Uniwersytet Bielsko-Bialski

LABORATORIUM

Programowanie dla Internetu w technologii ASP.NET

Sprawozdanie nr 5

ViewModel

GRUPA: 2B / SEMESTR: 6 / ROK: 3

Igor Gawłowicz / 59096

Warstwa ViewModelu

Na ostatnich zajęciach poruszyliśmy dodanie warstwy ViewModelu do aplikacji aby oddzielić fizycznie Front strony od bezpośredniego dostępu do bazy danych.

Zrobiliśmy to poprzez dodanie ViewModeli do plików gdzie korzystamy z konkretnych danych, np listujemy je lub wykorzystujemy w dowolny sposób.

Aby to dokonać musieliśmy zbadać które widoki mają bezpośredni dostęp do danych jak na przykład ta widok dla panelu klientów.

```
@model IEnumerable<TripApp.Models.Client>

<h1>Existing Clients</h1>

<table class="table">
    <thead>
        <tr>
            <th>
                Client ID
            </th>
            <th>
                Name
            </th>
            <th>
                Email
            </th>
            <th>
                Phone
            </th>
        </tr>
    </thead>
    <tbody>
        @foreach (var client in Model)
        {
            <tr>
                <td>@client.ClientId</td>
                <td>@client.Name</td>
                <td>@client.Email</td>
                <td>@client.Phone</td>
            </tr>
        }
    </tbody>
</table>
```

Rozdzieliliśmy go poprzez utworzenie nowego repozytorium `ViewModels` i utworzeniu tam pliku `ClientListViewModel`

```

```

```
namespace TripApp.ViewModels
{
    public class ClientListViewModel
    {
        public int ClientId { get; set; }
        public string Name { get; set; }
        public string Email { get; set; }
        public string Phone { get; set; }
    }
}
```

Następnie zmodyfikowaliśmy nasz widok aby korzystał teraz z ViewModelu zamiast bezpośrednio Modelu

```
@model IEnumerable<TripApp.ViewModels.ClientListViewModel>

<h1>Existing Clients</h1>

<table class="table">
    <thead>
        <tr>
            <th>
                Client ID
            </th>
            <th>
                Name
            </th>
            <th>
                Email
            </th>
            <th>
                Phone
            </th>
        </tr>
    </thead>
    <tbody>
        @foreach (var client in Model)
        {
            <tr>
                <td>@client.ClientId</td>
                <td>@client.Name</td>
                <td>@client.Email</td>
                <td>@client.Phone</td>
            </tr>
        }
    </tbody>
</table>
```

Po czym musieliśmy zmodyfikować kontroler, aby w odpowiedni sposób zajął się danymi, czyli przypisał dane pobrane z modelu do struktury we ViewModelu

przed

```
using Microsoft.AspNetCore.Mvc;
using System.Linq;
using TripApp.Data;

namespace TripApp.Controllers
{
    public class ClientController : Controller
    {
        private readonly TripContext _context;

        public ClientController(TripContext context)
        {
            _context = context;
        }

        // GET: Clients
        public IActionResult Index()
        {
            var clients = _context.Clients.ToList();
            return View(clients);
        }
    }
}
```

po

```
using Microsoft.AspNetCore.Mvc;
using System.Linq;
using TripApp.Data;
using TripApp.ViewModels;

namespace TripApp.Controllers
{
    public class ClientController : Controller
    {
        private readonly TripContext _context;

        public ClientController(TripContext context)
        {
            _context = context;
        }

        // GET: Clients
        public IActionResult Index()
        {
            var clientListViewModel = _context.Clients.Select(client => new
ClientListViewModel
```

```
            {
                ClientId = client.ClientId,
                Name = client.Name,
                Email = client.Email,
                Phone = client.Phone
            });

            return View(clientListViewModel);
        }
    }
}
```

W ten sposób strona zachowuje tą samą funkcjonalność z zewnątrz a jest bezpieczniejsza wewnątrz.

Aby dokończyć zadanie powtórzyliśmy ten sam proces dla pozostałych widoków, które wykorzystywały Modele.

```
@model List<TripApp.ViewModels.TripSummaryViewModel>

<h1>Trip List</h1>

<table class="table">
    <thead>
        <tr>
            <th>
                Destination
            </th>
            <th>
                Start Date
            </th>
            <th>
                End Date
            </th>
            <th>
                Price
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var trip in Model)
        {
            <tr>
                <td>
                    @trip.Destination
                </td>
                <td>
                    @trip.TripDateStart.ToString("dd/MM/yyyy")
                </td>
                <td>
                    @trip.TripDateEnd.ToString("dd/MM/yyyy")
```

```
                </td>
                <td>
                    @trip.Price.ToString("C")
                </td>
                <td>
                    <a href="@Url.Action("Create", "ClientReservation", new {
tripId = trip.TripId })">Create Reservation</a>
                </td>
            </tr>
        }
    </tbody>
</table>
```

```
namespace TripApp.ViewModels
{
    public class TripSummaryViewModel
    {
        public int TripId { get; set; }
        public string Destination { get; set; }
        public DateTime TripDateStart { get; set; }
        public DateTime TripDateEnd { get; set; }
        public decimal Price { get; set; }
    }
}
```

```
using BikeRentalSystemWeb.Data; // Assuming this namespace is correct
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Diagnostics;
using TripApp.Data;
using TripApp.ViewModels;

namespace TripApp.Controllers
{
    public class HomeController : Controller
    {
        private readonly ILogger<HomeController> _logger;

        private readonly TripContext _context;

        public HomeController(ILogger<HomeController> logger, TripContext context)
        {
            _logger = logger;
            _context = context;
            _context.Database.EnsureCreated();
            DbInitializer.Initialize(context);
        }
```

```
        public async Task<IActionResult> Index()
        {
            var trips = await _context.Trips
                .Select(trip => new TripSummaryViewModel
                {
                    TripId = trip.TripId,
                    Destination = trip.Destination,
                    TripDateStart = trip.TripDateStart,
                    TripDateEnd = trip.TripDateEnd,
                    Price = trip.Price
                })
                .ToListAsync();

            return View(trips);
        }

    }
}
```

```cs
@model IEnumerable<TripApp.ViewModels.ReservationViewModel>

<h1>Reservations</h1>

<table class="table">
    <thead>
        <tr>
            <th>
                Reservation ID
            </th>
            <th>
                Client Name
            </th>
            <th>
                Trip Destination
            </th>
            <th>
                Reservation Date
            </th>
        </tr>
    </thead>
    <tbody>
        @foreach (var reservation in Model)
        {
            <tr>
                <td>@reservation.ReservationId</td>
                <td>@reservation.ClientId</td>
                <td>@reservation.TripId</td>
                <td>@reservation.ReservationDate.ToString("dd/MM/yyyy") </td>
            </tr>
        }
    </tbody>
</table>
```

```
namespace TripApp.ViewModels
{
    public class ReservationViewModel
    {
        public int ReservationId { get; set; }
        public int ClientId { get; set; }
        public int TripId { get; set; }
        public DateTime ReservationDate { get; set; }
    }

}
```

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Linq;
using TripApp.Data;
using TripApp.ViewModels;

namespace TripApp.Controllers
{
    public class ReservationController : Controller
    {
        private readonly TripContext _context;

        public ReservationController(TripContext context)
        {
            _context = context;
        }

        // GET: Reservations
        public async Task<IActionResult> Index()
        {
            var reservations = await _context.Reservations
                .Select(reservation => new ReservationViewModel
                {
                    ReservationId = reservation.ReservationId,
                    ReservationDate = reservation.ReservationDate,
                    ClientId = reservation.ClientId,
                    TripId = reservation.TripId
                })
                .ToListAsync();

            return View(reservations);
        }
    }
}
```

Wnioski

Warstwa ViewModelu może wydawać się zbyteczna ponieważ nie wpływa znacząco na kod ani na expierience użytkownika, ale zdecydowanie wpływa na bezpieczeństwo danych będących w obiegu na stronie.