ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет програмування та комп’ютерних

і телекомунікаційних систем

Кафедра інженерії програмного забезпечення

**Лабораторна робота №** 1

з дисципліни «Програмування в інтернет»

на тему:

«Створення примітивного додатка на ASP.NET MVC 4.»

**Виконав:**

студент 1 курсу, групи ІПЗс-20-1, варіант 3 Н.В.Савич

**Перевірив:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.М.Яшина

(підпис)

Хмельницький – 2020 р.

Мета. Створити примітивний додаток на MVC 4, який покликаний дати деяке початкове розуміння роботи MVC. Як шаблон проекту використовується Empty і проектується міні-додаток з нуля.

Завдання

Розробити web – додаток, що буде представляти примітивну емуляцію служби страхування автомобілів, в якому ми зможемо переглядати надання виду страхування і замовити відповідне страхування.

Код Програми:

Моделі:

Type.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace lab1.Models

{

public class Type

{

public int Id { get; set; }

public string Name { get; set; }

public int Price { get; set; }

}

}

Order.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace lab1.Models

{

public class Order

{

public int OrderId { get; set; }

public string Person { get; set; }

public string Address { get; set; }

public int TypeId { get; set; }

public DateTime Date { get; set; }

}

}

Контролери:

HomeController.cs

using lab1.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace lab1.Controllers

{

public class HomeController : Controller

{

TypeContext db = new TypeContext();

public ActionResult Index()

{

IEnumerable<Models.Type> types = db.Types;

ViewBag.Types = types;

return View();

}

[HttpGet]

public ActionResult Buy(int id)

{

ViewBag.BookId = id;

return View();

}

[HttpPost]

public string Buy(Order order)

{

order.Date = DateTime.Now;

db.Orders.Add(order);

db.SaveChanges();

return "Спасибі," + order.Person + ", за покупку!";

}

}

}

Представлення:

Index.cshtml

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>Страхова компанія</title>

</head>

<body>

<div>

<h3>Види страхувань</h3>

<table>

<tr>

<td><p>Назва виду страхування</p></td>

<td><p>Цiна</p></td>

<td></td>

</tr>

@foreach (var b in ViewBag.Types)

{

<tr>

<td><p>@b.Name</p></td>

<td><p>@b.Price</p></td>

<td><p><a href="/Home/Buy/@b.Id">Купити</a></p></td>

</tr>

}

</table>

</div>

</body>

</html>

Buy.cshtml

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>Покупка</title>

</head>

<body>

<div>

<h3>Форма оформлення замовлення</h3>

<form method="post">

<input type="hidden" value="@ViewBag.TypeId" name="TypeId" />

<table>

<tr>

<td><p>Введіть своє ім'я </p></td>

<td><input type="text" name="Person" /> </td>

</tr>

<tr>

<td><p>Введіть адресу :</p></td>

<td>

<input type="text" name="Address" />

</td>

</tr>

<tr>

<td>

<input type="submit" value="Відправити" />

</td>

<td></td>

</tr>

</table>

</form>

</div>

</body>

</html>

Вигляд програми:

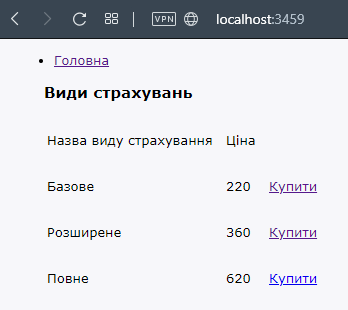


Рис 1.-Сторінка перегляду видів страхування

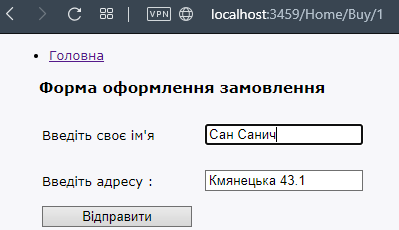


Рис 2.-Сторінка оформлення замовлення

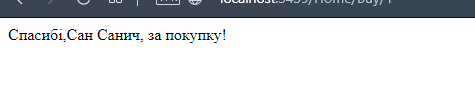


Рис 3.-Результат виконання замовлення

Контрольні питання:

1. Поняття технології MVC.

ASP.NET MVC є фреймворком для розробки від Microsoft, який поєднує в собі ефективність і акуратність архітектури MVC, найсучасніші ідеї та методи гнучкої розробки та кращі властивості існуючої платформи ASP.NET. Це альтернатива традиційним ASP.NET Web Forms, яка забезпечує істотну перевагу для всіх, крім найпростіших і тривіальних, проектів веб-розробки. Незважаючи на те, що Microsoft розробляв інструменти та фреймворки для веб-розробки протягом вже досить тривалого періоду, ASP.NET MVC стала проривом, оскільки, на відміну від попередніх розробок, робить наголос на чистий код, концепцію поділу і тестування.

1. Що таке шаблон MVC

Моде́ль–вигляд–контро́лер — [архітектурний шаблон](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%BD%D1%96_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F" \o "Архітектурні шаблони програмного забезпечення), який використовується під час проектування та розробки [програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F" \o "Програмне забезпечення).

Цей шаблон передбачає поділ системи на три взаємопов'язані частини: [модель даних](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), вигляд ([інтерфейс користувача](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0" \o "Інтерфейс користувача)) та модуль керування. Застосовується для відокремлення даних (моделі) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах сприяє впорядкованості їхньої структури і робить їх більш зрозумілими за рахунок зменшення складності.

1. Які функції виконує Контролер

Контролер одержує вхідні дані й перетворює їх на команди для моделі чи вигляду.

1. Які функції виконує Представлення

Представлення може являти собою будь-яке представлення інформації, одержуване на виході, наприклад графік чи діаграму. Одночасно можуть співіснувати кілька виглядів (представлень) однієї і тієї ж інформаці

1. Які функції виконує Модель?

Модель є центральним компонентом шаблону MVC і відображає поведінку застосунку, незалежну від інтерфейсу користувача. Модель стосується прямого керування даними, логікою та правилами застосунку.

1. Перерахуйте шаблони додатки MVC та їх призначення

Пустий проект, ASP.Net MVC Веб-додаток,SST

1. Структура проекту MVC

Структура проекту MVC 4. App\_Data. У цій папці зберігаються всі необхідні файли і ресурси, наприклад, бази даних, що використовуються додатком. Після розгортання програми лише безпосередньо додаток може працювати з цією папкою, доступ простих користувачів в цю папку заборонений. Файл Global.asax і папка App\_Start. У MVC 4 була додана папка App\_Start. Вона включає весь функціонал конфігурації програми, який в попередніх версіях містився у файлі Global.asax, а тепер перенесений в набір статичних класів, що викликаються в Global.asax. Ці статичні класи містять деяку логіку ініціалізації програми, виконуються при запуску. Файл Web.config. Файл конфігурації програми, який знаходиться в кореневій теці програми Content. Містить деякі допоміжні файли, які не включають код на С# або javascript, і які розгортаються разом з додатком. Зокрема, тут можуть розміщуватися файли стилів CSS. Так, в цій папці ви побачите файл Site.css, який містить стилі додатка, а також папку з темами, що включає стилі CSS та зображення для певних тем. Controllers. Містить контролери - класи, що відповідають за роботу програми. За замовчуванням тут знаходяться два контролера - HomeController і AccountController. Папки Images і Scripts. Папки Images і Scripts містять відповідно зображення і скрипти на JavaScript, використовувані в додатку. За замовчуванням ці папки вже містять файли, зокрема, в папку Scripts вже поміщені файли бібліотеки jQuery. Models. Містить моделі, використовувані додатком. За замовчуванням тут визначена одна модель - AccountModel, яка представляє окремий обліковий запис. Views. Тут розміщуються представлення. Представлення згруповані по папках, кожна з яких відповідає одному контролеру. Після отримання та обробки запиту контролер, відправляє одне з цих представлень, заповнених деякими даними, клієнту. Крім того, є папка загальних для контролерів представлень - папка Shared

1. Роль бібліотеки EntityFramework

Ця бібліотека відповідає за самостійне створення БД.

1. Призначення контексту класса

Контекст даних. Він потрібен для полегшення доступу до БД на основі деякої модел

1. Рядок підключення ConnectionStrings

Тим самим ми визначаємо шлях до бази даних, яка потім буде створюватися. Вираз | DataDirectory | представляє заступник, який вказує, що база даних буде створюватися в проекті в папці App\_Data

1. Призначення об'єкта ViewBag

Далі ми створюємо об'єкт Books в об'єкті ViewBag і присвоюємо йому цей список. Об'єкт ViewBag є таким об'єктом, який передається в представлення. І ми можемо визначити в цьому об'єкті будь-яку змінну і передати їй деяке значення, а потім в представленні витягти це значення.

1. Основи маршрутизації

Основні налаштування маршрутирізації знаходяться в папці RouteConfig.cs

1. Дані для моделей за замовчуванням

Models. Містить моделі, використовувані додатком. За замовчуванням тут визначена одна модель - AccountModel, яка представляє окремий обліковий запис.

1. Призначення файла Global.asax

Файл [global.asax](https://ssl.microsofttranslator.com/bv.aspx?ref=TAns&from=&to=uk&a=global.asax) дає змогу записувати обробники подій, які реагують на глобальні події. Користувачі не можуть запитувати файл [global.asax](https://ssl.microsofttranslator.com/bv.aspx?ref=TAns&from=&to=uk&a=global.asax) безпосередньо. Файл [global.asax](https://ssl.microsofttranslator.com/bv.aspx?ref=TAns&from=&to=uk&a=global.asax) автоматично запускає свій код у відповідь на деякі події програми.

1. Призначення, підключення файла Site.css

Задає css стилі для роботи представленнь

1. Призначення, підключення майстер сторінки

Дозволяє лише 1 раз оголосити зовнішній вигляд елементів які присутні в усіх представленнях

1. Призначення секції RenderBody ()

Метод @RenderBody(),є замысником і його місце потім буде замінено іншими представленнями, які використовують цю головну сторінку.