**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Руководитель проекта,  доцент департамента больших данных и информационного поиска, к.ф-м.н    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.В. Вознесенская  «20» \_\_\_\_мая\_\_\_\_\_\_ 2022 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия», кандидат технических наук  Автограф В В.В. Шилов  «20» мая 2022 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Интерактивная визуализация данных приемной кампании**  **Текст программы**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.10.03-01 12 01-1-ЛУ** | | |
|  |  | |
| Изображение выглядит как текст  Автоматически созданное описаниеИсполнитель:  студент группы БПИ203  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Я.Ю. Щербаков /  «19» \_\_\_\_\_\_\_мая\_\_\_\_\_\_\_\_\_\_ 2022 г. | |
|  | | |
| **Москва 2022** | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.10.03-01 12 01-1-ЛУ |  |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Интерактивная визуализация данных приемной кампании**  **Текст программы**  **RU.17701729.10.03-01 12 01-1**  **Листов 33** | | | | |
|  |  | | | |
|  | | | |
|  | | | | |
|  | | |  | |

**Москва 2022**

СОДЕРЖАНИЕ

[1. ТЕКСТ ПРОГРАММЫ 4](#_Toc103877611)

[**1.1. Проект AdmissionCampaign.API 4**](#_Toc103877612)

[**1.1.1.** **Класс EntrantController 4**](#_Toc103877613)

[**1.1.2. Класс SwaggerFileOperationFilter 6**](#_Toc103877614)

[**1.1.3. Класс HttpStatusCodeExceptionMiddleware 7**](#_Toc103877615)

[**1.1.4. Класс Program 8**](#_Toc103877616)

[**1.1.5. Класс Startup 8**](#_Toc103877617)

[**1.2. Проект AdmissionCampaign.DataAccess 10**](#_Toc103877618)

[**1.2.1. Класс Entrant 10**](#_Toc103877619)

[**1.2.2. Класс FilteredEntrantsModel 11**](#_Toc103877620)

[**1.2.3. Класс FilterEntrantsModel 11**](#_Toc103877621)

[**1.2.4. Класс GetStaticticByRegionsModel 12**](#_Toc103877622)

[**1.2.5. Класс PaginationModel 12**](#_Toc103877623)

[**1.2.6. Класс RegionStaticticModel 12**](#_Toc103877624)

[**1.2.7. Класс EntrantRepository 12**](#_Toc103877625)

[**1.2.8. Класс ApplicationContext 18**](#_Toc103877626)

[**1.2.9. Класс ApplicationContextFactory 18**](#_Toc103877627)

[**1.2.10. Класс StartupExtension 19**](#_Toc103877628)

[**1.3. Проект AdmissionCampaign.Services 19**](#_Toc103877629)

[**1.3.1. Класс CsvParser 19**](#_Toc103877630)

[**1.3.2. Класс EntrantMapper 21**](#_Toc103877631)

[**1.3.3. Класс EntrantService 22**](#_Toc103877632)

[**1.3.4. Класс StartupExtension 27**](#_Toc103877633)

[**1.4. Проект AdmissionCampaign.ViewModels 27**](#_Toc103877634)

[**1.4.1. Класс AddEntrantViewModel 27**](#_Toc103877635)

[**1.4.2. Класс EntrantView 28**](#_Toc103877636)

[**1.4.3. Класс FilterEntrantsViewModel 29**](#_Toc103877637)

[**1.4.4. Класс FiltredEntrantsView 30**](#_Toc103877638)

[**1.4.5. Класс GetEntrantsByEntryYearViewModel 30**](#_Toc103877639)

[**1.4.6. Класс GetStaticticByRegionsView 30**](#_Toc103877640)

[**1.4.7. Класс GetStaticticByRegionsViewModel 30**](#_Toc103877641)

[**1.4.8. Класс PaginationViewModel 31**](#_Toc103877642)

[**1.4.9. Класс RegionStaticticView 31**](#_Toc103877643)

[**1.4.10. Класс UpdateEntrantViewModel 31**](#_Toc103877644)

[2. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 32](#_Toc103877645)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 33](#_Toc103877646)

1. **ТЕКСТ ПРОГРАММЫ**

## Проект AdmissionCampaign.API

### Класс EntrantController

using AdmissionCampaign.ViewModels.Entrants;

using AdmissionСampaign.Services.Services;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace AdmissionСampaign.Controllers

{

[ApiController]

[Route("Entrant")]

public class EntrantController : ControllerBase

{

private readonly EntrantService \_entrantService;

public EntrantController(EntrantService entrantService)

{

\_entrantService = entrantService;

}

[HttpPut("AddEntrant")]

[ProducesResponseType(typeof(long), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult AddEntrant([FromBody] AddEntrantViewModel model)

{

var id = \_entrantService.AddEntrant(model);

return Ok(id);

}

[HttpPost("UpdateEntrant")]

[ProducesResponseType(typeof(void), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult UpdateEntrant([FromBody] UpdateEntrantViewModel model)

{

\_entrantService.UpdateEntrant(model);

return Ok();

}

[HttpDelete("RemoveEntrant/{id}")]

[ProducesResponseType(typeof(void), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult RemoveEntrant([FromRoute] long id)

{

\_entrantService.RemoveEntrant(id);

return Ok();

}

[HttpDelete("RemoveAllEntrants")]

[ProducesResponseType(typeof(void), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult RemoveAllEntrants()

{

\_entrantService.RemoveAllEntrants();

return Ok();

}

[HttpGet("GetByEntrantId/{id}")]

[ProducesResponseType(typeof(EntrantView), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult GetByEntrantId([FromRoute] long id)

{

var entrant = \_entrantService.FindEntrantById(id);

return Ok(entrant);

}

[HttpGet("GetAllEntrants")]

[ProducesResponseType(typeof(FiltredEntrantsView), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult GetAllEntrants([FromQuery] PaginationViewModel model)

{

var entrants = \_entrantService.GetAllEntrants(model);

return Ok(entrants);

}

[HttpGet("GetEntrantsByEntryYear")]

[ProducesResponseType(typeof(FiltredEntrantsView), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult GetEntrantsByEntryYear([FromQuery] GetEntrantsByEntryYearViewModel model)

{

var entrants = \_entrantService.GetEntrantsByEntryYear(model);

return Ok(entrants);

}

[HttpGet("FilterEntrants")]

[ProducesResponseType(typeof(FiltredEntrantsView), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult FilterEntrants([FromQuery] FilterEntrantsViewModel model)

{

var entrants = \_entrantService.FilterEntrants(model);

return Ok(entrants);

}

[HttpGet("GetStaticticByRegions")]

[ProducesResponseType(typeof(GetStaticticByRegionsView), 200)]

[ProducesResponseType(typeof(string), 400)]

public IActionResult GetStaticticByRegions([FromQuery] GetStaticticByRegionsViewModel model)

{

var entrants = \_entrantService.GetStaticticByRegions(model);

return Ok(entrants);

}

[HttpPost("UploadDocument")]

public IActionResult UploadDocument([FromQuery] int entryYear, [FromQuery] string educationProgram, [FromForm] IFormFile file)

{

if (string.IsNullOrWhiteSpace(educationProgram))

{

return BadRequest("EducationProgram is required");

}

if (entryYear <= 0)

{

return BadRequest("EntryYear is required and positive");

}

if (file.Length <= 0)

{

return BadRequest("File is empty");

}

using var fileStream = file.OpenReadStream();

\_entrantService.UploadDocument(entryYear, educationProgram, fileStream);

return Ok();

}

}

}

### Класс SwaggerFileOperationFilter

using Microsoft.AspNetCore.Http;

using Microsoft.OpenApi.Models;

using Swashbuckle.AspNetCore.SwaggerGen;

using System;

using System.Linq;

namespace AdmissionCampaign.API.Filters

{

public class SwaggerFileOperationFilter : IOperationFilter

{

public void Apply(OpenApiOperation operation, OperationFilterContext context)

{

var fileUploadMime = "multipart/form-data";

if (operation.RequestBody == null || !operation.RequestBody.Content.Any(x => x.Key.Equals(fileUploadMime, StringComparison.InvariantCultureIgnoreCase)))

return;

var fileParams = context.MethodInfo.GetParameters().Where(p => p.ParameterType == typeof(IFormFile));

operation.RequestBody.Content[fileUploadMime].Schema.Properties =

fileParams.ToDictionary(k => k.Name, v => new OpenApiSchema()

{

Type = "string",

Format = "binary"

});

}

}

}

### Класс HttpStatusCodeExceptionMiddleware

using Microsoft.AspNetCore.Http;

using System;

using System.Net;

using System.Threading.Tasks;

namespace AdmissionCampaign.API.Middlewares

{

public class HttpStatusCodeExceptionMiddleware

{

private readonly RequestDelegate \_next;

public HttpStatusCodeExceptionMiddleware(RequestDelegate next)

{

\_next = next;

}

public async Task InvokeAsync(HttpContext context)

{

try

{

Console.WriteLine("start");

await \_next(context);

Console.WriteLine("end");

}

catch (ApplicationException exception)

{

await HandleDefaultException(context, HttpStatusCode.BadRequest, exception);

}

catch (Exception)

{

await HandleDefaultException(context, HttpStatusCode.InternalServerError);

}

}

public async Task HandleDefaultException(HttpContext context, HttpStatusCode httpStatusCode, Exception exception = null)

{

context.Response.Clear();

context.Response.StatusCode = (int)httpStatusCode;

context.Response.ContentType = @"application/json";

var isCustomException = exception != null ? exception.GetType().IsSubclassOf(typeof(Exception)) : false;

if (isCustomException)

{

await context.Response.WriteAsync(exception?.Message);

return;

}

await context.Response.WriteAsync("Server is not responding. Try later");

}

}

}

### Класс Program

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Hosting;

namespace AdmissionСampaign

{

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>();

});

}

}

### Класс Startup

using AdmissionCampaign.API.Filters;

using AdmissionCampaign.API.Middlewares;

using AdmissionСampaign.Services;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.Http;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.OpenApi.Models;

namespace AdmissionСampaign

{

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)

{

services.AddSingleton<IHttpContextAccessor, HttpContextAccessor>();

var connectionString = Configuration.GetConnectionString("DefaultConnection");

services.AddServices(connectionString);

services.AddControllers();

services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo { Title = "AdmissionСampaign", Version = "v1" });

c.OperationFilter<SwaggerFileOperationFilter>();

});

}

// 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.UseSwagger();

app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "AdmissionСampaign v1"));

app.UseMiddleware<HttpStatusCodeExceptionMiddleware>();

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

});

}

}

}

## Проект AdmissionCampaign.DataAccess

### Класс Entrant

namespace AdmissionСampaign.DataAccess.Entities

{

public class Entrant

{

public long Id { get; set; }

public int EntryYear { get; set; }

public string EducationProgram { get; set; }

public int RegistrationNumber { get; set; }

public string FullName { get; set; }

public string SNILS { get; set; }

public string ApplicationType { get; set; }

public string WithoutExamsReason { get; set; }

public bool SpecialQuota { get; set; }

public bool TargetQuota { get; set; }

public int InformaticsUSE { get; set; }

public int MathUSE { get; set; }

public int RussianLanguageUSE { get; set; }

public int IndividualAchievementScore { get; set; }

public string EducationForm { get; set; }

public string PreemptiveRight { get; set; }

public bool IsDormitoryNeeded { get; set; }

public bool DocumentsReturned { get; set; }

public string Specialization { get; set; }

public string EducationCompetitions { get; set; }

public string RegistrationRegion { get; set; }

}

}

### Класс FilteredEntrantsModel

using AdmissionСampaign.DataAccess.Entities;

using System.Collections.Generic;

namespace AdmissionCampaign.DataAccess.models

{

public class FilteredEntrantsModel

{

public List<Entrant> Entrants { get; set; }

public int Total { get; set; }

}

}

### 1.2.3. Класс FilterEntrantsModel

namespace AdmissionCampaign.DataAccess.models

{

public class FilterEntrantsModel

{

public int? EntryYear { get; set; }

public string EducationProgram { get; set; }

public string NameSearch { get; set; }

public string RegistrationRegion { get; set; }

public bool? IsDormitoryNeeded { get; set; }

public bool? HasTargetQuota { get; set; }

public bool? HasSpecialQuota { get; set; }

public bool? IsDocumentsReturned { get; set; }

public int? MinTotalUse { get; set; }

public int? MaxTotalUse { get; set; }

public int? MinRussianLanguageUSE { get; set; }

public int? MaxRussianLanguageUSE { get; set; }

public int? MinInformaticsUSE { get; set; }

public int? MaxInformaticsUSE { get; set; }

public int? MinMathUSE { get; set; }

public int? MaxMathUSE { get; set; }

}

}

### 1.2.4. Класс GetStaticticByRegionsModel

using System.Collections.Generic;

namespace AdmissionCampaign.DataAccess.models

{

public class GetStaticticByRegionsModel

{

public List<RegionStaticticModel> RegionStatictic { get; set; }

public int Total { get; set; }

}

}

### 1.2.5. Класс PaginationModel

namespace AdmissionCampaign.DataAccess.models

{

public class PaginationModel

{

public int Skip { get; set; } = 0;

public int Take { get; set; } = 15;

}

}

### 1.2.6. Класс RegionStaticticModel

namespace AdmissionCampaign.DataAccess.models

{

public class RegionStaticticModel

{

public string Region { get; set; }

public int Count { get; set; }

}

}

### 1.2.7. Класс EntrantRepository

using AdmissionCampaign.DataAccess.models;

using AdmissionСampaign.DataAccess.Entities;

using Microsoft.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore.Storage;

using System.Collections.Generic;

using System.Linq;

namespace AdmissionСampaign.DataAccess.Repositories

{

public class EntrantRepository

{

protected readonly DbSet<Entrant> \_dbSet;

protected readonly ApplicationContext \_context;

private IDbContextTransaction \_transaction;

public EntrantRepository(ApplicationContext context)

{

\_context = context;

\_dbSet = \_context.Set<Entrant>();

}

public void BeginTransaction()

{

\_transaction = \_context.Database.BeginTransaction();

}

public void RollbackTransaction()

{

\_transaction.Rollback();

\_transaction.Dispose();

}

public void CommitTransaction()

{

\_transaction.Commit();

\_transaction.Dispose();

}

public long AddEntrant(Entrant entrant)

{

\_dbSet.Add(entrant);

\_context.SaveChanges();

return entrant.Id;

}

public void AddRange(List<Entrant> entrants)

{

\_dbSet.AddRange(entrants);

\_context.SaveChanges();

}

public void UpdateEntrant(Entrant entrant)

{

\_dbSet.Update(entrant);

\_context.SaveChanges();

}

public void RemoveEntrant(Entrant entrant)

{

\_dbSet.Remove(entrant);

\_context.SaveChanges();

}

public Entrant FindEntrantById(long id)

{

var entrant = \_dbSet.AsQueryable().FirstOrDefault(entrant => entrant.Id == id);

return entrant;

}

public FilteredEntrantsModel GetAllEntrants(PaginationModel paginationModel)

{

var query = \_dbSet.AsQueryable();

var retult = new FilteredEntrantsModel()

{

Entrants = query.Skip(paginationModel.Skip).Take(paginationModel.Take).ToList(),

Total = query.Count()

};

return retult;

}

public void RemoveAllEntrants()

{

\_dbSet.RemoveRange(\_dbSet);

\_context.SaveChanges();

}

public FilteredEntrantsModel GetEntrantsByEntryYear(int year, PaginationModel paginationModel)

{

var query = \_dbSet.AsQueryable();

query = AddEntryYearFilter(query, year, year);

var retult = new FilteredEntrantsModel()

{

Entrants = query.Skip(paginationModel.Skip).Take(paginationModel.Take).ToList(),

Total = query.Count()

};

return retult;

}

public GetStaticticByRegionsModel GetStaticticByRegions(FilterEntrantsModel model, PaginationModel paginationModel)

{

var query = \_dbSet.AsQueryable();

query = AddEntryYearFilter(query, model.EntryYear, model.EntryYear);

if (!string.IsNullOrWhiteSpace(model.EducationProgram))

{

query = query.Where(entrant => entrant.EducationProgram == model.EducationProgram);

}

var regionStatictic = query.GroupBy(x=>x.RegistrationRegion)

.Select(g => new RegionStaticticModel { Region = g.Key, Count = g.Count() })

.OrderByDescending(x=>x.Count);

var retult = new GetStaticticByRegionsModel()

{

RegionStatictic = regionStatictic.Skip(paginationModel.Skip).Take(paginationModel.Take).ToList(),

Total = regionStatictic.Count()

};

return retult;

}

public FilteredEntrantsModel FilterEntrants(FilterEntrantsModel model, PaginationModel paginationModel)

{

var query = \_dbSet.AsQueryable();

query = AddEntryYearFilter(query, model.EntryYear, model.EntryYear);

if (!string.IsNullOrWhiteSpace(model.EducationProgram))

{

query = query.Where(entrant => entrant.EducationProgram == model.EducationProgram);

}

if (!string.IsNullOrWhiteSpace(model.RegistrationRegion))

{

query = query.Where(entrant => entrant.RegistrationRegion == model.RegistrationRegion);

}

if (!string.IsNullOrWhiteSpace(model.NameSearch))

{

query = query.Where(entrant => entrant.FullName.Contains(model.NameSearch));

}

if (model.IsDormitoryNeeded is not null)

{

query = query.Where(entrant => entrant.IsDormitoryNeeded == model.IsDormitoryNeeded);

}

if (model.HasSpecialQuota is not null)

{

query = query.Where(entrant => entrant.SpecialQuota == model.HasSpecialQuota);

}

if (model.HasTargetQuota is not null)

{

query = query.Where(entrant => entrant.TargetQuota == model.HasTargetQuota);

}

query = AddDocumentsReturnedFilter(query, model.IsDocumentsReturned);

query = AddRussianLanguageUSEFilter(query, model.MinRussianLanguageUSE, model.MaxRussianLanguageUSE);

query = AddMathUSEFilter(query, model.MinMathUSE, model.MaxMathUSE);

query = AddInformaticsUseFilter(query, model.MinInformaticsUSE, model.MaxInformaticsUSE);

query = AddTotalUseFilter(query, model.MinTotalUse, model.MaxTotalUse);

var retult = new FilteredEntrantsModel()

{

Entrants = query.Skip(paginationModel.Skip).Take(paginationModel.Take).ToList(),

Total = query.Count()

};

return retult;

}

private IQueryable<Entrant> AddDocumentsReturnedFilter(IQueryable<Entrant> query, bool? IsDocumentsReturned)

{

if (IsDocumentsReturned is not null)

{

query = query.Where(entrant => entrant.DocumentsReturned == IsDocumentsReturned);

}

return query;

}

private IQueryable<Entrant> AddEntryYearFilter(IQueryable<Entrant> query, int? min, int? max)

{

if (min is not null)

{

query = query.Where(entrant => entrant.EntryYear >= min);

}

if (max is not null)

{

query = query.Where(entrant => entrant.EntryYear <= max);

}

return query;

}

private IQueryable<Entrant> AddRussianLanguageUSEFilter(IQueryable<Entrant> query, int? min, int? max)

{

if (min is not null)

{

query = query.Where(entrant => entrant.RussianLanguageUSE >= min);

}

if (max is not null)

{

query = query.Where(entrant => entrant.RussianLanguageUSE <= max);

}

return query;

}

private IQueryable<Entrant> AddMathUSEFilter(IQueryable<Entrant> query, int? min, int? max)

{

if (min is not null)

{

query = query.Where(entrant => entrant.MathUSE >= min);

}

if (max is not null)

{

query = query.Where(entrant => entrant.MathUSE <= max);

}

return query;

}

private IQueryable<Entrant> AddInformaticsUseFilter(IQueryable<Entrant> query, int? min, int? max)

{

if (min is not null)

{

query = query.Where(entrant => entrant.InformaticsUSE >= min);

}

if (max is not null)

{

query = query.Where(entrant => entrant.InformaticsUSE <= max);

}

return query;

}

private IQueryable<Entrant> AddTotalUseFilter(IQueryable<Entrant> query, int? min, int? max)

{

if (min is not null)

{

query = query.Where(entrant => entrant.RussianLanguageUSE +

entrant.MathUSE + entrant.InformaticsUSE + entrant.IndividualAchievementScore >= min);

}

if (max is not null)

{

query = query.Where(entrant => entrant.RussianLanguageUSE +

entrant.MathUSE + entrant.InformaticsUSE + entrant.IndividualAchievementScore <= max);

}

return query;

}

}

}

### 1.2.8. Класс ApplicationContext

using AdmissionСampaign.DataAccess.Entities;

using Microsoft.EntityFrameworkCore;

namespace AdmissionСampaign.DataAccess

{

public class ApplicationContext : DbContext

{

public DbSet<Entrant> Entrants { get; set; }

public ApplicationContext(DbContextOptions options) : base(options)

{

Database.Migrate();

}

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

optionsBuilder.UseLazyLoadingProxies();

}

}

}

### 1.2.9. Класс ApplicationContextFactory

using AdmissionСampaign.DataAccess;

using Microsoft.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore.Design;

using Microsoft.Extensions.Configuration;

using System.IO;

namespace AdmissionCampaign.DataAccess

{

class ApplicationContextFactory : IDesignTimeDbContextFactory<ApplicationContext>

{

public ApplicationContext CreateDbContext(string[] args)

{

IConfigurationRoot configuration = new ConfigurationBuilder()

.SetBasePath(Path.Combine(Directory.GetCurrentDirectory(), @"..\AdmissionСampaign"))

.AddJsonFile("appsettings.json")

.AddJsonFile("appsettings.Development.json")

.Build();

var builder = new DbContextOptionsBuilder<ApplicationContext>();

var connectionString = configuration.GetConnectionString("DefaultConnection");

//this code will be never executed in runtime only in design time

builder.UseSqlServer(connectionString);

return new ApplicationContext(builder.Options);

}

}

}

### 1.2.10. Класс StartupExtension

using AdmissionСampaign.DataAccess.Repositories;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.DependencyInjection;

namespace AdmissionСampaign.DataAccess

{

public static class StartupExtension

{

public static void AddDBContext(this IServiceCollection services, string connectionString)

{

services.AddDbContext<ApplicationContext>(options =>

options.UseSqlServer(connectionString));

// SQL configuration for non-injected dbcontext

DbContextOptionsBuilder<ApplicationContext> builder = new DbContextOptionsBuilder<ApplicationContext>();

builder.UseSqlServer(connectionString);

services.AddSingleton(builder.Options);

}

public static void AddRepositories(this IServiceCollection services)

{

services.AddTransient<EntrantRepository, EntrantRepository>();

}

}

}

## Проект AdmissionCampaign.Services

### 1.3.1. Класс CsvParser

namespace AdmissionCampaign.Services.Helpers

{

using System.Collections.Generic;

using System.Text;

class CsvParser

{

public static List<string> Parse(string line)

{

const char escapeChar = '"';

const char splitChar = ';';

bool inEscape = false;

bool priorEscape = false;

List<string> result = new List<string>();

StringBuilder sb = new StringBuilder();

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

{

char c = line[i];

switch (c)

{

case escapeChar:

if (!inEscape)

inEscape = true;

else

{

if (!priorEscape)

{

if (i + 1 < line.Length && line[i + 1] == escapeChar)

priorEscape = true;

else

inEscape = false;

}

else

{

sb.Append(c);

priorEscape = false;

}

}

break;

case splitChar:

if (inEscape)

sb.Append(c);

else

{

result.Add(sb.ToString());

sb.Length = 0;

}

break;

default:

sb.Append(c);

break;

}

}

if (sb.Length > 0)

result.Add(sb.ToString());

return result;

}

}

}

### 1.3.2. Класс EntrantMapper

using AdmissionCampaign.ViewModels.Entrants;

using AdmissionСampaign.DataAccess.Entities;

namespace AdmissionCampaign.Services.Mappers

{

public static class EntrantMapper

{

public static void MapEntrantViewModelToEntity(AddEntrantViewModel model, Entrant entrant)

{

entrant.EntryYear = model.EntryYear;

entrant.EducationProgram = model.EducationProgram;

entrant.RegistrationNumber = model.RegistrationNumber;

entrant.FullName = model.FullName;

entrant.SNILS = model.SNILS;

entrant.ApplicationType = model.ApplicationType;

entrant.WithoutExamsReason = model.WithoutExamsReason;

entrant.SpecialQuota = model.SpecialQuota;

entrant.TargetQuota = model.TargetQuota;

entrant.InformaticsUSE = model.InformaticsUSE;

entrant.MathUSE = model.MathUSE;

entrant.RussianLanguageUSE = model.RussianLanguageUSE;

entrant.IndividualAchievementScore = model.IndividualAchievementScore;

entrant.EducationForm = model.EducationForm;

entrant.PreemptiveRight = model.PreemptiveRight;

entrant.IsDormitoryNeeded = model.IsDormitoryNeeded;

entrant.DocumentsReturned = model.DocumentsReturned;

entrant.Specialization = model.Specialization;

entrant.EducationCompetitions = model.EducationCompetitions;

entrant.RegistrationRegion = model.RegistrationRegion;

}

public static EntrantView MapEntrantEntityToView(Entrant entrant)

{

var entrantView = new EntrantView();

entrantView.Id = entrant.Id;

entrantView.EntryYear = entrant.EntryYear;

entrantView.EducationProgram = entrant.EducationProgram;

entrantView.RegistrationNumber = entrant.RegistrationNumber;

entrantView.FullName = entrant.FullName;

entrantView.SNILS = entrant.SNILS;

entrantView.ApplicationType = entrant.ApplicationType;

entrantView.WithoutExamsReason = entrant.WithoutExamsReason;

entrantView.SpecialQuota = entrant.SpecialQuota;

entrantView.TargetQuota = entrant.TargetQuota;

entrantView.InformaticsUSE = entrant.InformaticsUSE;

entrantView.MathUSE = entrant.MathUSE;

entrantView.RussianLanguageUSE = entrant.RussianLanguageUSE;

entrantView.IndividualAchievementScore = entrant.IndividualAchievementScore;

int CompetitionPointsSum = entrantView.InformaticsUSE + entrantView.MathUSE +

entrantView.RussianLanguageUSE + entrantView.IndividualAchievementScore;

entrantView.CompetitionPoints = CompetitionPointsSum;

entrantView.EducationForm = entrant.EducationForm;

entrantView.PreemptiveRight = entrant.PreemptiveRight;

entrantView.IsDormitoryNeeded = entrant.IsDormitoryNeeded;

entrantView.DocumentsReturned = entrant.DocumentsReturned;

entrantView.Specialization = entrant.Specialization;

entrantView.EducationCompetitions = entrant.EducationCompetitions;

entrantView.RegistrationRegion = entrant.RegistrationRegion;

return entrantView;

}

}

}

### 1.3.3. Класс EntrantService

using AdmissionCampaign.DataAccess.models;

using AdmissionCampaign.Services.Helpers;

using AdmissionCampaign.Services.Mappers;

using AdmissionCampaign.ViewModels.Entrants;

using AdmissionСampaign.DataAccess.Entities;

using AdmissionСampaign.DataAccess.Repositories;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

namespace AdmissionСampaign.Services.Services

{

public class EntrantService

{

private readonly EntrantRepository \_entrantRepository;

public EntrantService(EntrantRepository entrantRepository)

{

\_entrantRepository = entrantRepository;

}

public long AddEntrant(AddEntrantViewModel entrant)

{

var newEntrant = new Entrant();

EntrantMapper.MapEntrantViewModelToEntity(entrant, newEntrant);

return \_entrantRepository.AddEntrant(newEntrant);

}

public void RemoveAllEntrants()

{

\_entrantRepository.RemoveAllEntrants();

}

public void UpdateEntrant(UpdateEntrantViewModel model)

{

var entrant = \_entrantRepository.FindEntrantById(model.Id);

if (entrant is null)

{

throw new ApplicationException("Entrant not found");

}

EntrantMapper.MapEntrantViewModelToEntity(model, entrant);

\_entrantRepository.UpdateEntrant(entrant);

}

public void RemoveEntrant(long id)

{

var entrant = \_entrantRepository.FindEntrantById(id);

if (entrant is null)

{

throw new ApplicationException("Entrant not found");

}

\_entrantRepository.RemoveEntrant(entrant);

}

public EntrantView FindEntrantById(long id)

{

var entrant = \_entrantRepository.FindEntrantById(id);

return EntrantMapper.MapEntrantEntityToView(entrant);

}

public FiltredEntrantsView GetAllEntrants(PaginationViewModel request)

{

var model = GetPaginationModel(request);

var entrantsResult = \_entrantRepository.GetAllEntrants(model);

var responseModel = new FiltredEntrantsView()

{

Entrants = entrantsResult.Entrants.Select(x => EntrantMapper.MapEntrantEntityToView(x)).ToList(),

Total = entrantsResult.Total

};

return responseModel;

}

private PaginationModel GetPaginationModel(PaginationViewModel request)

{

var paginationModel = new PaginationModel()

{

Skip = request.Page < 2 ? 0 : (request.Page - 1) \* request.ItemsPerPage,

Take = request.ItemsPerPage,

};

return paginationModel;

}

public FiltredEntrantsView GetEntrantsByEntryYear(GetEntrantsByEntryYearViewModel request)

{

var paginationModel = GetPaginationModel(request);

var entrantsResult = \_entrantRepository.GetEntrantsByEntryYear(request.EntryYear, paginationModel);

var responseModel = new FiltredEntrantsView()

{

Entrants = entrantsResult.Entrants.Select(x => EntrantMapper.MapEntrantEntityToView(x)).ToList(),

Total = entrantsResult.Total

};

return responseModel;

}

public FiltredEntrantsView FilterEntrants(FilterEntrantsViewModel request)

{

var filter = new FilterEntrantsModel()

{

EntryYear = request.EntryYear,

EducationProgram = request.EducationProgram,

NameSearch = request.NameSearch,

RegistrationRegion = request.RegistrationRegion,

IsDormitoryNeeded = request.IsDormitoryNeeded,

MinTotalUse = request.MinTotalUse,

MaxTotalUse = request.MaxTotalUse,

MinRussianLanguageUSE = request.MinRussianLanguageUSE,

MaxRussianLanguageUSE = request.MaxRussianLanguageUSE,

MinMathUSE = request.MinMathUSE,

MaxMathUSE = request.MaxMathUSE,

MinInformaticsUSE = request.MinInformaticsUSE,

MaxInformaticsUSE = request.MaxInformaticsUSE,

HasSpecialQuota = request.HasSpecialQuota,

HasTargetQuota = request.HasTargetQuota,

IsDocumentsReturned = request.IsDocumentsReturned

};

var paginationModel = GetPaginationModel(request);

var entrantsResult = \_entrantRepository.FilterEntrants(filter, paginationModel);

var responseModel = new FiltredEntrantsView()

{

Entrants = entrantsResult.Entrants.Select(x => EntrantMapper.MapEntrantEntityToView(x)).ToList(),

Total = entrantsResult.Total

};

return responseModel;

}

public GetStaticticByRegionsView GetStaticticByRegions(GetStaticticByRegionsViewModel request)

{

var filter = new FilterEntrantsModel()

{

EntryYear = request.EntryYear,

EducationProgram = request.EducationProgram

};

var paginationModel = GetPaginationModel(request);

var statisticResult = \_entrantRepository.GetStaticticByRegions(filter, paginationModel);

var responseModel = new GetStaticticByRegionsView()

{

RegionStatictic = statisticResult.RegionStatictic.Select(x => new RegionStaticticView()

{

Region = x.Region,

Count = x.Count

}).ToList(),

Total = statisticResult.Total

};

return responseModel;

}

public void UploadDocument(int entryYear, string educationProgram, Stream fileStream)

{

var entrants = new List<Entrant>();

using (var reader = new StreamReader(fileStream))

{

// Skip first row with headers

string line = reader.ReadLine();

var lineNumber = 1;

while ((line = reader.ReadLine()) != null)

{

++lineNumber;

try

{

var entrant = ParseEntrantFromCsv(line);

entrant.EntryYear = entryYear;

entrant.EducationProgram = educationProgram;

entrants.Add(entrant);

}

catch (Exception)

{

throw new ApplicationException($"Parse error in {lineNumber} ");

}

}

}

\_entrantRepository.AddRange(entrants);

}

private Entrant ParseEntrantFromCsv(string entrantString)

{

var entrantFields = CsvParser.Parse(entrantString);

var result = new Entrant();

result.RegistrationNumber = ParseStringToInt(entrantFields[1]);

result.FullName = entrantFields[2];

result.SNILS = entrantFields[3];

result.ApplicationType = entrantFields[4];

result.WithoutExamsReason = entrantFields[5];

result.SpecialQuota = ParseStringToBool(entrantFields[6]);

result.TargetQuota = ParseStringToBool(entrantFields[7]);

result.InformaticsUSE = ParseStringToInt(entrantFields[8]);

result.MathUSE = ParseStringToInt(entrantFields[9]);

result.RussianLanguageUSE = ParseStringToInt(entrantFields[10]);

result.IndividualAchievementScore = ParseStringToInt(entrantFields[11]);

result.EducationForm = entrantFields[13];

result.PreemptiveRight = entrantFields[14];

result.IsDormitoryNeeded = ParseStringToBool(entrantFields[15]);

result.DocumentsReturned = ParseStringToBool(entrantFields[16]);

result.Specialization = entrantFields[17];

result.EducationCompetitions = entrantFields[18];

result.RegistrationRegion = entrantFields.Count > 19 ? entrantFields[19] : string.Empty;

return result;

}

private int ParseStringToInt(string input)

{

if (int.TryParse(input, out int result))

{

return result;

}

return 0;

}

private bool ParseStringToBool(string input)

{

return input.Trim() == "+";

}

}

}

### 1.3.4. Класс StartupExtension

using AdmissionСampaign.Services.Services;

using Microsoft.Extensions.DependencyInjection;

using AdmissionСampaign.DataAccess;

namespace AdmissionСampaign.Services

{

public static class StartupExtension

{

public static void AddServices(this IServiceCollection services, string dbConnectionString)

{

services.AddDBContext(dbConnectionString);

services.AddRepositories();

services.AddTransient<EntrantService, EntrantService>();

}

}

}

## Проект AdmissionCampaign.ViewModels

### Класс AddEntrantViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class AddEntrantViewModel

{

public int EntryYear { get; set; }

public string EducationProgram { get; set; }

public int RegistrationNumber { get; set; }

public string FullName { get; set; }

public string SNILS { get; set; }

public string ApplicationType { get; set; }

public string WithoutExamsReason { get; set; }

public bool SpecialQuota { get; set; }

public bool TargetQuota { get; set; }

public int InformaticsUSE { get; set; }

public int MathUSE { get; set; }

public int RussianLanguageUSE { get; set; }

public int IndividualAchievementScore { get; set; }

public string EducationForm { get; set; }

public string PreemptiveRight { get; set; }

public bool IsDormitoryNeeded { get; set; }

public bool DocumentsReturned { get; set; }

public string Specialization { get; set; }

public string EducationCompetitions { get; set; }

public string RegistrationRegion { get; set; }

}

}

### 1.4.2. Класс EntrantView

namespace AdmissionCampaign.ViewModels.Entrants

{

public class EntrantView

{

public long Id { get; set; }

public int EntryYear { get; set; }

public string EducationProgram { get; set; }

public int RegistrationNumber { get; set; }

public string FullName { get; set; }

public string SNILS { get; set; }

public string ApplicationType { get; set; }

public string WithoutExamsReason { get; set; }

public bool SpecialQuota { get; set; }

public bool TargetQuota { get; set; }

public int InformaticsUSE { get; set; }

public int MathUSE { get; set; }

public int RussianLanguageUSE { get; set; }

public int IndividualAchievementScore { get; set; }

public int CompetitionPoints { get; set; }

public string EducationForm { get; set; }

public string PreemptiveRight { get; set; }

public bool IsDormitoryNeeded { get; set; }

public bool DocumentsReturned { get; set; }

public string Specialization { get; set; }

public string EducationCompetitions { get; set; }

public string RegistrationRegion { get; set; }

}

}

### 1.4.3. Класс FilterEntrantsViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class FilterEntrantsViewModel: PaginationViewModel

{

public int? EntryYear { get; set; }

public string EducationProgram { get; set; }

public string NameSearch { get; set; }

public string RegistrationRegion { get; set; }

public bool? IsDormitoryNeeded { get; set; }

public bool? HasTargetQuota { get; set; }

public bool? HasSpecialQuota { get; set; }

public bool? IsDocumentsReturned { get; set; }

public int? MinTotalUse { get; set; }

public int? MaxTotalUse { get; set; }

public int? MinRussianLanguageUSE { get; set; }

public int? MaxRussianLanguageUSE { get; set; }

public int? MinInformaticsUSE { get; set; }

public int? MaxInformaticsUSE { get; set; }

public int? MinMathUSE { get; set; }

public int? MaxMathUSE { get; set; }

}

}

### 1.4.4. Класс FiltredEntrantsView

using System.Collections.Generic;

namespace AdmissionCampaign.ViewModels.Entrants

{

public class FiltredEntrantsView

{

public List<EntrantView> Entrants { get; set; }

public int Total { get; set; }

}

}

### 1.4.5. Класс GetEntrantsByEntryYearViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class GetEntrantsByEntryYearViewModel: PaginationViewModel

{

public int EntryYear { get; set; }

}

}

### 1.4.6. Класс GetStaticticByRegionsView

using System.Collections.Generic;

namespace AdmissionCampaign.ViewModels.Entrants

{

public class GetStaticticByRegionsView

{

public List<RegionStaticticView> RegionStatictic { get; set; }

public int Total { get; set; }

}

}

### 1.4.7. Класс GetStaticticByRegionsViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class GetStaticticByRegionsViewModel : PaginationViewModel

{

public int? EntryYear { get; set; }

public string EducationProgram { get; set; }

}

}

### 1.4.8. Класс PaginationViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class PaginationViewModel

{

public int Page { get; set; } = 1;

public int ItemsPerPage { get; set; } = 15;

}

}

### 1.4.9. Класс RegionStaticticView

namespace AdmissionCampaign.ViewModels.Entrants

{

public class RegionStaticticView

{

public string Region { get; set; }

public int Count { get; set; }

}

}

### 1.4.10. Класс UpdateEntrantViewModel

namespace AdmissionCampaign.ViewModels.Entrants

{

public class UpdateEntrantViewModel : AddEntrantViewModel

{

public long Id { get; set; }

}

}

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
2. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |