НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської роботи

на ступінь вищої освіти бакалавр
на тему: «**Розробка адаптивного web-додатку за допомогою HTML, CSS та мови програмування JavaScript**»

Виконав: студент 4 курсу, групи ПД– 43
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Сарданов В.І.
(прізвище та ініціали)
Керівник Трінтіна Н.А.
(прізвище та ініціали)
Рецензент _____
(прізвище та ініціали)
Нормоконтроль _____
(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти - «Бакалавр»
Спеціальність - 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії програмного забезпечення
О.В. Негоденко

“_____”_____2022 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Сарданову Володимиру Ігоровичу
(прізвище, ім‘я, по батькові)

1. Тема роботи: «Розробка адаптивного web-додатку за допомогою HTML, CSS та мови програмування JavaScript»

Керівник роботи кандидат технічних наук Трінтіна Н.А.,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “18” лютого 2022 року № .

2. Строк подання студентом роботи 03.06.2022.
3. Вихідні дані до роботи:
   - 3.1. Середовище розробки Visual Studio Code;
   - 3.2. Структура веб додатку;
   - 3.3. Науково-технічна література, пов’язана з розробкою;
   - 3.4. Методи побудови адаптивних web-додатків;
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
   - 4.1. Аналіз предметної області
   - 4.2. Аналіз програмного забезпечення для створення адаптивного вебдодатку

4.3. Огляд засобів реалізації для створення адаптивного вебдодатку

4.4. Створення та реалізація адаптивного вебдодатку

5. Перелік графічного матеріалу.

5.1. Титульний слайд

5.2. Мета, об’єкт та предмет дослідження

5.3.Актуальність роботи

5.4.Аналоги 5.5.Порівняння з аналогами5.6.Технічне завдання

5.7.Програмні засоби реалізації

5.8.Інструменти використані для реалізації

5.9.Архітектура вебдодатку

5.10.Апробація результатів дослідження

5.11.Висновки

6. Дата видачі завдання: 11.04.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	19.04.2022	Виконано
2	Провести огляд та аналіз існуючих сайтів	20.04.2022	Виконано
3	Визначити недоліки існуючих методів сайтів для вирішення задач предметної галузі та проблемні питання	21.04.2022 – 22.04.2022	Виконано
4	Розробка адаптивного вебдодатку	23.04.2022 – 27.04.2022	Виконано
5	Висновки, оформлення роботи	27.04.2022 – 30.04.2022	Виконано
6	Розробка демонстраційних матеріалів	30.04.2022	Виконано
7	Попередній захист роботи	02.06.2022	Виконано
8	Здача роботи	03.06.2022	Виконано

Студент _____ Сарданов В.І..
(підпис) (прізвище та ініціали)

Керівник роботи_____ Трінтіна Н.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи: 68с., 38 рис, 1 дод., 3 табл., 26 джерел.

Ключеві слова: Visual Studio Code, HTML, CSS, JavaScript, React, GIT, бібліотека, адаптив, вебдодаток, сайт, статичний сайт, інтерфейс, вміст.

Об'єкт дослідження – процес динамічного виведення вмісту в адаптивному вебдодатку.

Предмет дослідження – адаптивний вебдодаток з динамічним виведенням вмісту.

Мета роботи – спрощення процесу виведення вмісту на сайті.

Аналіз особливостей процесу динамічного виведення вмісту в адаптивному вебдодатку та існуючих засобів для реалізації цього дозволяє сформувати вимоги до майбутнього програмного продукту.

Адаптивний вебдодаток повинен забезпечувати наступні функції:

- повинен бути сумісний з усіма платформами;
- коректно відображатись на будь-якому екрані;
- виводити актуальну інформацію;
- вірно відображати дані з сервера;
- постійно перевіряти наявність нових даних на сервері;
- динамічно виводити вміст на сайті по мірі запиту користувача;
- мати інтуїтивно зрозумілий інтерфейс.

Галузь використання – ресурс може використовувати будь-яка людина, яка має необхідність в фукнціоналі, яке надає розроблене програмне забезпечення.

ЗМІСТ

ВСТУП

Актуальність теми: Контент на різних екранах відображається по різному і саме тому, щоб контент веб-сайтів вірно відображався на моніторах з різною роздільною здатністю, в тому числі і на мобільних пристроях, необхідна динамічна адаптація. Тобто, адаптивний дизайн сайтів дозволяє відобразити весь контент веб-сайтів для їх зручного перегляду будь-де. Як результат, всі частини контенту будуть коректно розміщені на будь-якому екрані, буде то монітора комп'ютера або планшета.

Ступінь вивчення проблеми: За даними Pew Research Center, до кінця 2015 року 68% дорослих американців мали смартфон, а кількість власників планшетів зросла до 45%. Також встановлено, що деякі вікові групи наближаються до точки насичення смартфонами – 86% людей віком від 18 до 29 років мають смартфон. І це - розглядаючи приклад тільки Америки, але така тенденція зберігається по всьому світу, тим паче після початку пандемії, коли придбання гаджетів дуже виросло через перехід на дистанційне навчання та роботу. Тому це й означає, що потреба в повністю адаптивному дизайні значно зросла, адже адаптувати сайти під так велику кількість гаджетів більш трудомістка по часу та ресурсам робота.

Джерела дослідження: Самими головними джерелами дослідження, є навчально-наукова література, яка пов'язана з розробкою адаптивних вебдодатків на мові програмування JavaScript, та інтернет ресурси на тему програмування потрібного функціоналу й розробки мобільних сервісів.

Об'єктом дослідження є процес динамічного виведення вмісту в адаптивному вебдодатку.

Предметом роботи є адаптивний вебдодаток з динамічним виведенням вмісту

Метою роботи є спрощення процесу виведення вмісту на сайті.

Завданням роботи є розробка адаптивного вебдодатку, покращення сприйняття користувачами вмісту на сайтах при розважанні, навчанні чи навіть придбанні якихось товарів.

Методика дослідження: спочатку був створений адаптивний вебдодаток, з інтуїтивно зрозумілим інтерфейсом, щоб користувачі могли з легкістю переглядати наданий на сайті вміст й не запутуватись в белічі кнопок, за можливості підтримки його розробником з метою ефективного додавання нового функціоналу або виправлення виявлених недоліків при збивання налаштувань сервера, звідки динамічно вивантажується інформація.

Вебдодаток, на основі бібліотеки мови програмування JavaScript під назвою React, дозволяє вирішити всі перелічені вище проблеми, оскільки однією з переваг React являється його можливість «адаптувати» виведену інформацію динамічно та як завгодно програмістові.

Наукова новизна роботи: Таким чином, наукова новизна полягає в створенні адаптивного вебдодатку.

Практична значущість результатів: даний продукт орієнтований на можливість користуванням ним всіх вікових категорій.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Загальний аналіз та опис предметної області

Web-додаток (далі веб-додаток) - простіше кажучи, це програма, яка запускається в браузері користувача (клієнт-серверний додаток), а також може називатися адаптивним сайтом (далі адаптивний сайт або просто сайт). Інтернет-додатки можуть бути як простими (наприклад - сайт з особистим кабінетом), так і досить складними, наприклад, ERP-системи, розподілені за моделлю SaaS.

В Internet Explorer розмір макета може динамічно змінюватися в міру зміни розміру. Однак у Netscape сторінку довелося перезавантажувати з сервера, коли буде змінено розмір вікна сайту (тобто самого браузеру). Audi.com, який був запущений в кінці 2001 року, був першим сайтом, який використовував макет для всього браузера. Сайт розроблено razorfish.

Так само вони описали теорію, разом з практичним використанням адаптивного веб-дизайну у їх книзі Responsive Web Design, яка була опублікована у 2011 році. В травні 2010 року Ітан Маркотт ввів термін адаптивного веб-дизайну та описав найважливіші принципи у статті «A List Apart».

Адаптивний дизайн посів друге місце в списку найкращих трендів веб-дизайну 2012 року по версії журналу «.net». А сайт Mashable назвав 2013 рік роком адаптивного веб-дизайну. Інші джерела, яких забагато, також рекомендують адаптивний дизайн в вигляді економічно ефективної альтернативи мобільним додаткам, які тільки починають поширюватися.

Адаптивний дизайн – це здатність Інтернет-ресурсу адаптуватися до технічних параметрів будь-якого монітора персонального комп’ютера, ноутбука, планшета або смартфона. Сучасний адаптивний (гумовий) дизайн вимагає набагато більше праці та часу, витраченого на веб-розробку. Його головна мета — використовувати адаптивну верстку — підвищити рівень

зручності використання (використання) і досягти значно вищих коефіцієнтів конверсії (трафіку) для всіх користувачів, незалежно від типу пристрою.

Адаптивний дизайн є надзвичайно важливим елементом для успішного та прибуткового бізнесу в Інтернеті. Такий макет слід використовувати постійно при запуску будь-якого веб-проекту, незалежно від його тематики чи типу.

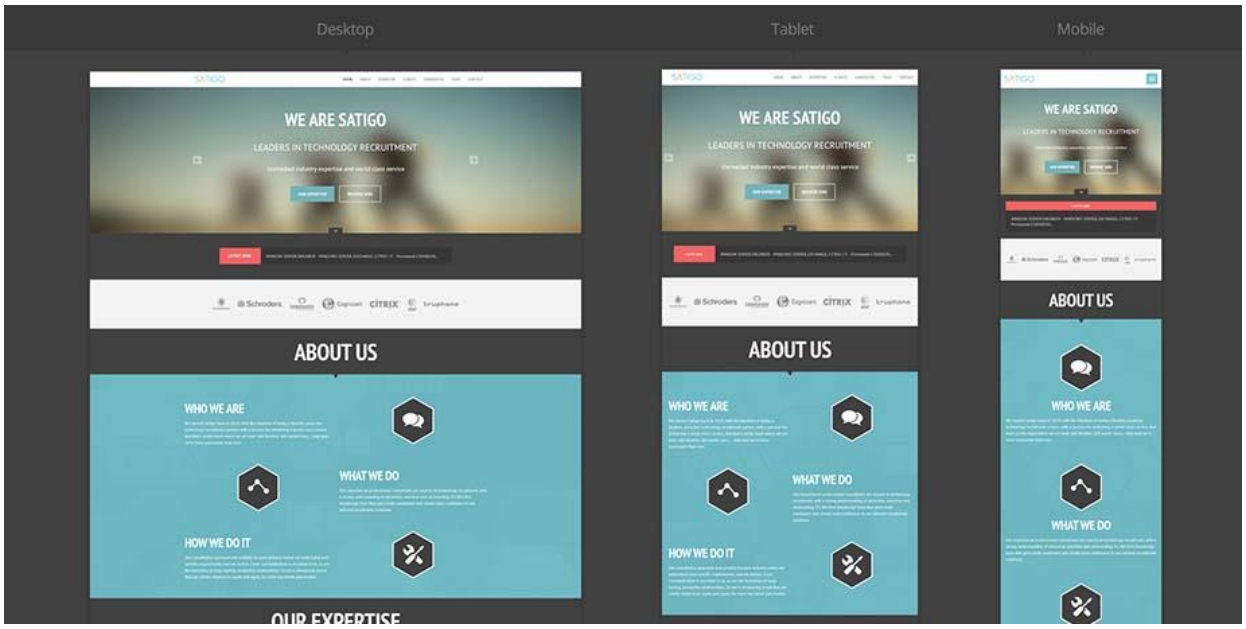


Рисунок 1.1 – Макет дизайну сайту

1.2. Аналіз потреб в адаптивному дизайну сайту

Сучасні інтернет-ресурси постійно змагаються за підтримку більшості користувачів. Сьогодні, окрім персональних комп’ютерів чи ноутбуків, в Інтернеті є мобільні гаджети. Якісний дизайн мобільної версії сайту забезпечує цільову аудиторію для замовлення послуг або покупки товарів онлайн за допомогою мобільних пристроїв. Взагалі кажучи, це легко зробити, не адаптуючи веб-сайт під різні пристрої та розширення екрану, але тепер це питання завжди чути при розробці кожного нового проекту чи вдосконаленні існуючого.

Найважливішим фактором, який впливає на перетворення цільового

трафіку в замовлення, є зручність використання сайту. А правильний дизайн і розробка сайту – основа зручних ресурсів. Якщо користувач задоволений сайтом, просто шукайте потрібну інформацію, переглядайте веб-сторінки ресурсу, такий сайт завжди запам’ятається і користувач захоче зайти ще раз, коли це необхідно. Поступово формується ваша звичайна аудиторія і підвищується коефіцієнт конверсії. Тому адаптивний і адаптивний дизайн підвищує рівень зручності використання. Користувачі можуть переглядати акуратно згруповані блоки вмісту та з легкістю користуватися сайтом, знаходячи все, що їм потрібно. Інакше практично кожен користувач покине ваш ресурс і почне переглядати сайт конкурента.

Для успішного просування також необхідний адаптивний дизайн сайту. Відсутність такого дизайну означає, що сайт втрачає більшість цільових споживачів, які використовують мобільні пристрої для веб-серфінгу. Результатом є сам дохід. Але, крім того, сайти з неадаптивним дизайном навчилися розпізнавати самі пошукові системи, які в першу чергу аналізують контент, що відображається на мобільних пристроях. Зміст мобільної та настільної версій ресурсів має бути повністю ідентичним.

Рисунок 1.2 –Зображення адаптивного сайту

Важливо розуміти, що перевага надається адаптивним сайтам. Активи, які не розроблені адаптивно, займають нижчі місця в рейтингу пошукових систем. Якщо ваші онлайн-проекти є основним джерелом онлайн-продажів, то ви не можете жити без адаптивного дизайну. Користувачі швидко залишають неякісні сайти і не люблять довго чекати і докладають багато зусиль, щоб знайти потрібну інформацію або здійснити заплановані дії.

Отже, цей дизайн веб-сайту необхідно використовувати для:

- Підвищити рівень доступності;
- Підвищити коефіцієнти конверсії;
- Успішне просування активів у провідних пошукових системах;
- Формування цільової аудиторії та лояльність користувачів до сайту.

1.3. Типи адаптивного сайту

Існує 5 основних типів адаптивного веб-дизайну:

1. Гумовий графік. Цей метод став найпоширенішим. Сайт налаштовується, стискаючи ширину блоків до розміру екрана мобільних пристроїв. Знизу розташовуються блоки, які не дають усадки.

2. Блокова передача. Цей варіант підходить для сайтів з багатоколонковою структурою. Якщо блоки не поміщаються на екрані - вони просто зміщуються вниз.

3. Розробка макета. Макет розроблений для кожної роздільної здатності екрана. Цей спосіб дуже трудомісткий, тому не підходить для простих сайтів (для них достатньо просто переміщення блоків). Але він використовується для бізнес-сайтів, веб-сайтів компаній, лендингов (односторінкових веб-сайтів) і особливо для інтернет-магазинів. При цьому розкладка розроблена для всіх базових дозволів екрану, тому що тут потрібно зосередити увагу користувача на важливих блоках і направити його на бажану цільову дію.

4. Масштабування тексту та малюнків. Чудовий варіант для

найпростіших сайтів. Не всі веб-ресурси масштабуються, лише деякі блоки. Це рішення не відрізняється великою гнучкістю, тому воно також не популярно.

5. Панелі. Ця опція прийшла з мобільних додатків. При горизонтальному або вертикальному натисканні тут з’являється додаткове меню. А це не завжди зручно, оскільки вимагає додаткових дій від користувача.

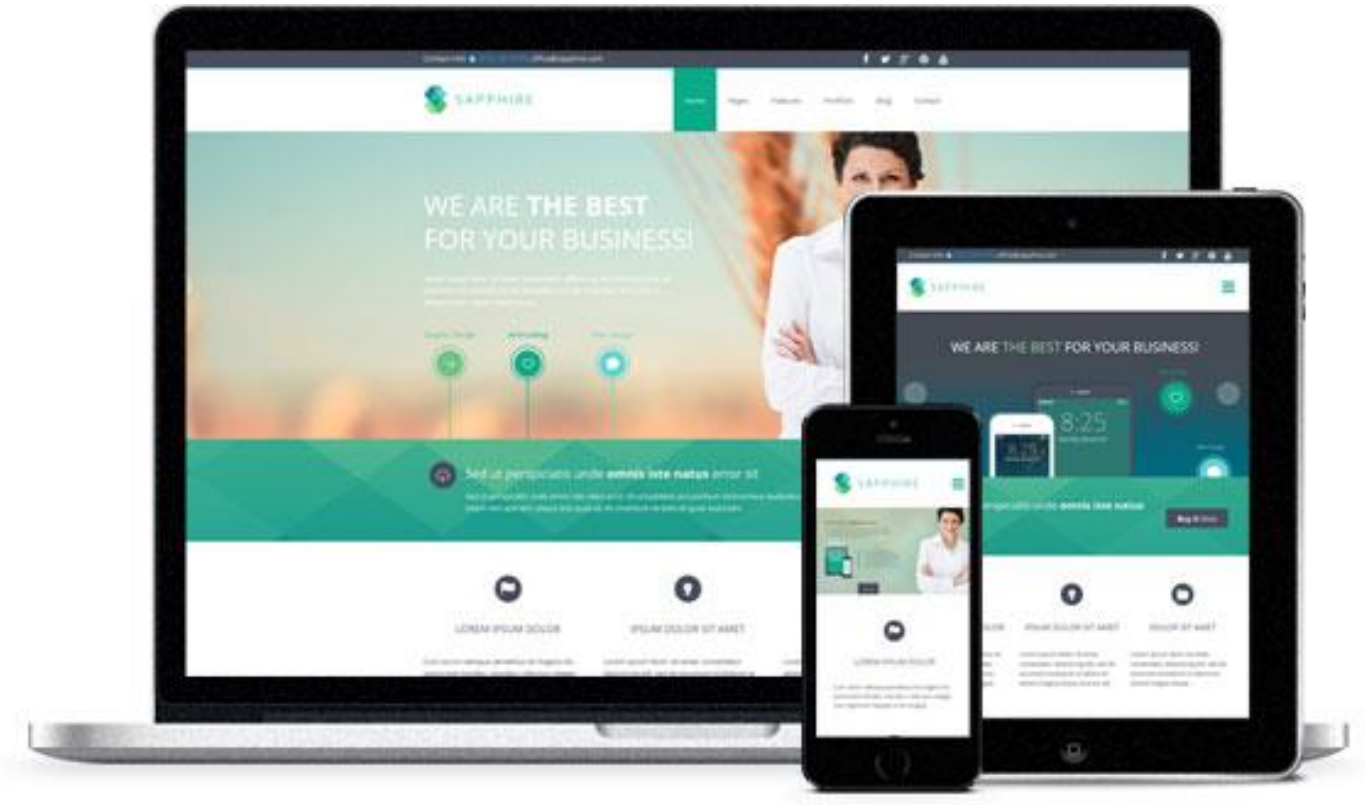


Рисунок 1.3 – Приклад мобільного дизайну в порівнянні з десктопною версією

Жоден з цих варіантів не є універсальним рішенням. Тому вибирається найкращий тип адаптивного дизайну для кожного сайту – враховуючи завдання, які він має виконувати.

1.4. Переваги та мінуси адаптивного дизайну сайту

1.4.1. Переваги адаптивного дизайну сайту

- відображати вміст правильно незалежно від роздільної здатності екрана та браузера;

- позитивне ранжування сайту в пошукових системах;

- Знижені витрати праці на створення мобільних версій;

- дозволяють охопити більш широку аудиторію;

- зробити мережевий маркетинг більш ефективним;

- Введення гнучких кодів CSS можна адаптувати до веб-сайту в будь-якій ситуації.

Процес розробки починається з оформлення сторінок з різною роздільною здатністю екрана для перегляду на комп'ютерах, ноутбуках, смартфонах, планшетах. Враховуйте, що користувачам потрібна максимально зручна навігація по сайту, інакше вони просто покинуть сайт і втратять частину потенційних клієнтів. Тому, розробляючи адаптивність веб-сайту, видаліть зі сторінки непотрібні елементи, збільште розмір кнопок і посилань для зручності використання, оберіть читабельний шрифт для тексту та виберіть зручні місця розташування блоків контенту.

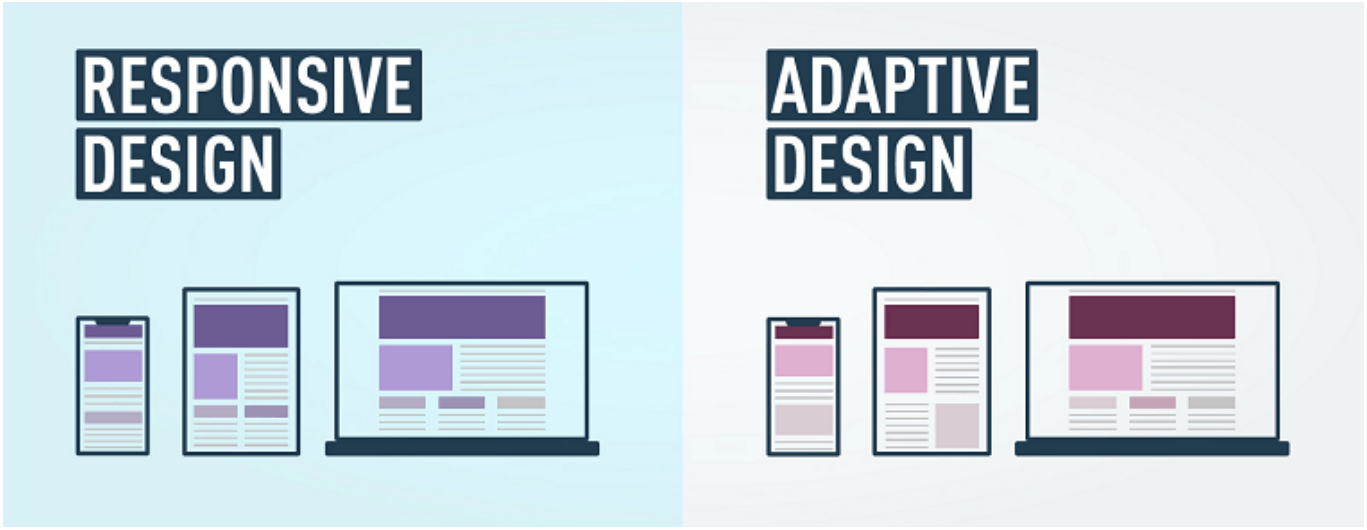


Рисунок 1.4 – Порівняння чутливого та адаптивного дизайну

1.4.2. Мінуси адаптивного дизайну для сайту

1. На адаптивному сайті потрібно виключити деякі графічні та технічні елементи, щоб зробити його зручнішим для користувачів.

2. Оскільки всі сторінки мають однакову адресу, неможливо вказати посилання на звичайну версію (це можливо в мобільній версії).

3. Адаптивні веб-сайти важчі, ніж звичайні настільні чи мобільні версії, і тому завантажуються повільніше.

Однак жоден з цих недоліків не є критичним. Пункти 1 і 2 не мають значення і насправді не працюють. Третій момент легко вирішується за допомогою грамотної технічної оптимізації (швидшого завантаження сайту). Підтверджує це той факт, що майже всі веб-сайти пропонують адаптивний дизайн. Особливо для тих, хто входить в ТОП-10 пошукових систем.

1.5. Відмінності адаптивного дизайну сайту і мобільної версії

Адаптивна та мобільна версії сайту – це два абсолютно різні рішення. Вони відрізняються за всіма аспектами: способом розробки, швидкістю завантаження і навіть вартістю. Візуально всі відмінності між адаптивним дизайном і мобільною версією наведені в таблиці 1.

Всі недоліки адаптивного дизайну – це переваги мобільної версії. Знову ж таки, всі мінуси мобільної версії – це переваги адаптивного дизайну. Отже, це два абсолютно різних рішення, які зазвичай не порівнюють.

Таблиця 1.1. Відмінності адаптивного дизайну і мобільної версії

	Адаптивний дизайн	Мобільна версія
Ключова особливість	Дизайн адаптується під роздільну здатність пристрою (будь-то комп'ютер, ноутбук чи смартфон). Той контент, що не підходить по розміру, або стискається по ширині (за можливості), або зміщується вниз. Тоді дизайн веб-сайту залишився таким самим – а лишень трохи змінилось розташування вмісту.	Веб-сайт досить простий, від якогось контенту та функціоналу доведеться позбутися. В результаті він не схожий на основну настільну версію.
URL-адрес	Має ту саму URL-адресу, що й основний сайт	Створено на окремому субдомені
Швидкість завантаження	Завантаження не дуже швидке, іноді повільне через слабке з'єднання, через те, що займає більше місця. Але це виправляється додатковою технічною оптимізацією.	Має спрощену конструкцію, тому завантаження йде швидко.
Вартість розробки	Він створюється відразу під час розробки веб-сайту, не вимагає поглибленних знань, а тому коштує недорого.	Це досить дороге рішення. Вартість у порівнянні з розробкою нового мобільного додатка.

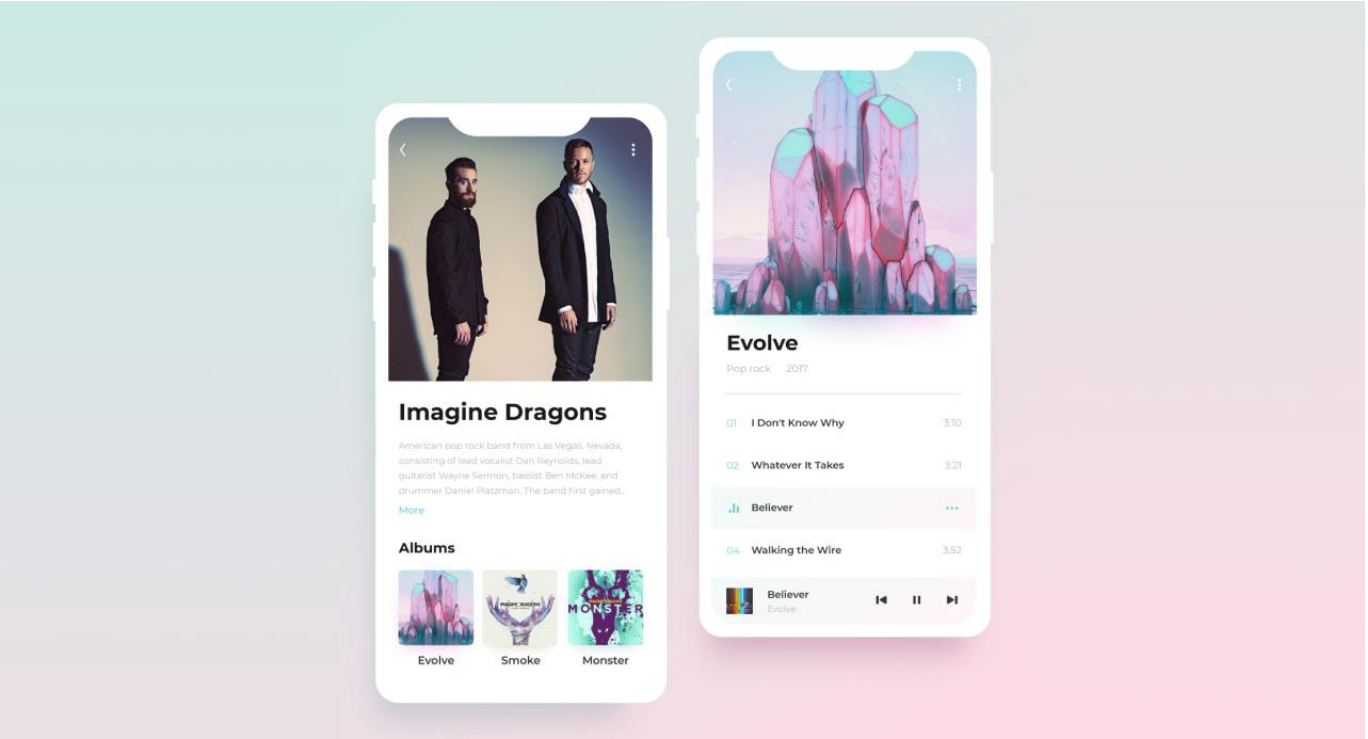


Рисунок 1.5 – Приклад мобільного дизайну

Але якщо ви вибираєте, на чому зосередитися, це, безумовно, адаптивний дизайн. Мобільна версія втратила свою актуальність і вважається пережитком минулого. Адаптивний дизайн зараз у моді, про що свідчать вимоги, які висуваються до веб-сайтів пошуковими системами.

Основні етапи розробки адаптивного дизайну сайту це підтверджують:

1. Дизайн інтерфейсу. Адаптивний дизайн для різних пристроїв.
2. Реалізуйте адаптивний макет.
3. Тест макета.
4. Розмітка та інтеграція системи CMS.

Загальні та високорівневі фреймворки інтерфейсу користувача, середовища програмування та мови для розробки для створення адаптивних сайтів. Такі мови, як CSS3 і HTML5, вже давно є важливими, оскільки вони дозволяють автоматично змінювати розмір веб-сторінок і гнучко отримувати мультимедійні дані. Фреймворки інтерфейсу користувача: Bootstrap Foundation, Materialize Skeleton, Cardinal, Semantic UI. Бібліотеки на основі мови програмування JavaScript: Angular.js, jQuery, React.js, Vue.js. і

препроцесори CSS3, такі як LESS і SASS

Через це була використана бібліотека React, оскільки вона вирішує всі проблеми звичайних сайтів, проблеми часткового оновлення вмісту веб-сторінки, які зустрічаються при розробці односторінкових додатків.

1.6. Порівняння адаптивних сайтів та звичайних

Аналіз програмного забезпечення для розробки адаптивного веб-сайту почався з детального аналізу існуючих сайтів, але без підтримки динамічного оновлення вмісту сторінки, з повним перезавантаженням, до якого більшість людей звикли, але не зручним для постійного використання.

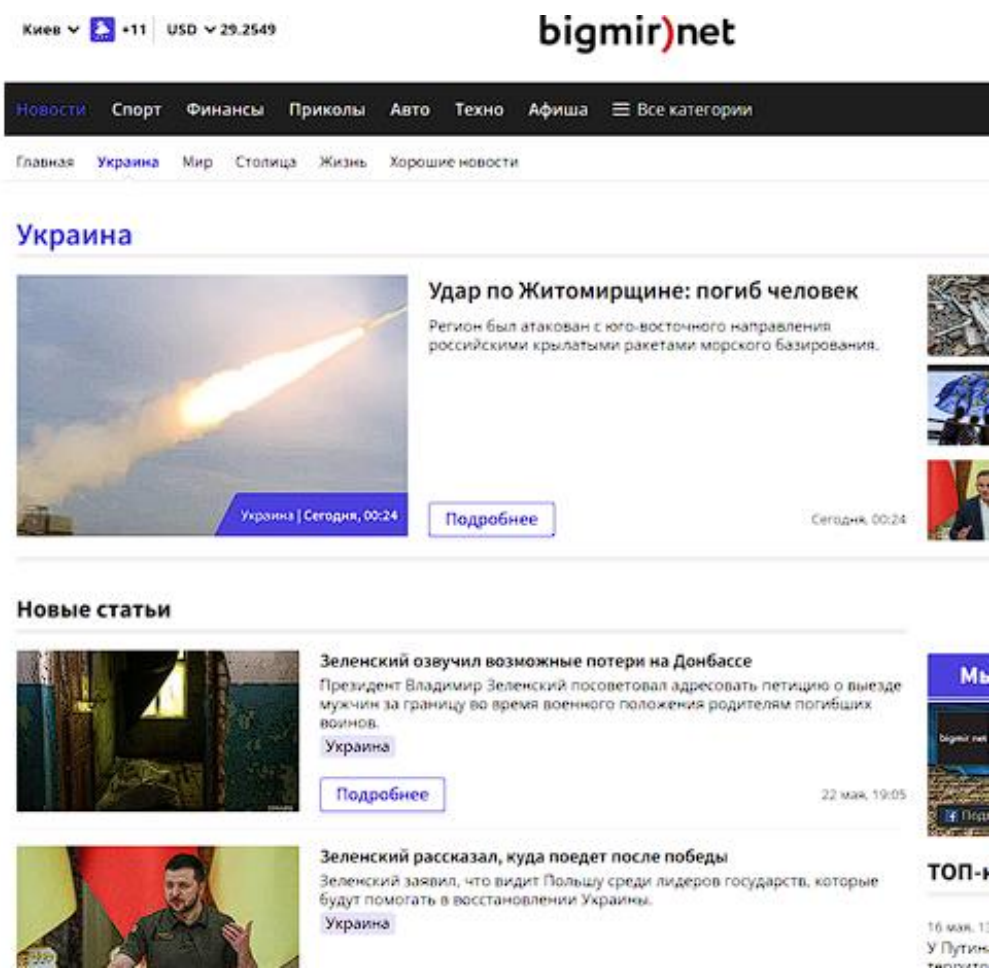


Рисунок 1.6 –Вигляд bigmir.net

Для цього ми порівняли сайти старої моделі bigmir.net і ukr.net, для

зручності порівняння їх розмістили в одну колонку, так як вони схожі один на одного, містять новини та інше, для порівняння лише використовувалася новина. Натомість вони взяли адаптивний веб-додаток instagram.com, оскільки він був розроблений з використанням бібліотеки React, мови програмування JavaScript. Це універсальний додаток, тому що, як і bigmir.net та ukr.net, містить, крім новинних «сайтів» (груп окремих тем), ще купу сторінок, просто живих людей. Він виділяється серед цих трьох кандидатів на порівняння тим, що при переході від сторінки до сторінки цей веб-додаток оновлює вміст так, що користувач навіть не помічає, що сторінки оновлюються, оскільки він динамічний, кожна новина подається як окремий пост із зображенням і кожен користувач може підписатися тільки на ті групи, які його цікавлять.

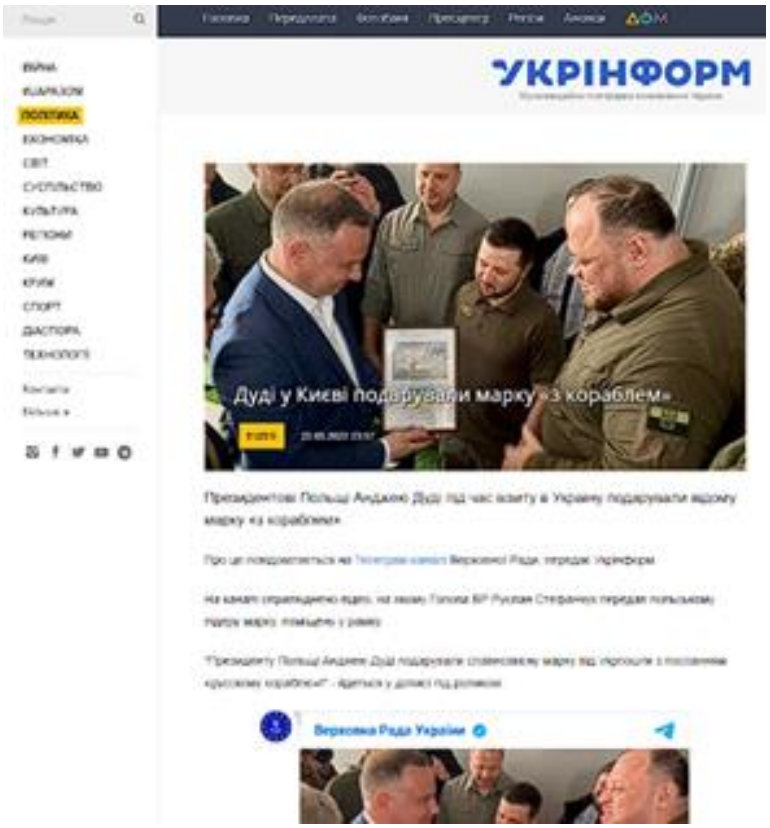


Рисунок 1.7 –Вигляд ukr.net

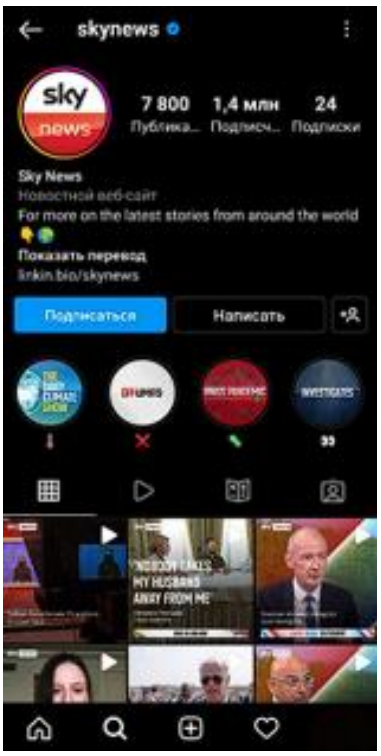


Рисунок 1.8 –Вигляд instagram.com

Таблиця 1.2. Порівняння статичних сайтів та динамічних.

	bigmir.net, ukr.net	instagram.com
Як відбувається оновлення сторінки	Вся сторінка оновлюється відразу	Оновлення динамічне
Фільтрація	Всі новини показуються відразу на вході, категорії ще потрібно знайти.	Ви можете вибрати окремо або все разом, все зрозуміло завдяки інтуїтивно зрозумілому інтерфейсу
Інтереси	Без доповнення.	Можливість налаштувати рекомендації.
Просування	Оскільки статичні сайти важче оновлювати без генераторів, вони можуть бути менш привабливими для пошукових систем.	Regularly updated resources with fresh content, including more dynamic sites, are now most often found in the distribution tops.
SEO	Ви можете проводити рекламні кампанії, але не так продуктивно.	Завдяки динамічному завантаженню, як правило, легше проводити рекламні кампанії з динамічними сторінками.

Таблиця 1.2. Продовження

Адміністрування	Статичні адміністратори сайту повинні відредагувати код, щоб оновити сайт. І якщо потрібно було зробити одне і те ж оновлення на кількох сторінках, то потрібно було змінити код кожної сторінки окремо. Зараз у цьому допомагають генератори статичних сайтів, але сайт перебудовується з нуля.	Є можливість редагувати проект через прості панелі прямо в браузері, що значно спрощує завдання оновлення сайту.
Розвиток	Статичні сайти все ще трохи складніше розробляти та оновлювати, ніж динамічні.	Динамічні сайти пропонують практично безмежні можливості для розвитку проектів, які досягаються за рахунок підключення розширень, плагінів та іншого програмного забезпечення.
Розробка	Статичні сайти з потребою прописувати код для кожної окремої сторінки динамічно програють, тому що розробка статичного сайту без знання програмування або допомоги експертів не вдасться.	Ви можете легко створювати з готових шаблонів і елементів. А створити динамічний сайт на базі будь-якої простої CMS буде легко, якщо потрібно швидко створити сайт. Це так само складно для глобальних проектів, але сама розробка набагато легша.
Розміщення на хостингу	Перенесення статичний веб-сайту на інший хостинг швидше, простіше і без особливих проблем.	Викладення динамічного веб-сайту завжди дорожче, ніж статичного, але вам не потрібно нічого додатково розміщувати на кожній сторінці.

2. ОГЛЯД ЗАСОБІВ РЕАЛІЗАЦІЇ ДЛЯ СТВОРЕННЯ АДАПТИВНОГО ВЕБДОДАТКУ

2.1. Опис середовища розробки Visual Studio Code

Visual Studio Code (далі — VSCode) — це інструмент для створення, редагування та налагодження сучасних веб-додатків і програм для хмарних систем. Visual Studio Code поширюється безкоштовно і доступний для платформ Windows, Linux і OS X (Рис. 2.1).

Microsoft запустила Visual Studio Code у квітні 2015 року на своїй конференції Build 2015, і середовище розробки стало першим кросплатформним продуктом у сімействі Visual Studio.

Visual Studio Code заснований на роботі безкоштовного проекту Atom, розробленого GitHub. Також Visual Studio Code — це надбудова до Atom Shell, яка використовує механізм браузера Chromium та Node.js. Цікаво, що використання роботи безкоштовного проекту Atom на веб-сайті VSCode, і в прес-релізі та офіційному блозі немає згадувань.

Редактор містить вбудований налагоджувач, інструменти для роботи з Git і інструменти рефакторингу, навігацію по коду, автозаповнення для стандартного дизайну та контекстні підказки. Продукт підтримує розробку на платформах ASP.NET і Node.js і позиціонується як просте рішення, яке можна обійтися без повністю інтегрованого середовища розробки. Підтримувані мови та технології включають: JavaScript, C++, C#, TypeScript, jade, PHP, XML, Java, F#, Coffee Script, HTML, R, Objective-C, Haxe, Luna, Visual Basic, JSON, Markdown, CSS, LESS, SASS, DockerFile, Python і PowerShell

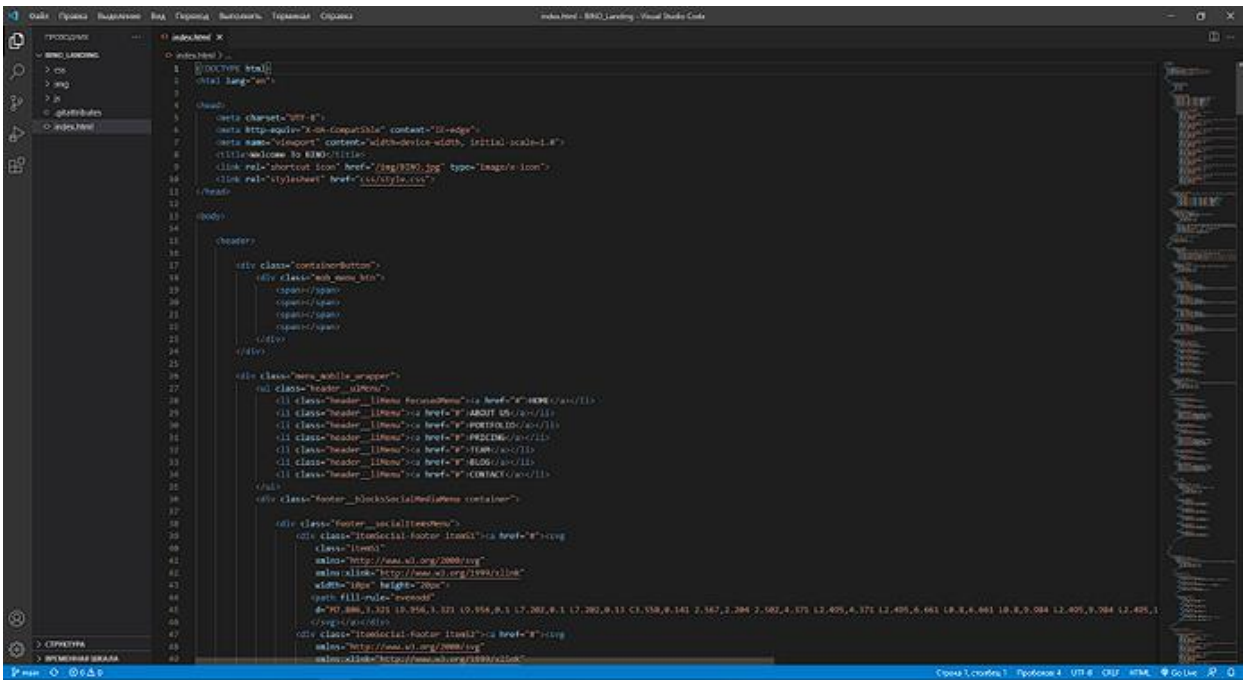


Рисунок 2.1 – Зображення відкритого проекту в VSCode

2.1.1. Багато каталоговість в Visual Studio Code

Це дозволяє користувачеві відкривати один або кілька каталогів замість системи проекту, які потім можуть бути збережені в робочому просторі для подальшого повторного використання (Рисунок 2.2). Це робить його незалежним від мови редактором коду для будь-якої мови. Він підтримує кілька мов програмування та набір функцій, які відрізняються для кожної мови. Ви можете використовувати налаштування для видалення небажаних файлів і папок з дерева проекту. Багато функцій VSCode недоступні через меню або інтерфейс користувача, але доступ до них можна отримати через палітру команд. Командний рядок також допомагає користувачеві, оскільки це дуже зручно, оскільки він повторно використовує підказку у вигляді останньої використаної команди або підказки про можливе продовження введеної вами команди.

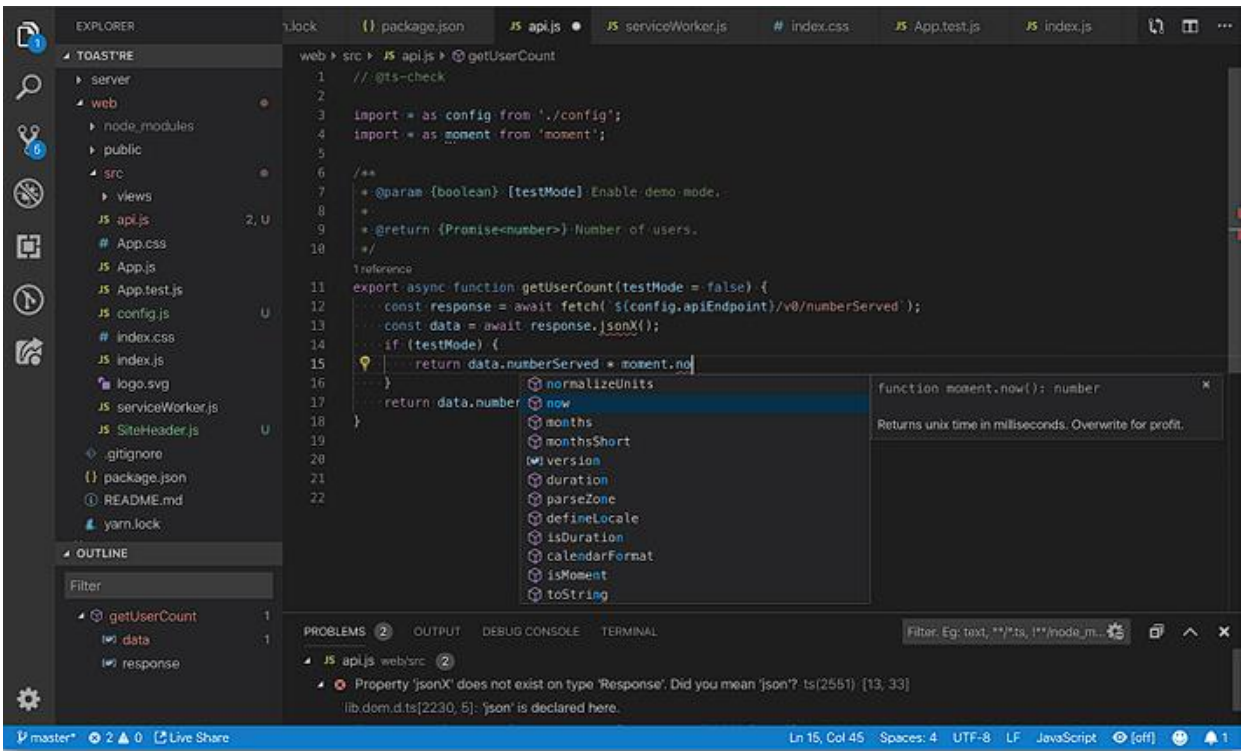


Рисунок 2.2 – Зображення одночасно відкритих каталогів

2.1.2. Магазин розширень в Visual Studio Code

Visual Studio Code може бути розширеним за допомогою інших розширень, наданих через центральне сховище (Рисунок 2.3). Туди входять доповнення та підтримка різних мов. Цікавою особливістю є можливість самостійно розробляти розширення, які додадуть в свою чергу підтримку нових мов, тем і нових налаштувань, постійно робити аналіз коду та додавати підказку при написанні коду за допомогою протоколу мовного сервера. Тим не менш, навіть якщо сам VSCode спочатку не підтримує жодних мов програмування, його підтримку можна легко додати через меню розширення. Там ви можете знайти багато корисних розширень, і навіть якщо у вас немає того, що вам потрібно, ви можете написати таке ж розширення самостійно і додати його до каталогу розширень VSCode. Крім того, це розширення може бути успішним, і ваша утиліта зробить вас популярним серед інших програмістів.

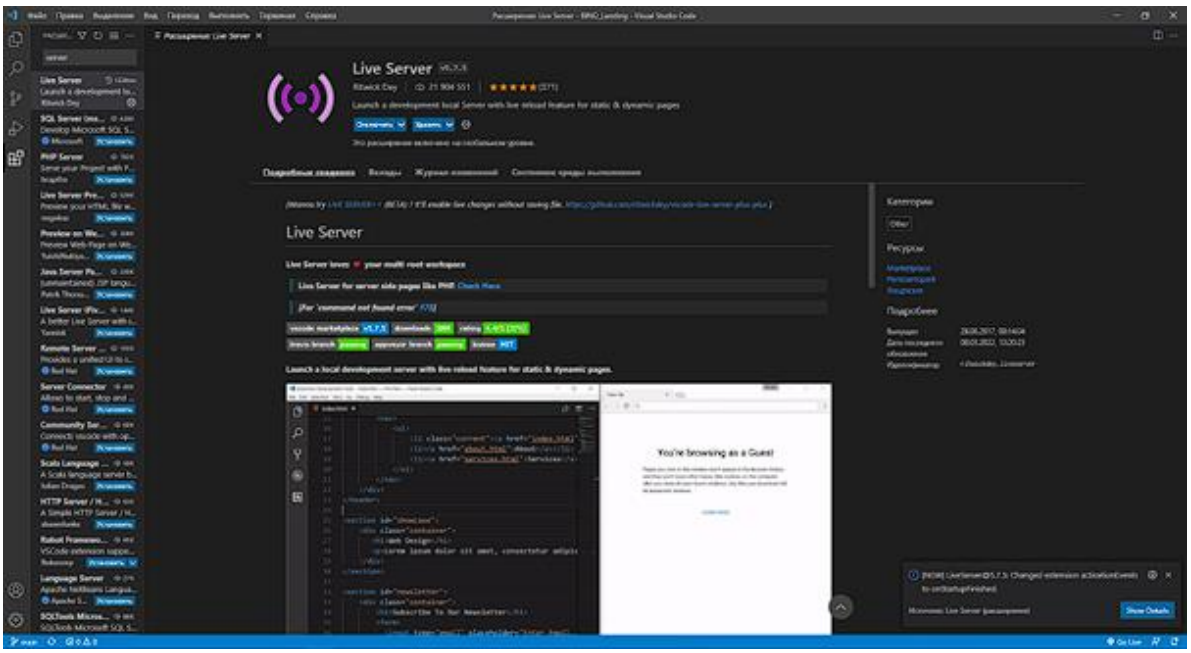


Рисунок 2.3 – Зображення безкоштовного магазину розширень

2.1.3. Контроль версій в Visual Studio Code через GIT

Контроль версій — це влаштована функція Visual Studio Code. В ній є окрема вкладку в рядку меню, де можна отримати доступ до налаштувань контролювання версії і переглядати зміни, що були зроблені в поточному проекті. Щоб користуватись цією функцією, потрібно прив’язати VSCode з будь-якою системою контролю версій, що підтримується (Git, Bitbucket, Apache Subversion, Perforce тощо). Це дозволить створити репозиторії та робити запити push і pull прямо з Visual Studio Code.

Visual Studio Code також включає кілька розширень FTP, які дозволяють використовувати програмне забезпечення безкоштовно для веб-розробки. Код можна синхронізувати між редактором і сервером без завантаження додаткового програмного забезпечення.

Нижче наведено приклад контролю версій як порівняння одного і того ж файлу, але з різницею в коді (Рисунок 2.4) і як виглядає розгалужене з’єднання (Рисунок 2.5):

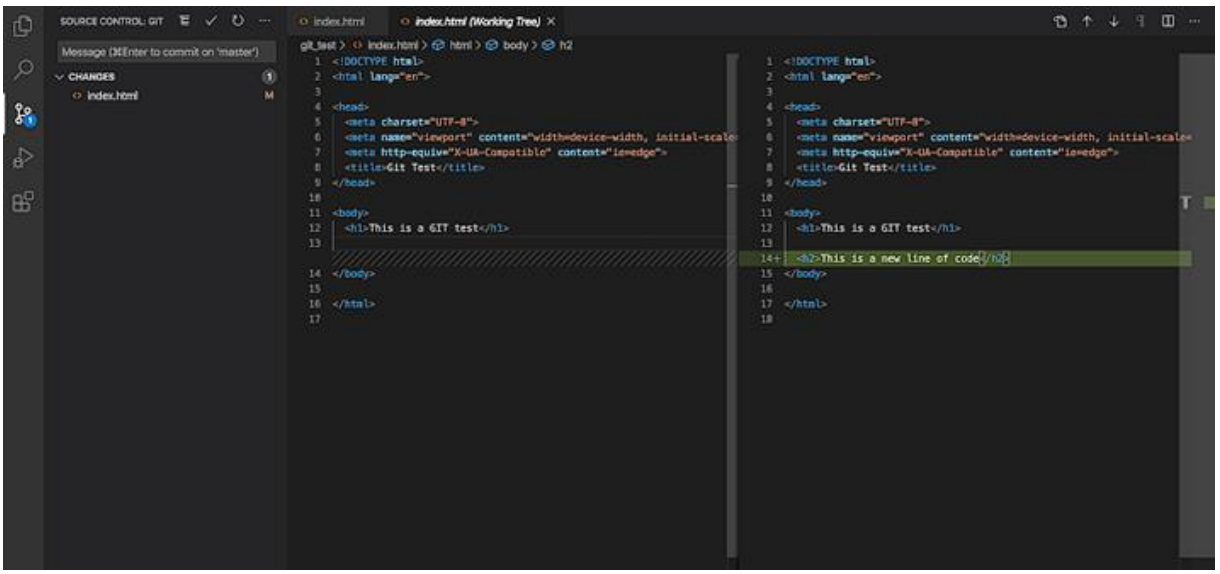


Рисунок 2.4 – Зображення різниці між файлами

Рисунок 2.5 – Зображення підключення до вітки GITa через VSCode

2.1.4. Популярність Visual Studio Code

У опитуванні Stack Overflow 2016 року Visual Studio Code зайняв 13-е місце серед найпопулярніших інструментів розробки, і лише 7% із 47 000 респондентів користувалися ним. Однак через два роки Visual Studio Code посів перше місце: ним скористалися 35% із 75 000 респондентів. В опитуванні розробників 2019 року Visual Studio Code також посів перше місце, оскільки ним користуються 50% із 87 000 респондентів. В Опитуванні розробників 2021 року Visual Studio Code продовжує займати перше місце, його використовують 70% із 82 000 респондентів.

Одним словом, VSCode з роками завоював свою популярність цілком заслужено, адже з року в рік постійно додаються нові можливості і видаляються баги (проблеми в коді самої програми).

2.2. Опис мови розмітки гіпертексту HTML

HTML (Hypertext Markup Language) — це стандартизована мова розмітки документів для перегляду веб-сторінок у браузері. Веб-браузер отримує HTML-документ із сервера через HTTP/HTTPS або відкриває його з локального диска та інтерпретує код як інтерфейс, який буде відображатися на екрані монітора.

Елементи HTML є будівельними блоками сторінок HTML. За допомогою структури HTML зображення та інші об’єкти (наприклад, інтерактивні форми) можуть бути вбудовані в скопійовані сторінки. HTML надає інструменти для створення структурованих документів, які вказують на структурну семантику тексту, наприклад заголовки, абзаци, списки, посилання, цитати та інші елементи. Елементи HTML розділені тегами, записаними в кутових дужках. Такі теги, як <img /> або <input />, безпосередньо відображають вміст сторінки. Інші теги, такі як <p>, оточують і надають інформацію про текст, а також можуть включати інші теги як дочірні елементи. Замість відображення тегів HTML браузери використовують їх для інтерпретації вмісту сторінки.

HTML можна вбудовувати в програми, написані мовами сценаріїв, такими як JavaScript, які впливають на поведінку та вміст веб-сторінок. Увімкнення CSS визначає зовнішній вигляд і макет вмісту. Консорціум World Wide Web Consortium (W3C), який підтримує стандарти HTML і CSS, заохочує використання CSS замість явного представлення HTML з 1997 року.

HTML реалізує такі інструменти:

- створює структурований документ, що вказує на структурний склад тексту: заголовки, абзаци, списки, таблиці, цитати тощо;
- отримання інформації зі всесвітньої мережі за допомогою гіперпосилань;
- створення інтерактивних форм;
- Включіть в текст зображення, звук, відео та інші об'єкти.

2.2.1. Різниця версій HTML

HTML5 – це наступна версія HTML. До робочої групи HTML5 входять AOL, Apple, Google, IBM, Microsoft, Mozilla, Nokia, Opera та сотні інших виробників.

Версії є дещо заплутаними, оскільки є дві окремі групи розробників, WHATWG і W3C.

Прийнявши специфікацію HTML, WHATWG відмовився від принципу «версії» на користь «безперервного розвитку». Це рішення було прийнято як спроба прискорити впровадження стандарту, тобто розробникам веб-браузерів не потрібно чекати офіційної затвердженої версії специфікації (специфікація стає рекомендаційною), тепер вони можуть реалізувати деякі частини специфікації. Тому, згідно з WHATWG, в Evolving є лише одна специфікація - HTML.

Дві групи працюють разом: WHATWG пише специфікації в моделі «живих стандартів», а W3C приймає ці специфікації як «знімки» та реалізує їх у чітких версіях своїх специфікацій. W3C працює набагато повільніше, тому що він повинен обслуговувати ширше коло користувачів, ніж лише веб-браузери.

28 жовтня 2014 року консорціум W3C оголосив про доступність набору специфікацій HTML5 зі статусом рекомендації. Цікаво, що в такому вигляді специфікація HTML 5.0 була сформована два роки тому, після чого робота зосередилася на тестуванні та оцінці сумісності доступних реалізацій. На момент стандартизації HTML5 вже давно став стандартом де-факто і широко використовувався у веб-додатках. Фактична ратифікація стандарту тільки що офіційно завершила просування HTML5 і підтвердила універсальність і правильність його впровадження.

Як приклад різниці у версіях показано на зображенні на рис. 2.6.

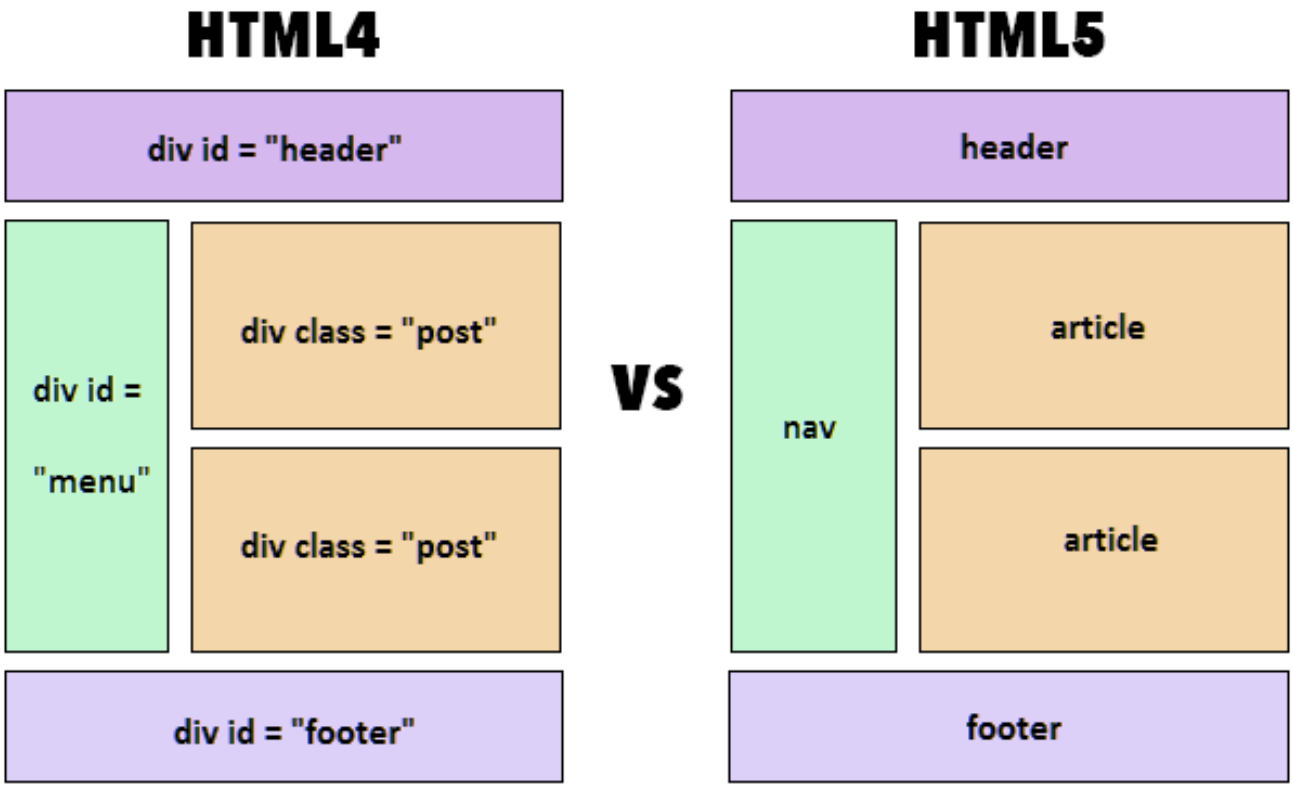


Рисунок 2.6 – Різниця специфікацій HTML4 та HTML5

Специфікації HTML5 не обмежуються розміткою і включають ряд веб-технологій, які разом утворюють відкриту веб-платформу - програмне середовище для кросплатформних додатків, які можуть взаємодіяти з апаратним забезпеченням, і які підтримують інструменти для роботи з відео, графікою та анімацією, забезпечуючи передові можливості мережі.

2.2.2. Основні переваги версії HTML5

Серед нововведень HTML5 можна виділити такі елементи (Рисунок – 2.7): Заголовок - заголовок сторінки; Розділ - великий блок сторінки; Нижній колонтитул - нижня частина сторінки; meta charset = "UTF-8" - оновлення кодування сторінки; Nav - навігація по сайту; Aside - додатковий вміст у вигляді бічної панелі; Стаття - стаття, переважна більшість змісту. Підтримка браузерами версії HTML5 (від останньої версії) показано на рис. 2.7.

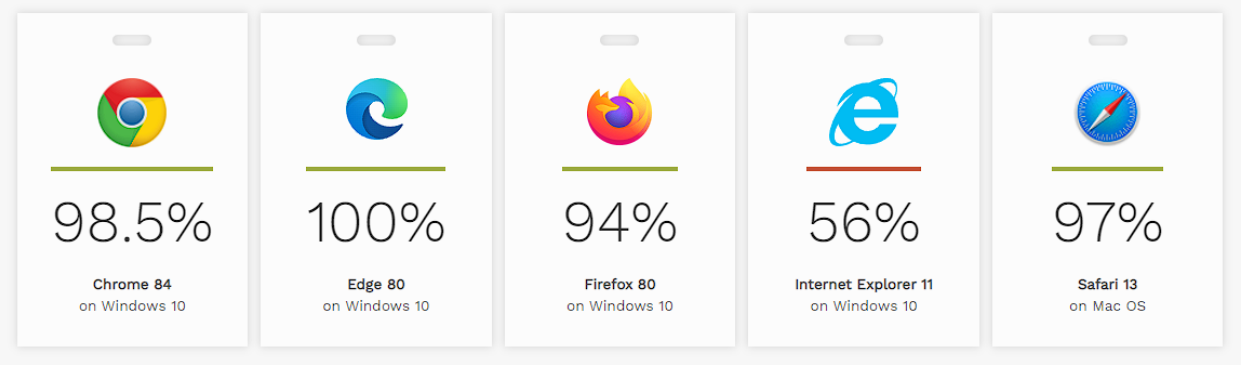


Рисунок 2.7 –Підтримка браузерами версії HTML5

HTML5 допомагає пошуковим системам знаходити цінну інформацію на сайті.

Наприклад: <article> показує пошуковим системам цінний вміст на сайті, який є важливішим за те, що вказано в <footer>. Ці теги дозволяють легко знаходити важливі фрагменти вмісту на сторінках ресурсів.

HTML5 дозволяє машинам «читати» зображення та анімацію. Раніше такого не було.

Список переваг:

- Використовуючи більш простий тип коду, наприклад, div замінюється більш досконалими елементами.

- Дизайнерські рішення, що дозволяють прийняти індивідуальне рішення для сайту. Користувальницький інтерфейс також був покращений, зробивши його зрозумілішим і функціональнішим. Форми можна перевірити за допомогою традиційних інструментів розмітки HTML.

- Ви можете використовувати нові поля введення для різних цілей, наприклад для пошуку.

— Нова розширена семантика HTML5 дозволяє швидко і легко розрізняти нижній колонтитул, заголовки, панель навігації. Для цього використовуйте нещодавно розроблені теги для спрощення присвоєння основних елементів у розмітці.

- Елементи HTML5 роблять семантичне ядро сторінок більш помітним. Для цього використовуйте стандартні коди.

- Зручне використання. З HTML5 ви можете використовувати нові елементи, зокрема <canvas>, <audio>, <video>. Покращено для інтеграції з SVG. Тепер ви можете використовувати плагіни, наприклад Flash.

- Установка музики, схем, відео та картинок може здійснюватися без використання сторонніх програм.

— Покращена структура документа з використанням нових тегів: <header>, <section> і <article>.

- Сприйняття макета сторінки: вихідний код став простим і зрозумілим.

- Розроблені комунікаційні дошки, вікі та drag-n-drops для покращення клієнтської частини програми.

— Значно збільшена швидкість відповіді сторінки (за допомогою localStorage і sessionStorage, які частково замінили файли cookie).

- Використовується API Geolocation, а отримані дані можна використовувати в програмах.

- HTML5 підтримує MathML (математична розмітка для формул) і SVG (формат зображення, масштабована векторна графіка).

HTML5 допомагає пошуковим системам знаходити цінну інформацію на сайті.

HTML5 дозволяє машинам «читати» зображення та анімацію. Раніше такого не було.

HTML5 підтримується популярними і широко використовуваними мобільними пристроями. Також використання цієї розмітки не вимагає придбання ліцензії, що, в свою чергу, дуже зручно і вигідно. Загрози безпеці проекту немає, оскільки в коді програми немає «прихованих» скриптів і SQL-запитів. Можна легко скласти набір підприємств для внутрішньої та зовнішньої оптимізації сайту, це дозволяє оптимізувати сайт в пошукових системах для відвідувачів, що призводить до збільшення відвідуваності та потенційних клієнтів.

2.3. Опис мови стилю сторінок CSS

CSS (каскадні таблиці стилів) — це спеціальна мова стилів сторінки, яка використовується для опису її зовнішнього вигляду. Самі сторінки написані мовою розмітки.

CSS є основною технологією всесвітньої мережі, поряд з HTML і JavaScript. CSS найчастіше використовується для візуалізації сторінок, написаних на HTML і XHTML, але CSS можна застосувати до інших типів документів XML.

Специфікація CSS була створена та розроблена Консорціумом World Wide Web.

CSS має різні рівні та профілі. Наступний рівень CSS базується на попередньому, додаючи нові функції або розширюючи існуючі. Рівні представлені як CSS1, CSS2 і CSS3. Профіль — це набір правил CSS з одним або кількома рівнями, створеними для певного типу пристрою чи інтерфейсу. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

CSS (каскадний або блоковий макет) замінив табличне розташування веб-сторінок. Основною перевагою блочного макета є поділ вмісту сторінки (даних) та його візуальне представлення.

2.3.1. Загальний огляд CSS

Веб-розробники та відвідувачі використовують CSS для визначення кольорів, шрифтів, макета та інших аспектів зовнішнього вигляду сторінки. Однією з головних переваг є можливість відокремити вміст сторінки (або вміст, вміст, зазвичай HTML, XML або подібну мову розмітки) від типу документа (описаного в CSS).

Такий поділ покращує сприйняття та доступність контенту, забезпечує більшу гнучкість та контроль над відображенням вмісту в різних контекстах, робить контент більш структурованим і простішим, усуває дублювання тощо.

CSS також дозволяє адаптувати вміст до різних умов відображення (на екранах моніторів, мобільних пристроях, друку, екранах телевізорів, пристроях, які підтримують Брайлів або голосові браузери тощо).

Один і той самий документ HTML або XML може виглядати по-різному залежно від використовуваного CSS. Стиль відображеної сторінки може бути:

- Авторський стиль (інформація надана автором сторінки):

• Зовнішні таблиці стилів, як правило, це окремі файли або файли .css

• Внутрішні таблиці стилів, що містяться в документі або блоці

• Стилі для окремих елементів

- Стилі користувача

• Визначений користувачем локальний файл .css для сторінки та зазначений у налаштуваннях браузера (наприклад, Opera)

- Стиль браузера

• Стилі браузера за замовчуванням, такі як стилі за замовчуванням, визначені браузером проекту, використовуються, коли немає або неповна інформація про стилі проекту.

Стандарти CSS визначають порядок та область застосування стилів, тобто порядок та елементи, у яких застосовуються стилі. Таким чином, каскадний принцип використовується, коли елемент задається лише з інформацією про стилі, які змінилися або не визначені більш загальним стилем.

Сайти виглядали би зовсім по іншому, для доказу цього приведено Рис. 2.8.

Рисунок 2.8 – Вигляд сайту в браузері без та з CSS

2.3.2. Переваги CSS

• Інформація про стилі для всього сайту або його частин може міститися у файлі .css, що дозволяє швидко змінювати дизайн і презентацію сторінок;

• Різна інформація про стилі для різних типів користувачів: наприклад, великі шрифти, стилі друку сторінок, мобільні стилі для користувачів із вадами зору;

• Сторінки зменшуються в розмірі та стають більш організованими, оскільки інформація про стилі відокремлюється від тексту, і існують певні правила, на основі яких будуються сторінки;

• Прискорює завантаження сторінки та зменшує обсяг переданої інформації, навантаження на сервер і канал передачі. Це пов’язано з тим, що сучасні браузери можуть кешувати (запам’ятати) інформацію про стиль і використовувати її для всіх сторінок замість того, щоб завантажувати її для кожної сторінки.

2.3.3. Синтаксис CSS

CSS має відносно простий синтаксис і використовує кілька англійських слів, щоб назвати різні компоненти стилю.

Стиль складається з низки правил. Кожне правило має один або кілька селекторів і рекламний блок. Блок визначення складається зі списку властивостей, укладених у фігурні дужки.

Властивості в списку виконуються у вигляді імені властивості, двокрапки (:), значення, крапки з комою (;).

Для спрощення розуміння наведено Рис. 2.9.

Рисунок 2.9 – Пояснення синтаксису CSS

2.4. Опис мови програмування JavaScript

JavaScript (надалі JS) — це динамічна, об'єктно-орієнтована мова програмування прототипів. Реалізація стандарту ECMAScript. Найчастіше він використовується для створення веб-скриптів, які дозволяють клієнтам (пристроям кінцевих користувачів) взаємодіяти з користувачами, керувати браузерами, асинхронно спілкуватися з серверами та змінювати структуру та зовнішній вигляд веб-сторінок.

JavaScript класифікується як прототип (підмножина об'єктно-

орієнтованих), мова сценаріїв з динамічним типом. Окрім прототипів, JavaScript частково підтримує інші парадигми програмування (імперативні та частково функціональні) та деякі пов'язані з ними архітектурні особливості, зокрема: динамічне та слабке введення, автоматичне керування пам'яттю, спадкування прототипів, функції як об'єкти першого класу. Функції також записані на рисунку 2.10.

```
const car1 = {
  maker: 'Ford',
  model: 'Fiesta',
  drive() {
    console.log(`Driving a ${this.maker} ${this.model} car!`)
  }
}
const anotherCar = {
  maker: 'Audi',
  model: 'A4'
}
car1.drive.bind(anotherCar)()
//Driving a Audi A4 car!

const car2 = {
  maker: 'Ford',
  model: 'Fiesta'
}
const drive = function(kmh) {
  console.log(`Driving a ${this.maker} ${this.model} car at ${kmh} km/h!`)
}
drive.call(car2, 100)
//Driving a Ford Fiesta car at 100 km/h!
drive.apply(car2, [100])
//Driving a Ford Fiesta car at 100 km/h!
```

Рисунок 2.10 – Приклад написання коду JS

2.4.1. Застосування JavaScript

JavaScript застосовують для:

- Написання сценаріїв веб-сторінок, щоб зробити їх інтерактивними
- створення односторінкових і прогресивних веб-додатків (React, AngularJS, Vue.js)
- програмування Бекенду (на сервері) (Node.js (Express.js))
- стаціонарні програми (Електрон)

- мобільні додатки (React Native)

- Скрипти в програмах (наприклад, в програмах з Adobe Creative Suite або Apache JMeter)

- всередині документів PDF тощо.

Незважаючи на схожі назви, Java і JavaScript є двома різними мовами з різною семантикою, хоча вони мають схожу функціональність у стандартній бібліотеці та умовах імен. Синтаксис обох мов успадковано від C, але семантика і дизайн JavaScript є результатом впливу Self і Scheme. Ілюстративні місця використання JavaScript показані на рис. 2.11.

Рисунок 2.11 – Місця використання JS

2.4.2. Опис JavaScript бібліотеки React

React (колишні назви: React.js, ReactJS) — це відкрита бібліотека JavaScript для створення інтерфейсів користувача, призначена для вирішення проблеми оновлення частин вмісту веб-сторінки при розробці односторінкових додатків. Розроблено Facebook, Instagram та спільнотою індивідуальних розробників. React дозволяє розробникам створювати великі веб-додатки, які використовують дані, які змінюються з часом, без

необхідності перезавантажувати сторінки. Він має на меті бути швидким, простим і масштабованим. React обробляє лише інтерфейс користувача в програмі. Це відповідає представленню в шаблонах Model-View-Controller (MVC), які можна використовувати разом з іншими бібліотеками JavaScript або більшими платформами MVC, такими як AngularJS. Його також можна використовувати з додатками React, щоб доглядати за частинами без інтерфейсу користувача для створення веб-додатків. Як бібліотека інтерфейсу користувача, React найчастіше використовується в поєднанні з іншими бібліотеками, такими як Redux.

Рисунок 2.12 – Логотипи бібліотек React & Redux

2.4.3. Особливості бібліотеки React

Одностороння передача даних

Атрибути передаються компоненту візуалізації як атрибути тегу html. Компонент не може безпосередньо змінювати властивості, передані йому, але їх можна змінити за допомогою функцій зворотного виклику. Цей механізм називається «властивості вниз, події вгору».

віртуальний DOM

React підтримує віртуальну DOM і не покладається лише на DOM браузера. Це дозволяє бібліотеці визначити, які частини DOM змінилися в

порівнянні з відмінностями між збереженими віртуальними версіями DOM, а отже, як найкраще оновити DOM браузера. Отже, програміст використовує сторінку, вважаючи, що вона повністю оновлена, але бібліотека вирішує, які компоненти сторінки оновити.

JSX

Компоненти React зазвичай пишуться на JSX. Код, написаний на JSX, компілюється у виклики методів бібліотеки React. Розробники також можуть писати код на чистому JavaScript. JSX подібний до XHP, іншої мови, створеної Facebook для розширення PHP.

Більше, ніж просто візуалізація HTML у браузері

React призначений не тільки для відтворення HTML у браузері. Наприклад, Facebook реплікує анімаційну графіку в тегу <canvas>, а Netflix і PayPal використовують ізоморфне завантаження для відтворення одного і того ж HTML на сервері та клієнті.

методи життєвого циклу

Методи життєвого циклу — це різні методи, вбудовані в ReactJS. Вони дозволяють розробникам працювати з даними на різних етапах життєвого циклу React. приклад:

shouldComponentUpdate – це метод життєвого циклу, який повідомляє JS , що потрібно оновлювати компонент за допомогою логічних змінних.

componentWillMount – це метод ЖЦ, який повідомляє JS, що потрібно налаштувати деякі дані перед монтуванням компонента (вставленням його у віртуальний DOM).

componentDidMount – це метод ЖЦ, який схожий на компонент WillMount, за винятком того, що він буде виконуватись після методу візуалізації та може бути використаний для додавання нових вхідних даних JSON, а також для визначення властивостей і стану.

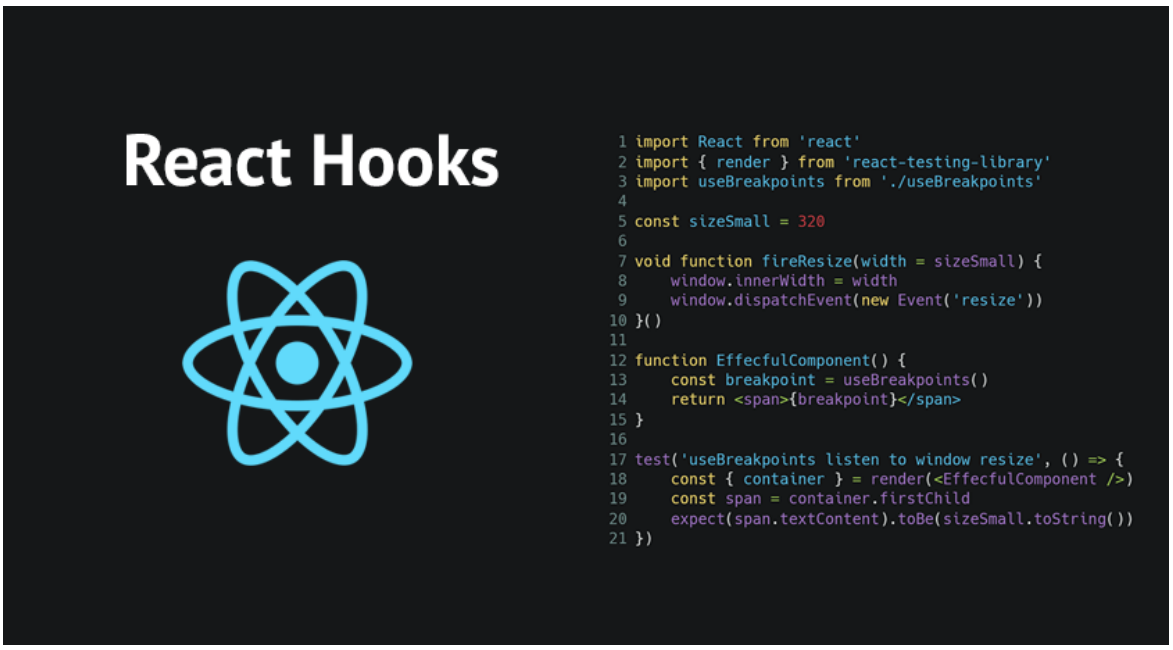


Рисунок 2.13 – Приклад коду на React

render є найважливішим методом життєвого циклу, потрібним будь-якому компоненту. Метод візуалізації — це те, що підключається до JSX і відображає власний JSX.

2.5. Опис вебдодатку

Веб-додаток (іноді веб-додаток) — це розподілена програма, де клієнт — браузер, а сервер — веб-сервер. Браузер може бути реалізацією так званого тонкого клієнта - логіка програми зосереджена на сервері, а функція браузера в основному полягає у відображенні мережевої інформації, завантаженої з сервера, і повернення даних користувача. Однією з переваг такого підходу є те, що клієнт не прив’язаний до конкретної операційної системи користувача, тому веб-додаток є кросплатформним сервісом. Завдяки такій універсальності та відносній простоті розробки веб-додатки стали широко популярними наприкінці 1990-х і на початку 2000-х років.

Рисунок 2.14 – Схематичне зображення, що вебдодаток вміщує в собі багато чого

2.5.1. Технічні особливості вебдодатку

Істотною перевагою створення веб-додатків для підтримки стандартних функцій браузера є те, що ці функції повинні виконуватися незалежно від операційної системи клієнта. Замість того, щоб писати різні версії для Microsoft Windows, Mac OS X, GNU/Linux та інших операційних систем, створіть програму один раз для випадково обраної платформи та розгорніть на ній. Однак різні реалізації HTML, CSS, DOM та інших специфікацій браузера можуть викликати проблеми з розробкою та підтримкою веб-додатків. Крім того, здатність користувача налаштовувати багато параметрів браузера (наприклад, розмір шрифту, кольори, вимкнення підтримки сценаріїв) може заважати нормальній роботі програми.

На початку 2000-х років був популярний інший (менш універсальний)

підхід, який використовував аплети Adobe Flash або Java для повної або часткової реалізації інтерфейсу користувача. Ці технології надали програмісту більше контролю над інтерфейсом і змогли обійти багато невідповідностей у конфігураціях браузера (хоча несумісність між реалізацією клієнта Java або Flash спричиняла різні ускладнення). Станом на 2020 рік аплети Java та технологія Flash практично застаріли.

Рисунок 2.15 – Схематичне зображення принципу роботи вебдодатку

Через архітектурну схожість із традиційними клієнт-серверними додатками, в чомусь «товстими» клієнтами, виникають суперечки щодо правильності включення таких систем у веб-додатки; альтернативний термін «Розширений Інтернет-додаток»).

2.5.2. Типи мобільних вебдодатків

Існує кілька підходів до створення веб-додатків для мобільних пристроїв:

• Адаптивний веб-дизайн можна використовувати для створення як зі звичайного веб-сайту, так і для односторінкового додатка, який зручно використовувати на пристроях з невеликим екраном.

• Прогресивний веб-додаток, що поєднує звичайні веб-сторінки з мобільним додатком.

• Мобільний додаток або «рідний додаток» працює безпосередньо на мобільному пристрої без веб-браузера і зазвичай не вимагає підключення до Інтернету. Зазвичай вони написані на Java для пристроїв Android або Objective-C або Swift для iOS. Зовсім недавно такі програмні фреймворки, як React Native, Flutter, Xamarin тощо, дозволяють розробляти мобільні додатки для кількох мобільних платформ, використовуючи одну мову програмування (зазвичай найпоширенішу, наприклад JavaScript), а не стандарт для мобільних додатків.

• Гібридний додаток вбудовує веб-сайт у мобільний додаток. Ви можете створювати фреймворки за допомогою гібридного програмного забезпечення, такого як Apache Cordova, Ionic або Appcelerator Titanium. Такий підхід дозволяє розробникам використовувати сучасні веб-технології при збереженні деяких переваг мобільних додатків: використання апаратного прискорення, роботи в автономному режимі, доступу до магазинів додатків тощо.

2.5.3. Архітектура вебдодатків

Веб-додаток отримує запити від клієнтів і виконує обчислення, потім створює веб-сторінку та надсилає її по мережі клієнту за допомогою HTTP. Веб-додаток може бути клієнтом інших служб, наприклад, бази даних або

стороннього веб-додатка, розташованого на іншому сервері. Яскравим прикладом веб-програми є система керування вмістом статей Вікіпедії: багато її членів можуть брати участь у створенні онлайн-енциклопедії за допомогою браузера своєї операційної системи (Microsoft Windows, GNU/Linux чи будь-якої іншої) без необхідності завантажте його. Додатковий виконуваний файл із базою даних статей.

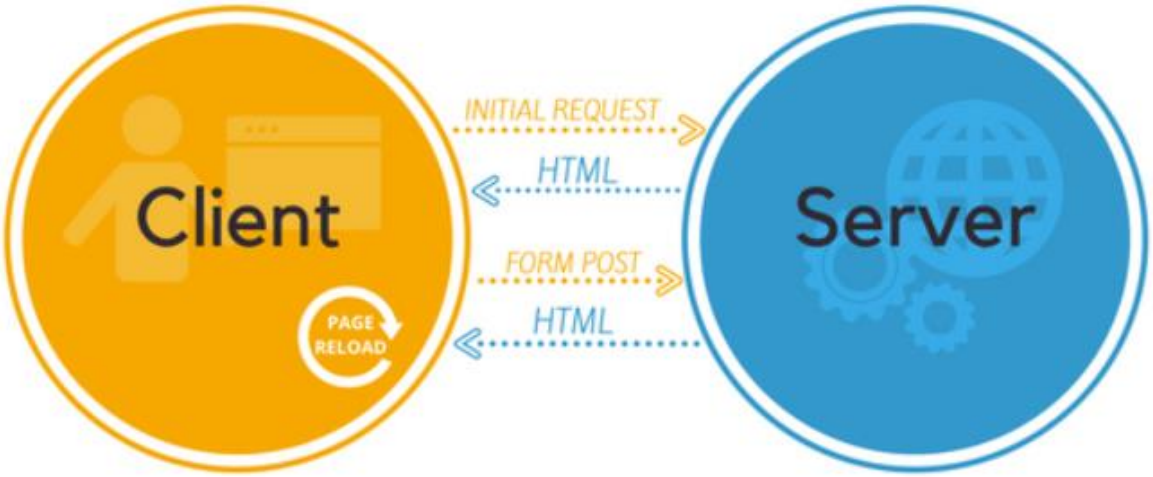


Рисунок 2.16 – Принцип роботи багатосторінкового додатку

Для покращення інтерактивності та продуктивності був розроблений новий метод розробки веб-додатків під назвою AJAX, який зараз є стандартом де-факто. При використанні Ajax сторінки веб-додатків можуть надсилати веб-запити на сервер у фоновому режимі без повного перезавантаження, а просто завантажувати необхідні дані з сервера, що значно прискорює роботу та робить її зручнішою.

Рисунок 2.17 –Принцип роботи односторінкового додатку

Для створення веб-додатків можна використовувати досить різні технології для серверів та мови програмування.

Таблиця 2.1. Використання серверних технологій та мов програмування.

Назва	Ліцензія	Вебсервер
ASP	власницька	спеціалізований
ASP.NET	власницька	спеціалізований
Java	вільна	безліч, зокрема вільних
Groovy	вільна	практично будь-який
Perl	вільна	практично будь-який
PHP	вільна	практично будь-який
Python	вільна	практично будь-який
Ruby	вільна	практично будь-який

Клієнтська частина може використовувати:

- Flash
- JavaScript
- ActiveX
- Java / JavaFX

3. СТВОРЕННЯ АДАПТИВНОГО ВЕБДОДАТКУ

3.1. Планування розробки проекту

Шаблон MVC

Для розробки гри буде використовуватися шаблон MVC.

MVC (Model-View-Controller) — це шаблон розробки програмного забезпечення, який зазвичай використовується для реалізації інтерфейсів, даних та логіки керування. Він підкреслює поділ між бізнес-логікою програмного забезпечення та відображенням. Такий «розподіл обов'язків» забезпечує кращий розподіл праці та покращене технічне обслуговування. Деякі інші шаблони дизайну засновані на MVC, наприклад MVVM (Model-View-Viewmodel), MVP (Model-View-Presenter) і MVW (Model-View-Whatever).

Три частини шаблону проектування програмного забезпечення MVC можна описати таким чином (див. Рисунок 3.1).

1. Модель: керує даними та бізнес-логікою.
2. Перегляд: керує макетом і відображенням.
3. Контролер: направляє команди на модель і частини виду.

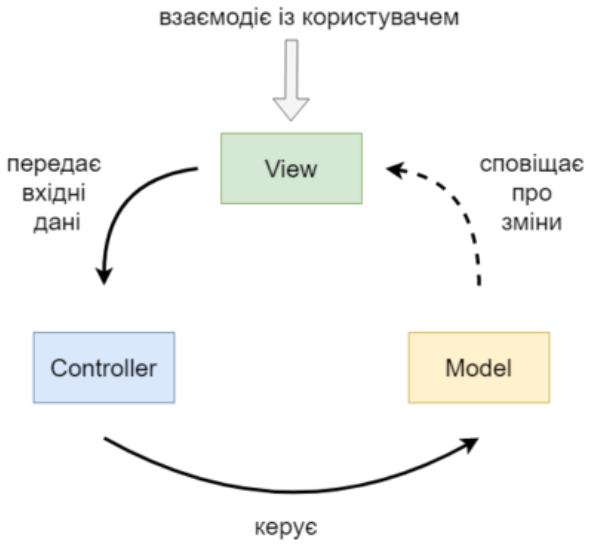


Рисунок 3.1 – Вигляд моделі MVC

- Модель

Модель визначає, які дані має містити програма. Якщо стан цих даних змінюється, модель зазвичай повідомляє про подання (тому відображення може змінюватися за потреби), а іноді й про контролер (якщо для керування оновленим поданням потрібна інша логіка).

Повертаючись до нашої програми списків покупок, модель вкаже, які дані мають містити елементи списку — продукти, ціни тощо — і які елементи списку вже існують.

- перегляд

Перегляди визначають, як відображаються дані програми.

У нашій програмі для списку покупок подання визначає, як представити список користувачеві та отримати дані для відображення моделі.

- Контролер

Контролер містить логіку, яка оновлює модель та/або презентацію у відповідь на дані, введені користувачем програми.

Наприклад, наш список покупок може містити форми введення та кнопки, які дозволяють нам додавати або видаляти товари. Ці дії вимагають оновлення моделі, тому вхідні дані надсилаються на контролер, який потім маніпулює моделлю, а модель надсилає оновлені дані до представлення.

Однак ви також можете просто оновити представлення даних, щоб відображати дані в іншому форматі, наприклад змінити порядок в алфавітному порядку або від найнижчої ціни до найвищої. У цьому випадку контролер може впоратися з цим безпосередньо без необхідності оновлення моделі.

3.2. Опис програми та її алгоритми

3.2.1. Опис програми

Функції системи представлені UML діаграма прецедентів(Див. рис. 3.2).

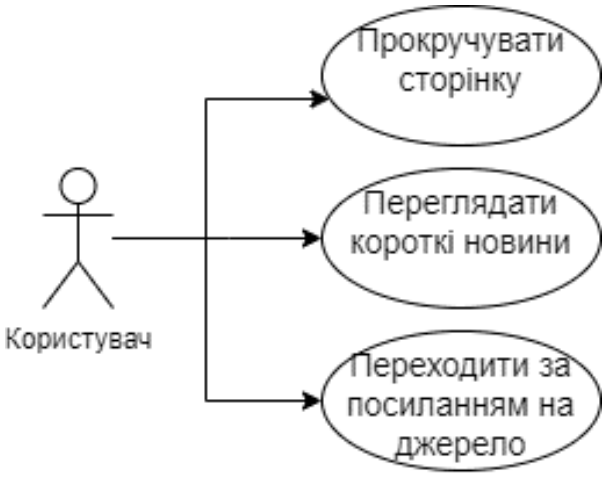


Рисунок 3.2 - UML діаграма прецедентів

Ця діаграма показує, що користувач може:

- Прокручувати (прокручувати) сайт;

- Перегляд новин у короткому форматі (тема та короткий опис);

- Якщо ви хочете відкрити джерело новин.

У процесі розробки програми потрібно чітко бачити взаємозв’язок між усіма класами, іншими словами, ієрархію класів (Див. рисунок 3.3).

Діаграма класів, яку також називають таксономією класів, — це група споріднених класів, які успадковуються для виконання подібних дій. Вершиною ієрархії може бути один базовий клас, з якого походять усі інші класи нижче, або ієрархія може мати кілька базових класів, функціональні можливості яких пізніше об’єднуються в один або кілька похідних класів. Відносини між класами можна проілюструвати як дерева, а кожне менше дерево у великій таксономії можна розглядати як ієрархію.

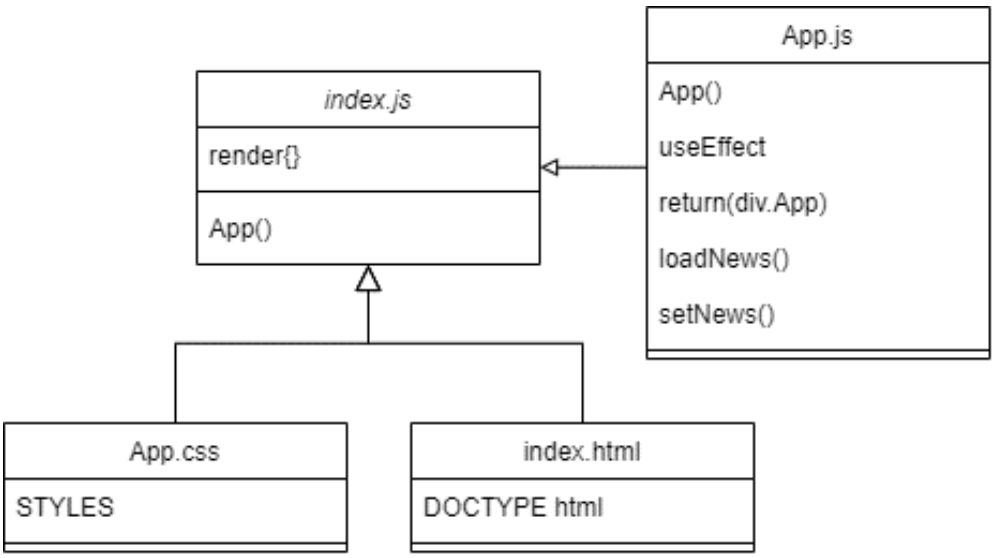


Рисунок 3.3 – Діаграма класів

В програмі є основні класи таки як

- index.js цей клас являється класом батьківськм, який відовідає за відображення всього;
- index.html клас який вже являється потомком, робить початкову структуру сайту;
- App.css – клас який відповідає за стилізацію сайту;
- App.js - клас являється батьківським класом відповідає за збирання та контроль всіх вхідних даних, перед передачею на відображення.

3.2.2. Алгоритми програми

Частина коду, показана на рис. 3.6, відповідає за рендеринг сторінки після виконання всього коду та відображення візуалізації в середині блоку, який програма знаходить за його id ‘root’.

src > JS index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import "antd/dist/antd.css";

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

Рисунок 3.4 – Функція рендеру

Далі на Рис. 3.7 зображено приклад додавання стилізації для класів з певною назвою, в даному випадку для блоків ‘code’. Font-family означає зміну стилю для шрифту текста.

```
code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

Рисунок 3.5 – Клас css ‘code’

На Рис. 3.8 Теж саме, але крім зміни стилю шрифту:

- background-color – змінює колір фону;
- min-height – задає мінімальну висоту;
- display – змінює стандартне розміщення на автоматичне;
- flex-direction – задає автоматичне направлення;
- align-items – вирівнює блоки;
- justify-content – центрує блоки;
- font-size – змінює розмір шрифту;
- color – змінює текст кольору.

```css
.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}
```

Рисунок 3.6 – Клас css ‘App-header’

На випадок, якщо у користувача в налаштуваннях браузеру відімкнутий JavaScript, то використовується команда з Рис. 3.9, щоб сказати користувачу, щоб він його ввімкнув, якщо хоче користуватись сайтом.

```
<noscript>You need to enable JavaScript to run this app.</noscript>
```

Рисунок 3.7 –Підказка ‘Щоб користуватись цим додатком, ви повинні ввімкнути JS”

Сам блок з id ‘root’ знаходиться всередині HTML документи, всередині тегу <body></body> й повністю порожній, поки не запрацює JavaScript (Рис. 3.10).

```
<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>

</body>
</html>
```

Рисунок 3.8 – Блок з id ‘root’

На Рис. 3.11 показано код, який відповідає за підключення до API новин.

```
function App() {
  const [news, setNews] = useState([]);

  useEffect(() => {
    const loadNews = async () => {
      const response = await axios.get(
        "http://newsapi.org/v2/top-headlines?country=us&category=business&apiKey=3da128da75bb4e819bb876090635ca8f"
      );
      setNews(response.data.articles);
    };
    loadNews();
  }, []);
```

Рисунок 3.9 – Підключення API новин

Наступний код віповідає за автоматичне наповнення HTML блоків картинками, текстом та кнопкою з переходом на першоджерело (Рис. 3.11).

```
  return (
    <div className="App">
      {news &&
        news.map((item, index) => {
          return (
            <Card
              key={index}
              hoverable
              style={{ width: "70%" }}
              cover={<img alt="image" src={item.urlToImage} />}
            >
              <Meta title={item.title} description={item.content} />
              <a href={item.url} target="_blank" rel="noopener noreferrer">
                <Button type="primary" style={{ marginTop: "10px" }}>
                  Read More
                </Button>
              </a>
            </Card>
          );
        })}
    </div>
  );
}
```

Рисунок 3.10 – Автоматичне наповнення блоків

3.3. Загальний вигляд програми

На Рис. 3.12 продемонстровано інтерфейс сайту на десктопному екрані.

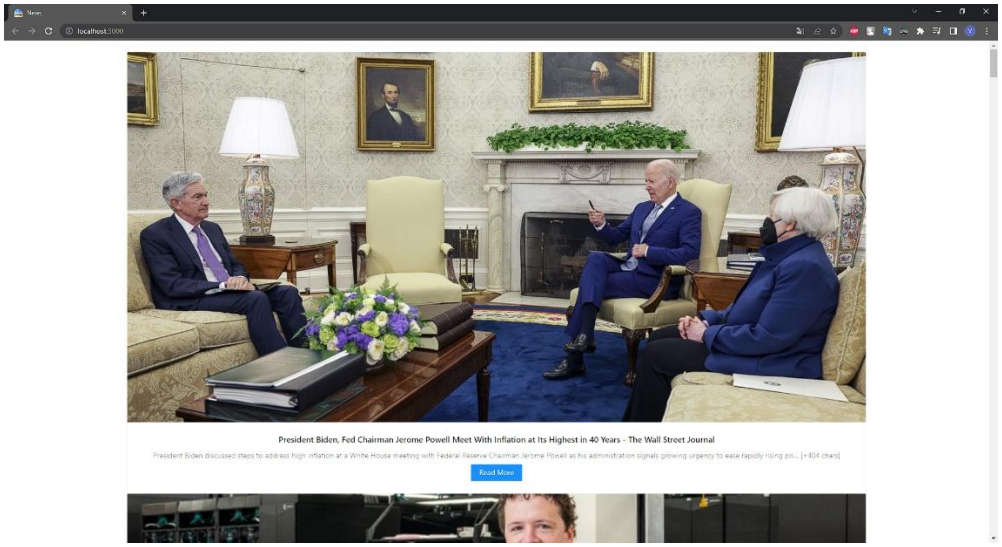


Рисунок 3.11 – Вигляд десктопної версії

На Рис. 3.13 продемонстровано інтерфейс сайту на планшетному екрані.

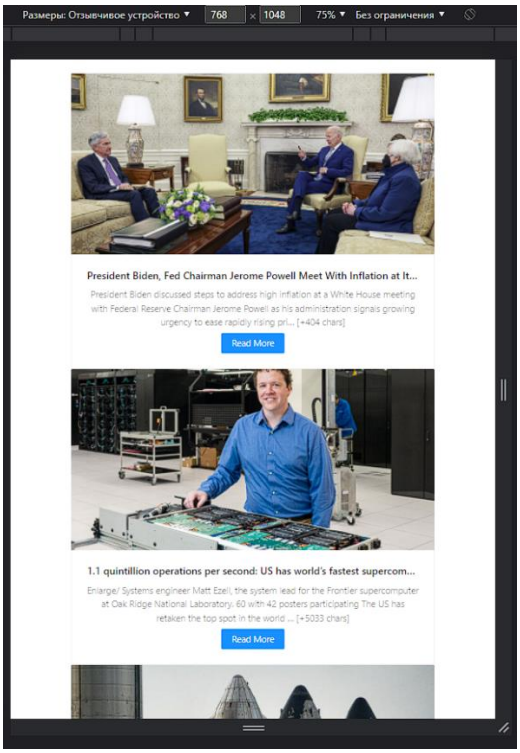


Рисунок 3.12 – Вигляд адаптації під планшет

На Рис. 3.14 продемонстровано інтерфейс сайту на мобільному екрані.

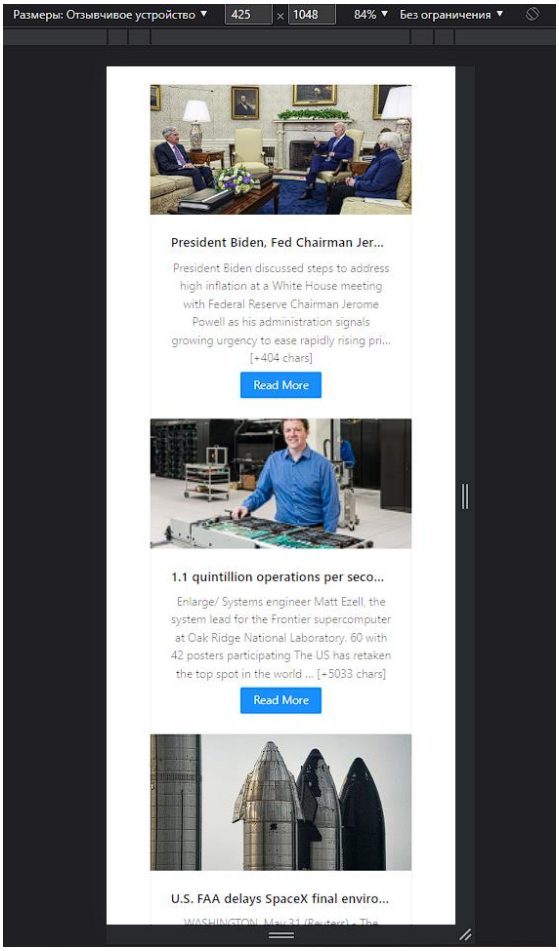


Рисунок 3.13 – Вигляд адаптації під мобільний телефон

ВИСНОВКИ

Метою даної роботи стало розроблення адаптивного вебдодатку за допомогою HTML, CSS та мови програмування JavaScript й її бібліотеки React.

Було проведено дослідження, які обгрунтовують актуальність цієї роботи та наукову новизну. А саме, який рівень зацікавленості сучасного суспільства в адаптивних вебдодатках, адже це рішення для створення сайтів було зроблено нещодавно та їх необхідність на сучасному ринку програмного забезпечення.

Була проаналізована предметна область. Проведений аналіз розробки веб сайтів. Отримані знання були використані для розробки власного алгоритму. Описано програмні засоби, які були застосовані для розробки програмного забезпечення. Визначено, що самим оптимальним рішенням є зв’язка із веб-стеку для фронтенду HTML, CSS, JavaScript та її бібліотеки React.

Підібрані відповідні технології, які максимально краще будуть сприяти розкриттю повного функціоналу програмного забезпечення, створено модулі взаємодії між технологіями, написано сторінка для автоматизації та взаємодії між сервером і вебдодатком.

Повністю був описаний процес розробки програмного забезпечення за домогою UML-діаграм та скріншотів розробленго адаптивного інтерфейсу користувача на пристроях різних розмірів. Проведено тестування за допомогою вбудованого інструменту розробника в Google Chrome, а саме зміну відображення вмісту від екрану до екрану, і показано, що дане ПЗ виконує всі функції, задля виконання яких воно було розроблене.

Робота пройшла апробацію на Науково-технічна конференція «Застосування програмного забезпечення в ІКТ». За результатами апробації опубліковано тези доповідей: «Дослідження розробки адаптивних web-додатків».

ПЕРЕЛІК ПОСИЛАНЬ

1. Створення адаптивного сайту. Адаптивне верстання – [Електронний ресурс] – Режим доступу: https://www.seotm.com/ua/services/web/adaptive_website.html

2. Адаптивний вебдизайн – [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/%D0%90%D0%B4%D0%B0%D0%BF%D1%82%D0%B8%D0%B2%D0%BD%D0%B8%D0%B9_%D0%B2%D0%B5%D0%B1%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD

3. Що таке адаптивний сайт? – [Електронний ресурс] – Режим доступу: https://brainlab.com.ua/uk/blog-uk/shho-take-adaptivnij-sajt

4. Visual Studio Code – [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Visual_Studio_Code

5. Visual Studio Code – [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Visual_Studio_Code

6. Web application vs. website: finally answered [Електронний ресурс] – Режим доступу до ресурсу:

https://www.scnsoft.com/blog/web-application-vs-website-finally-answered

7. Difference Between Web application and Website [Електронний ресурс] – Режим доступу до ресурсу:

https://www.geeksforgeeks.org/difference-between-web-application-and-website/

8. Web application (Web app) [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchsoftwarequality/definition/Web-application-Web-app

9. Використання інтеграції Git у Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: https://www.digitalocean.com/community/tutorials/how-to-use-git-integration-in-visual-studio-code-ru

10. HTML [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML

11. Visual Studio Code. Налаштування та застосування. Частина 2 [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@p1t1ch/visual-studio-code-%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0-%D0%B8-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5-%D1%87%D0%B0%D1%81%D1%82%D1%8C-2-8e4939bb1492

12. HTML5 [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML5

13. Особливості мови розмітки html5 [Електронний ресурс] – Режим доступу до ресурсу: https://hyperhost.ua/info/ru/osobennosti-yazyika-razmetki-html5

14. HTML5 Accessibility [Електронний ресурс] – Режим доступу до ресурсу: http://html5accessibility.com/

15. CSS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/CSS

16. Основи CSS [Електронний ресурс] – Режим доступу до ресурсу: http://lesson-css.blogspot.com/p/blog-page.html

17. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/JavaScript

18. ЩО МАЄ ЗНАТИ JS РОЗРОБНИК: TRAINEE, JUNIOR, MIDDLE, SENIOR РІВЕНЬ [Електронний ресурс] – Режим доступу до ресурсу: https://www.intellias.ua/blog/js-profession-guideline-by-intellias

19. React [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/React

20. Вебзастосунок [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA

21. Що таке веб-додаток? [Електронний ресурс] – Режим доступу до ресурсу: https://uk.education-wiki.com/7160560-what-is-web-application

22. Багатосторінкові та односторінкові додатки, їх переваги та недоліки [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/forums/topic/25444/

23. React Hooks — огляд можливостей нового API [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/articles/react-hooks-guide/

24. Типи адаптивних макетів [Електронний ресурс] – Режим доступу до ресурсу: https://voll.com.ua/uk/blog/adaptivnyj-dizajn-sajta-zachem-i-kogda-on-nuzhen

25. Важливість адаптивного дизайну для інтернет-магазинів [Електронний ресурс] – Режим доступу до ресурсу: https://turumburum.ua/ru/blog/vazhnost-adaptivnogo-dizayna-dlya-internet-magazinov/

26. Technology Device Ownership: 2015 [Електронний ресурс] – Режим доступу до ресурсу: https://www.pewresearch.org/internet/2015/10/29/technology-device-ownership-2015/#_ga=2.157598332.796598183.1653428824-540507356.1653428824

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА АДАПТИВНОГО WEB-ДОДАТКУ ЗА ДОПОМОГОЮ HTML, CSS ТА МОВИ ПРОГРАМУВАННЯ JAVASCRIPT

Виконав студент4 курсу

групи ПД-43
Сарданов Володимир Ігорович

Керівник роботи
к.т.н. доц. кафедри ІПЗ, Трінтіна Наталія Альбертівна

Київ – 2022

Порівняння з аналогами

	bigmir.net, ukr.net	Адаптивний вебдодаток
Легкість просування	-	+
Простота SEO	±	+
Легкість адміністрування	-	+
Простота розвитку	±	+
Простота розробки	-	+
Легкість розміщення на хостингу	+	±

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи - спрощення процесу виведення вмісту на сайті.

Об'єкт дослідження - процес динамічного виведення вмісту в адаптивному вебдодатку.

Предмет дослідження - адаптивний вебдодаток з динамічним виведенням вмісту.

ТЕХНІЧНІ ЗАВДАННЯ

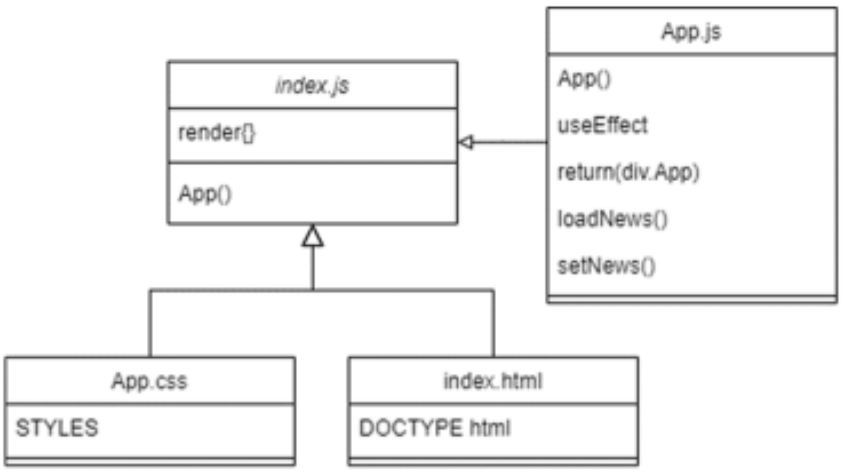
1. Вебдодаток повинен бути сумісний з основними платформами (Windows, Android, MacOS, IOS).
2. Повинен коректно відображатись на екрані з роздільною здатністю маленького смартфона до 4к екрану.
3. Повинен постійно перевіряти наявність нових даних на сервері.
4. Розробити юзер френдлі інтерфейс.

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Visual Studio Code

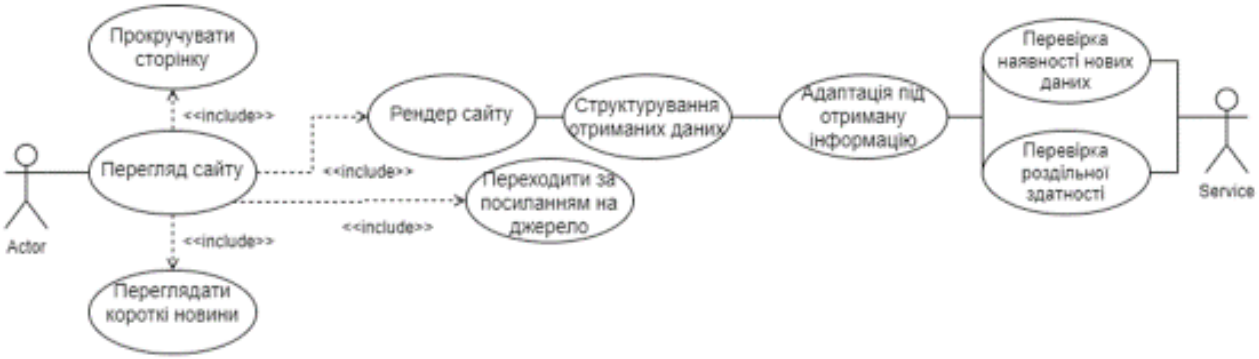
5

Діаграма класів



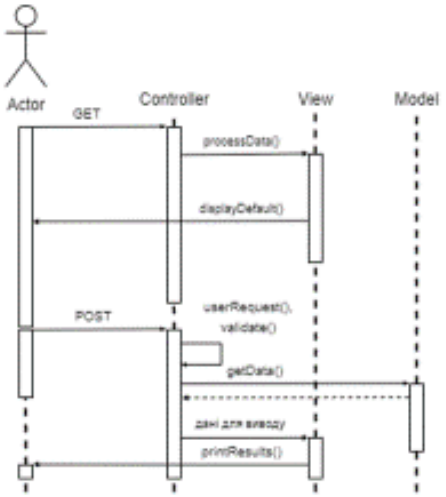
6

Схема діяльності користувача та сервісу



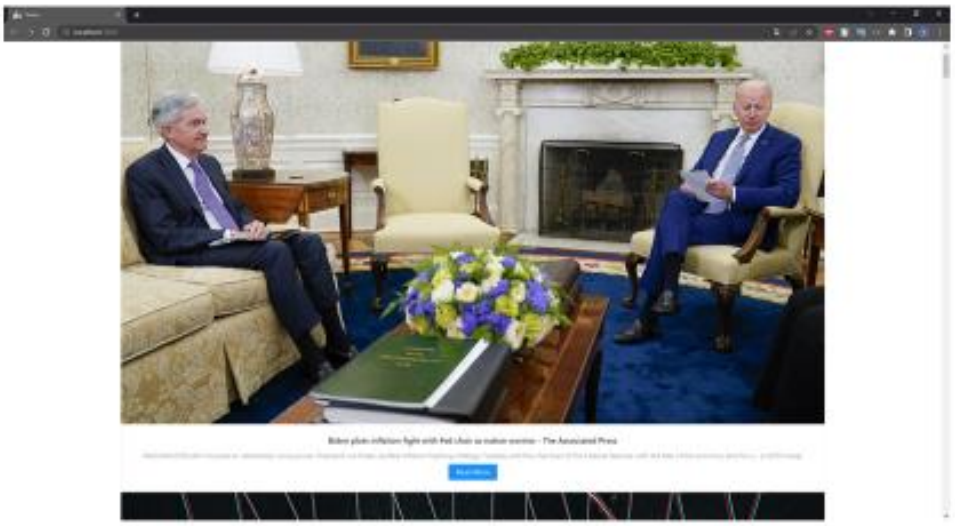
7

Схема роботи MVC в додатку



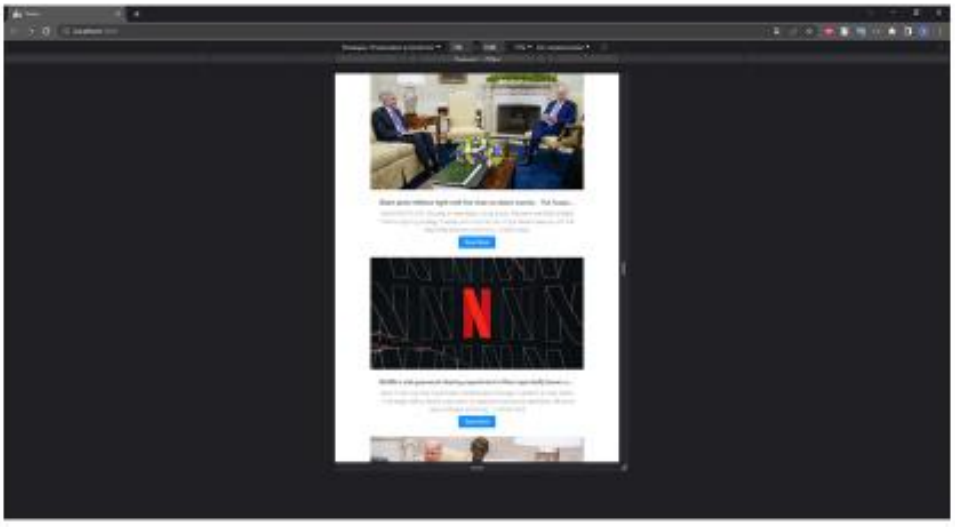
8

Робота вебдодатку (Desktop version)



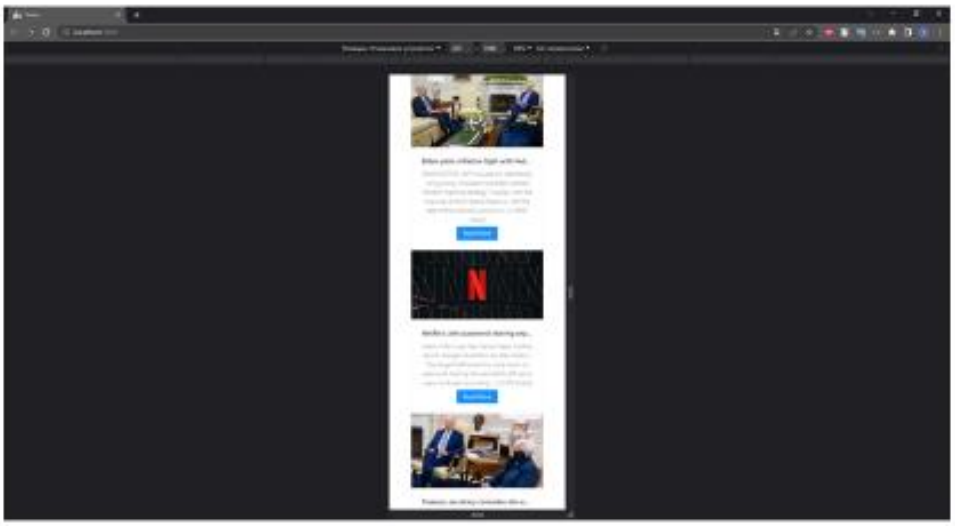
9

Робота вебдодатку (Tablet version)



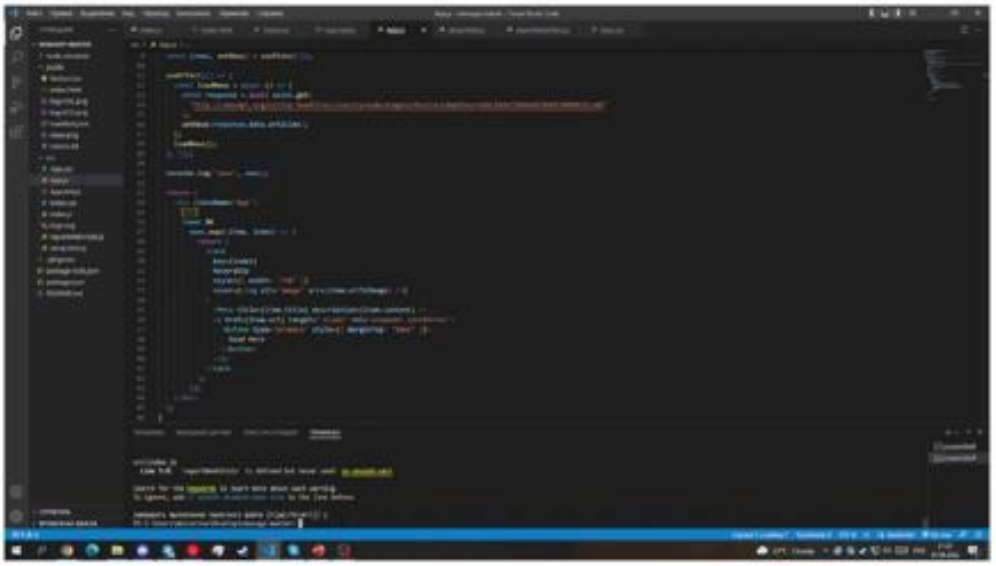
10

Робота вебдодатку (Mobile version)



11

Робота вебдодатку



12

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Сарданов В.І., Дослідження розробки адаптивних web-додатків: Матеріали науково-технічної конференції «Застосування програмного забезпечення в ІКТ». Збірник тез.\- 20.04.2022, ДУТ, м.Київ, ст. 19-22

Сарданов В.І., Дослідження впливу web-додатків на IoT: Матеріали науково-технічної конференції «Сучасний стан та перспективи розвитку IoT». Збірник тез.\- 08.04.2022, ДУТ, м.Київ

ВИСНОВКИ

1. Проаналізовано аналоги. Виявили переваги та недоліки кожного.
2. Реалізовано юзер френдлі інтерфейс.
3. Розроблено адаптивний вебдодаток, який коректно відображається на екрані з роздільною здатністю маленького смартфона до 4к екрану.
4. Вебдодаток сумісний з основними платформами (Windows, Android, MacOS, IOS).
5. Реалізована постійна перевірка на наявність нових даних на сервері.

ДЯКУЮ ЗА УВАГУ!