



Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»

Кафедра інформаційних технологій та фундаментальної підготовки

Випускна кваліфікаційна робота

Інформаційна система локалізованої е-комерції (на прикладі Jysk)

на здобуття бакалаврського рівня вищої освіти

зі спеціальності «122 Комп'ютерні науки»

Виконавці проєкту:

№	П.І.Б.	Група	Роль
1	Григор'єва Анна Павлівна	КН-П-212	Full stack

Керівник	Викладач кафедри ІТ та ФП, Туманов О.А
----------	---

АНОТАЦІЯ

Робота присвячена розробці Backend частини інтернет-магазину на прикладі JYSK.ua. Робота охоплює етапи аналізу, проектування, розробки та тестування інтернет-платформи для торгівлі товарами для дому — меблями, товарами для сну, інтер'єру та саду.

Інтернет-магазин JYSK надає користувачам можливість зручно та швидко купувати необхідні товари для облаштування дому, створюючи комфортний простір як у житлі, так і на присадибній ділянці. Платформа пропонує широкий асортимент продукції, зокрема: ліжка, матраци, шафи, стільці, текстиль, освітлення, декор та садові меблі.

Функціонал платформи включає: реєстрацію, автентифікацію, двофакторну автентифікацію, авторизацію, розширений пошук, фільтрацію за категоріями, розмірами, цінами та брендами, сортування товарів, систему відгуків, рейтингів та можливість поставити запитання щодо конкретного товару. Окрему увагу приділено зручному кошику, системі обліку наявності товарів, інтеграції з логістичними службами та підтримці декількох мов інтерфейсу.

Проект є актуальним з огляду на стрімкий розвиток електронної комерції в сфері товарів для дому. Все більше користувачів надають перевагу покупкам в інтернеті з можливістю попереднього ознайомлення з характеристиками, відгуками та фотографіями товарів.

Особливу увагу в роботі приділено реалізації системи безпеки, зокрема захисту персональних даних користувачів, запобіганню несанкціонованому доступу, шифруванню даних та забезпеченню безпечної автентифікації.

У результаті буде створено функціональний, надійний та масштабований інтернет-магазин, що відповідає потребам сучасного споживача. Проект передбачає можливість подальшої розробки, розширення функціоналу, додавання нових модулів та інтеграцій у відповідь на зміну ринкових умов та очікувань клієнтів.

Для реалізації Backend частини використано мову програмування C#, фреймворк ASP.NET Core 6, реляційну базу даних PostgreSQL для зберігання основних даних, а також Redis як нереляційне сховище для кешування та підвищення продуктивності системи.

ЗМІСТ

ВСТУП	3
Технічне завдання до проєкту	4
РОЗДІЛ 1. Загальна характеристика системи	6
РОЗДІЛ 2. Реалізація серверної частини	7
ВИСНОВКИ	10
ПЕРЕЛІК ВИКОРИСТАНИХ ТЕХНОЛОГІЙ	11
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	12

ВСТУП

Інтернет-технології відіграють ключову роль у трансформації сучасного ринку роздрібної торгівлі, відкриваючи нові можливості як для споживачів, так і для бізнесу. Зростаюча популярність онлайн-покупок та очікування користувачів щодо зручності, швидкості й надійності стимулюють розвиток цифрових рішень, які забезпечують ефективну взаємодію між продавцем і покупцем.

У цьому контексті онлайн-магазини нового покоління виступають важливим інструментом, що дозволяє реалізовувати широкий асортимент продукції, оптимізувати процеси обслуговування клієнтів і досягати ширшої аудиторії. Поєднання зрозумілого інтерфейсу, продуманої навігації, високої якості сервісу та технічної надійності — усе це формує основу успішної онлайн-торгівлі.

Даний документ присвячено дослідженню, розробці та впровадженню проєкту онлайн-магазину, який відповідає сучасним вимогам цифрового ринку. Основною метою є створення функціональної платформи, що забезпечує постійний доступ до товарів, інтуїтивно зрозумілу структуру та високий рівень зручності для користувачів.

У вступній частині аналізуються технічні й функціональні вимоги до системи. Далі розглядається процес побудови архітектури онлайн-магазину з акцентом на безпеку, масштабованість і продуктивність. Також детально описуються етапи реалізації, зокрема вибір технологічного стеку, розробка серверної частини, інтеграція з базами даних та системами управління контентом.

Цей автореферат дає уявлення про ключові аспекти створення сучасної електронної торгової платформи, яка здатна відповідати динамічним викликам ринку та потребам користувачів.

У підсумках наводяться результати тестування та досвід використання розробленого рішення. Отримані дані свідчать про ефективність обраного підходу до проєктування онлайн-магазину, що дозволяє підвищити рівень клієнтського досвіду та забезпечити стійку роботу бізнесу в умовах цифрової економіки.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЄКТУ

Загальний опис

Розробка цифрової системи для управління онлайн-торгівлею в сфері роздрібного продажу товарів для дому та саду

Призначення

Оперативне адміністрування веб-контенту онлайн-магазину, включаючи оновлення інформації про товари, акції, наявність і ціни. Надійне зберігання даних користувачів і замовлень із дотриманням правил безпеки та обмеженням доступу відповідно до прав користувачів.

Вимоги до функціоналу:

- Інтернет крамниця
 - Навігація по веб вітрині крамниці
 - Пошук товарів за критеріями
 - Оформлення замовлень
 - Зворотній зв'язок та відгуки
- Взаємодія із інтернет додатками
 - Web-API моделі та запити для вибору товару в кошик та оформлення замовлення користувачем
 - Web-API моделі та запити для створення питань та відгуків, а також для їх оцінок (лайків/дізлайків)
 - Web-API моделі та запити для пошуку товарів з необхідними фільтрацією та сортуванням
 - Web-API моделі та запити для отримання категорій товарів, банерів, тощо

Архітектура:

- Сервер бази даних - серверне програмне забезпечення, що забезпечує збереження та обробку поточних даних.
- Сервіс Web-API – серверне програмне забезпечення, що надає програмний інтерфейс для взаємодії (REST-API) з іншим програмним забезпеченням.
- Веб додаток керування контентом - доступний з веб-простору сайт, що забезпечує візуальний інтерфейс для менеджерів системи.
- Веб крамниця - доступний з веб-простору сайт, що дозволяє взаємодіяти з інтернет-магазином покупцям.

Технології:

- Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології та методології створення:
 - Бази даних – MSSQL
 - Сервіс Web-API – REST-API, ASP.NET Core MVC, C#
 - Веб додаток керування контентом – MVC, ASP.NET Core, C#
 - Веб додаток для покупця – React, TypeScript
- При розгортанні програмного забезпечення рекомендується задіяти хмарні технології
- При розробці програмного забезпечення необхідно задіяти зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контролю версій (вибір конкретної системи узгоджується групою проекту).
- Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU <http://www.gnu.org/prep/standards/>, у тому числі з дотриманням вимог до коментування та документування коду.

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

В результаті аналізу технічного завдання до проєкту, архітектурним стилем додатка став SPA (Single Page Application) завдяки своїй продуктивності, сучасності та зручності для користувачей. Проєкт складається з таких частин:

- Backend
- Frontend

Серверна частина Backend розроблена мовою програмування C#, за допомогою фреймворку ASP.NET Core 8. Для надійного зберігання персональних було використано СУБД MSSQL. Вибір саме цієї системи управління базами даних був зумовлений її ефективністю та популярністю.

Також ця СУБД підтримується популярним ORM (Object Relational Mapping) Entity Framework Core від Microsoft, який спрощує розробку та взаємодію з базою даних за допомогою мови запитів LINQ.

Сервер реалізує RESTful API, який обробляє мережеві запити від клієнтської частини за допомогою протоколу HTTP. Він проводить маніпуляції з даними, за потреби звертається до бази даних або до кешу і надсилає відповідь до клієнтської частини у форматі JSON.

Код серверної частини знаходиться за адресою:
<https://github.com/Dev-Anna-Gri/hyggy-backend>

Код клієнтської частини знаходиться за адресою:
<https://github.com/Dev-Anna-Gri/hyggy-frontend>

РОЗДІЛ 2

РЕАЛІЗАЦІЯ СЕРВЕРНОЇ ЧАСТИНИ

Серверна частина написана з використанням наступних технологій:

- Мова програмування C#
- Фреймворк ASP.NET Core 8
- База даних MSSQL
- Інструмент адміністрування бази даних SSMS
- Середовище розробки Visual Studio
- Технології: Dapper, LINQ
- Додаткові пакети: System.Data.SqlClient, dotnet.env, Microsoft.AspNetCore.Mvc, Newtonsoft.Json, Swashbuckle.AspNetCore

В основі серверної частини лежить RESTful API, який реалізовано згідно з Чистою Архітектурою. При написанні проєкту використовувались принципи проєктування, такі як SOLID, DRY, та патерни проєктування, зокрема MVC, OOP, CQRS. Це все необхідно для того, щоб мінімізувати наростання складності розробки додатка з часом.

Усі користувацькі та системні дані, необхідні для роботи додатка, зберігаються у базі даних MSSQL. Для маніпулювання та мапінгу даних с бази до класів в коді використовується Sql Client.

Код системи є асинхронним там, де це необхідно. Асинхронність необхідна у I/O-bound (Input/Output-bound) операціях. Прикладом такої операції є запит до бази даних.

Пошук товарів з урахуванням фільтрації та сортування доволі складний механізм. Але код, який це реалізує, вийшов доволі гнучким. Та якщо потрібно

буде додати параметр до сортування, то потрібно буде додати лише один рядок кода.

Загалом, серверна частина має 60+ кінцеві точки. Усі вони необхідні для функціонування системи. Їх перелік можна побачити тут:

Assets		^
GET	/api/Assets	▼
POST	/api/Assets	▼
GET	/api/Assets/{id}	▼
PUT	/api/Assets/{id}	▼
DELETE	/api/Assets/{id}	▼
BlogCategories		^
GET	/api/BlogCategories	▼
POST	/api/BlogCategories	▼
GET	/api/BlogCategories/{id}	▼
PUT	/api/BlogCategories/{id}	▼
DELETE	/api/BlogCategories/{id}	▼
Blogs		^
GET	/api/Blogs	▼
POST	/api/Blogs	▼
GET	/api/Blogs/{id}	▼
PUT	/api/Blogs/{id}	▼
DELETE	/api/Blogs/{id}	▼

BlogSubcategories		^
GET	/api/BlogSubcategories	▼
POST	/api/BlogSubcategories	▼
GET	/api/BlogSubcategories/{id}	▼
PUT	/api/BlogSubcategories/{id}	▼
DELETE	/api/BlogSubcategories/{id}	▼
Brands		^
GET	/api/Brands	▼
POST	/api/Brands	▼
GET	/api/Brands/{id}	▼
PUT	/api/Brands/{id}	▼
DELETE	/api/Brands/{id}	▼
ProductCategories		^
GET	/api/ProductCategories	▼
POST	/api/ProductCategories	▼
GET	/api/ProductCategories/{id}	▼
PUT	/api/ProductCategories/{id}	▼
DELETE	/api/ProductCategories/{id}	▼

Products

GET	/api/Products	▼
POST	/api/Products	▼
GET	/api/Products/{id}	▼
PUT	/api/Products/{id}	▼
DELETE	/api/Products/{id}	▼

ProductSpecifications

GET	/api/ProductSpecifications	▼
POST	/api/ProductSpecifications	▼
GET	/api/ProductSpecifications/{id}	▼
PUT	/api/ProductSpecifications/{id}	▼
DELETE	/api/ProductSpecifications/{id}	▼

ProductSubcategories

GET	/api/ProductSubcategories	▼
POST	/api/ProductSubcategories	▼
GET	/api/ProductSubcategories/{id}	▼
PUT	/api/ProductSubcategories/{id}	▼
DELETE	/api/ProductSubcategories/{id}	▼

Reviews

GET	/api/Reviews	▼
POST	/api/Reviews	▼
GET	/api/Reviews/{id}	▼
PUT	/api/Reviews/{id}	▼
DELETE	/api/Reviews/{id}	▼

Stores

GET	/api/Stores	▼
POST	/api/Stores	▼
GET	/api/Stores/{id}	▼
PUT	/api/Stores/{id}	▼
DELETE	/api/Stores/{id}	▼

Warehouse

GET	/api/Warehouse	▼
POST	/api/Warehouse	▼
GET	/api/Warehouse/{id}	▼
PUT	/api/Warehouse/{id}	▼
DELETE	/api/Warehouse/{id}	▼

ВИСНОВКИ

Було реалізовано веб-сайт YUSK — сучасний онлайн-магазин товарів для дому та саду. Проєкт складається з кількох компонентів, що взаємодіють між собою: клієнтська частина (FrontEnd), серверна логіка (BackEnd) та база даних. Для реалізації FrontEnd було використано React у зв'язці з TypeScript, що дозволило побудувати динамічний, адаптивний та зручний інтерфейс для користувачів.

BackEnd частина реалізована за допомогою ASP.NET Core з використанням патерну MVC та API-архітектури, що забезпечує гнучкість і масштабованість у розробці. Взаємодія з базою даних здійснюється через MS SQL Server — надійну СУБД, яка відповідає за зберігання інформації про користувачів, товари, замовлення та інші сутності системи.

Система також має високий рівень безпеки — реалізована автентифікація та авторизація з можливістю розширення, наприклад, за допомогою ASP.NET Identity. Це дозволяє захищати дані користувачів, обробляти реєстрації та логіни, а також забезпечувати рольовий доступ до окремих частин функціоналу.

Проєкт розроблявся з увагою до якості коду: здійснювався рефакторинг, уникались дублювання та неефективні рішення. Було впроваджено сучасні підходи до архітектури, що дозволяє легко розширювати функціонал у майбутньому.

Попри вже велику реалізовану частину, потенціал проєкту залишається відкритим: можливе додавання нових модулів, інтеграція платіжних систем, розширення фільтрації товарів тощо. Але вже зараз це стабільний, продуманий і технічно якісний продукт, що поєднує зручність користування з надійною серверною логікою.

ПЕРЕЛІК ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

- [C#](#)
- [.NET Core](#)
- [ASP.NET Core](#)
- [Visual Studio](#)
- [Postman](#)
- [Visual Studio Code](#)
- [SSMS](#)
- [MSSQL](#)
- [TypeScript](#)
- [React](#)
- [MaterialUI](#)
- [HTML](#)
- [CSS](#)
- [GitHub](#)
- [MVC](#)
- [HTTP](#)
- [REST-API](#)

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

- [Microsoft technical documentation](#)
- [Stackoverflow](#)
- [ChatGPT](#)

- [YouTube](#)
- [GitHub](#)
- [C# Corner](#)
- [Metanit](#)