

Ministerul Educației, Culturii și Cercetării al Republicii Moldova  
Universitatea Tehnică a Moldovei  
Departamentul Ingineria Software și Automatică

# **RAPORT**

Lucrare de laborator nr. 3, varianta II

Disciplina: PAM

Tema: Utilizare package state Management. Programarea asincrona.

A efectuat:

Mandiș Nichita TI-224

A verificat :

Buza D.

Chișinău 2025

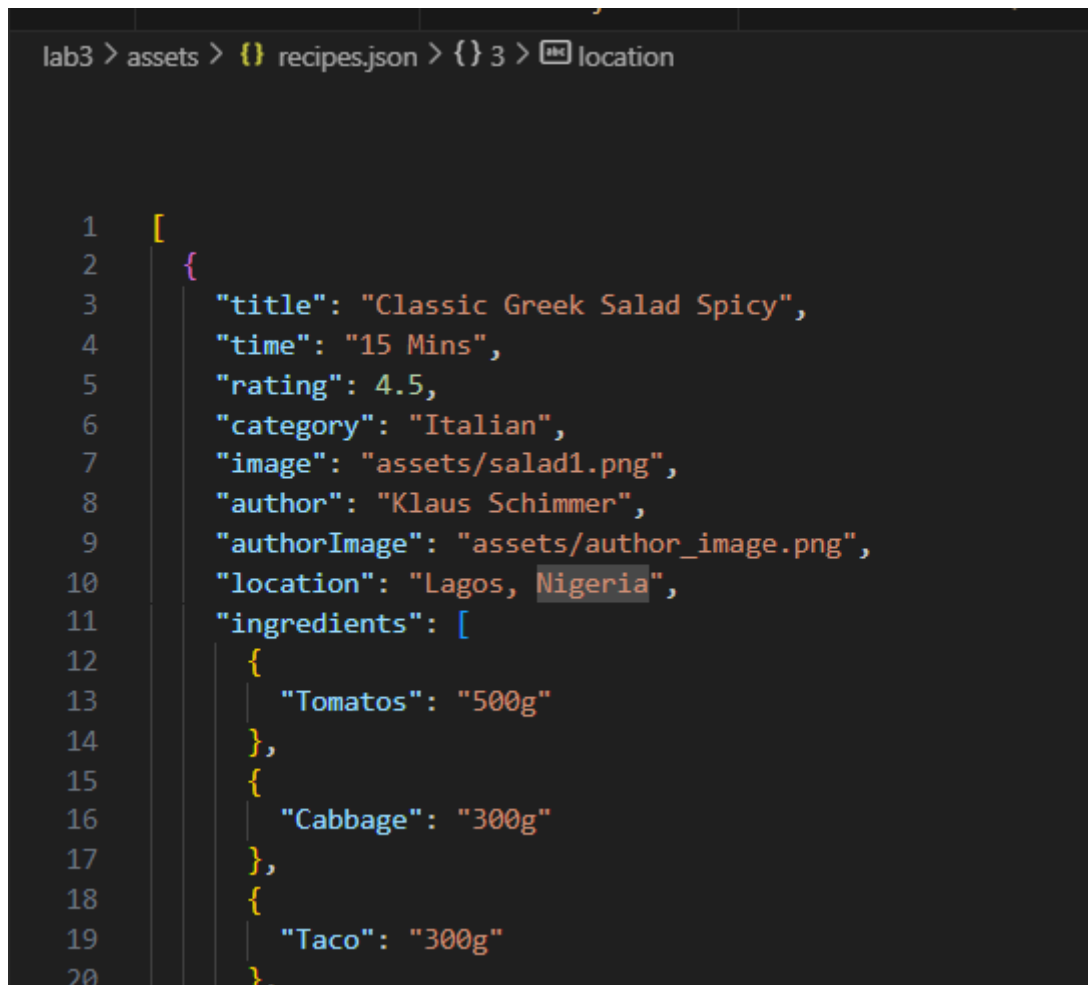
### Sarcina de lucru: Varianta 2.

Datele se vor incarca dintr-un fisier .json stocat in aplicatie. Să se utilizeze un package pentru State Management - BLoC sau GetX.

Ca baza interfeței UI a fost luat design-ul din laboratorul nr. 2.

### Mersul lucrării:

În fișierul assets/recipes.json am adăugat datele necesare.



```
lab3 > assets > {} recipes.json > {} 3 > location

1  [
2    {
3      "title": "Classic Greek Salad Spicy",
4      "time": "15 Mins",
5      "rating": 4.5,
6      "category": "Italian",
7      "image": "assets/salad1.png",
8      "author": "Klaus Schimmer",
9      "authorImage": "assets/author_image.png",
10     "location": "Lagos, Nigeria",
11     "ingredients": [
12       {
13         "Tomatos": "500g"
14       },
15       {
16         "Cabbage": "300g"
17       },
18       {
19         "Taco": "300g"
20     }
```

Figura 1 - assets/recipes.json

```

60 void onInit() {
61     super.onInit();
62     loadRecipes();
63 }
64
65 Future<void> loadRecipes() async {
66     isLoading(true);
67     final data = await rootBundle.loadString('assets/recipes.json');
68     final jsonResult = json.decode(data) as List;
69     recipes.value = jsonResult.map((e) => Recipe.fromJson(e)).toList();
70     isLoading(false);
71 }
72

```

Figura 2 - Încărcare asincronă din JSON

GetX este un framework pentru gestionarea stării, navigație și dependențe în aplicațiile Flutter. Acesta simplifică semnificativ logica aplicației prin eliminarea nevoii de metode tradiționale precum `setState()`, permițând actualizarea automată a interfeței grafice în momentul în care datele se modifică. Cu ajutorul variabilelor reactive (`.obs`) și al widgetului `Obx`, GetX oferă o arhitectură clară și eficientă, bazată pe reacții în timp real. De asemenea, include un sistem de injecție a dependențelor și un mecanism de rutare integrat, ceea ce face ca aplicațiile să fie mai rapide, mai curate și mai ușor de întreținut.

```

138 ), // AppBar
139 body: Obx(() {
140     if (controller.isLoading.value) {
141         return const Center(child: CircularProgressIndicator());
142     }
143

```

Figura 3 – Adăugarea unei iconițe Loading

`Obx` este un widget specific framework-ului GetX, utilizat pentru a reconstrui automat porțiuni din interfața grafică atunci când valorile reactive (declarații cu `.obs`) se modifică. Practic, `Obx` ascultă variabilele observabile dintr-un controller și actualizează în timp real doar acele elemente din UI care depind de ele, fără a reîncărca întreaga pagină. Acest mecanism face aplicația mai eficientă și mai receptivă.

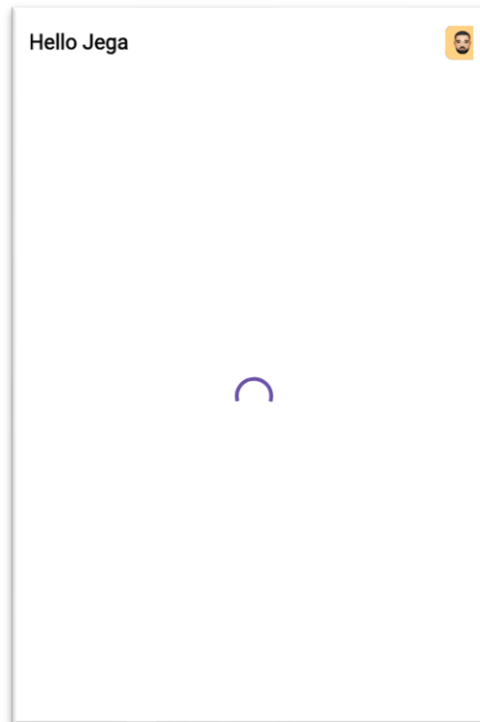


Figura 4 – Aplicația în stare de loading

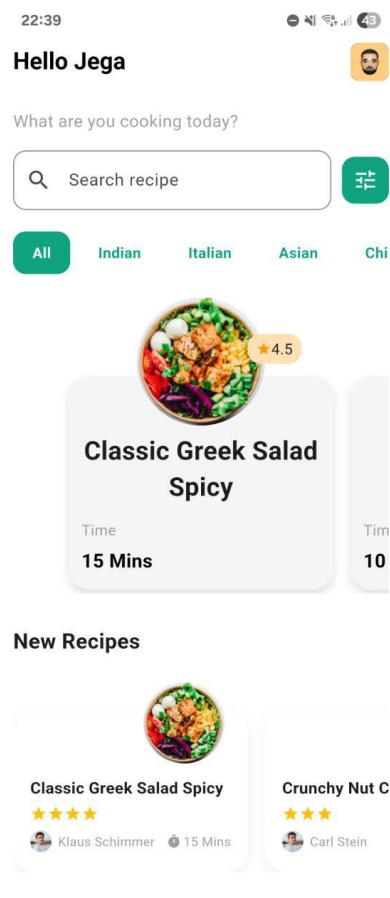


Figura 5 – Aplicația după încărcarea datelor

## Concluzie

Utilizarea Obx împreună cu framework-ul GetX oferă o metodă modernă, simplă și eficientă de gestionare a stării în Flutter. Obx permite actualizarea automată a interfeței grafice atunci când valorile reactive se modifică, eliminând nevoia de metode tradiționale precum `setState()`. Prin această abordare, codul devine mai curat, mai organizat și mai ușor de întreținut. În plus, GetX oferă un control clar al logicii aplicației prin controlere și o separare elegantă între interfață și date. În concluzie, Obx face ca aplicațiile Flutter să fie mai reactive și performante, asigurând o experiență de dezvoltare rapidă și intuitivă.