4Paw – Recommender dokumentacija

1. Opis sistema preporuke

Sistem preporuke zasnovan je na agregaciji podataka o terminima, prihodima i aktivnostima korisnika. Umjesto kompleksnog ML, koristi „rule-based" logiku:

- Grupisanje završenih termina po usluzi i zbrajanje prihoda/broja termina.

- Rangiranje (sortiranje opadajuće) → „Top usluge".

- Dodatno: „Top klijenti" (po broju termina/utrošku).

Primjena:

- Veterinaru: daje uvid u vlastite top usluge (gdje je najveća potražnja/učinak).

- Adminu: prikazuje globalne top usluge i prihode, što služi kao preporuka za poslovne odluke (fokus na isplativije usluge).

Prikaz:

- Desktop dashboard (pie chart „Prihod po uslugama", tabela „Usluge po prihodu", lista „Najbolji klijenti").

- Veterinarske statistike (moje top usluge, moji dnevni termini).

2. Opis implementacije

Tehnologije:

- Backend: ASP.NET Core Web API, Entity Framework Core, SQL.

- UI: Flutter (desktop i mobile), Dio HTTP klijent, Provider.

- Shared: veterinarska_shared biblioteka (servisi i modeli).

Ključni API endpointi:

GET /financial/veterinarian/my-stats

GET /financial/veterinarian/daily-appointments

GET /financial/veterinarian/top-services

GET /financial/admin/financial-summary

GET /financial/admin/revenue-by-services

Algoritam:

- Upit nad Appointments sa filterom Completed.

- Projekcija na ime usluge/razlog, suma prihoda i broj termina.

- Grupisanje, sortiranje, izračun procenta po usluzi (u UI).

Sigurnost:

- JWT autentikacija; [RoleRequired] atribute štite rute (npr. „top-services" samo za Veterinarian).

```
public class RoleRequiredAttribute : Attribute, IAuthorizationFilter
{
    private readonly UserRole[] _requiredRoles;

    public RoleRequiredAttribute(params UserRole[] requiredRoles)
    {
        _requiredRoles = requiredRoles;
    }

    public void OnAuthorization(AuthorizationFilterContext context)
    {
        // Check if user is authenticated
        if (!context.HttpContext.User.Identity?.IsAuthenticated ?? true)
        {
            context.Result = new UnauthorizedResult();
            return;
        }

        // Get user role from claims
        var roleClaim = context.HttpContext.User.FindFirst(ClaimTypes.Role)?.Value;

        if (string.IsNullOrEmpty(roleClaim) || !Enum.TryParse<UserRole>(roleClaim, out var userRole))
        {
            context.Result = new ForbidResult();
            return;
        }

        // Check if user has required role
        if (!_requiredRoles.Contains(userRole))
        {
            context.Result = new ForbidResult();
            return;
        }
    }
}
```

Komunikacija:

- UI poziva API kroz veterinarska_shared/lib/services/financial_service.dart, ubacuje token u Authorization: Bearer ....

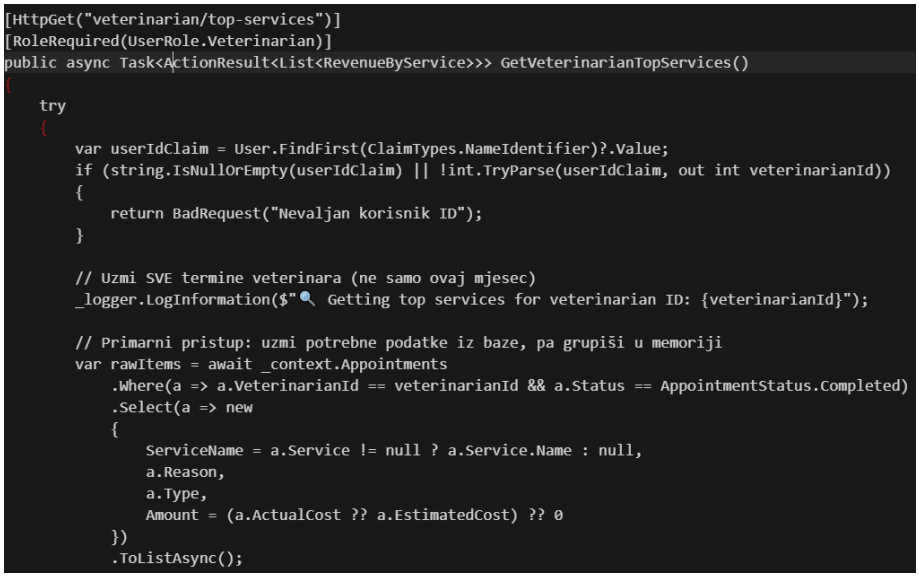
```
Future<VeterinarianStats> getVeterinarianStats() async {
  try {
    final token = await serviceLocator.authService.getAccessToken();

    if (token == null || token.isEmpty) {
      throw Exception('Niste prijavljeni. Molimo prijavite se ponovo.');
    }

    // Učitaj osnovne statistike
    final statsResponse = await serviceLocator.apiClient.dio.get(
      '/financial/veterinarian/my-stats',
      options: Options(
        headers: {
          'Authorization': 'Bearer $token',
          'Content-Type': 'application/json',
        },
      ),
    );
```

```
// Učitaj top usluge
final servicesResponse = await serviceLocator.apiClient.dio.get(
  '/financial/veterinarian/top-services',
  options: Options(
    headers: {
      'Authorization': 'Bearer $token',
      'Content-Type': 'application/json',
    },
  ),
);
```

3. Putanje i screenshot glavne logike (backend)

Glavna logika „Top usluge” (veterinar) – kontroler finansija:

```
[HttpGet("veterinarian/top-services")]
[RoleRequired(UserRole.Veterinarian)]
public async Task<ActionResult<List<RevenueByService>>> GetVeterinarianTopServices()
{
    try
    {
        var userIdClaim = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;
        if (string.IsNullOrEmpty(userIdClaim) || !int.TryParse(userIdClaim, out int veterinarianId))
        {
            return BadRequest("Nevaljan korisnik ID");
        }
```



```
[HttpGet("veterinarian/top-services")]
[RoleRequired(UserRole.Veterinarian)]
public async Task<ActionResult<List<RevenueByService>>> GetVeterinarianTopServices()
{
    try
    {
        var userIdClaim = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;
        if (string.IsNullOrEmpty(userIdClaim) || !int.TryParse(userIdClaim, out int veterinarianId))
        {
            return BadRequest("Nevaljan korisnik ID");
        }

        // Uzmi SVE termine veterinara (ne samo ovaj mjesec)
        _logger.LogInformation($"🔍 Getting top services for veterinarian ID: {veterinarianId}");

        // Primarni pristup: uzmi potrebne podatke iz baze, pa grupiši u memoriji
        var rawItems = await _context.Appointments
            .Where(a => a.VeterinarianId == veterinarianId && a.Status == AppointmentStatus.Completed)
            .Select(a => new
            {
                ServiceName = a.Service != null ? a.Service.Name : null,
                a.Reason,
                a.Type,
                Amount = (a.ActualCost ?? a.EstimatedCost) ?? 0
            })
            .ToListAsync();
```

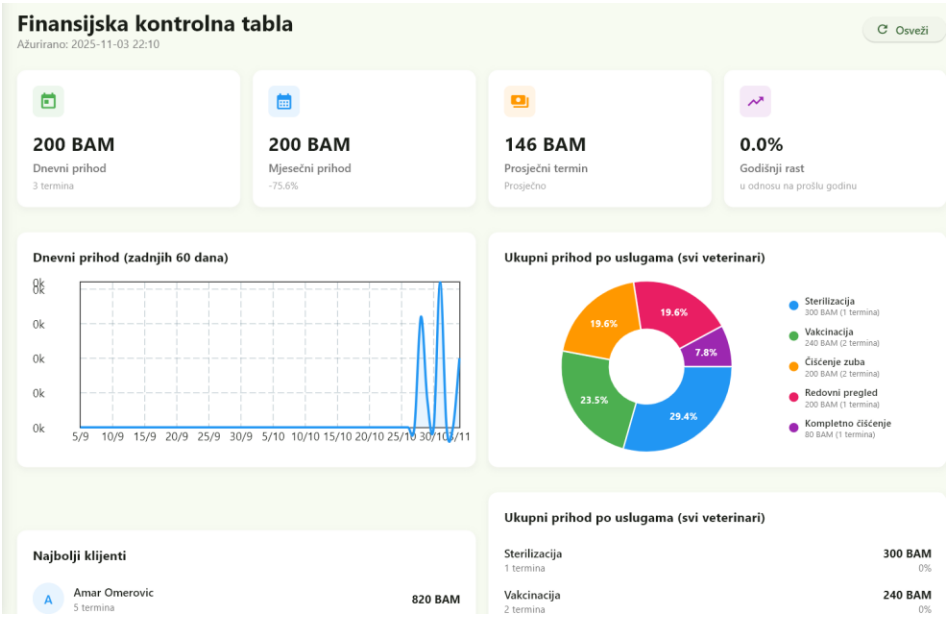
4. Putanja i screenshot orkestracije na UI-u (shared servis)

FinancialService.getVeterinarianStats – dohvaća my-stats, daily-appointments, top-services i spaja u jedan objekat:

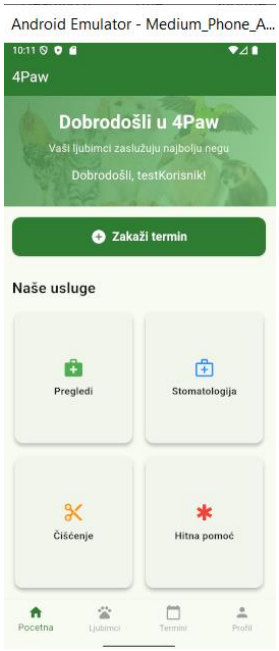
```
// Primarni pristup: uzmi potrebne podatke iz baze, pa grupiši u memoriji
var rawItems = await _context.Appointments
    .Where(a => a.VeterinarianId == veterinarianId && a.Status == AppointmentStatus.Completed)
    .Select(a => new
    {
        ServiceName = a.Service != null ? a.Service.Name : null,
        a.Reason,
        a.Type,
        Amount = (a.ActualCost ?? a.EstimatedCost) ?? 0
    })
    .ToListAsync();

string MapType(AppointmentType t)
{
    return t == AppointmentType.Checkup ? "Pregled" :
           t == AppointmentType.Vaccination ? "Vakcinacija" :
           t == AppointmentType.Surgery ? "Operacija" :
           t == AppointmentType.Emergency ? "Hitno" :
           t == AppointmentType.Grooming ? "Šišanje/Njega" :
           t == AppointmentType.Dental ? "Stomatologija" :
           t == AppointmentType.Consultation ? "Konsultacija" :
           t == AppointmentType.FollowUp ? "Kontrola" : t.ToString();
}

var topServices = rawItems
    .Select(x => new
    {
        Name = !string.IsNullOrWhiteSpace(x.ServiceName)
            ? x.ServiceName!
            : (!string.IsNullOrWhiteSpace(x.Reason) ? x.Reason! : MapType(x.Type)),
        x.Amount
    })
    .GroupBy(x => x.Name)
    .Select(g => new RevenueByService
    {
        ServiceName = g.Key,
        Revenue = g.Sum(i => i.Amount),
        Count = g.Count()
    })
    .OrderByDescending(r => r.Revenue)
    .Take(5)
    .ToList();
```

5. Putanja i screenshot iz pokrenute aplikacije (vizuelne preporuke)

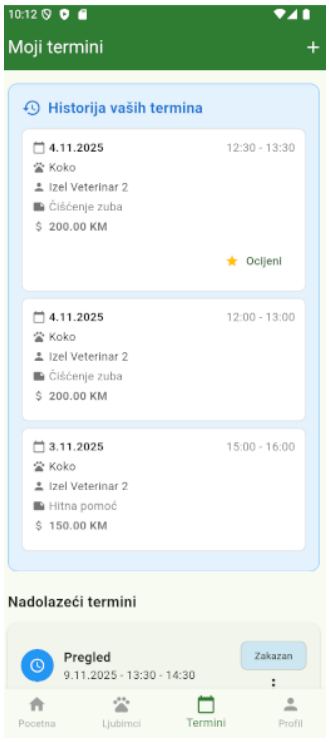
Desktop finansijski dashboard – pie chart i tabela „Usluge po prihodu”:



6. Početna (Mobile): veterinarska_mobile/lib/screens/home/dashboard_screen.dart – marketinški prikaz i CTA „Zakaži termin”.



7. Lista termina + ocjenjivanje: veterinarska_mobile/lib/screens/appointments/appointments_list_screen.dart.



8. Zaključak

Recommender u ovoj verziji je „explainable” i lagan: sve se može objasniti kroz izvještaje. Kasnije se može nadograditi na CF/CBF modele (npr. preporuke veterinara korisniku prema istoriji, vrstama ljubimaca i ocjenama).