ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Кафедра інженерії програмного забезпечення

ЛАБОРАТОРНА РОБОТА №6

з дисципліни «Програмування Інтернет»

.(назва дисципліни)

на тему: «Введення в мультимовні сайти. Ресурси. Логіка

мультимовного сайту. Фільтр локалізації»

ЛРІПЗ.2101075.01.06.ЛР

(шифр, назва)

Рівень вищої освіти                     Перший (бакалаврський)

Галузь знань                      12 «Інформаційні технології»

Спеціальність     121 «Інженерія програмного забезпечення»

Освітня програма   «Інженерія програмного забезпечення»

Студента. ІІI  курсу, група . ІПЗ-21-1. \_\_\_\_\_\_\_\_\_\_\_ Д.І. Кваснюк.

            (шифр)                         (підпис)     (ініціали, прізвище)

Викладач канд. техн. наук, доцент  .   \_\_\_\_\_\_\_\_\_\_\_ О.М. Яшина

              (посада, вчене звання, науковий ступінь)         (підпис)   (ініціали, прізвище)

Кількість балів \_\_\_\_\_\_\_

Оцінка за шкалою:

національною\_\_\_\_\_\_\_\_\_/ЄКТС\_\_\_\_

Хмельницький, 2023

ЗВІТ

**Варіант 6**

**Мета:** Отримати навики розробки веб-додатків з використанням мультимовних сайтів.

**ХІД РОБОТИ**

**Завдання №1.** Реалізувати логіку мультимовного сайту та фільтр локалізації для Web – додатка.

Для виконання цієї лабораторної роботи використовуватимемо бібліотеку XLocalizer ([посилання](https://github.com/LazZiya/XLocalizer)), яка забезпечує автоматичний переклад необхідного контенту на сайті шляхом переводу слів на сторонньому сервісі.

Для початку встановимо необхідні пакети, які забезпечуватимуть коректну роботу бібліотеки. Далі виконаємо початкову конфігурацію сервісу для локалізації сайту, для цього додамо код до Program.cs

|  |
| --- |
| builder.Services.AddControllersWithViews()  .AddXLocalizer<LocSource, MyMemoryTranslateService>(ops =>  {  ops.ResourcesPath = "LocalizationResources";  ops.AutoAddKeys = true;  ops.AutoTranslate = true;  ops.TranslateFromCulture = "en";  ops.UseExpressMemoryCache = true;  })  .AddViewLocalization(LanguageViewLocationExpanderFormat.Suffix)  .AddDataAnnotationsLocalization(options => {  options.DataAnnotationLocalizerProvider = (type, factory) =>  factory.Create(typeof(LangResource));  });  builder.Services.Configure<RequestLocalizationOptions>(ops =>  {  var cultures = new CultureInfo[] {  new CultureInfo("en"),  new CultureInfo("uk"),  }; ops.SupportedCultures = cultures;  ops.SupportedUICultures = cultures;  ops.DefaultRequestCulture = new RequestCulture("en");  ops.RequestCultureProviders.Insert(0, new RouteSegmentRequestCultureProvider(cultures));  });  builder.Services.AddSingleton<IXResourceProvider, XmlResourceProvider>();  builder.Services.AddRazorPages();  builder.Services.AddHttpClient<ITranslator, MyMemoryTranslateService>();  var app = builder.Build();  app.UseRequestLocalization(); |

Наступним кроком стане визначення налаштувань для автоматичного переводу. Для цього у відповідному сервісі згенеруємо API- ключ, який потім вкажемо appsettings.json:

|  |
| --- |
| {  "XLocalizer.Translate": {  "MyMemory": {  "Email": "...",  "Key": "..."  }  }  } |

Далі додамо відповідний навігаційний елемент для зміни мови, де визначимо необхідні атрибути.

|  |
| --- |
| <li class="drop with--one--item">  <**language-nav** **redirect-to-url**="@(Url.RouteUrl(routeData))" style="display: flex;align-items: center;" **cookie-handler-url**="@Url.Action("SetCultureCookie", "Home", new { cltr = routeData["culture"] })">  </**language-nav**>  </li> |

І для відповідної зміни локалізації певних елементів сайту ми можемо використовувати html-tag “ **localize-content** ” або “<**localize**></**localize**>” .

|  |
| --- |
| <li><**a** **localize-content** href="my-account.html">My Account</**a**></li>  <li><**a** **localize-content** href="cart.html">Cart Page</**a**></li>  <li><**a** **localize-content** href="checkout.html">Checkout Page</**a**></li>  <li><**a** **localize-content** href="wishlist.html">Wishlist Page</**a**></li>  <li><**a** **localize-content** href="error404.html">404 Page</**a**></li>  <li><**a** **localize-content** href="faq.html">Faq Page</**a**></li>  <li><**a** **localize-content** href="team.html">Team Page</**a**></li>  <**localize**>  <h2>Exchange <span>your </span></h2>  <h2>used <span>Book </span></h2>  <h2>from <span>Here </span></h2>  </**localize**> |

Також для збереження обраної локалізації у кукі створимо метод.

|  |
| --- |
| public IActionResult SetCultureCookie(string cltr)  {  Response.Cookies.Append(  CookieRequestCultureProvider.DefaultCookieName,  CookieRequestCultureProvider.MakeCookieValue(new RequestCulture(cltr)),  new CookieOptions { Expires = DateTimeOffset.UtcNow.AddYears(1) }  );    return LocalRedirect("/");  } |

Таким чином результати роботи відповідного функціоналу наявний на **Рисунках 1-4.**

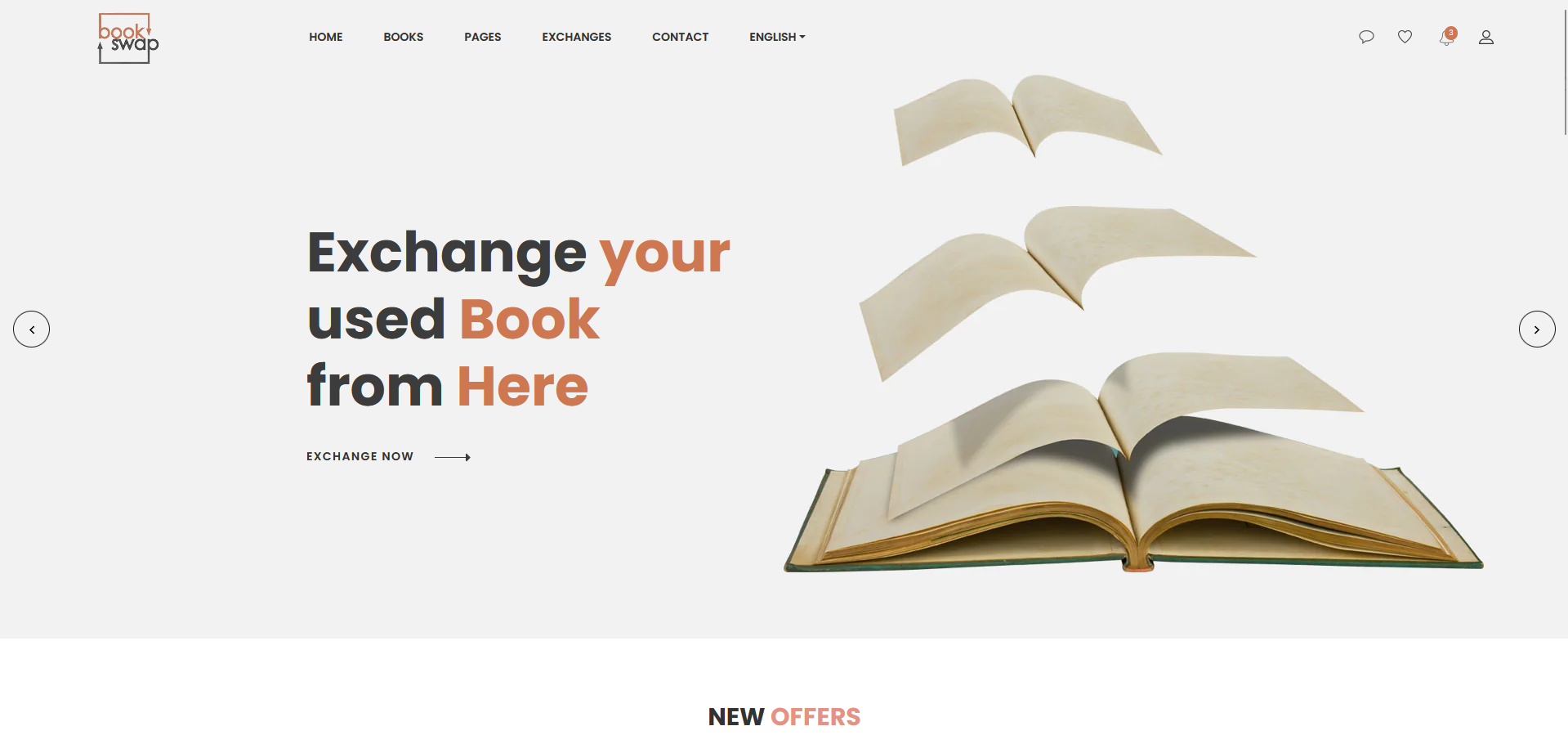


Рисунок 1 — Головна сторінка з англійською мовою

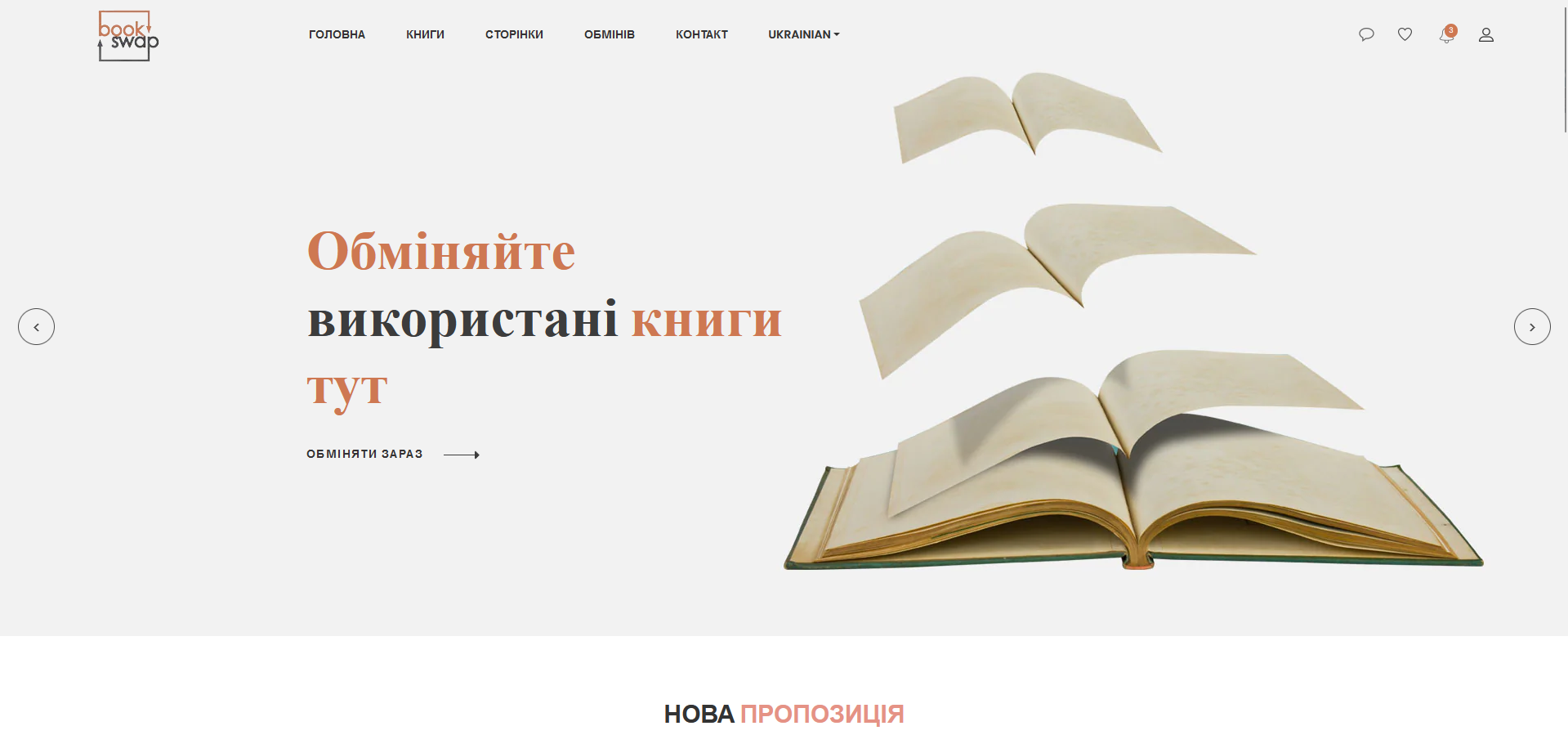


Рисунок 2 — Головна сторінка з українською мовою



Рисунок 3 — Форма підписки з англійською мовою

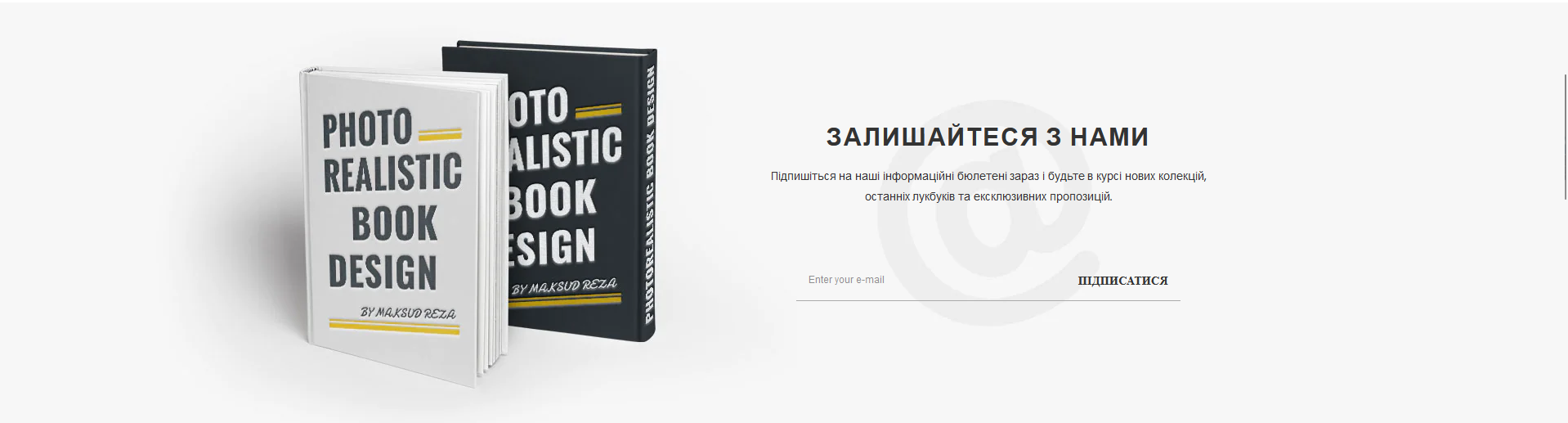


Рисунок 4 — Форма підписки з українською мовою

**Контрольні запитання**

1. *Об’єкт HttpContext*

Об’єкт HttpContext є глобальним контекстом для обробника запитів у ASP.NET Core. Він містить інформацію про запит, відповідь, користувача та інші дані, які необхідні для обробки запиту.

1. *Об’єкт ControllerContext*

Об’єкт ControllerContext є контекстом контролера в ASP.NET Core. Він містить інформацію про контролер, який обробляє запит, а також про інші дані, які необхідні для обробки запиту.

1. *Властивість Request*

Властивість Request об’єкта HttpContext містить інформацію про запит, наприклад метод запиту, URL-адресу, заголовки та тіло запиту.

1. *Властивість HttpContext.Request.Cookies*

Властивість HttpContext.Request.Cookies містить куки запиту.

1. *Властивість HttpContext.Response.Cookies*

Властивість HttpContext.Response.Cookies встановлює куки відповіді.

1. *Метод HttpContext.Response.Write*

Метод HttpContext.Response.Write виводить текст у тіло відповіді.

Синтаксис: HttpContext.Response.Write(text);

1. *Визначення користувача*

Користувач у ASP.NET Core визначається об’єктом User об’єкта HttpContext. Об’єкт User містить інформацію про користувача, наприклад ім'я користувача, роль та інші дані.

1. *Визначення типу ресурсу*

Тип ресурсу в ASP.NET Core визначається класом, який наслідує від класу Resource. Клас Resource містить інформацію про ресурс, наприклад його ім'я, тип даних та інші дані.

1. *Створення ресурсів*

Ресурси в ASP.NET Core можна створити за допомогою методу HttpContext.Resources.Add. Цей метод приймає в якості аргументу об'єкт ресурсу.

1. *Файл ресурсів*

Файл ресурсів в ASP.NET Core - це файл, який містить ресурси. Файл ресурсів має розширення .resx.

1. *Заповнення ресурсів*

Ресурси в файлі ресурсів можна заповнити за допомогою методу Resource.SetValue. Цей метод приймає в якості аргументів ім'я ресурсу та його значення.

1. *Звернення до ресурсів в представленні*

До ресурсів в представленні можна звернутися за допомогою атрибута Resource. Цей атрибут приймає в якості аргументу ім'я ресурсу. Наприклад: <p @Resource("MyResource")>Значення ресурсу</p>

1. *Умовності при створенні ресурсів*

При створенні ресурсів можна використовувати умови. Для цього можна скористатися методом HttpContext.Resources.AddIfExists. Цей метод приймає в якості аргументу об'єкт ресурсу. Якщо ресурс з таким ім'ям вже існує, то він не буде створений.

1. *Збереження культури в куки*

Культуру в куки можна зберегти за допомогою методу HttpContext.Response.Cookies.AppendCulture. Цей метод приймає в якості аргументу культуру.

1. *Отримання куки з контексту*

Куки з контексту можна отримати за допомогою методу HttpContext.Request.Cookies.Отримати куку за ім'ям можна за допомогою методу HttpContext.Request.Cookies["name"].