**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Доцент департамента программной инженерии факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Д. Брейман  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. |

|  |  |
| --- | --- |
| **Подп. и дата** |  |
| **Инв № дубл** |  |
| **Взам. инв. №** |  |
| **Подп. и дата** |  |
| **Инв. № подл** |  |

**Андроид приложение IndieWindy**

**Текст программы**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.01-01 ТЗ 01-1-ЛУ**

**Исполнитель**

Студент группы БПИ173

\_\_\_\_\_\_\_\_\_\_\_\_ / С. И. Ройтман /

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**УТВЕРЖДЕНО**

**RU.17701729.04.01-01 ТЗ 01-1-ЛУ**

**Андроид приложение IndieWindy**

**Текст программы**

**RU.17701729.04.01-01 ТЗ 01-1-ЛУ**

|  |  |
| --- | --- |
| **Подп. и дата** |  |
| **Инв № дубл** |  |
| **Взам. инв. №** |  |
| **Подп. и дата** |  |
| **Инв. № подл** |  |

**Листов 349**

Оглавление

[1. ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ 7](#_Toc37532543)

[1.1. Сервер: 7](#_Toc37532544)

[1.1.1. DatabaseAPI 7](#_Toc37532545)

[1.1.1.1. IndieWindyDbContext.cs 7](#_Toc37532546)

[1.1.1.2. Файл models/BaseEntity.cs 9](#_Toc37532547)

[1.1.1.3. Файл models/AppUser.cs 9](#_Toc37532548)

[1.1.1.4. Файл models/Artist.cs 10](#_Toc37532549)

[1.1.1.5. Файл models/ArtistConcertLink.cs 10](#_Toc37532550)

[1.1.1.6. Файл models/ArtistPostLink.cs 11](#_Toc37532551)

[1.1.1.7. Файл models/Album.cs 12](#_Toc37532552)

[1.1.1.8. Файл models/Concert.cs 13](#_Toc37532553)

[1.1.1.9. Файл models/Donation.cs 14](#_Toc37532554)

[1.1.1.10. Файл models/LatestPost.cs 15](#_Toc37532555)

[1.1.1.11. Файл models/Post.cs 15](#_Toc37532556)

[1.1.1.12. Файл models/ PostSongLink.cs 16](#_Toc37532557)

[1.1.1.13. Файл models/Song.cs 17](#_Toc37532558)

[1.1.1.14. Файл models/UserAlbumLink.cs 18](#_Toc37532559)

[1.1.1.15. Файл models/ UserArtistLink.cs 19](#_Toc37532560)

[1.1.1.16. Файл models/ UserConcertLink.cs 20](#_Toc37532561)

[1.1.1.17. Файл models/ UserSongLink.cs 21](#_Toc37532562)

[1.1.1.18. Файл services/DatabaseBaseService.cs 22](#_Toc37532563)

[1.1.1.19. Файл services/AppUserService.cs 25](#_Toc37532564)

[1.1.1.20. Файл services/ArtistService.cs 25](#_Toc37532565)

[1.1.1.21. Файл services/ConcertService.cs 28](#_Toc37532566)

[1.1.1.22. Файл services/AlbumService.cs 34](#_Toc37532567)

[1.1.1.23. Файл services/DonationService.cs 37](#_Toc37532568)

[1.1.1.24. Файл services/PostService.cs 37](#_Toc37532569)

[1.1.1.25. Файл services/SongService.cs 40](#_Toc37532570)

[1.1.1.26. Файл services/UserAlbumLinkService.cs 43](#_Toc37532571)

[1.1.1.27. Файл services/UserArtistLinkService.cs 43](#_Toc37532572)

[1.1.1.28. Файл services/UserConcertLinkService.cs 44](#_Toc37532573)

[1.1.1.29. Файл services/UserSongLinkService.cs 44](#_Toc37532574)

[1.1.1.30. Файл utils/DapperMappingConfiguration.cs 45](#_Toc37532575)

[1.1.1.31. Файл utils/ PasswordHasherUtil.cs 46](#_Toc37532576)

[1.1.1.32. Файл utils/SearchUtil.cs 48](#_Toc37532577)

[1.1.2. WebAPI 49](#_Toc37532578)

[1.1.2.1. Файл Program.cs 49](#_Toc37532579)

[1.1.2.2. Файл Startup.cs 49](#_Toc37532580)

[1.1.2.3. Файл appsettings.json 52](#_Toc37532581)

[1.1.2.4. Файл appsettings.Development.json 52](#_Toc37532582)

[1.1.2.5. Файл properties/launchSettings.json 53](#_Toc37532583)

[1.1.2.6. Файл controllers/AlbumController.cs 54](#_Toc37532584)

[1.1.2.7. Файл controllers/AppUserController.cs 55](#_Toc37532585)

[1.1.2.8. Файл controllers/ArtistController.cs 57](#_Toc37532586)

[1.1.2.9. Файл controllers/ConcertController.cs 58](#_Toc37532587)

[1.1.2.10. Файл controllers/DonationController.cs 61](#_Toc37532588)

[1.1.2.11. Файл controllers/PostController.cs 62](#_Toc37532589)

[1.1.2.12. Файл controllers/SongController.cs 63](#_Toc37532590)

[1.1.2.13. Файл controllers/UserAlbumLinkController.cs 64](#_Toc37532591)

[1.1.2.14. Файл controllers/UserArtistLinkController.cs 66](#_Toc37532592)

[1.1.2.15. Файл controllers/UserConcertLinkController.cs 67](#_Toc37532593)

[1.1.2.16. Файл controllers/UserSongLinkController.cs 68](#_Toc37532594)

[1.2. Android приложение 70](#_Toc37532595)

[1.2.1. Java 70](#_Toc37532596)

[1.2.1.1. Файл api/ApiController.java 70](#_Toc37532597)

[1.2.1.2. Файл api/ErrorHandler.java 75](#_Toc37532598)

[1.2.1.3. Файл api/SSLSocketFactoryProvider.java 76](#_Toc37532599)

[1.2.1.4. Файл api/ VolleyCallbackJSONArray.java 79](#_Toc37532600)

[1.2.1.5. Файл api/ VolleyCallbackJSONObject.java 79](#_Toc37532601)

[1.2.1.6. Файл api/ VolleyCallbackString.java 80](#_Toc37532602)

[1.2.1.7. Файл api/VolleyQueue.java 80](#_Toc37532603)

[1.2.1.8. Файл controller/AlbumController.java 83](#_Toc37532604)

[1.2.1.9. Файл controller/AppController.java 89](#_Toc37532605)

[1.2.1.10. Файл controller/ArtistController.java 95](#_Toc37532606)

[1.2.1.11. Файл controller/ConcertController.java 101](#_Toc37532607)

[1.2.1.12. Файл controller/DonationController.java 107](#_Toc37532608)

[1.2.1.13. Файл controller/LoginController.java 109](#_Toc37532609)

[1.2.1.14. Файл controller/PostController.java 114](#_Toc37532610)

[1.2.1.15. Файл controller/SearchController.java 117](#_Toc37532611)

[1.2.1.16. Файл controller/SongController.java 119](#_Toc37532612)

[1.2.1.17. Файл interfaces/ILinkAdd.java 123](#_Toc37532613)

[1.2.1.18. Файл interfaces/ ILinkEmpty.java 123](#_Toc37532614)

[1.2.1.19. Файл interfaces/ISearchableAlbum.java 123](#_Toc37532615)

[1.2.1.20. Файл interfaces/ ISearchableArtist.java 124](#_Toc37532616)

[1.2.1.21. Файл interfaces/ ISearchableSong.java 124](#_Toc37532617)

[1.2.1.22. Файл model/Album.java 125](#_Toc37532618)

[1.2.1.23. Файл model/AppUser.java 128](#_Toc37532619)

[1.2.1.24. Файл model/Artist.java 130](#_Toc37532620)

[1.2.1.25. Файл model/ArtistPostLink.java 133](#_Toc37532621)

[1.2.1.26. Файл model/Concert.java 136](#_Toc37532622)

[1.2.1.27. Файл model/Post.java 140](#_Toc37532623)

[1.2.1.28. Файл model/Song.java 142](#_Toc37532624)

[1.2.1.29. Файл model/UserAlbumLink.java 146](#_Toc37532625)

[1.2.1.30. Файл model/UserArtistLink.java 150](#_Toc37532626)

[1.2.1.31. Файл model/UserConcertLink.java 154](#_Toc37532627)

[1.2.1.32. Файл model/ UserSongLink.java 157](#_Toc37532628)

[1.2.1.33. Файл services/DownloadService.java 161](#_Toc37532629)

[1.2.1.34. Файл services/DownloadTracker.java 165](#_Toc37532630)

[1.2.1.35. Файл services/FragmentService.java 174](#_Toc37532631)

[1.2.1.36. Файл services/IconChangerService.java 175](#_Toc37532632)

[1.2.1.37. Файл services/KeyboardService.java 176](#_Toc37532633)

[1.2.1.38. Файл services/PlayerForegroundService.java 176](#_Toc37532634)

[1.2.1.39. Файл services/ PlayerServiceConnection.java 185](#_Toc37532635)

[1.2.1.40. Файл services/ SharedPrefsService.java 186](#_Toc37532636)

[1.2.1.41. Файл ui/activity/AlbumActivity.java 187](#_Toc37532637)

[1.2.1.42. Файл ui/activity/ArtistActivity.java 192](#_Toc37532638)

[1.2.1.43. Файл ui/activity/ConcertActivity.java 196](#_Toc37532639)

[1.2.1.44. Файл ui/activity/DonationActivity.java 200](#_Toc37532640)

[1.2.1.45. Файл ui/activity/LoginActivity.java 202](#_Toc37532641)

[1.2.1.46. Файл ui/activity/MoreActivity.java 205](#_Toc37532642)

[1.2.1.47. Файл ui/activity/NavigationActivity.java 207](#_Toc37532643)

[1.2.1.48. Файл ui/activity/PlayerActivity.java 208](#_Toc37532644)

[1.2.1.49. Файл ui/activity/SettingsActivity.java 216](#_Toc37532645)

[1.2.1.50. Файл ui/adapter/ ArtistPostLinkListAdapter.java 218](#_Toc37532646)

[1.2.1.51. Файл ui/adapter/ ConcertPagerAdapter.java 223](#_Toc37532647)

[1.2.1.52. Файл ui/adapter/ HomePagerAdapter.java 225](#_Toc37532648)

[1.2.1.53. Файл ui/adapter/ PersonalPagerAdapter.java 227](#_Toc37532649)

[1.2.1.54. Файл ui/adapter/ UserAlbumLinkListAdapter.java 229](#_Toc37532650)

[1.2.1.55. Файл ui/adapter/ UserArtistLinkListAdapter.java 235](#_Toc37532651)

[1.2.1.56. Файл ui/adapter/ UserConcertLinkListAdapter.java 240](#_Toc37532652)

[1.2.1.57. Файл ui/adapter/ UserSongLinkListAdapter.java 244](#_Toc37532653)

[1.2.1.58. Файл ui/custom\_views/ MyListView.java 249](#_Toc37532654)

[1.2.1.59. Файл ui/fragments /concert/ ConcertFragment.java 250](#_Toc37532655)

[1.2.1.60. Файл ui/fragments /concert/ NearestConcertFragment.java 256](#_Toc37532656)

[1.2.1.61. Файл ui/fragments /concert/ SavedConcertFragment.java 258](#_Toc37532657)

[1.2.1.62. Файл ui/fragments /concert/ SubscriptionConcertFragment.java 260](#_Toc37532658)

[1.2.1.63. Файл ui/fragments /home/ HomeFragment.java 261](#_Toc37532659)

[1.2.1.64. Файл ui/fragments /home/ LatestHomeFragment.java 264](#_Toc37532660)

[1.2.1.65. Файл ui/fragments /home/ SubscriptionHomeFragment.java 266](#_Toc37532661)

[1.2.1.66. Файл ui/fragments /links/ ArtistPostLinkListFragment.java 268](#_Toc37532662)

[1.2.1.67. Файл ui/fragments /links/ UserAlbumLinkListFragment.java 270](#_Toc37532663)

[1.2.1.68. Файл ui/fragments /links/ UserArtistLinkListFragment.java 273](#_Toc37532664)

[1.2.1.69. Файл ui/fragments /links/ UserConcertLinkListFragment.java 276](#_Toc37532665)

[1.2.1.70. Файл ui/fragments /links/ UserSongLinkListFragment.java 278](#_Toc37532666)

[1.2.1.71. Файл ui/fragments /personal/ PersonalFragment.java 281](#_Toc37532667)

[1.2.1.72. Файл ui/fragments /personal/ AlbumsPersonalFragment.java 286](#_Toc37532668)

[1.2.1.73. Файл ui/fragments /personal/ ArtistsPersonalFragment.java 288](#_Toc37532669)

[1.2.1.74. Файл ui/fragments /personal/ SongsPersonalFragment.java 290](#_Toc37532670)

[1.2.1.75. Файл ui/fragments /search SearchFragment.java/ 292](#_Toc37532671)

[1.2.2. Res 298](#_Toc37532672)

[1.2.2.1. Файл layout/activity\_album.xml 298](#_Toc37532673)

[1.2.2.2. Файл layout/activity\_artist.xml 300](#_Toc37532674)

[1.2.2.3. Файл layout/activity\_concert.xml 302](#_Toc37532675)

[1.2.2.4. Файл layout/activity\_donation.xml 305](#_Toc37532676)

[1.2.2.5. Файл layout/activity\_login.xml 307](#_Toc37532677)

[1.2.2.6. Файл layout/activity\_more.xml 310](#_Toc37532678)

[1.2.2.7. Файл layout/activity\_navigation.xml 310](#_Toc37532679)

[1.2.2.8. Файл layout/activity\_player.xml 311](#_Toc37532680)

[1.2.2.9. Файл layout/album\_list\_item.xml 312](#_Toc37532681)

[1.2.2.10. Файл layout/album\_list\_item.xml 314](#_Toc37532682)

[1.2.2.11. Файл layout/ artist\_list\_item.xml 316](#_Toc37532683)

[1.2.2.12. Файл layout/ concert\_list\_item.xml 318](#_Toc37532684)

[1.2.2.13. Файл layout/ custom\_exo\_controller\_view.xml 321](#_Toc37532685)

[1.2.2.14. Файл layout/ exo\_player\_view.xml 324](#_Toc37532686)

[1.2.2.15. Файл layout/ fragment\_album\_list.xml 328](#_Toc37532687)

[1.2.2.16. Файл layout/ fragment\_artist\_list.xml 329](#_Toc37532688)

[1.2.2.17. Файл layout/ fragment\_concert\_artist.xml 330](#_Toc37532689)

[1.2.2.18. Файл layout/ fragment\_concert\_list.xml 330](#_Toc37532690)

[1.2.2.19. Файл layout/ fragment\_concert\_nearest.xml 331](#_Toc37532691)

[1.2.2.20. Файл layout/ fragment\_concert\_saved.xml 331](#_Toc37532692)

[1.2.2.21. Файл layout/ fragment\_concert\_subscription.xml 332](#_Toc37532693)

[1.2.2.22. Файл layout/ fragment\_home.xml 332](#_Toc37532694)

[1.2.2.23. Файл layout/ fragment\_home\_latest.xml 333](#_Toc37532695)

[1.2.2.24. Файл layout/ fragment\_home\_subscription.xml 333](#_Toc37532696)

[1.2.2.25. Файл layout/ fragment\_personal.xml 334](#_Toc37532697)

[1.2.2.26. Файл layout/ fragment\_personal\_albums.xml 335](#_Toc37532698)

[1.2.2.27. Файл layout/ fragment\_personal\_artists.xml 335](#_Toc37532699)

[1.2.2.28. Файл layout/ fragment\_personal\_songs.xml 335](#_Toc37532700)

[1.2.2.29. Файл layout/ fragment\_post\_list.xml 336](#_Toc37532701)

[1.2.2.30. Файл layout/ fragment\_search.xml 336](#_Toc37532702)

[1.2.2.31. Файл layout/ fragment\_song\_list.xml 337](#_Toc37532703)

[1.2.2.32. Файл layout/ post\_list\_item.xml 339](#_Toc37532704)

[1.2.2.33. Файл layout/ settings\_activity.xml 341](#_Toc37532705)

[1.2.2.34. Файл layout/ song\_list\_item.xml 341](#_Toc37532706)

[1.2.2.35. Файл menu/ bottom\_nav\_menu.xml 344](#_Toc37532707)

[1.2.2.36. Файл menu/ concert\_search\_menu.xml 344](#_Toc37532708)

[1.2.2.37. Файл menu/ personal\_menu.xml 345](#_Toc37532709)

[1.2.2.38. Файл menu/ popup\_album\_menu.xml 346](#_Toc37532710)

[1.2.2.39. Файл menu/ popup\_song\_menu.xml 346](#_Toc37532711)

[1.2.2.40. Файл menu/ search\_menu.xml 347](#_Toc37532712)

[1.2.2.41. Файл navigation/ mobile\_navigation.xml 347](#_Toc37532713)

[2. ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 349](#_Toc37532714)

1. ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ

Весь исходный код можно найти в публично репозитории на GitHub: <https://github.com/Hetfield96/IndieWindy>

* 1. Сервер:
     1. DatabaseAPI
        1. IndieWindyDbContext.cs

using DatabaseAPI.Models;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI

{

/\* Migrations commands: from DatabaseAPI folder

dotnet ef --startup-project ../WebAPI/ migrations add \*name\*

dotnet ef --startup-project ../WebAPI/ database update

\*/

public class IndieWindyDbContext:DbContext

{

public static string ConnectionString;

public IndieWindyDbContext(DbContextOptions<IndieWindyDbContext> options) : base(options)

{

ConnectionString = Database.GetDbConnection()?.ConnectionString;

}

public DbSet<AppUser> AppUser { get; set; }

public DbSet<Artist> Artist { get; set; }

public DbSet<Album> Album { get; set; }

public DbSet<Song> Song { get; set; }

public DbSet<Concert> Concert { get; set; }

public DbSet<Donation> Donation { get; set; }

public DbSet<Post> Post { get; set; }

public DbSet<LatestPost> LatestPost { get; set; }

public DbSet<ArtistConcertLink> ArtistConcertLink { get; set; }

public DbSet<ArtistPostLink> ArtistPostLink { get; set; }

public DbSet<PostSongLink> PostSongLink { get; set; }

public DbSet<UserArtistLink> UserArtistLink { get; set; }

public DbSet<UserAlbumLink> UserAlbumLink { get; set; }

public DbSet<UserSongLink> UserSongLink { get; set; }

public DbSet<UserConcertLink> UserConcertLink { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<ArtistConcertLink>()

.HasKey(i => new { i.ArtistId, i.ConcertId });

modelBuilder.Entity<ArtistPostLink>()

.HasKey(i => new { i.ArtistId, i.PostId });

modelBuilder.Entity<PostSongLink>()

.HasKey(i => new { i.PostId, i.SongId });

modelBuilder.Entity<UserAlbumLink>()

.HasKey(i => new { i.AppUserId, i.AlbumId });

modelBuilder.Entity<UserArtistLink>()

.HasKey(i => new { i.AppUserId, i.ArtistId });

modelBuilder.Entity<UserConcertLink>()

.HasKey(i => new { i.AppUserId, i.ConcertId });

modelBuilder.Entity<UserSongLink>()

.HasKey(i => new { i.AppUserId, i.SongId });

}

}

}

* + - 1. Файл models/BaseEntity.cs

namespace DatabaseAPI.Models

{

public class BaseEntity

{

}

}

* + - 1. Файл models/AppUser.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("app\_user")]

public class AppUser : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("name")]

public string Name { get; set; }

[Column("password")]

public string Password { get; set; }

}

}

* + - 1. Файл models/Artist.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("artist")]

public class Artist : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("name")]

public string Name { get; set; }

[Column("description")]

public string Description { get; set; }

[Column("image\_url")]

public string ImageUrl { get; set; }

}

}

* + - 1. Файл models/ArtistConcertLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("artist\_concert\_link")]

public class ArtistConcertLink : BaseEntity

{

[Key]

[Column("artist\_id", Order=1)]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

[Key]

[Column("concert\_id", Order=2)]

public int ConcertId { get; set; }

[ForeignKey("ConcertId")]

public Concert Concert { get; set; }

}

}

* + - 1. Файл models/ArtistPostLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("artist\_post\_link")]

public class ArtistPostLink : BaseEntity

{

[Key]

[Column("artist\_id", Order=1)]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

[Key]

[Column("post\_id", Order=2)]

public int PostId { get; set; }

[ForeignKey("PostId")]

public Post Post { get; set; }

}

}

* + - 1. Файл models/Album.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("album")]

public class Album : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("name")]

public string Name { get; set; }

[Column("image\_url")]

public string ImageUrl { get; set; }

[Column("artist\_id")]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

}

}

* + - 1. Файл models/Concert.cs

using System;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("concert")]

public class Concert : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("name")]

public string Name { get; set; }

[Column("image\_url")]

public string ImageUrl { get; set; }

[Column("cost")]

public int Cost { get; set; }

[Column("start\_time")]

public DateTime StartTime { get; set; }

[Column("address")]

public string Address { get; set; }

[Column("ticket\_link")]

public string TicketLink { get; set; }

[Column("description")]

public string Description { get; set; }

}

}

* + - 1. Файл models/Donation.cs

using System;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("donation")]

public class Donation : BaseEntity

{

[Key]

[Column("id", Order=1)]

public int Id { get; set; }

[Column("app\_user\_id")]

public int AppUserId { get; set; }

[ForeignKey("AppUserId")]

public AppUser AppUser { get; set; }

[Column("artist\_id")]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

[Column("amount")]

public int Amount { get; set; }

[Column("date")]

public DateTime Date { get; set; }

}

}

* + - 1. Файл models/LatestPost.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("latest\_post")]

public class LatestPost : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("post\_id")]

public int PostId { get; set; }

}

}

* + - 1. Файл models/Post.cs

using System;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("post")]

public class Post : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("text")]

public string Text { get; set; }

[Column("time")]

public DateTime Time { get; set; }

[Column("artist\_id")]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

}

}

* + - 1. Файл models/ PostSongLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("post\_song\_link")]

public class PostSongLink : BaseEntity

{

[Key]

[Column("post\_id", Order=1)]

public int PostId { get; set; }

[ForeignKey("PostId")]

public Post Post { get; set; }

[Key]

[Column("song\_id", Order=2)]

public int SongId { get; set; }

[ForeignKey("SongId")]

public Song Song { get; set; }

}

}

* + - 1. Файл models/Song.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("song")]

public class Song : BaseEntity

{

[Key]

[Column("id")]

public int Id { get; set; }

[Column("name")]

public string Name { get; set; }

[Column("song\_url")]

public string SongUrl { get; set; }

[Column("artist\_id")]

public int ArtistId { get; set; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

[Column("album\_id")]

public int AlbumId { get; set; }

[ForeignKey("AlbumId")]

public Album Album { get; set; }

}

}

* + - 1. Файл models/UserAlbumLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("user\_album\_link")]

public class UserAlbumLink : BaseEntity

{

public UserAlbumLink() {}

public UserAlbumLink(int userId, int albumId)

{

AppUserId = userId;

AlbumId = albumId;

}

[Key]

[Column("app\_user\_id", Order=1)]

public int AppUserId { get; set; }

[ForeignKey("AppUserId")]

public AppUser AppUser { get; set; }

[Key]

[Column("album\_id", Order=2)]

public int AlbumId { get; }

[ForeignKey("AlbumId")]

public Album Album { get; set; }

}

}

* + - 1. Файл models/ UserArtistLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("user\_artist\_link")]

public class UserArtistLink : BaseEntity

{

public UserArtistLink() { }

public UserArtistLink(int appUserId, int artistId)

{

AppUserId = appUserId;

ArtistId = artistId;

}

[Key]

[Column("app\_user\_id", Order=1)]

public int AppUserId { get; set; }

[ForeignKey("AppUserId")]

public AppUser AppUser { get; set; }

[Key]

[Column("artist\_id", Order=2)]

public int ArtistId { get; }

[ForeignKey("ArtistId")]

public Artist Artist { get; set; }

}

}

* + - 1. Файл models/ UserConcertLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("user\_concert\_link")]

public class UserConcertLink : BaseEntity

{

public UserConcertLink()

{

}

public UserConcertLink(int appUserId, int concertId)

{

AppUserId = appUserId;

ConcertId = concertId;

}

[Key]

[Column("app\_user\_id", Order=1)]

public int AppUserId { get; set; }

[ForeignKey("AppUserId")]

public AppUser AppUser { get; set; }

[Key]

[Column("concert\_id", Order=2)]

public int ConcertId { get; }

[ForeignKey("ConcertId")]

public Concert Concert { get; set; }

}

}

* + - 1. Файл models/ UserSongLink.cs

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace DatabaseAPI.Models

{

[Table("user\_song\_link")]

public class UserSongLink : BaseEntity

{

public UserSongLink() {}

public UserSongLink(int appUserId, int songId)

{

AppUserId = appUserId;

SongId = songId;

}

[Key]

[Column("app\_user\_id", Order=1)]

public int AppUserId { get; set; }

[ForeignKey("AppUserId")]

public AppUser AppUser { get; set; }

[Key]

[Column("song\_id", Order=2)]

public int SongId { get; }

[ForeignKey("SongId")]

public Song Song { get; set; }

}

}

* + - 1. Файл services/DatabaseBaseService.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class DatabaseBaseService<T> where T : class

{

protected readonly IndieWindyDbContext \_indieWindyDb;

public DatabaseBaseService(IndieWindyDbContext indieWindyDb)

{

\_indieWindyDb = indieWindyDb;

}

/// <summary>

/// Made not implemented for catching error

/// when user forgot to override this

/// </summary>

/// <returns></returns>

public List<T> GetAll()

{

return \_indieWindyDb.Set<T>().ToList();

}

/// <summary>

/// Returns an item that was inserted in db

/// </summary>

/// <param name="item"></param>

/// <returns>null if no item was inserted</returns>

public virtual async Task<T> AddNewItem(T item)

{

var itemInDb = await \_indieWindyDb.AddAsync(item);

var success = await SaveChangesAsync();

return success ? itemInDb.Entity : null;

}

public virtual async Task<Boolean> DeleteItem<T>(T item)

{

try

{

\_indieWindyDb.Remove(item);

var success = await SaveChangesAsync();

return success;

}

catch (DbUpdateConcurrencyException )

{

// No element was removed

return false;

}

}

/// <summary>

/// Returns item found by it's identifier

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

public async Task<T> FindItemByIdAsync(string id)

{

return await \_indieWindyDb.FindAsync<T>(id);

}

/// <summary>

/// Returns true if delete of item is successful. Also updates the context

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

public async Task<bool> DeleteItemById(string id)

{

var item = await FindItemByIdAsync(id);

\_indieWindyDb.Remove(item);

return await SaveChangesAsync();

}

/// <summary>

/// Returns true if updating context is successful

/// </summary>

/// <returns></returns>

public async Task<bool> SaveChangesAsync()

{

try

{

await \_indieWindyDb.SaveChangesAsync();

return true;

}

catch (Exception e)

{

Console.Error.WriteLine("SaveChangesAsync: error: " + e.Message);

return false;

}

}

}

}

* + - 1. Файл services/AppUserService.cs

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Utils;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class AppUserService : DatabaseBaseService<AppUser>

{

public AppUserService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public override async Task<AppUser> AddNewItem(AppUser user)

{

user.Password = PasswordHasherService.GetMd5Hash(user.Password);

return await base.AddNewItem(user);

}

public async Task<AppUser> FindByName(string name)

{

return await \_indieWindyDb.AppUser.

SingleOrDefaultAsync(u => u.Name.Equals(name));

}

}

}

* + - 1. Файл services/ArtistService.cs

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Dapper;

using DatabaseAPI.Models;

using DatabaseAPI.Utils;

using Microsoft.EntityFrameworkCore;

using Npgsql;

namespace DatabaseAPI.Services

{

public class ArtistService : DatabaseBaseService<Artist>

{

public ArtistService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public async Task<List<UserArtistLink>> FindByName(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserArtistLink, Artist, UserArtistLink>(

@"select link.\*, a.\*

from user\_artist\_link as link

right join artist a on link.artist\_id = a.id and link.app\_user\_id = @user

where lower(a.name) like @name",

(link, artist) =>

{

link.Artist = artist;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserArtistLink>> FindByNameLinked(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

// TODO sort

var res = await con.QueryAsync<UserArtistLink, Artist, UserArtistLink>(

@"select link.\*, a.\*

from user\_artist\_link as link

right join artist a on link.artist\_id = a.id and link.app\_user\_id = @user

where lower(a.name) like @name and link.app\_user\_id is not null",

(link, artist) =>

{

link.Artist = artist;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserAlbumLink>> GetAlbums(int artistId, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserAlbumLink, Album, UserAlbumLink>(

@"select \* from user\_album\_link as link

right join album a on link.album\_id = a.id and link.app\_user\_id = @userId

where a.artist\_id = @artistId",

(link, album) =>

{

link.Album = album;

return link;

},

param: new {artistId, userId});

return res.ToList();

}

}

}

* + - 1. Файл services/ConcertService.cs

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Dapper;

using DatabaseAPI.Models;

using Npgsql;

namespace DatabaseAPI.Services

{

public class ConcertService : DatabaseBaseService<Concert>

{

public ConcertService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public async Task<List<UserConcertLink>> FindByName(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link as link

right join concert c on link.concert\_id = c.id and link.app\_user\_id = @user

where lower(c.name) like @name",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetNearest(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link as link

right join concert c on link.concert\_id = c.id and link.app\_user\_id = @user

where start\_time > now() and lower(c.name) like @name

order by c.start\_time",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetNearestByArtist(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link as link

right join concert c on link.concert\_id = c.id and link.app\_user\_id = @user

where c.start\_time > now() and c.id in

(select distinct acl.concert\_id from artist\_concert\_link acl

join artist a on acl.artist\_id = a.id

where lower(a.name) like @name)

order by c.start\_time;",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetBySubscription(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link link

right join concert c on link.concert\_id = c.id and app\_user\_id = @user

where lower(c.name) like @name and c.id in

(select distinct concert\_id from artist\_concert\_link

where artist\_id in

(select artist\_id from user\_artist\_link

where app\_user\_id = @user))

order by c.start\_time;",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetBySubscriptionByArtist(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link link

right join concert c on link.concert\_id = c.id and app\_user\_id = @user

where c.id in

(select distinct concert\_id from artist\_concert\_link

where artist\_id in

(select artist\_id from user\_artist\_link

join artist a on user\_artist\_link.artist\_id = a.id

where app\_user\_id = @user and lower(a.name) like @name))

order by c.start\_time;",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetSaved(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link link

right join concert c on link.concert\_id = c.id and app\_user\_id = @user

where link.app\_user\_id is not null and lower(c.name) like @name

order by c.start\_time;",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserConcertLink>> GetSavedByArtist(int userId, string query)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserConcertLink, Concert, UserConcertLink>(

@"select link.\*, c.\*

from user\_concert\_link link

right join concert c on link.concert\_id = c.id and app\_user\_id = @user

where link.app\_user\_id is not null and c.id in

(select distinct acl.concert\_id from artist\_concert\_link acl

join artist a on acl.artist\_id = a.id

where lower(a.name) like @name)

order by c.start\_time;",

(link, concert) =>

{

link.Concert = concert;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserArtistLink>> GetArtists(int userId, int concertId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserArtistLink, Artist, UserArtistLink>(

@"select link.\*, a.\*

from user\_artist\_link link

right join artist\_concert\_link acl

on link.artist\_id = acl.artist\_id and link.app\_user\_id = @user

join artist a on acl.artist\_id = a.id

where acl.concert\_id = @concert;",

(link, artist) =>

{

link.Artist = artist;

return link;

},

param: new {@user = userId, @concert = concertId});

return res.ToList();

}

}

}

* + - 1. Файл services/AlbumService.cs

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Dapper;

using DatabaseAPI.Models;

using DatabaseAPI.Utils;

using Microsoft.EntityFrameworkCore;

using Npgsql;

namespace DatabaseAPI.Services

{

public class AlbumService : DatabaseBaseService<Album>

{

public AlbumService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb)

{ }

public async Task<List<UserAlbumLink>> FindByName(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserAlbumLink, Album, Artist, UserAlbumLink>(

@"select link.\*, al.\*, a.\*

from user\_album\_link as link

right join album al on link.album\_id = al.id and link.app\_user\_id = @user

join artist a on al.artist\_id = a.id

where lower(al.name) like @name",

(link, album, artist) =>

{

album.Artist = artist;

link.Album = album;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserAlbumLink>> FindByNameLinked(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

// TODO sort

var res = await con.QueryAsync<UserAlbumLink, Album, Artist, UserAlbumLink>(

@"select link.\*, al.\*, a.\*

from user\_album\_link as link

right join album al on link.album\_id = al.id and link.app\_user\_id = @user

join artist a on al.artist\_id = a.id

where lower(al.name) like @name and link.app\_user\_id is not null",

(link, album, artist) =>

{

album.Artist = artist;

link.Album = album;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserSongLink>> GetSongs(int albumId, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserSongLink, Song, Artist, Album, UserSongLink>(

@"select \* from user\_song\_link as link

right join song s on link.song\_id = s.id and link.app\_user\_id = @userId

join artist a on s.artist\_id = a.id

join album al on s.album\_id = al.id

where s.album\_id = @albumId",

(link, song, artist, album) =>

{

album.Artist = artist;

song.Album = album;

song.Artist = artist;

link.Song = song;

return link;

},

param: new {albumId, userId});

return res.ToList();

}

}

}

* + - 1. Файл services/DonationService.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class DonationService : DatabaseBaseService<Donation>

{

public DonationService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

}

}

* + - 1. Файл services/PostService.cs

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Dapper;

using DatabaseAPI.Models;

using Npgsql;

namespace DatabaseAPI.Services

{

public class PostService : DatabaseBaseService<Concert>

{

public PostService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public async Task<List<ArtistPostLink>> GetBySubscription(int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<ArtistPostLink, Post, Artist, ArtistPostLink>(

@"select apl.\*, p.\*, a.\* from artist\_post\_link apl

join post p on apl.post\_id = p.id

join artist a on apl.artist\_id = a.id

where apl.artist\_id in

(select user\_artist\_link.artist\_id from user\_artist\_link

where app\_user\_id = @user)

order by p.time desc;",

(link, post, artist) =>

{

link.PostId = post.Id;

link.Post = post;

link.ArtistId = artist.Id;

link.Artist = artist;

return link;

},

param: new {@user = userId});

return res.ToList();

}

public async Task<List<ArtistPostLink>> GetLatest()

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<ArtistPostLink, Post, Artist, ArtistPostLink>(

@"select apl.\*, p.\*, a.\* from latest\_post lp

join post p on lp.post\_id = p.id

join artist a on p.artist\_id = a.id

join artist\_post\_link apl on apl.artist\_id = a.id and apl.post\_id = p.id

order by lp.id desc;",

(link, post, artist) =>

{

link.PostId = post.Id;

link.Post = post;

link.ArtistId = artist.Id;

link.Artist = artist;

return link;

});

return res.ToList();

}

public async Task<List<UserSongLink>> GetSongs(int userId, int postId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserSongLink, Song, Artist, Album, UserSongLink>(

@"select usl.\*, s.\*, a.\*, al.\*

from post\_song\_link psl

join song s on psl.song\_id = s.id

join artist a on s.artist\_id = a.id

join album al on s.artist\_id = al.id

left join user\_song\_link usl on s.id = usl.song\_id and usl.app\_user\_id = @userId

where psl.post\_id = @postId;",

(link, song, artist, album) =>

{

song.Album = album;

song.Artist = artist;

link.Song = song;

return link;

},

param: new {userId, postId});

return res.ToList();

}

}

}

* + - 1. Файл services/SongService.cs

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Dapper;

using DatabaseAPI.Models;

using DatabaseAPI.Utils;

using Microsoft.EntityFrameworkCore;

using Npgsql;

namespace DatabaseAPI.Services

{

public class SongService : DatabaseBaseService<Song>

{

public SongService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb)

{ }

public async Task<List<Song>> FindByName(string query)

{

var songs = await \_indieWindyDb.Song

.Include(s => s.Artist)

.Include(s => s.Album)

.ToListAsync();

return songs.Where(s => SearchUtil.StartsWith(s.Name, query)).ToList();

}

public async Task<List<UserSongLink>> FindByName(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

var res = await con.QueryAsync<UserSongLink, Song, Artist, Album, UserSongLink>(

@"select link.\*, s.\*, a.\*, al.\*

from user\_song\_link as link

right join song s on link.song\_id = s.id and link.app\_user\_id = @user

join artist a on s.artist\_id = a.id

join album al on s.album\_id = al.id

where lower(s.name) like @name",

(link, song, artist, album) =>

{

song.Artist = artist;

song.Album = album;

album.Artist = artist;

link.Song = song;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

public async Task<List<UserSongLink>> FindByNameLinked(string query, int userId)

{

await using var con = new NpgsqlConnection(IndieWindyDbContext.ConnectionString);

// TODO sort by position

var res = await con.QueryAsync<UserSongLink, Song, Artist, Album, UserSongLink>(

@"select link.\*, s.\*, a.\*, al.\*

from user\_song\_link as link

right join song s on link.song\_id = s.id and link.app\_user\_id = @user

join artist a on s.artist\_id = a.id

join album al on s.album\_id = al.id

where lower(s.name) like @name and link.app\_user\_id is not null",

(link, song, artist, album) =>

{

song.Artist = artist;

song.Album = album;

album.Artist = artist;

link.Song = song;

return link;

},

param: new {@name = $"{query.ToLower()}%", @user = userId});

return res.ToList();

}

}

}

* + - 1. Файл services/UserAlbumLinkService.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class UserAlbumLinkService: DatabaseBaseService<UserAlbumLink>

{

public UserAlbumLinkService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public async Task<Boolean> LinkExist(int userId, int albumId)

{

return await \_indieWindyDb.UserAlbumLink

.AnyAsync(l => l.AppUserId == userId && l.AlbumId == albumId);

}

}

}

* + - 1. Файл services/UserArtistLinkService.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class UserArtistLinkService : DatabaseBaseService<UserArtistLink>

{

public UserArtistLinkService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

public async Task<Boolean> LinkExist(int userId, int artistId)

{

return await \_indieWindyDb.UserArtistLink

.AnyAsync(l => l.AppUserId == userId && l.ArtistId == artistId);

}

}

}

* + - 1. Файл services/UserConcertLinkService.cs

using DatabaseAPI.Models;

namespace DatabaseAPI.Services

{

public class UserConcertLinkService: DatabaseBaseService<UserConcertLink>

{

public UserConcertLinkService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) {}

}

}

* + - 1. Файл services/UserSongLinkService.cs

using System;

using System.Linq;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using Microsoft.EntityFrameworkCore;

namespace DatabaseAPI.Services

{

public class UserSongLinkService: DatabaseBaseService<UserSongLink>

{

public UserSongLinkService(IndieWindyDbContext indieWindyDb) : base(indieWindyDb) { }

}

}

* + - 1. Файл utils/DapperMappingConfiguration.cs

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using Dapper;

using DatabaseAPI.Models;

namespace DatabaseAPI.Utils

{

public static class DapperMappingConfiguration

{

public static void Configure()

{

ConfigureType<Album>();

ConfigureType<AppUser>();

ConfigureType<Artist>();

ConfigureType<ArtistConcertLink>();

ConfigureType<ArtistPostLink>();

ConfigureType<Donation>();

ConfigureType<Concert>();

ConfigureType<Song>();

ConfigureType<Post>();

ConfigureType<PostSongLink>();

ConfigureType<UserAlbumLink>();

ConfigureType<UserArtistLink>();

ConfigureType<UserConcertLink>();

ConfigureType<UserSongLink>();

}

private static void ConfigureType<T>()

{

SqlMapper.SetTypeMap(typeof(T),

new CustomPropertyTypeMap(typeof(T),

(type, columnName) => type.GetProperties()

.FirstOrDefault(prop => prop.GetCustomAttributes(false)

.OfType<ColumnAttribute>()

.Any(attr => attr.Name == columnName))));

}

}

}

* + - 1. Файл utils/ PasswordHasherUtil.cs

using System;

using System.Security.Cryptography;

using System.Text;

namespace DatabaseAPI.Utils

{

public static class PasswordHasherService

{

public static MD5 \_md5Hasher = MD5.Create();

public static string GetMd5Hash(string input)

{

// Convert the input string to a byte array and compute the hash.

var dataBytes = \_md5Hasher.ComputeHash(Encoding.UTF8.GetBytes(input));

// Create a new Stringbuilder to collect the bytes

// and create a string.

var sBuilder = new StringBuilder();

// Loop through each byte of the hashed data

// and format each one as a hexadecimal string.

for (int i = 0; i < dataBytes.Length; i++)

{

sBuilder.Append(dataBytes[i].ToString("x2"));

}

// Return the hexadecimal string.

return sBuilder.ToString();

}

// Verify a hash against a string.

public static bool VerifyMd5Hash(string input, string hash, bool inputIsHashed = false)

{

// Hash the input.

var hashOfInput = inputIsHashed ? input : GetMd5Hash(input);

// Create a StringComparer an compare the hashes.

StringComparer comparer = StringComparer.OrdinalIgnoreCase;

if (0 == comparer.Compare(hashOfInput, hash))

{

return true;

}

else

{

return false;

}

}

}

}

* + - 1. Файл utils/SearchUtil.cs

using System;

namespace DatabaseAPI.Utils

{

public static class SearchUtil

{

public static bool StartsWith(string name, string query)

{

return name.IndexOf(query, StringComparison.OrdinalIgnoreCase) == 0;

}

}

}

* + 1. WebAPI
       1. Файл Program.cs

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Hosting;

namespace WebAPI

{

public class Program

{

public static void Main(string[] args)

{

CreateHostBuilder(args).Build().Run();

}

public static IHostBuilder CreateHostBuilder(string[] args) =>

Host.CreateDefaultBuilder(args)

.ConfigureWebHostDefaults(webBuilder => { webBuilder.UseStartup<Startup>(); });

}

}

* + - 1. Файл Startup.cs

using System;

using DatabaseAPI;

using DatabaseAPI.Services;

using DatabaseAPI.Utils;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

namespace WebAPI

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

// Database services

services.AddScoped<AppUserService>();

services.AddScoped<ArtistService>();

services.AddScoped<AlbumService>();

services.AddScoped<SongService>();

services.AddScoped<ConcertService>();

services.AddScoped<UserArtistLinkService>();

services.AddScoped<UserAlbumLinkService>();

services.AddScoped<UserSongLinkService>();

services.AddScoped<UserConcertLinkService>();

services.AddScoped<DonationService>();

services.AddScoped<PostService>();

services.AddControllers();

// Azure db in production, local db otherwise

String connectionString = Configuration.GetConnectionString(

Environment.GetEnvironmentVariable("ASPNETCORE\_ENVIRONMENT").Equals("Production") ?

"AzureConnection" : "LocalConnection");

services.AddEntityFrameworkNpgsql()

.AddDbContext<IndieWindyDbContext>(opt =>

opt.UseNpgsql(connectionString));

// Automatically perform database migration

services.BuildServiceProvider().GetService<IndieWindyDbContext>().Database.Migrate();

// Dapper mapping configuration

DapperMappingConfiguration.Configure();

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints => { endpoints.MapControllers(); });

}

}

}

* + - 1. Файл appsettings.json

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"ConnectionStrings": {

"LocalConnection": "User ID=postgres;Password=1234;Server=localhost;Port=5430;Database=indieWindyDB;Integrated Security=true;Pooling=true;",

"AzureConnection": "Server=indiewindydb.postgres.database.azure.com;Database=postgres;Port=5432;User Id=indie\_windy\_admin@indiewindydb;Password=enu872Tt;Ssl Mode=Require;"

}

}

* + - 1. Файл appsettings.Development.json

{

"Logging": {

"LogLevel": {

"Default": "Debug",

"System": "Information",

"Microsoft": "Information"

}

},

"DB" : {

"ConnectionStrings": "User ID=postgres;Password=1234;Server=localhost;Port=5430;Database=indieWindyDB;Integrated Security=true;Pooling=true;"

}

}

* + - 1. Файл properties/launchSettings.json

{

"$schema": "http://json.schemastore.org/launchsettings.json",

"iisSettings": {

"windowsAuthentication": false,

"anonymousAuthentication": true,

"iisExpress": {

"applicationUrl": "http://localhost:16366",

"sslPort": 44335

}

},

"profiles": {

"IIS Express": {

"commandName": "IISExpress",

"launchBrowser": true,

"launchUrl": "appuser",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

},

"IndieWindyServer": {

"commandName": "Project",

"launchUrl": "appuser",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

},

"applicationUrl": "https://localhost:5001;http://localhost:5000"

}

}

}

* + - 1. Файл controllers/AlbumController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class AlbumController : ControllerBase

{

private readonly AlbumService \_albumService;

public AlbumController(AlbumService albumService)

{

\_albumService = albumService;

}

[Route("find/{query}/{userId}")]

public async Task<List<UserAlbumLink>> FindByName(string query, int userId)

{

return await \_albumService.FindByName(query, userId);

}

[Route("findLinked/{query}/{userId}")]

public async Task<List<UserAlbumLink>> FindByNameLinked(string query, int userId)

{

query = query.Equals("null") ? "" : query;

return await \_albumService.FindByNameLinked(query, userId);

}

[HttpGet]

[Route("{userId}/{albumId}/songs")]

public async Task<List<UserSongLink>> GetSongs(int userId, int albumId)

{

return await \_albumService.GetSongs(albumId, userId);

}

}

}

* + - 1. Файл controllers/AppUserController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using DatabaseAPI.Utils;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class AppUserController : ControllerBase

{

private readonly AppUserService \_appUserService;

public AppUserController(AppUserService appUserService)

{

\_appUserService = appUserService;

}

[HttpGet]

[Route("all")]

public IEnumerable<AppUser> GetAll()

{

return \_appUserService.GetAll();

}

[HttpPost]

[Route("register")]

public async Task<AppUser> CreateAppUser([FromBody] AppUser user)

{

var duplicateCheck = await \_appUserService.FindByName(user.Name);

return duplicateCheck == null ? await \_appUserService.AddNewItem(user) : null;

}

[HttpPost]

[Route("login/{passwordIsHashed}")]

public async Task<AppUser> Login([FromBody] AppUser loginUser, bool passwordIsHashed)

{

var user = await \_appUserService.FindByName(loginUser.Name);

if (user == null)

return null;

var res = PasswordHasherService.VerifyMd5Hash(loginUser.Password, user.Password, passwordIsHashed);

return res ? user : null;

}

}

}

* + - 1. Файл controllers/ArtistController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class ArtistController : ControllerBase

{

private readonly ArtistService \_artistService;

public ArtistController(ArtistService artistService)

{

\_artistService = artistService;

}

[HttpGet]

public string Index()

{

return "This is ArtistController";

}

[HttpGet]

[Route("all")]

public IEnumerable<Artist> GetAll()

{

return \_artistService.GetAll();

}

[Route("find/{query}/{userId}")]

public async Task<List<UserArtistLink>> FindByName(string query, int userId)

{

return await \_artistService.FindByName(query, userId);

}

[Route("findLinked/{query}/{userId}")]

public async Task<List<UserArtistLink>> FindByNameLinked(string query, int userId)

{

query = query.Equals("null") ? "" : query;

return await \_artistService.FindByNameLinked(query, userId);

}

[HttpGet]

[Route("{userId}/{artistId}/albums")]

public async Task<List<UserAlbumLink>> GetAlbums(int userId, int artistId)

{

return await \_artistService.GetAlbums(artistId, userId);

}

}

}

* + - 1. Файл controllers/ConcertController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class ConcertController : ControllerBase

{

private readonly ConcertService \_concertService;

public ConcertController(ConcertService concertService)

{

\_concertService = concertService;

}

[HttpGet]

[Route("find/{query}/{userId}")]

public async Task<List<UserConcertLink>> FindByName(string query, int userId)

{

return await \_concertService.FindByName(query, userId);

}

[HttpGet]

[Route("getNearest/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetNearest(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetNearest(userId, query);

}

[HttpGet]

[Route("getNearestByArtist/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetNearestByArtist(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetNearestByArtist(userId, query);

}

// Returns UserConcertLinks for concerts where subscribed artists participate

[HttpGet]

[Route("getBySubscription/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetBySubscription(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetBySubscription(userId, query);

}

[HttpGet]

[Route("getBySubscriptionByArtist/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetBySubscriptionByArtist(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetBySubscriptionByArtist(userId, query);

}

[HttpGet]

[Route("getSaved/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetSaved(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetSaved(userId, query);

}

[HttpGet]

[Route("getSavedByArtist/{userId}/{query}")]

public async Task<List<UserConcertLink>> GetSavedByArtist(int userId, string query)

{

query = query.Equals("null") ? "" : query;

return await \_concertService.GetSavedByArtist(userId, query);

}

[HttpGet]

[Route("{userId}/{concertId}/artists")]

public async Task<List<UserArtistLink>> GetArtists(int userId, int concertId)

{

return await \_concertService.GetArtists(userId, concertId);

}

}

}

* + - 1. Файл controllers/DonationController.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class DonationController: ControllerBase

{

private readonly DonationService \_donationService;

public DonationController(DonationService donationService)

{

\_donationService = donationService;

}

[HttpPost]

[Route("add")]

public async Task<Donation> AddNewItem([FromBody] Donation item)

{

item.Date = DateTime.Now;

return await \_donationService.AddNewItem(item);

}

}

}

* + - 1. Файл controllers/PostController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class PostController : ControllerBase

{

private readonly PostService \_postService;

public PostController(PostService postService)

{

\_postService = postService;

}

[HttpGet]

[Route("getBySubscription/{userId}")]

public async Task<List<ArtistPostLink>> GetBySubscription(int userId)

{

return await \_postService.GetBySubscription(userId);

}

[HttpGet]

[Route("getLatest")]

public async Task<List<ArtistPostLink>> GetLatest()

{

return await \_postService.GetLatest();

}

[HttpGet]

[Route("getSongs/{userId}/{postId}")]

public async Task<List<UserSongLink>> GetSongs(int userId, int postId)

{

return await \_postService.GetSongs(userId, postId);

}

}

}

* + - 1. Файл controllers/SongController.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class SongController : ControllerBase

{

private readonly SongService \_songService;

public SongController(SongService songService)

{

\_songService = songService;

}

[Route("find/{query}/{userId}")]

public async Task<List<UserSongLink>> FindByName(string query, int userId)

{

return await \_songService.FindByName(query, userId);

}

[Route("findLinked/{query}/{userId}")]

public async Task<List<UserSongLink>> FindByNameLinked(string query, int userId)

{

query = query.Equals("null") ? "" : query;

return await \_songService.FindByNameLinked(query, userId);

}

}

}

* + - 1. Файл controllers/UserAlbumLinkController.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class UserAlbumLinkController : ControllerBase

{

private readonly UserAlbumLinkService \_userAlbumLinkService;

public UserAlbumLinkController(UserAlbumLinkService userAlbumLinkService)

{

\_userAlbumLinkService = userAlbumLinkService;

}

[HttpPost]

[Route("add")]

public async Task<UserAlbumLink> AddNewItem([FromBody] UserAlbumLink item)

{

return await \_userAlbumLinkService.AddNewItem(item);

}

[HttpDelete]

[Route("delete/{userId}/{albumId}")]

public async Task<Boolean> Delete(int userId, int albumId)

{

UserAlbumLink link = new UserAlbumLink(userId, albumId);

return await \_userAlbumLinkService.DeleteItem(link);

}

[HttpGet]

[Route("linkExist/{userId}/{albumId}")]

public async Task<Boolean> linkExist(int userId, int albumId)

{

return await \_userAlbumLinkService.LinkExist(userId, albumId);

}

}

}

* + - 1. Файл controllers/UserArtistLinkController.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class UserArtistLinkController : ControllerBase

{

private readonly UserArtistLinkService \_userArtistLinkService;

public UserArtistLinkController(UserArtistLinkService userArtistLinkService)

{

\_userArtistLinkService = userArtistLinkService;

}

[HttpPost]

[Route("add")]

public async Task<UserArtistLink> AddNewItem([FromBody] UserArtistLink item)

{

return await \_userArtistLinkService.AddNewItem(item);

}

[HttpDelete]

[Route("delete/{userId}/{artistId}")]

public async Task<Boolean> Delete(int userId, int artistId)

{

UserArtistLink link = new UserArtistLink(userId, artistId);

return await \_userArtistLinkService.DeleteItem(link);

}

[HttpGet]

[Route("linkExist/{userId}/{artistId}")]

public async Task<Boolean> linkExist(int userId, int artistId)

{

return await \_userArtistLinkService.LinkExist(userId, artistId);

}

}

}

* + - 1. Файл controllers/UserConcertLinkController.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class UserConcertLinkController : ControllerBase

{

private readonly UserConcertLinkService \_userConcertLinkService;

public UserConcertLinkController(UserConcertLinkService userConcertLinkService)

{

\_userConcertLinkService = userConcertLinkService;

}

[HttpPost]

[Route("add")]

public async Task<UserConcertLink> AddNewItem([FromBody] UserConcertLink item)

{

return await \_userConcertLinkService.AddNewItem(item);

}

[HttpDelete]

[Route("delete/{userId}/{concertId}")]

public async Task<Boolean> Delete(int userId, int concertId)

{

UserConcertLink link = new UserConcertLink(userId, concertId);

return await \_userConcertLinkService.DeleteItem(link);

}

}

}

* + - 1. Файл controllers/UserSongLinkController.cs

using System;

using System.Threading.Tasks;

using DatabaseAPI.Models;

using DatabaseAPI.Services;

using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers

{

[ApiController]

[Route("[controller]")]

public class UserSongLinkController : ControllerBase

{

private readonly UserSongLinkService \_userSongLinkService;

public UserSongLinkController(UserSongLinkService userSongLinkService)

{

\_userSongLinkService = userSongLinkService;

}

[HttpPost]

[Route("add")]

public async Task<UserSongLink> AddNewItem([FromBody] UserSongLink item)

{

return await \_userSongLinkService.AddNewItem(item);

}

[HttpDelete]

[Route("delete/{userId}/{songId}")]

public async Task<Boolean> Delete(int userId, int songId)

{

UserSongLink link = new UserSongLink(userId, songId);

return await \_userSongLinkService.DeleteItem(link);

}

}

}

* 1. Android приложение
     1. Java
        1. Файл api/ApiController.java

package com.siroytman.indiewindymobile.api;

import android.util.Log;

import com.android.volley.Response;

import com.android.volley.RetryPolicy;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.JsonArrayRequest;

import com.android.volley.toolbox.JsonObjectRequest;

import com.android.volley.toolbox.StringRequest;

import com.siroytman.indiewindymobile.BuildConfig;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import org.json.JSONArray;

import org.json.JSONObject;

public class ApiController {

private static final String serverUrl;

static {

// Choosing server url

int serverUrlId = BuildConfig.DEBUG ? R.string.server\_url : R.string.azure\_server\_url;

serverUrl = AppController.getContext().getString(serverUrlId);

}

/\*\*

\* Volley queue for executing requests to server

\*/

private VolleyQueue volleyQueue;

/\*\*

\* A singleton instance of the application class for easy access in other places

\*/

private static ApiController instance;

private ApiController() {

// initialize the singleton

instance = this;

// initialize volley queue

volleyQueue = new VolleyQueue(AppController.getContext());

}

/\*\*

\* @return ApplicationController singleton instance

\*/

public static synchronized ApiController getInstance() {

if (instance == null) {

instance = new ApiController();

}

return instance;

}

/\*\*

\* Returns string response from server

\* @param method restMethod: GET, POST, ... {@link com.android.volley.Request.Method}

\* @param apiUrl api address

\* @param callback function to call when got response

\*/

public void getStringResponse(int method, String apiUrl, final VolleyCallbackString callback) {

StringRequest request = new StringRequest(method,

serverUrl + "/" + apiUrl,

new Response.Listener <String> () {

@Override

public void onResponse(String response) {

callback.onSuccessResponse(response);

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError e) {

Log.d("ApiController", "Error: " + e.toString());

}

});

request.setRetryPolicy(new RetryPolicy() {

@Override

public int getCurrentTimeout() {

return 50000;

}

@Override

public int getCurrentRetryCount() {

return 50000;

}

@Override

public void retry(VolleyError error) throws VolleyError {

}

});

volleyQueue.addToRequestQueue(request);

}

public void getJSONObjectResponse(String apiUrl, JSONObject json, final VolleyCallbackJSONObject callback) {

JsonObjectRequest request = new JsonObjectRequest(serverUrl + "/" + apiUrl,

json,

new Response.Listener<JSONObject> () {

@Override

public void onResponse(JSONObject response) {

callback.onSuccessResponse(response);

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError e) {

callback.onErrorResponse(e);

Log.d("ApiController", "Error: " + e.toString());

}

});

request.setRetryPolicy(new RetryPolicy() {

@Override

public int getCurrentTimeout() {

return 50000;

}

@Override

public int getCurrentRetryCount() {

return 50000;

}

@Override

public void retry(VolleyError error) throws VolleyError {

}

});

volleyQueue.addToRequestQueue(request);

}

public void getJSONArrayResponse(int method, String apiUrl, JSONArray json, final VolleyCallbackJSONArray callback) {

JsonArrayRequest request = new JsonArrayRequest(method, serverUrl + "/" + apiUrl,

json,

new Response.Listener<JSONArray> () {

@Override

public void onResponse(JSONArray response) {

callback.onSuccessResponse(response);

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError e) {

callback.onErrorResponse(e);

Log.d("ApiController", "Error: " + e.toString());

}

});

request.setRetryPolicy(new RetryPolicy() {

@Override

public int getCurrentTimeout() {

return 50000;

}

@Override

public int getCurrentRetryCount() {

return 50000;

}

@Override

public void retry(VolleyError error) throws VolleyError {

}

});

volleyQueue.addToRequestQueue(request);

}

}

* + - 1. Файл api/ErrorHandler.java

package com.siroytman.indiewindymobile.api;

import android.content.Context;

import android.util.Log;

import android.widget.Toast;

import com.android.volley.NoConnectionError;

import com.android.volley.TimeoutError;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.R;

public class ErrorHandler {

public static final String TAG = "ErrorHandler";

// Return true if handled, false otherwise

public static Boolean HandleError(Context context, VolleyError error)

{

if (error.getClass() == NoConnectionError.class)

{

Toast.makeText(context, context.getString(R.string.error\_handler\_\_no\_connection), Toast.LENGTH\_LONG)

.show();

Log.d(TAG, context.getString(R.string.error\_handler\_\_no\_connection));

return true;

}

if (error.getClass() == TimeoutError.class)

{

Toast.makeText(context, context.getString(R.string.error\_handler\_\_timeout), Toast.LENGTH\_LONG)

.show();

Log.d(TAG, context.getString(R.string.error\_handler\_\_timeout));

return true;

}

Log.d(TAG, error.getMessage());

return false;

}

}

* + - 1. Файл api/SSLSocketFactoryProvider.java

package com.siroytman.indiewindymobile.api;

import android.content.Context;

import android.util.Log;

import com.siroytman.indiewindymobile.R;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStream;

import java.security.KeyManagementException;

import java.security.KeyStore;

import java.security.KeyStoreException;

import java.security.NoSuchAlgorithmException;

import java.security.cert.Certificate;

import java.security.cert.CertificateException;

import java.security.cert.CertificateFactory;

import java.security.cert.X509Certificate;

import javax.net.ssl.HostnameVerifier;

import javax.net.ssl.HttpsURLConnection;

import javax.net.ssl.SSLContext;

import javax.net.ssl.SSLSession;

import javax.net.ssl.SSLSocketFactory;

import javax.net.ssl.TrustManagerFactory;

/\*\*

\* Class that provides SSLSockets for connection to server

\*/

final class SSLSocketFactoryProvider {

// Returns SSLSocketFactory

static SSLSocketFactory getSocketFactory(final Context appContext) {

CertificateFactory cf;

try {

cf = CertificateFactory.getInstance("X.509");

InputStream caInput = appContext.getResources().openRawResource(R.raw.ssl\_certificate);

Certificate ca;

try {

ca = cf.generateCertificate(caInput);

Log.e("CERT", "ca=" + ((X509Certificate) ca).getSubjectDN());

} finally {

caInput.close();

}

String keyStoreType = KeyStore.getDefaultType();

KeyStore keyStore = KeyStore.getInstance(keyStoreType);

keyStore.load(null, null);

keyStore.setCertificateEntry("ca", ca);

String tmfAlgorithm = TrustManagerFactory.getDefaultAlgorithm();

TrustManagerFactory tmf = TrustManagerFactory.getInstance(tmfAlgorithm);

tmf.init(keyStore);

HostnameVerifier hostnameVerifier = new HostnameVerifier() {

@Override

public boolean verify(String hostname, SSLSession session) {

return hostname.compareTo(appContext.getString(R.string.server\_host))==0;

}

};

HttpsURLConnection.setDefaultHostnameVerifier(hostnameVerifier);

SSLContext context = SSLContext.getInstance("TLS");

context.init(null, tmf.getTrustManagers(), null);

HttpsURLConnection.setDefaultSSLSocketFactory(context.getSocketFactory());

return context.getSocketFactory();

} catch (CertificateException e) {

e.printStackTrace();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

} catch (KeyStoreException e) {

e.printStackTrace();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} catch (KeyManagementException e) {

e.printStackTrace();

}

return null;

}

}

* + - 1. Файл api/ VolleyCallbackJSONArray.java

package com.siroytman.indiewindymobile.api;

import com.android.volley.VolleyError;

import org.json.JSONArray;

public interface VolleyCallbackJSONArray {

void onSuccessResponse(JSONArray result);

void onErrorResponse(VolleyError error);

}

* + - 1. Файл api/ VolleyCallbackJSONObject.java

package com.siroytman.indiewindymobile.api;

import com.android.volley.VolleyError;

import org.json.JSONObject;

public interface VolleyCallbackJSONObject {

void onSuccessResponse(JSONObject result);

void onErrorResponse(VolleyError error);

}

* + - 1. Файл api/ VolleyCallbackString.java

package com.siroytman.indiewindymobile.api;

import com.android.volley.VolleyError;

public interface VolleyCallbackString

{

void onSuccessResponse(String result);

void onErrorResponse(VolleyError error);

}

* + - 1. Файл api/VolleyQueue.java

package com.siroytman.indiewindymobile.api;

import android.content.Context;

import android.text.TextUtils;

import com.android.volley.Network;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.VolleyLog;

import com.android.volley.toolbox.BasicNetwork;

import com.android.volley.toolbox.HurlStack;

import com.android.volley.toolbox.Volley;

import com.siroytman.indiewindymobile.BuildConfig;

/\*\*

\* Volley queue for executing requests to server

\*/

class VolleyQueue {

/\*\*

\* Log or request TAG

\*/

static final String TAG = "VolleyPatterns";

/\*\*

\* Global request queue for Volley

\*/

private RequestQueue mRequestQueue;

Context context;

VolleyQueue(Context context){

this.context = context;

}

/\*\*

\* @return The Volley Request queue, the queue will be created if it is null

\*/

RequestQueue getRequestQueue() {

// lazy initialize the request queue, the queue instance will be

// created when it is accessed for the first time

if (mRequestQueue == null) {

if (BuildConfig.DEBUG) {

// Https connection with localhost

mRequestQueue = Volley.newRequestQueue(context,

new HurlStack(null, SSLSocketFactoryProvider.getSocketFactory(context)));

} else {

// Http connection with Azure

mRequestQueue = Volley.newRequestQueue(context);

}

}

return mRequestQueue;

}

/\*\*

\* Adds the specified request to the global queue, if tag is specified

\* then it is used else Default TAG is used.

\*

\* @param req

\* @param tag

\*/

<T> void addToRequestQueue(Request<T> req, String tag) {

// set the default tag if tag is empty

req.setTag(TextUtils.isEmpty(tag) ? TAG : tag);

VolleyLog.d("Adding request to queue: %s", req.getUrl());

getRequestQueue().add(req);

}

/\*\*

\* Adds the specified request to the global queue using the Default TAG.

\*

\* @param req

\*/

<T> void addToRequestQueue(Request<T> req) {

// set the default tag if tag is empty

req.setTag(TAG);

getRequestQueue().add(req);

}

/\*\*

\* Cancels all pending requests by the specified TAG, it is important

\* to specify a TAG so that the pending/ongoing requests can be cancelled.

\*

\* @param tag

\*/

void cancelPendingRequests(Object tag) {

if (mRequestQueue != null) {

mRequestQueue.cancelAll(tag);

}

}

}

* + - 1. Файл controller/AlbumController.java

package com.siroytman.indiewindymobile.controller;

import android.util.Log;

import com.android.volley.Request;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.ErrorHandler;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONArray;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.api.VolleyCallbackString;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.interfaces.ISearchableAlbum;

import com.siroytman.indiewindymobile.model.Album;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.ui.activity.AlbumActivity;

import org.json.JSONArray;

import org.json.JSONObject;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

public class AlbumController {

private static final String TAG = "AlbumController";

private ApiController apiController;

private static AlbumController instance;

private AlbumController() {

apiController = ApiController.getInstance();

}

public static synchronized AlbumController getInstance() {

if (instance == null) {

instance = new AlbumController();

}

return instance;

}

public void searchAlbums(final ISearchableAlbum view, String query){

String url = "album/find/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Albums search: request completed");

ArrayList<UserAlbumLink> links = UserAlbumLink.parseLinks(result);

view.albumsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Albums search: request not completed!");

}

});

}

public void searchAlbumsLinked(final ISearchableAlbum view) {

searchAlbumsLinked(view, "null");

}

public void searchAlbumsLinked(final ISearchableAlbum view, String query){

String url = "album/findLinked/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Albums search: request completed");

ArrayList<UserAlbumLink> links = UserAlbumLink.parseLinks(result);

view.albumsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Albums search: request not completed!");

}

});

}

public void getAlbumSongs(final AlbumActivity albumActivity) {

final Album album = albumActivity.getItem();

String url = "album/" + AppController.user.getId() + "/" + album.getId() + "/songs";

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Songs found");

ArrayList<UserSongLink> links = UserSongLink.parseLinks(result);

albumActivity.songsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

if (!ErrorHandler.HandleError(albumActivity, error)) {

Log.d(TAG, "Songs not found!");

}

}

});

}

public void linkExist(final ILinkAdd<Album> view){

final Album album = view.getItem();

String url = "userAlbumLink/linkExist/" + AppController.user.getId() + "/" + album.getId();

apiController.getStringResponse(Request.Method.GET, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if (result.equals("true")) {

view.added();

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "linkExist error");

}

});

}

public void addUserAlbumLink(final ILinkAdd<Album> view) {

String url = "userAlbumLink/add";

final Album album = view.getItem();

Map<String, Integer> postParam = new HashMap<>();

postParam.put("AppUserId", AppController.user.getId());

postParam.put("AlbumId", album.getId());

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

Log.d(TAG, "UserAlbumLink added: " + album.getName());

view.added();

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "UserAlbumLink not added: " + album.getName());

}

});

}

public void removeUserAlbumLink(final ILinkAdd<Album> view) {

final Album album = view.getItem();

String url = "userAlbumLink/delete/" + AppController.user.getId() + "/" + album.getId();

apiController.getStringResponse(Request.Method.DELETE, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if(result.equals("true")) {

view.removed();

Log.d(TAG, "UserAlbumLink removed: " + album.getName());

} else {

Log.d(TAG, "UserAlbumLink not removed: " + album.getName());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "UserAlbumLink not removed: " + error.getMessage());

}

});

}

}

* + - 1. Файл controller/AppController.java

package com.siroytman.indiewindymobile.controller;

import android.app.Application;

import android.content.Context;

import com.google.android.exoplayer2.DefaultRenderersFactory;

import com.google.android.exoplayer2.RenderersFactory;

import com.google.android.exoplayer2.database.DatabaseProvider;

import com.google.android.exoplayer2.database.ExoDatabaseProvider;

import com.google.android.exoplayer2.offline.ActionFileUpgradeUtil;

import com.google.android.exoplayer2.offline.DefaultDownloadIndex;

import com.google.android.exoplayer2.offline.DefaultDownloaderFactory;

import com.google.android.exoplayer2.offline.DownloadManager;

import com.google.android.exoplayer2.offline.DownloaderConstructorHelper;

import com.google.android.exoplayer2.ui.DownloadNotificationHelper;

import com.google.android.exoplayer2.upstream.DataSource;

import com.google.android.exoplayer2.upstream.DefaultDataSourceFactory;

import com.google.android.exoplayer2.upstream.DefaultHttpDataSourceFactory;

import com.google.android.exoplayer2.upstream.FileDataSource;

import com.google.android.exoplayer2.upstream.HttpDataSource;

import com.google.android.exoplayer2.upstream.cache.Cache;

import com.google.android.exoplayer2.upstream.cache.CacheDataSource;

import com.google.android.exoplayer2.upstream.cache.CacheDataSourceFactory;

import com.google.android.exoplayer2.upstream.cache.NoOpCacheEvictor;

import com.google.android.exoplayer2.upstream.cache.SimpleCache;

import com.google.android.exoplayer2.util.Log;

import com.google.android.exoplayer2.util.Util;

import com.siroytman.indiewindymobile.BuildConfig;

import com.siroytman.indiewindymobile.model.AppUser;

import com.siroytman.indiewindymobile.services.DownloadTracker;

import java.io.File;

import java.io.IOException;

public class AppController extends Application {

public static final String DOWNLOAD\_NOTIFICATION\_CHANNEL\_ID = "download\_channel";

private static final String TAG = "AppController";

private static final String DOWNLOAD\_ACTION\_FILE = "actions";

private static final String DOWNLOAD\_TRACKER\_ACTION\_FILE = "tracked\_actions";

private static final String DOWNLOAD\_CONTENT\_DIRECTORY = "downloads";

protected String userAgent;

private DatabaseProvider databaseProvider;

private File downloadDirectory;

private Cache downloadCache;

private DownloadManager downloadManager;

private DownloadTracker downloadTracker;

private DownloadNotificationHelper downloadNotificationHelper;

public static AppUser user;

public static Context getContext() {

return mInstance.getApplicationContext();

}

private static AppController mInstance;

public static synchronized AppController getInstance() {

return mInstance;

}

@Override

public void onCreate() {

super.onCreate();

userAgent = Util.getUserAgent(this, "ExoPlayer");

mInstance = this;

}

/\*\* Returns a {@link DataSource.Factory}. \*/

public DataSource.Factory buildDataSourceFactory() {

DefaultDataSourceFactory upstreamFactory =

new DefaultDataSourceFactory(this, buildHttpDataSourceFactory());

return buildReadOnlyCacheDataSource(upstreamFactory, getDownloadCache());

}

/\*\* Returns a {@link HttpDataSource.Factory}. \*/

public HttpDataSource.Factory buildHttpDataSourceFactory() {

return new DefaultHttpDataSourceFactory(userAgent);

}

/\*\* Returns whether extension renderers should be used. \*/

public boolean useExtensionRenderers() {

return "withExtensions".equals(BuildConfig.FLAVOR);

}

public RenderersFactory buildRenderersFactory(boolean preferExtensionRenderer) {

@DefaultRenderersFactory.ExtensionRendererMode

int extensionRendererMode =

useExtensionRenderers()

? (preferExtensionRenderer

? DefaultRenderersFactory.EXTENSION\_RENDERER\_MODE\_PREFER

: DefaultRenderersFactory.EXTENSION\_RENDERER\_MODE\_ON)

: DefaultRenderersFactory.EXTENSION\_RENDERER\_MODE\_OFF;

return new DefaultRenderersFactory(/\* context= \*/ this)

.setExtensionRendererMode(extensionRendererMode);

}

public DownloadNotificationHelper getDownloadNotificationHelper() {

if (downloadNotificationHelper == null) {

downloadNotificationHelper =

new DownloadNotificationHelper(this, DOWNLOAD\_NOTIFICATION\_CHANNEL\_ID);

}

return downloadNotificationHelper;

}

public DownloadManager getDownloadManager() {

initDownloadManager();

return downloadManager;

}

public DownloadTracker getDownloadTracker() {

initDownloadManager();

return downloadTracker;

}

protected synchronized Cache getDownloadCache() {

if (downloadCache == null) {

File downloadContentDirectory = new File(getDownloadDirectory(), DOWNLOAD\_CONTENT\_DIRECTORY);

downloadCache =

new SimpleCache(downloadContentDirectory, new NoOpCacheEvictor(), getDatabaseProvider());

}

return downloadCache;

}

private synchronized void initDownloadManager() {

if (downloadManager == null) {

DefaultDownloadIndex downloadIndex = new DefaultDownloadIndex(getDatabaseProvider());

upgradeActionFile(

DOWNLOAD\_ACTION\_FILE, downloadIndex, /\* addNewDownloadsAsCompleted= \*/ false);

upgradeActionFile(

DOWNLOAD\_TRACKER\_ACTION\_FILE, downloadIndex, /\* addNewDownloadsAsCompleted= \*/ true);

DownloaderConstructorHelper downloaderConstructorHelper =

new DownloaderConstructorHelper(getDownloadCache(), buildHttpDataSourceFactory());

downloadManager =

new DownloadManager(

this, downloadIndex, new DefaultDownloaderFactory(downloaderConstructorHelper));

downloadTracker =

new DownloadTracker(/\* context= \*/ this, buildDataSourceFactory(), downloadManager);

}

}

private void upgradeActionFile(

String fileName, DefaultDownloadIndex downloadIndex, boolean addNewDownloadsAsCompleted) {

try {

ActionFileUpgradeUtil.upgradeAndDelete(

new File(getDownloadDirectory(), fileName),

/\* downloadIdProvider= \*/ null,

downloadIndex,

/\* deleteOnFailure= \*/ true,

addNewDownloadsAsCompleted);

} catch (IOException e) {

Log.e(TAG, "Failed to upgrade action file: " + fileName, e);

}

}

private DatabaseProvider getDatabaseProvider() {

if (databaseProvider == null) {

databaseProvider = new ExoDatabaseProvider(this);

}

return databaseProvider;

}

private File getDownloadDirectory() {

if (downloadDirectory == null) {

downloadDirectory = getExternalFilesDir(null);

if (downloadDirectory == null) {

downloadDirectory = getFilesDir();

}

}

return downloadDirectory;

}

protected static CacheDataSourceFactory buildReadOnlyCacheDataSource(

DataSource.Factory upstreamFactory, Cache cache) {

return new CacheDataSourceFactory(

cache,

upstreamFactory,

new FileDataSource.Factory(),

/\* cacheWriteDataSinkFactory= \*/ null,

CacheDataSource.FLAG\_IGNORE\_CACHE\_ON\_ERROR,

/\* eventListener= \*/ null);

}

}

* + - 1. Файл controller/ArtistController.java

package com.siroytman.indiewindymobile.controller;

import android.util.Log;

import com.android.volley.Request;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.ErrorHandler;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONArray;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.api.VolleyCallbackString;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.interfaces.ISearchableArtist;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.ui.activity.ArtistActivity;

import org.json.JSONArray;

import org.json.JSONObject;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

public class ArtistController {

private static final String TAG = "ArtistController";

private ApiController apiController;

private static ArtistController instance;

private ArtistController() {

apiController = ApiController.getInstance();

}

public static synchronized ArtistController getInstance() {

if (instance == null) {

instance = new ArtistController();

}

return instance;

}

public void searchArtists(final ISearchableArtist view, String query){

String url = "artist/find/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Artists search: request completed");

ArrayList<UserArtistLink> links = UserArtistLink.parseLinks(result);

view.artistsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Artists search: request not completed!");

}

});

}

public void searchArtistsLinked(final ISearchableArtist view) {

searchArtistsLinked(view, "null");

}

public void searchArtistsLinked(final ISearchableArtist view, String query){

String url = "artist/findLinked/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Artists search: request completed");

ArrayList<UserArtistLink> links = UserArtistLink.parseLinks(result);

view.artistsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Artists search: request not completed!");

}

});

}

public void getArtistAlbums(final ArtistActivity artistActivity, Artist artist){

String url = "artist/" + AppController.user.getId() + "/" + artist.getId() + "/albums";

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Albums found");

ArrayList<UserAlbumLink> links = UserAlbumLink.parseLinks(result);

artistActivity.albumsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

if (!ErrorHandler.HandleError(artistActivity, error)) {

Log.d(TAG, "Albums not found!");

}

}

});

}

public void linkExist(final ILinkAdd<Artist> view){

final Artist artist = view.getItem();

String url = "userArtistLink/linkExist/" + AppController.user.getId() + "/" + artist.getId();

apiController.getStringResponse(Request.Method.GET, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if (result.equals("true")) {

view.added();

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "linkExist error");

}

});

}

public void addUserArtistLink(final ILinkAdd<Artist> view) {

String url = "userArtistLink/add";

final Artist artist = view.getItem();

Map<String, Integer> postParam = new HashMap<>();

postParam.put("AppUserId", AppController.user.getId());

postParam.put("ArtistId", artist.getId());

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

Log.d(TAG, "UserArtistLink added: " + artist.getName());

view.added();

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "UserArtistLink not added: " + artist.getName());

}

});

}

public void removeUserArtistLink(final ILinkAdd<Artist> view) {

final Artist artist = view.getItem();

String url = "userArtistLink/delete/" + AppController.user.getId() + "/" + artist.getId();

apiController.getStringResponse(Request.Method.DELETE, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if(result.equals("true")) {

view.removed();

Log.d(TAG, "UserArtistLink removed: " + artist.getName());

} else {

Log.d(TAG, "UserArtistLink not removed: " + artist.getName());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "UserArtistLink not removed: " + error.getMessage());

}

});

}

}

* + - 1. Файл controller/ConcertController.java

package com.siroytman.indiewindymobile.controller;

import android.util.Log;

import com.android.volley.Request;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONArray;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.api.VolleyCallbackString;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Concert;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.ui.activity.ConcertActivity;

import com.siroytman.indiewindymobile.ui.fragments.concert.NearestConcertFragment;

import com.siroytman.indiewindymobile.ui.fragments.concert.SavedConcertFragment;

import com.siroytman.indiewindymobile.ui.fragments.concert.SubscriptionConcertFragment;

import org.json.JSONArray;

import org.json.JSONObject;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

public class ConcertController {

private static final String TAG = "ConcertController";

private ApiController apiController;

private static ConcertController instance;

private ConcertController() {

apiController = ApiController.getInstance();

}

public static synchronized ConcertController getInstance() {

if (instance == null) {

instance = new ConcertController();

}

return instance;

}

private String createGetConcertsUrl(String urlStart, String query, boolean byArtist) {

String urlEnd = "/" + AppController.user.getId() + "/" + query;

return byArtist ? urlStart + "ByArtist" + urlEnd : urlStart + urlEnd;

}

public void getNearestConcerts() {

getNearestConcerts("null", false);

}

public void getNearestConcerts(String query, boolean byArtist){

String url = createGetConcertsUrl("concert/getNearest", query, byArtist);

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getNearestConcerts: search request completed");

ArrayList<UserConcertLink> links = UserConcertLink.parseLinks(result);

NearestConcertFragment.getInstance().concertsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getNearestConcerts: search request not completed!" + error.getMessage());

}

});

}

public void getSubscriptionConcerts() {

getSubscriptionConcerts("null", false);

}

public void getSubscriptionConcerts(String query, boolean byArtist){

String url = createGetConcertsUrl("concert/getBySubscription", query, byArtist);

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getSubscriptionConcerts: search request completed");

ArrayList<UserConcertLink> links = UserConcertLink.parseLinks(result);

SubscriptionConcertFragment.getInstance().concertsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getSubscriptionConcerts: search request not completed!" + error.getMessage());

}

});

}

public void getSavedConcerts() {

getSavedConcerts("null", false);

}

public void getSavedConcerts(String query, boolean byArtist){

String url = createGetConcertsUrl("concert/getSaved", query, byArtist);

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getSavedConcerts: search request completed");

ArrayList<UserConcertLink> links = UserConcertLink.parseLinks(result);

SavedConcertFragment.getInstance().concertsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getSavedConcerts: search request not completed!" + error.getMessage());

}

});

}

public void getArtists(final ConcertActivity concertActivity, final int concertId){

String url = "concert/" + AppController.user.getId() + "/" + concertId + "/artists";

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Artists search: request completed");

ArrayList<UserArtistLink> links = UserArtistLink.parseLinks(result);

concertActivity.artistsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Artists search: request not completed!");

}

});

}

public void addUserConcertLink(final ILinkAdd<Concert> view) {

String url = "userConcertLink/add";

final Concert concert = view.getItem();

Map<String, Integer> postParam = new HashMap<>();

postParam.put("AppUserId", AppController.user.getId());

postParam.put("ConcertId", concert.getId());

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

Log.d(TAG, "userConcertLink added: " + concert.getName());

view.added();

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "userConcertLink not added: " + concert.getName());

}

});

}

public void removeUserConcertLink(final ILinkAdd<Concert> view) {

final Concert concert = view.getItem();

String url = "userConcertLink/delete/" + AppController.user.getId() + "/" + concert.getId();

apiController.getStringResponse(Request.Method.DELETE, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if(result.equals("true")) {

view.removed();

Log.d(TAG, "userConcertLink removed: " + concert.getName());

} else {

Log.d(TAG, "userConcertLink not removed: " + concert.getName());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "userConcertLink not removed: " + error.getMessage());

}

});

}

}

* + - 1. Файл controller/DonationController.java

package com.siroytman.indiewindymobile.controller;

import android.content.Context;

import android.util.Log;

import android.widget.Toast;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.model.Artist;

import org.json.JSONObject;

import java.util.HashMap;

import java.util.Map;

public class DonationController {

private static final String TAG = "DonationController";

private ApiController apiController;

private static DonationController instance;

private DonationController() {

apiController = ApiController.getInstance();

}

public static synchronized DonationController getInstance() {

if (instance == null) {

instance = new DonationController();

}

return instance;

}

public void addDonation(final Context context, final Artist artist, final int donationAmount) {

String url = "donation/add";

Map<String, Integer> postParam = new HashMap<>();

postParam.put("AppUserId", AppController.user.getId());

postParam.put("ArtistId", artist.getId());

postParam.put("Amount", donationAmount);

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

Log.d(TAG, "donation added:"

+ "user=" + AppController.user.getName()

+ "; artist=" + artist.getName()

+ "; amount=" + donationAmount);

Toast.makeText(context, R.string.donation\_activity\_\_donation\_message, Toast.LENGTH\_LONG)

.show();

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "donation not added:"

+ "user=" + AppController.user.getName()

+ "; artist=" + artist.getName()

+ "; amount=" + donationAmount);

Toast.makeText(context, R.string.donation\_activity\_\_donation\_message, Toast.LENGTH\_LONG)

.show();

}

});

}

}

* + - 1. Файл controller/LoginController.java

package com.siroytman.indiewindymobile.controller;

import android.content.Context;

import android.content.Intent;

import android.util.Log;

import android.widget.Toast;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.ErrorHandler;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.model.AppUser;

import com.siroytman.indiewindymobile.services.SharedPrefsService;

import com.siroytman.indiewindymobile.ui.activity.LoginActivity;

import com.siroytman.indiewindymobile.ui.activity.NavigationActivity;

import com.siroytman.indiewindymobile.ui.activity.PlayerActivity;

import org.json.JSONObject;

import java.util.HashMap;

import java.util.Map;

public class LoginController {

public static final String TAG = "LoginController";

private ApiController apiController;

private SharedPrefsService sharedPrefsService;

private static LoginController instance;

private LoginController() {

apiController = ApiController.getInstance();

sharedPrefsService = new SharedPrefsService();

}

public static synchronized LoginController getInstance() {

if (instance == null) {

instance = new LoginController();

}

return instance;

}

public void login(LoginActivity loginActivity, String name, String password){

login(loginActivity, name, password, false);

}

private void login(final LoginActivity loginActivity, String name, String password, Boolean passwordIsHashed){

String url = "appuser/login/" + passwordIsHashed;

Map<String, String> postParam = new HashMap<>();

postParam.put("Name", name);

postParam.put("Password", password);

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

try {

AppController.user = AppUser.ParseAppUser(result);

// Save to shared prefs

sharedPrefsService.saveAppUser(AppController.user);

String msg = String.format(loginActivity.getString(R.string.login\_\_hello), AppController.user.getName());

Toast.makeText(loginActivity, msg, Toast.LENGTH\_LONG)

.show();

// Swap to next activity

loginActivity.startActivity(new Intent(loginActivity, NavigationActivity.class));

loginActivity.finish();

}

catch (Exception e)

{

loginActivity.stopLoadingProgressBar();

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

loginActivity.stopLoadingProgressBar();

if (!ErrorHandler.HandleError(loginActivity, error)) {

Toast.makeText(loginActivity, loginActivity.getString(R.string.login\_\_incorrect\_login),

Toast.LENGTH\_LONG)

.show();

}

}

});

}

public void loginFromSharedPrefs(LoginActivity loginActivity){

AppUser user = sharedPrefsService.getAppUser();

if (user != null){

loginActivity.setLoginAndPasswordText(user.getName(), user.getPassword());

loginActivity.startLoadingProgressBar();

login(loginActivity, user.getName(), user.getPassword(), true);

Log.d(TAG, "Login by shared prefs");

}

else{

Log.d(TAG, "No shared prefs exists for login");

loginActivity.stopLoadingProgressBar();

}

}

public void logout(Context context) {

removeSharedPrefs();

AppController.user = null;

context.startActivity(new Intent(context, LoginActivity.class));

}

public void removeSharedPrefs(){

sharedPrefsService.remove();

}

public void register(final LoginActivity loginActivity, String name, String password){

String url = "appuser/register";

Map<String, String> postParam = new HashMap<>();

postParam.put("Name", name);

postParam.put("Password", password);

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

AppController.user = AppUser.ParseAppUser(result);

// Save to shared prefs

sharedPrefsService.saveAppUser(AppController.user);

String msg = String.format(loginActivity.getString(R.string.login\_\_hello), AppController.user.getName());

Toast.makeText(loginActivity, msg, Toast.LENGTH\_LONG)

.show();

// Swap to next activity

loginActivity.startActivity(new Intent(loginActivity, NavigationActivity.class));

loginActivity.finish();

}

@Override

public void onErrorResponse(VolleyError error) {

if (!ErrorHandler.HandleError(loginActivity, error)) {

Toast.makeText(loginActivity, loginActivity.getString(R.string.login\_\_user\_exist),

Toast.LENGTH\_LONG)

.show();

}

}

});

}

}

* + - 1. Файл controller/PostController.java

package com.siroytman.indiewindymobile.controller;

import android.util.Log;

import com.android.volley.Request;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONArray;

import com.siroytman.indiewindymobile.model.ArtistPostLink;

import com.siroytman.indiewindymobile.model.Post;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.ui.adapter.ArtistPostLinkListAdapter;

import com.siroytman.indiewindymobile.ui.fragments.home.LatestHomeFragment;

import com.siroytman.indiewindymobile.ui.fragments.home.SubscriptionHomeFragment;

import org.json.JSONArray;

import java.util.ArrayList;

public class PostController {

private static final String TAG = "PostController";

private ApiController apiController;

private static PostController instance;

private PostController() {

apiController = ApiController.getInstance();

}

public static synchronized PostController getInstance() {

if (instance == null) {

instance = new PostController();

}

return instance;

}

public void getSubscriptionPosts(){

String url = "post/getBySubscription/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getSubscriptionPosts: search request completed");

ArrayList<ArtistPostLink> posts = ArtistPostLink.parseArray(result);

SubscriptionHomeFragment.getInstance().postsFoundViewUpdate(posts);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getSubscriptionPosts: search request not completed!" + error.getMessage());

}

});

}

public void getLatestPosts(){

String url = "post/getLatest";

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getLatestPosts: completed");

ArrayList<ArtistPostLink> posts = ArtistPostLink.parseArray(result);

LatestHomeFragment.getInstance().postsFoundViewUpdate(posts);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getLatestPosts: not completed!" + error.getMessage());

}

});

}

public void getPostSongs(final ArtistPostLinkListAdapter.ViewHolder viewHolder, int postId){

String url = "post/getSongs/" + AppController.user.getId() + "/" + postId;

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "getPostSongs: completed");

if(result.length() > 0) {

ArrayList<UserSongLink> songLinks = UserSongLink.parseLinks(result);

viewHolder.updatePostSongsView(songLinks);

}

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "getPostSongs: not completed!" + error.getMessage());

}

});

}

}

* + - 1. Файл controller/SearchController.java

package com.siroytman.indiewindymobile.controller;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.interfaces.ISearchableAlbum;

import com.siroytman.indiewindymobile.interfaces.ISearchableArtist;

import com.siroytman.indiewindymobile.interfaces.ISearchableSong;

public class SearchController {

private static final String TAG = "SearchController";

private ApiController apiController;

private static SearchController instance;

private SearchController() {

apiController = ApiController.getInstance();

}

public static synchronized SearchController getInstance() {

if (instance == null) {

instance = new SearchController();

}

return instance;

}

public void searchSongs(final ISearchableSong view, final String query){

SongController.getInstance().searchSongs(view, query);

}

public void searchArtists(final ISearchableArtist view, String query){

ArtistController.getInstance().searchArtists(view, query);

}

public void searchAlbums(final ISearchableAlbum view, String query){

AlbumController.getInstance().searchAlbums(view, query);

}

}

* + - 1. Файл controller/SongController.java

package com.siroytman.indiewindymobile.controller;

import android.util.Log;

import com.android.volley.Request;

import com.android.volley.VolleyError;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONArray;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.api.ApiController;

import com.siroytman.indiewindymobile.api.VolleyCallbackJSONObject;

import com.siroytman.indiewindymobile.api.VolleyCallbackString;

import com.siroytman.indiewindymobile.interfaces.ISearchableSong;

import com.siroytman.indiewindymobile.model.Song;

import com.siroytman.indiewindymobile.model.UserSongLink;

import org.json.JSONArray;

import org.json.JSONObject;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

public class SongController {

private static final String TAG = "SongController";

private ApiController apiController;

private static SongController instance;

private SongController() {

apiController = ApiController.getInstance();

}

public static synchronized SongController getInstance() {

if (instance == null) {

instance = new SongController();

}

return instance;

}

public void searchSongs(final ISearchableSong view, final String query){

String url = "song/find/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Songs search: request completed");

ArrayList<UserSongLink> links = UserSongLink.parseLinks(result);

view.songsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Songs search: request not completed!");

}

});

}

public void searchSongsLinked(final ISearchableSong view) {

searchSongsLinked(view, "null");

}

public void searchSongsLinked(final ISearchableSong view, final String query){

String url = "song/findLinked/" + query + "/" + AppController.user.getId();

apiController.getJSONArrayResponse(Request.Method.GET, url, null, new VolleyCallbackJSONArray() {

@Override

public void onSuccessResponse(JSONArray result) {

try {

Log.d(TAG, "Songs search: request completed");

ArrayList<UserSongLink> links = UserSongLink.parseLinks(result);

view.songsFoundViewUpdate(links);

}

catch (Exception e) {

Log.d(TAG, "Unable to parse response: " + e.getMessage());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Songs search: request not completed!");

}

});

}

public void addUserSongLink(final ILinkAdd<Song> view) {

String url = "userSongLink/add";

final Song song = view.getItem();

Map<String, Integer> postParam = new HashMap<>();

postParam.put("AppUserId", AppController.user.getId());

postParam.put("SongId", song.getId());

apiController.getJSONObjectResponse(url, new JSONObject(postParam), new VolleyCallbackJSONObject() {

@Override

public void onSuccessResponse(JSONObject result) {

Log.d(TAG, "UserSongLink added: " + song.getName());

view.added();

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "Error: Song not added: " + song.getName());

}

});

}

public void removeUserSongLink(final ILinkAdd<Song> view) {

final Song song = view.getItem();

String url = "userSongLink/delete/" + AppController.user.getId() + "/" + song.getId();

apiController.getStringResponse(Request.Method.DELETE, url, new VolleyCallbackString() {

@Override

public void onSuccessResponse(String result) {

if(result.equals("true")) {

view.removed();

Log.d(TAG, "UserSongLink removed: " + song.getName());

}

}

@Override

public void onErrorResponse(VolleyError error) {

Log.d(TAG, "UserSongLink not removed: " + error.getMessage());

}

});

}

}

* + - 1. Файл interfaces/ILinkAdd.java

package com.siroytman.indiewindymobile.interfaces;

public interface ILinkAdd<T> {

T getItem();

void removed();

void added();

}

* + - 1. Файл interfaces/ ILinkEmpty.java

package com.siroytman.indiewindymobile.interfaces;

public interface ILinkEmpty {

boolean isEmpty();

void makeEmpty();

}

* + - 1. Файл interfaces/ISearchableAlbum.java

package com.siroytman.indiewindymobile.interfaces;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import java.util.ArrayList;

public interface ISearchableAlbum {

void albumsFoundViewUpdate(ArrayList<UserAlbumLink> links);

}

* + - 1. Файл interfaces/ ISearchableArtist.java

package com.siroytman.indiewindymobile.interfaces;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import java.util.ArrayList;

public interface ISearchableArtist {

void artistsFoundViewUpdate(ArrayList<UserArtistLink> links);

}

* + - 1. Файл interfaces/ ISearchableSong.java

package com.siroytman.indiewindymobile.interfaces;

import com.siroytman.indiewindymobile.model.UserSongLink;

import java.util.ArrayList;

public interface ISearchableSong {

void songsFoundViewUpdate(ArrayList<UserSongLink> links);

}

* + - 1. Файл model/Album.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import org.json.JSONException;

import org.json.JSONObject;

public class Album implements Parcelable {

public static final String TAG = "Album";

// Required

private int id;

private String name;

private String imageUrl;

// Possible

private Artist artist;

public Album() { }

protected Album(Parcel in) {

id = in.readInt();

name = in.readString();

imageUrl = in.readString();

artist = in.readParcelable(Artist.class.getClassLoader());

}

public static final Creator<Album> CREATOR = new Creator<Album>() {

@Override

public Album createFromParcel(Parcel in) {

return new Album(in);

}

@Override

public Album[] newArray(int size) {

return new Album[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(id);

dest.writeString(name);

dest.writeString(imageUrl);

dest.writeParcelable(artist, flags);

}

public static Album Parse(JSONObject jsonObject) {

Album album = new Album();

try {

album.setId(jsonObject.getInt("id"));

album.setName(jsonObject.getString("name"));

album.setImageUrl(jsonObject.getString("imageUrl"));

} catch (JSONException e)

{

Log.d(TAG, "Error: " + e.getMessage());

return null;

}

try {

album.artist = Artist.parse(jsonObject.getJSONObject("artist"));

} catch (JSONException e) {

Log.d(TAG, "No artist");

}

return album;

}

public static Album Parse(JSONObject jsonObject, Artist artist) {

try {

Album album = new Album();

album.setId(jsonObject.getInt("id"));

album.setName(jsonObject.getString("name"));

album.setImageUrl(jsonObject.getString("imageUrl"));

album.setArtist(artist);

return album;

} catch (JSONException e)

{

Log.d(TAG, "Error: " + e.getMessage());

return null;

}

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getImageUrl() {

return imageUrl;

}

public void setImageUrl(String imageUrl) {

this.imageUrl = imageUrl;

}

public Artist getArtist() {

return artist;

}

public void setArtist(Artist artist) {

this.artist = artist;

}

}

* + - 1. Файл model/AppUser.java

package com.siroytman.indiewindymobile.model;

import android.util.Log;

import org.json.JSONException;

import org.json.JSONObject;

public class AppUser {

private int id;

private String name;

private String password;

private AppUser() {

}

public static AppUser ParseAppUser(JSONObject json) {

try {

AppUser user = new AppUser();

user.setId(json.getInt("id"));

user.setName(json.getString("name"));

user.setPassword(json.getString("password"));

return user;

} catch (JSONException e)

{

Log.d("ParseAppUser", "Error: " + e.getMessage());

return null;

}

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

private void setPassword(String password) {

this.password = password;

}

}

* + - 1. Файл model/Artist.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import org.json.JSONException;

import org.json.JSONObject;

public class Artist implements Parcelable{

private int id;

private String name;

private String description;

private String imageUrl;

public Artist() { }

public static Artist parse(JSONObject jsonObject) {

try {

Artist artist = new Artist();

artist.setId(jsonObject.getInt("id"));

artist.setName(jsonObject.getString("name"));

artist.setDescription(jsonObject.getString("description"));

artist.setImageUrl(jsonObject.getString("imageUrl"));

return artist;

} catch (JSONException e)

{

Log.d("ParseArtist", "Error: " + e.getMessage());

return null;

}

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getImageUrl() {

return imageUrl;

}

private void setImageUrl(String imageUrl) {

this.imageUrl = imageUrl;

}

public String getDescription() {

return description;

}

private void setDescription(String description) {

this.description = description;

}

protected Artist(Parcel in) {

id = in.readInt();

name = in.readString();

description = in.readString();

imageUrl = in.readString();

}

public static final Creator<Artist> CREATOR = new Creator<Artist>() {

@Override

public Artist createFromParcel(Parcel in) {

return new Artist(in);

}

@Override

public Artist[] newArray(int size) {

return new Artist[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(id);

dest.writeString(name);

dest.writeString(description);

dest.writeString(imageUrl);

}

}

* + - 1. Файл model/ArtistPostLink.java

package com.siroytman.indiewindymobile.model;

import android.util.Log;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class ArtistPostLink {

public static final String TAG = "ArtistPostLink";

// Required

private int artistId;

private int postId;

// Possible

private Artist artist;

private Post post;

public ArtistPostLink(){}

public static ArtistPostLink Parse(JSONObject json){

ArtistPostLink link = new ArtistPostLink();

try {

link.artistId = json.getInt("artistId");

link.postId = json.getInt("postId");

} catch (JSONException e) {

Log.d(TAG, "Can't parse link");

return null;

}

try {

link.post = Post.parse(json.getJSONObject("post"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse post");

}

try {

link.artist = Artist.parse(json.getJSONObject("artist"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse artist");

}

return link;

}

public static ArrayList<ArtistPostLink> parseArray(JSONArray links)

{

ArrayList<ArtistPostLink> result = new ArrayList<>();

for(int i = 0; i < links.length(); ++i)

{

try {

JSONObject jsonObject = links.getJSONObject(i);

ArtistPostLink link = Parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of links");

}

}

return result;

}

public int getPostId() {

return postId;

}

public Artist getArtist() {

return artist;

}

public void setArtist(Artist artist) {

this.artist = artist;

}

public Post getPost() {

return post;

}

}

* + - 1. Файл model/Concert.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import org.json.JSONObject;

import java.sql.Timestamp;

public class Concert implements Parcelable {

public static final String TAG = "Concert";

private int id;

private String name;

private String imageUrl;

private int cost;

private Timestamp date;

private String address;

private String description;

private String ticketLink;

public static Concert Parse(JSONObject json) {

Concert concert = new Concert();

try {

concert.id = json.getInt("id");

concert.name = json.getString("name");

concert.imageUrl = json.getString("imageUrl");

concert.cost = json.getInt("cost");

concert.date = Timestamp.valueOf(json.getString("startTime").replace("T", " "));

concert.address = json.getString("address");

concert.description = json.getString("description");

concert.ticketLink = json.getString("ticketLink");

} catch (Exception e)

{

Log.d(TAG, "Can't parse concert: " + e.getMessage());

return null;

}

return concert;

}

public Concert() {}

protected Concert(Parcel in) {

id = in.readInt();

name = in.readString();

imageUrl = in.readString();

cost = in.readInt();

address = in.readString();

description = in.readString();

ticketLink = in.readString();

date = Timestamp.valueOf(in.readString());

}

public static final Creator<Concert> CREATOR = new Creator<Concert>() {

@Override

public Concert createFromParcel(Parcel in) {

return new Concert(in);

}

@Override

public Concert[] newArray(int size) {

return new Concert[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(id);

dest.writeString(name);

dest.writeString(imageUrl);

dest.writeInt(cost);

dest.writeString(address);

dest.writeString(description);

dest.writeString(ticketLink);

dest.writeString(date.toString());

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getImageUrl() {

return imageUrl;

}

public String getDateString() {

String res = date.toString();

return res.substring(0, res.lastIndexOf(':'));

}

public String getAddress() {

return address;

}

public String getDescription() {

return description;

}

public String getTicketLink() {

return ticketLink;

}

}

* + - 1. Файл model/Post.java

package com.siroytman.indiewindymobile.model;

import android.util.Log;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.sql.Timestamp;

import java.util.ArrayList;

public class Post {

public static final String TAG = "Post";

private int id;

private String text;

private Timestamp date;

private int artistId;

public static Post parse(JSONObject json) {

Post post = new Post();

try {

post.id = json.getInt("id");

post.text = json.getString("text");

post.date = Timestamp.valueOf(json.getString("time").replace("T", " "));

post.artistId = json.getInt("artistId");

} catch (Exception e)

{

Log.d(TAG, "Can't parse post: " + e.getMessage());

return null;

}

return post;

}

public static ArrayList<Post> parseArray(JSONArray posts)

{

ArrayList<Post> result = new ArrayList<>();

for(int i = 0; i < posts.length(); ++i)

{

try {

JSONObject jsonObject = posts.getJSONObject(i);

Post link = parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of posts");

}

}

return result;

}

public String getDateString() {

String res = date.toString();

return res.substring(0, res.lastIndexOf(':'));

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getText() {

return text;

}

public void setText(String text) {

this.text = text;

}

}

* + - 1. Файл model/Song.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import org.json.JSONException;

import org.json.JSONObject;

import androidx.annotation.Nullable;

public class Song implements Parcelable {

public static final String TAG = "Song";

private int id;

private String name;

private String songUrl;

private Artist artist;

private Album album;

public Song() {}

public static Song Parse(JSONObject json) {

Song song = new Song();

try {

song.id = json.getInt("id");

song.name = json.getString("name");

song.songUrl = json.getString("songUrl");

} catch (JSONException e)

{

Log.d(TAG, "Can't parse song");

return null;

}

try {

song.artist = Artist.parse(json.getJSONObject("artist"));

} catch (JSONException e) {

Log.d(TAG,"No artist");

}

try {

Album album = Album.Parse(json.getJSONObject("album"), song.artist);

song.album = album;

} catch (JSONException e) {

Log.d(TAG,"No album");

}

return song;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSongUrl() {

return songUrl;

}

public Artist getArtist() {

return artist;

}

public void setArtist(Artist artist) {

this.artist = artist;

}

public Album getAlbum() {

return album;

}

public void setAlbum(Album album) {

this.album = album;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

@Override

public boolean equals(@Nullable Object obj) {

// If the object is compared with itself then return true

if (obj == this) {

return true;

}

/\* Check if o is an instance of Song or not

"null instanceof [type]" also returns false \*/

if (!(obj instanceof Song)) {

return false;

}

// typecast o to Song so that we can compare data members

Song songObj = (Song) obj;

// Compare ids

return id == songObj.id;

}

protected Song(Parcel in) {

id = in.readInt();

name = in.readString();

songUrl = in.readString();

artist = in.readParcelable(Artist.class.getClassLoader());

album = in.readParcelable(Album.class.getClassLoader());

}

public static final Creator<Song> CREATOR = new Creator<Song>() {

@Override

public Song createFromParcel(Parcel in) {

return new Song(in);

}

@Override

public Song[] newArray(int size) {

return new Song[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(id);

dest.writeString(name);

dest.writeString(songUrl);

dest.writeParcelable(artist, flags);

dest.writeParcelable(album, flags);

}

}

* + - 1. Файл model/UserAlbumLink.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import com.siroytman.indiewindymobile.interfaces.ILinkEmpty;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class UserAlbumLink implements Parcelable, ILinkEmpty {

public static final String TAG = "UserAlbumLink";

// Required

private int appUserId;

private int albumId;

// Possible

private Album album;

public UserAlbumLink(){}

public static UserAlbumLink Parse(JSONObject json){

UserAlbumLink link = new UserAlbumLink();

try {

link.appUserId = json.getInt("appUserId");

link.albumId = json.getInt("albumId");

} catch (JSONException e) {

Log.d(TAG, "Can't parse link");

return null;

}

try {

link.album = Album.Parse(json.getJSONObject("album"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse album");

}

return link;

}

public static ArrayList<UserAlbumLink> parseLinks(JSONArray links)

{

ArrayList<UserAlbumLink> result = new ArrayList<>();

for(int i = 0; i < links.length(); ++i)

{

try {

JSONObject jsonObject = links.getJSONObject(i);

UserAlbumLink link = Parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of links");

}

}

return result;

}

@Override

public boolean isEmpty(){

return (appUserId == 0 && albumId == 0);

}

@Override

public void makeEmpty(){

appUserId = 0;

albumId = 0;

}

public Album getAlbum() {

return album;

}

public void setAppUserId(int appUserId) {

this.appUserId = appUserId;

}

public void setAlbumId(int albumId) {

this.albumId = albumId;

}

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(appUserId);

dest.writeInt(albumId);

dest.writeParcelable(album, flags);

}

protected UserAlbumLink(Parcel in) {

appUserId = in.readInt();

albumId = in.readInt();

album = in.readParcelable(Album.class.getClassLoader());

}

public static final Creator<UserAlbumLink> CREATOR = new Creator<UserAlbumLink>() {

@Override

public UserAlbumLink createFromParcel(Parcel in) {

return new UserAlbumLink(in);

}

@Override

public UserAlbumLink[] newArray(int size) {

return new UserAlbumLink[size];

}

};

}

* + - 1. Файл model/UserArtistLink.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import com.siroytman.indiewindymobile.interfaces.ILinkEmpty;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class UserArtistLink implements Parcelable, ILinkEmpty {

public static final String TAG = "UserArtistLink";

// Required

private int appUserId;

private int artistId;

// Possible

private Artist artist;

public UserArtistLink(){}

public static UserArtistLink Parse(JSONObject json){

UserArtistLink link = new UserArtistLink();

try {

link.appUserId = json.getInt("appUserId");

link.artistId = json.getInt("artistId");

} catch (JSONException e) {

Log.d(TAG, "Can't parse link");

return null;

}

try {

link.artist = Artist.parse(json.getJSONObject("artist"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse artist");

}

return link;

}

public static ArrayList<UserArtistLink> parseLinks(JSONArray links)

{

ArrayList<UserArtistLink> result = new ArrayList<>();

for(int i = 0; i < links.length(); ++i)

{

try {

JSONObject jsonObject = links.getJSONObject(i);

UserArtistLink link = Parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of links");

}

}

return result;

}

@Override

public boolean isEmpty(){

return (appUserId == 0 && artistId == 0);

}

@Override

public void makeEmpty(){

appUserId = 0;

artistId = 0;

}

public Artist getArtist() {

return artist;

}

public void setAppUserId(int appUserId) {

this.appUserId = appUserId;

}

public void setArtistId(int artistId) {

this.artistId = artistId;

}

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(appUserId);

dest.writeInt(artistId);

dest.writeParcelable(artist, flags);

}

protected UserArtistLink(Parcel in) {

appUserId = in.readInt();

artistId = in.readInt();

artist = in.readParcelable(Artist.class.getClassLoader());

}

public static final Creator<UserArtistLink> CREATOR = new Creator<UserArtistLink>() {

@Override

public UserArtistLink createFromParcel(Parcel in) {

return new UserArtistLink(in);

}

@Override

public UserArtistLink[] newArray(int size) {

return new UserArtistLink[size];

}

};

}

* + - 1. Файл model/UserConcertLink.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import com.siroytman.indiewindymobile.interfaces.ILinkEmpty;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class UserConcertLink implements Parcelable, ILinkEmpty {

public static final String TAG = "UserConcertLink";

// Required

private int appUserId;

private int concertId;

// Possible

private Concert concert;

public UserConcertLink(){}

public static UserConcertLink Parse(JSONObject json){

UserConcertLink link = new UserConcertLink();

try {

link.appUserId = json.getInt("appUserId");

link.concertId = json.getInt("concertId");

} catch (JSONException e) {

Log.d(TAG, "Can't parse link");

return null;

}

try {

link.concert = Concert.Parse(json.getJSONObject("concert"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse concert");

}

return link;

}

public static ArrayList<UserConcertLink> parseLinks(JSONArray links)

{

ArrayList<UserConcertLink> result = new ArrayList<>();

for(int i = 0; i < links.length(); ++i)

{

try {

JSONObject jsonObject = links.getJSONObject(i);

UserConcertLink link = Parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of links");

}

}

return result;

}

@Override

public boolean isEmpty(){

return (appUserId == 0 && concertId == 0);

}

@Override

public void makeEmpty(){

appUserId = 0;

concertId = 0;

}

public Concert getConcert() {

return concert;

}

public void setAppUserId(int appUserId) {

this.appUserId = appUserId;

}

public void setConcertId(int concertId) {

this.concertId = concertId;

}

protected UserConcertLink(Parcel in) {

appUserId = in.readInt();

concertId = in.readInt();

concert = in.readParcelable(Concert.class.getClassLoader());

}

public static final Creator<UserConcertLink> CREATOR = new Creator<UserConcertLink>() {

@Override

public UserConcertLink createFromParcel(Parcel in) {

return new UserConcertLink(in);

}

@Override

public UserConcertLink[] newArray(int size) {

return new UserConcertLink[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(appUserId);

dest.writeInt(concertId);

dest.writeParcelable(concert, flags);

}

}

* + - 1. Файл model/ UserSongLink.java

package com.siroytman.indiewindymobile.model;

import android.os.Parcel;

import android.os.Parcelable;

import android.util.Log;

import com.siroytman.indiewindymobile.interfaces.ILinkEmpty;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class UserSongLink implements Parcelable, ILinkEmpty {

public static final String TAG = "UserSongLink";

// Required

private int appUserId;

private int songId;

// Possible

private Song song;

public UserSongLink(){}

public static UserSongLink Parse(JSONObject json){

UserSongLink link = new UserSongLink();

try {

link.appUserId = json.getInt("appUserId");

link.songId = json.getInt("songId");

} catch (JSONException e) {

Log.d(TAG, "Can't parse link");

return null;

}

try {

link.song = Song.Parse(json.getJSONObject("song"));

} catch (JSONException e) {

Log.d(TAG, "Can't parse song");

}

return link;

}

public static ArrayList<UserSongLink> parseLinks(JSONArray links)

{

ArrayList<UserSongLink> result = new ArrayList<>();

for(int i = 0; i < links.length(); ++i)

{

try {

JSONObject jsonObject = links.getJSONObject(i);

UserSongLink link = Parse(jsonObject);

result.add(link);

}

catch (JSONException e)

{

Log.d(TAG, "Can't parse array of links");

}

}

return result;

}

@Override

public boolean isEmpty(){

return (appUserId == 0 && songId == 0);

}

@Override

public void makeEmpty(){

appUserId = 0;

songId = 0;

}

public Song getSong() {

return song;

}

public void setAppUserId(int appUserId) {

this.appUserId = appUserId;

}

public void setSongId(int songId) {

this.songId = songId;

}

protected UserSongLink(Parcel in) {

appUserId = in.readInt();

songId = in.readInt();

song = in.readParcelable(Song.class.getClassLoader());

}

public static final Creator<UserSongLink> CREATOR = new Creator<UserSongLink>() {

@Override

public UserSongLink createFromParcel(Parcel in) {

return new UserSongLink(in);

}

@Override

public UserSongLink[] newArray(int size) {

return new UserSongLink[size];

}

};

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeInt(appUserId);

dest.writeInt(songId);

dest.writeParcelable(song, flags);

}

}

* + - 1. Файл services/DownloadService.java

/\*

\* Copyright (C) 2017 The Android Open Source Project

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

package com.siroytman.indiewindymobile.services;

import android.app.Notification;

import android.content.Context;

import android.util.Log;

import com.google.android.exoplayer2.offline.Download;

import com.google.android.exoplayer2.offline.DownloadManager;

import com.google.android.exoplayer2.scheduler.PlatformScheduler;

import com.google.android.exoplayer2.ui.DownloadNotificationHelper;

import com.google.android.exoplayer2.util.NotificationUtil;

import com.google.android.exoplayer2.util.Util;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import java.util.List;

import static com.siroytman.indiewindymobile.controller.AppController.DOWNLOAD\_NOTIFICATION\_CHANNEL\_ID;

/\*\* A service for downloading media. \*/

public class DownloadService extends com.google.android.exoplayer2.offline.DownloadService {

public static final String TAG = "DownloadService";

private static final int JOB\_ID = 1;

private static final int FOREGROUND\_NOTIFICATION\_ID = 2;

public DownloadService() {

super(

FOREGROUND\_NOTIFICATION\_ID,

DEFAULT\_FOREGROUND\_NOTIFICATION\_UPDATE\_INTERVAL,

DOWNLOAD\_NOTIFICATION\_CHANNEL\_ID,

R.string.exo\_download\_notification\_channel\_name,

/\* channelDescriptionResourceId= \*/ 0);

}

@Override

protected DownloadManager getDownloadManager() {

// This will only happen once, because getDownloadManager is guaranteed to be called only once

// in the life cycle of the process.

AppController application = (AppController) getApplication();

DownloadManager downloadManager = application.getDownloadManager();

DownloadNotificationHelper downloadNotificationHelper =

application.getDownloadNotificationHelper();

downloadManager.addListener(

new TerminalStateNotificationHelper(

this, downloadNotificationHelper, FOREGROUND\_NOTIFICATION\_ID + 1));

return downloadManager;

}

@Override

protected PlatformScheduler getScheduler() {

return Util.SDK\_INT >= 21 ? new PlatformScheduler(this, JOB\_ID) : null;

}

@Override

protected Notification getForegroundNotification(List<Download> downloads) {

return ((AppController) getApplication())

.getDownloadNotificationHelper()

.buildProgressNotification(

R.drawable.ic\_download, /\* contentIntent= \*/ null, /\* message= \*/ null, downloads);

}

/\*\*

\* Creates and displays notifications for downloads when they complete or fail.

\*

\* <p>This helper will outlive the lifespan of a single instance of {@link DownloadService}.

\* It is static to avoid leaking the first {@link DownloadService} instance.

\*/

private static final class TerminalStateNotificationHelper implements DownloadManager.Listener {

private final Context context;

private final DownloadNotificationHelper notificationHelper;

private int nextNotificationId;

public TerminalStateNotificationHelper(

Context context, DownloadNotificationHelper notificationHelper, int firstNotificationId) {

this.context = context.getApplicationContext();

this.notificationHelper = notificationHelper;

nextNotificationId = firstNotificationId;

}

@Override

public void onDownloadChanged(DownloadManager manager, Download download) {

Log.d(TAG, "onDownloadChanged");

Notification notification;

if (download.state == Download.STATE\_COMPLETED) {

notification =

notificationHelper.buildDownloadCompletedNotification(

R.drawable.ic\_check,

/\* contentIntent= \*/ null,

Util.fromUtf8Bytes(download.request.data));

} else if (download.state == Download.STATE\_FAILED) {

notification =

notificationHelper.buildDownloadFailedNotification(

R.drawable.ic\_check,

/\* contentIntent= \*/ null,

Util.fromUtf8Bytes(download.request.data));

} else {

return;

}

NotificationUtil.setNotification(context, nextNotificationId++, notification);

}

}

}

* + - 1. Файл services/DownloadTracker.java

/\*

\* Copyright (C) 2017 The Android Open Source Project

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

package com.siroytman.indiewindymobile.services;

import android.content.Context;

import android.content.DialogInterface;

import android.net.Uri;

import android.widget.Toast;

import com.google.android.exoplayer2.C;

import com.google.android.exoplayer2.RenderersFactory;

import com.google.android.exoplayer2.offline.Download;

import com.google.android.exoplayer2.offline.DownloadCursor;

import com.google.android.exoplayer2.offline.DownloadHelper;

import com.google.android.exoplayer2.offline.DownloadIndex;

import com.google.android.exoplayer2.offline.DownloadManager;

import com.google.android.exoplayer2.offline.DownloadRequest;

import com.google.android.exoplayer2.offline.DownloadService;

import com.google.android.exoplayer2.trackselection.DefaultTrackSelector;

import com.google.android.exoplayer2.trackselection.MappingTrackSelector.MappedTrackInfo;

import com.google.android.exoplayer2.upstream.DataSource;

import com.google.android.exoplayer2.util.Log;

import com.google.android.exoplayer2.util.Util;

import com.siroytman.indiewindymobile.R;

import java.io.IOException;

import java.util.HashMap;

import java.util.concurrent.CopyOnWriteArraySet;

import androidx.annotation.Nullable;

import androidx.fragment.app.FragmentManager;

/\*\* Tracks media that has been downloaded. \*/

public class DownloadTracker {

/\*\* Listens for changes in the tracked downloads. \*/

public interface Listener {

/\*\* Called when the tracked downloads changed. \*/

void onDownloadsChanged();

}

private static final String TAG = "DownloadTracker";

private final Context context;

private final DataSource.Factory dataSourceFactory;

private final CopyOnWriteArraySet<Listener> listeners;

private final HashMap<Uri, Download> downloads;

private final DownloadIndex downloadIndex;

private final DefaultTrackSelector.Parameters trackSelectorParameters;

@Nullable private StartDownloadDialogHelper startDownloadDialogHelper;

public DownloadTracker(

Context context, DataSource.Factory dataSourceFactory, DownloadManager downloadManager) {

this.context = context.getApplicationContext();

this.dataSourceFactory = dataSourceFactory;

listeners = new CopyOnWriteArraySet<>();

downloads = new HashMap<>();

downloadIndex = downloadManager.getDownloadIndex();

trackSelectorParameters = DownloadHelper.getDefaultTrackSelectorParameters(context);

downloadManager.addListener(new DownloadManagerListener());

loadDownloads();

}

public void addListener(Listener listener) {

listeners.add(listener);

}

public void removeListener(Listener listener) {

listeners.remove(listener);

}

public boolean isDownloaded(Uri uri) {

Download download = downloads.get(uri);

return download != null && download.state != Download.STATE\_FAILED;

}

public DownloadRequest getDownloadRequest(Uri uri) {

Download download = downloads.get(uri);

return download != null && download.state != Download.STATE\_FAILED ? download.request : null;

}

public void toggleDownload(

FragmentManager fragmentManager,

String name,

Uri uri,

String extension,

RenderersFactory renderersFactory) {

Download download = downloads.get(uri);

if (download != null) {

DownloadService.sendRemoveDownload(

context, DownloadService.class, download.request.id, /\* foreground= \*/ false);

} else {

if (startDownloadDialogHelper != null) {

startDownloadDialogHelper.release();

}

startDownloadDialogHelper =

new StartDownloadDialogHelper(

fragmentManager, getDownloadHelper(uri, extension, renderersFactory), name);

}

}

private void loadDownloads() {

Log.d(TAG, "loadDownloads");

try (DownloadCursor loadedDownloads = downloadIndex.getDownloads()) {

while (loadedDownloads.moveToNext()) {

Download download = loadedDownloads.getDownload();

downloads.put(download.request.uri, download);

}

} catch (IOException e) {

Log.d(TAG, "Failed to query downloads", e);

}

}

private DownloadHelper getDownloadHelper(

Uri uri, String extension, RenderersFactory renderersFactory) {

int type = Util.inferContentType(uri, extension);

switch (type) {

case C.TYPE\_DASH:

return DownloadHelper.forDash(context, uri, dataSourceFactory, renderersFactory);

case C.TYPE\_SS:

return DownloadHelper.forSmoothStreaming(context, uri, dataSourceFactory, renderersFactory);

case C.TYPE\_HLS:

return DownloadHelper.forHls(context, uri, dataSourceFactory, renderersFactory);

case C.TYPE\_OTHER:

return DownloadHelper.forProgressive(context, uri);

default:

throw new IllegalStateException("Unsupported type: " + type);

}

}

private class DownloadManagerListener implements DownloadManager.Listener {

@Override

public void onDownloadChanged(DownloadManager downloadManager, Download download) {

downloads.put(download.request.uri, download);

for (Listener listener : listeners) {

listener.onDownloadsChanged();

}

}

@Override

public void onDownloadRemoved(DownloadManager downloadManager, Download download) {

downloads.remove(download.request.uri);

for (Listener listener : listeners) {

listener.onDownloadsChanged();

}

}

}

private final class StartDownloadDialogHelper

implements DownloadHelper.Callback,

DialogInterface.OnClickListener,

DialogInterface.OnDismissListener {

private final FragmentManager fragmentManager;

private final DownloadHelper downloadHelper;

private final String name;

// private TrackSelectionDialog trackSelectionDialog;

private MappedTrackInfo mappedTrackInfo;

public StartDownloadDialogHelper(

FragmentManager fragmentManager, DownloadHelper downloadHelper, String name) {

this.fragmentManager = fragmentManager;

this.downloadHelper = downloadHelper;

this.name = name;

downloadHelper.prepare(this);

}

public void release() {

downloadHelper.release();

// if (trackSelectionDialog != null) {

// trackSelectionDialog.dismiss();

// }

}

// DownloadHelper.Callback implementation.

@Override

public void onPrepared(DownloadHelper helper) {

if (helper.getPeriodCount() == 0) {

Log.d(TAG, "No periods found. Downloading entire stream.");

startDownload();

downloadHelper.release();

return;

}

// mappedTrackInfo = downloadHelper.getMappedTrackInfo(/\* periodIndex= \*/ 0);

// if (!TrackSelectionDialog.willHaveContent(mappedTrackInfo)) {

// Log.d(TAG, "No dialog content. Downloading entire stream.");

// startDownload();

// downloadHelper.release();

// return;

// }

// trackSelectionDialog =

// TrackSelectionDialog.createForMappedTrackInfoAndParameters(

// /\* titleId= \*/ R.string.exo\_download\_description,

// mappedTrackInfo,

// trackSelectorParameters,

// /\* allowAdaptiveSelections =\*/ false,

// /\* allowMultipleOverrides= \*/ true,

// /\* onClickListener= \*/ this,

// /\* onDismissListener= \*/ this);

// trackSelectionDialog.show(fragmentManager, /\* tag= \*/ null);

}

@Override

public void onPrepareError(DownloadHelper helper, IOException e) {

Toast.makeText(context, context.getString(R.string.download\_tracker\_\_download\_failed), Toast.LENGTH\_LONG)

.show();

Log.e(TAG,

e instanceof DownloadHelper.LiveContentUnsupportedException

? "Downloading live content unsupported"

: "Failed to start download",

e);

}

// DialogInterface.OnClickListener implementation.

@Override

public void onClick(DialogInterface dialog, int which) {

for (int periodIndex = 0; periodIndex < downloadHelper.getPeriodCount(); periodIndex++) {

downloadHelper.clearTrackSelections(periodIndex);

for (int i = 0; i < mappedTrackInfo.getRendererCount(); i++) {

// if (!trackSelectionDialog.getIsDisabled(/\* rendererIndex= \*/ i)) {

// downloadHelper.addTrackSelectionForSingleRenderer(

// periodIndex,

// /\* rendererIndex= \*/ i,

// trackSelectorParameters,

// trackSelectionDialog.getOverrides(/\* rendererIndex= \*/ i));

// }

}

}

DownloadRequest downloadRequest = buildDownloadRequest();

if (downloadRequest.streamKeys.isEmpty()) {

// All tracks were deselected in the dialog. Don't start the download.

return;

}

startDownload(downloadRequest);

}

// DialogInterface.OnDismissListener implementation.

@Override

public void onDismiss(DialogInterface dialogInterface) {

downloadHelper.release();

}

// Internal methods.

private void startDownload() {

startDownload(buildDownloadRequest());

}

private void startDownload(DownloadRequest downloadRequest) {

DownloadService.sendAddDownload(

context, DownloadService.class, downloadRequest, /\* foreground= \*/ false);

}

private DownloadRequest buildDownloadRequest() {

return downloadHelper.getDownloadRequest(Util.getUtf8Bytes(name));

}

}

}

* + - 1. Файл services/FragmentService.java

package com.siroytman.indiewindymobile.services;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentActivity;

import androidx.fragment.app.FragmentManager;

public class FragmentService {

// Replace fragment in activity

public static void replaceFragment(FragmentActivity activity, int container, Fragment f){

activity.getSupportFragmentManager().beginTransaction().replace(container, f).commit();

}

// Replace fragment in fragment

public static void replaceFragment(Fragment fragment, int container, Fragment f){

fragment.getChildFragmentManager().beginTransaction().replace(container, f).commit();

}

// Replace fragment

public static void replaceFragment(FragmentManager fragmentManager, int container, Fragment f){

fragmentManager.beginTransaction().replace(container, f).commit();

}

public static void clearFragment(Fragment parentFragment) {

FragmentManager fragmentManager = parentFragment.getChildFragmentManager();

for (Fragment frag : fragmentManager.getFragments()) {

fragmentManager.beginTransaction().remove(frag).commit();

}

}

}

* + - 1. Файл services/IconChangerService.java

package com.siroytman.indiewindymobile.services;

import android.widget.ImageView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.interfaces.ILinkEmpty;

public class IconChanger {

public static void setAddStateIcon(ILinkEmpty link, ImageView addButton) {

if(link.isEmpty()){

setIconAdd(addButton);

} else {

setIconCheck(addButton);

}

}

public static void setIconCheck(ImageView addButton) {

addButton.setImageResource(R.drawable.ic\_check);

}

public static void setIconAdd(ImageView addButton) {

addButton.setImageResource(R.drawable.ic\_add);

}

}

* + - 1. Файл services/KeyboardService.java

package com.siroytman.indiewindymobile.services;

import android.app.Activity;

import android.content.Context;

import android.view.inputmethod.InputMethodManager;

import androidx.fragment.app.FragmentActivity;

public class KeyboardService {

public static void hideKeyboard(FragmentActivity activity){

InputMethodManager inputManager = (InputMethodManager)

activity.getSystemService(Context.INPUT\_METHOD\_SERVICE);

inputManager.hideSoftInputFromWindow(activity.getCurrentFocus().getWindowToken(),

InputMethodManager.HIDE\_NOT\_ALWAYS);

}

}

* + - 1. Файл services/PlayerForegroundService.java

package com.siroytman.indiewindymobile.services;

import android.app.Notification;

import android.app.NotificationChannel;

import android.app.NotificationManager;

import android.app.PendingIntent;

import android.app.Service;

import android.content.Context;

import android.content.Intent;

import android.graphics.Bitmap;

import android.net.Uri;

import android.os.Build;

import android.os.Bundle;

import android.os.IBinder;

import android.util.Log;

import com.google.android.exoplayer2.ExoPlayerFactory;

import com.google.android.exoplayer2.Player;

import com.google.android.exoplayer2.SimpleExoPlayer;

import com.google.android.exoplayer2.source.ConcatenatingMediaSource;

import com.google.android.exoplayer2.source.ProgressiveMediaSource;

import com.google.android.exoplayer2.ui.PlayerNotificationManager;

import com.google.android.exoplayer2.upstream.DataSource;

import com.google.android.exoplayer2.upstream.DefaultDataSourceFactory;

import com.siroytman.indiewindymobile.model.UserSongLink;

import java.util.ArrayList;

import androidx.annotation.Nullable;

import androidx.core.content.ContextCompat;

public class PlayerForegroundService extends Service {

public static final String TAG = "PlayerForegroundService";

private static final String CHANNEL\_ID = "PlayerForegroundServiceChannel";

private static final int NOTIFICATION\_ID = 2;

public static PlayerForegroundService instance;

private UserSongLink songLink;

private ArrayList<UserSongLink> songLinks;

private ConcatenatingMediaSource concatenatingMediaSource;

private PlayerServiceConnection playerActivityConnection;

PlayerNotificationManager playerNotificationManager;

private SimpleExoPlayer player;

private boolean playWhenReady = true;

private int currentWindow;

private long playbackPosition = 0;

public static synchronized PlayerForegroundService getInstance() {

return instance;

}

public static void startService(Context context, ArrayList<UserSongLink> songLinks, int songPos) {

Log.d(TAG, "startService begin");

if(instance != null) {

Log.d(TAG, "instance not null");

if (instance.songLink.getSong().equals(songLinks.get(songPos).getSong())) {

Log.d(TAG, "Instance have same song");

// Set new playerView player

PlayerServiceConnection.getInstance().getPlayerActivity().setPlayer(instance.player);

return;

}

instance.stopService();

}

Intent serviceIntent = new Intent(context, PlayerForegroundService.class);

serviceIntent.putExtra("songLinks", songLinks);

serviceIntent.putExtra("songPos", songPos);

ContextCompat.startForegroundService(context, serviceIntent);

Log.d(TAG, "startService end");

}

// public static void stopService(Context context) {

// Intent serviceIntent = new Intent(context, PlayerForegroundService.class);

// stopService(serviceIntent);

// }

public void stopService() {

Log.d(TAG, "stopService begin");

instance = null;

stopSelf();

Log.d(TAG, "stopService end");

}

@Override

public void onDestroy() {

super.onDestroy();

releasePlayer();

Log.d(TAG, "Destroyed");

}

@Override

public void onCreate() {

Log.d(TAG, "onCreate begin");

super.onCreate();

instance = this;

playerNotificationManager = new PlayerNotificationManager(

getApplicationContext(),

CHANNEL\_ID,

NOTIFICATION\_ID,

new PlayerNotificationDescriptionAdapter());

Log.d(TAG, "onCreate end");

}

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

Bundle arguments = intent.getExtras();

if(arguments != null) {

songLinks = arguments.getParcelableArrayList("songLinks");

currentWindow = arguments.getInt("songPos");

songLink = songLinks.get(currentWindow);

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

initializePlayer();

createNotification();

return START\_NOT\_STICKY;

}

private void createNotification() {

Context context = getApplicationContext();

createNotificationChannel();

Notification notification = new Notification.Builder(context, CHANNEL\_ID)

// Make the transport controls visible on the lockscreen

.setVisibility(Notification.VISIBILITY\_PUBLIC)

// Take advantage of MediaStyle features

.setStyle(new Notification.MediaStyle()

.setShowActionsInCompactView(0, 1, 2)

)

.build();

startForeground(NOTIFICATION\_ID, notification);

}

private void initializePlayer() {

Log.d(TAG, "initializePlayer");

player = ExoPlayerFactory.newSimpleInstance(this);

concatenatingMediaSource = buildConcatenatingMediaSource(songLinks);

player.setPlayWhenReady(playWhenReady);

player.seekTo(currentWindow, playbackPosition);

player.prepare(concatenatingMediaSource, false, false);

Log.d(TAG, "Play: " + songLink.getSong().getName());

player.addListener(new Player.EventListener() {

@Override

public void onPositionDiscontinuity(int reason) {

//THIS METHOD GETS CALLED FOR EVERY NEW SOURCE THAT IS PLAYED

int latestWindowIndex = player.getCurrentWindowIndex();

if (latestWindowIndex == currentWindow) {

return;

}

// item selected in playlist has changed, handle here

currentWindow = latestWindowIndex;

songLink = songLinks.get(currentWindow);

Log.d(TAG, "Song in playlist changed: pos=" + currentWindow + "; songName=" + songLink.getSong().getName());

playerActivityConnection = PlayerServiceConnection.getInstance();

if (playerActivityConnection != null) {

playerActivityConnection.getPlayerActivity().updateSongView(currentWindow);

}

}

});

PlayerServiceConnection.getInstance().getPlayerActivity().setPlayer(player);

}

private ConcatenatingMediaSource buildConcatenatingMediaSource(ArrayList<UserSongLink> songLinks) {

DataSource.Factory dataSourceFactory =

new DefaultDataSourceFactory(this, "indiewindy\_exoplayer");

ConcatenatingMediaSource concatenatingMediaSource = new ConcatenatingMediaSource();

for (int i = 0; i < songLinks.size(); ++i) {

Uri uri = Uri.parse(songLinks.get(i).getSong().getSongUrl());

concatenatingMediaSource.addMediaSource(

new ProgressiveMediaSource.Factory(dataSourceFactory)

.createMediaSource(uri));

}

return concatenatingMediaSource;

}

@Nullable

@Override

public IBinder onBind(Intent intent) {

return null;

}

private void createNotificationChannel() {

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {

NotificationChannel serviceChannel = new NotificationChannel(

CHANNEL\_ID,

CHANNEL\_ID,

NotificationManager.IMPORTANCE\_DEFAULT

);

serviceChannel.setSound(null, null);

playerNotificationManager.setPlayer(player);

NotificationManager manager = getSystemService(NotificationManager.class);

manager.createNotificationChannel(serviceChannel);

}

}

private void releasePlayer() {

if (player != null) {

playWhenReady = player.getPlayWhenReady();

playbackPosition = player.getCurrentPosition();

currentWindow = player.getCurrentWindowIndex();

player.release();

player = null;

Log.d(TAG, "Stop: " + songLink.getSong().getName());

}

}

private class PlayerNotificationDescriptionAdapter implements

PlayerNotificationManager.MediaDescriptionAdapter

{

public static final String TAG = "PlayerNotificationDescriptionAdapter";

@Override

public String getCurrentContentTitle(Player player) {

return songLink.getSong().getName();

}

@Nullable

@Override

public String getCurrentContentText(Player player) {

return songLink.getSong().getArtist().getName();

}

@Nullable

@Override

public Bitmap getCurrentLargeIcon(Player player,

PlayerNotificationManager.BitmapCallback callback) {

return null;

}

@Nullable

@Override

public PendingIntent createCurrentContentIntent(Player player) {

int window = player.getCurrentWindowIndex();

return PendingIntent.getBroadcast(getApplicationContext(), 0, new Intent(), window);

}

}

}

* + - 1. Файл services/ PlayerServiceConnection.java

package com.siroytman.indiewindymobile.services;

import android.util.Log;

import com.siroytman.indiewindymobile.ui.activity.PlayerActivity;

public class PlayerServiceConnection {

public static final String TAG = "PlayerServiceConnection";

private static PlayerServiceConnection instance;

private PlayerActivity playerActivity;

private PlayerServiceConnection(PlayerActivity playerActivity) {

this.playerActivity = playerActivity;

instance = this;

}

public static void createInstance(PlayerActivity playerActivity) {

if (instance == null) {

new PlayerServiceConnection(playerActivity);

} else {

instance.playerActivity = playerActivity;

}

}

public static PlayerServiceConnection getInstance() {

if(instance == null) {

Log.d(TAG, "Instance is null");

}

return instance;

}

public static void destroyInstance() {

instance = null;

}

public PlayerActivity getPlayerActivity() {

return playerActivity;

}

}

* + - 1. Файл services/ SharedPrefsService.java

package com.siroytman.indiewindymobile.services;

import android.content.Context;

import android.content.SharedPreferences;

import com.google.gson.Gson;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.model.AppUser;

public class SharedPrefsService {

private SharedPreferences preferences;

private Gson gson;

public SharedPrefsService() {

Context context = AppController.getContext();

this.preferences = context.getSharedPreferences("IndieWindyPrefs", Context.MODE\_PRIVATE);

gson = new Gson();

}

public void saveAppUser(AppUser user){

String json = gson.toJson(user);

preferences.edit().putString("AppUser", json).apply();

}

public AppUser getAppUser(){

String json = preferences.getString("AppUser", "");

if (json.isEmpty()){

return null;

}

return gson.fromJson(json, AppUser.class);

}

public void remove(){

preferences.edit().clear().apply();

}

}

* + - 1. Файл ui/activity/AlbumActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.graphics.drawable.Icon;

import android.os.Bundle;

import android.util.Log;

import android.view.MenuItem;

import android.view.View;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AlbumController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Album;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.fragments.links.UserSongLinkListFragment;

import java.util.ArrayList;

import androidx.appcompat.view.menu.MenuBuilder;

import androidx.appcompat.view.menu.MenuPopupHelper;

import androidx.appcompat.widget.PopupMenu;

import androidx.fragment.app.FragmentActivity;

public class AlbumActivity extends FragmentActivity implements ILinkAdd<Album> {

public static final String TAG = "AlbumActivity";

@SuppressLint("StaticFieldLeak")

private static AlbumActivity activity;

private AlbumController albumController;

private Album album;

private Boolean linkExist = false;

private ImageView albumPhoto;

private TextView albumName;

private TextView albumArtistName;

private ImageView albumAddButton;

private ImageView albumOptionsButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_album);

activity = this;

// Get album from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

album = arguments.getParcelable(Album.class.getSimpleName());

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

albumName = findViewById(R.id.album\_activity\_\_album\_name);

albumArtistName = findViewById(R.id.album\_activity\_\_album\_artist\_name);

albumPhoto = findViewById(R.id.album\_activity\_\_album\_photo);

albumAddButton = findViewById(R.id.album\_activity\_\_album\_add\_button);

albumOptionsButton = findViewById(R.id.album\_activity\_\_album\_options\_button);

albumName.setText(album.getName());

albumArtistName.setText(album.getArtist().getName());

Glide.with(this).load(album.getImageUrl()).into(albumPhoto);

final AlbumActivity activity = this;

albumController = AlbumController.getInstance();

albumController.linkExist(activity);

albumController.getAlbumSongs(activity);

albumAddButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (linkExist) {

albumController.removeUserAlbumLink(activity);

} else {

albumController.addUserAlbumLink(activity);

}

}

});

albumOptionsButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

showPopupMenu(activity, v, album);

}

});

}

public void songsFoundViewUpdate(ArrayList<UserSongLink> links)

{

// Load list fragment

UserSongLinkListFragment fragment = new UserSongLinkListFragment(links, true, false);

FragmentService.replaceFragment(activity, R.id.album\_activity\_\_album\_song\_list\_container, fragment);

}

@Override

public Album getItem() {

return album;

}

@Override

public void removed() {

IconChanger.setIconAdd(albumAddButton);

linkExist = false;

}

@Override

public void added() {

IconChanger.setIconCheck(albumAddButton);

linkExist = true;

}

@SuppressLint("RestrictedApi")

public static void showPopupMenu(final Context context, View v, final Album album) {

PopupMenu menu = new PopupMenu(context, v);

menu.inflate(R.menu.popup\_album\_menu);

menu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {

@Override

public boolean onMenuItemClick(MenuItem item) {

switch (item.getItemId()) {

case R.id.album\_options\_menu\_artist:

// If not already in ArtistActivity

if (!(context instanceof ArtistActivity)) {

Log.d(TAG, "to artist of album: " + album.getName());

Intent intent = new Intent(context, ArtistActivity.class);

intent.putExtra(Artist.class.getSimpleName(), album.getArtist());

context.startActivity(intent);

return true;

}

return true;

default:

return false;

}

}

});

menu.setOnDismissListener(new PopupMenu.OnDismissListener() {

@Override

public void onDismiss(PopupMenu menu) {

}

});

MenuPopupHelper menuHelper = new MenuPopupHelper(context, (MenuBuilder) menu.getMenu(), v);

menuHelper.setForceShowIcon(true);

menuHelper.show();

}

}

* + - 1. Файл ui/activity/ArtistActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.annotation.SuppressLint;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ArtistController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.fragments.links.UserAlbumLinkListFragment;

import java.util.ArrayList;

import androidx.appcompat.app.AppCompatActivity;

public class ArtistActivity extends AppCompatActivity implements ILinkAdd<Artist> {

public static final String TAG = "ArtistActivity";

@SuppressLint("StaticFieldLeak")

private static ArtistActivity activity;

private ArtistController artistController;

private Artist artist;

private Boolean linkExist = false;

private ImageView artistPhoto;

private TextView artistName;

private TextView artistDescription;

private ImageView artistAddButton;

private ImageView artistDonateButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_artist);

activity = this;

// Get artist from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

artist = arguments.getParcelable(Artist.class.getSimpleName());

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

artistPhoto = findViewById(R.id.artist\_activity\_\_artist\_photo);

artistName = findViewById(R.id.artist\_activity\_\_artist\_name);

artistDescription = findViewById(R.id.artist\_activity\_\_artist\_description);

artistAddButton = findViewById(R.id.artist\_activity\_\_artist\_add\_button);

artistDonateButton = findViewById(R.id.artist\_activity\_\_artist\_donate\_button);

artistName.setText(artist.getName());

artistDescription.setText(artist.getDescription());

Glide.with(activity).load(artist.getImageUrl()).into(artistPhoto);

artistController = ArtistController.getInstance();

artistController.getArtistAlbums(activity, artist);

artistController.linkExist(activity);

artistAddButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (linkExist) {

artistController.removeUserArtistLink(activity);

} else {

artistController.addUserArtistLink(activity);

}

}

});

artistDonateButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(ArtistActivity.this, DonationActivity.class);

intent.putExtra(Artist.class.getSimpleName(), artist);

startActivity(intent);

}

});

}

public void albumsFoundViewUpdate(ArrayList<UserAlbumLink> links)

{

// Load list fragment

UserAlbumLinkListFragment fragment = new UserAlbumLinkListFragment(links, artist);

FragmentService.replaceFragment(activity, R.id.artist\_activity\_\_albums\_container, fragment);

}

@Override

public Artist getItem() {

return artist;

}

@Override

public void removed() {

IconChanger.setIconAdd(artistAddButton);

linkExist = false;

}

@Override

public void added() {

IconChanger.setIconCheck(artistAddButton);

linkExist = true;

}

}

* + - 1. Файл ui/activity/ConcertActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.annotation.SuppressLint;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.text.method.ScrollingMovementMethod;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Concert;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.fragments.links.UserArtistLinkListFragment;

import java.util.ArrayList;

import androidx.appcompat.app.AppCompatActivity;

public class ConcertActivity extends AppCompatActivity implements ILinkAdd<Concert> {

public static final String TAG = "ConcertActivity";

private UserConcertLink concertLink;

@SuppressLint("StaticFieldLeak")

private static ConcertActivity activity;

private ConcertController concertController;

private ImageView concertPhoto;

private TextView concertName;

private TextView concertDate;

private TextView concertDescription;

private TextView concertAddress;

private ImageView concertAddButton;

private Button concertTicketButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_concert);

activity = this;

// Get concertLink from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

concertLink = arguments.getParcelable(UserConcertLink.class.getSimpleName());

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

concertPhoto = findViewById(R.id.concert\_activity\_\_concert\_photo);

concertName = findViewById(R.id.concert\_activity\_\_concert\_name);

concertDate = findViewById(R.id.concert\_activity\_\_concert\_date);

concertDescription = findViewById(R.id.concert\_activity\_\_concert\_description);

concertAddress = findViewById(R.id.concert\_activity\_\_concert\_address);

concertAddButton = findViewById(R.id.concert\_activity\_\_concert\_add\_button);

concertTicketButton = findViewById(R.id.concert\_activity\_\_ticket\_button);

IconChanger.setAddStateIcon(concertLink, concertAddButton);

final Concert concert = concertLink.getConcert();

concertName.setText(concert.getName());

concertDescription.setText(concert.getDescription());

concertDate.setText(concert.getDateString());

concertDescription.setMovementMethod(new ScrollingMovementMethod());

concertAddress.setText(concert.getAddress());

concertAddress.setMovementMethod(new ScrollingMovementMethod());

Glide.with(activity).load(concert.getImageUrl()).into(concertPhoto);

concertController = ConcertController.getInstance();

concertController.getArtists(this, concert.getId());

concertAddButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (!concertLink.isEmpty()) {

concertController.removeUserConcertLink(activity);

} else {

concertController.addUserConcertLink(activity);

}

}

});

concertTicketButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent browserIntent = new Intent(Intent.ACTION\_VIEW, Uri.parse(concert.getTicketLink()));

startActivity(browserIntent);

}

});

}

public void artistsFoundViewUpdate(ArrayList<UserArtistLink> links)

{

// Load list fragment

UserArtistLinkListFragment fragment = new UserArtistLinkListFragment(links);

FragmentService.replaceFragment(activity, R.id.concert\_activity\_\_artists\_container, fragment);

}

@Override

public Concert getItem() {

return concertLink.getConcert();

}

@Override

public void removed() {

IconChanger.setIconAdd(concertAddButton);

concertLink.makeEmpty();

}

@Override

public void added() {

IconChanger.setIconCheck(concertAddButton);

concertLink.setAppUserId(AppController.user.getId());

concertLink.setConcertId(concertLink.getConcert().getId());

}

}

* + - 1. Файл ui/activity/DonationActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.DonationController;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.services.KeyboardService;

import androidx.appcompat.app.AppCompatActivity;

public class DonationActivity extends AppCompatActivity {

public static final String TAG = "ArtistActivity";

private DonationController donationController;

private Artist artist;

private TextView artistName;

private ImageView artistPhoto;

private EditText donationAmount;

private Button donationButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_donation);

// Get artist from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

artist = arguments.getParcelable(Artist.class.getSimpleName());

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

donationController = DonationController.getInstance();

artistName = findViewById(R.id.donation\_activity\_artist\_name);

artistPhoto = findViewById(R.id.donation\_activity\_artist\_image);

donationAmount = findViewById(R.id.donation\_activity\_amount);

donationButton = findViewById(R.id.donation\_activity\_donation\_button);

artistName.setText(artist.getName());

Glide.with(this).load(artist.getImageUrl()).into(artistPhoto);

donationButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

KeyboardService.hideKeyboard(DonationActivity.this);

int amount = Integer.parseInt(donationAmount.getText().toString());

donationController.addDonation(DonationActivity.this, artist, amount);

}

});

}

}

* + - 1. Файл ui/activity/LoginActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ProgressBar;

import android.widget.Toast;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.LoginController;

import androidx.appcompat.app.AppCompatActivity;

public class LoginActivity extends AppCompatActivity {

public static final String TAG = "LoginActivity";

LoginController loginController;

private EditText nameText;

private EditText passwordText;

private ProgressBar progressBar;

private Button loginButton;

private Button registerButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

loginController = LoginController.getInstance();

nameText = findViewById(R.id.login\_activity\_\_login);

passwordText = findViewById(R.id.login\_activity\_\_password);

progressBar = findViewById(R.id.login\_activity\_\_loading\_progress\_bar);

loginButton = findViewById(R.id.login\_activity\_\_login\_button);

registerButton = findViewById(R.id.login\_activity\_\_register\_button);

loginController.loginFromSharedPrefs(this);

}

public void setLoginAndPasswordText(String login, String password){

nameText.setText(login);

// Password is hashed here... Should be not

}

public void startLoadingProgressBar() {

Log.d(TAG, "startLoadingProgressBar");

passwordText.setEnabled(false);

nameText.setEnabled(false);

loginButton.setEnabled(false);

registerButton.setEnabled(false);

progressBar.setVisibility(View.VISIBLE);

}

public void stopLoadingProgressBar() {

Log.d(TAG, "stopLoadingProgressBar");

passwordText.setEnabled(true);

nameText.setEnabled(true);

loginButton.setEnabled(true);

registerButton.setEnabled(true);

progressBar.setVisibility(View.INVISIBLE);

}

// Authorization

public void loginOnClick(View view) {

String name = nameText.getText().toString();

String password = passwordText.getText().toString();

if (name.isEmpty() || password.isEmpty()) {

Toast.makeText(this, getString(R.string.login\_\_no\_login), Toast.LENGTH\_LONG)

.show();

return;

}

loginController.login(this, name, password);

}

// Registration

public void registerOnClick(View view) {

String name = nameText.getText().toString();

String password = passwordText.getText().toString();

if (name.isEmpty() || password.isEmpty()) {

Toast.makeText(this, getString(R.string.login\_\_no\_login), Toast.LENGTH\_LONG)

.show();

return;

}

loginController.register(this, name, password);

}

}

* + - 1. Файл ui/activity/MoreActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.os.Bundle;

import android.util.Log;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserAlbumLinkListFragment;

import com.siroytman.indiewindymobile.ui.fragments.links.UserArtistLinkListFragment;

import com.siroytman.indiewindymobile.ui.fragments.links.UserSongLinkListFragment;

import java.util.ArrayList;

import androidx.appcompat.app.AppCompatActivity;

public class MoreActivity extends AppCompatActivity {

public static final String TAG = "MoreActivity";

private ArrayList<UserSongLink> songLinks;

private ArrayList<UserAlbumLink> albumLinks;

private ArrayList<UserArtistLink> artistLinks;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_more);

// Get songLinks from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

songLinks = arguments.getParcelableArrayList("songLinks");

albumLinks = arguments.getParcelableArrayList("albumLinks");

artistLinks = arguments.getParcelableArrayList("artistLinks");

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

if (songLinks != null) {

// Load list fragment

UserSongLinkListFragment fragment = new UserSongLinkListFragment(songLinks, true, false);

FragmentService.replaceFragment(this, R.id.more\_activity\_\_container, fragment);

}

if (albumLinks != null) {

// Load list fragment

UserAlbumLinkListFragment fragment = new UserAlbumLinkListFragment(albumLinks, true, false);

FragmentService.replaceFragment(this, R.id.more\_activity\_\_container, fragment);

}

if (artistLinks != null) {

// Load list fragment

UserArtistLinkListFragment fragment = new UserArtistLinkListFragment(artistLinks, true);

FragmentService.replaceFragment(this, R.id.more\_activity\_\_container, fragment);

}

}

}

* + - 1. Файл ui/activity/NavigationActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.os.Bundle;

import com.google.android.material.bottomnavigation.BottomNavigationView;

import androidx.appcompat.app.AppCompatActivity;

import androidx.navigation.NavController;

import androidx.navigation.Navigation;

import androidx.navigation.ui.AppBarConfiguration;

import androidx.navigation.ui.NavigationUI;

import com.siroytman.indiewindymobile.R;

public class NavigationActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_navigation);

BottomNavigationView navView = findViewById(R.id.nav\_view);

// Passing each menu ID as a set of Ids because each

// menu should be considered as top level destinations.

AppBarConfiguration appBarConfiguration = new AppBarConfiguration.Builder(

R.id.navigation\_home, R.id.navigation\_search, R.id.navigation\_concert)

.build();

NavController navController = Navigation.findNavController(this, R.id.nav\_host\_fragment);

NavigationUI.setupActionBarWithNavController(this, navController, appBarConfiguration);

NavigationUI.setupWithNavController(navView, navController);

}

}

* + - 1. Файл ui/activity/PlayerActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.util.Log;

import android.view.MenuItem;

import android.view.View;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.google.android.exoplayer2.SimpleExoPlayer;

import com.google.android.exoplayer2.offline.DownloadService;

import com.google.android.exoplayer2.ui.PlayerView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.controller.SongController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Album;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.Song;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.DownloadTracker;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.services.PlayerForegroundService;

import com.siroytman.indiewindymobile.services.PlayerServiceConnection;

import java.util.ArrayList;

import androidx.appcompat.app.AppCompatActivity;

import androidx.appcompat.view.menu.MenuBuilder;

import androidx.appcompat.view.menu.MenuPopupHelper;

import androidx.appcompat.widget.PopupMenu;

public class PlayerActivity extends AppCompatActivity implements ILinkAdd<Song>, DownloadTracker.Listener {

public static final String TAG = "PlayerActivity";

private AppController appController;

private SongController songController;

public PlayerView playerView;

private UserSongLink songLink;

private int songPos;

private ArrayList<UserSongLink> songLinks;

private TextView songName;

private TextView artistName;

private ImageView songArtwork;

private ImageView optionsButton;

private ImageView addButton;

private ImageView downloadButton;

private DownloadTracker downloadTracker;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_player);

// Get album from bundle

Bundle arguments = getIntent().getExtras();

if(arguments != null) {

songLinks = arguments.getParcelableArrayList("songLinks");

songPos = arguments.getInt("songPos");

songLink = songLinks.get(songPos);

}

else {

Log.e(TAG, "Error: Arguments are null!");

}

appController = AppController.getInstance();

songController = SongController.getInstance();

playerView = findViewById(R.id.player\_view);

songArtwork = playerView.findViewById(R.id.artwork);

optionsButton = playerView.findViewById(R.id.player\_\_options\_button);

addButton = playerView.findViewById(R.id.player\_\_add\_button);

downloadButton = playerView.findViewById(R.id.player\_\_download\_button);

songName = playerView.findViewById(R.id.player\_\_song\_name);

artistName = playerView.findViewById(R.id.player\_\_artist\_name);

downloadTracker = appController.getDownloadTracker();

updateSongView(songPos);

optionsButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

showPopupMenu(PlayerActivity.this, optionsButton, songLink.getSong());

}

});

addButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (songLink.isEmpty()) {

songController.addUserSongLink(PlayerActivity.this);

} else{

songController.removeUserSongLink(PlayerActivity.this);

}

}

});

downloadButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Song song = songLink.getSong();

Uri uri = Uri.parse(song.getSongUrl());

downloadTracker.toggleDownload(

getSupportFragmentManager(),

song.getName(),

uri,

"",

null);

}

});

PlayerServiceConnection.createInstance(this);

PlayerForegroundService.startService(this, songLinks, songPos);

// Start the download service if it should be running but it's not currently.

// Starting the service in the foreground causes notification flicker if there is no scheduled

// action. Starting it in the background throws an exception if the app is in the background too

// (e.g. if device screen is locked).

try {

DownloadService.start(this, DownloadService.class);

} catch (IllegalStateException e) {

DownloadService.startForeground(this, DownloadService.class);

}

}

public void updateSongView(int songLinkPos) {

Log.d(TAG, "onSongChangedUpdateView");

songLink = songLinks.get(songLinkPos);

Glide.with(this).load(songLink.getSong().getAlbum().getImageUrl()).into(songArtwork);

IconChanger.setAddStateIcon(songLink, addButton);

songName.setText(songLink.getSong().getName());

artistName.setText(songLink.getSong().getArtist().getName());

}

public void setPlayer(SimpleExoPlayer player) {

playerView.setPlayer(player);

}

@Override

public void onDownloadsChanged() {

// TODO

Log.d(TAG, "onDownloadsChanged");

IconChanger.setIconCheck(downloadButton);

}

@Override

public void onStart() {

super.onStart();

downloadTracker.addListener(this);

}

@Override

public void onStop() {

downloadTracker.removeListener(this);

PlayerServiceConnection.destroyInstance();

super.onStop();

}

@Override

public Song getItem() {

return songLink.getSong();

}

@Override

public void removed() {

songLink.makeEmpty();

IconChanger.setIconAdd(addButton);

}

@Override

public void added() {

songLink.setAppUserId(AppController.user.getId());

songLink.setSongId(songLink.getSong().getId());

IconChanger.setIconCheck(addButton);

}

@SuppressLint("RestrictedApi")

public static void showPopupMenu(final Context context, View v, final Song song) {

PopupMenu menu = new PopupMenu(context, v);

menu.inflate(R.menu.popup\_song\_menu);

menu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {

@Override

public boolean onMenuItemClick(MenuItem item) {

switch (item.getItemId()) {

case R.id.song\_options\_menu\_artist:

// If not already in ArtistActivity

if (!(context instanceof ArtistActivity)) {

Log.d(TAG, "to artist of song: " + song.getName());

Intent intent = new Intent(context, ArtistActivity.class);

intent.putExtra(Artist.class.getSimpleName(), song.getArtist());

context.startActivity(intent);

return true;

}

return true;

case R.id.song\_options\_menu\_album:

// If not already in AlbumActivity

if (!(context instanceof AlbumActivity)) {

Log.d(TAG, "to album of song: " + song.getName());

Intent intent = new Intent(context, AlbumActivity.class);

intent.putExtra(Album.class.getSimpleName(), song.getAlbum());

context.startActivity(intent);

return true;

}

default:

return false;

}

}

});

menu.setOnDismissListener(new PopupMenu.OnDismissListener() {

@Override

public void onDismiss(PopupMenu menu) {

}

});

MenuPopupHelper menuHelper = new MenuPopupHelper(context, (MenuBuilder) menu.getMenu(), v);

menuHelper.setForceShowIcon(true);

menuHelper.show();

}

@Override

public void onWindowFocusChanged(boolean hasFocus) {

super.onWindowFocusChanged(hasFocus);

if (hasFocus) {

hideSystemUi();

}

}

@SuppressLint("InlinedApi")

private void hideSystemUi() {

playerView.setSystemUiVisibility(View.SYSTEM\_UI\_FLAG\_LOW\_PROFILE

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

| View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION);

}

}

* + - 1. Файл ui/activity/SettingsActivity.java

package com.siroytman.indiewindymobile.ui.activity;

import android.os.Bundle;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.LoginController;

import androidx.appcompat.app.ActionBar;

import androidx.appcompat.app.AppCompatActivity;

import androidx.preference.Preference;

import androidx.preference.PreferenceFragmentCompat;

public class SettingsActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.settings\_activity);

getSupportFragmentManager()

.beginTransaction()

.replace(R.id.settings, new SettingsFragment())

.commit();

}

public static class SettingsFragment extends PreferenceFragmentCompat {

@Override

public void onCreatePreferences(Bundle savedInstanceState, String rootKey) {

setPreferencesFromResource(R.xml.root\_preferences, rootKey);

Preference logout = findPreference("logout");

logout.setOnPreferenceClickListener(new Preference.OnPreferenceClickListener() {

@Override

public boolean onPreferenceClick(Preference preference) {

LoginController loginController = LoginController.getInstance();

loginController.logout(getContext());

return true;

}

});

}

}

}

* + - 1. Файл ui/adapter/ ArtistPostLinkListAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.text.method.ScrollingMovementMethod;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.FrameLayout;

import android.widget.ImageView;

import android.widget.RelativeLayout;

import android.widget.ScrollView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.PostController;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.ArtistPostLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.activity.ArtistActivity;

import com.siroytman.indiewindymobile.ui.fragments.links.UserSongLinkListFragment;

import java.util.ArrayList;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.cardview.widget.CardView;

import androidx.fragment.app.FragmentManager;

public class ArtistPostLinkListAdapter extends ArrayAdapter<ArtistPostLink> {

public static final String TAG = "PostsListAdapter";

private Context context;

private List<ArtistPostLink> links;

private FragmentManager fragmentManager;

public ArtistPostLinkListAdapter(Context context, int resource, List<ArtistPostLink> links, FragmentManager fragmentManager) {

super(context, resource, links);

this.context = context;

this.links = links;

this.fragmentManager = fragmentManager;

}

@NonNull

@Override

public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

ViewHolder viewHolder = null;

if(convertView == null) {

LayoutInflater inflater = (LayoutInflater) context.getSystemService(Activity.LAYOUT\_INFLATER\_SERVICE);

convertView = inflater.inflate(R.layout.post\_list\_item, null);

viewHolder = new ViewHolder(convertView, links.get(position));

convertView.setTag(viewHolder);

}

else {

viewHolder = (ViewHolder)convertView.getTag();

}

return convertView;

}

// Single element

public class ViewHolder implements View.OnClickListener {

public static final String TAG = "PostAdapter.VH";

private ArtistPostLink link;

private TextView postTextView;

private TextView postArtistNameView;

private ImageView postArtistPhotoView;

private TextView postDateView;

private RelativeLayout relativeLayout;

private FrameLayout frameLayout;

public ViewHolder(View convertView, final ArtistPostLink link) {

this.link = link;

postTextView = convertView.findViewById(R.id.post\_list\_item\_\_text);

postArtistNameView = convertView.findViewById(R.id.post\_list\_item\_\_artist\_name);

postArtistPhotoView = convertView.findViewById(R.id.post\_list\_item\_\_artist\_photo);

postDateView = convertView.findViewById(R.id.post\_list\_item\_\_date);

relativeLayout = convertView.findViewById(R.id.post\_list\_item\_\_layout);

Glide.with(convertView).load(link.getArtist().getImageUrl()).into(postArtistPhotoView);

getPostSongs();

postArtistNameView.setText(link.getArtist().getName());

postTextView.setText(link.getPost().getText());

postTextView.setMovementMethod(new ScrollingMovementMethod());

postDateView.setText(link.getPost().getDateString());

postArtistNameView.setOnClickListener(this);

postArtistPhotoView.setOnClickListener(this);

convertView.setOnClickListener(this);

}

private void getPostSongs() {

// Create frame layout

frameLayout = new FrameLayout(context);

frameLayout.setId(getPosition(link) + 1);

frameLayout.setScrollContainer(false);

PostController.getInstance().getPostSongs(this, link.getPostId());

}

public void updatePostSongsView(ArrayList<UserSongLink> links) {

// Dynamically add frame layout

RelativeLayout.LayoutParams layoutParams = new RelativeLayout.LayoutParams(

RelativeLayout.LayoutParams.MATCH\_PARENT, RelativeLayout.LayoutParams.WRAP\_CONTENT);

layoutParams.addRule(RelativeLayout.BELOW, postTextView.getId());

relativeLayout.addView(frameLayout, layoutParams);

UserSongLinkListFragment fragment = new UserSongLinkListFragment(links, true, false);

FragmentService.replaceFragment(fragmentManager, frameLayout.getId(), fragment);

}

@Override

public void onClick(View v) {

Log.d(TAG, "Post clicked");

Artist artist = link.getArtist();

switch (v.getId()) {

case R.id.post\_list\_item\_\_artist\_name:

case R.id.post\_list\_item\_\_artist\_photo:

Log.d(TAG, "to page of artist: " + artist.getName());

Intent intent = new Intent(context, ArtistActivity.class);

intent.putExtra(Artist.class.getSimpleName(), artist);

context.startActivity(intent);

return;

}

}

}

}

* + - 1. Файл ui/adapter/ ConcertPagerAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.content.Context;

import android.os.Parcelable;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.ui.fragments.concert.NearestConcertFragment;

import com.siroytman.indiewindymobile.ui.fragments.concert.SavedConcertFragment;

import com.siroytman.indiewindymobile.ui.fragments.concert.SubscriptionConcertFragment;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import androidx.fragment.app.FragmentStatePagerAdapter;

public class ConcertPagerAdapter extends FragmentStatePagerAdapter {

public static final String TAG = "ConcertPagerAdapter";

public static int NUM\_ITEMS = 3;

public ConcertPagerAdapter(FragmentManager fragmentManager) {

super(fragmentManager);

}

// Returns total number of pages

@Override

public int getCount() {

return NUM\_ITEMS;

}

// Returns the fragment to display for that page

@Override

public Fragment getItem(int position) {

switch (position) {

case 0:

return NearestConcertFragment.getInstance();

case 1:

return SubscriptionConcertFragment.getInstance();

case 2:

return SavedConcertFragment.getInstance();

default:

return null;

}

}

// Returns the page title for the top indicator

@Override

public CharSequence getPageTitle(int position) {

Context context = AppController.getContext();

// Drawable myDrawable = context.getDrawable(R.drawable.ic\_artist);

// String title = "DefaultTitle";

switch (position) {

case 0:

// myDrawable = context.getDrawable(R.drawable.ic\_artist);

// title = "MyTitle";

return context.getString(R.string.concert\_\_nearest\_title);

case 1:

// myDrawable = context.getDrawable(R.drawable.ic\_artist);

// title = "MyTitle";

return context.getString(R.string.concert\_\_subscriptions\_title);

case 2:

// myDrawable = context.getDrawable(R.drawable.ic\_check);

// title = "MyTitle";

return context.getString(R.string.concert\_\_saved\_title);

default:

return null;

}

// SpannableStringBuilder sb = new SpannableStringBuilder(" ");

// try {

// myDrawable.setBounds(5, 5, myDrawable.getIntrinsicWidth(), myDrawable.getIntrinsicHeight());

// ImageSpan span = new ImageSpan(myDrawable, DynamicDrawableSpan.ALIGN\_BASELINE);

// sb.setSpan(span, 0, 1, Spanned.SPAN\_EXCLUSIVE\_EXCLUSIVE);

// return sb;

// } catch (Exception e) {

// // handle exception

// return null;

// }

}

@Override

public void restoreState(Parcelable state, ClassLoader loader) {

//Do NOTHING;

}

}

* + - 1. Файл ui/adapter/ HomePagerAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.os.Parcelable;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.ui.fragments.home.LatestHomeFragment;

import com.siroytman.indiewindymobile.ui.fragments.home.SubscriptionHomeFragment;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import androidx.fragment.app.FragmentStatePagerAdapter;

public class HomePagerAdapter extends FragmentStatePagerAdapter {

public static final String TAG = "HomePagerAdapter";

public static int NUM\_ITEMS = 2;

public HomePagerAdapter(FragmentManager fragmentManager) {

super(fragmentManager);

}

// Returns total number of pages

@Override

public int getCount() {

return NUM\_ITEMS;

}

// Returns the fragment to display for that page

@Override

public Fragment getItem(int position) {

switch (position) {

case 0:

return SubscriptionHomeFragment.getInstance();

case 1:

return LatestHomeFragment.getInstance();

default:

return null;

}

}

// Returns the page title for the top indicator

@Override

public CharSequence getPageTitle(int position) {

switch (position) {

case 0:

return AppController.getContext().getString(R.string.home\_\_subscriptions\_title);

case 1:

return AppController.getContext().getString(R.string.home\_\_latest\_title);

}

return null;

}

@Override

public void restoreState(Parcelable state, ClassLoader loader) {

//Do NOTHING;

}

}

* + - 1. Файл ui/adapter/ PersonalPagerAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.ui.fragments.personal.AlbumsPersonalFragment;

import com.siroytman.indiewindymobile.ui.fragments.personal.ArtistsPersonalFragment;

import com.siroytman.indiewindymobile.ui.fragments.personal.SongsPersonalFragment;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import androidx.fragment.app.FragmentStatePagerAdapter;

public class PersonalPagerAdapter extends FragmentStatePagerAdapter {

public static final String TAG = "PersonalPagerAdapter";

public static int NUM\_ITEMS = 3;

public PersonalPagerAdapter(FragmentManager fragmentManager) {

super(fragmentManager);

}

// Returns total number of pages

@Override

public int getCount() {

return NUM\_ITEMS;

}

// Returns the fragment to display for that page

@Override

public Fragment getItem(int position) {

switch (position) {

case 0:

return SongsPersonalFragment.getInstance();

// case 1:

// return SongsPersonalFragment.getInstance();

case 1:

return ArtistsPersonalFragment.getInstance();

case 2:

return AlbumsPersonalFragment.getInstance();

default:

return null;

}

}

// Returns the page title for the top indicator

@Override

public CharSequence getPageTitle(int position) {

switch (position) {

case 0:

return AppController.getContext().getString(R.string.personal\_\_song\_title);

// case 1:

// return "Downloaded";

case 1:

return AppController.getContext().getString(R.string.personal\_\_artist\_title);

case 2:

return AppController.getContext().getString(R.string.personal\_\_album\_title);

}

return null;

}

}

* + - 1. Файл ui/adapter/ UserAlbumLinkListAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AlbumController;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Album;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.activity.AlbumActivity;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

public class UserAlbumLinkListAdapter extends ArrayAdapter<UserAlbumLink> {

public static final String TAG = "UserAlbumLinkListAdapter";

private AlbumController albumController;

private Context context;

private List<UserAlbumLink> linksList;

private Artist artist;

public UserAlbumLinkListAdapter(@NonNull Context context, int resource, @NonNull List<UserAlbumLink> linksList) {

super(context, resource, linksList);

this.context = context;

this.linksList = linksList;

this.albumController = AlbumController.getInstance();

}

public UserAlbumLinkListAdapter(@NonNull Context context, int resource, @NonNull List<UserAlbumLink> linksList, Artist artist) {

super(context, resource, linksList);

this.context = context;

this.linksList = linksList;

this.artist = artist;

this.albumController = AlbumController.getInstance();

}

@NonNull

@Override

public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

ViewHolder viewHolder = null;

final UserAlbumLink albumLink = linksList.get(position);

if(convertView == null) {

LayoutInflater inflater = (LayoutInflater) context.getSystemService(Activity.LAYOUT\_INFLATER\_SERVICE);

assert inflater != null;

convertView = inflater.inflate(R.layout.album\_list\_item, null);

if (artist != null){

viewHolder = new ViewHolder(convertView, albumLink, artist);

} else {

viewHolder = new ViewHolder(convertView, albumLink);

}

convertView.setTag(viewHolder);

}

else {

viewHolder = (ViewHolder)convertView.getTag();

}

return convertView;

}

// Single element

public class ViewHolder implements View.OnClickListener, ILinkAdd<Album> {

public static final String TAG = "UserAlbumLinkAdapter.VH";

private UserAlbumLink albumLink;

private ImageView albumPhotoView;

private TextView albumNameView;

private TextView albumArtistNameView;

private ImageView albumAddButton;

private ImageView albumOptionsButton;

public ViewHolder(View convertView, final UserAlbumLink albumLink) {

this.albumLink = albumLink;

albumPhotoView = convertView.findViewById(R.id.album\_list\_item\_\_album\_photo);

albumNameView = convertView.findViewById(R.id.album\_list\_item\_\_album\_name);

albumArtistNameView = convertView.findViewById(R.id.album\_list\_item\_\_album\_artist\_name);

albumAddButton = convertView.findViewById(R.id.album\_list\_item\_\_album\_add\_button);

albumOptionsButton = convertView.findViewById(R.id.album\_list\_item\_\_album\_options\_button);

Album album = albumLink.getAlbum();

albumNameView.setText(album.getName());

albumArtistNameView.setText(album.getArtist().getName());

Glide.with(convertView).load(album.getImageUrl()).into(albumPhotoView);

IconChanger.setAddStateIcon(albumLink, albumAddButton);

convertView.setOnClickListener(this);

albumAddButton.setOnClickListener(this);

albumOptionsButton.setOnClickListener(this);

}

public ViewHolder(View convertView, final UserAlbumLink albumLink, final Artist artist) {

this.albumLink = albumLink;

albumPhotoView = convertView.findViewById(R.id.album\_list\_item\_\_album\_photo);

albumNameView = convertView.findViewById(R.id.album\_list\_item\_\_album\_name);

albumArtistNameView = convertView.findViewById(R.id.album\_list\_item\_\_album\_artist\_name);

albumAddButton = convertView.findViewById(R.id.album\_list\_item\_\_album\_add\_button);

albumOptionsButton = convertView.findViewById(R.id.album\_list\_item\_\_album\_options\_button);

albumLink.getAlbum().setArtist(artist);

Album album = albumLink.getAlbum();

albumNameView.setText(album.getName());

albumArtistNameView.setText(album.getArtist().getName());

Glide.with(convertView).load(album.getImageUrl()).into(albumPhotoView);

IconChanger.setAddStateIcon(albumLink, albumAddButton);

convertView.setOnClickListener(this);

albumAddButton.setOnClickListener(this);

albumOptionsButton.setOnClickListener(this);

}

@Override

public void onClick(View v) {

Album album = albumLink.getAlbum();

Log.d(TAG, "onClick album: " + album.getName());

switch (v.getId()) {

case R.id.album\_list\_item\_\_album\_add\_button:

Log.d(TAG, "onClick album\_add\_button: " + album.getName());

if (albumLink.isEmpty()) {

albumController.addUserAlbumLink(this);

} else{

albumController.removeUserAlbumLink(this);

}

return;

case R.id.album\_list\_item\_\_album\_options\_button:

Log.d(TAG, "onClick album\_options\_button: " + album.getName());

AlbumActivity.showPopupMenu(getContext(), v, album);

return;

}

Log.d(TAG, "to page of album: " + album.getName());

Intent intent = new Intent(context, AlbumActivity.class);

intent.putExtra(Album.class.getSimpleName(), album);

context.startActivity(intent);

}

@Override

public Album getItem() {

return albumLink.getAlbum();

}

@Override

public void removed() {

albumLink.makeEmpty();

IconChanger.setIconAdd(albumAddButton);

}

@Override

public void added() {

albumLink.setAppUserId(AppController.user.getId());

albumLink.setAlbumId(albumLink.getAlbum().getId());

IconChanger.setIconCheck(albumAddButton);

}

}

}

* + - 1. Файл ui/adapter/ UserArtistLinkListAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.controller.ArtistController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.activity.ArtistActivity;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

public class UserArtistLinkListAdapter extends ArrayAdapter<UserArtistLink> {

public static final String TAG = "UserArtistLinkListAdapter";

private ArtistController artistController;

private Context context;

private List<UserArtistLink> linksList;

public UserArtistLinkListAdapter(@NonNull Context context, int resource, @NonNull List<UserArtistLink> linksList) {

super(context, resource, linksList);

this.context = context;

this.linksList = linksList;

this.artistController = ArtistController.getInstance();

}

@NonNull

@Override

public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

ViewHolder viewHolder = null;

final UserArtistLink artistLink = linksList.get(position);

if(convertView == null) {

LayoutInflater inflater = (LayoutInflater) context.getSystemService(Activity.LAYOUT\_INFLATER\_SERVICE);

assert inflater != null;

convertView = inflater.inflate(R.layout.artist\_list\_item, null);

viewHolder = new ViewHolder(convertView, artistLink);

convertView.setTag(viewHolder);

}

else {

viewHolder = (ViewHolder)convertView.getTag();

}

return convertView;

}

// Single element

public class ViewHolder implements View.OnClickListener, ILinkAdd<Artist> {

public static final String TAG = "UserArtistLinkAdapterVH";

private UserArtistLink artistLink;

private ImageView artistPhotoView;

private TextView artistNameView;

private ImageView artistAddButton;

public ViewHolder(View convertView, final UserArtistLink artistLink) {

this.artistLink = artistLink;

artistPhotoView = convertView.findViewById(R.id.artist\_list\_item\_\_artist\_photo);

artistNameView = convertView.findViewById(R.id.artist\_list\_item\_\_artist\_name);

artistAddButton = convertView.findViewById(R.id.artist\_list\_item\_\_add\_button);

Artist artist = artistLink.getArtist();

artistNameView.setText(artist.getName());

Glide.with(convertView).load(artist.getImageUrl()).into(artistPhotoView);

IconChanger.setAddStateIcon(artistLink, artistAddButton);

convertView.setOnClickListener(this);

artistAddButton.setOnClickListener(this);

}

@Override

public void onClick(View v) {

Artist artist = artistLink.getArtist();

Log.d(TAG, "onClick artist: " + artist.getName());

switch (v.getId()) {

case R.id.artist\_list\_item\_\_add\_button:

Log.d(TAG, "onClick artist\_add\_button: " + artist.getName());

if (artistLink.isEmpty()) {

artistController.addUserArtistLink(this);

} else{

artistController.removeUserArtistLink(this);

}

return;

}

Log.d(TAG, "to page of artist: " + artist.getName());

Intent intent = new Intent(context, ArtistActivity.class);

intent.putExtra(Artist.class.getSimpleName(), artist);

context.startActivity(intent);

}

@Override

public Artist getItem() {

return artistLink.getArtist();

}

@Override

public void removed() {

artistLink.makeEmpty();

IconChanger.setIconAdd(artistAddButton);

}

@Override

public void added() {

artistLink.setAppUserId(AppController.user.getId());

artistLink.setArtistId(artistLink.getArtist().getId());

IconChanger.setIconCheck(artistAddButton);

}

}

}

* + - 1. Файл ui/adapter/ UserConcertLinkListAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import com.bumptech.glide.Glide;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Concert;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.activity.ConcertActivity;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

public class UserConcertLinkListAdapter extends ArrayAdapter<UserConcertLink> {

public static final String TAG = "UserConcertLinkListAdapter";

private ConcertController concertController;

private Context context;

private List<UserConcertLink> linksList;

public UserConcertLinkListAdapter(@NonNull Context context, int resource, @NonNull List<UserConcertLink> linksList) {

super(context, resource, linksList);

this.context = context;

this.linksList = linksList;

this.concertController = ConcertController.getInstance();

}

@NonNull

@Override

public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

ViewHolder viewHolder = null;

final UserConcertLink concertLink = linksList.get(position);

if(convertView == null) {

LayoutInflater inflater = (LayoutInflater) context.getSystemService(Activity.LAYOUT\_INFLATER\_SERVICE);

convertView = inflater.inflate(R.layout.concert\_list\_item, null);

viewHolder = new ViewHolder(convertView, concertLink);

convertView.setTag(viewHolder);

}

else {

viewHolder = (ViewHolder)convertView.getTag();

}

return convertView;

}

// Single element

public class ViewHolder implements View.OnClickListener, ILinkAdd<Concert> {

public static final String TAG = "UserConcertLinkAdapt.VH";

private UserConcertLink concertLink;

private ImageView concertPhotoView;

private TextView concertNameView;

private TextView concertDateView;

private TextView concertDescriptionView;

private ImageView concertAddButton;

public ViewHolder(View convertView, final UserConcertLink concertLink) {

this.concertLink = concertLink;

concertPhotoView = convertView.findViewById(R.id.concert\_list\_item\_\_photo);

concertNameView = convertView.findViewById(R.id.concert\_list\_item\_\_name);

concertDateView = convertView.findViewById(R.id.concert\_list\_item\_\_date);

concertDescriptionView = convertView.findViewById(R.id.concert\_list\_item\_\_description);

concertAddButton = convertView.findViewById(R.id.concert\_list\_item\_\_\_add\_button);

Concert concert = concertLink.getConcert();

Glide.with(convertView).load(concert.getImageUrl()).into(concertPhotoView);

concertNameView.setText(concert.getName());

concertDateView.setText(concert.getDateString());

concertDescriptionView.setText(concert.getDescription());

IconChanger.setAddStateIcon(concertLink, concertAddButton);

convertView.setOnClickListener(this);

concertAddButton.setOnClickListener(this);

}

@Override

public void onClick(View v) {

Concert concert = concertLink.getConcert();

Log.d(TAG, "onClick concert: " + concert.getName());

switch (v.getId()) {

case R.id.concert\_list\_item\_\_\_add\_button:

Log.d(TAG, "onClick concert\_add\_button: " + concert.getName());

if (concertLink.isEmpty()) {

concertController.addUserConcertLink(this);

} else{

concertController.removeUserConcertLink(this);

}

return;

}

Log.d(TAG, "to page of concert: " + concert.getName());

Intent intent = new Intent(context, ConcertActivity.class);

intent.putExtra(UserConcertLink.class.getSimpleName(), concertLink);

context.startActivity(intent);

}

@Override

public Concert getItem() {

return concertLink.getConcert();

}

@Override

public void removed() {

concertLink.makeEmpty();

IconChanger.setIconAdd(concertAddButton);

}

@Override

public void added() {

concertLink.setAppUserId(AppController.user.getId());

concertLink.setConcertId(concertLink.getConcert().getId());

IconChanger.setIconCheck(concertAddButton);

}

}

}

* + - 1. Файл ui/adapter/ UserSongLinkListAdapter.java

package com.siroytman.indiewindymobile.ui.adapter;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AppController;

import com.siroytman.indiewindymobile.controller.SongController;

import com.siroytman.indiewindymobile.interfaces.ILinkAdd;

import com.siroytman.indiewindymobile.model.Song;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.IconChanger;

import com.siroytman.indiewindymobile.ui.activity.PlayerActivity;

import java.util.ArrayList;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

public class UserSongLinkListAdapter extends ArrayAdapter<UserSongLink> {

public static final String TAG = "UserSongLinkListAdapter";

private SongController songController;

private Context context;

private ArrayList<UserSongLink> songLinks;

public UserSongLinkListAdapter(@NonNull Context context, int resource, @NonNull ArrayList<UserSongLink> songLinks) {

super(context, resource, songLinks);

this.context = context;

this.songLinks = songLinks;

this.songController = SongController.getInstance();

}

@NonNull

@Override

public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

ViewHolder viewHolder = null;

final UserSongLink songLink = songLinks.get(position);

if(convertView == null) {

LayoutInflater inflater = (LayoutInflater) context.getSystemService(Activity.LAYOUT\_INFLATER\_SERVICE);

assert inflater != null;

convertView = inflater.inflate(R.layout.song\_list\_item, null);

viewHolder = new ViewHolder(convertView, songLink, position);

convertView.setTag(viewHolder);

}

else {

viewHolder = (ViewHolder)convertView.getTag();

}

return convertView;

}

// Single element

public class ViewHolder implements View.OnClickListener, ILinkAdd<Song> {

public static final String TAG = "UserSongLinkAdapter.VH";

private int songPos;

private UserSongLink songLink;

private TextView songNameView;

private TextView songArtistNameView;

private ImageView songAddButton;

private ImageView songOptionsButton;

private ImageView songDownloadButton;

public ViewHolder(View convertView, final UserSongLink songLink, final int songPos) {

this.songLink = songLink;

this.songPos = songPos;

songNameView = convertView.findViewById(R.id.song\_name);

songArtistNameView = convertView.findViewById(R.id.artist\_activity\_\_artist\_name);

songAddButton = convertView.findViewById(R.id.song\_add\_button);

songOptionsButton = convertView.findViewById(R.id.song\_options\_button);

songDownloadButton = convertView.findViewById(R.id.song\_download\_button);

songNameView.setText(songLink.getSong().getName());

songArtistNameView.setText(songLink.getSong().getArtist().getName());

IconChanger.setAddStateIcon(songLink, songAddButton);

convertView.setOnClickListener(this);

songAddButton.setOnClickListener(this);

songOptionsButton.setOnClickListener(this);

songDownloadButton.setOnClickListener(this);

}

@Override

public void onClick(View v) {

Song song = songLink.getSong();

Log.d(TAG, "onClick song: " + song.getName());

switch (v.getId()) {

case R.id.song\_add\_button:

if (songLink.isEmpty()) {

songController.addUserSongLink(this);

} else{

songController.removeUserSongLink(this);

}

return;

case R.id.song\_options\_button:

PlayerActivity.showPopupMenu(getContext(), v, song);

return;

case R.id.song\_download\_button:

// TODO

return;

}

Log.d(TAG, "to player starting starting with song: " + songLink.getSong().getName());

Intent intent = new Intent(context, PlayerActivity.class);

intent.putExtra("songLinks", songLinks);

intent.putExtra("songPos", songPos);

context.startActivity(intent);

}

@Override

public Song getItem() {

return songLink.getSong();

}

@Override

public void removed() {

songLink.makeEmpty();

IconChanger.setIconAdd(songAddButton);

}

@Override

public void added() {

songLink.setAppUserId(AppController.user.getId());

songLink.setSongId(songLink.getSong().getId());

IconChanger.setIconCheck(songAddButton);

}

}

}

* + - 1. Файл ui/custom\_views/ MyListView.java

package com.siroytman.indiewindymobile.ui.custom\_views;

import android.content.Context;

import android.util.AttributeSet;

import android.widget.ListView;

public class MyListView extends ListView {

public MyListView (Context context, AttributeSet attrs) {

super(context, attrs);

}

public MyListView (Context context) {

super(context);

}

public MyListView (Context context, AttributeSet attrs, int defStyle) {

super(context, attrs, defStyle);

}

@Override

public void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {

int expandSpec = MeasureSpec.makeMeasureSpec(Integer.MAX\_VALUE >> 2,

MeasureSpec.AT\_MOST);

super.onMeasure(widthMeasureSpec, expandSpec);

}

}

* + - 1. Файл ui/fragments /concert/ ConcertFragment.java

package com.siroytman.indiewindymobile.ui.fragments.concert;

import android.app.SearchManager;

import android.content.Context;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.Spinner;

import com.google.android.material.tabs.TabLayout;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.services.KeyboardService;

import com.siroytman.indiewindymobile.ui.adapter.ConcertPagerAdapter;

import androidx.annotation.Nullable;

import androidx.appcompat.widget.SearchView;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentStatePagerAdapter;

import androidx.viewpager.widget.ViewPager;

public class ConcertFragment extends Fragment {

private static final String TAG = "ConcertFragment";

private ConcertController concertController;

private SearchView searchView = null;

private FragmentStatePagerAdapter adapterViewPager;

private ViewPager pager;

private int currentPage = 0;

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_concert, container, false);

pager = view.findViewById(R.id.concert\_vp\_pager);

pager.setOffscreenPageLimit(ConcertPagerAdapter.NUM\_ITEMS);

adapterViewPager = new ConcertPagerAdapter(getChildFragmentManager());

pager.setAdapter(adapterViewPager);

TabLayout tabLayout = view.findViewById(R.id.concert\_tab\_layout);

tabLayout.setupWithViewPager(pager);

// Attach the page change listener inside the activity

pager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {

// This method will be invoked when a new page becomes selected.

@Override

public void onPageSelected(int position) {

currentPage = position;

switch (position) {

case 0:

concertController.getNearestConcerts();

break;

case 1:

concertController.getSubscriptionConcerts();

break;

case 2:

concertController.getSavedConcerts();

break;

default:

break;

}

}

// This method will be invoked when the current page is scrolled

@Override

public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {

// Code goes here

}

// Called when the scroll state changes:

// SCROLL\_STATE\_IDLE, SCROLL\_STATE\_DRAGGING, SCROLL\_STATE\_SETTLING

@Override

public void onPageScrollStateChanged(int state) {

// Code goes here

}

});

return view;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setHasOptionsMenu(true);

concertController = ConcertController.getInstance();

}

@Override

public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {

inflater.inflate(R.menu.concert\_search\_menu, menu);

// Spinner

MenuItem spinnerItem = menu.findItem(R.id.concert\_search\_menu\_\_spinner);

Spinner spinner = null;

if (spinnerItem != null) {

spinner = (Spinner) spinnerItem.getActionView();

ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(getContext(),

R.array.concert\_search\_menu\_\_spinner\_array, android.R.layout.simple\_spinner\_item);

adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

spinner.setAdapter(adapter);

}

// Search

MenuItem searchItem = menu.findItem(R.id.concert\_search\_menu\_\_search);

SearchManager searchManager = (SearchManager) getActivity().getSystemService(Context.SEARCH\_SERVICE);

if (searchItem != null) {

searchView = (SearchView) searchItem.getActionView();

}

if (searchView != null) {

searchView.setSearchableInfo(searchManager.getSearchableInfo(getActivity().getComponentName()));

final Spinner finalSpinner = spinner;

SearchView.OnQueryTextListener queryTextListener = new SearchView.OnQueryTextListener() {

@Override

public boolean onQueryTextChange(String newText) {

if(!newText.isEmpty()) {

return true;

}

// Text is empty

KeyboardService.hideKeyboard(getActivity());

switch (currentPage) {

case 0:

concertController.getNearestConcerts();

return true;

case 1:

concertController.getSubscriptionConcerts();

return true;

case 2:

concertController.getSavedConcerts();

return true;

default:

return false;

}

}

@Override

public boolean onQueryTextSubmit(String query) {

KeyboardService.hideKeyboard(getActivity());

boolean searchByArtist = finalSpinner.getSelectedItemId() == 1;

switch (currentPage) {

case 0:

concertController.getNearestConcerts(query, searchByArtist);

return true;

case 1:

concertController.getSubscriptionConcerts(query, searchByArtist);

return true;

case 2:

concertController.getSavedConcerts(query, searchByArtist);

return true;

default:

return false;

}

}

};

searchView.setOnQueryTextListener(queryTextListener);

}

super.onCreateOptionsMenu(menu, inflater);

}

}

* + - 1. Файл ui/fragments /concert/ NearestConcertFragment.java

package com.siroytman.indiewindymobile.ui.fragments.concert;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserConcertLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class NearestConcertFragment extends Fragment {

public static final String TAG = "NearestConcertFragment";

private ConcertController concertController;

private static NearestConcertFragment instance;

private NearestConcertFragment() {

}

public static synchronized NearestConcertFragment getInstance() {

if (instance == null) {

instance = new NearestConcertFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

concertController = ConcertController.getInstance();

concertController.getNearestConcerts();

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_concert\_nearest, container, false);

}

public void concertsFoundViewUpdate(ArrayList<UserConcertLink> links)

{

if (links.size() > 0) {

Log.d(TAG, "Concerts found");

// Load list fragment

UserConcertLinkListFragment fragment = new UserConcertLinkListFragment(links);

FragmentService.replaceFragment(this, R.id.fragment\_concert\_nearest\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /concert/ SavedConcertFragment.java

package com.siroytman.indiewindymobile.ui.fragments.concert;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserConcertLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class SavedConcertFragment extends Fragment {

public static final String TAG = "SavedConcertFragment";

private static SavedConcertFragment instance;

private SavedConcertFragment() {

}

public static synchronized SavedConcertFragment getInstance() {

if (instance == null) {

instance = new SavedConcertFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_concert\_saved, container, false);

}

public void concertsFoundViewUpdate(ArrayList<UserConcertLink> links)

{

if (links.size() > 0) {

Log.d(TAG, "Concerts found");

// Load list fragment

UserConcertLinkListFragment fragment = new UserConcertLinkListFragment(links);

FragmentService.replaceFragment(this, R.id.fragment\_concert\_saved\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /concert/ SubscriptionConcertFragment.java

package com.siroytman.indiewindymobile.ui.fragments.concert;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ConcertController;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserConcertLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class SubscriptionConcertFragment extends Fragment {

public static final String TAG = "SubscrConcertFragment";

private static SubscriptionConcertFragment instance;

private SubscriptionConcertFragment() {

}

public static synchronized SubscriptionConcertFragment getInstance() {

if (instance == null) {

instance = new SubscriptionConcertFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_concert\_subscription, container, false);

}

public void concertsFoundViewUpdate(ArrayList<UserConcertLink> links)

{

if (links.size() > 0) {

Log.d(TAG, "Concerts found");

// Load list fragment

UserConcertLinkListFragment fragment = new UserConcertLinkListFragment(links);

FragmentService.replaceFragment(this, R.id.fragment\_concert\_subscription\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /home/ HomeFragment.java

package com.siroytman.indiewindymobile.ui.fragments.home;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import androidx.annotation.Nullable;

import androidx.appcompat.widget.SearchView;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentStatePagerAdapter;

import androidx.viewpager.widget.ViewPager;

import com.google.android.material.tabs.TabLayout;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.PostController;

import com.siroytman.indiewindymobile.ui.adapter.HomePagerAdapter;

public class HomeFragment extends Fragment {

private static final String TAG = "HomeFragment";

private PostController postController;

private SearchView searchView = null;

private FragmentStatePagerAdapter adapterViewPager;

private ViewPager pager;

private int currentPage = 0;

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_home, container, false);

pager = view.findViewById(R.id.home\_vp\_pager);

pager.setOffscreenPageLimit(HomePagerAdapter.NUM\_ITEMS);

adapterViewPager = new HomePagerAdapter(getChildFragmentManager());

pager.setAdapter(adapterViewPager);

TabLayout tabLayout = view.findViewById(R.id.home\_tab\_layout);

tabLayout.setupWithViewPager(pager);

// Attach the page change listener inside the activity

pager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {

// This method will be invoked when a new page becomes selected.

@Override

public void onPageSelected(int position) {

currentPage = position;

switch (position) {

case 0:

postController.getSubscriptionPosts();

break;

case 1:

postController.getLatestPosts();

break;

default:

break;

}

}

// This method will be invoked when the current page is scrolled

@Override

public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {

// Code goes here

}

// Called when the scroll state changes:

// SCROLL\_STATE\_IDLE, SCROLL\_STATE\_DRAGGING, SCROLL\_STATE\_SETTLING

@Override

public void onPageScrollStateChanged(int state) {

// Code goes here

}

});

return view;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

postController = PostController.getInstance();

}

}

* + - 1. Файл ui/fragments /home/ LatestHomeFragment.java

package com.siroytman.indiewindymobile.ui.fragments.home;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.PostController;

import com.siroytman.indiewindymobile.model.ArtistPostLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.ArtistPostLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class LatestHomeFragment extends Fragment {

public static final String TAG = "LatestHomeFragment";

private static LatestHomeFragment instance;

private LatestHomeFragment() {

}

public static synchronized LatestHomeFragment getInstance() {

if (instance == null) {

instance = new LatestHomeFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_home\_latest, container, false);

}

public void postsFoundViewUpdate(ArrayList<ArtistPostLink> postLinks)

{

if (postLinks.size() > 0) {

Log.d(TAG, "Posts found");

// Load list fragment

ArtistPostLinkListFragment fragment = new ArtistPostLinkListFragment(postLinks);

FragmentService.replaceFragment(this, R.id.fragment\_home\_latest\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /home/ SubscriptionHomeFragment.java

package com.siroytman.indiewindymobile.ui.fragments.home;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.PostController;

import com.siroytman.indiewindymobile.model.ArtistPostLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.ArtistPostLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class SubscriptionHomeFragment extends Fragment {

public static final String TAG = "SubscrHomeFragment";

private static SubscriptionHomeFragment instance;

private PostController postController;

private SubscriptionHomeFragment() {

}

public static synchronized SubscriptionHomeFragment getInstance() {

if (instance == null) {

instance = new SubscriptionHomeFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

postController = PostController.getInstance();

postController.getSubscriptionPosts();

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_home\_subscription, container, false);

}

public void postsFoundViewUpdate(ArrayList<ArtistPostLink> postLinks)

{

if (postLinks.size() > 0) {

Log.d(TAG, "Posts found");

// Load list fragment

ArtistPostLinkListFragment fragment = new ArtistPostLinkListFragment(postLinks);

FragmentService.replaceFragment(this, R.id.fragment\_home\_subscription\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /links/ ArtistPostLinkListFragment.java

package com.siroytman.indiewindymobile.ui.fragments.links;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.model.ArtistPostLink;

import com.siroytman.indiewindymobile.ui.adapter.ArtistPostLinkListAdapter;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.fragment.app.ListFragment;

public class ArtistPostLinkListFragment extends ListFragment {

private List<ArtistPostLink> links;

private ArtistPostLinkListAdapter adapter;

public ArtistPostLinkListFragment() {

}

public ArtistPostLinkListFragment(List<ArtistPostLink> links) {

this.links = links;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

adapter = new ArtistPostLinkListAdapter(getActivity(), R.layout.post\_list\_item, links, getChildFragmentManager());

setListAdapter(adapter);

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_post\_list, container, false);

}

}

* + - 1. Файл ui/fragments /links/ UserAlbumLinkListFragment.java

package com.siroytman.indiewindymobile.ui.fragments.links;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.TextView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.ui.activity.MoreActivity;

import com.siroytman.indiewindymobile.ui.adapter.UserAlbumLinkListAdapter;

import com.siroytman.indiewindymobile.ui.fragments.search.SearchFragment;

import java.util.ArrayList;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.fragment.app.ListFragment;

public class UserAlbumLinkListFragment extends ListFragment {

public static final String TAG = "UserAlbumLinkListFrag";

private List<UserAlbumLink> albumLinks;

private UserAlbumLinkListAdapter adapter;

private Artist artist;

private Boolean rawList;

private Boolean cutList;

public UserAlbumLinkListFragment() {

}

public UserAlbumLinkListFragment(List<UserAlbumLink> albumLinks) {

this(albumLinks, false, false);

}

public UserAlbumLinkListFragment(List<UserAlbumLink> albumLinks, Boolean rawList, Boolean cutList) {

this.albumLinks = albumLinks;

this.rawList = rawList;

this.cutList = cutList;

}

public UserAlbumLinkListFragment(List<UserAlbumLink> albumLinks, Artist artist) {

this.albumLinks = albumLinks;

this.artist = artist;

this.rawList = true;

this.cutList = false;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

List<UserAlbumLink> subLinks;

if(cutList) {

// Cut to subLinks not to show all found songs

subLinks = albumLinks.subList(0, Math.min(SearchFragment.MAX\_ELEMENTS, albumLinks.size()));

} else {

subLinks = albumLinks;

}

if (artist != null) {

adapter = new UserAlbumLinkListAdapter(getActivity(), R.layout.album\_list\_item, subLinks, artist);

} else {

adapter = new UserAlbumLinkListAdapter(getActivity(), R.layout.album\_list\_item, subLinks);

}

setListAdapter(adapter);

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_album\_list, container, false);

Button albumMoreButton = view.findViewById(R.id.fragment\_song\_album\_\_more);

if (rawList) {

albumMoreButton.setVisibility(View.GONE);

TextView artistTitleView = view.findViewById(R.id.fragment\_album\_list\_\_title);

artistTitleView.setVisibility(View.GONE);

}

else {

albumMoreButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Log.d(TAG, "more button clicked");

Log.d(TAG, "albumLinks size = " + albumLinks.size());

Bundle bundle = new Bundle();

ArrayList<UserAlbumLink> arrayList = new ArrayList<>(albumLinks);

bundle.putParcelableArrayList("albumLinks", arrayList);

Intent intent = new Intent(getContext(), MoreActivity.class);

intent.putExtras(bundle);

startActivity(intent);

}

});

}

return view;

}

}

* + - 1. Файл ui/fragments /links/ UserArtistLinkListFragment.java

package com.siroytman.indiewindymobile.ui.fragments.links;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.TextView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.model.Artist;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.ui.activity.MoreActivity;

import com.siroytman.indiewindymobile.ui.adapter.UserArtistLinkListAdapter;

import java.util.ArrayList;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.fragment.app.ListFragment;

public class UserArtistLinkListFragment extends ListFragment {

public static final String TAG = "UserArtistLinkListFrag";

private List<UserArtistLink> artistLinks;

private UserArtistLinkListAdapter adapter;

private Boolean rawList;

public UserArtistLinkListFragment() {

}

public UserArtistLinkListFragment(List<UserArtistLink> artistLinks) {

this(artistLinks, false);

}

public UserArtistLinkListFragment(List<UserArtistLink> artistLinks, Boolean rawList) {

this.artistLinks = artistLinks;

this.rawList = rawList;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

adapter = new UserArtistLinkListAdapter(getActivity(), R.layout.artist\_list\_item, artistLinks);

setListAdapter(adapter);

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_artist\_list, container, false);

Button artistMoreButton = view.findViewById(R.id.fragment\_artist\_list\_\_more);

if (rawList) {

artistMoreButton.setVisibility(View.GONE);

TextView artistTitleView = view.findViewById(R.id.fragment\_artist\_list\_\_title);

artistTitleView.setVisibility(View.GONE);

}

else {

artistMoreButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Log.d(TAG, "more button clicked");

Log.d(TAG, "artistLinks size = " + artistLinks.size());

Bundle bundle = new Bundle();

ArrayList<UserArtistLink> arrayList = new ArrayList<>(artistLinks);

bundle.putParcelableArrayList("artistLinks", arrayList);

Intent intent = new Intent(getContext(), MoreActivity.class);

intent.putExtras(bundle);

startActivity(intent);

}

});

}

return view;

}

}

* + - 1. Файл ui/fragments /links/ UserConcertLinkListFragment.java

package com.siroytman.indiewindymobile.ui.fragments.links;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.model.UserConcertLink;

import com.siroytman.indiewindymobile.ui.adapter.UserConcertLinkListAdapter;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.fragment.app.ListFragment;

public class UserConcertLinkListFragment extends ListFragment {

private List<UserConcertLink> concertLinks;

private UserConcertLinkListAdapter adapter;

public UserConcertLinkListFragment() {

}

public UserConcertLinkListFragment(List<UserConcertLink> concertLinks) {

this.concertLinks = concertLinks;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

adapter = new UserConcertLinkListAdapter(getActivity(), R.layout.concert\_list\_item, concertLinks);

setListAdapter(adapter);

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_concert\_list, container, false);

}

}

* + - 1. Файл ui/fragments /links/ UserSongLinkListFragment.java

package com.siroytman.indiewindymobile.ui.fragments.links;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.TextView;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.ui.activity.MoreActivity;

import com.siroytman.indiewindymobile.ui.adapter.UserSongLinkListAdapter;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.ui.fragments.search.SearchFragment;

import java.util.ArrayList;

import java.util.List;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.fragment.app.ListFragment;

public class UserSongLinkListFragment extends ListFragment {

public static final String TAG = "UserSongLinkListFrag";

ArrayList<UserSongLink> songLinks;

private UserSongLinkListAdapter adapter;

private Boolean rawList;

private boolean cutList;

public UserSongLinkListFragment() {

}

public UserSongLinkListFragment(ArrayList<UserSongLink> songLinks) {

this(songLinks, false, false);

}

public UserSongLinkListFragment(ArrayList<UserSongLink> songLinks, Boolean rawList, Boolean cutList) {

this.songLinks = songLinks;

this.rawList = rawList;

this.cutList = cutList;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

ArrayList<UserSongLink> subLinks;

if(cutList) {

// Cut to subLinks not to show all found songs

subLinks = (ArrayList<UserSongLink>) songLinks.subList(0, Math.min(SearchFragment.MAX\_ELEMENTS, songLinks.size()));

} else {

subLinks = songLinks;

}

adapter = new UserSongLinkListAdapter(getActivity(), R.layout.song\_list\_item, subLinks);

setListAdapter(adapter);

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_song\_list, container, false);

Button songMoreButton = view.findViewById(R.id.fragment\_song\_list\_\_more);

if (rawList) {

songMoreButton.setVisibility(View.GONE);

TextView songTitleView = view.findViewById(R.id.fragment\_song\_list\_\_title);

songTitleView.setVisibility(View.GONE);

}

else {

songMoreButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Log.d(TAG, "more button clicked");

Log.d(TAG, "songLinks size = " + songLinks.size());

Bundle bundle = new Bundle();

ArrayList<UserSongLink> arrayList = new ArrayList<>(songLinks);

bundle.putParcelableArrayList("songLinks", arrayList);

Intent intent = new Intent(getContext(), MoreActivity.class);

intent.putExtras(bundle);

startActivity(intent);

}

});

}

return view;

}

}

* + - 1. Файл ui/fragments /personal/ PersonalFragment.java

package com.siroytman.indiewindymobile.ui.fragments.personal;

import android.app.SearchManager;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import com.google.android.material.tabs.TabLayout;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AlbumController;

import com.siroytman.indiewindymobile.controller.ArtistController;

import com.siroytman.indiewindymobile.controller.SongController;

import com.siroytman.indiewindymobile.services.KeyboardService;

import com.siroytman.indiewindymobile.ui.activity.SettingsActivity;

import com.siroytman.indiewindymobile.ui.adapter.PersonalPagerAdapter;

import androidx.annotation.Nullable;

import androidx.appcompat.widget.SearchView;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentStatePagerAdapter;

import androidx.viewpager.widget.ViewPager;

public class PersonalFragment extends Fragment {

private static final String TAG = "PersonalFragment";

private SearchView searchView = null;

private FragmentStatePagerAdapter adapterViewPager;

private ViewPager pager;

private SongController songController;

private ArtistController artistController;

private AlbumController albumController;

private int currentPage = 0;

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.fragment\_personal, container, false);

pager = view.findViewById(R.id.personal\_vp\_pager);

pager.setOffscreenPageLimit(PersonalPagerAdapter.NUM\_ITEMS);

adapterViewPager = new PersonalPagerAdapter(getChildFragmentManager());

pager.setAdapter(adapterViewPager);

TabLayout tabLayout = view.findViewById(R.id.personal\_tab\_layout);

tabLayout.setupWithViewPager(pager);

// Attach the page change listener inside the activity

pager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {

// This method will be invoked when a new page becomes selected.

@Override

public void onPageSelected(int position) {

currentPage = position;

switch (position) {

case 0:

songController.searchSongsLinked(SongsPersonalFragment.getInstance());

break;

case 1:

artistController.searchArtistsLinked(ArtistsPersonalFragment.getInstance());

break;

case 2:

albumController.searchAlbumsLinked(AlbumsPersonalFragment.getInstance());

break;

default:

break;

}

}

// This method will be invoked when the current page is scrolled

@Override

public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {

// Code goes here

}

// Called when the scroll state changes:

// SCROLL\_STATE\_IDLE, SCROLL\_STATE\_DRAGGING, SCROLL\_STATE\_SETTLING

@Override

public void onPageScrollStateChanged(int state) {

// Code goes here

}

});

return view;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setHasOptionsMenu(true);

songController = SongController.getInstance();

artistController = ArtistController.getInstance();

albumController = AlbumController.getInstance();

}

@Override

public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {

inflater.inflate(R.menu.personal\_menu, menu);

MenuItem searchItem = menu.findItem(R.id.personal\_search);

SearchManager searchManager = (SearchManager) getActivity().getSystemService(Context.SEARCH\_SERVICE);

if (searchItem != null) {

searchView = (SearchView) searchItem.getActionView();

}

if (searchView != null) {

searchView.setSearchableInfo(searchManager.getSearchableInfo(getActivity().getComponentName()));

SearchView.OnQueryTextListener queryTextListener = new SearchView.OnQueryTextListener() {

@Override

public boolean onQueryTextChange(String newText) {

return true;

}

@Override

public boolean onQueryTextSubmit(String query) {

KeyboardService.hideKeyboard(getActivity());

switch (currentPage) {

case 0:

songController.searchSongsLinked(SongsPersonalFragment.getInstance(), query);

return true;

case 1:

artistController.searchArtistsLinked(ArtistsPersonalFragment.getInstance(), query);

return true;

case 2:

albumController.searchAlbumsLinked(AlbumsPersonalFragment.getInstance(), query);

return true;

default:

return false;

}

}

};

searchView.setOnQueryTextListener(queryTextListener);

}

MenuItem settingsItem = menu.findItem(R.id.personal\_settings);

settingsItem.setOnMenuItemClickListener(new MenuItem.OnMenuItemClickListener() {

@Override

public boolean onMenuItemClick(MenuItem item) {

startActivity(new Intent(getContext(), SettingsActivity.class));

return true;

}

});

super.onCreateOptionsMenu(menu, inflater);

}

}

* + - 1. Файл ui/fragments /personal/ AlbumsPersonalFragment.java

package com.siroytman.indiewindymobile.ui.fragments.personal;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.AlbumController;

import com.siroytman.indiewindymobile.interfaces.ISearchableAlbum;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserAlbumLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class AlbumsPersonalFragment extends Fragment implements ISearchableAlbum {

public static final String TAG = "AlbumsPersonalFragment";

private static AlbumsPersonalFragment instance;

private AlbumController albumController;

private AlbumsPersonalFragment() {

albumController = AlbumController.getInstance();

}

public static synchronized AlbumsPersonalFragment getInstance() {

if (instance == null) {

instance = new AlbumsPersonalFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_personal\_albums, container, false);

}

@Override

public void albumsFoundViewUpdate(ArrayList<UserAlbumLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Albums found");

// Load list fragment

UserAlbumLinkListFragment fragment = new UserAlbumLinkListFragment(links, true, false);

FragmentService.replaceFragment(this, R.id.fragment\_personal\_albums\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /personal/ ArtistsPersonalFragment.java

package com.siroytman.indiewindymobile.ui.fragments.personal;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.ArtistController;

import com.siroytman.indiewindymobile.interfaces.ISearchableArtist;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserArtistLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class ArtistsPersonalFragment extends Fragment implements ISearchableArtist {

public static final String TAG = "ArtistsPersonalFragment";

private static ArtistsPersonalFragment instance;

private ArtistController artistController;

private ArtistsPersonalFragment() {

artistController = ArtistController.getInstance();

}

public static synchronized ArtistsPersonalFragment getInstance() {

if (instance == null) {

instance = new ArtistsPersonalFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_personal\_artists, container, false);

}

@Override

public void artistsFoundViewUpdate(ArrayList<UserArtistLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Artists found");

// Load list fragment

UserArtistLinkListFragment fragment = new UserArtistLinkListFragment(links, true);

FragmentService.replaceFragment(this, R.id.fragment\_personal\_artists\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /personal/ SongsPersonalFragment.java

package com.siroytman.indiewindymobile.ui.fragments.personal;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.SongController;

import com.siroytman.indiewindymobile.interfaces.ISearchableSong;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserSongLinkListFragment;

import java.util.ArrayList;

import androidx.fragment.app.Fragment;

public class SongsPersonalFragment extends Fragment implements ISearchableSong {

public static final String TAG = "SongsPersonalFragment";

private static SongsPersonalFragment instance;

private SongController songController;

private SongsPersonalFragment() {

}

public static synchronized SongsPersonalFragment getInstance() {

if (instance == null) {

instance = new SongsPersonalFragment();

}

return instance;

}

// Store instance variables based on arguments passed

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

songController = SongController.getInstance();

songController.searchSongsLinked(this);

}

// Inflate the view for the fragment based on layout XML

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_personal\_songs, container, false);

}

@Override

public void songsFoundViewUpdate(ArrayList<UserSongLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Songs found");

// Load list fragment

UserSongLinkListFragment fragment = new UserSongLinkListFragment(links, true, false);

FragmentService.replaceFragment(this, R.id.fragment\_personal\_songs\_\_container, fragment);

} else {

FragmentService.clearFragment(this);

}

}

}

* + - 1. Файл ui/fragments /search SearchFragment.java/

package com.siroytman.indiewindymobile.ui.fragments.search;

import android.app.SearchManager;

import android.content.Context;

import android.os.Bundle;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import com.siroytman.indiewindymobile.R;

import com.siroytman.indiewindymobile.controller.SearchController;

import com.siroytman.indiewindymobile.interfaces.ISearchableAlbum;

import com.siroytman.indiewindymobile.interfaces.ISearchableArtist;

import com.siroytman.indiewindymobile.interfaces.ISearchableSong;

import com.siroytman.indiewindymobile.model.UserAlbumLink;

import com.siroytman.indiewindymobile.model.UserArtistLink;

import com.siroytman.indiewindymobile.model.UserSongLink;

import com.siroytman.indiewindymobile.services.FragmentService;

import com.siroytman.indiewindymobile.services.KeyboardService;

import com.siroytman.indiewindymobile.ui.fragments.links.UserAlbumLinkListFragment;

import com.siroytman.indiewindymobile.ui.fragments.links.UserArtistLinkListFragment;

import com.siroytman.indiewindymobile.ui.fragments.links.UserSongLinkListFragment;

import java.util.ArrayList;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

import androidx.annotation.Nullable;

import androidx.appcompat.widget.SearchView;

import androidx.fragment.app.Fragment;

public class SearchFragment extends Fragment

implements ISearchableSong, ISearchableArtist, ISearchableAlbum {

private static final String TAG = "SearchFragment";

private SearchController searchController;

private SearchView searchView = null;

// Containers busy state

private LinkedHashMap<Integer, Boolean> containerBusy;

private String searchQuery;

public static final int MAX\_ELEMENTS = 2;

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_search, container, false);

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setHasOptionsMenu(true);

searchController = SearchController.getInstance();

}

@Override

public void songsFoundViewUpdate(ArrayList<UserSongLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Songs found");

int containerId = R.id.fragment\_search\_\_song\_container;

containerBusy.put(containerId, true);

// Load list fragment

UserSongLinkListFragment fragment = new UserSongLinkListFragment(links, false, true);

FragmentService.replaceFragment(this, containerId, fragment);

}

// When finished start next request

searchController.searchArtists(SearchFragment.this, searchQuery);

}

@Override

public void artistsFoundViewUpdate(ArrayList<UserArtistLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Artists found");

int containerId = pickFirstEmptyContainer();

containerBusy.put(containerId, true);

List<UserArtistLink> subLinks = links.subList(0, Math.min(MAX\_ELEMENTS, links.size()));

// Load list fragment

UserArtistLinkListFragment fragment = new UserArtistLinkListFragment(subLinks);

FragmentService.replaceFragment(this, containerId, fragment);

}

// When finished start next request

searchController.searchAlbums(SearchFragment.this, searchQuery);

}

@Override

public void albumsFoundViewUpdate(ArrayList<UserAlbumLink> links) {

if (links.size() > 0) {

Log.d(TAG, "Albums found");

int containerId = pickFirstEmptyContainer();

containerBusy.put(containerId, true);

List<UserAlbumLink> subLinks = links.subList(0, Math.min(MAX\_ELEMENTS, links.size()));

// Load list fragment

UserAlbumLinkListFragment fragment = new UserAlbumLinkListFragment(subLinks);

FragmentService.replaceFragment(this, containerId, fragment);

}

}

// Returns first empty containerId

private Integer pickFirstEmptyContainer() {

for (Map.Entry<Integer, Boolean> pair : containerBusy.entrySet()) {

if (!pair.getValue()) {

return pair.getKey();

}

}

return null;

}

@Override

public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {

inflater.inflate(R.menu.search\_menu, menu);

MenuItem searchItem = menu.findItem(R.id.search\_menu\_\_search);

SearchManager searchManager = (SearchManager) getActivity().getSystemService(Context.SEARCH\_SERVICE);

if (searchItem != null) {

searchView = (SearchView) searchItem.getActionView();

}

if (searchView != null) {

searchView.setSearchableInfo(searchManager.getSearchableInfo(getActivity().getComponentName()));

SearchView.OnQueryTextListener queryTextListener = new SearchView.OnQueryTextListener() {

@Override

public boolean onQueryTextChange(String newText) {

return true;

}

@Override

public boolean onQueryTextSubmit(String query) {

searchQuery = query;

KeyboardService.hideKeyboard(getActivity());

clearContainers();

searchController.searchSongs(SearchFragment.this, searchQuery);

return true;

}

};

searchView.setOnQueryTextListener(queryTextListener);

}

super.onCreateOptionsMenu(menu, inflater);

}

private void clearContainers() {

containerBusy = new LinkedHashMap<Integer, Boolean>() {{

put(R.id.fragment\_search\_\_song\_container, false);

put(R.id.fragment\_search\_\_artist\_container, false);

put(R.id.fragment\_search\_\_album\_container, false);

}};

FragmentService.clearFragment(this);

}

}

* + 1. Res
       1. Файл layout/activity\_album.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.AlbumActivity">

<ImageView

android:id="@+id/album\_activity\_\_album\_photo"

android:src="@drawable/ic\_album"

android:layout\_width="@dimen/activity\_image\_size"

android:layout\_height="@dimen/activity\_image\_size"

android:layout\_marginTop="10dp"

android:layout\_marginStart="10dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentStart="true"

android:contentDescription="Album photo" />

<TextView

android:id="@+id/album\_activity\_\_album\_name"

android:text="Album name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_toEndOf="@id/album\_activity\_\_album\_photo"

android:layout\_marginStart="10dp"

android:layout\_marginTop="20dp"

android:textSize="22sp"

style="@style/BigText" />

<TextView

android:id="@+id/album\_activity\_\_album\_artist\_name"

android:text="Artist name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/album\_activity\_\_album\_name"

android:layout\_toEndOf="@id/album\_activity\_\_album\_photo"

android:layout\_marginStart="10dp"

android:layout\_marginTop="6dp"

style="@style/SmallText"/>

<ImageView

android:id="@+id/album\_activity\_\_album\_options\_button"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:layout\_above="@+id/album\_activity\_\_album\_song\_list\_container"

android:layout\_alignParentEnd="true"

android:layout\_marginTop="115dp"

android:layout\_marginEnd="20dp"

android:layout\_marginBottom="15dp"

android:contentDescription="Options button"

android:src="@drawable/ic\_options" />

<ImageView

android:id="@+id/album\_activity\_\_album\_add\_button"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:layout\_above="@+id/album\_activity\_\_album\_song\_list\_container"

android:layout\_marginTop="115dp"

android:layout\_marginEnd="10dp"

android:layout\_marginBottom="15dp"

android:layout\_toLeftOf="@id/album\_activity\_\_album\_options\_button"

android:contentDescription="Add button"

android:src="@drawable/ic\_add" />

<FrameLayout

android:id="@+id/album\_activity\_\_album\_song\_list\_container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_below="@id/album\_activity\_\_album\_photo"

android:layout\_marginTop="10dp"/>

</RelativeLayout>

* + - 1. Файл layout/activity\_artist.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.ArtistActivity">

<ImageView

android:id="@+id/artist\_activity\_\_artist\_photo"

android:src="@drawable/ic\_artist"

android:layout\_width="@dimen/activity\_image\_size"

android:layout\_height="@dimen/activity\_image\_size"

android:layout\_marginTop="10dp"

android:layout\_marginStart="10dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentStart="true"

android:contentDescription="Artist photo" />

<TextView

android:id="@+id/artist\_activity\_\_artist\_name"

android:text="Artist name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_toEndOf="@id/artist\_activity\_\_artist\_photo"

android:layout\_marginStart="10dp"

android:layout\_marginTop="20dp"

android:textSize="22sp"

style="@style/BigText" />

<TextView

android:id="@+id/artist\_activity\_\_artist\_description"

android:text="Artist description"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/artist\_activity\_\_artist\_name"

android:layout\_toEndOf="@id/artist\_activity\_\_artist\_photo"

android:layout\_marginStart="10dp"

android:layout\_marginTop="3dp"

style="@style/SmallText"/>

<ImageView

android:id="@+id/artist\_activity\_\_artist\_donate\_button"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:layout\_above="@+id/artist\_activity\_\_albums\_container"

android:layout\_marginTop="115dp"

android:layout\_marginEnd="10dp"

android:layout\_marginBottom="15dp"

android:layout\_toStartOf="@id/artist\_activity\_\_artist\_add\_button"

android:contentDescription="Donate button"

android:src="@drawable/ic\_donate" />

<ImageView

android:id="@+id/artist\_activity\_\_artist\_add\_button"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:layout\_above="@+id/artist\_activity\_\_albums\_container"

android:layout\_marginTop="115dp"

android:layout\_marginEnd="10dp"

android:layout\_marginBottom="15dp"

android:layout\_alignParentEnd="true"

android:contentDescription="Add button"

android:src="@drawable/ic\_add" />

<FrameLayout

android:id="@+id/artist\_activity\_\_albums\_container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_below="@id/artist\_activity\_\_artist\_photo"

android:layout\_marginTop="10dp" />

</RelativeLayout>

* + - 1. Файл layout/activity\_concert.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.ConcertActivity">

<ImageView

android:id="@+id/concert\_activity\_\_concert\_photo"

android:src="@drawable/ic\_concert"

android:layout\_width="match\_parent"

android:layout\_height="370dp"

android:layout\_marginTop="40dp"

android:layout\_marginStart="10dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentStart="true"

android:contentDescription="Concert photo" />

<TextView

android:id="@+id/concert\_activity\_\_concert\_name"

android:text="Concert name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="10dp"

android:layout\_marginTop="5dp"

android:layout\_centerHorizontal="true"

android:textSize="22sp"

style="@style/BigText" />

<TextView

android:id="@+id/concert\_activity\_\_concert\_date"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_alignParentTop="true"

android:layout\_marginStart="10dp"

android:layout\_marginTop="10dp"

android:layout\_marginEnd="10dp"

android:text="Concert date"

android:textSize="17sp"

android:textStyle="italic"

/>

<TextView

android:id="@+id/concert\_activity\_\_concert\_address"

android:layout\_width="250dp"

android:layout\_height="40dp"

android:layout\_below="@id/concert\_activity\_\_concert\_photo"

android:layout\_marginTop="10dp"

android:layout\_marginStart="10dp"

android:text="Address"

android:scrollbars="vertical"

android:textSize="16sp"

style="@style/SmallText"/>

<TextView

android:id="@+id/concert\_activity\_\_concert\_description"

android:text="Concert description"

android:layout\_width="match\_parent"

android:layout\_height="100dp"

android:layout\_below="@id/concert\_activity\_\_ticket\_button"

android:textSize="18sp"

android:layout\_marginEnd="10dp"

android:layout\_marginStart="10dp"

android:layout\_marginTop="2dp"

android:scrollbars="vertical"/>

<ImageView

android:id="@+id/concert\_activity\_\_concert\_add\_button"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:layout\_below="@id/concert\_activity\_\_concert\_photo"

android:layout\_marginEnd="10dp"

android:layout\_marginBottom="15dp"

android:layout\_marginTop="11dp"

android:layout\_alignParentEnd="true"

android:contentDescription="Add button"

android:src="@drawable/ic\_add" />

<Button

android:id="@+id/concert\_activity\_\_ticket\_button"

android:layout\_width="wrap\_content"

android:layout\_height="50dp"

android:layout\_marginStart="10dp"

android:layout\_marginEnd="10dp"

android:layout\_marginTop="5dp"

android:layout\_marginBottom="5dp"

android:layout\_toLeftOf="@id/concert\_activity\_\_concert\_add\_button"

android:text="@string/activity\_concert\_\_ticket"

style="?attr/materialButtonStyle"

android:layout\_below="@id/concert\_activity\_\_concert\_photo"

android:textSize="@dimen/button\_text\_size"

/>

<FrameLayout

android:id="@+id/concert\_activity\_\_artists\_container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentBottom="true"

android:layout\_marginTop="10dp"

android:layout\_below="@id/concert\_activity\_\_concert\_description" />

</RelativeLayout>

* + - 1. Файл layout/activity\_donation.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.DonationActivity">

<TextView

android:id="@+id/donation\_activity\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="50dp"

android:layout\_centerHorizontal="true"

android:text="Artist name"

android:textSize="22sp"

android:textAppearance="@style/BigText"/>

<ImageView

android:id="@+id/donation\_activity\_artist\_image"

android:layout\_width="@dimen/activity\_image\_size"

android:layout\_height="@dimen/activity\_image\_size"

android:layout\_marginTop="16dp"

android:layout\_centerHorizontal="true"

android:layout\_below="@id/donation\_activity\_artist\_name"

app:srcCompat="@drawable/ic\_artist" />

<EditText

android:id="@+id/donation\_activity\_amount"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="10"

android:layout\_marginTop="16dp"

style="@style/SmallText"

android:hint="@string/donation\_\_amount\_hint"

android:inputType="numberDecimal"

android:layout\_centerHorizontal="true"

android:layout\_below="@id/donation\_activity\_artist\_image" />

<Button

android:id="@+id/donation\_activity\_donation\_button"

style="?attr/materialButtonStyle"

android:layout\_width="wrap\_content"

android:layout\_height="@dimen/button\_height"

android:layout\_marginTop="16dp"

android:text="@string/donation\_\_donate\_button"

android:textSize="@dimen/button\_text\_size"

android:layout\_centerHorizontal="true"

android:layout\_below="@id/donation\_activity\_amount"/>

</RelativeLayout>

* + - 1. Файл layout/activity\_login.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.LoginActivity">

<ImageView

android:id="@+id/login\_activity\_\_logo"

android:layout\_width="192dp"

android:layout\_height="192dp"

android:src="@mipmap/ic\_logo"

app:layout\_constraintBottom\_toTopOf="@+id/login\_activity\_\_login"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent" />

<EditText

android:id="@+id/login\_activity\_\_login"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginBottom="16dp"

android:ems="10"

android:hint="@string/login\_\_login\_hint"

android:inputType="textPersonName"

app:layout\_constraintBottom\_toTopOf="@+id/login\_activity\_\_password"

app:layout\_constraintEnd\_toEndOf="parent"

android:text="dimas"

app:layout\_constraintStart\_toStartOf="parent"

style="@style/BigText"/>

<EditText

android:id="@+id/login\_activity\_\_password"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="10"

android:hint="@string/login\_\_password\_hint"

android:inputType="textPassword"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

android:text="myTestPassword"

app:layout\_constraintTop\_toTopOf="parent"

style="@style/BigText"/>

<ProgressBar

android:id="@+id/login\_activity\_\_loading\_progress\_bar"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

android:indeterminateTint="@color/primary\_dark"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

tools:visibility="gone" />

<Button

android:id="@+id/login\_activity\_\_login\_button"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:paddingHorizontal="40dp"

android:layout\_marginTop="16dp"

android:onClick="loginOnClick"

android:text="@string/login\_\_login\_button"

android:textSize="@dimen/button\_text\_size"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/login\_activity\_\_password" />

<Button

android:id="@+id/login\_activity\_\_register\_button"

style="?attr/materialButtonOutlinedStyle"

android:textSize="@dimen/button\_text\_size"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:paddingVertical="10dp"

android:onClick="registerOnClick"

android:text="@string/login\_\_register\_button"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/login\_activity\_\_login\_button" />

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/activity\_more.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.MoreActivity">

<FrameLayout

android:id="@+id/more\_activity\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

/>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/activity\_navigation.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:id="@+id/container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<fragment

android:id="@+id/nav\_host\_fragment"

android:name="androidx.navigation.fragment.NavHostFragment"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:defaultNavHost="true"

app:layout\_constraintBottom\_toTopOf="@id/nav\_view"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:navGraph="@navigation/mobile\_navigation" />

<com.google.android.material.bottomnavigation.BottomNavigationView

android:id="@+id/nav\_view"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginStart="0dp"

android:layout\_marginEnd="0dp"

android:background="?android:attr/windowBackground"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

app:itemIconSize="33dp"

app:menu="@menu/bottom\_nav\_menu" />

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/activity\_player.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.activity.PlayerActivity">

<com.google.android.exoplayer2.ui.PlayerView

android:id="@+id/player\_view"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:use\_artwork="false"

app:controller\_layout\_id="@layout/custom\_exo\_controller\_view"/>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/album\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.card.MaterialCardView

android:layout\_width="match\_parent"

android:layout\_height="@dimen/list\_item\_height"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<ImageView

android:id="@+id/album\_list\_item\_\_album\_photo"

android:layout\_width="@dimen/list\_item\_image"

android:layout\_height="@dimen/list\_item\_image"

android:layout\_alignParentStart="true"

android:layout\_centerInParent="true"

android:src="@drawable/ic\_album" />

<TextView

android:id="@+id/album\_list\_item\_\_album\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_toEndOf="@id/album\_list\_item\_\_album\_photo"

android:layout\_alignParentTop="true"

android:paddingHorizontal="12dp"

android:paddingTop="8dp"

android:text="Album Name"

android:textAppearance="@style/BigText" />

<TextView

android:id="@+id/album\_list\_item\_\_album\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/album\_list\_item\_\_album\_name"

android:layout\_toEndOf="@id/album\_list\_item\_\_album\_photo"

android:paddingHorizontal="12dp"

android:text="Artist Name"

style="@style/SmallText" />

<ImageView

android:id="@+id/album\_list\_item\_\_album\_add\_button"

android:layout\_toStartOf="@id/album\_list\_item\_\_album\_options\_button"

android:layout\_centerInParent="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_add"

android:contentDescription="Add button" />

<ImageView

android:id="@+id/album\_list\_item\_\_album\_options\_button"

android:layout\_alignParentEnd="true"

android:layout\_centerInParent="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_options"

android:contentDescription="Options button" />

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/album\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.card.MaterialCardView

android:layout\_width="match\_parent"

android:layout\_height="@dimen/list\_item\_height"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<ImageView

android:id="@+id/album\_list\_item\_\_album\_photo"

android:layout\_width="@dimen/list\_item\_image"

android:layout\_height="@dimen/list\_item\_image"

android:layout\_alignParentStart="true"

android:layout\_centerInParent="true"

android:src="@drawable/ic\_album" />

<TextView

android:id="@+id/album\_list\_item\_\_album\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_toEndOf="@id/album\_list\_item\_\_album\_photo"

android:layout\_alignParentTop="true"

android:paddingHorizontal="12dp"

android:paddingTop="8dp"

android:text="Album Name"

android:textAppearance="@style/BigText" />

<TextView

android:id="@+id/album\_list\_item\_\_album\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/album\_list\_item\_\_album\_name"

android:layout\_toEndOf="@id/album\_list\_item\_\_album\_photo"

android:paddingHorizontal="12dp"

android:text="Artist Name"

style="@style/SmallText" />

<ImageView

android:id="@+id/album\_list\_item\_\_album\_add\_button"

android:layout\_toStartOf="@id/album\_list\_item\_\_album\_options\_button"

android:layout\_centerInParent="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_add"

android:contentDescription="Add button" />

<ImageView

android:id="@+id/album\_list\_item\_\_album\_options\_button"

android:layout\_alignParentEnd="true"

android:layout\_centerInParent="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_options"

android:contentDescription="Options button" />

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/ artist\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.card.MaterialCardView

android:layout\_width="match\_parent"

android:layout\_height="@dimen/list\_item\_height"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<ImageView

android:id="@+id/artist\_list\_item\_\_artist\_photo"

android:layout\_width="@dimen/list\_item\_image"

android:layout\_height="@dimen/list\_item\_image"

android:layout\_alignParentStart="true"

android:layout\_centerInParent="true"

android:src="@drawable/ic\_artist" />

<TextView

android:id="@+id/artist\_list\_item\_\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_toEndOf="@id/artist\_list\_item\_\_artist\_photo"

android:layout\_alignParentTop="true"

android:layout\_marginStart="14dp"

android:layout\_centerInParent="true"

android:text="Artist Name"

android:textAppearance="@style/BigText" />

<ImageView

android:id="@+id/artist\_list\_item\_\_add\_button"

android:layout\_alignParentEnd="true"

android:layout\_centerInParent="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_add"

android:contentDescription="Add button" />

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/ concert\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.card.MaterialCardView

android:layout\_width="match\_parent"

android:layout\_height="200dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<ImageView

android:id="@+id/concert\_list\_item\_\_photo"

android:layout\_width="180dp"

android:layout\_height="180dp"

android:layout\_alignParentStart="true"

android:layout\_centerInParent="true"

android:paddingHorizontal="12dp"

android:paddingVertical="5dp"

android:src="@drawable/ic\_concert"

android:contentDescription="Concert photo" />

<TextView

android:id="@+id/concert\_list\_item\_\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:paddingHorizontal="12dp"

android:paddingTop="12dp"

android:layout\_toEndOf="@id/concert\_list\_item\_\_photo"

android:text="Concert Name"

android:textAppearance="@style/BigText" />

<TextView

android:id="@+id/concert\_list\_item\_\_date"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="10dp"

android:layout\_marginTop="4dp"

android:text="Concert date"

android:textSize="17sp"

android:textStyle="italic"

android:layout\_toEndOf="@id/concert\_list\_item\_\_photo"

android:layout\_below="@id/concert\_list\_item\_\_name" />

<TextView

android:id="@+id/concert\_list\_item\_\_description"

android:text="Concert description"

android:layout\_width="match\_parent"

android:layout\_height="110dp"

android:layout\_toEndOf="@id/concert\_list\_item\_\_photo"

android:layout\_below="@id/concert\_list\_item\_\_date"

android:textSize="18sp"

android:layout\_marginEnd="10dp"

android:layout\_marginStart="10dp"

android:layout\_marginTop="4dp"

android:lines="5"

android:scrollHorizontally="true"

android:ellipsize="end"/>

<ImageView

android:id="@+id/concert\_list\_item\_\_\_add\_button"

android:layout\_alignParentEnd="true"

android:layout\_alignParentTop="true"

android:layout\_width="@dimen/options\_button\_big"

android:layout\_height="@dimen/options\_button\_big"

android:src="@drawable/ic\_add"

android:contentDescription="Add button" />

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/ custom\_exo\_controller\_view.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="bottom"

android:layoutDirection="ltr"

android:orientation="vertical"

android:paddingStart="20dp"

android:paddingEnd="20dp">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:orientation="horizontal"

android:paddingTop="4dp">

<ImageButton

android:id="@id/exo\_prev"

style="@style/ExoMediaButton.Previous"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_rew"

style="@style/ExoMediaButton.Rewind"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_shuffle"

style="@style/ExoMediaButton"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_repeat\_toggle"

style="@style/ExoMediaButton"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_play"

style="@style/ExoMediaButton.Play"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_pause"

style="@style/ExoMediaButton.Pause"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_ffwd"

style="@style/ExoMediaButton.FastForward"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_next"

style="@style/ExoMediaButton.Next"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

<ImageButton

android:id="@id/exo\_vr"

style="@style/ExoMediaButton.VR"

android:tint="@color/primary\_dark"

android:tintMode="src\_in" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="4dp"

android:gravity="center\_vertical"

android:orientation="horizontal">

<TextView

android:id="@id/exo\_position"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:includeFontPadding="false"

android:paddingLeft="4dp"

android:paddingRight="4dp"

android:textColor="#ff323232"

android:textSize="14sp"

android:textStyle="bold" />

<com.google.android.exoplayer2.ui.DefaultTimeBar

android:id="@id/exo\_progress"

android:layout\_width="0dp"

android:layout\_height="26dp"

android:layout\_weight="1"

app:played\_color="@color/primary\_dark" />

<TextView

android:id="@id/exo\_duration"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:includeFontPadding="false"

android:paddingLeft="4dp"

android:paddingRight="4dp"

android:textColor="#ff323232"

android:textSize="14sp"

android:textStyle="bold" />

</LinearLayout>

</LinearLayout>

* + - 1. Файл layout/ exo\_player\_view.xml

<?xml version="1.0" encoding="utf-8"?>

<merge xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:android="http://schemas.android.com/apk/res/android">

<com.google.android.exoplayer2.ui.AspectRatioFrameLayout android:id="@id/exo\_content\_frame"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_gravity="center">

<androidx.constraintlayout.widget.ConstraintLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content">

<TextView

android:id="@+id/player\_\_song\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Song name"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

android:layout\_marginTop="20dp"

style="@style/TextAppearance.AppCompat.Headline"

android:elevation="6dp"/>

<TextView

android:id="@+id/player\_\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Artist name"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@id/player\_\_song\_name"

android:layout\_marginTop="10dp"

android:elevation="6dp"/>

<ImageView

android:id="@+id/player\_\_options\_button"

android:layout\_width="35dp"

android:layout\_height="35dp"

android:layout\_alignParentEnd="true"

android:layout\_marginTop="10dp"

android:layout\_marginEnd="10dp"

android:elevation="6dp"

android:src="@drawable/ic\_options"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintTop\_toTopOf="parent" />

<View

android:id="@id/exo\_shutter"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@android:color/white" />

<ImageView

android:id="@+id/artwork"

android:layout\_width="350dp"

android:layout\_height="350dp"

android:elevation="12dp"

android:src="@color/primary\_dark"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="@+id/exo\_shutter" />

<ImageView

android:id="@+id/player\_\_add\_button"

android:layout\_width="35dp"

android:layout\_height="35dp"

android:layout\_below="@id/artwork"

android:layout\_marginTop="10dp"

android:layout\_marginEnd="10dp"

android:src="@drawable/ic\_add"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/artwork" />

<ImageView

android:id="@+id/player\_\_download\_button"

android:layout\_width="35dp"

android:layout\_height="35dp"

android:layout\_below="@id/artwork"

android:layout\_marginTop="10dp"

android:layout\_marginStart="10dp"

android:src="@drawable/ic\_download"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/artwork" />

<TextView

android:id="@id/exo\_error\_message"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@color/exo\_error\_message\_background\_color"

android:gravity="center"

android:padding="16dp"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>

</com.google.android.exoplayer2.ui.AspectRatioFrameLayout>

<FrameLayout android:id="@id/exo\_overlay"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<View android:id="@id/exo\_controller\_placeholder"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

</merge>

* + - 1. Файл layout/ fragment\_album\_list.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentTop="true">

<TextView

android:id="@+id/fragment\_album\_list\_\_title"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/album\_\_list\_title"

android:textSize="@dimen/fragment\_list\_title"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="5dp"

/>

<com.siroytman.indiewindymobile.ui.custom\_views.MyListView

android:id="@android:id/list"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/fragment\_album\_list\_\_title"/>

<Button

android:id="@+id/fragment\_song\_album\_\_more"

style="@style/Widget.MaterialComponents.Button.OutlinedButton"

android:layout\_marginEnd="4dp"

android:layout\_width="@dimen/fragment\_list\_more\_button"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_below="@android:id/list"

android:text="@string/more\_\_button"

android:textSize="@dimen/button\_text\_size"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_artist\_list.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentTop="true">

<TextView

android:id="@+id/fragment\_artist\_list\_\_title"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/artist\_\_list\_title"

android:textSize="@dimen/fragment\_list\_title"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="5dp"

/>

<com.siroytman.indiewindymobile.ui.custom\_views.MyListView

android:id="@android:id/list"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/fragment\_artist\_list\_\_title"/>

<Button

android:id="@+id/fragment\_artist\_list\_\_more"

style="@style/Widget.MaterialComponents.Button.OutlinedButton"

android:layout\_marginEnd="4dp"

android:layout\_width="@dimen/fragment\_list\_more\_button"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_below="@android:id/list"

android:text="@string/more\_\_button"

android:textSize="@dimen/button\_text\_size"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_concert\_artist.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_concert\_artist\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.ArtistConcertFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_concert\_list.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentTop="true">

<com.siroytman.indiewindymobile.ui.custom\_views.MyListView

android:id="@android:id/list"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</RelativeLayout>

* + - 1. Файл layout/ fragment\_concert\_nearest.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_concert\_nearest\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.NearestConcertFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_concert\_saved.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_concert\_saved\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.SavedConcertFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_concert\_subscription.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_concert\_subscription\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.SubscriptionConcertFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_home.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<androidx.viewpager.widget.ViewPager

android:id="@+id/home\_vp\_pager"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.tabs.TabLayout

android:id="@+id/home\_tab\_layout"

style="@style/Widget.MaterialComponents.TabLayout"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</androidx.viewpager.widget.ViewPager>

</LinearLayout>

* + - 1. Файл layout/ fragment\_home\_latest.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_home\_latest\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.LatestHomeFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_home\_subscription.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_home\_subscription\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.SubscriptionHomeFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_personal.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<androidx.viewpager.widget.ViewPager

android:id="@+id/personal\_vp\_pager"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<com.google.android.material.tabs.TabLayout

android:id="@+id/personal\_tab\_layout"

style="@style/Widget.MaterialComponents.TabLayout"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</androidx.viewpager.widget.ViewPager>

</LinearLayout>

* + - 1. Файл layout/ fragment\_personal\_albums.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_personal\_albums\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.personal.AlbumsPersonalFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_personal\_artists.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_personal\_artists\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.personal.ArtistsPersonalFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_personal\_songs.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_personal\_songs\_\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.personal.SongsPersonalFragment"/>

</RelativeLayout>

* + - 1. Файл layout/ fragment\_post\_list.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentTop="true">

<ListView

android:id="@android:id/list"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</RelativeLayout>

* + - 1. Файл layout/ fragment\_search.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:tools="http://schemas.android.com/tools">

<FrameLayout

android:id="@+id/fragment\_search\_\_song\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

tools:context=".ui.fragments.search.SearchFragment"/>

<FrameLayout

android:id="@+id/fragment\_search\_\_artist\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/fragment\_search\_\_song\_container"

android:isScrollContainer="false"

tools:context=".ui.fragments.search.SearchFragment" />

<FrameLayout

android:id="@+id/fragment\_search\_\_album\_container"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/fragment\_search\_\_artist\_container"

tools:context=".ui.fragments.search.SearchFragment" />

</RelativeLayout>

* + - 1. Файл layout/ fragment\_song\_list.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_alignParentTop="true">

<TextView

android:id="@+id/fragment\_song\_list\_\_title"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/song\_\_list\_title"

android:textSize="@dimen/fragment\_list\_title"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="5dp"

style="@style/SmallText"/>

<com.siroytman.indiewindymobile.ui.custom\_views.MyListView

android:id="@android:id/list"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/fragment\_song\_list\_\_title"/>

<Button

android:id="@+id/fragment\_song\_list\_\_more"

style="@style/Widget.MaterialComponents.Button.OutlinedButton"

android:layout\_marginEnd="4dp"

android:layout\_width="@dimen/fragment\_list\_more\_button"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_below="@android:id/list"

android:text="@string/more\_\_button"

android:textSize="@dimen/button\_text\_size"/>

</RelativeLayout>

* + - 1. Файл layout/ post\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<!--TODO card wrap content somehow-->

<com.google.android.material.card.MaterialCardView

android:id="@+id/post\_list\_item\_\_card"

android:layout\_width="match\_parent"

android:layout\_height="500dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:id="@+id/post\_list\_item\_\_layout"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<ImageView

android:id="@+id/post\_list\_item\_\_artist\_photo"

android:layout\_width="60dp"

android:layout\_height="60dp"

android:src="@drawable/ic\_artist"

android:layout\_alignParentStart="true"/>

<TextView

android:id="@+id/post\_list\_item\_\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_toEndOf="@id/post\_list\_item\_\_artist\_photo"

android:layout\_marginHorizontal="12dp"

android:layout\_marginTop="12dp"

android:text="Artist Name"

android:textAppearance="@style/BigText" />

<TextView

android:id="@+id/post\_list\_item\_\_date"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginHorizontal="12dp"

android:layout\_marginTop="12dp"

android:text="Date"

android:textStyle="italic"

style="@style/SmallText"

android:layout\_alignParentEnd="true"/>

<TextView

android:id="@+id/post\_list\_item\_\_text"

android:layout\_width="match\_parent"

android:layout\_height="100dp"

android:text="Post text"

android:layout\_below="@id/post\_list\_item\_\_artist\_photo"

android:layout\_marginHorizontal="12dp"

android:layout\_marginTop="6dp"

android:scrollbars="vertical"

style="@style/SmallText"/>

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл layout/ settings\_activity.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<FrameLayout

android:id="@+id/settings"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

</LinearLayout>

* + - 1. Файл layout/ song\_list\_item.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<com.google.android.material.card.MaterialCardView

android:layout\_width="match\_parent"

android:layout\_height="@dimen/list\_item\_height"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:strokeColor="@color/material\_on\_background\_disabled"

app:strokeWidth="@dimen/list\_item\_stroke">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="@dimen/list\_item\_padding">

<TextView

android:id="@+id/song\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentStart="true"

android:layout\_alignParentTop="true"

android:paddingHorizontal="12dp"

android:paddingTop="12dp"

android:text="Song Name"

android:textAppearance="@style/BigText" />

<TextView

android:id="@+id/artist\_activity\_\_artist\_name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/song\_name"

android:layout\_alignParentStart="true"

android:paddingHorizontal="12dp"

android:paddingTop="2dp"

android:text="Artist Name"

style="@style/SmallText"/>

<ImageView

android:id="@+id/song\_download\_button"

android:layout\_toStartOf="@id/song\_add\_button"

android:layout\_centerVertical="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_download"

android:layout\_marginEnd="5dp"

android:contentDescription="Download button" />

<ImageView

android:id="@+id/song\_add\_button"

android:layout\_toStartOf="@id/song\_options\_button"

android:layout\_centerVertical="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_add"

android:layout\_marginEnd="5dp"

android:contentDescription="Add button" />

<ImageView

android:id="@+id/song\_options\_button"

android:layout\_alignParentEnd="true"

android:layout\_centerVertical="true"

android:layout\_width="@dimen/list\_item\_options"

android:layout\_height="@dimen/list\_item\_options"

android:src="@drawable/ic\_options"

android:contentDescription="Options button" />

</RelativeLayout>

</com.google.android.material.card.MaterialCardView>

</androidx.constraintlayout.widget.ConstraintLayout>

* + - 1. Файл menu/ bottom\_nav\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android">

<item

android:id="@+id/navigation\_home"

android:icon="@drawable/ic\_home\_black\_24dp"

android:title="@string/navigation\_\_title\_home" />

<item

android:id="@+id/navigation\_search"

android:icon="@drawable/ic\_search\_black"

android:title="@string/navigation\_\_title\_search"/>

<item

android:id="@+id/navigation\_concert"

android:icon="@drawable/ic\_concert"

android:title="@string/navigation\_\_title\_concert" />

<item

android:id="@+id/navigation\_personal"

android:icon="@drawable/ic\_artist"

android:title="@string/navigation\_\_title\_personal" />

</menu>

* + - 1. Файл menu/ concert\_search\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

tools:context=".HomeActivity">

<item

android:id="@+id/concert\_search\_menu\_\_spinner"

android:title="@string/concert\_search\_menu\_\_spinner"

app:actionViewClass="android.widget.Spinner"

app:showAsAction="always" />

<item

android:id="@+id/concert\_search\_menu\_\_search"

android:icon="@android:drawable/ic\_menu\_search"

android:title="@string/concert\_search\_menu\_\_search\_title"

app:actionViewClass="androidx.appcompat.widget.SearchView"

app:showAsAction="always" />

</menu>

* + - 1. Файл menu/ personal\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

tools:context=".HomeActivity">

<item

android:id="@+id/personal\_search"

android:icon="@android:drawable/ic\_menu\_search"

android:title="Search"

app:actionViewClass="androidx.appcompat.widget.SearchView"

app:showAsAction="always" />

<item

android:id="@+id/personal\_settings"

android:icon="@drawable/ic\_settings"

android:title="Settings"

app:showAsAction="always"/>

</menu>

* + - 1. Файл menu/ popup\_album\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android" >

<group android:id="@+id/album\_options\_menu\_group">

<item

android:id="@+id/album\_options\_menu\_artist"

android:title="@string/popup\_\_artist"

android:icon="@drawable/ic\_artist"/>

</group>

</menu>

* + - 1. Файл menu/ popup\_song\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android" >

<group android:id="@+id/song\_options\_menu\_group">

<item

android:id="@+id/song\_options\_menu\_artist"

android:title="@string/popup\_\_artist"

android:icon="@drawable/ic\_artist"/>

<item

android:id="@+id/song\_options\_menu\_album"

android:title="@string/popup\_\_album"

android:icon="@drawable/ic\_album"/>

</group>

</menu>

* + - 1. Файл menu/ search\_menu.xml

<?xml version="1.0" encoding="utf-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

tools:context=".HomeActivity">

<item

android:id="@+id/search\_menu\_\_search"

android:icon="@android:drawable/ic\_menu\_search"

android:title="@string/concert\_search\_menu\_\_search\_title"

app:actionViewClass="androidx.appcompat.widget.SearchView"

app:showAsAction="always" />

</menu>

* + - 1. Файл navigation/ mobile\_navigation.xml

<?xml version="1.0" encoding="utf-8"?>

<navigation xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/mobile\_navigation"

app:startDestination="@+id/navigation\_home">

<fragment

android:id="@+id/navigation\_home"

android:name="com.siroytman.indiewindymobile.ui.fragments.home.HomeFragment"

android:label="@string/navigation\_\_title\_home"

tools:layout="@layout/fragment\_home" />

<fragment

android:id="@+id/navigation\_search"

android:name="com.siroytman.indiewindymobile.ui.fragments.search.SearchFragment"

android:label="@string/navigation\_\_title\_search"

tools:layout="@layout/fragment\_search" />

<fragment

android:id="@+id/navigation\_concert"

android:name="com.siroytman.indiewindymobile.ui.fragments.concert.ConcertFragment"

android:label="@string/navigation\_\_title\_concert"

tools:layout="@layout/fragment\_concert" />

<fragment

android:id="@+id/navigation\_personal"

android:name="com.siroytman.indiewindymobile.ui.fragments.personal.PersonalFragment"

android:label="@string/navigation\_\_title\_personal"

tools:layout="@layout/fragment\_personal" />

</navigation>

1. ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |