

Arkadiusz Datka

22 kwietnia 2020 r.

Dokumentacja projektu : Firebase

Narzędzia i zewnętrzne Źródła :

1. Android Studio
2. Flutter
3. Firebase Realtime Database
4. Firebase Authorization

Aplikacja zawiera :

- autoryzację użytkownika
- zapisywanie danych użytkownika w Firebase realtime database
- pobieranie danych z OMDb API

Opis problemów i ich rozwiązania :

1. Rejestrowanie nowego użytkownika - poprzez Google Firebase Auth dodanie nowego użytkownika za pomocą funkcji :

```
void _register() async {  
  
  final FirebaseUser user = (await _auth.createUserWithEmailAndPassword(  
  
    email: _emailController.text,  
  
    password: _passwordController.text,)).user;  
  
  if (user != null) {  
  
    setState(() {_success = true;  
  
      _userEmail = user.email;  
  
      Toast.show("Register complete", context, duration: Toast.LENGTH_LONG, gravity: Toast.BOTTOM);  
  
      Navigator.pop(context);});  
  
  } else {_success = false;}}
```

2. Dodawanie danych do Firebase Realtime Database - dodanie danych do bazy odbywa się poprzez nadanie unikalnego Id dla użytkownika oraz dla trzech kategorii (Books, Games, Movies) tworzone są podgałęzie i dla nich zapis pojedynczego zapisu

```
Future<void> _increment(String title, String type, bool check, String userID, String other) async {  
  DatabaseReference _counterRef;  
  
  _counterRef = FirebaseDatabase.instance.reference().child('counter');  
  
  final FirebaseDatabase database = FirebaseDatabase();  
  
  DatabaseReference _messagesRef;  
  
  _messagesRef = database.reference().child(userID).child(type);  
  
  database.reference().child('counter').once().then((DataSnapshot snapshot) {});  
  
  final TransactionResult transactionResult = await _counterRef.runTransaction((MutableData mutableData) async {  
    mutableData.value = (mutableData.value ?? 0) + 1;  
  
    return mutableData; });  
  
  if (transactionResult.committed) { _messagesRef.push().set({ 'title': title, 'check': check, 'type': type, 'other': other}); }}
```

3. Wyświetlanie danych odbywa się za pomocą FirebaseAnimatedList który automatycznie pobiera dane z bazy i wyświetla w ustalonej formie.
4. Usuwanie i Edytowanie odbywa się poprzez użycie podanych poniżej funkcji :

```
DBRef.child(userID).child(type).child(id).remove();  
  
DBRef.child(userID).child(type).child(id).update({  
  'title': myController.text,  
  'check': checkValue,  
  'type': dropdownValue,  
  'other': otherController.text });
```

5. Łączenie do api z nazwami filmów

```
Future<String> fetchWeather(String title, String type) async {  
  String Url = url + title;  
  
  final response = await http.get(Url);  
  
  if (response.statusCode == 200 && type == 'Movies') { setState(() {  
    suggestionController.text = Api.fromJson(json.decode(response.body)).Title; });  
    return Api.fromJson(json.decode(response.body)).Title;  
  } else { throw Exception('Failed to load weather'); }}
```

Najważniejsza część kodu to klasa Login i TabBarMain które zarządzają działaniem aplikacji i udzielają dostępu lub nie użytkownikowi.

```
class Login extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return DefaultTabController(  
      length: 3,  
      child: Scaffold(  
        appBar: AppBar( title: Text('Sign in panel'),),  
        body: SignInPage(),  
        floatingActionButton: FloatingActionButton(  
          child: Icon(Icons.add),  
          onPressed: () => _pushPage(context, RegisterPage()),),),));  
  }  
}  
  
class TabBarMain extends StatelessWidget {  
  String userID;  
  TabBarMain(this.userID);  
  
  @override  
  Widget build(BuildContext context) {  
    return DefaultTabController(  
      length: 3,
```

```

    child: Scaffold(
      appBar: AppBar(title: Text('Best Book/Game/Movie')),
      actions: <Widget>[
        IconButton(icon: const Icon(Icons.search), tooltip: 'Next choice',
          onPressed: () => _pushPage(context, Search(userID)),),],
      bottom: TabBar(tabs: [
        Tab(icon: Icon(Icons.book)),
        Tab(icon: Icon(Icons.videogame_asset)),
        Tab(icon: Icon(Icons.movie)),],),),
      body: TabBarView(children: [
        Column(
          children: <Widget>[
            Flexible(
              child: FirebaseAnimatedList(
                query: QReq('Books', userID).orderByChild('check'),
                itemBuilder: (BuildContext context, DataSnapshot snapshot,
                  Animation<double> animation, int index) {
                  return GestureDetector(
                    onLongPress: (){
                      Navigator.push(
                        context,
                          MaterialPageRoute(builder: (context) => EditRecord(
                            snapshot.value["title"],
                            snapshot.value["check"],
                            snapshot.value["type"],
                            snapshot.key,
                            userID,
                            snapshot.value["other"],
                            snapshot.value["type"]
                          )),

```

```

    ),
  ),
  child: Padding(
    padding: EdgeInsets.all(5.0),
    child: Container(
      padding: const EdgeInsets.all(5.0),
      decoration: BoxDecoration(
        border: Border.all(
          width: 1.0,
          color: Colors.black38
        ),
        borderRadius: BorderRadius.all(
          Radius.circular(10.0) // <--- border radius here
        ),
      ),
    ),
  ),
  // color: Colors.black12,
  child: Row(
    children: <Widget>[
      ikona(snapshot.value["check"]),
      Flexible(
        child: Container(
          margin: const EdgeInsets.only(left: 20.0),
          child: Text("${snapshot.value["title"]}",
            style: TextStyle(fontSize: 18.0),
            overflow: TextOverflow.ellipsis,
          ),
        ),
      ),
    ],
  ),
),
Column(
  children: <Widget>[
    Flexible(
      child: FirebaseAnimatedList(

```

```

        query: QReq('Games', userID),

        itemBuilder: (BuildContext context, DataSnapshot snapshot,
            Animation<double> animation, int index) {
            return GestureDetector(
                onLongPress: (){
                    Navigator.push(
                        context,
                        MaterialPageRoute(builder: (context) => EditRecord(
                            snapshot.value["title"],
                            snapshot.value["check"],
                            snapshot.value["type"],
                            snapshot.key,
                            userID,
                            snapshot.value["other"],
                            snapshot.value["type"]
                        )),
                    );
                },
                child: Padding(
                    padding: EdgeInsets.all(5.0),
                    child: Container(
                        padding: const EdgeInsets.all(5.0),
                        decoration: BoxDecoration(
                            border: Border.all(
                                width: 1.0,
                                color: Colors.black38
                            ),
                            borderRadius: BorderRadius.all(
                                Radius.circular(10.0) // <--- border radius here
                            ),
                        ),
                    ),
                ),
            );
        },
    );
}

```

[illegible]

```

        snapshot.value["type"]
      ),
    ),
  ),
},
    child: Padding(
      padding: EdgeInsets.all(5.0),
      child: Container(
        padding: const EdgeInsets.all(5.0),
        decoration: BoxDecoration(
          border: Border.all(
            width: 1.0,
            color: Colors.black38
          ),
          borderRadius: BorderRadius.all(
            Radius.circular(10.0) // <--- border radius here
          ),
        ),
        // color: Colors.black12,
        child: Row(
          children: <Widget>[
            ikona(snapshot.value["check"]),
            Flexible(
              child: Container(
                margin: const EdgeInsets.only(left: 20.0),
                child: Text("${snapshot.value["title"]}",
                  style: TextStyle(fontSize: 18.0),
                  overflow: TextOverflow.ellipsis,
                ),
              ),
            ),
          ],
        ),
        floatingActionButton: FloatingActionButton(
          child: Icon(Icons.add),

```

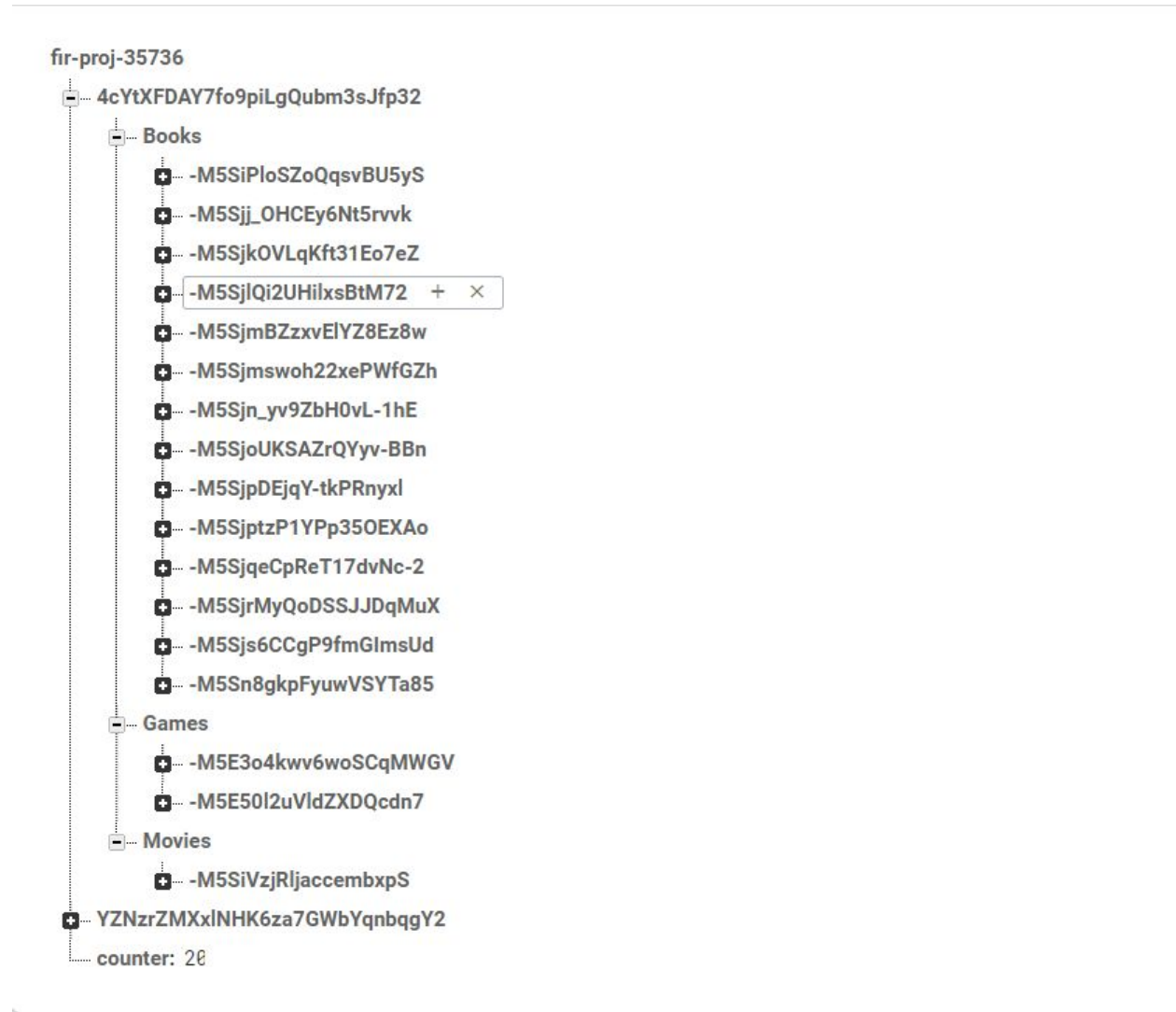


```

onPressed: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => AddNewRecord(userID)),
  );
}
}
}
}
}

```

Screenshots z działania aplikacji :



8:57

← Registration

Email

Password

Submit

8:58

Sign in panel

Email
hello@gmail.com

Password
qweqwe

Submit

+

qweqwe | qweqwe@gmail.com | qweqweqweqwe

q w e r t y u i o p

a s d f g h j k l

z x c v b n m

?123 , ☺ . ✓

8:59

← Best Book/Game/Movie

✓ Harry Potter and the Deathly Hallows: Par...

+

9:00

← Add new

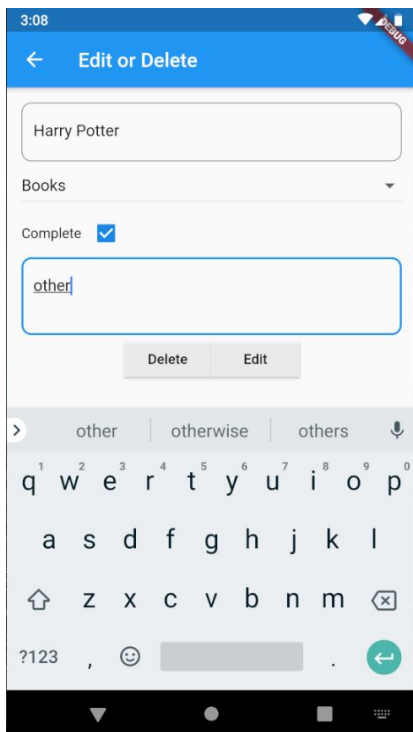
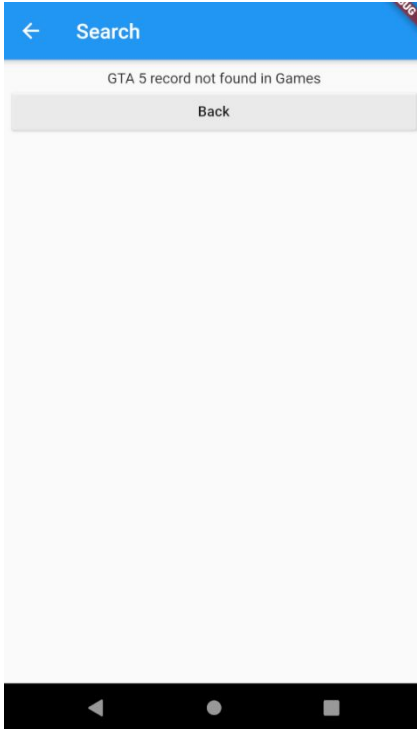
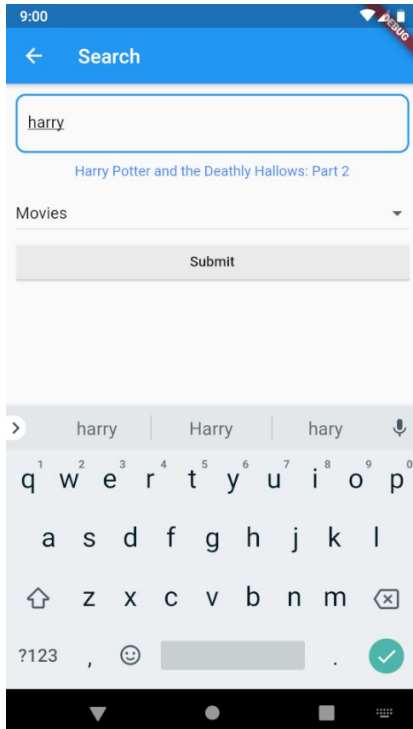
Title

Books

Complete ☐

Other

Add



Opinia własna autora :

Aplikacja ta została wykonana w technologii flutter i pozwoliła na lepsze zaznajomienie się z tą technologią oraz Firebasem. Uważam, że jeżeli aplikacja byłaby wciąż rozwijana można by było ją upublicznić.