Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» _____ 2024 р

Пояснювальна записка

до кваліфікаційної роботи
бакалавра

на тему: «Комп'ютерна система управління товарообігом магазину»

Захищено на засіданні
Атестаційної комісії № 42
протокол № __ від __.06.2024 р.
Оцінка _____ / _____
Голова Атестаційної комісії
_____ Скоб Ю. О.
(підпис) (прізвище та ініціали)

Виконав:
студент 4 курсу, групи КІ– 41
Галузь знань: 12 – Інформаційні технології
Спеціальність: 121 – «Комп'ютерна інженерія»

Шамрай Едуард Сергійович_____
(прізвище, ім'я та по батькові) (підпис)

Керівник:
Доцент кафедри, кандидат технічних наук

Бикова Тетяна Володимирівна
(прізвище, ім'я та по батькові) (підпис)

Рецензент:
зав.каф. електроніки та управляючих систем, к.ф.-м.наук, доцент
Хруслов Максим Михайлович _____
(підпис)

Харків – 2024

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи бакалавра складається зі вступу, 3 розділів, висновків, списку використаних джерел і 3 додатків. Загальний обсяг роботи складає 74 сторінки, із яких 39 сторінок основної частини з 10 рисунками, 8 найменуваннями списку використаних джерел та 4 додатками.

Метою кваліфікаційної роботи є створення системи товарообігу для магазину.

Об'єкт дослідження – розробка та впровадження сервісу для введення товарообігу магазину.

Предмет дослідження – дослідження є система товарообігу, яка включає в себе всі аспекти і процеси, пов'язані з обігом товарів у магазині.

Проблема, що вивчається в даній кваліфікаційній роботі, полягає в покращенні ефективності і точності системи товарообігу для магазину.

Основна мета що досліджується в даній роботі, полягає в оцінці ефективності та вибору оптимального підходу для реалізації системи товарообміну.

Ця дослідницька робота має значення для осіб які приймають рішення, оскільки надає можливість зробити інформований вибір щодо оптимальної архітектурної стратегії для автоматизації операцій. Ключові аспекти, що досліджуються включають зрозумілий інтерфейс, ефективність та гнучкість додатку, а також інтеграційність.

Ключові слова: інвентарізація, монолітна архітектура, Java, Spring Boot, MVC.

ABSTRACT

The explanatory note to the bachelor's thesis consists of an introduction, 3 chapters, conclusions, a list of references and 3 appendices. The total volume of the work is 74 pages, including 39 pages of the main part with 10 figures, 8 references and 4 appendices.

The purpose of the qualification work is to create a system of goods circulation for the store.

The object of research is the development and implementation of a store inventory system.

Subject of research - the study is a system of goods circulation, which includes all aspects and processes related to the circulation of goods in the store.

The problem studied in this qualification work is to improve the efficiency and accuracy of the inventory system for the store.

The main objective of the research work is to evaluate the effectiveness and select the optimal approach for the implementation of the merchandise circulation system.

This research work is important for decision makers, as it provides an opportunity to make an informed choice about the optimal architectural strategy for automating operations. The key aspects to be investigated include user-friendly interface, application efficiency and flexibility, and integrability.

Keywords: inventory, monolithic architecture, architecture, Java, Spring Boot, MVC.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

Фреймворк (Framework) – Платформа або основа для розробки програмного забезпечення - це структурована система, що містить готові компоненти та інструменти для спрощення процесу розробки. Вона надає набір загальних функцій, які можна використовувати для розробки програм в певній області або для вирішення конкретних завдань.

Java – Об'єктно-орієнтована, мультиплатформна мова програмування, яка була розроблена компанією Sun Microsystems у 1995 році.

API (Application Programming Interface) – це набір правил та протоколів, які визначають спосіб взаємодії між програмами або компонентами програмного забезпечення. Він встановлює набір функцій, методів, структур даних та конвенцій, які використовуються для комунікації між різними програмами або сервісами.

ВСТУП

У сучасному динамічному ринковому середовищі, де швидкість реагування на зміни споживчого попиту та ефективність управління запасами визначають конкурентоспроможність роздрібних мереж, важливість правильного вибору системи товарообміну стає ключовим чинником успіху. Впровадження ефективної системи товарообміну має стратегічне значення, оскільки вона безпосередньо впливає на оперативність постачання, зниження витрат на логістику та оптимізацію товарних запасів.

Ця кваліфікаційна робота фокусується на розробці API системи товарообміну, яка була спроєктована для інтеграції у різноманітні роздрібні системи у вигляді веб-сервісу. Такий підхід дозволяє роздрібним мережам швидко та легко інтегрувати розширені функції управління запасами, оптимізувати логістику та підвищити рівень задоволення клієнтів через автоматизацію взаємодій між постачальниками та точками продажу.

Ця робота має на меті дослідити потенціал запропонованої API, визначити її ключові можливості та обмеження.

Актуальність роботи полягає в створенні модульної API для системи товарообміну, реалізованої з використанням Spring Framework, що дозволяє легко інтегрувати її у веб-додатку в існуючі торгівельні системи. Основна увага в роботі приділена розробці високопродуктивної, гнучкої та легко адаптованої архітектури, яка підтримує широкий спектр операцій товарообміну між різними платформами.

Ця API надає рішення для автоматизації взаємодій між постачальниками та роздрібними точками, знижуючи ручні втручання та покращуючи точність управління запасами. Використання Spring дозволяє оптимізувати ресурсовитрати та забезпечує високу масштабованість та надійність системи. Розробка такої API має значення для розробників, які шукають ефективні шляхи інтеграції сучасних технологічних рішень у комерційні програмні

продукти, а також для керівників проектів, які відповідають за оптимізацію бізнес-процесів у роздрібній торгівлі

Метою дослідження цієї кваліфікаційної роботи є підвищення рівня автоматизації процесів управлінням товарообігом в магазині.

Об'єкт дослідження: процеси управління товарообігом в магазині.

Методи дослідження: включає вивчення принципів роботи систем інвентарізації з фокусом на їх можливість інетеграції з іншими додатками.

Предмет дослідження: методи та алгоритми компютерних систем управління процесами товарообігу в магазині.

Завдання дослідження

1. Аналіз існуючих додатків для інвентаризації та поставок для магазину
2. Розглянути способи реалізації даної системи та методів інтеграції з іншими додатками
3. Розробка та реалізація модулів для автоматизації внутрішнього обліку товарів та обробки онлайн-маніпуляцій з ними
4. Проведення тестування та аналіз отриманих даних для розробки системи автоматизації на основі виявлених результатів

РОЗДІЛ 1
АНАЛІЗ ІСНУЮЧИХ ДОДАТКІВ ДЛЯ ІНВЕНТАРИЗАЦІЇ ТА ПОСТАВОК ДЛЯ МАГАЗИНУ

1.1 Аналіз існуючих додатків для інвентаризації та поставок для магазину

Система товарообміну та інвентарізації магазину — це комплексне програмне забезпечення, яке допомагає управляти запасами продукції в магазині. Воно включає в себе кілька ключових функцій:

- **Контроль запасів:** API дозволяє відслідковувати наявність товарів, забезпечуючи точні дані про кількість і ціну кожного товару.
- **Інвентарізація:** програма допомагає проводити періодичні перевірки товарних запасів, щоб забезпечити їхню відповідність бухгалтерським записам. Це може включати ведення записів про отримання і видачу товарів.
- **Аналітика і звітність:** забезпечує збір і аналіз даних про продажі, що допомагає ухвалювати обґрунтовані рішення щодо закупівлі та управління товарними запасами.

API товарообміну та інвентарізації можуть бути інтегровані з іншими системами управління магазином, такими як касові системи, системи обліку, CRM-системи (управління взаємовідносинами з клієнтами) та інші, забезпечуючи єдину інформаційну базу для ефективного управління магазином.

Основна мета системи товарообміну та інвентарізації магазину полягає в оптимізації управління запасами та підвищенні ефективності роботи магазину. А саме забезпечення достатньої наявності товарів, зниження втрат і викрадень, покращення прийняття рішень.

У цьому підрозділі ми детально розглянемо деякі з ключових проблем:

1. Вибір правильної технології

Вибір правильної технології є критичним етапом у впровадженні автоматизації бізнесу. Вибір неправильної технології може призвести до того, що система не буде відповідати потребам бізнесу або буде недостатньо гнучкою для майбутніх змін. Ось аспекти які слід враховувати:

- **Відповідність потребам бізнесу**: Перш ніж вибрати технологічну платформу, необхідно чітко зрозуміти потреби вашого бізнесу. Це включає розгляд функціональних вимог, масштабування, інтеграції з існуючими системами та здатність до адаптації до майбутніх потреб.
- **Гнучкість і масштабованість**: Технологія повинна бути гнучкою та здатною масштабуватися, щоб вона можна було легко адаптувати до змін в бізнес-середовищі та збільшувати обсяги операцій при необхідності.
- **Безпека даних**: Обов'язково слід звернути увагу на заходи безпеки, які надає обрана технологія. Це включає захист від несанкціонованого доступу, шифрування даних, резервне копіювання та інші заходи забезпечення безпеки.
- **Підтримка та оновлення**: Важливо переконатися, що постачальник технології забезпечує ефективну підтримку та регулярні оновлення програмного забезпечення. Це допоможе забезпечити стабільну роботу системи та вирішення можливих проблем у майбутньому.
- **Вартість володіння**: Оцініка загальної вартістості володіння технології, включаючи витрати на впровадження, підтримку, оновлення та навчання персоналу. Тому важливо обратии рішення, яке максимально відповідає бюджету та вимогам.
- **Врахування майбутніх потреб**: При виборі технології слід розглядати не лише поточні потреби бізнесу, але і майбутні прогнози розвитку та змін у галузі. Оберіть технологію, яка буде здатна ефективно працювати із майбутніми викликами та потребами вашого бізнесу.

2. Інтеграція з існуючими системама

Під час вибору технології для автоматизації бізнесу слід детально розглянути питання щодо інтеграції з існуючими системами. Це означає:

- **Аналіз сумісності даних**: Перш ніж обирати нову систему, потрібно переконатися, що вона здатна ефективно взаємодіяти з існуючими базами даних і форматами даних бізнесу. Це допоможе уникнути проблем зі сумісністю та перенесенням даних.
- **Стандарти і протоколи**: Важливо врахувати, що нова система повинна підтримувати стандартні протоколи обміну даними, такі як API (Application Programming Interface), щоб забезпечити легку інтеграцію з існуючими програмами і сервісами.
- **План міграції даних**: Перед впровадженням нової системи необхідно розробити план міграції даних, який включає в себе перенесення існуючих даних у нову систему без втрат і збоїв. Це може включати імпорт даних, забезпечення їхньої цілісності та перевірку на коректність.
- **Оцінка і ризик-менеджмент**: Перед вибором технології важливо оцінити можливі ризики, пов'язані з інтеграцією нової системи з існуючими. Це допоможе прийняти розумні рішення та розробити стратегії мінімізації ризиків.
- **Документація і підтримка**: Після успішного впровадження нової системи важливо забезпечити належну документацію і підтримку для подальшої роботи з нею. Це включає навчання персоналу, розробку посібників користувача та надання технічної підтримки.

Ретельне врахування цих аспектів допоможе забезпечити успішну інтеграцію нової системи з існуючими системами та забезпечити ефективну роботу бізнесу.

3. Безпека даних

Цей аспект має ключове значення при виборі технології. Ось причини чому безпека грає велику роль при введеню бізнесу:

1. **Стабільність та продуктивність системи**: Регулярні оновлення програмного забезпечення дозволяють підтримувати стабільність та продуктивність системи. Вони включають виправлення помилок, покращення функціональності та оптимізацію продуктивності, що сприяє безперебійній роботі бізнесу.
2. **Захист від кібератак**: Оновлення програмного забезпечення також важливо для забезпечення безпеки даних та захисту від кібератак. Через те, що кіберзлочинці постійно шукають нові способи вторгнення, актуальні версії програм дозволяють уникнути вразливостей та зберегти конфіденційність інформації.
3. **Сумісність зі стандартами та вимогами**: Регулярні оновлення дозволяють підтримувати відповідність програмного забезпечення зі стандартами та вимогами, які часом змінюються. Це важливо для підтримки сумісності з іншими системами та виконання вимог законодавства.
4. **Підтримка нових функцій та можливостей**: Оновлення дозволяють отримувати доступ до нових функцій та можливостей, які поліпшують ефективність та конкурентоспроможність бізнесу. Це може включати нові інтеграції, інструменти аналітики, розширені можливості звітності та інші.
5. **Ефективне вирішення проблем**: Підтримка від виробника дозволяє ефективно вирішувати проблеми та отримувати допомогу у випадку виникнення неполадок. Це допомагає мінімізувати перерви в роботі бізнесу та забезпечує швидке відновлення роботи системи.

У цілому, підтримка та регулярні оновлення технології важливі для забезпечення безперебійної роботи системи, захисту даних та забезпечення відповідності вимогам та стандартам.

4. Опір співробітників

Опір співробітників під час впровадження автоматизації бізнесу може виникнути з різних причин і потребує уважного управління. Давайте розглянемо це більш детально:

- **Страх перед змінами**: Багато співробітників можуть боятися змін, пов'язаних із впровадженням нових технологій. Вони можуть перейматися тим, що автоматизація може вплинути на їхні робочі обов'язки або навіть призвести до втрати робочих місць.
- **Недовіра до технології**: Деякі співробітники можуть виявити опір через недовіру до нових технологій. Вони можуть сумніватися у здатності системи ефективно виконувати їхні завдання або боятися, що автоматизація призведе до помилок або непередбачуваних наслідків.
- **Неадекватна комунікація та навчання**: Недостатня комунікація зі сторони керівництва щодо мети та переваг автоматизації, а також недостатня підготовка та навчання персоналу можуть призвести до опору. Співробітники можуть відчувати себе збитими або непідготовленими до нових технологій.
- **Страх перед втратою робочих місць**: Один з основних страхів серед співробітників - це страх перед втратою робочих місць через автоматизацію. Вони можуть відчувати загрозу для своєї кар'єри або не впевнені, що зможуть адаптуватися до нової реальності.
- **Спосіб впровадження змін**: Якщо зміни введені неправильно або без належної підтримки, це може призвести до опору серед персоналу. Важливо залучати співробітників до процесу впровадження, вислуховувати їхні побої та пропозиції та надавати достатню підтримку під час переходу.

Для подолання опору співробітників важливо встановити відкритий та довірливий діалог з персоналом, пояснити переваги автоматизації для бізнесу

та їхньої власної роботи, надати достатню підтримку та навчання, а також включити їх у процес впровадження змін.

5. Фінансові витрати

Розгляд фінансових витрат у контексті впровадження автоматизації бізнесу вимагає уважного аналізу та детального розгляду різних аспектів. Ось ключові пункти для розгляду:

- **Витрати на програмне забезпечення та обладнання**: При впровадженні автоматизованих рішень можуть виникати значні витрати на закупівлю програмного забезпечення та необхідного обладнання. Це включає вартість ліцензій на програмне забезпечення, закупівлю серверів, комп'ютерів, сканерів, принтерів тощо.
- **Витрати на впровадження та налаштування системи**: Важливо врахувати витрати на впровадження та налаштування автоматизованої системи. Це може включати оплату консультантів з впровадження, навчання персоналу, налаштування програмного забезпечення під потреби бізнесу та інші пов'язані витрати.
- **Оплата підтримки та обслуговування**: Після впровадження необхідно враховувати витрати на підтримку та обслуговування автоматизованої системи. Це може включати оплату технічної підтримки, оновлення програмного забезпечення, ремонт та обслуговування обладнання та інші пов'язані витрати.
- **Втрати у зв'язку з перервами в роботі**: Необхідно врахувати можливі втрати від перерв у роботі бізнесу під час впровадження та налаштування автоматизованих систем. Це може включати втрати від призупинення виробничих процесів, втрату продажів або клієнтів тощо.
- **Планування бюджету на майбутнє**: При розгляді фінансових витрат важливо також планувати бюджет на майбутнє. Наприклад, необхідно врахувати витрати на розширення або модернізацію системи в

майбутньому, а також витрати на навчання персоналу та підтримку під час подальшого функціонування системи.

Ретельне розгляд фінансових аспектів допоможе підготувати ефективний бюджет на впровадження автоматизації бізнесу та мінімізувати ризики непередбачених витрат.

6. Підтримка і обслуговування

Ретельне планування та управління підтримкою та обслуговуванням автоматизованих систем є важливим аспектом успішного впровадження. Розглянемо цей пункт більш детально:

- **Технічна підтримка**: Забезпечення належної технічної підтримки включає у себе надання консультаційних послуг, відповіді на запитання користувачів та вирішення технічних проблем, що виникають під час роботи з системою.
- **Оновлення програмного забезпечення**: Регулярні оновлення програмного забезпечення дозволяють підтримувати систему у актуальному стані, виправляти помилки, покращувати функціональність та забезпечувати безпеку даних.
- **Навчання персоналу**: Проведення навчання персоналу щодо користування та ефективного використання автоматизованих систем є важливим елементом підтримки. Це допомагає максимізувати користь від використання системи та підвищує продуктивність праці співробітників.
- **Моніторинг та аналіз продуктивності**: Постійний моніторинг роботи системи та аналіз її продуктивності дозволяють вчасно виявляти можливі проблеми та вдосконалювати роботу системи з метою досягнення оптимальних результатів.

Забезпечення ефективної підтримки та обслуговування допомагає забезпечити стабільну та безперебійну роботу автоматизованих систем, що є важливим для успішної діяльності підприємства.

1.2 Розгляд існуючий додатків

Щоб провести аналіз додатків для інвентаризації та поставок у магазині, спочатку варто з'ясувати, що саме маємо на увазі під "існуючими додатками". Це можуть бути програми для управління запасами, системи обліку товарів, засоби для замовлення і поставок, інструменти для відстеження витрат та інші рішення, спрямовані на оптимізацію процесів у магазині.

Ось кроки, які були виконаны під час аналізу існуючих додатків для інвентаризації та поставок:

1. **Визначення потреб**: Спочатку важливо визначити, які саме завдання ви хочете вирішити за допомогою додатків. Це може бути відстеження запасів, автоматизація процесу замовлення та поставок, аналіз даних про продажі для прогнозування попиту тощо.
2. **Пошук додатків**: Після того як визначили потреби, треба провести пошук додатків, які можуть їх задовольнити. Це може бути через інтернет, додатки в магазинах програмного забезпечення або рекомендації від колег у галузі.
3. **Оцінка функціональності**: При оцінці додатків слід звернути увагу на їхні можливості та функціональність. Порівняйти їх з потребами і перевірити, чи можуть вони задовольнити вимоги.
4. **Відгуки та рейтинги**: Перед тим як приймати рішення, треба оцінити відгуки користувачів та рейтинги додатків. Це допоможе отримати уявлення про те, наскільки ефективно додаток працює в реальних умовах.
5. **Вартість і підтримка**: Враховати вартість додатка та можливості підтримки. Переконайтись, що обраний додаток вписується в бюджет, а також доступна якісна підтримка для вирішення будь-яких проблем чи запитань.

Таблиця 1.1

Порівняння існуючих додатків

Категорія	**TradeGecko**	**Zoho Inventory**	
Визначення потреб	Управління запасами та поставками.	Керування запасами та поставками.	Управління запасами і аналізування продажів товарів
Пошук додатків	Через веб-сайт або через магазини додатків.	Через веб-сайт або через магазини додатків.	Через веб-сайт
Оцінка функціональності	Широкий набір функцій, включаючи відстеження запасів в реальному часі, автоматизацію замовлень, аналітику продажів.	Широкий набір функцій, включаючи відстеження запасів в реальному часі, автоматизацію замовлень, аналітику продажів.	Широкий набір функцій, включаючи відстеження запасів в реальному часі, аналітику продажів.
Мінуси	Висока вартість деяких планів, складність інтерфейсу.	Складність інтеграції з іншими програмами Zoho.	Відсутність на даних момент автоматизації замовлень
Відгуки та рейтинги	Переважно позитивні, але є скарги на клієнтську підтримку та швидкість відгуку.	Змішані, з скаргами на складність інтеграції з іншими програмами Zoho.	-

Вартість і підтримка	Залежить від обраного плану, від доступних до дорогих корпоративних планів. Підтримка через тікет-систему.	Залежить від обраного плану, від безкоштовного до платних планів.	Наразі безкоштовний

1.2 Аналіз принципів роботи додатків системи товарообігом

1. Управління запасами

Управління запасами є критично важливим аспектом ефективного товарообігу в будь-якому бізнесі. Це включає в себе комплексні підходи та функціонал, спрямовані на ефективне управління рівнями товарних запасів, оптимізацію процесів поставок і замовлень, а також мінімізацію ризиків нестачі або перепродажу товарів.

Основний принцип роботи систем управління запасами полягає в систематичному відстеженні кількості товарів на складі та їхнього руху в межах бізнес-процесів. Це включає ведення детальних записів про прибуття та вибуття товарів.

Управління запасами також вимагає категоризації товарів та класифікації їх за різними критеріями. Це допомагає краще організувати і керувати асортиментом товарів, забезпечуючи ефективне використання складського простору та оптимізацію запасів.

Ці принципи роботи систем управління запасами допомагають бізнесам ефективно управляти своїми запасами, забезпечуючи плавний хід операцій та максимізуючи ефективність використання ресурсів.

2. Облік продажів та звітність

Облік продажів та звітність є важливими компонентами системи товарообігу, які допомагають бізнесам вести облік продажів, визначати

прибуток, аналізувати витрати та генерувати різноманітні звіти для прийняття рішень.

Основний принцип роботи обліку продажів полягає в систематичному відстеженні всіх транзакцій з продажу товарів або послуг. Це включає реєстрацію кожної угоди про продаж, включаючи кількість товарів, ціну тощо.

Одним з важливих аспектів обліку продажів є здатність генерувати різноманітні звіти та аналітику. Системи надають інструменти для складання звітів за різними параметрами, такими як період, тип товарів, клієнти тощо, що дозволяє отримувати детальну інформацію про різні аспекти бізнесу.

Ці принципи роботи обліку продажів та звітності допомагають бізнесам ефективно вести облік своїх торговельних операцій, аналізувати їх ефективність та приймати обгрунтовані рішення для подальшого розвитку.

3. Інтеграція з іншими системами

Інтеграція з іншими системами є важливим аспектом для систем управління товарообігом, оскільки вона дозволяє забезпечити безперервність бізнес-процесів та оптимізувати робочі процеси.

Одним з ключових принципів роботи систем інтеграції є забезпечення сумісності та взаємодії з різними програмними продуктами та сервісами. Це включає в себе можливість обміну даними між системами, автоматичне оновлення інформації та забезпечення її актуальності.

Інтеграція з іншими системами також передбачає розробку спеціальних інтерфейсів та протоколів обміну даними, що дозволяють різним системам ефективно співпрацювати між собою. Це може включати стандартизовані формати даних, API і технології веб-сервісів.

Інтеграція з іншими системами є важливим елементом ефективного управління товарообігом, оскільки вона дозволяє бізнесам оптимізувати процеси та забезпечувати безперервність бізнес-процесів.

1.3 Ознайомлення з системою продуктових магазинів

1. Роль системи продуктових магазинів

Роль системи продуктових магазинів полягає в автоматизації та оптимізації різних аспектів управління продажами та запасами в магазинах роздрібної торгівлі. Ця система виконує кілька важливих функцій, спрямованих на полегшення роботи персоналу та покращення обслуговування клієнтів.

Одним з важливих аспектів ролі системи продуктових магазинів є забезпечення точності та надійності даних. Вона використовує централізовану базу даних для управління інформацією про товари, клієнтів та транзакції, що дозволяє забезпечити актуальність та доступність інформації в будь-який момент часу.

Загалом, система продуктових магазинів відіграє ключову роль у забезпеченні ефективного функціонування магазинів роздрібної торгівлі. Вона допомагає оптимізувати робочі процеси, підвищує ефективність обслуговування клієнтів та сприяє успішному розвитку бізнесу.

2. Функціональність системи

Функціональність системи продуктових магазинів включає широкий спектр інструментів та можливостей, які спрямовані на автоматизацію та оптимізацію різних аспектів управління продажами та запасами.

Однією з ключових функцій системи є ведення каталогу товарів. Вона дозволяє додавати, оновлювати та керувати інформацією про товари, включаючи назву, опис, ціну, наявність та інші характеристики.

Крім того, система продуктових магазинів надає можливості для обліку продажів. Вона автоматично реєструє транзакції з продажу товарів, включаючи кількість проданих одиниць, ціну та інші важливі дані.

Додатково, система продуктових магазинів може включати функції управління клієнтською базою даних, створення звітності та аналітики, а також інші інструменти, спрямовані на підвищення ефективності та конкурентоспроможності бізнесу.

3. Ключові переваги

Однією з ключових переваг системи продуктових магазинів є покращення ефективності робочих процесів. Вона допомагає автоматизувати багато рутинних операцій, що дозволяє персоналу магазину зосередитися на більш важливих завданнях, таких як обслуговування клієнтів та вдосконалення сервісу.

Іншою важливою перевагою є можливість аналізу та відстеження результатів. Система надає можливість створювати різноманітні звіти та аналітику, які дозволяють аналізувати продажі тим самим допомогаючи прогнозувати майбутній розвиток бізнесу.

Загалом, система продуктових магазинів допомагає підвищити ефективність та конкурентоспроможність бізнесу, покращити обслуговування клієнтів та забезпечити успішний розвиток компанії.

4. Можливості розширення та налаштування

Однією з можливостей розширення системи продуктових магазинів є додавання додаткових модулів та функцій. Багато постачальників програмного забезпечення надають різноманітні додаткові модулі, які можуть бути легко інтегровані з основною системою та розширити її функціональність.

Ще однією можливістю є інтеграція з іншими системами. Система продуктових магазинів може бути легко інтегрована з іншими програмними продуктами та сервісами, такими як бухгалтерські програми, системи управління відносинами з клієнтами (CRM), системи управління запасами (ERP) та інші, що розширює можливості її використання та забезпечує більшу інтеграцію між різними аспектами бізнесу.

Загалом, можливості розширення та налаштування системи продуктових магазинів дозволяють бізнесам адаптувати її під свої конкретні потреби, розширювати її функціональність та забезпечувати більшу ефективність управління та розвитку.

Висновок за розділом 1

У цьому розділі проведено всебічний аналіз існуючих додатків для інвентаризації та поставок у продуктових магазинах. Було досліджено принципи роботи різних систем товарообігу та здійснено детальне ознайомлення з функціонуванням систем продуктових магазинів.

Під час аналізу додатків для інвентаризації та поставок встановлено, що сучасні рішення значно підвищують ефективність управління запасами та оптимізують процеси постачання. Розглянуті програми пропонують різноманітні функціональні можливості, такі як автоматичне оновлення даних про запаси, відстеження товарів у режимі реального часу та інтеграція з іншими системами управління підприємством.

Аналіз принципів роботи систем товарообігу показав, що ключовими аспектами ефективного функціонування є точність даних, швидкість обробки інформації та можливість налаштування під специфічні потреби магазину. Крім того, важливою є здатність систем адаптуватися до змін у попиті та пропозиції, що дозволяє знижувати втрати та збільшувати прибутковість.

Ознайомлення з системами продуктових магазинів виявило низку загальних тенденцій та практик, які впливають на ефективність їх роботи. Основна увага приділяється автоматизації процесів, інтеграції з постачальниками та забезпеченню високого рівня обслуговування клієнтів.

Таким чином, проведений аналіз свідчить про те, що впровадження сучасних додатків для інвентаризації та поставок може суттєво покращити ефективність управління продуктовими магазинами. Використання передових технологій та інтеграція різних систем дозволяють досягти високого рівня автоматизації, знизити витрати та підвищити задоволеність клієнтів.

РОЗДІЛ 2
ВИБІР ПЛАТФОРМИ ДЛЯ РОЗРОБКИ ДОДАТКУ

2.1 Вибір платформи для розробки додатку

1. Java

Java - це високорівнева, об'єктно-орієнтована мова програмування, яка була розроблена в компанії Sun Microsystems у 1995 році. Вона відома своєю крос-платформеністю, що означає, що програми, написані на Java, можуть запускатися на різних операційних системах без потреби переписування коду.

Переваги використання Java

По-перше, Java є досить простою для вивчення мовою програмування, особливо для тих, хто вже має деякий досвід у програмуванні. Вона має чітку синтаксичну структуру та велику кількість інструментів та ресурсів для вивчення.

Крім того, Java відома своєю високою надійністю та безпекою. Вона має вбудовані механізми безпеки, такі як контроль доступу та обмеження ресурсів, які дозволяють попереджувати багато типових помилок та атак.

Загалом, Java є потужною та універсальною мовою програмування, яка надає розробникам широкі можливості для створення різноманітних додатків. Її простота вивчення, надійність та крос-платформеність роблять її однією з найпопулярніших мов програмування в світі.

2. Spring boot

Spring Boot - це фреймворк для розробки веб-додатків на мові програмування Java. Він надає широкий набір інструментів та бібліотек для створення високоякісних, масштабованих та ефективних веб-додатків. Однією з ключових особливостей Spring Boot є його фокус на спрощенні розробки та зменшенні кількості необхідного коду.

Переваги використання Spring Boot

По-перше, він надає структуру та конвенції для швидкого старту проекту, що дозволяє розробникам швидко почати роботу над своїм додатком. Крім того, Spring Boot автоматизує багато стандартних процесів, таких як конфігурація, обробка винятків та управління залежностями, що дозволяє розробникам зосередитися на бізнес-логіці свого додатку.

Ще однією важливою перевагою Spring Boot є його модульність та гнучкість. Він дозволяє розробникам легко інтегрувати різноманітні фреймворки та бібліотеки для вирішення конкретних завдань, що робить його ідеальним вибором для різних типів проектів. Крім того, Spring Boot підтримує різні підходи до розробки, такі як MVC (Model-View-Controller), RESTful API та багато інших, що дозволяє розробникам вибрати найбільш підходящий для них підхід.

Іншою важливою перевагою Spring Boot є його активна спільнота та велика кількість документації та підтримки. Розробники можуть легко знайти відповіді на свої питання, а також взяти участь у різних форумах, дискусіях та спільнотних заходах, що дозволяє швидко розв'язувати проблеми та розвиватися як спеціалісти.

Загалом, Spring Boot є потужним та ефективним фреймворком для розробки веб-додатків на Java, який надає розробникам широкі можливості для створення високоякісних та масштабованих додатків. Його простота використання, гнучкість та підтримка спільноти роблять його одним з найпопулярніших виборів серед розробників.

3. Thymeleaf

Thymeleaf - це шаблонний движок для створення веб-сторінок, який використовується головним чином у розробці веб-додатків на Java. Однією з особливостей Thymeleaf є його можливість інтегруватися безпосередньо в HTML-код сторінок, що робить їх більш читабельними та зручними для розробників.

Переваги використання Thymeleaf

Однією з переваг використання Thymeleaf є його простота вивчення та використання. Синтаксис Thymeleaf є досить інтуїтивно зрозумілим і базується на звичайних HTML-тегах, що дозволяє розробникам швидко освоїти його та почати використовувати для створення динамічних веб-сторінок.

Ще однією важливою перевагою Thymeleaf є його інтеграція з різноманітними фреймворками та бібліотеками Java, зокрема з Spring Framework. Це дозволяє розробникам легко інтегрувати Thymeleaf у свої проекти та використовувати його разом із іншими інструментами для створення високоякісних веб-додатків.

Загалом, Thymeleaf є потужним та зручним інструментом для створення динамічних веб-сторінок на Java. Його простота використання, широкі можливості та інтеграція з іншими технологіями роблять його одним з популярних виборів серед розробників веб-додатків.

4. PostgreSQL

PostgreSQL - це потужна об'єктно-реляційна система управління базами даних (СУБД), яка використовується для зберігання та організації даних у веб-додатках та інших програмних продуктах.

Переваги використання PostgreSQL

Однією з переваг PostgreSQL є його висока надійність та стабільність. Він має вбудовані механізми забезпечення цілісності даних, транзакцій та відновлення після збоїв, що робить його ідеальним вибором для критичних застосунків, які вимагають стабільної роботи.

Крім того, PostgreSQL має широкий набір функціональності та можливостей. Він підтримує різні типи даних, включаючи числові, рядкові, дати, географічні та багато інших. Він також має багато розширень та доповнень, які дозволяють розширювати його функціональність відповідно до конкретних потреб проекту.

Загалом, PostgreSQL є потужною та надійною системою управління базами даних, яка надає розробникам широкий набір можливостей та функціональності для роботи з даними. Його стабільність, розширюваність та відкритість роблять його одним з популярних виборів для розробки різноманітних програмних продуктів.

2.2 Побудова архітектури додатку

При розробці додатку для інвентаризації та поставок у магазині, було обрано архітектурний патерн Model-View-Controller (MVC). Основні мотивації за вибір цього підходу полягають у моїй недостатній експертизі в області фронтенду та бажанні швидко розробити функціональний та надійний додаток. Використання MVC дозволить мені чітко розділити логіку додатку на компоненти, що спростило розробку та підтримку.

Технології

Для реалізації контролерів та бізнес-логіки додатку я обрав Spring Boot, який забезпечить мені швидкий старт та просте конфігурування проекту. Використання Spring Boot дозволить швидко розгорнути сервер та зосередитися на реалізації функціональності додатку, не витрачаючи багато часу на налаштування середовища.

Для шаблонізації веб-сторінок та відображення даних користувачам було обрано Thymeleaf. Через можливість Thymeleaf швидко створювати динамічні HTML-сторінки без необхідності в глибоких знаннях JavaScript та інших фронтенд-технологій.

База даних

Для зберігання даних про товари, категорії, замовлення постачання та іншу інформацію було обрано PostgreSQL. Використання PostgreSQL дозволить забезпечити ефективне зберігання та організацію даних, а також забезпечить високу швидкість операцій з базою даних.

Загалом, вибір Spring Boot, Thymeleaf та PostgreSQL дозволить швидко розробити функціональний та надійний додаток для інвентаризації та поставок

у магазині, зосереджуючись на реалізації бізнес-логіки та задоволенні потреб користувачів.

5. Model-View-Controller (MVC)

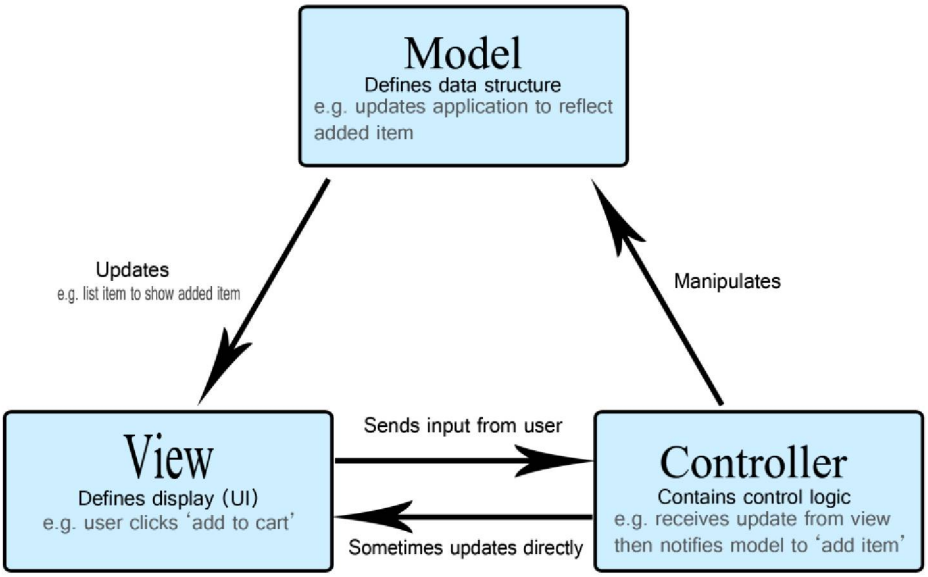


Рис 2.1– Схема MVC

Модель (Model)

Управляє даними та бізнес-логікою. Вона складається з POJO-класів, що представляють сутності додатку, такі як товари, категорії товарів, замовлення. Ці класи визначають поля та методи для роботи з даними та виконують валідацію.

Вид (View)

Відповідає за відображення даних користувачам. У цьому додатку використовуються HTML-шаблони та Thymeleaf для генерації сторінок, які відображають дані з моделі.

Контролер (Controller)

Обробляє HTTP-запити та забезпечує взаємодію між моделлю та представленням. Він містить методи для обробки різних типів запитів (GET, POST, PUT) та викликає відповідні сервіси або репозиторії для отримання даних та обробки бізнес-логіки. Це дозволяє відокремити бізнес-логіку від представлення, забезпечуючи чистоту та організованість коду.

Загальний до патерну MVC полягає в тому, що модель, представлення та контролери мають бути розділені та незалежні. Це сприяє перевикористовуванню коду та розширюваності додатку, оскільки кожен компонент може бути змінений або замінений без впливу на інші частини системи.

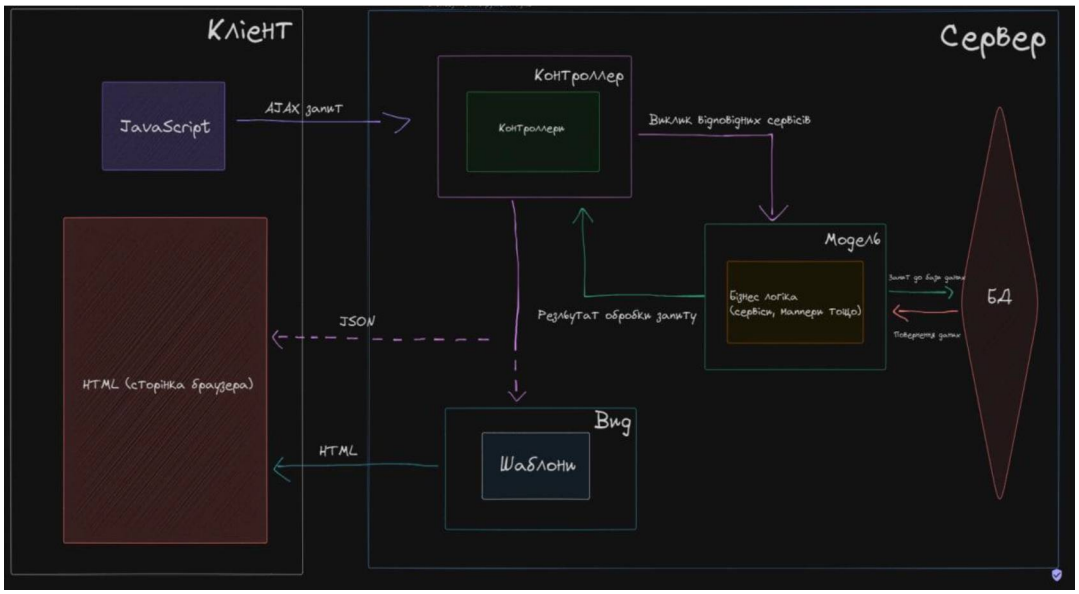


Рис 2.2– Схема системи товарообміну

6. База даних

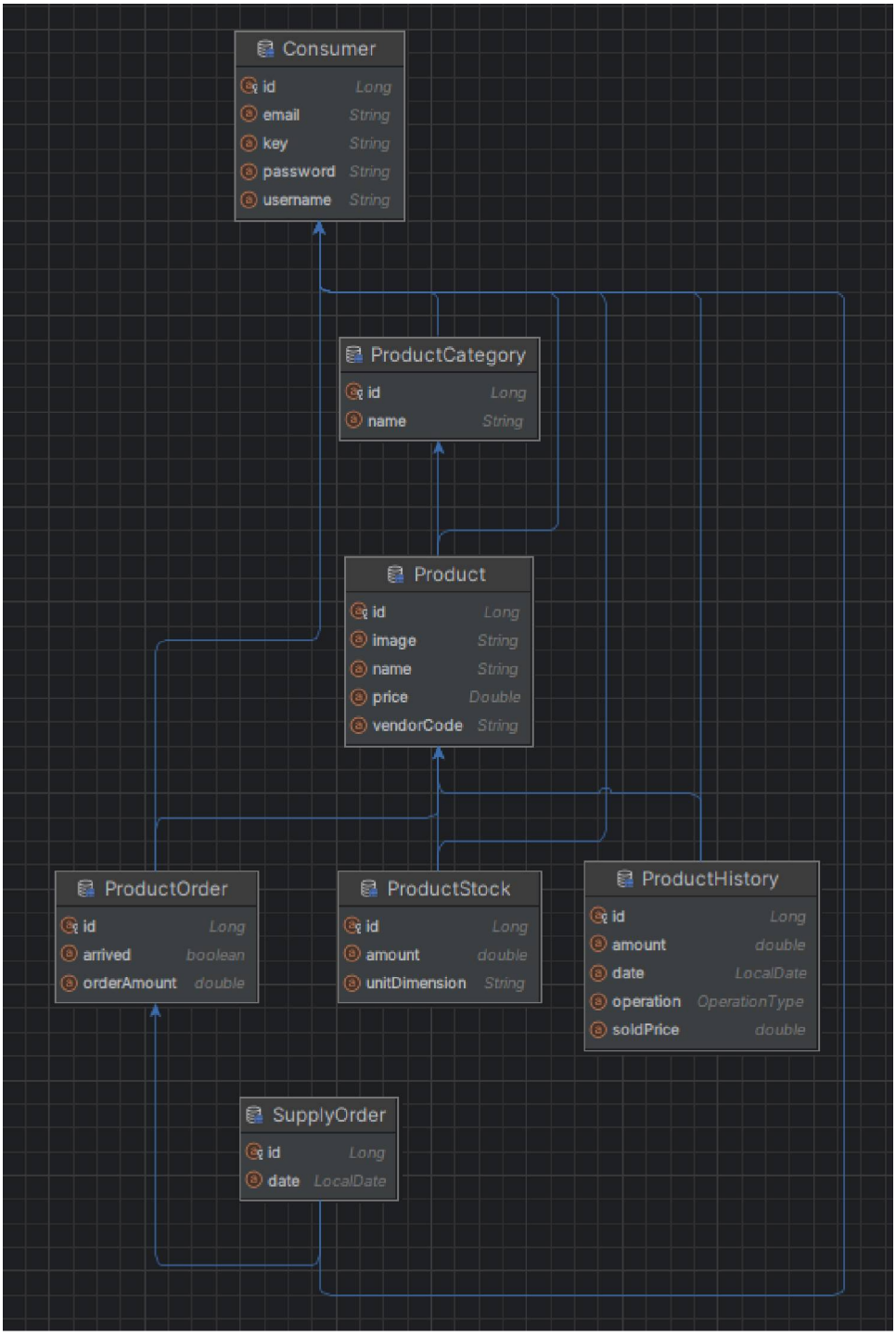


Рис 2.3– Схема бази даних

Детальний опис бази даних та зв'язків між сутностями

1. **Consumer:**
   - **id:** Унікальний ідентифікатор споживача.
   - **email:** Електронна пошта споживача.

- **key:** Укнікальний ключ, який використовується для ідентифікації товарів конкретного споживача.
- **password:** Пароль споживача.
- **username:** Ім'я користувача.
- Зв'язок з ProductCategory через id, що дозволяє співвідносити категорію продуктів з певним споживачем.

2. **ProductCategory:**
    - **id:** Унікальний ідентифікатор категорії продукту.
    - **name:** Назва категорії.
    - Зв'язок з Product через id, що дозволяє класифікувати продукти за категоріями.
3. **Product:**
    - **id:** Унікальний ідентифікатор продукту.
    - **image:** Зображення продукту.
    - **name:** Назва продукту.
    - **price:** Ціна продукту.
    - **vendorCode:** Код постачальника.
    - Зв'язок з ProductStock та ProductHistory через id для відстеження запасів та історії продажів/змін.
4. **ProductOrder:**
    - **id:** Унікальний ідентифікатор замовлення продукту.
    - **arrived:** Чи прибув продукт.
    - **orderAmount:** Кількість замовленого продукту.
    - Пов'язаний з Product через id для відстеження, які продукти були замовлені.
5. **ProductStock:**
    - **id:** Унікальний ідентифікатор запасів продукту.
    - **amount:** Кількість на складі.
    - **unitDimension:** Розмірність одиниці продукту.

- Пов'язаний з Product через id для відстеження запасів конкретних товарів.

6. **ProductHistory:**
   - **id:** Унікальний ідентифікатор історії продукту.
   - **amount:** Кількість продукту в операції.
   - **date:** Дата операції.
   - **operation:** Тип операції (покупка, продаж тощо).
   - **soldPrice:** Ціна продажу.
   - Зв'язок з Product через id для відстеження історії змін кожного продукту.
7. **SupplyOrder:**
   - **id:** Унікальний ідентифікатор замовлення постачання.
   - **date:** Дата замовлення.
   - Зв'язок з ProductOrder через id для координації постачань та замовлень.

Висновок за розділом 2

У цьому розділі здійснено вибір платформи для розробки додатку та побудову його архітектури. На основі проведеного аналізу було обрано наступні технології та інструменти: Java, Spring Boot, Thymeleaf та PostgreSQL. Архітектурний патерн, який було вирішено використати для даного додатку, – Model-View-Controller (MVC).

1. **Java:** Ця мова програмування була обрана за її широке розповсюдження, надійність та потужні можливості для розробки корпоративних додатків. Java також забезпечує високу продуктивність і масштабованість, що є важливими критеріями для розробки додатку для інвентаризації та поставок.
2. **Spring Boot:** Ця фреймворк для Java спрощує створення самостійних, готових до роботи додатків. Spring Boot надає можливість швидко

налаштувати і запустити додаток завдяки вбудованій підтримці багатьох корисних функцій та автоматичній конфігурації.

3. **Thymeleaf:** Як шаблонізатор, Thymeleaf дозволяє створювати динамічні веб-сторінки, які легко підтримувати та розширювати. Його інтеграція зі Spring Boot робить його ідеальним вибором для розробки представницького шару додатку.
4. **PostgreSQL:** Ця система управління базами даних була обрана за її надійність, потужні функціональні можливості та відкритий код. PostgreSQL забезпечує високу продуктивність, збереження даних та масштабованість, що є ключовими вимогами для додатку з управління товарообігом.
5. **Архітектурний патерн MVC:** Використання патерну Model-View-Controller дозволяє чітко розділити логіку додатку на три основні компоненти: модель (дані), представлення (інтерфейс користувача) та контролер (логіка управління). Це сприяє покращенню структури коду, полегшує його тестування та підтримку, а також дозволяє більш гнучко розвивати додаток у майбутньому.

Отже, обрані технології та архітектурний патерн MVC забезпечують надійну основу для розробки ефективного та масштабованого додатку для інвентаризації та поставок. Цей підхід дозволить створити потужний інструмент для управління запасами, що відповідатиме сучасним вимогам та потребам продуктових магазинів.

РОЗДІЛ 3
РОЗРОБКА ДОДАТКУ ДЛЯ УПРАВЛІННЯ ТОВАРООБІГОМ МАГАЗИНУ

3.1 Розробка додатку для управління товарообігом магазину

3.1.1 Створення проекту

Створення початкового проекту Spring Boot через веб-інтерфейс start.spring.io є швидким та зручним способом почати роботу над монолітним додатком. Цей веб-додаток надає можливість налаштувати основні параметри вашого проекту, такі як версія Spring Boot, мова програмування (Java, Kotlin, Groovy), залежності та інші конфігураційні налаштування.

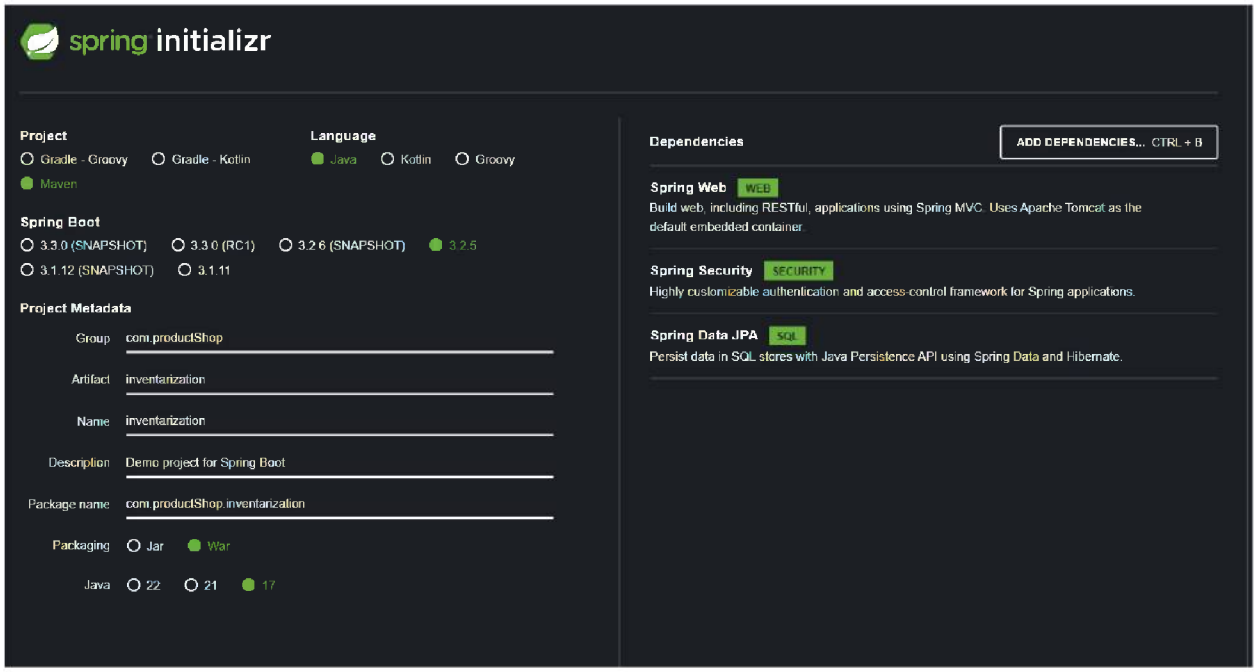


Рис 3.1 – Приклад створення проекту з використанням spring initializr

3.1.2 Впровадження монолітної архітектури

Монолітна архітектура є одним з традиційних підходів до розробки програмного забезпечення, де весь функціонал додатку реалізується як один великий блок коду, який виконується в одному процесі та на одній платформі,

що полегшує розгортання проєкту. Ось деякі детальні особливості впровадження монолітної архітектури:

1. **Єдиний модуль:** У монолітній архітектурі весь функціонал додатку зазвичай розміщується в єдиному модулі або додатку. Це означає, що всі компоненти, такі як інтерфейс користувача, логіка бізнес-процесів та доступ до даних, знаходяться в одному місці.
2. **Простота розгортання:** Так як усі частини додатку розміщені в одному монолітному зразку, розгортання додатку стає досить простим. Достатньо просто скомпілювати і запустити один артефакт.
3. **Легка масштабованість:** Хоча масштабування окремих частин додатку у монолітній архітектурі може бути викликом, масштабування всього додатку зазвичай вважається простим, оскільки він розгортається як єдиний блок.
4. **Залежності між компонентами:** У монолітних додатках компоненти можуть спільно використовувати дані та функціональні можливості без значних зусиль, оскільки вони зазвичай використовують одну базу даних і можуть безпосередньо взаємодіяти один з одним.
5. **Уніфікований технологічний стек:** У монолітних додатках зазвичай використовується один технологічний стек, оскільки всі частини додатку розгортаються разом. Це може спрощувати розробку, але також може виправдовувати технологічні обмеження.

Загалом, монолітна архітектура дозволяє швидко почати розробку додатку і забезпечує простоту управління та розгортанням, але може мати свої обмеження з точки зору масштабованості та розвитку з часом.

Проект був розбит на модулі і підмодулі, які відокремлюють класи за їх призначенням.

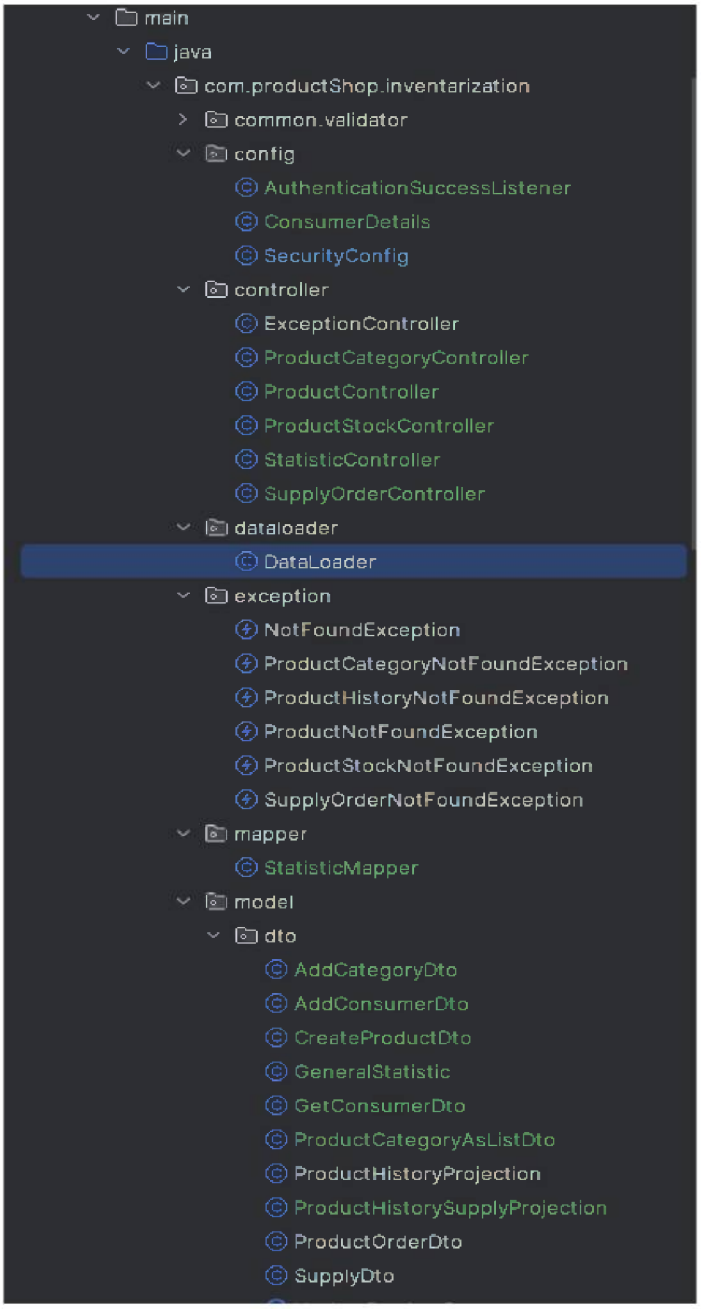


Рис 3.2 – Кодова база додатку

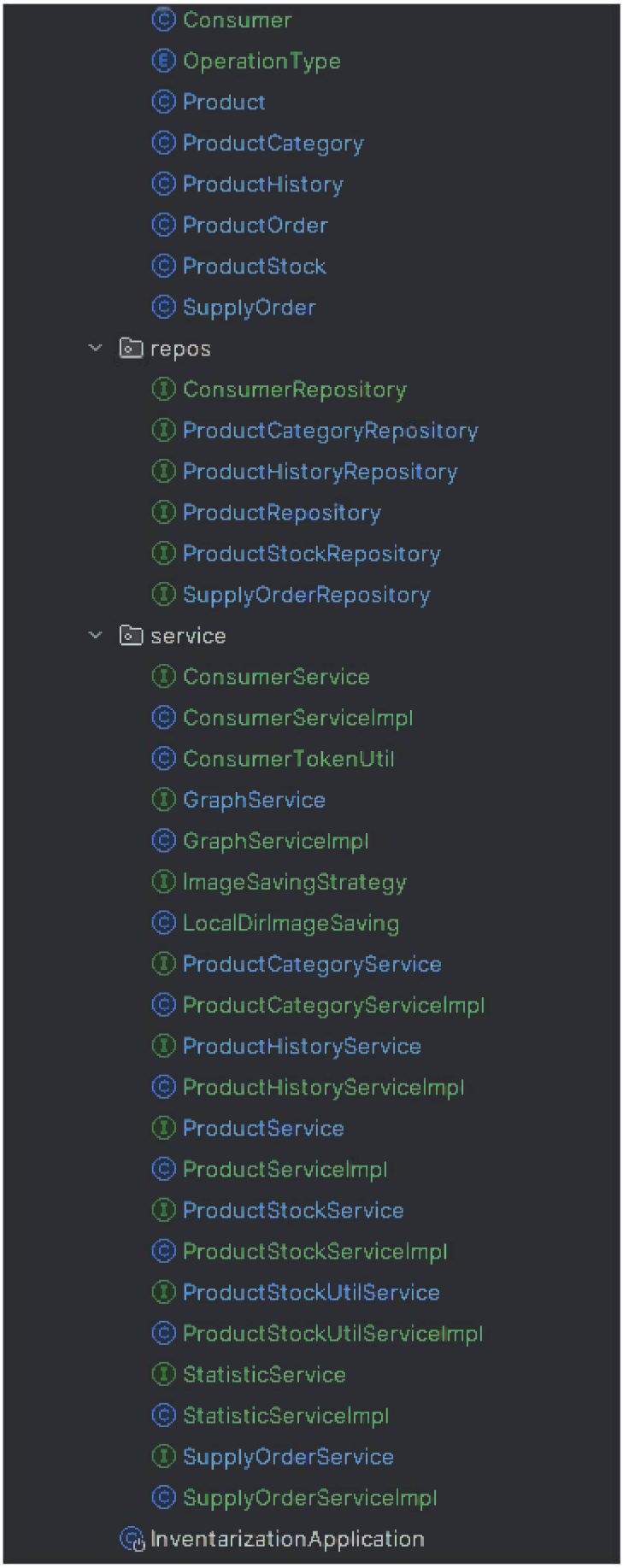


Рис 3.3 – Кодова база додатку

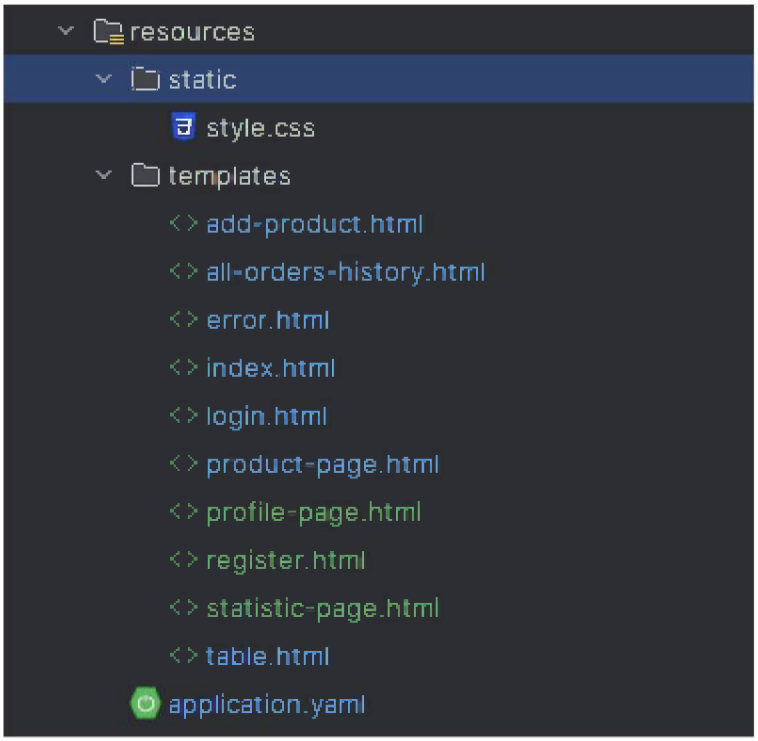


Рис 3.4 – Кодова база додатку

3.1.3 Детальний опис основних компонентів

Опис файлів конфігурації

application.yaml (або **application.yml**) є основним конфігураційним файлом для Spring Boot додатків. Він використовується для визначення різноманітних налаштувань, як-от параметри підключення до бази даних, конфігурації сервера, кастомні налаштування додатків та інше. В ньому описується:

- Налаштування обробки мультимедійних файлів
- Налаштування джерела даних (DataSource)
- Налаштування JPA (Java Persistence API)
- Кастомні налаштування

```yaml
spring:
  # Налаштування обробки мультимедійних файлів
  servlet:
    multipart:
      enabled: true
      max-file-size: 2MB
      max-request-size: 2MB
  # Налаштування джерела даних (DataSource)
  datasource:
    url: jdbc:postgresql://localhost:5433/inventoryOfShop
    username: superuser
    password: superuser
  # Налаштування JPA (Java Persistence API)
  jpa:
    hibernate:
      ddl-auto: create-drop
    show-sql: false
    properties:
      hibernate:
        dialect: org.hibernate.dialect.PostgreSQLDialect
# Налаштування базового шляху для зберігання зображень
image:
  basepath: product_images

```

Рис 3.5 – Файл конфігурації application.yaml

Для збереження інформації додаток використовує базу даних PostgreSQL, яка запускається у Docker-контейнері. Для досягнення цього потрібно налаштувати середовище виконання та запустити файл docker-compose.yml (рис. 6.6), виконавши команду docker-compose up у терміналі чи запустити в IDE. Для додатку використовується одна база даних, де зберігаються всі дані.

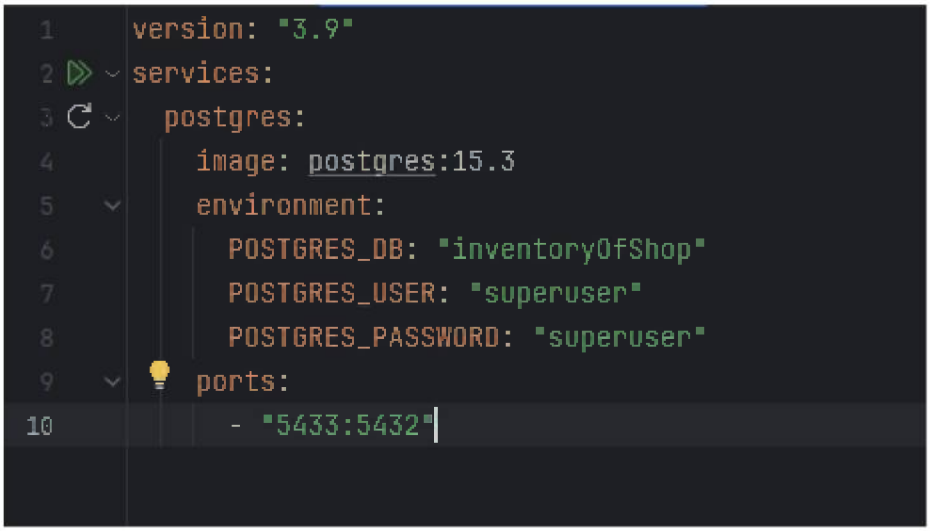


Рис 3.6 – Docker-compose файл бази даних.

Таблиця 3.1

Опис пакету config

Клас	**Опис**
SecurityConfig	Конфігурація безпеки додатку з використанням Spring Security. Встановлює правила доступу до API, налаштування сторінок входу та виходу, кодування паролів тощо
AuthenticationSuccessListener	Обробляє успішні спроби аутентифікації, забезпечуючи додаткові дії після входу користувача

Таблиця 3.2

Опис пакету common.validator

Клас	Опис
ProductHistoryValidator	Перевіряє валідність історії продукту, включаючи перевірку ціни та кількості продукту. Повертає **true**, якщо дані невалідні
ProductStockValidator	Перевіряє валідність інформації про складські запаси. Повертає **true**, якщо кількість продукту менше нуля, що свідчить про невалідність складських запасів

Таблиця 3.3

Опис пакету controller

Клас	Опис
ExceptionController	Обробляє виключення, що виникають в процесі роботи веб-інтерфейсу, забезпечуючи відповідне інформування користувачів про помилки
ProductCategoryController	Контролює операції, пов’язані з категоріями продуктів: повертає список усіх категорій, дозволяє додавання нової категорії
ProductStockController	Контролює дії над складськими запасами продуктів: реєстрація продажу, списання та додавання кількості продуктів

StatisticController	Контролює збір та представлення статистичних даних: виведення загальної статистики, оновлення даних діаграми
SupplyOrderController	Керує замовленнями на постачання продуктів: виведення історії замовлень, оновлення даних замовлень
ProductController	Керує загальними діями з продуктами: повернення списку продуктів, перегляд детальної інформації, оновлення даних та додавання нових продуктів

Таблиця 3.4

Опис пакету service

Клас	**Опис**
ProductStockService	Управління складськими запасами продуктів, включаючи додавання, списання та контроль залишків
ProductStockUtilService	Надає допоміжні утиліти для роботи із складськими запасами
StatisticService	Обробка та аналіз статистичних даних, збір інформації з різних джерел для звітності та аналізу
SupplyOrderService	Управління замовленнями на постачання, включаючи їх створення, оновлення та відслідковування

ConsumerService	Управління даними споживачів, включаючи авторизацію та обробку даних користувачів
GraphService	Створення та управління графічними даними для візуалізації статистики
ProductCategoryService	Управління категоріями продуктів, додавання, оновлення та отримання категорій
SupplyOrderService	Управління замовленнями на постачання, включаючи їх створення, оновлення та відслідковування
ProductHistoryService	Зберігання та обробка історичних даних про продукти

Таблиця 3.5

Опис пакету repository

Клас	**Опис**
ProductCategoryRepository	Здійснює операції з базою даних для об'єктів категорій продуктів, включаючи збереження, видалення та пошук категорій
ProductHistoryRepository	Здійснює операції з історією продуктів у базі даних, забезпечуючи збереження та пошук історичних даних продуктів

ProductRepository	Здійснює операції з продуктами у базі даних, включаючи збереження, видалення та пошук продуктів
ProductStockRepository	Здійснює операції зі складськими запасами продуктів у базі даних, забезпечуючи збереження, видалення та пошук складських запасів
SupplyOrderRepository	Здійснює операції з замовленнями на постачання продуктів у базі даних, включаючи збереження, видалення та пошук замовлень
ConsumerRepository	Здійснює операції з даними користувачів у базі даних, включаючи збереження, видалення та пошук даних користувачів

Таблиця 3.6

Опис пакету resources

Папка	**Опис**
static	Містить статичні файли, які не змінюються під час роботи додатку і віддаються клієнту "як є"
templates	Містить шаблони, які використовуються для візуалізації веб-інтерфейсу користувача

Ось краткі описи кожного шаблону:

- **add-product.html** - Сторінка для додавання нових продуктів у систему.

- **all-orders-history.html** - Сторінка для перегляду історії всіх замовлень.
- **error.html** - Сторінка помилки, яка показується користувачу в разі виникнення непередбачуваних помилок.
- **login.html** - Сторінка для входу користувачів.
- **index.html** - Головна сторінка додатку.
- **product-page.html** - Детальна сторінка конкретного продукту.
- **profile-page.html** - Сторінка профілю користувача, яка містить інформацію про користувача.
- **register.html** - Сторінка для реєстрації нових користувачів.
- **statistic-page.html** - Сторінка для відображення статистичних даних та графіків.
- **table.html** - Сторінка, яка містить таблицю з даними.

Ці шаблони використовуються для того, щоб надати користувачу взаємодію з додатком через веб-інтерфейс, забезпечуючи йому можливість здійснювати різноманітні операції, як-от перегляд сторінок, введення даних та отримання інформації.

Висновок за розділом 3

У цьому розділі було описано розробку додатку для управління товарообігом магазину. З використанням технологій та архітектурного патерну, обраних у попередньому розділі, було створено повнофункціональний додаток, який забезпечує ефективне управління інвентаризацією та постачаннями у продуктовому магазині.

Опис функціональних можливостей:

1. **Управління категоріями продуктів**: Користувачі можуть переглядати список категорій, додавати нові категорії та оновлювати існуючі.
2. **Управління складськими запасами**: Додаток дозволяє реєструвати продажі, списувати та додавати кількість продуктів на складі, що сприяє точному обліку товарів.

3. **Збір та аналіз статистичних даних**: Додаток надає можливість переглядати загальну статистику продажів, оновлювати дані діаграм за категоріями та періодами, що допомагає в аналізі товарообігу.
4. **Управління замовленнями на постачання**: Користувачі можуть переглядати історію замовлень на постачання, оновлювати дані про замовлення та отримувати інформацію за певний період.
5. **Управління продуктами**: Додаток забезпечує можливість переглядати список продуктів, оновлювати їх дані, додавати нові продукти та переглядати детальну інформацію про кожен продукт.

Таким чином, створений додаток для управління товарообігом магазину повністю відповідає поставленим вимогам та забезпечує ефективне управління запасами і постачаннями. Впровадження цього додатку дозволить продуктовому магазину оптимізувати процеси інвентаризації, знизити витрати та підвищити продуктивність.

ВИСНОВКИ

У результаті виконаної роботи було досягнуто мети розробки додатку для управління товарообігом магазину. Проведено всебічний аналіз існуючих додатків для інвентаризації та поставок, досліджено принципи роботи різних систем товарообігу та вивчено функціонування систем продуктових магазинів. Аналіз показав, що сучасні рішення значно підвищують ефективність управління запасами та оптимізують процеси постачання. Ключовими аспектами ефективного функціонування є точність даних, швидкість обробки інформації та можливість налаштування під специфічні потреби магазину.

На основі проведеного аналізу було обрано оптимальні технології та інструменти для розробки додатку: Java, Spring Boot, Thymeleaf та PostgreSQL. Використання архітектурного патерну Model-View-Controller (MVC) дозволило чітко розділити логіку додатку на три основні компоненти, що покращило структуру коду, полегшило його тестування та підтримку, а також забезпечило надійну основу для розробки ефективного та масштабованого додатку.

Розроблений додаток забезпечує ефективне управління інвентаризацією та постачаннями у продуктовому магазині. Основні функціональні можливості додатку включають управління категоріями продуктів, складськими запасами, збір та аналіз статистичних даних, управління замовленнями на постачання та продуктами. Впровадження цього додатку дозволить продуктовому магазину оптимізувати процеси інвентаризації, знизити витрати, підвищити продуктивність та задоволеність клієнтів, що сприятиме успішній роботі магазину.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Johnson, R., et al. (2014). *Spring Framework Documentation*. Pivotal Software, Inc. URL: https://spring.io/projects/spring-framework (Дата звернення: 01.04.2024).
2. Thymeleaf Documentation (2023). Thymeleaf. Retrieved from https://www.thymeleaf.org/documentation.html (Дата звернення: 16.04.2024).
3. PostgreSQL Global Development Group (2023). *PostgreSQL Documentation*. URL: https://www.postgresql.org/docs/ (Дата звернення: 01.04.2024).
4. Oracle (2023). *Java SE Documentation*. Oracle Corporation. URL: https://docs.oracle.com/en/java/ (Дата звернення: 02.04.2024).
5. Baeldung (2023). *Spring Boot Tutorials*. Retrieved from https://www.baeldung.com/spring-boot (Дата звернення: 05.04.2024).
6. GeeksforGeeks (2023). *Java Programming Language*. URL: https://www.geeksforgeeks.org/java/ (Дата звернення: 10.04.2024).
7. PostgreSQL Tutorial (2023). *PostgreSQL Tutorial*. Retrieved from URL: https://www.postgresqltutorial.com/ (Дата звернення: 05.04.2024).
8. Oracle (2023). *Java Persistence API (JPA) Documentation*. Oracle Corporation. URL:https://www.oracle.com/java/technologies/persistence-jsp.html (Дата звернення: 15.04.2024).

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) бакалавр
Галузь знань: 12 – Інформаційні технології
Спеціальність: 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри теоретичної
та прикладної системотехніки
_____ д.т.н., проф. Шматков С. І.
«21» грудня 2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

ШАМРАЯ ЕДУАРДА СЕРГІЙОВИЧА

1. Тема роботи **«КОМП'ЮТЕРНА СИСТЕМА УПРАВЛІННЯ ТОВАРООБІГОМ МАГАЗИНУ»**

керівник роботи Бикова Тетяна Володимирівна, кандидат технічних наук, доцент.

затверджені наказом по університету від «03» травня 2024року № 4101-5/909
2. Строк подання студентом роботи 31 травня 2024

3. Перелік питань, які потрібно розробити
   1) Аналіз існуючих додатків для інвентаризації та поставок для магазину.
   2) Аналіз принципів роботи додатків системи товарообігом.
   3) Ознайомлення з системою продуктових магазинів.
   4) Вибір платформи для розробки додатку.
   5) Побудова архітектури додатку.
   6) Розробка додатку для управління товарообігом магазину.
   7) Оформлення пояснювальної записки.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Аналіз існуючих додатків для інвентаризації та поставок для магазину	19.12.2023 - 25.01.2024
2	Аналіз принципів роботи додатків системи товарообігом	21.12.2023 - 2.01.2024
3	Ознайомлення з системою продуктових магазинів	2.01.2024 - 1.02.2024
4	Вибір платформи для розробки додатку	2.02.2024 - 2.02.2024
5	Побудова архітектури додатку	3.02.2024 - 16.02.2024
6	Розробка додатку для управління товарообігом магазину	17.02.2024 - 15.03.2024
7	Оформлення звіту за результатами переддипломної практики	15.05.2024 – 31.05.2024
8	Представлення кваліфікаційної роботи керівнику та рецензенту	31.05.2024
9	Оформлення пояснювальної записки та підготовка презентації	31.05.2024

5. Дата видачі завдання 21.12.2023

Студент Шамрай Е. С._____

Керівник роботи Бикова Т. В._____

ДОДАТОК Б

Технічне завдання
на розробку програмного вироби
«Комп'ютерна система управління товарообігом магазину»

Назва розділу	Назва і зміст підрозділу
1. Введення	1.1. Назва програмного виробу – Система управління товарообігом магазину. 1.2. Галузь застосування – 12 Інформаційні технології.
2. Підстава для розробки	2.1. Навчальний план за спеціальністю 121 – Комп'ютерна інженерія. 2.2. Завдання на дипломну роботу бакалавра, затверджено наказом ХНУ імені В. Н. Каразіна № 4101-5/909 від 03.05.2024 р. (представить як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3. Призначення розробки	3.1. Мета розробки програмного вироби – Створення гибкої системи товарообігу магазину. 3.2. Призначення програмного вироби для автоматизації введення обліку торгівельного бізнесу. 3.3. Початкові дані для розробки: лист задач, їх характеристики.
4. Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: 1) представляти з себе програмну реалізацію 2) додаток, який дозволяє зберігати, додавати та редагувати асортимент продуктів магазину, а також вести облік продажів цих продуктів 3) надавати інтегруванні з іншими системами 4) надавати додаток з візуальним інтерфейсом який буде зберігати в собі всі дані за наданими параметрами та умовами. 4.2. Вимоги до надійності: Система повинна бути надійною, забезпечуючи безпечне збереження даних, захист від несанкціонованого доступу, високу доступність та масштабованість. 4.3. Вимоги до умов експлуатації приміщення з необхідним обладненням. 4.4. Вимоги до складу і параметрів технічних засобів Персональний комп'ютер у повній комплектації (ноутбук) 4.5. Вимоги до інформаційної та програмної сумісності забезпечити сумісність з усіма обчислювальними засобами. 4.6. Вимоги до маркування та упаковки відсутні. 4.7. Вимоги до транспортування і зберігання відсутні. 4.8. Спеціальні вимоги не пред'являються.
5. Вимоги до програмної документації.	Програмною документацією до виробу «Система управління товарообігом магазину» вважати: 1) Справжнє Технічне завдання на розробку програмного вироби (представити у вигляді Додатку Б до пояснювальної записки до дипломної роботи). 2) Програму і методику випробувань розробленого програмного вироби (представити у вигляді Додатку В до пояснювальної записки до дипломної роботи).

<table>
<tr><td></td><td>3) Опис програмного вироби (представити в розділі 3 пояснювальної записки до дипломної роботи).
4) Текст програми (представити в Додатку Г до пояснювальної записки до дипломної роботи).</td></tr>
<tr><td>6. Техніко-економічні показники</td><td>Вимоги до розрахунку техніко-економічних показників не потрібні.</td></tr>
<tr><td>7. Стадії і етапи розробки</td><td>
<table>
<tr><th>№ з/п</th><th>Назва етапів роботи</th><th>Термін виконання етапів роботи</th></tr>
<tr><td>1</td><td>Аналіз існуючих додатків для інвентаризації та поставок для магазину</td><td>19.12.2023 - 25.01.2024</td></tr>
<tr><td>2</td><td>Аналіз принципів роботи додатків системи товарообігом</td><td>21.12.2023 - 2.01.2024</td></tr>
<tr><td>3</td><td>Ознайомлення з системою продуктових магазинів</td><td>2.01.2024 - 1.02.2024</td></tr>
<tr><td>4</td><td>Вибір платформи для розробки додатку</td><td>2.02.2024 - 2.02.2024</td></tr>
<tr><td>5</td><td>Побудова архітектури додатку</td><td>3.02.2024 - 16.02.2024</td></tr>
<tr><td>6</td><td>Розробка додатку для управління товарообігом магазину</td><td>17.02.2024 - 15.03.2024</td></tr>
<tr><td>7</td><td>Оформлення звіту за результатами переддипломної практики</td><td>15.05.2024 – 31.05.2024</td></tr>
<tr><td>8</td><td>Представлення кваліфікаційної роботи керівнику та рецензенту</td><td>31.05.2024</td></tr>
<tr><td>9</td><td>Оформлення пояснювальної записки та підготовка презентації</td><td>31.05.2024</td></tr>
</table>
</td></tr>
<tr><td>8. Порядок контролю і приймання</td><td>1) Перевірку ходу розробки програмного виробу керівнику робіт виконувати раз в 2 тижні.

2) Випробування програмного продукту провести відповідно до програми та методики випробувань на базі комп'ютерного класу.

3) Захист розробленої моделі провести на засіданні Атестаційної комісії.

4) Пояснювальну записку подати на паперових носіях в 1 примірнику і в електронному вигляді в 1 примірнику на CD R компакт-диску.</td></tr>
</table>

Виконавець

студент групи КІ-41

Шамрай Е.С.

Замовник

ДОДАТОК В

Програма і методика випробувань
програмного виробу
«Комп'ютерна система управління товарообігом магазину»

1 Об'єкт випробувань

1.1 Найменування випробуваного програмного виробу «Система управління товарообігом магазину».

1.2 Область його застосування торгівля, IT-індустрія і розробка ПЗ.

2. Мета випробувань

Покращення ефективності і точності системи товарообігу для магазину.

3. Загальні положення

3.1 Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

3.2 Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3.3 Обсяг випробувань

Приймальні випробування програмного виробу проводяться в обсязі відповідному цієї Програми і методики випробувань.

3.4 Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

4.1. Вимоги до функціональних характеристик:

1) представляти з себе програмну реалізацію

2) надати додаток, який дозволяє зберігати, додавати та редагувати асортимент продуктів магазину, а також вести облік продажів цих продуктів

3) надавати інтегруванні з іншими системами

4) надавати додаток з візуальним інтерфейсом який буде зберігати в собі всі дані за наданими параметрами та умовами.

4.2. Вимоги до надійності:

Система повинна бути надійною, забезпечуючи безпечне збереження даних, захист від несанкціонованого доступу, високу доступність та масштабованість.

4.3. Вимоги до умов експлуатації приміщення з необхідним обладненням.

4.4. Вимоги до складу і параметрів технічних засобів Персональний комп'ютер у повній комплектації (ноутбук)

4.5. Вимоги до інформаційної та програмної сумісності забезпечити сумісність з усіма обчислювальними засобами.

4.6. Вимоги до маркування та упаковки відсутні.

4.7. Вимоги до транспортування і зберігання відсутні.

4.8. Спеціальні вимоги відсутні.

5. Вимоги до програмної документації

Склад програмної документації, що подається на випробування, включає:

1) Технічне завдання на розробку програмного вироби (представлено в Додатку Б до пояснювальної записки до дипломної роботи).

2) Ця Програма і методика випробувань розробленого програмного вироби (представлена в Додатку В до пояснювальної записки до дипломної роботи).

3) Опис програмного вироби (представлено в розділі 3 пояснювальної записки до дипломної роботи).

6. Засоби і порядок випробувань

6.1 Засоби випробувань

Випробування проводяться на технічних засобах, яких персональний комп'ютер, ноутбук.

Випробування проводяться з використанням програмного засобу, яких EXCEL, командний рядок, браузер, IntelliJ IDEA, Postman.

6.2 Порядок проведення випробувань

6.2.1. Перевірка програмної документації.

Перевірка комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в технічному завданні документації.

6.2.2. Перевірка якості програмної документації.

Перевірку здійснювати за критерієм відповідності вимогам єдиної системи програмної документації (ЄСПД).

6.2.3. Перевірка виконання програми.

Тест 1: Перевірка відповідей розроблених моделей при однакових запитах. По вмісту запиту та відповіді робиться висновок про правильно працездатність моделей (рисунки 6.1, 6.2, 6.3).

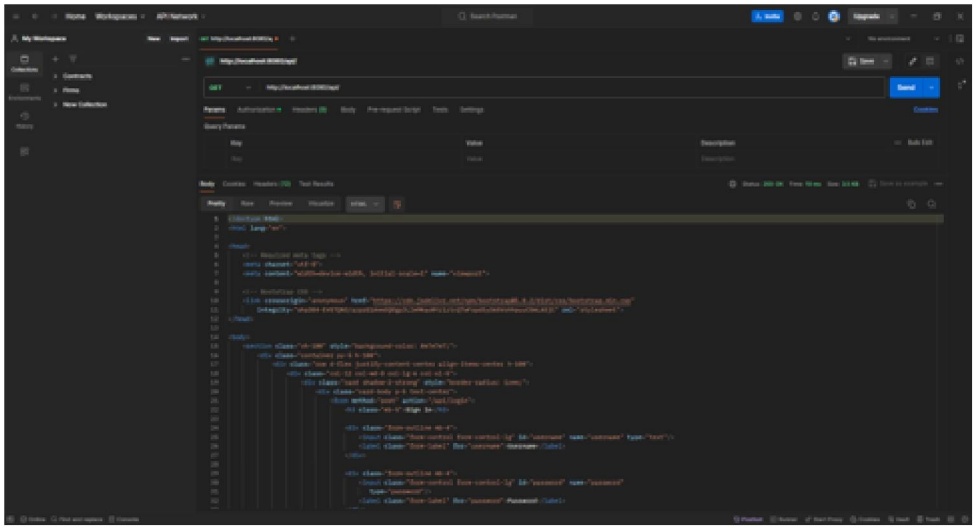


Рисунок В.1 – Перевірка сторінки усіх товарів магазину

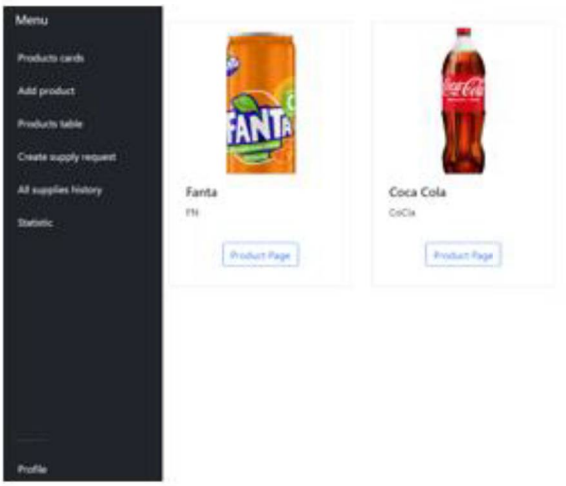


Рисунок В.2 – Представлення отриманої сторінки в браузері

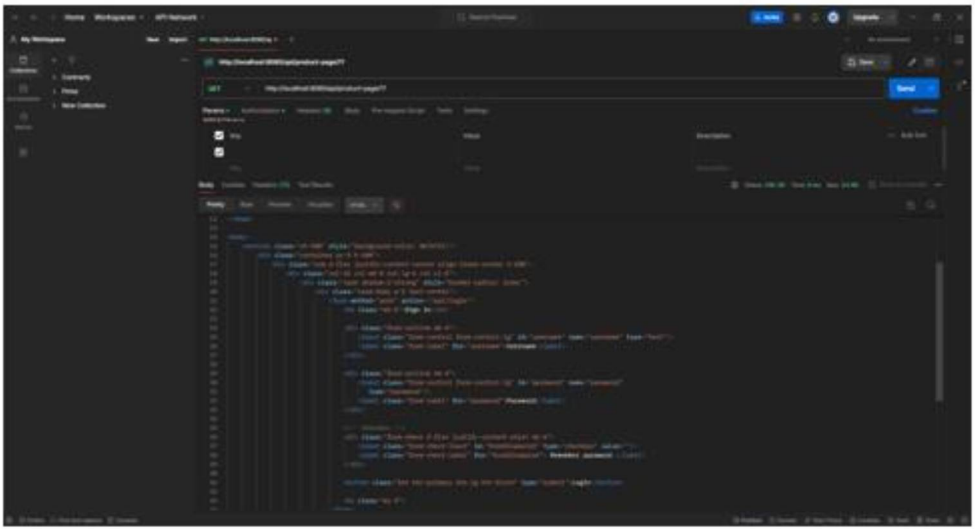


Рисунок В.3 – Перевірка отримання інформації конкретного товару

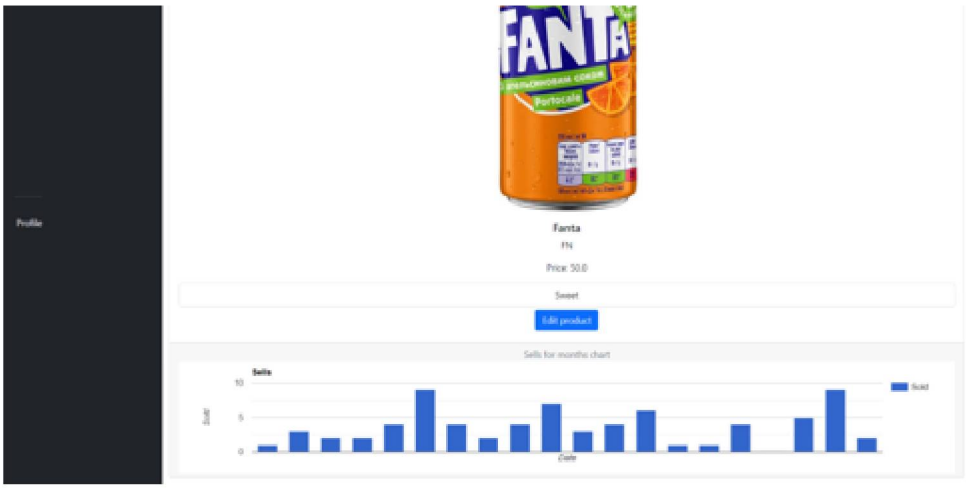


Рисунок В.4 – Представлення отриманої сторінки в браузері

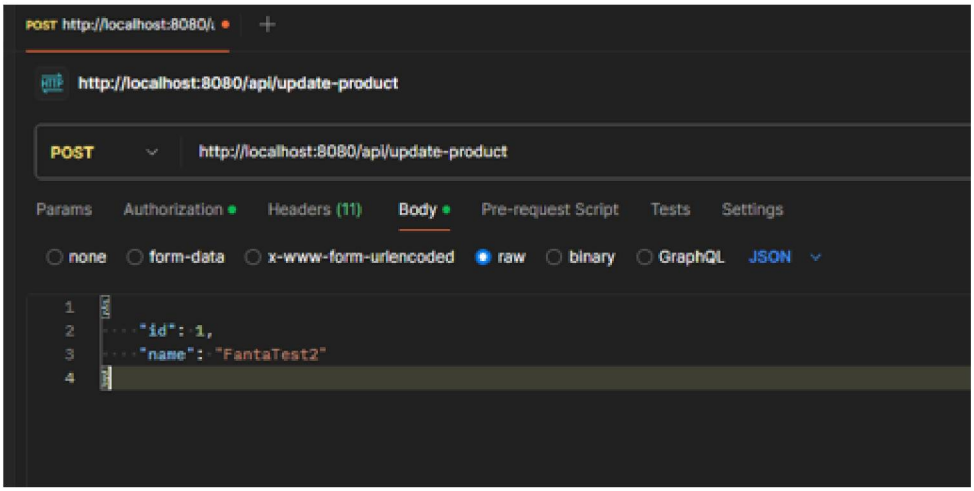


Рисунок В.5 – Перевірка оновлення інформації конкретного товару

Рисунок В.6 – Перевірка результату в браузері

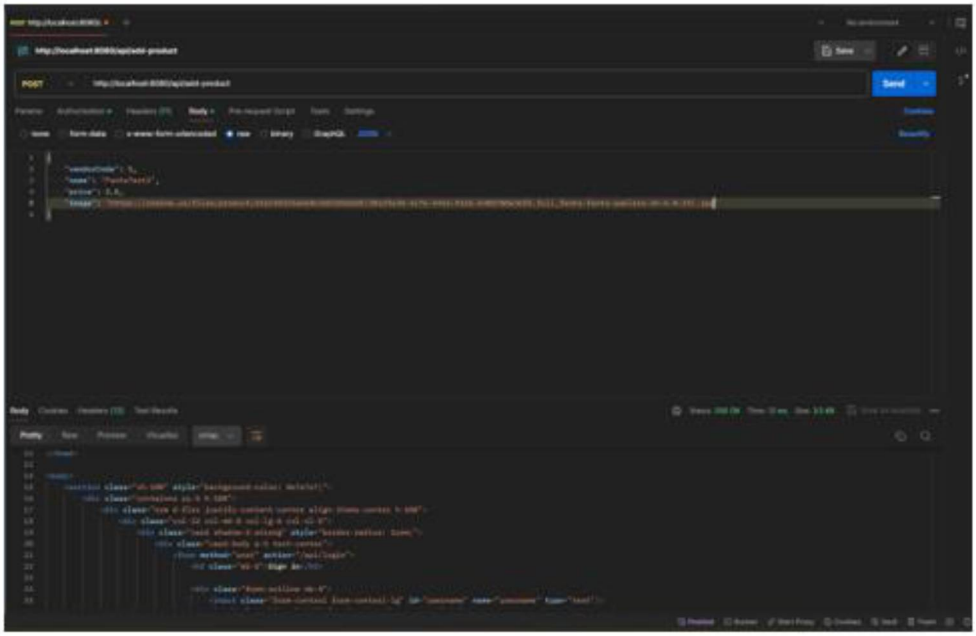


Рисунок В.7 – Перевірка створення нового товару

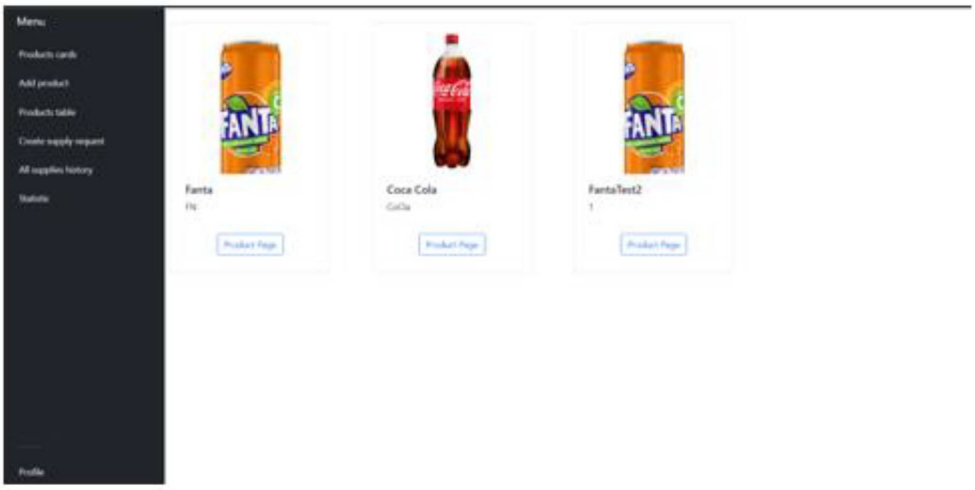


Рисунок В.8 – Перевірка результату в браузері

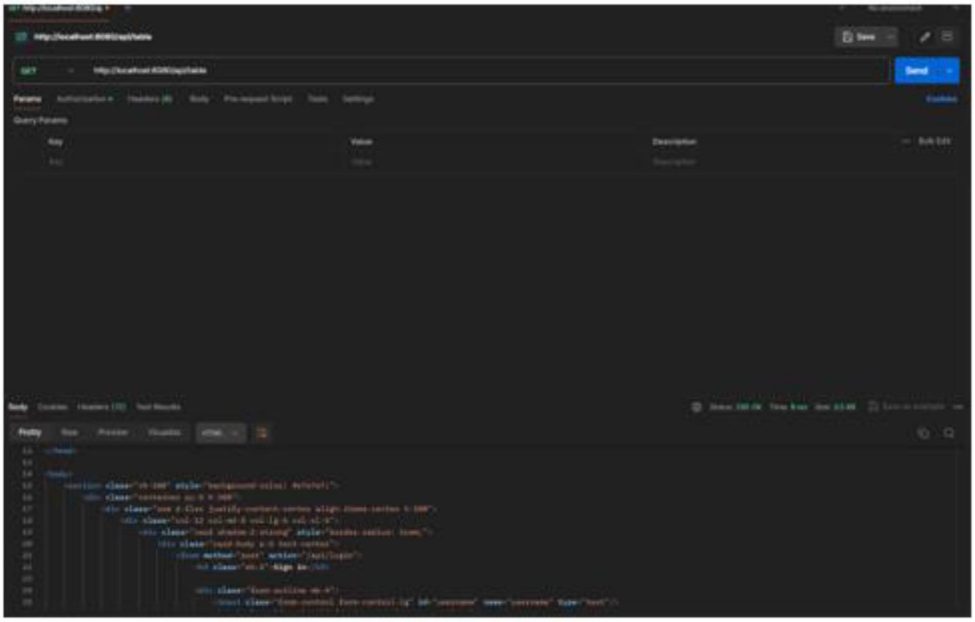


Рисунок В.9 – Перевірка отримання сторінки виводу таблиці товарів

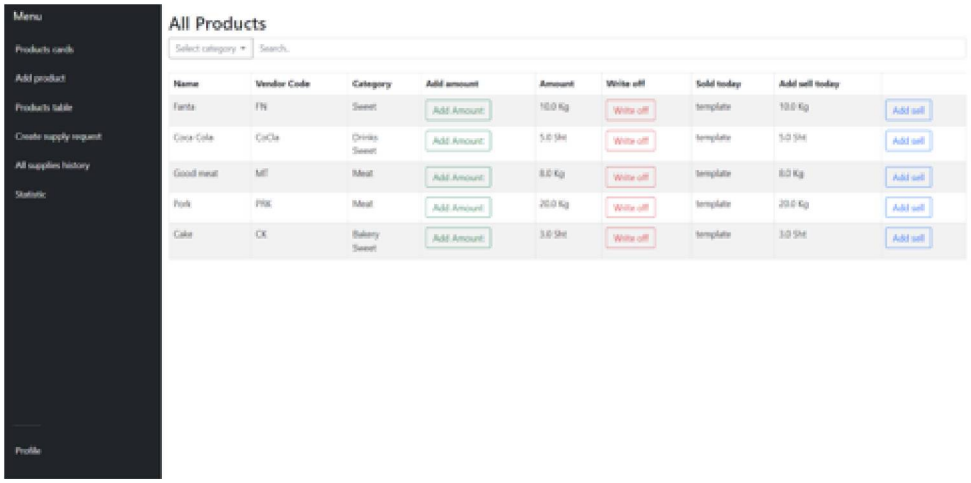


Рисунок В.10 – Перевірка результату в браузері

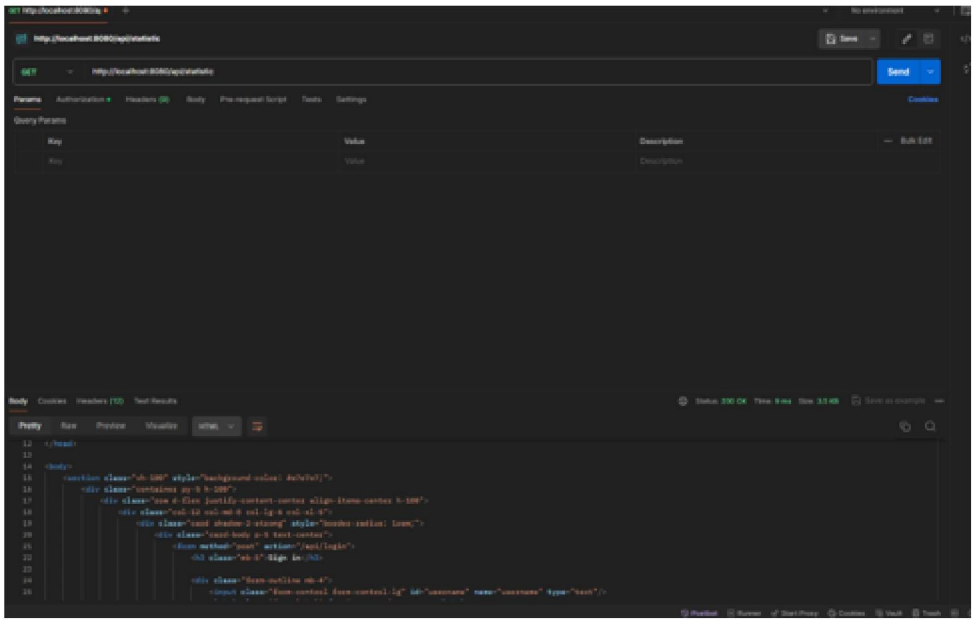


Рисунок В.11 – Перевірка отримання сторінки загальної статистики товарів

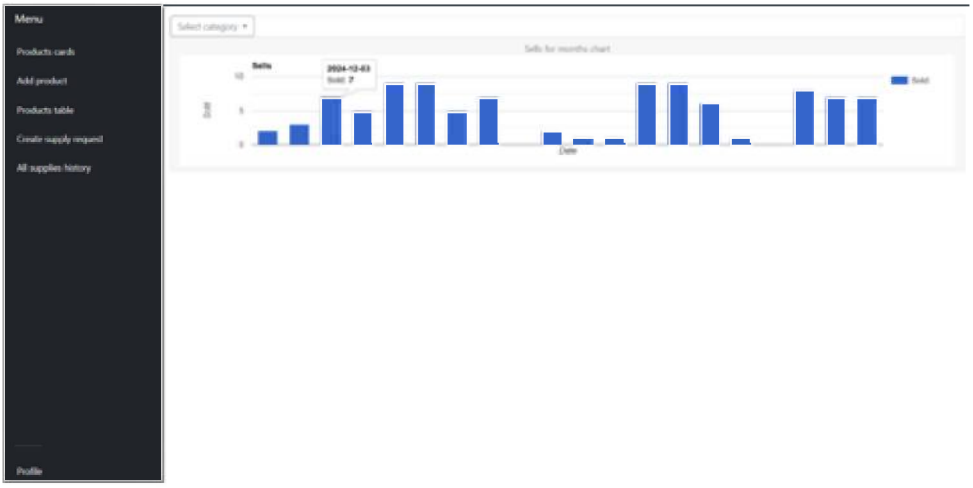


Рисунок В.12 – Перевірка результату в браузері

Висновок: : Перевірка відповідей розроблених моделей виконані запитів виконано успішно, всі відповіді відповідають перерахованому функціоналу додатка.

Виконавець

студент групи КІ-41

Шамрай Е.С.

Додаток Г

ПРИКЛАДИ РОБОТИ ДОДАТКУ.

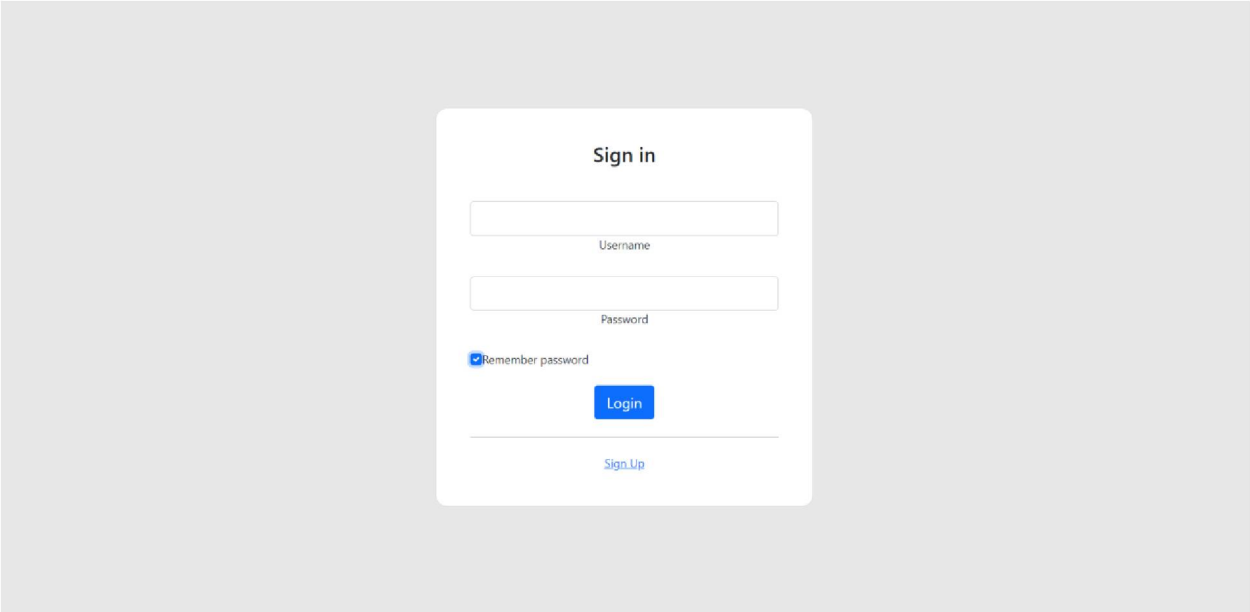


Рис Г.1 – Приклад логування користувача

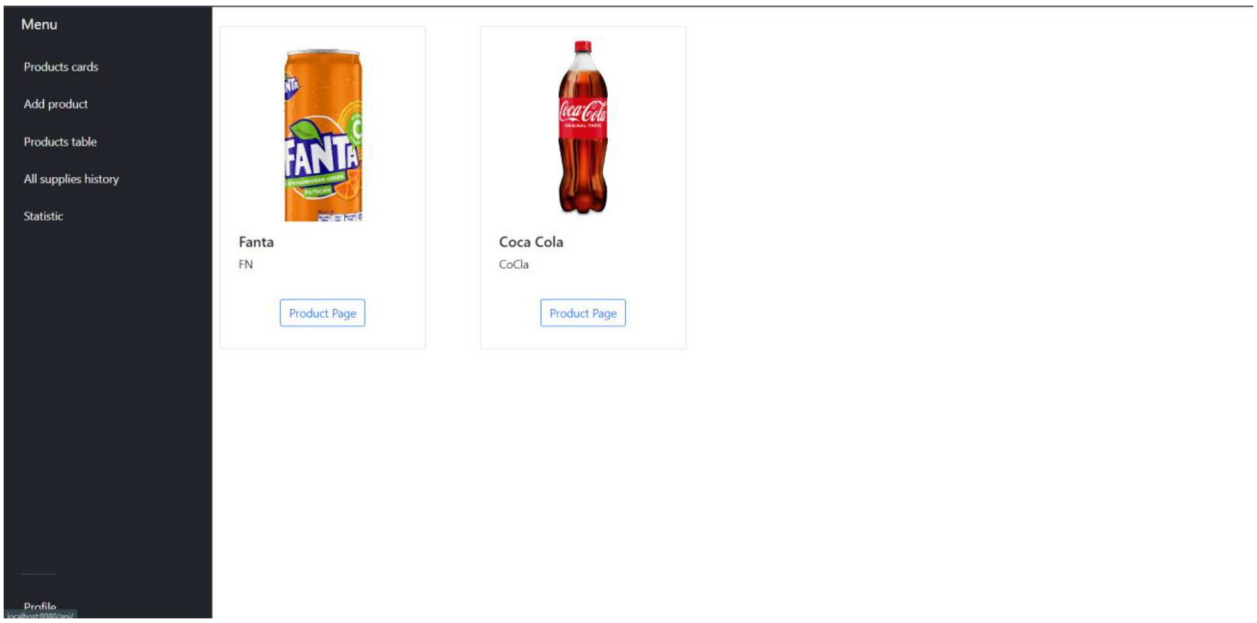


Рис Г.2 – Головна сторінка додатку

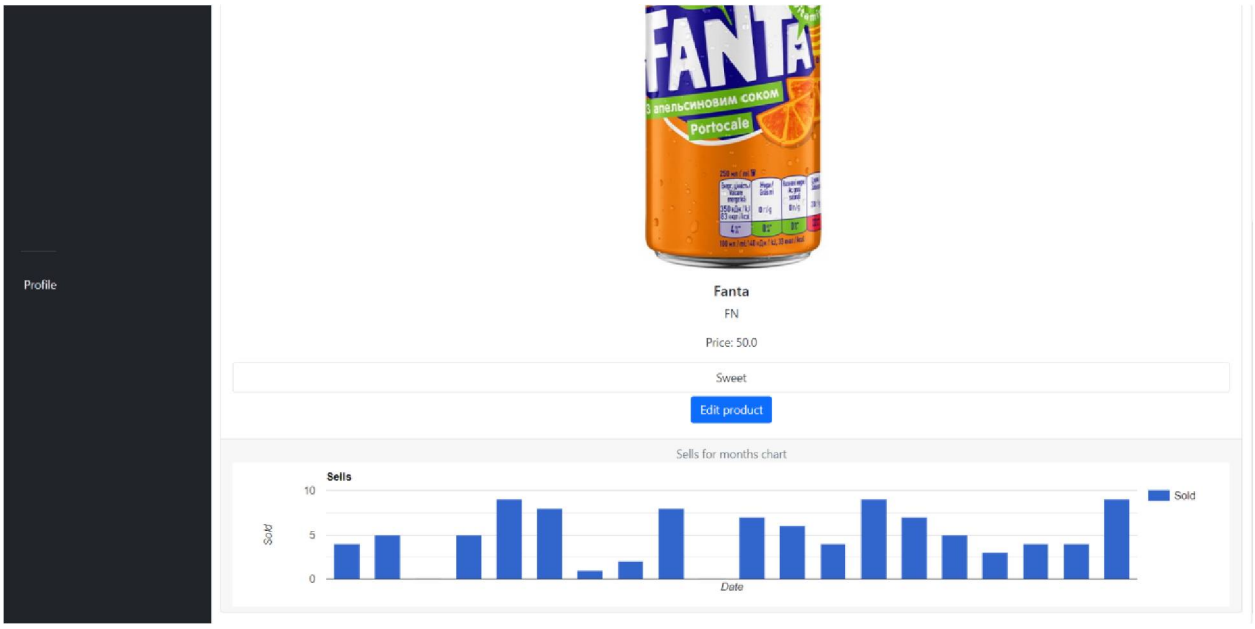


Рис Г.3 – Сторінка продукту

Update Film

Name:

Fanta

Vendor Code:

FN

Price:

50.0

Image:

Choose File | No file chosen

Categories:

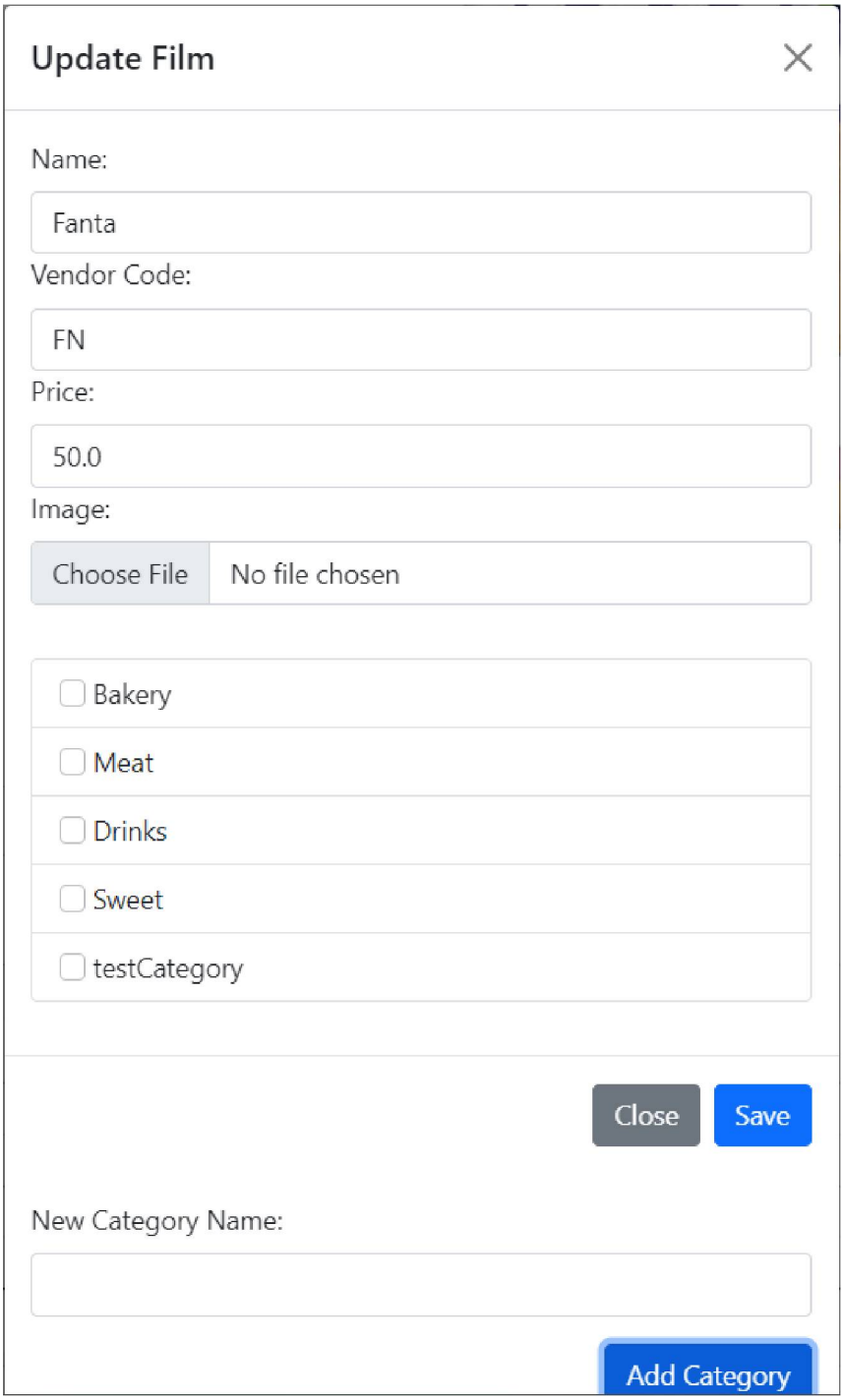
- [ ] Bakery
- [ ] Meat
- [ ] Drinks
- [ ] Sweet

Close | Save

New Category Name:

testCategory

Add Category


Update Film
Name:
Fanta
Vendor Code:
FN
Price:
50.0
Image:
Choose File
No file chosen
Bakery
Meat
Drinks
Sweet
testCategory
Close
Save
New Category Name:
Add Category

Update Film

Name:

FantaTest

Vendor Code:

FN

Price:

50.0

Image:

Choose File | No file chosen

- [ ] Bakery
- [ ] Meat
- [ ] Drinks
- [ ] Sweet
- [ ] testCategory

Close | Save

New Category Name:

Add Category

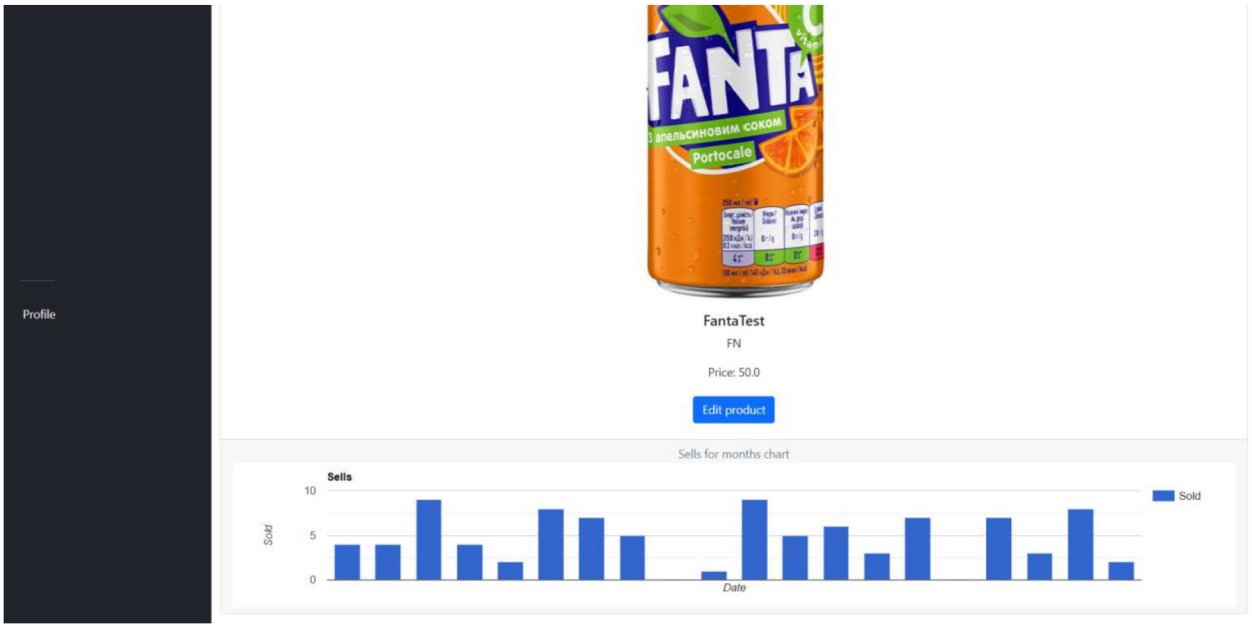


Рис Г.7 – Приклад оновленого продукту

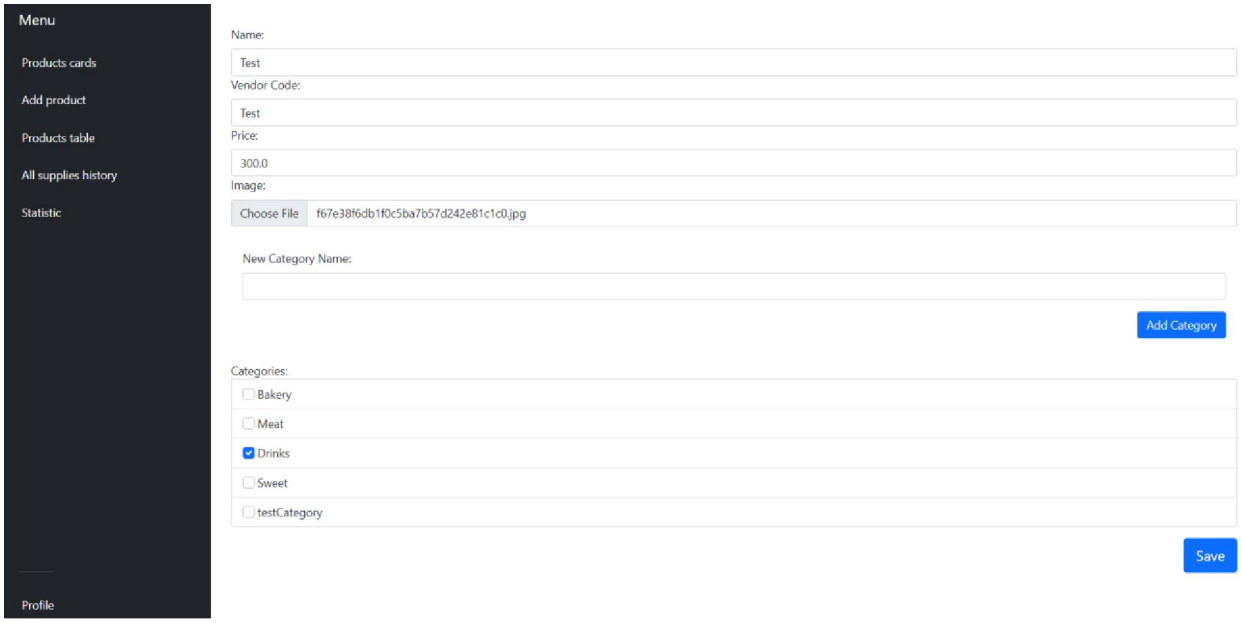


Рис Г.8 – Приклад додавання продукту

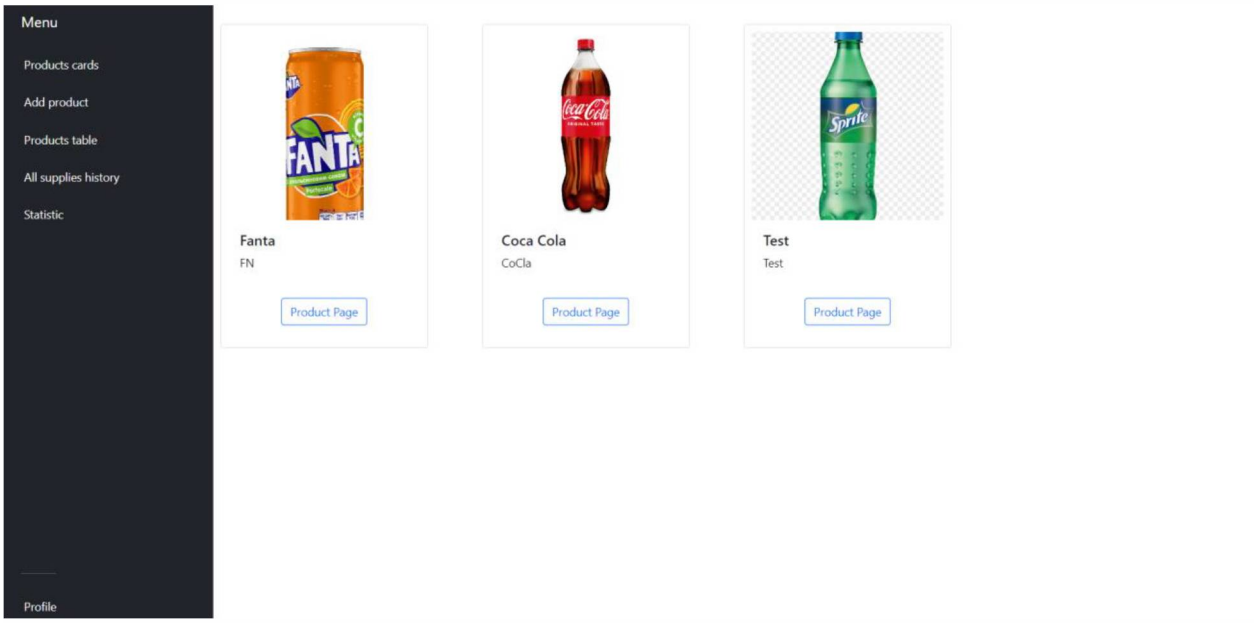


Рис Г.9 – Результат додавання продукту

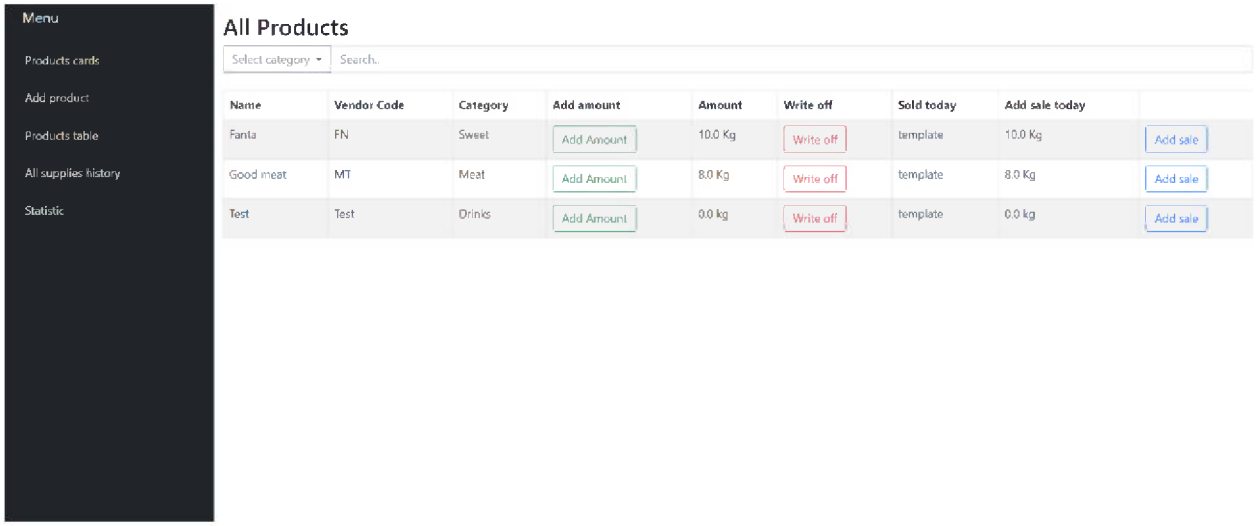


Рис Г.10 – Таблиця продуктів

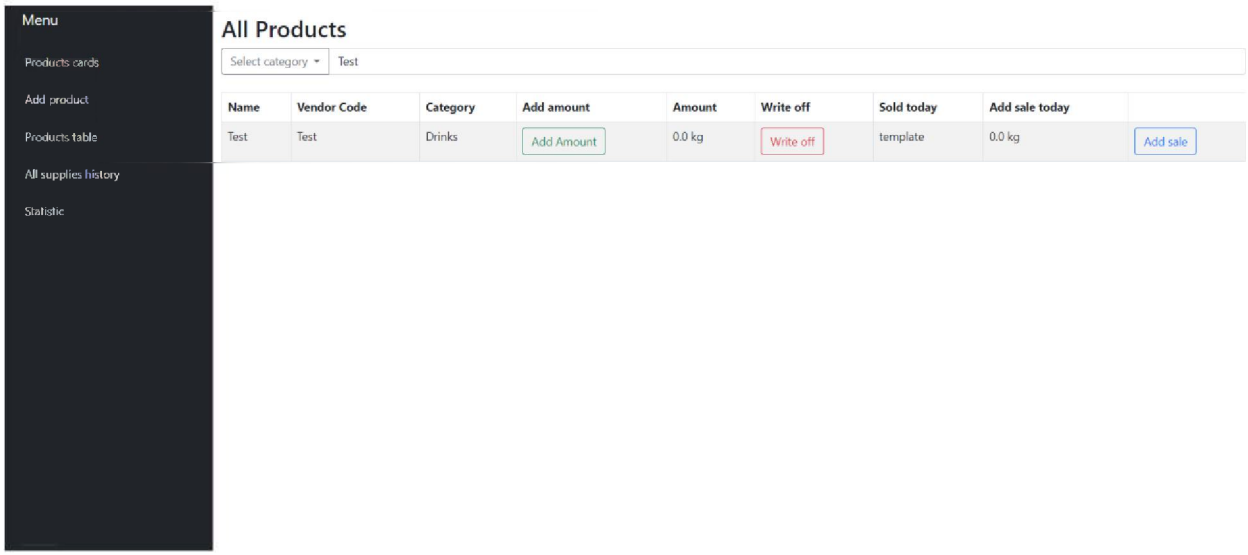


Рис Г.11 – Результат роботи пошуку

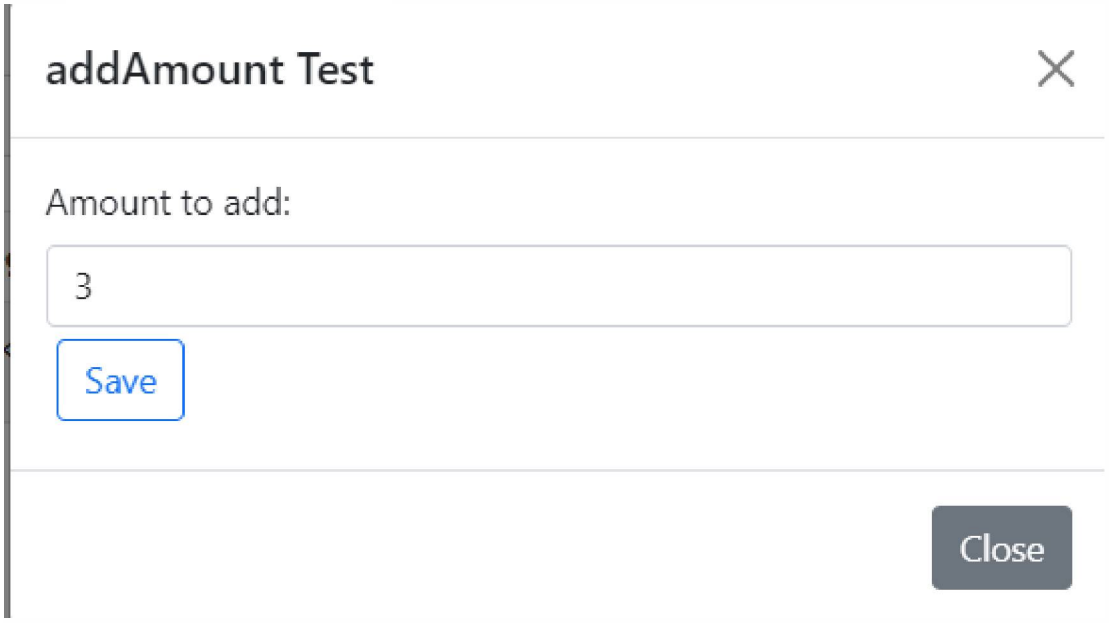


Рис Г.12 – Приклад додавання кількості продукту

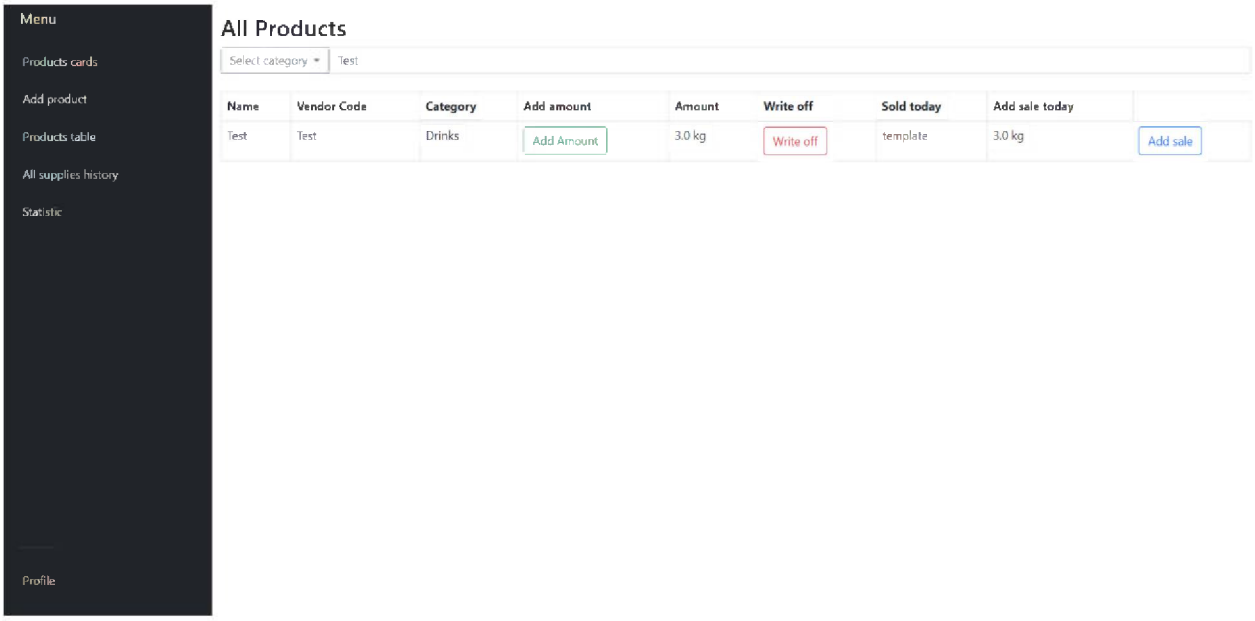


Рис Г.13 – Результат додавання кількості продукту

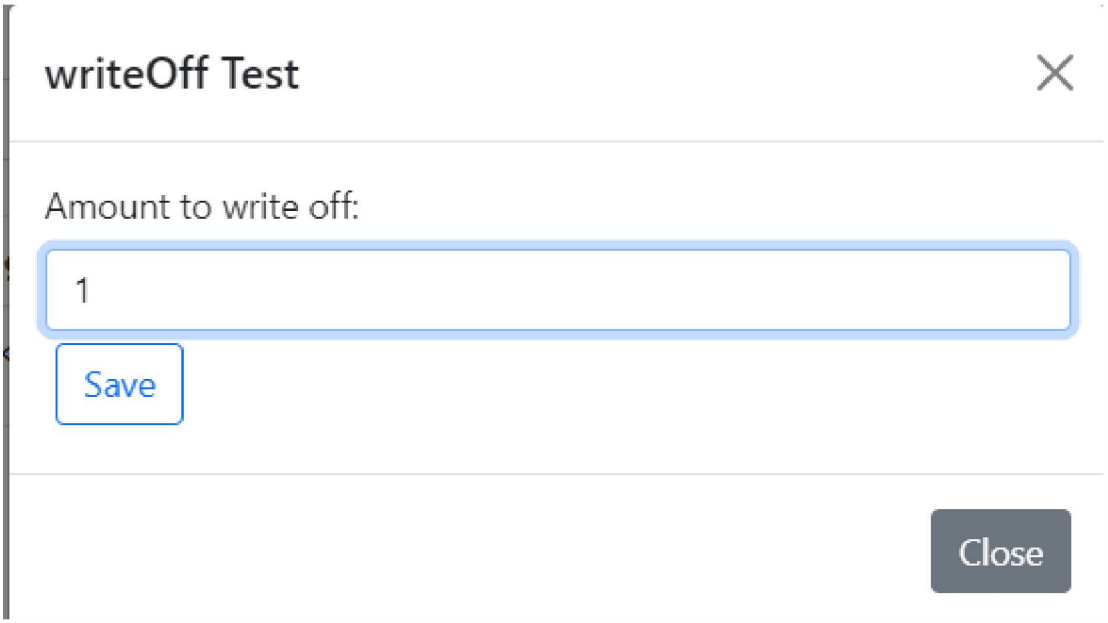


Рис Г.14 – Приклад списання кількості продукту

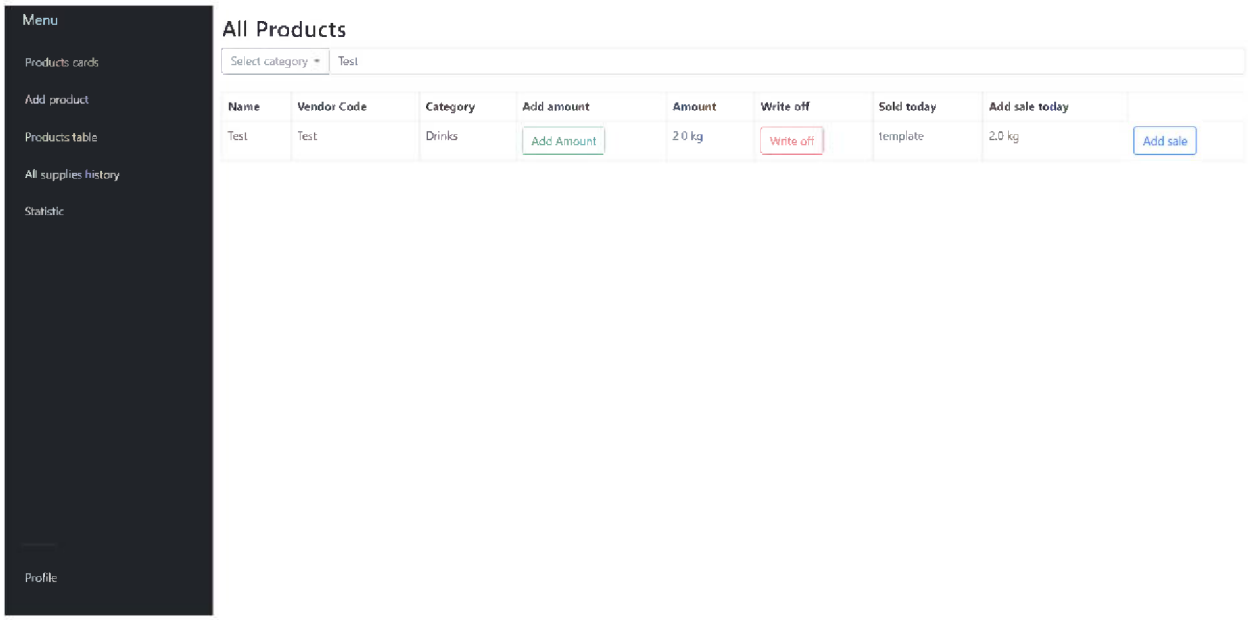


Рис Г.15 – Результат списання кількості продукту

Рис Г.16 – Приклад продажу продукту

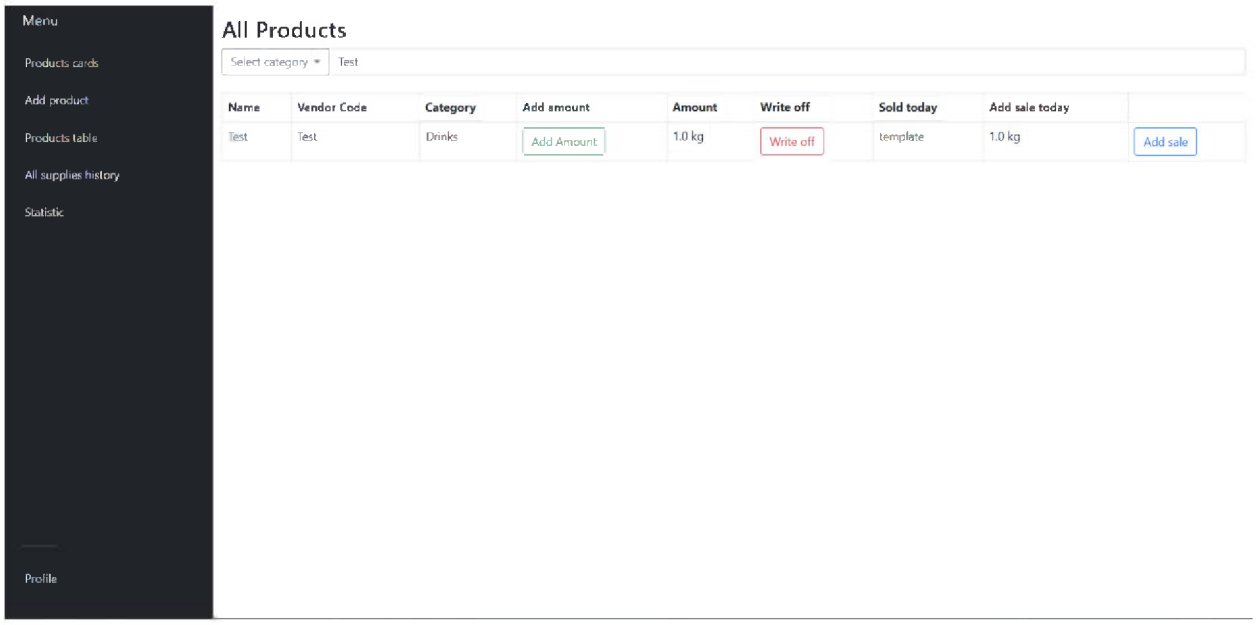


Рис Г.17 – Приклад продажу продукту

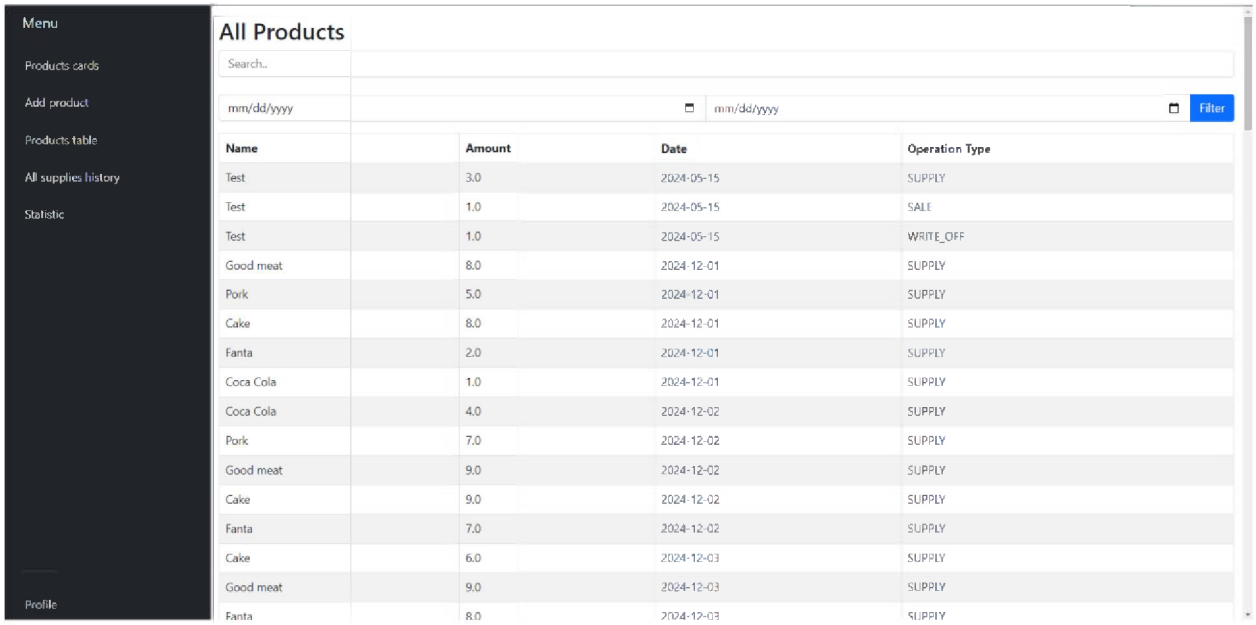


Рис Г.18 – Приклад історії продуктів

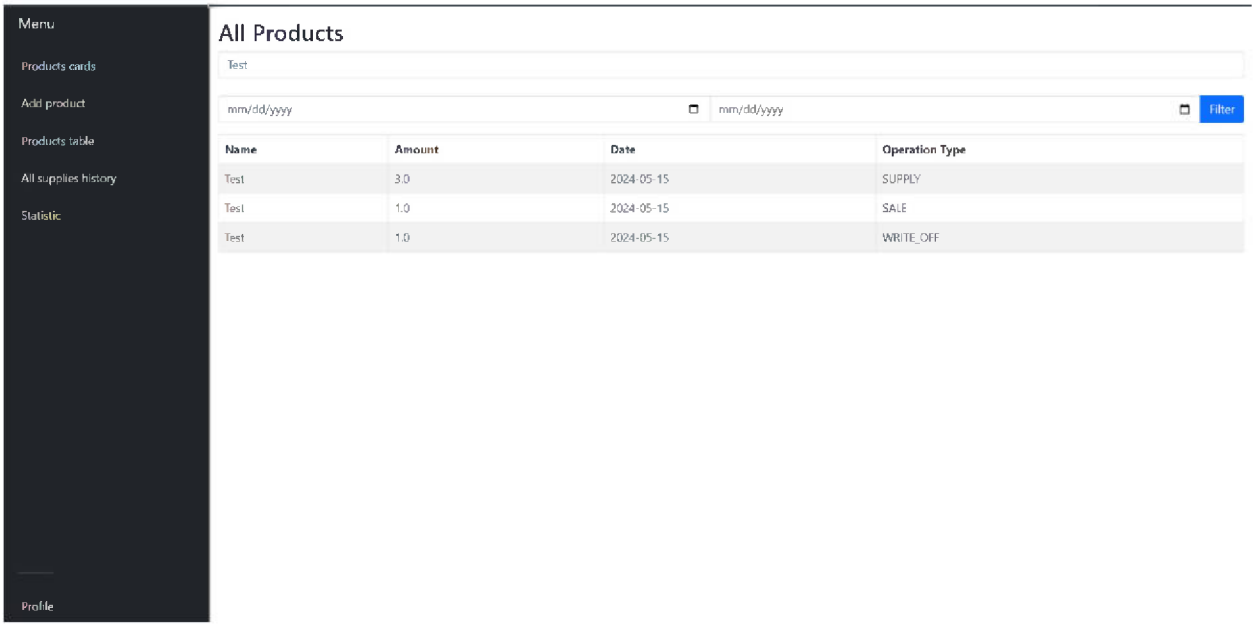


Рис Г.19 – Приклад пошуку історії продуктів за назвою

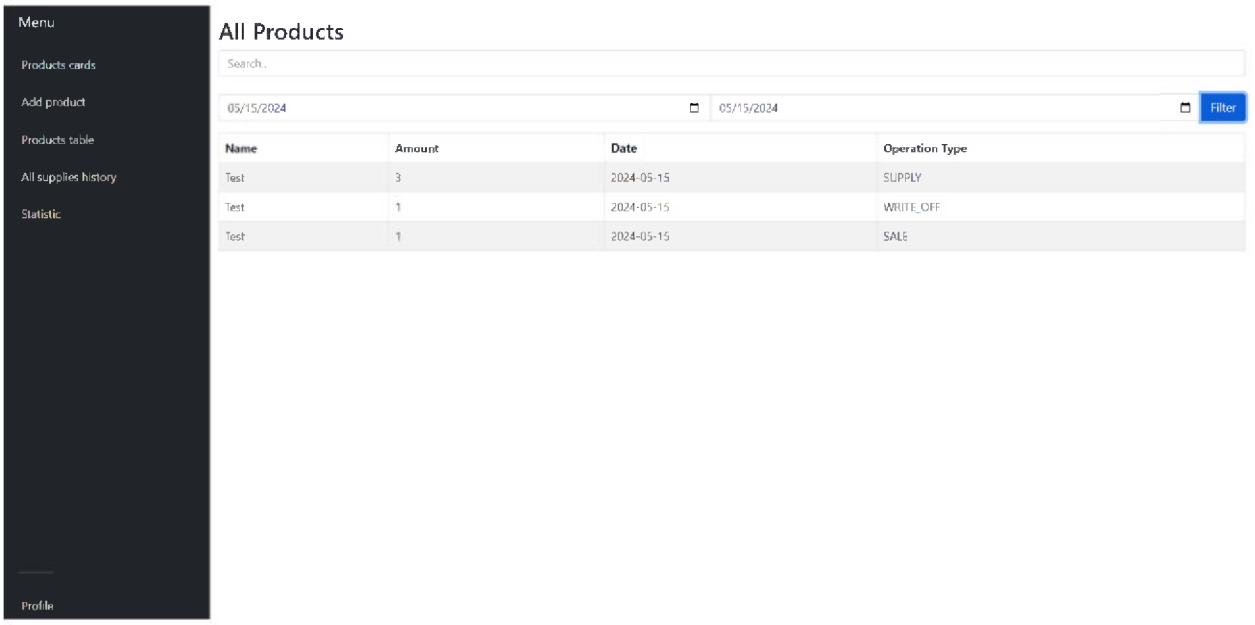


Рис Г.20 – Приклад пошуку історії продуктів за датою

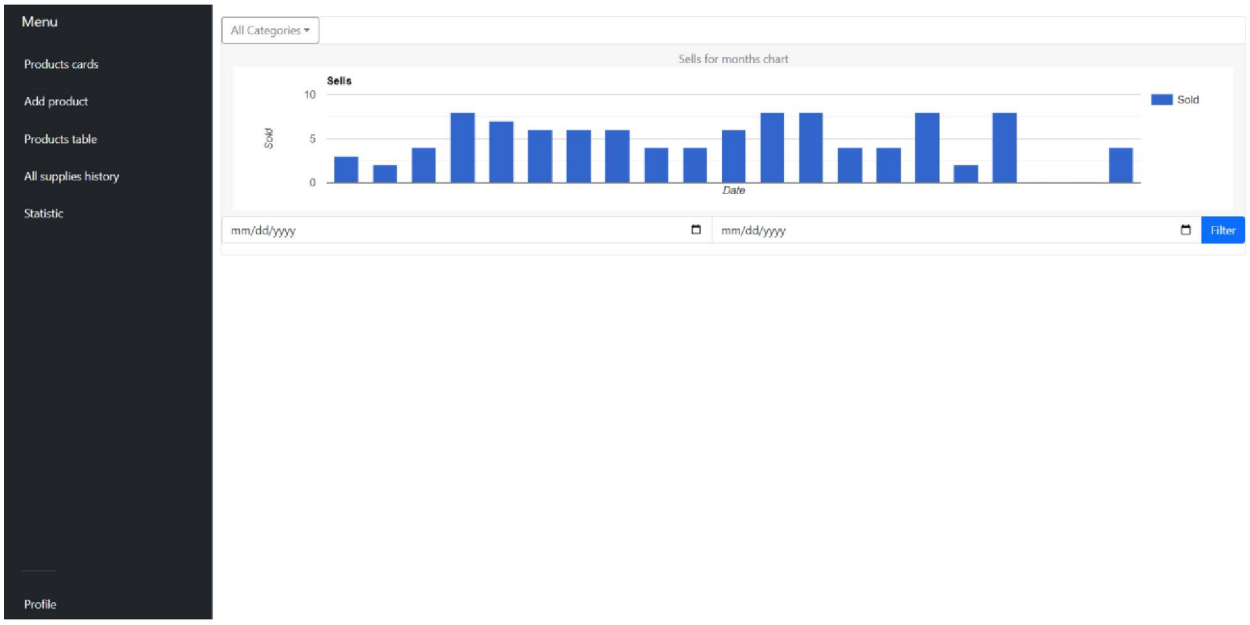


Рис Г.21 – Приклад діаграми продажів

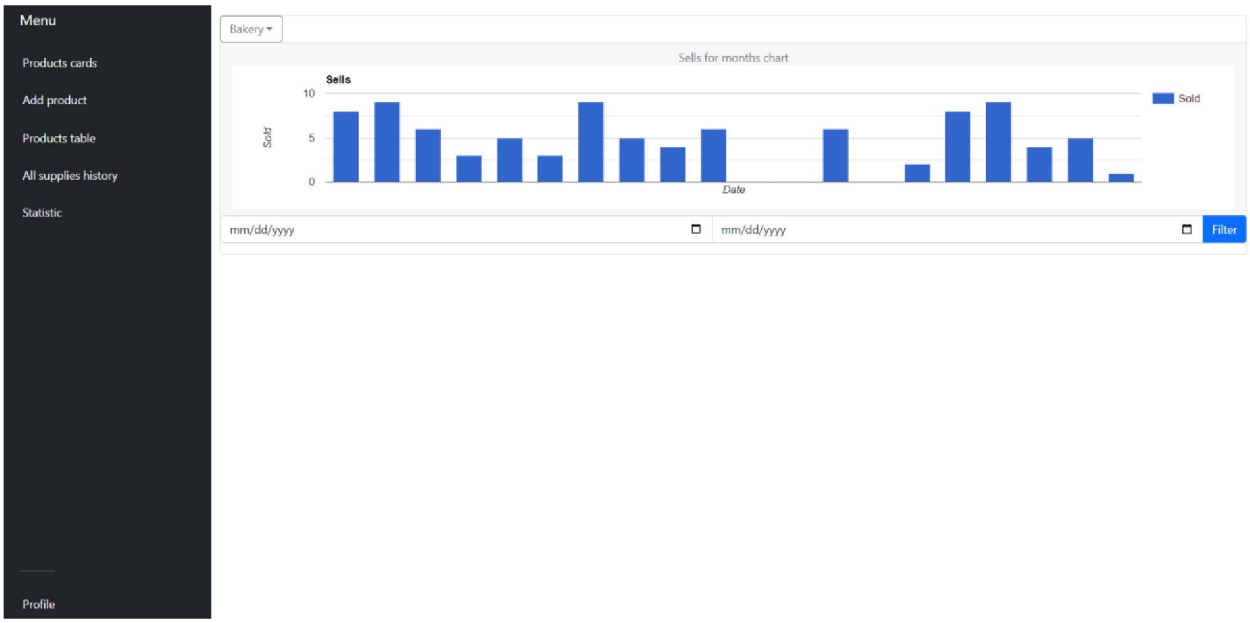


Рис Г.22 – Приклад діаграми продажів відповідної категорії

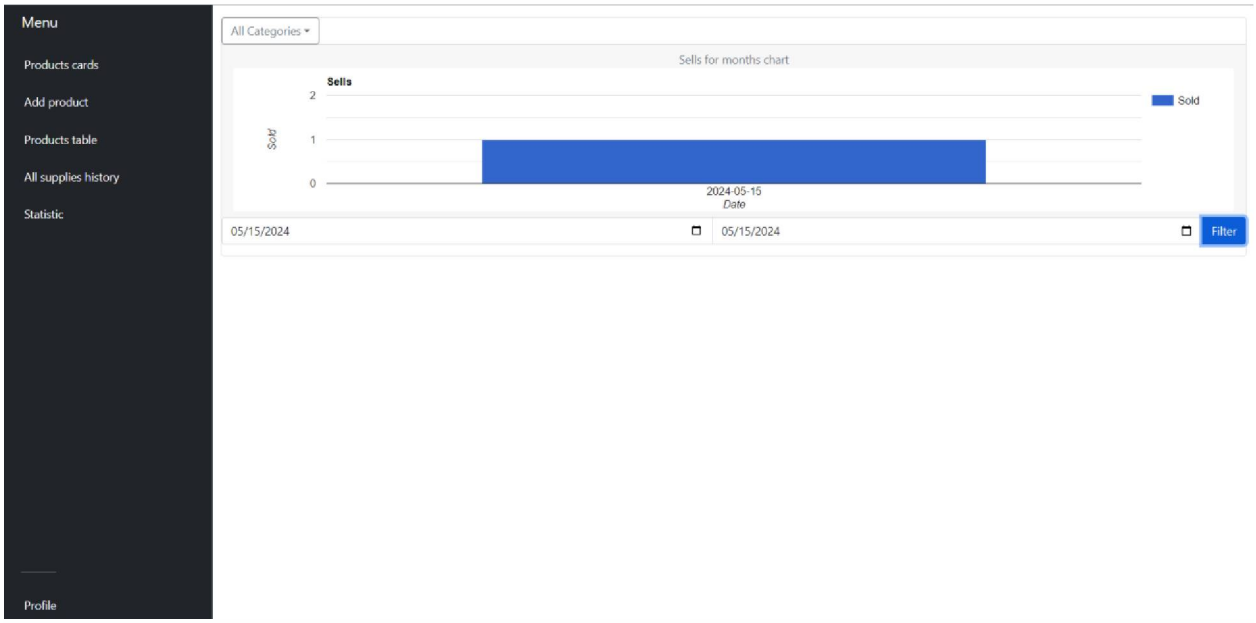


Рис Г.23 – Приклад діаграми продажів відповідно дат

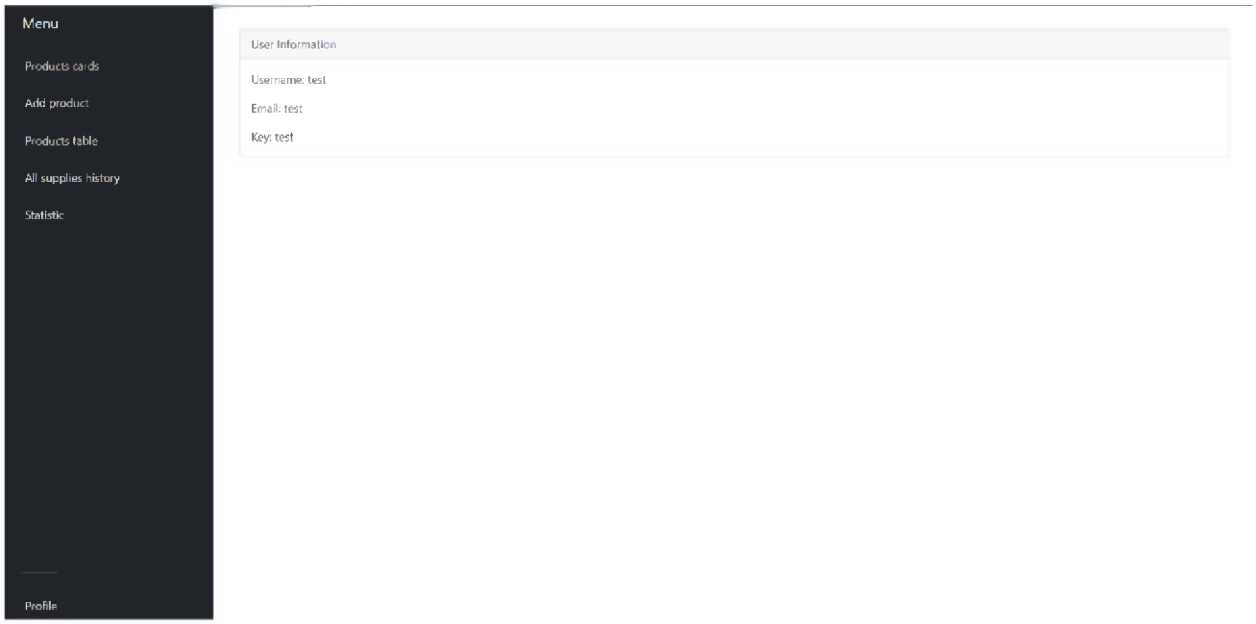


Рис Г.24 – Приклад профілю користувача