**Lastfm Servisi**

LastfmService sınıfı, Music Tracker uygulamsında Last.fm API’si ile entegrasyonunu sağlayan servis katmanıdır. Bu servis, uygulamaya dış kaynaklı müzik verisi ekleyerek şarkıların detaylı özelliklerine ve benzer parça listelerine erişilmesine olanak tanır. ILastfmService ara yüzü üzerinden tanımlanan bu servis, iki temel işlevi yerine getirir: Spotify üzerinden alınan bir parçaya karşılık gelen Last.fm şarkı bilgilerini getirmek ve bir şarkıya benzer diğer şarkıların listesini almak.

**Servis Metotları:**

**GetLastfmTrackAsync Metodu**

GetLastfmTrackAsync metodu, Spotify’dan elde edilen bir parçaya karşılık gelen Last.fm şarkı bilgilerini almak için kullanılır. Metot, Last.fm API’sinin track.getInfo uç noktasını kullanarak şarkının başlık, sanatçı, albüm, kapak görselleri, mbid ve tür gibi detaylı bilgileri sorgular.

Kodun Genel Akışı:

1. Parametre olarak alınan SpotifyTrack nesnesinden şarkı adı (Title) ve sanatçı adı (Artist) çıkarılır.
2. Last.fm API çağrısı için gerekli URL oluşturulur.
   * Temel URL (BaseURL)
   * method=track.getInfo
   * API anahtarı (api\_key)
   * Şarkı ve sanatçı bilgileri (artist, track) URL güvenli biçimde encode edilir.
3. HTTP GET isteği ile belirtilen URL’e istek gönderilir ve sonuç JSON formatında alınır.
4. Yanıt JSON olarak parse edilir (JsonDocument.Parse).
5. Yanıttaki “track” alanı kontrol edilir.
   * Eğer veri yoksa null döndürülür.
   * Veri varsa LastfmTrack nesnesi oluşturulur ve şu bilgilerle doldurulur.
     + mbid, name, artist, album, genres, coverImages
6. SpotifyTrack üzerinden gelen Id ve Duration bilgileri de LastfmTrack nesnesine eklenir.
7. Son olarak oluşturulan LastfmTrack nesnesi döndürülür.
8. Hata oluşması durumunda hata mesajı loglanır ve boş bir LastfmTrack nesnesi döndürülür.

Kod Bloğu:

|  |
| --- |
| public async Task<LastfmTrack> GetLastfmTrackAsync(SpotifyTrack spotifyTrack)  {  try  {  var artist = spotifyTrack.Artist;  var track = spotifyTrack.Title;  var url = $"{BaseUrl}?method=track.getInfo" +  $"&api\_key={\_lastfmApiKey}" +  $"&artist={Uri.EscapeDataString(artist)}" +  $"&track={Uri.EscapeDataString(track)}" +  "&format=json";  var response = await \_httpClient.GetStringAsync(url);  Console.WriteLine($"Response: {response}");  var jsonDocument = JsonDocument.Parse(response);  if (!jsonDocument.RootElement.TryGetProperty("track", out JsonElement trackElement))  {  Console.WriteLine("No track data found.");  return null;  }  var lastfmTrack = new LastfmTrack  {  Mbid = trackElement.TryGetProperty("mbid", out var mbidProp) ? mbidProp.GetString() : null,  Title = trackElement.TryGetProperty("name", out var nameProp) ? nameProp.GetString() : null,  Artist = trackElement.TryGetProperty("artist", out var artistElement)  ? new Artist  {  Mbid = artistElement.TryGetProperty("mbid", out var artistMbidProp) ? artistMbidProp.GetString() : null,  Name = artistElement.TryGetProperty("name", out var artistNameProp) ? artistNameProp.GetString() : null  }  : null,  Album = trackElement.TryGetProperty("album", out var albumElement)  ? new Album  {  Mbid = albumElement.TryGetProperty("mbid", out var albumMbidProp) ? albumMbidProp.GetString() : null,  Title = albumElement.TryGetProperty("title", out var albumTitleProp) ? albumTitleProp.GetString() : null,  CoverImages = albumElement.TryGetProperty("image", out var imagesElement)  ? imagesElement.EnumerateArray()  .Select(img => new CoverImage  {  Url = img.TryGetProperty("#text", out var urlProp) ? urlProp.GetString() : null,  Size = img.TryGetProperty("size", out var sizeProp) ? sizeProp.GetString() : null  })  .Where(c => !string.IsNullOrEmpty(c.Url)) .ToList()  : new List<CoverImage>()  }  : null,  Genres = trackElement.TryGetProperty("toptags", out var toptagsElement)  && toptagsElement.TryGetProperty("tag", out var tagsArrayElement)  ? tagsArrayElement.EnumerateArray()  .Select(tag => tag.TryGetProperty("name", out var tagNameProp) ? tagNameProp.GetString() : null)  .Where(name => !string.IsNullOrEmpty(name))  .ToList()  : new List<string>()  };  lastfmTrack.SpotifyId = spotifyTrack.Id;  lastfmTrack.Duration = spotifyTrack.Duration;  return lastfmTrack;  }  catch (Exception ex)  {  Console.WriteLine($"Error retrieving track from Last.fm: {ex.Message}");  return new();  }  } |

**GetSimilarTracksAsync Metodu**

GetSimilarTracksAsync metodu, belirli bir parçaya benzer olan şarkıları Last.fm API üzerinden sorgulamak amacıyla kullanılır. Metot, Last.fm’in track.getsimilar yöntemini çağırarak, gelen JSON yanıtındaki benzer parça bilgilerini LastfmTrack modeline dönüştürür ve liste olarak geri döndürür.

Kodun Genel Akışı:

1. Parametre olarak gelen mbid, trackName, ve artistName değerlerinden uygun olan sorgu parametresi oluşturulur.
   * Eğer mbid verilmişse, sorgu bu kimlik ile yapılır.
   * Aksi halde artist ve track parametreleri kullanılır.
2. Last.fm API isteiği için URL oluşturulur.
   * method= track.getsimilar
   * api\_key, mbid veya track + artist, limit ve format=json
3. HttpClient aracılığıyla oluşturulan URL’e GET isteği gönderilir.
4. Yanıtın başarılı olup olmadığı kontrol edilir.
   * Yanıt başarısızsa hata mesajı yazılır ve null döndürülür.
5. Başarılı yanıt içeriği JSON olarak okunur ve parse edilir.
6. JSON içinde “similartracks” -> “track” dizisi döngü ile taranır.
   * Her track nesnesi içerisinden:
     + mbid, name, duration, artsit bilgileri alınır.
     + artist nesnesi varsa alt alanları (mbid, name) alınır.
     + Kapar görselleri “image” dizisinden çekilir ve Album.CoverImages alanına atanır.
     + “toptags” alanı varsa şarkının tür bilgileri (Genres) alınır.
   * Elde edilen LastfmTrack nesnesi listeye eklenir.
7. Tüm parçalar tarandıktan sonra, oluşturulan List<LastfmTrack> döndürülür.
8. Hata oluşması durumunda: Hata mesajı konsola loglanır ve boş bir List<LastfmTrack> döndürülür.

Kod Bloğu:

|  |
| --- |
| public async Task<List<LastfmTrack>> GetSimilarTracksAsync(string? mbid, string? trackName, string? artistName, int limit)  {try{  var queryParams = !string.IsNullOrEmpty(mbid) ? $"&mbid={Uri.EscapeDataString(mbid)}&limit={limit}" : $"&artist={Uri.EscapeDataString(artistName)}&track={Uri.EscapeDataString(trackName)}&limit={limit}";  var url = $"{BaseUrl}?method=track.getsimilar" +  $"&api\_key={\_lastfmApiKey}" +  queryParams +  "&format=json";  Console.WriteLine(url);  HttpResponseMessage response = await \_httpClient.GetAsync(url);  if (!response.IsSuccessStatusCode)  {  Console.WriteLine($"Status:{response.StatusCode} Request:{response.RequestMessage}");  return null;  }  string json = await response.Content.ReadAsStringAsync();  using JsonDocument doc = JsonDocument.Parse(json);  List<LastfmTrack> similarTracks = new List<LastfmTrack>();  int i = 0;  foreach(var trackElement in doc.RootElement.GetProperty("similartracks").GetProperty("track").EnumerateArray())  {  LastfmTrack track = new LastfmTrack();  track.Mbid = trackElement.TryGetProperty("mbid", out var mbidProp) ? mbidProp.GetString() : null;  track.Title = trackElement.TryGetProperty("name", out var nameProp) ? nameProp.GetString() : null;  track.Duration = trackElement.TryGetProperty("duration", out var durationProp) && durationProp.TryGetInt32(out var duration) ? duration : 0;  track.Artist = trackElement.TryGetProperty("artist", out var artistElement)  ? new Artist  {  Mbid = artistElement.TryGetProperty("mbid", out var artistMbidProp) ? artistMbidProp.GetString() : null,  Name = artistElement.TryGetProperty("name", out var artistNameProp) ? artistNameProp.GetString() : null  }  : null;  track.Album = new Album {  CoverImages = trackElement.TryGetProperty("image", out var imagesElement)  ? imagesElement.EnumerateArray()  .Select(img => new CoverImage  {  Url = img.TryGetProperty("#text", out var urlProp) ? urlProp.GetString() : null,  Size = img.TryGetProperty("size", out var sizeProp) ? sizeProp.GetString() : null  })  .Where(img => !string.IsNullOrEmpty(img.Url))  .ToList()  : new List<CoverImage>()  };  track.Genres = trackElement.TryGetProperty("toptags", out var toptagsElement)  && toptagsElement.TryGetProperty("tag", out var tagsArrayElement)  ? tagsArrayElement.EnumerateArray()  .Select(tag => tag.TryGetProperty("name", out var tagNameProp) ? tagNameProp.GetString() : null)  .Where(name => !string.IsNullOrEmpty(name))  .ToList()  : new List<string>();  Console.WriteLine($"Similar Track Number {i}: {track.Mbid}");  i++;  similarTracks.Add(track);  }  return similarTracks;  }  catch (Exception ex)  {  Console.WriteLine($"Error retrieving similar tracks from Last.fm: {ex.Message}");  return new();  }  } |