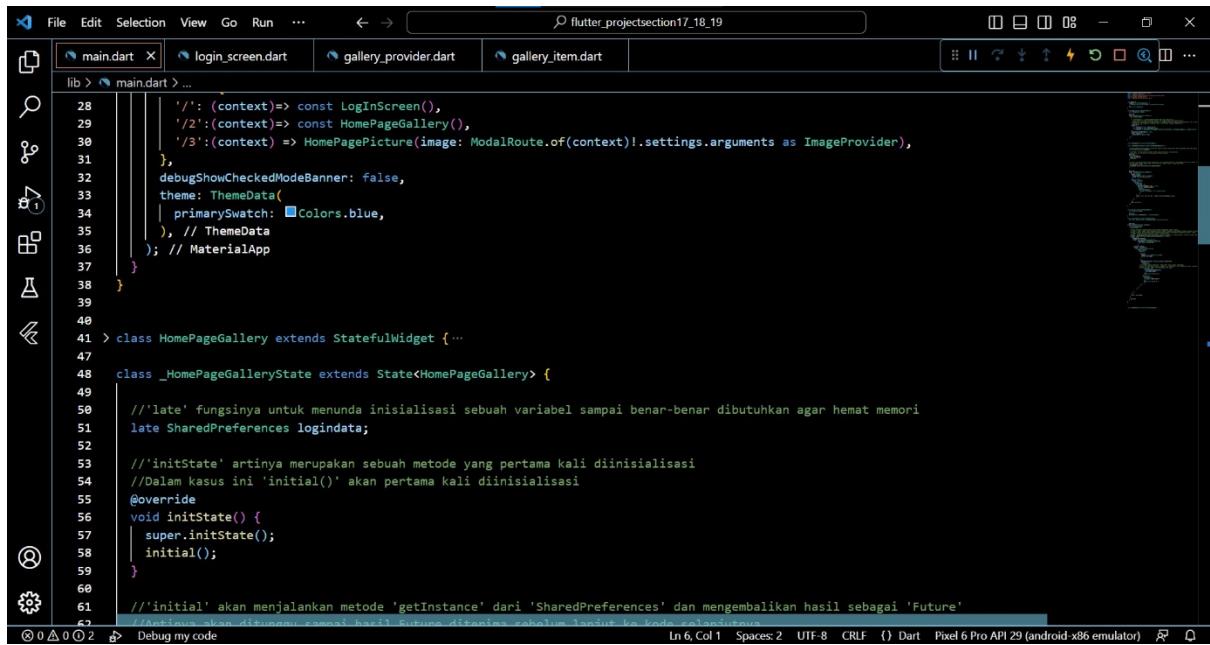


```
lib > main.dart > ...
1 import 'package:flutter/material.dart';
2 import 'login_screen.dart';
3 import 'package:shared_preferences/shared_preferences.dart';
4 import 'gallery_provider.dart';
5 import 'package:provider/provider.dart';
6
7 Run | Debug | Profile
8 void main() {
9   runApp(MultiProvider(providers: [
10   | ChangeNotifierProvider(create: (_)=>GalleryProvider())
11   ],
12   child: const MyApp())); // MultiProvider
13 }
14
15 class MyApp extends StatelessWidget {
16   const MyApp({super.key});
17
18   @override
19   Widget build(BuildContext context) {
20     return MaterialApp(
21
22       //initialRoute: '/' artinya adalah aplikasi akan mulai pada rute '/'
23       //routes adalah map yang menunjukkan daftar rute yang akan digunakan dalam aplikasi
24       //ModalRoute.of(context)!.settings.arguments as ImageProvider' akan ambil objek 'ModalRoute' dari konteks
25       //Setelah itu akan mengakses 'arguments' dari rute tersebut sebagai 'ImageProvider'
26       initialRoute: '/',
27       routes: {
28         '/': (context)=> const LogInScreen(),
29         '/2':(context)=> const HomePageGallery(),
30       }
31     );
32   }
33 }
```

'late' fungsinya untuk menunda inisialisasi sebuah variabel sampai benar-benar dibutuhkan agar hemat memori

'initState' artinya merupakan sebuah metode yang pertama kali diinisialisasi

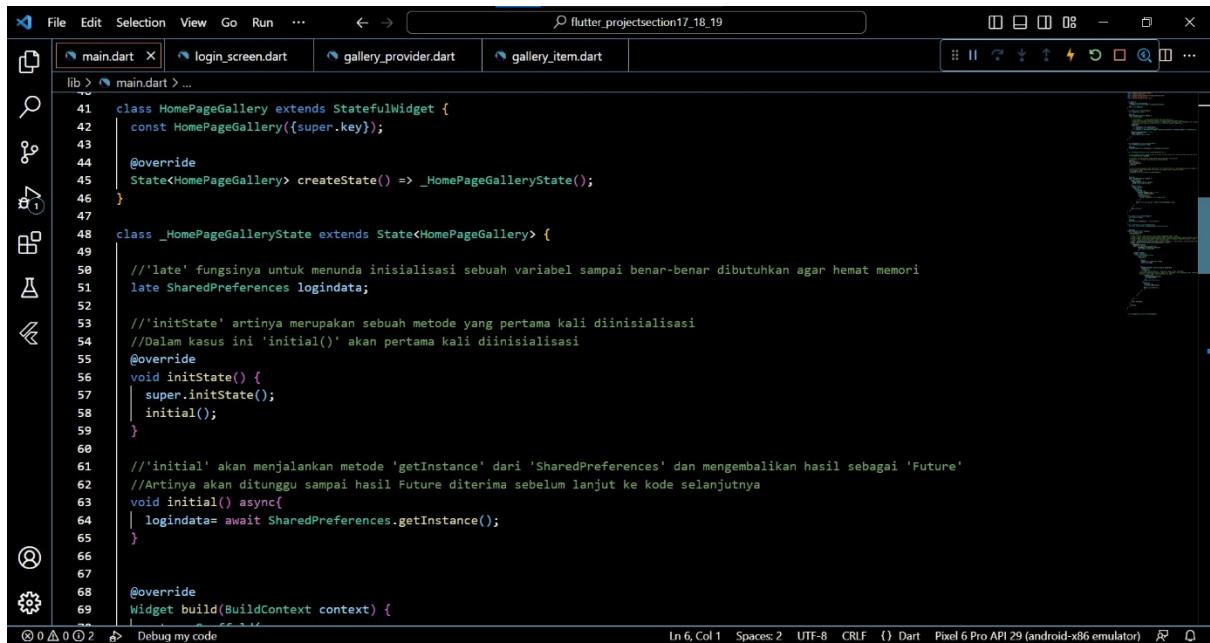
Dalam kasus ini 'initial()' akan pertama kali diinisialisasi



```
lib > main.dart > ...
28 |   '/': (context)=> const LogInScreen(),
29 |   '/2':(context)=> const HomePageGallery(),
30 |   '/3':(context) => HomePagePicture(image: ModalRoute.of(context)!.settings.arguments as ImageProvider),
31 | },
32 |   debugShowCheckedModeBanner: false,
33 |   theme: ThemeData(
34 |     primarySwatch: Colors.blue,
35 |   ), // ThemeData
36 | ); // MaterialApp
37 |
38 }
39
40 > class HomePageGallery extends StatelessWidget {
41   <> class _HomePageGalleryState extends State<HomePageGallery> {
42     <> //late' fungsinya untuk menunda inisialisasi sebuah variabel sampai benar-benar dibutuhkan agar hemat memori
43     <> late SharedPreferences logindata;
44     <>
45     //initState' artinya merupakan sebuah metode yang pertama kali diinisialisasi
46     //Dalam kasus ini 'initial()' akan pertama kali diinisialisasi
47     @override
48     void initState() {
49       super.initState();
50       initial();
51     }
52     <> // 'initial' akan menjalankan metode 'getInstance' dari 'SharedPreferences' dan mengembalikan hasil sebagai 'Future'
53     //Artinya akan ditunggu sampai hasil Future diterima sebelum lanjut ke kode selanjutnya
54     void initial() async{
55       logindata= await SharedPreferences.getInstance();
56     }
57     <>
58     @override
59     Widget build(BuildContext context) {
60       <>
61     }
62   }
63 }
```

'initial' akan menjalankan metode 'getInstance' dari 'SharedPreferences' dan mengembalikan hasil sebagai 'Future'

Artinya akan ditunggu sampai hasil Future diterima sebelum lanjut ke kode selanjutnya



```
lib > main.dart > ...
41   class HomePageGallery extends StatelessWidget {
42     const HomePageGallery({super.key});
43
44     @override
45     State<HomePageGallery> createState() => _HomePageGalleryState();
46   }
47
48   class _HomePageGalleryState extends State<HomePageGallery> {
49
50     //late' fungsinya untuk menunda inisialisasi sebuah variabel sampai benar-benar dibutuhkan agar hemat memori
51     late SharedPreferences logindata;
52
53     //initState' artinya merupakan sebuah metode yang pertama kali diinisialisasi
54     //Dalam kasus ini 'initial()' akan pertama kali diinisialisasi
55     @override
56     void initState() {
57       super.initState();
58       initial();
59     }
60
61     // 'initial' akan menjalankan metode 'getInstance' dari 'SharedPreferences' dan mengembalikan hasil sebagai 'Future'
62     //Artinya akan ditunggu sampai hasil Future diterima sebelum lanjut ke kode selanjutnya
63     void initial() async{
64       logindata= await SharedPreferences.getInstance();
65     }
66
67     @override
68     Widget build(BuildContext context) {
69       <>
70     }
71   }
72 }
```

```
lib > main.dart > ...
58  @override
59  Widget build(BuildContext context) {
60    return Scaffold(
61      appBar: AppBar(
62        backgroundColor: Colors.purple,
63        title: const Text('Gallery'),
64      ), // AppBar
65      drawer: Drawer(
66        child: ListView(
67          children: [
68            TextButton(
69              onPressed: () {
70                loginData.setBool('login', true);
71                Navigator.pushReplacement(
72                  context,
73                  MaterialPageRoute(
74                    builder: (context) => const LogInScreen(),
75                  ), // MaterialPageRoute
76                );
77              },
78              child: const Text('Log Out', style: TextStyle(fontSize: 20)),
79            ), // TextButton
80          ],
81        ), // ListView
82      ), // Drawer
83      body: Gallery(),
84    ); // Scaffold
85  }
86
87 
```

Ln 6, Col 1 Spaces: 2 UTF-8 CRLF {} Dart Pixel 6 Pro API 29 (android x86 emulator) ⚡

Setiap item pada 'myGalleryItems' akan dirender menggunakan widget 'Inkwell'

'Inkwell' adalah widget yang mengimplementasikan sebuah tindakan ketika pengguna menyentuh widget tersebut

Setiap 'Inkwell' akan menerima sebuah tindakan ketika diklik oleh pengguna menggunakan properti 'onTap'

'widget.' digunakan untuk mengakses properti 'myGalleryItems'

'map()' adalah metode pada List untuk ubah tiap item di dalam List 'myGalleryItems' ke bentuk widget yang akan dirender di dalam GridView

```
lib > gallery_provider.dart > ...
98
99 class Gallery extends StatefulWidget {
100   Gallery({super.key}); // Constructors in '@immutable' classes should be declared as 'const'. Try adding 'const' to the constructor decls
101
102   @override
103   State<Gallery> createState() => _GalleryState();
104 }
105
106 class _GalleryState extends State<Gallery> {
107   late final GalleryProvider _galleryProvider = GalleryProvider();
108
109
110   @override
111   Widget build(BuildContext context) {
112     return GridView.count(
113       crossAxisCount: 3,
114
115       // Setiap item pada 'myGalleryItems' akan dirender menggunakan widget 'Inkwell'
116       // 'Inkwell' adalah widget yang mengimplementasikan sebuah tindakan ketika pengguna menyentuh widget tersebut
117       // Setiap 'Inkwell' akan menerima sebuah tindakan ketika diklik oleh pengguna menggunakan properti 'onTap'
118       // 'widget.' digunakan untuk mengakses properti 'myGalleryItems'
119       // 'map()' adalah metode pada List untuk ubah tiap item di dalam List 'myGalleryItems' ke bentuk widget yang akan dirender di dalam Grid
120       children: _galleryProvider.myGalleryItems.map((item) => InkWell(
121         onTap: () {
122           showModalBottomSheet(
123             shape: const RoundedRectangleBorder(
124               borderRadius: BorderRadius.vertical(
125                 top: Radius.circular(20)
126               ), // BorderRadius.vertical
127             ), // RoundedRectangleBorder
128           ),
129         ),
130       ),
131     );
132   }
133 }
```

Ln 6, Col 1 Spaces: 2 UTF-8 CRLF {} Dart Pixel 6 Pro API 29 (android x86 emulator) ⚡

'item.image' adalah objek dari 'Image.asset' yang bertipe 'AssetImage'

Kita perlu menggunakan properti 'image' dari 'AssetImage' untuk mendapatkan objek 'ImageProvider'

Karena itu ada '.image' lagi dibelakang 'item.image'

```
lib > main.dart > ...
126     ) // BorderRadius.vertical
127     ), // RoundedRectangleBorder
128     context: context,
129     builder: (context) {
130         return SingleChildScrollView(
131             child: Column(
132                 children: [
133                     Padding(
134                         padding: const EdgeInsets.all(50),
135                         child: item.image,
136                     ), // Padding
137
138                     Row(
139                         mainAxisAlignment: MainAxisAlignment.spaceAround,
140                         children: [
141                             TextButton(
142                                 onPressed: () {
143                                     // 'item.image' adalah objek dari 'Image.asset' yang bertipe 'AssetImage'
144                                     // Kita perlu menggunakan properti 'image' dari 'AssetImage' untuk mendapatkan objek 'ImageProvider'
145                                     // Karena itu ada '.image' lagi dibelakang 'item.image'
146                                     item.onYesPressed(context,
147                                         item.image.image),
148                                     },
149                                     child: const Text('Yes'),
150                                 ), // TextButton
151                                 TextButton(
152                                     onPressed: () {
153                                         Navigator.pop(context);
154                                     },
155                                     child: const Text('No'),
156                                 ), // TextButton
157                             ],
158                         ), // Row
159                     ],
160                 ), // Column
161             ); // SingleChildScrollView
162
163         );
164     );
165 }
166 ),
167 child: item.image,
168 ) // InkWell
169 )
170 .toList(),
171 ); // GridView.count
172 }
173 }
174
175
176
177 > class HomePagePicture extends StatelessWidget { ... }
```

```
154     ),
155     child: const Text('No'),
156   ), // TextButton
157 ],
158 ), // Row
159 ],
160 ), // Column
161 ); // SingleChildScrollView
162
163 );
164 );
165 );
166 ),
167 child: item.image,
168 ) // InkWell
169 )
170 .toList(),
171 ); // GridView.count
172 }
173 }
174
175
176
177 > class HomePagePicture extends StatelessWidget { ... }
```

Disini 'floatingActionButton' ditambahkan pada halaman HomePagePicture untuk kembali ke halaman sebelumnya yaitu HomePageGallery

```
lib > main.dart > ...
175
176
177 class HomePagePicture extends StatelessWidget {
178   final ImageProvider image;
179
180   const HomePagePicture({super.key, required this.image});
181
182   @override
183   Widget build(BuildContext context) {
184     return Scaffold(
185       body: Center(
186         child: Image(image: image, height: 400),
187       ), // Center
188
189       //Disini 'floatingActionButton' ditambahkan pada halaman HomePagePicture untuk kembali ke halaman sebelumnya yaitu HomePageGallery
190       floatingActionButton: FloatingActionButton(
191         onPressed:(){
192           Navigator.pop(
193             context
194           );
195         },
196
197         child: const Icon(Icons.home),
198       ) // FloatingActionButton
199     ); // Scaffold
200
201   }
202 }
```

Ln 6, Col 1 Spaces: 2 UTF-8 CRLF {} Dart Pixel 6 Pro API 29 (android x86 emulator) ⚡

'logindata' merupakan variabel baru untuk menyimpan SharedPreference

'newUser' merupakan tipe data Boolean

'initState' artinya ini akan merupakan fungsi pertama kali yang dipanggil sebelum widget tree nya dibangun

```
lib > login_screen.dart > _LogInScreenState
1 import 'package:flutter/material.dart';
2 import 'main.dart';
3 import 'package:shared_preferences/shared_preferences.dart';
4 import 'package:email_validator/email_validator.dart';
5
6
7 class LogInScreen extends StatefulWidget {
8   const LogInScreen({super.key});
9
10  @override
11  State<LogInScreen> createState() => _LogInScreenState();
12 }
13
14 class _LogInScreenState extends State<LogInScreen> {
15
16   final formKey= GlobalKey<FormState>();
17
18   final _nameController= TextEditingController();
19   final _emailController= TextEditingController();
20   final _passwordController= TextEditingController();
21
22   // 'logindata' merupakan variabel baru untuk menyimpan SharedPreference
23   // 'newUser' merupakan tipe data boolean
24   late SharedPreferences logindata;
25   late bool newUser;
26
27   // 'initState' artinya ini akan merupakan fungsi pertama kali yang dipanggil sebelum widget tree nya dibangun
28   @override
29   void initState() {
30     super.initState();
31   }
32 }
```

Ln 50, Col 97 Spaces: 2 UTF-8 CRLF {} Dart Pixel 6 Pro API 29 (android x86 emulator) ⚡

'PageRouteBuilder' adalah sebuah kelas di Flutter yang digunakan untuk membuat rute transisi antara dua halaman dalam app

'pageBuilder' adalah parameter yang digunakan untuk menentukan bagaimana halaman tujuan akan dibangun

Dalam kasus ini, 'pageBuilder' digunakan untuk mengarahkan ke halaman 'HomePageGallery' ketika transisi dilakukan

'transitionBuilder' digunakan untuk menentukan bagaimana animasi transisi akan terjadi

Dalam kasus ini, 'transitionBuilder' digunakan untuk membuat animasi slide menggunakan 'SlideTransition'

'animation' dan 'secondaryAnimation' adalah 2 parameter yang digunakan pada 'pageBuilder'.  
'animation' mengendalikan animasi untuk halaman baru seperti animasi slide, fade, dll...

'secondaryAnimation' mewakili animasi yang digunakan untuk menggerakkan halaman lama keluar dari tampilan

Dalam kata lain, dengan 'secondaryAnimation' kita dapat kontrol bagaimana halaman lama dianimasikan saat keluar dari tampilan

'offset' digunakan dalam animasi untuk mengendalikan pergerakan atau perubahan posisi suatu objek atau widget

'curve' adalah parameter dalam animasi di flutter yang menentukan jenis kurva pergerakan animasi seperti Curves.linear, Curves.easeIn, dll...

'tween' adalah objek 'Tween' yang digunakan untuk mengatur perubahan nilai dalam animasi.

Dalam kasus ini, 'Tween' digunakan untuk mengatur perubahan posisi (begin dan end)

'SlideTransition' adalah sebuah widget yang digunakan untuk membuat animasi slide

'animation.drive(tween)' adalah cara untuk menghubungkan animasi yang didefinisikan dalam 'tween' dengan 'animation' yang diberikan 'PageRouteBuilder'

'child' adalah widget anak yang merupakan halaman tujuan yang diberikan dalam 'pageBuilder' akan dianimasikan oleh 'SlideTransition'

```
void initState() {
    super.initState();
    checkLogin();
}

// 'PageRouteBuilder' adalah sebuah kelas di Flutter yang digunakan untuk membuat rute transisi antara dua halaman dalam app
// 'pageBuilder' adalah parameter yang digunakan untuk menentukan bagaimana halaman tujuan akan dibangun
// Dalam kasus ini, 'pageBuilder' digunakan untuk mengarahkan ke halaman 'HomePageGallery' ketika transisi dilakukan
// 'transitionBuilder' digunakan untuk menentukan bagaimana animasi transisi akan terjadi
// Dalam kasus ini, 'transitionBuilder' digunakan untuk membuat animasi slide menggunakan 'SlideTransition'
// 'animation' dan 'secondaryAnimation' adalah 2 parameter yang digunakan pada 'pageBuilder'
// 'animation' mengendalikan animasi untuk halaman baru seperti animasi slide, fade, dll...
// 'secondaryAnimation' mewakili animasi yang digunakan untuk menggerakkan halaman lama keluar dari tampilan
// Dalam kata lain, dengan 'secondaryAnimation' kita dapat kontrol bagaimana halaman lama dianimasikan saat keluar dari tampilan
// 'offset' digunakan dalam animasi untuk mengendalikan pergerakan atau perubahan posisi suatu objek atau widget
// 'curve' adalah parameter dalam animasi di flutter yang menentukan jenis kurva pergerakan animasi seperti Curves.linear, Curves.easeIn,
// 'tween' adalah objek 'Tween' yang digunakan untuk mengetahui perubahan nilai dalam animasi.
// Dalam kasus ini, 'Tween' digunakan untuk mengatur perubahan posisi (begin dan end)
// 'SlideTransition' adalah sebuah widget yang digunakan untuk membuat animasi slide
// 'animation.drive(tween)' adalah cara untuk menghubungkan animasi yang diefisikan dalam 'tween' dengan 'animation' yang diberikan pada 'pageBuilder'
// 'child' adalah widget anak yang merupakan halaman tujuan yang diberikan dalam 'pageBuilder' akan dianimasikan oleh 'SlideTransition'
Route _createRoute(){
    return PageRouteBuilder(
        pageBuilder: (context, animation, secondaryAnimation)=> const HomePageGallery(),
        transitionsBuilder: (context, animation, secondaryAnimation, child){
            const beginOffset=<0, 10>;
            const endOffset.zero;
            const curve=Curves.ease;
        }
    );
}
```

'checkLogin' adalah fungsi asynchronous yang memeriksa apakah user telah login sebelumnya pada aplikasi menggunakan SharedPreference

'SharedPreferences' digunakan untuk menyimpan info yang perlu diakses kembali oleh aplikasi pada waktu yang beda seperti info login, data pengaturan, atau preferensi pengguna

'SharedPreferences.getInstance()' adalah untuk mendapatkan objek 'SharedPreferences' yang merepresentasikan file shared preferences default pada aplikasi

'await' digunakan untuk menunggu hasil pengembalian dari 'getInstance()' yang merupakan sebuah future sehingga kode selanjutnya akan tunggu sampai objek 'SharedPreferences' selesai

'newUser= logindata.getBool('login') ?? true' gunanya untuk ambil nilai boolean dari shared preferences dengan kunci 'login' kemudian simpan ke dalam variabel 'newUser'

Dalam kasus ini, 'newUser' dipakai untuk cek apakah user sudah pernah login atau belum

'???' adalah operator null-aware yang fungsinya untuk tahu nilai dari shared preferences null atau tidak dapat ditemukan

Kalau true, user akan disuruh login dulu dan kalau false maka akan diarahkan ke halaman HomePageGallery

```
//'checkLogin' adalah fungsi asynchronous yang memeriksa apakah user telah login sebelumnya pada aplikasi menggunakan SharedPreference
//SharedPreferences' digunakan untuk menyimpan info yang perlu diakses kembali oleh aplikasi pada waktu yang beda seperti info login, da
//SharedPreferences.getInstance()' adalah untuk mendapatkan objek 'SharedPreferences' yang merepresentasikan file shared preferences def
//'await' digunakan untuk menunggu hasil pengembalian dari 'getInstance()' yang merupakan sebuah future sehingga kode selanjutnya akan tu
//newUser= logindata.getBool('login') ?? true' gunanya untuk ambil nilai boolean dari shared preferences dengan kunci 'login' kemudian s
//Dalam kasus ini, 'newUser' dipakai untuk cek apakah user sudah pernah login atau belum
//???' adalah operator null-aware yang fungsinya untuk tahu nilai dari shared preferences null atau tidak dapat ditemukan
//Kalau true, user akan disuruh login dulu dan kalau false maka akan diarahkan ke halaman HomePageGallery
void checkLogin() async{
logindata= await SharedPreferences.getInstance();
newUser= logindata.getBool('login') ?? true;

if(newUser==false){
Navigator.pushAndRemoveUntil(
  context,
  MaterialPageRoute(),
  (Route route) => false)
}
}
```

'(route) => false' artinya aplikasi akan menentukan semua halaman sebelumnya harus dihapus dari stack karena nilai 'false'

Method dispose() digunakan untuk menghindari kebocoran memori

```
lib > login_screen.dart > _LoginScreenState
84     context,
85     _createRoute(),
86   );
87   // If '(route) => false' artinya aplikasi akan menentukan semua halaman sebelumnya harus dihapus dari stack karena nilai 'false'
88   // (route) => false,
89 );
90 }
91 }
92
93 //Method dispose() digunakan untuk menghindari kebocoran memori
94 @override
95 void dispose(){
96   _nameController.dispose();
97   _emailController.dispose();
98   _passwordController.dispose();
99
100 super.dispose();
101 }
102
103 @override
104 Widget build(BuildContext context) {
105   return Scaffold(
106     body: Container(
107       padding: const EdgeInsets.all(16),
108       child: Form(
109         autovalidateMode: AutovalidateMode.onUserInteraction,
110         key: formKey,
111         child: ListView(
112           children: [
113             TextFormField(
114               controller: _nameController,
115               decoration: const InputDecoration(
116                 prefixIcon: Icon(Icons.account_circle_rounded),
117                 labelText: 'Username',
118                 border: OutlineInputBorder(),
119               ), // InputDecoration
120               validator: (value){
121
122               //'(value!=null && value.length<4)' artinya jika inputan user tidak kosong dan kurang dari lima karakter maka akan ada pesan error
123               if (value!=null && value.length<4){
124                 return 'Enter at least 4 characters';
125               }
126               else{
127                 return null;
128               }
129             },
130           ), // TextFormField
131
132           TextFormField(
133             controller: _emailController,
134             decoration: const InputDecoration(
135               prefixIcon: Icon(Icons.email_rounded),
136               labelText: 'Email',
137               border: OutlineInputBorder(),
138             ), // InputDecoration
139             validator: (email){
140               if (email!=null && !EmailValidator.validate(email)){
```

'(value!=null && value.length<4)' artinya jika inputan user tidak kosong dan kurang dari lima karakter maka akan ada pesan error

```
lib > login_screen.dart > _LoginScreenState
112   children: [
113     TextFormField(
114       controller: _nameController,
115       decoration: const InputDecoration(
116         prefixIcon: Icon(Icons.account_circle_rounded),
117         labelText: 'Username',
118         border: OutlineInputBorder(),
119       ), // InputDecoration
120       validator: (value){
121
122         //'(value!=null && value.length<4)' artinya jika inputan user tidak kosong dan kurang dari lima karakter maka akan ada pesan error
123         if (value!=null && value.length<4){
124           return 'Enter at least 4 characters';
125         }
126         else{
127           return null;
128         }
129       },
130     ), // TextFormField
131
132     TextFormField(
133       controller: _emailController,
134       decoration: const InputDecoration(
135         prefixIcon: Icon(Icons.email_rounded),
136         labelText: 'Email',
137         border: OutlineInputBorder(),
138       ), // InputDecoration
139       validator: (email){
140         if (email!=null && !EmailValidator.validate(email)){
```

The screenshot shows a code editor with the file `login_screen.dart` open. The code defines a `TextFormField` for email and a `TextFormField` for password. Both fields have validators. The email field validator checks if the email is null or invalid. The password field validator checks if the password is null or less than 5 characters. Below the form fields is a `ElevatedButton`.

```
lib > login_screen.dart > _LoginScreenState > build
139 |     validator: (email){
140 |         if (email==null && !EmailValidator.validate(email)){
141 |             | return 'Enter a valid email';
142 |         }
143 |         else{
144 |             | return null;
145 |         }
146 |     },
147 |     ), // TextFormField
148 |
149 |     TextFormField(
150 |         controller: _passwordController,
151 |         decoration: const InputDecoration(
152 |             prefixIcon: Icon(Icons.password_rounded),
153 |             labelText: 'Password',
154 |             border: OutlineInputBorder(),
155 |         ), // InputDecoration
156 |         validator: (value){
157 |             if (value==null && value.length<5){
158 |                 | return 'Enter min. 5 Characters';
159 |             }
160 |             else{
161 |                 | return null;
162 |             }
163 |         },
164 |     ), // TextFormField
165 |
166 |     const SizedBox(height: 20),
167 |
168 |     ElevatedButton(
169 |         onPressed: (){
170 |             final isValidForm= formKey.currentState!.validate();
171 |
172 |             //Jika 'isValidForm' bernilai true, aplikasi akan menyimpan nilai 'false' pada SharedPreferences dengan key 'login'
173 |             // logindata.setBool('login', false)' artinya aplikasi memanggil metode 'setBool' untuk ubah nilai key 'login' menjadi 'false'
174 |             //Arti 'false' adalah menandakan kalau user sudah melakukan login
175 |             //'pushAndRemoveUntil' akan menghapus semua halaman yang ada pada stack dan menambahkan halaman baru ke stack
176 |             if (isValidForm){
177 |                 logindata.setBool('login', false);
178 |                 Navigator.pushAndRemoveUntil(
179 |                     context,
180 |                     MaterialPageRoute(
181 |                         builder: (context)=> const HomePageGallery(),
182 |                         ), // MaterialPageRoute
183 |                         (route) => false
184 |                     );
185 |                 }
186 |             },
187 |             child: const Text('Login'),
188 |         ), // ElevatedButton
189 |     ],
190 |     ), // ListView
191 |     ), // Form
192 |     ), // Container
193 | ); // Scaffold
194 |
195 | }
196 |
```

Jika 'isValidForm' bernilai true, aplikasi akan menyimpan nilai 'false' pada SharedPreferences dengan key 'login'

'logindata.setBool('login', false)' artinya aplikasi memanggil metode 'setBool' untuk ubah nilai key 'login' menjadi 'false'

Arti 'false' adalah menandakan kalau user sudah melakukan login

'pushAndRemoveUntil' akan menghapus semua halaman yang ada pada stack dan menambahkan halaman baru ke stack

The screenshot shows the same `login_screen.dart` file with additional code. It includes comments explaining the logic: if `isValidForm` is true, it sets `logindata.setBool('login', false)`. It also includes a note about `pushAndRemoveUntil` which removes all pages from the stack and adds a new one.

```
lib > login_screen.dart > _LoginScreenState > build
167 |
168 |         ElevatedButton(
169 |             onPressed: (){
170 |                 final isValidForm= formKey.currentState!.validate();
171 |
172 |                 //Jika 'isValidForm' bernilai true, aplikasi akan menyimpan nilai 'false' pada SharedPreferences dengan key 'login'
173 |                 // logindata.setBool('login', false)' artinya aplikasi memanggil metode 'setBool' untuk ubah nilai key 'login' menjadi 'false'
174 |                 //Arti 'false' adalah menandakan kalau user sudah melakukan login
175 |                 //'pushAndRemoveUntil' akan menghapus semua halaman yang ada pada stack dan menambahkan halaman baru ke stack
176 |                 if (isValidForm){
177 |                     logindata.setBool('login', false);
178 |                     Navigator.pushAndRemoveUntil(
179 |                         context,
180 |                         MaterialPageRoute(
181 |                             builder: (context)=> const HomePageGallery(),
182 |                             ), // MaterialPageRoute
183 |                             (route) => false
184 |                         );
185 |                     }
186 |                 },
187 |                 child: const Text('Login'),
188 |             ), // ElevatedButton
189 |         ],
190 |         ), // ListView
191 |         ), // Form
192 |         ), // Container
193 |     ); // Scaffold
194 |
195 | }
196 |
```

```
lib > gallery_provider.dart > GalleryProvider
1 import 'package:flutter/material.dart';
2 import 'gallery_item.dart';
3
4 class GalleryProvider extends ChangeNotifier {
5   final List<GalleryItem> _myGalleryItems = [
6     GalleryItem(
7       image: Image.asset('assets/images/image1.jpg', fit: BoxFit.cover),
8       onYesPressed: (BuildContext context, ImageProvider image) {
9         Navigator.pushNamed(
10           context, '/3', arguments: image
11         );
12       },
13       onNoPressed: (BuildContext context) {
14         Navigator.pop(context);
15       },
16     ), // GalleryItem
17     GalleryItem(
18       image: Image.asset('assets/images/image2.jpg', fit: BoxFit.cover),
19       onYesPressed: (BuildContext context, ImageProvider image) {
20         Navigator.pushNamed(
21           context, '/3', arguments: image
22         );
23       },
24       onNoPressed: (BuildContext context) {
25         Navigator.pop(context);
26       },
27     ), // GalleryItem
28     GalleryItem(
29       image: Image.asset('assets/images/image3.jpg', fit: BoxFit.cover),
30       onYesPressed: (BuildContext context, ImageProvider image) {
31         Navigator.pop(context);
32       },
33     ),
34   ];
35 }
36
37 }, // GalleryItem
38
39 GalleryItem(
40   image: Image.asset('assets/images/image4.jpg', fit: BoxFit.cover),
41   onYesPressed: (BuildContext context, ImageProvider image) {
42     Navigator.pushNamed(
43       context, '/3', arguments: image
44     );
45   },
46   onNoPressed: (BuildContext context) {
47     Navigator.pop(context);
48   },
49 ), // GalleryItem
50
51 GalleryItem(
52   image: Image.asset('assets/images/image5.jpg', fit: BoxFit.cover),
53   onYesPressed: (BuildContext context, ImageProvider image) {
54     Navigator.pushNamed(
55       context, '/3', arguments: image
56     );
57   },
58   onNoPressed: (BuildContext context) {
59     Navigator.pop(context);
60   },
61 ), // GalleryItem
62
63 GalleryItem(
64   image: Image.asset('assets/images/image6.jpg', fit: BoxFit.cover),
65   onYesPressed: (BuildContext context, ImageProvider image) {
66     Navigator.pushNamed(
67       context, '/3', arguments: image
68     );
69   },
70 }
```

The image shows two side-by-side screenshots of a code editor interface, likely from the Dart Editor or a similar IDE, displaying two files related to a Flutter project.

**Top Screenshot (File: gallery\_provider.dart):**

```
lib > gallery_provider.dart > GalleryProvider
72   GalleryItem(
73     image: Image.asset('assets/images/image7.jpg', fit: BoxFit.cover),
74     onYesPressed: (BuildContext context, ImageProvider image) {
75       Navigator.pushNamed(
76         context, '/3', arguments: image
77       );
78     },
79     onNoPressed: (BuildContext context) {
80       Navigator.pop(context);
81     },
82   ), // GalleryItem
83   GalleryItem(
84     image: Image.asset('assets/images/image8.jpg', fit: BoxFit.cover),
85     onYesPressed: (BuildContext context, ImageProvider image) {
86       Navigator.pushNamed(
87         context, '/3', arguments: image
88       );
89     },
90     onNoPressed: (BuildContext context) {
91       Navigator.pop(context);
92     },
93   ), // GalleryItem
94 ]
95 ];
96
97 List<GalleryItem> get myGalleryItems => _myGalleryItems;
98
99
100 ]
```

**Bottom Screenshot (File: gallery\_item.dart):**

```
lib > gallery_item.dart > GalleryItem
1 import 'package:flutter/material.dart';
2
3 class GalleryItem {
4   final Image image;
5   final Function(BuildContext, ImageProvider) onYesPressed;
6   final Function(BuildContext) onNoPressed;
7
8   GalleryItem({required this.image, required this.onYesPressed, required this.onNoPressed});
9 }
```

Both screenshots show the same basic UI elements: a top navigation bar with tabs for main.dart, login\_screen.dart, gallery\_provider.dart (active), and gallery\_item.dart; a left sidebar with icons for file operations like copy, paste, and search; and a bottom status bar with information like line number, column number, encoding, and build configuration.

File Edit Selection View Go Run ... ← → flutter\_projectsection17\_18\_19

login\_screen.dart gallery\_provider.dart gallery\_item.dart

lib > gallery\_item.dart > GalleryItem

```
1 import 'package:flutter/material.dart';
2
3 class GalleryItem {
4   final Image image;
5   final Function(BuildContext, ImageProvider) onYesPressed;
6   final Function(BuildContext) onNoPressed;
7
8   GalleryItem({required this.image, required this.onYesPressed, required this.onNoPressed});
9 }
```

11:30 @

Gallery

Debug my code

Ln 9, Col 2 Spaces: 2 UTF-8 CRLF {} Dart Pixel 6 Pro API 29 (android x86 emulator)