НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет міжнародних відносин

Кафедра комп’ютерних мультимедійних технологій

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

_____ Лобода С.М.

"_____"_____ 2020 р.

ДИПЛОМНА РОБОТА

(пояснювальна записка)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ

“МАГІСТР”

# Тема: “Методи та засоби проектування веб-сайту інтернет магазину мобільної техніки”

Виконавець: Волков О.С.

Керівник: Денисенко С.М.

Нормоконтролер:

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет міжнародних відносин

Кафедра комп’ютерних мультимедійних технологій

Напрям підготовки 186 "Видавництво та поліграфія"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Лобода С. М.

"___" _____ 2020 р.

ЗАВДАННЯ

на виконання дипломної роботи Волкова Олександра Сергійовича

1. Тема дипломного проекту: Методи та засоби проектування веб-сайту інтернет магазину мобільної техніки.

2. Термін виконання проекту: з 02.10.2020 по 02.12.2020

3. Вихідні данні до проєкту: теоретичні відомості видання, етапи його підготовки, практичний процес створення.

4. Зміст пояснювальної записки:

4.1. Розділ 1. Стан дослідженості проблеми розробки веб-сайтів інтернет магазинів мобільної техніки

4.2. Розділ 2. методи та засоби проєктування веб-сайту інтернет магазину мобільної техніки

4.3. Розділ 3. Практична реалізація веб-сайту інтернет -магазину

4.4. Висновок

5. Перелік обов'язкового графічного (ілюстративного) матеріалу: презентація Power Point.

Керівник дипломної роботи: _____ Денисенко С. М.

Завдання прийняв до виконання: _____ Волков О. С.

РЕФЕРАТ

Пояснювальна записка до дипломної роботи на тему “Методи та засоби проектування веб-сайту інтернет магазину мобільної техніки” складає: 93 сторінки, 57 рисунків, 2 додатки, 32 літературних джерела.

ІНТЕРНЕТ МАГАЗИН, WEB САЙТ, REACT, REDUX, TYPESCRIPT, ПРОГРАМУВАННЯ, WEB, ФРЕЙМВОРК.

**Метою** дипломної роботи є визначення та обґрунтування основних методів та засобів розробки веб-сайту інтернет магазину для продажу мобільної техніки

**Об’єкт дослідження** – веб-сайт інтернет магазину

**Предмет дослідження** – методи і засоби створення веб-сайту інтернет магазину мобільної техніки.

**Методи дослідження**: – аналіз, синтез, абстрагування, системний підход і узагальнення.

**Технічні та програмні засоби** – IDE VsCode, Google Chrome, node, npm cli.

**Рекомендації щодо використання результатів**: Інтернет магазин можна використовувати для продажу товарів.

ЗМІСТ

# ВСТУП

**Актуальність теми**. В умовах сьогодення абсолютно всі компанії і організації мають своє представлення у глобальній мережі Інтернет. Особливо критичним є перехід у веб є для компаній, що займаються реалізацією товарів. Адже інтернет магазин є значно вигіднішим ніж звичайний. Серед переваг інтернет магазинів можна виділити: дешевизна в обслуговуванні, оскільки не потрібно сплачувати за оренду приміщення з усіма входячими всюди витратами; менший обсяг обслуговуючого персоналу; можливість переглядати вміст магазину не виходячи з дому тощо.

Відповідно до цього, розробка інтернет магазину є пріоритетним напрямком для більшості компаній, що здійснюють реалізацію товарів повсякденного вжитку. Всі великі мережі мобільної і побутової техніки мають свої сайти, це говорить про те що наявність сайту є невід'ємною частиною ведення бізнесу сьогодні.

Сайти мають специфічну архітектуру і функціонал відповідно до їх категорії, і ніша інтернет магазинів не виключення. Подібні сайти мають містити такий функціонал: сортування за певними полями (ціна, популярність і т.п.); фільтрація за характеристиками та назвою; можливість додавати обрані елементі к кошик і роботи одне замовлення.

Цей функціонал є на більшості подібних сайтів і тому це фактично стандарт. Ці функцій допомагають користувачам значно простіше шукати потрібний їм товар і відповідно чим кращий юзабіліті тим більша конверсія для власника.

Важливим аспектом є адаптивність сайту, оскільки у майже всіх людей є мобільні телефони, і враховуючи потік інформації, люди звикли швидко гуглити те, що їх цікавить. Якщо хтось захоче перевірити характеристики певного телефону, а або просто по цікавитися новинками то це займе всього лишень 10 – 15 секунд доки ця людина не зайде на сайт через телефон. Ми не знаємо коли людина хоче отримати інформацію про товар, це може бути спонтанне бажання і вона не змінюватиме свої звички швидкого гуглення на користь походу в магазин. Тому наявність сайту так важлива, адже вони є у більшості компаній/конгломератів. І якщо в тебе немає сайта то знайдуть на сайт твого потенційного конкурента і куплять  в нього.

Сайти можна розроблювати в конструкторах сайтів, або наймати розробників для написання з нуля.

Сайти конструктори мають простий функціонал і тому для їх розробки не потрібно вміти програмувати, також вони швидше за все мати певну адмінку, що значно допомагаю власнику в наповненні сайту контентом і коректурі існуючого (зміна ціни, наявності на складі і т.п.). Однак такі сайти мають малу гнучкість і далеко не всякий функціонал можна підкрутити під такий проект.

Сайти що пишуться з нуля мають значно більшу гнучкість і в них можна реалізувати що забажається, однак вони і розробляються значно довше і не однією людиною.

В залежності від вмісту сайту обирається стек технологій за допомогою яких буде створюватися сайт. Оскільки ще на початку проектування потрібно передбачити напрямки можливого розширення і закласти для цього фундамент при проектування архітектури сайту.

Наразі особливо важливими стають дослідження, що дозволили б визначати та обирати найоптимальніші технології, методи та засоби проектування веб-сайтів інтернет магазинів, оскільки саме від них значним чином залежатиме якість функціонування, обслуговування та адаптованості до потреб користувача. А це безпосередньо позначиться на прибутку компанії.

Відтак, метою дипломної роботи є визначення та обґрунтування основних методів та засобів розробки веб-сайту інтернет магазину для продажу мобільної техніки.

**Завдання виконання дипломної роботи**:

1. Проаналізувати стан дослідженості проблеми розробки веб-сайтів інтернет магазинів мобільної техніки.
2. Провести аналіз вітчизняних та закордонних сайтів відповідної тематики.
3. Окреслити етапи створення веб-сайту інтернет магазину мобільної техніки.
4. Визначити методи, засоби та технології розробки веб-сайту інтернет магазину.

5. Визначити специфіку і архітектуру майбутнього проекту та здійснити практичну реалізацію веб-сайту інтернет магазину мобільної техніки.

**Об’єкт дослідження –** веб-сайт інтернет магазину

**Предмет дослідження –** методи і засоби створення веб-сайту інтернет магазину мобільної техніки.

**Методи дослідження**. Основними методами дослідження які використовувалися були методи аналізу, синтезу, абстрагування, системного підходу і узагальнення.

# 1. СТАН ДОСЛІДЖЕНОСТІ ПРОБЛЕМИ РОЗРОБКИ ВЕБ-САЙТІВ ІНТЕРНЕТ МАГАЗИНІВ МОБІЛЬНОЇ ТЕХНІКИ

## 1.1. Аналіз термінологічного апарату дослідження

Для проведення дослідження, насамперед, потрібно визначитися з формулюванням основних термінів і понять, якими будемо користуватися. Об'єктом дослідження є веб-сайт. Це широко відоме і вживане поняття. А ось такі поняття як "фреймворки", "шаблони", "фронт та бек частини веб-сайту" та інші, потребують додаткового розгляду, адже вони є важливим аспектом нашого дослідження.

Веб-сайт – це сторінка з інформацією, що розміщена в глобальній мережі.

Види веб-сайтів:

***Landing page* (односторінковий сайт)**

Посадкова сторінка або Лендінг (від англійського *landing page*) являє собою спеціально спроектовану сторінку сайту, яка покликана спонукати відвідувачів до вчинення певної дії: придбання товару, оформлення підписки, замовлення послуги тощо. Втім, не все так однозначно, адже на сьогоднішній день існує декілька видів посадочних сторінок, кожен з яких має свої певні особливості.

**Сайт візитка**

У наш час наявність власного веб-представництва в Інтернеті є одним з основних способів просування бізнесу. Різноманітність подібних представництв вражає уяву: тут і скромні персональні сторінки, і великі корпоративні ресурси, і інформаційно-розважальні портали. Однак, в тому випадку, якщо компанія тільки починає своє просування в Інтернеті, то найкращим варіантом буде розробка сайту-візитки.

**Корпоративні сайт**

Корпоративний сайт – це повноцінне представництво компанії в Інтернеті. Тут міститься повна інформація про фірму, послуги чи продукцію, які вона пропонує, тощо. Головна відмінність корпоративних сайтів від сайтів-візиток полягає в розширених функціональних можливостях а також можливої інтеграції з внутрішніми системами компанії.

**Промо-сайти**

Починаючи випуск нової продукції або запускаючи новий товарну пропозицію компанії роблять все можливе для отримання лояльності і визнання споживачів, а також для розширення кола постійних покупців або замовників. Зрозуміло, будь-яка новинка, будь то товар або послуга, потребує реклами. Власне, для просування нових пропозицій і створюють промо-сайти.

**Інтернет-магазин**

Багато компаній вже давно використовують інтернет-магазини для перенесення продажів товарів в онлайн-простір і отримання прибутку через інтернет. Від інших сайтів інтернет-магазини відрізняються, в першу чергу, тим, що тут представлено не тільки каталог товарів, а й надається можливість їх покупки безпосередньо на сайті. Популярність цього виду сайтів пояснюється тим, що відкрити свій інтернет-магазин можуть як великі компанії, так і представники малого бізнесу.

**Іміджевий сайт**

Іміджевий сайт – це сайт, основним завданням якого є створення позитивного враження про власника за допомогою візуального оформлення. У ролі замовників іміджевих сайтів можуть виступати як великі компанії, так і представники малого і середнього бізнесу, і навіть приватні особи.

**Персональний сайт**

На відміну від сайтів-візиток і корпоративних сайтів, персональні сторінки є представництвом не компаній, а приватних осіб. Найчастіше, подібні проекти відрізняються малим об'ємом, містять в собі відомості біографічного або особистого характеру, а також інформацію про рід діяльності власника або про послуги, які він надає

**On–line сервіси**

Незважаючи на те, що Інтернет став загальнодоступним зовсім недавно, вже сьогодні життя без Інтернету практично неможливо собі уявити, тим більше, що в даний час доступ до Інтернету можливий не тільки зі стаціонарних, але і з мобільних пристроїв. І якщо раніше для оплати квитанції потрібно було відстояти довгу чергу в банку, для пошуку книги доводилося йти в бібліотеку або книжковий магазин, то сьогодні все вищеперелічене і багато іншого можна зробити в Інтернеті.

**Портали**

Кажучи коротко, то портали – це максимальна кількість корисної інформації, інтерактивних сервісів, зручності для відвідувачів в одному сайті. Портали – це дуже насичені сайти, на яких можна знайти все необхідне: новини, авторські блоги, голосування, пошукові та поштові сервіси і багато іншого.

***Web-додатки***

Веб-додатками називають різні програмні продукти, доступ до яких здійснюється через браузер. Найчастіше, в розробці веб-додатків зацікавлені різні комерційні організації. Таким чином, функціональні можливості веб-додатків, як правило, включають потужні бізнес-орієнтовані інструменти.

Обраний проект відноситься до категорії інтернет магазинів.

Фреймворки – це програмне забезпечення, яке розробляється і використовується розробниками для створення додатків.

Оскільки їх часто будують, тестують та оптимізують кілька досвідчених програмних інженерів та програмістів, фреймворки є універсальними, надійними та ефективними.

Використання програмного забезпечення для розробки програм дозволяє зосередитись на високорівневій функціональності програми. Це пояснюється тим, що про будь-яку функціональність низького рівня піклується сам фреймворк.

Розробка програмного забезпечення – складний процес. Це вимагає безлічі завдань, включаючи кодування, проектування та тестування. Лише для частини кодування програмісти повинні були подбати про синтаксис, оголошення, збір сміття, оператори, винятки тощо.

Структури програмного забезпечення полегшують життя розробникам, дозволяючи їм контролювати весь процес розробки програмного забезпечення або більшу частину його з однієї платформи.

Переваги використання програмного забезпечення:

- Допомагає у встановленні кращих практик програмування та доречному використанні шаблонів дизайну
- Код є більш безпечним
- Повторного та надлишкового коду можна уникнути
- Допомагає послідовно розробляти код із меншою кількістю помилок
- Полегшує роботу над складними технологіями
- Можна створити їх програмне забезпечення або внести свій внесок у фреймворки з відкритим кодом. Отже, функціональність постійно вдосконалюється
- Кілька сегментів коду та функціональних можливостей попередньо створені та попередньо протестовані. Це робить додатки більш надійними
- Тестування та налагодження коду набагато простіше, і це можуть зробити навіть розробники, які не володіють кодом
- Час, необхідний для розробки програми, значно скорочується

Шаблони дизайну програмного забезпечення.

Шаблони проектування використовуються для представлення кращих практик, адаптованих досвідченими об’єктно-орієнтованими розробниками програмного забезпечення.

Шаблон проектування систематично показує, мотивує та пояснює загальний дизайн, який розглядає повторювані проблеми проектування в об'єктно-орієнтованих системах. Він описує проблему, рішення, коли застосовувати рішення та його наслідки. Він також надає підказки та приклади реалізації.

Сервер – це комп'ютерна програма або пристрій, що надає послугу іншій комп'ютерній програмі та її користувачеві, також відомій як клієнт. У центрі обробки даних фізичний комп'ютер, на якому працює серверна програма, також часто називають сервером [13].

Ця машина може бути виділеним сервером або використовуватись для інших цілей.

Фронт частина веб-сайту – це частина, з якою взаємодіють користувачі. Все, що ви бачите під час навігації по інтернету – від шрифтів та кольорів до випадаючих меню та повзунків – це комбінація *HTML*, *CSS* та *JavaScript*, які контролюються браузером комп'ютера.

Бек частина – це те що робить інтерфейс веб-сайту можливим, це те, де зберігаються всі ці дані. Бек частина веб-сайту складається з сервера, програми та бази даних. Бек-енд розробник розробляє і підтримує технологію, яка забезпечує ті компоненти, які разом дозволяють стороні веб-сайту, спрямованому на користувача, навіть існувати.

Бібліотека – це набір готових функцій, які встановлюються, як окремий модуль. Допомагають при розробці, оскільки не зручно постійно писати однакові шматки коду на різних проектах, тож це свого роду автоматизація і спрощення процесу розробки.

Таким чином, створюючи веб-сайт інтернет магазину, важливо особливу увагу приділяти відповідному програмному забезпеченню, бібліотекам, інтерфейсу та стратегіям взаємодії користувача з ресурсом.

## 1.2. Специфіка веб-сайтів інтернет-магазинів

Зараз в Інтернеті можна купити буквально все, від статті за пару доларів до автомобіля за сто тисяч. Однак різні товари і послуги продаються з різним успіхом, що пов'язано з особливостями Інтернету і особливою поведінкою покупців. Популярністю користуються недорогі товари в ціновому діапазоні від 15 $ до 350 $ (за даними RU метрика): книги, побутова техніка, комп'ютери, мобільні телефони, музика і відео, програмне забезпечення, меблі, ліки, парфумерія, квитки, продукти харчування, іграшки, одяг , платіжні карти та інше. Покупці в переважній більшості проживають у великих містах, таких як Київ. Платити за краще готівкою кур'єру, післяплатою, пластиковою карткою, банківським переказом або електронними грошима. За останніми тенденціями видно, що з року в рік збільшується середня вартість покупки, а торгівля в регіонах розвивається.

Все це призвело до дуже бурхливого розвитку торгівлі в Інтернеті. Вже зараз є багато онлайнових торгових компаній, таких як "*Amazon.com*" (найбільший в світі інтернет-магазин - http://www.amazon.com/ з оборотом понад 20 млрд. Доларів на рік), "*ROZETKA*" (найбільший в Україні інтернет-магазин – *http://rozetka.com.ua/)* і багато інші. Останнім часом помітна тенденція приходу в мережу великих торгових мереж з *off–line*.

**Структура інтернет-магазину: основні особливості**

Інтернет-магазин дозволяє перевести продажі товарів безпосередньо до онлайн-простору, і багато компаній сьогодні використовують саме такий тип сайтів для отримання прибутку через інтернет. Успішність роботи інтернет-магазину залежить, зрозуміло, від дуже великої кількості різних факторів, одним з яких є зокрема його структура. Давайте розглянемо, якою повинна бути структура інтернет-магазину в сучасних умовах.

Якщо структура інтернет-магазину незручна для відвідувачів, та ще й не оптимізована для просування в пошукових системах, то такий магазин втрачає майже половину своїх шансів на успіх, якщо навіть не більше. Тому важливо ретельно продумати майбутню структуру ще на ранніх етапах розробки інтренет-магазину

Звичайно, у залежності від тематики інтернет-магазинів та особливостей їх цільової аудиторії оптимальна структура в кожному випадку буде індивідуальною. Але є деякі її елементи, які в наші дні можна назвати стандартними і без яких онлайн-магазини практично не створюються.

**Головна сторінка**

Цей елемент структури інтернет-магазину є своєрідним "обличчям проекту і покликаний забезпечити відвідувачам зручне і приємне знайомство із запропонованими товарами. Як правило, головна сторінка інтернет-магазину є певним резюме всього його вмісту: відвідувачі тут можуть побачити і частину представлених товарів (наприклад, найновіші або найпопулярніші), і коротку інформацію про магазин, і якісь особливо цікаві акційні пропозиції або знижки, і контактні дані.

**Каталог товарів інтернет-магазину**

Якщо головна сторінка інтернет-магазину – це його “обличчя”, то каталог – це, звичайно ж, “серце”. Грамотно розроблена структура каталогу просто необхідна для зручної і, що важливо, результативної роботи з інтернет-магазином [15]. Зазвичай в інтернет-магазинах каталоги товарів проектуються за ієрархічним принципом.

При розробці структури каталогу інтернет-магазину вкрай важливо правильно визначити необхідну глибину вкладеності. Так, з одного боку, важливо, щоб до сторінки конкретного товару потенційний покупець зміг дістатися не більше ніж за 2 – 3 кліка, а з іншого – важливо, щоб кількість товарів, об'єднаних в одну категорію або розділ, не виявилося завеликою для зручного пошуку.

**Кошик покупця і сторінка оформлення замовлення**

Без кошика і сторінки оформлення замовлення жоден інтернет-магазин працювати не зможе, так що без них у структурі не обійтися. Але яку саме інформацію розміщувати на цих сторінках. Відповідь на це питання буде в кожній ситуації індивідуальна, але в загальному випадку можна сказати, що потрібна коротка інформація про замовлені товари (найменування, кількість, ціна), пропозиція повернутися до покупок, підказки для заповнення полів форми замовлення.

**Сторінки з інформацією про магазин, оплату й доставку, гарантії, повернення, контакти**

Користувачі інтернет-магазину не зможуть зробити покупку, якщо не отримають точної та повної інформації про те, що являє собою цей магазин, які пропонує умови оплати і доставки, які дає гарантії якості представлених товарів і на яких умовах робить повернення. Тому сторінки (або хоча б сторінка) з такою інформацією обов'язково повинні бути включені до структури інтернет-магазину. Рівень вкладеності, як правило, у подібних сторінок найвищий – перший.

**Контентний розділ**

І з точки зору користі для потенційних покупців, і з точки зору оптимізації для пошукового просування дуже важливо, щоб у структурі інтернет-магазину був присутній регулярно оновлюваний контентний розділ. Такий розділ може містити, наприклад, новини, корисні поради з використання або вибору запропонованих у магазині товарів, огляди новинок тощо. У навігаційній структурі інтернет-магазину подібний розділ зазвичай називається відповідно до його основного вмісту – “Корисні поради”, “Статті та огляди”, “Новини”, “Блог”.

**Особисті кабінети покупців**

Якщо в інтернет-магазині планується використовувати накопичувальну систему знижок або ж реалізувати якусь іншу програму лояльності для постійних покупців, то в такому магазині до структури повинні бути включені, крім іншого, особисті кабінети покупців, в яких могла б зберігатися інформація з історією замовлень, даними про накопичені бонуси, обрані товари тощо. Доступ до особистих кабінетів відкривається тільки для зареєстрованих покупців, і в інтернет-магазині передбачаються, відповідно, форми реєстрації і авторизації.

Звичайно ж, перераховані розділи і сторінки не описують повну можливу структуру інтернет-магазину, але вони дають загальне уявлення про те, які ж моменти слід урахувати, замислюючись про оптимальну структуру для свого магазину.

**Що містять кращі онлайн-магазини**

Якісний і сучасний сайт інтернет-магазину складається з величезної кількості елементів, від яких безпосередньо залежить зручність користувача, його поведінка на сайті і, як результат, бажання купити товар. Розглянемо основні з них.

Дизайн сайту інтернет-магазину. Коли ми заходимо в будь-який оффлайн-магазин, ми оцінюємо його інтер'єр – чистоту і обстановку, якість викладки магазину, кольору, які нас оточують. Все це в комплексі вже з першої секунди складає думку про магазин і ми приймаємо рішення будемо купувати що-небудь в цьому магазині або підемо в сусідній. Аналогічно відбувається з відвідувачем в інтернет-магазині. Отже, дизайн сайту грає найважливішу роль. Відвідувачу повинен бути зрозумілий дизайн для інтернет-магазину. Кольори повинні правильно поєднуватися, відповідати тематиці продукції [2].

Перед тим, як визначити зовнішній вигляд майбутнього онлайн-магазину, необхідно провести аналіз цільової аудиторії. Варто розуміти, що якщо майбутній інтернет-магазин буде продавати вузькоспеціалізований товар, то і його аудиторія буде аналогічною. Отже, дизайн повинен відповідати її бажанням, характеру. Якщо ж магазин буде продавати товари загального вжитку, наприклад, продукти харчування, то дизайн сайту повинен бути універсальним, щоб подобатися, як чоловікам, так і жінкам різного віку.

Як правило, інтернет-магазини містять велику кількість сторінок. І якщо на сайті юзабіліті відверто кульгає, то це загрожує користувачу плутаниною, а власнику сайту – втратою клієнтів. Сайт магазину повинен бути інтуїтивно зрозумілий і зручний у користуванні. Кожен інтернет-користувач вже звик до певного розташуванню навігації.

Він знає, що розділ контактів найчастіше розташовується в правому верхньому куті, а меню сайту часто починається в лівому верхньому кутку. Тому юзабіліті дуже важливо для сайту і, як результат, для зручності відвідувача. Йому повинно бути з першого погляду зрозуміло куди йому клікнути, щоб знайти те, що його цікавить. В інших випадках низький рівень юзабіліті прирікає сайт інтернет-магазину на провал.

Картка товару. Про те, якою має бути хороша картка товару ми розпорошуватися в цій статті не будемо. Якщо ви хочете докладніше заглибитися в це питання, рекомендуємо до прочитання статтю “Картка товару для інтернет-магазину”. Скажемо лише те, що від якості наповнення картки товару на 70% залежить вирішальний крок користувача [1].

Наповнення картки контентом, її дизайн, наявність закликів до дії, правильно розставлені заголовки і підзаголовки – буквально, все мотивує споживача до покупки. А, значить, все елементи повинні бути в картці товару і при цьому задіяні правильно продають елементи

Кожен елемент і функціональна можливість інтернет-магазину впливає на зручність його користування та стає мотиватором до дії споживача. Тому успішний онлайн-магазин містить необхідний набір таких елементів, а саме:

- функція пошуку по сайту за назвою товару, коду товару, певних запитах;
- фільтри для пошуку товарів за індивідуальними параметрами;
- функція онлайн-зв'язку з консультантом;
- функція порівняння товару, додавання продукту в обране;
- функції оплат з прив'язкою надійних банківських систем.

На головній сторінці розміщені хіти продажів для швидкого переходу до самої продукції, що продається. Сам же каталог товарів вміщується на одну сторінку, що дуже зручно для перегляду і пошуку.

При переході на картку товару відкривається його докладний опис. У дизайні немає нічого зайвого. На білому тлі акцентується увага на самому товарі. Є технічний опис позиції, блок рекомендованих товарів, посилання на вибір банківського сервісу для оплати, а також форма реєстрації із закликом до дії.

## 1.3. Аналіз ресурсів аналогічної тематики

Загальний відсоток продажів в інтернет-магазинах стрімко зростає. Сьогодні набагато простіше витратити 10 хвилин часу, вибрати те, що необхідно, і замовити доставку на будинок, ніж вийти в магазин і простояти в черзі на касі. При цьому купити в інтернет-магазинах можна абсолютно все: від коробки сірників до запчастин для кораблів-криголамів. Існують магазини різних масштабів – найбільші онлайн-портали *Amazon*, *Aliexpress*, *Ebay*, популярні інтернет-магазини в рамках країни (та ж *Rozetka*) або невеликі локальні магазинчики, які прагнуть до слави.

Створити хороший інтернет-магазин досить складно. Можна взяти ношу на себе і спробувати свої сили в онлайн-конструкторі, ризикую допустити ряд критичних помилок. Можна інвестувати в проект і замовити розробку онлайн-магазину в веб-студії, отримавши якісний сайт. Так чи інакше, успіх сайту і, отже, бізнесу, залежить від ряду факторів. І тут починається найцікавіше, адже більшість інтернет-магазинів закриваються вже через рік, не в силах витримати конкуренцію. Чому так відбувається? Чому деякі магазини виходять в дамки, обдаровуючи своїх власників приголомшливою конверсією і прибутком, а інші забуваються, практично, відразу після їх відкриття.

Як вже було зазначено вище, успіх інтернет-магазину залежить від ряду факторів, які, умовно, можна розділити на дві групи [4]:

- якість інтернет-магазину з точки зору його відповідності технічним і призначеним для користувача вимогам;
- якість ведення бізнесу, можливості самої компанії і т.д.

Розглянемо приклади інтернет-магазинів, що займається реалізацією мобільної техніки.

***Citrus.ua***

Інтернет-магазин електронної техніки “Цитрус” продає величезну кількість товарів, але при цьому його структура і дизайн повністю зрозумілі для користувача. Тут складно загубитися - правильна розбивка на категорії підкатегорії на інтуїтивному рівні дозволяє швидко знайти те, що необхідно (рис. 1.1).

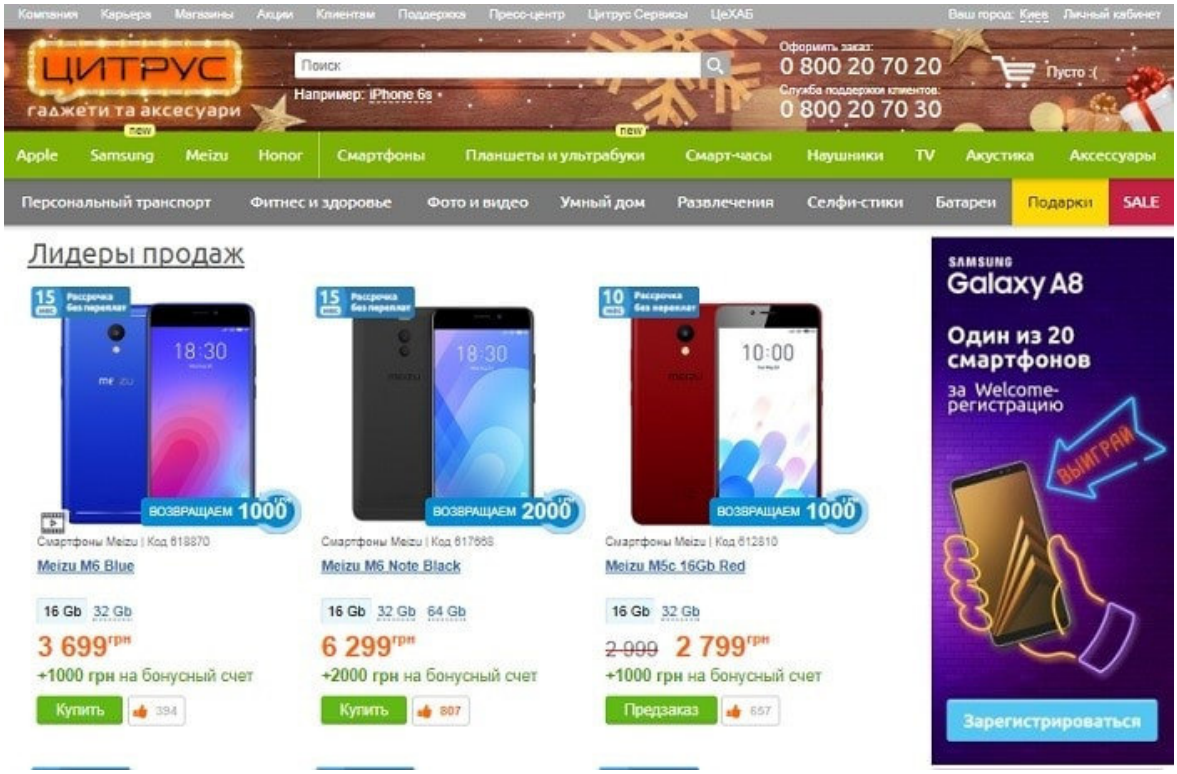



Рис. 1.1. Магазин “Цитрус”

Але окремої уваги заслуговує картка товару, яка являє собою повноцінний *landing page* (рис. 1.2).


Рис. 1.2. Картка товару магазина “Цитрус”

Тут продуманий кожен елемент, в кожному блоці є заклик до дії або тригер, який мотивує до покупки. Функціонал картки приголомшує – тут все в одному місці, не потрібно гортати сторінки для вибору відповідного варіанту моделі.

Сторінка ідеально зверстана, є вся інформація про продукт. Дуже зручно, просто і зрозуміло. А в поєднанні із закликами до дії досягається результат, заради якого і створюється продає сторінка.


Рис. 3. Сторінка інформації про певні додаткові функції

На окрему увагу заслуговує опис продукції. Воно взято з офіційного сайту продукту і дуже вигідно оформлено прямо тут на сторінці магазину. Яскраві і красиві фото лише додають ефектності (рис. 1.4).


Рис. 1.4 Інформація про телефон, магазин “Цитрус”

Переглянувши таку картку товару, у цільового відвідувача не залишається ні найменшого шансу піти з сайту почитати щось ще. І все це завдяки приголомшливому дизайну і функціоналу.

**Магазин *F.ua***

Ще один приклад відмінного українського інтернет-магазину електронної техніки. Дизайн сайту досить простий. Тут немає нічого незвичайного і відволікаючого уваги. Лише яскраві банери додають нових фарб (рис. 1.5).


Рис. 1.5. Головна сторінка магазина “*F.ua*”

Але простий і зрозумілий дизайн – лише плюс для великих сайтів з тисячами сторінок. Та й не тільки він став перевагою інтернет-магазину. Функціональні можливості сайту просто приголомшливі.

При виборі будь-якої категорії з правого боку відкривається фільтр з величезним переліком варіантів параметрів, частина з яких заточені під *SEO* (рис. 1.6).


Рис. 1.6. Сторінка фільтрації магазина “*F.ua*”

Можна знайти ідеальну модель телефону, відсортувавши його за ціною, розміром діагоналі екрана, виробнику, кольором, при цьому вказавши, що модель потрібна для роботи або в якості подарунка для дівчини. Такий фільтр максимально персоналізує покупку, дозволяючи споживачеві знайти саме те, що потрібно.

Але на цьому переваги інтернет-магазину не закінчуються. Переходимо в картку товару (рис. 1.7).

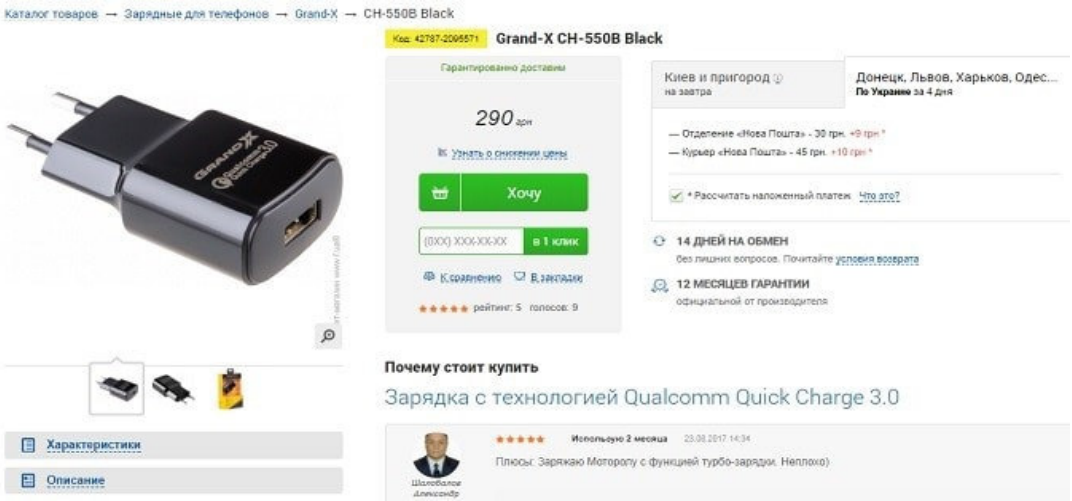



Рис. 1.7. Сторінка покупки товару в магазині “*F.ua*”

Картка досить проста і зручна. Є все необхідне з точки зору споживчого зручності. Фішка полягає в тому, що з правого боку нам відкривається зображення товару з посиланнями на його опис та характеристики. У міру пересування між

об'єктами фото з посиланнями переслідує користувача і це відмінне рішення (рисю 1.8).

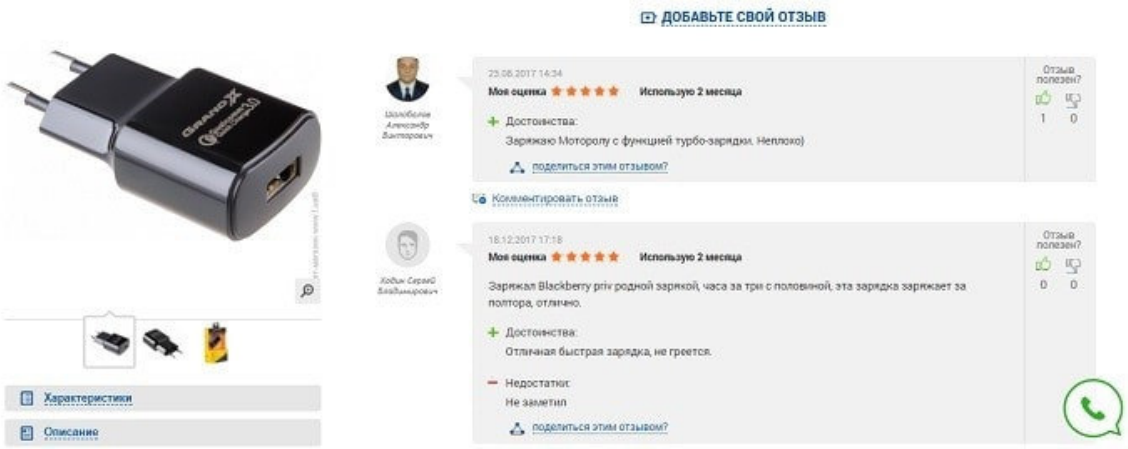



Рис. 1.8. Сторінка картки в магазині “*F.ua*”

Товар завжди на виду і можна швидко перейти до інформації про нього, що дуже зручно, особливо, якщо сторінка довга. До того ж, на картці товару відразу кілька лідогенерующих кнопок. Одна, за традицією, вгорі сторінці, а друга завжди присутня близько фотографії справа, що дозволяє відвідувачеві зробити замовлення в той момент, коли йому цього захочеться без необхідності скролити сторінку вгору.

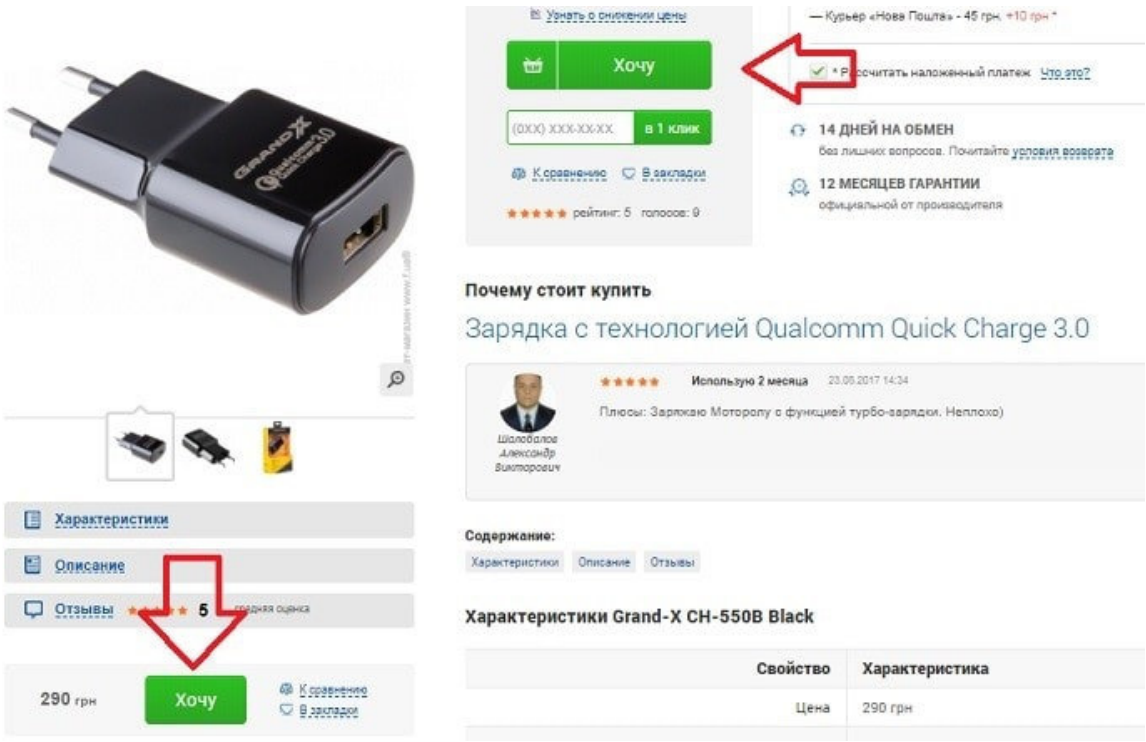



Рис. 1.9. Сторінка товару в магазині “*F.ua*”

**Магазин *Allo.ua***

Ще один український інтернет-магазин з продажу електронної та побутової техніки – АЛЛО. Яскравий і красивий дизайн виконаний у фірмовому стилі, зручна навігація і функції пошуку по фільтрам (рис. 1.10).

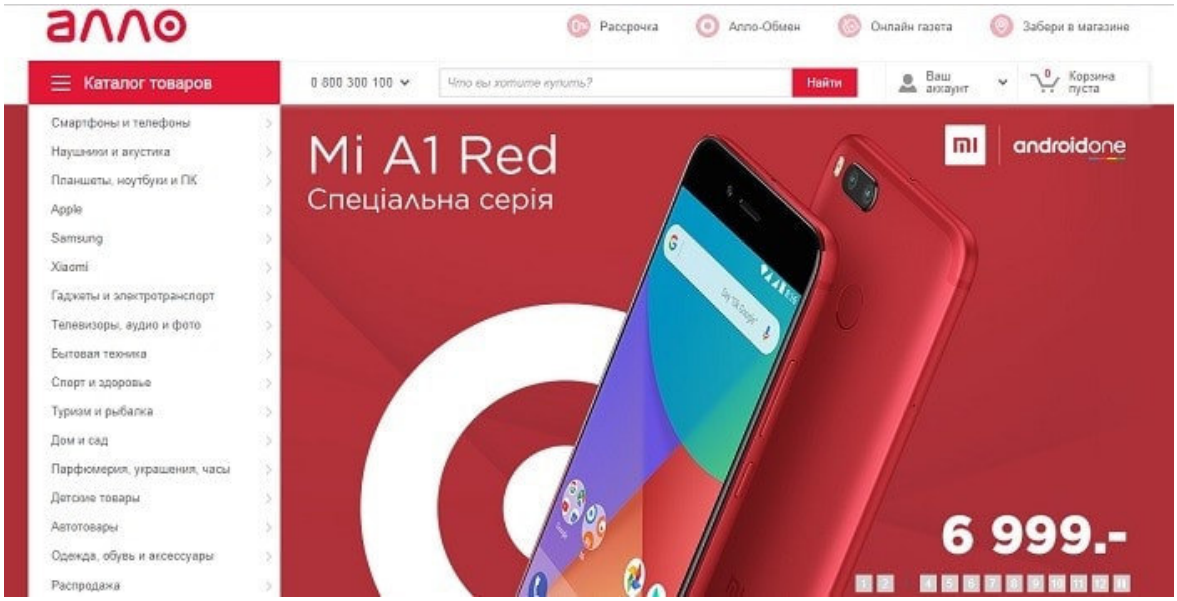



Рис. 1.10. Головна сторінка магазина “Алло”

Як і в попередніх прикладах, функціонал сайту детально продуманий. Він повністю орієнтований на покупця і створений для його комфорту. І ось яскравий тому приклад (рис. 1.11).


Рис. 1.11. Картка товару в магазині “Алло”

У картці товару, крім маркетингових продають тригерів, які мотивують людину до покупки, і зручних функцій вибору найбільш відповідної моделі та її дизайну є функція оцінки товару. При цьому оцінка виконується не в загальному за товар, а по його окремим ключовим характеристикам. Оцінка на початку сторінки є усереднена, формується виходячи оцінки клієнтів, які залишають її в своїх коментарях (рис. 12).

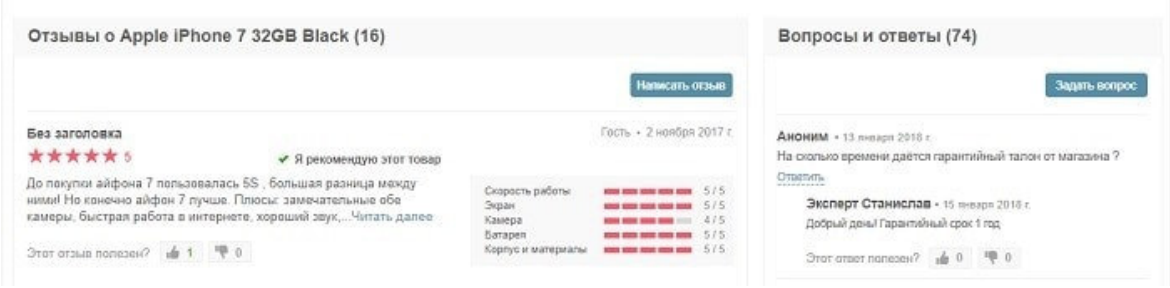



Рис. 12. Сторінка відгуків, магазин “Алло”

Це дуже зручна і корисна функція, адже на її основі можна робити суб'єктивні висновки щодо продукту. Ще один плюс в картці товару – взаємодія магазину зі своїми клієнтами. Крім відгуків тут є можливість залишити питання і отримати на нього відповідь від консультантів компанії. Відмінна ідея, як, втім, і сам магазин.

Всі наведені інтернет-магазини об'єднує одна найважливіша перевага. Крім того, що вони максимально орієнтовані на свою цільову аудиторію (про що свідчить і навігація, і дизайн, і функціонал) і мають структуру зручну для SEO-просування, що дуже важливо для ефективної розкрутки. Тому при створенні свого інтернет-магазину, обов'язково передбачте цей фактор. А якщо ви стикаєтеся з цим питанням вперше, рекомендуємо прочитати статтю як створити правильну структуру під SEO.

**Висновки до розділу**

Проведено аналіз основної термінології об’єкта дослідження. З’ясовано відмінності між фронт і бек частинами сайту, або мобільного застосунку. Наведено основні елементи програмного забезпечення для створення сайту.

Важливим аспектом є адаптивність сайту, оскільки у майже всіх людей є мобільні телефони, і враховуючи потік інформації, люди звикли швидко гуглити те, що їх цікавить. Якщо хтось захоче перевірити характеристики певного телефону, а або

просто по цікавитися новинками то це займе всього лишень 10 – 15 секунд доки ця людина не зайде на сайт через телефон. Ми не знаємо коли людина хоче отримати інформацію про товар, це може бути спонтанне бажання і вона не змінюватиме свої звички швидкого гуглення на користь походу в магазин. Тому наявність сайту так важлива, адже вони є у більшості компаній/конгломератів. І якщо в тебе немає сайта то знайдуть на сайт твого потенційного конкурента і куплять  в нього.

Сайти можна розроблювати в конструкторах сайтів, або наймати розробників для написання з нуля.

Сайти конструктори мають простий функціонал і тому для їх розробки не потрібно вміти програмувати, також вони швидше за все мати певну адмінку, що значно допомагаю власнику в наповненні сайту контентом і коректурі існуючого (зміна ціни, наявності на складі і т.п.). Однак такі сайти мають малу гнучкість і далеко не всякий функціонал можна підкрутити під такий проект.

Наведено основні приклади веб-сайтів в інтернеті, їх особливості. З’ясовано до якого з них відноситься дипломний проект. Приведено приклади реальних сайтів, розглянуто їх архітектури на прикладі *Rozetka* і *Allo*.

# 2. МЕТОДИ ТА ЗАСОБИ ПРОЄКТУВАННЯ ВЕБ-САЙТУ ІНТЕРЕНТ МАГАЗИНУ МОБІЛЬНОЇ ТЕХНІКИ

## 2.1. Етапи створення веб-сайт інтернет-магазину

**Маркетингові дослідження.** Як і будь-який бізнес, створення інтернет-магазину потрібно начитати з маркетингових досліджень. Мета цього важливого етапу - отримати інформацію, яка буде основою для успішного початку бізнесу. Потрібно обов'язково дослідити попит на продукцію, що продається, конкурентів, цільову аудиторію і інші важливі аспекти.

Правильний варіант – віддати дослідження в маркетингове агентство або, принаймні, проводити їх спільними зусиллями з фахівцем з досліджень. Економ варіант - провести дослідження власними силами, які теж дадуть багато корисної інформації. Якщо вже є чітке рішення робити магазин, можна знайти підрядника, який зможе забезпечити і розробку, і просування магазину, після чого попросити допомоги у нього в проведенні передпроєктних досліджень з метою створення якісного магазину. Спільними силами можна отримати інформацію про конкурентів і їх переваги, дізнатися очікуваний попит, останні технічні новинки в цій галузі і багато іншої інформації. Вся ця інформація повинна враховуватися як при розробки сайту магазину, так і при його просуванні.

**Комунікація.** Наступний важливий елемент даного бізнесу - організація зручною для споживачів комунікації. Мета етапу - зробити багато різних варіантів комунікації, щоб споживач з будь-якими перевагами зміг швидко зв'язатися з менеджерами магазину [17].

Перший канал комунікації - це телефон, на нього можна очікувати найбільше запитів. Зараз з телефонами особливих проблем немає: з будь-якої точки світу можна приймати дзвінки і дзвонити по прямому київським (044) або іншим номером, і коштує це зовсім недорого в залежності від технології (IP телефонія, CDMA і ін.) І телекомунікаційної компанії - 5-30 $ / міс.

Наприклад, інтернет-магазин “Зв'язковий” - http://www.svyaznoy.ru/ приймає дзвінки цілодобово. Можна організувати багатоканальну гарячу лінію (наприклад, інтернет-магазин «АЛЛО» – http://www.allo.ua/ має гарячу лінію).

Важливою частиною в магазині є *on-line* чат, який дозволяє спілкуватися за допомогою миттєвих повідомлень (наприклад, сервіс миттєвих повідомлень для сайту “*Siteheart*” - http://siteheart.com/, використовувати можна безкоштовно). У клієнтів часто виникають питання, і дуже зручно зробивши всього один клік, отримати відповідь на питання, що цікавить (наприклад, інтернет-магазин “*SVEN*” – http://shop.sven.ua/имеет on-line чат).

Наступне, на що варто звернути увагу – це меседжер, типу *Telegtam*, *Viber*, *Skype* і багато інших. Зареєструвати їх не складає труднощів. Однак для великих магазинів можу виникнути труднощі з розподілом навантаження на кожен контакт. У цьому випадку проблему можна вирішити технічно: кожному відвідувачу показується 1 – 2 контакти, але при цьому вони постійно змінюються (відвідувачеві “А” показують контакт “А”, відвідувачеві “Б” показують контакт “Б” і так по колу), таким чином , навантаження буде рівномірно розподілятися.

Потім можна згадати стару технологію *email*, яка до сих пір добре себе почуває. Мінімум потрібно створити один загальний *email* і по одному для кожного менеджера з продажу, на сайті розмістити форму зворотного зв'язку, якої до сих пір користуються деякі люди, а потім регулярно перевіряти пошту. Покупці часто пишуть електронні листи і нерідко не отримують зворотної відповіді від магазинів – це велика помилка, яка призводить до втрати клієнтів.

**Менеджери з продажу.** Чимало людей чомусь думають, що для інтернет-магазину майже не потрібні люди, зокрема, менеджери з продажу. Це оману. Будь-магазин без професійних продавців добре працювати не зможе. Уявімо ситуацію, коли дзвонить має сумнів потенційний клієнт і задає пару технічних питань: «технар» просто відповість на питання і попрощається, а “продажник” відповість на питання, потім з'ясує потреби і підбере такий продукт, який максимально підійде клієнту – звичайно, різниця величезна, особливо якщо перенести цю ситуацію на великий потік клієнтів і кінцеві прибутку [1].

Менеджери з продажу в інтернет-магазині повинні бути і повинні знати всі особливості продажів через інтернет, тому що вони значно відрізняються від традиційних продажів в реальному світі.

Доставка. За даними *ROMIR* покупці виділяють 2 основні переваги інтернет-магазинів: 68 % респондентів називають економію часу і 54 % економію грошей. Звичайно, доставка в економії часу грає велику роль, тому дуже важливо зробити її швидкої і зручної для користувачів. Покупець повинен зайти в магазин, швидко знайти потрібний товар, легко за нього заплатити і швидко його отримати у зручний йому час.

Зараз є багато служб доставки, яким це можна віддати на аутсорсинг (місцеві кур'єрські служби, міжміські, наприклад, "Автолюкс" – http://www.autolux.ua/, "*DHL*" – http://www.dhl.com/ , "Нічний експрес" – http://www.nexpress.com.ua/ і інші). Для великих магазинів можна побудувати власну службу доставки. На сайті це буде виглядати як опція при замовленні і окрема сторінка з описом доставки, а для магазину важливо.

**Оплата.** Ще один важливий для користувачів магазину елемент – оплата. В процесі оформлення покупки у користувача повинна бути можливість вибору зручного варіанту оплати. Сам процес може відбуватися по кроках: спочатку вибираємо сам спосіб оплати (готівкою, електронними грошима, пластиковою карткою і т.д.), потім система автоматично, в залежності від обраного способу, генерує рахунок на оплату, а також пропонує вибрати електронну платіжну систему або ввести дані карти і т.д. Суть в тому, щоб користувач оплатив так, як йому зручно, і витратив мінімум часу (наприклад, інтернет-магазин "Болеро" – http://www.bolero.ru/ пропонує багато різних способів оплати).

Для організації таких платежів можна використовувати популярні системи електронних платежів (наприклад, "*ASSIST*" – http://www.assist.ru/, "*ChronoPay*" – http://www.chronopay.com/, "Інтеркаса" – http: / /www.interkassa.com/ і інші, і всі вони беруть відсоток за послуги посередництва) або зробити все без посередників, прямі прийоми платежів.

**Сезонність.** Стандартна особливість торгового бізнесу - в різні пори року буде різний рівень продажів. Як правило, традиційні бізнес сезони весна і осінь, в цей час продажу вище, а "несезон" літо і зима. Також продажу деяких категорій товарів

ростуть в свята (наприклад, інтернет-магазин "*GOLD.ua*" – http: //gold.ua/ пропонує подарунки до традиційних свят на 23 лютого і 8 березня і навіть виділяє це в контекстному меню зліва).

Сезонність потрібно враховувати не тільки в плануванні продажів, але і перед запуском магазину. Розробку нового магазину варто починати не пізніше, ніж за 3 місяці до початку сезону, інакше можна не встигнути до початку сезону закінчити досягнення високого рівня продажів і втратити прибуток.

Сайт магазину. Один з найважливіших елементів успішного магазину. "Сайт повинен ефективно перетворювати відвідувачів в покупців" – стандартна фраза багатьох розробників, за якою ховається важливий сенс: на одних сайтах потенційний покупець швидко і без коливань робить покупку, а на інших не може знайти потрібний товар, хоча він там є. Для вимірювання ефективності сайту зазвичай використовують поняття "коефіцієнт конверсії", який показує співвідношення відвідувачів до покупців. На деяких сайтах цей коефіцієнт сягає значення 20 % і більше, тобто з 100 відвідувачів – 20 роблять покупку [3].

Існує цікава маркетингова модель *AIDA* (*Attention, Interest, Desire, Action* – увага, інтерес, бажання, дія), яка активно використовується на практиці в США. Її суть полягає в тому, щоб змусити відвідувача пройти 4 етапи: спочатку звернути його увагу на продукт, потім зацікавити його продуктом, після викликати бажання отримати продукт і в кінці змусити зробити дію (в нашому випадку купити продукт). Весь сайт повинен будуватися за цією моделлю, вона дуже допомагає збільшувати число покупців, але, на жаль, далеко не всі розробники знають про існування цієї моделі.

Зовні магазин повинен обов'язково викликати довіру, цьому сприяє багато чинників: популярність, вік бізнесу, хороший дизайн, сертифікати від виробників і т.д. Для Інтернету це дуже важливі елемент, зараз у багатьох користувачів рунеті все ще розвинене велика недовіра до *on-line* покупок.

**Конверсія.** Цьому показнику варто приділити особливу увагу. Про нього доведеться думати завжди: при розробці сайту, в процесі просування і в майбутніх прогнозах. Конверсія - універсальне мірило ефективності інтернет-магазину.

Середній рівень конверсії в рунеті 1 – 2%, тобто з 100 відвідувачів покупку зроблять всього 1 – 2 людини. На заході цей показник значно вищий. Пов'язано це,

перш за все, з неправильним підходом до побудови цього бізнесу у нас, вірніше до банального невміння, але про помилки розповім нижче. У той же час в деяких магазинах конверсія перевищує 20 % і більше (наприклад, за даними видання "*Marketing Charts*" – http://www.marketingcharts.com/, деякі магазини досягають конверсію в 20, 30 і навіть 40%; один з таких магазин "*ProFlowers*" – http://www.proflowers.com/ в листопаді 2009 отримав конверсію в 20,1 %, а в грудні – 22 %, і на такому рівні конверсії він знаходиться мінімум 2 роки і, до слова сказати, магазин дуже популярний) [13].

Дуже важливим для конверсії буде ціна: чим вона вища, тим менше конверсія. Це пояснює високу конверсію на сайтах з недорогою продукцією (наприклад, квіти) і низьку з дорогими товарами (наприклад, автомобілі).

Конверсія залежить, перш за все, від 3-х глобальних частин:

- Бізнес. Бренд, ціни на продукти, якість продуктів, умови оплати, доставки, гарантія тощо
- Сайт. Наскільки сайт підходить для продажів, що продає, який у нього функціонал, дизайн і т.д.
- Просування. Хто приходить на сайт, яка лояльність у відвідувачів, що шукають і т.д.

Якщо випустити з уваги хоча б одну з 3-х частин - конверсія буде низькою. Але, як не дивно, у нас примудряються втратити і 1-у, і 2-у, і, в особливо безнадійних випадках, навіть 3-ю частини чарівної формули.

Звичайно, вище описані в повному обсязі особливості, і, звичайно, є ще стандартні моменти торгового бізнесу: реальний офіс, логістика, кредитування покупців, програми лояльності та багато-багато інших важливих моментів.

Головне, щоб бізнес був цілком конкурентоспроможним на ринку, тільки в цьому випадку можна працювати над сайтом і його просуванням, і саме тому сильно виграють великі роздрібні мережі магазинів з реального світу, які багато років працювали над збільшенням конкурентоспроможності та власним брендом; таким гігантам досить тільки грамотно інтегруватися в мережу.

**Хостинг і домен.** Для звичайного інтернет-магазину з великою кількістю товару і високою передбачуваної відвідуваністю підійде простий хостинг на 5000 мб з необмеженим трафіком (100-150 $ / рік), для великого гіпермаркету потрібно брати VDS (від 200 $ / рік) або виділений сервер (від 1500 $ / рік): він і для навантаження потрібен, і багато місце на диску потрібно.

Доменне ім'я повинне бути просте, добре запам'ятовується і бажано відображати сенс магазину (наприклад, книжковий інтернет-магазин "*Books.Ru*" – http://www.books.ru/, відразу зрозуміло, що там продають, і складаються правильні асоціації).

**Платформа.** Зараз є дуже багато різних рішень для інтернет-магазинів. У всіх є переваги, недоліки і особливості (огляд CMS для інтернет-магазинів з цікавим обговоренням можна знайти тут: http: //www.nezname.com/? P = 352).

Перший варіант - створити власну систему управління сайтом (*CMS*) під свої вимоги. Цей варіант підходить в основному для дуже великих або нестандартних інтернет-магазинів, в яких потрібно враховувати багато нестандартних функцій, що найкраще робити з нуля. Однак у цього варіанту багато недоліків, варто виділити хоча б 3 основних:

- Вартість. Звичайно, створювати новий якісний рішення з нуля завжди набагато дорожче, це, напевно, основний недолік.
- Терміни. У цьому випадку вони збільшаться в кілька разів у порівнянні з розробкою на коробковому вирішенні.
- Безпека. Якщо багато коробочок CMS вже пройшли купу тестів на безпеку, то у власній системі цей момент потрібно буде враховувати своїми силами.

Через це варіант розробки власної платформи використовується дуже рідко, тим більше, що є багато якісних коробкових рішень з відкритим кодом, які можна самостійно допрацьовувати.

Другий варіант – використовувати безкоштовне коробочки рішення. Зараз є багато безкоштовних *CMS* з відкритим вихідним кодом, в тому числі і спеціалізованих для інтернет-магазинів. Одним з основних переваг такого рішення є ціна – вартість розробки буде низька, а за саму ліцензію платити нічого не потрібно. Недоліків втім,

набагато більше: відсутність технічної підтримки, часто функціональність гірше платних аналогів, можуть бути серйозні проблеми з безпекою, що для магазинів особливо погано. Різних систем багато, відразу можна подивитися в бік: “*osCommerce*”, “*PHPShop*”, “*Joomla*”, “*Drupal*” і ін.

Третій варіант – платне коробочки рішення. Вибір таких рішень на ринку дуже великий, з різною функціональністю і різними цінами. Переваг багато: офіційна технічна підтримка, хороша функціональність, висока безпека і продуктивність, безкоштовні оновлення і т.п., основний недолік один - за ліцензію потрібно заплатити, а кінцева вартість розробки буде більше, ніж при розробці на безкоштовному аналогу. У цій категорії варто звернути увагу на: “1С-Бітрікс”, “*Amiro*”, “*HostCMS*”, “*NetCat*”.

Будь-який з 3-х варіантів має своїх споживачів: свою платформу роблять великі проекти, безкоштовну *CMS* вибирають малобюджетні проекти, а платну - середні за розміром магазини з хорошими вимогами до якості. Я рекомендую дивитися в бік третього варіанту.

**Проектування і функціональність.** Це, мабуть, найважливіша частина розробки інтернет-магазину, відразу на думку спадає російське народне прислів'я: “Що посієш, те й пожнеш”. На цьому етапі дуже важливо все добре і однозначно спроектувати в майбутній інтернет-магазин. Підсумком проектування повинен стати спеціальний документ – технічне завдання, його зазвичай пишуть розробники сайту магазину, відштовхуючись від побажань клієнта [2].

ТЗ в загальному вигляді містить опис всього функціоналу, вимоги до дизайну (ці вимоги часто виносяться в окреме ТЗ, вони потребують особливо ретельного підходу, тому що саме тут процес розробки може зайти в глухий кут, якщо з'явиться нерозуміння між клієнтом і розробником щодо дизайну), макетування сторінок, вимоги до верстки, безпеки, продуктивності і т.д. Однак навіть дуже докладний і пророблена ТЗ може трохи змінюватися в процесі роботи, це цілком нормально.

В технічні подробиці не будемо вдаватися, зупинимося на спеціальних функціях, які приносять користь з точки зору продажів і тому обов'язково повинні закладатися в ТЗ. Сьогодні особливе місце потрібно відводити інформації про продаваних продуктах (за даними “*ROMIR*” такої інформації не вистачає 51 % покупців). Можна виділити десять основних функцій впливають на продажу:

- Перша така функція – фільтри за різними параметрами (наприклад, фільтр в інтернет-магазині “*Rozetka*” дозволяє впорядкувати товари по імені, ціною і іншим параметрам). Дуже зручно для вибору потрібного товару.
- Друга функція – порівняння однорідних товарів за параметрами (наприклад, таблиця порівняння програмного продукту “1С-Бітрікс” – http://www.1c-bitrix.ru/products/cms/editions/#tab-table-link). Дуже зручно при виборі потрібного продукту, а значить, допомагає потенційному клієнтові стати реальним.
- Третя функція – функціонал для демонстрації товарів. Важливо дати клієнту можливість по максимуму оцінити товар, для цього, в залежності від товару, підійдуть можливості подивитися фотографії, відео (наприклад, інтернет-магазин з продажу відео “*Bidеo-Shop*” – http://videoshop.com.ua/catalog/Filmi_DVD_Video/Rozhdestvenskaya_istoriya__DVD_.html пропонує переглянути трейлери продаваних фільмів) і багато іншого. Окремо можна виділити інтерактивне зміна параметрів, хоча це швидше перспектива майбутнього; Зараз подібних прикладів дуже мало, але даний функціонал може бути вельми корисним для користувачів, а зробити його не складно (наприклад, змінити колір у віртуального автомобіля або по заданим параметрам подивитися, як буде виглядати новий плазмовий телевізор на стіні власної кімнати, або певні квітам підібрати поєднуються відтінки для нового товару і т.д.). Допомагає покупцеві прийняти рішення.
- Четверта функція - розширений пошук за різними параметрами. Будь-який товар користувач повинен легко і швидко знайти, чи не риючись в каталозі, а просто ввівши його назву в рядок пошуку, або ж увівши потрібні йому параметри, в цьому випадку показуються схожі за функціями товари (наприклад, розширений пошук в інтернет-магазині “*Softkey*” – http://www.softkey.ru/catalog/search.php?adv=Y).
- П'ята функція - соціальні елементи. Сьогодні інтернет-магазини все більше перетворюються в своєрідні соціальні мережі. Звичайно, не варто робити черговий клон “*Facebook*” з можливістю здійснювати покупки, але безперечно варто зробити необов'язкову реєстрацію із заповненням інформації про користувачів (в тому числі інтересів), відгуки про товари, оцінок (наприклад, коментарі та оцінки користувачів є

в інтернет- магазині “*OZON.ru*” – http://www.ozon.ru/, які значно впливають на продажу), обговорення і т.д. Ці функції допоможуть при виборі товару і підвищать лояльність покупців в цілому до магазину.

- Шоста функція – спеціальні пропозиції для стимулювання збуту. Вони можуть бути як загальними для всіх покупців, так і персоналізованими, в залежності від інтересів користувача, історії покупок, активності і т.д. Загальні спеціальні пропозиції покликані підвищити попит на конкретні продукти, наприклад, ті, які залежалися на складі, і їх потрібно розпродати. А персоналізовані пропозиції покликані підвищити продажі конкретному покупцеві, наприклад, якщо користувач цікавиться маркетингом, в історії покупок у нього більше 50 % покупок книг цієї категорії, і відвідує він в магазині відповідні сторінки – йому буде не складно продати продукцію цієї або подібної тематики, причому в автоматичному режимі сайт аналізує інформацію і пропонує йому продукти, до яких у нього потенційно буде підвищений інтерес. По суті можна створити цілу аналітичну систему для цієї мети і, теоретично вона може в кілька разів збільшити прибуток інтернет-магазину, адже у всіх є приховані потреби, потрібно тільки докопатися до них.
- Сьома функція - мобільна версія сайту. У століття мобільних пристроїв потрібно обов'язково оптимізувати інтерфейс магазину під них, інакше можна втратити значну частку покупців (наприклад, мобільна версія магазину є у компанії “ТЕХНОТРЕЙД” – http://tehnotrade.com.ua/).
- Восьма функція – популярні товари в інтернет-магазині або окремої категорії. Такі товари вже зарекомендували себе, як добре продаються, і є сенс їх виділяти із загальної маси для інших покупців, причому не забувати про різних часових відрізках, іноді виходить, що товар залишається самим популярним цілий рік.
- Дев'ята функція – розсилка по підписці. Важливий елемент, особливо для постійних покупців, які хочуть бути в курсі новинок та акцій або очікують появи товару на складі. Безумовно, щоб не потрапити в папку спаму, потрібно продумати зручний функціонал з налаштування цієї розсилки і дати просту можливість відписатися, а за стандартом вона повинна бути ненав'язлива (наприклад, розсилка є у

інтернет-магазину “*Walmart*” – http://www.walmart .com /, правда відписатися від неї проблематично).

- Десята функція – інтеграція інтернет-магазину зі сторонніми системами. Ця функція побічно впливає на продажу, але при цьому значно допомагає роботі магазину. Не завжди, але досить часто потрібно інтеграція з *1C*, *ERP*, *CRM* і т.д. Це не складно зробити, у багатьох коробкових *CMS* є стандартний функціонал для інтеграції з деякими продуктами.

**Дизайн.** Чи не найголовніший, але досить важливий елемент сучасного інтернет-магазину, на який часто не звертають уваги. Особливо важливо правильно подати потрібну інформацію, часто інтернет-магазини настільки перевантажені інформацією, що знайти потрібний товар стає складно.

Дизайн повинен бути сучасним, легким, повітряним, з мінімальною графікою і без анімації, в даному випадку красива графіка і анімація будуть тільки відволікати користувача від товарів, що продаються, хоча в деяких товарних категоріях графіка цілком допустима. Важливо, щоб дизайн був розрахований на цільову аудиторію магазину, а не замовника (наприклад, хороший інформаційний дизайн у інтернет-магазину “ПАРФЮМЕРІЯ” – http://www.parfumeria.ua/ без надмірностей в графіку).

Основний упор в дизайні потрібно робити на інтерфейс і юзабіліті, сайт магазину повинен бути зручний для пошуку інформації, правильно доносити цю інформацію користувачеві і мотивувати зробити покупку або зробити запит в магазин. При цьому робити це демократично, не нав'язуючи будь-які дії, а лише надавати можливість їх здійснити.

Верстка. Цьому етапу замовник зазвичай приділяє мало уваги, тому що на перший погляд він нічого особливого не несе в собі. Здавалося б, немає нічого складного: з готового дизайну зверстати шаблон майбутнього магазину. Однак, тут варто звернути увагу на дві пов'язані між собою речі: по-перше, верстка повинна відповідати стандартам *W3C* (перевірити це можна тут: http://validator.w3.org/ – потрібно тільки ввести *url* сторінки), а по-друге , сайт повинен правильно відображатися усіма популярними браузерами (перевірити можна тут:

http://browsershots.org/ – зараз найпопулярніші браузери *Internet Explorer* (6,7,8), *Firefox (3x), Google Chrome (3x), Safari (4x ) і Opera (9x, 10x)).*

**Програмування.** Для інтернет-магазину досить великий і складний етап, навіть якщо ми будемо його розробляти на коробковому продукті. Часто застосовують ітераційний підхід, коли спочатку розробляється ядро (система з мінімальним функціоналом), а потім до цього ядра допрацьовуються додаткові функції, таким чином, проект можна запустити в обмеженому функціонал в будь-який момент після завершення ядра.

На даному етапі варто звернути особливу увагу на безпеку системи в цілому. Якщо використовується платна версія коробочки *CMS*, наприклад, *1С*-Бітрікс – вона вже добре захищена і пройшла багато тестів з безпеки, весь новий функціонал потрібно буде обов'язково окремо протестувати на *XSS*, *SQL* injection та ін.

Для великих інтернет-магазинів важливо подбати про продуктивність системи, вони повинні витримувати великі навантаження. Знову ж коробкові CMS вже оптимізовані, а хороший виділений сервер буде працювати стабільно, так що це швидше відноситься до самопісний *CMS* і модулів.

Важливо, щоб весь новий програмний код і особливо нові модулі коментувалися в коді. Для розробника це - правила хорошого тону, а для клієнта - мінімізація ризиків на той випадок, якщо з якихось причин доведеться міняти розробника.

Варто пам'ятати, що створений сайт буде просуватися, в тому числі і в пошукових системах, тому заздалегідь можна врахувати рекомендації по SEO від Google і Яндекс, та й взагалі, заздалегідь підготувати сайт інтернет-магазину до просування.

**Тестування.** Після того, як сам сайт готовий, його слід протестувати, щоб остаточно підтвердити якість створеного інтернет-магазину.

Для початку потрібно перевірити весь сайт на помилки (баги). Для цього спеціальна людина повинен пройтися по сайту і спробувати його використовувати по максимуму, часто робиться разом з клієнтом. Всі знайдені помилки передаються розробникам для виправлення [4].

Потім можна провести тестування навантаження (*Load-testing*) і тестування на стійкість (*Stress-testing*), це робиться за допомогою спеціальних програмних засобів.

І в кінці обов'язкове тестування безпеки, коли спеціально навчений фахівець з безпеки буде намагатися зламати або нашкодити системі, таким чином шукаючи «дірки». Всі знайдені вразливості також передаються на доопрацювання.

Контент. На готовий сайт потрібно завантажити інформацію про продаваних продуктах (тексти, фото, відео і т.д.). Це зазвичай робиться в напівавтоматичному режимі: частина інформації автоматично вивантажується з сторонніх баз даних (наприклад, інформація з 1С), а частина завантажується в ручну через систему адміністрування.

Тексти до продуктів повинні бути створеними спеціально для продажів і, в ідеалі, ще й оптимізованими під пошукові системи.

Таким чином, сайт інтернет-магазину вийде зручним, красивим, буде направляти користувачів, мотивувати на здійснення покупок, готовим до великих навантажень, з підвищеною безпекою, інтегрований в усі зовнішні системи підприємства. Залишилося тільки привести на цей чудовий сайт багато потенційних покупців і «справа в капелюсі», про що і говориться в наступному розділі.

Ринок безумовно є, і він досить значний, клієнтів багато, і вони охоче платять, темпи зростання тільки радують, і прогнози теж хороші. Створити дійсно прибутковий магазин зараз цілком реально, для цього потрібно врахувати три складові чарівної формули: комерційну частину, технічну частину і маркетингову частину. В результаті ми зможемо окупити вкладені інвестиції при оптимістичному прогнозі в перший квартал роботи, а при песимістичному – протягом року. У будь-якому випадку інтернет-магазини – дуже вигідний бізнес з високим коефіцієнтом *ROI*.

## 2.2. Основні методи та засоби проєктування веб-сайтів

Розробка веб-сайту інтернет магазину є загальним рішенням щодо інвестування серед більшості власників бізнесу, роздрібної торгівлі, підприємців та більшості комерційних підприємств, щоб перетворити конкурентні переваги інтернет-покупок у стабільне джерело доходу через професійний інтернет-магазин.

Веб-сайти електронної комерції “Інтернет-магазини” дозволяють споживачам знаходити, порівнювати та купувати бажані товари чи послуги в Інтернеті зручно вдома за кілька хвилин, за кілька кліків за допомогою своїх смартфонів, планшетів та ПК, які приймають рішення інвестувати час, зусилля та гроші на веб-сайт електронної комерції розробити процес прийняття раціонального рішення щодо збільшення вашого бізнесу за рахунок збільшення продажів та збільшення доходів.

На додаток до зручного досвіду онлайн-покупок, існує безліч інших конкурентних переваг, пропонованих веб-сайтами інтернет магазинами, які включають, але не обмежуючись ними:

1. Збільшення охоплення бізнесу за допомогою націлювання на різні місцеві та міжнародні ринки
2. Залучення більшої кількості потенційних клієнтів та збільшення обсягу продажів завдяки професійному інтернет-магазину, який працює цілодобово та без вихідних
3. Зниження операційних та маркетингових витрат, необхідних для ведення та просування фізичного магазину
4. Відстеження та аналіз поведінки споживачів “інтересів відвідувачів веб-сайту інтернет магазина” для налаштування вашої продукції та маркетингових кампаній
5. Створення більше можливостей для продажу та перехресних продажів за допомогою кампаній ремаркетингу.

Що стосується переваг розробки веб-сайту електронної комерції, щодня з’являється безліч ділових можливостей, коли з’являється професійний інтернет-магазин, налаштований відповідно до бізнес-потреб та цілей.

**Сучасні технології розробки інтернет-магазинів**

Інтернет-магазин є складним, динамічним веб-додатком. На даний момент існують безліч інструментів і технологій, що дозволяють реалізовувати такі програми.

Інтернет-магазин можна реалізувати на таких мовах програмування,

як *JavaScript, PHP, Java, Python, HTML, C, C* ++ і т.п.

Для зручної і спрощеної розробки інтернет-магазину існує клас систем управління контентом: широко використовуються системи *CMS* (*Content Management Systems*) [2].

*CMS* надають стандартні і базові набори певних функцій, придатних для створення сайтів. До того ж *CMS* мають безліч плагінів, готових до використання.

Як і будь-який інший програмний продукт, системи автоматизованого керування вмістом існують платні і безкоштовні, з відкритим і закритим вихідним кодом. Переваги платних систем складаються в постійній технічній підтримці з боку розробників, регулярних оновленнях продукту, а також більш високого ступеня безпеки в порівнянні з безкоштовними продуктами. У той же час системи, розповсюджуються безкоштовно, компенсують головний недолік платних *CMS* - їх вартість.

Переваги, які дає використання *CMS*, зручно розглянути під різними кутами: з точок зору розробника сайту і їх користувачів.

З позиції розробника сайту: завдяки наявності вже готових модулів, *CMS* дають можливість виробляти зміни в структурі сайтів якісно і оперативно шляхом компонування цих модулів. Використання модулів також забезпечує більш високу якість розробки, оскільки зменшує ймовірність появи помилок в програмному коді.

З точки зору користувача *CMS* пропонують насамперед зручний інтерфейс для редагування вмісту сайту, не вимагаючи основних знань в сфері веб-дизайну, *HTML*-верстки і т.п.

На даний момент існує безліч готових систем керування вмістом сайту. Найбільш відомі *CMS: WordPress, osCommerce, OpenCart ma Joomla, Tilda*.

В ході роботи були розглянуті найбільш поширені системи.

1С-Бітрікс – система орієнтована на корпоративні сайти, інформаційні та довідкові портали, соціальні мережі, інтернет-магазинів, сайти ЗМІ, придатна для створення інших видів веб-ресурсів.

Для зберігання даних сайту використовується файлова система сервера і реляційна СУБД. Підтримуються наступні СУБД: *MySQL, Oracle, MS SQL*. Продукт працює на *Microsoft Windows* і *UNIX*-подібних платформах, включаючи *Linux*. "1С-Бітрікс: Управління сайтом" продається в однією з восьми складених фірмою-розробником редакцій, що визначають функціональність системи і набір модулів.

Кількість модулів у встановлюваної системі залежить від редакції продукту. Головний недолік системи – вона є комерційною.

*WordPress* – система керування вмістом сайту з відкритим вихідним кодом, яка розповсюджується за ліцензією *GNU GPL*. *WordPress* написана на *PHP*, в якості бази даних використовує *MySQL*.

Сфера застосування – від блогів до досить складних новинних ресурсів і інтернет-магазинів. Вбудована система "тем" і "плагінів" разом з вдалою архітектурою дозволяє конструювати практично будь-які проекти. Система *WordPress* випущена під ліцензією *GPL* версії 2.

Універсальність *WordPress* прихована в сотнях доступних плагінів. Завдяки їм, функціонал цієї *CMS* наблизився до таких систем як "*Joomla*". Однак, установка занадто великої кількості плагінів може негативно позначитися на швидкості завантаження сторінок.

**Типи веб-сайтів інтернет магазинів**

Перш ніж обговорювати процес проектування та розробки веб-сайтів інтернет магазинів, потрібно спочатку обговоримо основні типи веб-сайтів інтернет магазинів, щоб визначити найбільш підходящий тип для потреб та цілей перед початком процесу розробки.

Основними типами веб-сайтів інтернет магазинів є:

- *B2C* – бізнес-користувач
- *B2B* – бізнес-бізнес
- *C2C* – користувач-користувач

Інтернет-магазини *B2C*: Поширений тип підприємств електронної комерції, що має веб-сайт електронної комерції, присвячений показу та просуванню товарів для кінцевих споживачів, таких як одяг та технічна продукція, включаючи мобільні телефони, ПК чи меблі.

Платформи електронної комерції *B2B*: Підприємства розробляють платформи електронної комерції в Інтернеті для просування своїх продуктів та послуг для інших підприємств, одночасно збільшуючи їх охоплення та орієнтуючись на міжнародні ринки.

Магазини електронної комерції *C2C*: Сторонні магазини, такі як *eBay*, дозволяють звичайним приватним та роздрібним продавцям просувати та продавати свою продукцію в Інтернеті кінцевим споживачам.

Існують інші типи онлайн-платформ, де приватні особи та спеціалісти пропонують свої послуги для бізнесу, наприклад, платформи-фрілансери, які, як правило, не відповідають критеріям веб-сайту електронної комерції.

Початок бізнесу в галузі електронної комерції – це вибір, який підходить для різних типів комерційних, технологічних та промислових проектів, оскільки дає змогу рекламувати та продавати фізичні товари, а також цифрові товари, такі як онлайн-курси чи електронні книги, або ексклюзивний цифровий вміст, як стокові зображення.

## 2.3. Основні типи шаблонів проєктування веб-сайтів

Шаблон проектування (патерн) в розробці програмного забезпечення – повторимо архітектурна конструкція, що представляє собою вирішення проблеми проектування в рамках деякого часто виникає контекст У порівнянні з самостійним проектуванням, шаблони мають ряд переваг. Головна користь від використання шаблонів полягає в зниженні складності розробки за рахунок готових абстракцій для вирішення цілого класу проблем. Застосування шаблонів концептуально близько до використання вже готових бібліотек.

У контексті розробки веб-додатки ми будемо використовувати класифікацію шаблонів проектування Мартіна Фаулера [3], згідно якої шаблони можна розділити по класах:

1. базові шаблони;
2. шаблони веб-уявлення;
3. шаблони архітектурних джерел даних;
4. шаблони об'єктно-реляційної логіки;
5. шаблони об'єктно-реляційного структурування;
6. шаблони логіки сутності;
7. шаблони розподілу даних;
8. шаблони локальної конкуренції.

Кожен з цих класів включає в себе певний набір шаблонів, одним з яких є MVC, а також похідні:

1. *MVP (Model – View – Presenter);*
2. *MVVM (Model – View -View – Model);*
3. *HMVC (Hierarchical MVC);*
4. *PAC (Presentation – Abstraction – Control).*

*MVC* є одним з популярних шаблонів проектування.

Більшість сучасних *CMS* є реалізацією архітектури *MVC*, яка дозволяє зосередитися на реалізації бізнес логіки, приділяючи менше слухаючи програмування.

Концепція *MVC* була описана задовго до появи *PHP*, в 1979 році. Основна мета застосування *MVC* полягає саме в тому, щоб відокремити бізнес-логіку (модель) від її візуалізації. При цьому контролер забезпечує зв'язок між користувачем і системою: відстежує дії користувача і реалізує необхідну реакцію за допомогою моделі і уявлення. Подання і контролер залежать від моделі, однак, модель не залежить ні від того, ні від іншого. Це дозволяє будувати модель незалежно від її візуального представлення, а також створювати кілька уявлень для однієї моделі [4].

**Model-View-ViewModel** (*MVVM*) – це структурний шаблон дизайну, який розділяє об'єкти на три окремі групи:

- Моделі містять дані програми. Зазвичай це конструкції або прості заняття.
- Подання відображають візуальні елементи та елементи керування на екрані. Зазвичай вони є підкласами UIView.
- Models Моделі подання перетворюють інформацію про модель у значення, які можуть відображатися у поданні. Зазвичай це класи, тому їх можна передавати як посилання.

*MVVM* з першого погляду

Розробимо MVVM на невеликі шматки і розглянемо їх окремо. Почнемо з основного будівельного блоку, який є ключовим для всіх додатків: даних та інформації. Це проводиться в моделі.

Модель – це те, що я називається об’єктом домену. Модель представляє фактичні дані та / або інформацію, з якою ми маємо справу. Прикладом моделі може бути контакт (що містить ім’я, номер телефону, адресу тощо) або характеристики пункту публікації в прямому ефірі.

Ключове, що слід пам’ятати з моделлю, полягає в тому, що вона зберігає інформацію, але не поведінку чи послуги, які маніпулюють інформацією. Він не несе відповідальності за форматування тексту, щоб виглядати красиво на екрані, або отримання списку елементів з віддаленого сервера (насправді, у цьому списку кожен елемент, швидше за все, буде власною моделлю). Бізнес-логіка зазвичай тримається

окремо від моделі та інкапсулюється в інші класи, які діють на модель. Це не завжди вірно: наприклад, деякі моделі можуть містити перевірку.

Часто є проблемою тримати модель повністю “чистою”. Під цим я маю на увазі справжнє уявлення про “реальний світ”. Наприклад, запис контакту може містити дату останньої зміни та особу користувача, що модифікується (інформація про аудит), та унікальний ідентифікатор (інформацію про базу даних чи збереження). Змінена дата не має реального значення для контакту в реальному світі, але є функцією того, як модель використовується, відстежується та зберігається в системі.

Погляд – це те, з чим знайомі більшість з нас, і єдине, з чим кінцевий користувач насправді взаємодіє. Це подання даних. Погляд має певні свободи, щоб зробити ці дані більш презентабельними. Наприклад, дата може зберігатися на моделі як кількість секунд з півночі 1 січня 1970 р. (Час *Unix*). Однак кінцевому користувачеві представлено назву місяця, дату та рік у місцевому часовому поясі. Представлення даних також може мати пов'язані з ним поведінку, наприклад, прийняття вводу користувача. Вигляд управляє введенням (натисканням клавіш, рухами миші, жестами тощо), що в кінцевому рахунку маніпулює властивостями моделі.

У *MVVM* перегляд активний. На відміну від пасивного подання, яке не має знань про модель і повністю ним керує контролер / презентатор, подання в *MVVM* містить поведінку, події та прив'язку даних, які в підсумку вимагають знання базової моделі та моделі перегляду. Хоча ці події та поведінку можуть бути зіставлені із властивостями, викликами методів та командами, представлення все ще відповідає за обробку власних подій і не передає це повністю в модель перегляду.

Варто пам’ятати про думку, що вона не відповідає за підтримку свого стану. Натомість він синхронізує це з моделью перегляду.

***ViewModel***

Модель перегляду є ключовим елементом тріади, оскільки вона вводить розділення презентацій або концепцію збереження відтінків вигляду окремо від моделі. Замість того, щоб інформувати модель про перегляд користувача датою, щоб вона перетворила дату у формат відображення, модель просто зберігає дані, подання просто утримує відформатовану дату, а контролер діє як зв’язок між двома . Контролер може взяти введення з подання і розмістити його на моделі, або він може взаємодіяти

зі службою для отримання моделі, а потім перекласти властивості та розмістити її на поданні.

Модель перегляду також виставляє методи, команди та інші моменти, які допомагають підтримувати стан вигляду, маніпулювати моделлю як результат дій над поданням та запускати події в самому поданні.

*MVVM* описується як реалізація презентаційної моделі, розробленої спеціально для WPF (і пізніше, *Silverlight*).

**Вид і модель *ViewMode***

- Вигляд та модель перегляду спілкуються за допомогою прив’язки даних, викликів методів, властивостей, подій та повідомлень
- Модель перегляду надає не тільки моделі, але й інші властивості (наприклад, інформацію про стан, наприклад, індикатор «зайнято») та команди
- Представлення обробляє власні події інтерфейсу користувача, а потім відображає їх у моделі перегляду за допомогою команд
- Моделі та властивості в viewmodel оновлюються з подання за допомогою двостороннього прив'язки даних

Два механізми, які часто враховують реалізації шаблону – це тригери (особливо тригери даних) у *WPF* та *Visual State Manager (VSM) у Silverlight*. Ці механізми допомагають реалізувати шаблон, прив'язуючи поведінку інтерфейсу користувача до базових моделей. У *Silverlight VSM* повинен бути основним вибором для координації переходів та анімації. Дізнайтеся більше про *VSM*.

***ViewModel* і модель**

Модель view повністю відповідає за модель у цьому сценарії. На щастя, це не самотність:

- Модель перегляду може виставляти модель безпосередньо або властивості, пов’язані з моделлю, для прив’язки даних
- Модель перегляду може містити інтерфейси до служб, дані конфігурації тощо для отримання та керування властивостями, які вона надає представленням

Можливо, ви чули дискусію про view first або viewmodel first. Загалом більшість розробників погоджуються з тим, що подання має мати рівно одну модель перегляду. Немає необхідності приєднувати кілька моделей перегляду до одного перегляду. Якщо подумати про розділення проблем, це має сенс, адже якщо на екрані є “віджет контакту”, прив'язаний до “моделі перегляду контактів”, і “віджет компанії”, прив'язаний до “моделі перегляду компанії”, це повинні бути окремі подання. , жодного перегляду з двома моделями перегляду.

Вигляд може складатися з інших поглядів, кожен із яких має свою модель перегляду. Моделі перегляду можуть створювати інші моделі перегляду, коли це необхідно (проте, часто я бачу, як люди складають та агрегують моделі перегляду, коли насправді те, що вони насправді хочуть – це обмін повідомленнями між моделями перегляду).

Хоча подання повинно мати лише одну модель перегляду, одна модель перегляду може використовуватися кількома переглядами (уявіть собі майстра, наприклад, який має три перегляди, але всі вони прив’язуються до тієї самої моделі перегляду, яка керує процесом).

*View First* просто означає, що саме вид є тим, що рухає створенням або відкриттям моделі перегляду. У перших сценаріях перегляду подання зазвичай прив'язується до моделі подання як ресурс, використовує шаблон локатора або модель подання вводиться за допомогою *MEF*, *Unity* або якимось іншим способом. Це дуже поширений метод управління поданнями та моделями переглядів.

Навігація – загальна проблема, з якою потрібно вирішити проблеми. Як потрібно керувати инавігацією з програми *MVVM*? У більшості прикладів на екрані відображається лише одна кнопка або віджет, і не вирішуються складені програми з кількома сторінками.

Коротка відповідь полягає в тому, що незалежно від того, як розробник орієнтується (чи використовує він власний двигун для введення подань, використовуєте систему навігації, що постачається *Silverlight*, чи використовує управління регіонами з Prism або їх комбінацію), слід абстрагувати механізм, що стоїть за інтерфейсом. Визначаючи *INavigation* або щось подібне, навігація більше не стає

проблемою *MVVM*. Якщо не вирішити це, viewmodel може імпортувати INavigation і просто перейти до або запустити перехід за необхідності.

Автовизначувані види за допомогою вільного інтерфейсу охоплюють відображення видів у регіони, а динамічне завантаження модулів за допомогою Prism має повне рішення з використанням системи навігації.

**Динамічні модулі**

Що робити, якщо надзвичайно велика програма? Часто немає сенсу завантажувати все відразу. Хочеться, щоб з'явилося головне меню та екран, а потім динамічно завантажували інші модулі, коли вони потрібні. Це скорочує початковий час для запуску та запуску програми, а також поважає браузер та / або пам'ять користувача та процесор користувача.

Питання динамічних модулів насправді не стосується *MVVM*, але обмін повідомленнями між моделями перегляду та між модулями, звичайно, важливий. Для них варто розглянути існуючі системи, такі як *MEF* та *Prism*, які вирішують конкретну проблему. *Prism* має модулі, які можна завантажувати "на вимогу", а *MEF* пропонує каталог розгортання, що дозволяє динамічно завантажувати файли *XAP*. Рішенням *Prism* для обміну повідомленнями в усіх додатках є агрегатор подій.

**Діалогове вікно**

Типовим шаблоном інтерфейсу є діалогове вікно (схоже на вікно повідомлення, але очікує відповіді). Часто люди стикаються з питаннями того, як це можна реалізувати як за допомогою *MVVM*, так і обмеження *Silverlight*, що весь код повинен бути асинхронним.

Найпростішим рішенням є абстрагування діалогового вікна, яке стоїть за інтерфейсом, і надання зворотного виклику для відповіді. Модель подання може імпортувати діалогове вікно, після чого на основі певної зміни стану або команди запускає службу діалогового вікна. Зворотний дзвінок поверне відповідь, і тоді подання може обробити відповідним чином.

**Анімація**

Вирішити цю проблему дуже часто: як можуть ініціюватись зміни в інтерфейсі користувача, а також у запуску анімації та інших переходів?

Існує кілька рішень проблеми. Ось кілька прикладів того, як вирішити проблему:

- агрегатор візуального стану дозволяє прив’язувати анімації на основі подій користувацького інтерфейсу, взагалі не залучаючи модель перегляду
- делегати анімації
- зворотна реалізація ICommand nRoute
- зразок фреймворку MVVM, який використовує візуальний менеджер стану

**Конфігурація або глобальні значення**

Ще одне питання, яке, піднімається досить часто – це як поводитися з глобальними змінними та інформацією про конфігурацію. Знову ж таки, це менше проблема *MVVM* і більше загальний розгляд архітектури. У більшості випадків можна виставити конфігурацію за допомогою інтерфейсу (*IConfiguration*), а потім підключити реалізацію зі своїми значеннями конфігурації. Будь-яка модель перегляду, яка вимагає інформації, просто імпортує реалізацію, будь то за допомогою *MEF*, *Unity* або якимсь іншим механізмом, і зберігається лише одна копія класу (шаблон Singleton, хоча, швидше за все, керований контейнером, а не самим класом).

**Асинхронні процеси**

Одним із пунктів плутанини з *Silverlight* є те, що він змушує дзвінки служби бути асинхронними. Це може здатися дивним під час побудови моделі перегляду: коли запускають виклик і як дізнатися, що він завершений? Як правило, це управляється шляхом реєстрації до події, коли процес завершується, та прив'язки результатів. Краще приховувати деталі реалізації подій за простою функцією *Action*.

Іноді може бути складніший робочий процес, який вимагає виконання та завершення декількох асинхронних викликів перед продовженням. У цьому випадку ви можете розглянути механізм спрощення кодування та читання послідовного робочого процесу.

**Величезні набори даних**

Існує твердження, що *MVVM* погано обробляє великі масиви даних. Певні реалізації мають цю проблему, а не сам шаблон. Рішенням часто є розміщення даних на сторінках, але справа в тому, що багато людей підходять до проблеми неправильно. З якоїсь причини розробники наполягають, що підкачування – це функція бази даних і повинна бути ізольована на рівні доступу до даних. Простий факт, що є елемент користувацького інтерфейсу з “поточною сторінкою” та “загальною кількістю сторінок”, припускає, що це не просто артефакт бази даних, але він бере участь у всіх рівнях програми та повинен керуватися як такий.

У найбільш примітивній формі модна створити колекцію, яка зростатиме, коли і сторінки користувача. Якщо даних мало, можна зібрати дуже велику колекцію і зберегти її в клієнті *Silverlight*, але використовувати віртуальну панель для відображення інформації (проблема деяких панелей полягає в тому, що вони створюють елемент керування для кожного пов'язаного елемента даних, який може зниження продуктивності - віртуалізовані панелі створюють лише достатньо елементів керування, щоб заповнити видиме вікно на екрані).

Такі технології, як *WCF RIA*, підтримують запити *LINQ*. Ці запити містять методи розширення, які дозволяють захоплювати лише перші кілька елементів у списку, а не отримувати повний список відразу. Фреймворк також надає допоміжні класи, такі як *PagedCollectionView*, які допомагають фільтрувати, сортувати та дані сторінки.

**Чим *MVVM* не є**

Жодне обговорення не було б повним, якби не зауважити, чим *MVVM* не є. *MVVM* не є повною структурою. Це шаблон і може бути частиною фреймворку, але це лише частина загального рішення для архітектури вашої програми. Це не стосується і насправді не хвилює того, що відбувається на вашому сервері або як ваші служби поєднані. Це робить розділення стресів проблем, що приємно.

***MVP*** (*minimum viable product*, іноді помилково розшифровується як *minimum valuable product або minimal valuable product*) – це мінімально життєздатний продукт, який дозволяє отримати осмислену зворотний зв'язок від користувачів, зрозуміти що їм потрібно і не створювати те, що їм нецікаво і за що вони не готові платити (рис. 2.1).

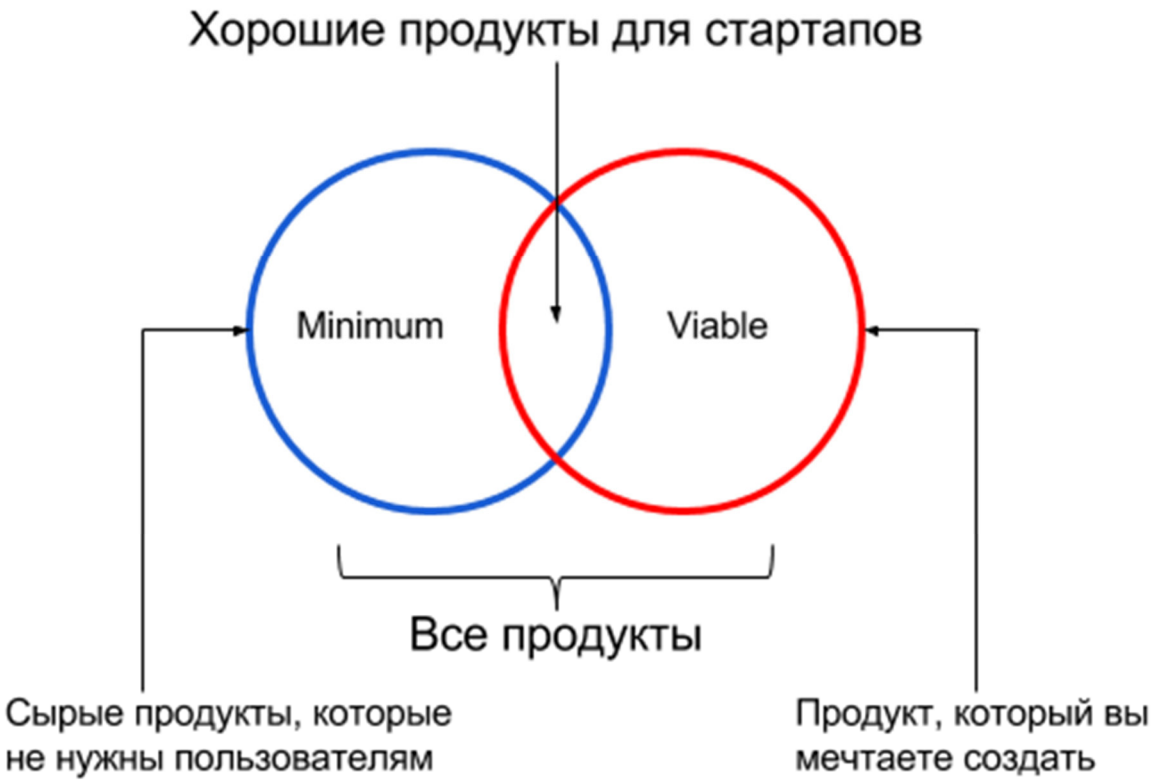



Рис. 2.1. Опис концепції *MVP*

В рамках концепції ідея вашого стартапу – це гіпотеза. Щоб перевірити її, необхідно зробити наступне:

1. Чітко сформулювати гіпотезу.
2. Визначити критерії, за якими буде визначатися її життєздатність.
3. Зробити мінімально життєздатний продукт для підтвердження гіпотези і запустити його.
4. Виміряти показники ефективності.
5. Зробити висновки і перевірити таку гіпотезу, якщо це необхідно.

*MVP* для стартапів ні в якому разі не означає сирий продукт, зроблений в поспіху. На його розробку просто витрачається мінімальний час і він містить тільки ключові функції, актуальність яких для реальних користувачів і слід перевірити. Дослідження показують, що 60 % фіч взагалі не використовуються, а, значить, і не є затребуваними серед користувачів [3]. Концепція *MVP* дозволяє скоротити час запуску проекту за

рахунок створення тільки необхідних функцій і почати отримувати реальний фідбек свого продукту (рис. 2.2).


Рис. 2.2. Концепція *MVP*

При цьому все не закінчується отриманням зворотного зв'язку. В основі методології *lean startup*, до якої належить концепція *MVP*, лежить цикл розробка-вимір-вивчення фідбек. Тому за отриманням фідбек слід доопрацювання вдалих фіч і їх повторне тестування. У разі успіху можна створювати повноцінний продукт і виходити на ринок.

Навіщо стартапу *MVP*

Який би геніальною вона не була, ідея - це не кінцевий результат. Створивши мінімально життєздатний продукт, ви зможете:

1. Заощадити гроші, не вкладаючи їх в провальний проект.
2. Перевірити, чи цікавить ваш продукт потенційних користувачів.
3. За допомогою ітерацій дізнатися, який напрямок розвитку буде найоптимальнішим.
4. Зібрати базу потенційних клієнтів і знайти ранніх прихильників (*early adopters)* свого продукту.

Стів Бланк, автор методики розвитку клієнтів (*customer development methodology*), стверджує, що головна причина провалу успішних за багатьма показниками проектів - це недостатнє знання своїх клієнтів. Замість того, щоб вивчити їх потреби ще на ранніх стадіях, розробивши *MVP product*, засновники стартапів з головою йдуть в роботу над ризикової витівкою, неперевіреної на реальних користувачів.

**Компанії, які почали свій шлях з *MVP***

*Spotify (рис. 2.3 – рис. 2.4)*

Розробники *MVP Spotify* сконцентрувалися на єдиної функції: потокової передачі музики. Вивчивши дані закритого бета-тестування програми для *Windows* засновники змогли укласти контракти з великими рекординговими лейблами і отримати значне фінансування для свого проекту. Зараз у сервісу 60 мільйонів користувачів, а його вартість оцінюють в 8,4 мільярда доларів США.

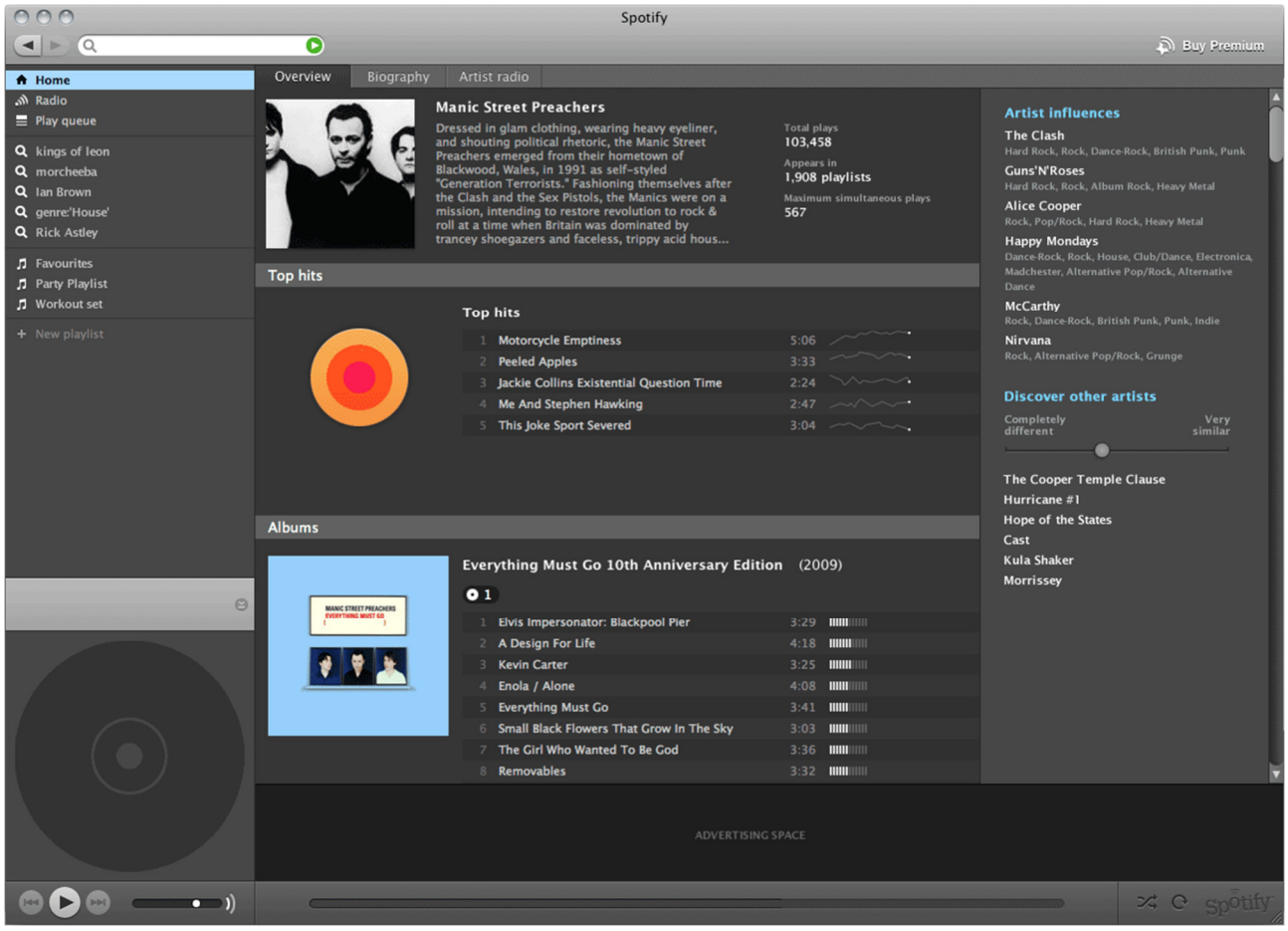



Рис. 2.3. *MVP Spotify*

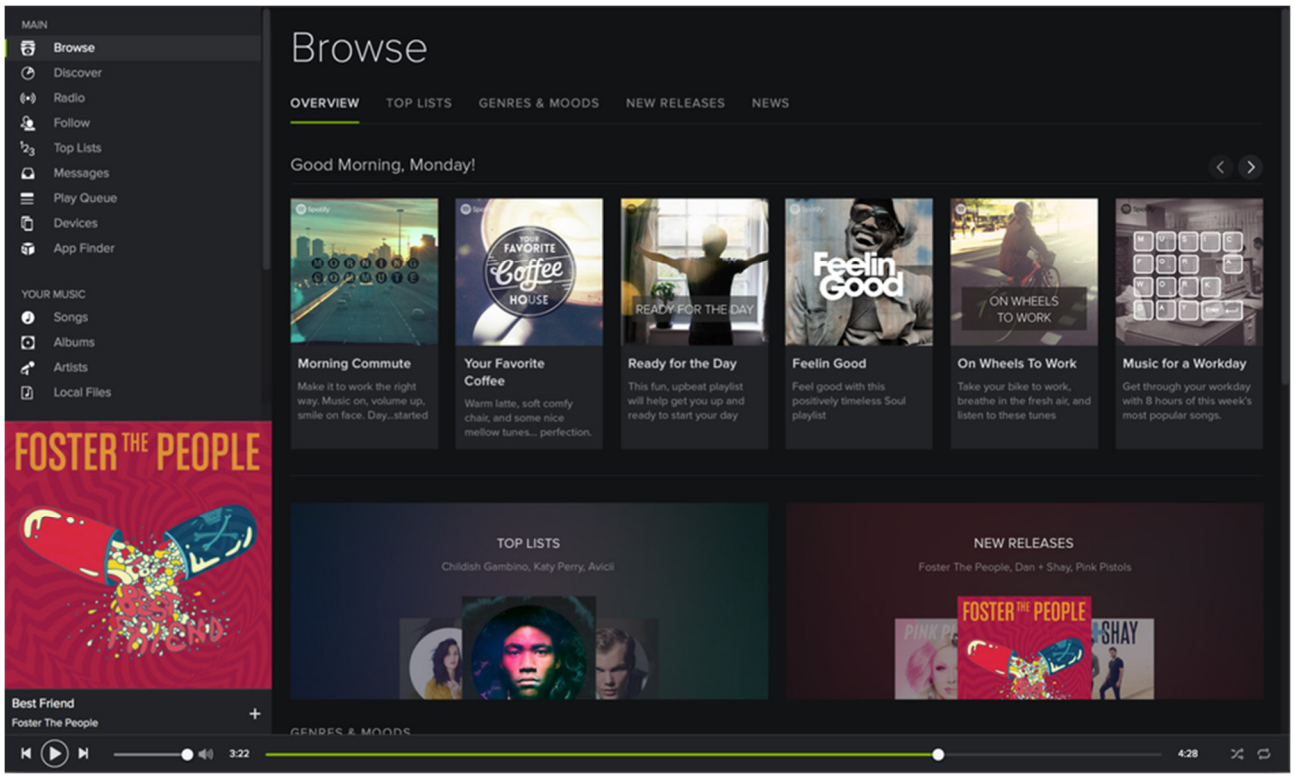



Рис. 2.4. *Spotify* зараз

*Spotify* також є одним з великих сайтів на *PHP-фреймворку Symfony2*.

***Foursquare*** (рис. 2.5 – рис. 2.6)

*MVP* компанії *Foursquare* містило чек-іни і нагороди за них у вигляді бейджів. Вивчивши реакцію користувачів, розробники *MVP* почали розширювати його можливості, додавши рекомендації та путівники по містах. Сьогодні сервіс об'єднує 50 мільйонів людей, які зачекінілісь 8 мільярдів раз.


Рис. 2.5. MVP Foursquare

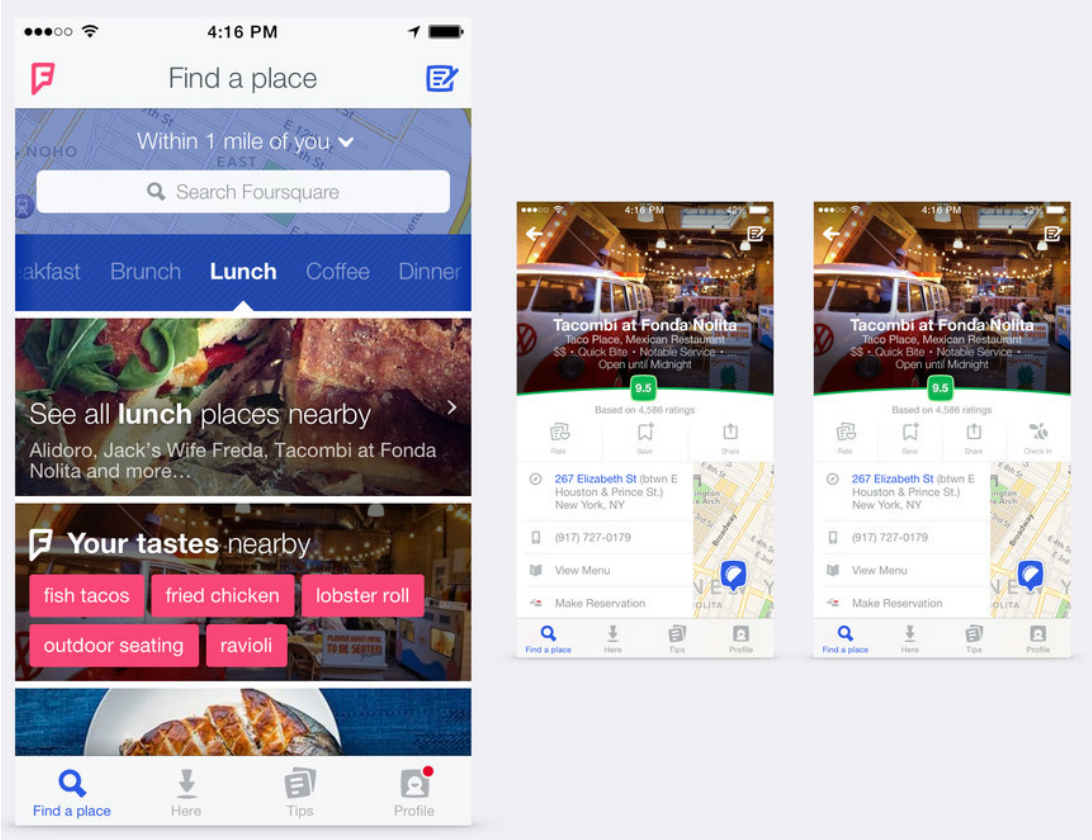



Рис. 2.6. *Foursquare* сьогодні

***Airbnb*** (рис. 2.7 – рис. 2.8)

Популярний серед мандрівників сервіс короткострокової оренди житла *Airbnb* почався з того, що його засновники Брайан Чески і Джо Гебб вирішили здати свою квартиру в Сан-Франциско учасникам конференції по дизайну. Вони сфотографували житло, запустили *MVP* у вигляді нехитрого сайту і вже незабаром приймали у себе перших гостей. Таким чином, стартаперам вдалося на практиці побачити, що ідея здачі власного житла на короткий термін зможе скласти конкуренцію готелям і буде затребувана.

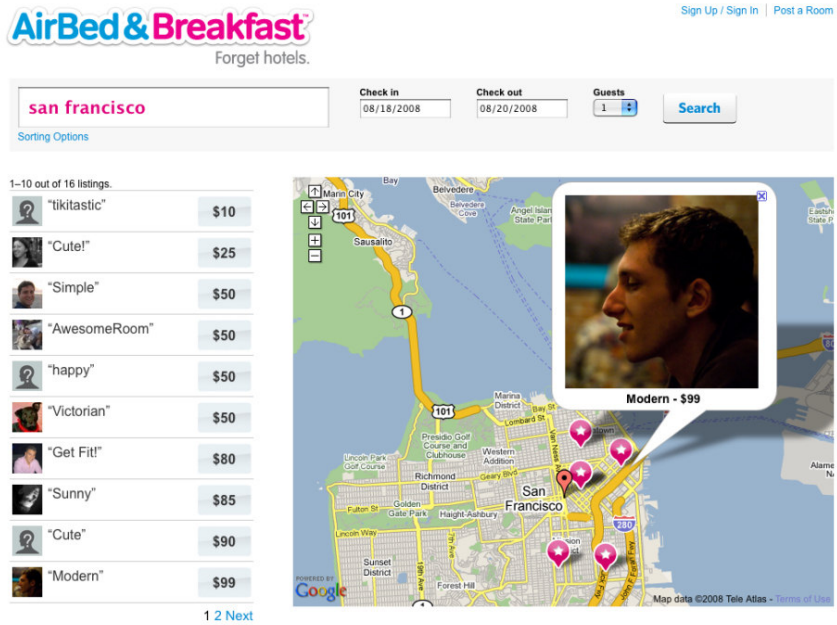



Рис. 2.7. *MVP Airbnb*

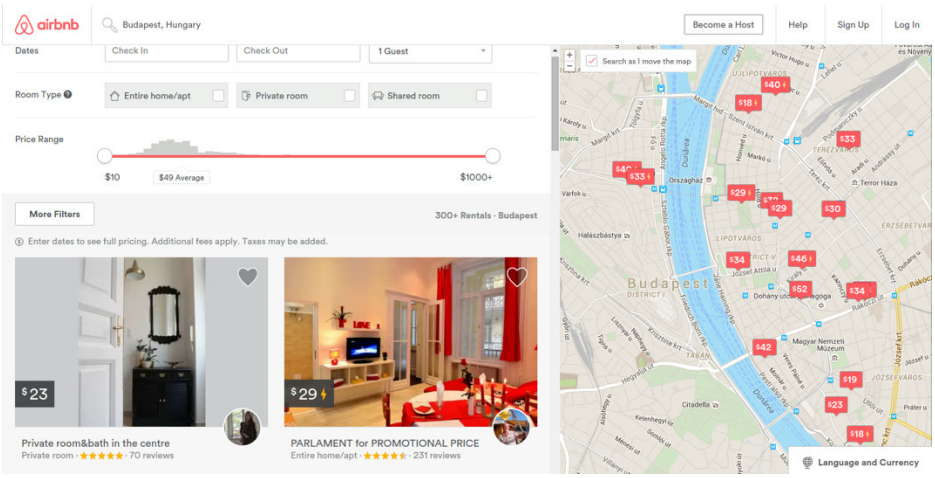



Рис. 2.8. *Airbnb* сьогодні

***Groupon*** (рис. 2.9 – рис. 2.10)

До того, як перевірити ідею сервісу колективних посилань, творці *Groupon* зробили сайт *The Point*, призначений для того, щоб люди, які не можуть виконати щось поодинці, могли знайти однодумців. Однак, ідея виявилася занадто загальною, тому вони запустили кастомізованих блог на платформі *Wordpress*, в який вручну додавали інформацію тільки про можливості колективних знижок. Коли користувачі підписувалися на певну знижку, *PDF*-файл з інформацією про неї вирушав їм на електронну пошту за допомогою *Apple Mail*. Таким чином, засновникам *Groupon* вдалося протестувати свою гіпотезу (людей цікавлять колективні знижки) з мінімальними витратами.

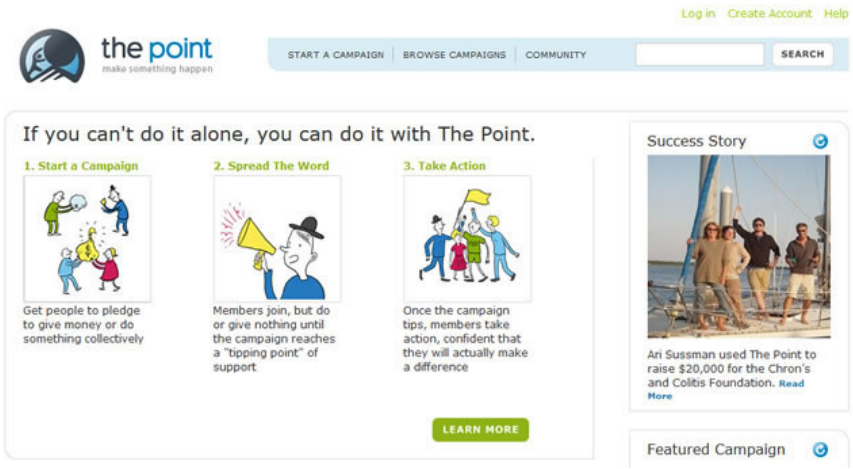



Рис. 2.9. *MVP Groupon*

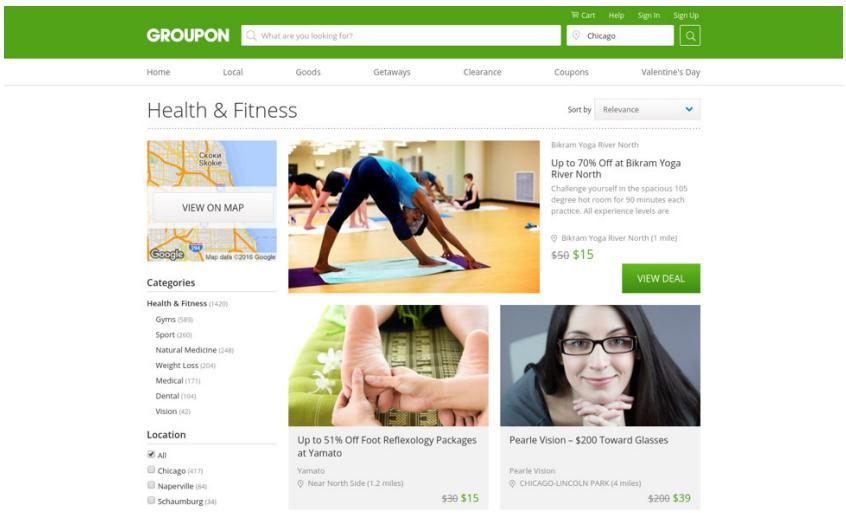



Рис. 2.10. *Groupon* сьогодні

Патерн *MVVM (Model-View – ViewModel)* дозволяє відокремити логіку додатки від візуальної частини (подання). Даний патерн є архітектурним, тобто він задає загальну архітектуру програми, а також відрізняється більш “тісному” зв'язком між моделлю і її поданням допомогою шару “Поданна-Модель”, який синхронізує дані, як при подію на стороні моделі, так і на стороні уявлення. Концепція реалізована в *WPF і Silverlight* [2].

Концепція *HMVC (Hierarchical model-view-controller)* – одне з розширень архітектурного паттерна *MVC*, що дозволяє вирішити деякі

проблеми масштабованості додатків, що мають класичну *MVC* архітектуру. Згідно парадигмі *HMVC*, кожна окрема *MVC* тріада використовується в якості шару в ієрархічній структурі.

При цьому, кожна тріада в цій ієрархії незалежна від інших, і може звернутися до контролера інший тріади. Такий підхід суттєво полегшує і прискорює розробку великих програм, полегшує їх подальшу підтримку і масштабування, сприяє повторному використанню коду.

У *CMS* “*OpenCart*” реалізований класичний шаблон проектування MVC, це дозволить створити гнучке веб-додаток, а також розділити бізнес-логіку і відображення даних програми

**Висновки до розділу**

Як і будь-який бізнес, створення інтернет-магазину потрібно начитати з маркетингових досліджень. Мета цього важливого етапу - отримати інформацію, яка буде основою для успішного початку бізнесу. Потрібно обов'язково дослідити попит на продукцію, що продається, конкурентів, цільову аудиторію і інші важливі аспекти.

Десять основних функцій впливають на продажу:

- Перша така функція – фільтри за різними параметрами.
- Друга функція – порівняння однорідних товарів за параметрами.
- Третя функція – функціонал для демонстрації товарів.
- Четверта функція - розширений пошук за різними параметрами.
- П'ята функція - соціальні елементи.

- Шоста функція – спеціальні пропозиції для стимулювання збуту. магазину, адже у всіх є приховані потреби, потрібно тільки докопатися до них.
- Сьома функція - мобільна версія сайту.
- Восьма функція – популярні товари в інтернет-магазині або окремої категорії.
- Дев'ята функція – розсилка по підписці.
- Десята функція – інтеграція інтернет-магазину зі сторонніми системами.

Шаблон проектування (патерн) в розробці програмного забезпечення – повторимо архітектурна конструкція, що представляє собою вирішення проблеми проектування в рамках деякого часто виникає контекст У порівнянні з самостійним проектуванням, шаблони мають ряд переваг. Головна користь від використання шаблонів полягає в зниженні складності розробки за рахунок готових абстракцій для вирішення цілого класу проблем. Застосування шаблонів концептуально близько до використання вже готових бібліотек.

*MVP* – це мінімально життєздатний продукт, який дозволяє отримати осмислену зворотний зв'язок від користувачів, зрозуміти що їм потрібно і не створювати те, що їм нецікаво і за що вони не готові платити

# 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ВЕБ-САЙТУ ІНТЕРНЕТ-МАГАЗИНУ

## 3.1. Розробка концепції веб-сайту інтернет-магазину

Сайт має бути швидким, однак через те що він повинен мати велику кількість елементів і функцій, це може серйозно сповільнювати роботу. Один з потенційних рішень є використання сучасних феріймворків, які будуть частково оптимізувати роботу сайта за нас.

По-перше концепція *SPA* дає можливість після того як сайт завантажився швидко переводити по його сторінках, оскільки код для їх відображення вже завантажився у перший раз, тож потрібно лише дочекатися завантаження самих товарів з серверу, а це краще, ніж чекати доки весь сайт завантажиться.

По-друге фреймфорки можуть мати гарний функціонал для кушевання, що знову ж скорочує час очікування.

Оскільки сайт буде розроблюватися без сторонніх *CMS*, вартно обрати базу даних і хостінг для зберігання самого сайта.

Серед сучасних *NoSQL* баз даних можна виокремити *mongoDb*, оскільки вона має простий інтерфейс для початку роботи з базам даних, а також зберігання у хмарі. Тобто не потрібно розміщавати у себе базу даних і прописувати повудінку для під'єднання, все це автоматично розбить *mongoDb*, якщо ми користуємось її хмарними технологіями. Також перевага у тому, що тип бази даних *NoSQL*, оскілки так простіше описувати об'єкт товару, він фактично описаний в одному *JSON* об'єкті, а не збирається із 10 таблиць. Через це простіше описувати інтерфейси, оскільки ми одразу розуміємо що це інтрефейс відповідає за певну сутність, і містить всі поля, що їй належать.

Розглянемо основні концептуальні рішення створюваного веб-сайт інтернет магазину.

**Ціль**: Створити повноцінний сайт інтернет магазин мобільних просторів. Він має відповідати сучасних напрямкам дизайну, бути швидким і зручним у користуванні. Приклад сторінки з товарами (рис. 3.1).

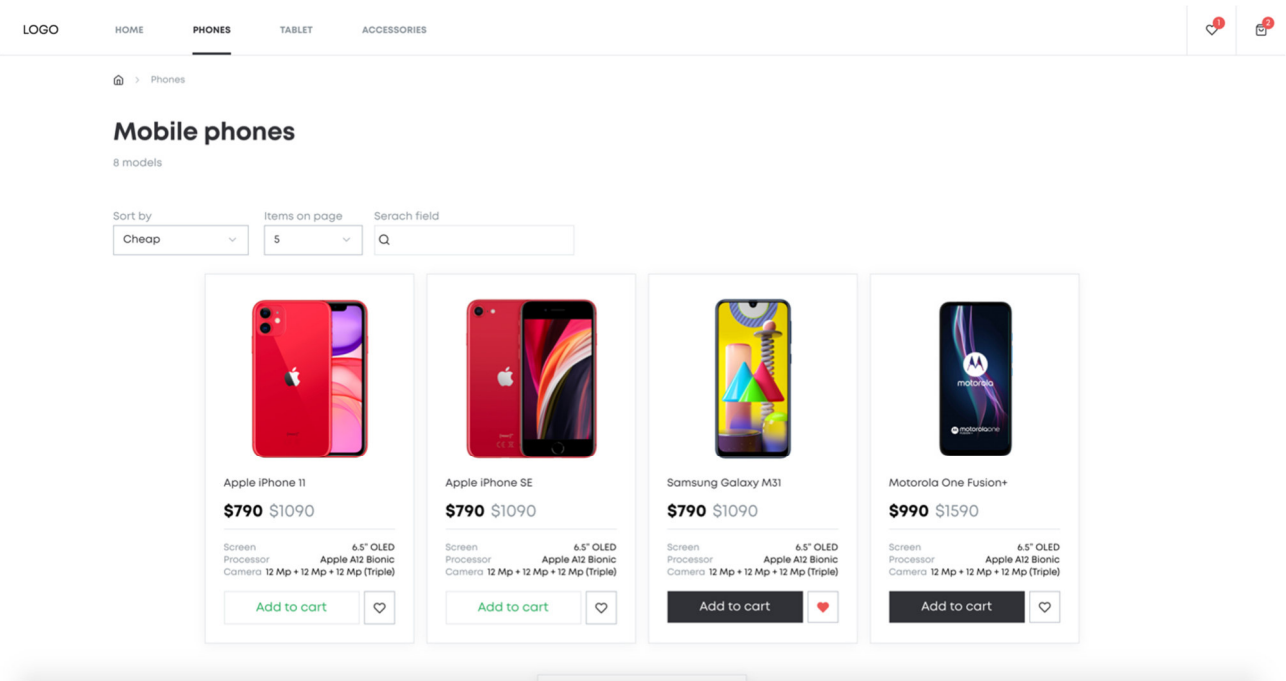



Рис. 3.1. Сторінка з товарами

Аудиторія: Зважаючи на специфіку товару потенційна аудиторія доволі широка, це люди віком від 12 до 50. Оскільки інтернет магазину загалом мають схожі функції, по типу фільтрації, сортування товарів, додавання у кошик або в улюблене, то складнощів у використанні людей, зо знайому з хоча ю один інтернет магазином виникнути не повинно (рис. 3.2).

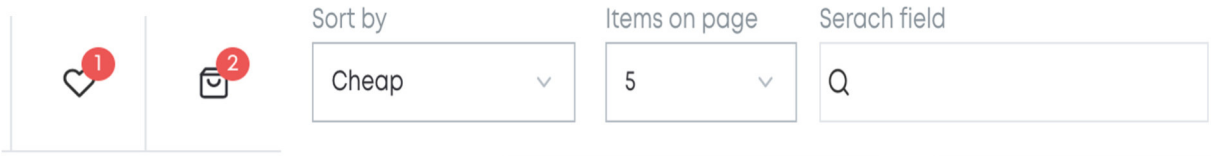



Рис. 3.2. Основні елементи інтернет магазинів

Структура: Сайт має стандарті елементи, хедер, футер, саме тіло, що розмірюється між ними.

Хедер містить навігацію по сайту (рис. 3.3)


Рис. 3.3. Хедер

Він є адаптованим, а точніше для різних розмірів екрану є два окремих хедера, це зроблено для простоти (рис. 3.4)

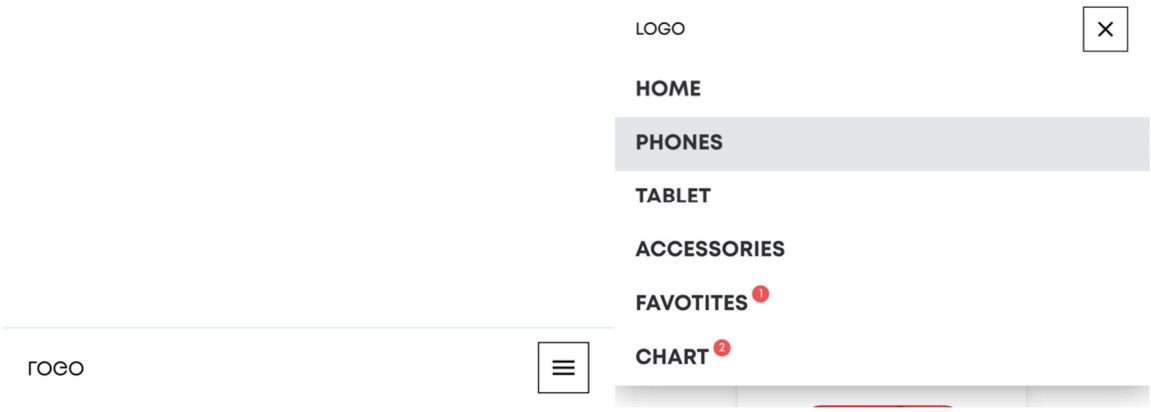



Рис. 3.4. Мобільний хедер

Типографіка: Використовувався шрифт *Mont* з різними нарисами і товщиною (рис. 3.5)

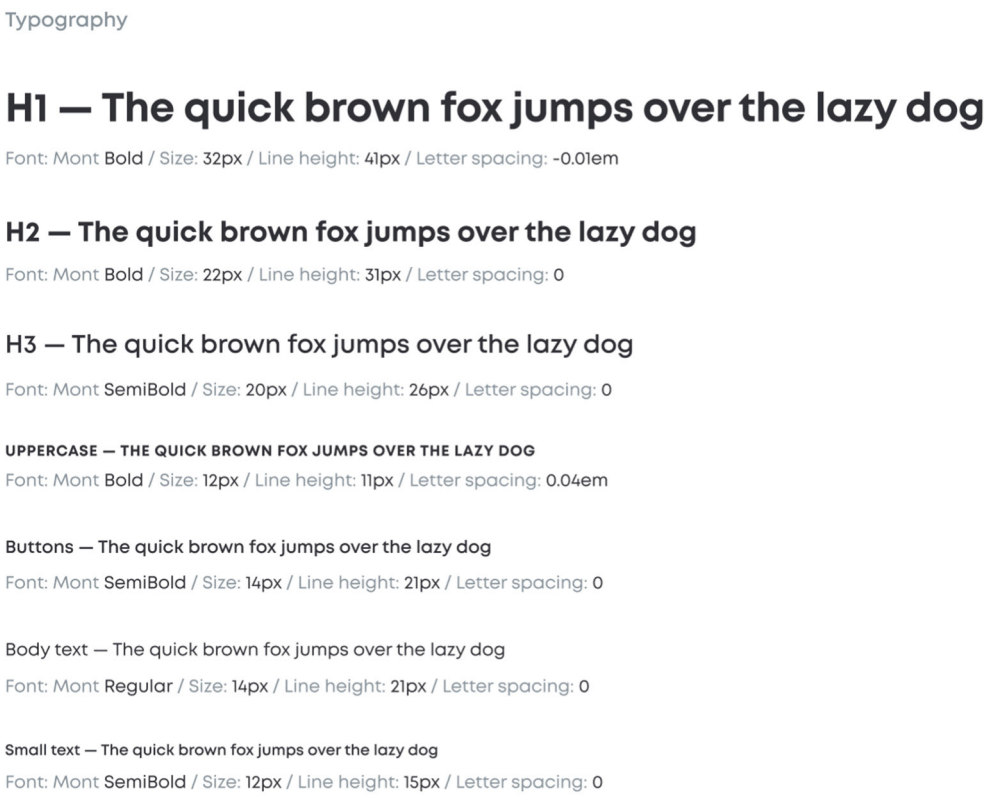



Рис. 3.5. Типографіка в проєкті

Колірне рішення (рис. 3.6): Загалом сайт зроблено в чорно-білому варіанті, з певними градаціями, виключенням являються ключові елементі для привернення уваги, наприклад кількість товарів у кошику, або доданий товар в “улюблене”, чи ні. Це ярко виокремлює важливі елементи.


Рис. 3.6. Колірне рішення проєкта

**Композиційне рішення**

Особливості використовуваних елементів: З пере використовуваних елементів варто виділити кнопки, та інтерактивні (клікабельні) іконки (рис. 3.7). Щоб показати що елемент не активний, змінимо його колір на більш світлий, в порівняння з активним кольором, це виглядає так наче він перестає бути інтерактивним, в на нього не можна клікнути

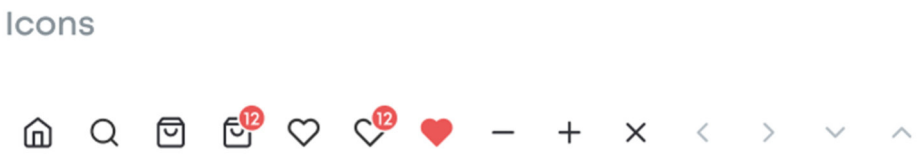



Рис. 3.7. Клікабельні іконки

Кнопки, та елементи, які змінюють свій стан при наведенні курсора (рис. 3.8). Вони можуть бути у таки станах:

1. Неактивні, коли курсор поза ними
2. Активні, коли курсор наведено на елемент
3. Вимкнені, коли неважливо наведений курсор чи ні, стан не змінюється

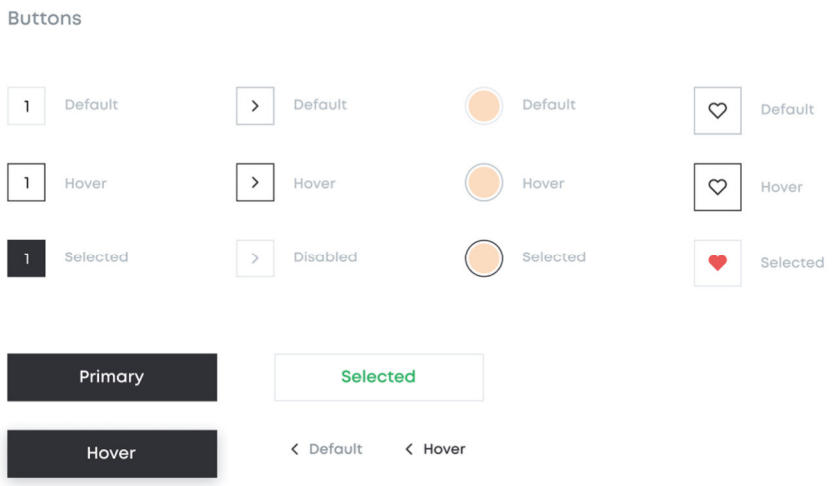



Рис. 38. Елементи, що змінюють свої стилі при наведенні

Оскільки в браузерах є стандартні селекти (рис. 3.9), можна використовувати їх, одна проблема в тому що вони практично не кастомізуються, тому часто роблять власні побідні елементи вже згідно стилю сайту. Вони повинні мати стилі для таких положень:

1. Неактивний, на нього не клацали мишкою
2. Активний, клацнули і випадає меню
3. Вибір з випадаючого меню певний елемент

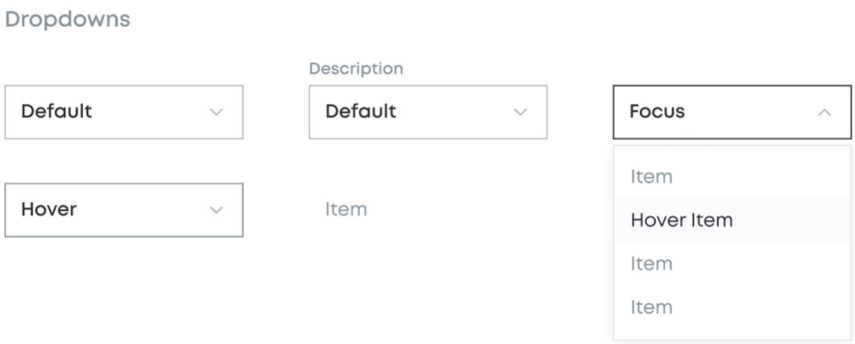



Рис. 3.9. Кастомні селекти

## 3.2. Програмна розробка

Для реалізації фронт частини було обрано сучасний фреймфорк *ReactJS* через те, що він дозволяє робити *SPA* застосунки, що збільшує швидкість роботи сайту і має зручний підхід у реалізації.

Першим кроком було отримання дизайну і ТЗ. Після цього було почато розробку самого проекту. Оглядаючи дизайн сайта на швидкоруч було виділено основні компоненти, на які можна розділити сайт.

Критерії для виокремлення фрагменту, як компонента:

- Відносна простота, має не багато елементів
- Використовується в декількох місцях

Після такого розділення, а такого самого дизайну і ТЗ було отримано знання, яку архітектуру краще закласти в сайт. В якості бази даних було обрано *MongoDb* як *NoSQL* базу даних, через її простоту, можливість безкоштовного використання на початку роботи (розробки), а також крутого допоміжного ПЗ *compass*.

Після обрання бази даних і вибору шляху будівництва архітектури, було створено сам проект, для цього було використано готовий шаблон реакту за допомогою команди *npx create-react-app phone-store*

*phone-store* – це назва проекту. Попередньо для цього було встановлено *node.js* https://nodejs.org/en/. Після цієї команди було отримано шаблон для початку роботи (рис. 3.10).

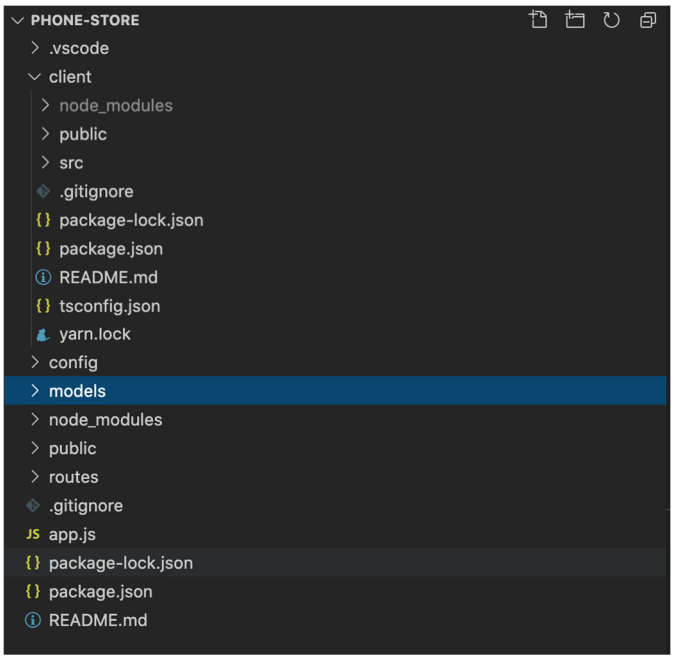



Рис. 3.10. Структура проєкта

Після цього було створено базу даних підключено її до проекту. Щоб створити базу даних було зареєстровано користувача на сайті https://www.mongodb.com/ і реєструю там нову базу даних, після цього, копіюю адресу, за якою до неї можна достукатися і під'єднаю *compass* (рис. 3. 11).

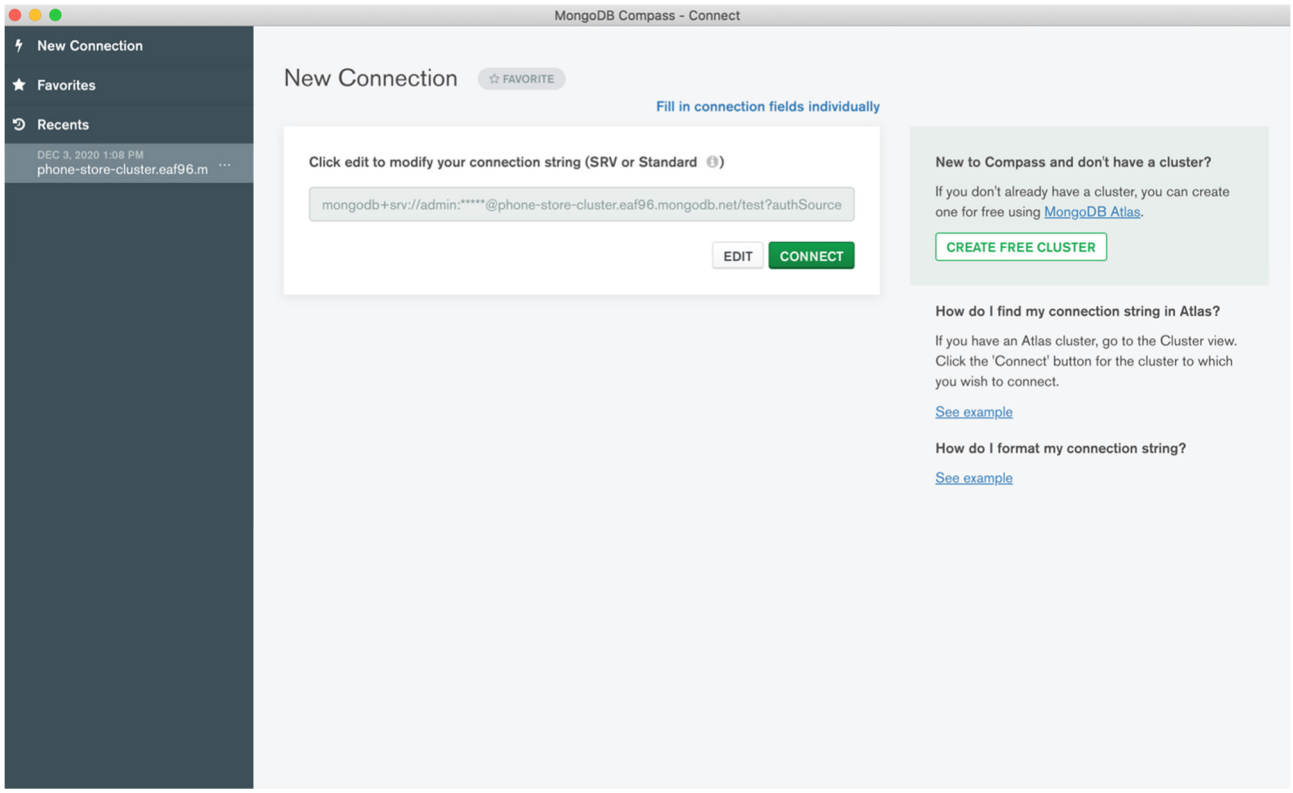



Рис. 3.11. Головний екран *compass*

Далі створюю колекції для телефонів, планшетів, для гарячих пропозицій і для кращих товарів (рис. 3.12). Колекціх – це відокремлені логічно і змістово групи в одній базі даних. Вони можуть бути повязані за допомгою id, або будь-якого іншого поля.

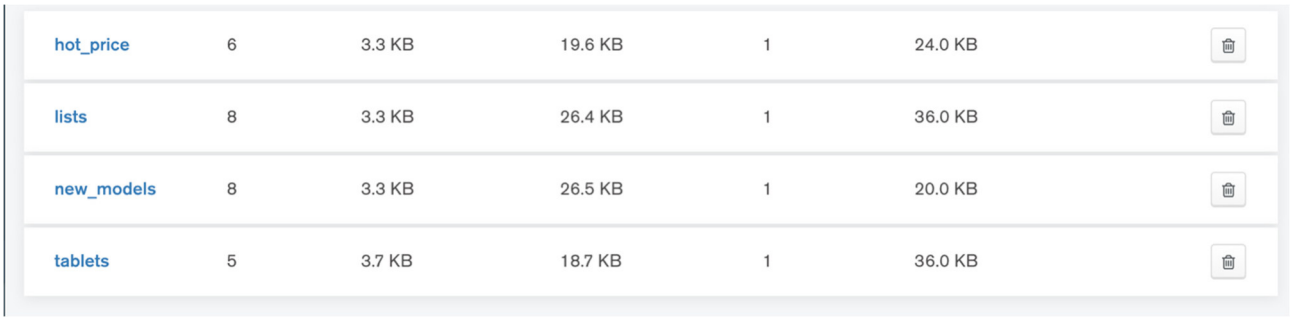



Рис. 3.12. Колекції для груп товарів

Приклад Товарів у колеції list, в які знаходяться списки всіх телефонів що доступні на сайті, їх представлено у вигляді об’єктів, для кращого сприйняття і редагування (рис. 13)

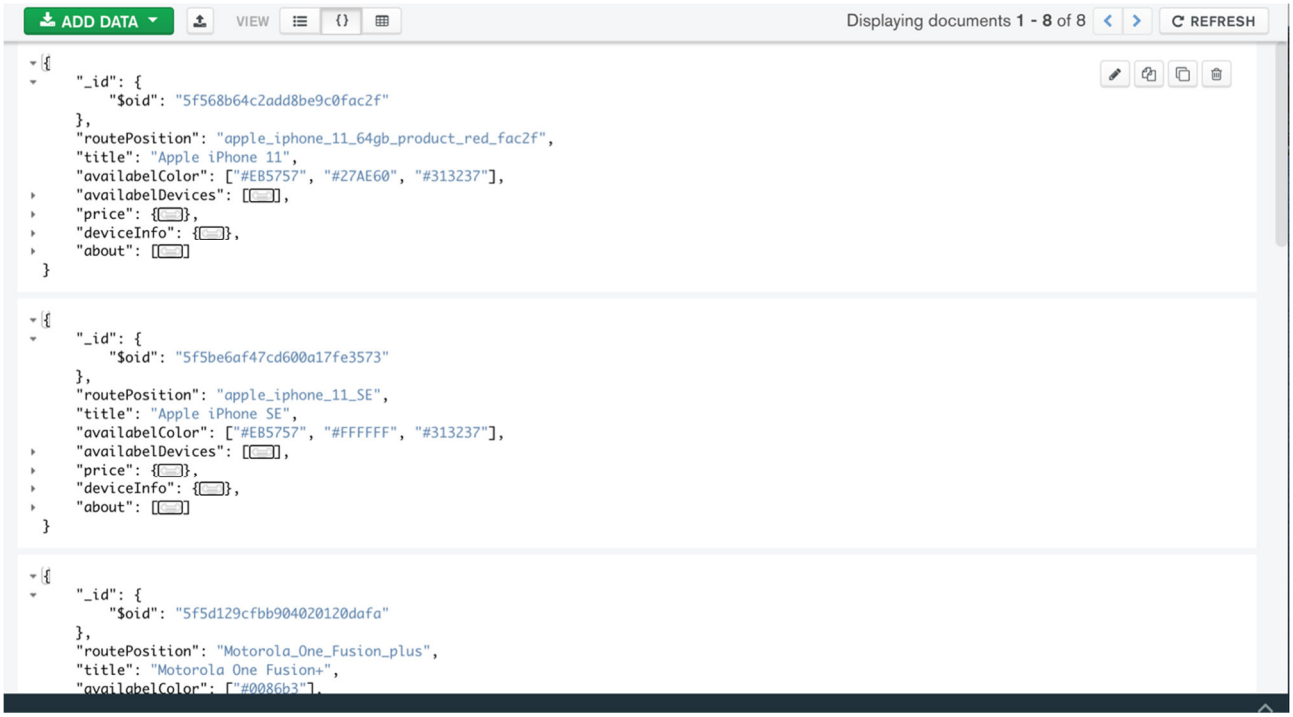
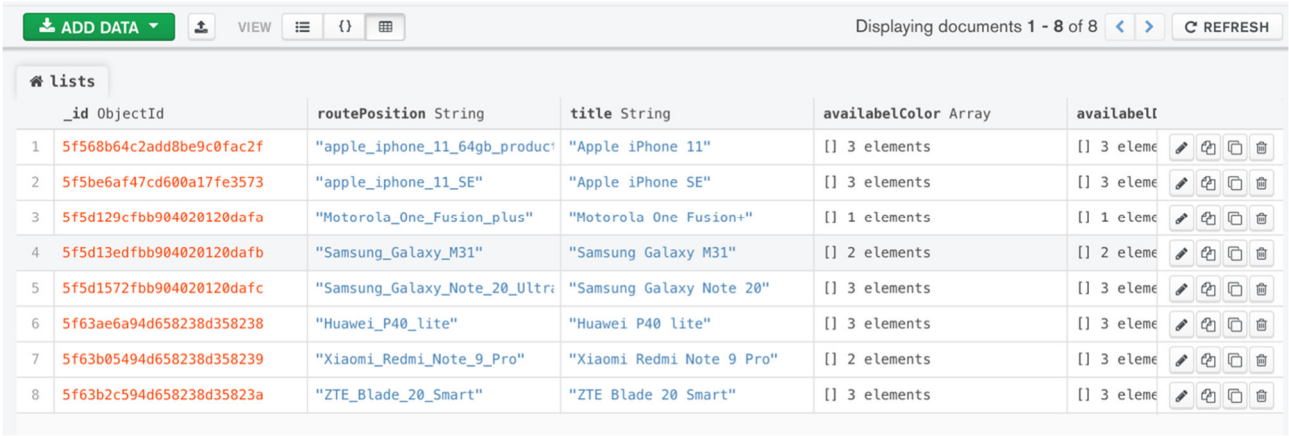



Рис. 3.13. Вигляд як об’єкти

Також можна представити їх у вигляді таблиці або списка (рис. 3.14)


Рис. 3.14. Вигляд списка

Або у вигляді таблиць (рис. 3.15).




| | _id ObjectId | routePosition String | title String | availabelColor Array | availabelI |
|---|---|---|---|---|---|
| 1 | 5f568b64c2add8be9c0fac2f | "apple_iphone_11_64gb_product | "Apple iPhone 11" | [] 3 elements | [] 3 eleme |
| 2 | 5f5be6af47cd600a17fe3573 | "apple_iphone_11_SE" | "Apple iPhone SE" | [] 3 elements | [] 3 eleme |
| 3 | 5f5d129cfbb904020120dafa | "Motorola_One_Fusion_plus" | "Motorola One Fusion+" | [] 1 elements | [] 1 eleme |
| 4 | 5f5d13edfbb904020120dafb | "Samsung_Galaxy_M31" | "Samsung Galaxy M31" | [] 2 elements | [] 2 eleme |
| 5 | 5f5d1572fbb904020120dafc | "Samsung_Galaxy_Note_20_Ultra | "Samsung Galaxy Note 20" | [] 3 elements | [] 3 eleme |
| 6 | 5f63ae6a94d658238d358238 | "Huawei_P40_lite" | "Huawei P40 lite" | [] 3 elements | [] 3 eleme |
| 7 | 5f63b05494d658238d358239 | "Xiaomi_Redmi_Note_9_Pro" | "Xiaomi Redmi Note 9 Pro" | [] 2 elements | [] 3 eleme |
| 8 | 5f63b2c594d658238d35823a | "ZTE_Blade_20_Smart" | "ZTE Blade 20 Smart" | [] 3 elements | [] 3 eleme |

Рис. 3.15 Вигляд табиць

Далі під’єднав базу даних у коді, для цього використав залежність *config*, щоб зручно зберігати якість константні дані. Далі створив папку *config* і додав туди адресу для отримання свої бази даних, номер порту, на якому буде розміщуватися сайт, доки він в розробці, і секретний ключ на випадок, якщо доведеться шифрувати особисті дані (рис. 3.16)

config > {} default.json > ...

```
{
    "port": 5000,
    "jwtSecret": "SenyaSecretKey",
    "mongoUri": "mongodb+srv://admin:admin@phone-store-cluster.eaf96.mongodb.net/phone-list?retryWrites=true&w=majority"
}
```

Рис. 3.16. Базові константи

Для того щоб використовувати *mongoDb* потрібно описати типи моделей, які будуть в нашій базі даних. Приклад опису моделі для мобільного телефону (рис. 3.17).

```
const phonePatternSchema = {
    // _id: String,
    routePosition: String,
    title: String,
    availabelColor: [String],
    availabelDevices: {
        red: {
            availableRAM: [String],
            images: {
                main: String,
                other: [String]
            },
            color: String
        },
        green: {
            availableRAM: [String],
            images: {
                main: String,
                other: [String]
            },
            color: String
        },
        black: {
            availableRAM: [String],
            images: {
                main: String,
                other: [String]
            },
            color: String
        },

    },
    price: {
        current: String,
        old: String,
    },
    deviceInfo: {
        screen: String,
        resolution: String,
        processor: String,
        camera: String,
        zoom: String,
        cell: String,
    },
    about: [{
        title: String,
        description: String
    }]

};
```

Рис. 3.17. Схема телефону для бази даних

Тепер створюємо саму схему і експортуємо її (рис. 3.18).

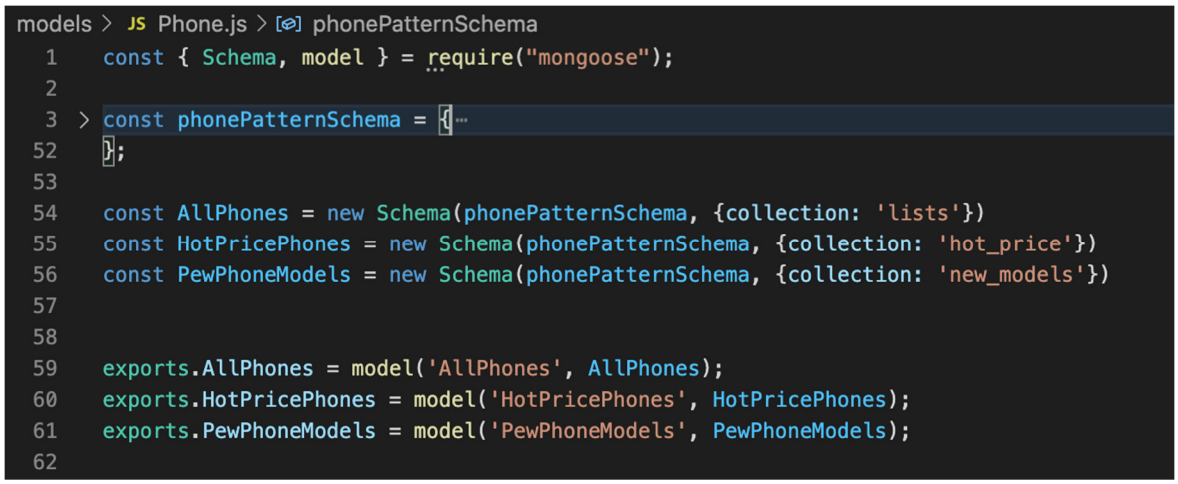



Рис. 3.18. Ініціалізація схеми

Тепер за допомогою бібліотеки *express* налаштовуємо роути на стороні бекенда (рис. 3.19)

```
app.use("/public", express.static("public"));

app.use('/api/auth', require('./routes/auth.routes'))

app.use('/api/phone', require('./routes/phone.routes'))
app.use('/api/tablet', require('./routes/tablets.routes'))
```

Рис. 3.19. Налаштування основних роутів

Тепер база даних готова, а роути для запитів на сервер написані, отже заливається реалізація фронтової частини. Фронтову частину починаємо писати з підєднання *redux* в проект, оскільки він допомагає керувати сховищем даних для сайта. Після цього потрібно додати екшети і редюсери щоб керувати станом редаксу (рис. 3.20 – рис. 3.21)

```
export const phoneLoading = (loading: boolean): AppStateActionTypes => ({
    type: PHONE_LIST_LOADING,
    loading
});
```

Рис. 3.20. Приклад екшену для заватаження моделі телефону

```
export const phonesState = (state = initialState, action: AppStateActionTypes) => {
    switch (action.type) {
        case PHONE_LIST_LOADING:
            return { ...state, loading: action.loading }

        case PHONE_LIST_SUCCESS:
            return { ...state, phoneList: action.phoneList }

        case PHONE_LIST_ERROR:
            return { ...state, error: action.error }

        case PHONE_LIST_STATE:
            return { ...state, phoneListState: action.phoneListState }

        case PHONE_ITEM_SUCCESS:
            return { ...state, currentModel: action.currentModel }

        default:
            return state
    }
}
```

Рис. 3.21. Приклад редюсера для зберігання стейта телефонів

Тепер можна перейти до основних перевикористовуваних компонентів, а саме хедеру, футеру і карток товарів.

**Розробка хедеру**

Одразу робимо два варіанти, один для версії з великим екраном, більше 700 px, інший менше, для мобільних пристроїв і планшетів (рис. 3.21).


Рис. 3.21. Структура мобільного і десктопного хедеру

Після цього була написана функція яка відстежує ширину екрану в даний момент і передає його в редакс, тож весь додаток миттєво дізнається, якщо ширина змінилася і це дає змогу нам визначати коли який хедер показати (рис. 3.22 – рис. 3.24).

```
// check device screen width
const checkDeviceSize = (event?: any) => {
  const screenWidth = event ? event.currentTarget.innerWidth : window.innerWidth;

  store.dispatch(setDeviceScreen({
    value: screenWidth,
    name: screenWidth > 0 && screenWidth <= 500
      ? 'phone'
      : screenWidth > 500 && screenWidth <= 900
      ? 'tablet'
      : 'desktop'
  }))
}
```

Рис. 3.22. Відстеження розміру екрана

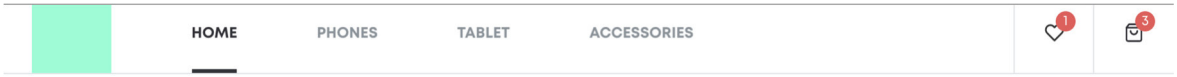



Рис. 3.23. Десктопний хедер

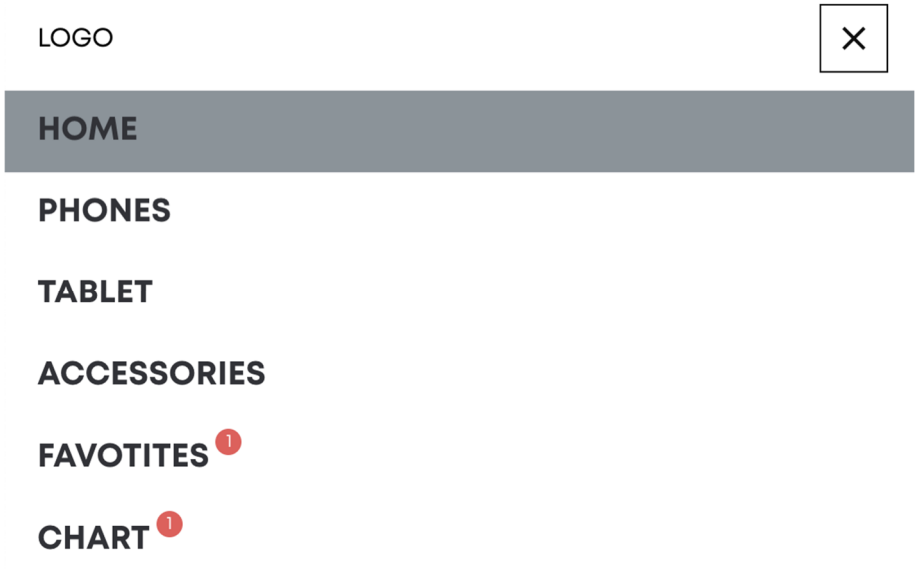



Рис. 3.24. Мобільний хедер

Важливим елементів є пошук потрібних товарів на сайті, тому потрібно вмонтовати пошукове меню у хедер сайта. Однак враховуючи адаптивність сайта і довжину пошукового меню, потрібно зробити декілька варіантів розміщення поля пошуку і хедері, для маленьких розмірів екрану і для великих (рис. 3.25 – рис. 3.30).


Рис. 3.25. Елемент пошуку


Рис. 3.26. Хедер для великих екранів


Рис. 3.27. Хедер для середніх екранів (планшетів)

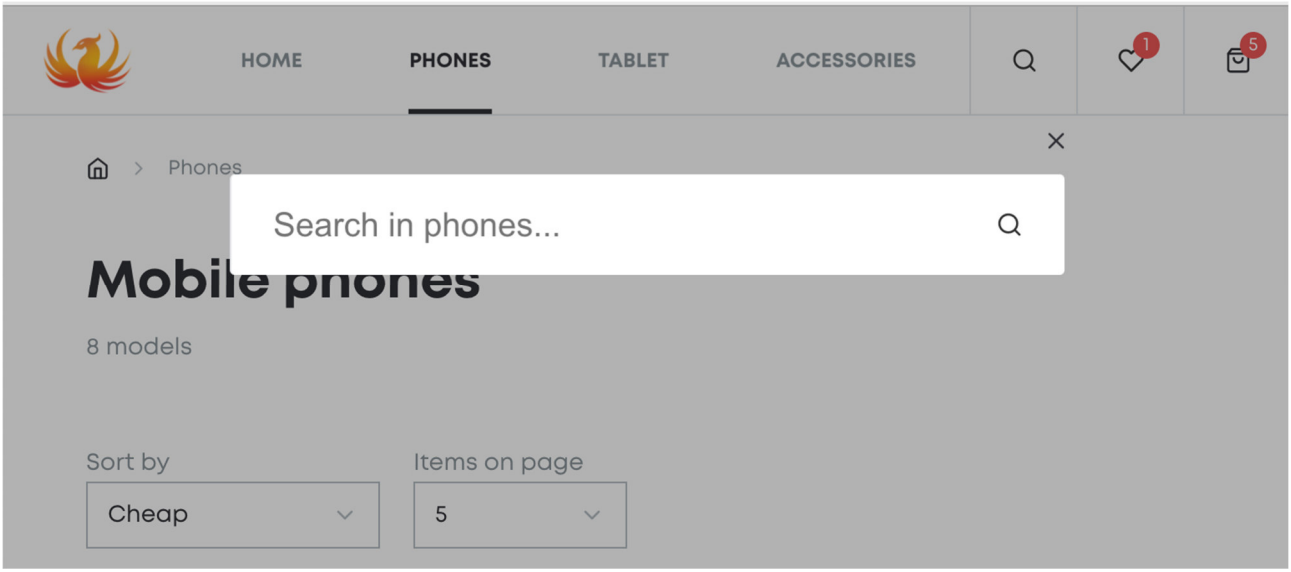



Рис. 3.28. Відкриття додатково поля для пошуку на середніх екранів

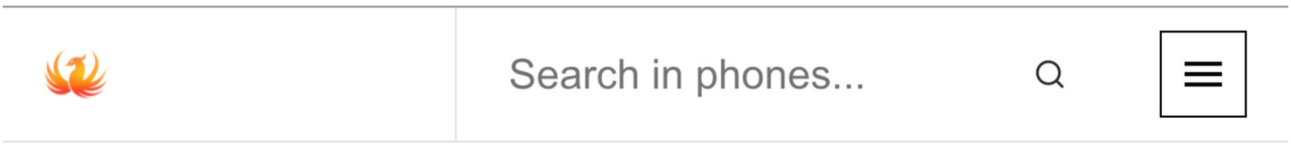



Рис. 3.29. Для маленьких пристроїв (телефонів)

Тепер час для футера, в цьому випадку не потрібно створювати два окремих варіанти для мобільного розширення і для великих моніторів, адже тут не так багато елементів і вони всі еластичні, тож можна просто попрацювати над адаптивністю (рис. 3.30 – рис. 3.31).

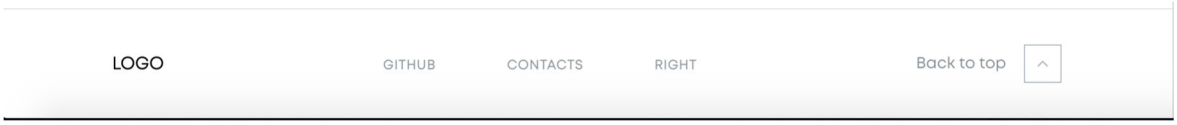



Рис. 3.30. Футер на повну ширину

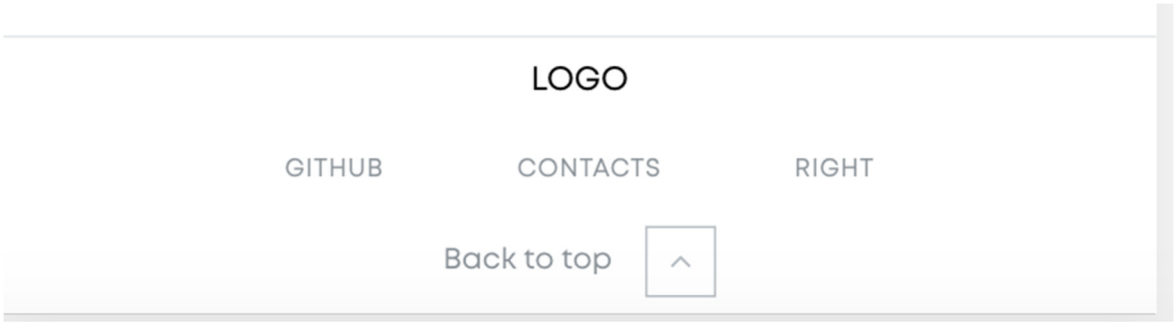



Рис. 3.31. Адаптований футер під малу ширину

На головному екрані має розміщуватися сладйер, однак не такий, як для товарів, а просто сладйер для зображень, тож потрібно трохи змінити висоту кнопок вперед / назад (рис. 3.32).

Рис. 3.32. Слайдер для зображень

В дизайні на головній сторінці присутні слайдери, тож варто створити шаблон для цього елемента, однак перед цим потрібно створити компонент картки товару (рис. 3.33).

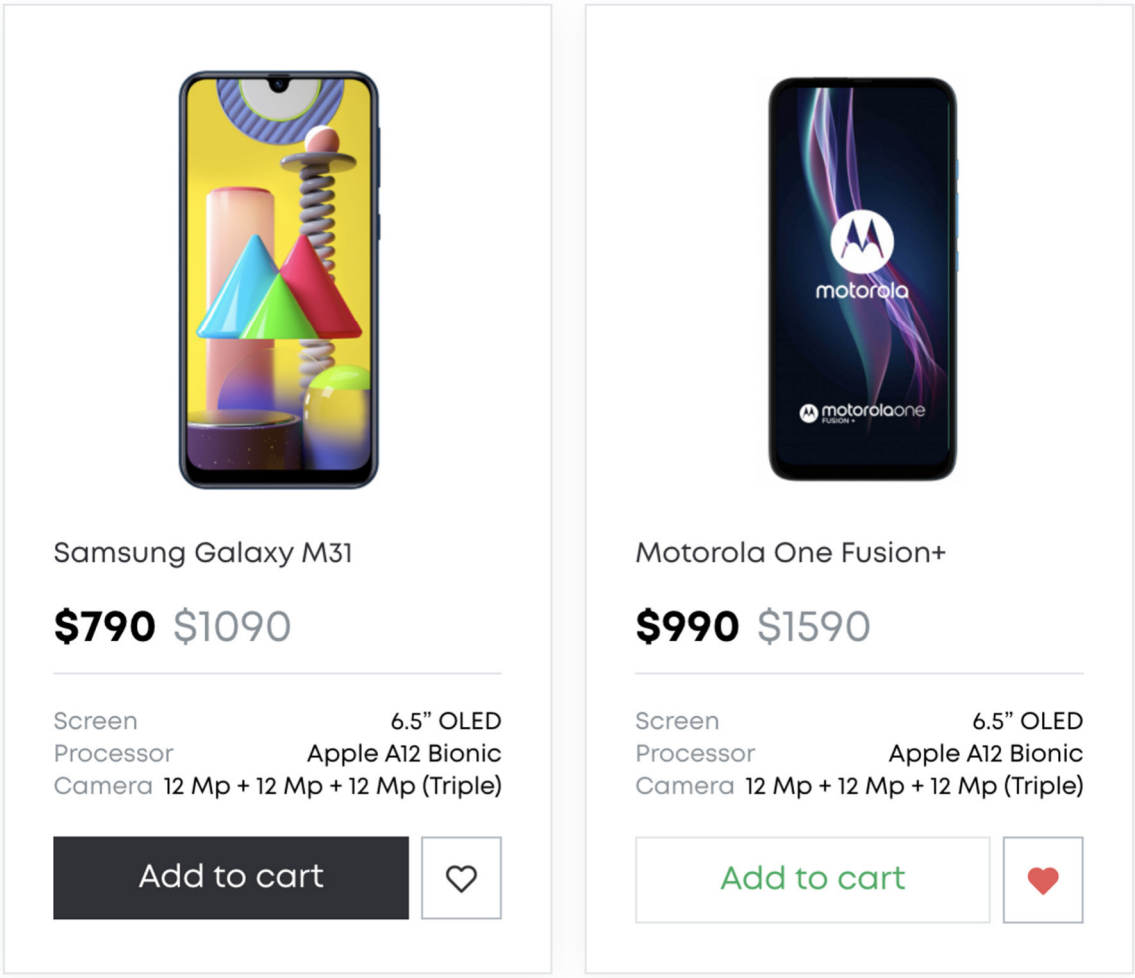



Рис. 3.33. Кратки товару

Для того щоб з цих карток зробити карусель (слайдер) було використано бібліотеку *slick-slider*. По суті просто передаючи в *slick-slider* дані для рендера карток карусель готова, однак для зручності використання потрібно додати клавіші для керування і отримаємо повноцінний слайдер (рис. 3.34).

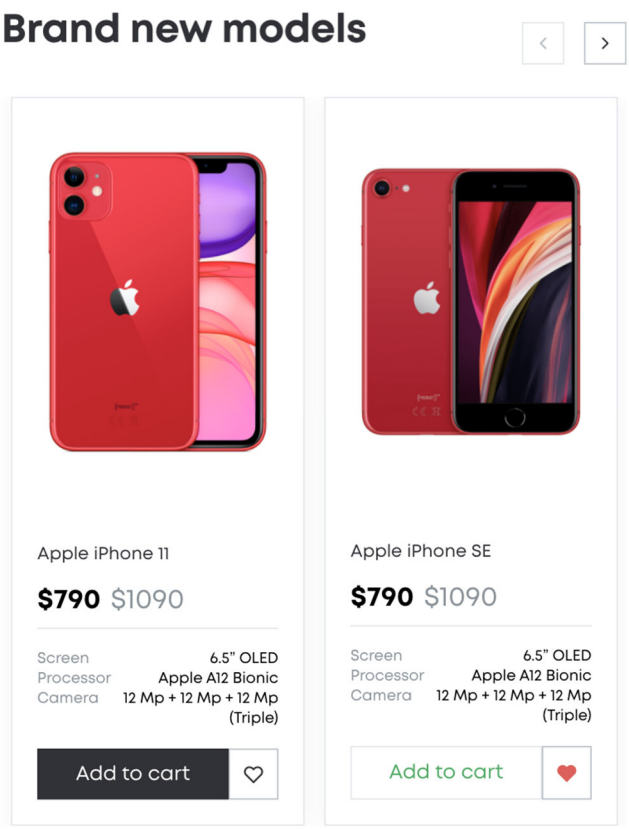



Рис. 3.34. Слайдер

Окрім слайдера за дизайном на головній сторінці сайта має бути список з плиток категорій товарів (рис. 3.35)

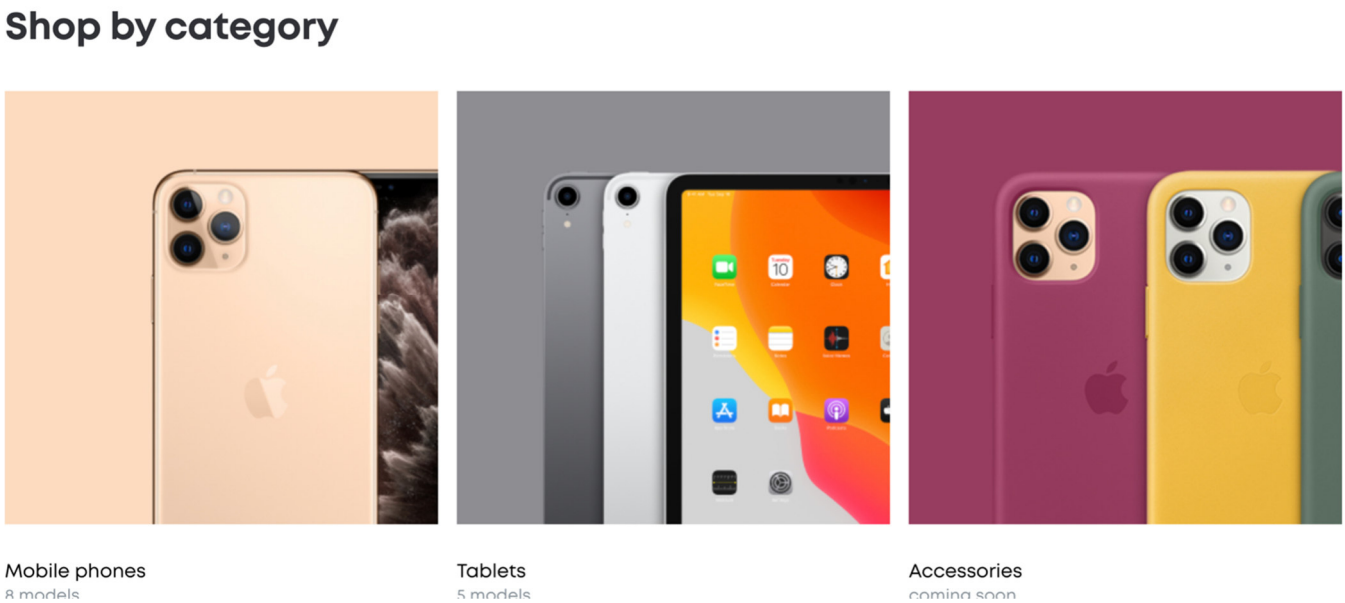



Рис. 3.35. Плитки на головній сторінці

Сторінка для улюблених товарі, що кристувач вподобав під час користування сайтом буде містити хедер, футер, а також список у вигляді плиток самих товарів (рис. 3.36)

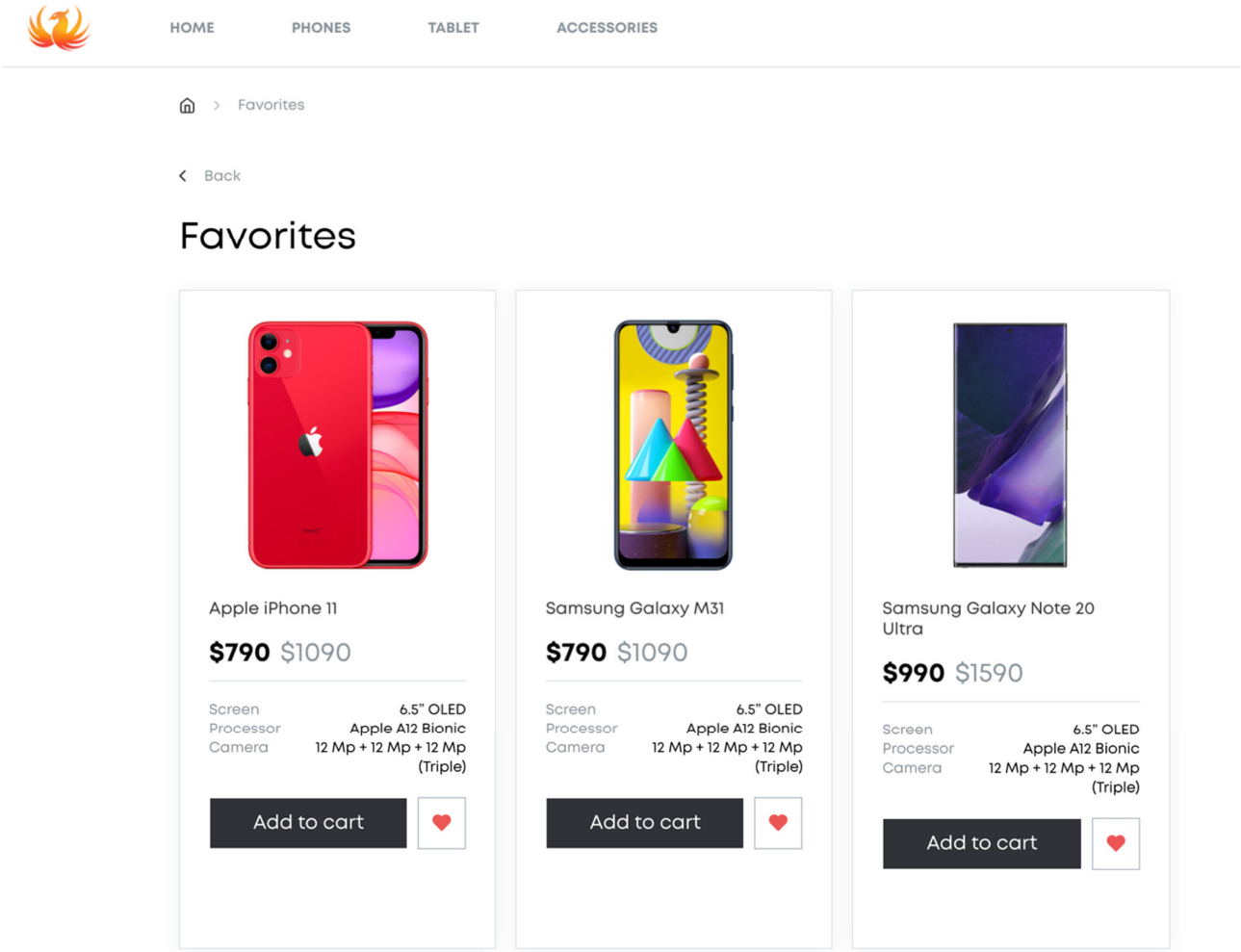



Рис. 3. 36. Сторінка вподобаних товарів

Окрім цього для кращої навігація створимо міні карту для того, аби юзер розумів де саме він знаходиться. Початковою точкою буде головна сторінка, а далі той шлях, який було пройдено. На рисунку 3.37 показано приклад коли юзер перейшов на сторінку з усіма телефонами і обрам там телефон Sansub Galaxy Note 20 а перейшов на сторінку конвертного товара. Як видо з рисунка остання точка шляху неактивна, оскільки користувач і так знаходиться на ній, а решта активні, і клікнувши на них можна перейти на ту сторінку.


Рис. 3.37. Маленька навігація

Тепер створимо сторінку для планшетів, вона буде такою самою як і сторінка для телефонів, і схожою на сторінку для аксесуарів (рис. 3.38).

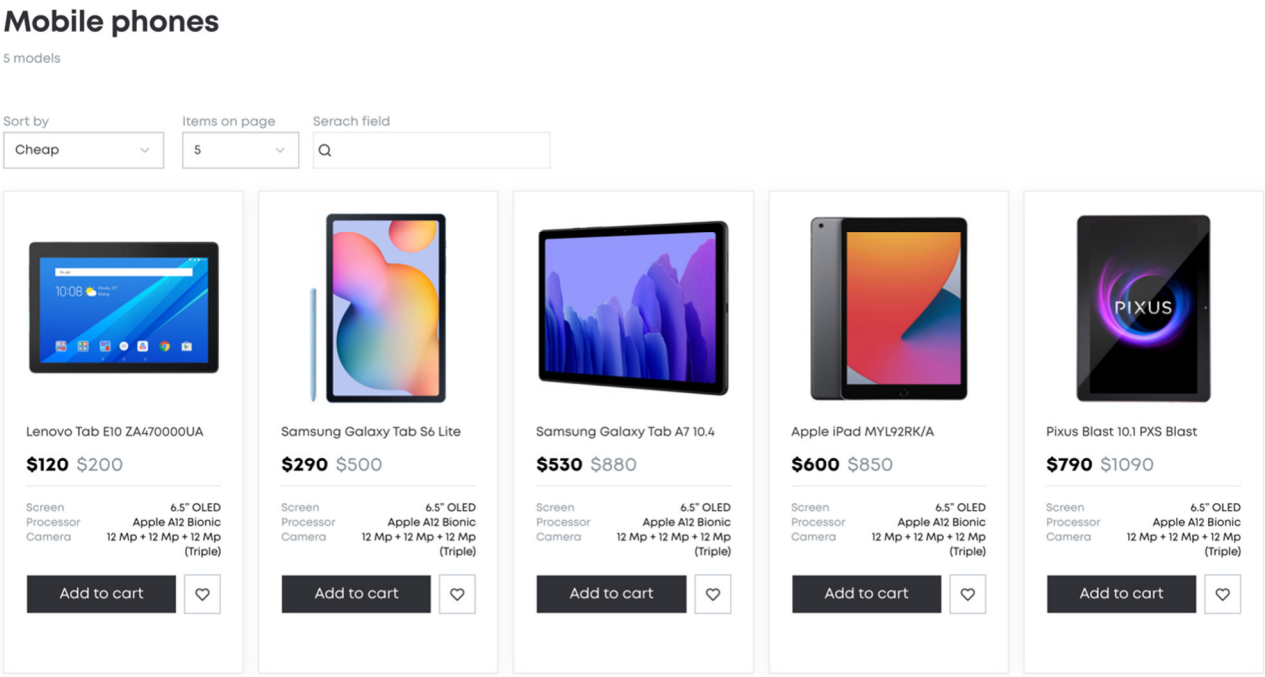



Рис. 3.38. Сторінка планшетів

Коли користувач тільки зайшов на сторінку, нам потрібно завантажити список товарів, однак про це потрібно якось повідомити юзера, тому додамо *preloader*, що буда повернитися, доки сторінка не завантажиться (рис. 3.39). Це стандартний елемент кожного сайта, десь він реалізований в більшій мірі, десь в меншій, одна його наявність на сайті стовідсоткова

Рис. 3.39. *Preloader*

Тепер створимо сторінку для конкретного товару (рис. 3.40), вона скрадатиметься з двох основних елементів, перший – це фото товару і можливі варіації кольору та інших опціональних характеристик.

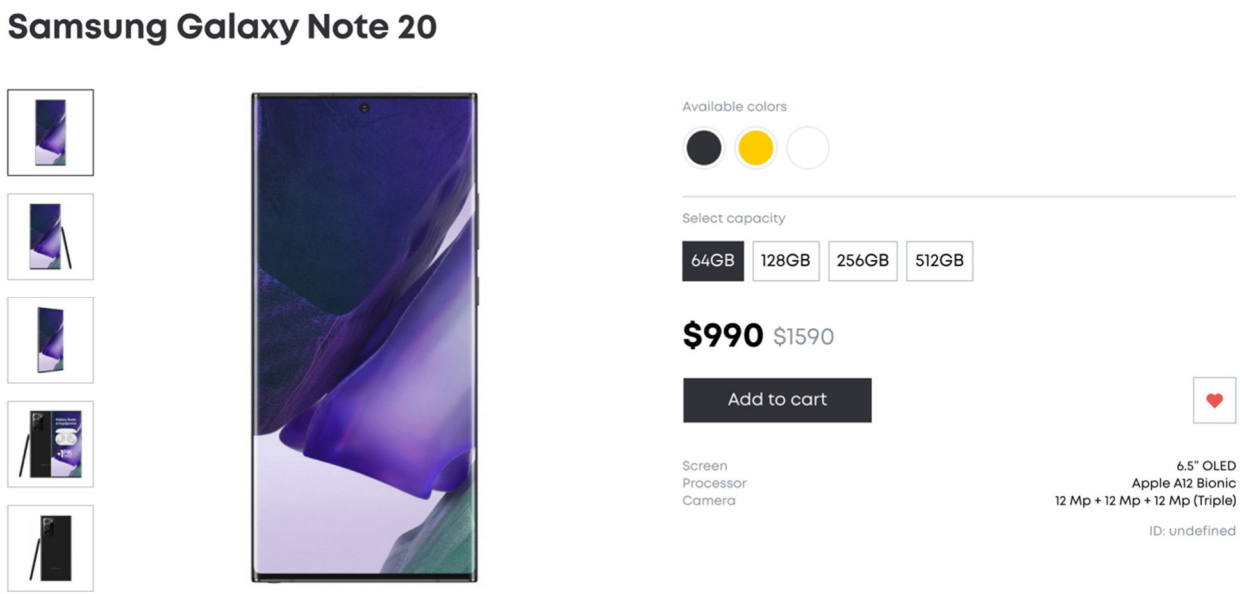



Рис. 3.40. Перша частина картки товара

Другий – це опис телефона або будь-кого іншого пристрою, а також перелік усіх інших характеристик, що вписані для цієї моделі товару (рис. 3.41).

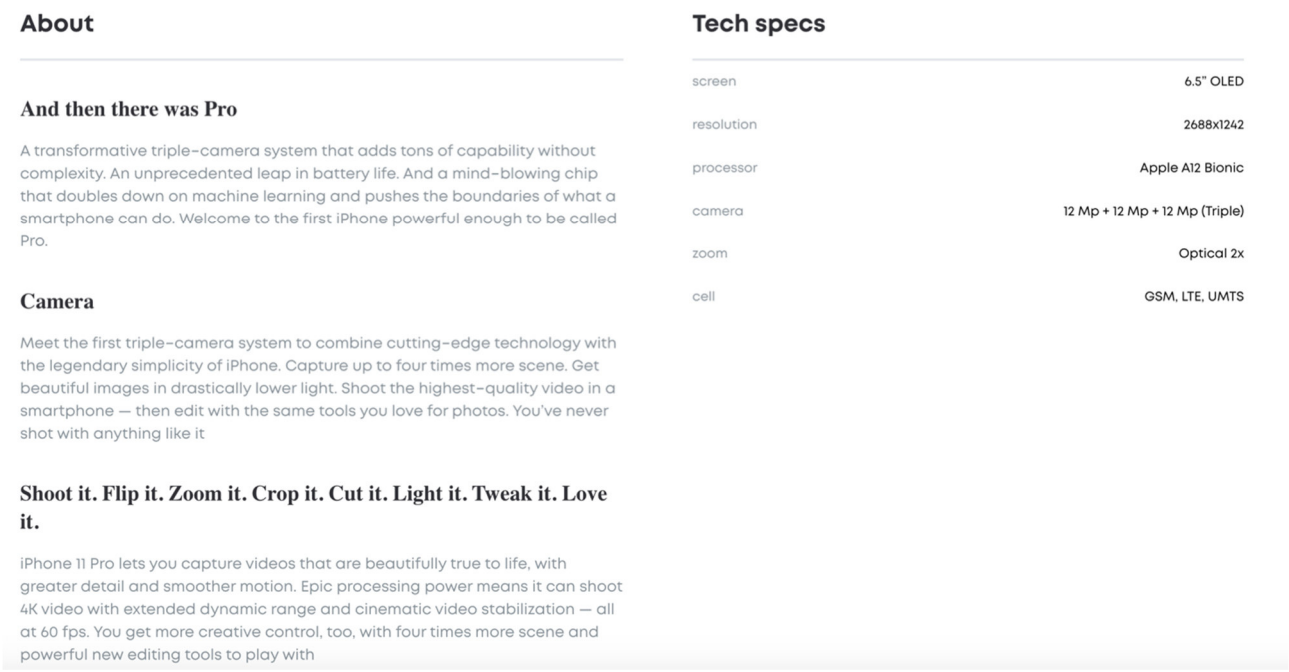



Рис. 3.41. Друга частина картки товара

В кінці була реалізована функція кошика (рис. 3.42), тобто людина просто клацає на іконку "додати в кошик в будь-якому місці де є ця іконка і товар автоматично додається до кошика. Додавання до кошика реалізовано у два етапи, перший - додання у динамічну пам’ять, тобто під час перезавантаження вона зникне, але це швидкий обмін даними. Другий етап – це додавання цих даних в *LocalStorage*, це сховище не залежить від перезавантаження сторінки чи вимикання комп'ютера. Тож якщо людина додала якісь товари вони будуть там доти, доки вона їх сама не видалить, або не обновить кеш і куки.

Перший елемент цієї сторінки – це сама картка доданого товару, на ній має обов’язково бути фото товара, його назва, блок регуляції кількості і сума.


Рис. 3.42. Товар, що було додано до кошику

Після цього йде блок де можна здійснити саму покупку, тут вказана кінцева сума, тобто якщо товарів декілька то тут буде результат суми всіх цін, а також сумарна кількість одиниць товару (рис. 3.43).

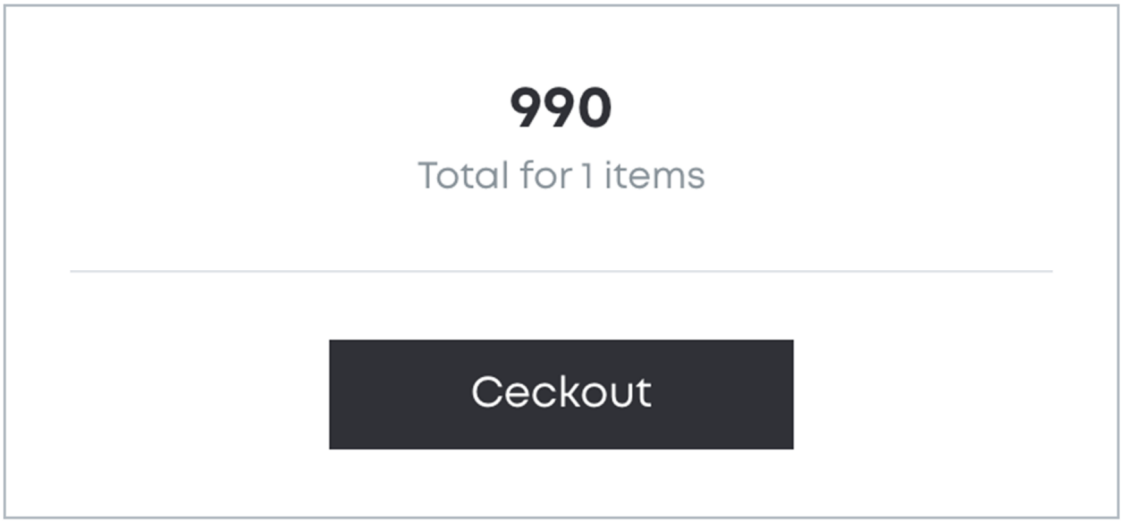



Рис. 3.43. Кнопка для покупки

На рисунку 3.44 представлено як вся конструкція виглядатиме на середньому і великому екрані.

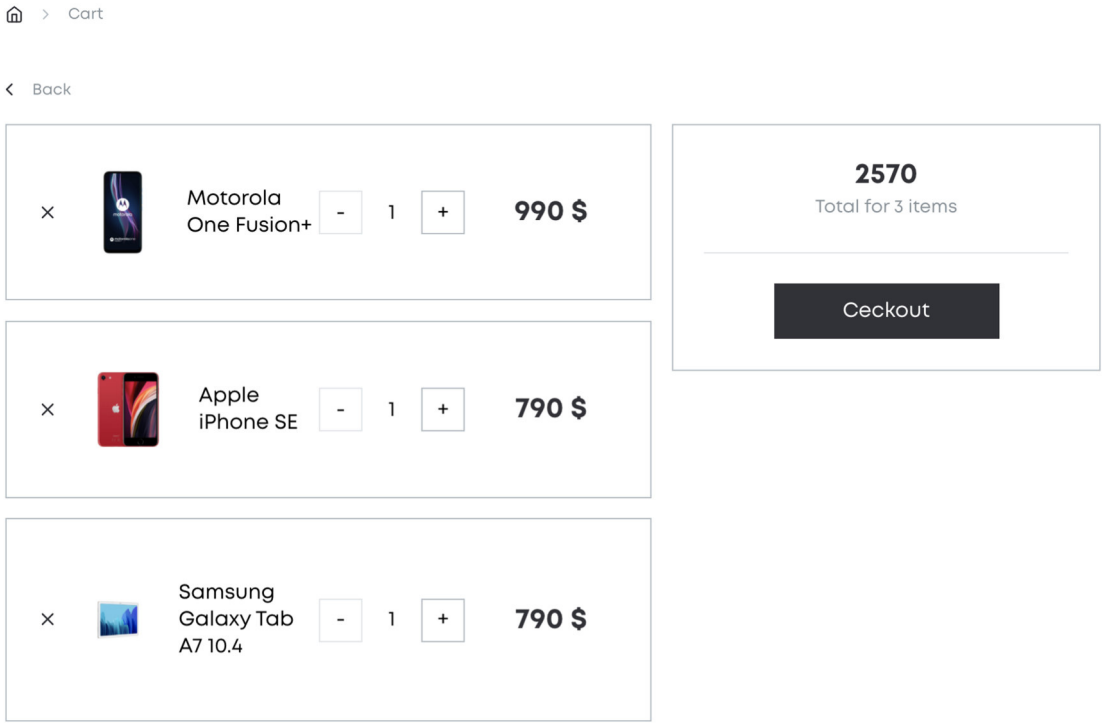


Рис. 3.44. Сторінка покупки товару для великих екарнів

Оскільки таку конструкцію вкрай важко масштабувати під маленькі пристрої, то просто переверстаємо під більш відповідну модель (рис. 3.45).

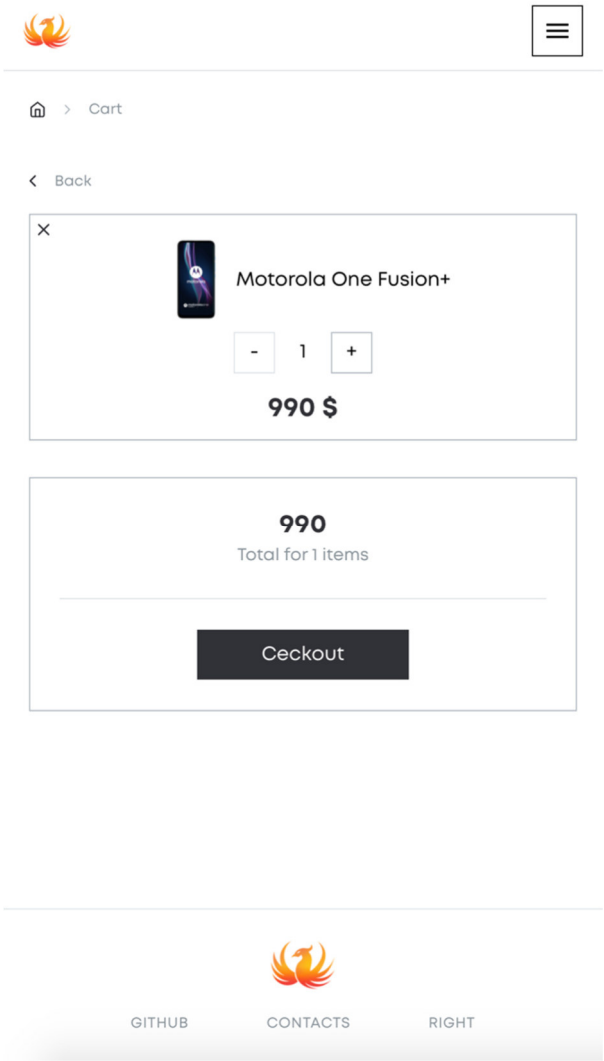



Рис. 3.45. Сторінка покупки товара для маленьких екранів

Вся розробка вимагає багато часу і багато коду. Тому очевидно що під час розробки будуть виникати проблеми і непривальні рішення. Які доведеться перероблювати. Тому для покращення цього моменту використовувалась система контроль версій *git*, а також сервіс *git-hub*.

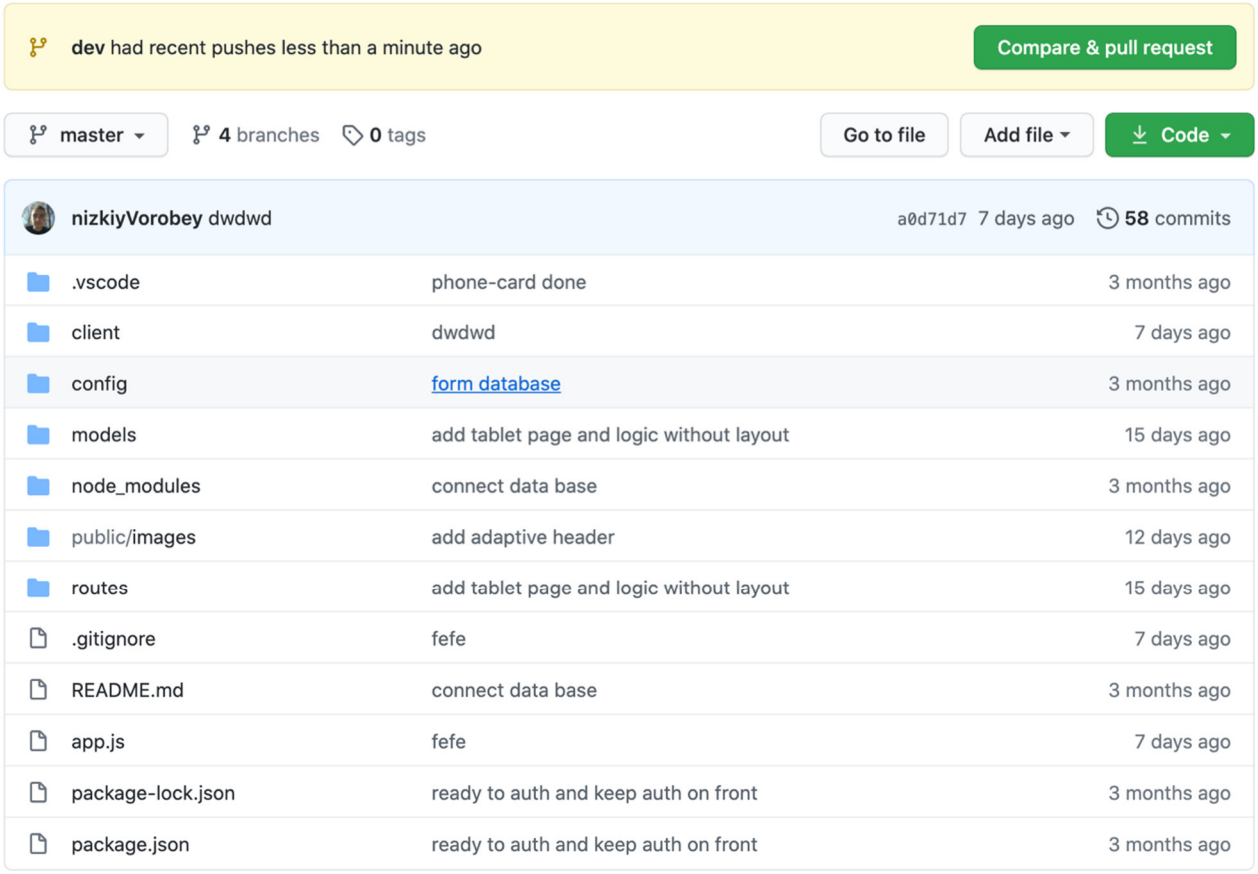



Рис. 3.46. Сторінка проєкта на гітхаб

Однією з особливостей цієї система є можливість вести розробку на різних гілках, а потім об’єднувати якщо потрібно, на рисунку показано деякі з гілок що були використані під час розробки (рис. 3.47).

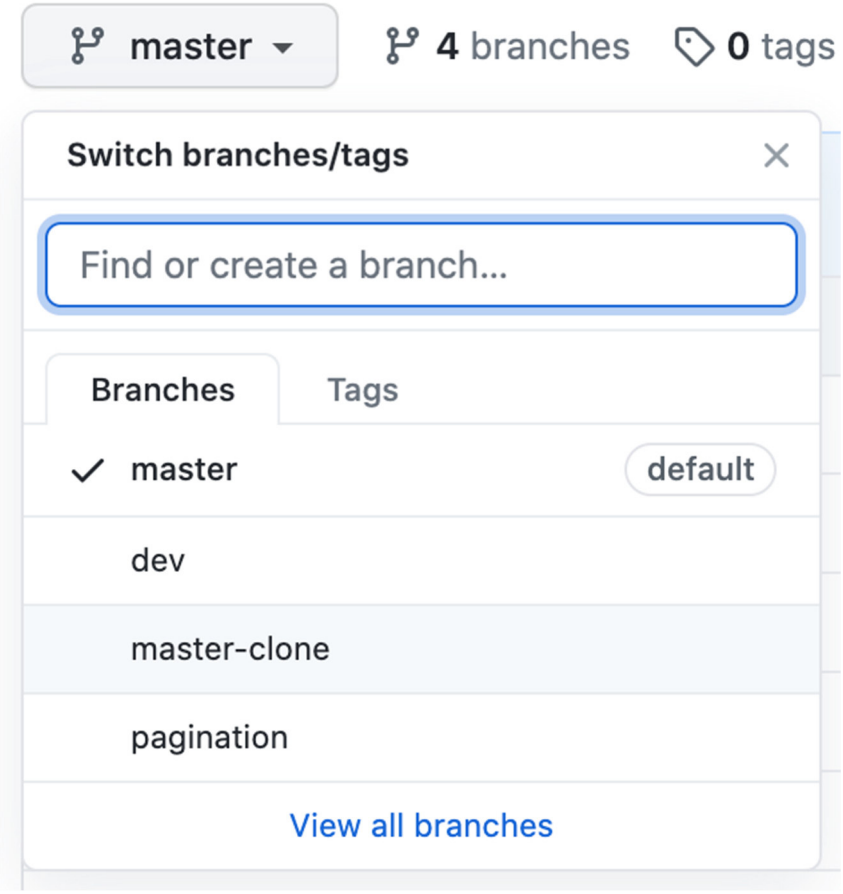



Рис. 3.47. Робочі гілки

Також цікавим доповненням є те, зо гітхаб самостійно сканує проєкт і дивиться які мовами програмування і технологій використовувалися, на рисунку … показано мій стек технологій

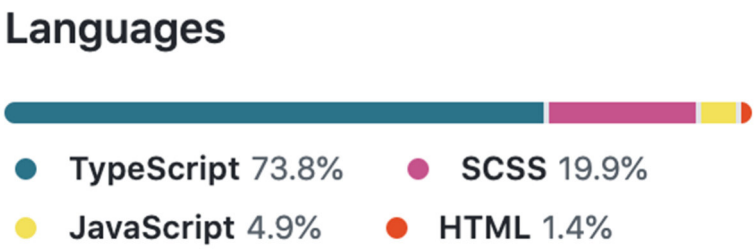



Рис. 3.48. Мови що використані в проєкті

Приклад роботи із терміналом показано на рисунку 3.49 Тут командою *git status* показано чи змінювалися якісь файли, з урахуванням що вони не були додані до

системи *git*. Другою командою *git branch* показано на якій гілці зараз заходиться проєкті.

```
senya@AnyMacs-MacBook-Pro phone-store % git status
На ветке dev
Ваша ветка обновлена в соответствии с «origin/dev».

нечего коммитить, нет изменений в рабочем каталоге
senya@AnyMacs-MacBook-Pro phone-store %
senya@AnyMacs-MacBook-Pro phone-store %
senya@AnyMacs-MacBook-Pro phone-store %
senya@AnyMacs-MacBook-Pro phone-store % git branch
* dev
  master
senya@AnyMacs-MacBook-Pro phone-store %
```

Рис. 3.49. Робота з терміналом

Ще один приклад роботи з терміном, командою *gut log -3* показуємо три остання коміта, їх назву, дату створення,  також авторах що їх зробив (рис. 3.50).

```
senya@AnyMacs-MacBook-Pro phone-store % git log -3
commit fcf0c86954bddc2c3a61176431b7430d37d35850 (HEAD -> dev, origin/dev)
Author: Senya <formobileprima@gmail.com>
Date:   Sat Dec 5 18:34:03 2020 +0200

    fixed bug header children

commit 67efe9f03c0bf1110570ebebaf28c99d3a7dde3c
Author: Senya <formobileprima@gmail.com>
Date:   Wed Dec 2 22:05:35 2020 +0200

    add route for countDevice count

commit 6421c0ce5ec6bada227ba9157ea91a2d679ce620
Author: Senya <formobileprima@gmail.com>
Date:   Wed Dec 2 20:43:41 2020 +0200

:...skipping...
commit fcf0c86954bddc2c3a61176431b7430d37d35850 (HEAD -> dev, origin/dev)
Author: Senya <formobileprima@gmail.com>
Date:   Sat Dec 5 18:34:03 2020 +0200

    fixed bug header children

commit 67efe9f03c0bf1110570ebebaf28c99d3a7dde3c
Author: Senya <formobileprima@gmail.com>
Date:   Wed Dec 2 22:05:35 2020 +0200

    add route for countDevice count

commit 6421c0ce5ec6bada227ba9157ea91a2d679ce620
Author: Senya <formobileprima@gmail.com>
Date:   Wed Dec 2 20:43:41 2020 +0200
```

Рис. 3.50. Список трьох останніх комітів

**Висновки до розділу**

Обрано модель *SPA*, оскільки вона дає швидкість і простоту використання. Під цю модель обрано базу даних *mongoDb* і зберігання цієї бази даних на серверах *mongoDb*. Ця база даних буде доступна за посиланням, в якому віключено місце для логіна і пароля. Створено роути під необхідні групи товарів і експортовано їх як модулі для обробки простоти написання.

Створено інтерфейси під групи товарів, а також під екшени в редаксі.

Написано модулі для хедера, адаптованість якого забезпечується не пружністю верстки, а наявністю мобільного і десктопного хедера, які вмикаються в залежності від ширини екрана.

Створено модель картки товару, а об’єднано їх в список товарів. Додано слайдер, щоб на головній сторінці ті ж самі картки можна було прокручувати як у слайдері.

Створено сторінки з товарами мобільних пристрої та таку саму для планшетів. Реалізовано сторінки з купівлею товарів та сторінки для улюблених товарів і кошика.

# ВИСНОВОК

Проведено аналіз інформаційних джерел за тематикою дипломного проекта, в результаті визначено завдання, предмет, тематику і цільове призначення сайтів з продажу мобільної техніки. З’ясовано, що основними завданнями при створенні сайтів-магазинів є розробка чіткого ТЗ, усвідомлення потенційного розвитку і закладання можливостей для цього у архітектурі, а також обрання відповідних засобів реалізації, з урахуванням їх популярності, складності і можливого розвитку.

В ході роботи над проектом, з'ясовано, що сайти мають специфічну архітектуру і функціонал відповідно до їх категорії, і ніша інтернет магазинів не виключення. Подібні сайти мають містити такий функціонал: сортування за певними полями (ціна, популярність і т.п.); фільтрація за характеристиками та назвою; можливість додавати обрані елементі к кошик і роботи одне замовлення.

Цей функціонал є на більшості подібних сайтів і тому це фактично стандарт. Ці функцій допомагають користувачам значно простіше шукати потрібний їм товар і відповідно чим кращий юзабіліті тим більша конверсія для власника.

Важливим аспектом є адаптивність сайту, оскільки у майже всіх людей є мобільні телефони, і враховуючи потік інформації, люди звикли швидко гуглити те, що їх цікавить. Якщо хтось захоче перевірити характеристики певного телефону, а або просто по цікавитися новинками то це займе всього лишень 10 – 15 секунд доки ця людина не зайде на сайт через телефон. Ми не знаємо коли людина хоче отримати інформацію про товар, це може бути спонтанне бажання і вона не змінюватиме свої звички швидкого гуглення на користь походу в магазин.

Визначено специфіку проєкту – сайт-магазин з продажу мобільних пристрій.

Реалізовано повноцінний сайт-магазин з використанням сучасних технологій, з використанням популярних підходів до реалізації, у вигляді *SPA*. Для того щоб сайт можна було масштабувати без великої кількості багів, використовувалася технологія typeScript. Це дозволило побудувати швидкий сайт з непоганою можливістю до розширення і нарощування нового функціоналу.

Предметом таких сайтів є будь-який товар, що відноситься до категорії електронних пристроїв і відповідних аксесуарів до них.

Проаналізовано стан дослідженості проблеми розробки веб-сайтів інтернет магазинів мобільної техніки. Визначено основну проблематику цієї категорії сайтів у каїнах СНД та на європейському ринку.

Проведено аналіз вітчизняних та закордонних сайтів відповідної тематики. Визначено схожі і відмінні риси на прикладі популярних інтернет магазинів.

Окреслено етапи створення веб-сайту інтернет магазину мобільної техніки. Приведено технічну документацію у відповідність технічним можливостям.

Визначено методи, засоби та технології розробки веб-сайту інтернет магазину.

Визначено специфіку і архітектуру майбутнього проекту.

Створено сайт інтернет-магазин мобільної техніки

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Бангал, Шэм ActionScript. Основы / Шэм Бангал. – М.: Символ-плюс, 2011. – 183 с.
2. Барысов Р .И. Постройте профессиональный сайт сами. – СПб., 2013. – 304 с.
3. Бер, Бибо jQuery. Подробное руководство по продвинутому JavaScript / Бибо Бер. – М.: Символ-плюс, 2013. – 347 с.
4. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySql. ДиаСофтЮП, 2013. – 672с.
5. Будилов, Вадим Интернет-программирование на Java / Вадим Будилов. – М.: БХВ-Петербург, 2003 – 383 с.
6. Веллинг Л., Томсон Л. Разработка веб-приложений с помощью PHP и MySQL. – М.: Вильямс – 2014. – 848 с.
7. Вуд, Кит Расширение библиотеки jQuery / Кит Вуд. – М.: ДМК Пресс, 2013. – 796 с.
8. Дронов В. А. Разработка современных Web-сайтов. – СПб.: БХВ-Петербург, 2013. – 414 с.
9. Документація фреймфорку ReactJs. – Режим доступу: https://reactjs.org/
10. Дунаев, В. В. HTML, скрипты и стили /В. В.Дунаев. – М.: БХВ-Петербург, 2014. – 328 с.
11. Документація бази даних mongoDb. – Режим доступу: https://www.mongodb.com/cloud
12. Клименко, Роман Веб-мастеринг на 100 % / Роман Клименко. – М.: Питер, 2013. – 512 с.
13. Документація мови програмування typeScript. – Режим доступу: https://www.typescriptlang.org/
14. Климов, Александр JavaScript на примерах / Александр Климов. – М.: БХВ-Петербург, 2009. – 336 с.

15. Документація мови програмування javaScript. – Режим доступу: https://javascript.info/

16. Машнин, Тимур Web-сервисы Java / Тимур Машнин. – М.: БХВ-Петербург, 2012. – 560 с.

17. Никсон, Робин Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript и CSS / Робин Никсон. – М.: Питер, 2013. – 597 с.

18. Резиг, Джон JavaScript для профессионалов / Джон Резиг, Расс Фергюсон , Джон Пакстон. – М.: Вильямс, 2015. – 240 с.

19. Роббинс, Дженнифер HTML5, CSS3 и JavaScript. Исчерпывающее руководство (+ DVD-ROM) / Дженнифер Роббинс. – М.: Эксмо, 2014. – 528 с.

20. Сухов К. К. Node.js. Путеводитель по технологии / К. К. Сухов. – М.: ДМК Пресс, 2015. – 254 с.

21. Херман Д. Сила JavaScript. 68 способов эффективного использования JS / Д. Херман. – М.: Питер, 2013. - 339 с.

22. Хэррон, Дэвид Node.js Разработка серверных веб-приложений на JavaScript / Дэвид Хэррон. – М.: ДМК Пресс, 2012. – 542 с.

23. Хэррон, Дэвид Node.js. Разработка серверных приложений на JavaScript / Дэвид Хэррон. – М.: ДМК Пресс, 2016. – 144 с.

24. Глушаков С. В. Программирование Web-страниц. JavaScript. VBScript / С. В. Глушаков, И. А. Жакин, Т. С. Хачиров. – М.: Фолио, 2013. – 390 с.

25. Климов Александр JavaScript на примерах / Александр Климов. – М.: БХВ-Петербург, 2009. – 300 с.

26. Макфарланд Дэвид JavaScript и jQuery. Исчерпывающее руководство (+ DVD-ROM) / Дэвид Макфарланд. – М.: Эксмо, 2012. – 688 с.

27. Машнин Тимур JavaFX 2.0. Разработка RIA-приложений / Тимур Машнин. – М.: БХВ-Петербург, 2012. – 320 с.

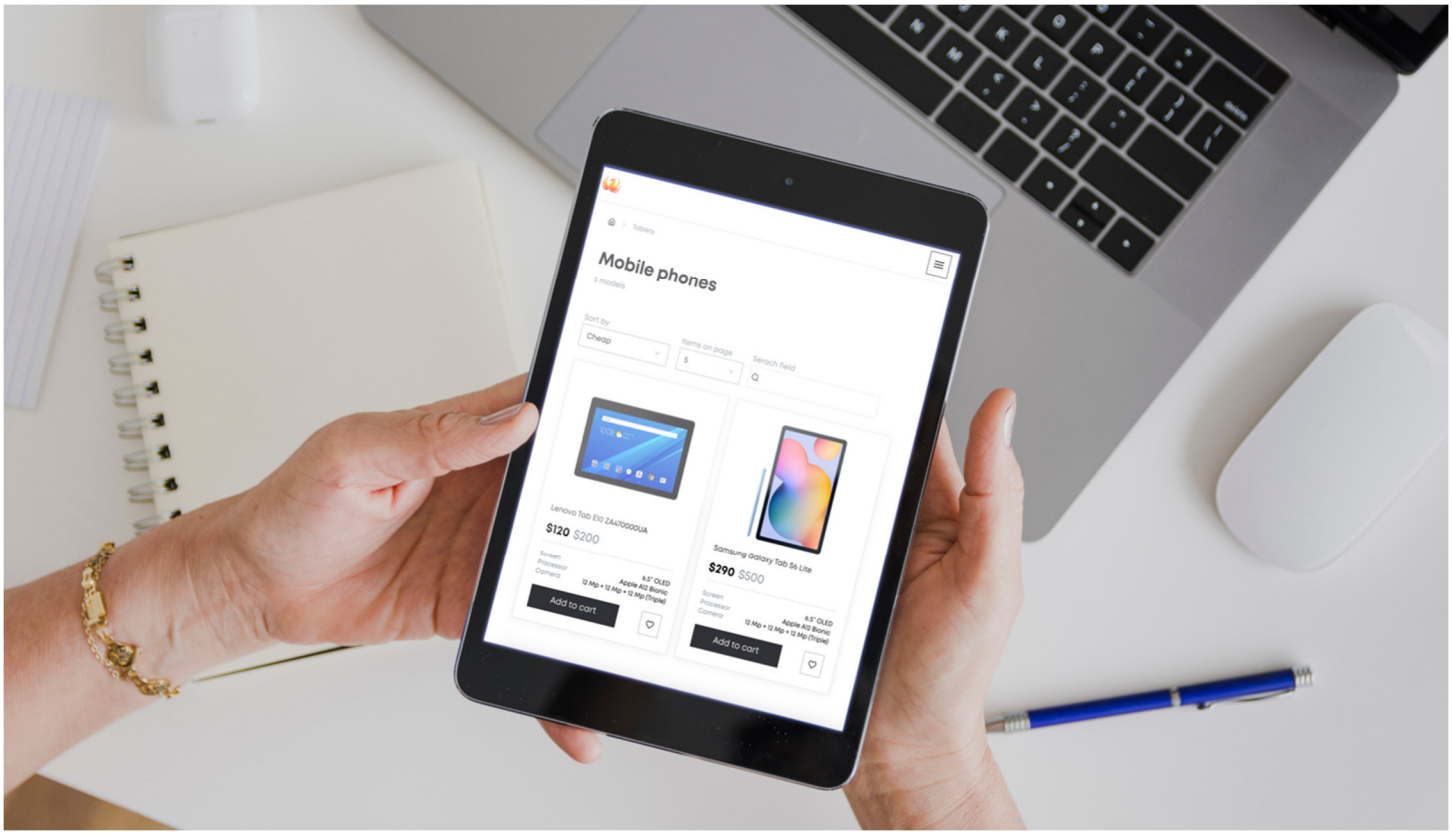
28. Резиг Джон JavaScript для профессионалов / Джон Резиг, Расс Фергюсон, Джон Пакстон. – М.: Вильямс, 2015. – 240 с.

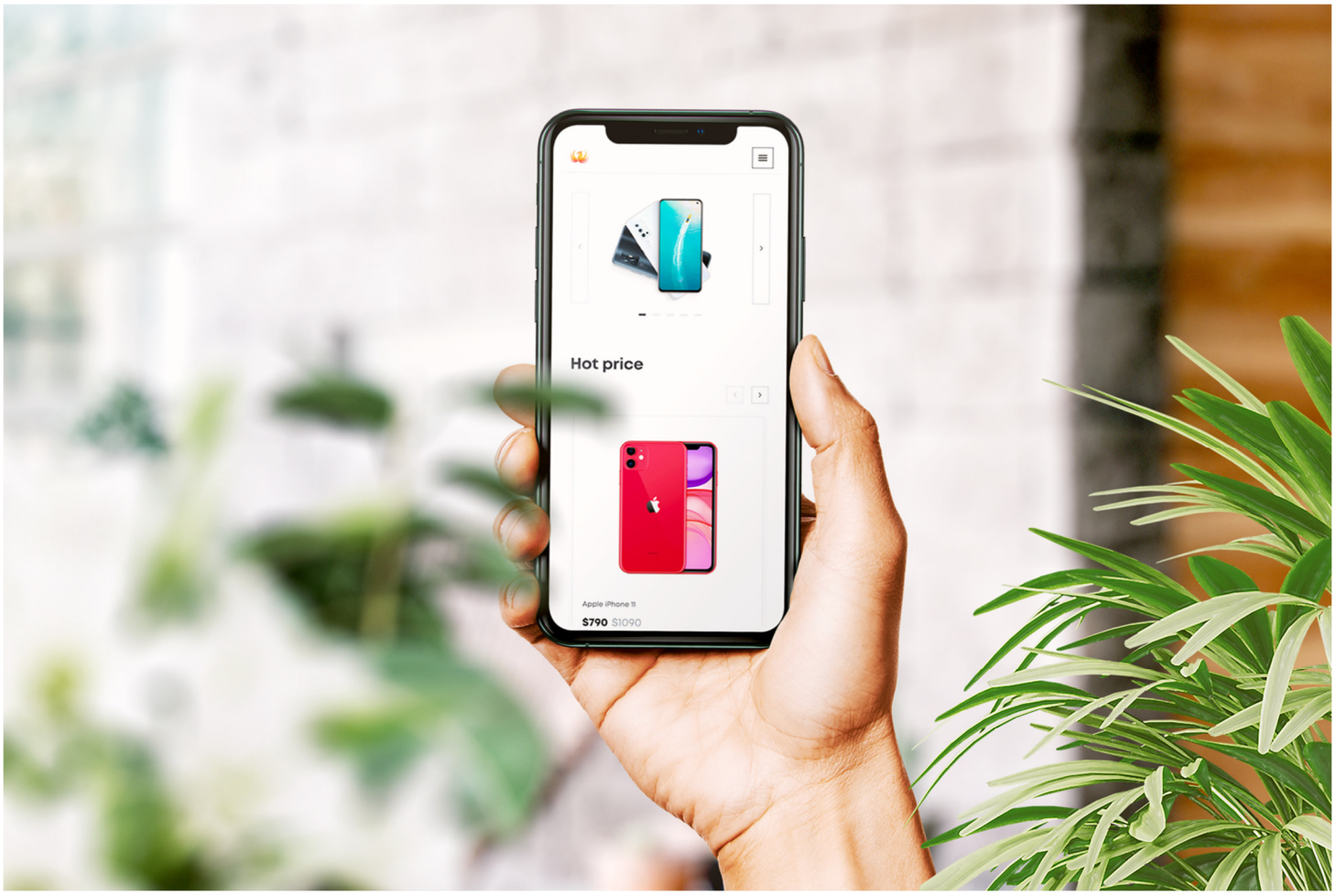
29. Флэнаган Дэвид JavaScript. Карманный справочник / Дэвид Флэнаган. – М.: Вильямс, 2013. - 320 с.

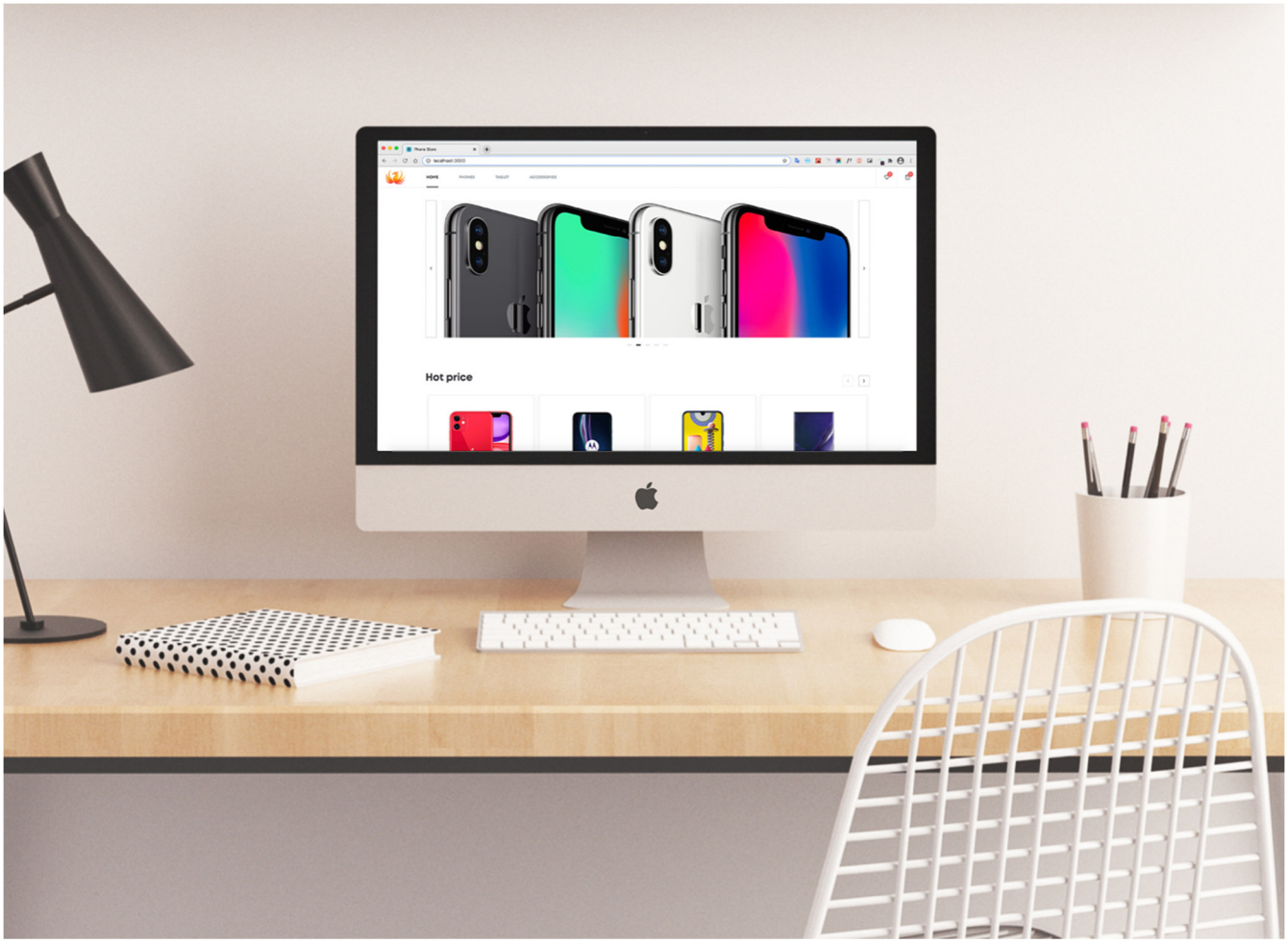
30. Фримен, Адам jQuery 2.0 для профессионалов / Адам Фримен. - М.: Вильямс, 2015. - 949 с.

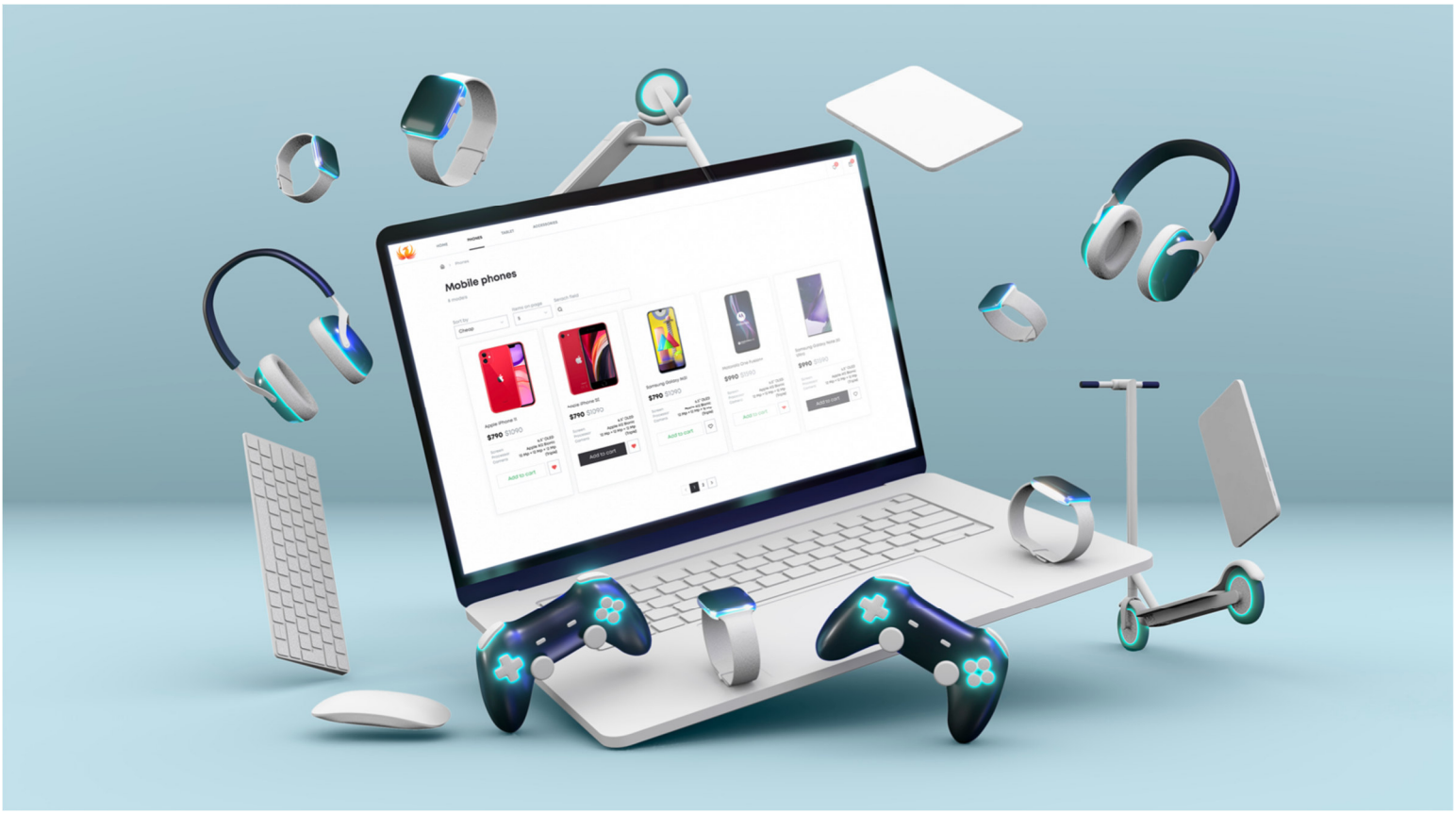
31. Чекко, Рафаэлло Графика на JavaScript / Рафаэлло Чекко. – М.: Питер, 2013. – 492 с.

32. Штефен, Вальтер Создание приложений для Windows 8 с использованием HTML5 и JavaScript / Вальтер Штефен. – М.: ДМК Пресс, 2013. – 735 с.


Mobile phones
Cheap
Lenovo Tab E10 ZA470000UA
$120 $200
Add to cart
Samsung Galaxy Tab S6 Lite
$290 $500
Add to cart


Hot price
Apple iPhone 11
$790 $1090


Hot price


Mobile phones