**Приложение А. Текст программы**

АННОТАЦИЯ

В данном программном документе приведен программный код программы склада микроэлектроники

ТЕКСТ ПРОГРАММЫ

1.1. Наименование программы – «Склад микроэлектроники»

1.2. Область применения программы –программа применяется на заводах и предприятиях, производящих микроэлектронику.

1.3. Модули

Таблица 1 – Модули приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название модуля | Описание модуля | Размер модуля | Количество строк |
| 1 | ApiBuilder.cs | Класс подключения к API | 1 кб | 26 |
| 2 | ResponseMessageHandler.cs | Класс обработчика ответа от API | 1 кб | 26 |
| 3 | ComeJournalData.cs | Модель журнала | 2 кб | 19 |
| 4 | CounterpartyData.cs | Модель отпусков | 2 кб | 18 |
| 5 | DismissData.cs | Модель учета рабочего времени | 2 кб | 19 |
| 6 | HolidayData.cs | Модель отпусков | 2 кб | 19 |
| 7 | LaborData.cs | Модель документации | 2 кб | 19 |
| 8 | ManufData.cs | Модель производства | 2 кб | 21 |
| 9 | ProductData.cs | Модель продукции | 2 кб | 18 |
| 10 | ResponseMessageData.cs | Модель ответа от API | 2 кб | 16 |
| 11 | SupplyData.cs | Модель поддержки | 2 кб | 19 |
| 12 | UserData.cs | Модель пользователей | 2 кб | 24 |
| 13 | VisitorData.cs | Модель посетителей | 2 кб | 23 |
| 14 | AddCounterpartyWindow.xaml | Окно отпусков | 4 кб | 65 |
| 15 | AddCounterpartyWindow.xaml.cs | Код окна отпусков | 5 кб | 34 |
| 16 | AddEmployeeWindow.xaml | Окно добавления сотрудников | 3 кб | 23 |
| 17 | AddEmployeeWindow.xaml.cs | Код окна добавления сотрудников | 2 кб | 64 |
| 18 | AddHolidayWindow.xaml | Окно добавления отпуска | 6 кб | 17 |
| 19 | AddHolidayWindow.xaml.cs | Код окна добавления отпуска | 7 кб | 35 |
| 20 | AddProductWindow.xaml | Окно добавления продукта | 4 кб | 49 |
| 21 | AddProductWindow.xaml.cs | Код окна добавления продукта | 3 кб | 23 |
| 22 | AddSupplyWindow.xaml | Окно поддержки | 4 кб | 26 |
| 23 | AddSupplyWindow.xaml.cs | Код окна поддержки | 2 кб | 21 |
| 24 | AddTaskWindow.xaml | Окно добавления задачи | 2 кб | 84 |
| 25 | AddTaskWindow.xaml.cs | Код окна добавления задачи | 15 кб | 42 |
| 26 | AddVisitor.xaml | Окно добавления посетителя | 4 кб | 34 |
| 27 | AddVisitor.xaml.cs | Код окна добавления посетителя | 5 кб | 17 |
| 28 | App.xaml | Главное окно | 6 кб | 15 |
| 29 | App.xaml.cs | Код главного окна | 5 кб | 15 |
| 30 | BookkeepWindow.xaml | Окно хранилища документации | 8 кб | 16 |
| 31 | BookkeepWindow.xaml.cs | Код окна хранилища документации | 1 кб | 73 |
| 32 | ChangeStorageMethodWindow.xaml | Окно смены места хранения | 2 кб | 63 |
| 33 | ChangeStorageMethodWindow.xaml.cs | Код окна смены места хранения | 4 кб | 67 |
| 34 | HRmanagerWindow.xaml | Окно HR | 4 кб | 52 |
| 35 | HRmanagerWindow.xaml.cs | Код окна HR | 4 кб | 23 |
| 36 | LoginWindow.xaml | Окно авторизации | 4 кб | 53 |
| 37 | LoginWindow.xaml.cs | Код окна авторизации | 4 кб | 23 |
| 38 | ManufactureWindow.xaml | Окно производства | 4 кб | 65 |
| 39 | ManufactureWindow.xaml.cs | Код окна производства | 3 кб | 24 |
| 40 | StorekeepWindow.xaml | Окно типа хранения | 7 кб | 14 |
| 41 | StorekeepWindow.xaml.cs | Код окна типа хранения | 3 кб | 61 |
| 42 | SupplyMoreInfoWindow.xaml | Окно подробностей | 2 кб | 34 |
| 43 | SupplyMoreInfoWindow.xaml.cs | Код окна подробностей | 8 кб | 63 |
| 44 | SupplyWindow.xaml | Окно транзакций | 9 кб | 87 |
| 45 | SupplyWindow.xaml.cs | Код окна транзакций | 1 кб | 36 |
| 46 | VisitorWindow.xaml | Окно посетителей | 1 кб | 15 |
| 47 | VisitorWindow.xaml.cs | Код окна посетителей | 1 кб | 64 |

Таблица 2 – Модули API

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название модуля | Описание модуля | Размер модуля | Количество строк |
| 1 | ComeJournalController.cs | Контроллер транзакций | 1 кб | 76 |
| 2 | ManufController.cs | Контроллер производства | 2 кб | 34 |
| 3 | ProductController.cs | Контроллер продукции | 3 кб | 86 |
| 4 | SupplyController.cs | Контроллер поддержки | 6 кб | 56 |
| 5 | UserController.cs | Контроллер пользователей | 14 кб | 34 |
| 6 | VisitorController.cs | Контроллер посетителей | 3 кб | 97 |
| 7 | HashHandler.cs | Класс расчета хеш-суммы | 1 кб | 67 |
| 8 | Counterparty.cs | Модель отпусков | 1 кб | 77 |
| 9 | Dismissal.cs | Модель производственного цикла | 1 кб | 83 |
| 10 | Employee.cs | Модель сотрудника | 1 кб | 54 |
| 11 | EmployeeStatus.cs | Модель статуса работы сотрудника | 1 кб | 34 |
| 12 | Holiday.cs | Модель отпусков | 1 кб | 23 |
| 13 | Labor.cs | Модель документов | 1 кб | 23 |
| 14 | MicroElectronsDBContext.cs | Класс подключения к базе данных | 25 кб | 232 |
| 15 | OperationType.cs | Класс типа операции | 1 кб | 24 |
| 16 | Post.cs | Модель вставки значений | 1 кб | 25 |
| 17 | Product.cs | Класс работы с продуктом | 1 кб | 22 |
| 18 | ProductCategory.cs | Класс работы с категориями | 1 кб | 24 |
| 19 | StorageMethod.cs | Класс работы с категориями хранения | 1 кб | 25 |
| 20 | Supply.cs | Класс поддержки | 1 кб | 45 |
| 21 | SupplyCompo.cs | Класс компонентов | 1 кб | 73 |
| 22 | Task.cs | Класс задач | 1 кб | 79 |
| 23 | TaskStatus.cs | Класс статуса выполнения | 1 кб | 28 |
| 24 | User.cs | Класс пользователя | 1 кб | 76 |
| 25 | Visitor.cs | Класс посетителя | 1 кб | 43 |
| 26 | VisitorJournal.cs | Класс учета посетителей | 1 кб | 25 |
| 27 | Warehouse.cs | Класс склада | 1 кб | 29 |
| 28 | Program.cs | Основной класс | 1 кб | 10 |
| 29 | Startup.cs | Класс запуска приложения | 1 кб | 18 |

1.4. Код API

* + 1. Листинг файла Startup.cs

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.Extensions.Logging;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using Microsoft.EntityFrameworkCore;

namespace MicroElectronsApi

{

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)

{

string connection = Configuration.GetConnectionString("DefaultConnection");

services.AddDbContext<MicroElectronsDBContext>(s => s.UseMySql(connection, new MySqlServerVersion(new Version(5, 0, 39))));

services.AddControllers();

}

// 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.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

});

}

}

}

* + 1. Листинг файла Program.cs

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.Hosting;

using Microsoft.Extensions.Logging;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace MicroElectronsApi

{

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. Листинг файла ComeJournalController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class ComeJournalController : ControllerBase

{

private MicroElectronsDBContext \_context;

public ComeJournalController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Список записей прихода/расхода

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllList()

{

try

{

var result = (from comeJournal in \_context.ComeJournals

select new

{

JournalId = comeJournal.Id,

SubjectName = comeJournal.SubjectName,

Quantity = comeJournal.Quantity,

DateTimeConfirm = comeJournal.DateTimeConfirm.ToString("dd.MM.yyyy"),

IsCome = (comeJournal.IsCome.Value) ? "Приход" : "Расход",

Operation = comeJournal.Operation.Name

}).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла ManufController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class ManufController : ControllerBase

{

private MicroElectronsDBContext \_context;

public ManufController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Добавление задачи

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Add(ManufData manufData)

{

try

{

Models.Task task = new Models.Task()

{

Warehouse = \_context.Warehouses.Where(w => w.ProductId == manufData.ProductId).FirstOrDefault(),

Quantity = manufData.Quantity,

DateStart = DateTime.ParseExact(manufData.DateStart, "dd.MM.yyyy", null),

DateDeadline = DateTime.ParseExact(manufData.DateDeadline, "dd.MM.yyyy", null),

Employee = \_context.Employees.Where(e => e.Id == manufData.EmployeeId).FirstOrDefault(),

Status = \_context.TaskStatuses.Where(s => s.Name == "Выполняется").FirstOrDefault()

};

\_context.Tasks.Add(task);

\_context.SaveChanges();

var result = new

{

Id = task.Id,

DateStart = task.DateStart,

DateDeadline = task.DateDeadline

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список задач

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllList()

{

try

{

var result = (from task in \_context.Tasks

select new

{

TaskId = task.Id,

DateStart = task.DateStart.ToString("dd.MM.yyyy"),

DateDeadline = task.DateDeadline.ToString("dd.MM.yyyy"),

DateEnd = (task.DateEnd != null) ? task.DateEnd.Value.ToString("dd.MM.yyyy") : "",

Quantity = task.Quantity,

ProductId = task.Warehouse.ProductId,

ProductName = task.Warehouse.Product.Name,

EmployeeId = task.EmployeeId,

Status = task.Status.Name

}).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Окончание задачи

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult End(ManufData manufData)

{

try

{

Models.Task task = \_context.Tasks.Where(t => t.Id == manufData.TaskId).FirstOrDefault();

task.DateEnd = DateTime.ParseExact(manufData.DateEnd, "dd.MM.yyyy", null);

task.Status = (task.DateEnd.Value > task.DateDeadline)

? \_context.TaskStatuses.Where(s => s.Name == "Завершено с опозданием").FirstOrDefault()

: \_context.TaskStatuses.Where(s => s.Name == "Завершено в срок").FirstOrDefault();

\_context.Tasks.Update(task);

\_context.SaveChanges();

var result = task;

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла ProductController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class ProductController : ControllerBase

{

private MicroElectronsDBContext \_context;

public ProductController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Добавление продукта

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Add(ProductData productData)

{

try

{

Product product = new Product()

{

Name = productData.Name,

Category = \_context.ProductCategories.Where(c => c.Name == productData.CategoryName).FirstOrDefault()

};

\_context.Products.Add(product);

\_context.SaveChanges();

var result = new

{

Name = product.Name,

Category = product.Category

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Поиск продукта по наименованию

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult FindByName(ProductData productData)

{

try

{

var result = (from product in \_context.Products

where product.Name == productData.Name

select new

{

ProductId = product.Id,

Name = product.Name,

Category = product.Category.Name

}).FirstOrDefault();

if (result == null)

{

throw new Exception("Product was not finded.");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список наименований продуктов

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllNameList()

{

try

{

var result = (from product in \_context.Products

select new String(product.Name)).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список наименований категорий

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllCategoryList()

{

try

{

var result = (from category in \_context.ProductCategories

select new String(category.Name)).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список товаров на складе

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllStorageList()

{

try

{

var result = (from storage in \_context.Warehouses

select new

{

ProductId = storage.ProductId,

Name = storage.Product.Name,

CategoryName = storage.Product.Category.Name,

Quantity = storage.Quantity,

StorageMethod = storage.StorageMethod.Name

}).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список способов хранения

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllStorageMethodList()

{

try

{

var result = (from method in \_context.StorageMethods

select new String(method.Name)).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Смена способа хранения

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult ChangeStorageMethod(ProductData productData)

{

try

{

Warehouse warehouse = \_context.Warehouses.Where(w => w.ProductId == productData.ProductId).FirstOrDefault();

warehouse.StorageMethod = \_context.StorageMethods.Where(s => s.Name == productData.StorageMethod).FirstOrDefault();

\_context.Warehouses.Update(warehouse);

\_context.SaveChanges();

var result = new

{

StorageMethod = warehouse.StorageMethod.Name

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла SupplyController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class SupplyController : ControllerBase

{

private MicroElectronsDBContext \_context;

public SupplyController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Добавление поставки

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Add(SupplyData supplyData)

{

try

{

Supply supply = new Supply()

{

IsSell = supplyData.IsSell,

DateSupply = DateTime.Now,

Counterparty = \_context.Counterparties.Where(c => c.Name == supplyData.CounterpartyName).FirstOrDefault()

};

\_context.Supplies.Add(supply);

foreach (var i in supplyData.Products)

{

SupplyCompo supplyCompos = new SupplyCompo()

{

Product = \_context.Products.Where(p => p.Name == i.Name).FirstOrDefault(),

Supply = supply,

Quantity = i.Quantity,

Summa = i.Price

};

\_context.SupplyCompos.Add(supplyCompos);

}

\_context.SaveChanges();

var result = new

{

Date = supply.DateSupply,

Counterparty = supply.Counterparty.Name,

SellOrBuy = (supply.IsSell.Value) ? "Продажа" : "Покупка"

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список поставок

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllList()

{

try

{

var result = (from supply in \_context.Supplies

select new

{

SupplyId = supply.Id,

Counterparty = supply.Counterparty.Name,

Date = supply.DateSupply.ToString("dd.MM.yyyy"),

SellOrBuy = (supply.IsSell.Value) ? "Продажа" : "Покупка",

Summa = GetSummById(\_context.SupplyCompos.Where(s => s.SupplyId == supply.Id).ToList())

}).ToList();

if (result.Count == 0)

{

throw new Exception("Supplys not finded");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Возвращает общую сумму поставки по её id

/// </summary>

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

/// <returns></returns>

public static double GetSummById(List<SupplyCompo> supplyCompos)

{

double result = 0.0;

foreach (var i in supplyCompos)

{

result += i.Summa;

}

return result;

}

/// <summary>

/// Подробная информация о поставке по id

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult MoreInfoById(SupplyData supplyData)

{

try

{

var result = (from supply in \_context.Supplies

where supply.Id == supplyData.SupplyId

select new

{

SupplyId = supply.Id,

Counterparty = supply.Counterparty.Name,

Date = supply.DateSupply.ToString("dd.MM.yyyy"),

SellOrBuy = (supply.IsSell.Value) ? "Продажа" : "Покупка",

Summa = GetSummById(\_context.SupplyCompos.Where(s => s.SupplyId == supply.Id).ToList()),

Products = (from product in \_context.SupplyCompos

where product.SupplyId == supply.Id

select new ProductData()

{

ProductId = product.ProductId,

Name = product.Product.Name,

CategoryName = product.Product.Category.Name,

Quantity = product.Quantity,

Price = product.Summa

}).ToList()

}).FirstOrDefault();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла UserController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class UserController : ControllerBase

{

private MicroElectronsDBContext \_context;

public UserController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Список всех неуволенных сотрудников

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllList()

{

try

{

var result = (from user in \_context.Users

where user.Employee.Status.Name != "Уволен"

join labor in \_context.Labors on user.EmployeeId equals labor.EmployeeId

select new UserData()

{

EmployeeId = user.EmployeeId,

LastName = user.Employee.LastName,

FirstName = user.Employee.FirstName,

Patronymic = user.Employee.Patronymic,

Birthday = user.Employee.Birthday.ToString("dd.MM.yyyy"),

Status = user.Employee.Status.Name,

Salary = labor.Salary,

LaborDate = labor.DateConfirm.ToString("dd.MM.yyyy"),

Post = user.Employee.Post.Name,

Login = user.Login,

Password = user.Password

}).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Регистрация сотрудника

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Registry(UserData userData)

{

try

{

Employee employee = new Employee()

{

LastName = userData.LastName,

FirstName = userData.FirstName,

Patronymic = userData.Patronymic,

Birthday = DateTime.ParseExact(userData.Birthday, "dd.MM.yyyy", null),

Post = \_context.Posts.Where(p => p.Name == userData.Post).FirstOrDefault(),

Status = \_context.EmployeeStatuses.Where(s => s.Name == "Работает").FirstOrDefault()

};

\_context.Employees.Add(employee);

Labor labor = new Labor()

{

Salary = userData.Salary,

DateConfirm = DateTime.Now,

Employee = employee

};

\_context.Labors.Add(labor);

User user = new User()

{

Login = userData.Login,

Password = HashHandler.Sha256(userData.Password),

Employee = employee

};

\_context.Users.Add(user);

\_context.SaveChanges();

var result = (from userDb in \_context.Users

where userDb.Login == user.Login

select new

{

UserId = userDb.Id,

Login = userDb.Login,

Post = userDb.Employee.Post.Name

}).FirstOrDefault();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Авторизация сотрудника

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Auth(UserData userData)

{

try

{

var result = (from user in \_context.Users

where user.Login == userData.Login && user.Password == HashHandler.Sha256(userData.Password)

select new UserData()

{

EmployeeId = user.EmployeeId,

LastName = user.Employee.LastName,

FirstName = user.Employee.FirstName,

Patronymic = user.Employee.Patronymic,

Birthday = user.Employee.Birthday.ToString("dd.MM.yyyy"),

Status = user.Employee.Status.Name,

Post = user.Employee.Post.Name,

Login = user.Login,

Password = user.Password

}).FirstOrDefault();

if (result == null)

{

throw new Exception("Login and/or password invalid");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Увольнение сотрудника

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Dismissal(UserData userData)

{

try

{

Employee employee = \_context.Employees.Where(e => e.Id == userData.EmployeeId).FirstOrDefault();

if (employee == null)

{

throw new Exception("That employee not exist");

}

Dismissal dismissal = new Dismissal()

{

Employee = employee,

DateConfirm = DateTime.Now

};

\_context.Dismissals.Add(dismissal);

employee.Status = \_context.EmployeeStatuses.Where(s => s.Name == "Уволен").FirstOrDefault();

\_context.Employees.Update(employee);

\_context.SaveChanges();

var result = new

{

EmployeeId = employee.Id,

Status = employee.Status.Name

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список должностей

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult PostList()

{

try

{

var result = (from post in \_context.Posts

select new String(post.Name)).ToList();

if (result.Count == 0)

{

throw new Exception("Должности не найдены");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список трудовых договоров

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult LaborsList()

{

try

{

var result = (from labor in \_context.Labors

select new LaborData()

{

EmployeeId = labor.EmployeeId,

LastName = labor.Employee.LastName,

FirstName = labor.Employee.FirstName,

Patronymic = labor.Employee.Patronymic,

Post = labor.Employee.Post.Name,

Date = labor.DateConfirm.ToString("dd.MM.yyyy")

}).ToList();

if (result.Count == 0)

{

throw new Exception("Трудовые договора не найдены");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список заявлений на увольнение

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult DismissList()

{

try

{

var result = (from dismiss in \_context.Dismissals

select new DismissData()

{

EmployeeId = dismiss.EmployeeId,

LastName = dismiss.Employee.LastName,

FirstName = dismiss.Employee.FirstName,

Patronymic = dismiss.Employee.Patronymic,

Post = dismiss.Employee.Post.Name,

Date = dismiss.DateConfirm.ToString("dd.MM.yyyy")

}).ToList();

if (result.Count == 0)

{

throw new Exception("Заявления на увольнение не найдены");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Список заявлений на отпуск

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult HolidayList()

{

try

{

var result = (from holiday in \_context.Holidays

select new HolidayData()

{

EmployeeId = holiday.EmployeeId,

LastName = holiday.Employee.LastName,

FirstName = holiday.Employee.FirstName,

Patronymic = holiday.Employee.Patronymic,

Post = holiday.Employee.Post.Name,

DateStart = holiday.DateStart.ToString("dd.MM.yyyy"),

DateEnd = holiday.DateEnd.ToString("dd.MM.yyyy")

}).ToList();

if (result.Count == 0)

{

throw new Exception("Заявления на отпуск не найдены");

}

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Отпуск сотрудника

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult Holiday(HolidayData holidayData)

{

try

{

Employee employee = \_context.Employees.Where(e => e.Id == holidayData.EmployeeId).FirstOrDefault();

if (employee == null)

{

throw new Exception("That employee not exist");

}

Holiday holiday = new Holiday()

{

Employee = employee,

DateStart = DateTime.ParseExact(holidayData.DateStart, "dd.MM.yyyy", null),

DateEnd = DateTime.ParseExact(holidayData.DateEnd, "dd.MM.yyyy", null)

};

\_context.Holidays.Add(holiday);

employee.Status = \_context.EmployeeStatuses.Where(s => s.Name == "В отпуске").FirstOrDefault();

\_context.Employees.Update(employee);

\_context.SaveChanges();

var result = new

{

EmployeeId = employee.Id,

Status = employee.Status.Name

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла VisitorController.cs

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MicroElectronsApi.Models;

using MicroElectronsApi.Logics;

using MicroElectronsApi.Models.Data;

namespace MicroElectronsApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class VisitorController : ControllerBase

{

private MicroElectronsDBContext \_context;

public VisitorController(MicroElectronsDBContext context)

{

\_context = context;

}

/// <summary>

/// Запись в журнал посещений (вход)

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult WriteEntry(VisitorData visitorData)

{

try

{

Visitor visitor = new Visitor()

{

LastName = visitorData.VisitorLastName,

FirstName = visitorData.VisitorFirstName,

Patronymic = visitorData.VisitorPatronymic,

Passport = visitorData.Passport

};

\_context.Visitors.Add(visitor);

VisitorJournal visitorJournal = new VisitorJournal()

{

DateTimeEntry = DateTime.Now,

EmployeeEntry = \_context.Employees.Where(e => e.Id == visitorData.EmployeeEntryId).FirstOrDefault(),

Visitor = visitor

};

\_context.VisitorJournals.Add(visitorJournal);

\_context.SaveChanges();

var result = new

{

DateTime = visitorJournal.DateTimeEntry,

VisitorName = $"{visitorData.VisitorLastName} {visitorData.VisitorFirstName} {visitorData.VisitorPatronymic}",

EmployeeName = $"${visitorJournal.EmployeeEntry.LastName} {visitorJournal.EmployeeEntry.FirstName} {visitorJournal.EmployeeEntry.Patronymic}"

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Запись в журнал посещений (выход)

/// </summary>

[Route("[action]")]

[HttpPost]

public ActionResult WriteExit(VisitorData visitorData)

{

try

{

VisitorJournal visitorJournal = \_context.VisitorJournals.Where(j => j.Visitor.LastName == visitorData.VisitorLastName &&

j.Visitor.FirstName == visitorData.VisitorFirstName &&

j.Visitor.Patronymic == visitorData.VisitorPatronymic &&

j.DateTimeExit == null).FirstOrDefault();

visitorJournal.DateTimeExit = DateTime.Now;

visitorJournal.EmployeeExit = \_context.Employees.Where(e => e.Id == visitorData.EmployeeExitId).FirstOrDefault();

\_context.VisitorJournals.Update(visitorJournal);

\_context.SaveChanges();

var result = new

{

DateTime = visitorJournal.DateTimeExit,

VisitorName = $"{visitorData.VisitorLastName} {visitorData.VisitorFirstName} {visitorData.VisitorPatronymic}",

EmployeeName = $"${visitorJournal.EmployeeExit.LastName} {visitorJournal.EmployeeExit.FirstName} {visitorJournal.EmployeeExit.Patronymic}"

};

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

/// <summary>

/// Возвращает список посетителей

/// </summary>

[Route("[action]")]

[HttpGet]

public ActionResult AllList()

{

try

{

var result = (from visitor in \_context.VisitorJournals

select new VisitorData()

{

EmployeeEntryId = visitor.EmployeeEntryId,

EmployeeEntryName = $"{visitor.EmployeeEntry.LastName} {visitor.EmployeeEntry.FirstName} {visitor.EmployeeEntry.Patronymic}",

EmployeeExitId = visitor.EmployeeExitId,

EmployeeExitName = $"{visitor.EmployeeExit.LastName} {visitor.EmployeeExit.FirstName} {visitor.EmployeeExit.Patronymic}",

DateTimeEntry = visitor.DateTimeEntry.ToString("dd.MM.yyyy"),

DateTimeExit = visitor.DateTimeExit.Value.ToString("dd.MM.yyyy"),

VisitorLastName = visitor.Visitor.LastName,

VisitorFirstName = visitor.Visitor.FirstName,

VisitorPatronymic = visitor.Visitor.Patronymic,

Passport = visitor.Visitor.Passport

}).ToList();

return new ObjectResult(result);

}

catch (Exception ex)

{

return new ObjectResult(new { message = ex.Message }) { StatusCode = 501 };

}

}

}

}

* + 1. Листинг файла HashHandler.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsApi.Logics

{

public class HashHandler

{

public static string Sha256(string text)

{

var crypt = new SHA256Managed();

string hash = String.Empty;

byte[] crypto = crypt.ComputeHash(Encoding.UTF8.GetBytes(text));

foreach (byte theByte in crypto)

{

hash += theByte.ToString("x2");

}

return hash;

}

}

}

* + 1. Листинг файла ComeJournalData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ComeJournalData

{

public int JournalId { get; set; }

public string SubjectName { get; set; }

public int Quantity { get; set; }

public string DateTimeConfirm { get; set; }

public string IsCome { get; set; }

public string Operation { get; set; }

}

}

* + 1. Листинг файла CounterpartyData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class CounterpartyData

{

public int Id { get; set; }

public string Name { get; set; }

public string Tin { get; set; }

public string Address { get; set; }

public string Bic { get; set; }

}

}

* + 1. Листинг файла DismissData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class DismissData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string Date { get; set; }

}

}

* + 1. Листинг файла HolidayData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class HolidayData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string DateStart { get; set; }

public string DateEnd { get; set; }

}

}

* + 1. Листинг файла LaborData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class LaborData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string Date { get; set; }

}

}

* + 1. Листинг файла ManufData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ManufData

{

public int TaskId { get; set; }

public string DateStart { get; set; }

public string DateDeadline { get; set; }

public string DateEnd { get; set; }

public int Quantity { get; set; }

public int ProductId { get; set; }

public string ProductName { get; set; }

public int EmployeeId { get; set; }

public string Status { get; set; }

}

}

* + 1. Листинг файла ProductData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ProductData

{

public int ProductId { get; set; }

public string Name { get; set; }

public string CategoryName { get; set; }

public int Quantity { get; set; }

public int Price { get; set; }

public string StorageMethod { get; set; }

}

}

* + 1. Листинг файла ResponseMessageData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.Json.Serialization;

namespace MicroElectronsManager.Models

{

public class ResponseMessageData

{

[JsonPropertyName("message")]

public string Message { get; set; }

}

}

* + 1. Листинг файла SupplyData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class SupplyData

{

public int SupplyId { get; set; }

public string Counterparty { get; set; }

public string Date { get; set; }

public string SellOrBuy { get; set; }

public int Summa { get; set; }

public List<ProductData> Products { get; set; }

}

}

* + 1. Листинг файла UserData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class UserData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Birthday { get; set; }

public string Status { get; set; }

public string Post { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public double Salary { get; set; }

public string LaborDate { get; set; }

}

}

* + 1. Листинг файла VisitorData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class VisitorData

{

public int EmployeeEntryId { get; set; }

public string EmployeeEntryName { get; set; }

public int? EmployeeExitId { get; set; }

public string EmployeeExitName { get; set; }

public string DateTimeEntry { get; set; }

public string DateTimeExit { get; set; }

public string VisitorLastName { get; set; }

public string VisitorFirstName { get; set; }

public string VisitorPatronymic { get; set; }

public string Passport { get; set; }

}

}

* + 1. Листинг файла ComeJournal.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class ComeJournal

{

public int Id { get; set; }

public string SubjectName { get; set; }

public int Quantity { get; set; }

public DateTime DateTimeConfirm { get; set; }

public bool? IsCome { get; set; }

public int OperationId { get; set; }

public virtual OperationType Operation { get; set; }

}

}

* + 1. Листинг файла Counterparty.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Counterparty

{

public Counterparty()

{

Supplies = new HashSet<Supply>();

}

public int Id { get; set; }

public string Name { get; set; }

public string Tin { get; set; }

public string Address { get; set; }

public string Bic { get; set; }

public virtual ICollection<Supply> Supplies { get; set; }

}

}

* + 1. Листинг файла Dismissal.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Dismissal

{

public int Id { get; set; }

public DateTime DateConfirm { get; set; }

public int EmployeeId { get; set; }

public virtual Employee Employee { get; set; }

}

}

* + 1. Листинг файла Employee.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Employee

{

public Employee()

{

Dismissals = new HashSet<Dismissal>();

Holidays = new HashSet<Holiday>();

Labors = new HashSet<Labor>();

Tasks = new HashSet<Task>();

Users = new HashSet<User>();

VisitorJournalEmployeeEntries = new HashSet<VisitorJournal>();

VisitorJournalEmployeeExits = new HashSet<VisitorJournal>();

}

public int Id { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public DateTime Birthday { get; set; }

public int PostId { get; set; }

public int StatusId { get; set; }

public virtual Post Post { get; set; }

public virtual EmployeeStatus Status { get; set; }

public virtual ICollection<Dismissal> Dismissals { get; set; }

public virtual ICollection<Holiday> Holidays { get; set; }

public virtual ICollection<Labor> Labors { get; set; }

public virtual ICollection<Task> Tasks { get; set; }

public virtual ICollection<User> Users { get; set; }

public virtual ICollection<VisitorJournal> VisitorJournalEmployeeEntries { get; set; }

public virtual ICollection<VisitorJournal> VisitorJournalEmployeeExits { get; set; }

}

}

* + 1. Листинг файла EmployeeStatus.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class EmployeeStatus

{

public EmployeeStatus()

{

Employees = new HashSet<Employee>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Employee> Employees { get; set; }

}

}

* + 1. Листинг файла Holiday.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Holiday

{

public int Id { get; set; }

public DateTime DateStart { get; set; }

public DateTime DateEnd { get; set; }

public int EmployeeId { get; set; }

public virtual Employee Employee { get; set; }

}

}

* + 1. Листинг файла Labor.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Labor

{

public int Id { get; set; }

public double Salary { get; set; }

public DateTime DateConfirm { get; set; }

public int EmployeeId { get; set; }

public virtual Employee Employee { get; set; }

}

}

* + 1. Листинг файла MicroElectronsDBContext.cs

using System;

using Microsoft.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore.Metadata;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class MicroElectronsDBContext : DbContext

{

public MicroElectronsDBContext()

{

}

public MicroElectronsDBContext(DbContextOptions<MicroElectronsDBContext> options)

: base(options)

{

}

public virtual DbSet<ComeJournal> ComeJournals { get; set; }

public virtual DbSet<Counterparty> Counterparties { get; set; }

public virtual DbSet<Dismissal> Dismissals { get; set; }

public virtual DbSet<Employee> Employees { get; set; }

public virtual DbSet<EmployeeStatus> EmployeeStatuses { get; set; }

public virtual DbSet<Holiday> Holidays { get; set; }

public virtual DbSet<Labor> Labors { get; set; }

public virtual DbSet<OperationType> OperationTypes { get; set; }

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

public virtual DbSet<Product> Products { get; set; }

public virtual DbSet<ProductCategory> ProductCategories { get; set; }

public virtual DbSet<StorageMethod> StorageMethods { get; set; }

public virtual DbSet<Supply> Supplies { get; set; }

public virtual DbSet<SupplyCompo> SupplyCompos { get; set; }

public virtual DbSet<Task> Tasks { get; set; }

public virtual DbSet<TaskStatus> TaskStatuses { get; set; }

public virtual DbSet<User> Users { get; set; }

public virtual DbSet<Visitor> Visitors { get; set; }

public virtual DbSet<VisitorJournal> VisitorJournals { get; set; }

public virtual DbSet<Warehouse> Warehouses { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

optionsBuilder.UseMySql("Server=MYSQL5039.site4now.net;Database=db\_a7d9ce\_jedof50;Uid=a7d9ce\_jedof50;Pwd=asdqwe123", Microsoft.EntityFrameworkCore.ServerVersion.Parse("5.0.39-mysql"));

}

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.HasCharSet("latin1")

.UseCollation("latin1\_swedish\_ci");

modelBuilder.Entity<ComeJournal>(entity =>

{

entity.ToTable("ComeJournal");

entity.HasIndex(e => e.OperationId, "operationId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateTimeConfirm)

.HasColumnType("datetime")

.HasColumnName("dateTimeConfirm");

entity.Property(e => e.IsCome)

.HasColumnName("isCome")

.HasDefaultValueSql("'1'");

entity.Property(e => e.OperationId)

.HasColumnType("int(11)")

.HasColumnName("operationId");

entity.Property(e => e.Quantity)

.HasColumnType("int(11)")

.HasColumnName("quantity");

entity.Property(e => e.SubjectName)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("subjectName")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.HasOne(d => d.Operation)

.WithMany(p => p.ComeJournals)

.HasForeignKey(d => d.OperationId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("ComeJournal\_ibfk\_1");

});

modelBuilder.Entity<Counterparty>(entity =>

{

entity.ToTable("Counterparty");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Address)

.IsRequired()

.HasMaxLength(200)

.HasColumnName("address")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.Bic)

.HasMaxLength(9)

.HasColumnName("bic");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.Tin)

.IsRequired()

.HasMaxLength(10)

.HasColumnName("tin");

});

modelBuilder.Entity<Dismissal>(entity =>

{

entity.ToTable("Dismissal");

entity.HasIndex(e => e.EmployeeId, "employeeId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateConfirm)

.HasColumnType("date")

.HasColumnName("dateConfirm");

entity.Property(e => e.EmployeeId)

.HasColumnType("int(11)")

.HasColumnName("employeeId");

entity.HasOne(d => d.Employee)

.WithMany(p => p.Dismissals)

.HasForeignKey(d => d.EmployeeId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Dismissal\_ibfk\_1");

});

modelBuilder.Entity<Employee>(entity =>

{

entity.ToTable("Employee");

entity.HasIndex(e => e.PostId, "postId");

entity.HasIndex(e => e.StatusId, "statusId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Birthday)

.HasColumnType("date")

.HasColumnName("birthday");

entity.Property(e => e.FirstName)

.IsRequired()

.HasMaxLength(50)

.HasColumnName("firstName")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.LastName)

.IsRequired()

.HasMaxLength(50)

.HasColumnName("lastName")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.Patronymic)

.HasMaxLength(50)

.HasColumnName("patronymic")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.PostId)

.HasColumnType("int(11)")

.HasColumnName("postId");

entity.Property(e => e.StatusId)

.HasColumnType("int(11)")

.HasColumnName("statusId");

entity.HasOne(d => d.Post)

.WithMany(p => p.Employees)

.HasForeignKey(d => d.PostId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Employee\_ibfk\_1");

entity.HasOne(d => d.Status)

.WithMany(p => p.Employees)

.HasForeignKey(d => d.StatusId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Employee\_ibfk\_2");

});

modelBuilder.Entity<EmployeeStatus>(entity =>

{

entity.ToTable("EmployeeStatus");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<Holiday>(entity =>

{

entity.ToTable("Holiday");

entity.HasIndex(e => e.EmployeeId, "employeeId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateEnd)

.HasColumnType("date")

.HasColumnName("dateEnd");

entity.Property(e => e.DateStart)

.HasColumnType("date")

.HasColumnName("dateStart");

entity.Property(e => e.EmployeeId)

.HasColumnType("int(11)")

.HasColumnName("employeeId");

entity.HasOne(d => d.Employee)

.WithMany(p => p.Holidays)

.HasForeignKey(d => d.EmployeeId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Holiday\_ibfk\_1");

});

modelBuilder.Entity<Labor>(entity =>

{

entity.ToTable("Labor");

entity.HasIndex(e => e.EmployeeId, "employeeId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateConfirm)

.HasColumnType("date")

.HasColumnName("dateConfirm");

entity.Property(e => e.EmployeeId)

.HasColumnType("int(11)")

.HasColumnName("employeeId");

entity.Property(e => e.Salary).HasColumnName("salary");

entity.HasOne(d => d.Employee)

.WithMany(p => p.Labors)

.HasForeignKey(d => d.EmployeeId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Labor\_ibfk\_1");

});

modelBuilder.Entity<OperationType>(entity =>

{

entity.ToTable("OperationType");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<Post>(entity =>

{

entity.ToTable("Post");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<Product>(entity =>

{

entity.ToTable("Product");

entity.HasIndex(e => e.CategoryId, "categoryId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.CategoryId)

.HasColumnType("int(11)")

.HasColumnName("categoryId");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.HasOne(d => d.Category)

.WithMany(p => p.Products)

.HasForeignKey(d => d.CategoryId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Product\_ibfk\_1");

});

modelBuilder.Entity<ProductCategory>(entity =>

{

entity.ToTable("ProductCategory");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<StorageMethod>(entity =>

{

entity.ToTable("StorageMethod");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<Supply>(entity =>

{

entity.ToTable("Supply");

entity.HasIndex(e => e.CounterpartyId, "counterpartyId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.CounterpartyId)

.HasColumnType("int(11)")

.HasColumnName("counterpartyId");

entity.Property(e => e.DateSupply)

.HasColumnType("date")

.HasColumnName("dateSupply");

entity.Property(e => e.IsSell)

.HasColumnName("isSell")

.HasDefaultValueSql("'0'");

entity.HasOne(d => d.Counterparty)

.WithMany(p => p.Supplies)

.HasForeignKey(d => d.CounterpartyId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Supply\_ibfk\_1");

});

modelBuilder.Entity<SupplyCompo>(entity =>

{

entity.HasIndex(e => e.ProductId, "productId");

entity.HasIndex(e => e.SupplyId, "supplyId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.ProductId)

.HasColumnType("int(11)")

.HasColumnName("productId");

entity.Property(e => e.Quantity)

.HasColumnType("int(11)")

.HasColumnName("quantity");

entity.Property(e => e.Summa).HasColumnName("summa");

entity.Property(e => e.SupplyId)

.HasColumnType("int(11)")

.HasColumnName("supplyId");

entity.HasOne(d => d.Product)

.WithMany(p => p.SupplyCompos)

.HasForeignKey(d => d.ProductId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("SupplyCompos\_ibfk\_1");

entity.HasOne(d => d.Supply)

.WithMany(p => p.SupplyCompos)

.HasForeignKey(d => d.SupplyId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("SupplyCompos\_ibfk\_2");

});

modelBuilder.Entity<Task>(entity =>

{

entity.ToTable("Task");

entity.HasIndex(e => e.EmployeeId, "employeeId");

entity.HasIndex(e => e.StatusId, "statusId");

entity.HasIndex(e => e.WarehouseId, "warehouseId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateDeadline)

.HasColumnType("date")

.HasColumnName("dateDeadline");

entity.Property(e => e.DateEnd)

.HasColumnType("date")

.HasColumnName("dateEnd");

entity.Property(e => e.DateStart)

.HasColumnType("date")

.HasColumnName("dateStart");

entity.Property(e => e.EmployeeId)

.HasColumnType("int(11)")

.HasColumnName("employeeId");

entity.Property(e => e.Quantity)

.HasColumnType("int(11)")

.HasColumnName("quantity");

entity.Property(e => e.StatusId)

.HasColumnType("int(11)")

.HasColumnName("statusId");

entity.Property(e => e.WarehouseId)

.HasColumnType("int(11)")

.HasColumnName("warehouseId");

entity.HasOne(d => d.Employee)

.WithMany(p => p.Tasks)

.HasForeignKey(d => d.EmployeeId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Task\_ibfk\_2");

entity.HasOne(d => d.Status)

.WithMany(p => p.Tasks)

.HasForeignKey(d => d.StatusId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Task\_ibfk\_3");

entity.HasOne(d => d.Warehouse)

.WithMany(p => p.Tasks)

.HasForeignKey(d => d.WarehouseId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Task\_ibfk\_1");

});

modelBuilder.Entity<TaskStatus>(entity =>

{

entity.ToTable("TaskStatus");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(100)

.HasColumnName("name")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<User>(entity =>

{

entity.ToTable("User");

entity.HasIndex(e => e.EmployeeId, "employeeId");

entity.HasIndex(e => e.Login, "login")

.IsUnique();

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.EmployeeId)

.HasColumnType("int(11)")

.HasColumnName("employeeId");

entity.Property(e => e.Login)

.IsRequired()

.HasMaxLength(50)

.HasColumnName("login");

entity.Property(e => e.Password)

.IsRequired()

.HasMaxLength(64)

.HasColumnName("password");

entity.HasOne(d => d.Employee)

.WithMany(p => p.Users)

.HasForeignKey(d => d.EmployeeId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("User\_ibfk\_1");

});

modelBuilder.Entity<Visitor>(entity =>

{

entity.ToTable("Visitor");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.FirstName)

.IsRequired()

.HasMaxLength(50)

.HasColumnName("firstName")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.LastName)

.IsRequired()

.HasMaxLength(50)

.HasColumnName("lastName")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

entity.Property(e => e.Passport)

.IsRequired()

.HasMaxLength(10)

.HasColumnName("passport");

entity.Property(e => e.Patronymic)

.HasMaxLength(50)

.HasColumnName("patronymic")

.UseCollation("utf8\_general\_ci")

.HasCharSet("utf8");

});

modelBuilder.Entity<VisitorJournal>(entity =>

{

entity.ToTable("VisitorJournal");

entity.HasIndex(e => e.EmployeeEntryId, "employeeEntryId");

entity.HasIndex(e => e.EmployeeExitId, "employeeExitId");

entity.HasIndex(e => e.VisitorId, "visitorId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.DateTimeEntry)

.HasColumnType("datetime")

.HasColumnName("dateTimeEntry");

entity.Property(e => e.DateTimeExit)

.HasColumnType("datetime")

.HasColumnName("dateTimeExit");

entity.Property(e => e.EmployeeEntryId)

.HasColumnType("int(11)")

.HasColumnName("employeeEntryId");

entity.Property(e => e.EmployeeExitId)

.HasColumnType("int(11)")

.HasColumnName("employeeExitId");

entity.Property(e => e.VisitorId)

.HasColumnType("int(11)")

.HasColumnName("visitorId");

entity.HasOne(d => d.EmployeeEntry)

.WithMany(p => p.VisitorJournalEmployeeEntries)

.HasForeignKey(d => d.EmployeeEntryId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("VisitorJournal\_ibfk\_1");

entity.HasOne(d => d.EmployeeExit)

.WithMany(p => p.VisitorJournalEmployeeExits)

.HasForeignKey(d => d.EmployeeExitId)

.HasConstraintName("VisitorJournal\_ibfk\_2");

entity.HasOne(d => d.Visitor)

.WithMany(p => p.VisitorJournals)

.HasForeignKey(d => d.VisitorId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("VisitorJournal\_ibfk\_3");

});

modelBuilder.Entity<Warehouse>(entity =>

{

entity.ToTable("Warehouse");

entity.HasIndex(e => e.ProductId, "productId");

entity.HasIndex(e => e.StorageMethodId, "storageMethodId");

entity.Property(e => e.Id)

.HasColumnType("int(11)")

.HasColumnName("id");

entity.Property(e => e.ProductId)

.HasColumnType("int(11)")

.HasColumnName("productId");

entity.Property(e => e.Quantity)

.HasColumnType("int(11)")

.HasColumnName("quantity")

.HasDefaultValueSql("'0'");

entity.Property(e => e.StorageMethodId)

.HasColumnType("int(11)")

.HasColumnName("storageMethodId");

entity.HasOne(d => d.Product)

.WithMany(p => p.Warehouses)

.HasForeignKey(d => d.ProductId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Warehouse\_ibfk\_1");

entity.HasOne(d => d.StorageMethod)

.WithMany(p => p.Warehouses)

.HasForeignKey(d => d.StorageMethodId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("Warehouse\_ibfk\_2");

});

OnModelCreatingPartial(modelBuilder);

}

partial void OnModelCreatingPartial(ModelBuilder modelBuilder);

}

}

* + 1. Листинг файла OperationType.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class OperationType

{

public OperationType()

{

ComeJournals = new HashSet<ComeJournal>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<ComeJournal> ComeJournals { get; set; }

}

}

* + 1. Листинг файла Post.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Post

{

public Post()

{

Employees = new HashSet<Employee>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Employee> Employees { get; set; }

}

}

* + 1. Листинг файла Product.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Product

{

public Product()

{

SupplyCompos = new HashSet<SupplyCompo>();

Warehouses = new HashSet<Warehouse>();

}

public int Id { get; set; }

public string Name { get; set; }

public int CategoryId { get; set; }

public virtual ProductCategory Category { get; set; }

public virtual ICollection<SupplyCompo> SupplyCompos { get; set; }

public virtual ICollection<Warehouse> Warehouses { get; set; }

}

}

* + 1. Листинг файла ProductCategory.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class ProductCategory

{

public ProductCategory()

{

Products = new HashSet<Product>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Product> Products { get; set; }

}

}

* + 1. Листинг файла StorageMethod.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class StorageMethod

{

public StorageMethod()

{

Warehouses = new HashSet<Warehouse>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Warehouse> Warehouses { get; set; }

}

}

* + 1. Листинг файла Supply.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Supply

{

public Supply()

{

SupplyCompos = new HashSet<SupplyCompo>();

}

public int Id { get; set; }

public bool? IsSell { get; set; }

public DateTime DateSupply { get; set; }

public int CounterpartyId { get; set; }

public virtual Counterparty Counterparty { get; set; }

public virtual ICollection<SupplyCompo> SupplyCompos { get; set; }

}

}

* + 1. Листинг файла SupplyCompo.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class SupplyCompo

{

public int Id { get; set; }

public int Quantity { get; set; }

public double Summa { get; set; }

public int ProductId { get; set; }

public int SupplyId { get; set; }

public virtual Product Product { get; set; }

public virtual Supply Supply { get; set; }

}

}

* + 1. Листинг файла Task.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Task

{

public int Id { get; set; }

public DateTime DateStart { get; set; }

public DateTime? DateEnd { get; set; }

public DateTime DateDeadline { get; set; }

public int Quantity { get; set; }

public int WarehouseId { get; set; }

public int EmployeeId { get; set; }

public int StatusId { get; set; }

public virtual Employee Employee { get; set; }

public virtual TaskStatus Status { get; set; }

public virtual Warehouse Warehouse { get; set; }

}

}

* + 1. Листинг файла TaskStatus.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class TaskStatus

{

public TaskStatus()

{

Tasks = new HashSet<Task>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Task> Tasks { get; set; }

}

}

* + 1. Листинг файла User.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class User

{

public int Id { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public int EmployeeId { get; set; }

public virtual Employee Employee { get; set; }

}

}

* + 1. Листинг файла Visitor.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Visitor

{

public Visitor()

{

VisitorJournals = new HashSet<VisitorJournal>();

}

public int Id { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Passport { get; set; }

public virtual ICollection<VisitorJournal> VisitorJournals { get; set; }

}

}

* + 1. Листинг файла VisitorJournal.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class VisitorJournal

{

public int Id { get; set; }

public DateTime DateTimeEntry { get; set; }

public DateTime? DateTimeExit { get; set; }

public int EmployeeEntryId { get; set; }

public int? EmployeeExitId { get; set; }

public int VisitorId { get; set; }

public virtual Employee EmployeeEntry { get; set; }

public virtual Employee EmployeeExit { get; set; }

public virtual Visitor Visitor { get; set; }

}

}

* + 1. Листинг файла Warehouse.cs

using System;

using System.Collections.Generic;

#nullable disable

namespace MicroElectronsApi.Models

{

public partial class Warehouse

{

public Warehouse()

{

Tasks = new HashSet<Task>();

}

public int Id { get; set; }

public int? Quantity { get; set; }

public int ProductId { get; set; }

public int StorageMethodId { get; set; }

public virtual Product Product { get; set; }

public virtual StorageMethod StorageMethod { get; set; }

public virtual ICollection<Task> Tasks { get; set; }

}

}

1.5. Код программы

* + 1. Листинг файла ApiBuilder.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using RestSharp;

namespace MicroElectronsManager.Logics

{

class ApiBuilder

{

private static string rootUrl = "http://jedof50245-001-site1.ctempurl.com/api";

private static RestClient restClient;

public static RestClient GetInstance()

{

if (restClient == null)

{

restClient = new RestClient(rootUrl);

}

return restClient;

}

}

}

* + 1. Листинг файла ResponseMessageHandler.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.Json;

using System.Threading.Tasks;

using MicroElectronsManager.Models;

namespace MicroElectronsManager.Logics

{

public static class ResponseMessageHandler

{

public static string GetMessage(string jsonMessage)

{

try

{

ResponseMessageData responseMessage = JsonSerializer.Deserialize<ResponseMessageData>(jsonMessage);

return responseMessage.Message;

}

catch

{

return "Not Found 404";

}

}

}

}

* + 1. Листинг файла ComeJournalData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ComeJournalData

{

public int JournalId { get; set; }

public string SubjectName { get; set; }

public int Quantity { get; set; }

public string DateTimeConfirm { get; set; }

public string IsCome { get; set; }

public string Operation { get; set; }

}

}

* + 1. Листинг файла CounterpartyData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class CounterpartyData

{

public int Id { get; set; }

public string Name { get; set; }

public string Tin { get; set; }

public string Address { get; set; }

public string Bic { get; set; }

}

}

* + 1. Листинг файла DismissData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class DismissData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string Date { get; set; }

}

}

* + 1. Листинг файла HolidayData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class HolidayData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string DateStart { get; set; }

public string DateEnd { get; set; }

}

}

* + 1. Листинг файла LaborData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class LaborData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Post { get; set; }

public string Date { get; set; }

}

}

* + 1. Листинг файла ManufData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ManufData

{

public int TaskId { get; set; }

public string DateStart { get; set; }

public string DateDeadline { get; set; }

public string DateEnd { get; set; }

public int Quantity { get; set; }

public int ProductId { get; set; }

public string ProductName { get; set; }

public int EmployeeId { get; set; }

public string Status { get; set; }

}

}

* + 1. Листинг файла ProductData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class ProductData

{

public int ProductId { get; set; }

public string Name { get; set; }

public string CategoryName { get; set; }

public int Quantity { get; set; }

public int Price { get; set; }

public string StorageMethod { get; set; }

}

}

* + 1. Листинг файла ResponseMessageData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.Json.Serialization;

namespace MicroElectronsManager.Models

{

public class ResponseMessageData

{

[JsonPropertyName("message")]

public string Message { get; set; }

}

}

* + 1. Листинг файла SupplyData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class SupplyData

{

public int SupplyId { get; set; }

public string Counterparty { get; set; }

public string Date { get; set; }

public string SellOrBuy { get; set; }

public int Summa { get; set; }

public List<ProductData> Products { get; set; }

}

}

* + 1. Листинг файла UserData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class UserData

{

public int EmployeeId { get; set; }

public string LastName { get; set; }

public string FirstName { get; set; }

public string Patronymic { get; set; }

public string Birthday { get; set; }

public string Status { get; set; }

public string Post { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public double Salary { get; set; }

public string LaborDate { get; set; }

}

}

* + 1. Листинг файла VisitorData.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MicroElectronsManager.Models

{

public class VisitorData

{

public int EmployeeEntryId { get; set; }

public string EmployeeEntryName { get; set; }

public int? EmployeeExitId { get; set; }

public string EmployeeExitName { get; set; }

public string DateTimeEntry { get; set; }

public string DateTimeExit { get; set; }

public string VisitorLastName { get; set; }

public string VisitorFirstName { get; set; }

public string VisitorPatronymic { get; set; }

public string Passport { get; set; }

}

}

* + 1. Листинг файла AddCounterpartyWindow.xaml

<Window x:Class="MicroElectronsManager.AddCounterpartyWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Добавление контрагента" Width="800" MinWidth="800" Height="450" MinHeight="450" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Наименование" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="ИНН" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbTin" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Адрес" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbAddress" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="3" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="БИК" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbBic" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="4" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddCounterpartyWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddCounterpartyWindow.xaml

/// </summary>

public partial class AddCounterpartyWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public AddCounterpartyWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("counterparty/Add")

.AddJsonBody(new

{

Name = TbName.Text.ToString(),

Tin = TbTin.Text.ToString(),

Address = TbAddress.Text.ToString(),

Bic = TbBic.Text.ToString()

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

(this.Owner as AddSupplyWindow).WriteListCounterparty();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void ValidForm()

{

if (TbAddress.Text.Trim() == "") { throw new Exception("Введите адрес"); }

if (TbName.Text.Trim() == "") { throw new Exception("Введите наименование"); }

if (TbBic.Text.Trim() == "") { throw new Exception("Введите бик"); }

if (TbTin.Text.Trim() == "") { throw new Exception("Введите инн"); }

}

}

}

* + 1. Листинг файла AddEmployeeWindow.xaml

<Window x:Class="MicroElectronsManager.AddEmployeeWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Регистрация сотрудника" Width="800" MinWidth="800" Height="600" MinHeight="600" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Фамилия" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbLastName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Имя" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbFirstName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Отчество" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbPatronymic" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="3" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Дата рождения" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<DatePicker x:Name="DpBirthday" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="4" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Должность" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<ComboBox x:Name="CbPosts" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="5" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Логин" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbLogin" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="6" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Пароль" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<PasswordBox x:Name="TbPassword" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="7" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Зарплата" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbSalary" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="8" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddEmployeeWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddEmployeeWindow.xaml

/// </summary>

public partial class AddEmployeeWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public AddEmployeeWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

try

{

var response = apiClient.Get<List<String>>(new RestRequest("user/postlist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbPosts.ItemsSource = response.Data;

DpBirthday.DisplayDateEnd = DateTime.Now;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("user/registry")

.AddJsonBody(new

{

LastName = TbLastName.Text.ToString(),

FirstName = TbFirstName.Text.ToString(),

Patronymic = TbPatronymic.Text.ToString(),

Birthday = DpBirthday.SelectedDate.Value.ToString("dd.MM.yyyy"),

Post = CbPosts.SelectedItem.ToString(),

Login = TbLogin.Text.ToString(),

Password = TbPassword.Password.ToString(),

Salary = TbSalary.Text.ToString()

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

MessageBox.Show("Сотрудник зарегистрирован", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

(this.Owner as HRmanagerWindow).GridEmployeeWrite();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void ValidForm()

{

if (TbLastName.Text.Trim() == "") { throw new Exception("Введите фамилию"); }

if (TbFirstName.Text.Trim() == "") { throw new Exception("Введите имя"); }

if (TbPatronymic.Text.Trim() == "") { throw new Exception("Введите отчество"); }

if (DpBirthday.SelectedDate == null) { throw new Exception("Введите дату рождения"); }

if (CbPosts.SelectedItem == null) { throw new Exception("Выберите должность"); }

if (TbLogin.Text.Trim() == "") { throw new Exception("Введите логин"); }

if (TbPassword.Password.Trim() == "") { throw new Exception("Введите пароль"); }

if (TbSalary.Text.Trim() == "") { throw new Exception("Введите зарплату"); }

int a = 0;

if (!int.TryParse(TbSalary.Text.Trim(), out a))

{

throw new Exception("Некорректный ввод зарплаты");

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

}

}

* + 1. Листинг файла AddHolidayWindow.xaml

<Window x:Class="MicroElectronsManager.AddHolidayWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Отпуск сотрудника" Width="800" MinWidth="800" Height="400" MinHeight="400" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock x:Name="TbEmployeeName" Text="Иванов Иван Иванович" Grid.Row="0" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"

Margin="0 0 0 20" Foreground="#FF212529"/>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 10">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Дата начала отпуска" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<DatePicker x:Name="DpDateStart" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 0 0 20">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Дата конца отпуска" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<DatePicker x:Name="DpDateEnd" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="3" VerticalAlignment="Center">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddHolidayWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddHolidayWindow.xaml

/// </summary>

public partial class AddHolidayWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public AddHolidayWindow()

{

InitializeComponent();

}

private void ValidForm()

{

if (DpDateStart.SelectedDate == null || DpDateEnd.SelectedDate == null)

{

throw new Exception("Укажите даты начала и конца отпуска.");

}

if (DpDateStart.SelectedDate.Value > DpDateEnd.SelectedDate.Value)

{

throw new Exception("Дата конца не может наступить быстрее даты начала.");

}

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("user/holiday")

.AddJsonBody(new { EmployeeId = user.EmployeeId,

DateStart = DpDateStart.SelectedDate.Value.ToString("dd.MM.yyyy"),

DateEnd = DpDateEnd.SelectedDate.Value.ToString("dd.MM.yyyy")

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

MessageBox.Show("Сотрудник отправлен в отпуск", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

(this.Owner as HRmanagerWindow).GridHolidayWrite();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbEmployeeName.Text = $"{user.LastName} {user.FirstName} {user.Patronymic}";

DpDateStart.DisplayDateStart = DateTime.Now;

DpDateEnd.DisplayDateStart = DateTime.Now;

}

}

}

* + 1. Листинг файла AddProductWindow.xaml

<Window x:Class="MicroElectronsManager.AddProductWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Добавление товаров" Width="800" MinWidth="800" Height="450" MinHeight="450" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Наименование" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Категория" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<ComboBox x:Name="CbCategory" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddProductWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddProductWindow.xaml

/// </summary>

public partial class AddProductWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public AddProductWindow()

{

InitializeComponent();

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("product/add")

.AddJsonBody(new

{

Name = TbName.Text.ToString(),

CategoryName = CbCategory.SelectedItem.ToString()

}));

(this.Owner as AddSupplyWindow).WriteListProducts();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

try

{

var response = apiClient.Get<List<string>>(new RestRequest("product/allcategorylist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbCategory.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void ValidForm()

{

if (TbName.Text.Trim() == "") { throw new Exception("Введите наименвоание"); }

if (CbCategory.SelectedItem == null) { throw new Exception("Выберите категорию"); }

}

}

}

* + 1. Листинг файла AddSupplyWindow.xaml

<Window x:Class="MicroElectronsManager.AddSupplyWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Добавление поставки" Width="1000" MinWidth="1000" Height="450" MinHeight="450" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="400"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid Grid.Column="0" HorizontalAlignment="Center" VerticalAlignment="Center">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<RadioButton x:Name="RbIsSell" Content="Продажа" IsChecked="True" Grid.Row="0" FontSize="18"/>

<RadioButton x:Name="RbIsBuy" Content="Покупка" Grid.Row="1" FontSize="18"/>

</Grid>

<Grid Grid.Row="1" Margin="0 0 0 25">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<ComboBox x:Name="CbCounterparty" Grid.Column="0" Width="200" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="18" Margin="0 0 10 0"/>

<Button x:Name="BtnAddCounterparty" Content="+" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" Padding="20 2" FontSize="18"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnAddCounterparty\_Click"/>

</Grid>

<Grid Grid.Row="2" Margin="0 0 0 15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<ComboBox x:Name="CbProducts" Grid.Column="0" Width="200" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="18" Margin="0 0 10 0"/>

<Button x:Name="BtnAddProduct" Content="+" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" Padding="20 2" FontSize="18"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnAddProduct\_Click"/>

</Grid>

<Grid Grid.Row="3" Margin="0 0 0 15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="120"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<TextBlock Text="Количество:" FontSize="18" VerticalAlignment="Center"/>

<TextBox x:Name="TbQuantity" Grid.Column="1" Width="150" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="18" Margin="0 0 10 0"/>

</Grid>

<Grid Grid.Row="4" Margin="0 0 0 15">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="120"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<TextBlock Text="Сумма:" FontSize="18" VerticalAlignment="Center"/>

<TextBox x:Name="TbPrice" Grid.Column="1" Width="150" VerticalAlignment="Center" HorizontalAlignment="Center" FontSize="18" Margin="0 0 10 0"/>

</Grid>

<Button x:Name="BtnAddProductInSupply" Content="Добавить в корзину" Grid.Row="5" HorizontalAlignment="Center" VerticalAlignment="Center" Padding="20 2" FontSize="18" Margin="0 0 0 25"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnAddProductInSupply\_Click"/>

<Grid Grid.Row="6">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Подтвердить" Width="120" Grid.Column="0" FontSize="18" Padding="0 2"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" VerticalAlignment="Center" HorizontalAlignment="Center" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Width="120" Grid.Column="1" FontSize="18" Padding="0 2"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click" VerticalAlignment="Center" HorizontalAlignment="Center"/>

</Grid>

</Grid>

<DataGrid x:Name="DataGridProducts" Grid.Column="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Наименование" Width="\*" Binding="{Binding Path=Name}"/>

<DataGridTextColumn Header="Количество" Width="140" Binding="{Binding Path=Quantity}"/>

<DataGridTextColumn Header="Сумма" Width="140" Binding="{Binding Path=Price}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла AddSupplyWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddSupplyWindow.xaml

/// </summary>

public partial class AddSupplyWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

private List<ProductData> products = new List<ProductData>();

public AddSupplyWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

try

{

WriteListCounterparty();

WriteListProducts();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

public void WriteListCounterparty()

{

var response = apiClient.Get<List<string>>(new RestRequest("counterparty/AllNameList"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbCounterparty.ItemsSource = response.Data;

}

public void WriteListProducts()

{

var response = apiClient.Get<List<string>>(new RestRequest("product/AllNameList"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbProducts.ItemsSource = response.Data;

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void BtnAddProductInSupply\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidProductForm();

ProductData productData = new ProductData()

{

Name = CbProducts.SelectedItem.ToString(),

Quantity = Convert.ToInt32(TbQuantity.Text),

Price = Convert.ToInt32(TbPrice.Text)

};

products.Add(productData);

DataGridProductsWrite();

ClearProductForm();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void DataGridProductsWrite()

{

DataGridProducts.ClearValue(ItemsControl.ItemsSourceProperty);

DataGridProducts.ItemsSource = products;

}

private void ValidProductForm()

{

if (CbProducts.SelectedItem == null) { throw new Exception("Выберите товар"); }

if (TbQuantity.Text.Trim() == "") { throw new Exception("Введите количество товара"); }

if (TbPrice.Text.Trim() == "") { throw new Exception("Введите сумму товара"); }

}

private void ValidForm()

{

if (CbCounterparty.SelectedItem == null) { throw new Exception("Выберите контрагента"); }

if (products.Count == 0) { throw new Exception("В корзине поставки ничего нету"); }

}

private void ClearProductForm()

{

CbProducts.SelectedItem = null;

TbQuantity.Text = "";

TbPrice.Text = "";

}

private void BtnAddCounterparty\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddCounterpartyWindow() { Owner = this }.Show();

}

private void BtnAddProduct\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddProductWindow() { Owner = this }.Show();

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("supply/add")

.AddJsonBody(new

{

IsSell = RbIsSell.IsChecked,

CounterpartyName = CbCounterparty.SelectedItem.ToString(),

Products = products

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

(this.Owner as SupplyWindow).DataGridSupplyWrite();

MessageBox.Show("Поставка зарегистрирована", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

* + 1. Листинг файла AddTaskWindow.xaml

<Window x:Class="MicroElectronsManager.AddTaskWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="AddTaskWindow" Width="800" MinWidth="800" Height="450" MinHeight="450" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed" >

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Товар" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<ComboBox x:Name="CbProducts" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Дата начала" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<DatePicker x:Name="DpStart" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Дата дедлайна" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<DatePicker x:Name="DpDeadline" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="3" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Количество" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbQuantity" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="4" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddTaskWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddTaskWindow.xaml

/// </summary>

public partial class AddTaskWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public AddTaskWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

WriteListProducts();

DpStart.DisplayDateStart = DateTime.Now;

DpDeadline.DisplayDateStart = DateTime.Now;

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var product = apiClient.Post<ProductData>(new RestRequest("product/FindByName")

.AddJsonBody(new

{

Name = CbProducts.SelectedItem.ToString()

})).Data;

var response = apiClient.Post(new RestRequest("manuf/add")

.AddJsonBody(new

{

DateStart = DpStart.SelectedDate.Value.ToString("dd.MM.yyyy"),

DateDeadline = DpDeadline.SelectedDate.Value.ToString("dd.MM.yyyy"),

Quantity = TbQuantity.Text.ToString(),

ProductId = product.ProductId,

EmployeeId = (this.Owner as ManufactureWindow).user.EmployeeId

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

(this.Owner as ManufactureWindow).DataGridManufsWrite();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void ValidForm()

{

if (CbProducts.SelectedItem == null) { throw new Exception("Выберите товар"); }

if (TbQuantity.Text.Trim() == "") { throw new Exception("Введите количество"); }

if (DpStart.SelectedDate == null) { throw new Exception("Выберите дату начала производства"); }

if (DpDeadline.SelectedDate == null) { throw new Exception("Выберите дату дедлайна"); }

if (DpStart.SelectedDate.Value > DpDeadline.SelectedDate.Value) { throw new Exception("Некорректные даты"); }

}

private void WriteListProducts()

{

try

{

var response = apiClient.Get<List<string>>(new RestRequest("product/AllNameList"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbProducts.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

* + 1. Листинг файла AddVisitor.xaml

<Window x:Class="MicroElectronsManager.AddVisitor"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Добавить посетителя" Width="800" MinWidth="800" Height="400" MinHeight="400" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Фамилия" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbLastName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Имя" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbFirstName" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Отчество" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbPatronymic" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="3" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Паспорт" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<TextBox x:Name="TbPassport" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="4" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла AddVisitor.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для AddVisitor.xaml

/// </summary>

public partial class AddVisitor : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public UserData user;

public AddVisitor()

{

InitializeComponent();

}

private void ValidForm()

{

if (TbLastName.Text.Trim() == "") { throw new Exception("Введите фамилию"); }

if (TbFirstName.Text.Trim() == "") { throw new Exception("Введите имя"); }

if (TbPatronymic.Text.Trim() == "") { throw new Exception("Введите отчество"); }

if (TbPassport.Text.Trim() == "") { throw new Exception("Введите серию и номер паспорта (10 чисел)"); }

if (TbPassport.Text.Trim().Length != 10) { throw new Exception("Некорректный ввод серии и номера паспорта (должно быть 10 чисел)"); }

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("visitor/writeentry")

.AddJsonBody(new

{

EmployeeEntryId = user.EmployeeId,

VisitorLastName = TbLastName.Text.ToString(),

VisitorFirstName = TbFirstName.Text.ToString(),

VisitorPatronymic = TbPatronymic.Text.ToString(),

Passport = TbPassport.Text.ToString()

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

(this.Owner as VisitorWindow).DataGridVisitorWrite();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

}

}

* + 1. Листинг файла App.xaml

<Application x:Class="MicroElectronsManager.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:MicroElectronsManager"

StartupUri="LoginWindow.xaml">

<Application.Resources>

</Application.Resources>

</Application>

* + 1. Листинг файла App.xaml.cs

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Threading.Tasks;

using System.Windows;

namespace MicroElectronsManager

{

/// <summary>

/// Interaction logic for App.xaml

/// </summary>

public partial class App : Application

{

}

}

* + 1. Листинг файла BookkeepWindow.xaml

<Window x:Class="MicroElectronsManager.BookkeepWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Бухгалтерия" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<DataGrid x:Name="DataGridComeJournal" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmExport" Header="Экспорт в эксель" Click="CmExport\_Click"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Предмет" Width="\*" Binding="{Binding Path=SubjectName}"/>

<DataGridTextColumn Header="Количество" Width="170" Binding="{Binding Path=Quantity}"/>

<DataGridTextColumn Header="Дата" Width="170" Binding="{Binding Path=DateTimeConfirm}"/>

<DataGridTextColumn Header="Приход/расход" Width="170" Binding="{Binding Path=IsCome}"/>

<DataGridTextColumn Header="Тип операции" Width="170" Binding="{Binding Path=Operation}"/>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=JournalId}" Visibility="Collapsed"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла BookkeepWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

using Excel = Microsoft.Office.Interop.Excel;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для BookkeepWindow.xaml

/// </summary>

public partial class BookkeepWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public BookkeepWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

DataGridComeJournalWrite();

}

public void DataGridComeJournalWrite()

{

try

{

DataGridComeJournal.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<ComeJournalData>>(new RestRequest("comejournal/alllist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridComeJournal.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void CmExport\_Click(object sender, RoutedEventArgs e)

{

Excel.Application ex = new Microsoft.Office.Interop.Excel.Application();

ex.Visible = true;

ex.SheetsInNewWorkbook = 1;

Excel.Workbook workBook = ex.Workbooks.Add(Type.Missing);

ex.DisplayAlerts = false;

Excel.Worksheet sheet = (Excel.Worksheet)ex.Worksheets.get\_Item(1);

sheet.Name = $"Отчет за {DateTime.Now.ToString("dd.MM.yyyy")}";

sheet.Cells[1, 1] = "Предмет";

sheet.Cells[1, 2] = "Количество";

sheet.Cells[1, 3] = "Дата";

sheet.Cells[1, 4] = "Приход/расход";

sheet.Cells[1, 5] = "Тип операции";

for (int i = 0; i < DataGridComeJournal.Items.Count; i++)

{

var selectedRow = DataGridComeJournal.Items[i] as ComeJournalData;

sheet.Cells[i + 2, 1] = selectedRow.SubjectName;

sheet.Cells[i + 2, 2] = selectedRow.Quantity;

sheet.Cells[i + 2, 3] = selectedRow.DateTimeConfirm;

sheet.Cells[i + 2, 4] = selectedRow.IsCome;

sheet.Cells[i + 2, 5] = selectedRow.Operation;

}

var headers = sheet.get\_Range("A1", "E1");

headers.Cells.Font.Size = 14;

var aRange = sheet.get\_Range("A1", "E100");

aRange.Columns.AutoFit();

}

}

}

* + 1. Листинг файла ChangeStorageMethodWindow.xaml

<Window x:Class="MicroElectronsManager.ChangeStorageMethodWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Изменение способа хранение" Width="800" MinWidth="800" Height="400" MinHeight="400" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid HorizontalAlignment="Center" VerticalAlignment="Center" Width="350">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock x:Name="TbProductName" Text="Product name" FontSize="19" Grid.Row="0" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Grid Grid.Row="1" VerticalAlignment="Center" Margin="0 0 0 5">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Способ хранения" Grid.Row="0" FontSize="18" Foreground="#FF212529"/>

<ComboBox x:Name="CbMethods" Grid.Row="1" TextBlock.FontSize="18" TextBlock.Foreground="#FF212529"/>

</Grid>

<Grid Grid.Row="2" VerticalAlignment="Center" Margin="0 10 0 0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="BtnConfirm" Content="Ок" Grid.Column="0" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnConfirm\_Click"/>

<Button x:Name="BtnCancel" Content="Отмена" Grid.Column="1" TextBlock.FontSize="18" HorizontalAlignment="Center" Width="150"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" Click="BtnCancel\_Click"/>

</Grid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла ChangeStorageMethodWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для ChangeStorageMethodWindow.xaml

/// </summary>

public partial class ChangeStorageMethodWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public ProductData productData;

public ChangeStorageMethodWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbProductName.Text = productData.Name;

WriteListMethods();

}

private void WriteListMethods()

{

try

{

var response = apiClient.Get<List<string>>(new RestRequest("product/AllStorageMethodList"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

CbMethods.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

private void BtnConfirm\_Click(object sender, RoutedEventArgs e)

{

try

{

ValidForm();

var response = apiClient.Post(new RestRequest("product/ChangeStorageMethod")

.AddJsonBody(new

{

ProductId = productData.ProductId,

StorageMethod = CbMethods.SelectedItem.ToString()

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

(this.Owner as StorekeepWindow).DataGridStorageWrite();

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void ValidForm()

{

if (CbMethods.SelectedItem == null) { throw new Exception("Выберите способ хранения"); }

}

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

}

}

* + 1. Листинг файла HRmanagerWindow.xaml

<Window x:Class="MicroElectronsManager.HRmanagerWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Отдел кадров" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<Menu Grid.Row="1" FontSize="18">

<MenuItem x:Name="MenuEmployee" Header="Сотрудники" IsEnabled="False" Foreground="#212529" Click="MenuEmployee\_Click" Padding="5 3"/>

<MenuItem x:Name="MenuLabors" Header="Трудовые договора" Foreground="#212529" Click="MenuLabors\_Click" Padding="5 3"/>

<MenuItem x:Name="MenuDismiss" Header="Заявления на увольнение" Foreground="#212529" Click="MenuDismiss\_Click" Padding="5 3"/>

<MenuItem x:Name="MenuHoliday" Header="Заявления на отпуск" Foreground="#212529" Click="MenuHoliday\_Click" Padding="5 3"/>

</Menu>

<!-- Сотрудники -->

<Grid x:Name="ContentEmployee" Grid.Row="2" Visibility="Visible">

<DataGrid x:Name="DataGridEmployee" Grid.Row="2" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True" SelectionChanged="DataGridEmployee\_SelectionChanged">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmAdd" Header="Добавить нового сотрудника" Click="CmAdd\_Click"></MenuItem>

<MenuItem x:Name="CmDismiss" Header="Уволить сотрудника" Click="CmDismiss\_Click" IsEnabled="False"></MenuItem>

<MenuItem x:Name="CmHoliday" Header="Отправить в отпуск" Click="CmHoliday\_Click" IsEnabled="False"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=EmployeeId}" Visibility="Collapsed"/>

<DataGridTextColumn Header="Фамилия" Width="\*" Binding="{Binding Path=LastName}"/>

<DataGridTextColumn Header="Имя" Width="\*" Binding="{Binding Path=FirstName}"/>

<DataGridTextColumn Header="Отчество" Width="\*" Binding="{Binding Path=Patronymic}"/>

<DataGridTextColumn Header="Дата рождения" Width="160" Binding="{Binding Path=Birthday}"/>

<DataGridTextColumn Header="Логин" Width="\*" Binding="{Binding Path=Login}"/>

<DataGridTextColumn Header="Должность" Width="\*" Binding="{Binding Path=Post}"/>

<DataGridTextColumn Header="Оклад" Width="160" Binding="{Binding Path=Salary}"/>

<DataGridTextColumn Header="Статус" Width="160" Binding="{Binding Path=Status}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

<!-- Трудовые договора -->

<Grid x:Name="ContentLabors" Grid.Row="2" Visibility="Collapsed">

<DataGrid x:Name="DataGridLabor" Grid.Row="2" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=EmployeeId}" Visibility="Collapsed"/>

<DataGridTextColumn Header="Фамилия" Width="\*" Binding="{Binding Path=LastName}"/>

<DataGridTextColumn Header="Имя" Width="\*" Binding="{Binding Path=FirstName}"/>

<DataGridTextColumn Header="Отчество" Width="\*" Binding="{Binding Path=Patronymic}"/>

<DataGridTextColumn Header="Должность" Width="\*" Binding="{Binding Path=Post}"/>

<DataGridTextColumn Header="Дата" Width="160" Binding="{Binding Path=Date}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

<!-- Заявления на увольнение -->

<Grid x:Name="ContentDismiss" Grid.Row="2" Visibility="Collapsed">

<DataGrid x:Name="DataGridDismiss" Grid.Row="2" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=EmployeeId}" Visibility="Collapsed"/>

<DataGridTextColumn Header="Фамилия" Width="\*" Binding="{Binding Path=LastName}"/>

<DataGridTextColumn Header="Имя" Width="\*" Binding="{Binding Path=FirstName}"/>

<DataGridTextColumn Header="Отчество" Width="\*" Binding="{Binding Path=Patronymic}"/>

<DataGridTextColumn Header="Должность" Width="\*" Binding="{Binding Path=Post}"/>

<DataGridTextColumn Header="Дата" Width="160" Binding="{Binding Path=Date}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

<!-- Заявления на отпуск -->

<Grid x:Name="ContentHoliday" Grid.Row="2" Visibility="Collapsed">

<DataGrid x:Name="DataGridHoliday" Grid.Row="2" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="HolidayOff" Header="Вызвать досрочно" Click="HolidayOff\_Click"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=EmployeeId}" Visibility="Collapsed"/>

<DataGridTextColumn Header="Фамилия" Width="\*" Binding="{Binding Path=LastName}"/>

<DataGridTextColumn Header="Имя" Width="\*" Binding="{Binding Path=FirstName}"/>

<DataGridTextColumn Header="Отчество" Width="\*" Binding="{Binding Path=Patronymic}"/>

<DataGridTextColumn Header="Должность" Width="\*" Binding="{Binding Path=Post}"/>

<DataGridTextColumn Header="Дата начала" Width="160" Binding="{Binding Path=DateStart}"/>

<DataGridTextColumn Header="Дата конца" Width="160" Binding="{Binding Path=DateEnd}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла HRmanagerWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для HRmanagerWindow.xaml

/// </summary>

public partial class HRmanagerWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public HRmanagerWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

GridEmployeeWrite();

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

public void GridEmployeeWrite()

{

try

{

DataGridEmployee.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<UserData>>(new RestRequest("user/alllist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridEmployee.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void GridLaborWrite()

{

try

{

DataGridLabor.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<LaborData>>(new RestRequest("user/laborslist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridLabor.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void GridDismissWrite()

{

try

{

DataGridDismiss.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<DismissData>>(new RestRequest("user/dismisslist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridDismiss.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

public void GridHolidayWrite()

{

try

{

DataGridHoliday.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<HolidayData>>(new RestRequest("user/holidaylist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridHoliday.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void CmAdd\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddEmployeeWindow() { Owner = this }.Show();

}

private void HolidayOff\_Click(object sender, RoutedEventArgs e)

{

try

{

if (DataGridHoliday.SelectedItem == null)

{

throw new Exception("Никакой пользователь не выбран");

}

var selectedUser = DataGridHoliday.SelectedItem as UserData;

var response = apiClient.Delete(new RestRequest("user/dismissal")

.AddJsonBody(new { EmployeeId = selectedUser.EmployeeId }));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

MessageBox.Show("Сотрудник успешно вызван", "Оповещение", MessageBoxButton.OK, MessageBoxImage.Information);

GridHolidayWrite();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void CmDismiss\_Click(object sender, RoutedEventArgs e)

{

try

{

if (DataGridEmployee.SelectedItem == null)

{

throw new Exception("Никакой пользователь не выбран");

}

var selectedUser = DataGridEmployee.SelectedItem as UserData;

if (MessageBox.Show($"Вы точно хотите уволить сотрудника: {selectedUser.LastName} {selectedUser.FirstName} {selectedUser.Patronymic}?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question)

== MessageBoxResult.Yes)

{

var response = apiClient.Post(new RestRequest("user/dismissal")

.AddJsonBody(new { EmployeeId = selectedUser.EmployeeId }));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

MessageBox.Show("Сотрудник успешно уволен", "Оповещение", MessageBoxButton.OK, MessageBoxImage.Information);

GridEmployeeWrite();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void CmHoliday\_Click(object sender, RoutedEventArgs e)

{

try

{

if (DataGridEmployee.SelectedItem == null)

{

throw new Exception("Никакой пользователь не выбран");

}

var selectedUser = DataGridEmployee.SelectedItem as UserData;

if (selectedUser.Status == "В отпуске")

{

throw new Exception("Сотрудник уже в отпуске");

}

this.IsEnabled = false;

new AddHolidayWindow() { Owner = this, user = selectedUser }.Show();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void ContentsClear()

{

MenuEmployee.IsEnabled = true;

MenuLabors.IsEnabled = true;

MenuDismiss.IsEnabled = true;

MenuHoliday.IsEnabled = true;

ContentEmployee.Visibility = Visibility.Collapsed;

ContentLabors.Visibility = Visibility.Collapsed;

ContentDismiss.Visibility = Visibility.Collapsed;

ContentHoliday.Visibility = Visibility.Collapsed;

}

private void MenuEmployee\_Click(object sender, RoutedEventArgs e)

{

ContentsClear();

MenuEmployee.IsEnabled = false;

ContentEmployee.Visibility = Visibility.Visible;

GridEmployeeWrite();

}

private void MenuLabors\_Click(object sender, RoutedEventArgs e)

{

ContentsClear();

MenuLabors.IsEnabled = false;

ContentLabors.Visibility = Visibility.Visible;

GridLaborWrite();

}

private void MenuDismiss\_Click(object sender, RoutedEventArgs e)

{

ContentsClear();

MenuDismiss.IsEnabled = false;

ContentDismiss.Visibility = Visibility.Visible;

GridDismissWrite();

}

private void MenuHoliday\_Click(object sender, RoutedEventArgs e)

{

ContentsClear();

MenuHoliday.IsEnabled = false;

ContentHoliday.Visibility = Visibility.Visible;

GridHolidayWrite();

}

private void DataGridEmployee\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

CmDismiss.IsEnabled = DataGridEmployee.SelectedItem != null;

CmHoliday.IsEnabled = DataGridEmployee.SelectedItem != null;

}

}

}

* + 1. Листинг файла LoginWindow.xaml

<Window x:Class="MicroElectronsManager.LoginWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Авторизация" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="180"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<Image Source="/res/logo.png" VerticalAlignment="Center" HorizontalAlignment="Center" Margin="0 15"/>

</Grid>

<Grid Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center" Width="320" Height="200">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" VerticalAlignment="Center">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Логин" Grid.Row="0" FontSize="18"/>

<TextBox x:Name="TbLogin" Text="" Grid.Row="1" TextBlock.FontSize="18"/>

</Grid>

<Grid Grid.Row="1" VerticalAlignment="Center">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="Пароль" Grid.Row="0" FontSize="18"/>

<PasswordBox x:Name="TbPassword" Password="" Grid.Row="1" TextBlock.FontSize="18"/>

</Grid>

<Button x:Name="BtnLogin" Grid.Row="2" HorizontalAlignment="Center" VerticalAlignment="Center" Content="Войти" Padding="40 5"

Foreground="#ffffff" BorderBrush="#212529" Background="#212529" TextBlock.FontSize="18" Click="BtnLogin\_Click"/>

</Grid>

</Grid>

</Window>

* + 1. Листинг файла LoginWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class LoginWindow : Window

{

private RestClient apiClient = ApiBuilder.GetInstance();

public LoginWindow()

{

InitializeComponent();

}

private void BtnLogin\_Click(object sender, RoutedEventArgs e)

{

try

{

var response = apiClient.Post<UserData>(new RestRequest("user/auth")

.AddJsonBody(new

{

Login = TbLogin.Text.ToString(),

Password = TbPassword.Password.ToString()

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

if (response.Data.Status == "Уволен")

{

throw new Exception("Вы уволены, у вас больше нет доступа к системе.");

}

if (response.Data.Status == "В отпуске")

{

throw new Exception("На время отпуска у вас нет доступа к системе, идите отдохните.");

}

this.Hide();

TbLogin.Text = "";

TbPassword.Password = "";

switch (response.Data.Post)

{

case "Бухгалтер":

new BookkeepWindow() { Owner = this, user = response.Data }.Show();

break;

case "Кладовщик":

new StorekeepWindow() { Owner = this, user = response.Data }.Show();

break;

case "HR менеджер":

new HRmanagerWindow() { Owner = this, user = response.Data }.Show();

break;

case "Вахтёр":

new VisitorWindow() { Owner = this, user = response.Data }.Show();

break;

case "Менеджер поставок":

new SupplyWindow() { Owner = this, user = response.Data }.Show();

break;

case "Начальник производства":

new ManufactureWindow() { Owner = this, user = response.Data }.Show();

break;

default:

this.Show();

break;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

* + 1. Листинг файла ManufactureWindow.xaml

<Window x:Class="MicroElectronsManager.ManufactureWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Производство" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<DataGrid x:Name="DataGridManufs" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True" SelectionChanged="DataGridManufs\_SelectionChanged">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmAdd" Header="Добавить задачу" Click="CmAdd\_Click"></MenuItem>

<MenuItem x:Name="CmEnd" Header="Окончить задачу" Click="CmEnd\_Click" IsEnabled="False"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Товар" Width="\*" Binding="{Binding Path=ProductName}"/>

<DataGridTextColumn Header="Количество" Width="170" Binding="{Binding Path=Quantity}"/>

<DataGridTextColumn Header="Дата начала" Width="170" Binding="{Binding Path=DateStart}"/>

<DataGridTextColumn Header="Дата дедлайна" Width="170" Binding="{Binding Path=DateDeadline}"/>

<DataGridTextColumn Header="Дата окончания" Width="170" Binding="{Binding Path=DateEnd}"/>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=TaskId}" Visibility="Collapsed"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла ManufactureWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для ManufactureWindow.xaml

/// </summary>

public partial class ManufactureWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public ManufactureWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

DataGridManufsWrite();

}

public void DataGridManufsWrite()

{

try

{

DataGridManufs.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<ManufData>>(new RestRequest("manuf/alllist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridManufs.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void CmEnd\_Click(object sender, RoutedEventArgs e)

{

if (MessageBox.Show("Задача действительно выполнена?", "Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.No)

{

return;

}

try

{

var selectedTask = DataGridManufs.SelectedItem as ManufData;

var response = apiClient.Post(new RestRequest("manuf/End")

.AddJsonBody(new

{

TaskId = selectedTask.TaskId,

DateEnd = DateTime.Now.ToString("dd.MM.yyyy")

}));

DataGridManufsWrite();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void CmAdd\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddTaskWindow() { Owner = this }.Show();

}

private void DataGridManufs\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

var selectedTask = DataGridManufs.SelectedItem as ManufData;

CmEnd.IsEnabled = DataGridManufs.SelectedItem != null && selectedTask.DateEnd == "";

}

}

}

* + 1. Листинг файла StorekeepWindow.xaml

<Window x:Class="MicroElectronsManager.StorekeepWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Склад" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<DataGrid x:Name="DataGridStorage" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True" SelectionChanged="DataGridStorage\_SelectionChanged">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmChangeMethod" Header="Изменить способ хранения" Click="CmChangeMethod\_Click" IsEnabled="False"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Width="\*" Binding="{Binding Path=ProductId}"/>

<DataGridTextColumn Header="Наименование" Width="\*" Binding="{Binding Path=Name}"/>

<DataGridTextColumn Header="Категория" Width="\*" Binding="{Binding Path=CategoryName}"/>

<DataGridTextColumn Header="Количество" Width="170" Binding="{Binding Path=Quantity}"/>

<DataGridTextColumn Header="Способ хранения" Width="180" Binding="{Binding Path=StorageMethod}"/>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=ProductId}" Visibility="Collapsed"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла StorekeepWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для StorekeepWindow.xaml

/// </summary>

public partial class StorekeepWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public StorekeepWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

DataGridStorageWrite();

}

public void DataGridStorageWrite()

{

try

{

DataGridStorage.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<ProductData>>(new RestRequest("product/AllStorageList"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridStorage.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void CmChangeMethod\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new ChangeStorageMethodWindow() { Owner = this, productData = DataGridStorage.SelectedItem as ProductData }.Show();

}

private void DataGridStorage\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

CmChangeMethod.IsEnabled = DataGridStorage.SelectedItem != null;

}

}

}

* + 1. Листинг файла SupplyMoreInfoWindow.xaml

<Window x:Class="MicroElectronsManager.SupplyMoreInfoWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Состав поставки" Width="800" MinWidth="800" Height="400" MinHeight="400" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Loaded="Window\_Loaded" Closed="Window\_Closed">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="200"/>

</Grid.ColumnDefinitions>

<TextBlock x:Name="TbCounterpartyName" Grid.Column="0" HorizontalAlignment="Center" VerticalAlignment="Center" Text="ООО Микроэлектроника" FontSize="18"/>

<TextBlock x:Name="TbSellOrBuy" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" Text="Покупка " FontSize="18"/>

<TextBlock x:Name="TbDate" Grid.Column="2" HorizontalAlignment="Center" VerticalAlignment="Center" Text="20000.0 Руб" FontSize="18"/>

</Grid>

<DataGrid x:Name="DataGridProducts" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Наименование" Width="\*" Binding="{Binding Path=Name}"/>

<DataGridTextColumn Header="Категория" Width="\*" Binding="{Binding Path=CategoryName}"/>

<DataGridTextColumn Header="Количество" Width="150" Binding="{Binding Path=Quantity}"/>

<DataGridTextColumn Header="Сумма" Width="150" Binding="{Binding Path=Price}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла SupplyMoreInfoWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для SupplyMoreInfoWindow.xaml

/// </summary>

public partial class SupplyMoreInfoWindow : Window

{

public SupplyData supplyData;

public SupplyMoreInfoWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbCounterpartyName.Text = supplyData.Counterparty;

TbSellOrBuy.Text = supplyData.SellOrBuy;

TbDate.Text = supplyData.Date;

DataGridProducts.ClearValue(ItemsControl.ItemsSourceProperty);

DataGridProducts.ItemsSource = supplyData.Products;

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.IsEnabled = true;

}

}

}

* + 1. Листинг файла SupplyWindow.xaml

<Window x:Class="MicroElectronsManager.SupplyWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Поставки" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<DataGrid x:Name="DataGridSupply" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True" SelectionChanged="DataGridSupply\_SelectionChanged">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmAdd" Header="Добавить поставку" Click="CmAdd\_Click"></MenuItem>

<MenuItem x:Name="CmCompos" Header="Состав поставки" Click="CmCompos\_Click" IsEnabled="False"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Контрагент" Width="\*" Binding="{Binding Path=Counterparty}"/>

<DataGridTextColumn Header="Тип поставки" Width="160" Binding="{Binding Path=SellOrBuy}"/>

<DataGridTextColumn Header="Сумма" Width="160" Binding="{Binding Path=Summa}"/>

<DataGridTextColumn Header="Дата" Width="160" Binding="{Binding Path=Date}"/>

<DataGridTextColumn Header="Id" Width="\*" Binding="{Binding Path=SupplyId}" Visibility="Collapsed"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла SupplyWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для SupplyWindow.xaml

/// </summary>

public partial class SupplyWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public SupplyWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

DataGridSupplyWrite();

}

public void DataGridSupplyWrite()

{

try

{

DataGridSupply.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<SupplyData>>(new RestRequest("supply/alllist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridSupply.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void CmAdd\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddSupplyWindow() { Owner = this }.Show();

}

private void CmCompos\_Click(object sender, RoutedEventArgs e)

{

try

{

var selectedItem = DataGridSupply.SelectedItem as SupplyData;

var response = apiClient.Post<SupplyData>(new RestRequest("supply/moreinfobyid")

.AddJsonBody(new

{

SupplyId = selectedItem.SupplyId

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

this.IsEnabled = false;

new SupplyMoreInfoWindow() { Owner = this, supplyData = response.Data }.Show();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void DataGridSupply\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

CmCompos.IsEnabled = DataGridSupply.SelectedItem != null;

}

}

}

* + 1. Листинг файла VisitorWindow.xaml

<Window x:Class="MicroElectronsManager.VisitorWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:MicroElectronsManager"

mc:Ignorable="d"

Title="Журнал посещений" Width="1366" MinWidth="1366" Height="768" MinHeight="768" WindowState="Maximized" WindowStyle="SingleBorderWindow" WindowStartupLocation="CenterScreen" Closed="Window\_Closed" Loaded="Window\_Loaded">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#212529">

<TextBlock x:Name="TbWelcome" Text="Здравствуйте, юзернейм" Foreground="#ffffff" FontSize="18" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<Button x:Name="BtnExit" Content="Выход" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0" Padding="30 5"

Background="#ffffff" BorderBrush="#ffffff" Foreground="#212529" FontSize="15" TextBlock.FontSize="18" Click="BtnExit\_Click"/>

<Image Source="/res/logo.png" HorizontalAlignment="Right" Margin="0 15 20 15"/>

</Grid>

<DataGrid x:Name="DataGridVisitors" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" FontSize="18"

CanUserAddRows="False" CanUserDeleteRows="False" CanUserResizeRows="False" AutoGenerateColumns="False" SelectionMode="Single"

RowHeaderWidth="0" Padding="2" HorizontalGridLinesBrush="LightGray" Foreground="#212529"

GridLinesVisibility="Horizontal" RowHeight="35" IsReadOnly="True" SelectionChanged="DataGridVisitors\_SelectionChanged">

<DataGrid.ContextMenu>

<ContextMenu>

<MenuItem x:Name="CmAdd" Header="Добавить посетителя" Click="CmAdd\_Click"></MenuItem>

<MenuItem x:Name="CmExit" Header="Выход посетителя" Click="CmExit\_Click" IsEnabled="False"></MenuItem>

<MenuItem x:Name="CmWhoEntry" Header="Кто впустил" Click="CmWhoEntry\_Click" IsEnabled="False"></MenuItem>

<MenuItem x:Name="CmWhoExit" Header="Кто выпустил" Click="CmWhoExit\_Click" IsEnabled="False"></MenuItem>

</ContextMenu>

</DataGrid.ContextMenu>

<DataGrid.Columns>

<DataGridTextColumn Header="Фамилия" Width="\*" Binding="{Binding Path=VisitorLastName}"/>

<DataGridTextColumn Header="Имя" Width="\*" Binding="{Binding Path=VisitorFirstName}"/>

<DataGridTextColumn Header="Отчество" Width="\*" Binding="{Binding Path=VisitorPatronymic}"/>

<DataGridTextColumn Header="Паспорт" Width="160" Binding="{Binding Path=Passport}"/>

<DataGridTextColumn Header="Дата входа" Width="160" Binding="{Binding Path=DateTimeEntry}"/>

<DataGridTextColumn Header="Дата выхода" Width="160" Binding="{Binding Path=DateTimeExit}"/>

<DataGridTextColumn Header="Кто впустил" Width="\*" Binding="{Binding Path=EmployeeEntryName}" Visibility="Collapsed"/>

<DataGridTextColumn Header="Кто выпустил" Width="\*" Binding="{Binding Path=EmployeeExitName}" Visibility="Collapsed"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Window>

* + 1. Листинг файла VisitorWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using RestSharp;

using MicroElectronsManager.Logics;

using MicroElectronsManager.Models;

namespace MicroElectronsManager

{

/// <summary>

/// Логика взаимодействия для VisitorWindow.xaml

/// </summary>

public partial class VisitorWindow : Window

{

public UserData user;

private RestClient apiClient = ApiBuilder.GetInstance();

public VisitorWindow()

{

InitializeComponent();

}

private void Window\_Closed(object sender, EventArgs e)

{

this.Owner.Show();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

TbWelcome.Text = $"Здравствуйте, {user.FirstName} {user.Patronymic}";

DataGridVisitorWrite();

}

public void DataGridVisitorWrite()

{

try

{

DataGridVisitors.ClearValue(ItemsControl.ItemsSourceProperty);

var response = apiClient.Get<List<VisitorData>>(new RestRequest("visitor/alllist"));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridVisitors.ItemsSource = response.Data;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void BtnExit\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void CmAdd\_Click(object sender, RoutedEventArgs e)

{

this.IsEnabled = false;

new AddVisitor() { Owner = this, user = user }.Show();

}

private void CmExit\_Click(object sender, RoutedEventArgs e)

{

try

{

if (MessageBox.Show("Посетитель действительно вышел?", "Подтверждение.", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.No)

{

return;

}

var selectedVisistor = DataGridVisitors.SelectedItem as VisitorData;

var response = apiClient.Post(new RestRequest("visitor/writeexit")

.AddJsonBody(new

{

EmployeeExitId = user.EmployeeId,

VisitorLastName = selectedVisistor.VisitorLastName,

VisitorFirstName = selectedVisistor.VisitorFirstName,

VisitorPatronymic = selectedVisistor.VisitorPatronymic

}));

if (response.StatusCode != System.Net.HttpStatusCode.OK)

{

throw new Exception(ResponseMessageHandler.GetMessage(response.Content));

}

DataGridVisitorWrite();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void DataGridVisitors\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

var selectedVisistor = DataGridVisitors.SelectedItem as VisitorData;

CmExit.IsEnabled = selectedVisistor != null && selectedVisistor.DateTimeExit == null;

CmWhoEntry.IsEnabled = selectedVisistor != null;

CmWhoExit.IsEnabled = selectedVisistor != null && selectedVisistor.DateTimeExit != null;

}

private void CmWhoEntry\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show($"{(DataGridVisitors.SelectedItem as VisitorData).EmployeeEntryName}", "Кто впустил?", MessageBoxButton.OK, MessageBoxImage.Information);

}

private void CmWhoExit\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show($"{(DataGridVisitors.SelectedItem as VisitorData).EmployeeExitName}", "Кто выпустил?", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

}