Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Веб-додаток для продажу жіночого одягу за типом фігури

(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-4

Юлія Васильченко
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект

(повна назва освітньої програми)
Керівник ас. Ірина Малєєва
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Васильченко Юлії Сергіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Веб-додаток для продажу жіночого одягу за типом фігури

затверджена наказом університету від 19 травня 20 25 р. № 378Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 червня 20 25 р.
3. Вихідні дані до роботи Аналітика онлайн-продажу жіночого одягу, типи жіночих фігур, вимоги до вебдодатків, ASP.NET Core MVC, Razor Pages, Entity Framework Core, PostgreSQL, ML.NET, адаптивний дизайн, персоналізація інтерфейсу.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Теоретичні відомості обраних технологій
3) Проєктування та розробка вебдодатку
4) Аналіз розробленого вебдодатку

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	20.05 – 24.05	виконано
3	Постановка задачі	25.05 – 27.05	виконано
4	Обґрунтування вибору використаних технологій	28.05 – 31.05	виконано
5	Програмна реалізація	01.06 – 06.06	виконано
6	Оформлення пояснювальної записки	07.06 – 15.06	виконано
7	Підготовка до захисту	16.06 – 19.06	виконано
8	Захист перед ЕК	20.06.2025	виконано

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Ірина Малєєва
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 85 с., 41 рис., 2 дод., 17 джерел.

ВЕБДОДАТОК, ЕЛЕКТРОННА КОМЕРЦІЯ, ТИП ФІГУРИ, ASP.NET CORE MVC, CSS, ENTITY FRAMEWORK CORE, HTML, JAVASCRIPT, POSTGRESQL, RAZOR PAGES.

Об’єкт дослідження – вебдодаток для онлайн-продажу жіночого одягу з елементами персоналізації користувацького досвіду.

Предмет дослідження – розробка та реалізація інтелектуального вебдодатку на основі технологій ASP.NET Core MVC, Entity Framework Core для визначення типу фігури та формування рекомендацій одягу.

Мета роботи – створити вебдодаток, який дозволяє користувачці на основі введених параметрів визначити тип фігури та отримати рекомендації щодо одягу, що найкраще відповідає її індивідуальним особливостям.

Методи дослідження – у процесі виконання роботи застосовувалися методи аналізу предметної галузі, проєктування архітектури вебсистеми, реалізація серверної логіки за допомогою ASP.NET Core MVC, побудова інтерфейсу з використанням Razor Pages і Bootstrap, моделювання структури бази даних у PostgreSQL з використанням EF Core.

У результаті роботи було реалізовано багатофункціональний

вебдодаток із каталогом одягу, інструментами фільтрації та калькулятором типу фігури, що дозволяє формувати індивідуальні рекомендації для користувачів.

Дана робота є прикладом ефективного поєднання вебтехнологій, принципів адаптивного дизайну та персоналізованого підходу в онлайн-комерції, та має перспективи подальшого розвитку – зокрема, розширення бази одягу, впровадження стилістичних трендів або інтеграції мобільної версії додатку.

ABSTRACT

Bachelor’s thesis contains: 85 pp., 41 fig., 2 ann., 17 references.

ASP.NET CORE MVC, CSS, E-COMMERCE, ENTITY FRAMEWORK CORE, FIGURE TYPE, HTML, JAVASCRIPT, POSTGRESQL, RAZOR PAGES, WEB APPLICATION.

The object of the study is a web application for online sales of women’s clothing with elements of user experience personalization.

The subject of the study is the development and implementation of an intelligent web application based on ASP.NET Core MVC, Entity Framework Core technologies for determining the type of figure and forming clothing recommendations.

The purpose of the work is to create a web application that allows the user to determine the type of figure based on the entered parameters and receive recommendations for clothing that best suits her individual characteristics.

Research methods – in the process of performing the work, methods of subject area analysis, web system architecture design, server logic implementation using ASP.NET Core MVC, interface construction using Razor Pages and Bootstrap, database structure modeling in PostgreSQL using EF Core.

As a result of the work, a multifunctional

web application with a clothing catalog, filtering tools and a figure type calculator was implemented, which allows you to form individual recommendations for users.

This work is an example of an effective combination of web technologies, adaptive design principles and a personalized approach in online commerce, and has prospects for further development – in particular, expanding the clothing base, introducing stylistic trends or integrating a mobile version of the application.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface – інтерфейс програмування додатків;

ASP.NET Core – Active Server Pages .NET Core – фреймворк для створення вебдодатків;

CSS – Cascading Style Sheets – каскадні таблиці стилів;

DB – Database – база даних;

EF Core – Entity Framework Core – об’єктно-реляційний фреймворк для роботи з базами даних;

HTML – HyperText Markup Language – мова розмітки гіпертексту;

HTTP – HyperText Transfer Protocol – протокол передачі гіпертексту;

JS – JavaScript – мова сценаріїв для веброзробки;

LINQ – Language Integrated Query – вбудована в мову система запитів;

MVC – Model-View-Controller – модель-вид-контролер, архітектурний шаблон;

ORM – Object-Relational Mapping – об’єктно-реляційне відображення;

Razor Pages – технологія генерації вебсторінок в ASP.NET Core;

SQL – Structured Query Language – мова структурованих запитів;

UI – User Interface – інтерфейс користувача.

ВСТУП

У сучасних умовах розвитку інформаційних технологій електронна комерція стрімко трансформується, пропонуючи нові можливості як для бізнесу, так і для споживачів. Однією з найбільш динамічних галузей онлайн-торгівлі є продаж жіночого одягу, де індивідуальний підхід до клієнтки відіграє вирішальну роль. Попри очевидні переваги онлайн-шопінгу – такі як зручність, широкий вибір, доступність з будь-якого місця – покупці часто стикаються з труднощами при виборі одягу через неможливість фізичної примірки. Як наслідок, це призводить до повернень, незадоволення від покупки та зниження лояльності до бренду.

Однією з ключових особливостей купівлі одягу є відповідність виробу до типу фігури. Невдало підібраний крій може зіпсувати враження від зовнішнього вигляду, тоді як вдало обраний фасон – підкреслити переваги силуету. Тип фігури має безпосередній вплив на естетичне сприйняття образу, а його врахування дає змогу зменшити кількість повернень товарів та підвищити рівень задоволеності споживачів. Однак сьогодні більшість інтернет-магазинів не враховують цей аспект, обмежуючись лише стандартними фільтрами за розміром, кольором або ціною.

Актуальність теми полягає в необхідності впровадження персоналізованого підходу до вибору одягу в умовах онлайн-продажів. У міру зростання популярності електронної торгівлі жінки все частіше зіштовхуються з труднощами під час купівлі одягу через інтернет. Відсутність можливості примірки та індивідуального підбору часто призводить до незадоволеності покупкою й значного рівня повернень. Саме тому розробка вебдодатку, який дозволяє визначити тип фігури на основі введених параметрів і формувати відповідні рекомендації щодо одягу, є актуальною задачею. Такий інструмент дозволить не лише підвищити якість користувацького досвіду, а й посилити конкурентоспроможність інтернет-магазинів, що особливо важливо в умовах високої ринкової конкуренції.

Метою даної роботи є створення вебдодатку, що поєднує традиційний каталог одягу з інтегрованою системою визначення типу фігури та персоналізованих рекомендацій. У межах дослідження буде виконано аналіз предметної галузі, розглянуто існуючі рішення, визначено переваги й недоліки різних підходів. Крім того, буде розроблено програмну архітектуру та реалізовано функціональні модулі, включаючи калькулятор типу фігури, фільтрацію товарів, рекомендаційний блок і зручний інтерфейс користувача.

Результати даної роботи можуть бути використані для поліпшення взаємодії між онлайн-продавцями та покупцями, сприяючи розвитку персоналізованої електронної комерції.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

Жіночий одяг є ключовою складовою модної індустрії, поєднуючи функціональність, естетику та відображення особистого стилю. До асортименту входять різноманітні категорії, такі як сукні, блузи, штани, спідниці та верхній одяг. Виробники постійно оновлюють дизайн, використовуючи сучасні матеріали та враховуючи модні тенденції, сезонність та індивідуальні потреби споживачів.

Важливим аспектом при виборі одягу є тип фігури, оскільки правильно підібрані фасони можуть візуально корегувати силует, підкреслюючи переваги та мінімізуючи недоліки [1]. Серед основних типів фігур виділяють такі варіанти (рисунок 1.1): «пісочний годинник», який характеризується врівноваженими обхватами грудей та стегон разом із вираженою талією; «груша» (або трикутник), де спостерігаються вузький верх та ширші стегна; «яблуко» з об’ємом у торсі та менш вираженою талією; «прямокутник», де груди, талія та стегна мають приблизно однакові обхвати; та «перевернутий трикутник», для якого характерний широкий верх та вузькі стегна.

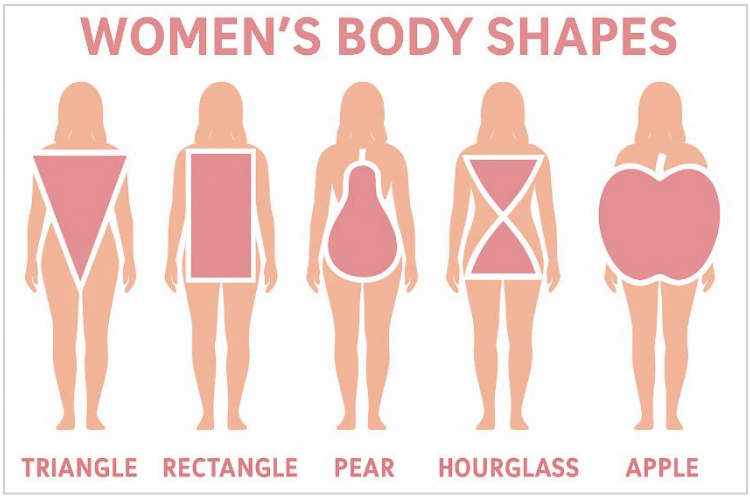


Рисунок 1.1 – Типи жіночих фігур

Зростання популярності електронної комерції зумовило потребу в інноваційних рішеннях для онлайн-шопінгу. Однак, на відміну від офлайн-покупок, клієнтки часто стикаються з труднощами через відсутність можливості приміряти одяг. В свою чергу, це призводить до помилкового вибору розмірів або фасонів, що не відповідають особливостям фігури, а отже – до повернень товарів та незадоволеності покупкою.

Сучасні технології, такі як алгоритми аналізу параметрів фігури та персоналізовані рекомендації, допомагають вирішити цю проблему. Впровадження подібних інструментів у вебдодатки для продажу одягу дозволяє запропонувати користувачкам оптимальні моделі, враховуючи їхній тип фігури, зменшити кількість повернень за рахунок більш точного підбору та підвищити задоволеність клієнтів завдяки індивідуалізованому підходу [2].

Таким чином, поєднання знань про типи фігур, тренди моди та цифрові технології створює основу для розвитку інноваційних рішень у сфері електронної торгівлі, спрямованих на покращення якості обслуговування та зростання лояльності клієнтів.

1.1 Особливості онлайн-продажу жіночого одягу

Сучасний ринок онлайн-продажу жіночого одягу переживає період активного розвитку, що обумовлено зростанням популярності електронної комерції та зміною поведінки споживачів. На відміну від традиційних магазинів, інтернет-торгівля пропонує покупцям унікальні переваги, такі як зручність, доступ до глобальних брендів і можливість швидкого порівняння пропозицій. Однак ця сфера має ряд специфічних характеристик, які впливають як на клієнтів, так і на продавців.

Однією з ключових особливостей є необхідність якісної візуалізації товару. Оскільки покупець не може фізично приміряти одяг, велике значення мають детальні фотографії, відеопрезентації та точні описи

матеріалів і розмірів. Також важливим елементом є відгуки покупців із реальними фото, які допомагають зменшити кількість повернень.

Персоналізація послуг займає особливе місце в онлайн-продажах жіночого одягу. Сучасні алгоритми аналізують історію переглядів і попередніх покупок, пропонуючи користувачам індивідуальні підбірки. Деякі магазини також запрошують професійних стилістів, які консультують клієнтів у режимі онлайн, допомагаючи створити гардероб з урахуванням їхніх уподобань.

Логістика залишається одним із найскладніших аспектів у цій сфері. Високий рівень повернень пов’язаний із тим, що покупці часто замовляють кілька варіантів одягу, щоб потім повернути непідходящі речі. Тому бренди розробляють гнучкі умови повернення, пропонують безкоштовну доставку або співпрацюють із мережами пунктів самовивозу для підвищення лояльності клієнтів.

Соціальні мережі відіграють важливу роль у просуванні жіночого одягу в інтернеті. Платформи, такі як Instagram, TikTok і Pinterest, стають основним джерелом натхнення для покупців. Бренди активно використовують візуальний контент, співпрацюють із інфлюенсерами та створюють естетично оформлені сторінки, щоб залучати аудиторію.

Таким чином, успішний онлайн-продаж жіночого одягу вимагає комплексного підходу, який поєднує технологічні інновації, розуміння потреб клієнтів та ефективні маркетингові стратегії. Ключовим фактором залишається здатність брендів адаптуватися до змін у поведінці споживачів і забезпечувати комфортний досвід покупок.

1.1.1 Переваги та недоліки онлайн-шопінгу

Онлайн-шопінг став невід’ємною частиною сучасного споживчого досвіду, особливо у сфері продажу жіночого одягу. Цей формат покупок має

як значні переваги, так і певні обмеження у порівнянні з традиційними офлайн-магазинами.

До основних переваг онлайн-шопінгу можна віднести зручність та доступність. Покупці можуть переглядати асортимент та оформляти замовлення в будь-який час доби, не виходячи з дому, що особливо важливо для зайнятих жінок, які не мають можливості витрачати час на походи по магазинах. Крім того, інтернет-магазини пропонують набагато ширший вибір товарів у порівнянні з фізичними торговими точками, включаючи доступ до міжнародних брендів та ексклюзивних колекцій.

Важливою перевагою є можливість порівняння цін та характеристик товарів від різних продавців за кілька кліків. Багато платформ пропонують зручні фільтри по розміру, кольору, стилю та іншим параметрам, що значно спрощує пошук потрібного товару. Додатковим плюсом є наявність знижок, акцій та кешбеку, які часто доступні саме при онлайн-покупках.

Однак онлайн-шопінг має і певні недоліки. Найбільшою проблемою залишається неможливість фізично приміряти одяг перед покупкою, що часто призводить до невідповідності очікувань та реальності, а також зумовлює високий рівень повернень товарів, що створює додаткові складнощі як для покупців, так і для продавців.

Важливим недоліком є суб’єктивність сприйняття товару через екран пристрою. Технічні особливості передачі зображення, налаштування дисплеїв та умови освітлення під час зйомки можуть спричиняти розбіжності між тим, що бачить покупець на фото, і реальними характеристиками товару. Крім того, цифрове представлення не дозволяє повною мірою оцінити тактильні якості матеріалів, їх щільність та фактичну структуру, створюючи певний ризик розчарування при отриманні замовлення, особливо коли йдеться про товари, де важливі нюанси кольору, фактури та якості обробки. Такі технологічні обмеження онлайн-шопінгу потребують від продавців додаткових заходів для забезпечення максимально точної візуалізації продукції.

Доставка товарів також може становити певні проблеми – від затримок у відправленні до додаткових витрат на пересилання. Крім того, деякі покупці відзначають відсутність емоційного досвіду, який зазвичай супроводжує походи по магазинах, зокрема можливість одразу отримати покупку та порадитися з продавцем.

Таким чином, хоча онлайн-шопінг пропонує численні переваги у вигляді зручності, доступності та широкого вибору, він також має певні обмеження, пов’язані з технічними аспектами та відсутністю фізичного контакту з товаром. Оптимальним рішенням для багатьох споживачів стає поєднання обох форматів – онлайн-дослідження товарів з подальшою покупкою в офлайн-магазині або навпаки.

1.1.2 Типи жіночих фігур та їх вплив на вибір одягу

У сучасній індустрії моди розуміння типів жіночих фігур є ключовим аспектом для створення ефективного асортименту одягу та забезпечення високого рівня клієнтської задоволеності. Аналіз антропометричних характеристик дозволяє визначити оптимальні моделі та крій, які найкраще підкреслять переваги та коректують особливості кожної фігури.

Існує кілька основних типів жіночих фігур, які відрізняються співвідношенням верхньої частини тіла, талії та стегон. Фігура «пісочний годинник» має приблизно однакову ширину плечей (або грудей) і стегон, при чітко вираженій тонкій талії. Тип «груша» (або «трикутник») характеризується вузькими верхом і більш широкими стегнами, в той час як для «перевернутого трикутника» притаманні широкий верх та вужчі стегна. Фігура «прямокутник» має слабко виражену талію, а груди і стегна майже однакового обхвату. Тип «яблуко» зазвичай має широку талію з менш вираженою різницею між грудьми і стегнами, а основний об’єм тіла зосереджений у середній частині.

Вплив типу фігури на вибір одягу має кілька ключових аспектів. В першу чергу, це вибір оптимальних силуетів: наприклад, жінкам з типом «трикутник» (груша) ідеально підходять А-силуети та трапецієподібні моделі, які врівноважують пропорції, тоді як від обтягуючих спідниць та вузьких брюк варто утриматись. Для типу «пісочний годинник» чудово пасуть приталені моделі, які підкреслюють тонку талію, проте мішкуватий одяг може приховати гармонійні пропорції.

По-друге, велике значення має підбір фактур та матеріалів. Жінкам з типом «прямокутник» варто обирати тканини зі структурою, які створюють ілюзію вигинів, у той час як тонкі тканини, що облягають, можуть підкреслити прямолінійність силуету. Для типу «овал» (яблуко) доречними будуть тканини середньої щільності з вертикальною орієнтацією, тоді як важкі об’ємні матеріали можуть додати зайвого об’єму.

По-третє, важливо враховувати розміщення декоративних елементів. Жінкам з типом «перевернутий трикутник» варто уникати масивних деталей у верхній частині одягу (накладні кишені, об’ємні рукави), оскільки вони можуть підкреслити ширину верхньої частини тіла. Натомість, декор у нижній частині (кишені на стегнах, оборки) допоможе урівноважити силует. Для всіх типів фігур актуальним правилом є уникання принтів, що контрастують з проблемними зонами – великі квіткові принти можуть візуально збільшувати ті ділянки, які бажано приховати.

1.2 Особливості онлайн-продажу жіночого одягу

Сучасний ринок онлайн-продажу жіночого одягу пропонує різні підходи до організації покупок. Традиційні інтернет-магазини зосереджені на каталогах товарів без персоналізації, тоді як більш інноваційні платформи вже інтегрують інструменти для визначення типу фігури та підбору відповідного одягу. Окремо існують спеціалізовані сервіси, які допомагають користувачам визначити свої параметри, але не завжди

пов’язують ці дані з конкретними товарами. Аналіз цих рішень дозволить виявити їхні переваги, недоліки та потенційні можливості для вдосконалення.

1.2.1 Zara

Zara є офіційним інтернет-магазином відомої іспанської мережі Zara, яка входить до складу холдингу Inditex [3]. Бренд спеціалізується на швидкій моді (fast fashion), пропонуючи клієнткам актуальні трендові моделі одягу, взуття та аксесуарів за доступними цінами. Онлайн-платформа магазину має сучасний інтерфейс із зручною навігацією, що дозволяє користувачам швидко знаходити потрібні товари за категоріями, кольорами, розмірами або ціновими діапазонами.

Однією з ключових особливостей Zara є оперативне оновлення асортименту – нові колекції з’являються на сайті щотижня, що дозволяє покупцям завжди бути в тренді. Магазин також пропонує послугу швидкої доставки та можливість безкоштовного повернення товарів, що робить покупки більш комфортними. Додатково на сайті є розділ із lookbook, де представлені готові образи, які допомагають клієнткам краще уявити, як поєднувати різні предмети одягу.

Проте, незважаючи на популярність бренду, Zara має низку недоліків, особливо з точки зору персоналізації. Магазин не надає інструментів для визначення типу фігури чи індивідуальних рекомендацій. Система підбору одягу обмежується стандартними фільтрами за розміром і кольором, що не враховує особливості різних типів фігур. В результаті покупці часто стикаються з проблемами, коли обраний одяг не ідеально підходить за фасоном, що призводить до необхідності повернення товарів.

Таким чином, хоча Zara є зручною та популярною платформою для покупок, її основним недоліком залишається відсутність персоналізованого підходу до вибору одягу.

1.2.2 The Concept Wardrobe

The Concept Wardrobe є спеціалізованим онлайн-ресурсом, який пропонує унікальний підхід до вибору жіночого одягу на основі теорії кольорових типів і аналізу фігури [4]. На відміну від традиційних інтернет-магазинів, цей проєкт зосереджений на створенні капсульного гардеробу, де кожен предмет одягу підбирається з урахуванням індивідуальних особливостей зовнішності та стилю користувача. Основна мета платформи – допомогти жінкам формувати стильний та функціональний гардероб, який ідеально підкреслює їхні природні дані.

Ключовою особливістю The Concept Wardrobe є комплексний підхід до аналізу фігури. Сайт надає детальні матеріали про різні типи фігур («пісочний годинник», «груша», «яблуко» тощо) і конкретні рекомендації щодо вибору фасонів, тканин і кольорів для кожного типу. На відміну від більшості звичайних магазинів, де підбір відбувається лише за розміром, цей ресурс враховує такі важливі фактори, як пропорції тіла, довжина конечностей, тип обличчя та інші індивідуальні характеристики.

Однак The Concept Wardrobe має певні обмеження. Найсуттєвішим недоліком є відсутність повноцінного інтернет-магазину з можливістю безпосередньої покупки рекомендованих товарів. Ресурс виконує переважно інформаційну та консультаційну функцію, не пропонуючи інтегрованого сервісу для придбання одягу, що насамперед змушує користувачів самостійно шукати рекомендовані моделі, що значно ускладнює процес створення ідеального гардероба.

Ще одним слабким місцем є дещо застарілий інтерфейс сайту, який не завжди зручний для користування на мобільних пристроях. Система навігації іноді виявляється надто складною для початківців. Крім того, відсутні інтерактивні інструменти для автоматичного аналізу фігури – всі рекомендації базуються на самостійному вивченні матеріалів та самоаналізі, що може бути неточним.

Незважаючи на ці недоліки, The Concept Wardrobe залишається цінним ресурсом для тих, хто серйозно підходить до створення свого стилю. Його унікальність полягає в науковому підході до підбору одягу, що вирізняє його серед більшості комерційних інтернет-магазинів. Впровадження подібних принципів у системи електронної комерції могло б значно підвищити якість онлайн-шопінгу, особливо для клієнтів, які цінують індивідуальний підхід і хочуть робити усвідомлений вибір одягу.

1.2.3 Gemini woman

Gemini woman є спеціалізованим британським онлайн-магазином жіночого одягу, який інтегрує систему підбору на основі типів фігури з можливістю безпосередньої покупки рекомендованих товарів [5]. Цей проєкт відрізняється від традиційних електронних торгових майданчиків своєю спробою поєднати теоретичні знання про особливості жіночих силуетів з практичною реалізацією персоналізованого шопінгу.

Основним інструментом підбору на сайті є спрощена система самодіагностики типу фігури. Користувачкам пропонується ознайомитись із візуальними схемами та короткими текстовими описами чотирьох основних типів фігур: «груша», «пісочний годинник», «прямокутник» та «перевернутий трикутник». Після візуального порівняння з прикладами, відвідувачки сайту можуть обрати свій варіант і перейти до відповідного розділу з підібраними моделями одягу.

Важливою перевагою Gemini woman є безпосередня інтеграція рекомендацій з торговим асортиментом. На відміну від багатьох інформаційних ресурсів, які лише описують принципи підбору, цей сайт дозволяє відразу придбати запропоновані моделі. Для кожного типу фігури підібрані конкретні фасони суконь, блузок, штанів та інших категорій одягу, що спрощує процес вибору для клієнток.

Однак система має ряд істотних методологічних обмежень. Відсутні точні алгоритми визначення параметрів тіла – весь аналіз ґрунтується на суб’єктивній візуальній оцінці та може призводити до помилок, особливо у випадках проміжних або нестандартних типів фігур.

Незважаючи на ці обмеження, досвід Gemini woman демонструє потенціал поєднання теоретичних знань про типи фігур з практичною реалізацією онлайн-продажів. Приклад свідчить, що навіть відносно прості системи персоналізації можуть покращити досвід клієнтів у порівнянні зі звичайними інтернет-магазинами.

1.2.4 Body shape calculator

Сучасні онлайн-калькулятори типу фігури представляють собою автоматизовані інструменти, які дозволяють визначити індивідуальні особливості жіночого силуету на основі об’єктивних вимірів. Принцип їх роботи ґрунтується на математичному аналізі співвідношення трьох ключових параметрів: обхвату грудей, талії та стегон. Користувач вводить свої виміри у відповідні поля, після чого алгоритм порівнює отримані дані зі стандартними співвідношеннями для різних типів фігур.

Основною перевагою таких калькуляторів є об’єктивність оцінки, оскільки вона базується на точних числових даних, а не на суб’єктивній візуальній оцінці. Після аналізу система надає короткий опис визначеного типу фігури («груша», «яблуко», «пісочний годинник» тощо) разом із базовими рекомендаціями щодо вибору одягу. Деякі розширені версії також включають поради щодо оптимальних фасонів, тканин та візуальних прийомів, що допомагають корегувати силует.

Проте ця методика має недолік у практичній реалізації. Користувач отримує лише загальні поради, але не має можливості одразу переглянути приклади підходячого одягу чи моделей, які б ідеально відповідали його типу фігури. Після визначення свого силуету людина змушена самостійно

шукати відповідні речі в інтернет-магазинах, що значно ускладнює процес створення гардероба. Більшість калькуляторів не надають візуальних прикладів, що знижує їх практичну цінність для споживачів. Крім того, навіть якщо рекомендації містять опис ідеальних фасонів, відсутність наочних зразків ускладнює їх правильне тлумачення, що суттєво для тих, хто не має досвіду в підборі одягу за типом фігури.

Незважаючи на цей недолік, калькулятори типу фігури залишаються популярним інструментом завдяки своїй простоті та доступності.

1.3 Порівняння та аналіз рішень

Аналіз сучасних рішень у сфері онлайн-продажу жіночого одягу демонструє різні підходи до інтеграції систем підбору за типом фігури. Традиційні магазини, такі як Zara, пропонують зручний інтерфейс та широкий асортимент, але не надають персоналізованих рекомендацій, що часто призводить до проблем із підбором одягу. На противагу цьому, спеціалізовані ресурси, як The Concept Wardrobe, пропонують глибокий аналіз фігури та детальні стилістичні поради, проте відокремлені від процесу покупки, що знижує їх практичну цінність.

Більш прогресивні рішення, зокрема Gemini Woman, демонструють спробу поєднати теоретичні знання про типи фігур з практичним шопінгом. Проте їх методика самодіагностики залишається надто спрощеною та суб’єктивною. Найбільш об’єктивний підхід пропонують калькулятори типу фігури, які базуються на точних вимірах, але не інтегровані з торговими платформами.

Ключовим висновком є необхідність створення комплексного рішення, яке поєднуватиме точні методи визначення типу фігури з безпосереднім доступом до відповідного асортименту одягу. Ідеальна система має включати об’єктивні алгоритми аналізу, наочні приклади підходящих моделей та можливість їх миттєвого придбання.

1.4 Постановка задачі

Метою даної роботи є розробка вебдодатку для продажу жіночого одягу з інтегрованою системою визначення типу фігури та персоналізованими рекомендаціями. Аналіз існуючих рішень показав, що більшість сучасних платформ мають суттєві обмеження: відсутність точних алгоритмів аналізу фігури або недостатню інтеграцію рекомендацій з торговим асортиментом.

Основна задача полягає у створенні комплексного рішення, яке поєднуватиме зручність традиційного інтернет-магазину з інноваційними інструментами персоналізації. Платформа передбачає реалізацію таких функціональних можливостей:

– каталог жіночого одягу з описами, зображеннями, цінами та категоріями;

– система фільтрації товарів за розміром, кольором, типом фігури, ціною, категорією, що дозволяє швидко знаходити відповідні варіанти;

– точний калькулятор визначення типу фігури на основі введених користувачкою параметрів;

– рекомендації одягу відповідно до визначеного типу фігури;

– адаптивний інтерфейс із реалізацією на базі сучасних вебтехнологій, серверною частиною на C# (ASP.NET Core), збереженням даних у базі PostgreSQL та підтримкою основних користувацьких сценаріїв.

2 ТЕОРЕТИЧНІ ВІДОМОСТІ ОБРАНИХ ТЕХНОЛОГІЙ

2.1 Технології створення серверної частини вебдодатку

Серверна частина вебдодатку є ключовим компонентом, що забезпечує взаємодію користувача з даними, бізнес-логікою та зберіганням інформації. Для створення сучасних вебсистем необхідно використовувати надійні, масштабовані та продуктивні технології, які дозволяють реалізувати обробку запитів, захист даних, авторизацію, інтеграцію з базами даних, а також дотримання принципів розподіленої архітектури.

У рамках даної роботи для реалізації серверної частини вебдодатку було обрано фреймворк ASP.NET Core, який є потужним засобом для створення вебдодатків, API та мікросервісів на базі платформи .NET [6]. Його особливістю є кросплатформеність, підтримка високої продуктивності, модульність та інтеграція з сучасними стандартами безпеки.

Разом із ASP.NET Core застосовується архітектурна модель MVC (Model-View-Controller), яка дозволяє логічно розділити додаток на три окремі компоненти: модель, представлення та контролер [7]. Такий підхід значно полегшує підтримку, масштабування та тестування проєкту.

У поєднанні з іншими інструментами, зокрема Entity Framework Core, Razor Pages, засобами автентифікації, авторизації, ASP.NET Core дає змогу створити надійну серверну інфраструктуру для вебдодатків будь-якої складності. У контексті реалізації системи персоналізованих рекомендацій для жіночого одягу ці технології забезпечують гнучке керування даними користувачів, обробку запитів щодо визначення типу фігури та генерації рекомендацій, а також інтеграцію з базою даних PostgreSQL.

2.1.1 Фреймворк ASP.NET Core та його можливості

ASP.NET Core – це сучасний високопродуктивний фреймворк від компанії Microsoft, призначений для створення вебдодатків, API та хмарних сервісів. Його перевагою є відкритий вихідний код, кросплатформеність (Windows, Linux, macOS) і активна підтримка спільноти. ASP.NET Core є повністю переписаною версією попереднього ASP.NET, зосередженою на модульності, продуктивності та зменшенні залежностей [6].

Однією з ключових можливостей ASP.NET Core є вбудована підтримка залежностей (dependency injection), що спрощує організацію коду та підвищує його тестованість. Крім того, він має потужну систему маршрутизації, яка дозволяє легко обробляти HTTP-запити.

Фреймворк також підтримує інтеграцію з шаблонами Razor, які дозволяють створювати динамічні вебсторінки з розділенням логіки й уявлення. Для роботи з базами даних ASP.NET Core легко поєднується з Entity Framework Core, що забезпечує ORM-підхід до управління даними.

Для даного проєкту ASP.NET Core забезпечує основу для створення серверної логіки, обробки запитів користувача, зберігання даних параметрів фігури, формування рекомендацій на основі введених значень, а також підтримки інтерактивного інтерфейсу з клієнтською частиною.

2.1.2 Патерн MVC

Архітектурний патерн MVC (Model-View-Controller) є одним із найбільш поширених підходів у веброзробці. Його основною метою є розділення логіки додатку на три взаємопов’язані, але незалежні компоненти, кожен з яких виконує чітко визначену функцію (рисунок 2.1). Це дозволяє підвищити гнучкість, спростити масштабування та покращити підтримку програмного забезпечення [7].

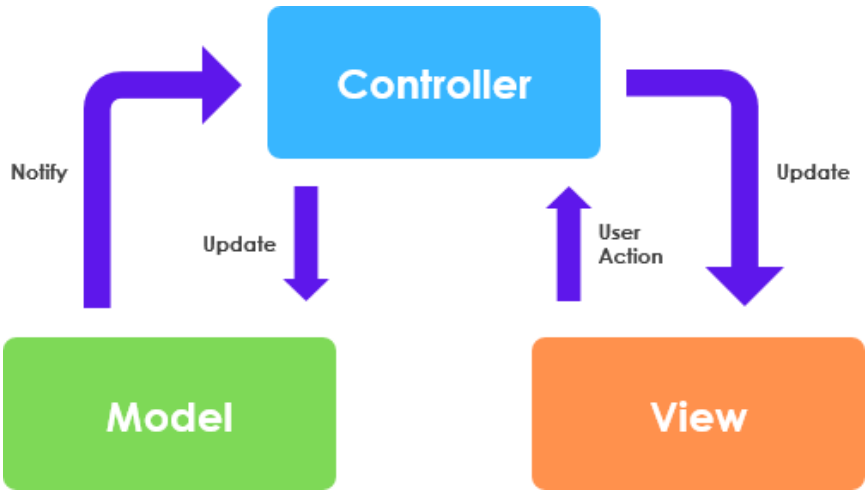


Рисунок 2.1 – Схема роботи патерну MVC

Модель (Model) представляє собою частину, відповідальну за доступ до даних і бізнес-логіку. Вона включає сутності, які описують структуру даних, та методи для взаємодії з базою даних. У межах проєкту ця складова реалізується за допомогою класів C# та бібліотеки Entity Framework Core, що забезпечує зручну ORM-інтеграцію з PostgreSQL.

Представлення (View) відповідає за відображення інформації кінцевому користувачеві. Воно формує інтерфейс у вигляді HTML-сторінок, які створюються з використанням Razor-шаблонів. Razor дозволяє комбінувати C#-код із HTML, що дає змогу динамічно змінювати вміст сторінок у залежності від даних, отриманих із моделі.

Контролер (Controller) є центральним елементом логіки обробки запитів у архітектурі MVC. Його основне завдання – отримувати HTTP-запити від користувача, аналізувати їх вміст, ініціювати відповідну дію в моделі, а потім формувати відповідь і передавати її до представлення. Контролер фактично виступає «координатором» між користувачем, моделлю та представленням.

Перевага застосування MVC полягає у чіткому розділенні обов’язків між компонентами, що дозволяє одночасно розробляти різні частини проєкту незалежними командами, які підвищують зрозумілість і

структурованість коду, спрощують модифікацію окремих частин без ризику порушити роботу всієї системи, а також значно покращують тестування й налагодження.

У рамках реалізації даного проєкту використання патерну MVC забезпечує чисту архітектуру, що легко розширюється, адаптується до нових функцій і підтримує інтеграцію з іншими сервісами.

2.2 Робота з базою даних у вебдодатку

База даних відіграє ключову роль у функціонуванні сучасних вебдодатків, забезпечуючи зберігання, пошук, оновлення та видалення даних. Для реалізації доступу до бази даних у цьому проєкті використовується Entity Framework Core – сучасна ORM-бібліотека, яка дозволяє працювати з даними на рівні об’єктів, не оперуючи безпосередньо SQL-запитами. Завдяки підтримці LINQ, механізму міграцій і високій інтегрованості з ASP.NET Core, EF Core забезпечує зручну та безпечну роботу з даними, мінімізуючи ризики помилок і прискорюючи розробку [8].

Як система керування базами даних обрано PostgreSQL – потужну, стабільну та масштабовану СУБД з відкритим кодом. Вона має багатий функціонал, підтримує складні запити, транзакції та індексацію, що робить її придатною для додатків з великим обсягом інформації. У поєднанні з EF Core, PostgreSQL дозволяє легко реалізувати гнучку структуру даних, необхідну для роботи алгоритмів персоналізованих рекомендацій і зберігання історії дій користувачів [9].

2.2.1 Використання Entity Framework Core

Entity Framework Core (EF Core) є сучасним об’єктно-орієнтованим фреймворком для роботи з базами даних, який належить до технологій ORM (Object-Relational Mapping). Він дозволяє розробникам взаємодіяти з

реляційними базами даних, використовуючи об’єктну модель, без необхідності писати складні SQL-запити вручну. Підхід значно спрощує процес розробки, оскільки забезпечує абстракцію над схемою бази даних, автоматизує створення запитів і керування даними, а також підтримує різні СУБД, такі як SQL Server, PostgreSQL, MySQL та SQLite [8].

Однією з ключових переваг EF Core є підтримка підходу Code First, який дає змогу спочатку визначити модель даних у вигляді класів C#, а потім автоматично генерувати схему бази даних на основі цих класів. Це дозволяє зосередитися на логіці додатку, а не на деталях реалізації бази даних. Крім того, EF Core підтримує міграції, що дає можливість інкрементально оновлювати структуру бази даних при зміні моделі, зберігаючи існуючі дані.

Фреймворк також забезпечує високий рівень продуктивності завдяки механізмам кешування, лінивого завантаження (Lazy Loading) та явного завантаження (Eager Loading), що дозволяє оптимізувати запити до бази даних. Підтримка LINQ (Language Integrated Query) дозволяє будувати складні запити прямо в коді C#, що робить роботу з даними більш інтуїтивною та безпечною з точки зору типізації.

Ще одна важлива характеристика EF Core – це його модульність та крос-платформенність. Він може використовуватися не лише у традиційних вебдодатках на ASP.NET Core, але й у мобільних та десктопних програмах.

Таким чином, Entity Framework Core є потужним інструментом для роботи з даними, який поєднує у собі зручність розробки, гнучкість налаштувань та високу продуктивність. Його використання дозволяє прискорити створення програмного забезпечення, зменшити кількість шаблонного коду та забезпечити стабільну роботу з різними типами баз даних.

2.2.2 СУБД PostgreSQL

PostgreSQL – це об’єктно-реляційна система керування базами даних (СУБД) з відкритим кодом, яка відповідає стандарту SQL і вирізняється високою відповідністю ACID-принципам, що гарантує цілісність і надійність даних у будь-яких сценаріях використання [9].

PostgreSQL підтримує JSON, XML, GIS, повнотекстовий пошук, а також паралельне виконання запитів, що дозволяє будувати високопродуктивні та масштабовані рішення.

Завдяки відкритості архітектури PostgreSQL, вона легко інтегрується з різними мовами та фреймворками. У середовищі .NET для роботи з PostgreSQL зазвичай використовується офіційний провайдер Npgsql, який забезпечує повну сумісність з Entity Framework Core. Npgsql дозволяє використовувати переваги ORM-підходу без втрати продуктивності, підтримує функції PostgreSQL, такі як типи масивів, збережені процедури, специфічні типи даних (наприклад, uuid, inet, jsonb) тощо.

Завдяки поєднанню з EF Core, PostgreSQL отримує ще один рівень абстракції – розробники можуть зосередитись на моделюванні предметної галузі через класи, а не на деталях SQL-запитів. Такий підхід підвищує продуктивність розробки, зменшує кількість помилок та спрощує обслуговування бази впродовж життєвого циклу додатку.

2.3 Розробка інтерфейсу користувача

Інтерфейс користувача (UI) є однією з найважливіших складових будь-якого вебдодатку, оскільки саме через нього відбувається безпосередня взаємодія з користувачем. Якісний UI забезпечує зручність навігації, швидкість доступу до функцій, а також формує загальне враження від системи. Розробка інтерфейсу має враховувати як технічні вимоги, так і аспекти ергономіки, дизайну та доступності.

У межах сучасного веброзроблення широкого поширення набули такі технології, як Razor Pages – для генерації динамічного вмісту сторінок, та Bootstrap – як CSS-фреймворк для побудови адаптивного, зручного та привабливого візуального оформлення. Обидві технології тісно інтегруються з платформою ASP.NET Core і дозволяють будувати повноцінні інтерфейси без використання надлишкових бібліотек або зовнішніх рішень.

Застосування Razor Pages спрощує структуру розробки сторінок, дозволяє відокремити логіку від уявлення та створювати динамічний HTML-контент. Водночас, використання Bootstrap значно прискорює процес стилізації інтерфейсу, забезпечуючи при цьому сумісність із мобільними пристроями, правильне вирівнювання елементів, сучасний вигляд форм, таблиць, навігаційних панелей тощо. У сукупності ці технології забезпечують побудову UI, який поєднує гнучкість, зручність і професійний дизайн.

2.3.1 Razor Pages

Razor Pages – це сучасна технологія від Microsoft для створення вебдодатків на платформі ASP.NET Core, яка спрощує розробку динамічних сторінок завдяки своїй модульній та орієнтованій на сторінки архітектурі. Razor Pages об’єднує логіку обробки запитів і відображення даних в єдиному компоненті – сторінці, що робить код більш структурованим і легшим для розуміння, особливо при створенні невеликих та середніх вебдодатків.

Ключовою перевагою Razor Pages є використання синтаксису Razor, який дозволяє поєднувати HTML-розмітку з C#-кодом прямо у файлах представлення. Це дає змогу динамічно генерувати контент на стороні сервера, використовуючи дані з моделей, без необхідності застосовувати додаткові JavaScript-фреймворки. Наприклад, можна легко ітерувати

колекції, виводити умовний контент або форматувати дані безпосередньо у розмітці, що значно прискорює процес розробки.

Завдяки своїй простоті, продуктивності та гнучкості, Razor Pages є відмінним вибором для побудови динамічних вебдодатків, де важлива швидкість розробки та зручність підтримки. Вони дозволяють розробникам створювати сучасні та інтерактивні сторінки з мінімальними зусиллями, зберігаючи при цьому високу продуктивність і безпеку.

2.3.2 CSS-фреймворк Bootstrap

Bootstrap – це популярний CSS-фреймворк з відкритим кодом, який використовується для розробки адаптивних та кросбраузерних вебінтерфейсів [10].

Однією з основних переваг Bootstrap є адаптивна сітка (grid system), яка дозволяє будувати інтерфейси, що автоматично підлаштовуються під розмір екрана. Важливість цього особливе для забезпечення коректного відображення сайту на мобільних пристроях, планшетах та моніторах різної роздільної здатності.

Фреймворк включає велику бібліотеку готових UI-елементів, таких як кнопки, форми, панелі навігації, модальні вікна, сповіщення, та інші. Він також підтримує JavaScript-компоненти (наприклад, каруселі, випадаючі списки), що легко підключаються без написання складного коду.

Bootstrap добре інтегрується з Razor Pages та іншими технологіями ASP.NET, дозволяючи швидко створювати сучасні, візуально привабливі інтерфейси. Його використання значно прискорює процес верстки та полегшує підтримку єдиного стилю в межах усього вебдодатку.

3 ПРОЄКТУВАННЯ ТА РОЗРОБКА ВЕБДОДАТКУ

3.1 Технічне завдання та функціональні вимоги

Вебдодаток планується як онлайн-майданчик для реалізації жіночого одягу, обладнаний функцією індивідуального підбору з урахуванням особливостей фігури. Мета полягає у створенні не лише можливості придбати конкретну річ, а й допомогти спростити процес її вибору. Користувач при введені своїх параметрів має отримати від системи згенеровані рекомендації таким чином, щоби інтерфейс не створював тиску на людину, а також виключав її потребу в глибшої обізнаності про фасони.

Сервіс, не обмежуючись стандартним маркетплейсом, будується навколо досвіду користувача – з можливістю визначити тип фігури, побачити речі, які пасують саме їй, а не слідувати виключно трендам. Додаток не просто демонструє асортимент, а й навчається на введених даних, щоб видавати якомога влучніші варіанти. Тобто, виконує певну роль стиліста

Сторінки з одягом має мати функціонал фільтрації за наступними категоріями: типом одягу, кольором, розміром, вартістю, типом фігури, – що є найважливішим. Слідуючи цим принципам, користувач уникає ситуації, коли фасон виглядає привабливо на моделі, але абсолютно не пасує їй. Очевидно, для побудови процесу інтернет-маркетингу має бути реалізованим додавання товарів до кошику, можливість оформити замовлення, створення адресної книжки для зручності оформлення замовлення авторизованому користувачеві, а також можливість перегляду історії покупок.

Також обов’язковою умовою для працездатності ресурсу має бути виокремлення ролі адміністратора. Він повинен мати доступ до списку товарів, можливість редагувати або додавати нові позиції, оновлювати фото,

змінювати ціни та описи. Також йому має бути доступне управління категоріями, статусами замовлень та перегляд статистики.

З приводу можливого хибного введення даних в поля, то всі вони мають бути обладнаним конкретними обмеженнями: за розміром числа, допустимим діапазоном, відсутністю пустих значень. За виникненням помилки, система має показати зрозуміле повідомлення для того, щоб уникнути помилок в працездатності додатку.

Всі процеси відбуватимуться в режимі реального часу. Введені параметри миттєво аналізуються, і після короткої обробки користувач має спостерігати результат.

Ще однією умовою є кросплатформеність ресурсу. Додаток повинен коректно працювати на середньостатистичному ноутбуці або смартфоні.

3.2 Архітектура та структура проєкту

Під час реалізації вебдодатку передбачено застосування архітектурної моделі Model-View-Controller (MVC), яка сприятиме впорядкованій організації програмної логіки та розподілу відповідальностей між її ключовими частинами. В свою чергу, підхід гарантує керованість системи на етапах розробки, спростить тестування окремих модулів та полегшить майбутнє масштабування функціональності.

Додаток функціонуватиме за принципом клієнт-серверної моделі, де сервер відповідатиме за обробку запитів, зберігання даних і виконання бізнес-логіки. Клієнтська частина, у свою чергу, відповідатиме за взаємодію з користувачем, відображення інтерфейсу та передавання запитів відповідним обробникам. Комунікація між сторонами здійснюватиметься за допомогою стандартних протоколів HTTP.

У межах проєкту сформовано чітку структуру каталогів. Моделі описуватимуть ключові сутності предметної галузі, контролери відповідатимуть за логіку обробки запитів, а представлення – за відтворення

даних у вигляді сторінок. Крім того, буде виділено допоміжні модулі, які забезпечать підтримку специфічних сервісів, наприклад, механізм персоналізованих рекомендацій або розпізнавання типу фігури. Структура додатку зазначена на рисунку 3.1.

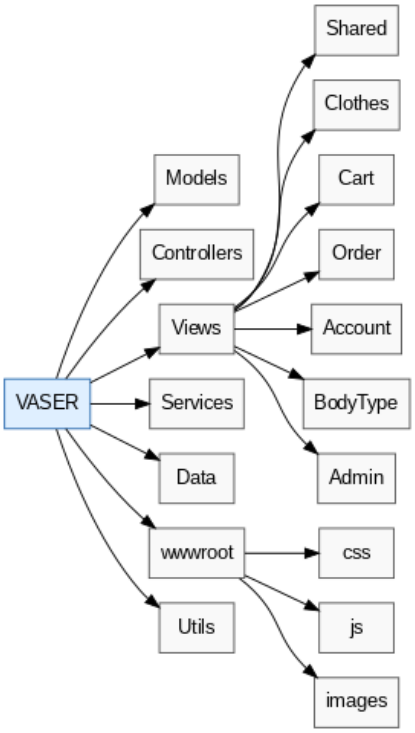


Рисунок 3.1 – Архітектура вебдодатку

На етапі проєктування передбачено створення окремої папки Data для роботи з базою даних, де буде визначено контекст та початкові налаштування з’єднання. Окремо будуть розглянуті питання структури даних, взаємозв’язків між таблицями, а також ініціалізації тестовими даними, що дозволить спростити перевірку основного функціоналу у подальшому.

Передбачається, що всі статичні ресурси (зображення, стилі, скрипти) будуть згруповані в окремому каталозі wwwroot, що забезпечить зрозумілу структуру та спростить побудову зовнішнього вигляду додатку та функціоналу скриптів.

З точки зору вибору технологій, ASP.NET Core є основним каркасом, на який буде нашаровано решту. Для роботи з базою даних планується застосування ORM-фреймворку Entity Framework Core, що дасть можливість описувати структуру, використовуючи засоби задавання структури бази даних через моделі, що включає в собі підтримку автоматичного створення таблиць. Для зберігання даних передбачається використовувати PostgreSQL як надійну та масштабовану систему управління базами даних.

Також в клієнтській частині планується використання Razor-шаблонів і загальновживаних технологій, таких як: HTML, CSS та JavaScript. Дані дії дозволять створити інтерфейс, адаптований під різні пристрої, та забезпечити базову валідацію даних без використання сторонніх фреймворків.

В результаті застосування обраної архітектури, структура додатку буде логічно впорядкованою та придатною до подальшого розвитку та забезпечення ефективної реалізації поставлених задач, зберігши при цьому гнучкість у зміні або доповненні окремих компонентів системи.

3.3 Проєктування бази даних

Для втілення основного функціоналу вебдодатку критично важливо забезпечити коректне моделювання предметної галузі шляхом створення системи взаємопов’язаних сутностей. У рамках цієї системи планується розробка таблиць, які відображатимуть основні об’єкти: користувачі, замовлення, товари, категорії, параметри фігури та інші. Схема зображена на рисунку 3.2. Окрім структури самих моделей, особливу увагу слід приділити зв’язкам між ними, що дозволить зберегти цілісність даних та гарантувати логічну навігацію всередині програми.

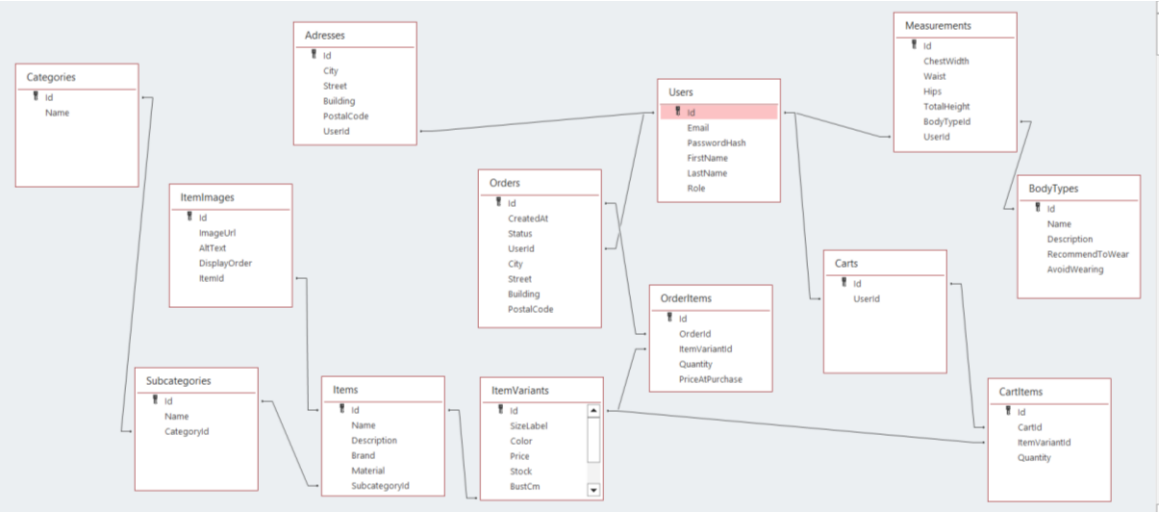


Рисунок 3.2 – Схема бази даних проєкту

Відповідність між таблицями та логікою предметної галузі буде встановлюватись з використанням ORM-фреймворку. Передбачається застосування конфігурації через ApplicationDbContext, де для кожної сутності буде визначено опис, зв’язки, обмеження, а також стратегію поведінки при видаленні. Такий підхід дозволить мінімізувати повторення коду, зберегти структурованість проєкту та автоматизувати синхронізацію моделі з базою даних.

3.3.1 Користувачі

У структурі бази даних модель користувача займає ключову позицію, адже більшість операцій в системі пов’язані з акаунтом (рисунок 3.3). Запис у таблиці Users міститиме унікальний ідентифікатор, електронну пошту, пароль, а також особисту інформацію, як-от ім’я та прізвище. Поле Email планується як обов’язкове для заповнення, з обмеженням довжини та унікальністю, оскільки саме на нього спиратиметься система аутентифікації. Пароль зберігатиметься у вигляді хешу (PasswordHash), без відображення відкритого тексту.

Додатково в таблиці передбачено атрибут Role, що дасть змогу визначати права доступу, наприклад, розділяти між адміністративним обліковим записом і звичайним користувачем. Ім’я (FirstName) та прізвище (LastName) – обов’язкові поля, що дозволяють персоналізувати взаємодію з інтерфейсом.

Рисунок 3.3 – Клас User

Один із ключових аспектів цієї моделі – зв’язок із параметрами тіла. Планується, що кожен користувач зможе один раз або декілька разів вводити свої антропометричні дані, що зберігатимуться в окремій таблиці вимірювань. Через властивість навігації Measurement буде реалізовано зв’язок «один до одного», що дозволить системі точно визначати тип фігури та формувати рекомендації.

Також користувач матиме можливість зберігати декілька адрес доставки. Зв’язок між таблицями Users та Addresses буде реалізовано за принципом «один до багатьох», що дозволить, за необхідності, додавати декілька варіантів отримання товарів.

Крім цього, через таблицю замовлень передбачено логіку збереження історії покупок. Кожен користувач матиме доступ до переліку своїх замовлень, які будуть прив’язані до нього через зовнішній ключ, що дозволить не лише переглядати попередні покупки, але й формувати аналітику або повторні замовлення на основі попереднього досвіду.

3.3.2 Адреси доставки

Таблиця адрес доставки буде призначена для зберігання локацій, де користувачі можуть отримувати свої замовлення (рисунок 3.4). Структура передбачає зберігання основних атрибутів адреси: назву міста, вулицю, номер будинку та, за потреби, поштовий індекс. Поля «City» та «Street» є обов’язковими – без них неможливо сформувати коректну адресу. «Building» та «PostalCode», навпаки, будуть опціональними, оскільки в деяких випадках доставка може відбуватися без уточнення номера будинку або індексу.

Кожен запис у цій таблиці буде жорстко прив’язаний до певного користувача, який забезпечуватиметься зовнішнім ключем «UserId», що вказуватиме на відповідний запис у таблиці «Users». Таким чином, реалізується зв’язок типу «один до багатьох», що дозволить зберігати історію попередніх адрес або надавати вибір під час оформлення нового замовлення.

У системі планується реалізація адресної книги в особистому кабінеті користувача. Кожну адресу можна буде переглянути, відредагувати або видалити. Така можливість особливо зручна для користувачів, які часто замовляють товари на різні локації – наприклад, додому та на роботу. Під час оформлення покупки система автоматично пропонуватиме останню використану адресу, але з можливістю вибору іншої з переліку.

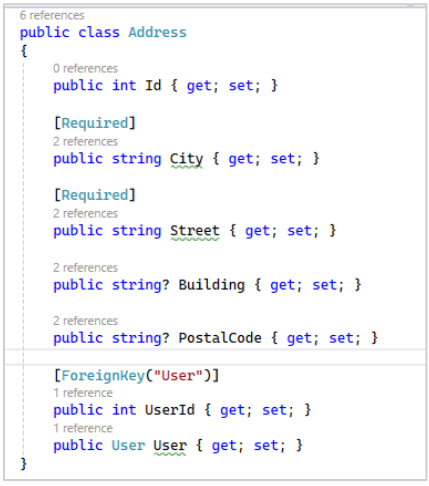


Рисунок 3.4 – Клас Adress

З точки зору зберігання даних, таблиця є простою, але доволі важливою для коректної логістики. Саме на основі цієї інформації система формуватиме маршрут доставки, розраховуватиме вартість послуги та надсилатиме замовлення у відповідне місце.

3.3.3 Вимірювання та типи фігур

Для персоналізованого підбору одягу необхідно враховувати індивідуальні особливості тіла користувача. З цією метою в структурі бази даних передбачено окрему таблицю вимірів, де будуть зберігатися антропометричні дані, такі як обхват грудей, талії, стегон та загальний зріст (рисунок 3.5). Поля ChestWidth, Waist, Hips та TotalHeight будуть числовими з обмеженнями на мінімальні та максимальні значення, що дозволить уникнути помилок під час введення даних.

Ці дані зберігатимуться, будучи пов’язаними з користувачем за допомогою зовнішнього ключа UserId. Проте, запис у таблиці Measurements може бути відсутнім, наприклад, якщо користувач ще не ввів свої параметри. Цей зв’язок реалізується як «один до одного», що забезпечує логічну відповідність: один користувач – один набір вимірів.

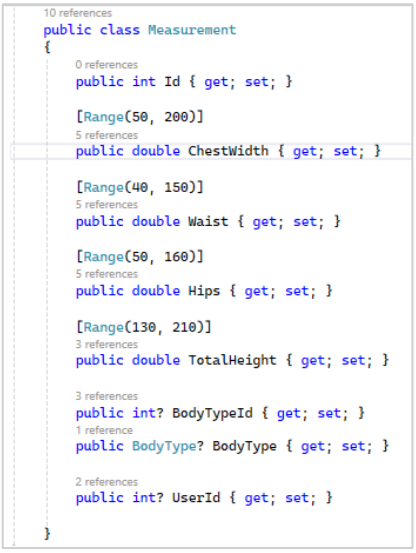


Рисунок 3.5 – Клас Measurment

На основі введених вимірів (рисунок 3.6) буде визначатися тип фігури. Для цього проєктом передбачено окрему таблицю BodyTypes, яка міститиме перелік попередньо визначених типів: пісочний годинник, груша, прямокутник, яблуко та перевернутий трикутник. Кожен тип супроводжуватиметься описом та текстовими рекомендаціями щодо вибору одягу.

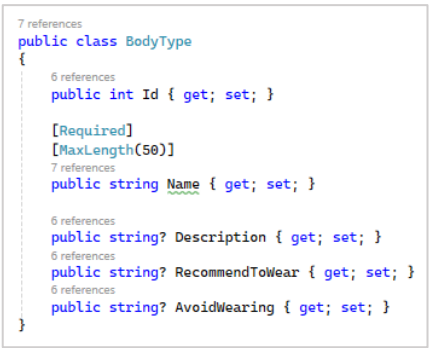


Рисунок 3.6 – Клас BodyType

Зв’язок між вимірами та типом фігури встановлюється через зовнішній ключ BodyTypeId, який може бути порожнім до моменту

визначення результату. Визначення типу фігури відбуватиметься автоматично після обробки введених вимірів, використовуючи окремий сервіс, але зберігатиметься в цій таблиці.

Поєднання двох сутностей дозволяє створити не просто каталог одягу, а справді адаптивну систему, яка враховує пропорції тіла та генерує індивідуальні рекомендації.

3.3.4 Категорії та підкатегорії

Для приведення асортименту до ладу в структурі бази даних планується запровадження системи класифікації, що включатиме категорії та підкатегорії, що дозволить не тільки логічно організувати наповнення каталогу, а й забезпечити гнучку систему фільтрації, комфортну як для користувачів, так і для адміністраторів (рисунки 3.7–3.8).

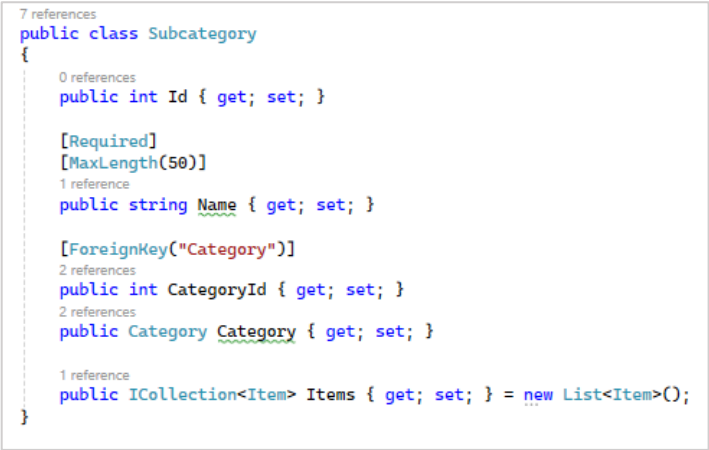
```
5 references
public class Category
{
    0 references
    public int Id { get; set; }

    [Required]
    [MaxLength(50)]
    1 reference
    public string Name { get; set; }


    2 references
    public ICollection<Subcategory> Subcategories { get; set; } = new List<Subcategory>();
}
```

Рисунок 3.7 – Клас Category

Таблиця Categories зберігатиме основні розділи товарів, наприклад: верхній одяг, низ, сукні, та інші. Кожна категорія матиме унікальний ідентифікатор та обов’язкове поле з назвою. Всередині кожної категорії зберігатиметься список підкатегорій, що конкретизують вид товару, наприклад, в категорії «верхній одяг» можуть бути «светри», «куртки» або «пальта».



```
7 references
public class Subcategory
{
    0 references
    public int Id { get; set; }

    [Required]
    [MaxLength(50)]
    1 reference
    public string Name { get; set; }

    [ForeignKey("Category")]
    2 references
    public int CategoryId { get; set; }
    2 references
    public Category Category { get; set; }

    1 reference
    public ICollection<Item> Items { get; set; } = new List<Item>();
}
```

Рисунок 3.8 – Клас Subcategory

Така деталізація буде реалізована через таблицю Subcategories, яка міститиме посилання на свою батьківську категорію за допомогою зовнішнього ключа CategoryId. Відтак, структура формуватиметься за принципом «один до багатьох»: кожна категорія – кілька підкатегорій. Найменування підкатегорій також фіксуватимуться у вигляді окремих записів, і жодна з них не зможе існувати окремо, без прив’язки до певної категорії.

З технічного боку, цей розподіл дасть змогу підтримувати каталог в організованому вигляді, забезпечить можливість швидкого пошуку товарів за типом, а також дозволить вільно оновлювати або розширювати номенклатуру, не вносячи змін у загальну структуру даних.

Для адміністраторської частини така класифікація є вкрай важливою: саме вона визначає, де і як показуватиметься товар, які фільтри будуть до нього застосовані, та до якої групи він відноситься. З боку користувача – це фундамент інтуїтивної навігації по сайту, коли спершу обирається загальна категорія, а потім – уточнюється тип товару в межах обраної групи.

3.3.5 Товари

Основою у системі продажу виступає товар. У базі даних інформація про нього зберігатиметься через взаємопов’язані таблиці, що разом створюватимуть вичерпний опис кожної позиції: основної інформації, характеристики кольору та розміру, зображень.

Таблиця Items зберігатиме основну інформацію про товар – його назву, опис, бренд, матеріал, а також прив’язку до певної підкатегорії (рисунок 3.9). В межах одного запису зберігатиметься лише узагальнена характеристика моделі, без деталізації розмірів або ціни, що дозволить уникнути дублювання інформації, оскільки одна й та сама модель одягу може мати різні конфігурації.

```
25 references
public class Item
{
    8 references
    public int Id { get; set; }

    [Required]
    [MaxLength(100)]
    9 references
    public string Name { get; set; }

    1 reference
    public string? Description { get; set; }

    [MaxLength(50)]
    1 reference
    public string? Brand { get; set; }

    [MaxLength(50)]
    2 references
    public string? Material { get; set; }

    [ForeignKey("Subcategory")]
    1 reference
    public int SubcategoryId { get; set; }
    5 references
    public Subcategory Subcategory { get; set; }

    16 references
    public ICollection<ItemVariant> Variants { get; set; } = new List<ItemVariant>();

    11 references
    public ICollection<ItemImage> Images { get; set; } = new List<ItemImage>();
}
```

Рисунок 3.9 – Клас Item

Кожен товар матиме один або більше варіантів відображатиметься в таблиці ItemVariants (рисунок 3.10). Тут буде зафіксовано конкретний розмір (у вигляді розмірної мітки, наприклад S або M), колір, вартість, залишок на складі, а також, за потреби, граничні параметри фігури, для якої

варіант рекомендований. В свою чергу, це відкриє можливість поєднання варіантів із системою персоналізованих рекомендацій.

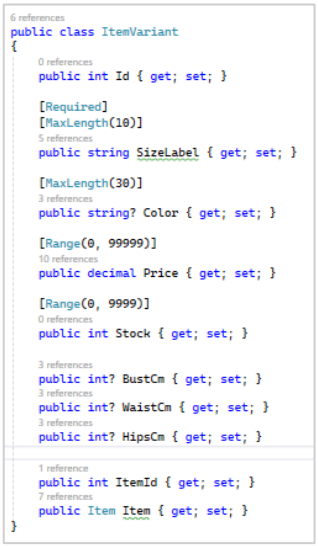


Рисунок 3.10 – Клас ItemVariant

Для візуального представлення товару буде залучено таблицю ItemImages, яка дозволить прив’язувати до кожної моделі набір зображень. Знімки можуть мати порядковий номер відображення (наприклад, для формування галереї), а також короткий альтернативний текст, що забезпечить базову доступність і кращу індексацію (рисунок 3.11).

Рисунок 3.11 – Клас ItemImage

Зв’язки між цими таблицями будуть реалізовані за принципом «один до багатьох»: кожен Item матиме декілька ItemVariants і декілька ItemImages. Всі зміни, як у вартісних характеристиках, так і в асортименті, відбуватимуться саме на рівні варіантів, що дозволить гнучко оновлювати дані без потреби повторного прив’язування або створення дублікатів моделей.

3.3.6 Кошик

Кошик – це тимчасове місце, де зберігаються товари, які користувач має намір придбати. У базі даних ця логіка втілюватиметься через дві взаємопов’язані таблиці: Carts та CartItems, що допомагає сформувати приналежність до користувача та наповнювати його вміст товарами.

Таблиця Carts (рисунок 3.12) зберігатиме посилання на користувача (UserId), якому належить цей кошик. В системі передбачено, що кожен користувач матиме лише один активний кошик, який оновлюється під час додавання або видалення товарів. Після завершення оформлення замовлення його вміст очищується.

```
4 references
public class Cart
{
    1 reference
    public int Id { get; set; }

    [ForeignKey("User")]
    4 references
    public int UserId { get; set; }
    0 references
    public User User { get; set; }

    9 references
    public ICollection<CartItem> Items { get; set; } = new List<CartItem>();
}
```

Рисунок 3.12 – Клас Cart

У таблиці CartItems (рисунок 3.13) описуються конкретні товари у кошику. Кожний запис тут являє собою окрему позицію, що включає посилання на конкретний варіант товару (ItemVariantId) та вказівку

кількості одиниць. Зв’язок реалізується через зовнішній ключ CartId, що об’єднує позиції з відповідним кошиком. Один кошик може містити кілька позицій, кожна з яких має власну кількість.

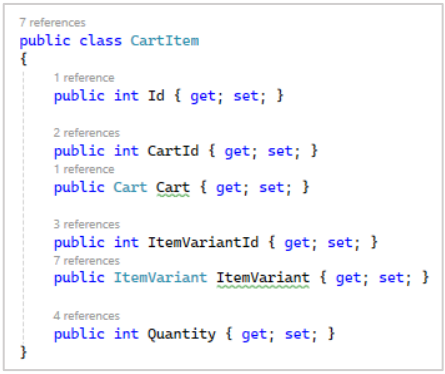


Рисунок 3.13 – Клас CartItem

Окреме використання такої структури дозволяє легко маніпулювати вмістом кошика: змінювати кількість, видаляти елементи, перевіряти наявність на складі. Під час оформлення замовлення ці записи переносяться до відповідної структури замовлення, після чого кошик очищується для подальшого використання.

3.3.7 Замовлення

Система замовлень є фінальною ланкою у ланцюзі покупки, її архітектура зобов’язана зберігати всю важливу інформацію про транзакцію: хто, що та коли замовив, куди доставити та який поточний стан кожного замовлення. З цією метою передбачено дві основні сутності – Orders та OrderItems, які вміщують як загальну інформацію, так і детальний опис вмісту (рисунки 3.14–3.15).

В таблиці Orders розміщується загальна інформація про замовлення. Вона містить посилання на користувача (UserId), дату створення (CreatedAt), статус обробки («Нове», «Сплачено», «Виконано»,

«Скасовано»), а також дані про адресу доставки – не важливо, була вона вибрана з адресної книги чи вказана вручну при оформленні та дозволяє зберігати інформацію навіть у випадках, коли користувач пізніше видалить чи оновить свої адреси.

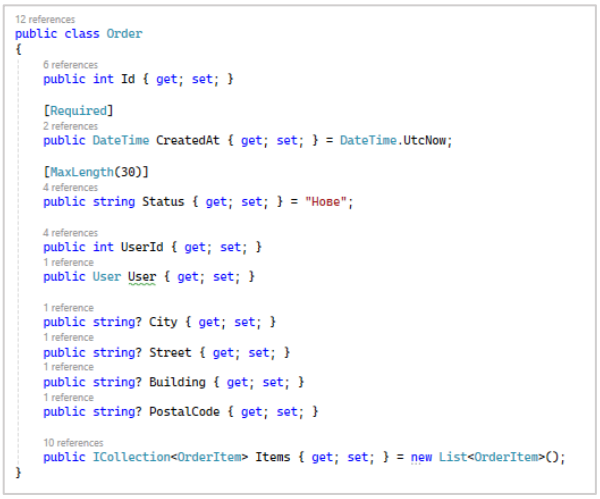


Рисунок 3.14 – Клас Order

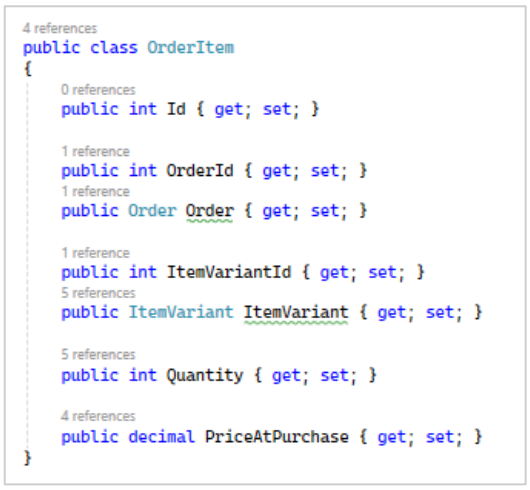


Рисунок 3.15 – Клас OrderItem

Зміст замовлення деталізується в окремій таблиці OrderItems. Кожен рядок тут представляє окрему позицію товару, яка входила до замовлення. Тут зберігаються посилання на конкретний варіант товару (ItemVariantId),

кількість одиниць та ціна на момент покупки (PriceAtPurchase). Ціна зберігається окремо, тому що вона може змінитися в каталозі з часом, а замовлення має залишатися історично коректним.

Зв’язок між таблицями реалізований як класичне відношення «один до багатьох»: одне замовлення включає кілька товарних позицій. Для користувачів це забезпечує повну історію покупок, а адміністратори отримують змогу переглядати склад кожного замовлення, його поточний статус, управляти обробкою та формувати необхідні звіти.

3.4 Реалізація логіки

3.4.1 Користувачі

Формування порад у цій програмі розпочинається з чіткого розуміння того, як виглядає постать користувача. З цією метою не передбачено ручного класифікування або умовних операторів у коді – система вчиться на конкретних прикладах. В процесі проєктування було передбачено використання механізмів машинного навчання, що дозволяє самостійно виявляти взаємозв’язки між введеними розмірами та відповідним типом фігури.

Навчання моделі реалізовано в межах служби BodyTypeService, де використовується бібліотека ML.NET. При першому запуску система перевіряє, чи є вже збережена модель у форматі .zip. Якщо ні, вона тренується автоматично. Для навчання застосовується набір даних у форматі CSV-файлу (body_data.csv), в якому містяться реальні або згенеровані приклади з конкретними параметрами: обхват грудей, талії, стегон, зріст, та номер типу фігури як мітка (Label) (фрагмент тренування наведено у лістингу 3.1).

Лістинг 3.1 – Навчання моделі класифікації

```
var pipeline = mlContext.Transforms.Conversion.MapValueToKey("Label",nameof(MeasurementInput.BodyType)).Append(mlContext.Transforms.Concatenate("Features", nameof(MeasurementInput.ChestWidth), nameof(MeasurementInput.Waist), nameof(MeasurementInput.Hips), nameof(MeasurementInput.TotalHeight))).Append(mlContext.Transforms.NormalizeMinMax("Features")).Append(mlContext.MulticlassClassification.Trainers.SdcaMaximumEntropy()).Append(mlContext.Transforms.Conversion.MapKeyToValue("PredictedLabel"));
```

У цьому фрагменті цільова змінна конвертується у внутрішній числовий формат, поля об’єднуються у вектор ознак, нормалізуються, подаються до класифікатора, а результат перетворюється назад у зрозумілий формат. Реалізація дозволяє моделі самостійно визначати типи фігури, ґрунтуючись на пропорціях тіла.

Навчена модель зберігається у файлі bodytype_model.zip. Всі подальші обчислення відбуваються на її основі, без потреби повторного тренування. Коли користувач вводить свої параметри, система створює об’єкт вхідних даних MeasurementInput, передає його до PredictionEngine, і отримує передбачений результат – номер відповідного типу фігури.

Отже, тип фігури не призначається довільно, а визначається попередньо навченою моделлю, що не тільки зменшує ймовірність помилок, а й надає можливість у майбутньому перенавчити систему на більшому обсязі даних, підвищуючи точність класифікації.

3.4.2 Рекомендаційна система

Визначивши тип фігури, наступним етапом логіки додатку стає формування персоналізованих рекомендацій. Це ключовий функціональний компонент, адже саме він створює для користувача відчуття

індивідуального підходу – не просто показує товари, а підбирає ті, що з більшою ймовірністю пасуватимуть конкретній фігурі.

Реалізація рекомендаційної системи передбачена у вигляді окремого сервісу – RecommendationService, який опрацьовує введені користувачем дані та співвідносить їх з наявним асортиментом товарів (методи наведені на рисунку 3.16). Логіка роботи така: для зареєстрованого користувача збережені мірки формують набір числових параметрів – обхвати грудей, талії, стегон. Потім система шукає варіанти одягу, розміри яких максимально наближені до цих значень.

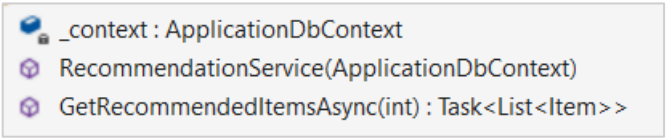


Рисунок 3.16 – Функції сервісу RecommendationService

При пошуку враховуються допустимі похибки, що дозволяють уникнути надмірної точності – наприклад, різниця у кілька сантиметрів вважається прийнятною. Відповідні варіанти одягу агрегуються, і на виході користувач бачить перелік товарів, які підходять йому не абстрактно, а за конкретними пропорціями тіла. Такий підбір стає можливим лише за наявності таблиці варіантів товарів, де зафіксовані не тільки розмірні мітки, а й їх відповідність у сантиметрах.

Рішення дозволяє відійти від стандартного підходу «обери розмір – отримай одяг» і перейти до більш гнучкої моделі, де сам сервіс визначає, які моделі можуть бути релевантними. У перспективі це створює фундамент для складнішої рекомендаційної системи, яка враховуватиме не тільки антропометричні параметри, а й стиль, вподобання, сезонність тощо.

В результаті користувач отримує не просто фільтрований каталог, а динамічно згенеровану вибірку, що враховує саме його дані, надаючи змогу

збільшити точність підбору речей, зменшити кількість невдалих покупок та зробити взаємодію з додатком максимально персоналізованою.

3.4.3 Завантаження зображень

Для наповнення каталогу товарів візуальним контентом система передбачає можливість завантаження зображень. Функція реалізована через окремий сервіс ImageUploaderService, який відповідає за збереження файлів, що надсилаються адміністратором, у відповідну директорію проєкту.

У разі надходження нового зображення, сервіс виконує перевірку на валідність, генерує унікальну назву для файлу, зберігає його у внутрішній папці /wwwroot/images/uploads, після чого повертає шлях, що використовуватиметься для відображення фото у інтерфейсі.

Отже, адміністратор має можливість швидко додати візуальну складову до товару, не вдаючись до додаткових технічних налаштувань. Модуль є ізольованим від решти логіки системи, що дозволяє у разі потреби замінити спосіб зберігання файлів (наприклад, на хмарний) без жодного впливу на основну систему.

3.5 Основні компоненти вебдодатку

3.5.1 Контролери

Контролери – це своєрідні мости між тим, що хоче користувач, і тим, як працює додаток. Вони забирають інформацію з форм, активують потрібні сервіси, відправляють результати в представлення та направляють увесь процес взаємодії. В організації додатку кожен контролер відповідає за окрему задачу, наприклад, роботу з кошиком, оформлення замовлення або керування асортиментом. Завдяки такому поділу дуже легко слідкувати за структурою маршрутів та уникати повторного написання коду.

Контролер AccountController відповідає за аутентифікацію, управління акаунтом та взаємодію з персональним профілем юзера. Він реалізує основні сценарії: реєстрацію, вхід, вихід та оновлення персональних даних.

Під час реєстрації відбувається створення нового акаунту, після чого система негайно виконує вхід – без потреби повторно вводити дані. Вхід в систему здійснюється через пошук за email та перевірку паролю. У випадку успішної аутентифікації, користувача ідентифікує набір claims, що містить ID, email та роль (фрагмент зазначений у лістингу 3.2).

Лістинг 3.2 – Ідентифікаційні claims користувача для сесії

```
new Claim(ClaimTypes.NameIdentifier, user.Id.ToString()),
new Claim(ClaimTypes.Email, user.Email),
new Claim(ClaimTypes.Role, user.Role ?? "User")
```

Профіль доступний для авторизованого користувача, який має змогу змінювати ім’я, прізвище або адресу електронної пошти, в якому також відображаються адреси доставки, збережені виміри та історія замовлень. Усі зміни передаються через POST запити та зберігаються у базі даних. Вихід із системи реалізовано через механізм cookie.

Контролер BodyTypeController відповідає за визначення типу статури користувача на основі введених антропометричних показників. Він також ініціює процес підбору одягу, що найкраще відповідає виявленому типу (рисунок 3.17).

GET-запит Identify() повертає порожню форму з полями для вказування замірів: об’єм грудей, об’єм талії, об’єм стегон та зріст. Після заповнення та надсилання цієї форми, POST-запит Identify(input) отримує введені дані, передає їх сервісу BodyTypeService та отримує передбачуваний тип фігури. Результат (ідентифікатор типу) зберігається у полі BodyTypeId

в моделі вимірювання. У випадку авторизованого користувача, значення також прив’язується до його профілю.

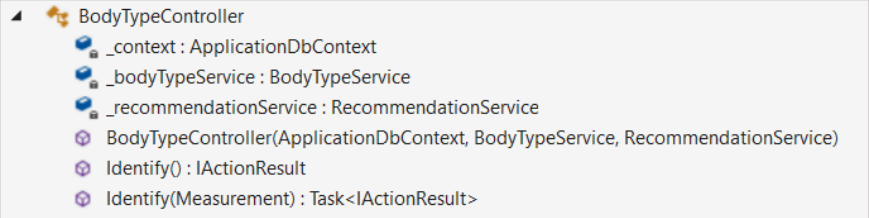


Рисунок 3.17 – Методи та функції BodyTypeController

Контролер CartController відповідає за функціональність кошика: перегляд товарів, які додано, управління ними та створення на основі вмісту повноцінного замовлення. Операції прив’язані до користувача, чий ідентифікатор отримується з сесії через claims.

Метод Index генерує сторінку для перегляду кошика. Для цього завантажується не тільки сам кошик, але й пов’язані об’єкти: варіанти товарів, відповідні товари, їхні зображення, що є чинником формування необхідної інформації для відображення (лістинг 3.3).

Лістинг 3.3 – Отримання інформації про вміст кошику

```
var cart = await _context.Carts
    .Include(c => c.Items)
    .ThenInclude(ci => ci.ItemVariant)
    .ThenInclude(v => v.Item)
    .ThenInclude(i => i.Images)
    .FirstOrDefaultAsync(c => c.UserId == userId);
```

Додавання товару (Add) відбувається через систему перевірок: якщо корзина ще не створена, вона негайно ініціалізується. Якщо товар вже наявний, його кількість збільшується на одиницю; в іншому випадку –

додається новий товар до корзини. Всі зміни автоматично записуються в базу даних без потреби оновлення сторінки.

Видалення (Remove) здійснюється просто: знаходить запис за його унікальним ID та видаляє його з таблиці. Оформлення (Checkout) передбачає створення нового замовлення на основі вмісту корзини. Позиції товарів переносяться в новий об’єкт Order, після чого корзина очищується. Для забезпечення зручності, передбачено автоматичне перенаправлення на сторінку з детальною інформацією про створене замовлення:

Контролер ClothesController відповідає за всю взаємодію з каталогом товарів. Тут зосереджено логіку, що керує пошуком, фільтрацією, сортуванням, посторінковою навігацією та відображенням детальної інформації про кожен екземпляр одягу.

Метод Index отримує низку параметрів: рядок для пошуку, списки розмірів, кольорів, типів фігур, цінові діапазони, критерії сортування та номер сторінки. Параметри формують запит до бази даних, який налаштовується відповідно до введених умов. Ключем є колекція товарів, до якої підключаються відповідні дані: зображення, варіанти, підкатегорії та категорії. Далі застосовується пошук. Якщо вказано текст, він фільтрується за кількома критеріями (лістинг 3.4).

Лістинг 3.4 – Побудова алгоритму пошуку товарів

```
if (!string.IsNullOrWhiteSpace(searchQuery)){
    var normalized = searchQuery.Trim().ToLower();
    query = query.Where(i =>
        i.Name.ToLower().Contains(normalized) ||
        i.Brand.ToLower().Contains(normalized) ||
        i.Material.ToLower().Contains(normalized));}
```

Далі застосовуються фільтри. Система опрацьовує тільки ті параметри, що були вказані: якщо, скажімо, обрано розміри S та M, то до

результатів потраплять лише вироби з відповідними позначками у варіантах. Аналогічно – з кольорами та цінами.

Сортування реалізовано через окремий функціональний блок: за ціною (збільшення/зменшення), за новизною або ж за замовчуванням. Результати поділяються на сторінки по 12 елементів на кожна, з розрахунком загальної кількості сторінок та поточного статусу пагінації.

Паралельно формуються додаткові дані для фільтрів: список унікальних розмірів, кольорів, типів фігур, мінімальна та максимальна ціна. Уся ця інформація передається у ViewBag, щоб динамічно генерувати панель вибору на інтерфейсі. Окремо реалізовано метод Details, який завантажує всю детальну інформацію про обраний товар: з описом, зображеннями, доступними варіантами розміру та кольору.

Контролер OrderController відповідає за управління переглядом та оновленням замовлень (рисунок 3.18). Він обслуговує два головні кейси: перегляд історії покупок користувача та детальне відображення кожного конкретного замовлення. Також передбачено функцію зміни статусу замовлення, призначену переважно для адміністраторів.

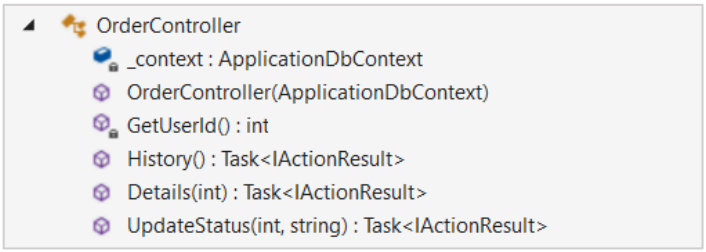


Рисунок 3.18 – Методи та функції OrderController

Метод History генерує перелік усіх замовлень, що належать активному користувачеві. Запит на замовлення включає завантаження супутніх даних: позиції замовлення, характеристики товарів та самі товари. Сформований перелік сортується за спаданням ID, тобто нові замовлення розміщуються на початку.

Details показує детальну інформацію про конкретне замовлення. Окрім загальних даних, цей метод також підвантажує зображення товарів, які входять до замовлення, що дозволяє користувачеві переглянути не лише назви та кількість, але й візуальну частину – які саме товари були придбані.

Метод UpdateStatus дає можливість оновити поточний стан замовлення. Зміна статусу зберігається в базі даних і застосовується до відповідного запису. Зважаючи на функціональність, цей метод орієнтований на адміністративний доступ, а не на взаємодію зі звичайними користувачами.

Контролер HomeController виконує роль точки входу у додаток. Він обслуговує головну сторінку, що показує декілька товарів з каталогу, відображення яких в подальшому можна формувати використовуючи алгоритми рейтингу. Для кожного товару підвантажуються зображення та доступні опції, а дані без змін передаються у вигляд.

Контролер AdminController реалізовує адміністраційну панель, де йде управління товарами, категоріями та замовленнями (рисунок 3.19). Головний метод Dashboard відображає стислу статистику системи: загальну кількість зареєстрованих користувачів, товарів і зроблених замовлень. За допомогою методу Items адміністратор отримує доступ до списку всіх товарів із завантаженими зображеннями, варіантами й прив’язками до підкатегорій, а також може редагувати їх через функцію EditItem.

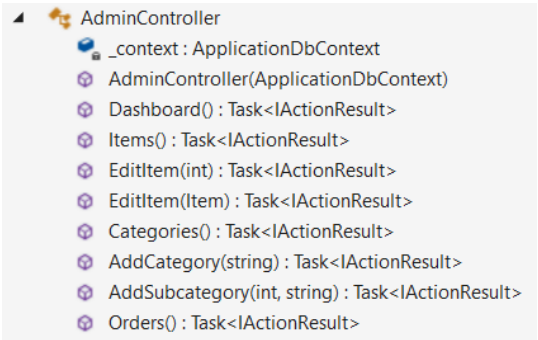


Рисунок 3.19 – Методи та функції AdminController

Управління категоріями спрощено: окремий метод виводить всі наявні категорії разом із підкатегоріями, а методи AddCategory та AddSubcategory дають можливість додавати нові. Перелік замовлень доступний через метод Orders, де кожне замовлення детально описує товари, варіанти та кількість, що забезпечує візуальний контроль над структурою та обсягом всіх транзакцій.

3.5.2 Представлення

Структура папки Views зорганізована відповідно до логіки контролерів. Кожен функціональний фрагмент програми (рисунок 3.20) має окрему піддиректорію, в якій містяться пов’язані з ним представлення. Підхід забезпечує порядок та прискорює навігацію в компонентах інтерфейсу.

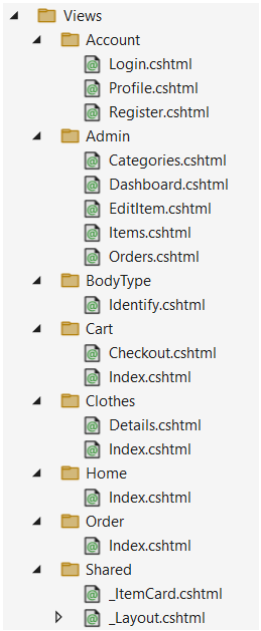


Рисунок 3.20 – Структура представлень додатку

Для взаємодії з обліковими даними передбачено окрему директорію. В ній розташовані представлення для авторизації, реєстрації та редагування

особистого профілю. Окремо виділено каталог, де міститься найбільш інтерактивна частина, де формується сторінка з товарами, реалізується фільтрація, сортування, пошук та пагінація. Сюди також входить перегляд конкретного товару з детальною інформацією.

Блок кошика та замовлень також мають власні директорії. В них містяться представлення, які дозволяють користувачеві переглядати список товарів у кошику та перехід до оформлення замовлення. Перегляд статусу та історії замовлень реалізований у профілі користувача.

Спільні елементи – макет блоку товару та шапка сайту, виділені в спеціальній папці. В свою чергу, вони підключаються до кожного представлення, забезпечуючи єдиний стиль та повторне використання компонентів без дублювання коду.

3.5.3 Статичні ресурси та маршрутизація

Всі статичні файли, які є в додатку: стилі, скрипти, зображення, розташовані у теці wwwroot. Структура цієї теки розділена на певні відділи: css – для стилів, js – для клієнтської логіки, images – для зберігання картинок товарів.

Сторінки стилізовані, використовуючи власний CSS в поєднанні з іншими бібліотеками. Застосовуються змінні для колірної схеми, адаптивна верстка, а також спрощена система класів для кнопок, форм та сітки каталогу. JavaScript застосовується для валідації полів, обробки введених значень, фільтрації та підвищення інтерактивності. Наприклад, у каталозі при взаємодії з фільтрами та формою для введення значення ціни (лістинг 3.5).

Маршрутизація базується на шаблоні, в якому шлях відповідає схемі controller/action/id, що дозволяє системі автоматично знаходити необхідне представлення, спираючись на назву контролера та дії. За замовчуванням,

головна сторінка відкриває Home/Index. Додаткові параметри передаються через запит або формуються як частина URL.

Лістинг 3.5 – Відсіювання невірно введених даних

```
if (value < fieldData.min || value > fieldData.max) {
  input.classList.add('invalid');
  errorText += `Поле "${fieldData.label}" має бути в межах ${fieldData.min}-${fieldData.max}.<br>`;}
```

Маршрути визначені на рівні Program.cs, де встановлюється точка входу у додаток. При цьому система самостійно визначає, до якого контролера необхідно звернутись, виходячи з адреси сторінки.

4 АНАЛІЗ РОЗРОБЛЕНОГО ВЕБДОДАТКУ

4.1 Опис функціональних можливостей та інтерфейсу

Розроблений вебдодаток призначений для продажу жіночого одягу, з функцією підбору товарів, враховуючи тип фігури. Основні можливості: реєстрація, управління профілем, пошук та фільтрація товарів, оформлення замовлень і персоналізовані рекомендації. Функціонал реалізовано за моделлю взаємодії клієнт-сервер з чітким зв’язком між представленнями та контролерами. Інтерфейс додатку базується на Razor Views, де сторінки розділені за функціональними зонами. Сторінки адаптовано до різних типів пристроїв, що забезпечує зручність роботи як на комп’ютерах, так і на мобільних телефонах. Структура сайту спроєктована таким чином, щоб користувач швидко знаходив потрібні розділи, мінімізуючи кількість переходів.

Головна сторінка вебдодатку – це точка входу для користувачів, що формує перше враження про сервіс (рисунок А.1). На всіх сторінках вгорі розміщене навігаційне меню, яке дає можливість швидко переходити між розділами сайту: каталогу товарів, модуля підбору за типом фігури, новинок та розділу зі знижками. Праворуч в меню розміщені функціональні іконки – пошук, профіль користувача, обране та кошик, які надають змогу отримати доступ до зазначених елементів взаємодії.

Центральну частину головної сторінки займає банер, мета якого є акцентувати увагу на актуальній пропозиції або новій колекції товарів. Нижче банера розташована добірка популярних товарів, сформована на основі критеріїв популярності та новизни. Кожен товар на всіх сторінках, де він наявний у виді попереднього перегляду, представлено у вигляді окремої картки із зображенням, назвою та ціною, а також двома кнопками – для додавання до списку обраного та для миттєвого переміщення в кошик, що дозволяє користувачеві оперативно здійснювати покупки без потреби

переходити на сторінку товару. Внизу сторінки знаходиться футер, що забезпечує швидкий доступ до інформаційних ресурсів.

Сторінка визначення типу фігури є одним з найважливіших функціональних елементів вебдодатку (рисунок А.2). Під головним меню розміщений ланцюжок навігації, який допомагає користувачеві швидко зорієнтуватися на сайті, розуміючи, де він знаходиться. У центральній частині розташована форма для введення антропометричних даних: зросту, обхвату грудей, талії та стегон. Біля кожного параметра розміщено текстове роз’яснення, яке пояснює, як правильно знімати мірки, щоб зменшити вірогідність введення неправильних даних. Після заповнення полів користувач натискає кнопку для визначення типу фігури. Якщо введені дані не відповідають вимогам, система сповістить про помилки, коли зріст або обхвати не відповідають допустимим значенням. Повідомлення підсвічуються, щоб було чітко видно, які саме поля містять помилки, що забезпечує зворотний зв’язок та зводить нанівець проходження хибних даних (рисунок А.3).

Після успішного введення даних, нижче з’являється блок з результатом – визначеним типом фігури та коротким описом. Блок містить опис особливостей фігури, поради щодо вибору одягу та інформацію про те, яких фасонів слід уникати. Поруч з текстовою інформацією розміщена ілюстрація, що наглядно демонструє виявлений тип фігури. Під рекомендаціями система автоматично генерує підбірку товарів, що найкраще підходять для конкретного типу фігури. Товари представлені у вигляді карток з фотографіями, цінами та кнопками для додавання в кошик або обране. В свою чергу, це дозволяє користувачеві відразу перейти до вибору одягу без необхідності повертатися до каталогу, що позитивно впливає на зручність використання. Внизу сторінки розташована пагінація для зручного переходу між сторінками з рекомендаціями.

Каталог – це функціонально насичений розділ, структура якого розроблена таким чином, щоб максимально спростити користувачеві пошук

та вибір товарів (рисунок А.4). Ліворуч розташована панель фільтрації, що дає змогу вибрати товари за категоріями, розмірами, кольором, матеріалом, брендом та ціною. Блок фільтрів реалізований у вигляді розкривних списків і чекбоксів, що дозволяє гнучко поєднувати умови пошуку. Особливістю є підтримка вибору декількох фільтрів одночасно, а також можливість фільтрації за типом фігури, що інтегрує функціонал персоналізованих рекомендацій безпосередньо в каталог.

Центральна частина сторінки зайнята сіткою товарів. Кожен товар відображається у вигляді картки з зображенням, назвою, ціною та іконками для додавання у список обраного або кошик. Зверху над сіткою розташований елемент для сортування товарів за ціною, новизною чи популярністю. Вибір сортування автоматично оновлює вивід товарів без повного перезавантаження сторінки, що позитивно впливає на швидкість взаємодії.

Каталог підтримує пагінацію, що дозволяє користувачам переходити між сторінками результатів пошуку. Кнопки сторінок розташовані під сіткою товарів, також є окрема кнопка для завантаження додаткових товарів без необхідності переходу. У разі активних фільтрів і сортування, всі параметри зберігаються при переході на нову сторінку, що забезпечує сталість пошукового запиту (рисунок А.5).

Сторінка товару сконцентрована на представленні інформації про вибрану позицію з каталогу (рисунок А.6). З лівого боку розміщено блок з зображеннями товару, що дозволяє перемикатися між фото. З правого боку розташовані основні відомості: назва товару, його ціна, доступні кольори та розміри. Вибір кольору реалізовано як мініатюрні індикатори (завдяки переведенню спеціальному сервісу переведення словесного значення в кольорове), а вибір розміру – у форматі кнопок з розміткою стандартних розмірних сіток. Поряд є кнопка для додавання товару до кошика.

Нижче розташовані інформаційні блоки щодо умов оплати, гарантії та доставки. Основна текстова частина сторінки містить опис товару, де

наведені ключові характеристики моделі та її переваги. Окремим блоком виділено інформацію про склад матеріалів та рекомендації щодо догляду. В нижній частині сторінки виводиться список аналогічних товарів – система пропонує альтернативи на основі категорії або характеристик. Кожна картка аналогічного товару містить стандартний набір елементів: фото, назву, ціну та кнопки для взаємодії. Завершує сторінку блок відгуків. Якщо відгуків поки немає, користувачеві пропонується залишити перший коментар за допомогою відповідної кнопки.

Сторінка з кошиком відображає перелік товарів, які користувач обрав для замовлення (рисунок А.7). Кожна позиція реалізована як окремий блок, що містить зображення продукту, його назву, обраний відтінок, розмір та вартість за одиницю. Поруч знаходяться інструменти для регулювання кількості товару та кнопка видалення позиції з кошика.

Загальна вартість замовлення обчислюється автоматично на основі кількості одиниць кожного товару та їхньої ціни. Блок із загальною сумою розташований внизу списку товарів та динамічно оновлюється у відповідь на зміни кількості або видалення позицій. Під списком товарів знаходиться кнопка переходу до оформлення замовлення, що виводить користувача на заключний етап покупки.

Окрім основної сторінки кошика, передбачено спливаюче вікно-кошик, яке з’являється при наведенні на відповідну іконку у верхній частині вебсайту (рисунок А.8). У цьому вікні повторюється вміст кошика: список товарів з короткою інформацією, загальна сума та кнопка переходу до оформлення. Спливаюча версія дозволяє користувачеві швидко переглядати вміст кошика, не покидаючи поточну сторінку.

Сторінка оформлення замовлення структурована на два розділи: персональна інформація та деталі доставки (рисунок А.9). Перша секція передбачає заповнення особистих даних покупця, таких як прізвище, ім’я, по батькові, контактний телефон та електронна адреса. У другому блоці потрібно визначити метод доставки за допомогою випадаючого меню,

вказати населений пункт, назву вулиці, номер будівлі та поштовий індекс. Окремо, під блоком доставки, знаходиться поле для вибору способу оплати. Перед відправленням даних на сервер, усі поля проходять перевірку на наявність введеної інформації.

Під формою відображається підсумкова вартість замовлення, що автоматично змінюється відповідно до складу кошика. Для підтвердження покупки передбачено кнопку «Замовити», натискання якої ініціює відправку всієї інформації на сервер, а при виборі оплати онлайн, користувач переадресовується на сторінку платіжного сервісу LiqPay, щоби виконати транзакцію.

Блок акаунту користувача охоплює реєстрацію, вхід і керування особистим кабінетом. Реєстрація реалізована через стандартну форму, де заповнюються основні дані: прізвище, ім’я, по батькові, електронна пошта, пароль і підтвердження пароля (рисунок А.10). Усі поля є обов’язковими для заповнення, із базовою валідацією на коректність введених даних. Після успішної реєстрації користувач отримує доступ до особистого кабінету. Процедура входу спрощена – користувач вводить лише електронну пошту та пароль (рисунок А.11).

Усередині особистого кабінету реалізовано декілька функціональних розділів. Перший блок – контактна інформація, де можна переглянути та змінити персональні дані: прізвище, ім’я, по батькові, номер телефону та електронну пошту (рисунок А.12). Всі дані доступні для редагування через окрему форму з кнопкою збереження змін.

Другий розділ – адресна книга. Тут користувач бачить збережені адреси доставки у вигляді списку (рисунок А.13). Додати нову адресу можна через окрему форму, де потрібно заповнити місто, вулицю, будинок і поштовий індекс, що дозволяє мати декілька адрес і вибирати потрібну під час оформлення замовлення.

Третій розділ присвячено історії замовлень (рисунок А.14). У ньому відображається список усіх оформлених замовлень із зазначенням номера,

дати створення та статусу доставки. Також користувач бачить підсумкову суму кожного замовлення. Деталізація замовлення доступна через розгортання елемента списку.

Ще один важливий функціонал – список бажань, де відображаються всі товари, які користувач додав до обраного під час перегляду каталогу (рисунок А.15). Кожна позиція виводиться у вигляді стандартної картки товару з фото, назвою, ціною та іконками для швидкого додавання в кошик або видалення зі списку.

Система також передбачає можливість видалення акаунту (рисунок А.16). При переході до цієї функції користувачу пропонується вказати причину видалення профілю та підтвердити дію через відповідну галочку. Лише після підтвердження активується кнопка остаточного видалення, що запобігає випадковому видаленню акаунту і вимагає від користувача усвідомленого підтвердження своїх дій.

Адміністративна панель інтегрована в особистий кабінет користувача з розширеними правами (рисунок А.17). Вона надає доступ до ключових операцій управління контентом сайту: додавання нових товарів, корегування існуючих позицій, перегляд здійснених замовлень та управління структурою категорій. Після успішної авторизації адміністратора, на панелі з’являється відповідний пункт «Адмін. панель», який відкриває меню з доступними функціями.

Функція додавання товару реалізована у формі з двох етапів (рисунок А.18). На першому етапі адміністратор заповнює загальну інформацію про товар: назву, належність до категорії та підкатегорії, опис продукту, склад використаних матеріалів, інструкцію з догляду та вартість. Усі поля, що вимагають заповнення, містять необхідні списки для зручного вибору. На другому етапі адміністратор додає інформацію про варіації товару: колір, розмір та доступну кількість. Також передбачена можливість завантаження зображень для кожної варіації за допомогою стандартного механізму вибору файлів.

Редагування існуючих товарів побудоване за аналогічною логікою (рисунок А.19). Перша частина форми дозволяє вносити зміни в текстову інформацію про товар та коригувати його ціну. Друга частина надає можливість редагувати наявність товарів за кольорами та розмірами, а також замінювати прикріплені фотографії. Інтерфейс форми характеризується гнучкістю та дозволяє оперативно змінювати як текстові дані, так і візуальні аспекти, уникнувши потреби створювати товар заново.

Управління замовленнями забезпечує огляд всіх покупок, зроблених користувачами. Адміністратор отримує доступ до основної інформації про замовлення: номер замовлення, дату його оформлення, поточний статус доставки, а також загальну суму замовлення. Це дає змогу швидко контролювати процес виконання замовлень та змінювати їх статуси за необхідності.

Розділ управління категоріями дозволяє адміністратору додавати нові або змінювати існуючі категорії та підкатегорії товарів. Це важливо для підтримання актуальної структури каталогу при оновленні асортименту.

4.2 Оцінка якості реалізації вебдодатку

Розроблений вебдодаток цілком відповідає поставленим вимогам, як за функціональною наповненістю, так і за якістю реалізації основних сценаріїв взаємодії з користувачем. Перевірка функціоналу засвідчила коректне виконання усіх ключових процесів: реєстрації, авторизації, перегляду товарів, навігації каталогом і підбору товарів за типом фігури, без будь-яких суттєвих затримок при завантаженні сторінок.

Процедура реєстрації користувача функціонує надійно: всі необхідні поля підлягають перевірці на заповнення та відповідність даних. Механізм входу у систему не викликає труднощів – після успішної авторизації користувач отримує доступ до особистого кабінету. Функціонал кабінету реалізовано логічно: редагування персональної інформації, адресна книга,

історія замовлень і список бажань функціонують без збоїв. Усі форми збереження даних передбачають базову валідацію, що унеможливлює введення некоректних даних.

Оцінка сторінки визначення типу фігури підтвердила правильність обробки антропометричних даних. Система коректно реагує на недійсні значення, підсвічуючи помилки. При введенні коректної інформації алгоритм класифікації працює стабільно, а результат із поясненням та рекомендаціями генерується миттєво. Додатково система пропонує персоналізовану підбірку товарів, що підвищує рівень індивідуалізації взаємодії з користувачем.

Каталог товарів підтримує багаторівневу фільтрацію за кількома критеріями одночасно. Фільтри пошуку працюють правильно, а параметри сортування оновлюють результати без перезавантаження сторінки. Всі вибрані користувачем фільтри зберігаються при перемиканні між сторінками результатів, що спрощує пошук і забезпечує стабільність сеансу перегляду товарів. Пагінація та можливість поступового завантаження нових товарів функціонують коректно.

Сторінка товару повною мірою демонструє всю необхідну інформацію: зображення, варіації кольорів та розмірів, умови доставки й оплати. Функціонал додавання до кошика працює оперативно, обрані варіанти товару зберігаються правильно. Блок аналогічних товарів та форма для відгуків доступні й функціонують належним чином.

Кошик реалізовано стабільно, всі додані товари відображаються з коректною інформацією. Можливість редагування кількості та видалення товарів функціонує швидко, а загальна сума замовлення оновлюється автоматично без перезавантаження сторінки. Спливаюче вікно-кошик також функціонує правильно, даючи змогу переглядати замовлення без втрати поточного контенту сторінки.

Процедура оформлення замовлення перевірена на повноту заповнення всіх форм. Всі поля проходять перевірку на обов’язковість заповнення, а

також обробляються належним чином на сервері. Сума замовлення розраховується коректно, вибір способу доставки та оплати доступний і працює відповідно до очікувань.

Особистий кабінет користувача охоплює весь заявлений функціонал: зміну контактної інформації, управління адресами, перегляд історії замовлень та роботу зі списком бажань. Видалення облікового запису реалізовано з обов’язковим підтвердженням, що зменшує ризик випадкової втрати даних користувача.

Адміністративна панель також розроблена відповідно до вимог. Операції додавання, редагування товарів, оновлення інформації про кількість і варіації функціонують стабільно. Завантаження зображень відбувається без помилок. Перегляд і управління замовленнями дозволяє оперативно відстежувати статуси виконання. Редагування категорій каталогу працює коректно, забезпечуючи гнучкість у підтримці актуального асортименту.

Тестування інтерфейсу на різних пристроях показало належний рівень адаптивності: сторінки коректно відображаються на екранах різних розмірів, елементи інтерфейсу не виходять за межі екрану, навігація зберігає стабільність. Всі клієнтські сценарії завершуються успішно без помилок або зависань. Вебдодаток демонструє швидкий час відгуку при основних діях користувачів, що свідчить про оптимізацію як серверної, так і клієнтської частин.

На підставі результатів оцінювання можна констатувати, що розроблений вебдодаток повністю відповідає функціональним та нефункціональним вимогам, забезпечуючи стабільну роботу, зручність використання та належний рівень персоналізації для кожного користувача.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було успішно реалізовано вебдодаток для продажу жіночого одягу з функцією визначення типу фігури та надання персоналізованих рекомендацій. Поставлену мету досягнуто шляхом поєднання сучасних технологій веброзробки та методів персоналізації користувацького досвіду.

У ході роботи проведено детальний аналіз предметної галузі, що дозволило виявити основні проблеми онлайн-продажу жіночого одягу, серед яких найбільш суттєвою є відсутність можливості примірки та складність у виборі одягу, який відповідає індивідуальним особливостям фігури. Було вивчено існуючі рішення, проаналізовано їх переваги й недоліки, що дало змогу сформулювати вимоги до створення продукту.

Розроблений вебдодаток базується на використанні ASP.NET Core MVC для серверної частини, Razor Pages для побудови інтерфейсу та PostgreSQL для зберігання даних. Було створено інтелектуальний модуль для визначення типу фігури на основі введених користувачкою параметрів для формування персоналізовані рекомендації з підбору одягу. У системі реалізовано функціонал фільтрації товарів за різними характеристиками, адаптивний інтерфейс та можливість управління замовленнями.

Виконане тестування підтвердило стабільність роботи додатку, зручність інтерфейсу, коректність реалізації основних функцій та високу якість персоналізованого підбору товарів. Результати роботи свідчать про доцільність використання інтеграції індивідуальних рекомендацій у сфері електронної комерції для підвищення задоволеності користувачів та зменшення кількості повернень. Таким чином, розроблений вебдодаток не лише відповідає актуальним вимогам сучасного ринку, маючи потенціал для розвитку, а саме за рахунок розширення асортименту, удосконалення алгоритмів рекомендацій та створення мобільної версії для покращення доступності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bolaji J. T. Impact of the female body shape on clothing size and fit: comfort versus safety. *Research journal of textile and apparel*. 2025. URL: https://doi.org/10.1108/rjta-06-2024-0095 (дата звернення: 14.05.2025).

2. Perception of body appearance and its relation to clothing. *Clothing appearance and fit*. 2004. URL: https://doi.org/10.1201/9781439823446.ch1 (дата звернення: 14.05.2025).

3. Zara.com. Офіційний сайт. *Zara*. URL: https://www.zara.com/ (дата звернення: 14.05.2025).

4. Гід по типам фігури. *The Concept Wardrobe*. URL: https://theconceptwardrobe.com (дата звернення: 14.05.2025).

5. Підбір одягу за типом фігури. *Gemini Woman*. URL: https://www.gemini-woman.co.uk/ (дата звернення: 14.05.2025).

6. ASP.NET documentation. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/ukua/aspnet/core/?view=aspnetcore-9.0 (дата звернення: 14.05.2025).

7. Overview of ASP.NET Core MVC. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/ukua/aspnet/core/mvc/overview?view=aspnetcore-9.0 (дата звернення: 14.05.2025).

8. Overview of Entity Framework Core – EF Core. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/ukua/ef/core/ (дата звернення: 14.05.2025).

9. PostgreSQL: Documentation. *PostgreSQL: The world's most advanced open-source database*. URL: https://www.postgresql.org/docs/ (дата звернення: 14.05.2025).

10. Introduction. *Bootstrap · The most popular HTML, CSS, and JS library in the world*. URL: https://getbootstrap.com/docs/5.0/gettingstarted/introduction/ (дата звернення: 14.05.2025).

11. ASP.NET Core MVC Tutorial for Beginners | .NET Core Tutorial. *Pragim Tech*. URL: https://www.pragimtech.com/courses/asp-net-core-mvc-tutorial-for-beginners/ (дата звернення: 23.05.2025).

12. Flowmapp. Personalization UI Design Trends in 2023: 7 Key Methods. *Medium*. URL: https://medium.com/@Flowmapp/personalization-ui-design-trends-in-2023-7-key-methods-3216f94f0705 (дата звернення: 23.05.2025).

13. Kasym M. How Much Personalization Is Enough in UX Design?. *UX Magazine*. URL: https://uxmag.com/articles/how-much-personalization-is-enough-in-ux-design (дата звернення: 24.05.2025).

14. UI Personalization – The Key to Effective Interface Design. Fuselab Creative | *User Experience Design*. URL: https://fuselabcreative.com/personalization-the-key-to-effective-interface-design/ (дата звернення: 24.05.2025).

15. Schade A. Customization vs. Personalization in the User Experience. *Nielsen Norman Group*. URL: https://www.nngroup.com/articles/customization-personalization/ (дата звернення: 25.05.2025).

16. Business V. Global Fashion Agenda and PDS announce Trailblazer Award finalists. *Vogue Business*. URL: https://www.voguebusiness.com/story/sustainability/global-fashion-agenda-and-pds-announce-trailblazer-award-finalists (дата звернення: 03.06.2025).

17. Team B. M. Sustainable Fashion Trends to Watch Out For in 2025. Browzwear | *#1 Digital Apparel Design & Development Software*. URL: https://browzwear.com/blog/sustainable-fashion-trends (дата звернення: 03.06.2025).