

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №4**

**з дисципліни «Розробка ПЗ для мобільних пристроїв»**

**на тему:**

**«Можливості Firebase. Використання тем та анімації.»**

Виконав:

Студент гр. ПЗ2011

Кулик С.В.

Прийняв:

Іванов О.П.

Дніпро, 2023

**Тема.** Можливості Firebase. Використання тем та анімації.

**Мета.** Ознайомлення з безсерверними технологіями Firebase. Налаштування анімації та теми додатків.

**Завдання**

Створення додатку, що використовує сервіс Firebase. Додаток з функцією обміну повідомленнями, на зразок форуму(спільного чату). Повідомлення зберігаються у хмарному сховищі, розділені по папках (folders), користувач вказує своє ім’я та може переглядати/надсилати повідомлення до різних папок.

Удосконалити приклад, додати функції:

1. зміна імені користувача, збереження/відновлення при вході;

2. вводити назву нової папки при додаванні;

3. вивести додаткові дані в список папок (дата створення, користувач);

4. сортувати повідомлення за датою;

5. виділити повідомлення поточного користувача;

6. налаштувати світлу/темну тему додатку;

7. анімація діалогових вікон.

**Текст програми**

import 'package:flutter/material.dart';

import 'package:firebase\_core/firebase\_core.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

class ThemeSettings with ChangeNotifier {

bool \_dark = false;

get mode => \_dark ? ThemeMode.dark : ThemeMode.light;

get isDark => \_dark;

swap() {

\_dark = !\_dark;

notifyListeners();

}

light() => ThemeData(

brightness: Brightness.light,

primaryColor: Colors.*greenAccent*,

floatingActionButtonTheme:

FloatingActionButtonThemeData(backgroundColor: Colors.*greenAccent*),

useMaterial3: true,

);

dark() => ThemeData.dark();

}

ThemeSettings curTheme = ThemeSettings(); //global

void main() => runApp(MyApp());

class MyApp extends StatefulWidget {

MyApp({Key? key}) : super(key: key);

@override

\_MyAppState createState() => \_MyAppState();

}

class \_MyAppState extends State<MyApp> {

@override

void initState() {

super.initState();

curTheme.addListener(() {

setState(() {

//update all

});

});

}

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Lr4',

debugShowCheckedModeBanner: false,

themeMode: curTheme.mode,

theme: curTheme.light(),

darkTheme: curTheme.dark(),

home: MyHomePage(title: 'Forum app'),

);

}

}

class MyHomePage extends StatefulWidget {

final title;

MyHomePage({Key? key, this.title}) : super(key: key);

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

List \_folderList = [];

var \_currentFolder = "";

var \_currentUser = 'noName';

TextEditingController \_newFolderController = TextEditingController();

@override

\_MyHomePageState() {

final FirebaseOptions firebaseConfig = FirebaseOptions(

apiKey: "YOUR\_API\_KEY",

projectId: "YOUR\_PROJECT\_ID",

storageBucket: "YOUR\_STORAGE\_BUCKET",

appId: "YOUR\_APP\_ID",

messagingSenderId: '');

WidgetsFlutterBinding.*ensureInitialized*();

Firebase.*initializeApp*(options: firebaseConfig).whenComplete(() {

print("connected...");

\_getFolderList();

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(\_currentFolder),

actions: [

IconButton(

icon: Icon(Icons.*refresh*),

onPressed: \_getFolderList,

),

],

),

body: Center(child: \_folderItems()),

drawer: Drawer(

child: ListView(

children: [

DrawerHeader(

child: Text(\_currentUser),

decoration: BoxDecoration(color: Theme.*of*(context).primaryColor),

),

ListTile(

title: Text('FOLDERS'),

trailing: IconButton(

icon: Icon(Icons.*add*),

onPressed: \_newFolder,

),

),

for (var x in \_folderList)

Padding(

padding: EdgeInsets.fromLTRB(10, 0, 0, 0),

child: ListTile(

title: Text(x),

dense: true,

selected: \_currentFolder == x,

onTap: () => setState(() {

Navigator.*pop*(context);

\_currentFolder = x;

}),

),

),

ListTile(

title: Text('OPTIONS'),

),

Padding(

padding: EdgeInsets.fromLTRB(10, 0, 0, 0),

child: ListTile(

dense: true,

title: Text('Dark theme:'),

trailing: Switch(value: curTheme.isDark, onChanged: (x) => curTheme.swap()),

),

),

],

),

),

floatingActionButton: FloatingActionButton(

onPressed: \_displayDialog,

tooltip: 'new message',

child: Icon(Icons.*add*),

),

);

}

\_folderItems() {

if (\_currentFolder == "") return Text('Select folder');

CollectionReference msgs = FirebaseFirestore.*instance*

.collection('folders')

.doc(\_currentFolder)

.collection('items');

return StreamBuilder(

stream: msgs.snapshots(),

builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {

if (snapshot.hasError) return Text('Something went wrong');

if (snapshot.connectionState == ConnectionState.waiting) return Text("Loading");

final docs = snapshot.data!.docs;

print('items $\_currentFolder');

return ListView(

children: docs.map((DocumentSnapshot? document) {

if (document != null) {

final data = document.data() as Map;

return ListTile(

title: Text(data['text']),

subtitle: Text(data['user']),

trailing: Text(data['date'].toDate().toString() ?? '', textScaleFactor: 0.5),

);

}

}).toList().cast<Widget>(),

);

},

);

}

\_getFolderList() async {

var res = [];

print('folders:');

CollectionReference folders = FirebaseFirestore.*instance*.collection('folders');

var querySnapshot = await folders.get();

querySnapshot.docs.forEach((doc) {

Map d = doc.data() as Map;

print("${doc.id} ${d['creator'] ?? ''} ${d['date'] ?? ''}");

res += [doc.id];

});

setState(() {

\_folderList = res;

});

}

\_displayDialog() async {

final \_textFieldController = TextEditingController();

var valueText = '';

showGeneralDialog(

context: context,

pageBuilder: (context, anim1, anim2) {

return Text('');

},

barrierDismissible: true,

barrierColor: Colors.*black*.withOpacity(0.4),

barrierLabel: '',

transitionBuilder: (context, anim1, anim2, child) {

return Transform.rotate(

angle: anim1.value \* 360 \* (3.1416 / 180.0),

child: Opacity(

opacity: anim1.value,

child: AlertDialog(

title: Text('New message'),

content: TextField(

onChanged: (value) {

setState(() {

valueText = value;

});

},

controller: \_textFieldController,

minLines: null,

maxLines: null,

expands: true,

decoration: InputDecoration(hintText: "Text"),

),

actions: [

ElevatedButton(

child: Text('Send'),

onPressed: () => setState(() {

\_newMess(valueText);

Navigator.*pop*(context);

}),

),

],

),

),

);

},

transitionDuration: Duration(milliseconds: 500),

);

}

\_newMess(text) async {

CollectionReference fld =

FirebaseFirestore.*instance*.collection('folders').doc(\_currentFolder).collection('items');

await fld.add({'user': \_currentUser, 'text': text, 'date': DateTime.now()});

ScaffoldMessenger.*of*(context).showSnackBar(SnackBar(content: const Text('Message sent')));

setState(() {

//update

});

}

\_newFolder() async {

TextEditingController \_newFolderNameController = TextEditingController();

showDialog(

context: context,

builder: (BuildContext context) {

return AlertDialog(

title: Text('New Folder'),

content: TextField(

controller: \_newFolderNameController,

decoration: InputDecoration(hintText: 'Folder Name'),

),

actions: [

TextButton(

child: Text('Cancel'),

onPressed: () {

Navigator.*of*(context).pop();

},

),

TextButton(

child: Text('Create'),

onPressed: () {

\_createNewFolder(\_newFolderNameController.text);

Navigator.*of*(context).pop();

},

),

],

);

},

);

}

\_createNewFolder(folderName) async {

CollectionReference fld = FirebaseFirestore.*instance*.collection('folders');

await fld.doc(folderName).set({'creator': \_currentUser, 'date': DateTime.now()});

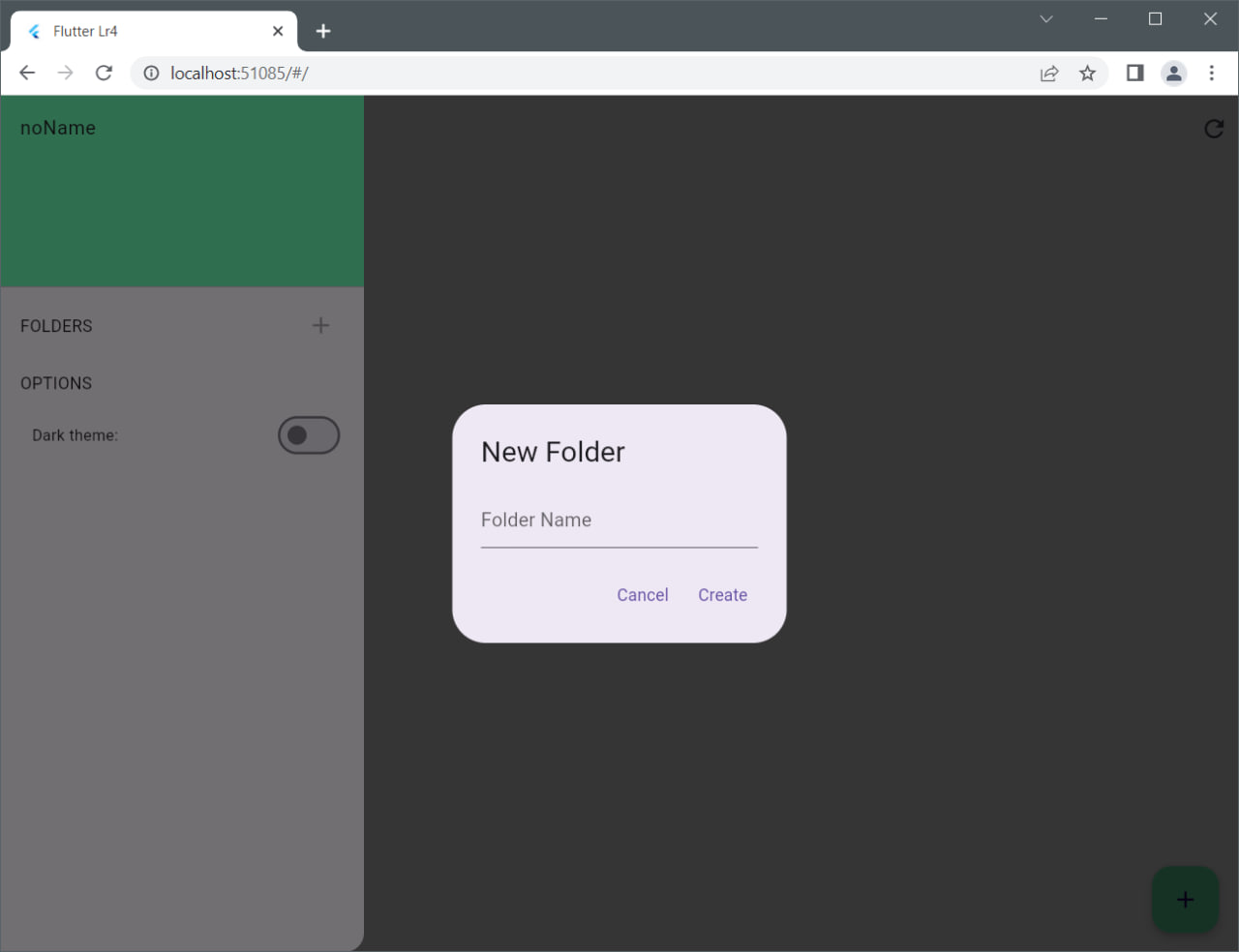
ScaffoldMessenger.*of*(context).showSnackBar(SnackBar(content: const Text('New folder created')));

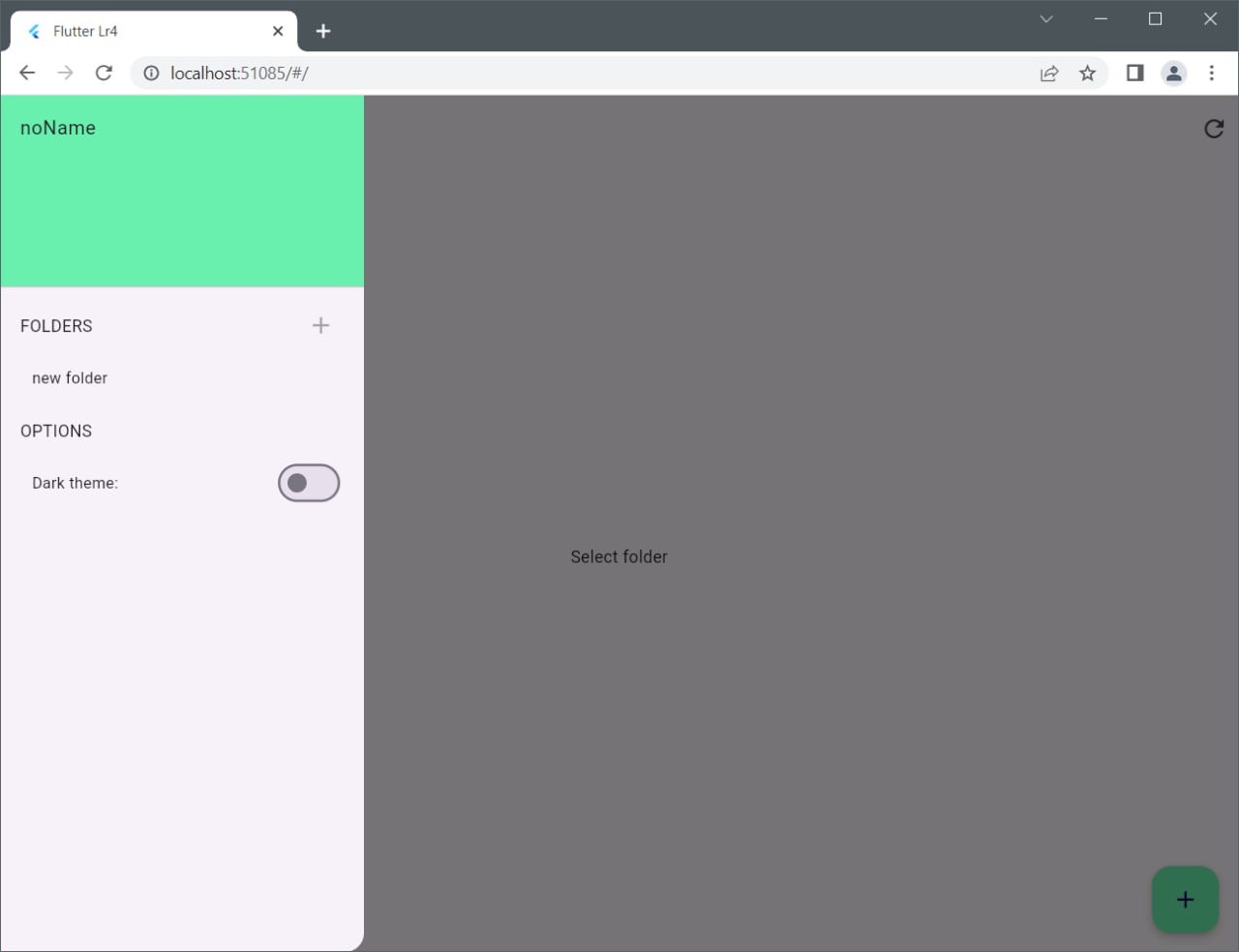
\_getFolderList();

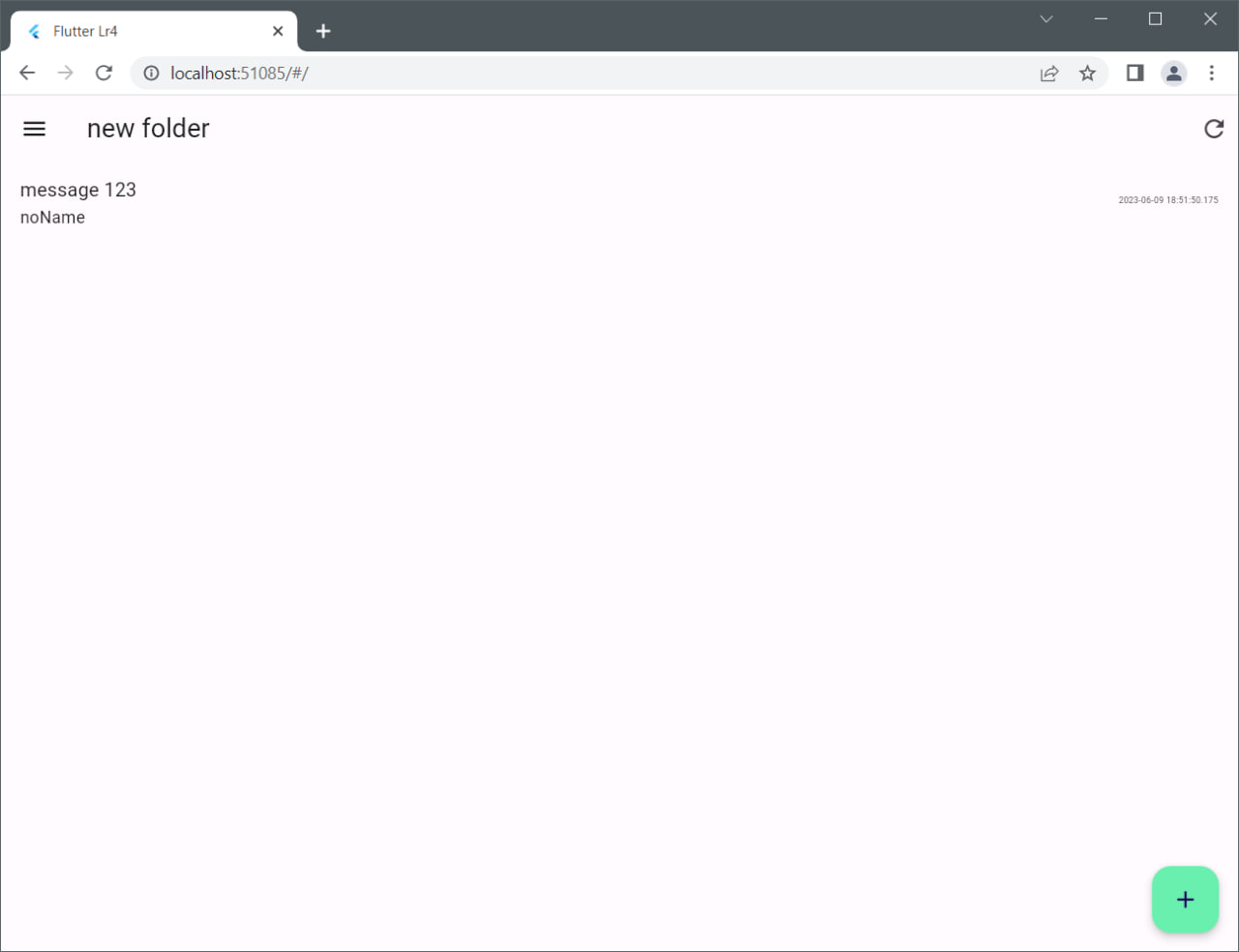
}

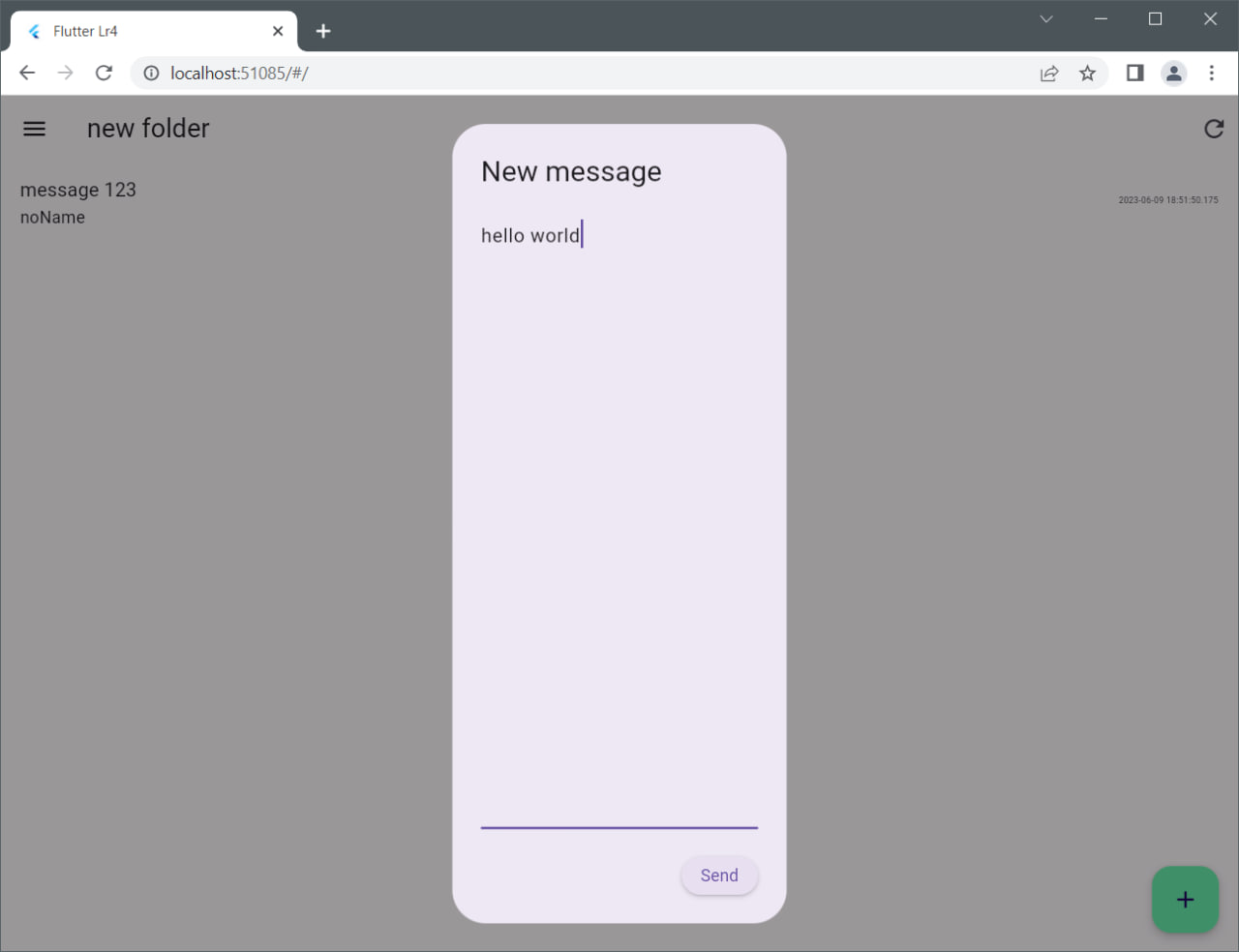
}

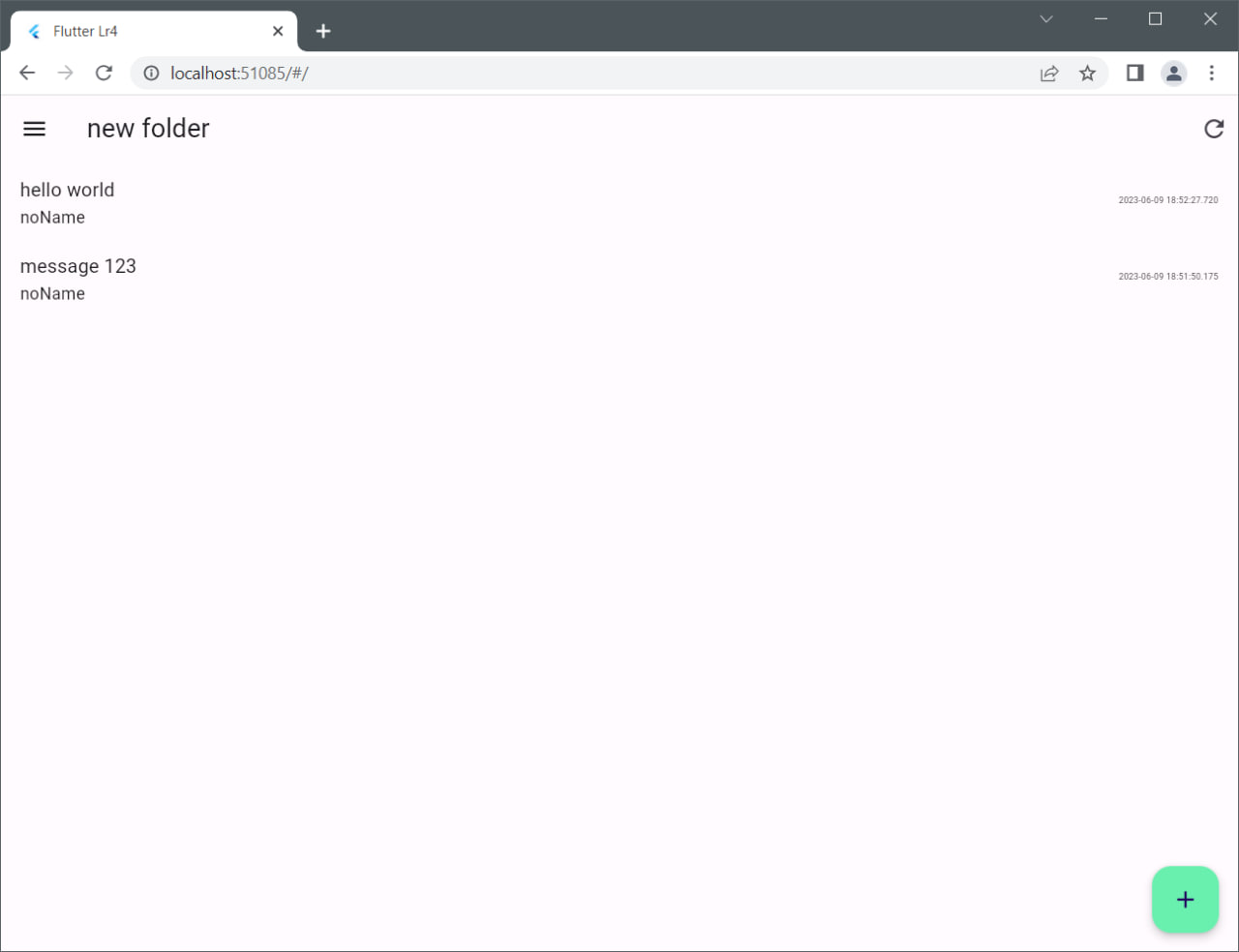
**Результати роботи програми**











Можна побачити що програма працює коректно, та виконує поставлені завдання.

**Висновок**

Firebase є потужною платформою, що надає безліч можливостей для розробки веб- та мобільних додатків. Використовуючи Firebase, розробники можуть створювати безсерверні додатки, що дозволяють писати функціональність на клієнтській стороні, використовуючи хмарні служби для збереження даних та обробки логіки.

Одна з ключових можливостей Firebase полягає у його базі даних реального часу. За допомогою Firebase Realtime Database, розробники можуть зберігати та синхронізувати дані між різними клієнтськими пристроями в реальному часі. Це дає змогу створювати додатки, які миттєво реагують на зміни даних та забезпечують зручне спілкування між користувачами.