ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

НА ПРОЄКТНУ-ТЕХНОЛОГІЧНУ ПРАКТИКУ

Студенту групи *К12-19-1*

спеціальності *123* «*Комп’ютерна інженерія*»

*Дідуху Вадиму Анатолійовичу*  .

Прізвище, ім’я, по батькові

База практики *ПП «АВІВІ», м. Хмельницький*

Термін практики з 06.06.2022 р. по 26.06.2022 р.

Тема індивідуального завдання  *Розробити веб-сайт з імітацією реального замовлення*

Завдання видав: *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Завдання одержав: \_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис, дата

Хмельницький національний університет

Кафедра комп’ютерної інженерії та програмування

Звіт

з науково-дослідної практики

база практики *ПП «АВІВІ», м. Хмельницький*

                          ЗППКІ. XXXXX. XX. XX

Шифр Номер ІНПС, номер за списком, номер теми в наказі

Галузь знань *12* «*Інформаційні технології*»

Спеціальність *123* «*Комп’ютерна інженерія*»

Студента III курсу, група КІ2–19-1 А. В. Дідуха

Керівник від кафедри Керівник від бази практики

Павлова О.О. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис, прізвище Підпис, прізвище

Кількість балів \_\_\_\_

Оцінка за шкалою: національною\_\_\_\_\_\_/ЄКТС\_\_\_\_

Члени комісії: \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

Хмельницький 2022

ЗМІСТ

[ВСТУП 4](#_Toc107160587)

[1 ЗАГАЛЬНА ХАРАКТЕРИСТИКА БАЗИ ПРАКТИКИ 5](#_Toc107160588)

[1.1 Організаційна структура бази практики (ПП «АВІВІ») 5](#_Toc107160589)

[1.3 Аналіз предметної області та її актуальність 6](#_Toc107160590)

[1.4 Подання в ASP.NET Core MVC 7](#_Toc107160591)

[2 ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ ТА РЕАЛІЗІЦІЯ ПРОЄКТА 10](#_Toc107160592)

[2.1  Реалізація проєкта 10](#_Toc107160593)

[2.2  Створення уявлення та передача даних у подання 11](#_Toc107160594)

[2.3  Аутентифікація, авторизація та Клайми 12](#_Toc107160595)

[Висновок 14](#_Toc107160596)

[Список використаних джерел 15](#_Toc107160597)

[Додаток А 16](#_Toc107160598)

[Скріни сайту 16](#_Toc107160599)

ВСТУП

Завданням проєктної-технологічної практики розробити веб-сайт з імітацією реального замовлення клієнта. Таким чином за розробку сайта взято Asp .NET Core 6.

Базова структура стандартного проекту ASP.NET Core. Проект ASP.NET Core Empty містить дуже просту структуру - необхідний мінімум для запуску програми:

* Connected Services: підключені сервіси з Azure
* Dependencies: всі додані до проекту пакети та бібліотеки, інакше кажучи, залежності
* Properties: вузол, який містить деякі параметри проекту. Зокрема, у файлі launchSettings.json описані налаштування запуску проекту, наприклад, адреси, за якими запускатиметься програма.
* appsettings.json: файл конфігурації проекту у форматі json
* Program.cs: головний файл програми, з якого починається його виконання. Код цього файлу налаштовує та запускає веб-додаток

Це базова мінімальна структура проекту веб-програми на ASP.NET. При створенні інших типів проектів ASP.NET структура буде відрізнятися, відповідно початковий проект матиме більше функціоналу, проте це каркас, від якого ми можемо відштовхуватися, додаючи до нього якісь свої файли і папки.

1. ЗАГАЛЬНА ХАРАКТЕРИСТИКА БАЗИ ПРАКТИКИ

## 1.1 Організаційна структура бази практики (ПП «АВІВІ»)

Базою практики виступає приватне підприємство «АВІВІ» м. Хмельницький. Це підприємство було засновано у 2008 році.

Основним видом діяльності є впровадження CRM, розробка сучасного програмного забезпечення від невеличкого сайту чи інтернет-магазину до великого корпоративного порталу.

Основою підприємства є команда ІТ-професіоналів, сертифікованих компанією «Бітрікс».

Сьогодні «АВІВІ» використовує всі сучасні технології і тенденції з метою реалізації найефективніших інструментів для розвитку бізнесу будь-якого масштабу.

Штат підприємства складає понад 50 осіб. Основним напрямком робіт ТОВ «АВІВІ» є допомога клієнтам заробляти більше з використанням сучасних технологій шляхом впровадження сучасних інформаційних технологій

Персонал підприємcтва – висококваліфіковані фахівці у галузі програмування веб-сервісів. Підприємство має 4 офіси, більше 20 країн-клієнтів, та більше 100 успішних імплементованих у бізнес ІТ-проектів.

Компанія «АВІВІ» включає дирекцію, бухгалтерію, відділ кадрів, відділ технічної підтримки, віддів софтверного девелопменту з командами керівників проектів та розробників програмного забезпечення.

Дирекцію складає два співзасновники, яким підпорядковуються усі відділи компанії.

Відділ технічної підтримки нараховує 5 співробітників, серед яких 2 девопси та три системних інженери.

Відділ софтверного девелопменту включає PR-менеджера, 5 керівників проектів та більше 40 розробників програмного забезпечення, які в залежності від замовлення формують необхідну команду розробки ПЗ, та команда з 5 веб-дизайнерів.

1.2 Постановка задачі для проєктування веб-сайту

Завдання включає: розробити веб-сайт з імітацією реального замовлення

клієнта згідно запропонованого нижче варіанту технічного завдання. Сайт повинен містити галерею продуктів (картки товарів) для вибору та можливість замовлення товару через форму на сайті (клієнт може залишити свої контактні дані на сайті (ім’я, номер телефону) та вказати бажаний товар, щоб з ним можна було сконтактувати). А також сторінку адміністратора, де можна переглянути інформацію про клієнтів у вигляді таблиці - ім’я, номер телефону та повідомлення.

## 1.3 Аналіз предметної області та її актуальність

Інтернет-магазин — місце в інтернеті, де відбувається прямий продаж товарів споживачеві (юридичній або фізичній особі), враховуючи доставку. При цьому розміщення споживацької інформації, замовлення товару і угода відбуваються там само, всередині мережі (на сайті інтернет-магазину).

Інтернет-магазин обійдеться набагато дешевше, ніж ваш, вже існуючий, бізнес. Вам не потрібно буде купувати приміщення або щомісяця платити за його оренду, платити за обладнання, на кшталт касового апарату, холодильників (якщо мова про продуктовий магазин), вітрин і стелажів. Відповідно, про комунальні платежі, придбання камер відеоспостереження і про заробітну плату для співробітників теж можна не хвилюватися.

Для онлайн-магазину потрібно набагато менше працівників. Контент-менеджер, який буде викладати нові товари на сайт, ще один менеджер, який буде відповідати на запитання покупців, бухгалтер що приходить — в принципі, ви можете взяти все це в свої руки. В такому випадку економія буде ще крутіша.

Важливі елементи інтернет-магазину — оновлення наявного асортименту (продукти/товари та їх кількість), можливість додавати товари до «кошику», вхід для зареєстрованих користувачів. У деяких випадках можна використовувати систему оплати через інтернет (електронні гроші), у простішому випадку оплата відбувається звичайними грішми через банк на рахунок, роздрукований з сайту. Оплата через інтернет передбачає необхідність створення кількох облікових записів (принаймні двох), але у деяких випадках системи оплати можуть бути пов'язаними з магазином.

Для того, щоб інтернет-магазин видавав інформацію залежно від запитів, надавав можливості пошуку тощо — на сервер встановлюється підтримка скриптів (наприклад PHP, Perl). У більш комплексному варіанті, програма інтернет-магазину — це система управління вмістом сайту, яка вже має підтримку скриптів, надає можливість в он-лайновому режимі (головним чином через інтернет) і в межах наявного асортименту виконувати купівлю потрібних товарів.

## 1.4 Подання в ASP.NET Core MVC

У шаблоні MVC (Model-View-Controller, модель - подання - контролер) подання відповідає за відображення даних програми та взаємодія з користувачем. Подання це HTML-шаблон з вбудованою Razor розміткою. Razor розмітка - це код, що взаємодіє з розміткою HTML для створення веб-сторінки, що надсилається клієнту.

У ASP.NET Core MVC уявлення - це .cshtml файли, що використовують мову програмування C# в Razor розмітці. Як правило, файли уявлень об'єднуються в папки з іменами, що відповідають окремим контролерам програми.

У шаблоні MVC (Model-View-Controller, модель - подання - контролер) подання відповідає за відображення даних програми та взаємодія з користувачем. Подання це HTML-шаблон з вбудованою Razor розміткою. Razor розмітка - це код, що взаємодіє з розміткою HTML для створення веб-сторінки, що надсилається клієнту.

У ASP.NET Core MVC уявлення - це .cshtml файли, що використовують мову програмування C# в Razor розмітці. Як правило, файли уявлень об'єднуються в папки з іменами, що відповідають окремим контролерам програми.

Контролер Home представлений папкою Home у папці Views. Папка Home містить уявлення для AboutContact веб-сторінок та Index (домашньої сторінки). Коли користувач запитує одну з цих трьох веб-сторінок, дії контролера в Home контролері визначають, які з трьох уявлень використовуються для збирання та повернення веб-сторінки користувачеві.

Макети дозволяють створювати однакові розділи веб-сторінок та скоротити повтори у коді. Вони часто містять верхній та нижній колонтитули, а також елементи навігації та меню. Колонтитул зазвичай містять стандартну розмітку для багатьох елементів метаданих і посилання на ресурси скриптів і стилів. Макети дозволяють уникнути використання цієї стандартної розмітки у виставах.

Часткові уявлення скорочують дублювання коду, забезпечуючи управління часто використовуваними частинами уявлень. Наприклад, часткова вистава буде корисною для біографії автора, яка присутня в кількох виставах на веб-сайті блогу. Біографія автора — це звичайний вміст уявлення, який не вимагає виконання коду для створення вмісту веб-сторінки. Для доступу до вмісту біографії автора подання досить прив'язки моделі, тому часткове подання ідеально підходить для такого вмісту.

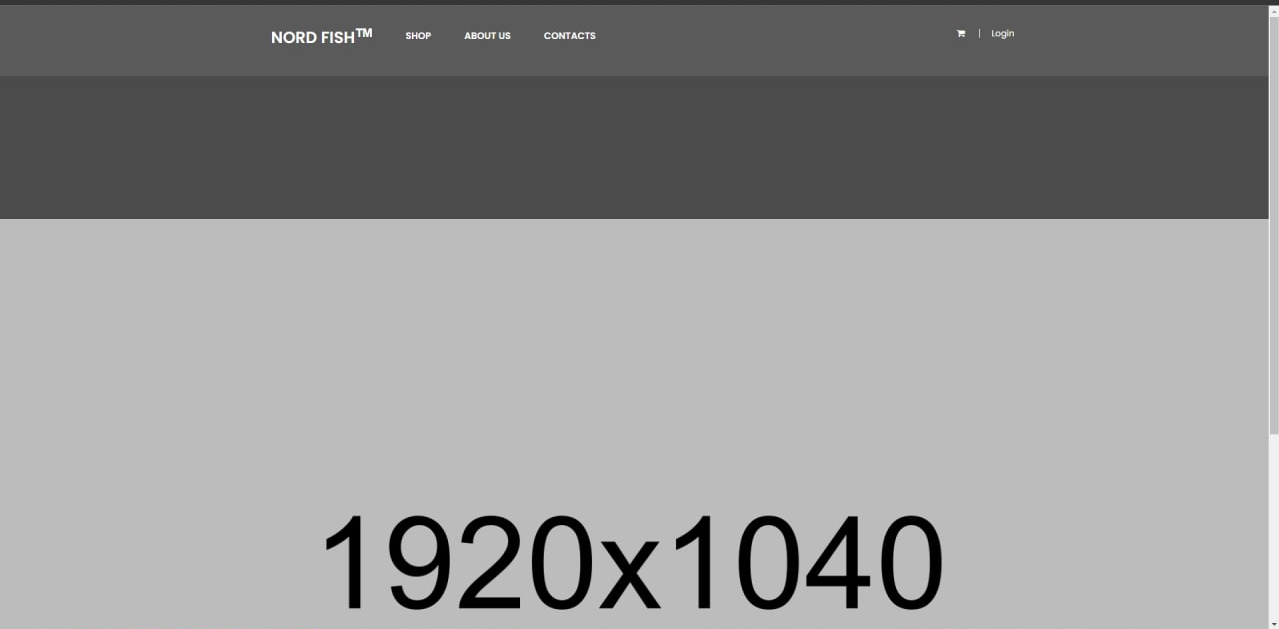
Компоненти уявлення схожі на часткові уявлення тим, що вони також дозволяють скоротити повтори коду, однак вони підходять для вмісту уявлення, який вимагає виконання коду на сервері для перетворення веб-сторінки для перегляду. Компоненти представлення корисні в тих випадках, коли для підготовки вмісту для перегляду потрібна взаємодія з базою даних, наприклад, для кошика на веб-сайті електронного магазину. Під час формування вихідних даних веб-сторінки компоненти подання не обмежені прив'язкою моделі.

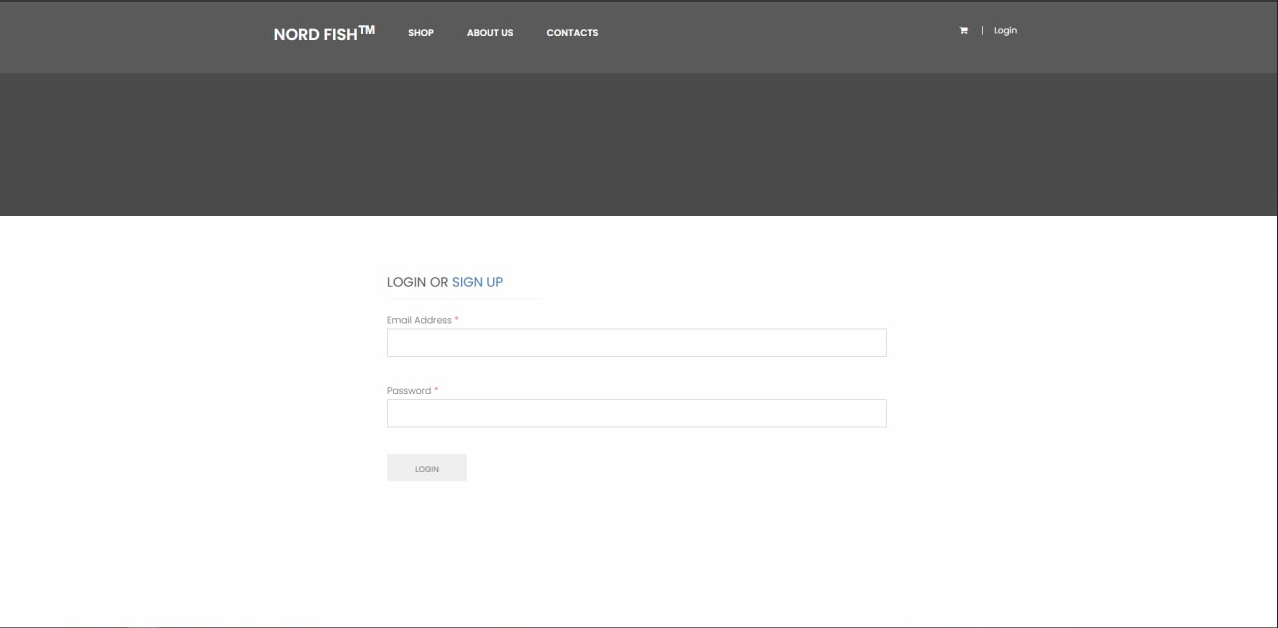
Уявлення допомагають реалізувати поділ завдань у додатку MVC, ізолюючи розмітку інтерфейсу користувача від інших частин програми. Дотримання принципу SoC робить програму модульною, що дає кілька переваг.

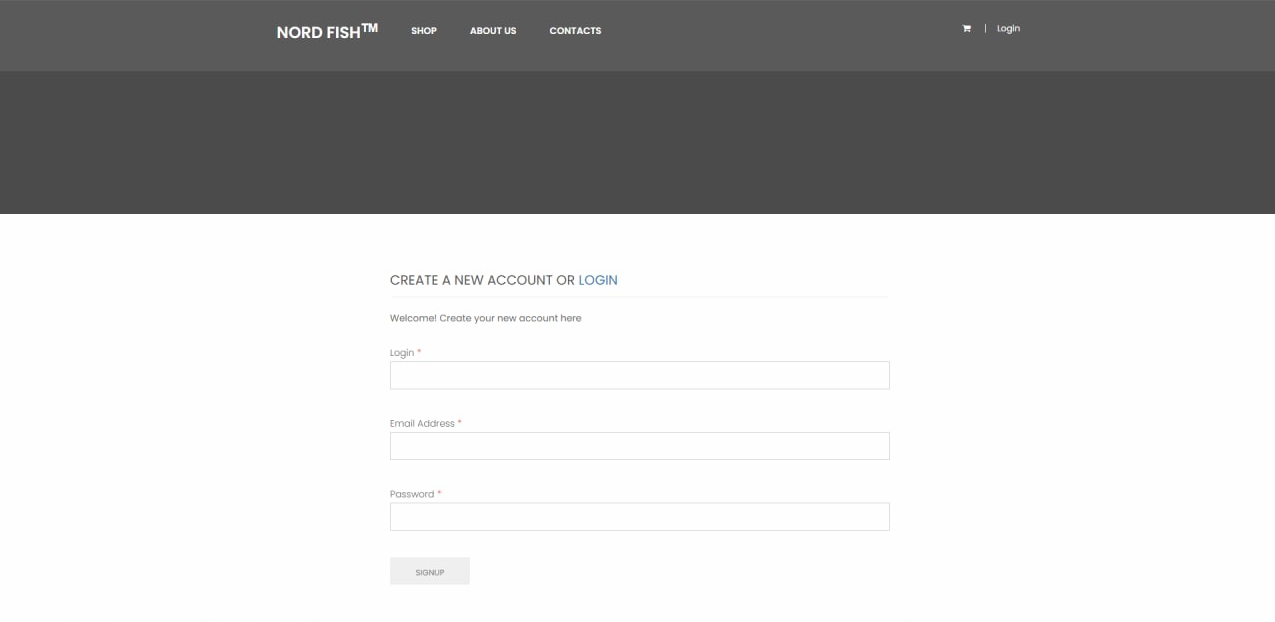
* Додаток простіше обслуговувати, оскільки він краще організований. Подання, як правило, групуються за функціональними можливостями програми. Це спрощує пошук пов'язаних уявлень під час роботи над певною функцією.
* Частини програми слабо пов'язані один з одним. Ви можете розробляти та оновлювати подання програми окремо від компонентів бізнес-логіки та доступу до даних. При оновленні подань додатка не обов'язково оновлювати інші його частини.
* Частини інтерфейсу користувача простіше тестувати, так як уявлення є окремими модулями.
* Завдяки більш упорядкованій структурі менша ймовірність того, що розділи інтерфейсу користувача будуть випадково повторюватися.

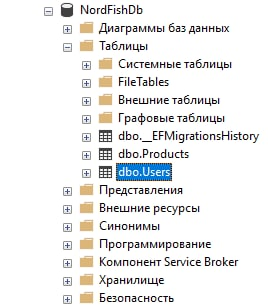
1. ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ ТА РЕАЛІЗІЦІЯ ПРОЄКТА

2.1  Реалізація проєкта









2.2  Створення уявлення та передача даних у подання

Подання щодо контролера створюються в папці Views/[ControllerName] . Подання, які спільно використовуються між контролерами, містяться в папці Views/Shared . Щоб створити уявлення, додайте новий файл і надайте йому те саме ім'я, що і пов'язане з ним дію контролера з розширенням .cshtml файлу. Щоб створити уявлення, що відповідає About дії в контролері Home, створіть About.cshtml файл у папці Views/Home :

@{

ViewData["Title"] = "About";

}

<h2>@ViewData["Title"].</h2>

<h3>@ViewData["Message"]</h3>

<p>Use this area to provide additional information.</p>

Razor розмітка починається з символу @. Запустіть інструкції C#, помістивши код C# у Razor блоки коду, задані фігурними дужками ({...}). Прикладом може бути наведений вище оператор присвоєння значення "About" властивості ViewData["Title"]. Для відображення значень у коді HTML можна просто посилатися на них за допомогою символу @. Див. вміст елементів <h2> та <h3> вище.

Наведений вище вміст — це лише частина веб-сторінки, що відображається для користувача. Решта макета сторінки та інші стандартні аспекти представлення визначаються інших файлах представлення. Додаткові відомості див. у статті Макет.

Існує кілька підходів до передачі даних у подання:

* Строго типізовані дані: viewmodel
* Слабко типізовані дані
* ViewData (ViewDataAttribute)
* ViewBag

2.3  Аутентифікація, авторизація та Клайми

ASP.NET Core має вбудовану підтримку автентифікації на основі cookie. Для цього в ASP.NET визначено спеціальний компонент middleware, який серіалізує дані користувача в зашифровані автентифікаційні куки та передає їх на бік клієнта. При отриманні запиту від клієнта, у якому містяться аутентифікаційні куки, відбувається їх валідація, десеріалізація та ініціалізація властивості User об'єкта HttpContext.

Ключовим інструментом для авторизації є атрибут AuthorizeAttribute із простору імен Microsoft.AspNetCore.Authorization. Наприклад, у минулій темі цей атрибут обмежував доступ до методу Index контролера HomeController:

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Authorization;

public class HomeController : Controller

{

    [Authorize]

    public IActionResult Index()

    {

        return Content(User.Identity.Name);

    }

    // остальные методы

}

У цьому випадку доступ до методу Index мають лише користувачі, які залогінилися в додатку. Анонімні користувачі в даному випадку при доступі до методу Index будуть переадресовані на форму входу до програми.

При цьому атрибут Authorize не вказує, як користувач повинен автентифікуватись, він тільки виконує перевірку. Для перевірки статусу користувача атрибут використовує властивості User об'єкта HttpContext, яке встановлюється інфраструктурою ASP.NET.

Властивість HttpContext.User представляє об'єкт інтерфейсу IPrincipal, визначений у просторі імен System.Security.Principal. Цей інтерфейс визначає метод IsInRole() та властивість Identity.

Властивість Identity повертає об'єкт інтерфейсу IIdentity, який пов'язаний із запитом.

Метод IsInRole() як параметр бере участь і повертає true, якщо поточний користувач належить цій ролі.

Об'єкт IIdentity, у свою чергу, надає інформацію про поточного користувача через такі властивості:

* AuthenticationType: тип аутентифікації у рядковому вигляді
* IsAuthenticated: повертає true, якщо користувач автентифікований
* Name: Повертає ім'я користувача. Як правило, як подібне ім'я використовується логін, за яким користувач входить додаток

Для визначення, чи автентифікований користувач, ASP.NET Core використовує автентифікаційні куки.

Ми можемо застосовувати ці властивості у контролерах або уявленнях:

public IActionResult Index()

{

    if(User.Identity.IsAuthenticated)

    {

        return Content(User.Identity.Name);

    }

    return Content("не аутентифицирован");

}

Висновок

Дана проєктна-технологічна практика надала змогу розширити накопичений обсяг знань в сфері програмування на ASP .NET Core 6.

Проектно-технологічна практика має своєю метою підготувати студентів до роботи на підприємствах, в установах, наукових і проектних організаціях та оволодіння студентами передовими методами праці, виробничими навичками й уміннями. Ця практика покликана закріпити і поглибити теоретичні знання, отримані студентами в Університеті з фундаментальних та спеціалізованих дисциплін.

Метою проектно-технологічної практики є ознайомлення студентів з процесом проектування, розробки, тестування та експлуатації елементів комп’ютерних систем і інформаційних технологій в умовах профільних підприємств та власна участь студентів у цьому процесі, а також набуття студентами досвіду самостійної науково-дослідної роботи та опрацювання методики її проведення, поглиблення теоретичних знань у сфері комп’ютерних наук, формування вмінь і навичок опрацювання наукових та інформаційних джерел.

## Список використаних джерел

1. ASP .NET Tutorial. Metanit URL: https://metanit.com/sharp/aspnet6/1.2.php https://itvdn.com/ua/specialities/backend-developer
2. ASP .NET Tutorial. Wikipedia URL: <https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BDhttps://uk.wikipedia.org/wiki/Bootstrap#:~:text=Bootstrap%20%E2%80%94%20%D1%86%D0%B5%20%D0%B1%D0%B5%D0%B7%D0%BA%D0%BE%D1%88%D1%82%D0%BE%D0%B2%D0%BD%D0%B8%D0%B9%20%D0%BD%D0%B0%D0%B1%D1%96%D1%80%20%D1%96%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%96%D0%B2,%D0%B0%20%D1%82%D0%B0%D0%BA%D0%BE%D0%B6%20%D0%B4%D0%BE%D0%B4%D0%B0%D1%82%D0%BA%D0%BE%D0%B2%D1%96%20%D1%80%D0%BE%D0%B7%D1%88%D0%B8%D1%80%D0%B5%D0%BD%D0%BD%D1%8F%20JavaScript>.
3. ASP .NET Tutorial. Microsoft URL: <https://docs.microsoft.com/ua-ua/aspnet/core/mvc/views/overview?view=aspnetcore-6.0>
4. ASP .NET Tutorial. Metanit URL: https://metanit.com/sharp/aspnet5/15.1.php

## Додаток А

## Скріни сайту

