НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра автоматизованих систем обробки інформації та управління

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(повна назва кафедри, циклової комісії)

**КУРСОВА РОБОТА**

з                                           «Технології розробки ПЗ»

(назва дисципліни)

на тему:          «Склад: електронний облік товарів»

Студента 3 курсу ІП-51 групи

напряму підготовки 6.050103 «Програмна інженерія»

спеціальності «Програмне забезпечення систем»

Гладишка В. Ю.

(прізвище та ініціали)

Керівник Крамар Ю. М.

Доцент кафедри АСОІУ

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Члени комісії |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |
|  |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |

Київ ‑ 2018 рік

Національний технічний університет України “КПІ” ім. І. Сікорського

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна Технології розробки ПЗ

Напрям "Програмна інженерія"

Курс 3 Група ІП-51 Семестр 6

**ЗАВДАННЯ**

**на курсову роботу студента**

|  |
| --- |
| Гладишка Валентина Юрійовича |

(прізвище, ім’я, по батькові)

|  |  |
| --- | --- |
| 1. Тема роботи | Склад: електронний облік товарів |
|  | |
|  | |

|  |  |
| --- | --- |
| 2. Строк здачі студентом закінченої роботи |  |

|  |  |
| --- | --- |
| 3. Вихідні дані до роботи | Спроектувати та реалізувати систему класів, в основу якої покладено |
| логічну структуру даних, наведену у варіанті, для накопичення та обробки даних домену | |
| відповідно варіанту курсової роботи. В якості типу застосування обрати застосування на основі | |
| технологій asp.net MVC. Код програмної системи має складатись не менш як з трьох частин. | |

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

|  |
| --- |
|  |
|  |
|  |
|  |

5. Перелік графічного матеріалу ( з точним зазначенням обов’язкових креслень )

|  |
| --- |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| 6. Дата видачі завдання |  |

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Підписи керівника, студента |
| 1. | Отримання теми курсової роботи | 14.03.18-26.03.18 |  |
| 2. | Підготовка ТЗ | 26.03.18-03.04.18 |  |
| 3. | Пошук та вивчення літератури з питань курсової роботи | 04.04.18-17.04.18 |  |
| 4. | Розробка алгоритму вирішення задачі | 18.04.18-24.04.18 |  |
| 6. | Узгодження алгоритму з керівником | 18.04.18-24.04.18 |  |
| 5. | Розробка сценарію роботи програми | 25.04.18-01.05.18 |  |
| 6. | Узгодження сценарію роботи програми з керівником | 25.04.18-01.05.18 |  |
| 7. | Узгодження з керівником інтерфейсу користувача | 25.04.18-01.05.18 |  |
| 8. | Розробка програмного забезпечення | 02.05.18-22.05.18 |  |
| 10. | Розробка та налагодження інтерфейсної частини програми | 23.05.18-29.05.18 |  |
| 11. | Тестування програми | 30.05.18-03.06.18 |  |
| 13. | Підготовка пояснювальної записки | 30.05.18-03.06.18 |  |
| 14. | Здача курсової роботи на перевірку | 04.06.18-12.06.18 |  |
| 15. | Захист курсової роботи |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | Гладишко В. Ю. |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

|  |  |  |  |
| --- | --- | --- | --- |
| Керівник |  |  | Крамар Ю. М. |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 р.

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 20 сторінок, 13 рисунків, 4 посилання.

Об’єкт дослідження: веб-сайт для електронного обліку товарів на складі.

Мета роботи: навчитись створювати веб-сайти за допомогою ASP.NET MVC фреймворку, розробити тришарове програмне забазпечення (DAL, BLL, PL).

Виконана програмна реалізація механізми створення, додавання, редагування та видалення обєктів до бази даних. Також були додані можливості фільтрації даних за певними ознаками, реалізації валідації на стороні сервера, логічної перевірки даних. Розроблена клієнтська частина веб-сайту зі зрозумілим інтерфейсом для користувача.

ASP.NET MVC, БАГАТОШАРОВА АРХІТЕКТУРА, DAL, BLL, PL, BOOTSTRAP, AJAX, JQUERY

Зміст

[Вступ 5](#_Toc515984098)

[1 Постановка задачі 6](#_Toc515984099)

[2 Опис шарів проекту 7](#_Toc515984100)

[2.1. Шар доступу до даних (DAL) 7](#_Toc515984101)

[2.1.1. Моделі даних 7](#_Toc515984102)

[2.1.2. Класи, призначені для взаємодії з моделями даних 7](#_Toc515984103)

[2.2. Шар бізнес-логіки (DLL) 10](#_Toc515984104)

[2.2.1. Моделі даних 10](#_Toc515984105)

[2.2.2. Бізнес моделі 10](#_Toc515984106)

[2.2.3. Класи які описують бізнес-логіку 11](#_Toc515984107)

[2.3. Шар представлення (PL) 14](#_Toc515984108)

[2.3.1. Моделі 14](#_Toc515984109)

[2.3.2. Контролери 14](#_Toc515984110)

[2.3.3. Представлення 15](#_Toc515984111)

[Висновки 22](#_Toc515984112)

[Перелік посилань 23](#_Toc515984113)

Вступ

Платформа ASP.NET MVC представляє собою фреймворк для створення сайтів і веб-додатків за допомогою реалізації паттерна MVC.

Концепція шаблону MVC (model-view-controller) передбачає розділення додатку на три компоненти: представлення, контроллери, і моделі. Представлення служать для відображення користувацького інтерфейсу програми. Контроллер – клас, який обробляє дані, отримані з представлення і відправляє їх до сховища (бази даних). Модель – клас, який відображає структуру даних, які використовуються додатком.

Використання цього фреймворка доречне, коли треба створити веб-сайт, який має тісний зв'язок з базою даних (постійно додає, змінює та видаляє дані), а також значна частина функціоналу якого має виконуватись на сервері. Привабливим бонусом є те, що серверна частина відокремлена від користувацької, тому можна без проблем використовувати сторонні бібліотеки для покращення зовнішнього вигляду додатку, такі як bootstrap, JQuery, AJAX та інші. Написання серверної частини на мові С# дає змогу використовувати багатомодульність проекту, застосовувати різні бібліотеки, не повязані з веб-розробкою, а також покривати проект модульними тестами.

Мені потрібно було зробити веб-сайт для введення обліку товарів на складі.

Для створення такого додатку потрібна база даних, яка вміє зберігати великі обсяги даних до 100000 записів в таблиці. Також потрібно врахувати що в базу даних будуть регулярно вноситись зміни. Тому мені було необхідно також створити іструменти для додавання, оновлення та видалення даних. Саме тому я обрав цю технологію.

Метою моєї роботи є створення веб-сайту, який дав би змогу користувачеві повністю контролювати діяльність складу. А саме, реалізувати:

1. створення, редагування, видалення, сортування та пошук інформації про товари та можливість додавати та видаляти товари з категорії;
2. створення, редагування та видалення інформації про категорії;
3. створення, редагування та видалення інформації про постачальників;
4. створення, редагування та видалення інформації про замовників;

# Постановка задачі

Спроектувати та реалізувати систему класів, в основу якої покладено логічну структуру даних, наведену у варіанті, для накопичення та обробки даних домену відповідно варіанту курсової роботи. Варіант: «Склад: електронний облік товарів».

Структура програмної системи курсового проекту:

Код програмної системи має складатись не менш як з трьох частин відповідно до багатошарової архітектури системи, де шарами архітектури є:

1. Шар доступу до даних (DAL) організувати таким чином, щоб він забезпечував збереження даних у БД із застосуванням Entity Framework. Тип програмного модулю – DLL. У проектуванні шару використати шаблони Репозиторій та Одиниця роботи.
2. В шарі бізнес-логіки (BLL) побудувати класи, що представляють об’єкти та дії над ними відповідно до предметної області. Якщо дані потрібно отримати зі сховища даних або зберегти у сховищі, шар бізнес-логіки звертається до шару доступу до даних, передаючи\отримуючи збережені дані. Тип програмного модулю – DLL.
3. Шар представлення (PL) – це додаток типу відповідного до обраного інтерфейсу (asp.net MVC чи web API). Для виконання дій на вимогу користувача шар представлення звертається до шару бізнес-логіки, передаючи в нього ведені користувачем дані та команди виконати певні дії. Результати виконаних дій, отримані від шару бізнес-логіки, шар представлення відображує користувачу у представленнях (web-сторінках). Тип програмного модулю – web-застосування.

Реалізувати обов’язковий контроль коректності введених даних в шарі представлення.

Реалізувати перевірку виняткових ситуацій в шарі бізнес-логіки та доступу до даних; в тому числі – при роботі з даними. При необхідності створити власні класи виключень.

# Опис шарів проекту

## Шар доступу до даних (DAL)

Даний шар програми представлений DLL модулем (Storage.DAL.dll).

Він містить у собі усі моделі даних, які необхідно зберігати в БД, а також класи, призначені для взаємодії з цими даними.

### Моделі даних

В шарі представлені такі моделі даних:

1. Товар (Product). Містить інформацію про назву, бренд, категорію, ціну за одиницю товару, та кількість товару на складі.
2. Категорія (Category). Містить інформацію про назву категорії.
3. Постачальник (Provider). Містить інформацію про ім’я та фамілію постачальника.
4. Замовник (Customer). Містить інформацію про ім’я та фамілію замовника.
5. Продаж (Sale). Містить інформацію про назву товару, ім’я та фамілію замовника, дату продажу, та кількість проданого товару.
6. Поставка (Supply). Містить інформацію про ціну товару за одиницю, назву та кількість товару, ім’я та фамілію постачальника, дату поставки.

### Класи, призначені для взаємодії з моделями даних

Були використані шаблони Репозиторій та Одиниця роботи. Таким чином були розроблені такі класи:

1. IRepository. Інтерфейс, призначений для роботи з різними репозиторіями проекту. Містить у собі методи додавання, зміни, пошуку, видалення, отримання даних та збереження змін у базі.
2. ProductRepository, CategoryRepository, ProviderRepository, CustomerRepository, SaleRepository, SupplyRepository. Класи, які наслідуюють IRepository та призначені для роботи з логічно відокремленою частиною бази даних.
3. UnitOfWork. Клас, через який буде відбуватись уся взаємодія з базою даних. Містить у собі усі репозиторії проекту.

Підключення до бази даних відбувається за допомогою класу StorageContext. Він повязує усі моделі з сутностями в базі даних. Для генерації початкових даних був створений клас StorageDbInitializer, який реаліує інтерфейс DropCreateDatabaseAlways<StorageContext>. В методі Seed() на початку роботи програми відбувається наповнення бази даних випадковими початковими даними.

Надаю схему класів шару доступу до даних (Рисунок 2.1 – 2.3).

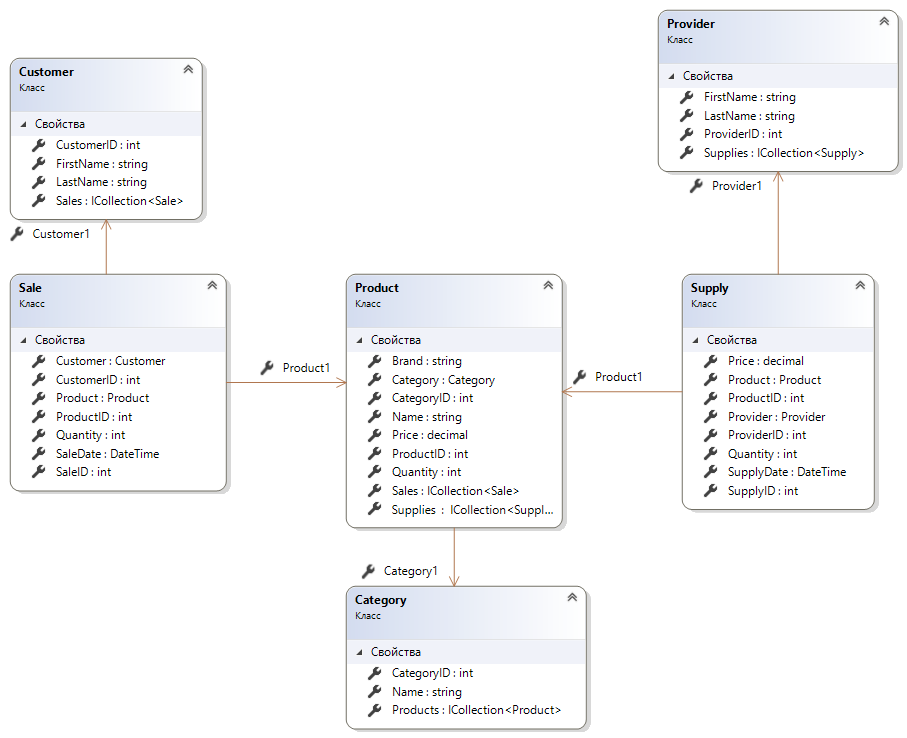


Рисунок 2.1 – Схема моделей даних Storage.DAL.dll

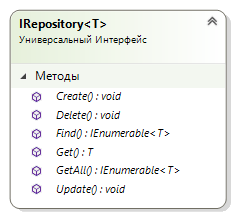


Рисунок 2.2 – Схема інтерфейсів Storage.DAL.dll

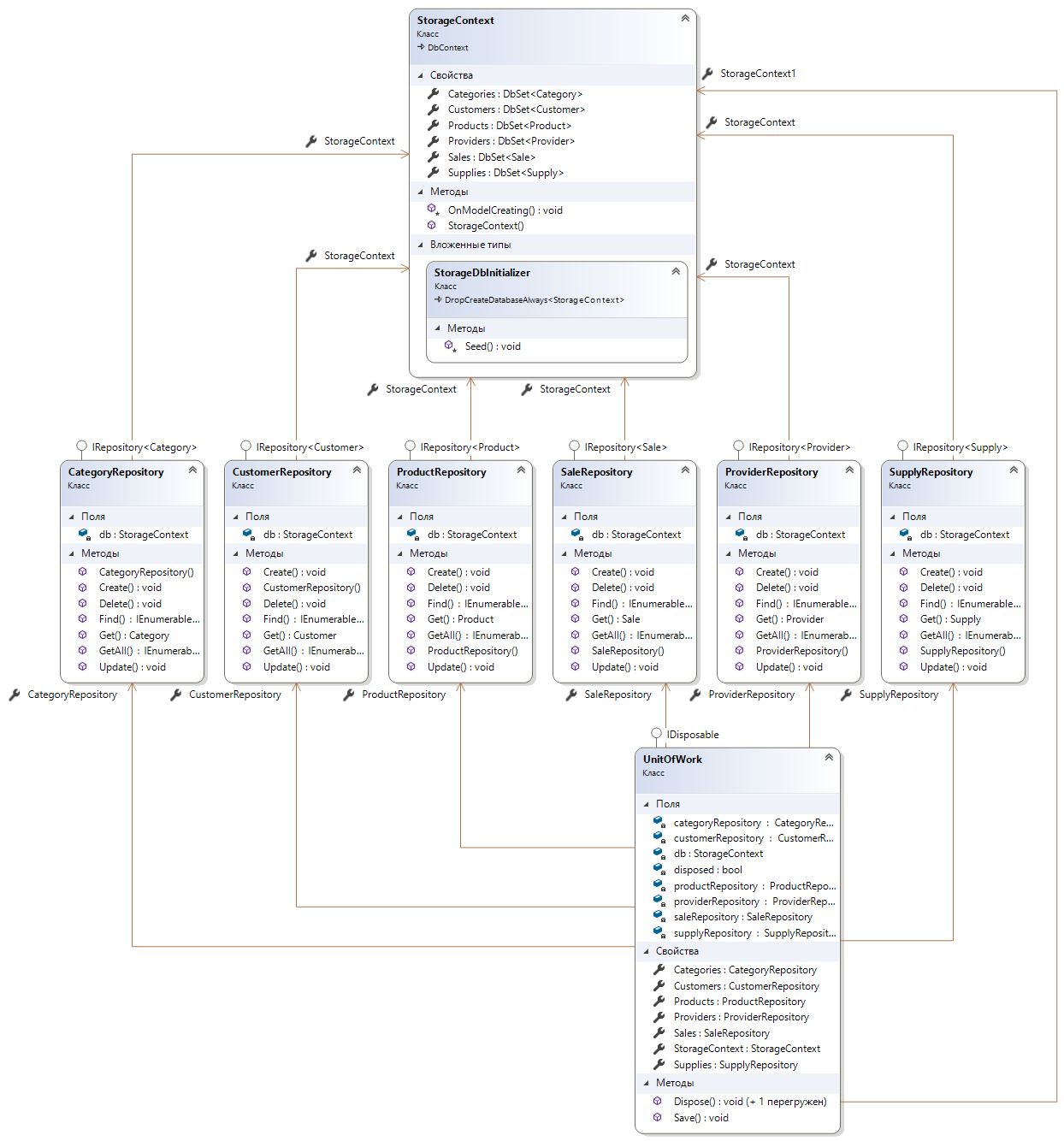


Рисунок 2.3 – Схема класів Storage.DAL.dll

## Шар бізнес-логіки (DLL)

Даний шар програми представлений DLL модулем (Storage.BLL.dll).

Він містить у собі моделі даних, інструменти для виконання операцій над цими даними та бізнес моделі, які описують додаткову бізнес-логіку. Цей шар призначений для для виконання логіки додатку, обчислень та внутрішніх дій, спрощення використання додатку, та організації взаємодії між шаром доступу до даних та шаром представлення.

### Моделі даних

Моделі даних, як проміжні сутності, повністю копіюють моделі з шару DAL та також містять методи для перетворення даних.

### Бізнес моделі

Бізнес моделі це додаткові класи, які містять вибіркову інформацію з моделей даних та методи для аналізу цих даних.

Були створені такі бізнес моделі:

1. SaleProduct. Містить стислу інформацію про продажі по товарам: назву товару, сумарну кількість та ціну для кожного товару та методи для отримання цих даних.
2. SaleCustomer. Містить стислу інформацію про продажі по замовникам: ім’я та прізвище замовника, кількість куплених товарів, сумарну ціну та методи для отримання цих даних.
3. SupplyProduct. Містить стислу інформацію про поставки по товарам: назву товару, сумарну кількість та ціну для кожного товару та методи для отримання цих даних.
4. SupplyProvider. Містить стислу інформацію про поставки по постачальникам: ім’я та прізвище постачальника, кількість поставленних товарів, сумарну ціну та методи для отримання цих даних.

### Класи які описують бізнес-логіку

Для реалізації бізнес-логіку проекту були створені «сервіси» - класи, які містять у собі методи для обробки даних, отриманих від користувача, перед їх додаванням у базу, спрощення даних, отриманих з бази, пред передачею їх до користувача. Також був створений клас виключень.

Були створені такі сервіси:

1. ServiceCategories. Містить методи для отримання даних про категорії, отримання інформації про товари в певній категорії, створення та редагування категорії, її видалення. Сервіс не дає користувачу видалити категорію якщо вона містить хоча б один товар.
2. ServiceCustomers. Містить методи для отримання інформації про замовників, отримання інформації про покупки певного замовника, методи додавання, редагування та видалення замовника. Додатково містить методи для пошуку та сортування замовників по імені та фамілії.
3. ServiceProducts. Містить методи отримання інформації про товари, методи додавання, редагування та видалення товару, додавання та видалення товару з категорії. Додатково містить метод пошуку товару по його назві, бренду, категорії, а також метод для сортування товарів за назвою, ціною та брендом.
4. ServiceProviders. Містить методи для отримання інформації про постачальників, отримання інформації про поставки певного постачальника, методи додавання, редагування та видалення постачальника. Додатково містить методи для пошуку та сортування постачальників по імені та фамілії.
5. ServiceSales. Містить методи для отримання інформації про продажі, методи додавання, редагування та видалення продажу. Додатково містить методи для групування даних по товарам або замовникам.
6. ServiceSupplies. Містить методи для отримання інформації про поставки, методи додавання, редагування та видалення поставки. Додатково містить методи для групування даних по товарам або постачальникам.

Надаю схему класів шару бізнес-логіки (Рисунок 2.4 – 2.6).



Рисунок 2.4 – Схема моделей даних Storage.BLL.dll

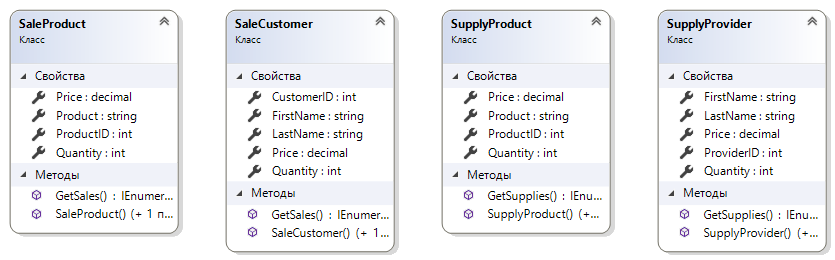


Рисунок 2.5 – Схема бізнес моделей Storage.BLL.dll

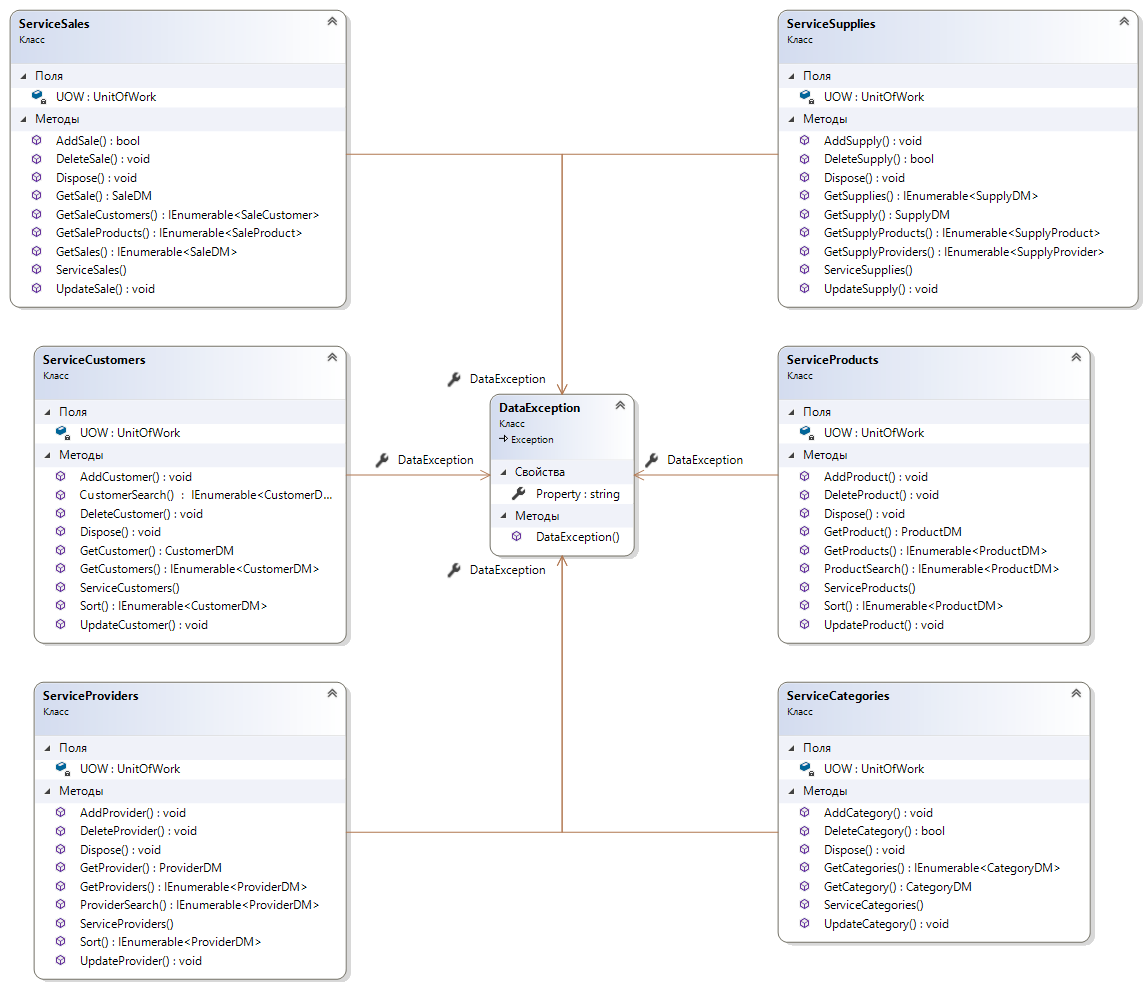


Рисунок 2.6 – Схема класів Storage.BLL.dll

## Шар представлення (PL)

Даний шар програми представлений ASP.NET MVC модулем (Storage.Web).

Він містить у собі моделі представлення, контролери та представлення які реалізують інтерфейс додатку, видимий користувачу. А також містить функціонал, який забезпечує валідацію даних, додавання, редагування та видалення даних з бази.

### Моделі

Моделі, копіюють моделі з шару BLL. Додатково в моделях прописані умови валідації даних, їх необхідність, відображення на сайті.

### Контролери

Шар містить 6 контролерів, кожен з яких відповідає за свою логічну частину програми.

1. ProductsController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про товари, пошук товарів за назвою, брендом та категорією. Також реалізована можливість відсортувати товари за назвою, ціною та брендом. Для своїх дій використовує об’єкти типу ServiceProducts, ServiceCategories.
2. CategoriesController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про категорію. Також реалізована можливість відображення списку товарів в певній категорії. Для своїх дій використовує об’єкт типу ServiceCategories.
3. ProvidersController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про постачальника, пошук та сортування постачальників за ім’ям та фамілією. Також реалізована можливість відображення списку поставок певного постачальника. Для своїх дій використовує об’єкт типу ServiceProviders.
4. CustomersController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про замовника, пошук та сортування замовників за ім’ям та фамілією. Також реалізована можливість відображення списку покупок певного замовника. Для своїх дій використовує об’єкт типу ServiceCustomers.
5. SalesController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про продаж. Також реалізована можливість відображення продажів згрупованих по товарам або замовникам. Для своїх дій використовує об’єкти типу ServiceSales, ServiceProducts, ServiceCustomers.
6. SuppliesController. Містить у собі методи реалізації додавання, редагування, видалення та відображення повної інформації про поставку. Також реалізована можливість відображення поставок згрупованих по товарам або постачальникам. Для своїх дій використовує об’єкти типу ServiceSupplies, ServiceProducts, ServiceProviders.

### Представлення

Шар містить представлення, які відповідають кожному методу контролера, розбиті по папкам з назвою контролера.

1. Products. Містить у собі представлення Index (використовує ProductView), Create (використовує ProductEdit), Edit (використовує ProductEdit), Details (використовує ProductView), Delete (використовує ProductView), GetProducts (використовує ProductView).
2. Categories. Містить у собі представлення Index (використовує CategoryView), Create (використовує CategoryView), Edit (використовує CategoryView), Details (використовує CategoryView), Delete (використовує CategoryView).
3. Providers. Містить у собі представлення Index (використовує ProviderView), Create (використовує ProviderView), Edit (використовує ProviderView), Details (використовує ProviderView), Delete (використовує ProviderView), GetProviders (використовує ProviderView).
4. Customers. Містить у собі представлення Index (використовує CustomerView), Create (використовує CustomerView), Edit (використовує CustomerView), Details (використовує CustomerView), Delete (використовує CustomerView), GetCustomer (використовує CustomerView).
5. Sales. Містить у собі представлення Index (використовує SaleView), Create (використовує SaleEdit), Edit (використовує SaleEdit), Details (використовує SaleView), Delete (використовує SaleView), SaleProducts (використовує SaleProductView), SaleCustomers (використовує SaleCustomerView).
6. Supplies. Містить у собі представлення Index (використовує SupplyView), Create (використовує SupplyEdit), Edit (використовує SupplyEdit), Details (використовує SupplyView), Delete (використовує SupplyView), SupplyProducts (використовує SupplyProductView), SupplyProviders (використовує SupplyProviderView).

Надаю схему класів шару представлення (Рисунок 2.5 – 2.8).



Рисунок 2.5 – Схема класів Storage.Web

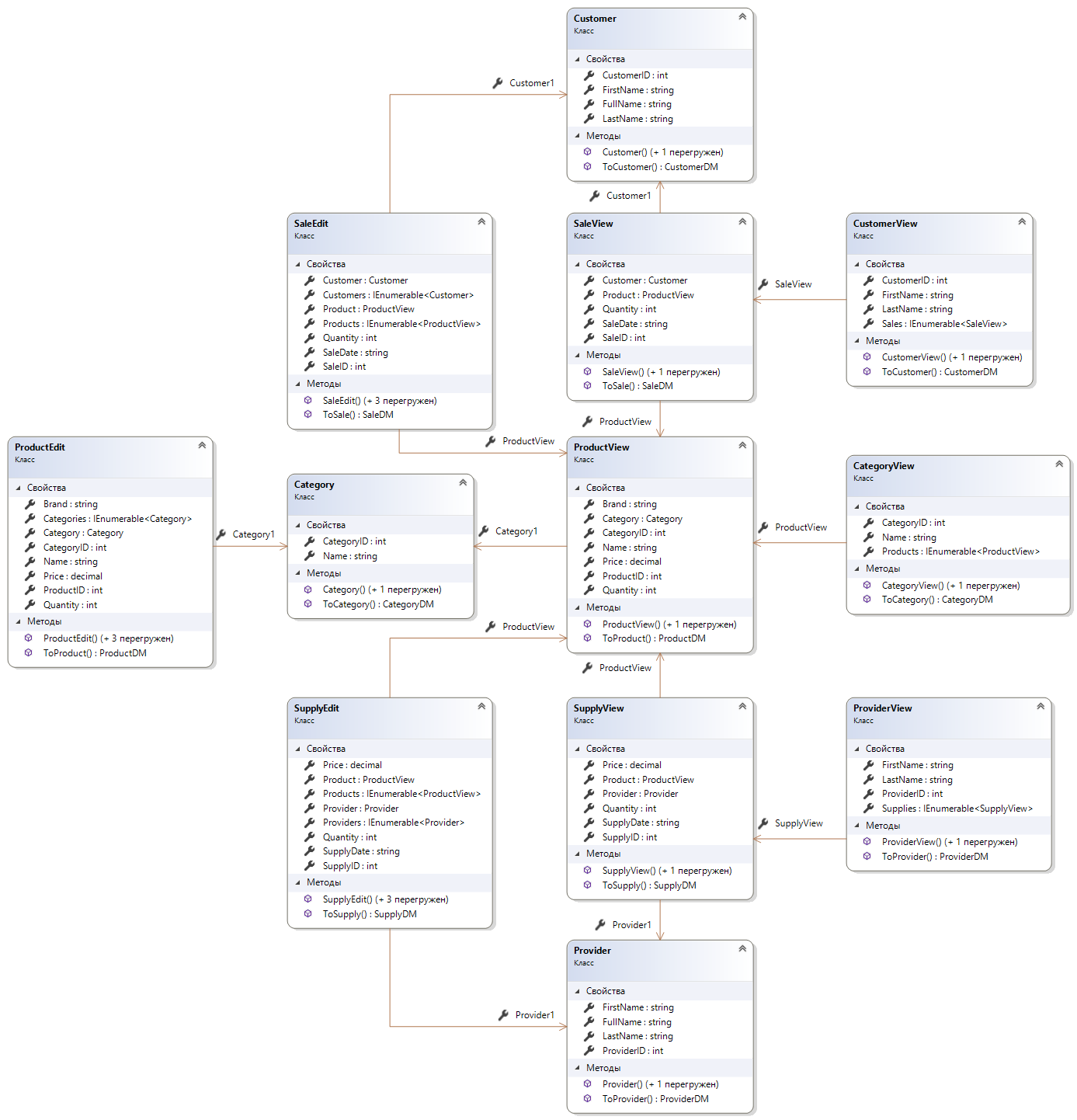


Рисунок 2.6 – Схема моделей представлення Storage.Web

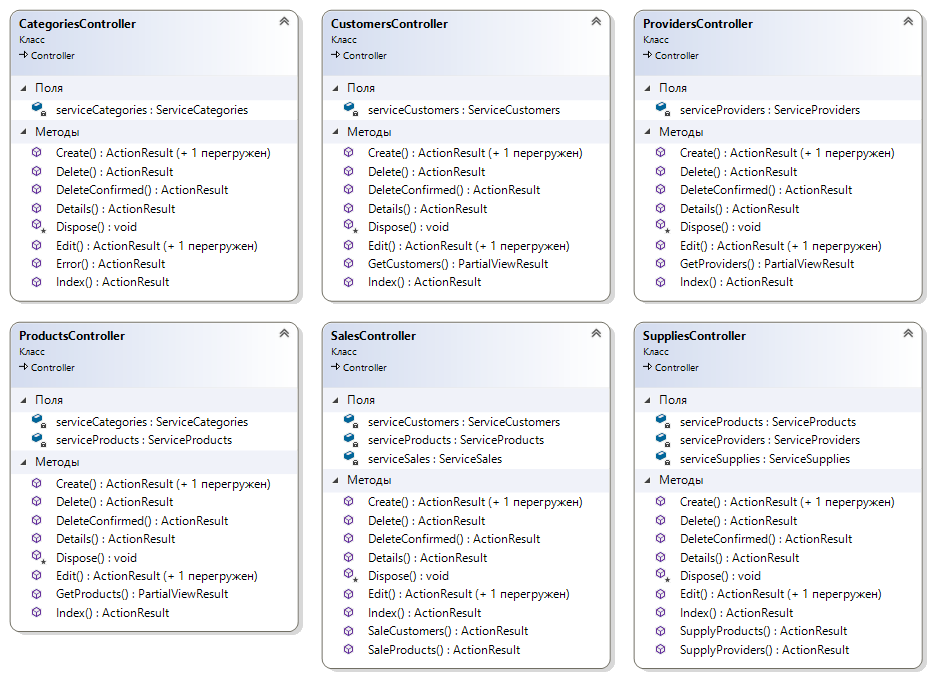


Рисунок 2.7 – Схема контролерів Storage.Web

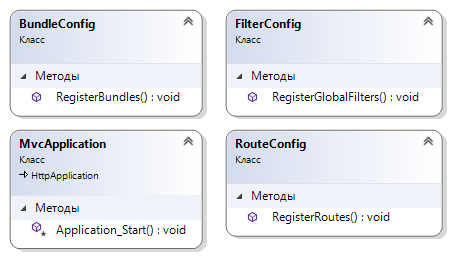


Рисунок 2.8 – Схема класів конфігурації Storage.Web

Нижче представлені приклади роботи програми (Рисунок 2.9 – 2.13).

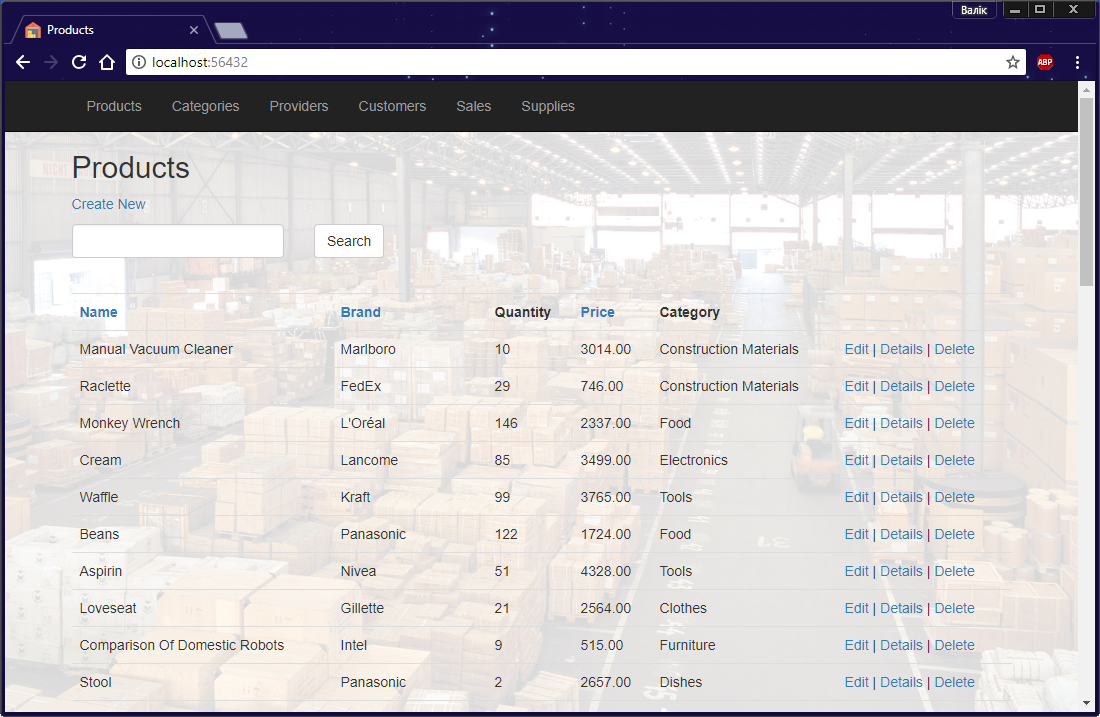


Рисунок 2.9 – Перегляд всіх товарів

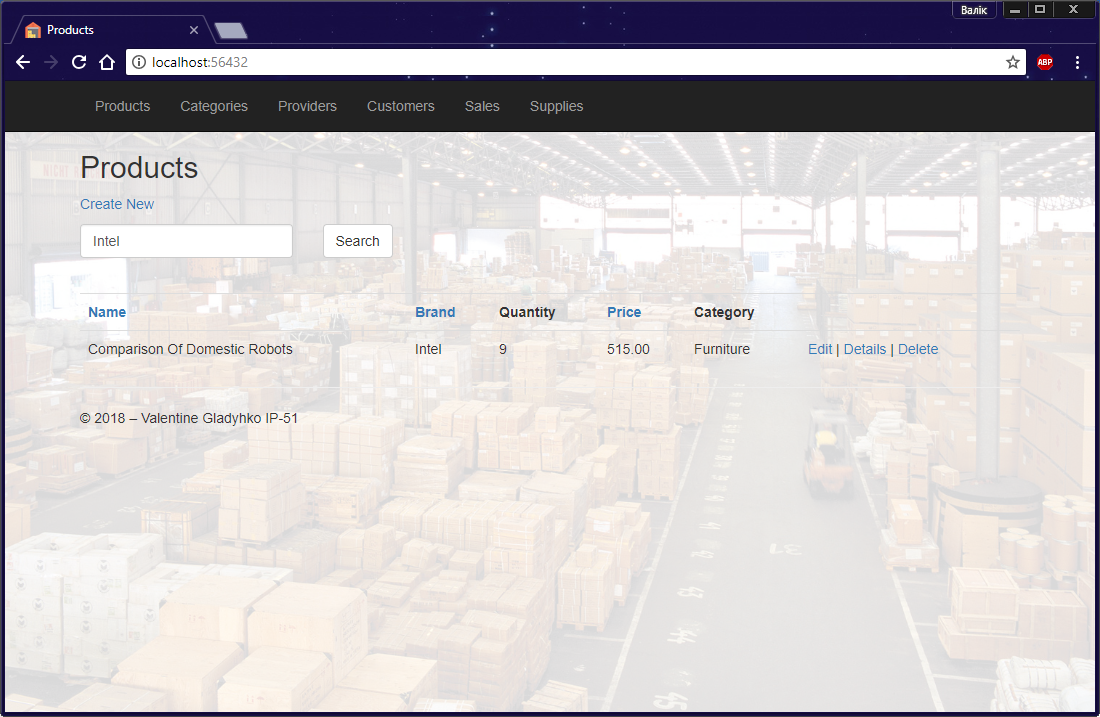


Рисунок 2.10 – Пошук продуктів по слову “Intel”

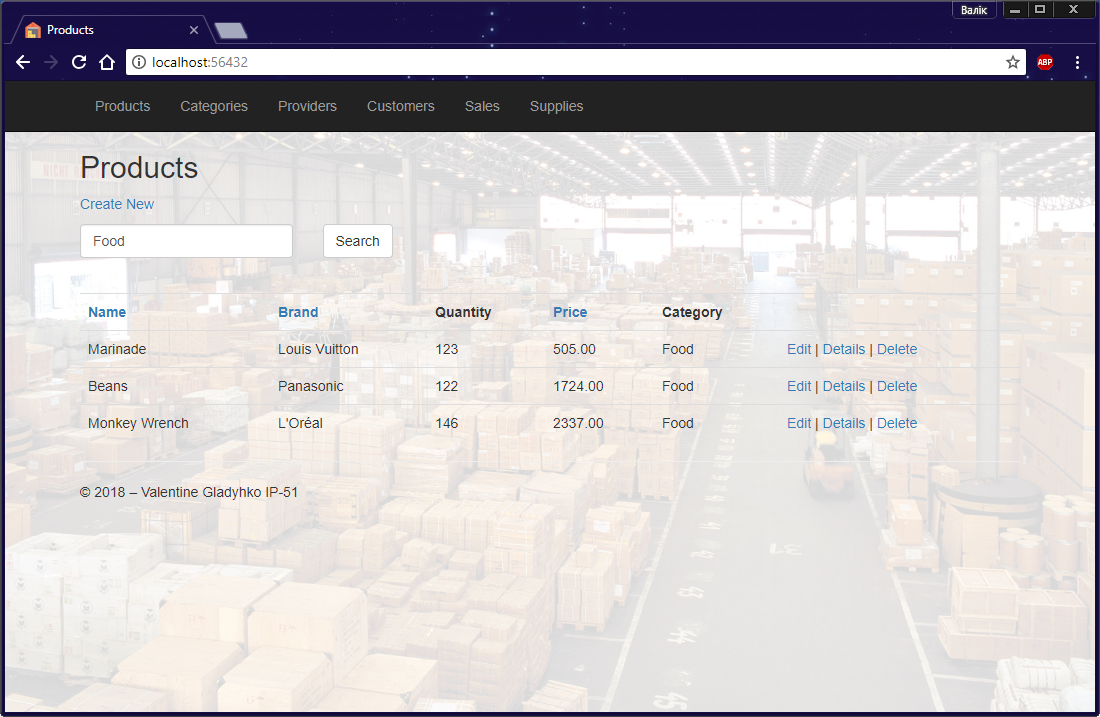


Рисунок 2.11 – Список товарів відсортованих по ціні

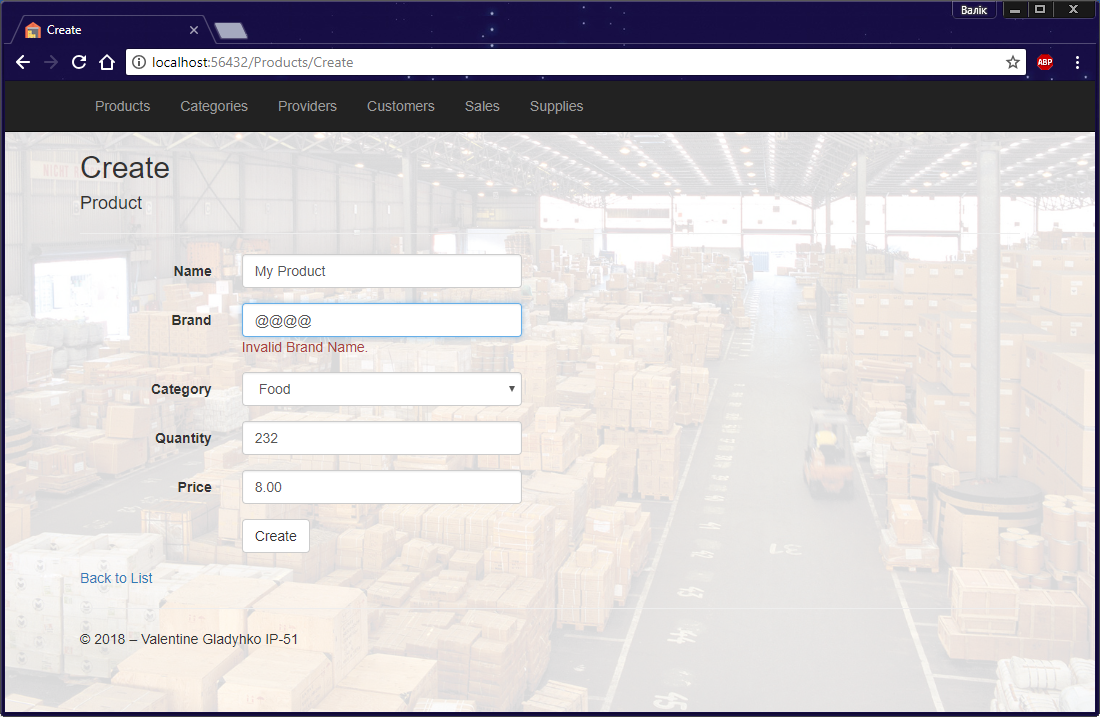


Рисунок 2.12 – Створення продукту, валідація

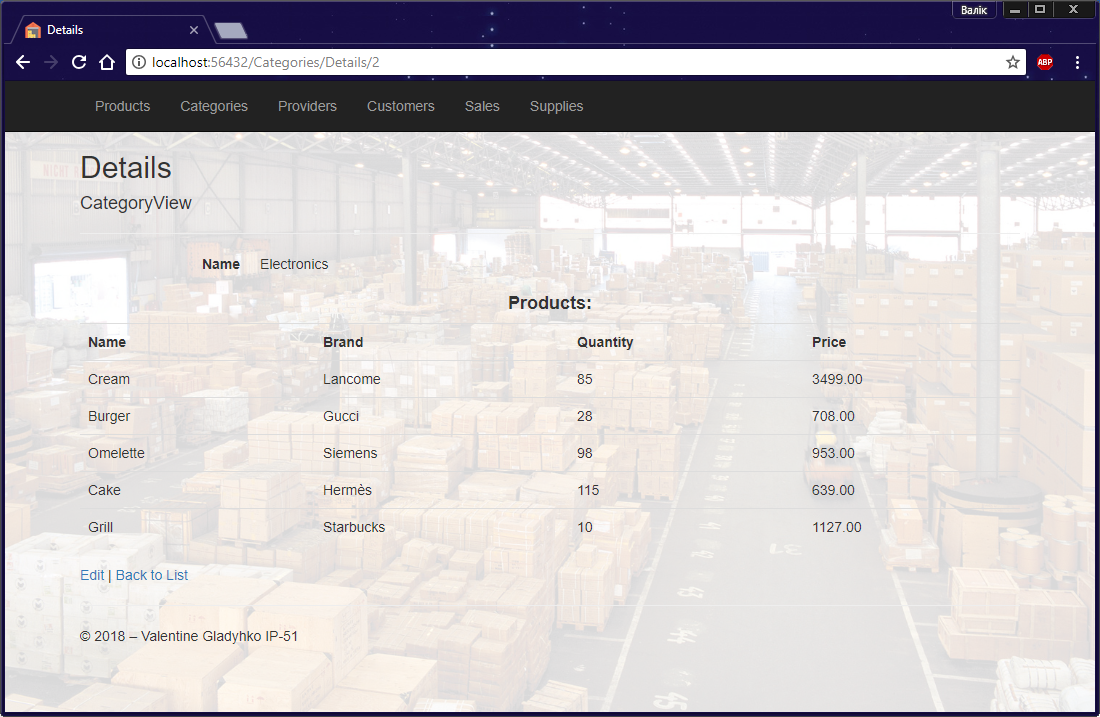


Рисунок 2.13 – список товарів в категорії “Electronics”

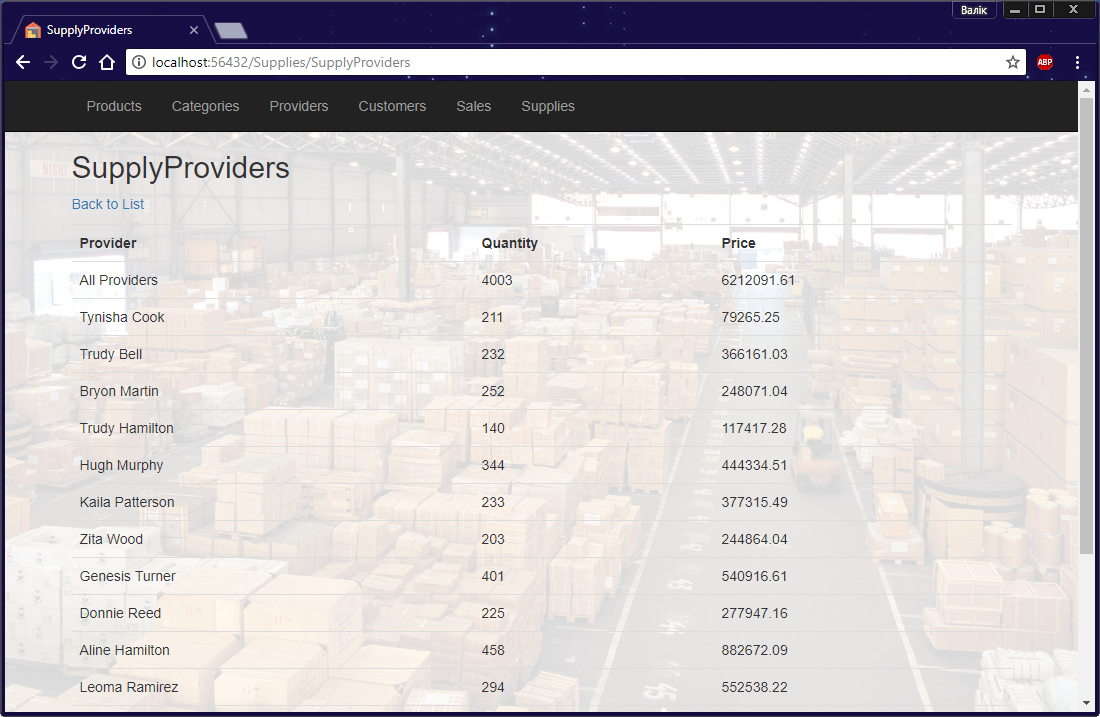


Рисунок 2.14 – список поставок згрупованих по постачальникам

Висновки

На етапі вивчення теоретичних основ завдання даної курсової роботи було розглянуто способи розробки веб-додатку на мові C#. Були досліджені шаблон Model-View-Controller (Модель-Представлення-Контролер) та особливості його реалізації в ASP.NET MVC фреймворку, способи використання баз даних у проекті, особливості розробки багатошарового програмного забезпечення, а також взаємодію програмних шарів між собою.

На етапі проектування програмного забезпечення реалізовано механізми створення, додавання, редагування та видалення об’єктів до бази даних. Також були додані можливості пошуку та сортування даних за певними ознаками, валідації на стороні сервера та логічної перевірки даних.

Для взаємодії з користувачем був розроблений зрозумілий графічний інтерфейс. Для його розробки були застосовані навички розробки веб-сторінок на основі движка Razor (C#, HTML, JS, CSS).

На етапі тестування було визначено, що програма вірно обробляє всі вхідні дані та видає очікуваний результат.

Результатом курсової роботи є створений веб-додаток, який може використовувати склад для полегшення обліку товарів на складі. Такий додаток дає можливість швидкого пошуку повної інформації про певний товар, певного постачальника чи замовника.

Перелік посилань

1. Простір імен Microsoft.SqlServer.Server .NET Framework [Електронний ресурс] – Режим доступу

<https://msdn.microsoft.com/ru-ru/library/microsoft.sqlserver.server(v=vs.110).aspx>

1. Простір імен System.Web.Mvc [Електронний ресурс] – Режим доступу <https://msdn.microsoft.com/ru-ru/library/system.web.mvc(v=vs.118).aspx>
2. Бібліотека CSS стилів Bootstrap 3 [Електронний ресурс] – Режим доступу <https://www.w3schools.com/bootstrap/>
3. Контролери [Електронний ресурс] – Режим доступу <https://metanit.com/sharp/mvc5/3.1.php>
4. Представлення [Електронний ресурс] – Режим доступу <https://metanit.com/sharp/mvc5/4.1.php>
5. Моделі [Електронний ресурс] – Режим доступу <https://metanit.com/sharp/mvc5/5.1.php>
6. JavaScript і AJAX [Електронний ресурс] – Режим доступу <https://metanit.com/sharp/mvc5/10.1.php>
7. Репозиторій проекту (GitHub) [Електронний ресурс] – Режим доступу <https://github.com/ValentineGladyshko/KPI-Csharp-labs/tree/master/KPI-Semester-6/Coursework>