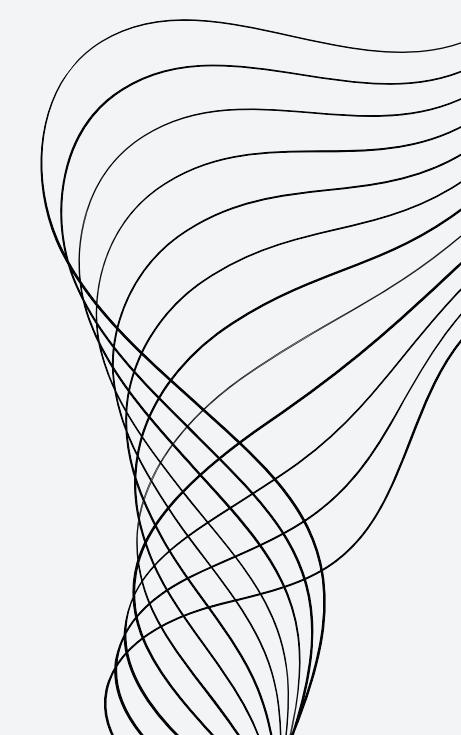


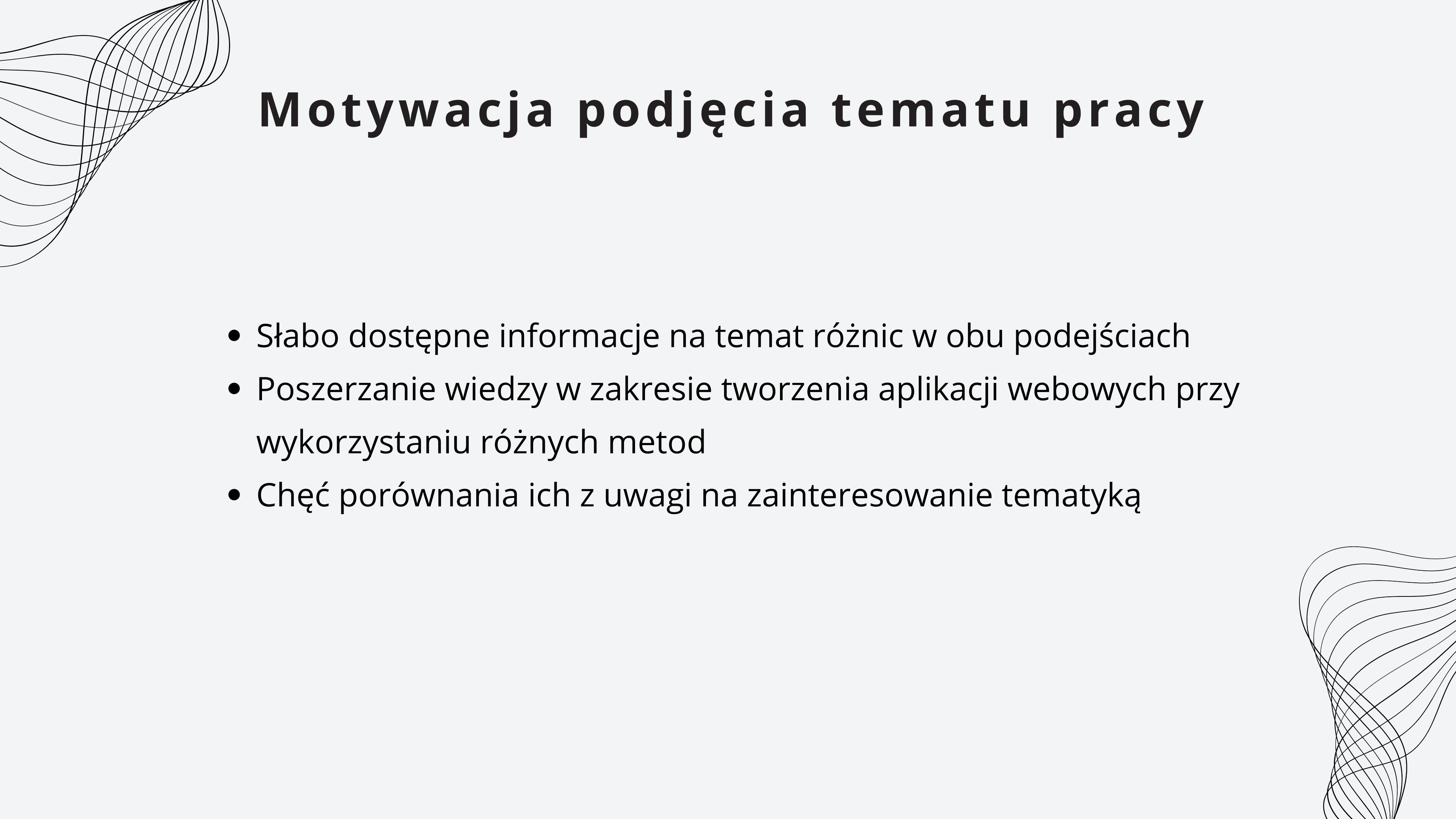
Praca Magisterska

PORÓWNANIE APLIKACJI WEBOWYCH TYPU API + KLIENT ORAZ APLIKACJI MVC



Marcin Drelewski

Promotor: dr Wojciech Gwizdała



Motywacja podjęcia tematu pracy

- Słabo dostępne informacje na temat różnic w obu podejściach
- Poszerzanie wiedzy w zakresie tworzenia aplikacji webowych przy wykorzystaniu różnych metod
- Chęć porównania ich z uwagi na zainteresowanie tematyką

Cel Pracy

Zaprezentowanie i omówienie wybranych różnic w obu podejściach do tworzenia aplikacji internetowych na przykładzie aplikacji podzielonej na warstwę serwerową, opartą o platformę .Net oraz warstwę kliencką na platformie Angular oraz aplikacji MVC stworzonej przy wykorzystaniu nowoczesnej platformy Microsoftu o nazwie Blazor

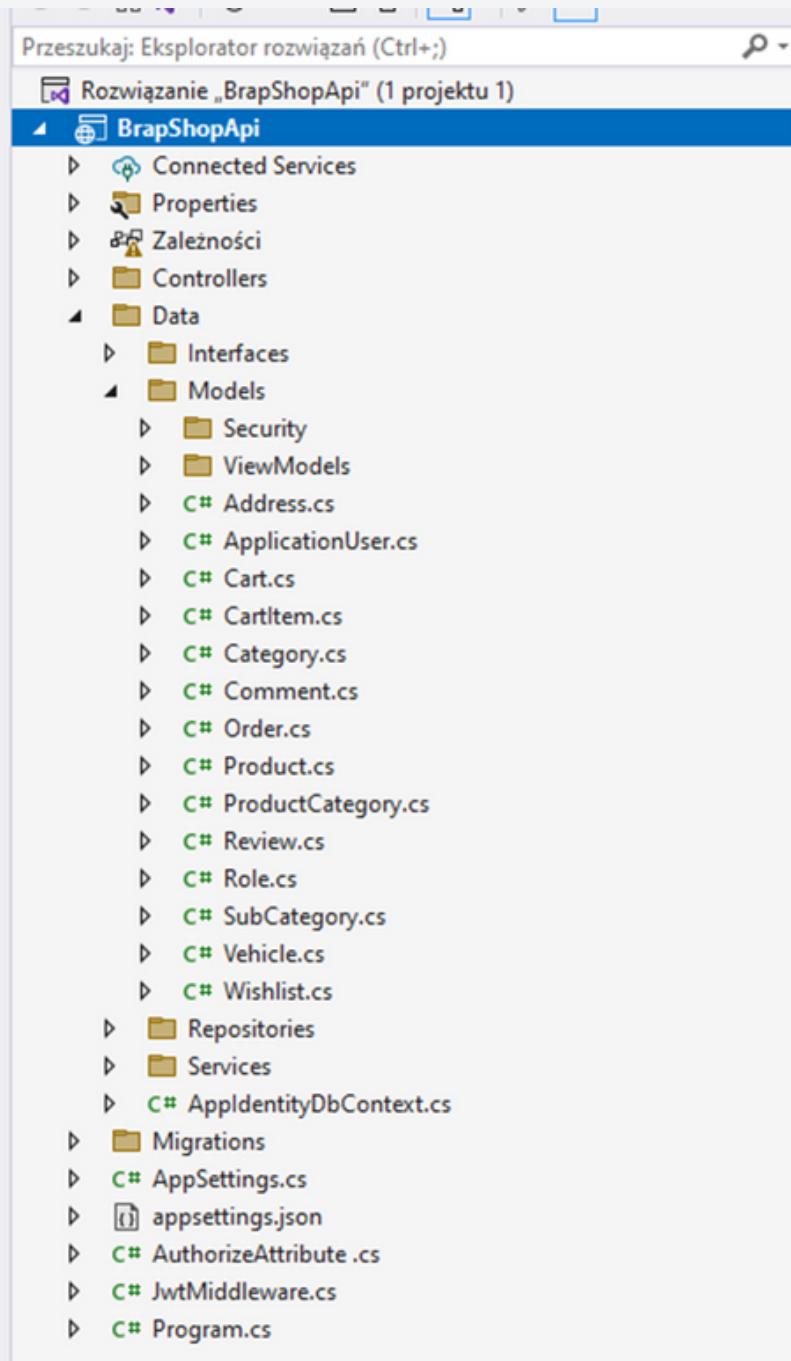


Zastosowane Technologie

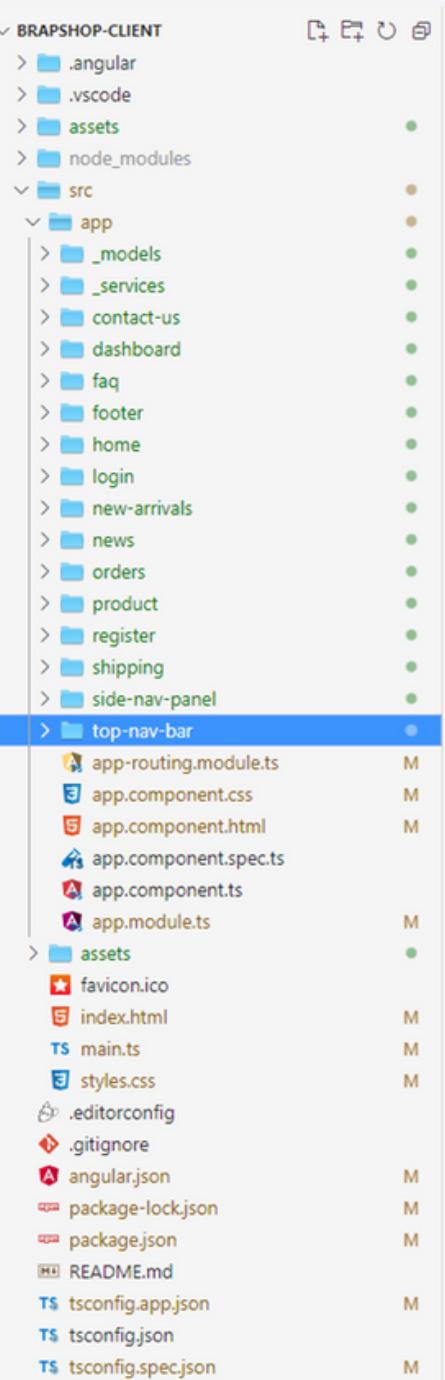
- .NET
- Angular
- Blazor
- Linq
- Entity Framework Core
- Bootstrap

Struktura aplikacji

API

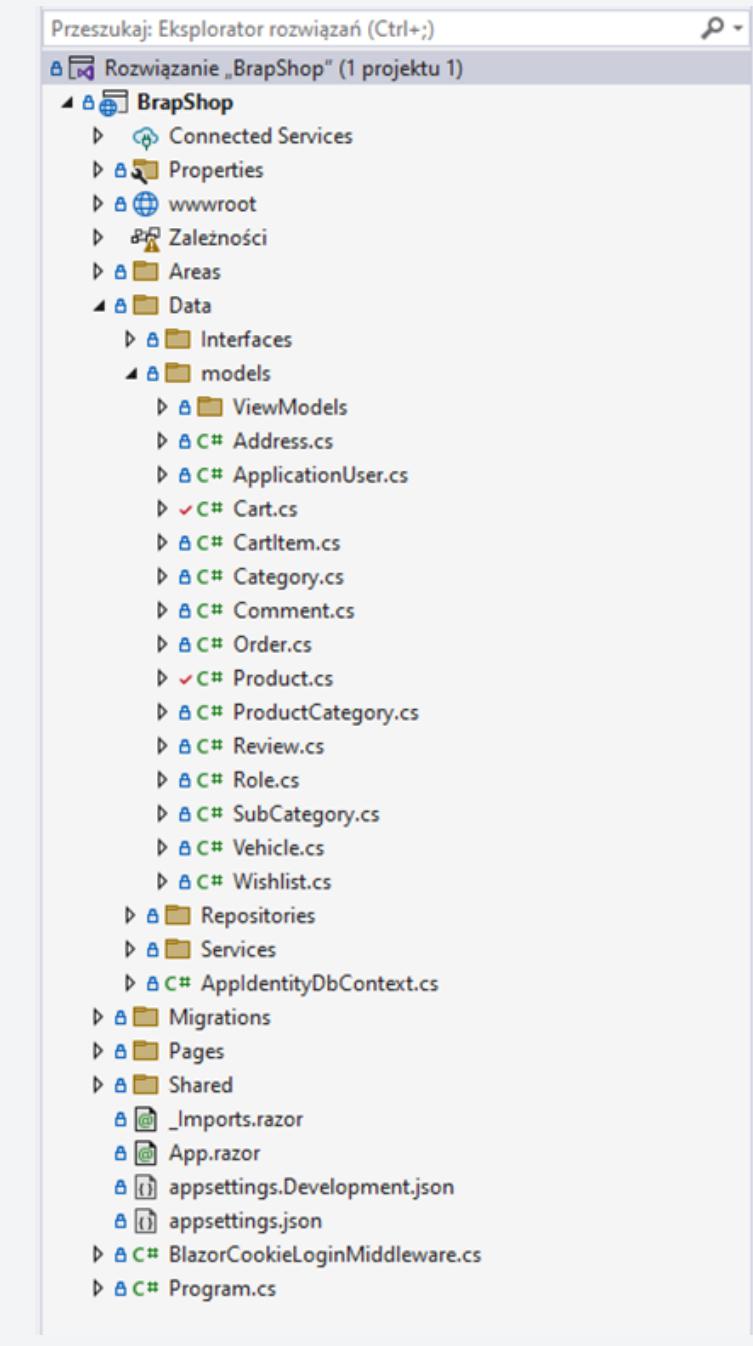


Klient



VS

Blazor



Plik Programs.cs

API

```
7  var builder = WebApplication.CreateBuilder(args);
8
9  var connectionStrings = new Dictionary<string, string>()
10 {
11     { "identity", builder.Configuration.GetConnectionString("IdentityDBConnection") },
12     { "data", builder.Configuration.GetConnectionString("DataDBConnection") }
13 };
14
15 builder.Services.AddDbContext<AppIdentityDbContext>(options =>
16     options.UseSqlServer(connectionStrings["identity"]));
17 builder.Services.AddControllers();
18 builder.Services.AddScoped < ICartRepository, CartRepository > ();
19 builder.Services.AddScoped < IUserRepository, UserRepository > ();
20 builder.Services.AddScoped<ITokenService, TokenService>();
21 builder.Services.AddScoped<IUserService, UserService>();
22 builder.Services.AddScoped<ProductService>();
23 builder.Services.AddScoped<CategoryService>();
24 builder.Services.AddScoped<CartService>();
25
26 builder.Services.AddCors(options =>
27 {
28     options.AddDefaultPolicy(builder =>
29     {
30         builder.AllowAnyOrigin()
31             .AllowAnyMethod()
32             .AllowAnyHeader();
33     });
34 });
35 );
36
37 var app = builder.Build();
38
39 app.UseAuthorization();
40 app.MapControllers();
41 app.UseCors();
42
43 app.Run();
44
```

Blazor

```
15 var builder = WebApplication.CreateBuilder(args);
16
17 var connectionStrings = new Dictionary<string, string>()
18 {
19     { "identity", builder.Configuration.GetConnectionString("IdentityDBConnection") },
20     { "data", builder.Configuration.GetConnectionString("DataDBConnection") }
21 };
22
23 builder.Services.AddDbContext<AppIdentityDbContext>(options =>
24     options.UseSqlServer(connectionStrings["identity"]));
25 builder.Services.AddDatabaseDeveloperPageExceptionFilter();
26 builder.Services.AddDefaultIdentity<ApplicationUser>(options => options.SignIn.RequireConfirmedAccount = true)
27     .AddRoles<IdentityRole>()
28     .AddEntityFrameworkStores<AppIdentityDbContext>().AddDefaultUI();
29 builder.Services.AddRazorPages();
30 builder.Services.AddServerSideBlazor();
31 builder.Services.AddScoped<AuthenticationStateProvider, RevalidatingIdentityAuthenticationStateProvider<IdentityUser>>();
32 builder.Services.AddScoped<ProductService>();
33 builder.Services.AddScoped<CategoryService>();
34 builder.Services.AddScoped<CartService>();
35 builder.Services.AddScoped<ICartRepository, CartRepository>();
36 builder.Services.AddScoped<IUserRepository, UserRepository>();
37 builder.Services
38     .AddBlazorise(options =>
39     {
40         options.Immediate = true;
41     })
42     .AddBootstrapProviders()
43     .AddFontAwesomeIcons();
44
45 var app = builder.Build();
46
47 if (app.Environment.IsDevelopment())
48 {
49     app.UseMigrationsEndPoint();
50 }
51 else
52 {
53     app.UseExceptionHandler("/Error");
54     app.UseHsts();
55 }
56
57 app.UseMiddleware<BlazorCookieLoginMiddleware>();
58 app.UseHttpsRedirection();
59 app.UseStaticFiles();
60 app.UseRouting();
61 app.UseAuthentication();
62 app.UseAuthorization();
63 app.MapControllers();
64 app.MapBlazorHub();
65 app.MapFallbackToPage("/_Host");
66
67 app.Run();
68
```

VS

Własna aplikacja wykorzystana w pracy

The screenshot shows a web-based application for a car parts store. At the top, there's a navigation bar with links for BrapShop, Shipping, FAQ, NEW!, Orders, and Contact Us. Below the navigation is a header section featuring a red Mazda RX-7 with its headlights on. To the left of the car is a large circular image of a flywheel ring gear. On the right side of the car, there's some placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum." Below this, there's a "Categories" sidebar listing various car parts categories like Block Off Plates, Engine Management, Hoses, etc. In the center, there's a "Featured Products" section showing three placeholder camera icons. At the bottom, there's a product listing for "79011 Rx7 & Rx8 Flywheel Ring Gear Brake (ARE923)" priced at \$20, with a shopping cart and heart icon.

This screenshot shows a modal dialog box titled "New account" overlaid on the same car parts store interface. The dialog contains fields for "Email" (test@test.com), "Password" (a masked password), and "Confirm password". A blue "Create" button is at the bottom. In the background, the login form is visible with fields for "Email" (test@test.com) and "Password" (a masked password), along with a "Login" button and a "Not a member? Create Account" link. The overall design is clean and modern, using a light gray color scheme.

Możliwości dalszego rozwoju

W przyszłości można rozszerzyć pracę nad różnicami między tymi, jak widać, mocno odmiennymi podejściami, o badania nad wydajnością obu aplikacji bądź dodać do porównania również aplikację opartą o architekturę mikroserwisów.