LABORATORY ASSIGNMENT №3

Partial Views. Navigation. Filtering. Sessions

Goal: Gain skills in working with partial views, learn to work with query string parameters, and acquire skills in working with sessions.

Workflow:

Task 1. Create Partial View

First I've created partial view, I it looks like event card.

```
@model TicketBookingWebsite.Models.Event

<div class="card mb-3 shadow-sm">
    <div class="card-body">
        <h5 class="card-title">@Model.Name</h5>
        <p class="card-text text-muted">
            <i class="bi bi-calendar-event"></i> @Model.Date.ToString("dd MMMM, yyyy")
<br>
            <i class="bi bi-geo-alt"></i> @Model.Location
        </p>
        <p class="card-text">@Model.Description</p>
        <p class="card-text fw-bold text-success">
            <i class="bi bi-ticket"></i> @Model.AvailableTickets tickets available
        </p>
        <p class="card-text">
            <span class="badge bg-primary fs-6">$@Model.Price</span>
        </p>
    </div>
</div>
```

Next step was importing it to page, so I can display it. Here is how I modified my Index.cshtml:

```
@model TicketBookingWebsite.ViewModels.EventListViewModel
@using System.Globalization

<div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">
    @foreach (var e in Model.Events)
    {
        <div class="flex-grow-1" style="min-width: 300px; max-width: 400px;">
            <partial name="EventSummary" model="e" />
        </div>
```

```
        }
    </div>

    <!-- Pagination Controls -->
    <nav class="mt-4">
        <ul class="pagination justify-content-center">
            @for (int i = 1; i <= Model.PagingInfo.TotalPages; i++)
            {
                <li class="page-item @(i == Model.PagingInfo.CurrentPage ? "active" : "")">
                    <a class="page-link" asp-action="Index" asp-route-page="@i">@i</a>
                </li>
            }
        </ul>
    </nav>
```

Task 2. Category Navigation

First I've created NavigationMenuViewComponent.cs:

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

namespace TicketBookingWebsite.Components
{
    public class NavigationMenuViewComponent : ViewComponent
    {
        private readonly WebsiteDbContext _context;

        public NavigationMenuViewComponent(WebsiteDbContext context)
        {
            _context = context;
        }

        public IViewComponentResult Invoke()
        {
            var categories = _context.Events
                .Select(e => e.Location)
                .Distinct()
                .OrderBy(c => c)
                .ToList();

            return View(categories);
        }
    }
}
```

Then I've created the component is Views/Shared/Components/NavigationMenu/Default.cshtml:

```
<div class="sidebar-sticky">
    <h4 class="text-primary">Categories</h4>
    <ul class="nav flex-column">
        <li class="nav-item">
```

```html
            <a class="nav-link text-dark fw-bold" asp-controller="Home" asp-
action="Index" asp-route-category="">
                All Events
            </a>
        </li>
        @foreach (var category in Model)
        {
            <li class="nav-item">
                <a class="nav-link text-dark" asp-controller="Home" asp-action="Index"
asp-route-category="@category">
                    @category
                </a>
            </li>
        }
    </ul>
</div>
```

Added it to _Layout.chtml and modified the controllers Index method:

```csharp
public IActionResult Index(string? category, int page = 1)
{
    var filteredEvents = string.IsNullOrEmpty(category)
        ? repository.Events
        : repository.Events.Where(e => e.Location == category);

    var totalEvents = filteredEvents.Count();

    var events = filteredEvents
        .OrderBy(e => e.Date)
        .Skip((page - 1) * PageSize)
        .Take(PageSize);

    var viewModel = new EventListViewModel
    {
        Events = events,
        PagingInfo = new PagingInfo
        {
            CurrentPage = page,
            TotalItems = totalEvents,
            ItemsPerPage = PageSize
        },
        CurrentCategory = category
    };

    return View(viewModel);
}
```
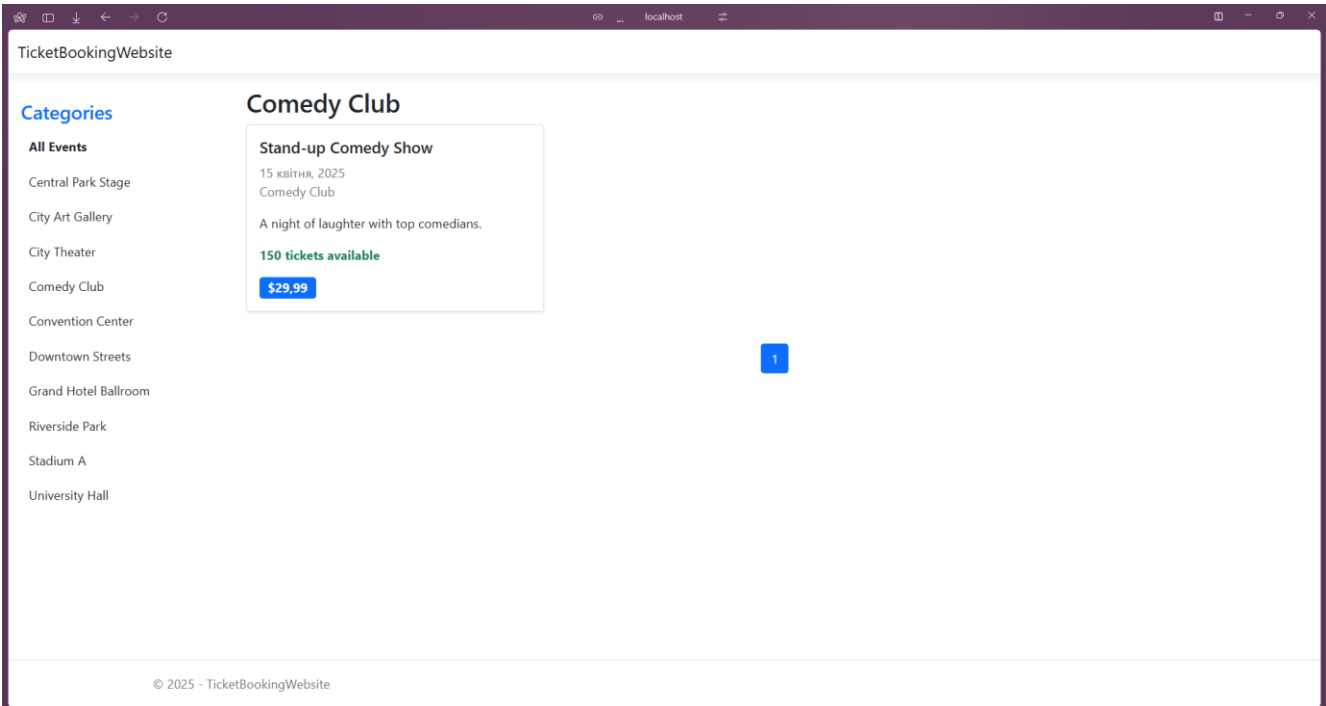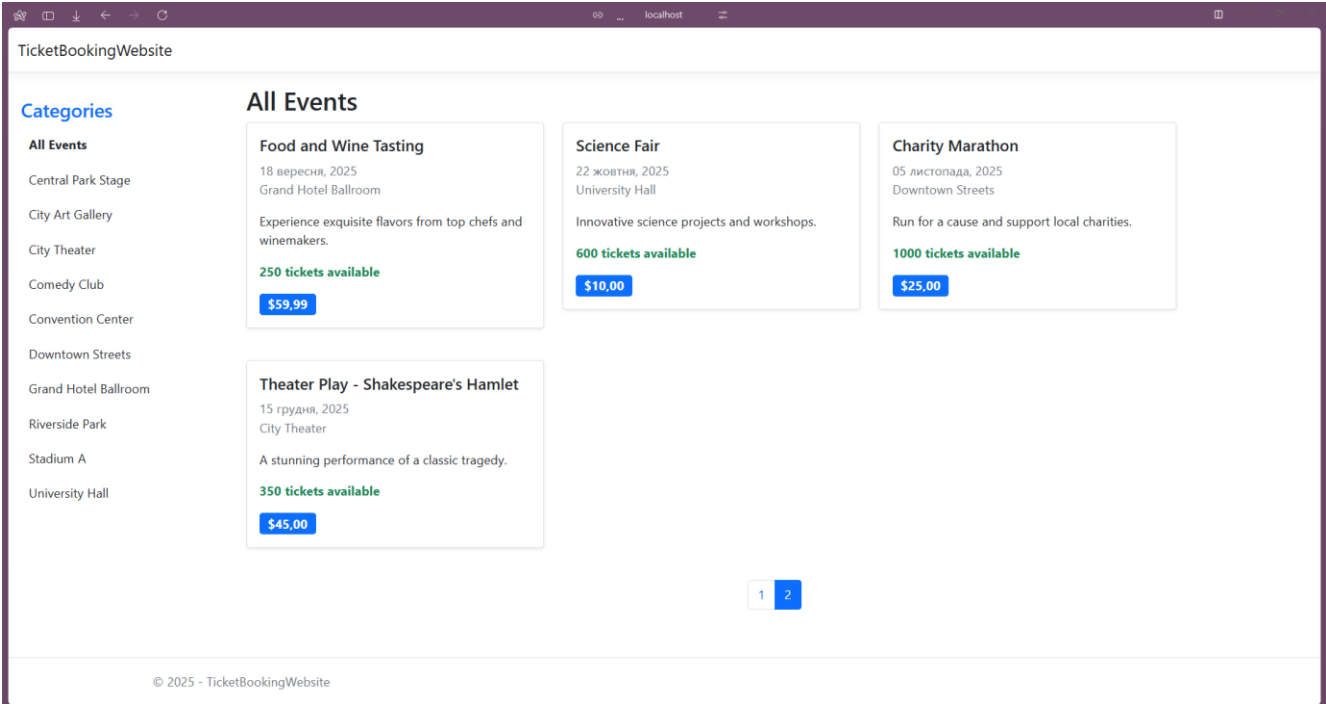
Here is the result:

Task 3. Sessions

Number on group list	Project Name	Feature Description
1	Event Ticket Booking Website	Use sessions to temporarily store selected event tickets before checkout, preventing loss of selection if the user navigates away.

Firstly I added this lines to Program.cs to include sessions:

```
builder.Services.AddDistributedMemoryCache();
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});


app.UseSession();
```

Then I created SessionHelper:

```
using Microsoft.AspNetCore.Http;
using Newtonsoft.Json;

public static class SessionHelper
{
    public static void SetObjectAsJson(this ISession session, string key, object value)
    {
        session.SetString(key, JsonConvert.SerializeObject(value));
    }

    public static T GetObjectFromJson<T>(this ISession session, string key)
    {
        var value = session.GetString(key);
        return value == null ? default(T) : JsonConvert.DeserializeObject<T>(value);
    }
}
```

To save something in session, firstly I need to create thing that I will save. So I decided to create Cart, where all selected event tickets will be stored before checkout.

To start I've created CartViewModel and cart page:

```csharp
using TicketBookingWebsite.Models;

namespace TicketBookingWebsite.ViewModels
{
    public class CartViewModel
    {
        public List<Event> Events { get; set; }
        public decimal TotalPrice { get; set; }
    }

}
```

```cshtml
@model TicketBookingWebsite.ViewModels.CartViewModel

<h2>Your Cart</h2>

@if (Model.Events != null && Model.Events.Count > 0)
{
    <table class="table">
        <thead>
            <tr>
                <th>Event</th>
                <th>Price</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var item in Model.Events)
            {
                <tr>
                    <td>@item.Name</td>
                    <td>@item.Price.ToString("C")</td>
                </tr>
            }
        </tbody>
    </table>

    <h3>Total: @Model.TotalPrice.ToString("C")</h3>

    <a href="@Url.Action("ClearCart")" class="btn btn-danger">Clear Cart</a>
    <a href="@Url.Action("Checkout")" class="btn btn-primary">Proceed to Checkout</a>
}
else
{
    <p>Your cart is empty.</p>
}
```
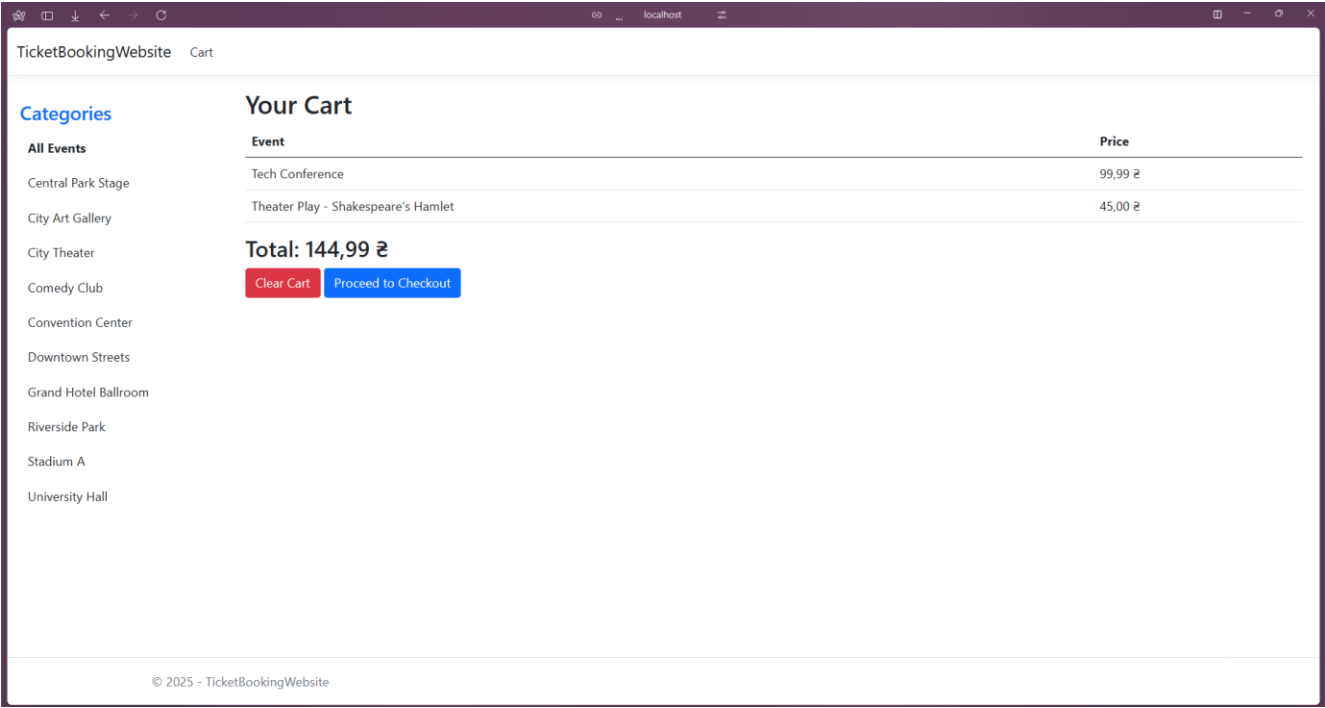
Added Add to Cart button in event summary:

```html
<a href="@Url.Action("AddToCart", "Home", new { eventId = Model.Id })" class="btn btn-primary">
    Add to Cart
</a>
```
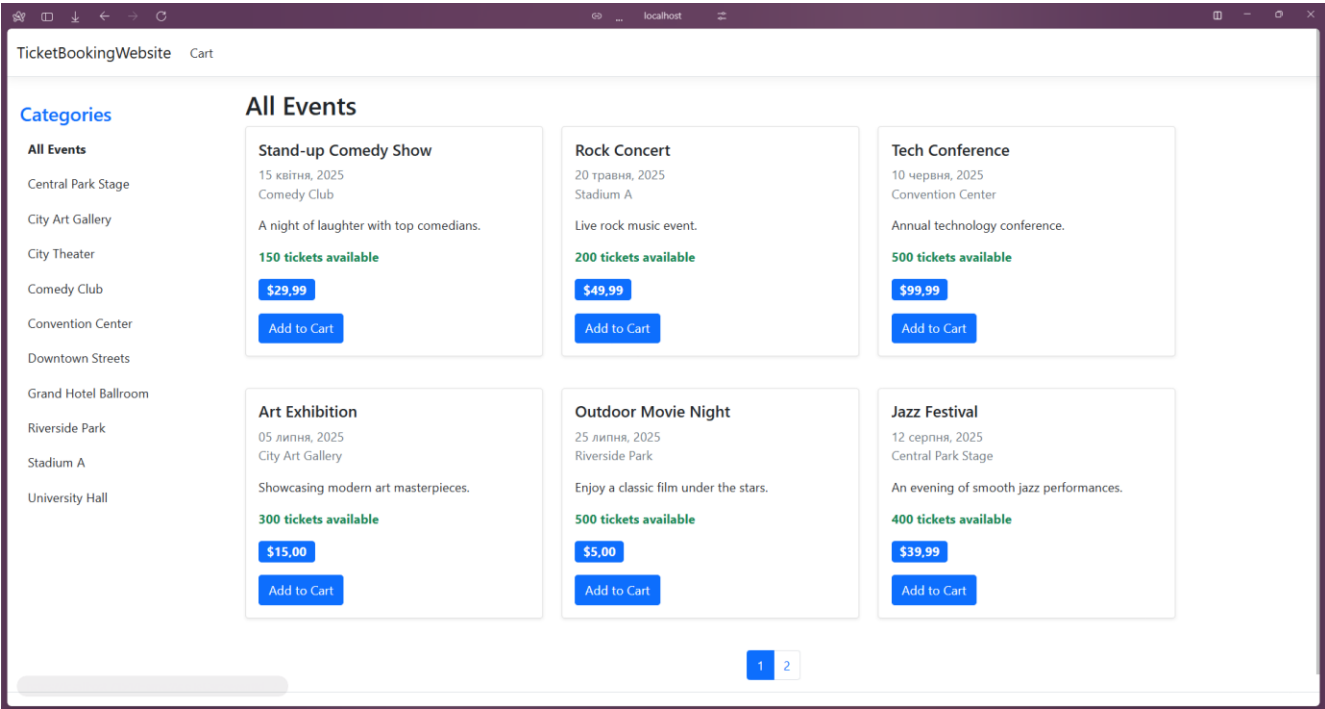
Modified HomeController, added handlers for Cart:

```csharp
    public IActionResult AddToCart(int eventId)
    {
        var selectedEvent = repository.Events.FirstOrDefault(e => e.Id == eventId);
        if (selectedEvent != null)
        {
            var cart = HttpContext.Session.GetObjectFromJson<List<Event>>("Cart") ?? new List<Event>();
            cart.Add(selectedEvent);
            HttpContext.Session.SetObjectAsJson("Cart", cart);
        }

        return RedirectToAction("Index");
    }

    public IActionResult Cart()
    {
        var cart = HttpContext.Session.GetObjectFromJson<List<Event>>("Cart");
        var viewModel = new CartViewModel
        {
            Events = cart,
            TotalPrice = cart?.Sum(e => e.Price) ?? 0
        };

        return View(viewModel);
    }

    public IActionResult ClearCart()
    {
        HttpContext.Session.Remove("Cart");
        return RedirectToAction("Cart");
    }
```

Result:

Conclusion: I gained skills in working with partial views, learned how to work with query string parameters, and acquired skills in working with sessions.