**Форма № Н–9.02**

Національний університет «Полтавська політехніка імені Юрія Кондратюка»

(повне найменування вищого навчального закладу)

Навчально науковий інститут інформаційних технологій та робототехніки

(повна назва факультету)

Кафедра комп’ютерних та інформаційних технологій і систем

(повна назва кафедри)

**Пояснювальна записка**

**до дипломного проекту (роботи)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_бакалавра\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(освітньо–кваліфікаційний рівень)

на тему \_\_\_\_\_\_\_Розроблення інформаційної системи інтернет–магазину автозапчастин

Виконав: студент 4 курсу, групи 402–ТН

спеціальності

122 Комп’ютерні науки

(шифр і назва напряму)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Маслов А. В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Керівник \_\_\_\_\_\_\_\_\_\_\_Деркач Т. М.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Полтава – 2021 року

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**«ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

**НАВЧАЛЬНО НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ**

**КАФЕДРА КОМП’ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ**

**КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА**

**спеціальність 122 «Комп’ютерні науки»**

**на тему**

**«Розроблення інформаційної системи інтернет–магазину автозапчастин»**

**Студента групи 402–ТН Маслова Антона Валерійовича**

Керівник роботи

кандидат технічних наук,

доцент Деркач Т.М.

в.о.завідувача кафедри

кандидат технічних наук,

доцент Головко Г.В.

Полтава – 2021

# РЕФЕРАТ

Кваліфікаційна робота бакалавра: 73с.,19 рисунків, 2 додатки, 18 джерел.

**Об’єктдослідження**:розроблення інформаційної системи інтернет торгівлі.

**Метароботи**: розроблення веб-ресурсу для Інтернет торгівлі автозапчастинами.

**Методи**: проектування та розробка бази даних для функціонування інформаційної системи Інтернет торгівлі, програмна реалізація інтерфейсу користувача та розробка інтерфейсу для адміністрування веб-ресурсу.

**Ключові слова**: автоматизована інформаційна система, модуль, модель, база даних.

**ANNOTATION**

Qualification work of bachelor’s degree: 73 p., 19 drawings, 2 appendices, 18 sources.

**Object of research**: the activity of the information system of e–commerce.

**Purpose**: development of a web resource for Internet trade in auto parts.

**Methods**: design and development of a database for the functioning of the Internet commerce information system, software implementation of the user interface and development of an interface for web resource administration.

**Keywords**: automated information system, module, model, database.

**ЗМІСТ**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ…………………6

[ВСТУП 6](#_Toc74032974)

[РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД ІНФОРМАЦІЙНОЇ СИСТЕМИ ІНТЕРНЕТ-МАГАЗИНУ АВТОЗАПЧАСТИН 8](#_Toc74032975)

[1.1 Опис предметної області 8](#_Toc74032976)

[1.2 Огляд аналогів веб-ресурсів з торгівлі та доставки 11](#_Toc74032977)

[1.3 Огляд технологій розробки веб-ресурсів 15](#_Toc74032978)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ ІНФОРМАЦІНОЇ СИСТЕМ 23](#_Toc74032979)

[2.1 Інструментальні засоби проектування 23](#_Toc74032980)

[2.2 Вибір інструментів та фреймворків для розробки проекту 24](#_Toc74032981)

[2.3 Засоби розробки інтерфейсу 28](#_Toc74032982)

[2.4 Проектування бази даних 30](#_Toc74032983)

[2.5 Проектування інформаційної системи діяльності інтернет торгівлі 34](#_Toc74032984)

[РОЗДІЛ 3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ ПРОДАЖУ АВТОЗАПЧАСТИН 46](#_Toc74032985)

[3.1 Користувацький інтерфейс 46](#_Toc74032986)

[3.2 Адміністративна панель 51](#_Toc74032987)

[РОЗДІЛ 4 ТЕСТУВАННЯ 55](#_Toc74032988)

[4.1 Функціональне тестування 55](#_Toc74032989)

[4.2 Мобільне тестування 57](#_Toc74032990)

[ВИСНОВКИ 59](#_Toc74032991)

[CПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 60](#_Toc74032992)

[ДОДАТОК А ЛІСТИНГ КОДУ ДЛЯ РЕАЛІЗАЦІЇ ДОМАШНЬОЇ СТОРІНКИ 62](#_Toc74032993)

[ДОДАТОК Б ФРАГМЕНТ ЛІСТИНГУ КОДУ ДЛЯ РЕАЛІЗАЦІЇ СТИЛЕЙ ПРОЕКТУ69](#_Toc74032994)

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

**ER–діаграма** – Entity-relationship діаграма

**UML** – Unified Modeling Language

**JS** – Java Script

**БД** – база даних

**ВР** – веб-ресурс

**ІС** – інформаційна система

**CMS** – Система управління контентом

**СУБД** – Система управління базами даних

**ООП** – об’єктно–орієнтоване програмування

**SSMS** –S QL Server Management Studio

**QA** – quality assurance тестування

# ВСТУП

В даний час потреба у фізичних пунктах з продажу товарів є абсолютно засторілою моделью бізнес комерції. З появою глобальної мережі розвиток електроної комерції з кожним днем здобуває стрімкого розвитку. В наш час майже кожен володіє компютером чи смартфоном і в більшості з них є доступ в інтернет. Споживач під час комунікації з інтернетом здійснює незаплановані покупки.Істотним є і той факт, що Інтернет стає зручним і досить дешевим «торговим майданчиком». Сучасні підприємці розуміючи сьогодені тенденції розміщують свою продукцію у онлайн середовищі.

Сьогодні існує два види способу продажу через глобальну мережу перший це розміщення товарів на так званих «маркетплейсах» або ж створення інформаційної системи під власну категорію товару, що неодмінно має велику кількість переваг порівнянно з попереднім способом.

У кваліфікаційнійроботі роботі поставлена задача – розробити веб-ресурс для Інтернет торгівлі автозапчастин.Вочевидь стала потреба в розробленні декількох інтерфейсів: для відвідувача інтренет-­магазину так і для адміністратора. Відповідно до повноважень користувача буде наданий різний доступ. Редагувати, створювати видаляти товар, обробляти замовлення має право лише адмінінстратор інформаційної системи. Інтерфейс для адміністрування інформаційної системи має бути строгим та чітким у оформленні, так як головною метою цього інтерфейсу є обслуговування та обік продажів інформаційної системи.

Що до користувацької частини інтерфейсу вона має містити фірмовий дизайн підприемства виконанний у певній кольоровій палітрі. Головним завданням інтерфейсу є те що він має бути зручним для користувача і надавати усі можливості для вибору потрібного товару та безпосередньо для створення замовлення.

# РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД ІНФОРМАЦІЙНИХ СИСТЕМ ІНТЕРНЕТ-ТОРГІВЛІ АВТОЗАПЧАСТИН

## **1.1 Опис предметної області**

Інформаційна система інформаційна система інтернет магазину автозапчастин забезпечує продаж, автоматизований підбір автозапчастин, різноманітні методи сортування. Система спрямована на вдосконалення продажу автомобільних товарів в глобальній мережі.

Інформаційна система має задовольняти потреби користувача, а також аналіз потреб клієнта та бути маркетинг-орієнтованою на потенційних користувачів.

З кожним днем поширення автомобілів серед людей зростає, актуальність створення даної інформаційної системи зумовлена постійним технічним обслуговуванням автомобілів які постійно ламаються.

Відмінність європейських країн від України у відношенні до технічного обслуговування автомобіля полягає в тому що українські водії зацікавлені в тому щоб самостійно ремонтувати та обслуговувати свій транспортний засіб на відміну від європейців які звикли звертатися до спеціалізованих станцій технічного обслуговування, де слюсарі професіонали займаються профілактикою ремонтом та підбором автозапчастин відповідно.

Інформаційна система інтернет магазину автозапчастин має бути доступна та зрозуміла як фахівцям професіональних станцій технічного обслуговування так і простим водіям, у яких виникла потреба самостійно налагодити свій транспортний засіб. Доступність термінології складових автозапчастин має бути різноманітною, тому що дійсні технічні назви деталей досвідчені автомобілісти можуть не використовувати так як за часи автомобільного буму більшість назв деталей набувала народних назв які так чи інакше витісняли оригінальну термінологію. Так як головною метою розробки інформаційної системі є доступність, елементи каталогу товарів повинні мати декілька назв для зручності пошуку товарів.

Інформаційна система має надавати такі можливості як:

* Вибір товару та подальше оформлення заказу за допомогою «Price»-листів та каталогу товару. Ця можливість є необхідною для функціональної частини інформаційна система.
* Оплата товару через глобальну мережу за допомогою відомих платіжних систем. Від обсягу різноманіття платіжних систем буде залежати зручність вибору найоптимальнішої для покупця.
* Обробка замовлення в реальному часі. Ця можливість повинна надати інформацію потенційному покупцю що до наявності товару на складі в реальному часі, та виходячи з характеристик товару та адреси доставки клієнта обраховувати час доставки та її вартість.
* Реєстрація користувача інформаційної системи. Можливість зареєструватися дає користувачу такі переваги як особистий кабінет в я кому можна переглядати замовлення. Таким чином поставник автозапчастин може розповсюджувати адресну рекламну та актуальну інформацію або ж проводити маркетингові дослідження.
* Автентифікація та авторизація користувачів. Ця система повинна перевіряти користувачів на відповідність до їх облікових записів.
* Надавати потенційному покупцеві повну інформацію про авто товари (назва, виробник, матеріал, розмір, вагу деталей, інформацію про марку автомобіля).
* Наявність розширеної системи пошуку. Пошук за категоріями (марка, виробник, ціна) і за назвою, шляхом введення частини слова або цілком.
* Надавати клієнту всю необхідну інформацію для здійснення покупки в Інтернет–магазині.
* Надавати потенційному покупцеві можливість задавати питання по роботі Інтернет-магазину. Користувач інформаційної системи повинен мати змогу поставити питання якщо йому щось буде не зрозуміло у користуванні сайтом. Онлайн підтримка відповість на всі питання потенційного покупця та проконсультує у випадку виникнення проблем.
* Наявність швидкої і зручної системи замовлення авто продукції.
* Мати кілька різних способів оплати продукції. Під час оформлення замовлення товару, покупцю має надаватися вибір яким чином він бажає провести оплату. Це може бути оплата товару одразу через платіжні системи або ж оплата при отримані у відділені пошти.
* Наявність системи знижок для постійних покупців. Особистий кабінет користувача має містити систему лояльності, таким чином надавати бонуси або знижки покупцям які є постійними користувачами інформаційної системи.
* Наявність розділу з історією замовлень. В даному розділу користувач може бачити всі свої попередні замовлення або ж відстежувати стан доставки актуальних.
* Можливість залишати відгуки про товар. Покупець повинен мати змогу поділитися досвідом про придбання тієї чи іншої автозапчастини. Після публікації відгуку клієнт буде мати зворотній зв'язок від менеджерів інформаційна система інтернет магазину.
* Мати систему відновлення пароля. Ця функція є необхідною в такому випадку якщо користувач не пам’ятатиме пароль від облікового запису. Наявність цієї функції дозволяє відновити доступ до запису та всіх даних записаних у ньому.
* Можливість підписки на новини Інтернет-магазину. За згодою користувача він може отримувати останні новини що до товару інформаційна система інтернет магазину зручним для нього способом. Це може бути розсилка по електронній пошті або за допомогою СМС повідомлень на номер телефону користувача.
* Політика конфіденційності. Забезпечення конфіденційності про особисті дані покупців і інформацію про їх замовлення.
* Можливість редагування інформації про товари. Менеджер інтернет–магазину буде мати доступ до додавання, видалення і редагування характеристики автозапчастин.
* Доступ до інформації про покупця. Цими правами буде володіти виключно менеджер інтернет-магазину за для формування товарних накладних або до формування спеціальних пропозицій.
* Можливість відповідати на запитання покупців. Менеджер має доступ до онлайн чату в якому він веде діалог з користувачами інформаційної системи у яких виникають питання.

Вимоги до дизайну та інтерфейсу інформаційної системи мають бути сучасними. Проблема інших інформаційних систем автозапчастин полягає в тому що вони орієнтуються виключно на чоловіків. Про це говорить оформлення каталогів товару, кольорова палітра яка переважно всюди синього кольору. У наш час жінки є повнозначними водіями які також доглядають та обслуговують власній автомобіль тому тема сайту має бути універсальною та привабливою для жінок так само як і для чоловіків.

## **1.2 Огляд аналогів веб**-**ресурсів з торгівлі та доставки**

Інтернет–магазини автозапчастин є досить популярними так як з розвитком інформаційних технологій покупка деталей до свого авто на авторинках стає не актуальним через це водії надають перевагу придбанню комплектуючих через інтернет. Інформаційних систем з продажу деталей значна кількість про те у всіх є недоліки та переваги, проаналізувавши які можна буде створити оптимальну інформаційну систему яка буде задовольняти усі потреби потенційних покупців.

**Інтернет–магазин «elit.ua»**

Відмінний інтернет-магазин, який містить величезну різноманітність товарів різних постачальників. Даний інтернет-магазин має наступні переваги:

– величезний вибір з наданих товарів;

– можливість замовлення поштовою посилкою;

– докладний опис товарів;

– різні види оплати;

– підтримка знижок;

– наявність онлайн–аукціону;

До недоліків даного сайту можна віднести:

– не привабливий дизайн;

– погана адаптивність;

– перенасичення непотрібною корпоративною інформацією;

– малий розмір шрифтів

Програмна реалізація даної інформаційної системи за допомогою мови гіпертекстової розмітки HTML та каскадних таблиць стилів CSS та мови програмування Java Script.

**інтернет–магазин «autohim.com.ua»**

Даний магазин володіє наступними перевагами:

– великий вибір автозапчастин;

– можливість порівняння цін від різних фірм;

– можливість проконсультуватися

До недоліків можна віднести застарілий дизайн сайту та погану адаптивність під смартфони.

Програмна реалізація даної інформаційно системи за допомогою мови гіпертекстової розмітки HTML та каскадних таблиць стилів CSS та мови програмування Java Script.

**інтернет–магазин «avtoskorost.com.ua»**

Інтернет–магазин з можливістю доставки продуктів додому. Інтернет–магазин має наступні переваги:

– величезний вибір з наданих автозапчастин;

– можливість замовлення поштовою посилкою;

– докладний опис товарів;

– різні види оплати;

– цілодобова доставка.

До недоліків даного сайту належать:

– складність навігації по каталогам і результатів пошуку;

– відсутність можливостей залишати коментарі.

Інформаційна система зроблена на основі CMS системи WordPress це передбачає свої певні недоліки такі як простоту в дизайні іншими словами її не унікальність та малий функціонал.

Виходячи з перерахованих недоліків, а також переваг існуючих аналогів необхідно реалізувати програмний засіб, що володіє наступними властивостями:

– зручний призначений для користувача дизайн;

– можливість обмеження функціональності користувача в залежності від ролі;

– легкість в освоєнні сайту;

– простота реєстрації;

– простота навігації по сайту;

– простота управління сайтом для адміністратора.

Сторінки з описом товару повинні бути найбільш значущі в інтернет-магазині. Тому повинні містити вичерпну інформацію, достатню, щоб розвіяти сумніви покупця в необхідності здійснення покупки: назва товару, ємний правдивий опис, якісні картинки, гарантії, короткий опис умов доставки, оплати і повернення, обміну. Розглянуті інтернет–магазини бояться згадувати на сайті про можливе повернення або обмін товару, вважаючи, що подібна інформація може збільшити кількість повернень, підштовхуючи покупця на невигідне для них дію. Але на практиці справа йде трохи інакше: зростає не кількість повернень, а кількість продажів. Заява про те, що в магазині проводиться обмін і повернення куплених товарів, показує, що тут впевнені в своїй продукції та сервісі.

Речі інтуїтивно зрозумілі, але, як показує практика, багато про них забувають або вважають необов'язковими. Наявність такої інформації, як прямий міський номер телефон, є обов'язковим. Присутність тільки мобільних номерів недостатньо. Також обов'язкова наявність адреси офісу або пункту видачі, куди клієнт може приїхати. На сторінці «Контакти» повинна бути розміщена карта сайту із зазначенням найближчих орієнтирів. Рекомендую розмістити інтерактивну карту від пошукової систем Google, створену за допомогою спеціального конструктора. Найпростіше це зробити за допомогою конструктора карт Google. Наявність такої інформації позитивно відіб'ється і на довірі користувачів, і на «довірі» пошукових систем.

Також не варто забувати про правдиву актуальну інформацію про ціни та наявність товарів. Не повинно бути такого, щоб біля товарної позиції стояла ціна «0.00» або написаний текст такого змісту: «Щоб дізнатися ціну зв'яжіться з менеджером». На кожен товар має бути встановлена конкретна фактична ціна. Також слід прагнути до того, щоб інформація про наявність товарів на сайті відповідала дійсності.

Наявність позитивних відгуків як на самому сайті, так і на сторонніх ресурсах сприятливо позначається на ставлення до інтернет-магазину і часто стає вирішальним фактором при прийнятті рішення про здійснення покупки.

## **1.3 Огляд технологій розробки веб**-**ресурсів**

Чи слід розробляти веб-сайт електронної комерції з нуля або використовувати існуюче рішення? Тривають суперечки щодо найкращого підходу для бізнесу та того, який варіант має найбільший фінансовий сенс. Обидві сторони мають дійсні бали. Для одних підприємстввикористанняіснуючихінформаційнихсистем з продажу є більш розумнимрішенням, тоді як для інших розробка на замовлення є єдиним розумним варіантом. То що до моєї інформаційної системи? Щоб дізнатись, почнемо із питань які я собі поставив для знаходження відповіді на це питання.

* Чи потрібно моєму веб-сайту обробляти тисячі або навіть мільйони користувачів?
* Чи потрібно моєму веб-сайту витримувати великі навантаження та забезпечувати швидку роботу?
* Чи вимагає мій проект унікальних можливостей?

Відповідь на всі ці запитання – так, це означає що створення інтернет-магазину на інших фрейм ворках буде не доцільною. Створимо веб-сайт електронної комерції за допомогою Python. Щоб зрозуміти, чому, проаналізуємо розробку електронної комерції за допомогою Python у світлі трьох важливих факторів: масштабованості, продуктивності та унікальних функцій.

Для платформ електронної комерції масштабованість має значення. Роздрібні торговці прагнуть залучити нових клієнтів та збільшити продажі. І коли кількість клієнтів зростає, платформа також повинна зростати.

За допомогою спеціальної розробки можна масштабувати свій проект двома способами: вертикально або горизонтально.

Вертикальне масштабування – це коли додаємо більше потужності своїй платформі шляхом оновлення інфраструктури проекту. Щоб масштабувати платформу вертикально, додаємо до своєї платформи більше серверів баз даних, серверів кешування та веб-серверів. Django пропонує механізми для цього за замовчуванням.

Горизонтальне масштабування – це масштабування примірників Django. Іншими словами, додаємо більше машин. Горизонтальне масштабування не є обов’язковим; застосовується, коли вертикального масштабування недостатньо.

Під час масштабування важливо правильно налаштувати ситуацію з самого початку. Аналізувати кількість користувачів потрібно с самого початку. Роблячи це, можемо визначити, який метод масштабування підходить для програми інформаційної системи.

Вертикальне масштабування є найбільш економічно вигідним, оскільки є можливість заплатити Amazon Web Services (AWS), щоб додати більше серверів, і надати розробнику можливість для їх підтримки. При горизонтальному масштабуванні платформи потрібно платити за сервери та фактичну розробку програмного забезпечення[13].

Масштабування на платформах електронної комерції дещо інше. Наприклад, за допомогою Shopify існує можливість запустити веб-сайт з будь-яким планом – Basic Shopify, Shopify, Advanced Shopify – і масштабувати свій магазин, змінюючи свій план.



Рисунок 1.1 – Ціна за підписку Shopify на місяць

“Gastronom” Оскільки платформа буде зростати, існує можливість перейти на Shopify Plus, масштабоване рішення для підприємств. Цей корпоративний план починається приблизно з 2000 доларів на місяць, а це означає, що вартість такого проекту буде щонайменше 24000 доларів на рік.

За допомогою Django можна створити інформаційну систему з високою продуктивністю з самого початку. Існує безліч методів та інструментів Django, які можуть допомогти змусити код працювати ефективніше, швидше та використовувати менше системних ресурсів. Ці інструменти спрямовані на оптимізацію кешування, продуктивності бази даних та продуктивності HTTP.

Крім того, Django пропонує такі інструменти, як django-debug-toolbar, щоб отримати уявлення про те, де у коді є неефективність, щоб знати, що потрібно оптимізувати.

Офіційна документацію від Django Software Foundation. Там можна знайти огляд інструментів і технік Django для покращення продуктивності інтернет–магазину та мінімізації споживання пам'яті[14].

Всесвітньо відомі компанії використовують Python вже багато років і є доказом його продуктивності та масштабованості. Торгові платформи, такі як Wish та Instacart, – чудові приклади. До некомерційних веб-сайтів, які використовують Python / Django, належать Disqus, Instagram, Pinterest, YouTube, Mozilla, The Washington Post, Spotify, Eventbrite та Dropbox.

Висока продуктивність – це не їх сильна сторона. Основною причиною цього є розширення сторонніх розробників. Завдяки фрейм воркам, швидше за все, знадобляться сторонні розширення, щоб додати необхідну функціональність. Проблема в тому, що вони займають багато місця, що збільшує час навантаження і знижує продуктивність. Отже, чим більше сторонніх додатків має веб-сайт, тим гірша його ефективність.

Проаналізуємо швидкість відгуку веб сторінки. Згідно з Littledata, середня швидкість завантаження сторінки Magento становить 2,8 секунди, тоді як для Shopify – 2,6 секунди. У великих електронних магазинах час завантаження повільніший. У середньому для повного завантаження магазину Magento потрібно 6 секунд, а магазину Shopify – 6,8 секунди.

У деяких випадках використання спеціальна розробка може бути не найкращим варіантом. Розглянемо деякі випадки, коли має сенс використовувати платформу електронної комерції для розвитку Інтернет-магазину.

Готова платформа може бути єдиним варіантом, якщо є ціль побудувати магазин без фактичного кодування. Однак це можливо лише для невеликих веб-сайтів з невеликою функціональністю та без налаштувань.

За допомогою готового програмного забезпечення можна створити просту інформаційну систему без знань кодування, але знадобиться кодування будь-яких розширень, преміум-шаблонів та налаштувань. Візьмемо Shopify як приклад. Це дає можливість контролювати свої проекти, дозволяючи редагувати HTML та CSS. Для цього потрібно мати принаймні базові навички HTML / CSS.

Більшість фрейм ворків пропонують необхідну функціональність для створення веб-сайту електронної комерції та управління ним. Наприклад, BigCommerce має систему управління вмістом (CMS), функціональність платежів, управління замовленнями та запасами, виконання та відвантаження замовлення, знижки, управління поверненням та інтеграцію додатків. Завдяки OpenCart власники магазинів можуть скористатися необмеженою кількістю товарів і категорій, рейтингами та відгуками, мультивалютною та багатомовною підтримкою, інформаційною панеллю адміністратора, інтеграцією платіжних шлюзів, інтеграцією доставки та звітами про продажі.

Готові рішення пропонують досить багато шаблонів (як безкоштовних, так і платних), як правило, з обмеженими можливостями їх налаштування.

Примітка: Часто для налаштування теми потрібні технічні навички. Потрібне базове редагування вашого макета без HTML / CSS.

Якщо гроші є проблемою, то можна обрати створення інформаційної системи за допомогою фрейм ворка. Спочатку не доведеться вкладати занадто багато коштів для побудови платформи. Але кожного разу, коли потрібно буде додати функціональність, застосувати доповнення або налаштувати функції, потрібно буде платити.

Потрібно розуміти, що готова платформа не завжди коштує дешево. Візьмемо Shopify як приклад. Часто потрібно покластися на сторонні програми, щоб додати функціональність, а це означає, що потрібно буде платити додаткові збори.

Python – це рішення для цілком власної розробки, тобто при розробці веб–сайту є можливість вирішувати все, починаючи від дизайну і закінчуючи функціоналом. За допомогою Python можна створити масштабну та високопродуктивну програму. Розвиток може сильно зашкодити бюджету, але це одноразова інвестиція такого розміру. У майбутньому також виникнуть певні витрати, коли буде вирішено впровадити нові функції та ввести зміни [12].

Натомість із розробниками електронної комерції ви платите менше за налаштування веб-сайту, але потім продовжуєте платити за його використання.

Сьогодні готові платформи є дуже настроюваними та включають обов’язкові функції платформи електронної комерції. Але якщо потрібно буде створити масштабовану платформу з готовим рішенням, є велика ймовірність, що заплатимо більше, ніж мали б, якби розробляли з нуля.

SEO (Search Engine Optimization, пошукова оптимізація) – це всебічний розвиток і просування сайту для його виходу на перші позиції в результатах видачі пошукових систем (SERPs) по вибраних запитах з метою збільшення відвідуваності і подальшого отримання доходу [9].

Незважаючи на те, що SEO має найвищий рентабельність інвестицій серед будь-якої маркетингової кампанії електронної комерції, більшість інтернет-магазинів складаються разом, майже не враховуючи пошукових систем.

SEO, з іншого боку, вимагає лише зусиль заздалегідь – як тільки налаштувати інтернет-магазин, інформаційна система практично здійснює продажі на автопілоті без постійних витрат.

Безкоштовний, повторюваний трафік з високою конверсією. Це те, що потрібно досягти в останньому етапі створення інтернет-магазину.

Основні тактики, які слід включити у стратегію SEO щодо електронної комерції

Найкраща стратегія SEO для електронної комерції включає:

* Дослідження ключових слів для пошуку типів ключових слів, які шукають клієнти.
* Архітектура сайту на основі дослідження ключових слів.
* SEO на сторінці за допомогою стратегічної оптимізації ключових слів у мета–тегах та вмісті.
* Технічне SEO, яке допомагає забезпечити ефективне сканування веб–сайтів пошуковими системами.
* Місцевий SEO для сприяння залученню місцевого органічного трафіку (якщо у вас є цегла та розчин).
* Контент–маркетинг для залучення додаткових органічних відвідувачів.
* Створення посилань, щоб допомогти покращити авторитет вашого веб–сайту.
* Вимірювання успіху SEO за допомогою таких інструментів, як Google Analytics та Ahrefs [6].

Оптимізація пошукової системи (SEO) – це наукове мистецтво оптимізувати свій веб–сайт за певними ключовими словами, щоб отримати вищий рейтинг у результатах пошуку, таких як Google.

Наукове мистецтво, оскільки, хоча багато відомо про технічні аспекти SEO, у цьому є творчий досвід користувача та дизайн.

Але оптимізація інформаційної системи, зрештою, означає одне: створення найкращого можливого результату для цільового ключового слова.

Мета Google полягає в ранжуванні результатів пошуку, які відповідають на всі запитання шукача, настільки добре, що їм не потрібно повертатися до Google для отримання іншої відповіді.

Забезпечте кращу взаємодію з користувачем через швидший веб–сайт, кращий мобільний досвід, інтуїтивній інтерфейс тощо.

Дослідження Outbrain показує, що пошук є основним фактором трафіку на тематичних сайтах, перевершуючи соціальні мережі більше ніж на 300%.

Крім того, дослідження SEMrush, що вивчало 13 вертикалей електронної комерції, показало, що у 5 (музика, книги, меблі, будинок і сад, електроніка) переважає органічний пошук, а на всі 13 органічних та прямих припадає 80% всього трафіку[7].

У результатах пошуку Google відображає кілька оголошень, після чого відображаються звичайні списки. Більшість кліків, однак, спрямовуються на органічні результати. (Звичайно, це залежить від кількості оголошень та ключового слова, але здебільшого це відповідає дійсності.)

І оскільки ~ 95% людей не проходять повз першу сторінку, досягнення вершини рядка – єдиний спосіб отримати реальні результати. (Знову ж таки, це різниться, але здебільшого справедливо.)

Ключове слово "наконечник кульової опори" отримує в середньому близько 11 000 пошуків на місяць, за даними сервісу **Ahrefs** (інструмент який аналізує запити користувачів глобальної мережі). Якщо припустити, що 35% цих кліків спрямовуються на перший результат (середнє значення за ключовими словами), рейтинг №1 за цим ключовим словом отримає 3850 кліків.

Тепер припустимо, що коефіцієнт конверсії 10% (досить низький для ключового слова з високим наміром покупця).

Рейтинг № 1 за цим ключовим словом призвів би до додаткових 385 продажів на місяць!

І це лише одне ключове слово. Більшість сторінок класифікуються за кількома ключовими словами, а більшість сайтів класифікують кілька сторінок.

Маючи гарно налаштовану інформаційну систему можна отримати тисячі додаткових продажів на місяць, лише трохи додаткових зусиль щодо SEO.

# РОЗДІЛ 2 ПРОЕКТУВАННЯ ІНФОРМАЦІНОЇ СИСТЕМ

## **2.1 Інструментальні засоби проектування**

Для проектування інформаційної системи продажів використовувалися діаграми нотації UML, а також CASE засіб Draw.io.

Уніфікована мова моделювання UML є загально цільовою мову графічного опису моделей, який розроблений для візуалізації, специфікації, проектування та документування компонентів ПЗ, взаємодій об'єктів системи, бізнес–процесів та інших систем. Щоб підтримувати процес моделювання програмних засобів використовуючи об'єктно–орієнтований підхід, організовувати взаємозв'язок концептуальних і програмних понять і відображати проблеми масштабування складних систем потрібен UML. Моделі графічної мови UML застосовуються на всіх етапах життєвого циклу програмного забезпечення, від бізнес–аналізу до супроводу системи. Різноманітні організації застосовують UML на свій розсуд в залежності використовуваних технологій або своїх проблемних областей [2].

Виходячи з головних функції методології об'єктно-орієнтованого аналізу і проектування відносно повна модель складної системи представляє собою відповідну кількість уявлень які пов’язані між собою, кожне з яких відповідно ілюструє структуру системи чи аспект поведінки. Ієрархічна побудова це принцип в якому моделі складних систем наказується розглядати процес побудови моделей на різних рівнях деталізації або абстрагування не виходячи за рамки фіксованих уявлень. Прийнято вважати, що мова складається з правил комбінування. Його відмінністю є те, що словник мови складається з графічних елементів. Таким чином що для кожного графічного символу є своя відповідна семантика, тому модель яку створював один розробник, може бути зрозуміла іншим однозначно, а також програмним засобом, інтерпретують UML. Словник мови UML включає три види блоків:

* суті – основні елементи моделі, так звані абстракції;
* відносини – частини, що зв'язують різні сутності;
* діаграми – блоки, представляють інтерес сутностей.

Таким чиномдіаграми UML дозволяють виконуватисупровід проектута облегшують процес розробки документації до програмної системи. UML може бути застосований під час життєвого циклу аналізу бізнес-систем і розробки додатків [2].

Для проектування бази даних веб-додатків було використано CASE засіб ERwin.

CASE засіб ERwin – це засіб який використовують для розробки структури бази даних. ERwin об’єднує графічний інтерфейс Windows та спеціальні інструменти які знадобляться для побудови ER-діаграм, також містить різноманітні редактори для створення фізичного і логічного опису моделі даних та прозору підтримку провідних реляційних СУБД. ERwin дозволяє проводити або створювати зворотне проектування баз даних.

Сімейство продуктів ERwin є набором засобів концептуального моделювання даних, що використовують метод IDEF1X. Завдяки ERwin реалізується проектування БД схеми, опис та генерацію її на мові цільової СУБД (DB2, Oracle, Microsoft SQL Server і ін.) І реверсний інжиніринг існуючої БД. ERwin існує в декількох конфігураціях, всі вони орієнтовані на найбільш поширені засоби розробки додатків 4GL. Інтегрується з популярними засобами розробки клієнтської частини додатків, що дозволяє автоматично генерувати код додатків [3].

## **2.2 Вибір інструментів та фреймворків для розробки проекту**

Для реалізації інтернет-магазину використовується веб-фреймворк Django. Django – безкоштовний і вільний фреймворк для web-додатків написаний на мові програмування Python. Фреймворк підтримується організацією Django Software Foundation. Розглянемо технології розробки більш докладніше.

Python – мова програмування яку належить до мов високого рівня загального призначення, спеціалізована за для підвищення продуктивності розробника програмного продукту, а також для читання коду. Мінімалістичний синтаксис є характерною особливістю ядра Python. Проте до стандартної бібліотеки Python входить дійсно великий обсяг корисних функцій [4].

Python є уособленням кількох парадигм програмування, до них звісно належать: структурний підхід, функціональний та об'єктно-орієнтований, імперативно і аспектно-орієнтований. Код організовується у функції та класи, які можуть об'єднуватися в модулі (вони в свою чергу об'єднані в пакети) [4].

Переваги Python:

* відкрита розробка;
* надає кошти швидкого прототипування і динамічної семантики;
* безліч корисних бібліотек і розширень мови можна легко використовувати у власних проектах завдяки гранично уніфікованого механізму імпорту та програмним інтерфейсами;
* механізми модульності добре проаналізовані і завжди можуть бути використані;
* все вказує на те що все в Python є об'єктами в сенсі ООП, проте при цьому об'єктний підхід не нав'язується програмісту.

Django – вільний програмний каркас для веб–додатків на мові Python, що використовує шаблон проектування MVC.

Model-view-controller (MVC, «модель-уявлення-контролер») – схема використання декількох шаблонів проектування, за допомогою яких модель додатки, призначений для користувача інтерфейс і взаємодія з користувачем розділені на три окремих компонента таким чином, щоб модифікація одного з компонентів надавала мінімальну вплив на інші. Дана схема проектування часто використовується для побудови архітектурного каркаса, коли переходять від теорії до реалізації в конкретній предметній області [5].

Основна мета застосування цієї концепції полягає в відділенні бізнес- логіки (моделі) від її візуалізації (уявлення, виду). За рахунок такого поділу підвищується можливість повторного використання.

Найбільшкорисно застосування даної концепції в тих випадках, коли користувач повинен бачити ті ж самі дані одночасно в різних контекстах або з різних точок зору.

Зокрема, виконуються наступні завдання:

* до однієї моделі можна приєднати кілька видів, при цьому не зачіпаючи реалізацію моделі (наприклад, деякі дані можуть бути одночасно представлені у вигляді електронної таблиці, гістограми і кругової діаграми);
* не зачіпаючи реалізацію видів, можна змінити реакції на дії користувача (натискання мишею на кнопці, введення даних), для цього досить використовувати інший контролер;
* ряд розробників спеціалізується тільки в одній з областей: або розробляють графічний інтерфейс, або розробляють бізнес-логіку (тому можливо добитися того, що програмісти, які займаються розробкою бізнес–логіки (моделі), взагалі не будуть обізнані про те, яке уявлення буде використовуватися).

Концепція MVC дозволяє розділити дані, подання та обробку дій користувача на три окремих компонента.

Модель (англ. Model). Модель надає знання: дані та методи роботи з цими даними, реагує на запити, змінюючи свій стан. Не містить інформації, як ці знання можна візуалізувати.

Подання, вид (англ. View). Відповідає за відображення інформації (візуалізацію). Часто в якості уявлення виступає форма (вікно) з графічними елементами. В Django за подання зазначають шаблони.

Контролер (англ. Controller). Забезпечує зв'язок між користувачем і системою: контролює введення даних користувачем і використовує модель і уявлення для реалізації необхідної реакції [5].

Фреймворк розділений на незалежні компоненти. Модель нічого не знає про таку річ, як HTTP-запит, а з шаблону не можна навіть випадково змінити дані. Всі модулі можна використовувати окремо від всієї системи,передавати з проекту в проект, підключати готові рішення. Існує величезна кількість готових модулів для додавання до проектів.

Для роботи з базою даних Django використовує власний ORM, в якому модель даних описується класами Python, і по ній генерується схема бази даних. Вона являє собою опис даних в базі і є еквівалентом SQL-коду CREATE TABLE [6].

Шаблон служить для відображення сторінок. Тут можна зустріти безліч шаблонних тегів, які спрощують побудову логічних конструкцій і різноманітні фільтри.

Крім того, Django поставляється з чималою кількістю вже написаних базових речей, які в тому чи іншому вигляді присутні майже в будь-якому сучасному веб-додатку:

* сесії (досить підключити в додаток потрібний модуль і в кожному запиті з'являється request.session, в якій можна зберігати будь–які дані);
* реєстрація, авторизація, система прав на об'єкти моделі даних, генерація паролів, розсилка повідомлень по email;
* кешування

Django дотримується принципу DRY (Do not repeat yourself) – не повторюй себе, тому програмісти намагаються не повторювати вже використаний код, отже, кількість рядків зменшується [7].

Django використовує такий потужний інструмент, як успадкування шаблонів. Сенс спадкування полягає в розробці скелетного батьківського і дочірніх шаблонів. Головна сторінка містить загальні розділи і в ній описуються переобумовленої в успадкованих шаблонах блоки.

Django спрощує вирішення проблем із забезпеченням безпеки програми. Багато вбудовані механізми не дозволяють застосовувати небезпечні рішення. Django за замовчуванням екранують все значення перед додаванням в базу, а перш ніж можна буде екранує HTML-код [7].

## **2.3 Засоби розробки інтерфейсу**

Інтерфейс інформаційної системи розробляється за допомогою трьох основних елементів: HTML + CSS + JS. Це зв'язка використовується для будь-якого веб–додатка.

HTML (англ. HyperText Markup Language – мова гіпертекстової розмітки) – стандартизований мову розмітки документів у Всесвітній павутині. CSS (англ. Cascading Style Sheets – каскадні таблиці стилів) – формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки HTML [8].

HTML задає структуру вмісту і його сенс, визначаючи такий контент як, наприклад, заголовки, абзаци або зображення. CSS же використовується для оформлення зовнішнього вигляду контенту, який використовує, наприклад, шрифти або кольори.

Ці дві мови – HTML і CSS незалежні один від одного і повинні такими і залишатися. CSS не повинен бути написаний всередині HTML-документа і навпаки. Як правило, HTML завжди буде представляти вміст, а CSS завжди буде визначати його оформлення [9].

JavaScript – мультипарадигменний мову програмування. Підтримує об'єктно-орієнтована, імперативний і функціональний стилі. Є реалізацією мови ECMAScript. JavaScript зазвичай використовується як вбудований мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінок [10].

Для реалізації інформаційноїсистеми автозапчастин використовується HTML5 + CSS3 + jQuery.

HTML5 є останньою версією HTML і характеризується масою переваг, основними з яких можна назвати:

* валідація форм тепер доступна класичними засобами HTML;
* вдосконалена семантика HTML5 дозволяє швидше і простіше розрізняти футер, заголовки, панель навігації;
* використання елементів HTML5 дозволяє зробити більш видимим семантичне ядро кожної сторінки (для реалізації такої задумки відтепер використовуються тільки стандартні коди);
* виконані поліпшення для інтеграції з вмістом SVG (таким чином, при використанні нових елементів відпала необхідність в установці таких плагінів, як Flash);
* музику, чарти, відео та малюнки можна вбудовувати без сторонніх програм, оскільки така функція розроблена в самому браузері;
* завдяки зменшенню розміру кукі розробники змогли істотно збільшити швидкість відгуку сторінки [8].

CSS3 відрізняється від попередньої версії тим, що підтримує ряд нових властивостей, таких як закруглення кутів, градієнти, прозорість, тіні, трансформації, плавні переходи і інше [9].

jQuery – бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними. Також бібліотека jQuery надає зручний API для роботи з AJAX [11].

Для відтворення та взаємодії з картами використовуємо бібліотеку Leaflet.

Leaflet – бібліотека з відкритим вихідним кодом, написана на JavaScript, призначена для відображення карт на веб-сайтах. Підтримує більшість мобільних і стаціонарних платформ з числа тих, що підтримують HTML5 та CSS3.

## **2.4 Проектування бази даних**

СУБД повинна забезпечувати реляційну модель роботи з даними. Сама модель має на увазі певний тип зв'язку між сутностями зрізних таблиць. Щоб зберігати і працювати з даними, такий тип СУБД повинен мати певну структуру (таблиці). У таблицях кожен стовпець може містити дані різного типу. Кожен запис складається з безлічі атрибутів (стовпців) і має унікальний ключ, що зберігається в тій же таблиці – всі ці дані взаємопов'язані між собою, як описано в реляційної моделі.

Розглянемо наступні СУБД.

SQLite – легко вбудовується в додатки база даних. Так як це система базується на файлах, то вона надає досить широкий набір інструментів для роботи з нею, в порівнянні з мережевими СУБД. При роботі з цією СУБД звернення відбуваються безпосередньо до файлів (в ці файлах зберігаються дані), замість портів і гнізд в мережевих СУБД. Саме тому SQLite дуже швидка, а також потужна завдяки технологіям обслуговуючих бібліотек.

Переваги SQLite:

* файлова структура – вся база даних складається з одного файлу, тому її дуже легко переносити на різні машини;
* відмінна при розробці та тестуванні – в процесі розробки додатків часто з'являється необхідність масштабування (SQLite пропонує все що необхідно для цих цілей, так як складається всього з одного файлу і бібліотеки написаної на мові C).

Недоліки SQLite:

* відсутність системи користувачів – більші СУБД включають в свій склад системи управління правами доступу користувачів (Зазвичай застосування цієї функції не так критично, так як ця СУБД використовується в невеликих додатках);
* відсутність можливості збільшення продуктивності – знову, виходячи з проектування, досить складно вичавити щось більш продуктивне з цієї СУБД [11].

MySQL – це найпоширеніша повноцінна серверна СУБД. MySQL дуже функціональна, що вільно розповсюджується СУБД, яка успішно працює з різними сайтами і веб додатками. Навчитися використанню цієї СУБД досить просто, так як на просторах інтернету ви легко знайдете більшу кількість інформації.

Незважаючи на те, що в ній не реалізований весь SQL функціонал, MySQL пропонує досить багато інструментів для розробки додатків. Так як це серверна СУБД, додатки для доступу до даних, на відміну від SQLite працюють зі службами MySQL.

Переваги MySQL:

* простота в роботі – встановити MySQL досить просто. Існують різноманітні програми, наприклад GUI, дозволяє досить легко працювати з БД;
* багатий функціонал – MySQL підтримує більшість функціоналу SQL;
* безпека – велика кількість функцій забезпечують безпеку, які підтримується за замовчуванням;
* масштабованість – MySQL легко працює з великими обсягами даних і легко масштабується;
* швидкість – спрощення деяких стандартів дозволяє MySQL значно збільшити продуктивність.

Недоліки MySQL:

* відомі обмеження – за задумом в MySQL закладені деякі обмеження функціонала, які іноді необхідні в особливо вимогливих додатках;
* проблеми з надійністю – через деяких способів обробки даних MySQL (зв'язку, транзакції, аудити) іноді поступається іншим СУБД по надійності [11].

PostgreSQL є професійним з усіх трьох розглянутих СУБД. Вона вільно розповсюджується і максимальновідповідає стандартам SQL. PostgreSQL намагається повною мірою використовувати ANSI / ISO SQL стандарти своєчасно з виходом нових версій.

Від інших СУБД PostgreSQL відрізняється підтримкою затребуваного об'єктно–орієнтованого і / або реляційного підходу до баз даних. Наприклад, повна підтримка надійних транзакцій, тобто атомарность, послідовність, ізоляційні, міцність. Завдяки потужним технологіям PostgreSQL дуже продуктивна. Паралельність досягнута не за рахунок блокування операцій читання, а завдяки реалізації управління різноманітним паралелізмом (MVCC). PostgreSQL дуже легко розширювати своїми процедурами, які називаються збережені процедури. Ці функції спрощують використання постійно повторюваних операцій.

Хоча PostgreSQL і не може похвалитися великою популярністю на відміну від MySQL, існує досить велика кількість додатків полегшують роботу з PostgreSQL, незважаючи на всю потужність функціоналу. Зараз досить легко встановити цю СУБД використовуючи стандартні менеджери пакетів операційних систем.

Переваги PostgreSQL:

* відкрите ПЗ відповідає стандарту SQL;
* велика кількість доповнень – незважаючи на величезну кількість вбудованих функцій, існує дуже багато доповнень, що дозволяють розробляти дані для цієї СУБД і управляти ними;
* розширення – існує можливість розширення функціоналу за рахунок збереження своїх процедур;
* об'єктно – PostrgreSQL це не тільки реляційна СУБД, але також і об'єктно–орієнтована.

Недоліки PostgreSQL:

* продуктивність – при простих операціях читання PostgreSQL може значно уповільнити сервер і бути повільніше своїх конкурентів, таких як MySQL;
* популярність – за своєю природою, популярністю ця СУБД похвалитися не може, хоча і є досить велика спільнота;
* хостинг – в силу названих вище чинників іноді досить складно знайти хостинг з підтримкою цієї СУБД [11].

Розглянувши подані СУБД, можна зробити висновок, що оптимальним рішенням дляінформаційної системи, що розробляється є SQLite. Недоліки даної СУБД для розроблюваної системи некритичні, тому що система користувачів контролюється фреймворком Django, і наявність даної функції в СУБД не є необхідністю, а швидкість роботи для інтернет–магазину не є важливим критерієм. Плюси є куди більш значущими, вся база даних складається з одного файлу і не вимагає установки різних служб і сервісів, що допоможе заощадити місце на хостингу. Крім цього СУБД легко встановлюється і налаштовується, а що більш важливо є масштабується.

## **2.5** **Проектування інформаційної системи діяльності інтернет торгівлі**

В рамках процесу проектування розглянемо основні вимоги, що пред'являються до інформаційної системи магазину. Функціональні вимоги досліджуються і формуються у вигляді моделі варіантів використання.

Варіант використання – це незалежне від реалізації багаторівневе представлення конкретної функції розроблюваної системи. Варіант використання прийнято вважати послідовністю дій, виконуваних системою у відповідь на подію, котра зініціюється зовнішнім об'єктом (дійовою особою так званим актором) [12]. Іншими словами, таке уявлення призначене для спрощення взаємодії з майбутніми користувачами системи, з клієнтами, і особливо знадобляться для визначення необхідних характеристик системи.

Складання варіантів використання проводиться на основі аналізу вимог до інформаційних систем з електронною комерцією. Головними акторами в контексті програмного засобу є адміністратор і користувач цієї системи.

Адміністратор в праві додавати товари, встановлювати їм характеристики, ціну та категорію, а також видалити або змінити вже існуючий товар. У міру здійснення продажів адміністратор може відстежувати статистику по продажах, відвідування, активності та ін.

Користувачі повинні мати можливість по перегляду, пошуку, сортування та категоризації товарів, перед безпосередньою покупкою. Також повинна бути можливість переглянути відгуки по зацікавленому товару, а також переглянути рейтинг обраного товару.

На рис. 2.1 зображена діаграма варіантів використання в нотаціїUML.

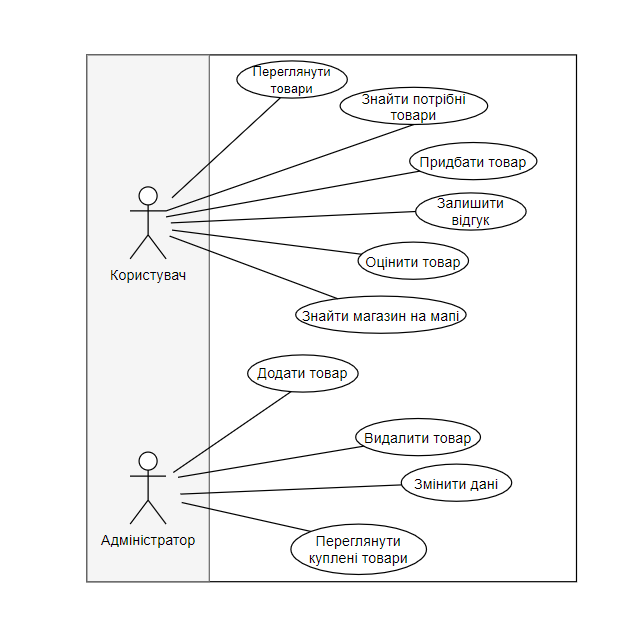


Рисунок 2.1 – Діаграма варіантів використання

Виділення основних варіантів використання полегшить процес подальшого проектування.

Для того щоб спроектувати моделі БД використовувалося середовище Draw.io і його методологія IDEF1X, де доступні два рівня уявлення – логічний і фізичний.

Логічний рівень загалом вказує на відображення конкретних фактів з реального життя. В той час як логічний рівень не розглядає використання конкретної СУБД, не визначає типи даних ( ціле чи дійсне число) і не визначає індекси для таблиць [3].

Спроектуємо логічну модель бази даних інформаційної системи інтернет-магазину. Дана модель буде містити 12 таблиць пов'язаних між собою логічними зв'язками.

Для того щоб зберігати дані в базі необхідно створити такі таблиці:

* «Категорії товарів» – для зберігання даних про категоріях: ідентифікатор категорії, найменування;
* «Характеристики товарів» – для зберігання даних про властивості різних товарів: ідентифікатор характеристики, найменування;
* «Магазини» – для зберігання даних про магазинах: ідентифікатор магазину, найменування, адреса, координати;
* «Товари» – для зберігання даних про різні товари: ідентифікатор товару, найменування, ціна, дата додавання, зображення, гарантія, відображення, перегляди, опис, новинка, ідентифікатор категорії;
* «Товар–Характеристики товарів» – допоміжна таблиця для зберігання даних про властивості притаманні окремому товару, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор характеристики, значення;
* «Товар–Магазини» – допоміжна таблиця для зберігання даних про магазинах в яких доступний товар, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор магазину, кількість;
* «Профіль клієнта» – для зберігання даних про клієнтів: ім'я, пошта, пароль, телефон, стать, дата народження;
* «Список бажань» – допоміжна таблиця для зберігання даних про товари, які клієнт збирається купити пізніше, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор клієнта;
* «Історія продажів» – для зберігання даних про продажі: ідентифікатор клієнта, дата;
* «Позиція товару» – допоміжна таблиця для зберігання даних про товари в продажу, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор продажу, кількість;
* «Кошик» – допоміжна таблиця для зберігання даних про товари, які додані для безпосередньої покупки, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор користувача, кількість;
* «Коментарі» – допоміжна таблиця для зберігання коментарів про товари, що дозволяє уникнути зв'язку багато до багатьох: ідентифікатор товару, ідентифікатор користувача, дата, оцінка, текст коментаря.

Логічна модель має наступний вигляд (рисунок 2.2):

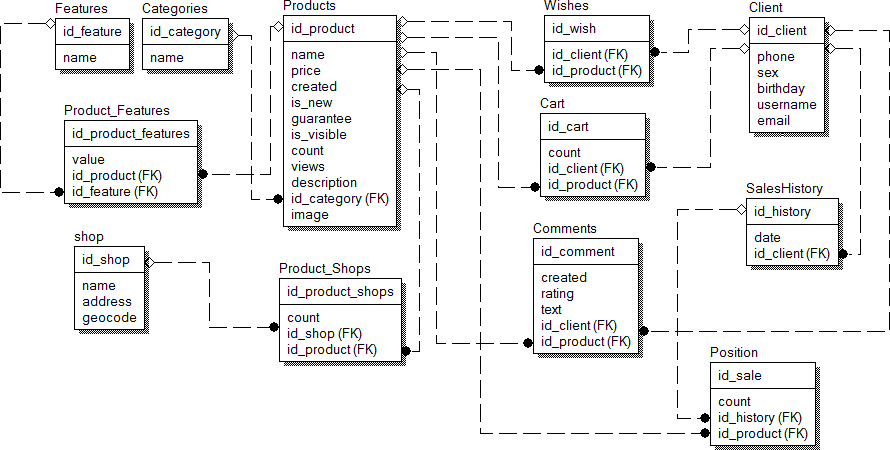


Рисунок 2.2 – Логічна модель даних

Фізичний рівень моделі складають імена об'єктів і типи даних, індекси. Фізична модель представлена на рисунку 2.3.

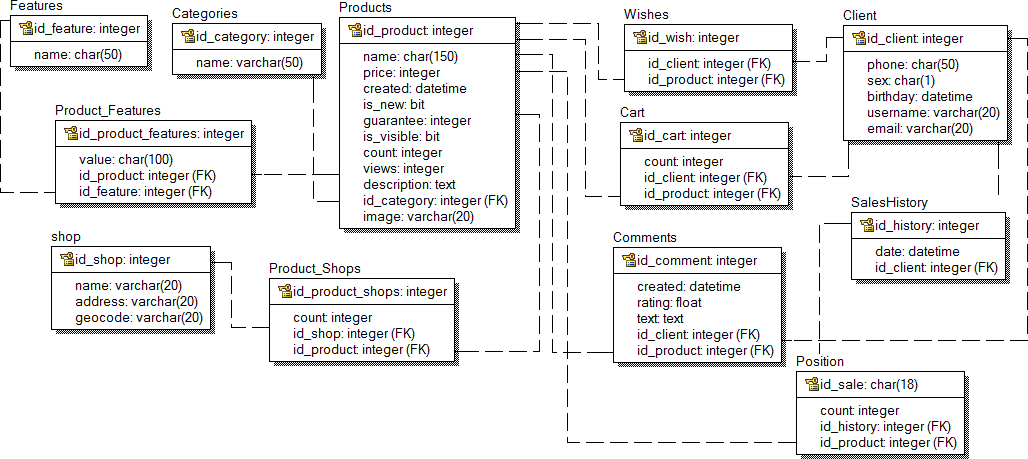


Рисунок 2.3 – Фізична модель даних

Фізична і логічна моделі дозволяють розглянути всі зв'язки і типи даних перед реалізацією, що полегшує створення бази даних.

Інтерфейс прийнято вважати так званим комунікаційним каналом, за допомогою якого здійснюється взаємодія користувача інформаційної системи і комп'ютера.

Гарним користувацьким інтерфейсом можна вважати той, використовуючи який користувач не буде зважати і практично не помічати його. Такий інтерфейс називають прозорим – користувача нічого не відволікає від основного наповнення сторінки [13].

Користувачі, які відвідують інформаційну систему, в більшості випадків, приходять з певною метою, і пріоритетним для них є швидкий і зручний доступ до необхідної інформації або товарів, тому для розробки інтерфейсу вкрай необхідно організувати простір таким чином, щоб врахувати всі особливості сприйняття і потреби покупців.

Виділимо основні принципи, які слід враховувати при розробці користувальницького інтерфейсу.

Ясність – це перша і найголовніша мета будь-якого інтерфейсу. Для того, щоб ефективно використовувати розроблений інтерфейс, люди повинні легко виявляти його, усвідомлювати, навіщо вони його використовують, розуміти, взаємодія з яким об'єктом він забезпечує, представляти, що станеться при його використанні і, нарешті, бути здатні взаємодіяти з ним.

Перше знайомство з інтерфейсом – момент критичний, і, в той же час, дизайнери про нього дуже часто забувають. Для того, щоб допомогти користувачам максимально швидко освоїтися з інтерфейсом, розробляти дизайн потрібно з урахуванням точки входу – моменту, коли ще нічого не сталося. Початкова сторінка повинна вказувати напрямок руху. Більшість проблем взаємодії виникає в контексті початку робіт, зрозумівши правила, користувачі швидше досягнуть успіху.

Функціональність важливіше зовнішності. Люди найбільш комфортно відчувають себе в оточенні об'єктів, поведінка яких вони здатні передбачити – інших людей, тварин, речей, програм. Коли хтось або щось діє очікуваним нами чином, ми відчуваємо до нього симпатію. При тривалій взаємодії з інтернетом користувачі можуть зіткнутися з певним стереотипом, що являє собою певну систему умовних інстинктів. Тому зовнішній вигляд елементів дизайну повинен відповідати уявленням користувача. Заходячи на сайт, людина вже заздалегідь передбачає, в якій частині сторінки буде знаходитися потрібна йому інформація або елемент меню. На практиці це означає, що будь–який користувач повинен розуміти, як діє той чи інший елемент інтерфейсу, просто глянувши на нього.

В основі загальної композиції сторінки існує такий принцип, – текст сприймається зліва направо і зверху вниз. З цього слідує що, найбільш важливі елементи повинні знаходитися в зоні початкового уваги, за для того щоб користувач не відволікався на другорядні інформацію. Тобто злівого боку верхньої частини веб–сторінки. Найбільш змістовна інформація зазвичай розташовується в центральній частині сторінки. Менш важлива, допоміжна інформація повинна знаходитися в правій і в правій нижній частинах сторінки. Однотипні дані на різних сторінках повинні розташовуватися в одній і тій же області [13].

Існують інші способи привернення уваги користувача не лише просторовим розташуванням, а ще й за допомогою виділення розміром, залежність між розмірами об’єктів проста чим більше об’єкт ті скоріше користувач зверне на нього увагу. Яскраві деталі сприймаються краще в порівняні з затемненими, кольорові шрифти також перевертають увагу користувача.

Розглянуті особливості та принципи побудови призначений для користувача інтерфейсів слід враховувати при розробці дизайну інформаційної системи з продажу автозапчастин.

Django-додаток описує повне безліч дій, можливих при взаємодії сервера і клієнта. Ці дії можуть містити відображення веб–сторінки, виконання запитів до бази даних, запис даних на диск і багато іншого. Django використовує патерн MVC, тому структура програми буде виглядати наступним чином.

У файлі models.py описуються моделі даних у вигляді класів на мові Python. Дані будуть зберігатися в реляційної базі даних, проте завдяки ORM технології, дані використовуються як об'єкти моделей.

Інтерфейс являє собою взаємозв'язок шаблонів Django. Структура сторінок представлена базовим шаблоном, що містить в собі верхню частину сайту. Цей шаблон успадковують інші, які замінюють один одного при переході з розділу в розділ. Це досягається завдяки системі успадкування шаблонів. Сторінки заповнюються даними, які беруться з моделей.

У views.py міститися всі можливі керуючі функції і класи. Вони контролюють який шаблон слід використовувати і які дані і методи моделі задіяти. Діаграма компонентів представлена на рисунку 2.4.

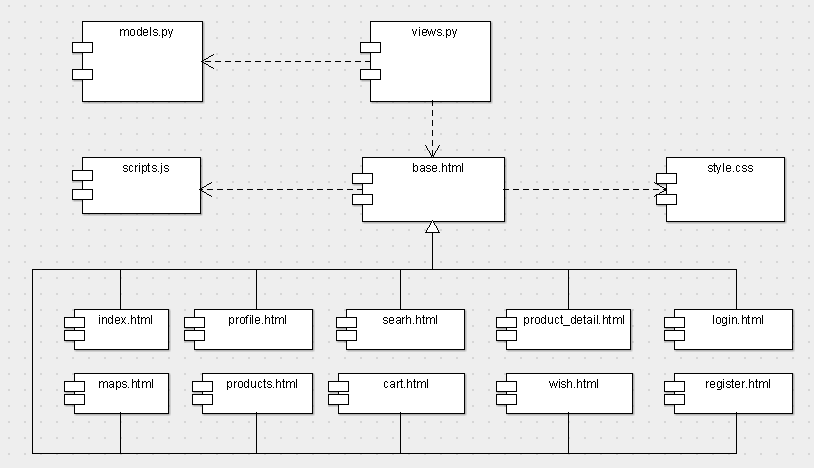


Рисунок 2.4 – Діаграма компонентів

Автоматизована система продажу складається з двох частин. Перша, клієнтська частина, відповідає за взаємодію з покупцями. Вона являє собою інтернет-магазин і надає користувачам різні можливості.

Незареєстровані користувачі можуть переглядати каталог товарів (усі товари, новинки, лідери продажів), сортувати товари (за рейтингом, вартості, результатам пошуку, відгуками), а також користуватися пошуком.

Користувачі можуть додавати товари в кошик, встановлювати потрібну кількість товару та оформити покупку. При необхідності товари можна видалити з кошика. У профілі доступна історія покупок, а також можливість змінити особисті дані або пароль. Якщо не вистачає грошей на покупку, можна додати товар, який сподобався в список бажань, щоб його не втратити. Діаграма діяльності для реєстрації користувача представлена на рисунку 2.5



Рисунок 2.5 – Діаграма діяльності

Друга частина являє собою адміністративну панеллю. Доступ право лише ті користувачі з правами адміністратора. Адміністратор за допомогою панелі може заповнювати базу даних товарами, категоріями, властивостями і іншою необхідною інформацією, а також редагувати або видаляти дані. У розділі продажів можна подивитися історію покупок скоєних користувачами.

У розділі з магазинами адміністратор може вказати географічні координати, для того, щоб користувачі могли знайти на карті найближчий до них магазин з товаром, який їм сподобався.Код додатку представлений в додатку Б.

В Django таблиці бази даних представлені у вигляді моделей. Модель – це основне джерело даних. Він містить набір полів і поведінку даних які зберігаються в базі.

Django використовує принцип ORM, отже підтримуються основні принципи ООП як успадкування, поліморфізм, інкапсуляції та інші. Опис моделі даних є аналогом SQL CREATE TABLE і має особливості:

* назви таблиць створюються автоматично з назви програми і назви моделі в нижньому регістрі;
* первинні ключі додаються автоматично;
* Django додає «\_id» до назви зовнішнього ключа.

Структура бази даних описується за допомогою класів на мові Python. Кожен клас успадковує базовий клас Django «Model», який, в свою чергу, вже містить методи по додаванню, видалення і зміни даних.

Діаграма класів представлена на рисунку 2.6.

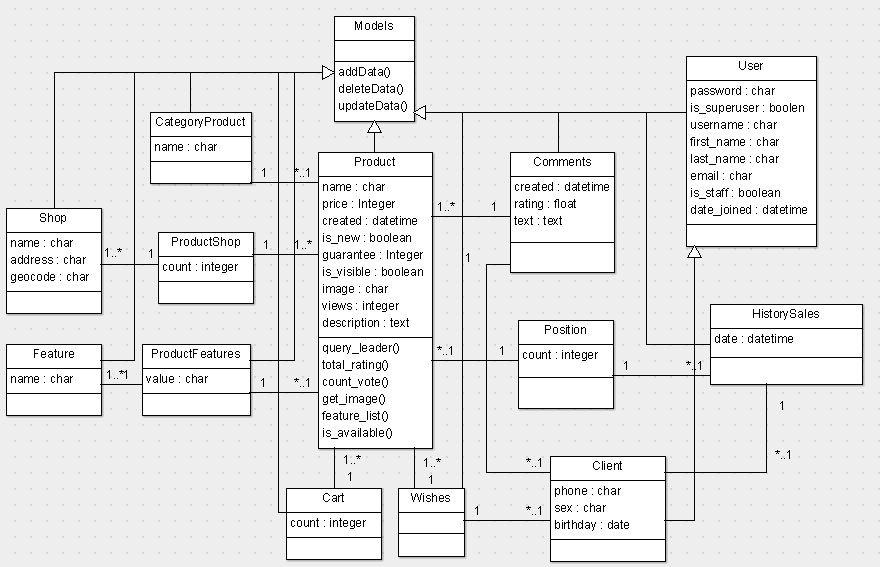


Рисунок 2.6 – Діаграма класів

Представлені класи:

* клас «CategoryProduct» – описує таблицю «Категорії товарів»;
* клас «Feature» – описує таблицю «Характеристики»;
* клас «Shop» – описує таблицю «Магазини»;
* клас «Product» – описує таблицю «Товари»;
* клас «ProductFeature» –опісиваеттабліцу «Товар–Характеристики товарів »;
* клас «ProductShop» – описує таблицю «Товар–Магазини»;
* клас «Client» – описує таблицю «Профіль користувача»;
* клас «ClientWish» – описує таблицю «Список бажань»;
* клас «HistorySales» – описує таблицю «Історія продажів»;
* клас «ClientCart» – описує таблицю «Кошик»;
* клас «Comment» – описує таблицю «Коментарі»;
* клас «Position» – описує таблицю «Позиція товару».

Крім моделей даних для роботи магазину знадобляться кілька властивостей і методів:

* query\_leader (cls, top), метод класу «Product», визначає лідерів з продажу;
* total\_rating (self), властивість об'єкта класу «Product», вираховує загальний рейтинг певного товару;
* count\_vote (self), властивість об'єкта класу «Product», визначає кількість голосів певного товару;
* get\_image (self), властивість об'єкта класу «Product», повертає url до зображення в html вигляді;
* feature\_list (self), метод об'єкта класу «Product», повертає список характеристик зазначеного товару;
* is\_available (self), метод об'єкта класу «Product», повертає True, якщо товар є в наявності.

Базовим шаблоном є base.html. Він містить в собі шапку сайту, з логотипом, пошуковим блогом, і навігацією. Доступ до цих елементів необхідний з будь–якої частини сайту, тому базовий шаблон завжди буде використовуватися при рендерингу сторінки. Його успадковують інші шаблони (рисунок2.7).

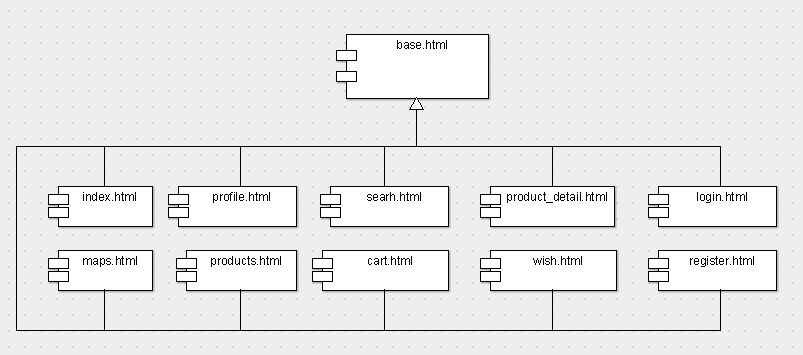


Рисунок 2.7 – Спадкування шаблонів

На рисунку представлені наступні шаблони:

* index.html, містить в собі структуру головної сторінки;
* products.html, є каталогом товарів, в залежності від параметрів вибірки відображає потрібні товари;
* product\_detail.html, сторінка с детальною інформацією про конкретний товар, характеристиками, рейтингом, відгуками та ін.;
* cart.html, користувацька корзина з можливістю змінювати кількість товарів, видаляти або оформити покупку;
* wish.html, список бажаних товарів користувача;
* login.html, сторінка авторизації користувача;
* register.html, сторінка реєстрації нового користувача;
* profile.html, профіль користувача, є можливість змінювати інформацію, змінити пароль, подивитися історію покупок;
* maps.html, сторінка з відображенням магазинів на карті;
* search.html, сторінка з результатами пошуку.

# РОЗДІЛ 3

# РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ ПРОДАЖУ АВТОЗАПЧАСТИН

## **3.1 Користувацький інтерфейс**

Прийнявши до уваги вимоги, представлені в розділі проектування, був розроблений для користувача інтерфейс.

Розглянемо основні сторінки веб-додатку, такі як: головна сторінка інтернет-магазину, де представлені акції та пропозиції, каталог товарів з можливістю вибірки і сортування за різними параметрами, сторінка з докладним описом товару, його характеристиками і відгуками, карта магазинів в яких даний товар є в наявності, а також призначена для користувача корзина, в яку потрапляють всі товари, перед оформленням покупки.

Верхня частина сайту є окремим шаблонами, і несе в собі функції, які необхідні з будь-якої частини інтернет-магазину. Даний шаблон успадковують всі інші.

Інтерфейс головної сторінки представлений на рисунку3.1.

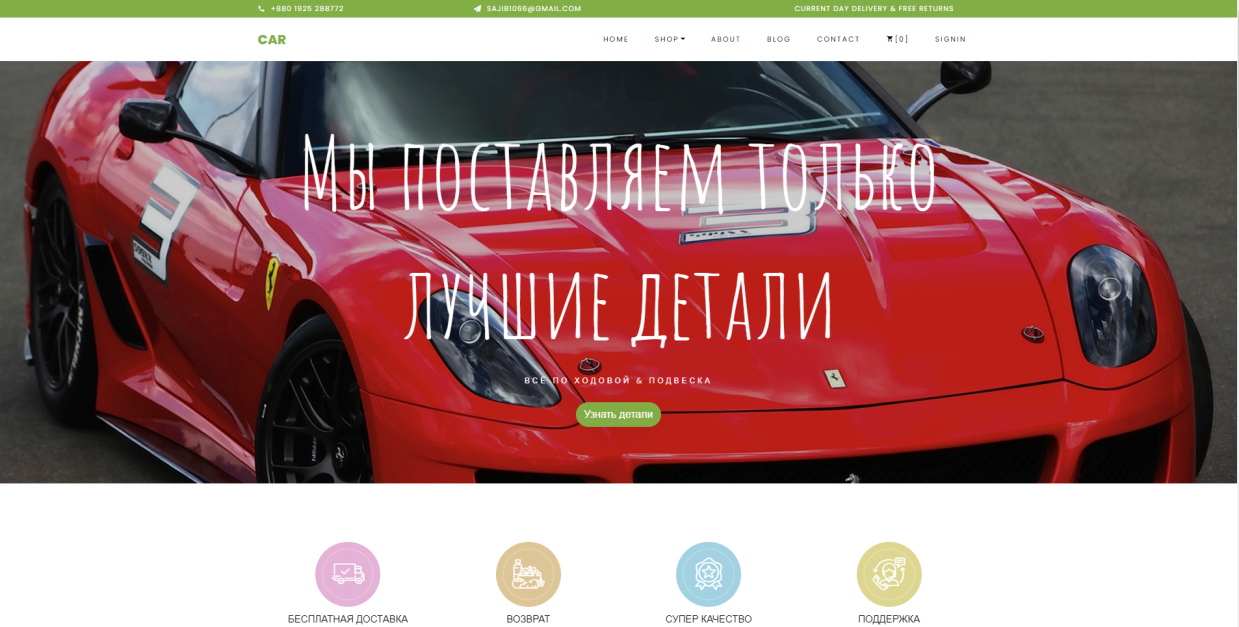


Рисунок 3.1 – Головна сторінка

Інтерфейс умовно можна розділити на три частини. Верхня – «шапка» сайту, центральна – слайдер, панель вибору категорії товару і блок з популярними товарами, а також нижня, яка містить інформацію про магазин.

Верхня частина сайту включає основні функції та елементи навігації інтернет–магазину. Там розташовується інформації про товари, що знаходяться в кошику. У верхній частині знаходиться контактна інформація: електрона пошта та номер гарячої лінії. Навігація по інформаційній системі здійснюється за допомогою кнопок–посилань, за допомогою них у користувача є можливість перейти до головної сторінки «Home», кнопка «Shop» дозволяє перейти до перегляду товарів, «ABOUT» перехід на сторінку з довідковою інформацією, «BLOG» посилання на блог магазину автозапчастин, «CONTACNT» контактна інформація для зворотного зв’язку.

Навігаційна панель представлена на рисунку 3.2.



Рисунок 3.2 – Навігаційна панель

Центральна частина сайту представлена анімованим слайдером з цікавими пропозиціями і знижками. Над слайдером розташовується панель вибору категорії товарів. Далі представлена панель з найпопулярнішихтоварів магазину. Є можливість відразу додати товар в корзину або ж перейти до їх докладного опису.

Нижня частина сайту містить інформацію про магазин, номера телефону, адреса головного магазину, а також доступні способи оплати товарів.

Інтерфейс каталогу товарів представлений на рисунку 3.3.

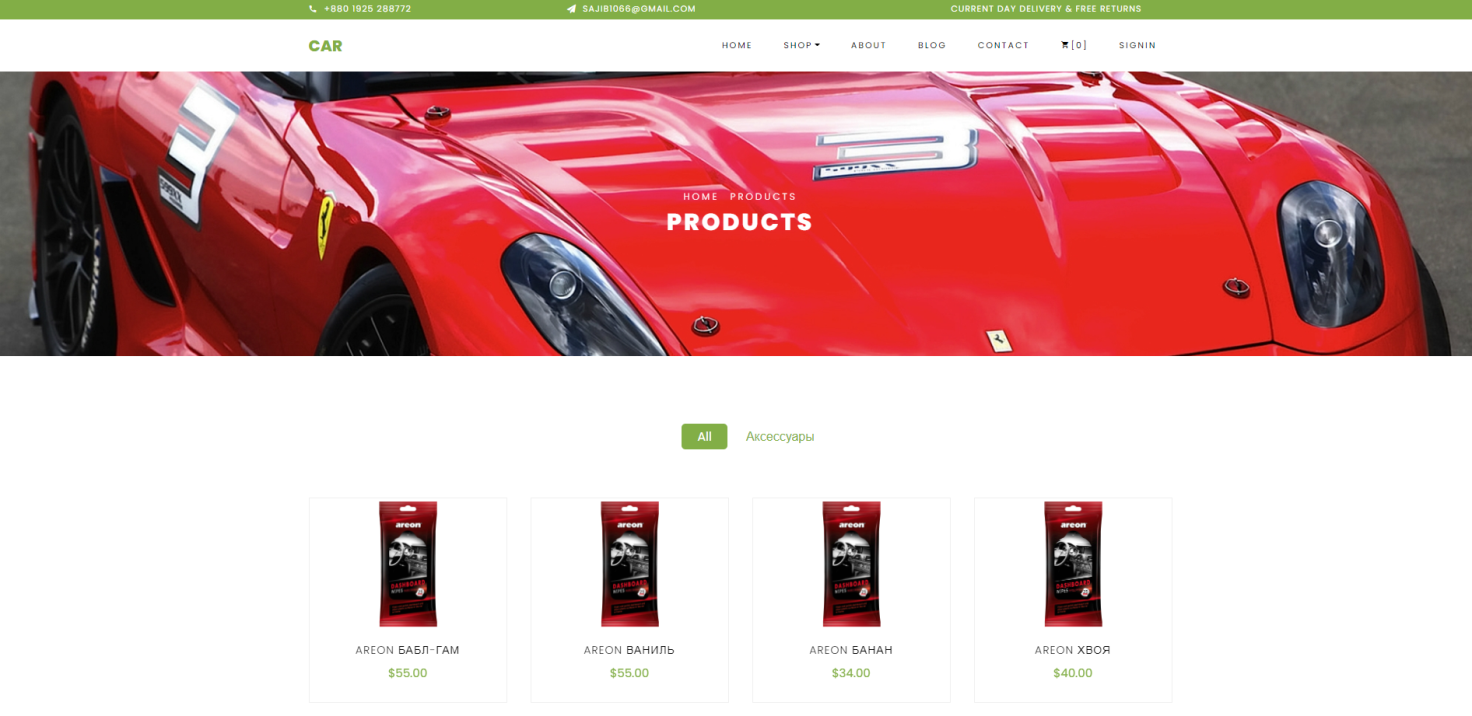


Рисунок 3.3 – Інтерфейс каталогу товарів

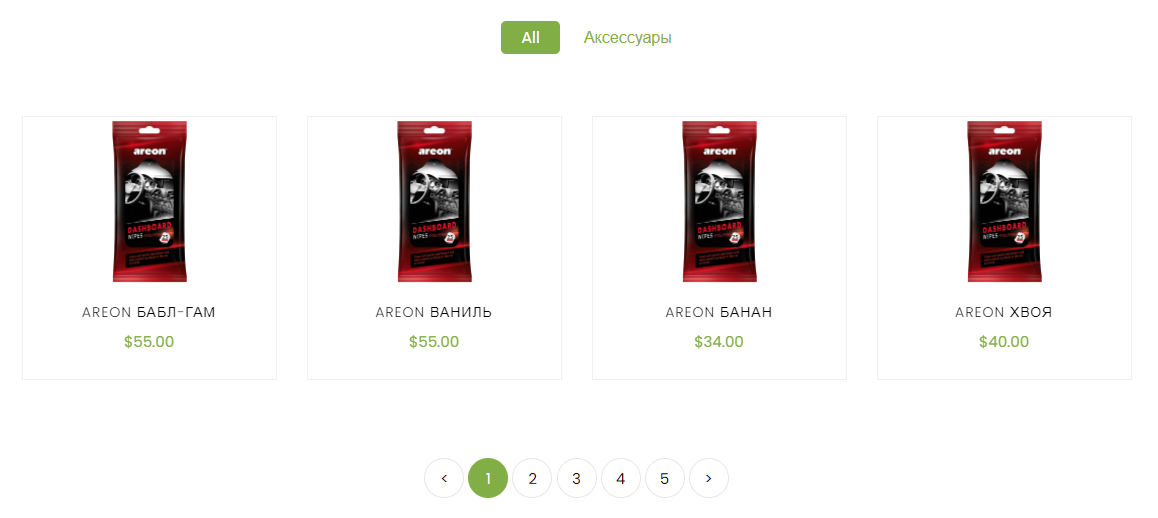


Рисунок 3.4 – Каталог товарів

Тут верхня і нижня частина інтерфейсу залишається колишньою, а центральна частина є каталогом товарів.

Товари виводяться посторінково. Є меню вибору сортування товарів. Спеціальними маркерами позначаються новинки. Присутні іконки свідчать про переглядах і коментарях конкретного товару. Товар, який був оцінений користувачами має рейтинг.

Інтерфейс сторінки певного товару представлений на рисунку3.5.

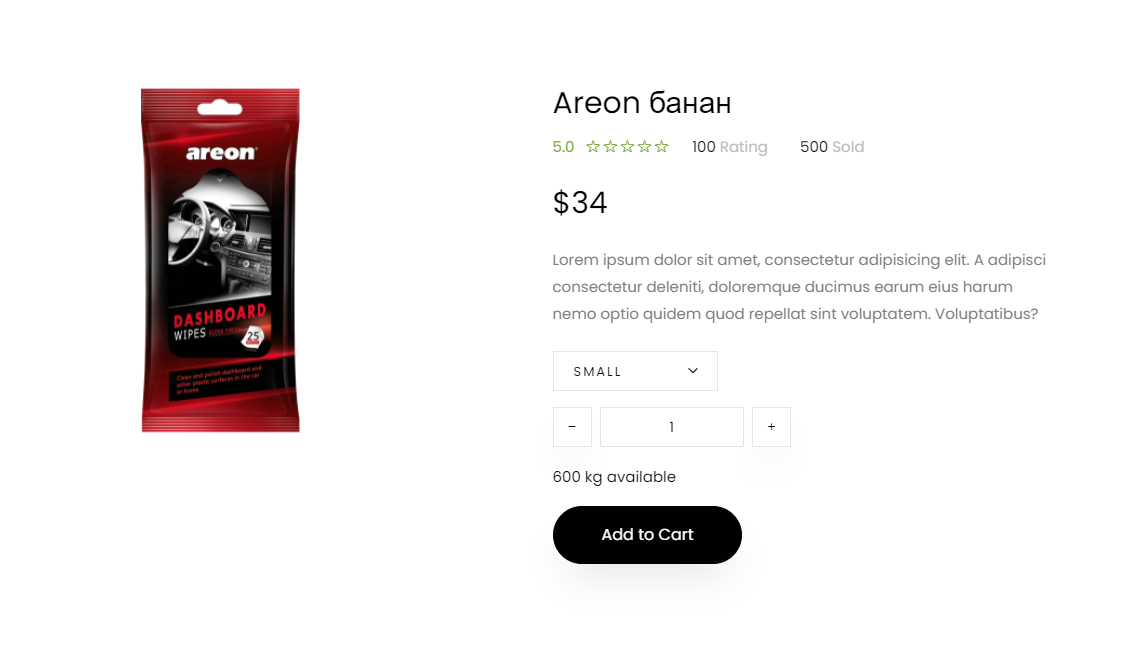


Рисунок 3.5 – Сторінка товару

Сторінка товару надає докладну інформацію про конкретний товар. У верхній частині представлено найменування, зображення, вартість, рейтинг, гарантія товару, а також кнопки додавання в корзину. Є можливість обрати розмір товару.

У розділі «Опис» зазначено докладний опис товару, в розділі «Коментарі» можна подивитися відгуки інших покупців або ж залишити свій.

Інтерфейс сторінки з картою міста та мітками магазинів представлена ​​на рисунку3.6.

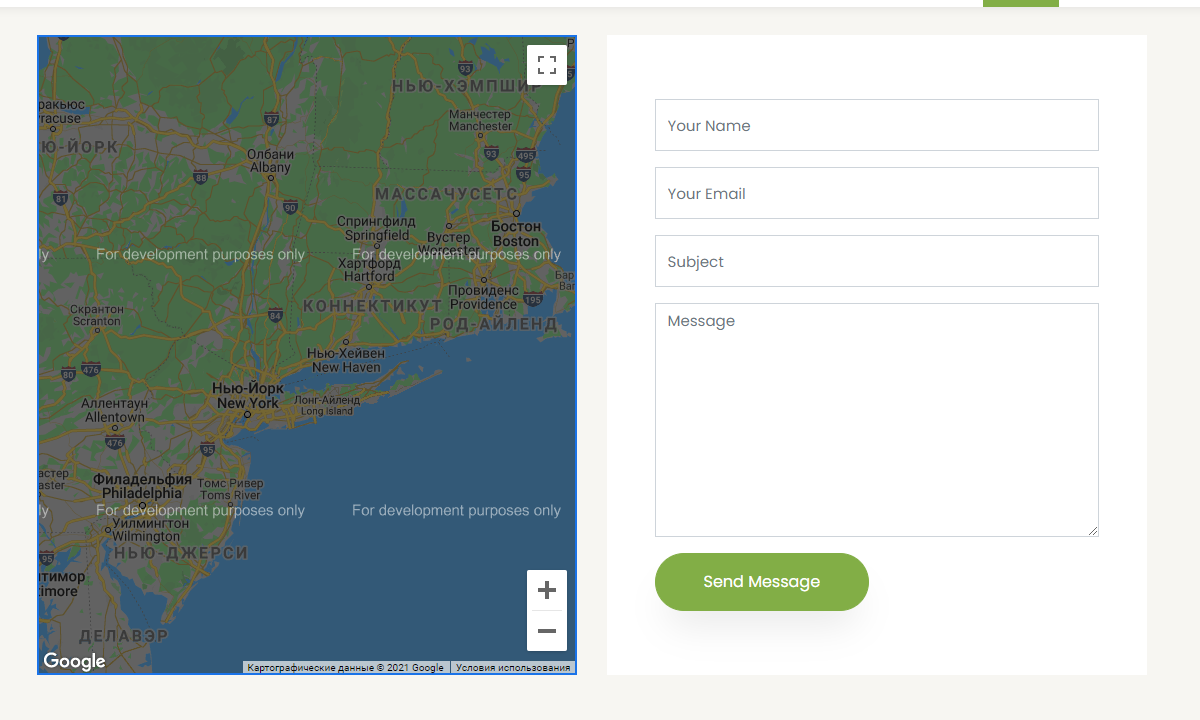


Рисунок 3.6 – Карта магазину

На карті буде відзначено магазини, де є обраний товар в наявності. На правій панелі можна залишити своє повідомлення.

Інтерфейс сторінки з товарами в кошику представлена на рисунку3.7.



Рисунок 3.7 – Кошик

У кошику є можливість змінити кількість товару, видалити непотрібний, а також оформити покупку.

## **3.2 Адміністративна панель**

Адмін–панель – це панель управління сайтом, з якої може працювати будь-який користувач, який не має навичок програмування. Всі матеріали вносяться на сучасні сайти за допомогою системи управління контентом (CMS).

Наведемо приклад, щоб розібратися, що значить адмінка для сайту. Перед капітаном космічного корабля – пульт з різними кнопками, важелями і екранами. І він знає, на яку кнопку натиснути, щоб комп'ютер видав інформацію про обстановку на борту, а корабель пішов вліво, або, наприклад, зробив стрибок в гіперпростір.

Адмін-панель – це пульт управління космічним кораблем. Головна відмінність веб–ресурсів на Django полягає в тому, що робота з базами даних, текстовими, графічними та іншими файлами відбувається за допомогою управління за допомогою зрозумілої системи.

Сайт без адмінки також може працювати з базою даних і всіма перерахованими вище компонентами. Але, наприклад, для того щоб в Django на сторінку встановити картинку, потрібно зробити всього лише одну дію: натиснути на відповідну іконку у вбудованому редакторі, а далі слідувати вказівкам. Ризик помилки виключений на 99,9%.

А ось для установки зображення на сторінку в коді (без адмін-панелі) доведеться зробити масу кроків:

1. Завантажити картинку на сервер.
2. Знайти відповідний розділ в базі даних або в html-коді на сторінці, куди ви збираєтеся розмістити картинку.
3. За допомогою тега прописати код відображення картинки, при цьому ніде не помилитися.
4. Відкрити сторінку і перевірити, як відображається картинка. Повернутися в кодову частину і виправити помилки в міру необхідності.

Для редагування інформації в інформаційній системі потрібно володіти доступом до адмін-панелі, а саме мати обліковий запис з логіном та паролем, котрий потрібно ввести в поле авторизації(Рис 3.8)

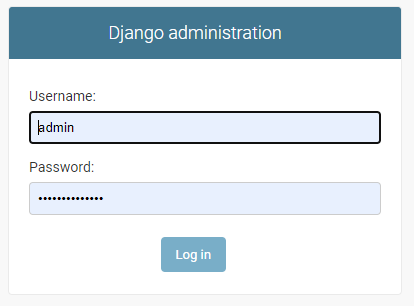


Рисунок 3.8 – Поле авторизації

Адміністратором прийнято вважати людину яка керує роботою інформаційною системою. Створення постів та додавання товару на сайт здійснюється безпосередньо через адмін-панель. Після авторизації адміністратор має доступ до зручного інтерфейсу керування інформаційної системи(Рис 3.9)

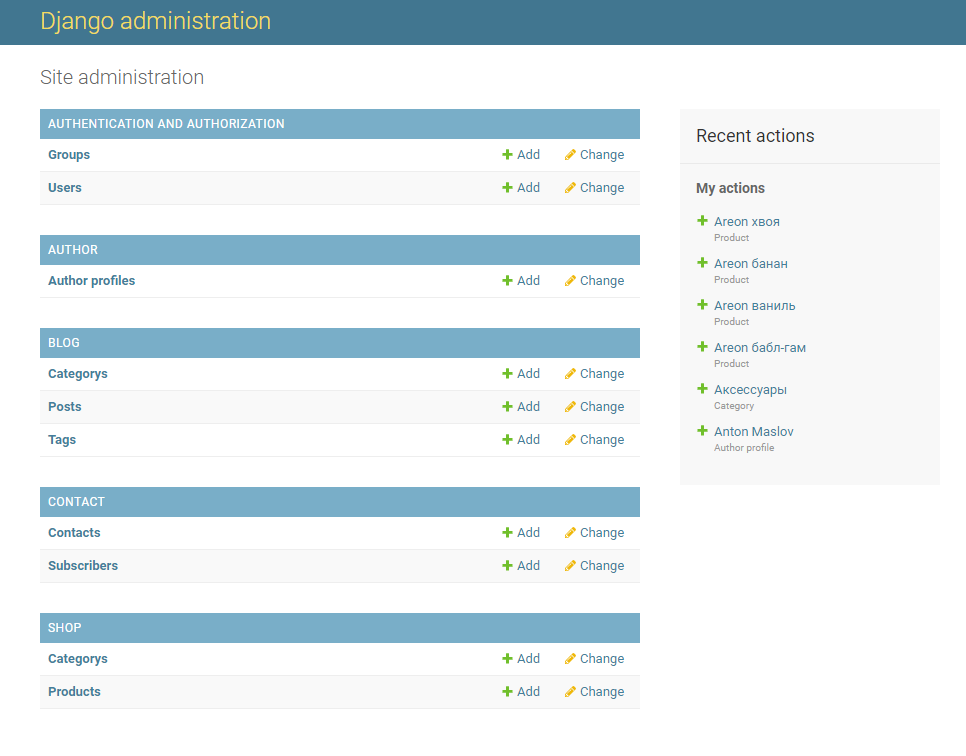


Рисунок 3.9 – Інтерфейс адміністративної панелі

У розділі «Products» знаходиться список товарів інформаційної системи. (рис 3.10). Адміністратор має можливість редагувати, видаляти та додавати нові товари. Додані товари будуть відображатися у користувацькому інтерфейсі після збереження.

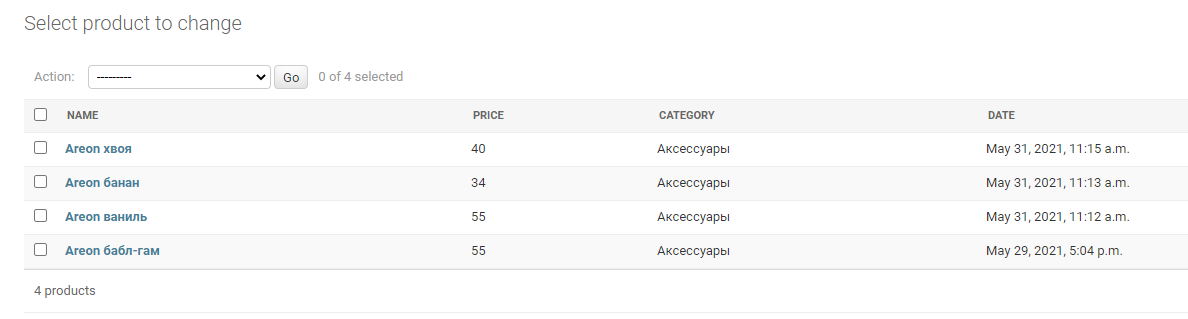


Рисунок 3.10 – Розділ «Products»

Для того щоб додати новий товар, адміністратору потрібно заповнити поля товару вказати ціну, завантажити фото товару, створити опис тощо.( Рис 3.11)



Рисунок 3.11 – Розділ «Products»

Після натискання кнопки Save, товар буде додано до бази даних та відображено в користувацькому інтерфейсі.

# РОЗДІЛ 4 ТЕСТУВАННЯ

Тестування програмного забезпечення – процес виявлення помилок в програмному забезпеченні (ПЗ). Існуючі на сьогоднішній день методи тестування ПО не дозволяють однозначно і повністю усунути всі дефекти і помилки і встановити коректність функціонування аналізованої програми особливо в закритих приватних програмах. Тому всі існуючі методи тестування діють в рамках формального процесу перевірки досліджуваного або розробляється ПО [18].

Такий процес формальної перевірки або верифікації може довести, що дефекти відсутні, з точки зору використовуваного методу – тобто немає ніякої можливості точно встановити чи гарантувати відсутність дефектів в програмному продукті з урахуванням людського фактора, присутнього на всіх етапах життєвого циклу ПЗ.

Існує безліч підходів до вирішення завдання тестування і верифікації ПЗ, але ефективне тестування складних програмних продуктів – це процес надзвичайно творчий, що не зводиться до проходження суворим і чітким процедурам або створення таких.

Одними з підходів до тестування є методологія функціонального тестування.

## **4.1 Функціональне тестування**

Функціональне тестування – це тестування ПО з метою перевірки можливості бути реалізованим функціональних вимог, тобто проведення функціонального тестування дозволяє перевірити здатність інформаційної системи в певних умовах вирішувати завдання, потрібні користувачам [18].

Функціональне тестування є одним з ключових видів тестування, завдання якого – встановити відповідність розробленого програмного забезпечення вихідним функціональним вимогам замовника.

Основні переваги функціонального тестування:

* функціональне тестування ПЗ повністю імітує фактичне використання системи;
* дозволяє своєчасно виявити системні помилки ПЗ і, тим самим, уникнути безлічі проблем при роботі з ним надалі;
* економія за рахунок виправлення помилок на більш ранньому етапі життєвого циклу ПЗ.

Виконаємо функціональне тестування автоматизованої системи магазину автозапчастин.(Таблиця 4.1)



Таблиця 4.1 –Дефекти функціонального тестування сайту

## **4.2 Мобільне тестування**

Мобільний тестування ‒ одна з найбільш активно розвиваються сфер через швидко зростаючого ринку мобільних додатків. Розберемо, на що потрібно звертати увагу при тестуванні мобільних додатків і яким інструментарієм для цього необхідно володіти.

Починають тестування з перевірки на відповідність вимогам і дизайну. Мова тут не тільки про те, щоб перевірити наявність всіх картинок або працездатність посилань, а про повноцінний UX / UI аналізі. Це означає, що тестувальник повинен вміти працювати з Figma, Zeplin, використовувати інструменти перевірки інтерфейсів як Appium Viewer та інші. Також необхідно опрацювати всі можливі маршрути переміщення користувача по додатку. З цією метою складають mindmap – діаграму зв'язків між сторінками. Для цього підійде будь–який mindmap–продукт – наприклад, Mindomo або xMind.

Також необхідно проаналізувати мережевий трафік: обрив мережі і слабкий інтернет, вихідні запити і отримані відповіді. Для цього використовують сніфери Charles / Fiddler, Proxyman і інші.

При необхідності виконують тестування API. Для цього завдання використовують спеціалізовані інструменти: Swagger, Postman, SOAPUI. Вони допомагають документувати запити і виконувати їх інтерактивну перевірку.

Для тестування на різних пристроях використовують емулятори на кшталт Genymotion, BlueStacks. Однак успішні тести на емуляторі не гарантують, що додаток буде працювати без збоїв на реальних пристроях. Щоб підключитися до реальних мобільних пристроїв і інтегрувати туди Автотест, використовують ферми BrowserStack, Xamarin або AWS. Або можна підняти власну ферму на базі OpenSTF – це дозволить всім співробітникам мати рівний доступ до тестових пристроїв, що особливо важливо в умовах розподілених команд і віддаленої роботи.Всі функції, які були перевірені – занесені у систему відслідковування помилок до таблиці 4.2.



Таблиця 4.2 –Дефекти мобільного тестування сайту

# ВИСНОВКИ

У процесі виконання випускної кваліфікаційної роботи був виконаний аналіз предметної області, в рамках якого було проведено огляд існуючих веб-додатків, які здійснють продаж товарів, завдяки чому виявлено структура і основні функції системи, що розробляється.Зроблено аналіз варіантів використання інформаційної системи електронної комерції автозапчастин, розглянуті програмні засоби проектування та розробки.

В результаті була розроблена інформаційна система магазину автозапчастин, яка дозволяє здійснювати продаж товарів в інтернеті, надає зручний інтерфейс для адміністрування магазину, а інтеграція з картами допомогає клієнту знайти найближчий магазин.

Тестування програмних засобів полягало у визначенні методу тестування і виконання тестів. Результати функціонального тестування показали коректність роботи системи.

Завдяки фреймворку Django вдалося значно прискорити розробку за рахунок реалізації компонентного підходу і принципу DRY. Компонентна архітектура спростила визначення і підтримання єдиного стилю додатки і правил поведінки.

Розроблена система в подальшому може бути вдосконалена впровадженням в додаток можливостей з аналізу продажів, наприклад з використанням кластеризації або асоціативних зв'язків.

# CПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Елементи інтернет–магазину [Електронний ресурс]. – URL:<http://www.web.starport.ru/ecommerce/pg3_0.php>
2. Златопольский Д.М. Основи програмування на мові Python. - М .: ДМК Пресс, 2017. - 284 с.
3. Рейтц К., Шлюссер Т. Автостопом по Python. - СПб .: Пітер, 2017. - 336 с .
4. Геддіс Т. Починаємо програмувати на Python. - 4 вид .: Пер. з англ. - СПб .: БХВ-Петербург, 2019. - 768 с.
5. Model View Controller [Електронний ресурс]. – URL: https://ru.wikipedia.org/wiki/Model–View–Controller
6. Фреймворк Django [Електронний ресурс]. – URL:<http://djbook.ru/rel1.8/intro/overview.html>
7. Загальний огляд архітектури Django [Електронний ресурс]. – URL:<http://kutaloweb.com/jeff_forcier_glava_3/obschiy_obzor_arhitektury_django/>
8. Розробка сайтів з використанням HTML5 [Електронний ресурс]. – URL: https: //[www.prcomm–spb.ru/html–5.html](http://www.prcomm-spb.ru/html-5.html)
9. СSS3 властивості [Електронний ресурс]. – URL: https://webformyself.com/sss3–svojstva/
10. Сучасний підручник Javascript [Електронний ресурс]. – URL: https://learn.javascript.ru/
11. Порівняння систем управління базами даних [Електронний ресурс].

– URL: [http://devacademy.ru/posts/sqlite–vs–mysql–vs–postgresql/](http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/)

1. Діаграма варіантів використання [Електронний ресурс]. – URL:<http://www.intuit.ru/studies/courses/32/32/lecture/1004>
2. Основи побудови інтерфейсів [Електронний ресурс]. – URL:[http://www.info–system.ru/interface/begin\_design\_interface.html](http://www.info-system.ru/interface/begin_design_interface.html)
3. Принципи розробки інтерфейсу [Електронний ресурс]. – URL:[http://www.cmsmagazine.ru/library/items/usability/principles–of–user–interface–](http://www.cmsmagazine.ru/library/items/usability/principles-of-user-interface-)design /
4. Функціонально–вартісний аналіз [Електронний ресурс]. – URL:<http://www.iso.ru/print/rus/document5954.phtml>
5. Метрики коду програмного забезпечення [Електронний ресурс]. – URL: https: //[www.viva64.com/ru/a/0045/](http://www.viva64.com/ru/a/0045/)
6. Якісний аналіз програмного модуля на основі метрик коду [Електронний ресурс]. – URL: https://vunivere.ru/work18973
7. Тестування програмного забезпечення [Електронний ресурс]. – URL:<http://www.dpgrup.ru/testing.htm>

# ДОДАТОК А ЛІСТИНГ КОДУ ДЛЯ РЕАЛІЗАЦІЇ ДОМАШНЬОЇ СТОРІНКИ

{% extends 'base/base.html' %}  
{% load static %}  
  
{% block content %}  
<section id="home–section" class="hero">  
<div class="home–slider owl–carousel">  
<div class="slider–item" style="background–image: url({% static 'images/bg\_111.jpg' %});">  
<div class="overlay"></div>  
<div class="container">  
<div class="row slider–text justify–content–center align–items–center" data–scrollax–parent="true">  
  
<div class="col–md–12 ftco–animate text–center">  
<h1 class="mb–2">Мы поставляем только лучшие детали</h1>  
<h2 class="subheading mb–4">Всё по ходовой &amp; подвеска</h2>  
<p><a href="#" class="btn btn–primary">Узнать детали</a></p>  
</div>  
  
</div>  
</div>  
</div>  
  
<div class="slider–item" style="background–image: url({% static 'images/bg\_211.jpeg' %});">  
<div class="overlay"></div>  
<div class="container">  
<div class="row slider–text justify–content–center align–items–center" data–scrollax–parent="true">  
  
<div class="col–sm–12 ftco–animate text–center">  
<h1 class="mb–2">100% Гарантия &amp; Товары от производителя</h1>  
<h2 class="subheading mb–4">Всё по ходовой &amp; подвеска</h2>  
<p><a href="#" class="btn btn–primary">Узнать детали</a></p>  
</div>  
  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<section class="ftco–section">  
<div class="container">  
<div class="row no–gutters ftco–services">  
<div class="col–md–3 text–center d–flex align–self–stretch ftco–animate">  
<div class="media block–6 services mb–md–0 mb–4">  
<div class="icon bg–color–1 active d–flex justify–content–center align–items–center mb–2">  
<span class="flaticon–shipped"></span>  
</div>  
<div class="media–body">  
<h3 class="heading">Бесплатная доставка</h3>  
<span>On order over $100</span>  
</div>  
</div>  
</div>  
<div class="col–md–3 text–center d–flex align–self–stretch ftco–animate">  
<div class="media block–6 services mb–md–0 mb–4">  
<div class="icon bg–color–2 d–flex justify–content–center align–items–center mb–2">  
<span class="flaticon–diet"></span>  
</div>  
<div class="media–body">  
<h3 class="heading">Возврат</h3>  
<span>Возможен возврат товара</span>  
</div>  
</div>  
</div>  
<div class="col–md–3 text–center d–flex align–self–stretch ftco–animate">  
<div class="media block–6 services mb–md–0 mb–4">  
<div class="icon bg–color–3 d–flex justify–content–center align–items–center mb–2">  
<span class="flaticon–award"></span>  
</div>  
<div class="media–body">  
<h3 class="heading">Супер качество</h3>  
<span>Качество продуктов</span>  
</div>  
</div>  
</div>  
<div class="col–md–3 text–center d–flex align–self–stretch ftco–animate">  
<div class="media block–6 services mb–md–0 mb–4">  
<div class="icon bg–color–4 d–flex justify–content–center align–items–center mb–2">  
<span class="flaticon–customer–service"></span>  
</div>  
<div class="media–body">  
<h3 class="heading">Поддержка</h3>  
<span>24/7 Поддержка</span>  
</div>  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<section class="ftco–section ftco–category ftco–no–pt">  
<div class="container">  
<div class="row">  
<div class="col–md–8">  
<div class="row">  
<div class="col–md–6 order–md–last align–items–stretch d–flex">  
<div class="category–wrap–2 ftco–animate img align–self–stretch d–flex" style="background–image: url({% static 'images/category11.jpg' %});">  
<div class="text text–center">  
<h2>Запчасти</h2>  
<p>Покупай качество</p>  
<p><a href="#" class="btn btn–primary">Купить сейчас</a></p>  
</div>  
</div>  
</div>  
<div class="col–md–6">  
<div class="category–wrap ftco–animate img mb–4 d–flex align–items–end" style="background–image: url({% static 'images/category–111.jpg' %});">  
<div class="text px–3 py–1">  
<h2 class="mb–0"><a href="#">Комплектующие</a></h2>  
</div>  
</div>  
<div class="category–wrap ftco–animate img d–flex align–items–end" style="background–image: url({% static 'images/category–211.jpg' %});">  
<div class="text px–3 py–1">  
<h2 class="mb–0"><a href="#">Запчасти</a></h2>  
</div>  
</div>  
</div>  
</div>  
</div>  
  
<div class="col–md–4">  
<div class="category–wrap ftco–animate img mb–4 d–flex align–items–end" style="background–image: url({% static 'images/category–311.jpg' %});">  
<div class="text px–3 py–1">  
<h2 class="mb–0"><a href="#">Детали</a></h2>  
</div>  
</div>  
<div class="category–wrap ftco–animate img d–flex align–items–end" style="background–image: url({% static 'images/category–411.jpg' %});">  
<div class="text px–3 py–1">  
<h2 class="mb–0"><a href="#">Аксесуары</a></h2>  
</div>  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<section class="ftco–section">  
<div class="container">  
<div class="row justify–content–center mb–3 pb–3">  
<div class="col–md–12 heading–section text–center ftco–animate">  
<span class="subheading">Рекомендуемые товары</span>  
<h2 class="mb–4">Наша продукция</h2>  
<p>Доставка со всего мира, только оригинал</p>  
</div>  
</div>  
</div>  
<div class="container">  
<div class="row">  
{% if products %}  
 {% for product in products %}  
<div class="col–md–6 col–lg–3 ftco–animate">  
<div class="product">  
<a href="{% url 'product–details' product.id %}" class="img–prod"><img class="img–fluid" src="{{ product.photo.url }}" alt="Colorlib Template">  
<div class="overlay"></div>  
</a>  
<div class="text py–3 pb–4 px–3 text–center">  
<h3><a href="{% url 'product–details' product.id %}">{{ product.name }}</a></h3>  
<div class="d–flex">  
<div class="pricing">  
<p class="price"><span class="price–sale">${{ product.price }}</span></p>  
</div>  
</div>  
<div class="bottom–area d–flex px–3">  
<div class="m–auto d–flex">  
<a href="#" class="add–to–cart d–flex justify–content–center align–items–center text–center">  
<span><i class="ion–ios–menu"></i></span>  
</a>  
<a href="{% url 'cart' %}" class="buy–now d–flex justify–content–center align–items–center mx–1">  
<span><i class="ion–ios–cart"></i></span>  
</a>  
<a href="#" class="heart d–flex justify–content–center align–items–center ">  
<span><i class="ion–ios–heart"></i></span>  
</a>  
</div>  
</div>  
</div>  
</div>  
</div>  
{% endfor %}  
 {% else %}  
<h2 style="margin: auto">Простите! Товар не доступен!</h2>  
{% endif %}  
</div>  
</div>  
</section>  
  
<section class="ftco–section img" style="background–image: url({% static 'images/bg3111.jpg' %});">  
<div class="container">  
<div class="row justify–content–end">  
<div class="col–md–6 heading–section ftco–animate deal–of–the–day ftco–animate">  
<span class="subheading">Лучшая цена для тебя</span>  
<h2 class="mb–4">Товар дня</h2>  
<p>Доставка со всего мира, только оригинал</p>  
<h3><a href="#">Омыватель стекла</a></h3>  
<span class="price">$10 <a href="#">сейчас $5 за литр</a></span>  
<div id="timer" class="d–flex mt–5">  
<div class="time" id="days"></div>  
<div class="time pl–3" id="hours"></div>  
<div class="time pl–3" id="minutes"></div>  
<div class="time pl–3" id="seconds"></div>  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<section class="ftco–section testimony–section">  
<div class="container">  
<div class="row justify–content–center mb–5 pb–3">  
<div class="col–md–7 heading–section ftco–animate text–center">  
<span class="subheading">Отзывы</span>  
<h2 class="mb–4">Наш довольный клиент говорит</h2>  
<p>Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in</p>  
</div>  
</div>  
<div class="row ftco–animate">  
<div class="col–md–12">  
<div class="carousel–testimony owl–carousel">  
<div class="item">  
<div class="testimony–wrap p–4 pb–5">  
<div class="user–img mb–5" style="background–image: url({% static 'images/person\_1.jpg' %})">  
<span class="quote d–flex align–items–center justify–content–center">  
<i class="icon–quote–left"></i>  
</span>  
</div>  
<div class="text text–center">  
<p class="mb–5 pl–4 line">Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts.</p>  
<p class="name">Garreth Smith</p>  
<span class="position">Marketing Manager</span>  
</div>  
</div>  
</div>  
<div class="item">  
<div class="testimony–wrap p–4 pb–5">  
<div class="user–img mb–5" style="background–image: url({% static 'images/person\_2.jpg' %})">  
<span class="quote d–flex align–items–center justify–content–center">  
<i class="icon–quote–left"></i>  
</span>  
</div>  
<div class="text text–center">  
<p class="mb–5 pl–4 line">Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts.</p>  
<p class="name">Garreth Smith</p>  
<span class="position">Interface Designer</span>  
</div>  
</div>  
</div>  
<div class="item">  
<div class="testimony–wrap p–4 pb–5">  
<div class="user–img mb–5" style="background–image: url({% static 'images/person\_3.jpg' %})">  
<span class="quote d–flex align–items–center justify–content–center">  
<i class="icon–quote–left"></i>  
</span>  
</div>  
<div class="text text–center">  
<p class="mb–5 pl–4 line">Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts.</p>  
<p class="name">Garreth Smith</p>  
<span class="position">UI Designer</span>  
</div>  
</div>  
</div>  
<div class="item">  
<div class="testimony–wrap p–4 pb–5">  
<div class="user–img mb–5" style="background–image: url({% static 'images/person\_1.jpg' %})">  
<span class="quote d–flex align–items–center justify–content–center">  
<i class="icon–quote–left"></i>  
</span>  
</div>  
<div class="text text–center">  
<p class="mb–5 pl–4 line">Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts.</p>  
<p class="name">Garreth Smith</p>  
<span class="position">Web Developer</span>  
</div>  
</div>  
</div>  
<div class="item">  
<div class="testimony–wrap p–4 pb–5">  
<div class="user–img mb–5" style="background–image: url({% static 'images/person\_1.jpg' %})">  
<span class="quote d–flex align–items–center justify–content–center">  
<i class="icon–quote–left"></i>  
</span>  
</div>  
<div class="text text–center">  
<p class="mb–5 pl–4 line">Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts.</p>  
<p class="name">Garreth Smith</p>  
<span class="position">System Analyst</span>  
</div>  
</div>  
</div>  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<hr>  
  
<section class="ftco–section ftco–partner">  
<div class="container">  
<div class="row">  
<div class="col–sm ftco–animate">  
<a href="#" class="partner"><img src="{% static 'images/partner–1.png' %}" class="img–fluid" alt="Colorlib Template"></a>  
</div>  
<div class="col–sm ftco–animate">  
<a href="#" class="partner"><img src="{% static 'images/partner–2.png' %}" class="img–fluid" alt="Colorlib Template"></a>  
</div>  
<div class="col–sm ftco–animate">  
<a href="#" class="partner"><img src="{% static 'images/partner–3.png' %}" class="img–fluid" alt="Colorlib Template"></a>  
</div>  
<div class="col–sm ftco–animate">  
<a href="#" class="partner"><img src="{% static 'images/partner–4.png' %}" class="img–fluid" alt="Colorlib Template"></a>  
</div>  
<div class="col–sm ftco–animate">  
<a href="#" class="partner"><img src="{% static 'images/partner–5.png' %}" class="img–fluid" alt="Colorlib Template"></a>  
</div>  
</div>  
</div>  
</section>  
  
<section class="ftco–section ftco–no–pt ftco–no–pb py–5 bg–light">  
<div class="container py–4">  
<div class="row d–flex justify–content–center py–5">  
<div class="col–md–6">  
<h2 style="font–size: 22px;" class="mb–0">Subcribe to our Newsletter</h2>  
<span>Get e–mail updates about our latest shops and special offers</span>  
</div>  
<div class="col–md–6 d–flex align–items–center">  
<form class="subscribe–form" method=POST>  
<div class="form–group d–flex">  
{% csrf\_token %}  
 {{ forms.email }}  
<input type="submit" value="Subscribe" class="submit px–3">  
</div>  
</form>  
</div>  
</div>  
</div>  
</section>  
{% endblock %}

# ДОДАТОК Б ФРАГМЕНТ ЛІСТИНГУ КОДУ ДЛЯ РЕАЛІЗАЦІЇ СТИЛЕЙ ПРОЕКТУ

:root {  
––blue: #007bff;  
––indigo: #6610f2;  
––purple: #6f42c1;  
––pink: #e83e8c;  
––red: #dc3545;  
––orange: #fd7e14;  
––yellow: #ffc107;  
––green: #28a745;  
––teal: #20c997;  
––cyan: #17a2b8;  
––white: #fff;  
––gray: #6c757d;  
––gray–dark: #343a40;  
––primary: #007bff;  
––secondary: #6c757d;  
––success: #28a745;  
––info: #17a2b8;  
––warning: #ffc107;  
––danger: #dc3545;  
––light: #f8f9fa;  
––dark: #343a40;  
––breakpoint–xs: 0;  
––breakpoint–sm: 576px;  
––breakpoint–md: 768px;  
––breakpoint–lg: 992px;  
––breakpoint–xl: 1200px;  
––font–family–sans–serif: –apple–system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans–serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji";  
––font–family–monospace: SFMono–Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace; }  
  
\*,  
\*::before,  
\*::after {  
–webkit–box–sizing: border–box;  
 box–sizing: border–box; }  
  
html {  
 font–family: sans–serif;  
 line–height: 1.15;  
–webkit–text–size–adjust: 100%;  
–webkit–tap–highlight–color: rgba(0, 0, 0, 0); }  
  
article, aside, figcaption, figure, footer, header, hgroup, main, nav, section {  
 display: block; }  
  
body {  
 margin: 0;  
 font–family: –apple–system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans–serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji";  
 font–size: 1rem;  
 font–weight: 400;  
 line–height: 1.5;  
 color: #212529;  
 text–align: left;  
 background–color: #fff; }  
  
[tabindex="–1"]:focus {  
 outline: 0 !important; }  
  
hr {  
–webkit–box–sizing: content–box;  
 box–sizing: content–box;  
 height: 0;  
 overflow: visible; }  
  
h1, h2, h3, h4, h5, h6 {  
 margin–top: 0;  
 margin–bottom: 0.5rem; }  
  
p {  
 margin–top: 0;  
 margin–bottom: 1rem; }  
  
abbr[title],  
abbr[data–original–title] {  
 text–decoration: underline;  
–webkit–text–decoration: underline dotted;  
 text–decoration: underline dotted;  
 cursor: help;  
 border–bottom: 0;  
 text–decoration–skip–ink: none; }  
  
address {  
 margin–bottom: 1rem;  
 font–style: normal;  
 line–height: inherit; }  
  
ol,  
ul,  
dl {  
 margin–top: 0;  
 margin–bottom: 1rem; }  
  
ol ol,  
ul ul,  
ol ul,  
ul ol {  
 margin–bottom: 0; }  
  
dt {  
 font–weight: 700; }  
  
dd {  
 margin–bottom: .5rem;  
 margin–left: 0; }  
  
blockquote {  
 margin: 0 0 1rem; }  
  
b,  
strong {  
 font–weight: bolder; }  
  
small {  
 font–size: 80%; }  
  
sub,  
sup {  
 position: relative;  
 font–size: 75%;  
 line–height: 0;  
 vertical–align: baseline; }  
  
sub {  
 bottom: –.25em; }  
  
sup {  
 top: –.5em; }  
  
a {  
 color: #007bff;  
 text–decoration: none;  
 background–color: transparent; }  
 a:hover {  
 color: #0056b3;  
 text–decoration: underline; }  
  
a:not([href]):not([tabindex]) {  
 color: inherit;  
 text–decoration: none; }  
 a:not([href]):not([tabindex]):hover, a:not([href]):not([tabindex]):focus {  
 color: inherit;  
 text–decoration: none; }  
 a:not([href]):not([tabindex]):focus {  
 outline: 0; }  
  
pre,  
code,  
kbd,  
samp {  
 font–family: SFMono–Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace;  
 font–size: 1em; }  
  
pre {  
 margin–top: 0;  
 margin–bottom: 1rem;  
 overflow: auto; }  
  
figure {  
 margin: 0 0 1rem; }  
  
img {  
 vertical–align: middle;  
 border–style: none; }  
  
svg {  
 overflow: hidden;  
 vertical–align: middle; }  
  
table {  
 border–collapse: collapse; }  
  
caption {  
 padding–top: 0.75rem;  
 padding–bottom: 0.75rem;  
 color: #6c757d;  
 text–align: left;  
 caption–side: bottom; }  
  
th {  
 text–align: inherit; }  
  
label {  
 display: inline–block;  
 margin–bottom: 0.5rem; }  
  
button {  
 border–radius: 0; }  
  
button:focus {  
 outline: 1px dotted;  
 outline: 5px auto –webkit–focus–ring–color; }  
  
input,  
button,  
select,  
optgroup,  
textarea {  
 margin: 0;  
 font–family: inherit;  
 font–size: inherit;  
 line–height: inherit; }  
  
button,  
input {  
 overflow: visible; }  
  
button,  
select {  
 text–transform: none; }  
  
button,  
[type="button"],  
[type="reset"],  
[type="submit"] {  
–webkit–appearance: button; }  
  
button::–moz–focus–inner,  
[type="button"]::–moz–focus–inner,  
[type="reset"]::–moz–focus–inner,  
[type="submit"]::–moz–focus–inner {  
 padding: 0;  
 border–style: none; }  
  
input[type="radio"],  
input[type="checkbox"] {  
–webkit–box–sizing: border–box;  
 box–sizing: border–box;  
 padding: 0; }  
  
input[type="date"],  
input[type="time"],  
input[type="datetime–local"],  
input[type="month"] {  
–webkit–appearance: listbox; }  
  
textarea {  
 overflow: auto;  
 resize: vertical; }  
  
fieldset {  
 min–width: 0;  
 padding: 0;  
 margin: 0;  
 border: 0; }  
  
legend {  
 display: block;  
 width: 100%;  
 max–width: 100%;  
 padding: 0;  
 margin–bottom: .5rem;  
 font–size: 1.5rem;  
 line–height: inherit;  
 color: inherit;  
 white–space: normal; }  
  
progress {  
 vertical–align: baseline; }  
  
[type="number"]::–webkit–inner–spin–button,  
[type="number"]::–webkit–outer–spin–button {  
 height: auto; }  
  
[type="search"] {  
 outline–offset: –2px;  
–webkit–appearance: none; }  
  
[type="search"]::–webkit–search–decoration {  
–webkit–appearance: none; }  
  
::–webkit–file–upload–button {  
 font: inherit;  
–webkit–appearance: button; }  
  
output {  
 display: inline–block; }  
  
summary {  
 display: list–item;  
 cursor: pointer; }  
  
template {  
 display: none; }  
  
[hidden] {  
 display: none !important; }  
  
h1, h2, h3, h4, h5, h6,  
.h1, .h2, .h3, .h4, .h5, .h6 {  
 margin–bottom: 0.5rem;  
 font–family: inherit;  
 font–weight: 500;  
 line–height: 1.2;  
 color: inherit; }  
  
h1, .h1 {  
 font–size: 2.5rem; }  
  
h2, .h2 {  
 font–size: 2rem; }  
  
h3, .h3 {  
 font–size: 1.75rem; }  
  
h4, .h4 {  
 font–size: 1.5rem; }  
  
h5, .h5 {  
 font–size: 1.25rem; }  
  
h6, .h6 {  
 font–size: 1rem; }  
  
.lead {  
 font–size: 1.25rem;  
 font–weight: 300; }  
  
.display–1 {  
 font–size: 6rem;  
 font–weight: 300;  
 line–height: 1.2; }  
  
.display–2 {  
 font–size: 5.5rem;  
 font–weight: 300;  
 line–height: 1.2; }  
  
.display–3 {  
 font–size: 4.5rem;  
 font–weight: 300;  
 line–height: 1.2; }  
  
.display–4 {  
 font–size: 3.5rem;  
 font–weight: 300;  
 line–height: 1.2; }