COOKBOOK

# Preparación

## Creación del Proyecto

Creamos el Proyecto Flutter en la carpeta **D:\Flutter\cookbook**

## Repositorio

Creamos el repositorio en GitHub y lo llamamos **CookBook**.

Con el explorador de archivos vamos ala carpeta **D:\Flutter\cookbook**

Hacemos clic derecho **Git Bash Here**

Escribimos el comando

git init

git add .

git commit -m "Basic Project creation"

git remote add origin https://github.com/LuisPulenta/CookBook.git

git push origin master

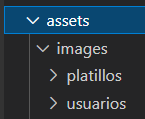
Luego abrimos el **GitHub DeskTop** y elegimos **File – Add local Repository** y buscamos la carpeta **D:\Flutter\cookbook**

Y nos queda listo para hacer los Push desde aquí

## Assets

Creamos la carpeta **D:\Flutter\cookbook\assets**

Dentro ponemos los archivos:



En el archivo **pubspec.yaml** ponemos

assets:

  - assets/logo.png

  - assets/images/platillos/

  - assets/images/usuarios/

## Fake BackEnd

Creamos la carpeta **D:\Flutter\cookbook\flutter\_modulo1\_fake\_backend**

Dentro ponemos los archivos del Fake BackEnd

En el archivo **pubspec.yaml** ponemos

cupertino\_icons: ^1.0.2

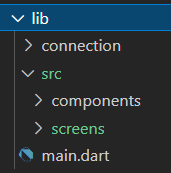
  image\_picker: ^0.5.0+9

  flutter\_modulo1\_fake\_backend:

    path: D:\Flutter\cookbook\flutter\_modulo1\_fake\_backend

## Carpetas

Dentro de la carpeta **lib** creamos las carpetas:



# Configuración del Proyecto

## Archivos iniciales

|  |  |
| --- | --- |
| **main.dart** | **Comentarios** |
| import 'package:cookbook/src/screens/app.dart';  import 'package:flutter/material.dart';  void main() {    runApp(const MyApp());  } | La App arranca por este archivo.  Llama a la pantalla MyApp |

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| import 'package:cookbook/src/screens/login\_screen.dart';  import 'package:flutter/material.dart';  class MyApp extends StatelessWidget {    const MyApp({Key key}) : super(key: key);    // This widget is the root of your application.    @override    Widget build(BuildContext context) {      return MaterialApp(        debugShowCheckedModeBanner: false,        title: 'Material App',        initialRoute: '/',        onGenerateRoute: (RouteSettings settings) {          return MaterialPageRoute(builder: (BuildContext context) {            switch (settings.name) {              case "/":                return LoginScreen();              default:                return LoginScreen();            }          });        },      );    }  } | Aquí se configuran las rutas.  La primera es LoginScreen |

|  |  |
| --- | --- |
| **login\_screen.dart** | **Comentarios** |
| import 'package:flutter/material.dart';  class LoginScreen extends StatefulWidget {    const LoginScreen({Key key}) : super(key: key);    @override    State<LoginScreen> createState() => \_LoginScreenState();  }  class \_LoginScreenState extends State<LoginScreen> {    @override    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("Login"),          centerTitle: true,        ),        body: Center(          child: Text("Login"),        ),      );    }  } |  |

# Login Screen

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/screens/home\_screen.dart';  import 'package:cookbook/src/screens/login\_screen.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  ServerController \_serverController = ServerController();  class MyApp extends StatelessWidget {    const MyApp({Key key}) : super(key: key);    // This widget is the root of your application.    @override    Widget build(BuildContext context) {      return MaterialApp(        debugShowCheckedModeBanner: false,        title: 'Material App',        initialRoute: '/',        onGenerateRoute: (RouteSettings settings) {          return MaterialPageRoute(builder: (BuildContext context) {            switch (settings.name) {              case "/":                return LoginScreen(\_serverController, context);              case "/home":                User loggedUser = settings.arguments;                \_serverController.loggedUser = loggedUser;                return HomeScreen(\_serverController, loggedUser);              default:                return LoginScreen(\_serverController, context);            }          });        },      );    }  } |  |

|  |  |
| --- | --- |
| **server\_controller.dart** | **Comentarios** |
| import 'package:flutter/widgets.dart';  import 'package:flutter\_modulo1\_fake\_backend/flutter\_modulo1\_fake\_backend.dart'      as server;  import 'package:flutter\_modulo1\_fake\_backend/models.dart';  class ServerController {    User loggedUser;    void init(BuildContext context) {      server.generateData(context);    }    Future<User> login(String userName, String password) async {      return await server.backendLogin(userName, password);    }  } |  |

|  |  |
| --- | --- |
| **login\_screen.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  class LoginScreen extends StatefulWidget {    ServerController serverController;    BuildContext context;    LoginScreen(this.serverController, this.context, {Key key}) : super(key: key);    @override    State<LoginScreen> createState() => \_LoginScreenState();  }  class \_LoginScreenState extends State<LoginScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Iniciación de variables \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    bool \_loading = false;    GlobalKey<FormState> \_formKey = GlobalKey<FormState>();    String userName = "";    String password = "";    String \_errorMessage = "";  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Init State \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    void initState() {      // TODO: implement initState      super.initState();      widget.serverController.init(widget.context);    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    Widget build(BuildContext context) {      return Scaffold(        body: Form(          key: \_formKey,          child: Stack(            children: <Widget>[              Container(                width: double.infinity,                padding: EdgeInsets.symmetric(vertical: 60),                decoration: BoxDecoration(                  gradient: LinearGradient(                    colors: [                      Color(0xff4dd0e1),                      Color(0xff00838f),                    ],                  ),                ),                child: Image.asset(                  "assets/logo.png",                  color: Colors.white,                  height: 200,                ),              ),              Transform.translate(                offset: Offset(0, -60),                child: Center(                  child: SingleChildScrollView(                    child: Card(                      shape: RoundedRectangleBorder(                          borderRadius: BorderRadius.circular(20)),                      elevation: 3,                      margin:                          const EdgeInsets.only(left: 20, right: 20, top: 260),                      child: Padding(                        padding: const EdgeInsets.symmetric(                            horizontal: 35, vertical: 20),                        child: Column(                          mainAxisSize: MainAxisSize.min,                          mainAxisAlignment: MainAxisAlignment.center,                          children: <Widget>[                            TextFormField(                              decoration: InputDecoration(labelText: "Usuario: "),                              onSaved: (value) {                                userName = value == null ? "" : value;                              },                              validator: (value) {                                if (value.isEmpty) {                                  return "Este campo es obligatorio";                                }                              },                            ),                            SizedBox(                              height: 40,                            ),                            TextFormField(                              decoration:                                  InputDecoration(labelText: "Contraseña: "),                              obscureText: true,                              onSaved: (value) {                                password = value == null ? "" : value;                              },                              validator: (value) {                                if (value.isEmpty) {                                  return "Este campo es obligatorio";                                }                              },                            ),                            SizedBox(                              height: 40,                            ),                            ElevatedButton(                              child: Row(                                mainAxisAlignment: MainAxisAlignment.center,                                children: [                                  \_loading                                      ? Container(                                          child: CircularProgressIndicator(                                            color: Colors.white,                                          ),                                          height: 20,                                          width: 20,                                        )                                      : Icon(Icons.login),                                  SizedBox(                                    width: 40,                                  ),                                  Text('Iniciar sesión'),                                ],                              ),                              style: ElevatedButton.styleFrom(                                primary: Color(0xff4dd0e1),                                minimumSize: Size(double.infinity, 50),                                shape: RoundedRectangleBorder(                                  borderRadius: BorderRadius.circular(5),                                ),                              ),                              onPressed: () => \_login(context),                            ),                            if (\_errorMessage.isNotEmpty)                              Padding(                                padding: const EdgeInsets.all(8),                                child: Text(                                  \_errorMessage,                                  style: TextStyle(                                    color: Colors.red,                                    fontWeight: FontWeight.bold,                                  ),                                  textAlign: TextAlign.center,                                ),                              ),                            SizedBox(                              height: 20,                            ),                            Row(                              mainAxisAlignment: MainAxisAlignment.center,                              children: <Widget>[                                Text("¿No estás registrado?"),                                SizedBox(                                  width: 20,                                ),                                TextButton(                                  child: Text(                                    'Registrarse',                                    style: TextStyle(color: Color(0xff4dd0e1)),                                  ),                                  onPressed: () => \_showRegister(context),                                ),                              ],                            )                          ],                        ),                      ),                    ),                  ),                ),              ),            ],          ),        ),      );    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Métodos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    void \_login(BuildContext context) async {      if (!\_loading) {        if (\_formKey.currentState.validate()) {          \_formKey.currentState.save();          setState(() {            \_loading = true;            \_errorMessage = "";          });          User user = await widget.serverController.login(userName, password);          if (user != null) {            Navigator.of(context).pushReplacementNamed("/home", arguments: user);          } else {            setState(() {              \_errorMessage = "Usuario o contraseña incorrectos";              \_loading = false;            });          }        }      }    }    void \_showRegister(BuildContext context) {      Navigator.of(context).pushNamed('/register');    }  } |  |

|  |  |
| --- | --- |
| **home\_screen.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  class HomeScreen extends StatefulWidget {    ServerController serverController;    User loggedUser;    HomeScreen(this.serverController, this.loggedUser, {Key key})        : super(key: key);    @override    State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("Home"),          centerTitle: true,        ),        body: Center(          child: Text("Home"),        ),      );    }  } |  |

# Register Screen

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/screens/home\_screen.dart';  import 'package:cookbook/src/screens/login\_screen.dart';  import 'package:cookbook/src/screens/register\_screen.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  ServerController \_serverController = ServerController();  class MyApp extends StatelessWidget {    const MyApp({Key key}) : super(key: key);    // This widget is the root of your application.    @override    Widget build(BuildContext context) {      return MaterialApp(        debugShowCheckedModeBanner: false,        title: 'Material App',        initialRoute: '/',        onGenerateRoute: (RouteSettings settings) {          return MaterialPageRoute(builder: (BuildContext context) {            switch (settings.name) {              case "/":                return LoginScreen(\_serverController, context);              case "/home":                User loggedUser = settings.arguments;                \_serverController.loggedUser = loggedUser;                return HomeScreen(\_serverController, loggedUser);              case "/register":                User loggedUser = settings.arguments;                return RegisterScreen(\_serverController, context,                    userToEdit: loggedUser);              default:                return LoginScreen(\_serverController, context);            }          });        },      );    }  } |  |

|  |  |
| --- | --- |
| **image\_picker.dart** | **Comentarios** |
| import 'dart:io';  import 'package:flutter/material.dart';  import 'package:image\_picker/image\_picker.dart';  typedef OnImageSelected = Function(File imageFile);  class ImagePickerWidget extends StatelessWidget {    final File imageFile;    final OnImageSelected onImageSelected;    ImagePickerWidget({@required this.imageFile, @required this.onImageSelected});    @override    Widget build(BuildContext context) {      return Container(          width: double.infinity,          height: 320,          decoration: BoxDecoration(              gradient: LinearGradient(                colors: [                  Color(0xff4dd0e1),                  Color(0xff00838f),                ],                begin: Alignment.topLeft,                end: Alignment.bottomRight,              ),              image: imageFile != null                  ? DecorationImage(                      image: FileImage(imageFile), fit: BoxFit.cover)                  : null),          child: IconButton(            icon: Icon(Icons.camera\_alt),            onPressed: () {              \_showPickerOptions(context);            },            iconSize: 90,            color: Colors.white,          ));    }    void \_showPickerOptions(BuildContext context) {      showDialog(          context: context,          builder: (context) {            return SimpleDialog(              children: <Widget>[                ListTile(                  title: Text("Cámara"),                  leading: Icon(Icons.camera\_alt),                  onTap: () {                    Navigator.pop(context);                    \_showPickImage(context, ImageSource.camera);                  },                ),                ListTile(                  title: Text("Galería"),                  leading: Icon(Icons.image),                  onTap: () {                    Navigator.pop(context);                    \_showPickImage(context, ImageSource.gallery);                  },                )              ],            );          });    }    void \_showPickImage(BuildContext context, source) async {      var image = await ImagePicker.pickImage(source: source);      this.onImageSelected(image);    }  } |  |

|  |  |  |
| --- | --- | --- |
| **server\_controller.dart** | **Comentarios** | |
| import 'package:flutter/widgets.dart';  import 'package:flutter\_modulo1\_fake\_backend/flutter\_modulo1\_fake\_backend.dart'      as server;  import 'package:flutter\_modulo1\_fake\_backend/models.dart';  class ServerController {    User loggedUser;    void init(BuildContext context) {      server.generateData(context);    }    Future<User> login(String userName, String password) async {      return await server.backendLogin(userName, password);    }    Future<bool> addUser(User nUser) async {      if (nUser != null) {        return await server.addUser(nUser);      } else {        return null;      }    }    Future<bool> updateUser(User user) async {      loggedUser = user;      return await server.updateUser(user);    }  } |  | |
| **register\_screen.dart** | | **Comentarios** |
| import 'dart:io';  import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/image\_picker.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  ServerController \_serverController = ServerController();  class RegisterScreen extends StatefulWidget {    ServerController serverController;    BuildContext context;    User userToEdit;    RegisterScreen(this.serverController, this.context,        {Key key, this.userToEdit})        : super(key: key);    @override    \_RegisterScreenState createState() => \_RegisterScreenState();  }  class \_RegisterScreenState extends State<RegisterScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Iniciación de variables \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    bool \_loading = false;    GlobalKey<FormState> \_formKey = GlobalKey<FormState>();    GlobalKey<ScaffoldState> \_scaffKey = GlobalKey<ScaffoldState>();    String userName = "";    String password = "";    Genrer genrer = Genrer.MALE;    String \_errorMessage = "";    File imageFile;    bool showPassword = false;    bool editingUser = false;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Init State \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    void initState() {      // TODO: implement initState      super.initState();      editingUser = (widget.userToEdit != null);      if (editingUser) {        userName = widget.userToEdit.nickname;        password = widget.userToEdit.password;        imageFile = widget.userToEdit.photo;        genrer = widget.userToEdit.genrer;      }    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    Widget build(BuildContext context) {      return Scaffold(        backgroundColor: Colors.cyan[50],        body: Form(          key: \_formKey,          child: Stack(            children: <Widget>[              ImagePickerWidget(                imageFile: this.imageFile,                onImageSelected: (File file) {                  setState(() {                    imageFile = file;                  });                },              ),              SizedBox(                child: AppBar(                  title: Text("Registrarse"),                  elevation: 0,                  backgroundColor: Colors.transparent,                  iconTheme: IconThemeData(color: Colors.white),                ),                height: kToolbarHeight + 25,              ),              Center(                child: Transform.translate(                  offset: Offset(0, -20),                  child: SizedBox(                    height: 620,                    child: Card(                      shape: RoundedRectangleBorder(                          borderRadius: BorderRadius.circular(20)),                      elevation: 15,                      margin: EdgeInsets.only(                          left: 20, right: 20, top: 260, bottom: 20),                      child: Padding(                        padding:                            EdgeInsets.symmetric(horizontal: 15, vertical: 20),                        child: ListView(                          children: <Widget>[                            TextFormField(                              initialValue: userName,                              decoration: InputDecoration(                                label: Text("Usuario:"),                                prefixIcon: Icon(Icons.alternate\_email),                              ),                              onSaved: (value) {                                userName = value == null ? "" : value;                              },                              validator: (value) {                                if (value.isEmpty) {                                  return "Este campo es obligatorio";                                }                              },                            ),                            TextFormField(                              initialValue: password,                              decoration: InputDecoration(                                label: Text("Contraseña:"),                                prefixIcon: Icon(Icons.lock),                                suffixIcon: IconButton(                                  icon: Icon(showPassword                                      ? Icons.visibility                                      : Icons.visibility\_off),                                  onPressed: () {                                    setState(() {                                      showPassword = !showPassword;                                    });                                  },                                ),                              ),                              obscureText: !showPassword,                              onSaved: (value) {                                password = value == null ? "" : value;                              },                              validator: (value) {                                if (value.isEmpty) {                                  return "Este campo es obligatorio";                                }                              },                            ),                            SizedBox(                              height: 20,                            ),                            Padding(                              padding: const EdgeInsets.all(0.0),                              child: Row(                                children: [                                  Text(                                    "Género:",                                    style: TextStyle(                                        fontWeight: FontWeight.bold,                                        color: Colors.grey[700]),                                  ),                                  Expanded(                                    child: RadioListTile(                                      activeColor: Color(0xff4dd0e1),                                      title: Text(                                        "Masc.",                                        style: TextStyle(fontSize: 12),                                      ),                                      value: Genrer.MALE,                                      groupValue: genrer,                                      onChanged: (value) {                                        setState(() {                                          genrer = value;                                        });                                      },                                    ),                                  ),                                  Expanded(                                    child: RadioListTile(                                      activeColor: Color(0xff4dd0e1),                                      title: Text(                                        "Femen.",                                        style: TextStyle(fontSize: 12),                                      ),                                      value: Genrer.FEMALE,                                      groupValue: genrer,                                      onChanged: (value) {                                        setState(() {                                          genrer = value;                                        });                                      },                                    ),                                  ),                                ],                              ),                            ),                            Theme(                              data: Theme.of(context)                                  .copyWith(accentColor: Colors.white),                              child: ElevatedButton(                                child: Row(                                  mainAxisAlignment: MainAxisAlignment.center,                                  children: [                                    \_loading                                        ? Container(                                            child: CircularProgressIndicator(                                              color: Colors.white,                                            ),                                            height: 20,                                            width: 20,                                          )                                        : Icon(Icons.person\_add),                                    SizedBox(                                      width: 40,                                    ),                                    Text(                                        editingUser ? "Actualizar" : "Registrar"),                                  ],                                ),                                style: ElevatedButton.styleFrom(                                  primary: Color(0xff4dd0e1),                                  minimumSize: Size(double.infinity, 50),                                  shape: RoundedRectangleBorder(                                    borderRadius: BorderRadius.circular(5),                                  ),                                ),                                onPressed: () => \_doProcess(context),                              ),                            ),                            if (\_errorMessage.isNotEmpty)                              Padding(                                padding: const EdgeInsets.all(8),                                child: Text(                                  \_errorMessage,                                  style: TextStyle(                                    color: Colors.red,                                    fontWeight: FontWeight.bold,                                  ),                                  textAlign: TextAlign.center,                                ),                              ),                            SizedBox(                              height: 20,                            ),                          ],                        ),                      ),                    ),                  ),                ),              )            ],          ),        ),      );    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Métodos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    void showSnackbar(BuildContext context, String title, Color backColor) {      ScaffoldMessenger.of(context).showSnackBar(        SnackBar(          content: Text(            title,            textAlign: TextAlign.center,          ),          backgroundColor: backColor,          // duration: Duration(seconds: 3),        ),      );    }    \_doProcess(BuildContext context) async {      if (\_formKey.currentState.validate()) {        \_formKey.currentState.save();        if (imageFile == null) {          showSnackbar(context, "Seleccione una imagen por favor", Colors.red);          return;        }        User user = User(            id: 0,            genrer: this.genrer,            nickname: this.userName,            password: this.password,            photo: this.imageFile);        var state;        if (editingUser) {          user.id = widget.serverController.loggedUser.id;          state = await widget.serverController.updateUser(user);        } else {          state = await widget.serverController.addUser(user);        }        final action = editingUser ? "actualizar" : "guardar";        final action2 = editingUser ? "actualizado" : "guardado";        if (state == false) {          showSnackbar(context, "No se pudo $action", Colors.orange);        } else {          showDialog(            context: context,            builder: (BuildContext context) {              return AlertDialog(                title: Text("Información"),                content: Text("Su usuario ha sido $action2 con éxito."),                actions: <Widget>[                  TextButton(                    child: Text("Ok"),                    onPressed: () {                      Navigator.pop(context);                      Navigator.pop(context);                    },                  )                ],                shape: RoundedRectangleBorder(                    borderRadius: BorderRadius.circular(10)),              );            },          );        }      }      ;    }  } | |  |

# Home Screen

|  |  |
| --- | --- |
| **my\_drawer.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/recipe\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  import '../components/my\_drawer.dart';  class HomeScreen extends StatefulWidget {    final ServerController serverController;    const HomeScreen(this.serverController, {Key key}) : super(key: key);    @override    State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("My CookBook"),          backgroundColor: Color(0xff4dd0e1),          centerTitle: true,        ),        body: FutureBuilder<List<Recipe>>(          future: widget.serverController.getRecipeList(),          builder: (context, snapshot) {            if (snapshot.hasData) {              final list = snapshot.data;              return ListView.builder(                itemCount: list.length,                itemBuilder: (context, index) {                  Recipe recipe = list[index];                  return RecipeWidget(                    recipe: recipe,                    serverController: widget.serverController,                    onChange: () {                      setState(() {});                    },                  );                },              );            } else {              return Center(                child: CircularProgressIndicator(),              );            }          },        ),        floatingActionButton: FloatingActionButton(          child: Icon(Icons.add),          backgroundColor: Color(0xff4dd0e1),          onPressed: () {            Navigator.of(context).pushNamed("/add\_recipe");          },        ),        drawer: MyDrawer(serverController: widget.serverController),      );    }  } |  |

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/screens/home\_screen.dart';  import 'package:cookbook/src/screens/login\_screen.dart';  import 'package:cookbook/src/screens/register\_screen.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  ServerController \_serverController = ServerController();  class MyApp extends StatelessWidget {    const MyApp({Key key}) : super(key: key);    // This widget is the root of your application.    @override    Widget build(BuildContext context) {      return MaterialApp(        debugShowCheckedModeBanner: false,        title: 'Material App',        initialRoute: '/',        onGenerateRoute: (RouteSettings settings) {          return MaterialPageRoute(builder: (BuildContext context) {            switch (settings.name) {              case "/":                return LoginScreen(\_serverController, context);              case "/home":                User loggedUser = settings.arguments;                \_serverController.loggedUser = loggedUser;                return HomeScreen(\_serverController);              case "/register":                User loggedUser = settings.arguments;                return RegisterScreen(\_serverController, context,                    userToEdit: loggedUser);              default:                return LoginScreen(\_serverController, context);            }          });        },      );    }  } |  |

|  |  |
| --- | --- |
| **home\_screen.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  import 'package:flutter\_modulo1\_fake\_backend/user.dart';  import '../components/my\_drawer.dart';  class HomeScreen extends StatefulWidget {    final ServerController serverController;    const HomeScreen(this.serverController, {Key key}) : super(key: key);    @override    State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("My CookBook"),          backgroundColor: Color(0xff4dd0e1),          centerTitle: true,        ),        body: FutureBuilder<List<Recipe>>(          future: widget.serverController.getRecipeList(),          builder: (context, snapshot) {            if (snapshot.hasData) {              final list = snapshot.data;              return ListView.builder(                itemCount: list.length,                itemBuilder: (context, index) {                  Recipe recipe = list[index];                  return Padding(                    padding: const EdgeInsets.all(8.0),                    child: ClipRRect(                      borderRadius: BorderRadius.circular(25),                      child: Card(                        child: Container(                          alignment: Alignment.bottomLeft,                          height: 250,                          width: double.infinity,                          decoration: BoxDecoration(                            image: DecorationImage(                                image: FileImage(recipe.photo),                                fit: BoxFit.cover),                          ),                          child: Container(                            color: Colors.black.withOpacity(0.35),                            child: ListTile(                              title: Text(                                recipe.name,                                style:                                    TextStyle(color: Colors.white, fontSize: 16),                              ),                              subtitle: Text(                                recipe.user.nickname,                                style:                                    TextStyle(color: Colors.white, fontSize: 14),                              ),                              trailing: IconButton(                                icon: Icon(Icons.favorite),                                onPressed: () {},                                iconSize: 32,                              ),                            ),                          ),                        ),                      ),                    ),                  );                },              );            } else {              return Center(                child: CircularProgressIndicator(),              );            }          },        ),        floatingActionButton: FloatingActionButton(          child: Icon(Icons.add),          backgroundColor: Color(0xff4dd0e1),          onPressed: () {            Navigator.of(context).pushNamed("/add\_recipe");          },        ),        drawer: MyDrawer(serverController: widget.serverController),      );    }  } |  |

# Editar Usuario

Ya está hecho dentro de Register\_Screen

# Mis Favoritos

|  |  |
| --- | --- |
| **server\_controller.dart** | **Comentarios** |
| Future<List<Recipe>> getFavoritesList() async {      return await server.getFavorites();    }    Future<bool> getIsFavorite(Recipe recipeTocheck) async {      return await server.isFavorite(recipeTocheck);    }    Future<bool> deleteFavorite(Recipe favoriteRecipe) async {      return await server.deleteFavorite(favoriteRecipe);    }    Future<Recipe> addFavorite(Recipe nFavorite) async {      return await server.addFavorite(nFavorite);    } |  |

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| case "/favorites":                return MyFavoritesScreen(\_serverController); |  |

|  |  |
| --- | --- |
| **my\_drawer.dart** | **Comentarios** |
| ListTile(              title: Text(                "Mis favoritos",                style: TextStyle(fontSize: 18),              ),              leading: Icon(Icons.favorite, color: Colors.red),              onTap: () {                Navigator.pop(context);                Navigator.of(context).pushNamed("/favorites");              },            ), |  |

|  |  |
| --- | --- |
| **recipe\_widget.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class RecipeWidget extends StatelessWidget {    final Recipe recipe;    final ServerController serverController;    final VoidCallback onChange;    RecipeWidget({this.recipe, this.serverController, this.onChange, key})        : super(key: key);    @override    Widget build(BuildContext context) {      return Padding(        padding: EdgeInsets.fromLTRB(15, 20, 15, 0),        child: GestureDetector(          onTap: () => \_showDetails(context),          child: ClipRRect(            borderRadius: BorderRadius.circular(30),            child: Card(                child: Container(              height: 250,              width: double.infinity,              decoration: BoxDecoration(                  image: DecorationImage(                      image: FileImage(recipe.photo), fit: BoxFit.cover)),              alignment: Alignment.bottomLeft,              child: Container(                color: Colors.black.withOpacity(0.35),                child: ListTile(                  title: Text(                    recipe.name,                    style: TextStyle(                      color: Colors.white,                      fontWeight: FontWeight.bold,                      fontSize: 16,                    ),                  ),                  subtitle: Text(                    recipe.user.nickname,                    style: TextStyle(                      color: Colors.white,                    ),                  ),                  trailing: \_getFavoriteWidget(),                ),              ),            )),          ),        ),      );    }    Widget \_getFavoriteWidget() {      return FutureBuilder<bool>(          future: serverController.getIsFavorite(recipe),          builder: (context, snapshot) {            if (snapshot.hasData) {              final isFavorite = snapshot.data;              if (isFavorite) {                return IconButton(                  icon: Icon(Icons.favorite),                  color: Colors.red,                  onPressed: () async {                    await serverController.deleteFavorite(recipe);                    onChange();                  },                  iconSize: 32,                );              } else {                return IconButton(                  icon: Icon(Icons.favorite\_border),                  color: Colors.white,                  onPressed: () async {                    await serverController.addFavorite(recipe);                    onChange();                  },                  iconSize: 32,                );              }            } else {              return CircularProgressIndicator();            }          });    }    \_showDetails(BuildContext context) {      Navigator.pushNamed(context, "/details", arguments: recipe);    }  } |  |

|  |  |
| --- | --- |
| **home\_screen.dart** | **Comentarios** |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/my\_drawer.dart';  import 'package:cookbook/src/components/recipe\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class HomeScreen extends StatefulWidget {    final ServerController serverController;    const HomeScreen(this.serverController, {Key key}) : super(key: key);    @override    \_HomeScreenState createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {    @override    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("Recetas"),        ),        body: FutureBuilder<List<Recipe>>(          future: widget.serverController.getRecipeList(),          builder: (context, snapshot) {            if (snapshot.hasData) {              final list = snapshot.data;              return ListView.builder(                itemCount: list.length,                itemBuilder: (context, index) {                  Recipe recipe = list[index];                  return RecipeWidget(                    recipe: recipe,                    serverController: widget.serverController,                    onChange: () {                      setState(() {});                    },                  );                },              );            } else {              return Center(                child: CircularProgressIndicator(),              );            }          },        ),        floatingActionButton: FloatingActionButton(          child: Icon(Icons.add),          onPressed: () {            Navigator.of(context).pushNamed("/add\_recipe");          },        ),        drawer: MyDrawer(serverController: widget.serverController),      );    }  } |  |

|  |  |
| --- | --- |
| **myFavorites\_screen.dart** | **Comentarios** |
| // ignore\_for\_file: file\_names  import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/recipe\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class MyFavoritesScreen extends StatefulWidget {    final ServerController serverController;    const MyFavoritesScreen(this.serverController, {Key key}) : super(key: key);    @override    \_MyFavoritesScreenState createState() => \_MyFavoritesScreenState();  }  class \_MyFavoritesScreenState extends State<MyFavoritesScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("Mis Favoritos"),          backgroundColor: Color(0xff4dd0e1),          centerTitle: true,        ),        body: FutureBuilder<List<Recipe>>(          future: widget.serverController.getFavoritesList(),          builder: (context, snapshot) {            if (snapshot.hasData) {              final list = snapshot.data;              if (list.length == 0) {                return SizedBox(                  width: double.infinity,                  child: Column(                    crossAxisAlignment: CrossAxisAlignment.center,                    mainAxisAlignment: MainAxisAlignment.center,                    children: <Widget>[                      Icon(                        Icons.info,                        size: 120,                        color: Colors.grey[300],                      ),                      Text(                        "Ud. no tiene favoritos",                        textAlign: TextAlign.center,                      )                    ],                  ),                );              }              return ListView.builder(                itemCount: list.length,                itemBuilder: (context, index) {                  Recipe recipe = list[index];                  return RecipeWidget(                    recipe: recipe,                    serverController: widget.serverController,                    onChange: () {                      setState(() {});                    },                  );                },              );            } else {              return Center(                child: CircularProgressIndicator(),              );            }          },        ),      );    }  } |  |

# Mis Recetas

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| case "/myrecipes":                return MyRecipesScreen(\_serverController); |  |

|  |  |
| --- | --- |
| **server\_controller.dart** | **Comentarios** |
| Future<List<Recipe>> getUserRecipesList() async {      return await server.getUserRecipes(loggedUser);    } |  |

|  |  |
| --- | --- |
| **myRecipes\_screen.dart** | **Comentarios** |
| // ignore\_for\_file: file\_names  import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/recipe\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class MyRecipesScreen extends StatefulWidget {    final ServerController serverController;    const MyRecipesScreen(this.serverController, {Key key}) : super(key: key);    @override    \_MyRecipesScreenState createState() => \_MyRecipesScreenState();  }  class \_MyRecipesScreenState extends State<MyRecipesScreen> {    @override  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: Text("Mis Recetas"),          backgroundColor: Color(0xff4dd0e1),          centerTitle: true,        ),        body: FutureBuilder<List<Recipe>>(          future: widget.serverController.getUserRecipesList(),          builder: (context, snapshot) {            if (snapshot.hasData) {              final list = snapshot.data;              if (list.length == 0) {                return SizedBox(                  width: double.infinity,                  child: Column(                    crossAxisAlignment: CrossAxisAlignment.center,                    mainAxisAlignment: MainAxisAlignment.center,                    children: <Widget>[                      Icon(                        Icons.info,                        size: 120,                        color: Colors.grey[300],                      ),                      Text(                        "Ud. no tiene recetas propias",                        textAlign: TextAlign.center,                      )                    ],                  ),                );              }              return ListView.builder(                itemCount: list.length,                itemBuilder: (context, index) {                  Recipe recipe = list[index];                  return RecipeWidget(                    recipe: recipe,                    serverController: widget.serverController,                    onChange: () {                      setState(() {});                    },                  );                },              );            } else {              return Center(                child: CircularProgressIndicator(),              );            }          },        ),      );    }  } |  |

# Detalle de una Receta

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| case "/details":                Recipe recipe = settings.arguments;                return DetailsScreen(recipe, \_serverController); |  |

|  |  |
| --- | --- |
| **tabIngredients\_widget.dart** | **Comentarios** |
| // ignore\_for\_file: prefer\_const\_constructors, file\_names  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/models.dart';  class TabIngredientsWidget extends StatelessWidget {    final Recipe recipe;    const TabIngredientsWidget({this.recipe, Key key}) : super(key: key);    @override    Widget build(BuildContext context) {      return ListView(        padding: EdgeInsets.only(top: 20, left: 20, right: 20),        children: <Widget>[          Text(            recipe.name,            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),          ),          SizedBox(            height: 10,          ),          Text(            recipe.description,            style: TextStyle(fontSize: 16),          ),          SizedBox(            height: 20,          ),          Text(            "Ingredientes",            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),          ),          SizedBox(            height: 15,          ),          Column(            children: List.generate(              recipe.ingredients.length,              (int index) {                final ingredient = recipe.ingredients[index];                return ListTile(                  leading: Container(                    height: 15,                    width: 15,                    decoration: BoxDecoration(                      shape: BoxShape.circle,                      color: Theme.of(context).primaryColor,                    ),                  ),                  title: Text(ingredient),                );              },            ),          )        ],      );    }  } |  |

|  |  |  |
| --- | --- | --- |
| **tabPreparation\_widget.dart** | | **Comentarios** |
| // ignore\_for\_file: prefer\_const\_constructors, file\_names  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/models.dart';  class TabPreparationWidget extends StatelessWidget {    final Recipe recipe;    const TabPreparationWidget({this.recipe, Key key}) : super(key: key);    @override    Widget build(BuildContext context) {      return ListView(        padding: EdgeInsets.only(top: 20, left: 20, right: 20),        children: <Widget>[          Text(            recipe.name,            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),          ),          SizedBox(            height: 10,          ),          Text(            recipe.description,            style: TextStyle(fontSize: 16),          ),          SizedBox(            height: 20,          ),          Text(            "Preparación",            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),          ),          SizedBox(            height: 15,          ),          Column(            children: List.generate(              recipe.steps.length,              (int index) {                final step = recipe.steps[index];                return ListTile(                  leading: Text(                    "${index + 1}",                    style: TextStyle(                        fontSize: 25,                        color: Theme.of(context).primaryColor,                        fontWeight: FontWeight.bold),                  ),                  title: Text(step),                );              },            ),          )        ],      );    }  } | |  |
| **recipe\_widget.dart** | **Comentarios** | |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class RecipeWidget extends StatelessWidget {    final Recipe recipe;    final ServerController serverController;    final VoidCallback onChange;    RecipeWidget({this.recipe, this.serverController, this.onChange, key})        : super(key: key);    @override    Widget build(BuildContext context) {      return Padding(        padding: EdgeInsets.fromLTRB(15, 20, 15, 0),        child: GestureDetector(          onTap: () => \_showDetails(context),          child: ClipRRect(            borderRadius: BorderRadius.circular(30),            child: Card(                child: Container(              height: 250,              width: double.infinity,              decoration: BoxDecoration(                  image: DecorationImage(                      image: FileImage(recipe.photo), fit: BoxFit.cover)),              alignment: Alignment.bottomLeft,              child: Container(                color: Colors.black.withOpacity(0.35),                child: ListTile(                  title: Text(                    recipe.name,                    style: TextStyle(                      color: Colors.white,                      fontWeight: FontWeight.bold,                      fontSize: 16,                    ),                  ),                  subtitle: Text(                    recipe.user.nickname,                    style: TextStyle(                      color: Colors.white,                    ),                  ),                  trailing: \_getFavoriteWidget(),                ),              ),            )),          ),        ),      );    }    Widget \_getFavoriteWidget() {      return FutureBuilder<bool>(          future: serverController.getIsFavorite(recipe),          builder: (context, snapshot) {            if (snapshot.hasData) {              final isFavorite = snapshot.data;              if (isFavorite) {                return IconButton(                  icon: Icon(Icons.favorite),                  color: Colors.red,                  onPressed: () async {                    await serverController.deleteFavorite(recipe);                    onChange();                  },                  iconSize: 32,                );              } else {                return IconButton(                  icon: Icon(Icons.favorite\_border),                  color: Colors.white,                  onPressed: () async {                    await serverController.addFavorite(recipe);                    onChange();                  },                  iconSize: 32,                );              }            } else {              return CircularProgressIndicator();            }          });    }    \_showDetails(BuildContext context) {      Navigator.pushNamed(context, "/details", arguments: recipe);    }  } |  | |

|  |  |
| --- | --- |
| **main.dart** |  |
| import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/tabIngredients\_widget.dart';  import 'package:cookbook/src/components/tabPreparation\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/recipe.dart';  class DetailsScreen extends StatefulWidget {    Recipe recipe;    final ServerController serverController;    DetailsScreen(this.recipe, this.serverController, {Key key})        : super(key: key);    @override    \_DetailsScreenState createState() => \_DetailsScreenState();  }  class \_DetailsScreenState extends State<DetailsScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Iniciación de variables \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    bool favorite = false;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Init State \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    void initState() {      super.initState();      loadState();    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    Widget build(BuildContext context) {      return Scaffold(          body: DefaultTabController(        length: 2,        child: NestedScrollView(          headerSliverBuilder: (BuildContext context, bool innerBoxIsScrolled) {            return [              SliverAppBar(                title: Text(widget.recipe.name),                expandedHeight: 320,                flexibleSpace: Container(                  decoration: BoxDecoration(                    image: DecorationImage(                      fit: BoxFit.cover,                      image: FileImage(widget.recipe.photo),                    ),                  ),                  child: Container(                    color: Colors.black.withOpacity(.5),                  ),                ),                pinned: true,                bottom: TabBar(                  indicatorWeight: 5,                  labelColor: Colors.white,                  tabs: <Widget>[                    Tab(                      child: Text(                        "Ingredientes",                        style: TextStyle(fontSize: 18),                      ),                    ),                    Tab(                      child: Text(                        "Preparación",                        style: TextStyle(fontSize: 18),                      ),                    ),                  ],                ),                actions: <Widget>[                  if (widget.recipe.user.id ==                      widget.serverController.loggedUser.id)                    IconButton(                        icon: Icon(Icons.edit),                        onPressed: () async {                          final nRecipe = await Navigator.of(context).pushNamed(                              "/edit\_recipe",                              arguments: widget.recipe);                          setState(() {                            widget.recipe = nRecipe;                          });                        }),                  getFavoriteWidget(),                  IconButton(                      onPressed: () {                        \_showAboutIt(context);                      },                      icon: Icon(Icons.help\_outline)),                ],              )            ];          },          body: TabBarView(            children: <Widget>[              TabIngredientsWidget(                recipe: widget.recipe,              ),              TabPreparationWidget(                recipe: widget.recipe,              ),            ],          ),        ),      ));    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Métodos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Widget getFavoriteWidget() {      if (favorite != null) {        if (favorite) {          return IconButton(            icon: Icon(Icons.favorite),            color: Colors.red,            onPressed: () async {              await widget.serverController.deleteFavorite(widget.recipe);              setState(() {                favorite = false;              });            },          );        } else {          return IconButton(            icon: Icon(Icons.favorite\_border),            color: Colors.white,            onPressed: () async {              await widget.serverController.addFavorite(widget.recipe);              setState(() {                favorite = true;              });            },          );        }      } else {        return Container(            margin: EdgeInsets.all(15),            width: 30,            child: CircularProgressIndicator());      }    }    void loadState() async {      final state = await widget.serverController.getIsFavorite(widget.recipe);      setState(() {        this.favorite = state;      });    }    void \_showAboutIt(BuildContext context) {      showDialog(          context: context,          builder: (BuildContext context) {            return AlertDialog(              title: Text("Acerca de la Receta"),              content: Column(                mainAxisSize: MainAxisSize.min,                children: <Widget>[                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: <Widget>[                      Text(                        "Nombre:  ",                        style: TextStyle(fontWeight: FontWeight.bold),                      ),                      Text(widget.recipe.name),                    ],                  ),                  Divider(),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: <Widget>[                      Text(                        "Usuario:  ",                        style: TextStyle(fontWeight: FontWeight.bold),                      ),                      Text(widget.recipe.user.nickname),                    ],                  ),                  Divider(),                  Row(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: <Widget>[                      Text(                        "Fecha:  ",                        style: TextStyle(fontWeight: FontWeight.bold),                      ),                      Text(                          "${widget.recipe.date.day}/${widget.recipe.date.month}/${widget.recipe.date.year}"),                    ],                  ),                ],              ),              actions: <Widget>[                TextButton(                  child: Text("Cerrar"),                  onPressed: () {                    Navigator.pop(context);                  },                )              ],            );          });    }  } |  |

# Nueva Receta

|  |  |
| --- | --- |
| **server\_controller.dart** | **Comentarios** |
| Future<Recipe> addRecipe(Recipe nRecipe) async {      return await server.addRecipe(nRecipe);    }    Future<bool> updateRecipe(Recipe recipe) async {      return await server.updateRecipe(recipe);    } |  |

|  |  |
| --- | --- |
| **ingredient\_widget.dart** | **Comentarios** |
| import 'package:flutter/material.dart';  typedef OnIngredientDeleteCallback = void Function(int index);  class IngredientWidget extends StatelessWidget {    final int index;    final String ingredientName;    final OnIngredientDeleteCallback onIngredientDeleteCallback;    final OnIngredientDeleteCallback onIngredientEditCallback;    const IngredientWidget({      this.index,      this.ingredientName,      this.onIngredientDeleteCallback,      this.onIngredientEditCallback,      key,    }) : super(key: key);    @override    Widget build(BuildContext context) {      Color backColor = Colors.white;      if ((index % 2) == 1) {        backColor = Colors.grey[300];      }      return DecoratedBox(        decoration: BoxDecoration(color: backColor),        child: ListTile(          leading: Text(            "${index + 1}",            style: TextStyle(              fontSize: 16,            ),          ),          title: Text(            ingredientName,            style: TextStyle(fontSize: 16),          ),          trailing: Row(            mainAxisSize: MainAxisSize.min,            children: <Widget>[              IconButton(                icon: Icon(Icons.edit),                onPressed: () {                  this.onIngredientEditCallback(index);                },              ),              IconButton(                icon: Icon(Icons.delete),                onPressed: () {                  this.onIngredientDeleteCallback(index);                },              )            ],          ),        ),      );    }  } |  |

|  |  |
| --- | --- |
| **app.dart** | **Comentarios** |
| case "/add\_recipe":                return AddRecipeScreen(                  \_serverController,                );              case "/edit\_recipe":                Recipe recipe = settings.arguments;                return AddRecipeScreen(                  \_serverController,                  recipe: recipe,                ); |  |

|  |  |
| --- | --- |
| **addRecipe\_screen.dart** | **Comentarios** |
| // ignore\_for\_file: missing\_return, annotate\_overrides, prefer\_const\_constructors\_in\_immutables, prefer\_function\_declarations\_over\_variables, deprecated\_member\_use, unnecessary\_this, file\_names, prefer\_const\_constructors, prefer\_is\_empty  import 'dart:io';  import 'package:cookbook/connection/server\_controller.dart';  import 'package:cookbook/src/components/image\_picker.dart';  import 'package:cookbook/src/components/ingredient\_widget.dart';  import 'package:flutter/material.dart';  import 'package:flutter\_modulo1\_fake\_backend/models.dart';  class AddRecipeScreen extends StatefulWidget {    final ServerController serverController;    final Recipe recipe;    AddRecipeScreen(this.serverController, {Key key, this.recipe})        : super(key: key);    \_AddRecipeScreenState createState() => \_AddRecipeScreenState();  }  class \_AddRecipeScreenState extends State<AddRecipeScreen> {  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Iniciación de variables \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    final formKey = GlobalKey<FormState>();    final scaffoldKey = GlobalKey<ScaffoldState>();    String name = "", description = "";    List<String> ingredientsList = [], stepsList = [];    File photoFile;    final nIngredientControler = TextEditingController();    final nPasoController = TextEditingController();    File imageFile;    bool editing = false;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Init State \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    void initState() {      // TODO: implement initState      super.initState();      editing = widget.recipe != null;      if (editing) {        name = widget.recipe.name;        description = widget.recipe.description;        ingredientsList = widget.recipe.ingredients;        stepsList = widget.recipe.steps;        imageFile = widget.recipe.photo;      }    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pantalla \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    @override    Widget build(BuildContext context) {      return Scaffold(        key: scaffoldKey,        body: Form(          key: formKey,          child: Stack(            children: <Widget>[              ImagePickerWidget(                imageFile: this.imageFile,                onImageSelected: (File file) {                  setState(() {                    imageFile = file;                  });                },              ),              SizedBox(                child: AppBar(                  elevation: 0,                  backgroundColor: Colors.transparent,                  iconTheme: IconThemeData(color: Colors.white),                  actions: <Widget>[                    IconButton(                      icon: Icon(Icons.save),                      onPressed: () {                        \_save(context);                      },                    )                  ],                ),                height: kToolbarHeight + 25,              ),              Center(                child: Card(                  elevation: 2,                  shape: RoundedRectangleBorder(                      borderRadius: BorderRadius.circular(20)),                  margin: const EdgeInsets.only(                      left: 20, right: 20, top: 260, bottom: 20),                  child: Padding(                    padding:                        const EdgeInsets.symmetric(horizontal: 35, vertical: 20),                    child: ListView(children: <Widget>[                      TextFormField(                        initialValue: name,                        decoration:                            InputDecoration(labelText: "Nombre de receta"),                        onSaved: (value) {                          this.name = value;                        },                        validator: (value) {                          if (value.isEmpty) {                            return "Llene este campo";                          }                        },                      ),                      TextFormField(                        initialValue: description,                        decoration: InputDecoration(labelText: "Descripción"),                        onSaved: (value) {                          this.description = value;                        },                        validator: (value) {                          if (value.isEmpty) {                            return "Llene este campo";                          }                        },                        maxLines: 6,                      ),                      SizedBox(                        height: 20,                      ),                      ListTile(                        title: Text("Ingredientes"),                        trailing: FloatingActionButton(                          heroTag: "uno",                          child: Icon(Icons.add),                          onPressed: () {                            \_ingredientDialog(context);                          },                        ),                      ),                      SizedBox(                        height: 20,                      ),                      getIngredientsList(),                      SizedBox(                        height: 20,                      ),                      ListTile(                        title: Text("Pasos"),                        trailing: FloatingActionButton(                          heroTag: "dos",                          child: Icon(Icons.add),                          onPressed: () {                            \_stepDialog(context);                          },                        ),                      ),                      SizedBox(                        height: 20,                      ),                      getStepsList()                    ]),                  ),                ),              )            ],          ),        ),      );    }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Métodos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Widget getStepsList() {      if (stepsList.length == 0) {        return Text(          "Listado vacío",          textAlign: TextAlign.center,          style: TextStyle(color: Colors.grey),        );      } else {        return Column(          children: List.generate(stepsList.length, (index) {            final ingredient = stepsList[index];            return IngredientWidget(              index: index,              ingredientName: ingredient,              onIngredientDeleteCallback: \_onStepDelete,              onIngredientEditCallback: \_onStepEdit,            );          }),        );      }    }    Widget getIngredientsList() {      if (ingredientsList.length == 0) {        return Text(          "Listado vacío",          textAlign: TextAlign.center,          style: TextStyle(color: Colors.grey),        );      } else {        return Column(          children: List.generate(ingredientsList.length, (index) {            final ingredient = ingredientsList[index];            return IngredientWidget(              index: index,              ingredientName: ingredient,              onIngredientDeleteCallback: \_onIngredientDelete,              onIngredientEditCallback: \_onIngredientEdit,            );          }),        );      }    }    void \_ingredientDialog(BuildContext context, {String ingredient, int index}) {      final textController = TextEditingController(text: ingredient);      final editing = ingredient != null;      final onSave = () {        final text = textController.text;        if (text.isEmpty) {          \_showSnackBar("El nombre está vacío", backColor: Colors.orange);        } else {          setState(() {            if (editing) {              ingredientsList[index] = text;            } else {              ingredientsList.add(text);            }            Navigator.pop(context);          });        }      };      showDialog(          context: context,          builder: (context) {            return AlertDialog(              title: Text(                  editing ? "Editando ingrediente" : "Agregando ingrediente"),              content: TextField(                controller: textController,                decoration: InputDecoration(labelText: "Ingrediente"),                onEditingComplete: onSave,              ),              actions: <Widget>[                FlatButton(                  child: Text("Cancelar"),                  onPressed: () {                    Navigator.pop(context);                  },                ),                FlatButton(                  child: Text(editing ? "Actualizar" : "Guardar"),                  onPressed: onSave,                ),              ],            );          });    }    void \_stepDialog(BuildContext context, {String step, int index}) {      final textController = TextEditingController(text: step);      final editing = step != null;      final onSave = () {        final text = textController.text;        if (text.isEmpty) {          \_showSnackBar("El paso está vacío", backColor: Colors.orange);        } else {          setState(() {            if (editing) {              stepsList[index] = text;            } else {              stepsList.add(text);            }            Navigator.pop(context);          });        }      };      showDialog(          context: context,          builder: (context) {            return AlertDialog(              title: Text(editing ? "Editando paso" : "Agregando paso"),              content: TextField(                controller: textController,                decoration: InputDecoration(                  labelText: "Paso",                ),                textInputAction: TextInputAction.newline,                maxLines: 6,                //onEditingComplete: onSave,              ),              actions: <Widget>[                FlatButton(                  child: Text("Cancelar"),                  onPressed: () {                    Navigator.pop(context);                  },                ),                FlatButton(                  child: Text(editing ? "Actualizar" : "Guardar"),                  onPressed: onSave,                ),              ],            );          });    }    void \_showSnackBar(String message, {Color backColor = Colors.black}) {      scaffoldKey.currentState.showSnackBar(        SnackBar(          content: Text(message),          backgroundColor: backColor,        ),      );    }    void \_onIngredientEdit(int index) {      final ingredient = ingredientsList[index];      \_ingredientDialog(context, ingredient: ingredient, index: index);    }    void \_onIngredientDelete(int index) {      questionDialog(context, "¿Seguro desea eliminar el ingrediente?", () {        setState(() {          ingredientsList.removeAt(index);        });      });    }    void \_onStepEdit(int index) {      final step = stepsList[index];      \_stepDialog(context, step: step, index: index);    }    void \_onStepDelete(int index) {      questionDialog(context, "¿Seguro desea eliminar el paso?", () {        setState(() {          stepsList.removeAt(index);        });      });    }    void questionDialog(BuildContext context, String message, VoidCallback onOk) {      showDialog(          context: context,          builder: (context) {            return AlertDialog(              content: Text(message),              actions: <Widget>[                FlatButton(                  child: Text("No"),                  onPressed: () {                    Navigator.pop(context);                  },                ),                FlatButton(                  child: Text("Si"),                  onPressed: () {                    Navigator.pop(context);                    onOk();                  },                ),              ],            );          });    }    void \_save(BuildContext context) async {      if (formKey.currentState.validate()) {        formKey.currentState.save();        if (imageFile == null) {          \_showSnackBar("La imágen está vacía");          return;        }        if (ingredientsList.length == 0) {          \_showSnackBar("No tiene ingredientes");          return;        }        if (stepsList.length == 0) {          \_showSnackBar("No tiene pasos");          return;        }        final recipe = Recipe(            name: this.name,            description: this.description,            ingredients: this.ingredientsList,            steps: this.stepsList,            photo: this.imageFile,            user: widget.serverController.loggedUser,            date: DateTime.now());        if (editing) {          recipe.id = widget.recipe.id;        }        bool saved = false;        if (editing) {          saved = await widget.serverController.updateRecipe(recipe);        } else {          final recipe2 = await widget.serverController.addRecipe(recipe);          saved = recipe2 != null;        }        if (saved) {          Navigator.pop(context, recipe);          showDialog(              context: context,              builder: (context) {                return AlertDialog(                  content: Text("Receta guardada exitosamente"),                  actions: <Widget>[                    FlatButton(                      child: Text("Ok"),                      onPressed: () {                        Navigator.pop(context);                      },                    )                  ],                );              });        } else {          \_showSnackBar("No se pudo realizar");        }      }    }  } |  |