

# Laboratorium nr 5

Temat: Tworzenie aplikacji rozproszonych

# 1 Inicjalizacja projektu

Projekt Aspire utworzono komendą:

```
dotnet new aspire -n ShopPlatform
```

Wygenerowana struktura:

```
ShopPlatform/  
  ShopPlatform.AppHost (Orchestrator)  
  ShopPlatform.ServiceDefaults (Wspólne konfiguracje)
```

## 1.1 Aktualizacja do .NET 9.0

Zarówno w `ShopPlatform.AppHost.csproj` jak i `ShopPlatform.ServiceDefaults.csproj` zmieniono:

```
<TargetFramework>net8.0</TargetFramework>
```

na:

```
<TargetFramework>net9.0</TargetFramework>
```

Dostępny SDK: 9.0.307 [C:\Program Files\dotnet\sdk]

## 1.2 Uruchomienie

```
dotnet run --project ShopPlatform.AppHost
```

Aplikacja uruchomiła się pomyślnie. Dashboard dostępny pod:

<http://localhost:18888>

## 1.3 ServiceDefaults

Plik `Extensions.cs` zawiera:

```
public static IServiceCollection AddServiceDefaults(  
    this IServiceCollection services)  
{  
    services.AddOpenTelemetry()  
        .WithTraces(tracing => tracing  
        .AddAspNetCoreInstrumentation()  
        .AddHttpClientInstrumentation())  
        .WithMetrics(metrics => metrics  
        .AddAspNetCoreInstrumentation()  
        .AddHttpClientInstrumentation());  
  
    services.AddHealthChecks();  
    return services;  
}
```

## 1.4 Rola

- **OpenTelemetry:** Obserwabilność (traces, metryki)
- **Health Checks:** Sprawdzanie stanu usług
- **Wspólna konfiguracja:** Do użytku przez wszystkie mikrousługi

## 2 Dodanie API i frontendu

### 2.1 Tworzenie projektów

```
dotnet new webapi -n Shop.Api --framework net9.0
dotnet new blazor -n ShopPlatform.Frontend --interactivity Server --framework net9.0
dotnet sln add Shop.Api/Shop.Api.csproj
dotnet sln add ShopPlatform.Frontend/ShopPlatform.Frontend.csproj
```

### 2.2 Konfiguracja AppHost

```
var builder = DistributedApplication.CreateBuilder(args);

builder.AddProject("api", "../Shop.Api/Shop.Api.csproj");
builder.AddProject("frontend", "../ShopPlatform.Frontend/ShopPlatform.Frontend.csproj");

builder.Build().Run();
```

### 2.3 ServiceDefaults w obydwu projektach

Dodano referencję i w Program.cs:

```
builder.AddServiceDefaults();
```

### 2.4 Dashboard

Po uruchomieniu widoczne są 2 resources:

- **api** – WebAPI na porcie 5001
- **frontend** – Blazor na porcie 5002

**URL:** https://localhost:17178