Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ

завідувач кафедри
мережевих та інтернет технологій

_____ Ю.В. Кравченко

«_____»_____2021 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему:

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ ШАБЛОНУ «МОДЕЛЬ-ВІДОБРАЖЕННЯ-КОНТРОЛЕР» ДЛЯ РОЗРОБКИ ІНТЕРНЕТ-МАГАЗИНУ

Виконав: студент групи МІТ-41

Дев'яткін Михайло Дмитрович _____
(прізвище ім'я по-батькові) (підпис)

Керівник: доцент кафедри мережевих та інтернет технологій

к.т.н., асистент Махович О.І. _____
(посада, прізвище ім'я по-батькові) (підпис)

Київ 2021

Міністерство освіти і науки України
«Київський Національний університет імені Тараса Шевченка»

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2021 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Дев'яткіну Михайлу Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи:
Особливості застосування шаблону «Модель-відображення-контролер» для розробки інтернет-магазину

затверджена на засіданні кафедри МІТ, протокол «11» грудня 2020 р. протокол № 6

2. Термін здачі закінченої роботи «30» травня 2021 р.

3. Вихідні дані до проекту (роботи)

Мова програмування – Python, HTML

Технології: Flask, SQLAlchemy

4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)

Вступ

1. Аналіз і порівняння сучасних способів реалізації інтернет-магазину на шаблоні MVC

1.1 Огляд шаблону MVC та його особливості.

1.2 Аналіз сучасних вимог до інтернет-магазину.

1.3 Постанова задачі.

1.4 Аналіз мов та фреймворків для роботи з шаблоном MVC.

1.5 Вибір типу бази даних для роботи.

2. Аналіз існуючих компонентів для реалізації веб-магазину на мікрофреймвроку Flask

2.1 Шаблонізатор Jinja2.

2.2 SqlAlchemy.

2.3 Flask-Login.

3. Реалізація інтернет-магазину на шаблоні MVC

3.1. Аналіз способів реалізації.

3.2. Розробка структури інтернет-магазину.

3.3. Реалізація веб-додатку.

3.4. Створення інтерфейсу користувача.

Висновки

Дата видачі завдання

Керівник роботи асистент Махович О.І.
(підпис) (посада, прізвище, ім'я, по батькові)

Завдання прийняв до виконання Дев'яткін М.Д.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	16.03.2021	
2	Розділ 1	26.03.2021	
3	Розділ 2	05.04.2021	
4	Розділ 3	19.04.2021	
5	Доповідь та слайди	25.05.2021	
6	Пояснювальна записка	30.05.2021	

Здобувач вищої освіти _____ М.Д. Дев’яткін
(підпис)

Керівник _____ О.І. Махович
(підпис)

РЕФЕРАТ

Пояснювальна записка: 37 с., 8 рис., 12 джерел.

Об’єкт дослідження: особливості використання шаблону «Модель-відображення-контролер».

Предмет дослідження: шаблон «Модель-відображення-контролер» на прикладі інтернет-магазину.

Мета роботи: створення інтернет-магазину на шаблоні «Модель-відображення-контролер».

Методи дослідження: сучасні засоби розробки.

В роботі проведено аналіз: шаблону «Модель-відображення-контролер», сучасні способи реалізувати шаблон MVC, вимог до інтернет-магазинів.

Запропоновано: використання мови програмування Python для написання веб-додатку за допомогою шаблонізатора Jinja, використання Flask-Login для ведення контролю сесії користувача.

Побудовано: схему бази даних інтернет-магазину.

Розроблено: веб-додаток для інтернет-покупок, база даних для зберігання даних о користувачах та продуктів покупок.

Практичне значення роботи полягає у тому, що веб-додаток може працювати на будь якій базі даних. За рахунок цього його можна вбудувати у інші системи. Користувачом через систему реєстрації може стати будь яка людина.

Результати здійснених у дипломному проекті досліджень можуть бути використані для написання нових інтернет-магазинів обравши за основу веб-додаток у цій роботі.

Наукова новизна дослідження полягає у сучасному представлені шаблону «Модель-відображення-контролер» для інтернет-магазинів.

Галузь використання – комерційні проекти України.

Напрямки подальших досліджень: поліпшення розуміння продавця та покупця у інтернет середовищі, нові методики розробки інтернет-магазинів.

Ключові слова: МОДЕЛЬ-ВІДОБРАЖЕННЯ-КОНТРОЛЕР, PYTHON, ІНТЕРНЕТ-МАГАЗИН, ФРЕЙМВОРК, FLASK, MVC.

ВСТУП

Зараз, у сучасному світі, коли інформаційні технології швидко розвиваються в усіх областях діяльності людини, магазини та продаж через Інтернет не є винятком. Через сучасне положення речей, у тому числі через коронавірусні захворювання, прискорюється загальна діджиталізація та онлайн розвиток малого, і не тільки, бізнесу в Україні.

Інтернет-магазини вже давно вийшли з тієї категорії послуг, для яких потрібно було спочатку витратити багато коштів на розробку, потім на підтримку системи, сплачувати не тільки за домен та хостинг, а ще й за регулярні оновлення та додавання нових функцій, що також потребують значних вкладень.

Майже кожен більш-менш успішний бізнес може дозволити собі особистий сайт, на якому, наприклад, може розмістити меню своїх товарів або послуг, організувати онлайн-продаж, рекламу на сайті, зручну адміністрацію та модерацію, авторизацію та аутентифікацію. Крім зручності по продажам, це також збільшує їх об'єм, розширює клієнтську базу та полегшує рекламу магазину.

Нині є різні шаблони для розробки веб-додатків, але основним є MVC (Model-View-Controller) чи українською Модель-відображення-контролер через її популярність та легкість в розумінні. Основним концептом Модель-відображення-контролер є розділення та розбиття коду, який відповідає за частину, що побачить користувач, та обробку даних. Завдяки цьому конструювати, розробляти та масштабувати веб-додаток стає набагато легше.

Різні фреймворки на різних мовах програмування майже цілком задовольняють потреби сучасного клієнта, тому під час виконання роботи я оберу мову програмування та буду використовувати один з сучасних фреймворків.

1 АНАЛІЗ І ПОРІВНЯННЯ СУЧАСНИХ СПОСОБІВ РЕАЛІЗАЦІЇ ІНТЕРНЕТ-МАГАЗИНУ НА ШАБЛОНІ MVC

1.1 Огляд шаблону MVC та його особливості

Шаблон MVC – один з найважливіших шаблонів дизайну веб-додатків. У той час як більшість шаблонів вирішують конкретні проблеми, MVC описує архітектуру системи об’єктів. Він може застосовуватися як до ізольованих підсистем, так і до цілих додатків. Шаблон дизайну MVC також менш чітко визначений, ніж багато інших моделей, залишаючи розробнику багато свободи для альтернативних реалізацій.

Основні пункти MVC, згідно з назвою, – це модель, відображення, контролер.

Модель надає дані, реагуючі на команди контролера та змінюючи стан згідно цих самих команд. У більшості ситуацій модель працює безпосередньо з базою даних. Серед можливих операцій: оновлення, видалення, отримання та вставка даних.

Відображення відображає стан даних моделі для користувача. Воно містить HTML, CSS, JS або будь-які інші мову розмітки, які ми можемо використовувати для створення інтерфейсу. Єдине, що має зробити відображення, – це показувати дані клієнту в Інтерфейсі користувача та реагувати на події. Наприклад, що робити, коли користувач натискає кнопку «Оновити» або «Видалити». Відповідь полягає в тому, що користувача слід перенаправити до форми оновлення або до спливаючого вікна підтвердження видалення.

Контролер служить зв'язком або, іншими словом, інтерфейсом між користувачем та системою. Він отримує вхідні даних користувача і вирішує, що з цим робити. Контролер вирішує, яким був вхід користувача, як модель повинна змінитися в результаті цього введення і який результуючий вигляд слід використовувати.

Шаблон MVC набув популярністі та такого успіху завдяки розділенню бізнес-логіки та її візуалізації, через це збільшується процент повторного використання програмного кода та зрозумілість у великих проектах, який компонент за що відповідає.

Не останнім пунктом буде те, що багато основних мов програмування мають перевіренні поширені фреймворки з наглядною документацією, а також реалізацію у самих мовах цього шаблону.

Переваги шаблону MVC:

- Розробка програми стає швидкою.
- Кілька розробників можуть легко співпрацювати та працювати разом.
- Простіше оновити додаток.
- Простіше налаштувати, оскільки у нас є кілька рівнів, правильно записаних у програмі.

Недоліки шаблону MVC:

- Важко зрозуміти архітектуру MVC у перший раз.
- Повинні бути суворі правила щодо методів.

Недоліків архітектури мало та вони не такі великі, тому їх можна ігнорувати порівняно з усіма перевагами, які ми отримуємо.

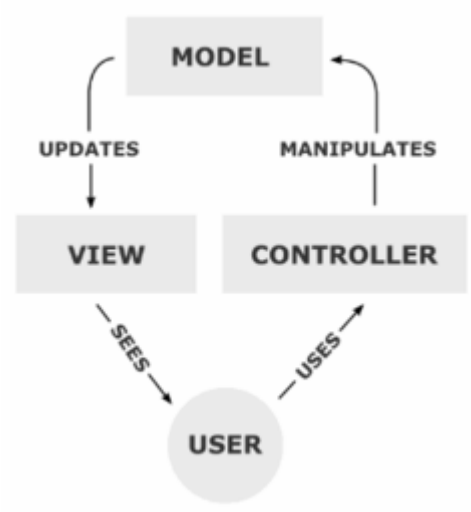

Рисунок 1.1 – Схематичне відображеня шаблону MVC

1.2 Аналіз сучасних вимог до інтернет-магазину

1.2.1 Роль користувальницького інтерфейсу на сайті

Користувальницький досвід охоплює широту відвідування магазину, замовлень товарів в Інтернеті за допомогою програм для обслуговування клієнтів. Дотримання фундаментальних і послідовних принципів дизайну полегшує клієнтам здійснення покупок.

Користувальницький досвід ніколи не є просто естетикою; це наука використання дизайну, щоб допомогти клієнтам досягти цілей, заради яких вони прийшли до вашого магазину. Усе, від кольору кнопок до шрифтів та кількості натискань, має бути розроблено відповідно до принципів орієнтованого на користувача дизайну.

Люди зазвичай дуже швидко вирішують, подобається їм веб-сайт чи ні; саме тому вам слід викликати їх зацікавленість за лічені секунди після їх входу на сайт, інакше вони просто знайдуть інший магазин з більш привабливою домашньою сторінкою.

1.2.2 Промо-пропозиції

Ніщо не приваблює краще та швидше, ніж торгові пропозиції. Багато людей хочуть купити під час розпродажів та знижок та купують щось за зниженими цінами лише тому, що воно продається зі знижкою.

Деяких клієнтів приваблює безкоштовна доставка, деяких – значні знижки, деяких – подарунки, але майже всіх цікавить той чи інший вид промо-пропозиції. Таким чином, знижки та ексклюзивні пропозиції – це, як правило, перше, на що звертають увагу відвідувачі веб-сайтів. Привабливі обіцянки та унікальні ціни стимулюють відвідувачів витрачати кошти.

1.2.3 Розміщення матеріалу на сайті

Коли є новини, періоди розпродажів або майбутні події, які покупці повинні знати про товар, то більш логічним місцем для розміщення цієї інформації буде головна сторінка магазину.

Якщо розміщати це на інших сторінках, це лише ускладнить покупцям пошук ексклюзивних пропозицій та гарячих цін. Більше того, постійні клієнти швидше переглядатимуть кілька «нових» товарів у продажу, а не витрачатимуть півгодини, переглядаючи повний інвентар для чогось нового.

Не завжди можливо передбачити, що шукатиме наступний клієнт, але це не означає, що потрібно розміщувати всі товари безпосередньо на домашній сторінці.

Потрібно розташовувати найпривабливіші та найцікавіші пропозиції у зоні легкого доступу. Якщо є фірмові товари у продажу, то потрібно показувати їх заздалегідь. Це привертає увагу, особливо серед відвідувачів, які вперше на сайті та ще точно не знають, що саме вони шукають.

1.2.4 Роль мобільних користувачів

Особливо заслуговує на увагу зростання мобільних покупок. Покращення досвіду електронної комерції для мобільних клієнтів може стати можливістю для бізнесу на залучення нових клієнтів.

Зростання електронної комерції частково зумовлене збільшенням використання мобільних пристроїв. Це пов’язано з тим, що все більше споживачів використовують свої мобільні пристрої для перегляду чи дослідження, перш ніж вирішити питання про свою покупку.

Зі збільшенням довіри до покупок в Інтернеті споживачі починають відчувати себе комфортніше та робити покупки на мобільних пристроях частіше, ніж коли-небудь. Особливо це стосується споживачів, які виросли в оточенні

комп’ютерів та Інтернету. Ці покоління частіше здійснюють покупки в Інтернеті за допомогою своїх мобільних пристроїв у порівнянні зі старшими поколіннями.

1.2.5 Візуальна комерція

При веденні інтернет-магазину вважається труднощами саме необхідність продавати товар споживачам, які не мають шансів на фізичну взаємодію з товаром до самої покупки. Ось тут візуальна комерція набуває велике значення.

Візуальна комерція – це наступне покоління звичайних статичних способів візуалізації. Вона виводить маркетинг на зовсім інший рівень. Замість того, щоб для просування вашого бізнесу просто використовувати фотографії товару, візуальна комерція іде далі, включаючи інші типи візуальних матеріалів, таких як створені споживачами медіа, інтерактивний контент, цікаві відео.

Візуальна комерція повільно, але впевнено стає невід’ємною частиною електронної комерції, про що свідчить постійний розвиток глибоких технологій, що стоять за нею.

1.3 Постанова задачі

Результатом виконання дипломної роботи має бути працюючий інтренет-магазин на основі MVC шаблону. Інтернет-магазин має включати в себе:

- Веб-додаток;
- База даних.

Веб-додаток розроблений, щоб приймати команди користувача, показ, оновлення, видалення чи додавання даних. Також додаток має надавати можливість користувачам реєструватись, авторизуватись та аутентифікуватись.

Веб-додаток повинен мати приємний для користувачів інтерфейс, на котрому будуть сторінка з товарами, сторінка користувача, корзина.

База даних має приймати дані та зберігати їх до часу витягування.

1.4 Аналіз мов та фреймворків для роботи з шаблоном MVC

Перш за все потрібно було обрати мову програмування, яка більш підходить до цілей завдання. А тобто, є легка в розумінні та освоєні, має можливість за допомогою фреймворків чи вбудованих компонентів реалізувати веб-додаток на основі MVC.

Список мов, які більш за все підходять до задачі:

- C#
- Python
- Java
- Ruby

1.4.1 C#

C# – це проста, сучасна та об'єктно-орієнтована мова, яка забезпечує сучасним розробникам гнучкість та можливості для побудови програмного забезпечення, яке не тільки буде працювати сьогодні, але й буде застосовуватися протягом багатьох років у майбутньому. Розроблена в 1998-2001 роках груповою інженерною компанією Microsoft, як мова програмування для платформ Microsoft .NET Framework.

C# відноситься до сім'ї мов з C-подібним синтаксисом, із них її синтаксис найближче до C++ та Java.

Основні характеристики мови C# включають:

- Сучасна і проста
- Швидкий і відкритий код
- Крос-платформена
- Безпечна

- Універсальна
- Еволюціонує

Серед фреймворків для реалізації MVC в основному використовується ASP.NET MVC чи ASP.NET Web Forms.

Платформа ASP.NET MVC базується на взаємодії трьох компонентів: контролера, моделі та відображення. Контролер приймає запити, обробляє користувача введення, взаємодіє з моделлю і відображенням та повертає користувачеві результат обробки запиту.

Модель представляє шар, що описує логіку організації даних в додатку. Відображення отримує дані з контролера і генерує елементи призначеного для користувача інтерфейсу для відображення інформації.

ASP.NET Web Forms – це фреймворк для веб-додатків і одна з декількох моделей програмування, що підтримуються технологією Microsoft ASP.NET. Програми Web Forms можуть бути написані будь-якою мовою програмування, яка підтримує Common Language Runtime, наприклад C # або Visual Basic. Основними будівельними блоками сторінок веб-форм є серверні елементи керування, які є багаторазовими компонентами, що відповідають за розмітку HTML-розмітки та реагування на події. Техніка, яка називається станом перегляду, використовується для збереження стану серверних елементів керування між HTTP-запитами, які зазвичай не мають стану.

ASP.NET MVC і ASP.NET Web Forms є двома спорідненими технологіями, в основі яких лежить одна платформа ASP.NET. І все ж ASP.NET MVC має ряд переваг перед ASP.NET Web Forms:

- поділ відповідальності (окрема розробка різних компонентів – контролера, моделей, відображення);
- поліпшене тестування;
- підвищена гнучкість і зручні налаштування під власні потреби.

1.4.2 Python

Python – інтерпретована, об’єктно-орієнтована мова програмування високого рівня. Одна з тих рідкісних мов, яка може претендувати як на просту, так і на потужність. Python є прикладом вільного програмного забезпечення та програмного забезпечення з відкритим кодом. Це базується на концепції спільноти, яка ділиться знаннями. Це одна з причин, чому Python настільки хороший - він був створений і постійно вдосконалюється спільнотою, яка просто хоче бачити кращу мову програмування.

Так як Python мова високого рівня, то не потрібно турбуватися про деталі низького рівня, такі як управління пам'яттю, що використовується програмою. Завдяки своїй природі відкритого коду, Python перенесено на багатьох платформах. Усі програми Python можуть працювати на будь-якій з цих платформ, не вимагаючи жодних змін, якщо уникнути будь-яких системно-залежних функцій. Можливо використовувати Python на GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE та PocketPC.

Python не потребує компіляції в двійковий файл. Внутрішньо Python перетворює вихідний код у проміжну форму, що називається байт-кодами, потім перекладає це на мову комп'ютера, а потім запускає його. Все це, власне, значно полегшує використання Python, оскільки не потрібно турбуватися про компіляцію програми, переконавшись, що належні бібліотеки пов’язані та завантажені тощо.

Стандартна бібліотека Python має багато функціонала. Це допомагає робити різні справи, включаючи регулярні вирази, модульне тестування, створення потоків, бази даних, веб-браузерів, CGI, FTP, електронну пошту, XML, XML-RPC, HTML, файли WAV, криптографію, графічний інтерфейс користувача.

Для реалізації MVC у Python використовують фреймворки Flask та Django.

Flask – фреймворк для створення веб-додатків на мові програмування Python, що використовує набір інструментів Werkzeug, а також шаблонізатор Jinja2. Відноситься до категорії так званих мікрофреймворков – мінімалістичний каркасів веб-додатків, свідомо надаються лише базові функції.

В мікрофреймворку «мікро» означає, що Flask прагне зробити ядро простим, але розширюваним. Flask не прийме за розробника багато рішень, наприклад, яку базу даних використовувати. Ті рішення, які він приймає, наприклад, який механізм шаблонування використовувати, легко змінити. Все інше залежить від розробника, так що Flask може бути всім, що потрібно розробнику.

За замовчуванням Flask не включає рівень абстракції бази даних, перевірку форми чи багато іншого, де вже існують різні бібліотеки, які можуть з цим впоратись. Натомість Flask підтримує розширення для додавання такої функціональності до вашої програми, як якщо б вона була реалізована в самій Flask. Численні розширення забезпечують інтеграцію баз даних, перевірку форми, обробку завантажень, різні технології відкритої аутентифікації тощо. Flask може бути «мікро», але він готовий до використання для різних потреб розробника.

Django – вільний фреймворк для веб-додатків на мові Python, що використовує шаблон проектування MVC. Проект підтримується організацією Django Software Foundation.

Django, фреймворк охоплює практично все, що вам знадобиться у веб-програмі. Пакети фреймворка включають API, системи управління вмістом, автентифікацію користувачів, перевірку форми. Запуск проекту на Django дозволяє побудувати всю модель даних програми на Python без необхідності використання SQL. Використовуючи об’єктно-реляційне відображення, Django перетворює традиційну структуру бази даних у класи Python, щоб полегшити роботу в середовищі Python. Django-MySQL підтримує тип даних JSON та необхідні функції.

Веб-фреймворк Django полегшує масштабування проекту. Оскільки програма на Django може керувати сеансами користувачів, можливо додавати більше екземплярів програми та передавати досвід користувача між екземплярами без втрати даних. Багато розробників проектів Django також використовують кеш-менеджер, такий як Varnish, для попереднього завантаження статичних елементів сайту для користувачів.

1.4.3 Java

Java – строго типізована об'єктно-орієнтована мова програмування загального призначення, розроблена компанією Sun Microsystems. Права на торговельну марку належать корпорації Oracle.

Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якої комп'ютерної архітектурі, для якої існує реалізація віртуальної Java-машини.

Ось деякі важливі функції Java:

- Це одна з простих у використанні мов програмування для вивчення.
- Написаний код один раз можливо запустити практично на будь-якій обчислювальній платформі.
- Java не залежить від платформи. Програми, розроблені на одній машині, можуть виконуватися на іншій машині.
- Вона призначена для побудови об'єктно-орієнтованих додатків.
- Це багатопотокова мова з автоматичним управлінням пам'яттю.
- Вона створена для розподіленого середовища Інтернету.
- Сприяє розподіленим обчисленням як орієнтованим на мережу.

Для реалізації MVC на Java самим відомим фреймворком є Spring.

Spring - це найпопулярніший механізм розробки додатків для корпоративної Java. Один із найпопулярніших фреймворків Java Enterprise Edition, Spring допомагає розробникам створювати високопродуктивні додатки, використовуючи

звичайні об'єкти Java. Spring framework - це платформа Java з відкритим кодом. Спочатку він був написаний Родом Джонсоном і вперше був випущений за ліцензією Apache 2.0 у червні 2003 року. Окрім Spring є багато популярних фреймворків Java, включаючи Java Server Faces, Maven, Hibernate, Struts.

Spring вважається надійним, недорогим та гнучким фреймворком. Spring покращує ефективність кодування та скорочує загальний час розробки додатків, оскільки він легкий, ефективно використовує системні ресурси та має багато підтримки. Він видаляє нудну роботу з конфігурацією, щоб розробники могли зосередитися на написанні бізнес-логіки, також він обробляє інфраструктуру, щоб розробники могли зосередитись на додатку. Spring надає зручний API для перекладу винятків, специфічних для технологій , наприклад, JDBC, Hibernate або JDO, у послідовні, неперевірені винятки. Spring забезпечує послідовний інтерфейс управління транзакціями, який може масштабуватися до локальної транзакції та масштабуватися до глобальних транзакцій , наприклад, за допомогою JTA.

Веб-додаток (багатошарова архітектура) зазвичай включає три шари:

- Шар презентації (UI) - це самий зовнішній шар, який обробляє презентацію вмісту та взаємодію з користувачем.
- Рівень бізнес-логіки - центральний рівень, який займається логікою програми.
- Рівень доступу до даних - глибокий рівень, який займається пошуком даних із джерел.

Кожен рівень залежить від іншого, щоб програма працювала.

Основною логікою Spring є введення залежностей. Введення залежності означає, що Spring розуміє різні анотації Java, які розробник ставить поверх класів. Spring знає, що розробник хоче створити екземпляр класу і що Spring повинен ним керувати. Spring також розуміє залежність і гарантує, що всі створені екземпляри мають належним чином заповнені залежності.

1.4.4 Ruby

Ruby – динамічна, рефлективна, інтерпретована високорівнева мова програмування. Мова має незалежну від операційної системи реалізацію багатопоточності, сильну динамічну типізацію, збирача сміття і багато інших можливостей. За особливостями синтаксису вона близька до мов Perl і Eiffel, по об'єктно-орієнтованому підходу – до Smalltalk. Також деякі риси мови взяті з Python, Lisp, Dylan і CLU.

Особливості Ruby:

- Ruby є програмою з відкритим кодом і знаходиться у вільному доступі в Інтернеті, але на нього поширюється ліцензія.
- Ruby - інтерпретована мова програмування загального призначення.
- Ruby - справжня об'єктно-орієнтована мова програмування.
- Ruby - це мова сценаріїв на стороні сервера, подібна до Python та Perl.
- Ruby можна вбудувати в мову розмітки гіпертексту (HTML).
- Ruby має чіткий і простий синтаксис, який дозволяє новому розробнику вчитися дуже швидко і легко.
- Ruby має синтаксис, подібний до синтаксису багатьох мов програмування, таких як C ++ та Perl.
- Ruby дуже масштабований, і великі програми, написані на Ruby, легко підтримувати.
- Ruby можна використовувати для розробки програм Інтернету та Інтранета.
- Ruby можна встановити в середовищах Windows і POSIX.
- Ruby підтримує багато інструментів графічного інтерфейсу, таких як Tcl / Tk, GTK та OpenGL.
- Ruby можна легко підключити до DB2, MySQL, Oracle та Sybase.

Ruby має багатий набір вбудованих функцій, які можна використовувати безпосередньо в скриптах Ruby.Rails – це фреймворк для розробки веб-додатків, написаний мовою програмування Ruby. Він розроблений для полегшення програмування веб-додатків, роблячи припущення щодо того, що потрібно кожному розробнику для початку роботи. Це дозволяє писати менше коду, виконуючи більше, ніж багато інших мов та фреймворків. Rails робить припущення, що існує «найкращий» спосіб зробити щось, і він покликаний заохочувати такий шлях – а в деяких випадках і перешкоджати альтернативам. Створено Давидом Хейнемейер Ханссоном на основі його роботи в компанії 37signals над засобом управління проектами Basecamp і випущений в липні 2004 року.

Філософія Rails включає два основних керівних принципи:

- Не повторюйся (Don't Repeat Yourself): DRY – це принцип розробки програмного забезпечення, який стверджує, що "кожна частина знань повинна мати єдине, однозначне, авторитетне представлення в системі". Якщо ми не пишемо одну і ту ж інформацію знову і знову, наш код стає більш ремонтопридатним, більш розширюваним і має менше помилок.
- Конвенції понад конфігурацій: Rails має настройки щодо найкращого способу робити багато речей у веб-програмі та за замовчуванням застосовує цей набір конвенцій, а не вимагає, щоб ви вказували дрібниці через велику кількість файлів конфігурацій.

1.4.5 Вибір мови та фреймворків для роботи

Після розбору сучасних мов та фреймворків, я можу визначитись з якою технологією буду працювати, порівнюючи їх один з одним.

Перш за все потрібно обирати з мов та фреймворків, в яких я маю досвід роботи та вже зроблені проекти, бо краще розкрити тему можна у тій системі, у

котрій ти маєш навички та яка підходе до проекту. Таким чином, відсіюється мова Ruby, хоч вона і має свої переваги.

Обираючи з мов C#, Python та Java, для того щоби показати роботу й особливості шаблону MVC для інтернет-магазину, кожна з них має свої недоліки та переваги. Java разом з C# більше підходить для великих проектів, які будуть розширюватись. У моєму випадку треба більш простіше рішення задачі, яка після роботи навряд чи буде масштабуватись. Також великою перевагою у моєму проекті буде мова з гарною продуктивністю роботи та швидким циклом редагування-тестування-налагодження. Саме через ці причини я обрав мову програмування Python.

У Python є два популярні фреймворки для розробки додатків на основі MVC. Це Django – фреймворк у котрому відразу встроєно багато елементів, більшість яких потрібні для реалізації MVC. Але моя задача не потребує так багато функціоналу який надає цей фреймворк, тому на заміну підходить більш модульний мікрофреймворк Flask. Так, його базові можливості обмежені, але вони легко доповнюються за допомоги додаткових модулів. Наприклад, якщо нам потрібно зробити таку базову функцію як реєстрація, то ми скористаємося flask-login, а якщо потрібно підключати БД, то використаємо flask-sqlalchemy.

1.5 Вибір типу бази даних для роботи

Для розробки інтернет-магазинів використовують такі три основні типа систем управління баз даними: реляційна, об’єктно-орієнтована, реляційна-об’єктна.

1.5.1 Реляційні та нереляційні бази даних

Реляційна система управління базами даних (RDBMS або просто RDB) - це загальноприйнятий тип бази даних, що зберігає дані в таблицях, тому її можна використовувати щодо інших збережених наборів даних. Більшість баз даних, які

сьогодні використовуються підприємствами, є реляційними базами даних, на відміну від простого файлу або ієрархічної бази даних. Більшість сучасних систем та додатків базуються на реляційній СУБД.

Реляційні бази даних мають сили для обробки безлічі даних та складних запитів. Дані часто зберігаються в багатьох таблицях, які також називаються "відносинами". Ці таблиці розділені на рядки, які також називаються записами та стовпцями (полями). У базі даних може бути дуже багато рядків. Стовпці складаються з одного конкретного типу даних, наприклад, назви чи ціни.

На відміну від нереляційних баз даних, які також названі базою даних NoSQL, були в основному розроблені з урахуванням управління великими наборами даних.

Нереляційні бази даних не зберігають дані в рядках і стовпцях. Вони можуть використовувати будь-який формат, який є оптимальним для даних, які вони містять. На відміну від реляційних баз даних, вони приймають лише конкретні запити. Але оскільки вони не засновані на SQL, вони уникають жорстких схем для структурування даних.

Подібно до зв’язків між даними на діаграмі зв’язків сутності, таблиці в реляційній базі даних можуть бути пов’язані кількома способами

- Характеристики одного запису таблиці можуть бути пов'язані із записом в іншій таблиці
- Запис таблиці може бути пов'язаний з багатьма записами в іншій таблиці
- Багато записів таблиці можуть бути пов’язані з багатьма записами в іншій таблиці.

Ці відносини - це те, що бачите, переглядаючи, наприклад, зведені таблиці у візуалізаціях інформаційної панелі. Таким чином це робить складні взаємозв’язки легкими для розуміння.

1.5.2 Об’єктно-орієнтована база даних

Об’єктно-орієнтована база даних (ООБД) – це база даних, яка підписується на модель з інформацією, представленою об’єктами. Об'єктно-орієнтовані бази даних є нішевими пропозиціями в галузі реляційної системи управління базами даних (СУБД) і не настільки успішні чи відомі, як основні механізми баз даних.

Як випливає з назви, основною особливістю об’єктно-орієнтованих баз даних є надання можливості визначення об’єктів, які відрізняються від звичайних об’єктів баз даних. Об’єкти в об’єктно-орієнтованій базі даних посилаються на здатність розробляти продукт, а потім визначають і називають його. Потім на об’єкт можна посилатись або називати його пізніше як одиницю, не вдаючись до його складностей. Це дуже схоже на об'єкти, що використовуються в об'єктно-орієнтованому програмуванні.

Реальна паралель об’єктам – це автомобільний двигун. Він складається з декількох частин: головного блоку циліндрів, вихлопної системи, впускного колектора тощо. Кожен із них є самостійним компонентом; але при механічній обробці та закріпленні болтами в одному об’єкті їх тепер спільно називають двигуном. Подібним чином, при програмуванні можна визначити декілька компонентів, таких як вертикальна лінія, що перетинає перпендикулярну горизонтальну лінію, тоді як обидві лінії мають градуйоване вимірювання. Потім цей об’єкт може бути спільно позначений графіком. Використовуючи можливість побудови компонентів, немає необхідності спочатку визначати графік; скоріше можна назвати екземпляр створеного графіка.

1.5.3 Об'єктно-реляційна база даних

Об'єктно-реляційна база даних (ОРБД) – це система керування базами даних (СУБД), яка складається як з реляційної бази даних (РБД), так і з об'єктно-орієнтованої бази даних (ООБД). ОРБД підтримує основні компоненти будь-якої

об'єктно-орієнтованої моделі бази даних у своїх схемах та використовуваній мові запитів, таких як об'єкти, класи та успадкування.

Об'єктно-реляційна база даних може також бути відома як об'єктна реляційна система управління базами даних (ОРСУБД).

Зазначається, що ОРБД є посередником між реляційними та об'єктно-орієнтованими базами даних, оскільки він містить аспекти та характеристики обох моделей. В ОРБД основний підхід заснований на РБД, оскільки дані зберігаються в традиційній базі даних, маніпулюють та отримують доступ за допомогою запитів, написаних мовою запитів, як SQL. Однак ОРБД також демонструє об’єктно-орієнтовану характеристику в тому, що база даних вважається сховищем об’єктів, як правило, для програмного забезпечення, що написане об’єктно-орієнтованою мовою програмування. Тут API використовуються для зберігання та доступу до даних як об’єктів.

Однією з цілей ОРБД є подолання розриву між концептуальними методами моделювання даних для реляційних та об'єктно-орієнтованих баз даних, таких як діаграма взаємозв'язку сутності та об'єктно-реляційне відображення. Вона також має мету з'єднати розділення між реляційними базами даних та об'єктно-орієнтованими методами моделювання.

Традиційні продукти ОРСУБД концентруються на ефективній організації даних, яка отримується з обмеженого набору типів даних. З іншого боку, ОРСУБД має функцію, яка дозволяє розробникам створювати та впроваджувати власні типи даних та методи, які можна застосувати до СУБД. Завдяки цьому ОРСУБД дозволяє розробникам збільшити абстракцію, за допомогою якої вони розглядають проблемну область.

1.5.4 Аналіз типів баз даних

Так як у інтернет-магазину багато речей будуть уявлятись як об’єкти у ORM та буде залежність між різними об’єктами, то найбільш доричним до використання стане реляційно-об’єктна система управління. А так як я

використовував Python разом із мікрофреймворком Flask у своєму проекті, то найбільш сумісна з Flask, а також з вимогами є реляційно-об’єктна система управління базами даних PostgreSQL.

1.5.5 PostgreSQL

PostgreSQL - це вдосконалена реляційна база даних з відкритим кодом корпоративного класу, яка підтримує як запити SQL (реляційні), так і JSON (нереляційні). Це стабільна система управління базами даних, що підтримується більш ніж 20-річним розвитком громади, що сприяло її високому рівню стійкості, цілісності та коректності. PostgreSQL використовується як сховище даних для багатьох веб, мобільних, геопросторових та аналітичних програм.

PostgreSQL має багату історію підтримки вдосконалених типів даних і підтримує рівень оптимізації продуктивності, який є загальним серед його комерційних аналогів баз даних, таких як Oracle та SQL Server.

PostgreSQL має надійні набори функцій, включаючи Multi-Version Concurrency Control (MVCC), відновлення в точці часу, детальний контроль доступу, табличні простори, асинхронну реплікацію, вкладені транзакції, гарячі резервні копії, вдосконалений оптимізатор запитів та реєстрацію.

Він підтримує міжнародні набори символів, багатобайтове кодування символів, Unicode, а також розпізнає місцевість для сортування, чутливості до регістру та форматування. PostgreSQL є дуже масштабованим як за кількістю даних, якими він може управляти, так і за кількістю одночасних користувачів, які він може вмістити.

Код PostgreSQL доступний за ліцензією з відкритим кодом, що дає свободу користуватися, модифікувати та впроваджувати його, як вважаєтеся за потрібне, безкоштовно. PostgreSQL не несе ліцензійних витрат, що виключає ризик надмірного розгортання. Спеціальна спільнота учасників та ентузіастів PostgreSQL регулярно знаходить помилки та виправлення, що сприяє загальній

безпеці системи баз даних.

2 АНАЛІЗ ІСНУЮЧИХ КОМПОНЕНТІВ ДЛЯ РЕАЛІЗАЦІЇ ВЕБ-МАГАЗИНУ НА МІКРОФРЕЙМВРОКУ FLASK

2.1 Шаблонізатор Jinja2

Для відокремлення функціональної частини від візуальної використовується шаблонізатори. Можливості шаблонізатора Jinja великі. Він здатний виводити стандартну HTML розмітку, а також виконувати: перевірку умов, виведення масивів через цикл, виведення змінних і багато іншого.

Раніше для виведення інформації використовували HttpResponse. Подібний підхід невірний, так як виводити багато HTML тегів в форматі одного рядка не зручно. Замість такого підходу використовують функцію render, що дозволяє виводити великі HTML-шаблони, в які ми вже додаємо код Jinja.

Особливості Jinja 2:

- Спадкування та включення шаблону.
- Можливість визначати та імпортувати макроси в шаблонах.
- Шаблони HTML можуть використовувати автоматичне сканування, щоб запобігти XSS (міжсайтовий скриптинг) від введення ненадійного користувача.
- Навчальне середовище може безпечно робити ненадійні шаблони.
- Підтримка AsyncIO для генерації шаблонів та виклику асінхронних функцій.
- Шаблони компілюються для оптимізованого коду Python під час виконання та кешуються, або можуть бути скомпільовані заздалегідь.
- Точки виняток вказують на правильний рядок у шаблонах, щоб полегшити налагодження.
- Розширювані фільтри, тести, функції та синтаксис.

2.2 SqlAlchemy

SQLAlchemy – це фреймворк, який на практиці використовується для роботи з реляційними базами даних на Python. Він був створений Майком Баєром у 2005 році. SQLAlchemy підтримує наступні бази даних: MySQL, PostgreSQL, Oracle, MS-SQL, SQLite та інші.

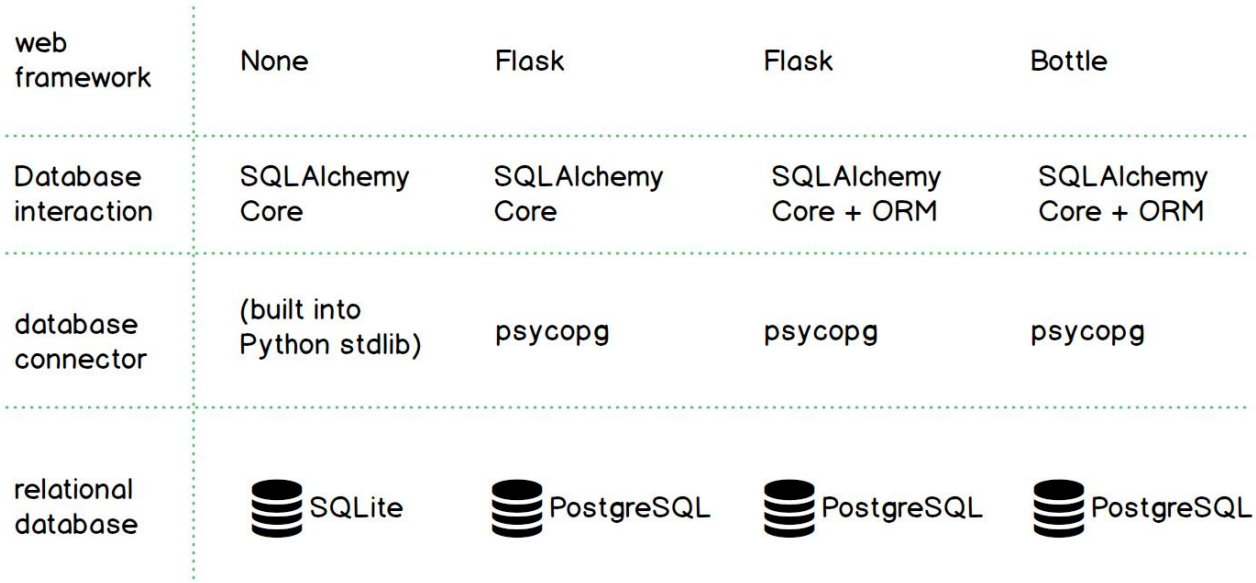

Рис 2.1 — приклад конфігурацій SQLAlchemy з різними веб-фреймворками.

SQLAchemy поставляється з потужним ORM (технологія об'єктно-реляційного відображення), що дозволяє працювати з різними базами даних за допомогою об'єктно-орієнтованого коду, а не сирого SQL (мова структурованих запитів). Звичайно, це не зобов'язує використовувати тільки ORM. У будь-який момент можна задіяти можливості SQL.

Причини використання ORM замість SQL-скрипту:

- Прискорення веб-розробки, адже зникає необхідність переключатися між написанням коду Python і SQL;
- Усунення повторюваного коду;
- Оптимізація робочого процесу і більш ефективні запити до бази даних;

- Абстрагування системи бази даних, так що перемикання між декількома базами стає більш плавним;
- Генерація шаблонного коду для основних операцій CRUD;
- SQLAlchemy зазвичай використовується з Flask як ORM бази даних через розширення Flask-SQLAlchemy.

Flask-SQLAlchemy – це розширення, яке інтегрує SQLAlchemy у фреймворк Flask. Він також пропонує додаткові методи, завдяки яким працювати з SQLAlchemy стає трохи простішим.

2.3 Flask-Login

Flask-Login забезпечує управління сеансами користувача для Flask. Він обробляє загальні завдання входу, виходу та запам'ятовування сеансів користувачів протягом тривалого періоду часу.

Він буде:

- Зберігати ідентифікатор активного користувача у сеансі та дозволяйте легко входити та виходити з системи.
- Дозволяти обмежувати перегляди користувачів, які увійшли або вийшли з системи.
- Обробляти звичайно функціональність "запам'ятай мене".
- Допоможіть захистити сеанси користувачів від викрадення злодіями cookie.
- Можлива інтеграція з Flask-Principal або іншими розширеннями авторизації пізніше.

Однак він не:

- Нав'язує конкретну базу даних або інший спосіб зберігання.

- Обмежує використання імен користувачів та паролів, OpenID або будь-якого іншого способу автентифікації.
- Обробляє дозволи, крім "ввійшли чи ні".
- Обробляє реєстрацію користувачів або відновлення облікового запису.

3 РЕАЛІЗАЦІЯ ІНТЕРНЕТ-МАГАЗИНУ НА ШАБЛОНІ MVC

3.1 Аналіз способів реалізації

Проаналізувавши усі доступні засоби розробки, я прийшов до висновку, що потрібно використовувати мову розробки програмного забезпечення Python разом з мікрофреймворком Flask, на якому буде розроблятись веб-додаток. Як систему управління баз даних буду використовувати PostgreSQL, через те що вона об'єктно-реляційна, а саме це дуже підходить до інтернет-магазину. Також важними факторами є її надійність, потужність, підтримка інших мов. За допомоги бази даних ми будемо реєструвати нових користувачів, додавати товари до магазину, проводити операції з кошиком та інше.

3.2 Розробка структури інтернет-магазину

База даних веб-додатку складається з декількох таблиць.

Таблиця User, у якій зберігаються дані про користувача: ім'я користувача, електронна пошта, файл з картинкою, яку користувач може завантажити собі у профіль, та пароль, який зберігається у шифрованому виді.

Таблиця Product, у якій зберігається: назва продукту, ціна за одну одиницю, кількість товару на складу та опис товару.

У таблиці Tag зберігається назва тегу.

Далі на схемі також представлені зв’язуючі таблиці tags та basket. Вони потрібні для можливості зв’язку багато до багатьох(many-to-many), бо на прикладі тегів, один продукт може мати багато тегів, а один тег, у свою чергу, може мати багато продуктів. У таблиці basket, окрім зв’язуючих полів product_id та user_id, є ще поле quantity – кількість продукту, тобто скільки даного продукту замовив користувач у свою корзину.

Схему бази даних інтернет-магазину можно побачити на рисунку 3.1.

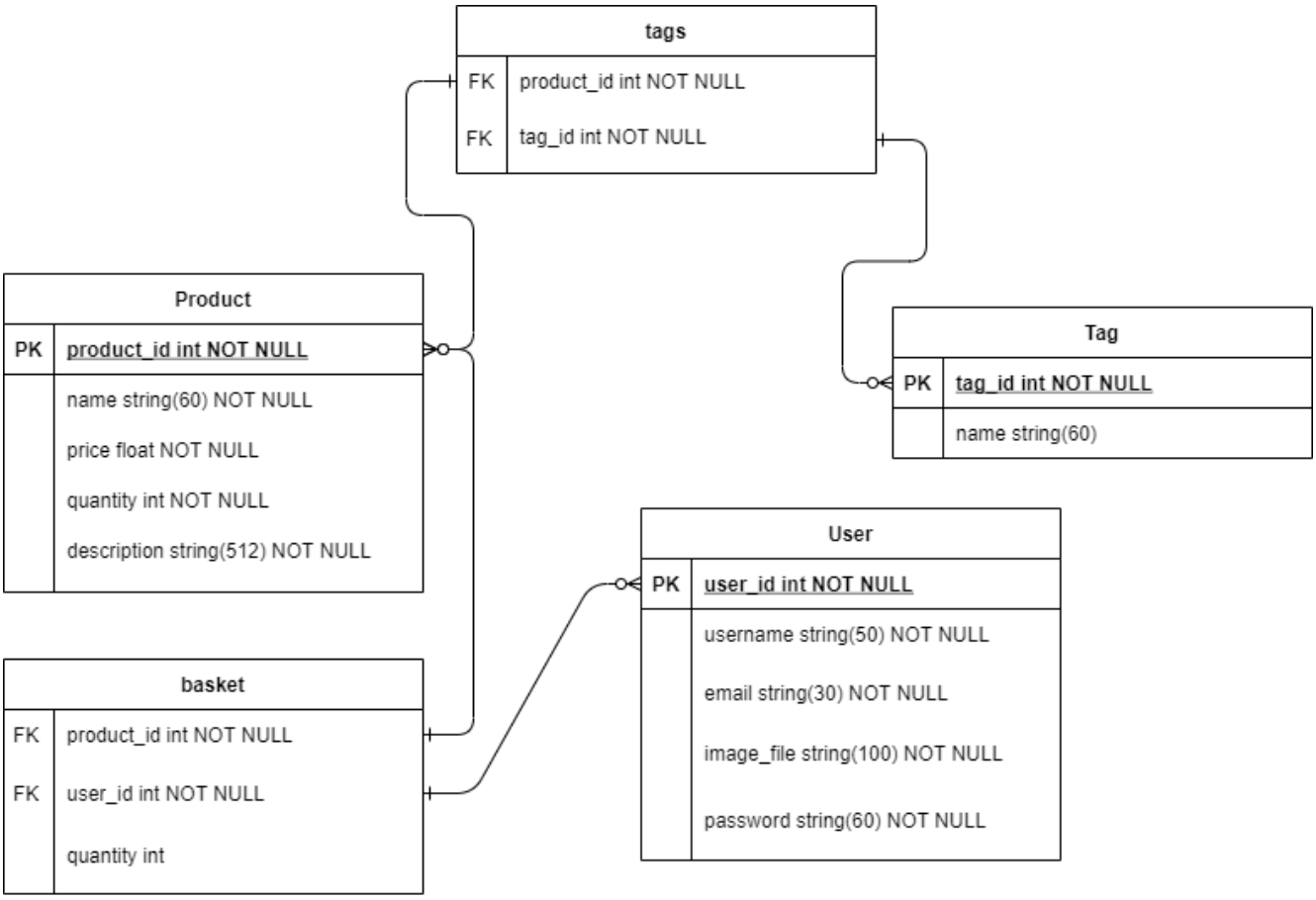

Рисунок 3.1 – Схема бази даних

3.3 Реалізація веб-додатку

Для взаємодії з користувачем використовуємо інтерфейс, який буде реалізовувати вхід команд від користувача до контролеру. Контролер, у свою чергу, буде відправляти чи перенаправляти дані згідно із вводом користувача.

Для показу роботи веб-додатку я обрав декілька прикладів роботи контролеру.

На рисунку 3.2 видно роботу блоку коду контролеру котрий змінює теги. Він генерує сторінку із усіма тегами, а потім приймає команди з інтерфейсу по

додаванню, зміни та видалення тегу із параметром id тегу. Завдяки формі ми далі отримаємо користувальницьке введення та робимо зміни у БД і інтерфейсі.

```python
@app.route('/editTags', methods=['POST', 'GET'])
def editTags():
    tagsList = Tag.query.order_by(Tag.name).all()
    if request.method == 'POST':
        editingRequest=request.form.get('edit')
        deletingRequest=request.form.get('delete')
        addRequest=request.form.get('add')
        if editingRequest:
            print(editingRequest)
            return redirect(url_for('editTag', id=editingRequest))
        elif deletingRequest:
            print(Tag.query.filter_by(id=id))
            Tag.query.filter_by(id=id).delete()
            db.session.commit()
            return redirect(url_for('editTag'))
    return render_template('editTags.html', tags=tagsList)

@app.route('/editTag/<id>', methods=['POST', 'GET'])
def editTag(id):
    tag = Tag.query.filter_by(id=id).first_or_404()
    print(tag.name)
    if request.method == 'POST':
        insertString=request.form.getlist('name')
        print(insertString[0])
        tag.name = insertString[0]
        db.session.commit()
    return render_template('editTag.html', tag=tag)
```

Рисунок 3.2 – Контролер тегу та редагування тегу

Рисунок 3.3 демонструє нам виведення списку усіх тегів, а вже обравши саме який тег нам потрібен по id тегу знаходимо через таблицю багато до багатьох продукти, які зв’язані із цим тегом.

```python
@app.route('/searchByTag', methods=['GET'])
def searchAllTags():
    tagsList = Tag.query.order_by(Tag.name).all()
    print(tagsList)
    return render_template('searchAllTags.html', tags=tagsList)

@app.route('/searchByTag/<id>', methods=['GET'])
def searchByTag(id):
    tag = Tag.query.filter_by(id=id).first_or_404()
    productList = tag.products
    print(productList)
    return render_template('searchByTag.html', tag=tag, products=productList)
```

Рисунок 3.3 – Контролер пошуку за тегом та пошуку тегу

3.4 Створення інтерфейсу користувача

Інтерфейс користувача складається з декількох сторінок. Він відправляє команди на контролер, а той у свою чергу витягує, оновлює, видаляє чи додає потрібні дані з бази даних.

На рисунку 3.4 ми можемо побачити головну сторінку інтернет-магазину. У верхній частині знаходиться «шапка», у котрій ми маємо такі елементи інтерфейсу: Home Page – перехід до головної сторінки, Basket – перехід до кошика, Products – перехід до сторінки з продуктами магазину, Tags – перехід до сторінки з тегами, Your Page – перехід до сторінки вашого профіля.

Також у «тілі» сторінки виводиться декілька продуктів магазину для звернення уваги клієнта, який перший раз заходить до сайту, саме на них.

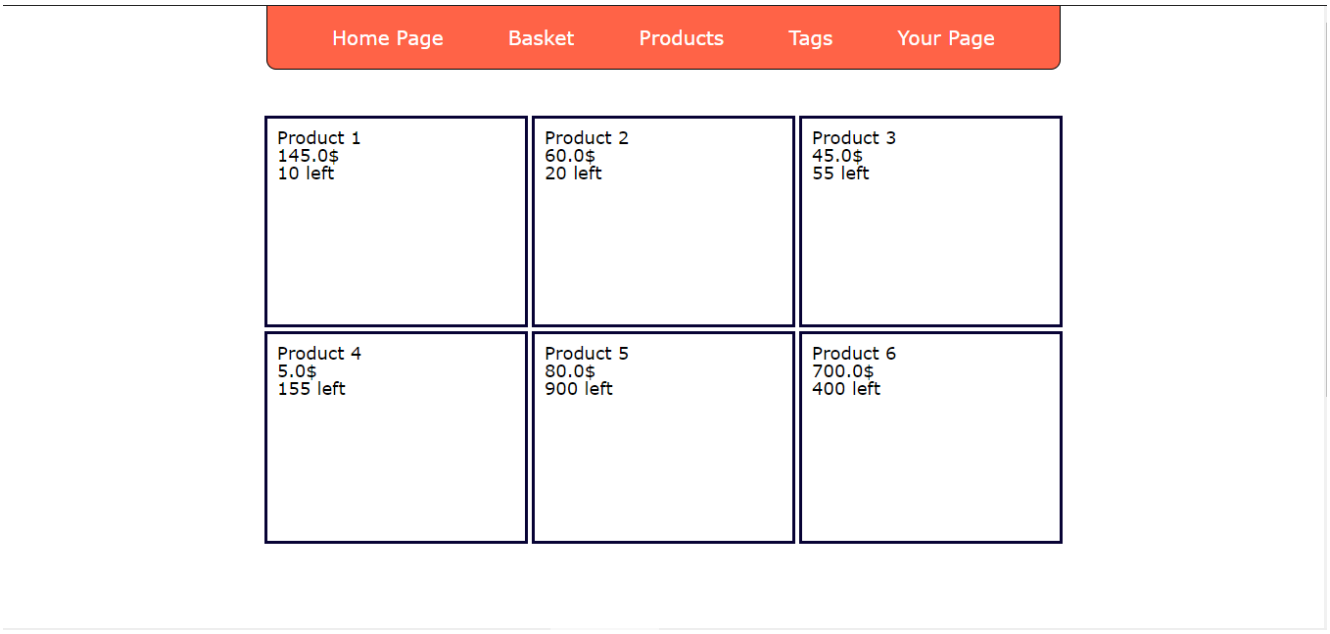

Рисунок 3.4 – Головна сторінка інтернет магазину

На рисунку 3.5 показується механізм пошуку за тегом. Ми обираємо який тег нам потрібен та система виведе на інтерфейсі список продуктів за цим самим тегом.

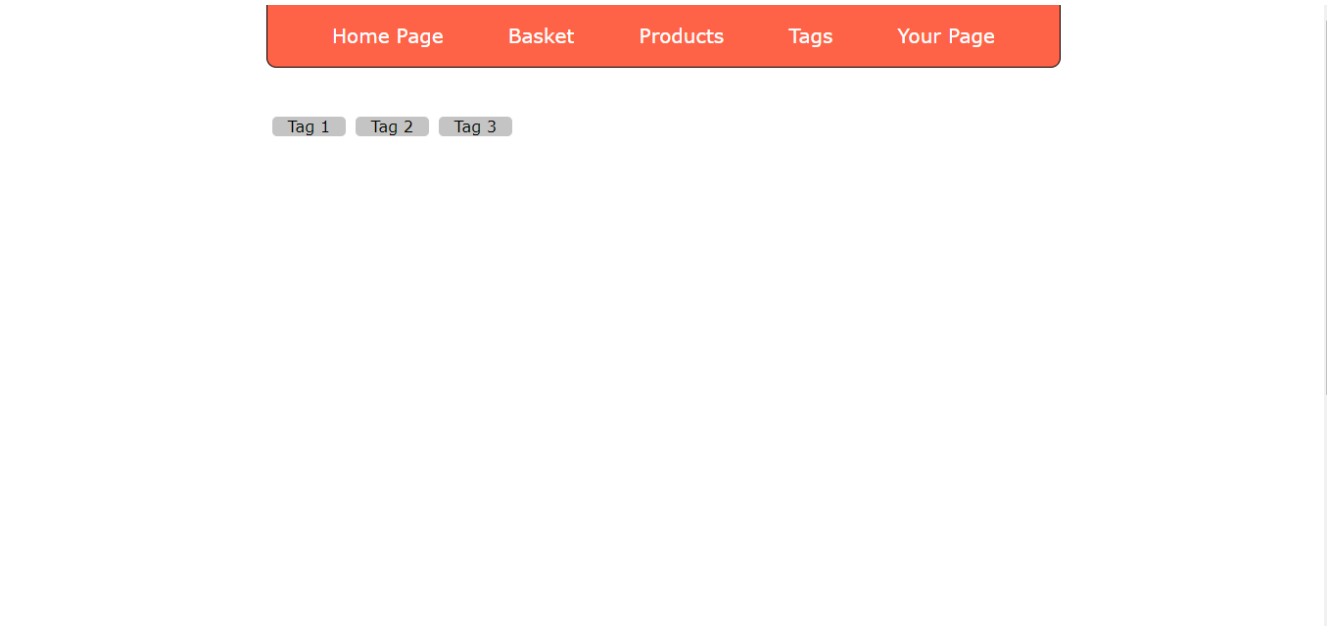

Рисунок 3.5 – Список тегів та пошук за ними

Рисунок 3.6 демонструє нам сторінку продукта. У нашому випадку це продукт з назвою Product 2, із вартістю 60 та кількістю залишків на складу у 20 одиниць. Далі йде опис продукта та кнопки «Додати», «Змінити» та «Видалити», які згідно назві додають продукт до кошика, змінюють продукт чи видаляють його.

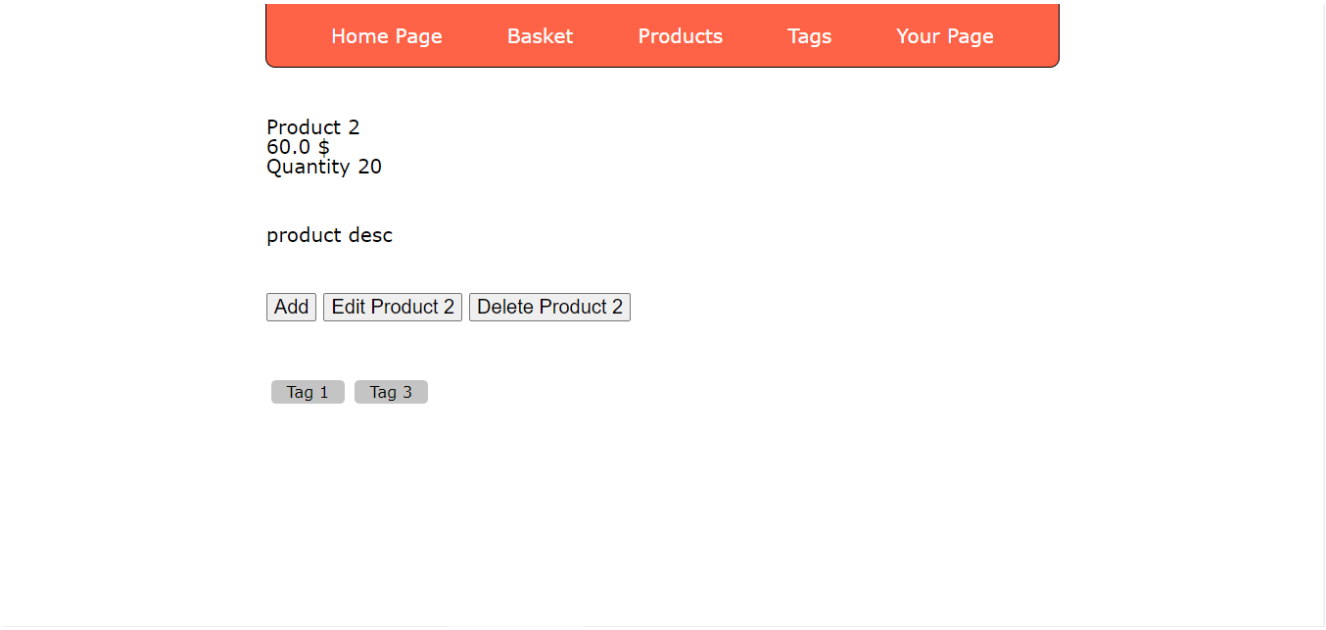

Рисунок 3.6 – Сторінка продукту

ВИСНОВКИ

Результатом даної роботи став функціональний інтернет-магазин на шаблоні MVC.

Інтернет-магазин складається з веб-додатку та бази даних.

Для користувацького інтерфейсу веб-додатку використовується HTML, CSS, JS. Користувацький інтерфейс був розроблений для зручного використання та інформативного показу даних користувачу.

Веб-додаток надає зручні можливості для додавання, оновлення, видалення та зчитування даних, а також реєстрації, авторизації та аутентифікації та контроль сесії. Має робочий механізм зручного пошуку по тегам та систему тегування продуктів інтеренет-магазину. А у кожного користувача є свій кошик, в котрому зберігається усі замовлені товари.

База даних, використовуючи об’єктно-реляційну систему управління PostgreSQL, повністю виконую свої обов’язки в зберіганні та видачі даних.

Результати, отримані під час написання дипломної роботи, відповідають постановленої задачі, тому поставлена мета досягнута.

ПЕРЕЛІК ПОСИЛАНЬ

1. Full Stack Python SQLAlchemy [Електронний ресурс]. — Режим доступу: https://www.fullstackpython.com/sqlalchemy.html
2. Flask SQLAlchemy Basics [Електронний ресурс]. — Режим доступу: https://medium.com/swlh/flask-sqlalchemy-basics-60d4f7f122
3. What is Python? [Електронний ресурс]. — https://pythoninstitute.org/what-is-python/
4. MVC - Model, View, Controller [Електронний ресурс]. — Режим доступу: https://dev.to/temmietope/mvc-model-view-controller-bon
5. Must-Have Features for Ecommerce Sites [Електронний ресурс]. — Режим доступу: https://www.searchenginejournal.com/ecommerce-guide/must-have-website-features/
6. Grinberg M. Flask Web Development. Developing web applications with Python — O'Reilly Media, 2014. — ISBN 9781449372613, ISBN 9781449372620
7. Spring Framework Overview [Електронний ресурс]. — Режим доступу: https://docs.spring.io/spring-framework/docs/current/reference/html/overview.html
8. Руководство по SQLAlchemy в Flask [Електронний ресурс]. — Режим доступу: https://pythonru.com/biblioteki/sqlalchemy-v-flask
9. WTForms Crash Course [Електронний ресурс]. — Режим доступу: https://wtforms.readthedocs.io/en/2.3.x/crash_course/
10. SkipMontanaro. Why is Python a dynamic language and also a strongly typed language - Python Wiki [Електронний ресурс]. — Режим доступу: https://wiki.python.org/
11. Copperwaite M., Leifer C. Learning Flask Framework. Build dynamic, data driven websites and modern web applications with Flask. — Packt Publishing, 2015. ISBN 9781783983360
12. Kalyani Adawadkar. Python Programming - Applications and Future International Journal of Advance Engineering and Research Development. — 2017. — April (iss. SIEICON-2017). — P. 1—4. — ISSN 2348-447