Національний університет “Острозька академія”

Звіт про виконання лабораторних робіт

з дисципліни

Програмування на с#

Виконав:

студент групи КН-31

Марчук Микола

Острог 2023

Зміст

[**Завдання №1**](#_zcoupnnivqcq) **2**

[Завдання:](#_3znysh7) 2

[Робота з гітом:](#_j8qrfaqii368) 2

[Сайт-аналог:](#_d593b2g6n7y2) 4

[Список задач для проекту](#_ydbhj58f7b22) 5

[Базові функції роботи з даними на основі створеної БД:](#_9dx0kvxwpd0s) 6

[Висновки:](#_vquvfeb11yvj) 6

[**Завдання №2**](#_3wx5hugtef92) **7**

[Завдання:](#_y1yajedz2da) 7

[Завдання 1](#_9hm74tlxw8yg) 7

[Завдання 2](#_qop6jo4jbd1s) 8

[Висновки:](#_hl11be6xaafz) 13

# 

# 

# 

# 

# 

# Завдання №1

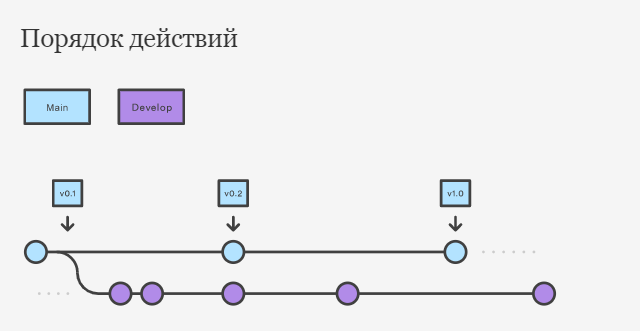
## Завдання:

**Основне завдання лабораторної роботи №1: реалізувати базові функції додавання/редагування та видалення даних на основі створеної БД.**

* Обрати тему.
* Cформувати групу (2 максимум, поки вистачить) або прийняти рішення про індивідуальне виконання. *Примітка. Дедлайн затвердження теми* ***14:00 17 жовтня.*** *Зверніть увагу, що тему може також запропонувати і студент, не обов’язково обирати з даного переліку.*
* Визначити ролі у проєкті: Team Lead, Developer.
* Обрати сайт-аналог, з якого братиметься орієнтовна структура БД.
* Розділити завдання на підзадачі (скласти чіткий список).
* Розподілити задачі між учасниками виконання проєкту.
* Створити репозиторій на github/ bitbucket / azure devops тощо.
* Запросити учасників команди.
* Кожну задачу виконує визначений учасник у окремій гілці, Team Lead робить merge у main.
* Підготовка звіту.
* Публічний захист.

## Робота з гітом:

Я працював сам, тому і не було розподілення на ролі. Коміти я робив у main одразу, але в реальних проектах ніхто так не робить. Main - це production бранча, тому будь-які зміни мають проходити по gitflow <https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflow-workflow>



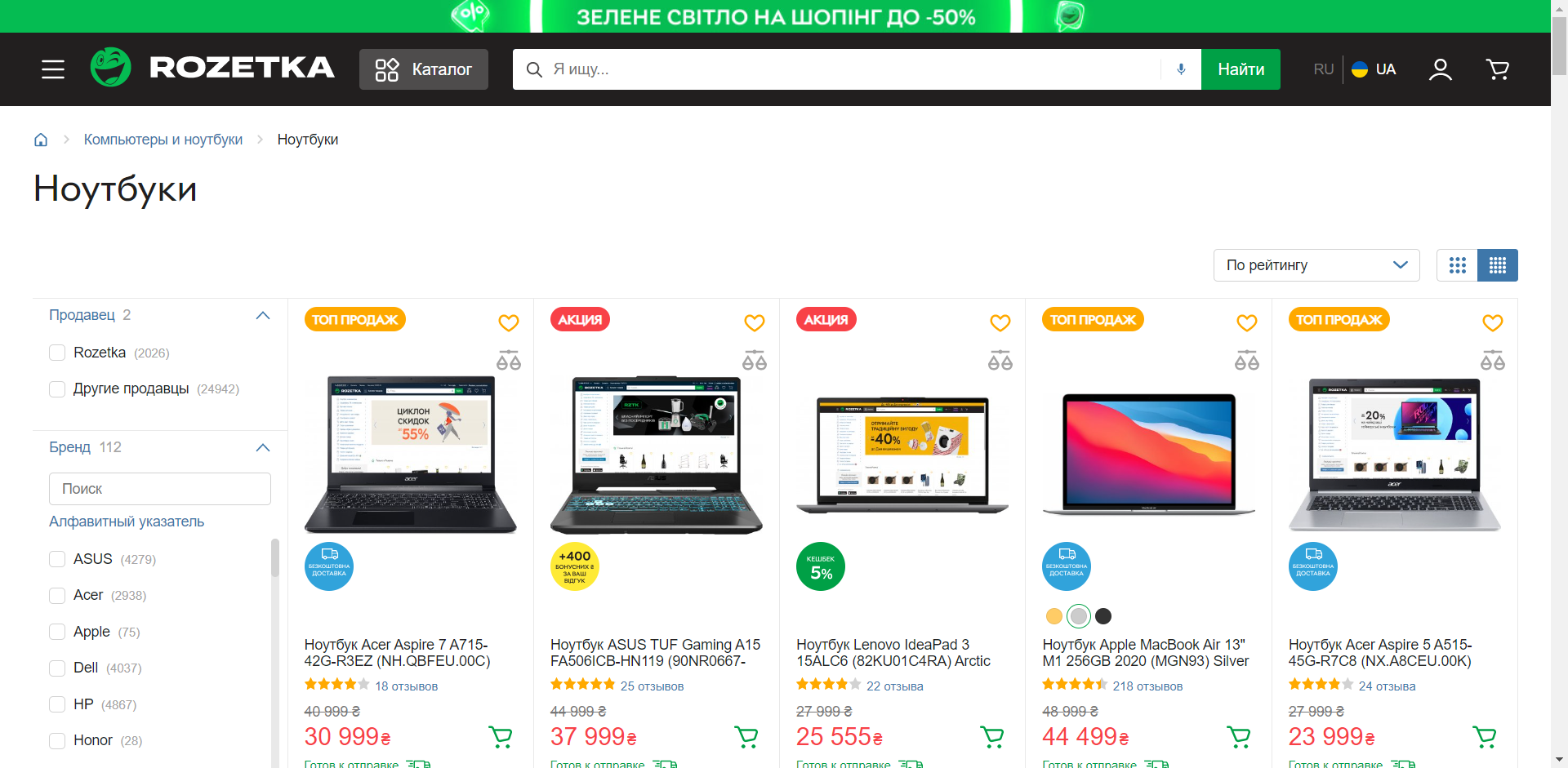
Також хорошою практикою є ще одна прослойка між production та готовими до тестування задачами - develop або ще називають stage.

Зазвичай процес delivery задачі до production виглядає наступним чином:

* формується реліз з деяких задач
* розробник асайнить на себе задачу
* розробник заходить у бранчу develop
* спулює всі актуальні зміни (git pull)
* створює нову бранчу від develop (git checkout -b branchName)
* робить якісь зміни
* git add .
* git commit -m “message”
* git push --set-upstream origin branchName
* створює merge request у github/gitbucket тощо до develop
* за потреби вирішує конфлікти, проходить рев’ю, отримує апрув
* змержує задачу з develop

## Сайт-аналог:

Натхнення брав з <https://rozetka.com.ua/>



Доволі успішний бізнес, якщо б не виникло ускладнень пов’язаних з війною, rozetka мали розширюватись на міжнародний ринок. У них доволі інтуїтивний UX/UI та потужний функціонал для пошуку та фільтрації потрібного користувачу товару серед десятків тисяч.

## Список задач для проекту

Нижче буде список, червоним позначені ті задачі, які ще в процесі розробки:

1. Створення asp.net проекту,
2. Додавання початкового контролеру продуктів, структура база даних, перша міграція, swagger тести
3. Створення клієнту на базі react та mui, обробка запитів на локальний бекенд з локального клієнту, обробка cors
4. Налаштування роутів на клієнті, логування помилок та тестові помилки на клієнті
5. Логіка та контролер корзини та відображення ui логіки на фронті
6. Додавання централізованого сховища для фронта
7. фільтрація та пагінація продуктів
8. Додавання сутності юзеру та авторизації на бекенді
9. Авторизація на фронті
10. Створення чекауту та сутності order(замовлення), історія замовлень
11. Додавання обробки платежів за допомогою провайдера Stripe
12. Відправка емейлу на пошту для юзера, де буде інформація по замовленню та інформація по доставці
13. Додавання сутності категорій та підкатегорій та CRUD для них, який доступний тільки для Адміна, дати можливість завантажувати картинки через url з firebase або подібного сервісу на asp.net, а не брати їх напряму з клієнта
14. Додавання CRUD для продуктів, доступний тільки для Адміна
15. Додавання UI для адміна для редагування продуктів, категорій тощо…
16. редизайн з допомогою дизайнера та адаптація під мобільні пристрої

## Базові функції роботи з даними на основі створеної БД:

описано у лабораторній по wcf [Лаб1-2\_Марчук](https://docs.google.com/document/d/1Wv9pXJZHglndkn1uISzQNE6AM3rDdXm4dbSGjFGwY-o/edit)

## Висновки:

На цій лабораторній роботі я навчився створювати Restful api для комунікації бекенду з фронтом, розібрався з тим як працює Cors, розібрався з авторизацією на asp net

Зони для покращення

1. Дороблення існуючих задач

# 

# 

# 

# 

# Завдання №2

## Завдання:

Потрібно виконати наступні завдання:

1. Реалізувати категорії та підкатегорії різної складності вкладеності.
2. Додати зображення до категорій
3. Реалізувати пошук по каталогу товарів: як по усіх товарах так і по конкретній категорії окремо.
4. Додати можливість формування замовлення з вибором типу доставки, типу оплати і т.д.
5. Реалізувтаи можливість підтвердження замовлення зі сповіщенням на електронну пошту :)

## Завдання 1-2:

Наразі сутності у проекті як категорія та підкатегорія немає, але є розуміння що це має бути відношення one-to-many(Одна категорія може мати багато підкатегорій).

Але кожна підкатегорія також є сутністю Category. В такому разі ми можемо мати велику кількість вкладень, для прикладу як у rozetka. На приклад: у нас є категорія Електроніка, у ній є підкатегорії Ноутбуки та Смартфони, у Ноутбуків є розділення на категорії ігрових та ноутбуків для роботи тощо...

Нижче приклад для entity

<https://habr.com/ru/post/516596/>

## Завдання 3:

Реалізовано, детальніший опис у [Лаб1-2\_Марчук](https://docs.google.com/document/d/1Wv9pXJZHglndkn1uISzQNE6AM3rDdXm4dbSGjFGwY-o/edit)

## Завдання 4-5:

Наразі не реалізовано, але є у планах на майбутнє

<https://www.c-sharpcorner.com/blogs/sending-email-in-asp-net-mvc>

Приклад відправки емейлу для asp net

### Висновки:

На цій лабораторній роботі я зробив справжню фільтрацію за багатьма параметрами, а також створив вивід та фільтрацію продуктів на клієнті

Використані джерела

* <https://khalilstemmler.com/articles/object-oriented/programming/4-principles/>
* <https://jakeydocs.readthedocs.io/en/latest/security/authentication/identity.html>
* <https://learn.microsoft.com/en-us/aspnet/core/web-api/action-return-types?view=aspnetcore-7.0>
* <https://jwt.io/introduction>
* <https://learn.microsoft.com/ru-ru/dotnet/api/system.linq.iqueryable-1?view=net-7.0>
* <https://code-maze.com/paging-aspnet-core-webapi/>
* <https://mui.com/>
* <https://redux-toolkit.js.org/>