

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

RAPORT

Lucrare de laborator nr. 2, varianta II

Disciplina: PAM

Tema: Utilizare avansata de componente UI. Realizare app dupa
design definit complex.

A efectuat:

Russu Iulia TI-224

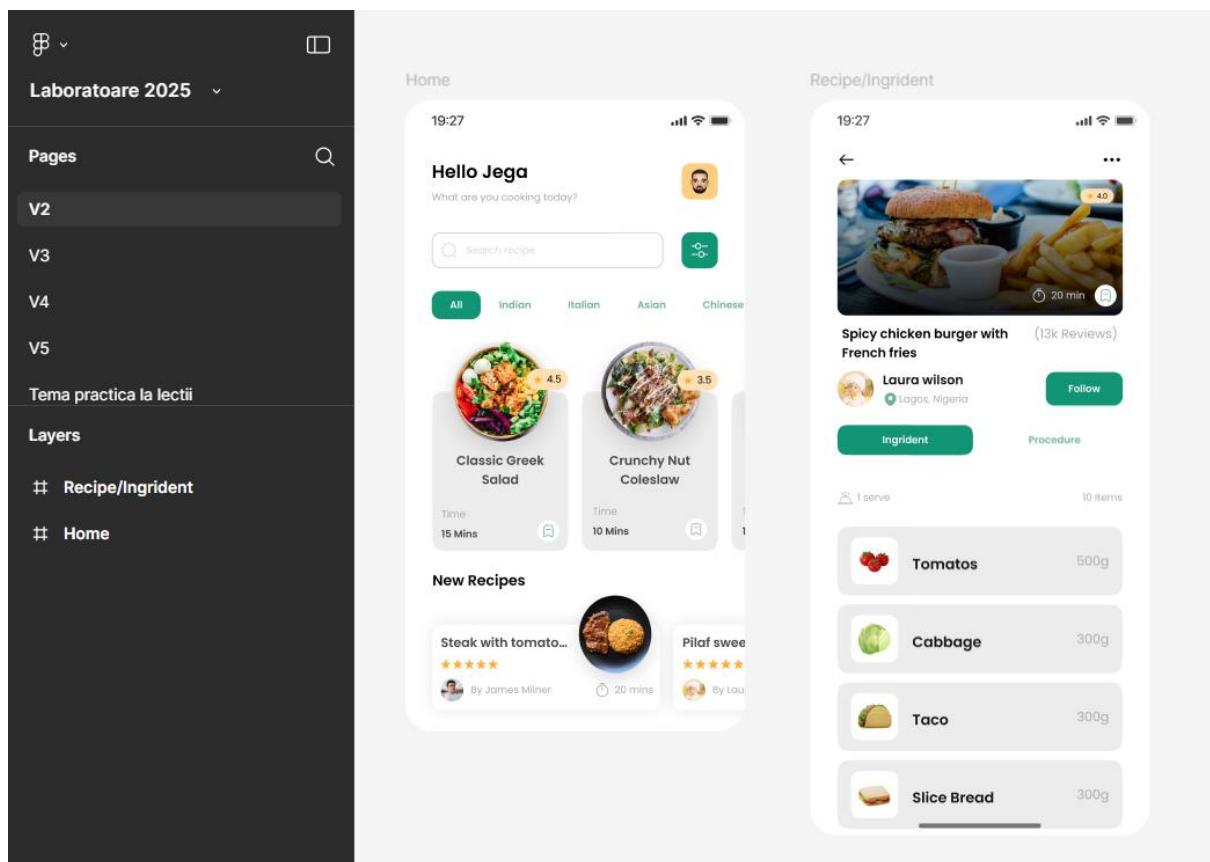
A verificat :

Buza D.

Chișinău 2025

Sarcina de lucru: Varianta 2.

Utilizare avansata de componente UI. Realizare app dupa design definit complex.
Designul prezentat în Figma:



Mersul lucrării:

Hello Jega



What are you cooking today?



Figura 1 – Secțiunea cu search bar

```

116 return Scaffold(
117   appBar: AppBar(
118     backgroundColor: Colors.white,
119     elevation: 0,
120     title: const Text(
121       'Hello Jega',
122       style: TextStyle(color: Colors.black, fontWeight: FontWeight.bold),
123     ), // Text
124     actions: [
125       Padding(
126         padding: const EdgeInsets.only(right: 16),
127         child: Container(
128           width: 36,
129           height: 36,
130           decoration: BoxDecoration(
131             color: Colors.grey,
132             borderRadius: BorderRadius.circular(8),
133             image: const DecorationImage(
134               image: AssetImage('assets/avatar.png'),
135               fit: BoxFit.cover,
136             ), // DecorationImage
137           ), // BoxDecoration
138         ), // Container
139       ), // Padding

```

Figura 1.2 – Titlul + imaginea cu avatar-ul user-ului

În acest fragment de cod sunt utilizate elemente pentru crearea unei interfețe grafice. Scaffold este structura de bază a unei pagini în Flutter și oferă cadrul principal în care pot fi plasate componente precum bara de sus, conținutul sau butoanele de navigare. AppBar reprezintă bara de titlu aflată în partea superioară a ecranului; în acest caz, are un fundal alb, fără umbră (elevation: 0) și conține un titlu și o acțiune în partea dreaptă. Text este utilizat pentru afișarea textului „Hello Jega”, având un stil personalizat — culoare neagră și text îngroșat. În interiorul actions, este folosit un Padding pentru a adăuga spațiu la dreapta, iar un Container cu proprietăți de BoxDecoration afișează un avatar, aplicând o culoare gri, colțuri rotunjite și o imagine din fișierul local assets/avatar.png. Astfel, aceste elemente lucrează împreună pentru a construi o bară de aplicație estetică și funcțională.

```

152 Row(
153   children: [
154     Expanded(
155       child: TextField(
156         decoration: InputDecoration(
157           hintText: "Search recipe",
158           prefixIcon: const Icon(Icons.search),
159           border: OutlineInputBorder(
160             borderRadius: BorderRadius.circular(12),
161             borderSide: BorderSide(
162               color: const Color(0xFF888888),
163               width: 1.0,
164             ), // BorderSide
165           ), // OutlineInputBorder
166           filled: true,
167           fillColor: Colors.white,
168           enabledBorder: OutlineInputBorder(
169             borderRadius: BorderRadius.circular(12),
170             borderSide: BorderSide(
171               color: const Color(0xFF888888),
172               width: 1.0,
173             ), // BorderSide
174           ), // OutlineInputBorder
175           focusedBorder: OutlineInputBorder(
176             borderRadius: BorderRadius.circular(12),
177             borderSide: BorderSide(
178               color: const Color(
179                 0xFF888888,
180               ), // Color
181               width: 2.0,
182             ), // BorderSide
183           ), // OutlineInputBorder
184         ), // InputDecoration
185       ), // TextField
186     ), // Expanded
187     const SizedBox(width: 10),
188     Container(
189       decoration: BoxDecoration(
190         color: Color(0xFF10A37F),

```

Figura 1.3 – SearchBar + butonul cu filtre

În acest fragment de cod sunt utilizate mai multe elemente UI pentru a construi secțiunea principală a paginii. SingleChildScrollView permite derularea conținutului pe

ecran, astfel încât toate elementele din interiorul său pot fi vizualizate chiar și pe ecrane mici. Proprietatea padding adaugă un spațiu uniform în jurul conținutului. În interior se află un Column, care aliniază elementele vertical, iar crossAxisAlignment: CrossAxisAlignment.start asigură alinierea lor la stânga. Primul element este un Text care afișează întrebarea „What are you cooking today?” cu o dimensiune mică și culoare gri. SizedBox este folosit pentru a adăuga spațiu între elemente. Urmează un Row, care aliniază orizontal un TextField și un Container cu o pictogramă. TextField este câmpul de căutare, având un InputDecoration ce include un indiciu (hintText), o pictogramă de căutare (prefixIcon), un fundal alb și margini rotunjite. OutlineInputBorder definește conturul câmpului atât pentru starea activă (focusedBorder) cât și pentru cea normală (enabledBorder). Alături, Container are un fundal verde, colțuri rotunjite și conține o pictogramă de tip Icon(Icons.tune), reprezentând un buton de filtrare. Toate aceste elemente împreună creează o interfață clară și modernă pentru introducerea și filtrarea rețetelor.

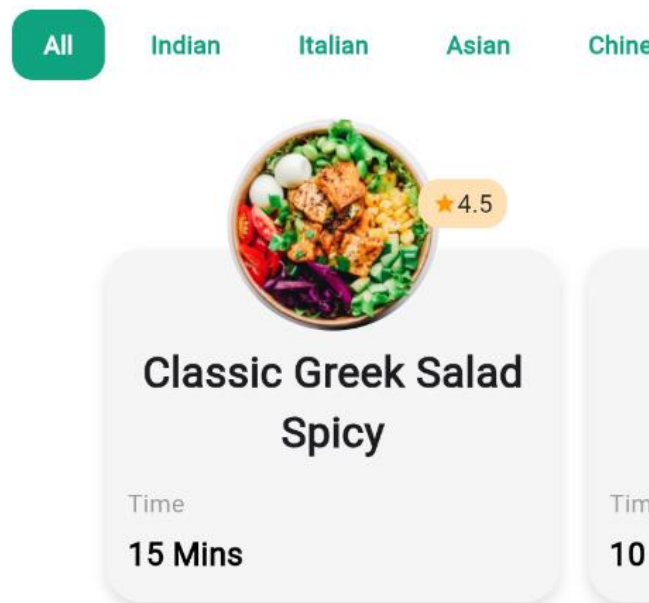


Figura 2 – Secțiunea cu meniul principal, cu butoane pentru categorii

```

43 final List<Map<String, dynamic>> recipes = [
44   {
45     'title': 'Classic Greek Salad Spicy',
46     'time': '15 Mins',
47     'rating': 4.5,
48     'category': 'Italian',
49     'image': 'assets/salad1.png',
50     'author': "Klaus Schimmer",
51     'authorImage': 'assets/author_image.png',
52     'location': "Lagos, Nigeria",
53     'ingredients': [
54       {"Tomatos": "500g"},
55       {"Cabbage": "300g"},
56       {"Taco": "300g"},
57       {"Slice Bread": "300g"},
58     ],
59   },
60   {
61     'title': 'Crunchy Nut Coleslaw',
62     'time': '10 Mins',
63     'rating': 3.5,
64     'category': 'Asian',
65     'image': 'assets/salad2.png',
66     'author': "Carl Stein",
67     'authorImage': 'assets/author_image.png',
68     'location': "Lagos, Nigeria",
69     'ingredients': [
70       {"Tomatos": "500g"},
71       {"Cabbage": "300g"},
72       {"Taco": "300g"},
73       {"Slice Bread": "300g"},
74     ],
75   },
76   {
77     'title': 'Steak with tomato sauce',

```

Figura 2.1 – Lista declarată cu toate elementele meniului (recete)

```

200 SizedBox(
201   height: 40,
202   child: ListView.separated(
203     scrollDirection: Axis.horizontal,
204     itemBuilder: (_, index) {
205       final cat = categories[index];
206       final isSelected = cat == selectedCategory;
207       return GestureDetector(
208         onTap: () {
209           setState(() => selectedCategory = cat);
210         },
211         child: Container(
212           padding: const EdgeInsets.symmetric(
213             horizontal: 18,
214             vertical: 8,
215           ), // EdgeInsets.symmetric
216           decoration: BoxDecoration(
217             color: isSelected ? Color(0xFF10A37F) : Colors.white,
218             borderRadius: BorderRadius.circular(12),
219           ), // BoxDecoration
220           child: Center(
221             child: Text(
222               cat,
223               style: TextStyle(
224                 color: isSelected
225                   ? Colors.white
226                   : Color(0xFF10A37F),
227                 fontWeight: isSelected
228                   ? FontWeight.bold
229                   : FontWeight.normal,
230               ), // TextStyle
231             ), // Text
232           ), // Center
233         ), // Container
234       ); // GestureDetector
235     },
236     separatorBuilder: (_, __) => const SizedBox(width: 8),

```

Figura 2.2 – Butoanele cu categorii

În acest fragment de cod este creat un rând de butoane orizontale care reprezintă categoriile de rețete, utilizând un `ListView.separated`. Acesta afișează o listă derulabilă

orizontal, unde fiecare element este separat de un spațiu de 8 pixeli, adăugat prin `separatorBuilder`. Containerul este inclus într-un `SizeBox` cu înălțimea de 40 de pixeli, pentru a controla dimensiunea întregului rând. În interiorul funcției `itemBuilder`, se obține fiecare categorie din lista `categories`, iar prin variabila `isSelected` se verifică dacă categoria curentă este cea selectată. Fiecare element este înfășurat într-un `GestureDetector`, care detectează atingerea utilizatorului și, prin metoda `setState`, actualizează categoria selectată. Vizual, fiecare categorie este reprezentată printr-un `Container` cu margini rotunjite și culori diferite în funcție de selecție: verde pentru cea activă și alb pentru celelalte. Textul din interiorul butonului își schimbă, de asemenea, culoarea în alb când este selectat și în verde când nu este.

```

243   SizeBox(
244     height: 280,
245     child: PageView.builder(
246       controller: PageController(viewportFraction: 0.75),
247       itemCount: filtered.length,
248       itemBuilder: (_, index) {
249         final recipe = filtered[index];
250         return GestureDetector(
251           onTap: () {
252             Navigator.push(
253               context,
254               MaterialPageRoute(
255                 builder: (_) => RecipeDetailScreen(recipe: recipe),
256               ), // MaterialPageRoute
257             );
258           },
259           child: AnimatedContainer(
260             duration: const Duration(milliseconds: 300),
261             margin: const EdgeInsets.symmetric(
262               horizontal: 8,
263               vertical: 6,
264             ), // EdgeInsets.symmetric
265             child: Stack(
266               clipBehavior: Clip.none,
267               children: [
268                 Positioned(
269                   top: 70,
270                   left: 0,
271                   right: 0,
272                   child: Container(
273                     width: 250,
274                     height: 280,
275                     decoration: BoxDecoration(
276                       color: Colors.grey.shade100,
277                       borderRadius: BorderRadius.circular(20),
278                       boxShadow: [
279

```

Figura 2.3 – Containerul cu scroll pentru elementele meniului

În acest fragment de cod este creat un carusel interactiv de rețete folosind `PageView.builder`, care permite derularea orizontală a cardurilor. Acesta este plasat într-un `SizeBox` cu înălțimea de 280 pixeli pentru a controla spațiul ocupat pe ecran. Proprietatea `viewportFraction: 0.75` face ca fiecare pagină să ocupe 75% din lățimea ecranului, oferind un efect vizual modern cu următoarea și precedenta rețetă parțial vizibile. Fiecare card este creat dinamic din lista `filtered`, care conține rețetele filtrate. Fiecare element este înconjurat de un `GestureDetector`, care detectează atingerea utilizatorului și, la apăsare, deschide o pagină nouă — `RecipeDetailScreen` — cu detaliile rețetei selectate, prin intermediul `Navigator.push`.

New Recipes

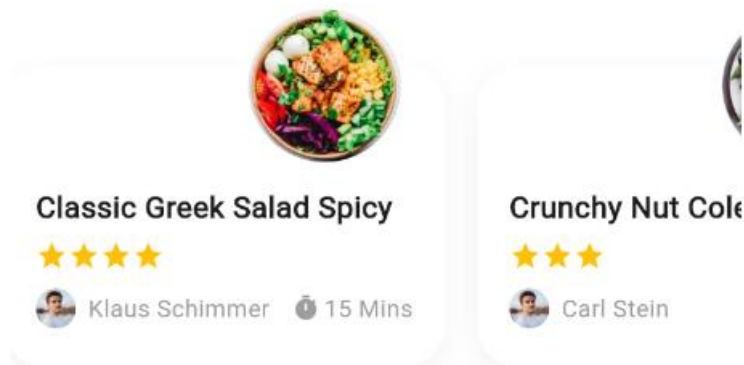


Figura 3 – Secțiunea “New Recipes”

```
380     const SizedBox(height: 30),
381     const Text(
382       "New Recipes",
383       style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
384     ), // Text
385     const SizedBox(height: 12),
386
387     SizedBox(
388       height: 190,
389       child: ListView.builder(
390         scrollDirection: Axis.horizontal,
391         itemCount: recipes.length,
392         itemBuilder: (_, index) {
393           final r = recipes[index];
394
395           return GestureDetector(
396             onTap: () {
397               Navigator.push(
398                 context,
399                 MaterialPageRoute(
400                   builder: (_) => RecipeDetailScreen(recipe: r),
401                 ), // MaterialPageRoute
402               );
403             },
404             child: Stack(
405               clipBehavior: Clip.none,
406               children: [
407                 Container(
408                   width: 220,
409                   margin: const EdgeInsets.only(right: 16, top: 40),
410                   decoration: BoxDecoration(
411                     color: Colors.white,
412                     borderRadius: BorderRadius.circular(20),
413                     boxShadow: [
414                       BoxShadow(
415                         color: Colors.black.withOpacity(0.05),
416                         blurRadius: 10,
417                         offset: const Offset(0, 4),
418                       ), // BoxShadow

```

Figura 3.1 – Secțiunea “New Recipes” cu scroll container

În acest fragment de cod este construită o secțiune intitulată „New Recipes”, care afișează o listă orizontală de carduri cu rețete noi. Secțiunea începe cu un Text evidențiat printr-un stil îngroșat și o dimensiune a fontului de 20, urmat de un spațiu vertical (SizedBox(height: 12)) pentru separare vizuală. Lista de rețete este creată cu ajutorul unui ListView.builder cu derulare orizontală, ce generează dinamic fiecare element din lista recipes.

Fiecare card de rețetă este un GestureDetector, care permite navigarea către o pagină de detaliu (RecipeDetailScreen) atunci când utilizatorul atinge cardul. Cardul propriu-zis este realizat cu un Container alb, rotunjit la colțuri și cu umbră subtilă, oferind un aspect elegant.

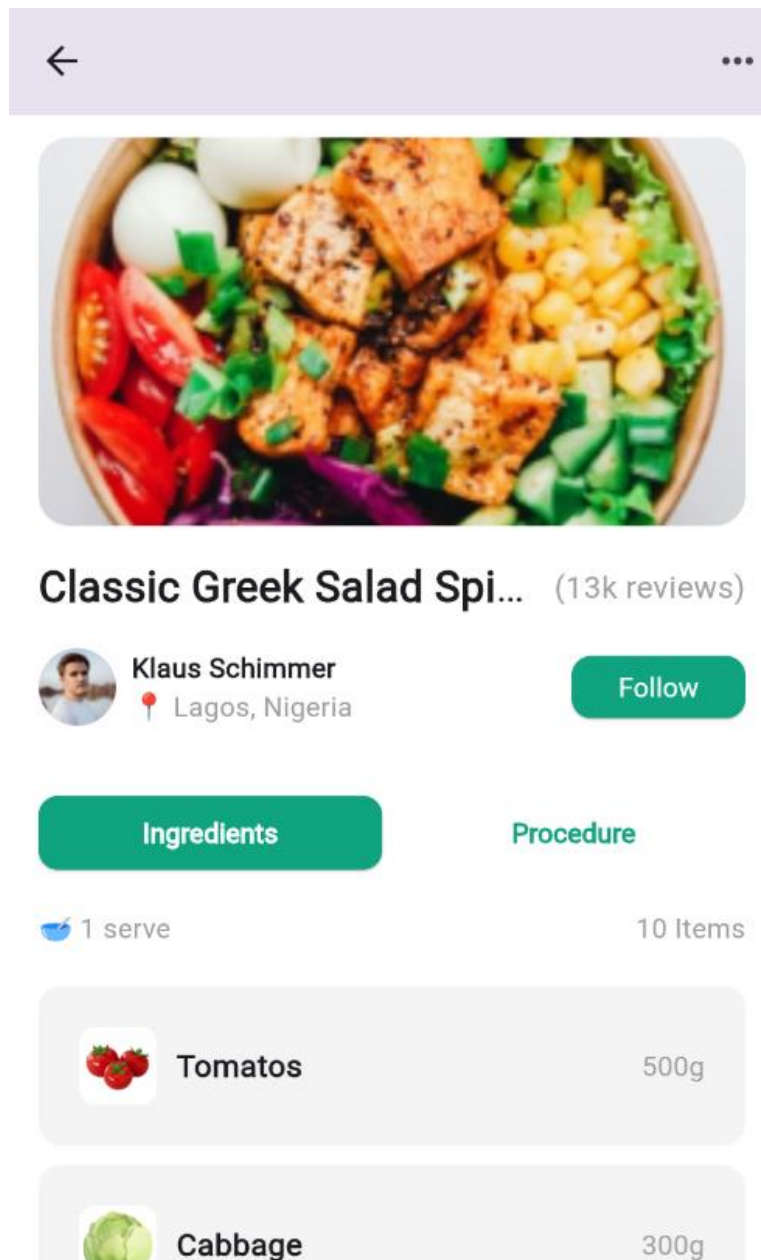


Figura 4 – Pagina cu detalii


```

514 class RecipeDetailScreen extends StatelessWidget {
515   final Map<String, dynamic> recipe;
516   const RecipeDetailScreen({super.key, required this.recipe});
517
518   @override
519   Widget build(BuildContext context) {
520     var selectedTab;
521     return Scaffold(
522       appBar: AppBar(
523         title: const Text(''),
524         actions: [
525           IconButton(icon: const Icon(Icons.more_horiz), onPressed: () {}),
526         ],
527       ), // AppBar
528       body: SingleChildScrollView(
529         padding: const EdgeInsets.all(16),
530         child: Column(
531           crossAxisAlignment: CrossAxisAlignment.start,
532           children: [
533             ClipRect(
534               borderRadius: BorderRadius.circular(16),
535               child: SizedBox(
536                 width: double.infinity,
537                 height: 200,
538                 child: ClipRect(
539                   borderRadius: BorderRadius.circular(16),
540                   child: Image.asset(recipe['image'], fit: BoxFit.cover),
541                 ), // ClipRect
542               ), // ClipRect
543             const SizedBox(height: 16),
544             Row(
545               mainAxisAlignment: MainAxisAlignment.spaceBetween,
546               children: [
547                 Expanded(
548                   flex: 9,
549                   child: Text(
550                     recipe['title'],
551                     style: const TextStyle(
552                       fontSize: 22,
553                       fontWeight: FontWeight.bold,
554

```

```

return GestureDetector(
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (_) => RecipeDetailScreen(recipe: r),
      ), // MaterialPageRoute
    );
  },
);

```

Figura 4.1 – Pagina cu detalii și acțiunea OnTap

Acest fragment de cod Flutter definește ecranul detaliat al unei rețete în aplicație, prin intermediul clasei `RecipeDetailScreen`, care este un `StatelessWidget` și primește un `Map<String, dynamic>` cu datele rețetei. Structura principală este oferită de `Scaffold`, care conține `AppBar` cu un buton de acțiune (`more_horiz`) și un `SingleChildScrollView` pentru a permite derularea verticală a întregului conținut.

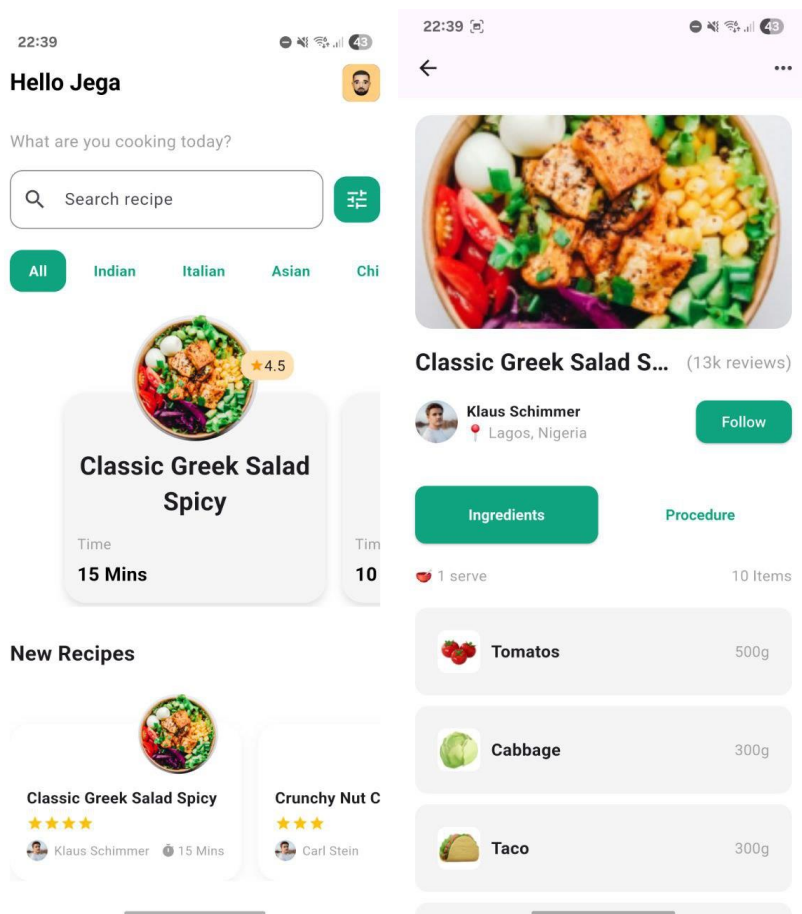


Figura 5 – Rezultatul final

Concluzie

Aplicația reușește să combine funcționalitatea și estetica într-un mod intuitiv și atractiv pentru utilizator. Prin utilizarea eficientă a widgeturilor precum Scaffold, AppBar, ListView, PageView, Container, Row, Column, TextField și ElevatedButton, interfața oferă o navigare ușoară între secțiuni, afișează informațiile rețetelor clar și permite interacțiuni dinamice, cum ar fi selectarea categoriilor, vizualizarea detaliilor rețetelor și derularea imaginilor într-un carusel. Designul responsiv, elementele vizuale suprapuse și butoanele interactive contribuie la o experiență plăcută și modernă, demonstrând modul în care Flutter facilitează dezvoltarea rapidă a aplicațiilor mobile complexe, cu o interfață estetică și funcționalitate completă.