МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»

Навчально науковий інститут інформаційних технологій і робототехніки
Кафедра комп'ютерних та інформаційних технологій і систем

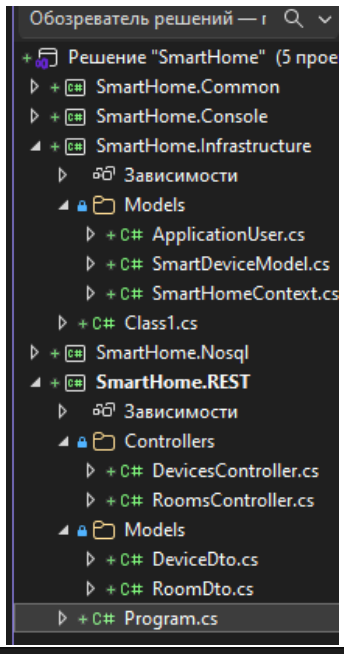
Лабораторна робота № 5

з навчальної дисципліни
«Технології на платформі NET»

Виконав:
Студент групи 403-ТН
Солонецький Роман Миколайович

Перевірив:
Тютюнник Петро Богданович

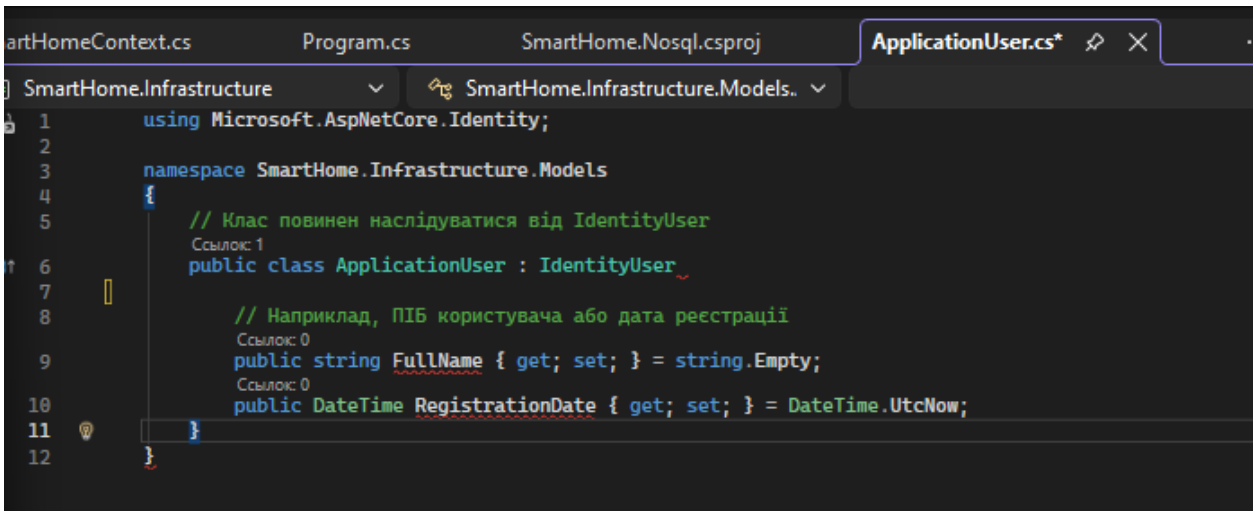
Полтава 2025



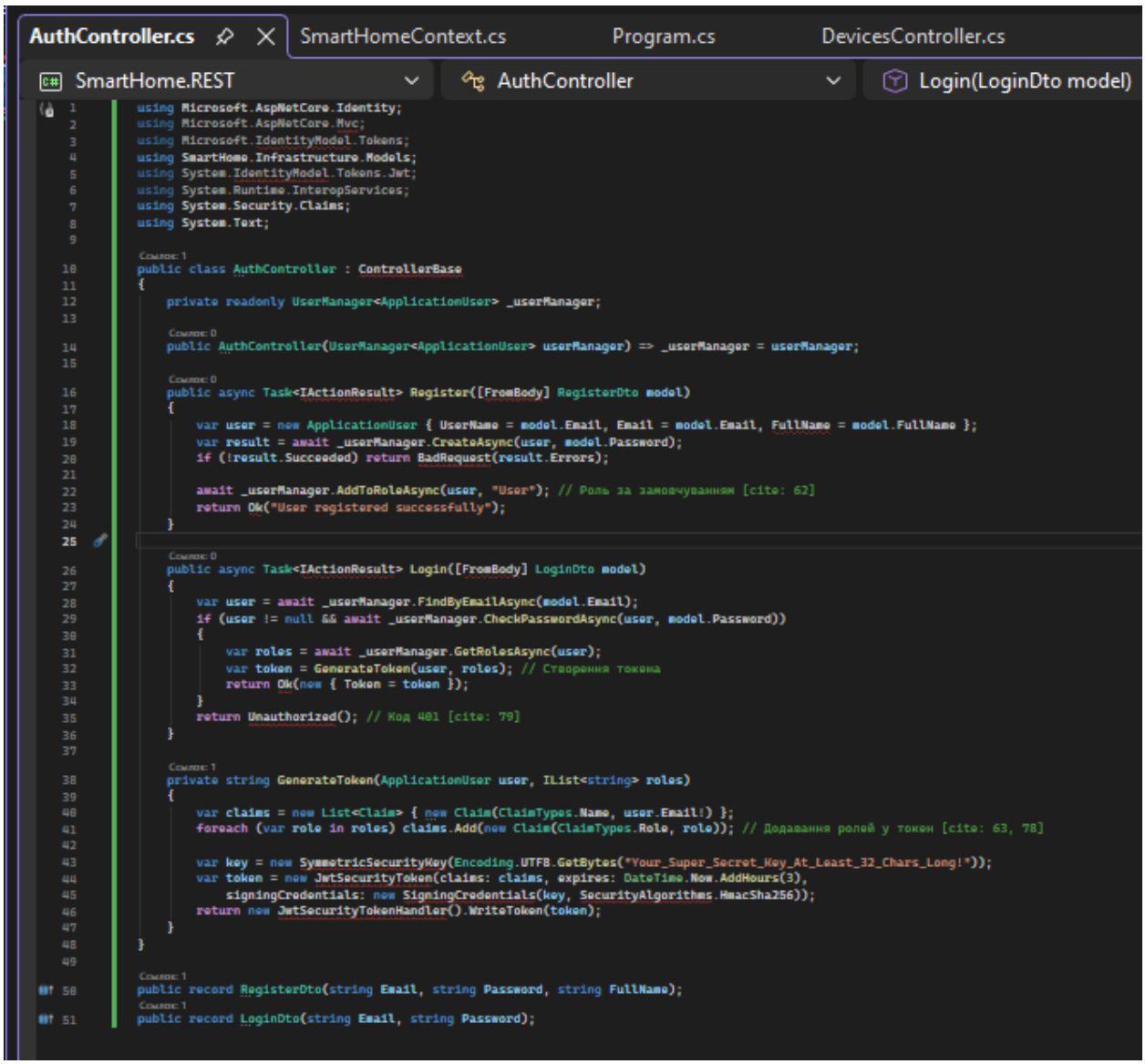
SmartHomeContext.cs | SmartHome.REST.csproj | Program.cs | SmartHome.Nosql.csproj

SmartHome.Infrastructure | SmartHome.Infrastructure.SmartH | OnModelCreating(ModelBuilder bu

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using SmartHome.Infrastructure.Models;

namespace SmartHome.Infrastructure
{
    // Контекст наслідується від IdentityDbContext із вказанням нашої сутності користувача [cite: 57]

    public class SmartHomeContext : IdentityDbContext<ApplicationUser>
    {

        public SmartHomeContext(DbContextOptions<SmartHomeContext> options)
            : base(options)
        {
        }

        // Сутності з вашої тематики (Лабораторні %3 та %4)

        public DbSet<RoomModel> Rooms { get; set; }

        public DbSet<SmartDeviceModel> Devices { get; set; }

        public DbSet<LightDeviceModel> LightDevices { get; set; }

        public DbSet<ClimateDeviceModel> ClimateDevices { get; set; }

        public DbSet<TechnicalPassportModel> TechnicalPassports { get; set; }


        protected override void OnModelCreating(ModelBuilder builder)
        {
            // Обов'язково викликаємо базовий метод для налаштування таблиць Identity [cite: 57]
            base.OnModelCreating(builder);

            // Налаштування TPT (Table-per-Type) наслідування для пристроїв
            builder.Entity<SmartDeviceModel>().ToTable("Devices");
            builder.Entity<LightDeviceModel>().ToTable("LightDevices");
            builder.Entity<ClimateDeviceModel>().ToTable("ClimateDevices");

            // Налаштування зв'язку 1:1 (Технічний паспорт)
            builder.Entity<SmartDeviceModel>()
                .HasOne(d => d.Passport)
                .WithOne(p => p.Device)
                .HasForeignKey<TechnicalPassportModel>(p => p.SmartDeviceModelId);

            // Налаштування зв'язку 1:N (Кімната -> Пристрої)
            builder.Entity<RoomModel>()
                .HasMany(r => r.Devices)
                .WithOne(d => d.Room)
                .HasForeignKey(d => d.RoomId);
        }
    }
}
```




```
// Налаштування Identity
builder.Services.AddIdentity<SmartHomeUser, IdentityRole>()
    .AddEntityFrameworkStores<SmartHomeContext>()
    .AddDefaultTokenProviders();

// Налаштування автентифікації JWT
builder.Services.AddAuthentication(options => {
    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
})
.AddJwtBearer(options => {
    options.TokenValidationParameters = new TokenValidationParameters
    {
        ValidateIssuer = true,
        ValidateAudience = true,
        ValidateLifetime = true,
        IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("Ваш_Дуже_Секретний_Ключ_32_Символи"))
    };
});

Ссылок: 0
public class DevicesController : ControllerBase
{
    // Дозволено всім авторизованим
    [Authorize]
    [HttpGet]

    public async Task<IActionResult> GetAll() { ... }

    // Тільки для Адміна (наприклад, видалення)
    [Authorize(Roles = "Admin")]
    [HttpDelete("{id}")]

    public async Task<IActionResult> Delete(Guid id) { ... }
}
```



Контрольні запитання

1. Для чого в проєкт {Назва тематики}.Infrastructure додається пакет Microsoft.AspNetCore.Identity.EntityFrameworkCore, і які можливості він надає?

Цей пакет додається для інтеграції системи ASP.NET Core Identity з Entity Framework Core. Він надає готові класи для автоматичного створення та керування таблицями користувачів, ролей, токенів та їхніх зв’язків у базі даних.

2. Яким чином сутність користувача (User) успадковується від IdentityUser, та які властивості ви додатково додали до цієї сутності?

Сутність успадковується через стандартне оголошення класу public class User : IdentityUser, що дає доступ до базових полів (логін, пароль). Додатково я додав властивості FullName та RegistrationDate для зберігання персональних даних користувача.

3. Чому DbContext потрібно наслідувати від IdentityDbContext<TUser>, і які зміни це вносить у конфігурацію бази даних?

Наслідування від IdentityDbContext необхідно для того, щоб EF Core знав про необхідність створення системних таблиць Identity (AspNetUsers, AspNetRoles тощо). Це дозволяє контексту автоматично конфігурувати зв'язки між сутностями безпеки та основними моделями програми.

4. Як налаштовуються служби Identity у проєкті {Назва тематики}.REST, зокрема менеджери користувачів (UserManager) та ролей (RoleManager)?

Служби реєструються у файлі Program.cs за допомогою методу AddIdentity<TUser, TRole>(), який підключає менеджери до контейнера залежностей. Це дозволяє впроваджувати UserManager та RoleManager у контролери для створення користувачів та призначення їм ролей.

5. Які кроки потрібні для створення та підписування JWT або Bearer-токенів в ASP.NET Core?

Для цього потрібно створити набір заяв (Claims), визначити час життя токена та підписати його за допомогою секретного ключа (SymmetricSecurityKey). Готовий токен генерується через JwtSecurityTokenHandler і повертається клієнту після успішної автентифікації.

6. Яким чином налаштовується Middleware для автентифікації та авторизації на основі JWT у вашому REST API?

У конвеєр обробки запитів додаються методи app.UseAuthentication() та app.UseAuthorization(). Також налаштовується служба AddJwtBearer, яка перевіряє валідність підпису, видавця та термін дії токена в кожному запиті.

7. Що містить HTTP-заголовок Authorization, і як його обробляє сервер при отриманні запиту?

Заголовок містить тип схеми та сам токен (наприклад, Bearer <token_value>). Сервер витягує токен, перевіряє його валідність і наповнює об’єкт User даними про поточного користувача та його права.

8. Які кінцеві точки REST API залишилися загальнодоступними, а які захищені — та чому саме такий розподіл? Наведіть приклади.

Загальнодоступними залишилися методи перегляду (наприклад, список пристроїв), а захищеними — методи зміни даних (бронювання, видалення). Такий розподіл дозволяє будь-кому бачити інформацію, але обмежує можливість модифікації лише для ідентифікованих осіб.

9. Як ви реалізували поділ повноважень між ролями (мінімум три ролі) та якими атрибутами (наприклад, [Authorize(Roles = "...")]) це забезпечується?

Я створив ролі Admin, Manager та User, призначаючи їх користувачам під час реєстрації. Обмеження доступу реалізовано за допомогою атрибута [Authorize(Roles = "...")] над методами контролерів для перевірки прав клієнта.

10. Як обробляються ситуації, коли користувачі неавторизовані або мають недостатні права?

Які HTTP-коди повертаються в цих випадках? Для неавторизованих користувачів сервер повертає код 401 Unauthorized. Якщо користувач увійшов, але його роль не дозволяє виконати дію, повертається код 403 Forbidden.

11. У чому відмінність між автентифікацією та авторизацією, і як ці процеси взаємодіють у вашому застосунку?

Автентифікація — це процес перевірки особи користувача (хто ви?), а авторизація — перевірка його прав доступу (що вам дозволено?). У застосунку автентифікація видає JWT-токен, а авторизація перевіряє ролі всередині цього токена при кожному запиті.

12. Як ви тестували кінцеві точки реєстрації, автентифікації та авторизованих запитів?

Які приклади потрібно включити в PDF-звіт? Тестування проводилося через Swagger UI та Postman шляхом імітації запитів з токенами та без них. У звіт включені скріншоти успішної реєстрації, входу, а також прикладів доступу дозволених та заборонених запитів.

13. Які кроки виконані для розгортання застосунку у вибраній хмарній/хостинговій платформі, і які сервіси були задіяні? (Додаткове завдання)

Застосунок було розгорнуто на платформі Azure (або аналогічній) шляхом створення Web App та бази даних у хмарі. Використовувалися сервіси Azure App Service для коду та Azure SQL/PostgreSQL для збереження даних.

14. Як ви налаштували перенос налаштувань (connection string, JWT-секрети тощо) у середовище розгортання, та як забезпечили безпеку цих даних? (Додаткове завдання)

Налаштування були перенесені в змінні середовища (Environment Variables) хмарної платформи. Це забезпечує безпеку, оскільки секретні ключі не зберігаються у відкритому вигляді в коді.

15. Які перевірки виконувалися, щоб упевнитися, що розгорнута версія застосунку працює коректно, та які матеріали потрібно додати до звіту? (Додаткове завдання)

Виконувалися тестові виклики API за публічним посиланням для перевірки реєстрації та доступу до бази даних. До звіту додано саме посилання на застосунок та скріншоти панелі керування хмарною інфраструктурою.