

# OrderS

## Dokumentacija Sistema Preporuke

Razvoj softvera II (RS2) · Akademski 2024/2025

Merzuk Šišić · IB220060

### 1. Uvod

OrderS sistem za upravljanje kafićem implementira višeslojni sistem preporuke koji pomaže konobarima i administratorima u donošenju odluka o narudžbama i nabavci. Sistem koristi kombinaciju tri algoritma: preporuke zasnovane na popularnosti, preporuke zasnovane na vremenu i collaborative filtering na bazi korisničke historije.

Sistem preporuke je implementiran kao poseban servis na backend strani (.NET 9 Web API) i dostupan je kroz REST API endpointe. Flutter mobilna i desktop aplikacija konzumiraju ove endpoint-e putem RecommendationsProvider state management klase.

### 2. Arhitektura Sistema Preporuke

Komponenta	Putanja	Opis
IRecommendationService	OrdersAPI/Application/Interfaces/IRecommendationService.cs	Interface — definiše ugovor servisa
RecommendationService	OrdersAPI/Infrastructure/Services/RecommendationService.cs	Glavna logika — svi algoritmi
RecommendationsController	OrdersAPI/API/Controllers/RecommendationsController.cs	REST API controller — 3 endpointa
RecommendationsProvider	orders_mobile/lib/providers/notification_recommendation_providers.dart	Flutter state management
RecommendationsApiService	orders_mobile/lib/core/services/api/misc_api_services.dart	HTTP klijent za API pozive

### 3. Opis Algoritama

#### 3.1 Popularity-Based Filtering

Algoritam analizira historiju narudžbi za posljednjih 30 dana i rangira proizvode prema ukupnoj prodatoj količini. Koristi se kao osnovna metoda preporuke jer ne zahtijeva korisničku historiju i primjenjiva je za sve korisnike. Primarni kriterij sortiranja je ukupna prodana količina (TotalQuantity), a sekundarni je ukupan prihod (TotalRevenue) za razrješavanje jednakih slučajeva.

Putanja source koda:

```
OrdersAPI/Infrastructure/Services/RecommendationService.cs → metoda  
GetPopularProductsInternalAsync()
```

Implementacija:

```
var popularProductIds = await context.OrderItems .AsNoTracking() .Where(oi =>  
oi.Order.CreatedAt >= startDate && oi.Order.Status == OrderStatus.Completed)  
.GroupBy(oi => oi.ProductId) .Select(g => new { ProductId = g.Key, TotalQuantity =  
g.Sum(oi => oi.Quantity), TotalRevenue = g.Sum(oi => oi.Subtotal) })  
.OrderByDescending(x => x.TotalQuantity) .ThenByDescending(x => x.TotalRevenue)  
.Take(count) .Select(x => x.ProductId) .ToListAsync();
```

## 3.2 Time-Based Filtering

Algoritam prilagođava preporuke prema trenutnom dijelu dana, koristeći kategorije proizvoda kao proxy za vremensku relevantnost. Logika je zasnovana na uobičajenim obrascima konzumacije u ugostiteljskim objektima i segmentira dan u četiri perioda.

Vremenski period	Sati	Kategorije proizvoda
Doručak	06:00 – 10:59	Doručak, Kafa, Topli napici, Sokovi
Ručak	11:00 – 14:59	Glavna jela, Hrana, Sendviči, Burgeri
Poslijepodne	15:00 – 17:59	Deserti, Kafa, Torte, Slatkiši
Večer/Piće	18:00 – 23:59	Piće, Alkoholna pića, Bezalkoholna, Kokteli

Putanja source koda:

```
OrdersAPI/Infrastructure/Services/RecommendationService.cs → metode  
GetTimeBasedProductsInternalAsync() i GetCategoryFiltersForHour()
```

Implementacija:

```
private static List GetCategoryFiltersForHour(int hour) { if (hour >= 6 && hour < 11)  
return new List { "Doručak", "Kafa", "Topli napici", "Sokovi" }; if (hour >= 11 && hour  
< 15) return new List { "Glavna jela", "Hrana", "Sendviči", "Burgeri" }; if (hour >= 15  
&& hour < 18) return new List { "Deserti", "Kafa", "Torte", "Slatkiši" }; //  
Evening/Drinks (18-23h) return new List { "Piće", "Alkoholna pića", "Bezalkoholna  
pića", "Kokteli" }; }
```

## 3.3 User-Based Collaborative Filtering

Algoritam implementira item-based collaborative filtering pristup: pronalazi korisnike sa sličnom historijom narudžbi (koji su naručili iste proizvode) i preporučuje proizvode koje su ti slični korisnici

naručivali, a trenutni korisnik još nije. Svaki preporučeni proizvod dobiva score jednak broju sličnih korisnika koji su ga naručili.

Korak	Opis
1. Korisnikova historija	Dohvata sve ProductId-ove koje je korisnik naručio (distinct)
2. Slični korisnici	Pronalazi korisnike koji su naručili iste proizvode (max 20)
3. Kandidati za preporuku	Dohvata proizvode sličnih korisnika koje trenutni nije naručio
4. Rangiranje	Sortira po score-u (broj sličnih korisnika koji su naručili taj proizvod)
5. Rezultat	Vraća top 5 rangiranih dostupnih proizvoda

Putanja source koda:

```
OrdersAPI/Infrastructure/Services/RecommendationService.cs → metoda  
GetUserBasedRecommendationsInternalAsync()
```

Implementacija (ključni dio):

```
// 2. Find similar users (users who ordered the same products) var similarUserIds =  
await context.OrderItems .AsNoTracking() .Where(oi =>  
userProductIds.Contains(oi.ProductId) && oi.Order.WaiterId != userId) .Select(oi =>  
oi.Order.WaiterId) .Distinct() .Take(SIMILAR_USERS_LIMIT) // max 20 korisnika  
.ToListAsync(); // 3. Get products ordered by similar users but NOT by current user var var  
recommendedProductIds = await context.OrderItems .AsNoTracking() .Where(oi =>  
similarUserIds.Contains(oi.Order.WaiterId) && !userProductIds.Contains(oi.ProductId))  
.GroupBy(oi => oi.ProductId) .Select(g => new { ProductId = g.Key, Score = g.Count() //  
Broj sličnih korisnika koji su naručili }) .OrderByDescending(x => x.Score) .Take(5)  
.Select(x => x.ProductId) .ToListAsync();
```

## 4. Hibridni Sistem — Kombinacija Algoritama

Metoda GetRecommendedProductsAsync() kombinuje sve tri metode u jedinstven odgovor s ukupno do 5 preporuka (configurable). Redoslijed prioriteta:

Prioritet	Metoda	Broj stavki	Uvjet
1.	Time-Based	2 stavke	Uvijek se primjenjuje
2.	Popularity-Based	3 stavke	Popunjava do ukupno 5, bez duplikata
3.	Collaborative Filtering	2 stavke	Samo ako postoji userId
4.	Random Fallback	Preostalo	Ako nema dovoljno preporuka

Putanja source koda (hibridna metoda):

```
OrdersAPI/Infrastructure/Services/RecommendationService.cs → metoda  
GetRecommendedProductsAsync()
```

## 5. REST API Endpointi

Putanja source koda:

```
OrdersAPI/API/Controllers/RecommendationsController.cs
```

HTTP Metoda	Endpoint	Auth	Opis
GET	/api/Recommendations	Obavezan JWT	Hibridne personalizirane preporuke (userId iz tokena)
GET	/api/Recommendations/popular	Javni endpoint	Top 10 najpopularnijih proizvoda (30 dana)
GET	/api/Recommendations/time-based	Javni endpoint	Preporuke zasnovane na trenutnom satu

## 6. Flutter Integracija

Flutter aplikacija konzumira sistem preporuke kroz Provider pattern. RecommendationsProvider upravlja stanjem i asinhronim pozivima API-a:

Putanja source koda:

```
orders_mobile/lib/providers/notification_recommendation_providers.dart → klasa RecommendationsProvider
```

```
orders_mobile/lib/core/services/api/misc_api_services.dart → klasa RecommendationsApiService
```

Metode RecommendationsProvider-a:

Metoda	Opis
fetchRecommendedProducts()	Dohvata personalizirane preporuke za prijavljenog korisnika
fetchPopularProducts()	Dohvata top 10 najpopularnijih proizvoda
fetchTimeBasedRecommendations()	Dohvata preporuke na osnovu trenutnog sata
fetchAllRecommendations()	Paralelno poziva sve tri metode (Future.wait)

Primjer korištenja u Flutteru:

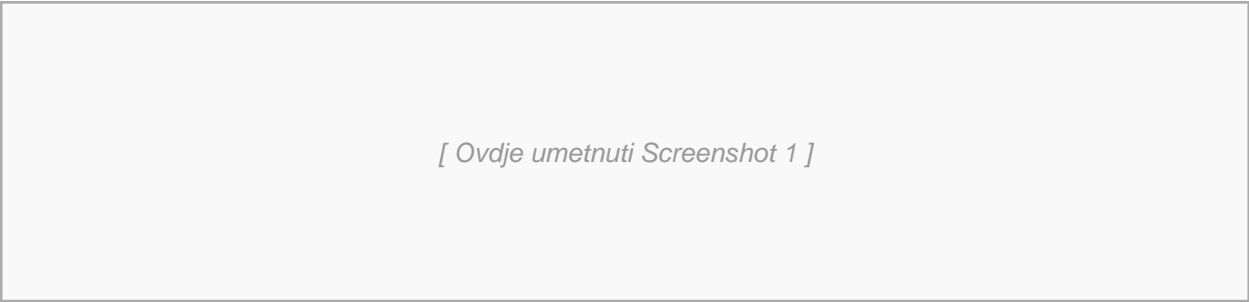
```
// Inicijalizacija u ekranu final provider = context.read(); await
provider.fetchAllRecommendations(userId: currentUserId); // Prikaz u UI-u Consumer(
builder: (context, provider, child) { if (provider.isLoading) return
CircularProgressIndicator(); return ListView.builder( itemCount:
provider.recommendedProducts.length, itemBuilder: (context, index) =>
ProductCard(product: provider.recommendedProducts[index]), ); }, )
```

## 7. Source Code — Screenshotovi

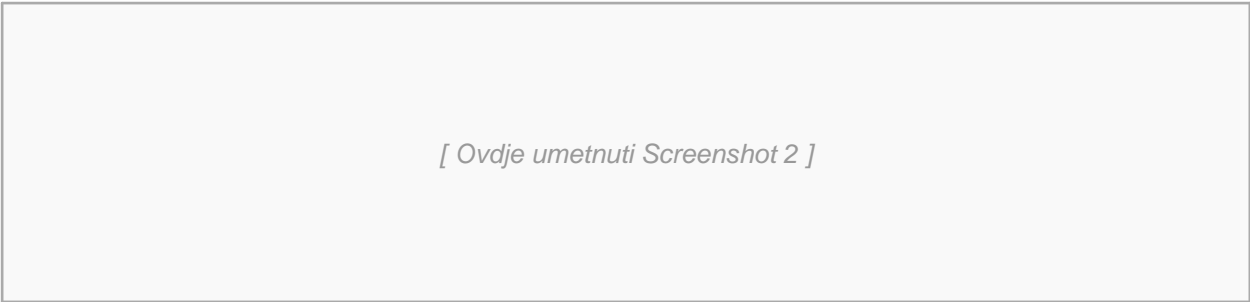
Napomena: Screenshotovi source koda i pokrenute aplikacije su priloženi u nastavku. Potrebno ih je dodati u dokument prema uputama profesora.

Screenshot	Šta prikazati	Putanja fajla
Screenshot 1	RecommendationService.cs — metoda GetRecommendations() u IDE-u (Visual Studio / VS Code)	OrdersAPI/API/Services/RecommendationService.cs
Screenshot 2	RecommendationService.cs — metoda GetPopularProducts() u IDE-u (Visual Studio / VS Code)	OrdersAPI/API/Services/RecommendationService.cs
Screenshot 3	RecommendationsController.cs — sva tri endpointa	OrdersAPI/API/Controllers/RecommendationsController.cs
Screenshot 4	Flutter: notification_recommendation_providers.dart — RecommendationProviders klasa	notification_recommendation_providers.dart
Screenshot 5	Pokrenuta aplikacija — ekran gdje se prikazuju preporuke proizvoda (emulatori i fizični uređaji)	Flutter izlazna aplikacija (emulatori i fizični uređaji)
Screenshot 6	Swagger UI — GET /api/Recommendations endpoint s response-om koji vraća listu preporučenih proizvoda	https://localhost:8080/swagger

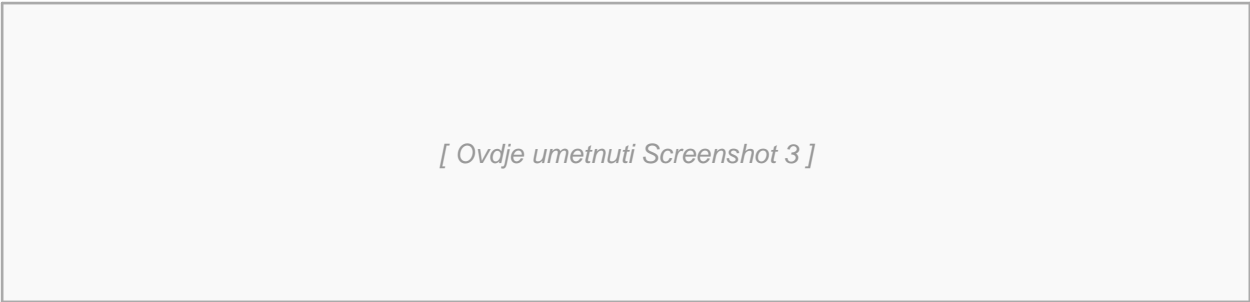
### Screenshot 1:



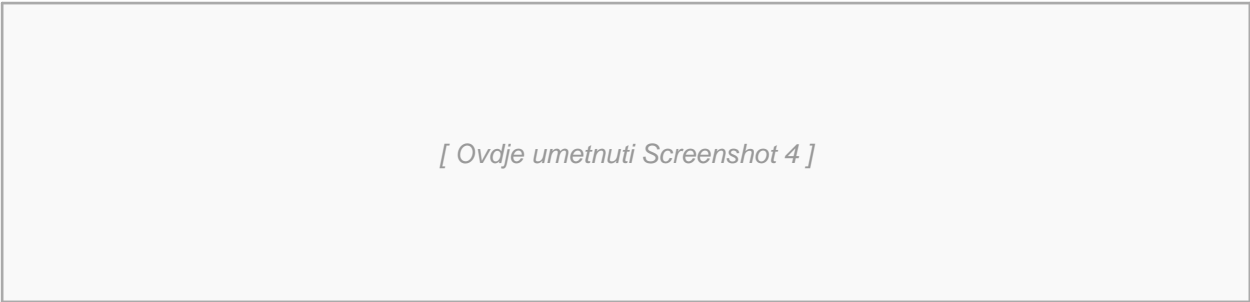
**Screenshot 2:**



**Screenshot 3:**

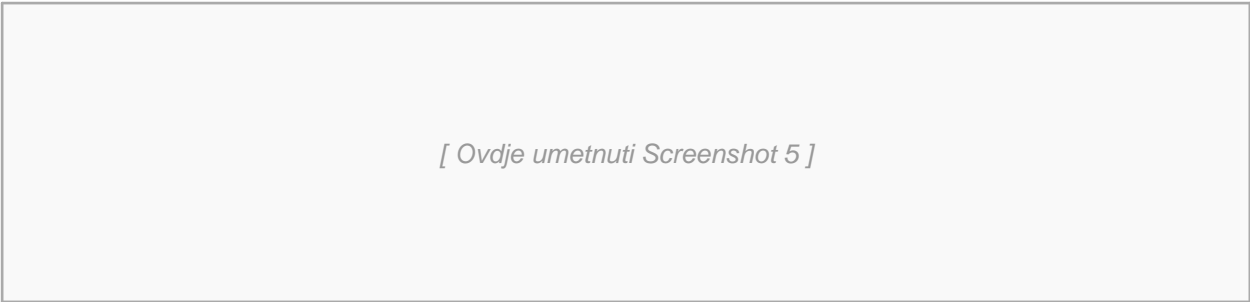


**Screenshot 4:**





**Screenshot 5:**



### Screenshot 6:

[ Ovdje umetnuti Screenshot 6 ]