# реализация программной системы

## 1. Общие принципы организации работы системы.

Разрабатываемая система реализована на платформе .NET Core, с использованием Entity Framework Core. Для представлений был использован ReactJS. В качестве хранилища выступает MS SQL. Вся разработка велась в Visual Studio Code.

## 3. Реализация механизма регистрации, аутентификации и авторизации

Для реализации регистрации, аутентификации и авторизации пользователей были использованы стандартные методы шаблона Microsoft Identity. На втором шаге регистрации пользователям назначаются роли, в соответствии с их принадлежностью к системе. Незарегистрированные пользователи не имеют доступа к функционалу сайта.

## 4. Пользовательский интерфейс

Пользовательский интерфейс представлен на рисунках 1-.

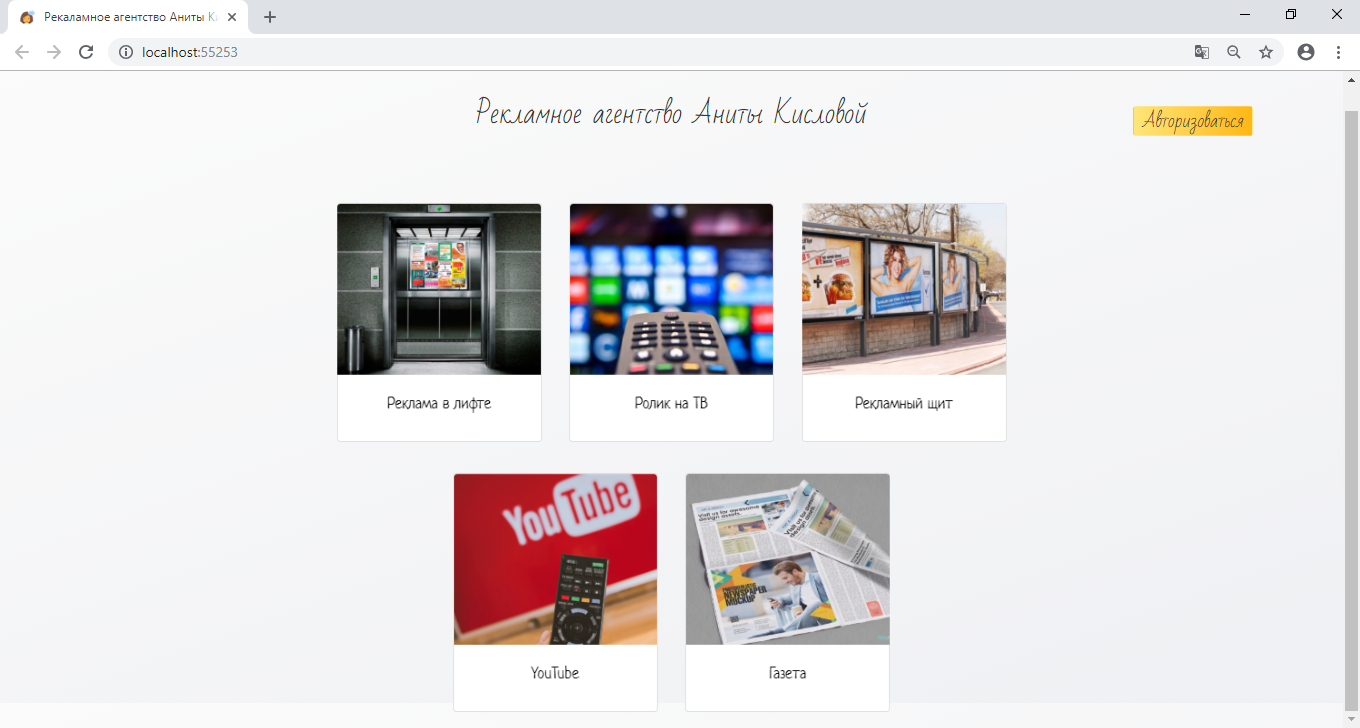


Рисунок 1. Главная страница для гостя

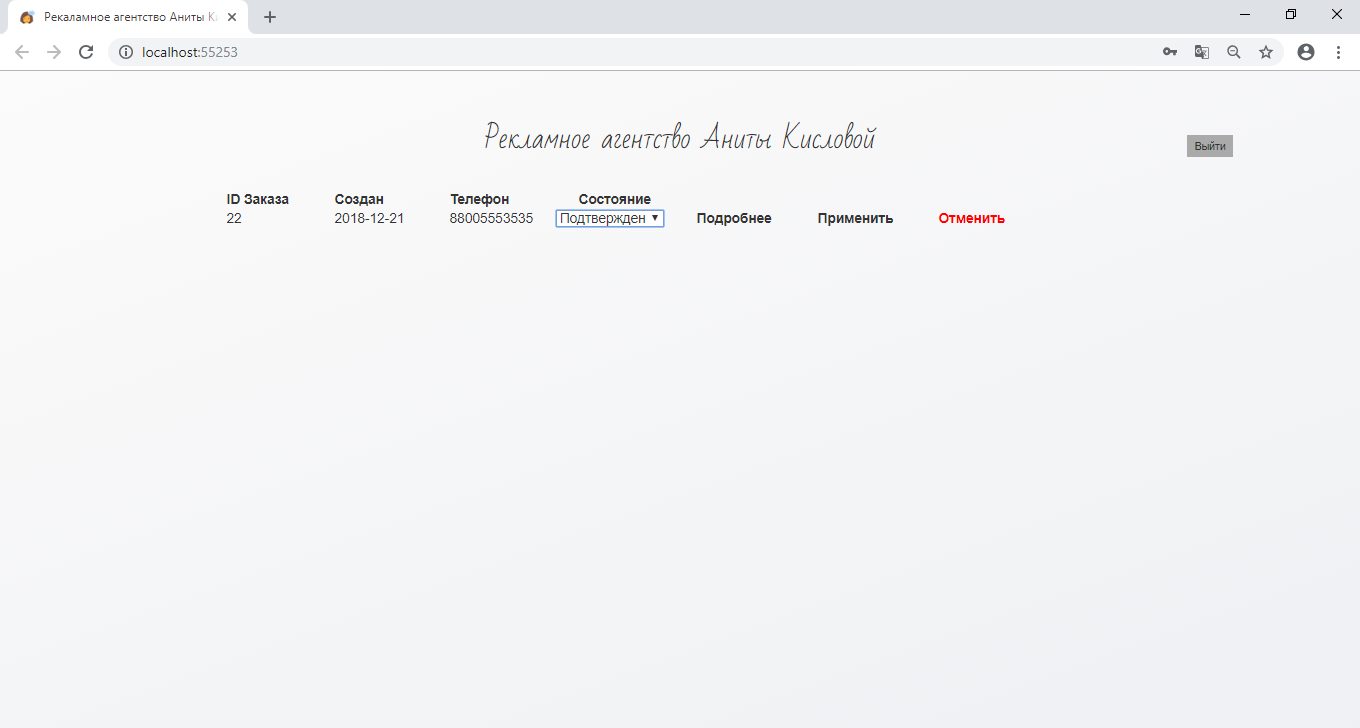


Рисунок 2. Главная страница для менеджера

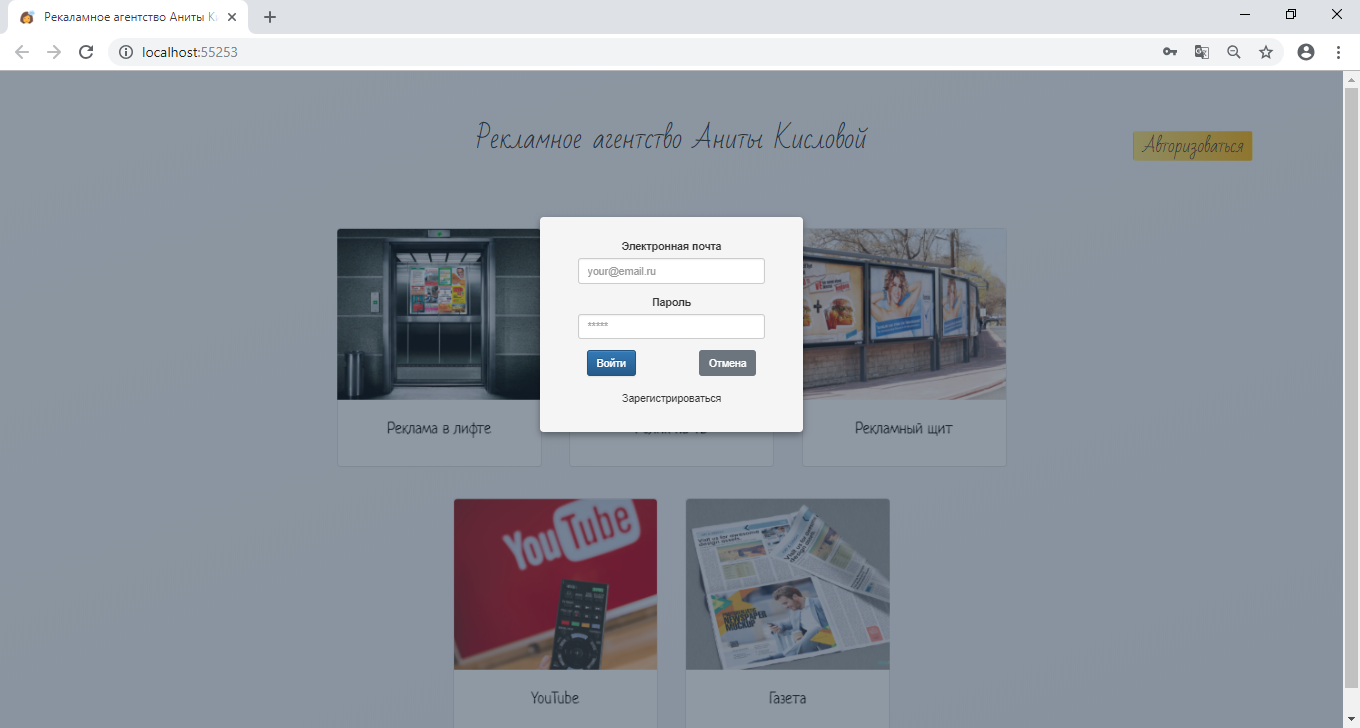


Рисунок 3. Окно авторизации

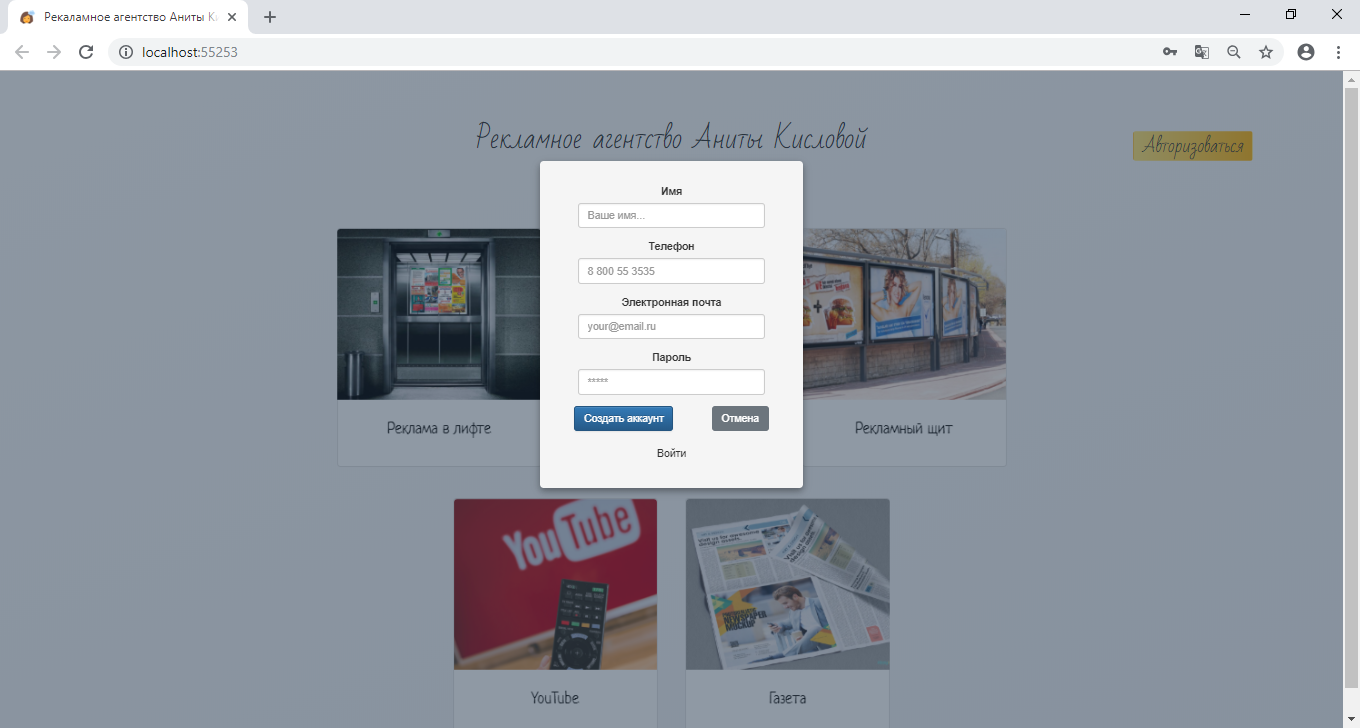


Рисунок 4. Окно регистрации.

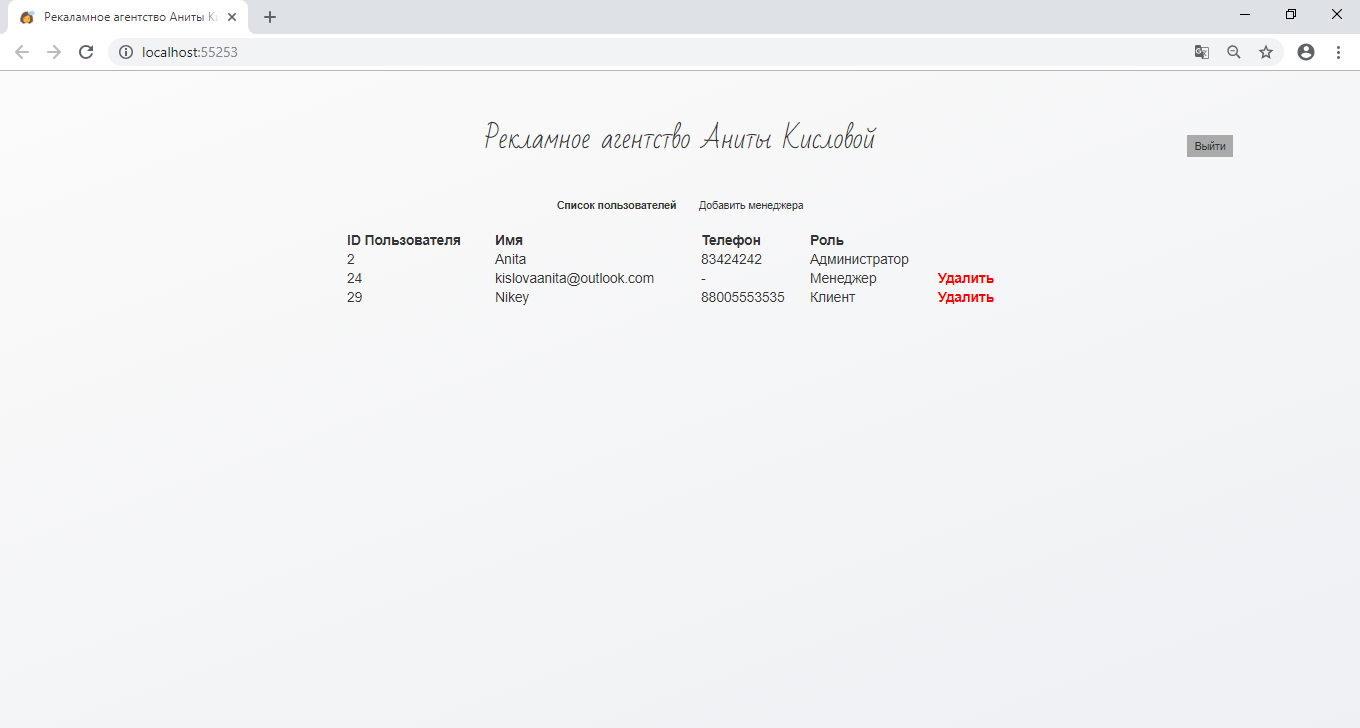


Рисунок 5. Главная страница администратора.

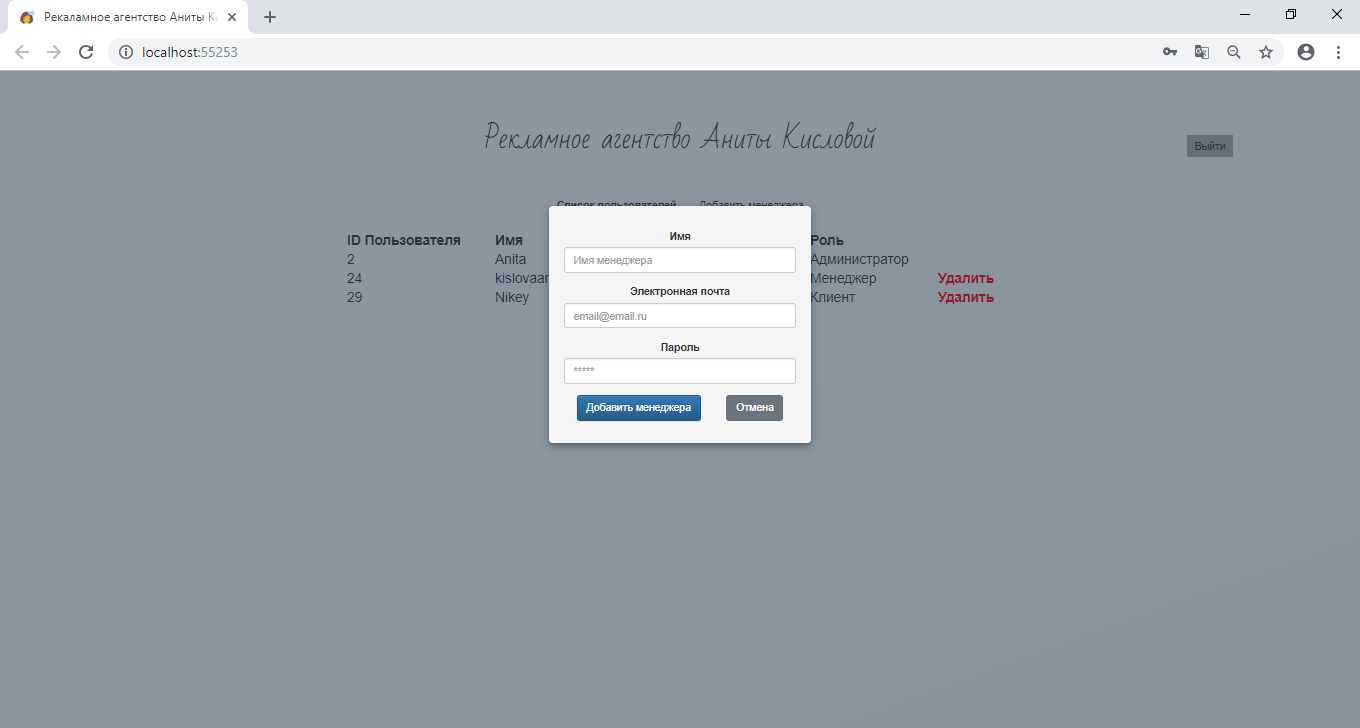


Рисунок 6. Добавление нового менеджера.

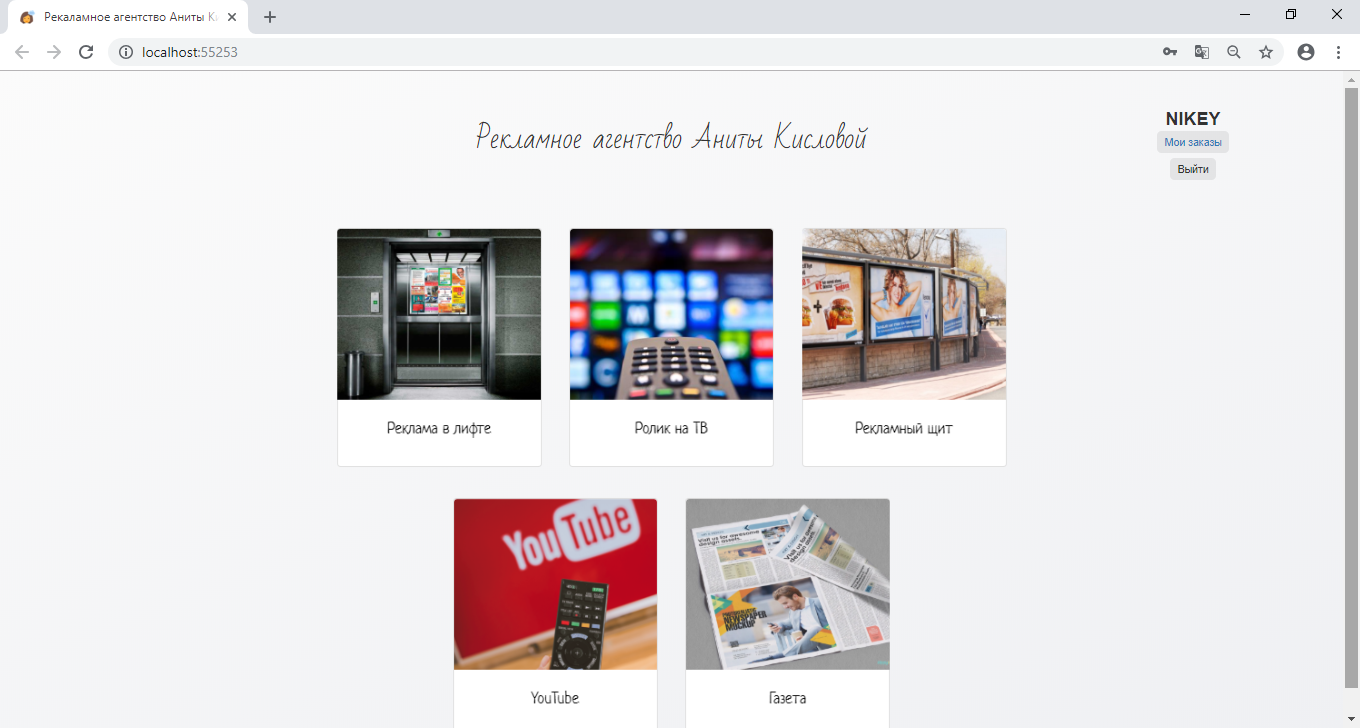


Рисунок 7. Главная страница заказчика.



Рисунок 8. Страница со списком заказов у заказчика.

# исходный код программной системы

Модели:

Класс AuthUserViewModel.cs

namespace AdvAgency.Models

{

public class AuthUserViewModel

{

public string Email { get; set; }

public string Password { get; set; }

}

}

Класс Order.cs

namespace AdvAgency.Models

{

public class Order

{

public int Id { get; set; }

public int ServiceId { get; set; }

public Service Service { get; set; }

public int UserId { get; set; }

public User User { get; set; }

public string Comment { get; set; }

public string Duration { get; set; }

public OrderState Status { get; set; }

public DateTime Created { get; set; }

}

}

Класс OrderState.cs

namespace AdvAgency.Models

{

public enum OrderState

{

Reicived,//принят

Confirmed,//подтвержден

Proceed,//выполняется

Canceled,//отменен

Done

}

}

Класс OrderViewModel.cs

namespace AdvAgency.Models

{

public class OrderViewModel

{

public int UserId { get; set; }

public int ServiceId { get; set; }

public string Duration { get; set; }

public string Comment { get; set; }

public static explicit operator Order(OrderViewModel m)

{

return new Order()

{

UserId = m.UserId,

ServiceId = m.ServiceId,

Duration = m.Duration,

Comment = m.Comment

};

}

}

}

Класс RegisterUserViewModel.cs

namespace AdvAgency.Models

{

public class RegisterUserViewModel

{

public string Name { get; set; }

public string Password { get; set; }

public string Phone { get; set; }

public string Email { get; set; }

public int RoleId { get; set; }

public static explicit operator User(RegisterUserViewModel m)

{

return new User()

{

FullName = m.Name,

Phone = m.Phone,

Email = m.Email,

RoleId = m.RoleId

};

}

}

}

Класс Role.cs

namespace AdvAgency.Models

{

public class Role

{

public int Id { get; set; }

public String Name { get; set; }

}

}

Класс Service.cs

namespace AdvAgency.Models

{

public class Service

{

public Service() { }

public int Id { get; set; }

public string Title { get; set; }

public string ImageUrl { get; set; }

}

}

}

Класс User.cs

namespace AdvAgency.Models

{

public class User

{

public User() { }

public int Id { get; set; }

public String FullName { get; set; }

public String Email { get; set; }

public String Phone { get; set; }

public int RoleId { get; set; }

public Role Role { get; set; }

public byte[] PasswordHash { get; set; }

public byte[] PasswordSalt { get; set; }

}

}

Класс 20181125164828\_initial.cs

namespace AdvAgency.Migrations

{

public partial class initial : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.CreateTable(

name: "Services",

columns: table => new

{

Id = table.Column<int>(nullable: false)

.Annotation("MySQL:AutoIncrement", true),

Title = table.Column<string>(nullable: true),

ImageUrl = table.Column<string>(nullable: true)

},

constraints: table =>

{

table.PrimaryKey("PK\_Services", x => x.Id);

});

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "Services");

}

}

}

Класс 20181130083213\_addUsers.cs

namespace AdvAgency.Migrations

{

public partial class addUsers : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.CreateTable(

name: "Role",

columns: table => new

{

Id = table.Column<int>(nullable: false)

.Annotation("MySql:ValueGenerationStrategy", MySqlValueGenerationStrategy.IdentityColumn),

Name = table.Column<string>(nullable: true)

},

constraints: table =>

{

table.PrimaryKey("PK\_Role", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Users",

columns: table => new

{

Id = table.Column<int>(nullable: false)

.Annotation("MySql:ValueGenerationStrategy", MySqlValueGenerationStrategy.IdentityColumn),

FullName = table.Column<string>(nullable: true),

Phone = table.Column<string>(nullable: true),

RoleId = table.Column<int>(nullable: false),

PasswordHash = table.Column<byte[]>(nullable: true),

PasswordSalt = table.Column<byte[]>(nullable: true)

},

constraints: table =>

{

table.PrimaryKey("PK\_Users", x => x.Id);

table.ForeignKey(

name: "FK\_Users\_Role\_RoleId",

column: x => x.RoleId,

principalTable: "Role",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateIndex(

name: "IX\_Users\_RoleId",

table: "Users",

column: "RoleId");

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "Users");

migrationBuilder.DropTable(

name: "Role");

}

}

}

Класс 20181130225328\_addRoles.cs

namespace AdvAgency.Migrations

{

public partial class addRoles : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropForeignKey(

name: "FK\_Users\_Role\_RoleId",

table: "Users");

migrationBuilder.DropPrimaryKey(

name: "PK\_Role",

table: "Role");

migrationBuilder.RenameTable(

name: "Role",

newName: "Roles");

migrationBuilder.AddPrimaryKey(

name: "PK\_Roles",

table: "Roles",

column: "Id");

migrationBuilder.AddForeignKey(

name: "FK\_Users\_Roles\_RoleId",

table: "Users",

column: "RoleId",

principalTable: "Roles",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropForeignKey(

name: "FK\_Users\_Roles\_RoleId",

table: "Users");

migrationBuilder.DropPrimaryKey(

name: "PK\_Roles",

table: "Roles");

migrationBuilder.RenameTable(

name: "Roles",

newName: "Role");

migrationBuilder.AddPrimaryKey(

name: "PK\_Role",

table: "Role",

column: "Id");

migrationBuilder.AddForeignKey(

name: "FK\_Users\_Role\_RoleId",

table: "Users",

column: "RoleId",

principalTable: "Role",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

}

}

}

Класс 20181201093609\_userAddEmail.cs

namespace AdvAgency.Migrations

{

public partial class userAddEmail : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.AddColumn<string>(

name: "Email",

table: "Users",

nullable: true);

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropColumn(

name: "Email",

table: "Users");

}

}

}

Класс 20181217220204\_AddOrdersTable.cs

namespace AdvAgency.Migrations

{

public partial class AddOrdersTable : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.CreateTable(

name: "Orders",

columns: table => new

{

Id = table.Column<int>(nullable: false)

.Annotation("MySql:ValueGenerationStrategy", MySqlValueGenerationStrategy.IdentityColumn),

ServiceId = table.Column<int>(nullable: false),

UserId = table.Column<int>(nullable: false),

Comment = table.Column<string>(nullable: true),

Duration = table.Column<string>(nullable: true),

Status = table.Column<int>(nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Orders", x => x.Id);

table.ForeignKey(

name: "FK\_Orders\_Services\_ServiceId",

column: x => x.ServiceId,

principalTable: "Services",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

table.ForeignKey(

name: "FK\_Orders\_Users\_UserId",

column: x => x.UserId,

principalTable: "Users",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateIndex(

name: "IX\_Orders\_ServiceId",

table: "Orders",

column: "ServiceId");

migrationBuilder.CreateIndex(

name: "IX\_Orders\_UserId",

table: "Orders",

column: "UserId");

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "Orders");

}

}

}

Класс 20181218192056\_RenamedStatus.cs

namespace AdvAgency.Migrations

{

public partial class RenamedStatus : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

}

protected override void Down(MigrationBuilder migrationBuilder)

{

}

}

}

Класс 20181218195419\_AddCreatedPropToOrder.cs

namespace AdvAgency.Migrations

{

public partial class AddCreatedPropToOrder : Migration

{

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.AddColumn<DateTime>(

name: "Created",

table: "Orders",

nullable: false,

defaultValue: new DateTime(1, 1, 1, 0, 0, 0, 0, DateTimeKind.Unspecified));

}

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropColumn(

name: "Created",

table: "Orders");

}

}

}

Контроллеры:

Класс AuthController.cs

namespace AdvAgency.Controllers

{

[Authorize]

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

public class AuthController : Controller

{

const string Admin = "1";

const string Manager = "2";

const string Client = "3";

private readonly IUserService \_userService;

private readonly IConfiguration \_config;

public AuthController(IUserService userService, IConfiguration config)

{

this.\_userService = userService;

this.\_config = config;

}

[AllowAnonymous]

[HttpPost("register")]

public IActionResult Create([FromBody] RegisterUserViewModel model)

{

model.RoleId = 3;

try

{

this.\_userService.Create((User)model, model.Password);

return Ok();

}

catch (AppException e)

{

return BadRequest(new

{

message = e.Message

});

}

catch

{

return BadRequest(new

{

message = "Произошла ошибка, попробуйте позже"

});

}

}

[AllowAnonymous]

[HttpPost("auth")]

public IActionResult Authenticate([FromBody] AuthUserViewModel model) {

if(model == null || model.Email == null || model.Password == null)

return BadRequest(new { message = "Электронная почта или пароль не валидны" });

var user = \_userService.Authenticate(model.Email,model.Password);

if (user == null)

return BadRequest(new { message = "Электронная почта или пароль не валидны" });

var tokenHandler = new JwtSecurityTokenHandler();

var key = Encoding.ASCII.GetBytes(\_config.GetSection("Config:SecretKey").Value);

var tokenDescriptor = new SecurityTokenDescriptor

{

Subject = new ClaimsIdentity(new Claim[]

{

new Claim(ClaimTypes.Name, user.Id.ToString()),

new Claim(ClaimTypes.Role, user.RoleId.ToString()),

}),

Expires = DateTime.UtcNow.AddDays(7),

SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key), SecurityAlgorithms.HmacSha256Signature)

};

var token = tokenHandler.CreateToken(tokenDescriptor);

var tokenString = tokenHandler.WriteToken(token);

return Ok(new {

token = tokenString,

email = user.Email,

fullName = user.FullName,

phone = user.Phone

});

}

}

}

Класс ManagementController.cs

namespace AdvAgency.Controllers

{

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

public class ManagementController : Controller

{

const string Admin = "1";

const string Manager = "2";

const string Client = "3";

private readonly IUserService \_userService;

private readonly IUnitOfWork \_uof;

public ManagementController(IUserService userService, IUnitOfWork unitOfWork)

{

this.\_userService = userService;

this.\_uof = unitOfWork;

}

[Authorize(Roles = Admin)]

[HttpPost("createmanager")]

public IActionResult NewManager([FromBody] RegisterUserViewModel model)

{

try

{

model.RoleId = 2;

\_userService.Create((User)model, model.Password);

return Ok();

} catch (Exception ex)

{

return BadRequest();

}

}

[Authorize(Roles = Admin)]

[HttpGet("getusers")]

public IActionResult GetUsers()

{

try

{

var users = \_userService.Fetch(u => true);

users = users.Select(u =>

{

u.PasswordHash = null;

u.PasswordSalt = null;

return u;

});

return Ok( new

{

users

});

}

catch (Exception ex)

{

return BadRequest();

}

}

[Authorize(Roles = Admin)]

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

public IActionResult RemoveUser([FromRoute]int id)

{

try

{

if (\_userService.Get(id).RoleId == 1) throw new Exception("Can't remove administrator");

\_userService.Delete(id);

\_uof.Complete();

return Ok();

}

catch (Exception ex)

{

return BadRequest( new {

message = ex.Message

});

}

}

}

}

Класс OrderController.cs

namespace AdvAgency.Controllers

{

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

public class OrderController : Controller

{

const string Admin = "1";

const string Manager = "2";

const string Client = "3";

private readonly IRepository<Order> \_orderRepo;

private readonly IUnitOfWork \_uof;

public OrderController(IRepository<Order> orderRepo, IUnitOfWork uof)

{

\_orderRepo = orderRepo;

\_uof = uof;

}

[Authorize(Roles = Client)]

[HttpPost("makeorder")]

public IActionResult MakeOrder([FromBody] OrderViewModel model)

{

try

{

var order = (Order)model;

order.UserId = int.Parse(User.Identity.Name);

order.Created = DateTime.Now;

\_orderRepo.Create(order);

\_uof.Complete();

return Ok();

}

catch (Exception e)

{

return BadRequest(new

{

message = "На сервере произошла ошибка, попробуйте позже"

});

}

}

[Authorize(Roles = (Manager + "," + Client))]

[HttpGet("getorders")]

public IActionResult GetOrders()

{

try

{

IEnumerable<Order> orders;

if (User.GetRoleId() == Client)

{

var userId = int.Parse(User.Identity.Name);

orders = \_orderRepo.Fetch(order => order.UserId == userId);

} else

{

orders = \_orderRepo.Fetch(o => true);

}

return Ok(new

{

orders

});

}

catch (Exception e)

{

return BadRequest(new

{

message = "На сервере произошла ошибка, попробуйте позже"

});

}

}

[Authorize(Roles = Manager)]

[HttpPut("setstatus/{id}")]

public IActionResult SetStatus([FromRoute] int id, [FromQuery] int status )

{

try

{

var order = \_orderRepo.Get(id);

if (order != null)

{

order.Status = (OrderState) status;

\_uof.Complete();

return Ok();

} else

{

return BadRequest("Заказ не найден");

}

} catch (Exception)

{

return BadRequest();

}

}

}

}

Класс ServiceController.cs

namespace AdvAgency.Controllers

{

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

public class ServiceController : Controller

{

private readonly IRepository<Service> \_servicesRepo;

public ServiceController(IRepository<Service> servRep)

{

\_servicesRepo = servRep;

}

[HttpGet("[action]")]

public IActionResult GetServices()

{

try

{

return Ok(new

{

services = \_servicesRepo.Fetch(s => true)

});

}

catch

{

return BadRequest(new

{

message = "Can't fetch data"

});

}

}

}

}