МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕКОНОМІКИ ТА ПІДПРИЄМНИЦТВА

КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ЕКОНОМІЧНОЇ КІБЕРНЕТИКИ

СЛОБОДЯНЮК Владислав Вікторович

РОЗРОБКА ОДНОСТОРІНКОВОГО WEB-САЙТУ ДЛЯ ПІДТРИМКИ ДІЯЛЬНОСТІ ПІДПРИЄМСТВ

Дипломна робота на здобуття освітнього ступеня «Бакалавр»
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Комп'ютерні науки»

Науковий керівник: доктор філософії з економіки, старший викладач кафедри комп'ютерних наук та економічної кібернетики Чіков Ілля Анатолійович

Вінниця – 2023

АНОТАЦІЯ

Дипломна робота виконана студентом групи КН-41 *Слободянюком Владиславом Вікторовичем*. Тема «Розробка односторінкового WEB-сайту для підтримки діяльності підприємств». Робота направлена на здобуття ступеня бакалавр за спеціальністю 122 «Комп'ютерні науки».

Загальний обсяг дипломної роботи становить 83 сторінки комп'ютерного тексту. Робота містить 6 таблиць та 75 рисунки. Список використаних джерел включає 37 найменувань, викладених на 4 сторінках

Метою дипломної роботи є розробка односторінкового WEB-сайту та його завантаження на хостинг.

Для досягнення поставленої мети, було досліджено загальні положення використання WEB-сайтів, зокрема в першому розділі було здійснено опис предметного середовища, огляд наявних аналогів та сформовано постановку задачі дипломної роботи.

В рамках другого розділу було обрано та здійснено опис програмного середовища для WEB-розробки, охарактеризовано технології створення WEB-сайтів та досліджено їх архітектуру.

В третьому розділі описано алгоритм розробки та завантаження односторінкового WEB-сайту на хостинг, зокрема було здійснено моделювання роботи WEB-сайту, наведено файлову структура проєкту та описано програмну реалізацію WEB-сайту у вигляді фрагментів лістингу програмного коду.

В результаті виконання дипломної роботи було розроблено адаптивний WEB-ресурс, який направлений на підвищення результативності діяльності підприємств у контексті просування пропонованих ними товарів та послуг.

Ключові слова: WEB-сайт, розробка, хостинг, мова розмітки, каскадна таблиця стилів, автоматизація, препроцесори.

ЗМІСТ

ВСТУП

Сьогодні переважна більшість підприємств різної специфіки функціонування починає використовувати мережу Інтернет для зберігання, обміну та обробки інформації, а також оптимізації роботи з клієнтами. Найпоширенішою формою взаємодії підприємств та користувачів у мережі є WEB-сайт – «посередник» між підприємством та його оточенням.

Сучасний онлайн-маркетинг включає в себе безліч підходів, прийомів і методів для просування товарів чи послуг. Одним із інструментів, який направлений на досягнення одразу кількох маркетингових цілей, зокрема на залучення нових користувачів, привертання уваги на свою діяльність, рекламування пропонованої продукції тощо – є посадкова сторінка або односторінковий WEB-сайт. Посадкова сторінка створюється з єдиною метою – обмежити можливості для користувача сайту таким чином, щоб опинившись на сторінці, контент на сайті ненав'язливо «вів» його до виконання конкретних дій (замовлення, реєстрація, оформлення підписки тощо).

Односторінкові WEB-сайти є базисом маркетингової політики підприємства, яке займається просування в інтернеті, або розширює свою діяльність. За допомогою посадкових сторінок, підприємство може досягти відмінних результатів відразу за декількома напрямками, зокрема підвищення впізнаваності бренду, поширення інформації, генерації потоку клієнтів. Сайт має відповідати сучасним стандартам веб-розробки, бути зручним та корисним для цільового споживача. А саме, контент сайту має допомагати споживачу, вирішувати його проблеми і задовольняти потребу в товарі чи послузі, та інформації про них.

Метою кваліфікаційної роботи є розробка та завантаження на хостинг односторінкового WEB-сайту для підтримки діяльності підприємств.

Для досягнена поставленої мети, були визначені наступні задачі:

1) проаналізувати предметну область дослідження;

2) дослідити теоретичні засади розробки WEB-сайтів;

3) обрати та описати інструменти розробки WEB-сайтів;

4) здійснити верстку WEB-сайту;

5) завантажити WEB-сайт на хостинг.

Об’єктом дослідження є односторінковий WEB-сайт, метою якого є спонукання користування до замовлення комерційних пропозицій.

Предметом дослідження є технології розробки WEB-сайтів.

Для розробки односторінкового WEB-сайту для підтримки діяльності підприємств використовувались: препроцесорна мова розмітки PUG, препроцесорна мова опису зовнішнього виду документа SASS, мова програмування JavaScript та інструмент автоматизації Gulp. У якості програмного середовища було обрано Microsoft Visual Studio Code. Для можливості завантаження сайту на хостинг використовувався FTP менеджер FileZilla.

Розроблений WEB-сайт складається із восьми екранів, серед яких головний екран, «Наші послуги», «Індивідуальні рішення», «Готові рішення», «Калькулятор», «Наші переваги», «Статистика» та футер. Окрім цього, сайт адаптований до перегляду на різних пристроях.

РОЗДІЛ 1
ЗАГАЛЬНІ ПОЛОЖЕННЯ ВИКОРИСТАННЯ WEB-САЙТІВ

1.1. Опис предметного середовища

WEB-сайт підприємства – це його інформаційний центр, у якому зібрана уся необхідна і актуальна інформація, яка дозволяє дати відвідувачам вичерпну відповідь на питання, що їх цікавлять. Зокрема, ознайомитися з діяльністю підприємства, відгуками клієнтів, дізнатися контакти, замовити зворотний дзвінок, уточнити умови доставки та повернення товарів тощо. Інакше кажучи, відвідуючи WEB-сайт ми маємо за мету отримати актуальну інформацію про проблемну область.

WEB-сайт підприємства направлений на забезпечення комунікативної (за допомогою WEB-сайту підприємство спонукає клієнта до необхідності покупки певного товару або послуги), збутової (через мережу Інтернет легко зробити замовлення товару чи послуги без необхідності витрачають свій час на офлайн-магазини) та інформаційної функції (надання інформації про підприємства, товари або послуги) [1].

У розрізі технічного аспекту WEB-сайт – це цифровий об’єкт у мережі Інтернет, який характеризується своєю унікальною Інтернет-адресою (URL-адреса) і містить у собі одну і більше WEB-сторінок, які об’єднані спільною тематикою, структурою, оформленням та доменним ім’ям. Основним елементом будь-якого WEB-сайту є WEB-сторінка, яка є окремою логічною складовою, що написана на гіпертекстовій мові розмітки HTML або XHTML і може містити гіпертекст із навігаційними гіперпосиланнями на інші WEB-сторінки.

WEB-сторінка є прикладом складеного документа оскільки вона містить дані різних типів, таких як окремі тестові документи (DOC, PDF), графічні зображення (растрові, векторні), анімацію (CSS-графіка, GIF-зображення) тощо. Враховуючи зазначене, під час створення WEB-сторінок важливо враховувати їх тип, структуру та функціональне призначення, а також специфіку WEB-сайту, до складу якого будуть входити ці сторінки.

Сьогодні WEB-сайти за технологією їх створення поділяються на статичні, динамічні та flash-сайти. Статичні сайти складаються виключно із сторінок написаних на гіпертекстовій мові HTML та гіперпосилань для навігації між ними. Особливістю таких сайтів є можливість їх завантаження для локального перегляду користувачем, однак будь-яке оновлення або доповнення контексту на сайт є неможливим без спеціального редактору коду. Разом з цим, даний тип сайтів не передбачає наявність виконання на стороні користувача скриптів або запитів до бази даних [2].

Динамічні WEB-сайти, у свою чергу, характеризуються наявністю скриптів, форм і методів, які дають можливість встановлювати зв'язок із базою даних і за запитом користувача відображати різний контекст залежно від його уподобань. Оновлення інформації на WEB-сайті із динамічною архітектурою передбачає внесення змін до бази даних, шляхом передачі і підтвердженні запиту до кінцевого серверу. При наступному завантаженні WEB-сайту, сервер надішле нову для відображення інформацію.

Flash-сайти – це підвид динамічних WEB-сайтів, однак на відміну від останніх, архітектура Flash-сайтів не передбачає оновлення даних та інших елементів шляхом опрацювання запитів серверів. Даний тип сайтів, як правило, розміщуються на сайтах дизайнерів, художників, фотографів, а тому такі сторінки привабливі, яскраві, містять багато анімації та звукових ефектів.

Окрім класифікації за технологією створення, WEB-сайти поділяються за видом представлення інформації, до яких належать, сайт-візитка, тематичний сайт, корпоративні сайти, каталог продукції (вітрина), інтернет-магазин та односторінкові WEB-сайти. Сайт-візитка – це один із найпростіших WEB-сайтів з усіх вище наведених. Такий сайт зазвичай складається із декількох сторінок і має нейтральний або навіть строгий дизайн, з основним акцентом на інформативність цільової аудиторії. Сайт візитка містить інформацію про компанію, її контакти, вид діяльності та послуги. Таким тип сайтів переважно підходить для підприємців-початківців, які лише розпочинають свою діяльність.

Тематичний сайт – це вид сайту, який представляє інформацію з однієї

конкретної теми. Такі сайти, як правило, складаються більше аніж з десяти WEB-сторінок з детальним висвітлення інформації на будь-яке питання. Даний вид сайтів є одними з найбільш відвідуваних, коли мова йде про пошук і отримання інформації з різних тем.

Корпоративні сайти призначені для великих бізнес-структур. На корпоративному сайті, крім стандартних розділів, розміщують більш розгорнуті відомості про реалізовану продукції та послуги, що надаються. Зокрема інформація про постачальників, договори про співпрацю, фінансові звітності тощо. Часто на таких WEB-ресурсах місить новинна стрічка, форма зворотного зв'язку, карта сайту тощо.

Каталог продукції (вітрина) призначена для продажу товарів чи групи товарів. Такі сайти мають вигляд структурованого каталогу продукції на яких розміщується інформація про товари або послуги, яку неможливо помістити в прайс-лист, зокрема детальний опис, фото, відеоматеріали. На інтернет-каталогах, як і на корпоративних, також може бути присутня секція з новинами, блог, відгуки тощо.

На відміну від інтернет-каталогу, де користувач може переглянути наявність всіх товарів, інтернет-магазини дають можливість зробити замовлення, вибрати варіант розрахунку, спосіб отримання замовлення та одержати рахунок на оплату. Інтернет-магазин – це один з найбільш поширених і популярних WEB-ресурсів з продажу товарів та послуг. На відмінну від інтернет-каталогу інтернет-магазини передбачають надання гарантійних зобов'язань, а також їх діяльність регламентується вимогами чинного законодавства.

Односторінковий WEB-сайт – це невеликий сайт, сконструйований та оптимізований так чином, аби клієнт, який відвідав ресурс виконав цільову дію. Розглядаючи даний тип сайтів дещо ширше, то їх функціональне призначення полягає у описі певного продукту чи послуги, зокрема у переконанні користувачів у перевагах пропонованих продуктів та необхідності у їх придбанні. Інакше кажучи – це сайт, який просуває конкретну пропозицію. Загальна структура односторінкових WEB-сайтів наведена на рис. 1.1.

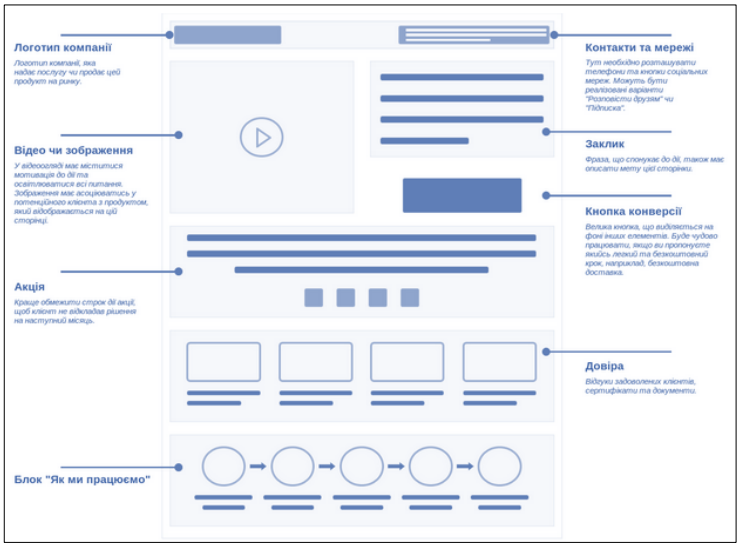


Рисунок 1.1. – **Загальна структура односторінкових WEB-сайтів**

Джерело: [2]

Як правило, односторінкові WEB-сайти складаються з однієї довгої сторінки, на якій розміщено стислу та акценту інформацію про підприємство, товар або послугу. Основною метою таких сайтів є привертання уваги клієнтів, їх спонукання до оформлення замовлення. Залежно від того, на яку цільову аудиторію орієнтується підприємство, сайт може рекламувати товар чи послугу, реєструвати користувачів на розсилку контенту, рекламувати подію тощо. Особливістю односторінкових WEB-сайтів – грамотно продумане текстове наповнення, яке зможе зацікавити людину та вмотивувати її до дій.

Згідно загальноприйнятоїї класифікації існує два типи односторінкових WEB-сайтів: 1) «Short List» – найбільш популярний тип посадкового сайту, метою якого є спонукання користувача до дій (купівля товару, реєстрація, завантаження матеріалу тощо); 2) «Long List» – багато секційні сайти, метою яких є інформування користувачів про товар чи послугу.

Ціллю розробки та використання односторінкових WEB-сайтів є спонукання користувачів до реєстрації на прослуховування вебінару, реєстрації у відвідування певного заходу, оформленні заявок на придбання послуги або продукту, завантаження електронної каталогу товару, книги або іншого документу тощо. Окрім цього, через такі сайти можна з легкістю інформувати клієнтів про нові надходження товару, про знижки, акції та розпродажі. Тобто,

головною метою односторінкових WEB-сайтів є привабити і спонукати користувача до придбання послуги або товару.

Залежно від того, яку мету переслідує підприємство – маркетинг чи продаж, виділяють транзитні та контактні односторінкові WEB-сайти. Транзитні WEB-сайти привертають користувачів до пропонованого товару чи послуги, а потім перенаправляють на сторінки, де вони можуть його придбати. Контактні, у свою чергу, направлені на збір даних користувачів, зокрема їх адреси електронних скриньок. Зазвичай це робиться через заповнення форми із пропозицією надання безкоштовної електронної книги, консультації, фрагмента відео тощо [3].

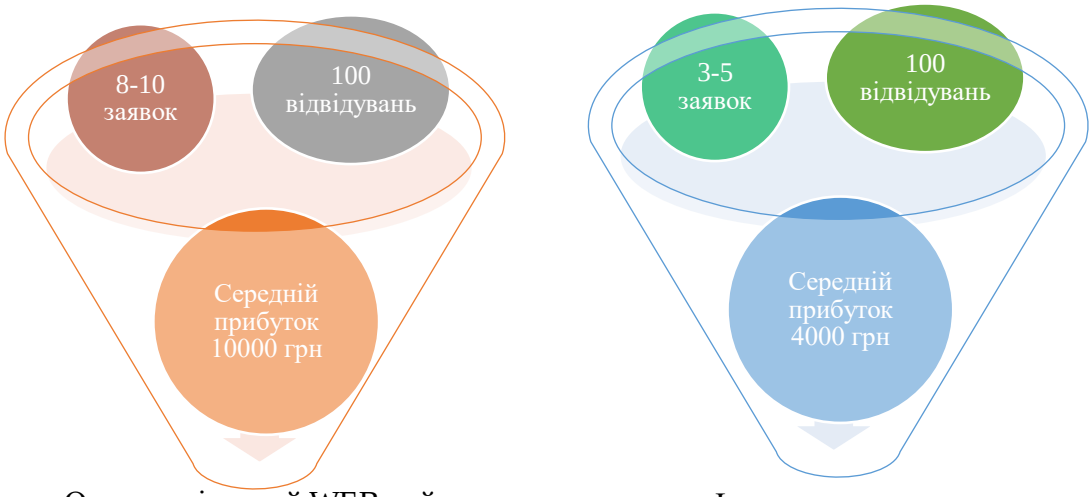
Односторінкові сайти створюють таким чином, щоб користувачі, яким невідомий продукт чи послуга, після відвідування сайту та ознайомившись з інформацію опублікованою на ньому, зробили замовлення. Як правило, для оформлення замовлення, на відміну від інтернет-магазинів, на односторінкових сайтах не має кошику чи особистого кабінету, однак бувають і виключення. Для замовлення товару чи послуги, користувач повинен заповнити певну форму із запитом щодо замовлення пропонованого продукту чи послуги. В рамках задачі поставленої в кваліфікаційній роботі, увага буде зосереджена на односторінкових WEB-сайтах [4].

В силу своєї принципової відмінності від традиційного інтернет-магазину, які мають багатоцільовий характер, односторінкові WEB-сайти концентруються увагу користувача на одній пропозиції (рис. 1.2).

Рисунок 1.2. – **Порівняльна структура односторінкових WEB-сайтів та інтернет-магазинів**

Джерело: [3, 4]

Як показали дослідження компанії HubSpot, яке спеціалізується на розгортанні хмарного програмного забезпечення для автоматизації маркетингу і продажів, підприємства, що збільшують число посадкових сторінок спостерігаються збільшення потенційних покупців на 55% [5]. При правильному підході до розробки та оформлення WEB-сайту гарантується результат у вигляді збільшення кількості завершених цільових дій. Відповідно, це принесе підприємстві високі прибутки (рис. 1.3).



Односторінковий WEB-сайт Інтернет-магазин

Рисунок 1.3. – **Порівняння воронок продажу односторінкових WEB-сайтів та Інтернет-магазину**

Джерело: [6]

Вищезазначене і робить даний тип сайтів надзвичайно потужним маркетинговим інструментом (табл. 1.1).

Таблиця 1.1

Відмінності між односторінковими WEB-сайтами та інтернет-магазинами

Односторінковий WEB-сайт	**Інтернет-магазин**
Вища конверсія за відвідування	Низька конверсія за відвідування
Інформація наводиться невеликими обсягами	Інформація наводиться у максимально можливому обсязі
Фокусує користувача на одному товарі чи послузі	Великий асортимент може негативно вплинути на початку ціль користувача
Сайти створюються у короткі терміни, а отже веде до швидкого старту продажів	Інтернет-магазин складний у розробці, потребує більше часу на старт продажів
Підключення аналітичних інструментів дозволяє визначити поведінку користувачів	Масштабність сайту не дозволяє виконувати аналіз дій користувачів

Джерело: сформовано автором на основі [1]

Односторінкові WEB-сайти рекламують один товар або конкретну товарну категорію, акцентують увагу на їх головних перевагах і особливостях. Фокусування на одному об'єкті створює умови для ефективної мотивації користувача до цільової дії – замовлення товару чи послуги. Такі сайти дають можливість підвищити «специфічність контенту», що лягає в основу зростання якості пошуку сторінки (SEO) в мережі.

Не залежно від того, який односторінковий WEB-сайт необхідно розробити і запустити – для збору контактних даних чи онлайн-продажів, існує п'ять ключових правил якісного сайту [2]:

1) «Один сайт – одна пропозиція» – основна ідея полягає у тому, що сайт повинен розроблятися і структуруватися таким чином, аби фокусувати увагу користувача виключно на одній пропозиції. Якщо сайт буде пропонувати декілька варіантів одного товару увага користувача неминуче втратиться. Якщо є необхідність рекламувати декілька продуктів, краще розробити ряд сторінок – так конверсія буде набагато вищою;

2) дизайн – в першу чергу, в дизайні не повинно нічого відволікати. Ключовий аспект у дизайні – привабливість. Сайт повинен привертати погляд користувача і змусити його переглянути сторінку;

3) контент – текст повинен бути побудований таким чином, аби донести читачу ідею пропонованого продукту чи послуги. Текст немає бути перенасиченими ключами, містити складні мовні конструкції, важким для сприйняття – все це відволіче увагу користувача;

4) заклики до дій – на сторінці обов'язково має бути декілька елементів із закликом до дії, які «підштовхнуть» користувача зробити цільову дію;

5) «секція довіри» – додатковим стимулом до цільової дії користувача є його переконання у тому, що пропонованому товару та підприємству можна довіряти. Зазначене полягає у розміщенні на сайті секції із відгуками клієнтів, які вже придбали товар чи послугу.

Створення односторінкового WEB-сайту буде доцільно тоді, коли підприємство має за мету виділити окремо свій товар чи послугу створивши

окремий сайт з іншим потоком користувачів; підприємство лише започатковує свою діяльність, потрібен швидкий запуск і швидке отримання заявок; підприємство потребує сайт з додатковою генерацією потоку клієнтів; підприємству потрібен сайт, який «змушує» користувача залишити свої контактні дані.

З огляду на проведене порівняння видів, можна навести переваги від створення і запуску односторінкових WEB-сайтів (рис. 1.4).

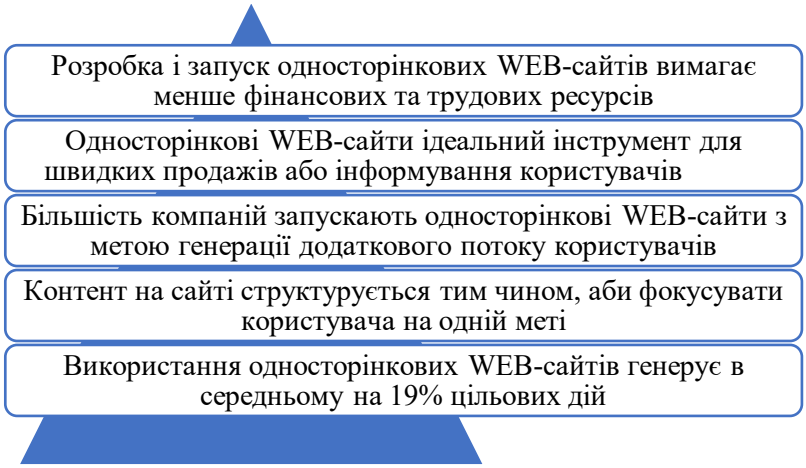


Рисунок 1.4. – **Переваги розробки і запуску односторінкових WEB-сайтів**

Джерело: сформовано автором на основі [6]

Не зважаючи на переваги односторінкових WEB-сайтів, недоліки також присутні, і найвагоміший з них те, що такі сайти викликають менше довіри у користувачів, аніж великі інтернет-магазин. Окрім цього, такі сайти гірше просувається в SEO і індексується пошуковими системами, а тому, розробка і запуск односторінкових WEB-сайтів має місце тоді коли підприємство вже користується увагою у користувачів, наприклад має інтернет-магазин або інший сайт на якому висвітлена інформація про його діяльність. В іншому випадку такі сайти не принесуть бажаного результату.

В першу чергу, розробка WEB-сайту починатися з визначення цілі створення сайту, серед яких може бути, наприклад інформування споживачів щодо пропонованої продукції, в такому випадку сайт буде мати яскраві та акцентні елементи або створення іміджу надійності, тоді використовуються стриманий стиль викладенні інформації із увагою на досягнення підприємства за

час своєї діяльності. Інакше кажучи, визначення приналежності до однієї чи іншої групи є вкрай важливим, адже це дає змогу сформулювати основну концепцію сайту. Залежно від того, яку ціль переслідує підприємство, обирається тип сайту (short або long list), структура контенту та система зв'язків сторінок або секції між собою. В загальному вигляді, можна виділити шість етапів створення WEB-сайтів (рис. 1.5):

Рисунок 1.5. – **Етапи створення сайту**

Джерело: сформовано автором на основі [7]

Головним завданням, яке стоїть перед розробниками WEB-сайту, є визначення структури сайту та створення карти зв'язків сторінок або секцій. Структура сайту – це тип організації веб-контенту для представлення його у зручному для користувача вигляді [8]. Організації веб-контенту полягає у структуруванні даних (текстової, графічної) про предметну галузь групуючи у вигляді окремих WEB-сторінок.

Залежно від того, якої складності передбачається розробка WEB-сайту, кількості і типів зв'язків WEB-сторінок, розрізняють декілька видів структур їх організації, зокрема лінійна, блокова, стандартна, каскадна, хмарочос, павутинна та деревоподібна [9]. Серед наведеного переліку варто розглянути найбільш часто використовувані на практиці типи структур, зокрема лінійну, довільну, стандартну та деревоподібна. Зазначені концепції структурування сторінок та контенту є найбільш поширеними при розробці WEB-ресурсів будь-якості складності – від односторінкових сайтів до потужних, багатосторінкових WEB-порталів [10].

WEB-сайти з лінійною структурою побудови передбачають реалізацію послідовної навігації між WEB-сторінками. За даної структури WEB-сторінки

утворюють логічний ланцюжок навігації, тобто з головної сторінки можна перейти на другу, з неї – на третю тощо (рис. 1.6).

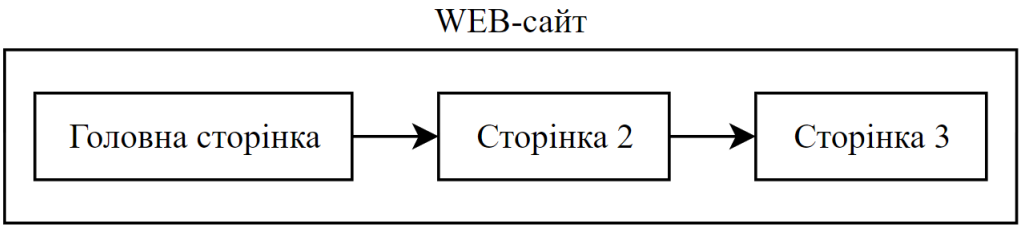


Рисунок 1.6. – **Лінійна структура WEB-сайту**

Джерело: сформовано автором на основі [1, 10, 11]

Варто зауважити на тому, що в рамках специфіки односторінкових WEB-сайтів, навігація на сторінці будується переважно на принципах лінійної структуру, за тим винятком, що навігація відбувається не між рядом сторінок, як наприклад, в інтернет-магазинах, а між блоками (секціями) WEB-сторінки.

Блоковий підхід до організації навігації між WEB-сторінками є одним із найбільш зручних для структурування невеликої кількості сторінок (рис. 1.7).

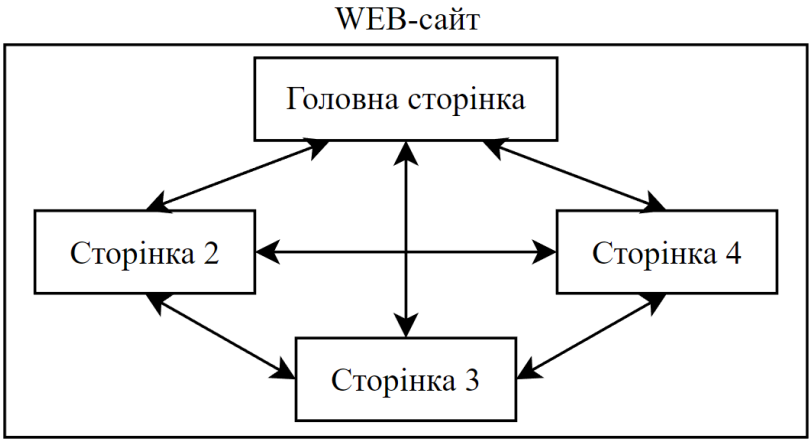


Рисунок 1.7. – **Блокова структура WEB-сайту**

Джерело: сформовано автором на основі [1, 10, 11]

На перший погляд сайт із такою структурою є неорганізованим, однак суть його полягає у розташуванні на усіх WEB-сторінках посилань на інші сторінки сайту, що дозволяє користувачу переходити з однієї його сторінки на інші в різні способи [11]. Особливістю такої структури WEB-сайту є побудова системи зворотної навігації сайту із вертикальними та горизонтальними елементами сайту (сторінками), тобто є можливість швидкого переходу від однієї до іншої

сторінки без відвідування проміжних сторінок.

В рамках стандартної структури організації WEB-сайту навігація між WEB-сторінками реалізовується через посилання на головній сторінці. Тобто, головна WEB-сторінка містить посилання на внутрішні сторінки сайту, а останні, в свою чергу, містять посилання, на головну веб-сторінку. Недоліком такої структури є неможливість навігації між сторінками без відвідування «посередника» у якості головної сторінки (рис. 1.8).

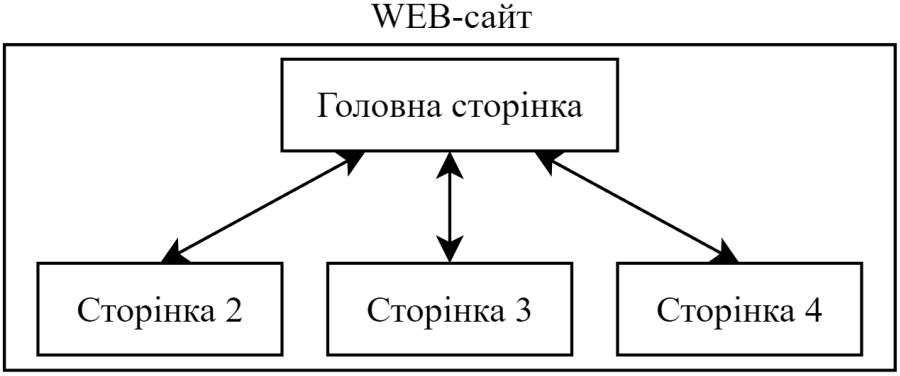


Рисунок 1.8. – **Стандартна структура WEB-сайту**

Джерело: сформовано автором на основі [1, 10, 11]

Найбільш часто використовуваним на практиці підходом до організації зв'язків між сторінками є деревоподібна або ієрархічна структура (рис. 1.9).

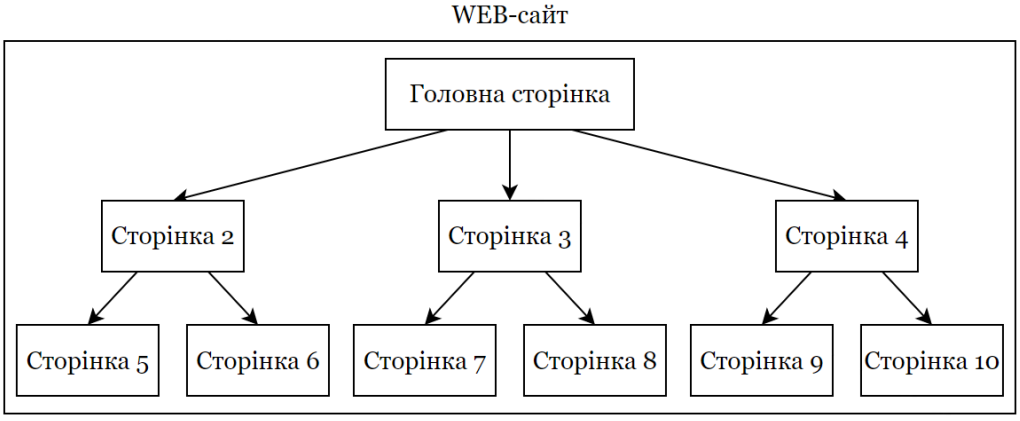


Рисунок 1.9. – **Деревоподібна (ієрархічна) структура WEB-сайту**

Джерело: сформовано автором на основі [1, 10, 11]

В рамках цієї структури основні розділи сайту (батьківські сторінки) діляться на ряд більш конкретних (дочірніх) сторінок. За даною структурою, сайт ділиться на ряд основних розділів за певною тематикою, а внутрішні сторінки

пов’язуються між собою навігацією в межах даного розділу. Якщо сайт передбачає висвітлювання великої кількості інформації, ієрархічну структуру потрібно розглядати першою в якості організації зв’язків сторінок [7]. Гарним прикладом сайтів де застосовується ієрархічна структура зв’язків сторінок – це інтернет-магазини, форуми, тематичні та інформацій портали тощо.

Таким чином, вибір структури визначається особливостями завдань та типом WEB-сайту, який направлений на вирішення поставлених задач. У цьому контексті розглянемо основні види WEB-сайтів, які покликані забезпечити підтримку діяльності підприємств будь-якої специфіки їх функціонування.

1.2. Огляд наявних аналогів

Розробка і запуск будь-якого сайту – це багатоетапний процес, який вимагає від команди розробників детального аналізу ринку, планування, прототипування та тестування WEB-сайтів. Перед розробкою дизайну або структури WEB-сайту необхідно здійснити аналіз односторінкових сайтів різного напряму їх використання та виявити сучасні тренди до організації контенту, дизайнерські підходи до оформлення сторінок, «юзабіліті» досвід користувачів, а також недоліки при користуванні функціоналом на основі чого сформувати власне бачення майбутнього WEB-сайту.

З метою аналізу односторінкових WEB-сайтів було обрано сайти з різним функціональним наповнення, напрямом їх використання, дизайном і контентом. На нашу думку, зазначене дасть можливість сформувати основні концепції нашого односторінкового WEB-сайту.

Для досягнення поставленої мети було проведено дослідження наступних веб-сайтів:

1) Курси англійської мови у м. Вінниця «Project12» – https://p12.com.ua/vinnytsia

2) Онлайн-курси з WEB-розробки «WezomAcademy» – https://wezom.academy/ua/

3) Сервіс замовлення їжі «Health&Food» – https://health-food.com.ua/ua/

Сайт «Курси англійської мови у м. Вінниця» гарний приклад використання інтерактивного підходу до інформування користувачів. Сайт виконаний у лаконічному стилі із використанням лише трьох акцентних кольорів – чорний, жовтий, білий. На сайті присутня секція із відео-роликом для більш детального представлення своїх послуг. Цікавим моментом, що є не типовим для односторінкових WEB-сайтів, є те, що на даному сайті реалізований особистий кабінет користувача у якому можна зареєструватися та отримувати інформацію про послуги безпосередньо у власному кабінеті (рис. 1.10).

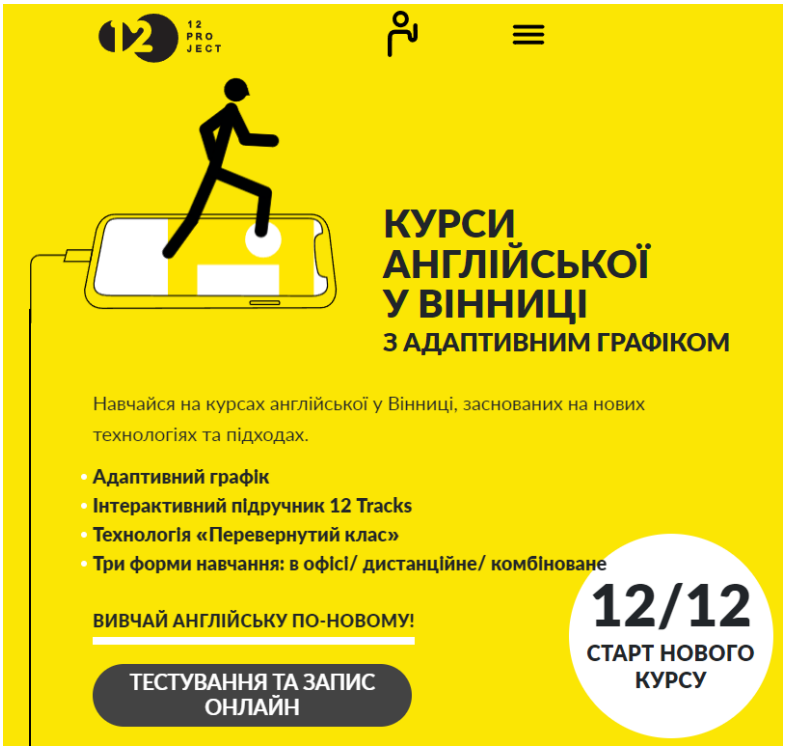


Рисунок 1.10. – **Сайт «Project12»**

Джерело: [12]

На WEB-сайті присутня анімація, яка у «ігровому» стилі «веде» користувача до оформлення заявки на безкоштовне пробне заняття (рис. 1.11).

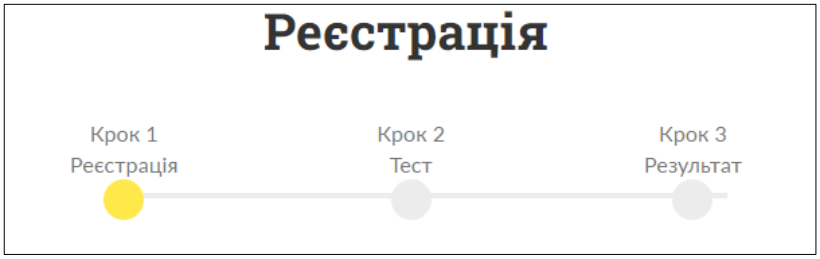


Рисунок 1.11. – **Форма реєстрації користувача на сайті «Project12»**

Джерело: [12]

Також на сайті є посилання та демонстрація роботи мобільного додатку для вивчення іноземної мови. Окрім вищенаведеного, сайт має адаптивний дизайн та високу оптимізацію для перегляду контенту на різних пристроях. Варто зауважити, що адаптивний дизайн – це особливий вид верстки сайту, який враховує характеристики різних пристроїв, забезпечуючи правильне відображення WEB-ресурсу на екранах різного розміру. Таким чином, відвідувач може без проблем скористатися всіма можливостями сайту за допомогою свого смартфону або планшету (рис. 1.12).

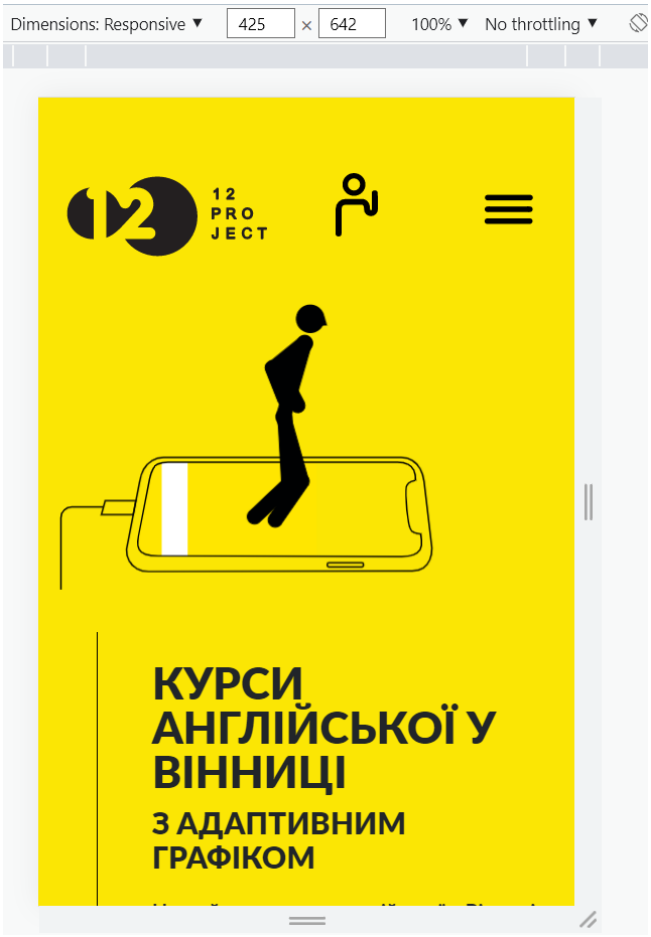


Рисунок 1.12. – **Адаптований дизайн сайту «Project12»**

Джерело: [12]

У якості недоліку ми вбачаємо відсутність панель навігації для інформування користувача, яку секцію на даний момент він переглядає, а також існують некоректне відображення окремих секції в рамках адаптивного дизайну на планшетних розширеннях.

Односторінковий WEB-сайт з реєстрації на IT курси містить особистий

кабінет та форму реєстрації користувача. На головному екрані розміщується акцентна інформація щодо заповнення форми на пробне безкоштовне заняття (рис. 1.13).

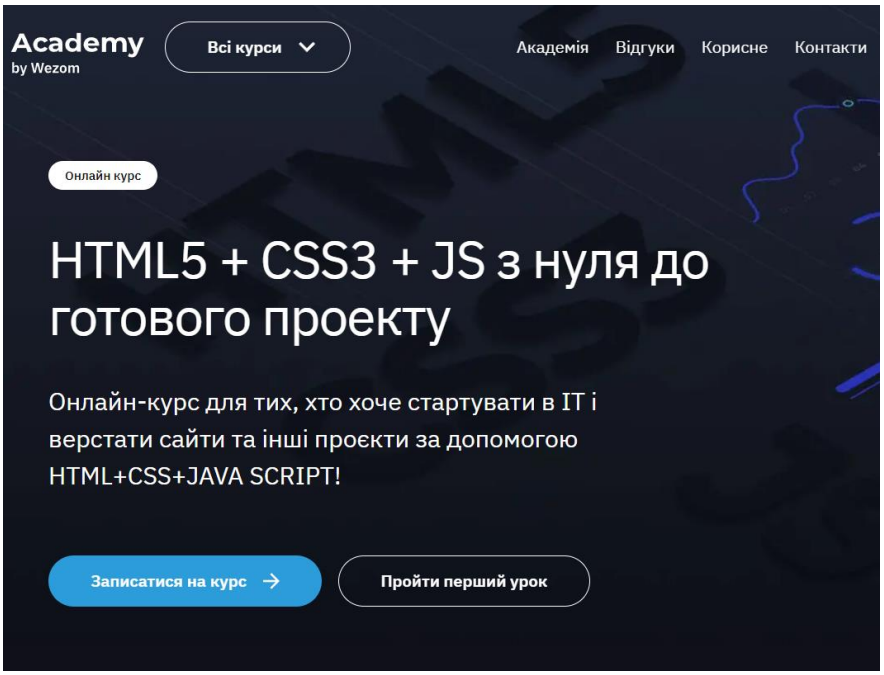


Рисунок 1.13. – **Сайт «WezomAcademy»**

Джерело: [13]

При бажанні зареєструватися на сайті для оформлення замовлення або отримання консультації, користувача переводить на внутрішню сторінку з реєстраційною формою, однак без можливості зміни мови (рис. 1.14).

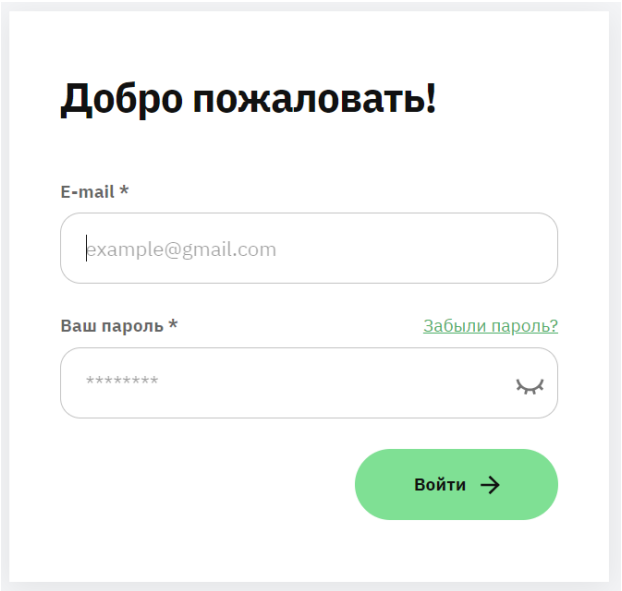


Рисунок 1.14. – **Форма реєстрації користувача на сайті «WezomAcademy»**

Джерело: [13]

Як і на попередньому сайті, на даному також присутня секція з відеоматеріалом та фотогалерея для демонстрації організації навчання. Присутня інформація про команду викладачів курсів, секція із відгуками та інформацією про випускників. Також присутня можливість зміни мови сайту.

Загалом, сайт складається більше ніж 15 секцій, які логічно відділяються одна від одної дизайном та чіткими заголовками. Навігація між секціями супроводжується м'якою анімацією. Аналізований сайт, як і попередній, теж має адаптований дизайн для перегляду на мобільних пристроях, що є позитивним аспектом (рис. 1.15).

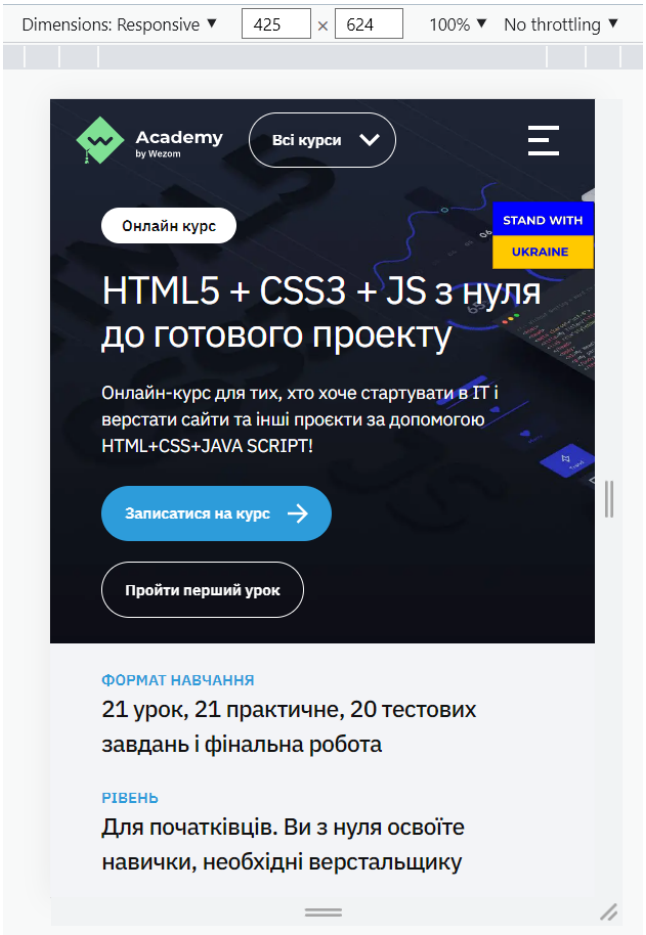


Рисунок 1.15. – **Адаптивний дизайн сайту «WezomAcademy»**

Джерело: [13]

До недоліків, на нашу думку, можна віднести перевантаженість інформацією окремих секцій, також відсутня контактна форма зв'язку для онлайн-спілкування. Також, під час переглядання сайту, періодично з'являється реклама, яка відволікає від переглядання контенту. Окрім цього, існують

проблеми із відображенням плаваючих форм на розширеннях менше 425 пікселів.

Наступний сайт відрізняється від решти оригінальним дизайном та структуруванням інформації – він має вигляд інтернет-магазину, але у стилі односторінкового WEB-сайту (рис. 1.16).

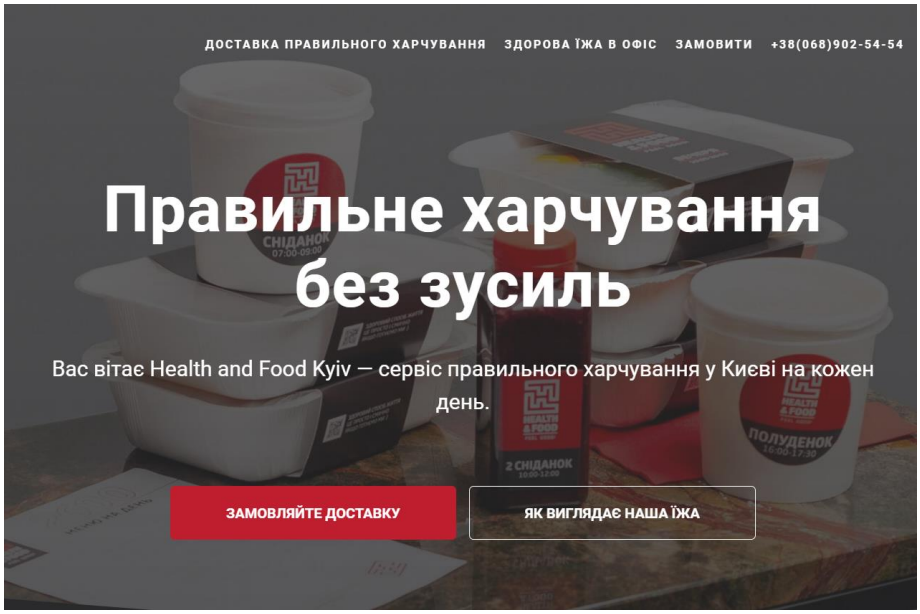


Рисунок 1.16. – **Сайт «Health&Food»**

Джерело: [14]

Сайт виконаний у мінімалістичному стилі з невеликою кількістю використання анімаційних скриптів. Даний сайт є чудовим прикладом синтезу структури односторінкового сайту та функціоналу інтернет-магазину. При натисканні на пропоновану продукцію, відкривається внутрішня сторінка, на якій наведена інформація про обрану позицію, а також корзина для замовлення. Даний сайт містить фото-галерею, форму зворотного зв’язку, а також контактна форма зв’язку для онлайн-спілкування. Існує функція попереднього перегляду інформації про обрану позицію, для цього на картці товару є спеціальна шрифтова іконка, натиснувши на яку, відкривається плаваюче вікно з галереєю.

Навігація по сайту супроводжується цікавою паралакс-анімацією, яка по мірі швидкості перемикання між секціями, який дає ефект того, що задній фон сайту рухається повільніше (або швидше) аніж об’єкт, що знаходяться на передньому плані.

Цікавою особливістю сайту є відображення контенту на усю можливу висоту адаптуючись до пристрою користувача (рис. 1.17).

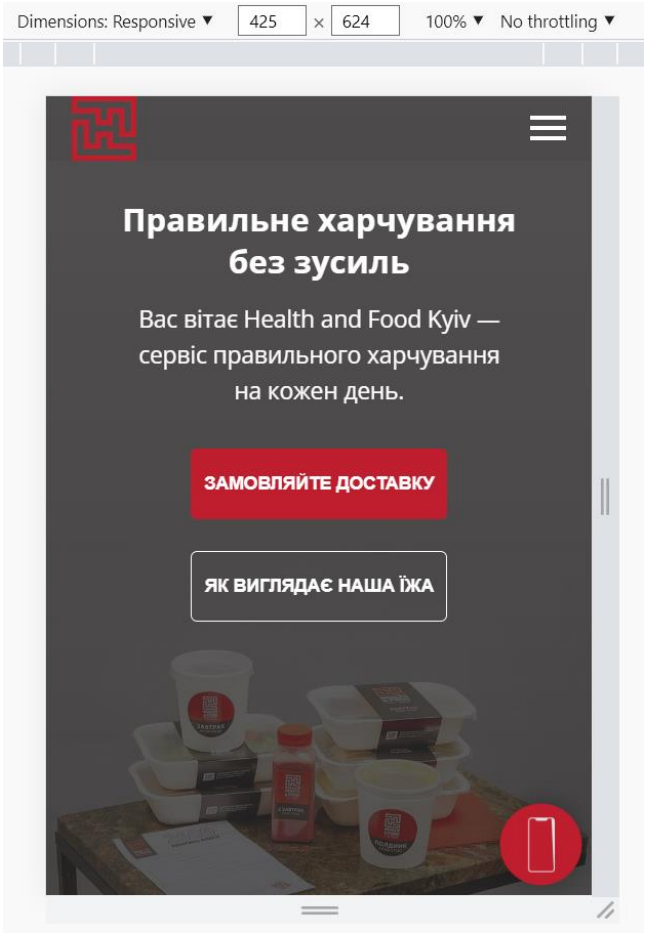


Рисунок 1.17. – **Адаптивний дизайн сайту «Health&Food»**

Джерело: [14]

Усі секції сайту мають відмінне від попередньої секцій оформлення, що візуально фокусує користувача на інформації. Присутній відеоматеріал із коментарями та відгуками клієнтів. На сайті присутній калькулятор калорій та конструктор індивідуального меню. Серед недоліків можна виділити некоректне відображення галереї зображень, а також проблеми з функціонуванням калькулятора на розширеннях менше 768 пікселів.

Проаналізувавши представлені односторінкові WEB-сайт сайти можна виокремити спільні переваги, зокрема усі аналізовані сайти мають мобільну версію для можливості перегляду контексту на різних пристроях. Окремо варто зауважити на тому, що на всіх сайтах присутні секції із тарифами за продукти чи послуги, секції з відеоматеріалами, а також форми зв’язку.

Розглянемо детальніше інформацію про сайти та проведемо загальний аналіз (табл. 1.2).

Таблиця 1.2

Порівняльна характеристика аналізованих односторінкових WEB-сайтів

Критерії оцінки	WEB-сайти		
	Курси англійської мови у м. Вінниця	Онлайн-курси з WEB-розробки	Сервіс замовлення їжі
Відсутність реклами	+	-	+
Форма зворотного зв'язку	+	+	+
Форма для онлайн-спілкування	+	-	+
Адаптований дизайн	+	+	+
Особистий кабінет користувача	+	+	-
Наявність секції із тарифною сіткою	+	+	+
Наявність секції із відеоматеріалами	+	+	+
Наявність анімації	+	-	+
Не переповненість інформацією	+	-	+
SEO оптимізація *	+	-	+

Джерело: [15]

Окремо варто проаналізувати дані сайти на наявність правильної розмітки у відповідності з визначенням типу документа. Перевірка розмітки є важливим кроком до забезпечення технічної якості WEB-сторінок та можливості їх подальшого оновлення та масштабування. Даний аналіз можна здійснити за допомогою валідатора Консорціуму Всесвітньої павутини W3C (W3C Markup Validation Service), який дозволяє користувачам Інтернету перевіряти документи HTML і XHTML до HTML5.

Валідатор W3C покликаний перевірити, наскільки сайт відповідає єдиним стандартам і при виявленні помилок вказує на них, та формує пропозиції щодо їх виправлення. Якщо ігнорувати найдені помилки, може виявитися, що навіть неправильно закритої дужки, сайт може невірно завантажуватися у браузері. Зазначене призводять до того, що користувачі швидко залишають WEB-сторінку і підприємство втрачає потенційний потік користувачів.

Перевагами валідатора W3C є [16]: 1) підняття сайту у рейтингу пошукових системах; 2) покращення ефективності сайту; 3) забезпечення універсальності сайту для будь-якого браузера; 4) адаптивність для різних видів пристрій; 5) спрощення технічного обслуговування сайту

Таким чином, нами було здійснено перевірку аналізованих сайтів. Результати аналізу наведено у табл. 1.3.

Таблиця 1.3

Результатами порівняння відповідності структури сайтів стандартам W3C

WEB-сайт	Тип помилки *			
	Info	Warning	Error	Fatal Error
https://p12.com.ua/vinnytsia	9	3	5	1
https://wezom.academy/ua/	21	1	8	0
https://health-food.com.ua/ua/	51	15	17	0

*Примітка: *Info – пропущені або існують зайві символи у розмітці сторінки; Warning – конструкції коду, які можуть не підтримуватися застарілими браузерами; Error – конструкції коду, у яких пропущені параметри та (або) атрибути; Fatal Error – конструкції коду, які не відповідають загальноприйнятим стандартам розмітки*

Джерело: [16]

Резюмуючи вище наведене, можна виділити основні риси якісних односторінкових WEB-сайтів, зокрема – це лаконічність у тексті і дизайні; стильна та достатня кількість анімації; використання векторних зображень; наявність секцій із відеоматеріалами, відгуками, тарифними сітками; ненав'язливі форм реєстрації користувачів та замовлення послуг; адаптивний дизайн для комфортного перегляду сайту на різних пристроях. Окремою перевагою можна вважати наявність інтерактивної або «ігрової» стилістики сайту.

1.3. Постановка задачі

В умовах стрімкого розвитку ринкових відносин, перед підприємствами висуваються нові вимоги та виклики, де першочерговими завданнями є утримання своїх позицій та збереження конкурентних позицій. З появою та стрімким поширенням мережі Інтернет, який є ефективним інструментом

розвитку бізнесу та торгівлі, традиційні канали комунікації втрачають свою актуальність, натомість ним все швидше розвивається інтернет-комунікація. Одним із таких каналів комунікації із користувачами, а в наслідку потенційними клієнтами, є WEB-сайт підприємства.

WEB-сайт є одним із основних інструментів інтернет-маркетингу будь-якого підприємства, ціль якого полягає у забезпеченні автоматизованого виконання завдань для підвищення економічної ефективності функціонування організації. В сучасних умовах розробка та використання WEB-сайту є запорукою успішного бізнесу, оскільки за його допомогою підприємство може проаналізувати настрої користувачів, утримати наявних та залучити нових клієнтів, сформувати клієнтську базу, прорекламувати свої послуги та/або продукти, популяризувати свій бренд тощо [17]. Перераховане дозволяє розглядати WEB-сайт як результат та інструмент економічної діяльності підприємств та основою формування їх іміджу.

Актуальність і необхідність розробки WEB-сайтів неможливо переоцінити, оскільки якісно побудований інформаційний ресурс лягає в основу формування комунікаційної політики підприємств та їх високої конкурентоспроможності на ринку.

Таким чином, метою кваліфікаційної роботи є розробка односторінкового WEB-сайту для підтримки діяльності підприємств, загальні риси якого можуть бути використані підприємствами будь-якої сфери діяльності.

В рамках поставленої мети були поставлені наступні задачі:

1) проаналізувати предметну область дослідження;

2) дослідити теоретичні засади розробки та використання WEB-сайтів;

3) обрати та описати інструменти розробки WEB-сайтів;

4) здійснити верстку WEB-сайту;

5) завантажити WEB-сайт на хостинг.

Використання розробленого WEB-сайту спрямовано на підтримку діяльності підприємств у контексті інформування користувачів про товари (послуги) та збору контактної інформації.

РОЗДІЛ 2
СУЧАСНІ ПІДХОДИ, ІНСТРУМЕНТИ ТА ТЕХНОЛОГІЇ РОЗРОБКИ ОДНОСТОРІНКОВИХ WEB-САЙТІВ

2.1. Опис програмного середовища для WEB-розробки

Текстові редактори або так звані редактори коду є важливим інструментом для веб-розробників. Питання вибору оптимального редактора коду для розробника є одним із найважливіших етапів у процесі створення WEB-сайту. При виборі відповідного середовища розробник орієнтуються на низку факторів, зокрема чи підтримує програмний засіб необхідні мови програмування; чи є існує підсвічування синтаксису для декількох мов програмування; чи є вбудований реєстр помилок; чи є можливість спільної розробки для командної роботи; чи є можливість додаткового налаштування програмного середовища під власні потреби тощо [18]. Сучасні редактори коду не лише пришвидшують роботу, вони надають широкий асортимент інструментів, які зменшують кількість кроків, необхідних для виконання певних завдань. Таким чином, вибір і використання зручного та зрозумілого інструменту розробки – це основа написання якісного та зрозумілого програмного коду за мінімальний час, що є рисою висококваліфікованого, продуктивного та конкурентоспроможного фахівця у сучасний цифровий час.

Редактор коду або IDE – це програмне забезпечення, яке за допомогою спеціальних інструментів надає можливість розробляти додатки різного рівня складності, а також здійснювати їх тестування в єдиному програмному середовищі. Інакше кажучи, IDE – це текстовий редактор, який надає широкі можливість до написання, форматування, компілювання та тестування програмного коду. Такі програмні середовища розробки дозволяють створювати великі додатки, а також підключати GIT (розподілену систему керування версіями файлів) для спільної командної роботи над проєктами.

В загальному вигляді IDE будується безпосередньо з текстового редактора для написання та редагування коду; компілятора або інструмента, що дозволяє

«трансформувати» написаний розробником програмний текст у набір машинних кодів; модуля тестування, який перевіряє код і допомагає виявити помилки в процесі розробки; інструментів для автоматизації форматування коду, що прискорює процес розробки.

Сьогодні, найбільш популярними редакторами коду для WEB-розробки є такі програми як Visual Studio Code, Sublime Text, Atom, Eclipse. В рамках даної роботи, для розробки односторінкового WEB-сайту нами був обраний Visual Studio Code – надійний, потужний та невибагливий до ресурсів редактор коду. Visual Studio Code, сьогодні, є одним із найбільш популярних редакторів коду розроблений компанією Microsoft.

Згідно з опитування проведеного найбільшою WEB-спільнотою Stack Overflow, станом на кінець 2021 р. більше ніж 71% користувачів WEB-порталу вважають, що Visual Studio Code є одним із найбільш зручних і швидких програмних середовищ для розробки проєктів будь-якої складності (рис. 2.1).

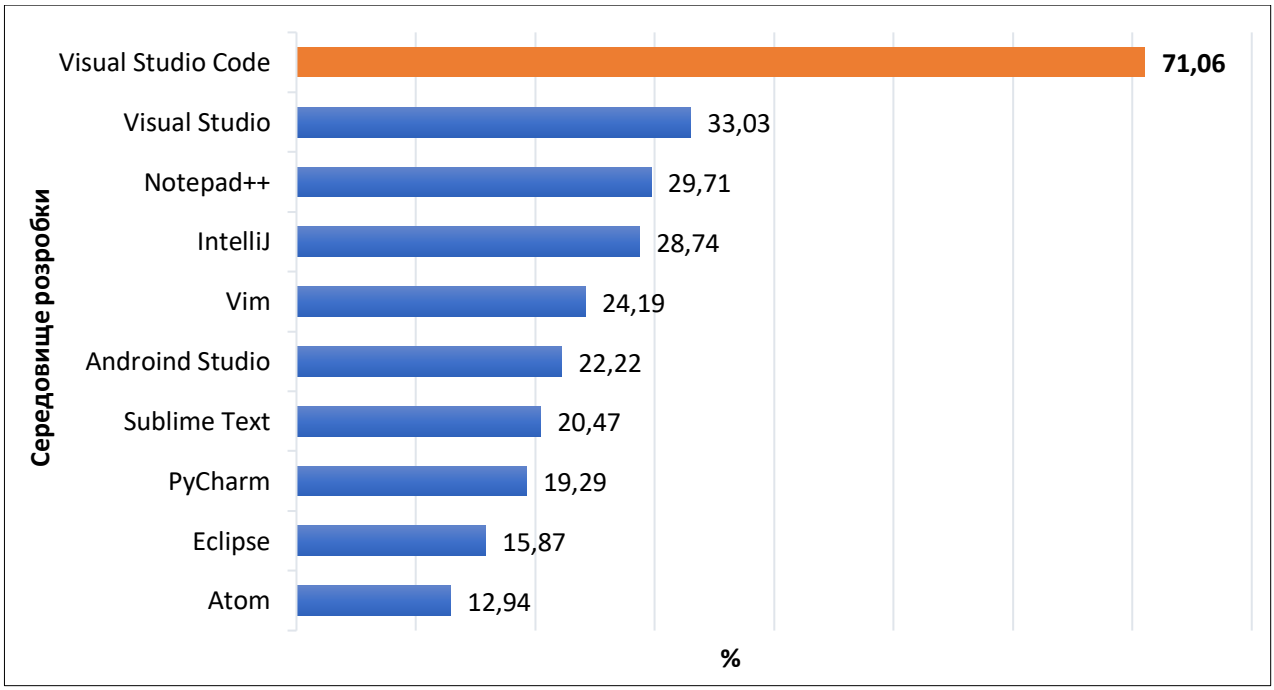


Рисунок 2.1. – **Рейтинг програмних середовищ за оцінкою користувачів Stack Overflow у 2021 р., %**

Джерело: побудовано на основі [19]

Visual Studio Code – це крос-платформний редактор коду, з відкритим вихідним кодом, який підтримує автоматичне інтелектуальне доповнення тексту

програм з використанням технології Microsoft IntelliSense та підсвічування синтаксису для більшості популярних мов. Visual Studio Code поширюється безкоштовно і може бути встановлений на всі популярні платформи, такі як Windows, Linux або MacOS. Дане програмне забезпечення є звичайним текстовим редактором з можливістю підключення різних плагінів, що дає можливість працювати з різними мовами програмування для розробки IT-продукту будь-якої складності

Завантажити Visual Studio Code можна з офіційного сайту програмного продукту, де можна знайти версії для усіх популярних операційних систем. Особливістю даного середовища розробки є його інтеграція у контекстне меню та реєстрація усіх можливих для відкриття розширень файлів. Для ввімкнення даної опції, в інсталяторі програми необхідно поставити відмітки як показано на рис. 2.2.

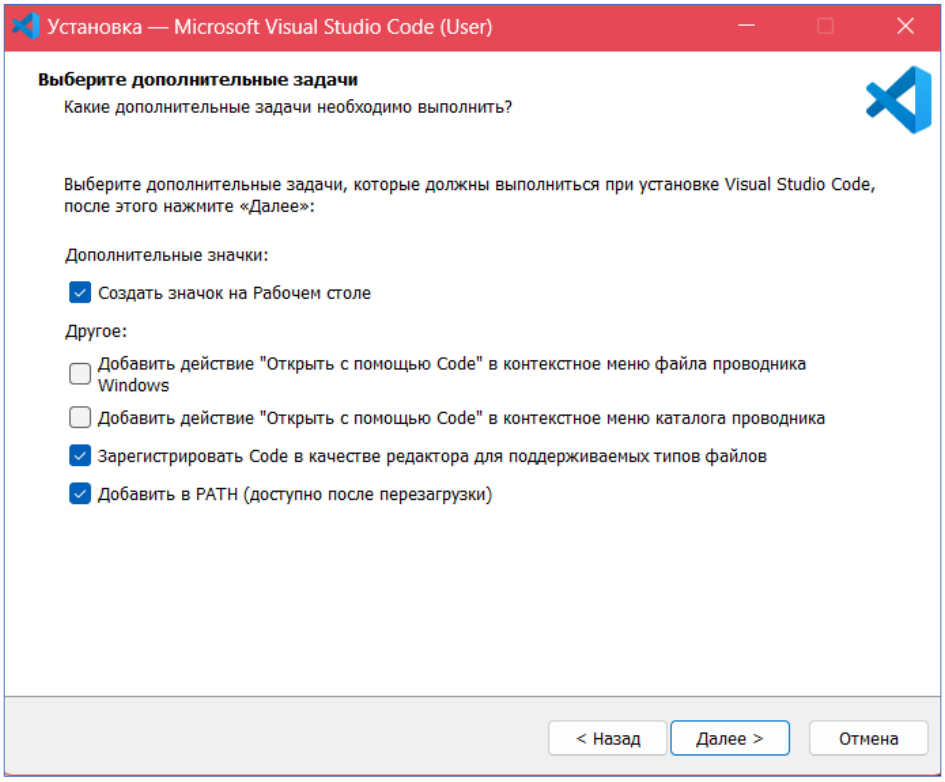


Рисунок 2.2. – **Вікно інсталятора середовища розробки Visual Studio Code**

Джерело: [20]

Середовище розробки Visual Studio Code має зручний інтерфейс з можливістю одночасного відкриття декількох незалежних вікон і панелей та

візуальному розділенню програмної області робочого вікна. Зазначеною опцією було зручно користуватися під час розробки сайту – в одній частині екрану був відкритий файл з PUG-розміткою, в іншому – з SCSS кодом (рис. 2.3).

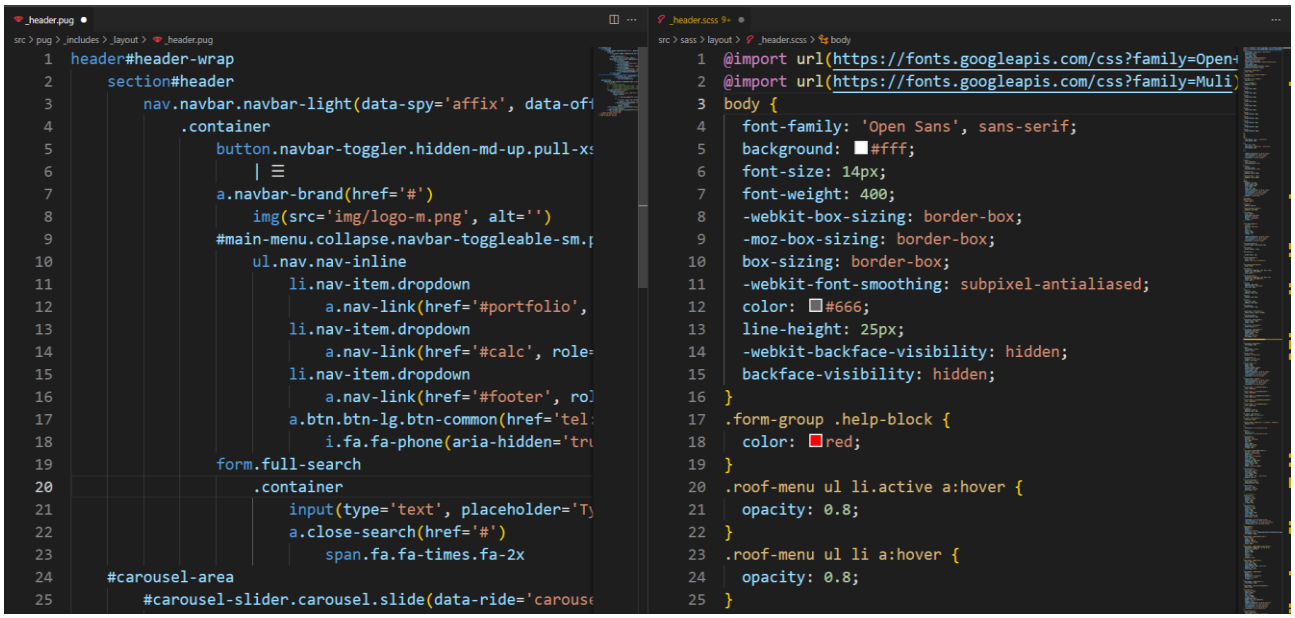


Рисунок 2.3. – **Інтерфейс редактора Visual Studio Code**

Джерело: власна розробка

Завдяки підтримці багатьох мов програмування, Visual Studio Code містить автоматичне підсвічування синтаксису, перевірку на відповідність дужок, форматуванням відступів, виділення вікон, фрагментів коду тощо. Окрім цього, у редакторі вбудований засіб для роботи з GIT-репозиторіями та інструменти для рефакторингу фрагментів коду, а також присутня фірмова функція Live Share, яка надає можливість спільного програмування, завдяки чому команда може без додаткового налаштування свого програмного середовища почати працювати над одним проєктом, просто завантаживши через хмарне сховище файл конфігурації.

Даний редактор підтримує такі мови програмування та розмітки як HTML, XHTML, XML, CSS, JavaScript, C#, C++, Python, Java, PHP, Ruby та багато інших. Такі мови, як JavaScript, TypeScript, CSS і HTML відразу доступні після встановлення додатку; для мов, наприклад Python, Java, PHP, Ruby Visual Studio Code містить розширену екосистему та окреме середовище виконання програмного коду (для сімейства .NET мов). Щоб мати можливість

використовувати інші мови програмування необхідно додатково встановлювати та налаштовувати окремі розширення за допомогою інтерфейсу програми Visual Studio Code або платформи Visual Studio Code Marketplace.

На рис. 2.4 наведено приклад встановлення розширень для підтримки мов програмування C# та PHP. Аналогічним чином можна встановляти розширення та плагіни для інших мов програмування, а також для оптимізації та підвищення продуктивності праці.

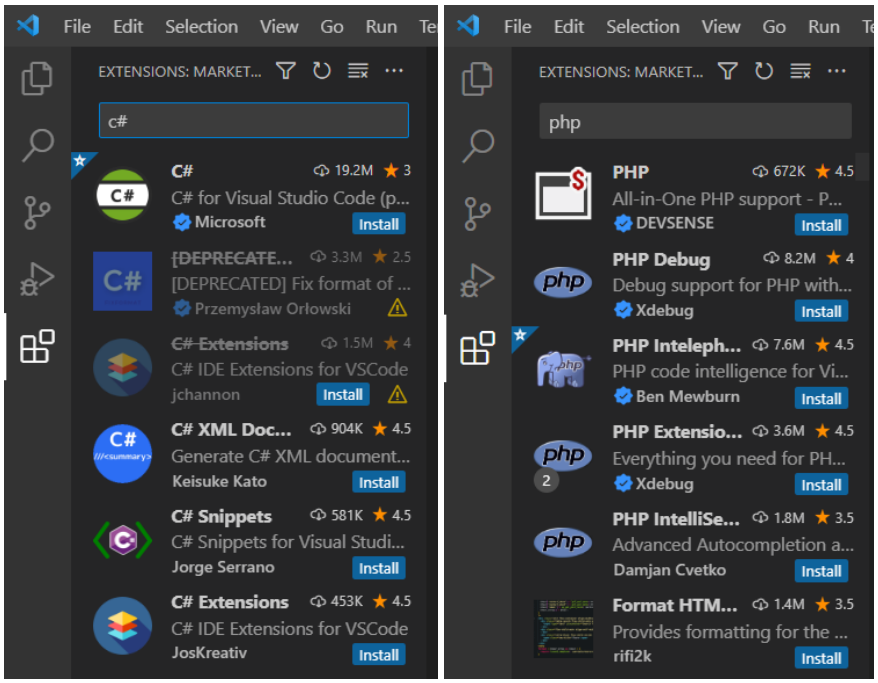


Рисунок 2.4. – **Принцип встановлення розширень через інтерфейс Visual Studio Code Marketplace**

Джерело: власна розробка

В рамках нашого дослідження, нами було встановлення ряд додаткових розширень та плагінів для більш зручної роботи над проєктом. Зокрема було встановлено розширення Pug/Jade Snippets, яке дозволяє зробити шаблон розмітки певної секції або блоку, та при необхідності викликати команду для динамічної інтеграції фрагментів розмітки. Також, було встановлено плагін Pug beautify, який несе в собі функцію візуальної правки написаного коду, для підтримки усіх відступів та пробілів між ключовими тегами розмітки.

Для більш зручної роботи з SCSS файлами препроцесора SASS було встановлено розширення SCSS Formatter, яке дозволяє оформляти файли стилів

згідно вимог та спростити пошук необхідного селектора. Окрім цього, нами було встановлено розширення SCSS IntelliSense, яке дозволяє спростити написання властивостей та їх значень, шляхом формування підказок.

В рамках роботи з файлами JS ми встановили розширення JSHint, яке дає можливість використовувати раніше заготовлені фрагменти коду. Також, був встановлений плагін JS Refactor, який дозволяє реструктурувати написаний JS код, у більш компактні фрагменти коду, що дає можливість мінімізувати появу помилок при формуванні кінцевої збірки проєкту (рис. 2.5).

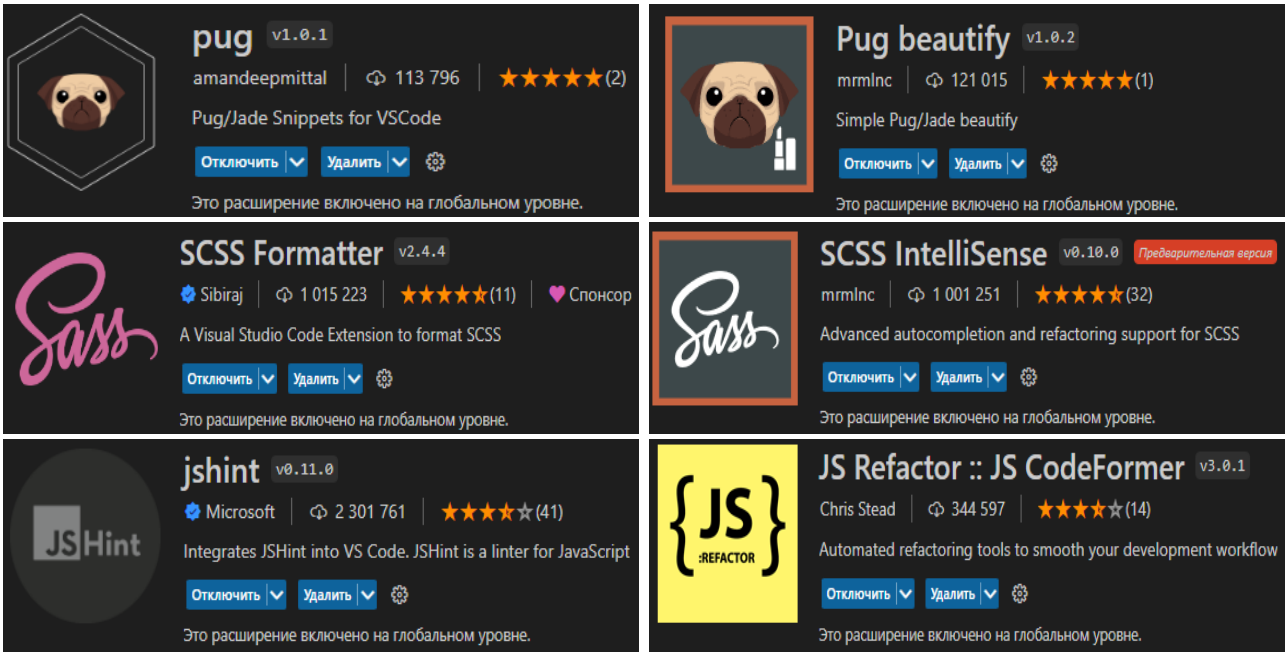


Рисунок 2.5. – **Встановленні розширення Visual Studio Code**

Джерело: власна розробка

Окрім наведених вище особливостей Visual Studio Code, даний редактор містить фірмові функції, які відсутні в інших програмних середовищах, зокрема це потужна інтеграція з хмарним програмним забезпеченням Azure. Говорячи про тісну інтеграцію Azure та VS Code, варто звернути увагу на те, що це хмарна платформа, яка поєднує в собі рішення для обчислювальної інфраструктури IaaS (сервери, сховища даних, мережі, операційні системи), так і набір інструментів і сервісів, що полегшують розробку і розгортання хмарних додатків (PaaS). Інакше кажучи, Azure – це хмарна платформа призначена для забезпечення розгортання онлайн застосунків та виконання обчислень. Функції Azure

дозволяють в меншому обсязі писати та підтримувати об’ємний програмний код. Серед окремих особливостей використання інтеграції Azure є тестування розробленого додатку у хмарному середовищі Azure поклавши складні обчислення на сервера шляхом розгортання спеціальних емуляторів (рис. 2.6).

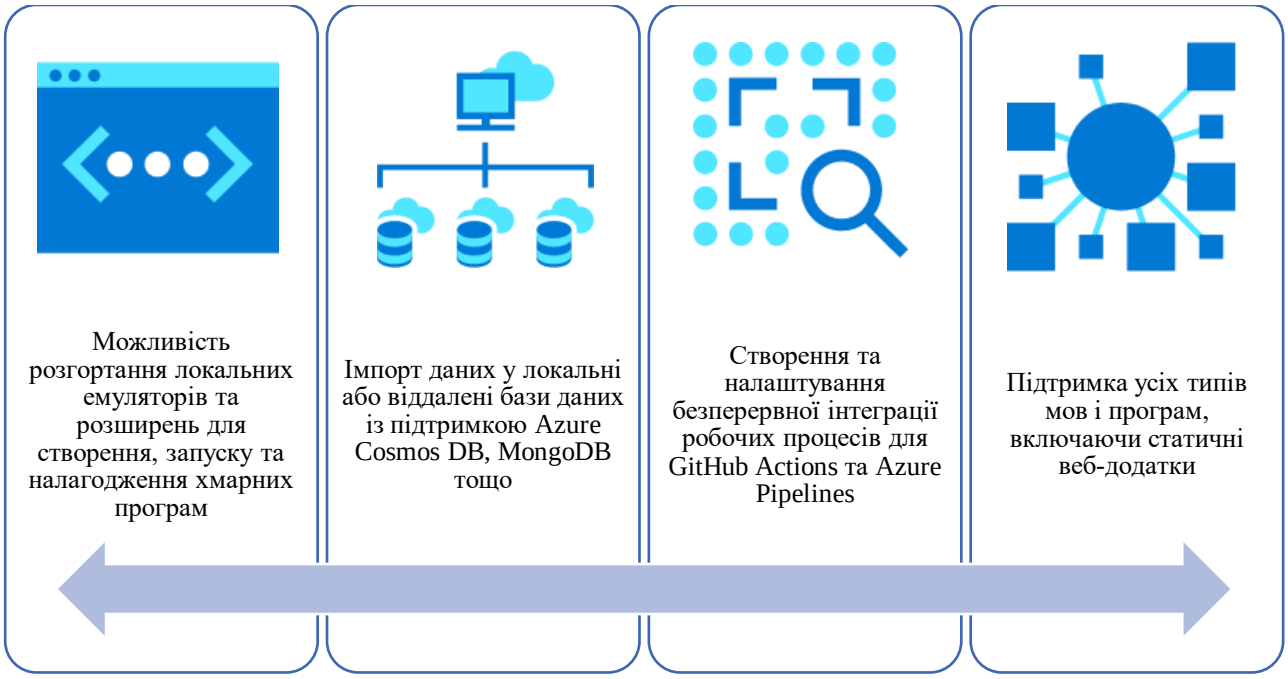


Рисунок 2.6. – **Інструменти розгортання проєктів у хмарному середовищі**

Джерело: [21]

Отже, можна виділити ряд основних переваг Visual Studio Code у порівнянні з іншими редакторами коду, зокрема це: 1) широкі можливості до налаштувань програми в цілому або окремо інтерфейсу; 2) велика бібліотека з доповненнями і готовими рішеннями; 3) підтримка усіх мов, які використовуються для створення програм; 4) інтеграція із хмарним середовищем Azure; 5) загальнодоступна модель поширення програмного засобу.

2.2. Характеристика технологій створення WEB-сайтів

Всесвітня павутина (World Wide Web) складається з WEB-сайтів, які у свою чергу є набором WEB-сторінок написаних мовою HTML (HyperText Markup Language). По своїй суті, всесвітня павутина – це колекція текстових документів, розмічених мовою HTML, яка диктує форму подання інформації і структуру зв’язків між цими файлами й іншими інформаційними ресурсами.

Перший у світі сайт був розроблений у 1991 році, на якому Тім Бернерс-Лі, за допомогою гіпертекстової мови розмітки HTML (якої він є розробником), опублікував інформацію про технологію World Wide Web, а також принципи встановлення, налаштування та роботу серверів та WEB-переглядачів [22]. Сайт містив інформацію про технічні особливості гіпертекстової мови розмітки HTML, опис базових тегів та структуру простої WEB-сторінки. В силу обмежених на той час технологічних можливостей, сайт мав вигляд статичної WEB-сторінки. Сьогодні, з розвитком WEB-технологій, статичні WEB-сторінки майже не користуються популярністю, їх замінили повнофункціональні динамічні WEB-сайти.

Як зазначалося вище, HTML – це гіпертекстова мова розмітки або мова розмітки гіпертексту. За її допомогою розробник може створювати та створювати ієрархію елементів, блоків, секцій, абзаців, заголовків, гіперпосилань, зображень тощо для WEB-сторінок та додатків. Тобто, мова HTML дозволяє створювати лише каркас WEB-сторінки, вона не дає можливості реалізовувати динамічну функціональність. Лише у поєднанні з такими технологіями як CSS та JavaScript можна створити складні, динамічні та повнофункціональні WEB-сайти (рис. 2.7).

Рисунок 2.7. – **Сучасні технології WEB-розробки**

Джерело: [11, 23]

Гіперпосилання (Hyperlink) – це базовий функціональний елемент HTML мови, який реалізовує зв’язок певного елемента WEB-сторінки з іншим елементом, наприклад блоком, секцією, сторінкою тощо. У якості

гіперпосилання може використовуватися як фрагмент тексту, так і графічний об’єкт, а сам зв’язок можна встановлювати як між елементами одного сайту, так і між об’єктами, що розміщені на різних сайтах в мережі Інтернет [71].

Мова HTML передбачає використання спеціальних конструкцій коду та їх модифікацій, які називаються тегами та атрибутами для розмітки сторінки WEB-сайту. Дані теги вказують у якому вигляді та де буде виведено текстовий чи інший елемент у вікні браузера. Варто зауважити, сьогодні, сучасні браузери мають підтримку відображення усіх можливих тегів HTML п’ятої редакції (HTML5). Однак, як показує практика, на більшості підприємств досі використовуються браузери версій, які можуть не підтримувати нові стандарти тегів, а тому, якщо певного тегу немає у бібліотеці того чи іншого WEB-переглядача, він його просто ігнорує.

Гіпертекстова мова розміти HTML є мовою що інтерпретується, тобто для виконання написаного коду, його не потрібно окремо компілювати сторонніми засобами. Інтерпретатор мови HTML інтегровано у браузер, а тому WEB-сторінка компілюється безпосередньо під час відкривання її користувачем. Якщо під час компіляції було виявлено помилку, браузер, як правило, не повідомляє про це, а просто пропускає фрагмент який містить помилку і «намагається» далі завантажити сторінку [23].

Теги у мові HTML складаються з лівої відкриваючої дужки «<», назви тегу з параметрами атрибутів або без них, та закриваючої правої дужки «>» (рис. 2.8).

Рисунок 2.8. – **Структура HTML тегів**

Джерело: [11, 23]

Найпростіший WEB-сайт повинен містити мінімум три теги для відображення базової структури WEB-сторінки браузером, зокрема це теги

<HTML>, <HEAD> і <BODY> (рис. 2.9).

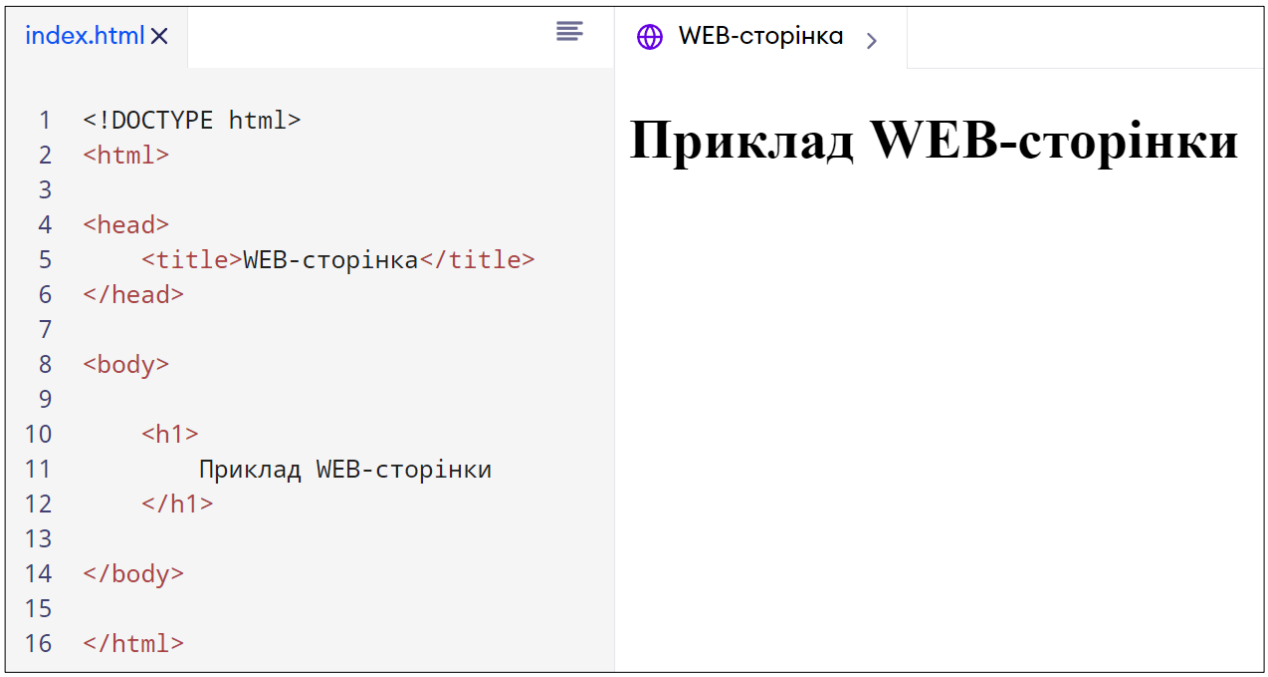


Рисунок 2.9. – **Структура найпростішої WEB-сторінки**

Джерело: [11, 24]

Тег <HTML> свідчить про те, що вся інформація, яка міститься у документі – є гіпертекстовою розміткою, тобто WEB-сторінкою. Це є найголовнішим тегом, який містить всі інші теги, конструкції, атрибути, параметри тощо.

Наступний тег <HEAD> містить у собі службову інформацію, яка необхідна для того, щоб браузер міг «зрозуміти», що саме та як йому потрібно відобразити. Ця інформація не виводиться на екран, вона виступає у якості інструкцій, яка містить посилання на файли скриптів, стилів та мета-тегів.

Тег <BODY> – це контейнер, що містить інформацію для виводу на сторінку. Все, що міститься у цьому контейнері зчитується і відображається WEB-переглядачем. Інакше кажучи, між тегами <BODY> і </BODY> зберігається весь зміст сторінки. Варто зауважити, теги HTML, HEAD і BODY можуть бути використані лише один раз.

Варто зауважити, що в рамках даної роботи, для написання HTML-розмітки нами використовувався препроцесор PUG (JADE), який написаний на JavaScript. Особливістю PUG – є його зберігати фрагменти HTML-сторінки у різних файлах та підключати їх за необхідності до відповідного шаблону.

Синтаксис на принцип роботи препроцесора PUG наведено на рис. 2.10.

```
doctype html
html(lang="ru-RU")
    block head
        include ./_includes/_head/_head.pug
    body
        .wrapper
            .maincontent
                block header
                    include ./_includes/_layout/_header.pug
                main
                    block main
                        include ./_includes/_layout/_main.pug
            block footer
                include ./_includes/_layout/_footer.pug
        block script
            include ./_includes/_layout/_script.pug
```

Рисунок 2.10. – **Приклад структури PUG-розмітки**

Джерело: авторська розробка

Для оформлення HTML сторінки використовується спеціальна мова стилів – каскадні таблиці стилів CSS, за допомогою яких налаштовують вигляд WEB-сторінки. Варто зауважити, що CSS не є мовою програмування чи навіть мовою розмітки, як HTML, це мова таблиці стилів, яка дозволяє застосовувати стилістичні властивості вибірково до будь-якого елемента на WEB-сторінці.

За допомогою CSS можна керувати кольором тексту, стилем шрифтів, налаштовувати фонові зображення або кольори на кожній окремій WEB-сторінці, відображення для різних пристроїв та розмірів екрану тощо. Серед особливостей каскадних таблиць стилів CSS варто виділити можливість відображати один і той же документ у різних стилях залежно від пристрою користувача, простоту подальшої зміни дизайну – необхідно лише змінити CSS-файл без внесення змін структуру документа, а також створення складного дизайну із використанням анімацій тощо. Інакше кажучи, CSS-файл можна розглядати як шаблон для форматування текстів, таблиць та інших елементів. CSS-файл можна імпортувати до різних WEB-сторінок, без яких-небудь обмежень по відношення до налаштувань серверів.

Усі CSS-правила складаються із селектора, що вказує який конкретний

елемент на сторінці підлягає оформленню, після чого, у фігурних дужка вказуються властивості і їх значення. Всередині блоку оголошень може знаходитися одне або декілька властивостей та їх параметрів, розділених крапкою з комою (рис. 2.11).

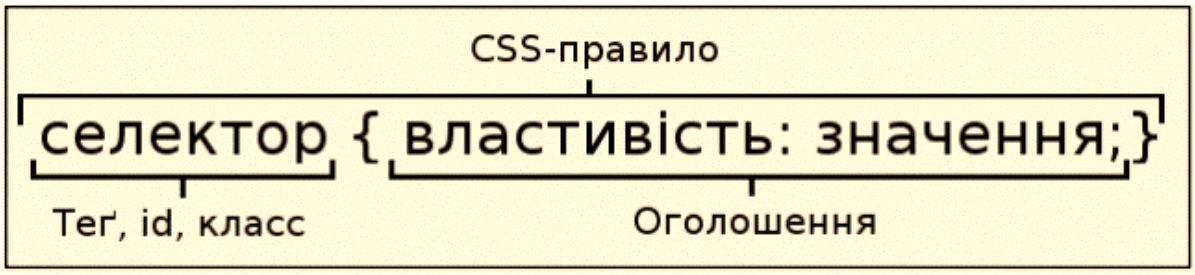


Рисунок 2.11. – **Структура CSS-правила**

Джерело: [11, 23, 24]

У якості селектора, зазвичай, використовується певний клас тегу або його ідентифікатор. У блоці оголошень відбуватиметься встановлення правил відображення обраних елементів, визначення їх властивості – розмір елемента, колір шрифту або блоку, рамок елемента, полів, позиціювання об'єкта на екрані тощо (рис. 2.12).

Рисунок 2.12. – **Принцип оголошення HTML-класів та CSS-властивостей**

Джерело: авторська розробка

Правила в CSS працюють по пріоритету їх написання, тобто правила будуть виконуватися по черговості їх написання. Зазначене дозволяє отримати прогнозований результат у випадку, коли до одного елемента застосовуються декілька правил оформлення зовнішнього вигляду.

В рамках даної роботи був обраний препроцесор для написання CSS-стилів – SCSS. Препроцесор SASS дає можливість використовувати такі функції та конструкції як [25]:

1) змінні – дозволяють зберігати інформацію, яку можна повторно застосовувати при написанні стилів, що значно підвищує продуктивність розробника та якість написаного коду;

2) вкладеність – дає можливість компонувати CSS-правила усередину інших;

3) імпорт – здатність підключати та імпортувати стилі з інших файлів;

4) шаблони – дозволяють використовувати фрагменти стилів у різних частинах таблиці стилів;

5) міксини – динамічні конструкції, в середині яких написані фрагменти стилів, які за допомогою параметрів можна швидко міняти (рис. 2.13).

```
@import "../../bower_components/app/assets/stylesheets";

@mixin inlineblock {
    display: inline-block;
    vertical-align: top;
}
```

Рисунок 2.13. – **Приклад оголошення SCSS-конструкцій**

Джерело: авторська розробка

Отже, CSS – це не класична, у своєму розумінні, мова написання коду – це технологія опису зовнішнього вигляду документів, яка дозволяє відтворити в життя будь-який сучасний дизайн WEB-сайту. Каскадні таблиці стилів CSS призначені для форматування текстових, табличних, графічних та інших елементів WEB-сторінки та є загальноприйнятим стандартом, який використовується WEB-розробниками.

Для надання сайтам елементів інтерактивності, в порівнянні зі звичайним статичними WEB-сторінками, застосовується мова програмування JavaScript. Вона використовується при реалізації динамічних сценарії для взаємодії з користувачем.

JavaScript – інтерпретована мова програмування з об’єктно-орієнтованою концепцією написання сценаріїв [23]. Як і мова HTML для компіляції коду на JavaScript не потрібне спеціальне програмне забезпечення – весь код

обробляється і виконується WEB-переглядачем.

Код на мові JavaScript називають скриптами [26]. Як правило, скрипти зберігаються в окремому файлі з розширенням *.js, а для того щоб його виконати, у HTML розмітці використовують спеціальний тег підключення. Окрім цього, для інтерпретації WEB-переглядачем JavaScript коду, він може бути інтегрований безпосередньо у HTML розмітку (рис. 2.14).

```
<!-- Під'єднання файлу JS сценаріїв  -->
<script defer="defer" src="js/script.js"></script>

<!-- Інтеграція JS сценаріїв  -->
<script>
  $('.container').appear(function() {
    $('.main').each(function() {
      $('.progress-main').css('width', function() {
        return ($(this).attr('data-percentage') + '%')
      });
    });
  }, {
    accY: -100
  });
</script>
```

Рисунок 2.14. – **Принцип під'єднання JavaScript сценаріїв**

Джерело: авторська розробка

Отже, мова програмування JavaScript дає можливість забезпечити динамічне оновлення вмісту WEB-сторінок; «спілкування» з користувачем через реакцію на дії останнього, наприклад на натискання клавіш миші або клавіатури; отримувати запити від користувачів; надсилати файли для завантажування тощо.

На практиці, процес розробки передбачає створення десятків файлів і каталогів, які лягають в основу бажаної структури, яка найкраще підходить для кожного окремого проєкту. У цьому випадку здійснюється багато часто повторюваних дій, таких як оновлення загального каталогу, сторінок для перегляду змін що вносяться, діагностика проєкту на наявність помилок тощо. Все це збільшує загальний час на розробку та тестування IT-продукту. Таким чином, з метою оптимізації робочого процесу, управління проєктом, забезпечення роботи препроцесорів PUG, SASS, додаткових плагінів тощо, нами використовувався спеціальний інструмент (менеджер) автоматизації проєктів

GULP (рис. 2.15).

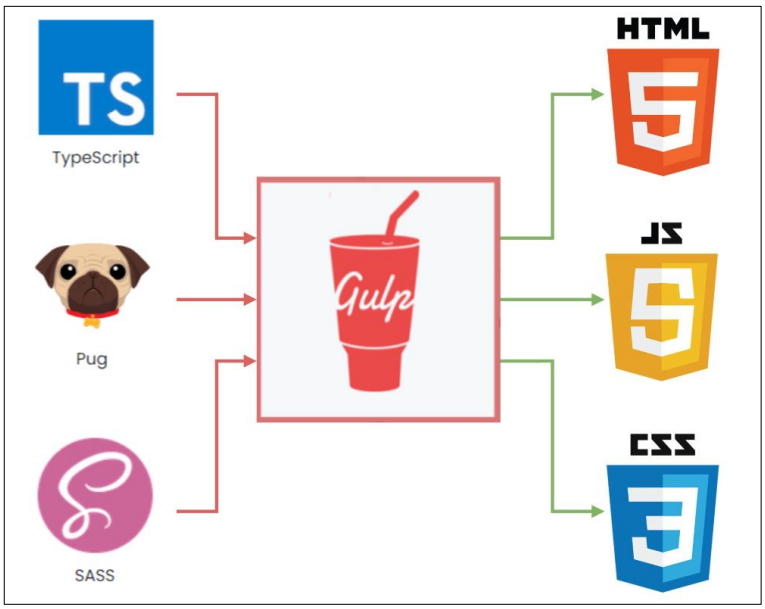


Рисунок 2.15. – **Структура GULP-проєкту**

Джерело: авторська розробка

GULP використовує лише код JavaScript та допомагає запускати зовнішні скрипти та великомасштабні веб-програми. Він створює системні автоматизовані завдання, такі як мінімізація CSS, HTML та JS; об'єднання бібліотечних файлів та компіляція файлів SASS, SCSS, PUG; оптимізація зображень для браузерів, автоматичне оновлення каталогу після внесення змін тощо.

2.3. Архітектура односторінкових WEB-сайтів

Будь-який сучасний WEB-сайт характеризується наявністю трьох основних функціональних блоків, серед яких модуль зберігання даних, модуль обробки інформації та модуль генерації відповідного представлення для відображення оброблених запитів користувачу. Розробка логічного макета стосовно того, як зазначені функціональні блоки та компоненти, представляють собою WEB-сайт та взаємодіють між собою, лягає в основу вибору типу архітектури планування та проектування технічних особливостей WEB-ресурсу до його розробки та розгортання.

Архітектура WEB-сайту дає можливість визначити основні та допоміжні

елементи системи, сформувати логічну модель взаємозв'язку компонентів сайту (типи зв'язків, організацію захисту інформації), описати підходи до кодування (вибір мови програмування, типу автоматизації проєктів), визначити технічне та програмне забезпечення (серверне обладнання, сховище, типи зв'язків).

Сьогодні найбільш поширеною архітектурою розробки IT-продуктів будь-якої складності є клієнт-серверна архітектура. Власне, даний тип архітектури нами був обраний для розробки односторінкового WEB-сайту для підтримки діяльності підприємств. Базова схема роботи клієнт-серверної архітектри наведена на рис. 2.16.

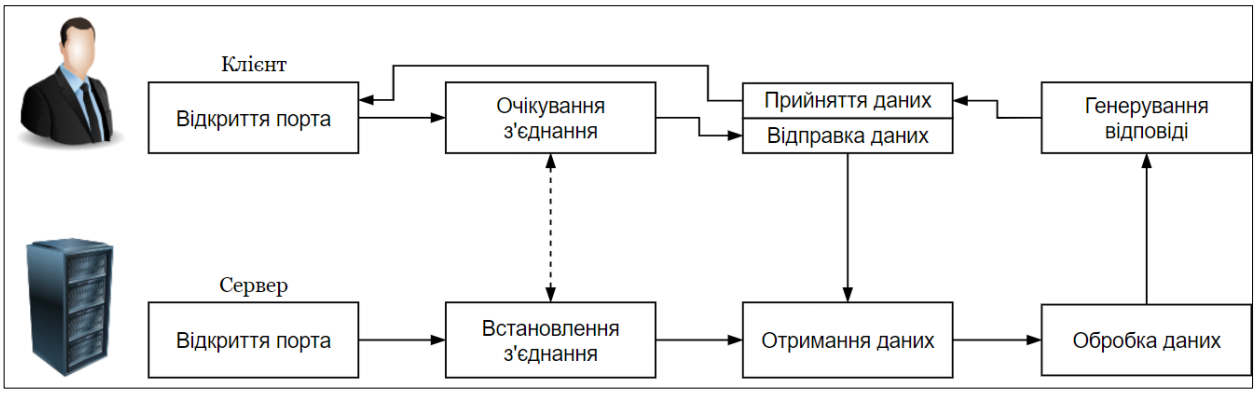


Рисунок 2.16. – **Узагальнена схема роботи клієнт-серверної архітектури**

Джерело: авторська розробка

За даної архітектури функціонування WEB-сайту відбувається за схемою «запит-відповідь». Модель функціонування умовно можна зобразити у наступній послідовності: 1) користувач формує і відправляє запит на сервер; 2) сервер отримує і обробляє запит; 3) генерується відповідь на запит і готовий результат відправляється назад користувачу.

Модель клієнт-серверної архітектури передбачає наявність двох ключових компонентів, зокрема клієнту та сервера. Під клієнтом можна розуміти персональний комп'ютер, смартфон чи інший пристрій на стороні користувача, який відправляє запит до сервера з метою отримання інформації або виконання певних дій. У класичному розумінні, для того, щоб користувач побачив графічний інтерфейс у вікні браузера, останній повинен обробити отриману відповідь від WEB-сервера, в якому буде міститися інформація, реалізована із

застосуванням HTML, CSS, JS. Зазначені технології, власне і «надають» інструкції, як браузеру потрібно відобразити все, що він отримав у відповідь на запит клієнта.

WEB-сервер – це сервер, який приймає HTTP-запити від клієнтів та повертає користувачам HTTP-відповіді. WEB-сервер – це комплекс програмних засобів та технічного обладнання, який призначене для:

1) обробки програмних кодів;

2) забезпечення виконання сервісних функцій;

3) надання користувачам доступу до певних ресурсів;

4) підтримки CRUD операцій тощо.

Дворівнева архітектура забезпечує формує середовище «клієнт-сервер», яке направлене на зберігання інтерфейсу користувача в клієнтській системі, а вся база даних зберігається на серверній робочій станції. Бізнес-логіка та принципи роботи бази даних, у свою чергу, існують на клієнтській стороні. Особливістю клієнт-серверною архітектури є направленість запитів під час комунікації. Клієнт призначений для запитів, а сервер для реагування на запити та генерацію відповіді для клієнта. Сервер в цьому випадку є централізованою машиною, від якої залежать клієнти [27]. Комунікація між клієнтом та сервером відбувається через мережу Інтернет (рис. 2.17).

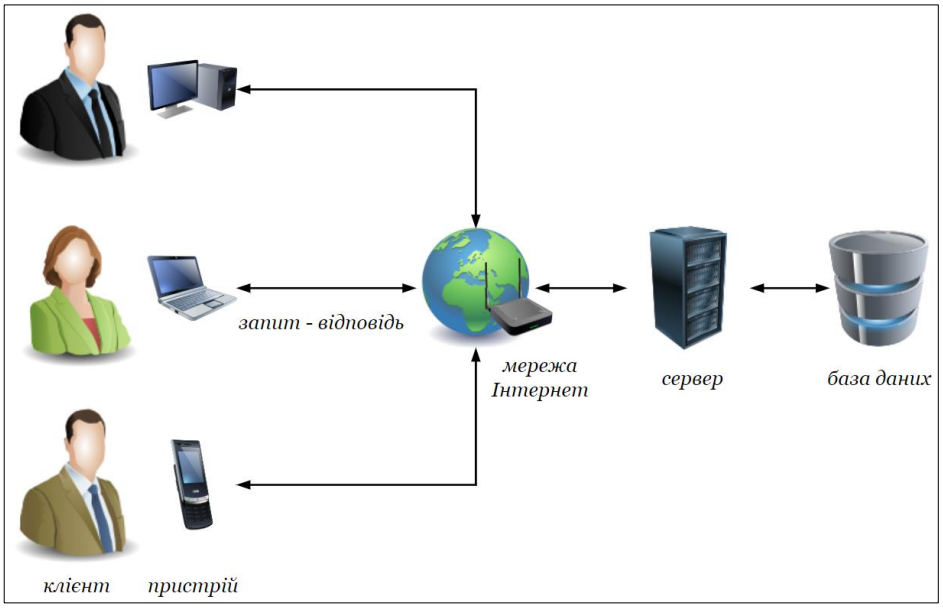


Рисунок 2.17. – **Структура дворівневої клієнт-серверної архітектури**

Джерело: [28]

Таким чином, традиційна клієнт-серверна архітектура має два рівні – клієнт та сервер: клієнт представляє собою користувацький інтерфейс, а сервер відповідає за отримання запитів і відправку відповідей клієнту, використовуючи при цьому лише власні ресурси. Сервер може обслуговувати кілька клієнтів одночасно [29]. Якщо одночасно приходить більше одного запиту, то вони встановлюються в чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети, в такому випадку запити з більш високими пріоритетами повинні виконуватися раніше.

Дворівневу клієнт-серверну архітектуру, нерідко розширюють до трирівневих систем, за рахунок встановлення додаткового обладнання для зниження навантаження на сервер, шляхом розподілу виконання операцій на декілька потоків. Як правило, ця тип архітектури складається з наступних компонентів: представлення даних – для відображення інтерфейсу користувачу; прикладний компонент – сервер додатків; керування ресурсами – сервер бази даних, який надає генерує відповідь у вигляді шуканої інформації (рис. 2.18).

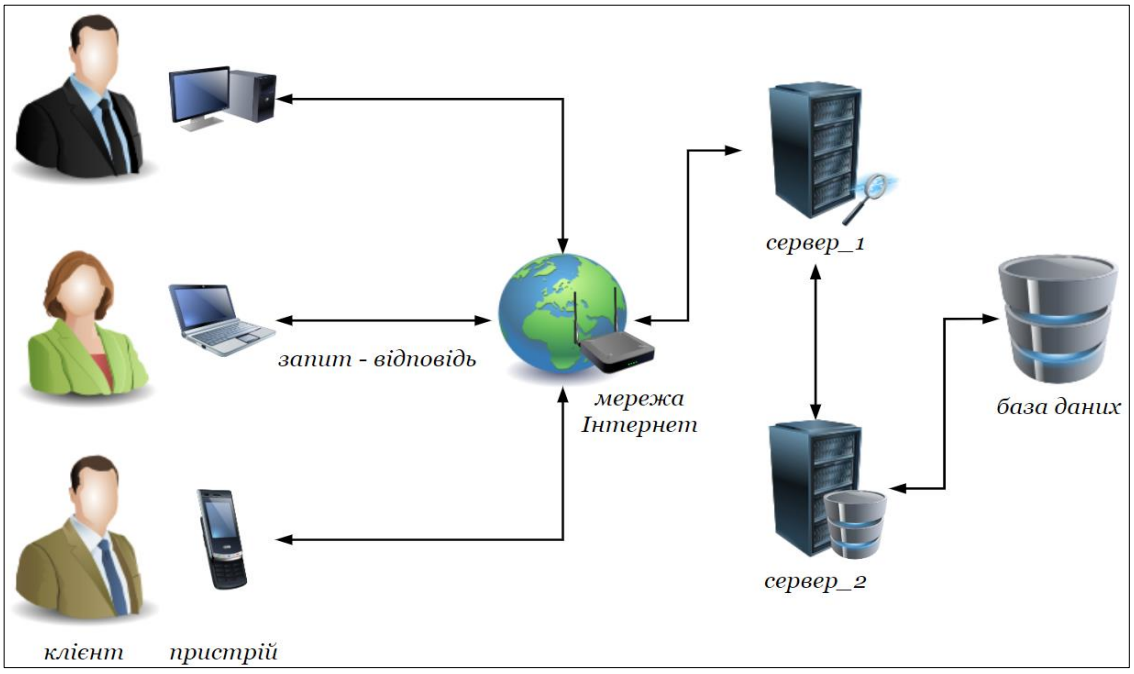


Рисунок 2.18. – **Структура трирівневої клієнт-серверної архітектури**

Джерело: [28, 30]

Трирівнева клієнт-серверна архітектура функціонує у Intranet та Internet мережах. Уся програмна логіка винесена на сторону сервера, що забезпечує

формування запитів до бази даних, які передаються відповідному серверу для їх обробки та виконання [30]. Даний тип клієнт-серверної архітектури лежить в основі розробки великих сайтів, WEB-порталів та інших інтернет-ресурсів, в яких використовуються реляційні бази даних. Наприклад, Internet-магазини (Rozetka.ua), пошукові системи (Google, Bing, Yahoo), системи для Internet-телефонії й обміну повідомленнями в реальному часі (Google Meet, Telegram, Viber), сервіси передавання відео через мережу Internet (YouTube, Twitch).

Коли програмний продукт призначений для використання у сферах з великим навантаженням на систему, наприклад розробка багато користувацьких мереж, розгортання хмарних програмних засобів, забезпечення роботи віртуальних робочих станцій тощо використовується розширена версія трирівневої системи – багаторівнева клієнт-серверна архітектура. Це надзвичайно потужне як в технічному, так і програмному сенсі обладнання, яке відрізняється від вище наведених типів більшою кількістю встановлених серверів для оброки запитів в багатопотоковому режимі (рис. 2.19).

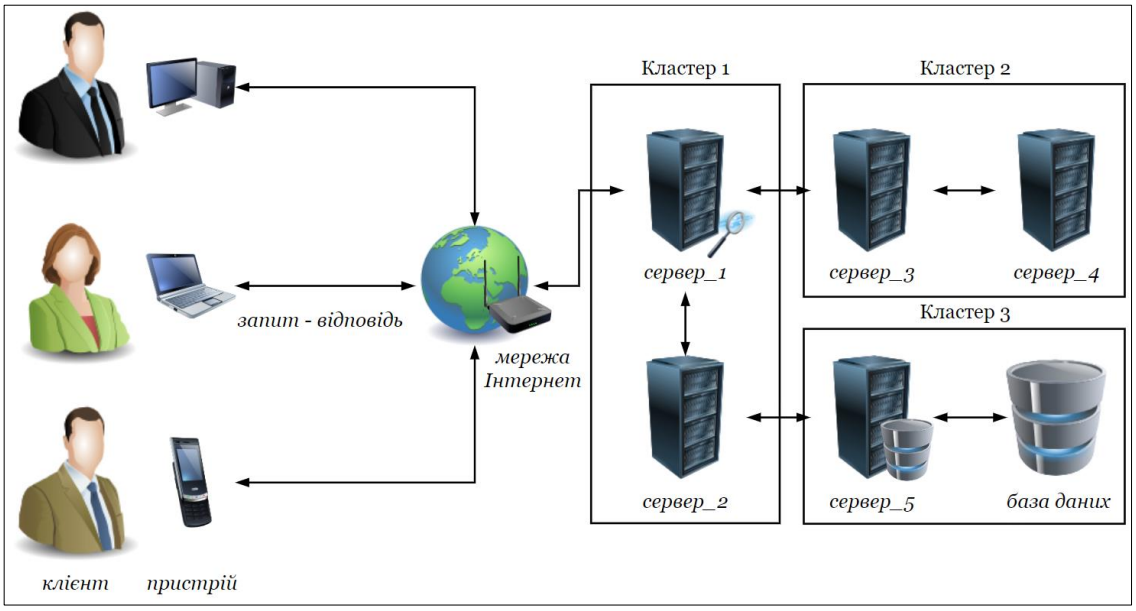


Рисунок 2.19. – **Структура багаторівневої клієнт-серверної архітектури**

Джерело: [28, 30]

Прикладом WEB-ресурсів, які функціонують на такій архітектурі є сервіси для розгортання хмарного середовища розробки (Microsoft Azure), онлайн-офіси (Microsoft Office 365), сервіси для автоматизації бізнес-процесів (Бітрікс24),

хмарні сервіси для зберігання і синхронізації даних (Dropbox, Google Drive, OneDrive, Mega).

Модель клієнт-серверної взаємодії визначається перш за все розподілом обов'язків між клієнтом та сервером. Можна відокремити три рівні операцій (рис. 2.20):

Рисунок 2.20. – **Схема запиту і генерації WEB-сторінки**

Джерело: [28, 29, 30]

Окрім вищевикладеного, варто навести функції, які виконуються в рамках клієнт-серверної архітектурі додатків (табл. 2.1).

Таблиця 2.1

Функції клієнт-серверної архітектури

Клієнт	Сервер
керування користувальницьким інтерфейсом	відображення клієнтського інтерфейсу
формування запиту до сервера і його написання	зберігання, доступ, захист і резервне копіювання даних
отримання результатів запиту	обробка клієнтського запиту
відправка додаткових команд (додавання, оновлення або видалення даних)	відправлення результату (відповіді) клієнту

Джерело: [29]

В залежності від того, які функції розподіляються між елементами архітектури, розрізняють модель «тонкого» та «товстого» клієнта (рис. 2.21, 2.22). Тонкий клієнт – це тип клієнта, який перенаправляє завдання з обробки даних на сервер, не використовуючи свої обчислювальні можливості для їх

реалізації. Обчислювальні ресурси такого клієнта дуже обмежені, їх повинно бути досить лише для запуску і відображення WEB-інтерфейсу.

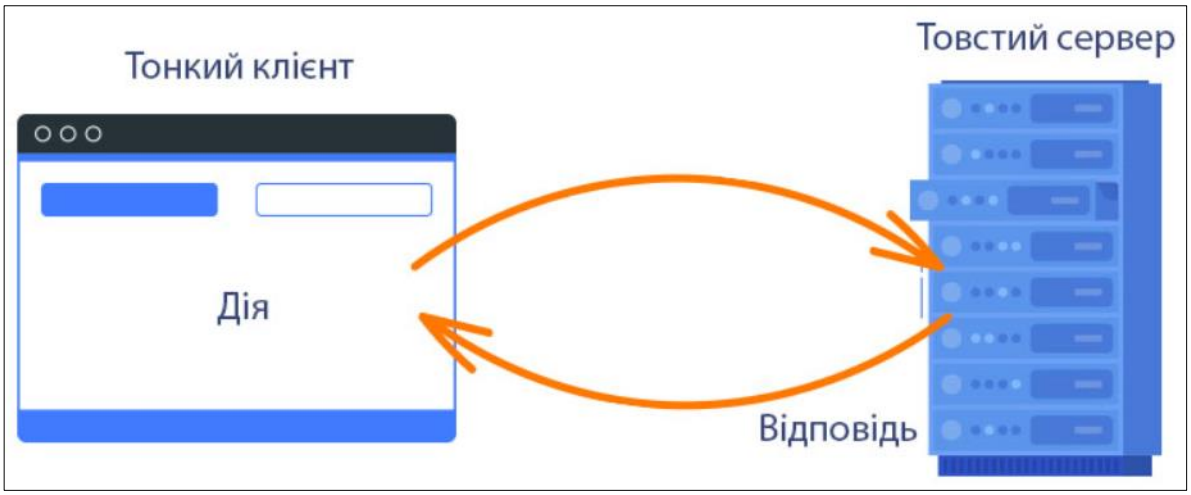


Рисунок 2.21. – **Модель «тонкого» клієнта**

Джерело: [28-31]

У якості «тонких» клієнтів можна вважати усі пристрої користувачів із встановленим WEB-браузером, які використовується для роботи з WEB-ресурсами. «Товстий» клієнт, у свою чергу, це клієнт, який виконує запити користувачів незалежно від потужностей центрального сервера. Сервер може використовуватися як сховище даних, обробка та отримання яких лягає пристрій користувача.

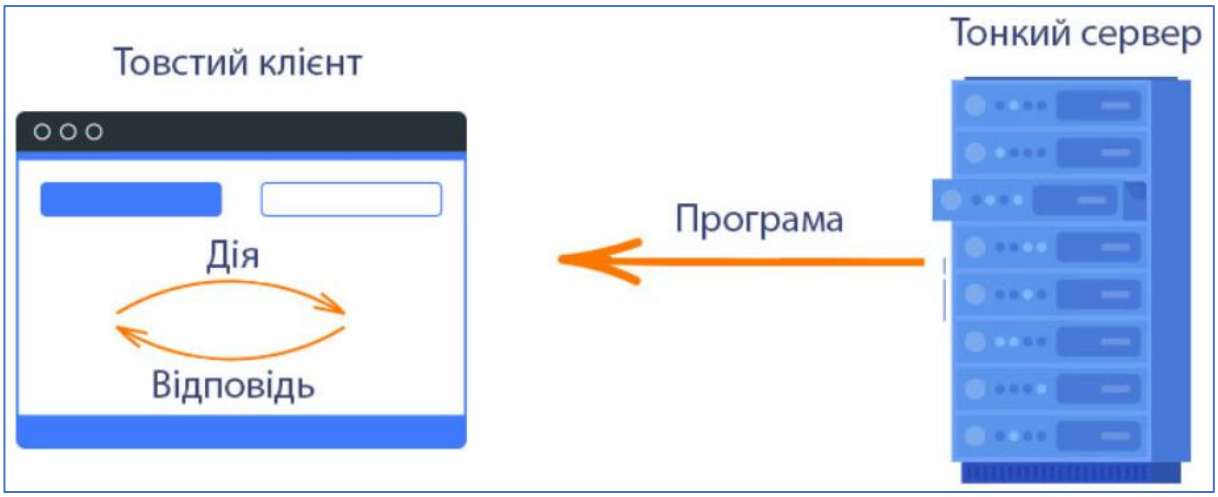


Рисунок 2.22. – **Модель «товстого» клієнта**

Джерело: [28-31]

Прикладом «товстих» клієнтів можна вважати робочі станції або

персональні комп’ютери, які працюють під управлінням спеціальної операційної системи, що мають необхідний набір програмного забезпечення для реалізації завдань користувача.

Таким чином, основною відмінністю між «тонкими» і «товстими» клієнтами є спосіб обробки інформації. Товстий клієнт здійснює роботу з даними використовуючи свої апаратні і програмні можливості, а з сервером пов’язується лише для отримання даних з бази. Тонкі клієнти, в свою чергу, використовують потужності центрального сервера для обробки інформації, надаючи лише необхідний інтерфейс для роботи користувача. Як правило, у більшості випадків, користувач навіть не помічає, з яким типом архітектури він працює, а тому, обираючи типу клієнта для проєкту, необхідно враховувати його технічні особливості та логіку взаємодії з кінцевим споживачем.

Коли WEB-сервер отримує запит від користувача на відображення WEB-сторінки з певною інформацією, сервер починає пошук відповідного каталогу, зчитує код, що знаходиться на шуканій сторінці і передає її спеціальній програмі, яка формує кінцевий вигляд сторінки у відповідності до запиту користувача, і лише після цього згенерована сторінка відправляється клієнту. Схема запиту і генерації WEB-сторінки наведена на рис. 2.23.

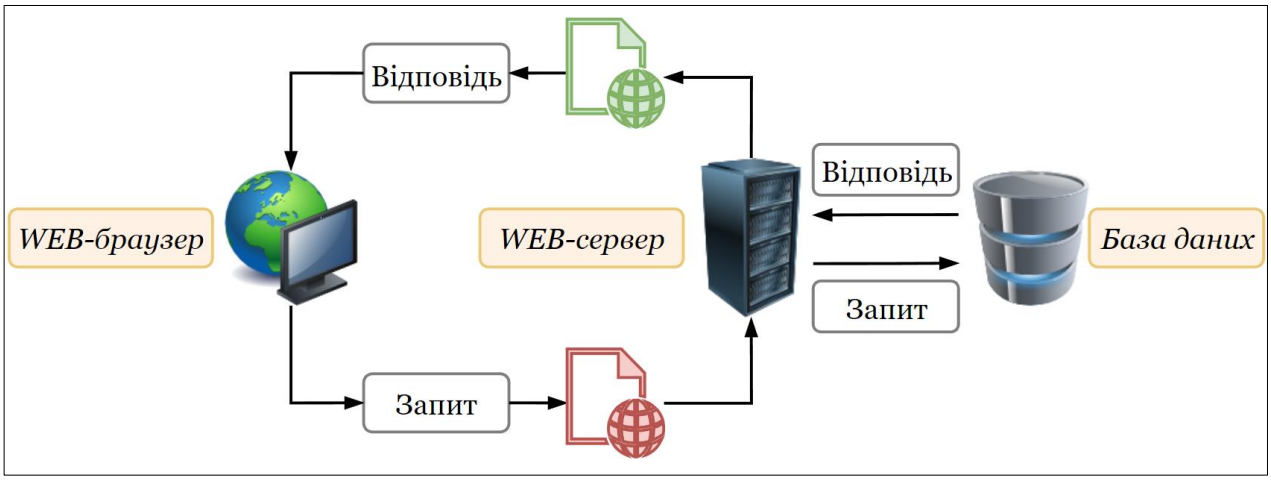


Рисунок 2.23. – **Схема запиту і генерації WEB-сторінки**

Джерело: [28, 30]

По своїй суті, база даних – це інформаційна модель, яка дозволяє впорядковано зберігати дані про об’єкт або групу об’єктів, які мають набір

властивостей, які можна привести до спільної категорії. Фактично, база даних не є частиною WEB-сервера, але більшість програм, в силу їх функціоналу, не можуть виконувати покладені на них задачі без під’єднання до неї, оскільки саме в базі даних зберігається вся динамічна інформація програми (облікові дані користувача, адміністратора тощо).

Варто зауважити, що у результаті всіх описаних операцій генерується статична WEB-сторінка, яка передається WEB-сервером клієнтському браузеру. Весь функціонал динамічного складу виконується на стороні сервера, а клієнту надсилається кінцева сторінка, яка містять лише HTML-код. створюються безпосередньо в процесі обробки запиту від користувача.

На прикладі односторінкових WEB-сайтів, клієнт-серверної архітектура дає можливість динамічною змінювати наповнення сайту; під’єднувати та використовувати чати для онлайн-спілкування між користувачами; збирати, зберігати та обробляти дані про відвідування сайту, наприклад час перебування користувача на сайті, на якій секції сайту користувач затримався найдовше, скільки разів один і той самий користувач повертався на сайт тощо; використовувати онлайн-форми для збирання реєстраційних даних користувачів; формування реєстру замовлень тощо.

Отже, ідея клієнт-серверної архітектури полягає у поділі WEB-сторінки на декілька компонентів, кожен з яких виконує специфічний набір функцій. Компоненти можуть виконуватися на різних комп’ютерах, виконуючи серверні або клієнтські функції. Це дозволяє підвищити надійність, безпеку і продуктивність мережевих додатків і мережі в цілому.

РОЗДІЛ 3
РОЗРОБКА ОДНОСТОРІНКОВОГО WEB-САЙТУ

3.1. Моделювання роботи односторінкового WEB-сайту для підтримки діяльності підприємств

Одним з основаних етапів розробки односторінкового WEB-сайту є моделювання процесу його роботи, зокрема представлення динаміки зміни станів компонентів сайту, опис функцій учасників проєкту, а також деталізація алгоритму надходження та обробки запитів. Для цього, нами було використано уніфіковану мову моделювання (UML), як універсального інструмента для візуалізації, специфікації, конструювання та документування інформаційних систем. Діаграми у нотації UML – це потужний, стандартизований інструмент у сфері розробки програмного забезпечення, моделювання IT-інфраструктури та бізнес-систем [32]. За допомогою цих діаграм формується схема додатку, за якою розробники пишуть програмний код. В рамках даної роботи, нами було виконано моделювання роботи односторінкового WEB-сайту за допомогою діаграми варіантів використання, діаграми послідовності та діаграми станів.

На нульовому рівні процес роботи односторінкового WEB-сайту розглядається, як блок із всіма відповідними вхідними і вихідними робочими та керуючими зв’язками (рис. 3.1).

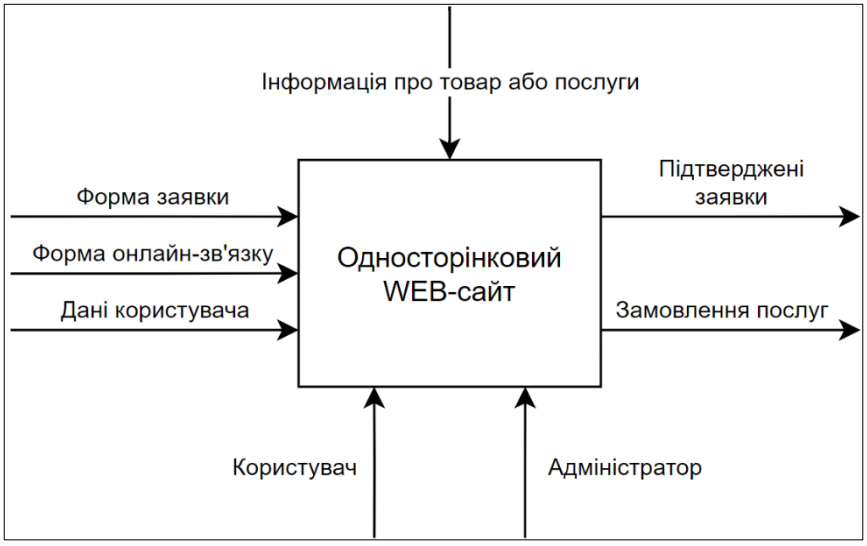


Рисунок 3.1. – **Контекстна діаграма роботи односторінкового WEB-сайту**

Джерело: авторська розробка

Згідно наведеної діаграми формується загальне уявлення про вхідні так вихідні атрибути системи, такі як кількість та вид форм зв’язку, тип користувачів системою, вид ресурсу який надає WEB-сайт та що є результатом роботи WEB-сайту. Варто зауважити на тому, що в даній роботі створення бази даних не передбачається завданням, однак для загального розуміння роботи системи WEB-сайту, діаграми з описом зв’язків зі сховищем даних можуть бути корисними для майбутнього масштабування WEB-ресурсу.

Контекстна діаграма роботи односторінкового WEB-сайту описує процес функціонування системи на нульовому рівні, тобто на такому, який відображає взаємодію усіх інших складових і процесів що протікають у системі. Таким чином, для наочного відображення роботи сайту, доцільно виконати декомпозицію функціонального блоку нульового рівня на набір взаємозалежних складових, які представлені на рис. 3.2.

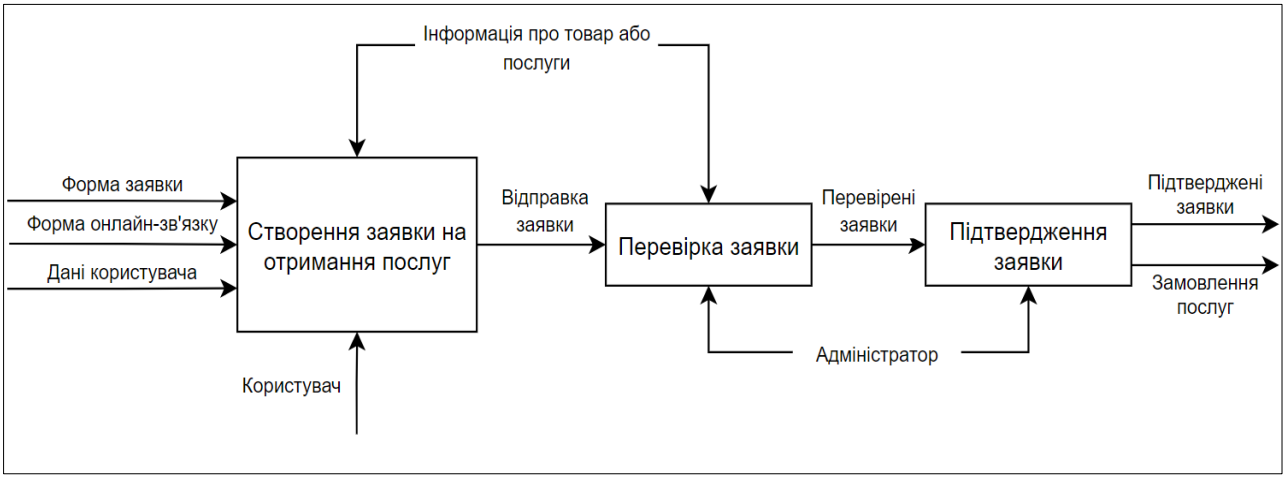


Рисунок 3.2. – **Декомпозиція діаграми функціонування односторінкового WEB-сайту**

Джерело: авторська розробка

З наведеної декомпозиції ми можемо бачити процес функціонування роботи сайту у розрізі його ключових складових та підфункцій, зокрема створення заявки на пропоновані підприємством послуги, перевірка заявки, підтвердження заявки та замовлення послуг. Таким чином, дослідивши роботу сайту можна виділити два глобальні типи користувачів, зокрема авторизованого та неавторизованого користувача (табл. 3.1).

Таблиця 3.1

Типи користувачів WEB-сайтом

Тип користувача	Короткий опис
Авторизований	Має можливість редагувати керувати інформаціє на сайті та обробляти запити не авторизованих користувачів
Не авторизований	Має можливість переглядати інформацію на сайті, оформляти замовлення та зв'язуватися з менеджером для отримання консультації

Джерело: авторська розробка

В рамках зазначеного, можна сформувати комунікативну екосистему нашого односторінкового WEB-сайту із трьох типів осіб-акторів, зокрема користувача, адміністратора та бази даних (табл. 3.2).

Таблиця 3.2

Колекція акторів односторінкового та їх опис

Актор	Сутність актора	Опис актора
Користувач	Внутрішня система	Користувач WEB-сайту, який має права «гість»
Адміністратор		Користувач WEB-сайту, який має права «адміністратор»
База даних	Зовнішня система	База даних – сховище інформації

Джерело: авторська розробка

На основі вищенаведеного, сформуємо ієрархічне дерево користувачів WEB-сайту ґрунтуючись на їх функціях та приналежності до системи (рис. 3.3).

Рисунок 3.3. – **Ієрархічне дерево користувачів WEB-сайту**

Джерело: авторська розробка

Розмежування користувачів за правами дає можливість налаштувати рівень доступу до операцій, які дозволені відвідувачам, а також, що є більш важливим з точки зору забезпечення роботи серверного компоненту – це

відмежувати програмний код на окремі частини, кожна з яких відповідає за виконання покладених на неї функцій. Мова йде про CRUD операції – створення (create), зчитування (read), оновлення (update) та видалення (delete) записів бази даних.

Для забезпечення роботи даних операцій, у виконуваному файли, наприклад index.php, якщо серверна логіка реалізована на мові PHP, або main.js – якщо на Node.js передбачається використання чотирьох HTTP-методів:

1) для POST, тобто для запису даних у базу даних, буде використовуватися метод create();

2) для GET, тобто для зчитування даних з бази даних, буде виконуватися метод read();

3) для PUT, тобто для оновлення раніше записаних даних у базу даних, буде виконуватися метод update();

4) для DELETE, тобто для видалення певних даних із бази даних, буде виконуватися метод delete().

Залежно від того, які права має користувач, йому доступні ті чи інші операції. В рамках нашого сайту передбачається наступна система розподілу прав та доступів до операцій користувачів на сайті (рис. 3.4):

Рисунок 3.4. – **Операції користувачів на сайті**

Джерело: авторська розробка

Вище наведені моделі дають базове уявлення про те, які користувачі

передбачені системою, однак для глибшого розуміння їх взаємодії між собою та графічного представлення цього, на практиці застосовуються діаграми варіантів використання. Діаграма варіантів використання – це найпростіша із поведінкових діаграм UML, та найкорисніша при висвітленні функціональних особливостей програми. Вона використовується з метою опису функціональних вимог до програмного продукту, що формує базове представлення того, як програма буде використовуватися [33].

Діаграма варіантів використання є основним видом діаграм при моделюванні поведінки систем, у тому числі WEB-ресурсів. В UML, діаграми варіантів використання застосовуються з метою візуалізації поведінки системи у розрізі документування процесу формування запитів та їх обробки користувачами, аби розробник розумів, як реалізувати функціонал того чи іншого елемента [34].

Таким чином, представимо візуальну інтерпретацію зв'язків між акторами односторінкового WEB-сайту (рис. 3.5).

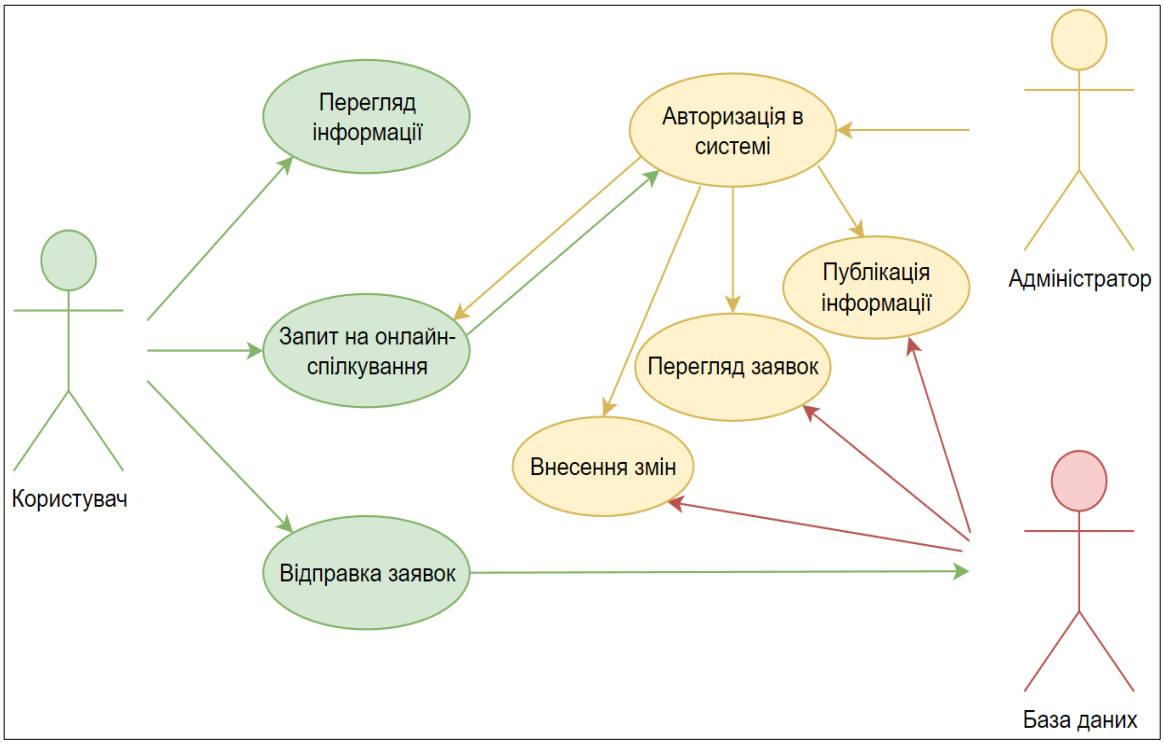


Рисунок 3.5. – **Діаграма варіантів використання WEB-сайту**

Джерело: авторська розробка

Розглянемо діаграму варіантів використання більш детально. Користувач,

потрапивши на сайт може без обмежень переглядати інформацію, викликати форму для онлайн-спілкування з менеджером, а також відправляти заявки на замовлення товарів чи послуг. Користувачу надаються права «гість», згідно з якими, він може користуватися інформацію, яку йому надає користувач з правами «адміністратор».

Адміністратор, у свою чергу, може переглядати заявки від користувачів, вносити зміни до реєстраційних даних (редагувати, видаляти, оновлювати), керувати контентом на сайті. Окрім цього, адміністратор може відстежувати зміни на сайті, та самотужки пропонувати клієнту допомогу через форму онлайн-спілкування. Однак, варто зауважити, що для користування цим функціоналом необхідно пройти етап «Авторизація в системі». Власне, даний етап і розмежовує права «гість» і «адміністратор», зокрема визначає чи користувач має доступ до закритих функцій системи. Разом з цим, авторизація дає доступ до сховища бази даних, де зберігаються усі заявки користувачів, інформація для відображення на сайті, реєстраційні даній клієнтів тощо. Без авторизації в систему і під'єднання до бази даних, адміністратор не має можливості виконувати свої керівні функції.

Наведена вище діаграма дає лише уяву про те, що саме виконується на сайті, зокрема який функціонал здійснюється з точки зору користувача. Для того, аби деталізувати поведінкові аспекти системи, а саме зобразити як взаємодіють користувачі (актори з діаграми варіантів використання) з іншими компонентами WEB-сайту під час реалізації тих чи інших варіантів використання програми, та як при цьому взаємодіють інші компоненти програмної системи використовується діаграма послідовності.

У вузькому розумінні, діаграма послідовності відображає принцип взаємодії елементів системи у динаміці, тобто демонструє часові особливості передачі, прийому та обробки повідомлень об'єктів системи. Діаграми послідовності використовуються для уточнення діаграми варіантів використання через формування опису із зображенням вхідних запитів і результатів обробки цих запитів. Вони є одним із способів формалізації сценаріїв користування компонентами системи, шляхом документування користування функціоналом

системи з точки зору набору сценаріїв. На діаграмі послідовностей відображаються лише ті об’єкти, які безпосередньо беруть участь у виконанні прописаних сценаріїв. Ці об’єкти супроводжуються блоками з повідомленнями та результатами які генеруються в процесу роботи системи.

Перевагою цих діаграм є можливість формування колекції взаємодіючих компонентів та описати потік інформації від одних компонентів до інших. Зазначені компоненти та потоки інформації, в рамках термінології мови програмування JavaScript, будуть трансформовані в конкретні класи та методи цих об’єктів [35]. Таким чином, формується модель системи функціонування, а також тип даних, які будуть підтримуватися та оброблятися системою. З огляду на діаграму варіантів використання, пропонуємо деталізувати декілька з них, зокрема процес перегляду інформації, авторизації в системі, перегляд заявок та внесення змін.

Діаграма послідовності процесу перегляду інформації користувачем наведена на рис. 3.6.

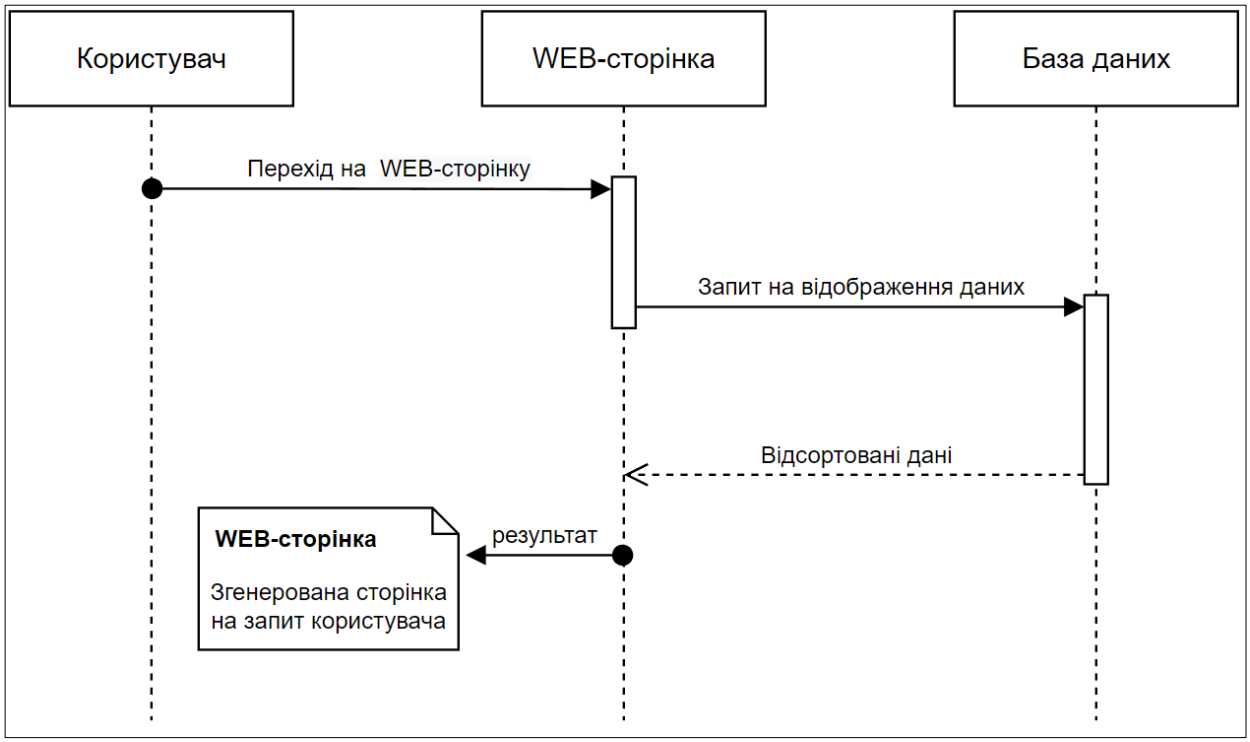


Рисунок 3.6. – **Діаграма послідовності перегляду інформації**

Джерело: авторська розробка

З діаграми можна бачити як відбувається відображення інформації

користувачу. Зокрема, надсилається запит на сервер щодо відображення актуальної версії WEB-сторінки, на сервері відбувається пошук необхідних даних, і, як результат – генерується WEB-сторінка для перегляданння.

Як зазначалося вище, для керування контентом на сайті, зокрема для додавання, редагування та видалення інформації, а також приймання й обробки заявок від користувачів передбачається користувач з правами «адміністратор». Таким чином, забезпечення реалізації модуля з функціоналом авторизації користувачів та надання їм прав «адміністратор» є одним із основних етапів розробки WEB-сайту. Для авторизації користувача та надання йому прав керування сайтом полягає у порівнянні введених персональних даних користувачем із тими, що існують у базі даних. При успішній верифікації даних, в базу даних записуються дані про час авторизації та відкривається панель адміністрування контентом. Алгоритм простої авторизації користувача наведено на рис. 3.7.

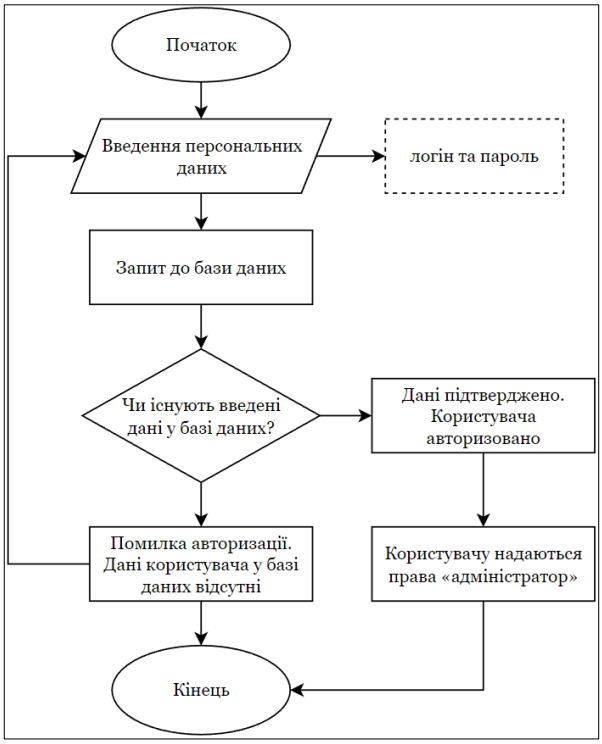


Рисунок 3.7. – **Блок-схема авторизації користувача**

Джерело: авторська розробка

Наведений алгоритм демонструє підхід до так званої «простої» авторизації, яка полягає у порівняні введеного логіну та паролю користувача.

При підтвердженні введених даних на стороні серверу, реєстраційна форма закривається і модуль авторизації більше не доступний. У випадку відсутності даних у базі даних, реєстраційна форма може бути відкрита повторно для додаткової спроби авторизації.

Діаграма послідовності процесу авторизації користувача в системі наведена на рис. 3.8.

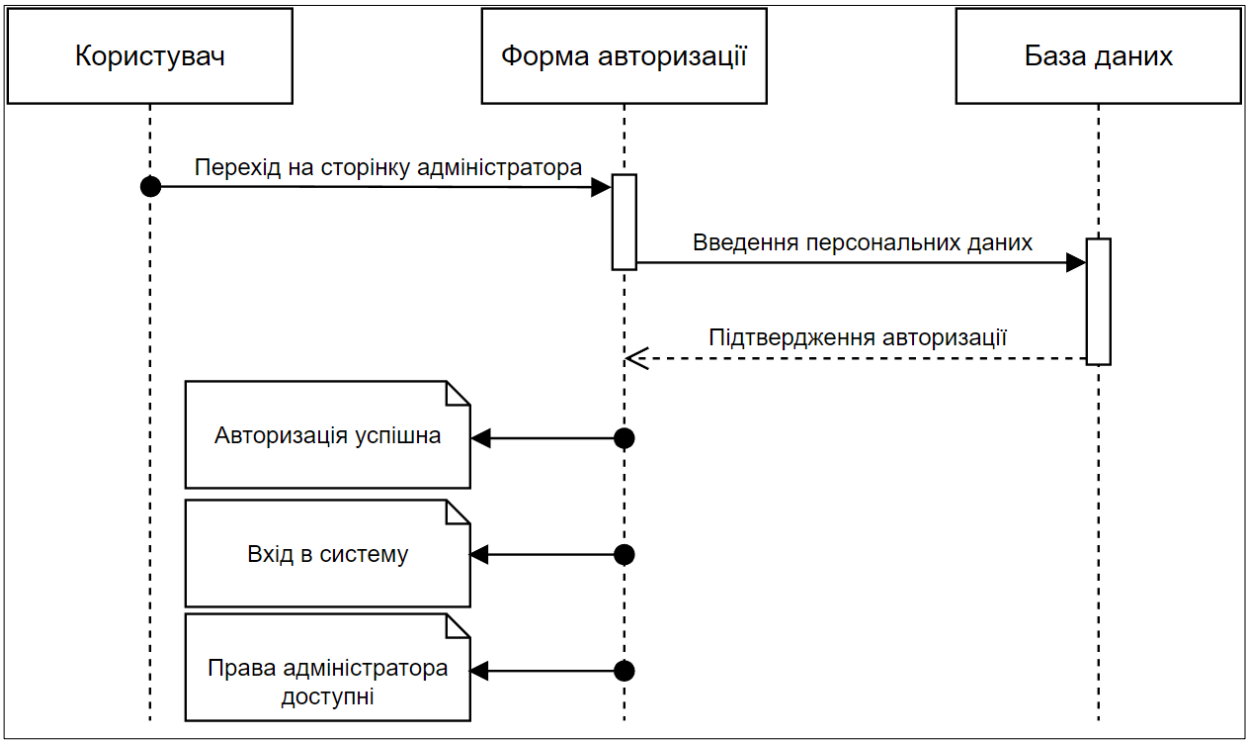


Рисунок 3.8. – **Діаграма послідовності процесу авторизації в системі**

Джерело: авторська розробка

Наведена діаграма демонструє процес надання користувачу прав «адміністратор». В якості механізму верифікації користувача виступає форма авторизації, через яку введені персональні дані останнього порівнюються з даними у базі даних. В результаті перевірки, користувачу надаються права адміністрування інформаційним ресурсом.

Наступною до розгляду пропонується діаграма послідовності процесу перегляду заявок надісланих від користувачів (рис. 3.9). Реалізація процесу перегляду заявок передбачає наявність компоненту, функціонал якого полягає в опрацюванні сервером запиту адміністратора на отримання списку записаних в базу даних заявок та сортування їх згідно з певними параметрами.

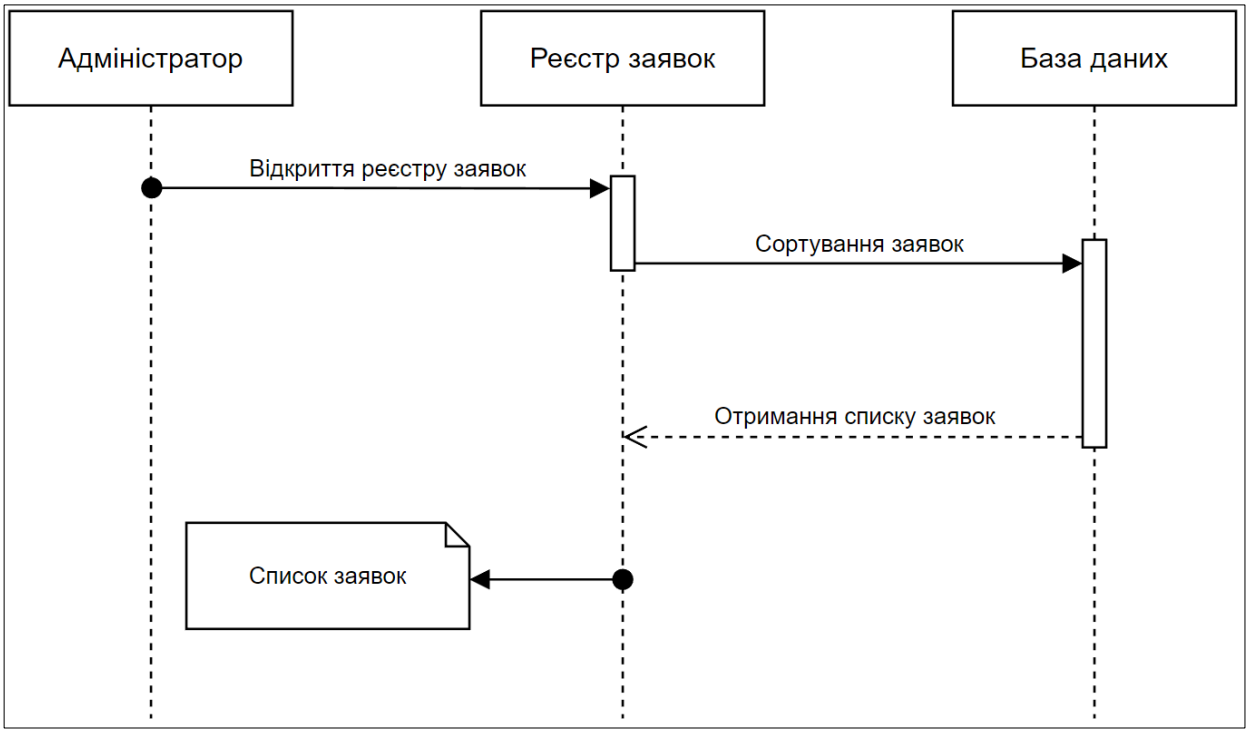


Рисунок 3.9. – **Діаграма послідовності процесу перегляду заявок від користувачів**

Джерело: авторська розробка

Діаграма послідовності процесу внесення змін наведена на рис. 3.10.

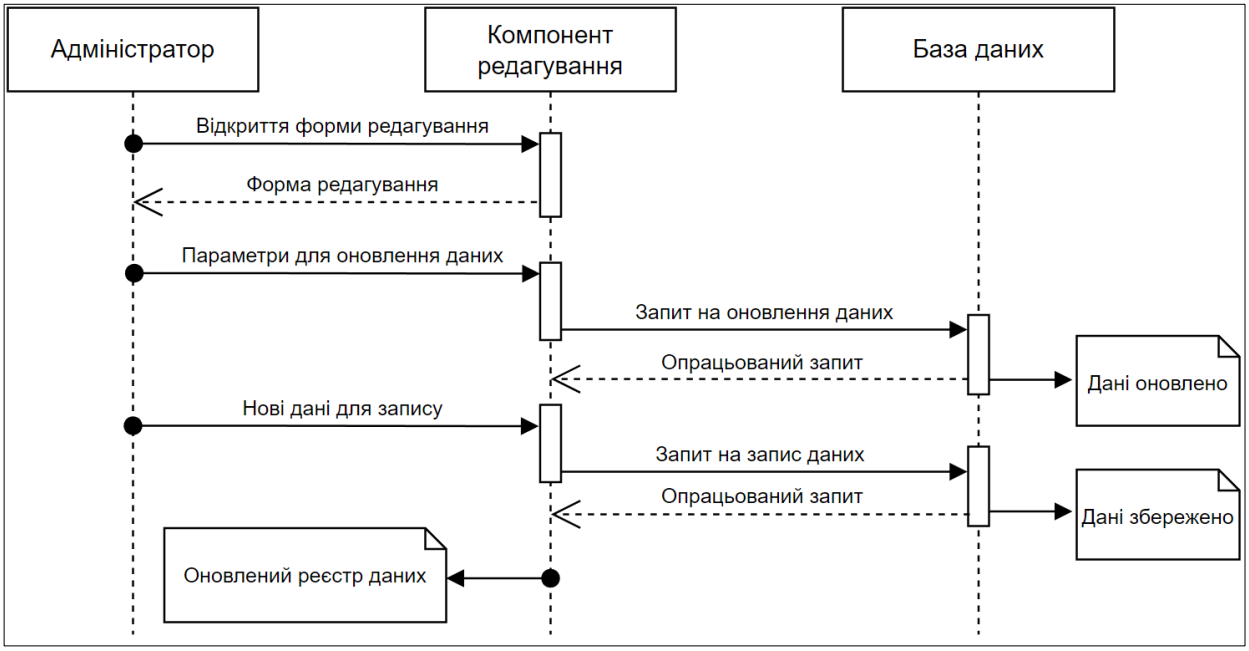


Рисунок 3.10. – **Діаграма послідовності процесу внесення змін**

Джерело: авторська розробка

Наведена діаграма демонструє принцип функціонування компоненту

редагування, який призначений як для оновлення наявних даних, так і реєстрації нових. Очевидно, функціонал оновлення або введення нових даних можливий лише при наявності авторизованого користувача з правами «адміністратор». Окрім цього, варто зауважити, що внесення змін в базу даних можливе лише тоді, коли параметри для змін відрізняються від тих, що вже існують в базі.

Кожна система, зокрема її компоненти характеризується не лише структурою її складових, зв'язками, сценаріями чи функціональністю, а й станами цих компонентів в процесі роботи системи. Для опису динаміки зміни станів компонентів, їх дій в рамках тих чи інших станах використовуються діаграми станів [36].

У вузькому розумінні, діаграма стану використовується для наочного подання тих чи інших станів програмного засобу, у яких він може знаходитися в різні моменти часу. Ключовою особливістю даного типу діаграм є опис реакцій, тобто ключових точок зупинки роботи програми. Кожен стан компонента має точки входу та виходу, тобто що веде до зміни стану, і який результат очікується отримати. Інакше кажучи, за допомогою діаграм станів можна сформувати «лінію життя» компонентів системи.

При описі динаміки станів об'єктів, специфікуються наступні ключові моменти [37]:

1) стабільні стани, у яких може перебувати об'єкт;

2) події, які спонукають зміну одного стану об'єкта на інший;

3) дії, які виконуються в рамках зміни станів.

Основним призначенням діаграм станів є опис послідовності переходів один станів в інші, які в сукупності характеризують поведінку елемента системи протягом його часу функціонування. Хоча діаграми станів часто використовуються для опису поведінки окремих об'єктів системи, нерідко вони застосовуються для специфікації функціональності інших компонентів моделей, таких як варіанти використання, актори, підсистеми, операції і методи.

За допомогою діаграми станів можна зобразити усі можливі стани одного примірника WEB-сайту і сценарії їх перемикання в інші. Тобто побудувати

модель зміни станів компонентів сайту та динаміку їх впливу на інші пов’язані з ним компоненти (рис. 3.11).

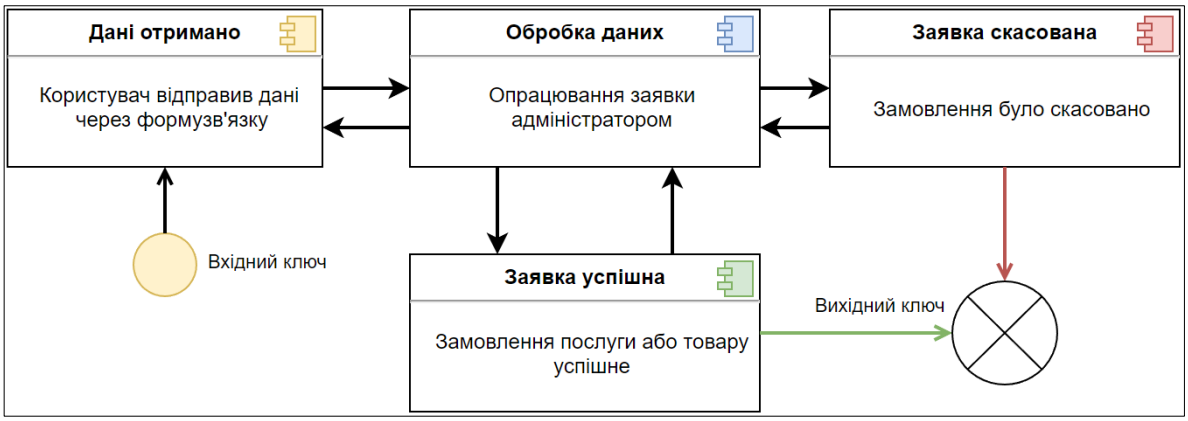


Рисунок 3.11. – **Діаграма станів роботи односторінкового WEB-сайту**

Джерело: авторська розробка

Наш майбутній односторінковий WEB-сайт характеризується наявністю чотирьох станів: «дані отримано», «обробка даних», «заявка успішна» і «заявка скасована». В рамках першого стану, в базу даних записується нова заявка з реєстраційними даними користувача. Якщо від одного і того ж самого користувача надходить декілька заявок, то в базу даних записується остання. Другий стан свідчить про розгляд заявок користувачів та їх обробку, зокрема уточняються дані пов’язана з процесом погодження заявки. Третій і четвертий стани є результатом обробки даних адміністратором – при погодженні заявки клієнтом стан системи характеризується міткою «заявка успішна», у цьому випадку виконується замовлення клієнта та закривається угода. У випадку якщо угода скасована однієї зі сторін – клієнтом або адміністратором, стан системи характеризується міткою «заявка скасована».

3.2. Файлова структура односторінкового WEB-сайту

Розробка односторінкового WEB-сайту починається з розгортання програмного середовища та під’єднання усіх залежностей для функціонування проєкту. Як зазначалося вище, нами використовується комплекс технологій для WEB-розробки, зокрема препроцесори PUG та SASS для швидкого формування HTML-розмітки та написання стилів для елементів WEB-сторінки відповідно, а

також інструмент для автоматичної збірки, управління та забезпечення коректної роботи препроцесорів – GULP.

Для базового налаштування робочого середовища, необхідно встановити менеджер пакетів для мови програмування JavaScript (рис. 3.12).

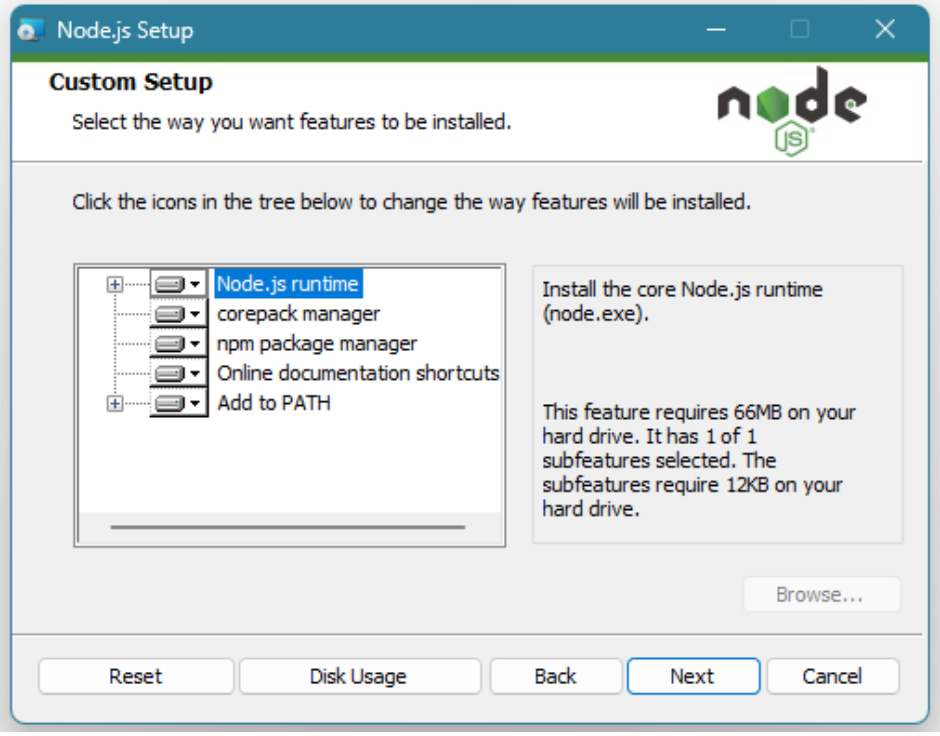


Рисунок 3.12. – **Процес встановлення менеджера пакетів Node.js**

Джерело: авторська розробка

Встановлення менеджера пакета дає нам можливість використовувати команду npm у командному рядку для завантаження та встановлення усіх залежностей для роботи з проєктами. Після встановлення Node.js, зокрема бібліотеки пакетів npm, необхідно встановити GULP та створити каталог для майбутнього WEB-сайту. Для цього, у командному рядку потрібно почергово ввести та виконати наступні команди:

1) npm install --global gulp-cli – завантажити та встановити у систему пакети для роботи інструмента GULP;

2) npx mkdirp singlePageWS – створити каталог під назвою singlePageWS для зберігання усіх необхідних для роботи файлів та налаштувань;

3) cd singlePageWS && npm init – зареєструвати файл налаштування package.json у створеному раніше каталозі.

Окрім цього, для завантаження та використання додаткових бібліотек та плагінів для нашого WEB-сайту ми під'єднали систему керування пакетами BOWER. Для цього, у командному рядку потрібно написати команду для встановлення пакетного менеджера bower з прапорцем -g. Зазначене дасть можливість викликати спеціальні команди інсталяції з будь-якого каталогу. Команда дає можливість встановити файл конфігурації bower.json у якому можна вказати, які бібліотеки, тобто залежності, та якої версії необхідно встановити для роботи проєкту. Фрагмент авторського файлу bower.json наведено на рис. 3.13.

```
"dependencies": {
  "hover": "latest",
  "960-grid-system": "latest",
  "bourbon": "latest",
  "bPopup": "bpopup#latest",
  "jquery": "1.12.0",
  "normalize-css": "latest",
  "owl.carousel": "latest",
  "bootstrap-grid-only": "latest",
  "simple-line-icons": "^2.4.1"
},
"devDependencies": {
  "reset-css": "latest",
  "remodal": "latest",
  "jeet": "^7.0.0",
  "bxslider": "latest",
  "bxslider-4": "^4.2.5",
  "slick": "*",
  "jquery-mousewheel": "^3.1.13"
}
```

Рисунок 3.13. – **Лістинг налаштування bower.json**

Джерело: авторська розробка

Після налаштування файлу конфігурації bower.json, необхідно завантажити пакети для роботи з препроцесорами. В рамках даної розробки ми використовуємо допоміжні інструменти оптимізації проєкту, зокрема мінімізації файлу стилів, автоматичного перезавантаження сторінки при реєстрації змін, оптимізації зображень для WEB-перегляду тощо. Для встановлення цих залежностей можна скористатися бібліотекою npm, або в ручному режимі у файл package.json вписати, які пакети залежностей нам необхідно встановити.

Фрагмент авторського файлу package.json наведено на рис. 3.14.

```
"devDependencies": {
  "gulp-changed": "latest",
  "gulp-clean": "^0.3.2",
  "gulp-cssnano": "latest",
  "gulp-group-css-media-queries": "^1.2.0",
  "gulp-plumber": "latest",
  "gulp-rename": "latest",
  "gulp-rigger": "^0.5.8",
  "gulp-shorthand": "latest",
  "gulp-uncss": "latest",
  "gulp-watch": "latest",
  "gulp-zip": "^3.2.0"
},
"dependencies": {
  "browser-sync": "latest",
  "gulp": "latest",
  "gulp-autoprefixer": "latest",
  "gulp-concat": "latest",
  "gulp-imagemin": "latest",
  "gulp-pug": "latest",
  "gulp-sass": "latest",
  "gulp-sourcemaps": "latest",
  "gulp-uglify": "latest"
}
```

Рисунок 3.14. – **Лістинг налаштування package.json**

Джерело: авторська розробка

В загальному вигляді, початкова файлова структура каталогу проєкту має наступний вигляд (рис. 3.15):

Рисунок 3.15. – **Файлова структура каталогу WEB-проєкту**

Джерело: авторська розробка

Пропонуємо детальніше розглянути наведену файлову структуру нашого WEB-проєкту. Каталог src є батьківським каталогом, який містить вкладені каталоги з файлами стилів, розмітки, скриптів, шрифтами та зображеннями.

Інструмент GULP відстежує усі зміни в даному каталозі та вносить зміни у структуру вкладених файлів. Вся робота над проєктом відбувається в межах даного каталогу.

Фрагмент лістинга GULP інструмента для відстеження змін у каталозі src наведений на рис. 3.16.

```
watch: {
    pug: './src/pug/**/*.pug',
    pugIncludes: './src/pug/_includes/**/*.pug',
    js: './src/js/*.js',
    jsVendor: './src/js/vendor/*.js',
    scss: ["./src/sass/**/*.scss",'./src/sass/_*.scss'],
    img: './src/img/**/*',
    favicon: './src/favicon/*',
    fonts: './src/fonts/**/**'
}
```

Рисунок 3.16. – **Фрагмент лістинга GULP інструмента**

Джерело: авторська розробка

Ієрархічне представлення каталогу src наведено на рис. 3.17.

```
C:.
└───src
    ├───favicon
    ├───fonts
    │   └───line-icons
    ├───img
    │   └───icons
    ├───js
    │   └───vendor
    ├───pug
    │   └───_includes
    │       ├───_head
    │       └───_layout
    └───sass
        ├───base
        ├───layout
        └───vendor
```

Рисунок 3.17. – **Структура каталогу src**

Джерело: авторська розробка

Найскладнішою за своєю структурою є каталоги, у яких зберігаються файли розмітки *.pug та стилів *.scss – каталоги pug та scss відповідно.

Каталог pug містить колекцію підкаталогів для розмітки WEB-сторінок

сайту. Зокрема каталог _head містить файли із системною і метаінформацією про WEB-сторінку, _layout, в свою чергу, файли із розміткою основних секцій сторінки, таких як «шапка», «тіло» та «підвал». Файл _layout.pug є основним файлом – ядром, який під'єднує усі файли розмітки і компонує їх у вихідний файл index.html (рис. 3.18).

```
doctype html
html(lang="ru-RU")
    block head
        include ./_includes/_head/_head.pug
    body
        .wrapper
            .maincontent
                block header
                    include ./_includes/_layout/_header.pug
                main
                    block main
                        include ./_includes/_layout/_main.pug
            block footer
                include ./_includes/_layout/_footer.pug
        block script
            include ./_includes/_layout/_script.pug
```

Рисунок 3.18 – **Фрагмент лістинга файлу _layout.pug**

Джерело: авторська розробка

Ієрархічне представлення каталогу pug наведено на рис. 3.19.

```
C:.
    index.pug
    _layout.pug

└───_includes
    ├───_head
            _head.pug
            _socials.pug

    └───_layout
            _footer.pug
            _header.pug
            _hiddenScript.pug
            _main.pug
            _script.pug
```

Рисунок 3.19. – **Структура каталогу pug**

Джерело: авторська розробка

Каталог scss містить систему файлів для оформлення розмітки WEB-сторінок сайту. Даний каталог містить три підкаталоги (base, layout, vendor) та кінцевий файл styles.scss який, як і _layout.pug, поєднує усі файли оформлення в кінцевий документ styles.css.

Ієрархічне представлення каталогу scss наведено на рис. 3.20.

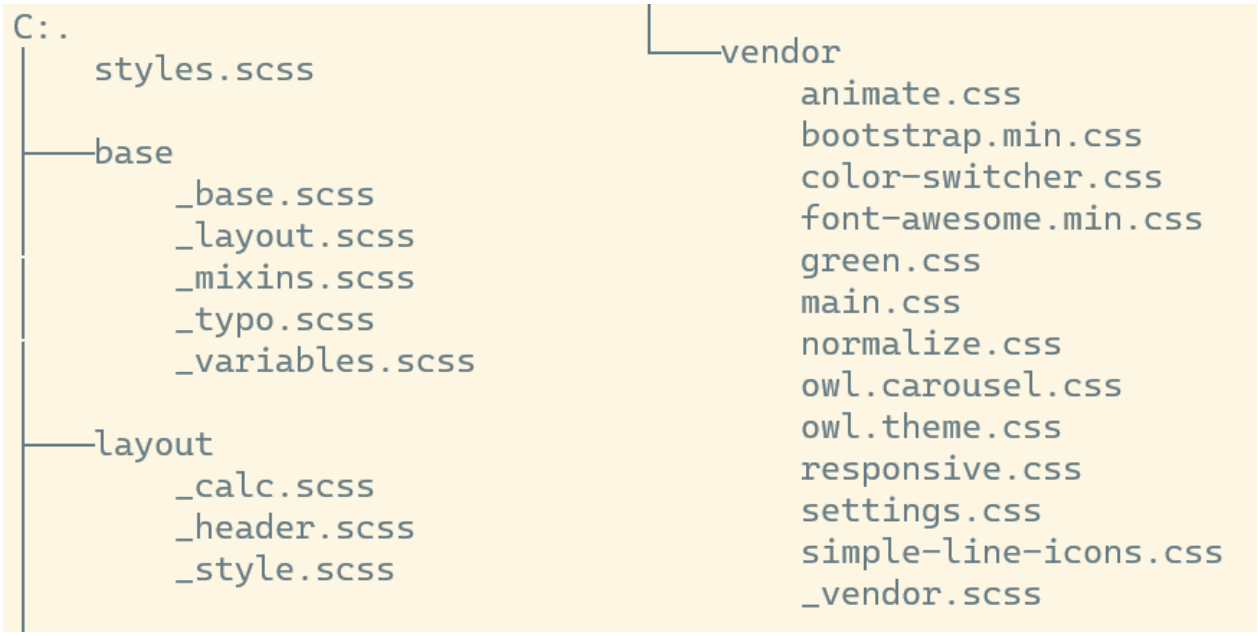


Рисунок 3.20. – **Структура каталогу scss**

Джерело: авторська розробка

Каталог base призначений для зберігання файлів із початковим налаштуванням зовнішнього вигляду WEB-сторінок, зокрема _base.scss – для написання базових стилів, які використовуються на усіх сторінках; _layout.scss – для під'єднання сторонніх бібліотек стилів; _mixins.scss – для написання власних міксінів та шаблонів; _typo.scss – для налаштування шрифтів та їх відображення; _variables.scss – для формування системи змінних із кольорами.

Каталог layout призначений для зберігання файлів, які містить стилі для оформлення WEB-сторінок, наприклад файл _header.scss містить стилі, які призначені виключно для секції «шапка». При певній необхідності, в даному каталозі можуть створюватися додаткові файли для оформлення різних блоків WEB-сторінок. Наприклад файл _calc.scss був створений для написання стилів щодо оформлення секції, яка містить цифрову інформацію. Файл _style.scss, по аналогії з файлами styles.scss та _layout.pug, об'єднує в собі усі файли стилів, за

тим виключенням, що лише ті, які містяться у каталозі layout.

Також варто розглянути каталог js. Він містить файли, які призначені для написання сценаріїв мовою програмування JavaScript. В корні даного каталогу є файл script.js, який є основним файлом зі скриптами, в який під'єднуються сторонні файли із каталогу vendor. Даний файл, при відкритті WEB-сторінки, завантажується першим.

Ієрархічне представлення каталогу js наведено на рис. 3.21.

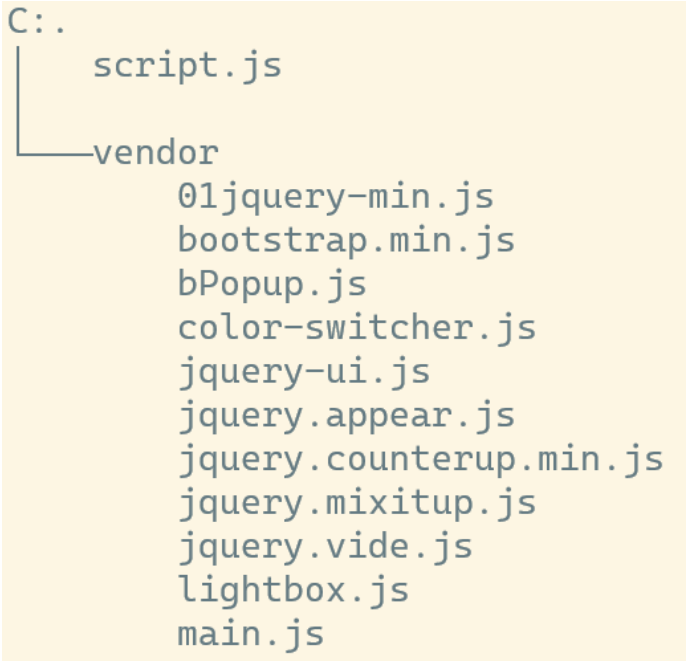


Рисунок 3.21. – **Структура каталогу js**

Джерело: авторська розробка

Наведена файлова структура є оптимальною для розробки WEB-проєктів будь-якої складності. Дана структура характеризується високим рівнем модульності та незалежності одних компонентів від інших. Тобто, при виникненні помилок в одному із файлів, програмне середовище ізолює помилку та забезпечить цілісність проєкту і мінімізує пошкодження сторонніх файлів. Окрім цього, модульність дає змогу з легкістю як під'єднувати нові, так і від'єднувати наявні бібліотеки без додаткового налаштування кореневих файлів. За допомогою інструмента GULP, можна оптимізувати процес розробки шляхом автоматизації часто повторюваних операцій, що підвищує загальну продуктивність та зменшує час на розробку.

3.3. Програмна реалізація односторінкового WEB-сайту

Наш односторінковий WEB-сайт є прикладом відображення акцентної та статистичної інформації про роботу підприємства та сервісної служби по встановленню та обслуговуванню систем відео нагляду. Варто зауважити, що підхід до розборки односторінкового WEB-сайту та принцип структурування інформації справедливий для усіх підприємств, незалежно від специфіки їх функціонування.

В рамках нашого проєкту передбачається вісім секцій, серед яких головний екран, «Наші послуги», «Індивідуальні рішення», «Готові рішення», «Калькулятор», «Наші переваги», «Статистика» та футер. Розробка нашого односторінкового WEB-сайту починається з формування розмітки головного екрану. У даному блоці користувачу надається можливість навігації по сайту та заповнення форми зворотного зв'язку.

Для відмежовування блоку навігації від інших секцій, а також для структурування інформації у них, використовуються HTML5 теги section та nav, функціональне призначення яких формування меж секцій сайту. Лістинг розмітки блоків навігації та головного екрану наведено на рисунках 3.22, 3.23.

```
<section id="header">
  <nav class="navbar navbar-light" data-spy="affix" data-offset-top="50">
    <div class="container">
      <button class="navbar-toggler hidden-md-up pull-xs-right" data-target="#main-menu"
      data-toggle="collapse" type="button">☰</button>
      <div class="collapse navbar-toggleable-sm pull-xs-left pull-md-right" id="main-menu">
        <ul class="nav nav-inline">
          <li class="nav-item dropdown"><a class="nav-link" href="#portfolio" role="button"
          aria-haspopup="true" aria-expanded="false">Готові рішення</a></li>
          <li class="nav-item dropdown"><a class="nav-link" href="#calc" role="button"
          aria-haspopup="true" aria-expanded="false">Калькулятор</a></li>
          <li class="nav-item dropdown"><a class="nav-link" href="#footer" role="button"
          aria-haspopup="true" aria-expanded="false">Контакти</a></li><a class="btn btn-lg btn-common"
          href="tel:88005504460"><i class="fa fa-phone" aria-hidden="true"></i> Замовити дзвінок </a>
        </ul>
      </div>
      <form class="full-search">
        <div class="container">
          <input type="text" placeholder="Type to Search"><a class="close-search" href="#"><span class="fa
          fa-times fa-2x"></span></a>
        </div>
      </form>
    </div>
  </nav>
</section>
```

Рисунок 3.22. – **Фрагмент лістингу розмітки блоку навігації**

Джерело: авторська розробка

```
<div id="carousel-area">
  <div class="carousel slide" id="carousel-slider" data-ride="carousel">
    <!-- Indicators-->
    <div class="carousel-inner" role="listbox">
      <div class="carousel-item active"><img src="img/bg-11.jpg" alt="First slide">
        <div class="carousel-caption">
          <h2>Встановлення прихованих систем відеоспостереження</h2>
          <h3>
            Готові рішення для приватних та промислових об'єктів у цифровому та аналоговому варіантах
            <br><br>
            Дізнайтесь точну вартість системи - інженер складе кошторис під час безкоштовного виїзду на
            ваш об'єкт
          </h3><a class="btn btn-lg btn-common" href="#"><i class="fa fa-download"></i>Виклик майстра</a>
        </div>
      </div>
    </div><a class="left carousel-control nav-prev" href="#carousel-slider" role="button"
    data-slide="prev"><span class="sr-only">Previous</span></a><a class="right carousel-control nav-next"
    href="#carousel-slider" role="button" data-slide="next"><span class="sr-only">Next</span></a>
  </div>
</div>
```

Рисунок 3.23. – **Фрагмент лістингу розмітки головного екрану**

Джерело: авторська розробка

Секція «Наші послуги» представляє собою блок, у якому міститься коротка інформація про послуги підприємства, з якою може ознайомитися користувач. Лістинг розмітки секції наведено нижче (рис. 3.25):

```
<div class="col-md-4 col-sm-12 col-xs-12 wow animated fadeInLeft animated" data-wow-delay=".3s"
style="visibility: visible;-webkit-animation-delay: .3s; -moz-animation-delay: .3s; animation-delay: .3s;">
  <div class="images"><img src="img/plain-why-choose-us-2.png" alt=""></div>
</div>
<div class="col-md-8 col-sm-12 col-xs-12 wow animated fadeInRight animated" data-wow-delay=".3s"
style="visibility: visible;-webkit-animation-delay: .3s; -moz-animation-delay: .3s; animation-delay: .3s;">
  <div class="content-inner">
    <p class="lead">Ми займаємося встановленням систем відеоспостереження як у приватних та багатоквартирних
    будинках, так і у торгових приміщеннях, офісах, на складах та парковках. Кожен клієнт отримує оптимальний
    варіант обладнання в залежності від своїх цілей, бюджету та особливостей об'єкта.</p>
    <div class="details-list">
      <div class="row">
        <div class="col-sm-6 col-xs-12">
          <h3>Цифрові системи</h3>
          <p>Чітке відображення рухомих об'єктів, відтворення деталей за рахунок мегапіксельної роздільної
          здатності та синхронізація аудіо з відео</p>
        </div>
        <div class="col-sm-6 col-xs-12">
          <h3>Аналогові системи</h3>
          <p>Бюджетний, простий у використанні та перевірений часом варіант для невеликих приміщень та територій.
          </p>
        </div>
```

Рисунок 3.24. – **Фрагмент лістингу розмітки секції «Наші послуги»**

Джерело: авторська розробка

Розширена інформація про послуги підприємства, а також їх опис міститься у секції «Індивідуальні рішення». Як правило, в даній секції наводяться найбільш популярні послуги підприємства. Лістинг розмітки описаної секції наведено на рис. 3.25.

```
<h1 class="section-title wow fadeIn animated animated" data-wow-delay=".2s" style="visibility: visible;
-webkit-animation-delay: .2s; -moz-animation-delay: .2s; animation-delay: .2s;">Індивідуальні рішення</h1>
<p class="section-subcontent">
  У нашому асортименті – понад 100 видів товарів для комплектації систем безпеки. Звернувшись до компанії, Ви
  отримуєте індивідуальне складання системи за прараметрами вашого об'єкта.
</p>
<div class="col-sm-6 col-md-3">
  <div class="service-item wow fadeInUpQuick animated animated" data-wow-delay=".5s" style="visibility: visible;
  -webkit-animation-delay: .5s; -moz-animation-delay: .5s; animation-delay: .5s;">
    <div class="icon-wrapper"><i class="icon-layers pulse-shrink"></i></div>
    <h2>Внутрішнє відеоспостереження</h2>
    <p>Камери для установки в приміщенні, у тому числі приховані.</p>
  </div>
</div>
<div class="col-sm-6 col-md-3">
  <div class="service-item wow fadeInUpQuick animated animated" data-wow-delay=".8s" style="visibility: visible;
  -webkit-animation-delay: .8s; -moz-animation-delay: .8s; animation-delay: .8s;">
    <div class="icon-wrapper"><i class="icon-settings pulse-shrink"></i></div>
    <h2>Охорона периметра</h2>
    <p>Вуличні камери, зокрема, укомплектовані датчиками руху.</p>
  </div>
</div>
```

Рисунок 3.25. – **Фрагмент лістингу розмітки секції «Індивідуальні рішення»**

Джерело: авторська розробка

Для демонстрації можливостей або типів робіт підприємства, було реалізовано секцію «Готові рішення». В ній наведено інформацію про вже реалізовані підприємством проєкти, їх вартість, комплектуючі тощо. Секція виконана у вигляді галереї, де рішення згруповані за певним класифікатором. Користувач може перемикатися між об’єктами, натискаючи на певний елемент керування. Фрагмент лістингу розмітки секції наведено на рис. 3.26.

```
<div class="row">
  <h1 class="section-title wow fadeInUpQuick" data-wow-delay=".3s" style="visibility: hidden;
  -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .3s;
  -moz-animation-delay: .3s; animation-delay: .3s;">Готові рішення</h1>
  <p class="section-subcontent wow fadeInUpQuick" data-wow-delay=".4s" style="visibility: hidden;
  -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .4s;
  -moz-animation-delay: .4s; animation-delay: .4s;">Монтаж будь-якої системи не зіпсує зовнішнього вигляду будівлі:
  непомітні камери оснащені онлайн-доступом для <br> віддаленого перегляду, а передача Wi-Fi забезпечить цілодобове
  спостереження за об'єктом з будь-якої точки планети.</p>
  <!-- Portfolio Controller/Buttons-->
  <div class="controls text-center wow fadeInUpQuick" data-wow-delay=".6s" style="visibility: hidden;
  -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .6s;
  -moz-animation-delay: .6s; animation-delay: .6s;"><a class="filter btn btn-common active" data-filter="all">Усі
  рішення</a><a class="filter btn btn-common" data-filter=".house">Будинок</a><a class="filter btn btn-common"
  data-filter=".sellHouse">Торгові приміщення</a><a class="filter btn btn-common" data-filter=".office">Офіси</a><a
  class="filter btn btn-common" data-filter=".room">Квартири</a><a class="filter btn btn-common" data-filter=".
  access">Під'їзди</a><a class="filter btn btn-common" data-filter=".parking">Парковки</a></div>
  <div class="wow fadeInUpQuick" id="portfolio-list" data-wow-delay=".8s" style="visibility: hidden;
  animation-name: none; animation-delay: 0.8s;">
```

Рисунок 3.26. – **Фрагмент лістингу розмітки секції «Готові рішення»**

Джерело: авторська розробка

Секція «Калькулятор» призначена для надання користувачам інформації

про вартість послуг, ввівши параметри свого приміщення та бажані комплектуючі. Дана секція є гарним прикладом комунікації з адміністратором, адже відправлена заявка на обрахунок вартості, як правило, в базі даних не зберігається, а надсилається у вигляді листа на пошту, або в систему керування контентом. Лістинг розмітки секції наведено на рис. 3.27.

```
    <div class="row">
      <h1 class="section-title wow fadeInUpQuick" data-wow-delay=".3s" style="visibility: hidden;
      -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .3s;
      -moz-animation-delay: .3s; animation-delay: .3s;">ШВИДКИЙ РОЗРАХУНОК ВАРТОСТІ</h1>
    </div>
  </div>
</div>
<div class="constr">
  <div class="inconstructor">
    <form action="order.php" method="post">
      <div class="step1">
        <div class="line-constr wow fadeInLeft">
          <h4>1. Скільки камер ви плануєте встановити?</h4>
          <table class="constructor grey-color">
            <tbody>
              <tr>
                <td width="25%">приховані
                  <div class="inputBoxDiv" style="width: 44.4px;">
                    <input class="inputBox spinner" name="hid_cam" type="text" value="1">
                  </div>
                </td>
                <td width="25%">вуличні
```

Рисунок 3.27. – **Фрагмент лістингу розмітки секції «Калькулятор»**

Джерело: авторська розробка

Для надання інформації про ключові переваги даного підприємства серед інших підприємств на ринку призначена секція «Наші переваги». Лістинг розмітки секції наведено на рис. 3.28.

```
<h1 class="section-title wow fadeInUpQuick" style="visibility: hidden; -webkit-animation-name: none;
-moz-animation-name: none; animation-name: none;"> Наші переваги </h1>
<div class="col-md-4 col-sm-6" data-animation="fadeIn" data-animation-delay="01">
  <div class="featured-box">
    <div class="featured-icon"><i class="icon-present"></i></div>
    <div class="featured-content">
      <h4>Економимо ваші гроші</h4>
      <p>Після безкоштовного огляду складається фіксований кошторис. Налаштування системи виконується
      безкоштовно!</p>
    </div>
  </div>
</div>
<div class="col-md-4 col-sm-6 wow fadeInUpQuick" data-wow-delay=".2s" data-animation="fadeIn"
data-animation-delay="01" style="visibility: hidden; -webkit-animation-name: none; -moz-animation-name: none;
animation-name: none;-webkit-animation-delay: .2s; -moz-animation-delay: .2s; animation-delay: .2s;">
  <div class="featured-box">
    <div class="featured-icon"><i class="icon-rocket"></i></div>
    <div class="featured-content">
      <h4>Цінуємо час клієнтів</h4>
      <p>Доставляємо системи протягом 1-2 днів і виконуємо монтаж до 10 камер за 24 години.</p>
```

Рисунок 3.28. – **Фрагмент лістингу розмітки секції «Наші переваги»**

Джерело: авторська розробка

В даному блоці висвітлюється найбільш цінна інформація, яка «змусить» користувача оформити замовлення. Зокрема мова йде про унікальні особливості та функціонал пропонованої продукції (послуги), або про переваги, які отримає клієнт від співпраці з даним підприємством.

Секція статистики несе в собі інформацію про «історію успіху» підприємства, зокрема про досвід підприємства, кількість виконаних проєктів, залишених заявок від користувачів тощо. Дана секція має на меті привабити користувача до співпраці, продемонструвавши йому свою динаміку діяльності. Лістинг розмітки секції наведено на рис. 3.29.

```
-moz-animation-name: none; animation-name: none;"> Трішки статистики </h1>
<div class="col-sm-6 col-md-3 col-lg-3">
  <div class="fact-block clearfix wow fadeInUp" data-wow-delay=".3s" style="visibility: hidden;
  -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .3s;
  -moz-animation-delay: .3s; animation-delay: .3s;">
    <div class="facts-item"><i class="icon-trophy"></i>
      <div class="fact-count">
        <h3><span class="counter">5</span></h3>
        <h4>років досвіду у сфері відеоспостереження</h4>
      </div>
    </div>
  </div>
</div>
<div class="col-sm-6 col-md-3 col-lg-3">
  <div class="fact-block clearfix wow fadeInUp" data-wow-delay=".8s" style="visibility: hidden;
  -webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .8s;
  -moz-animation-delay: .8s; animation-delay: .8s;">
    <div class="facts-item"><i class="icon-heart"></i>
```

Рисунок 3.29. – **Фрагмент лістингу розмітки секції статистики**

Джерело: авторська розробка

Інформація про місцезнаходження підприємства, робочі часи та інші контактні дані, зокрема форма зв’язку та посилання на соціальні мережі знаходяться у футері WEB-сторінки (рис. 3.30).

```
-webkit-animation-name: none; -moz-animation-name: none; animation-name: none;-webkit-animation-delay: .2s;
-moz-animation-delay: .2s; animation-delay: .2s;">
  <h3 class="small-title">Контакти</h3>
  <ul class="recent-tweets">
    <li class="tweet">
      Багатоканальний телефон:<br>
      +3 8 (096) 660-660-1 <br><br>
      Адреси<br>
      м. Вінниця, вул. Гоголя 1  <br>
      1-й поверх <br><br>
      м.Вінниця вул.Некрасова 73 <br>
      1-й поверх <br><br>
    </li>
```

Рисунок 3.30. – **Фрагмент лістингу розмітки футера WEB-сторінки**

Джерело: авторська розробка

Окрім основних секцій, нами було також розроблено «плаваючу» форму зворотного зв’язку (рис. 3.31).

```
.popup-form
    .icon-close
    p.
        Заповніть контактну форму, і ми Вам зателефонуємо протягом 30 хв
    form
        .form-group
            input#enterData.form-control(type='text', placeholder="Введіть Ваш E-mail")
            input#enterData.form-control(type='text', placeholder="Введіть Ваше питання")
        button.btn.btn-common.form-btnn(type='submit') Відправити
```

Рисунок 3.31. – **Фрагмент лістингу розмітки футера WEB-сторінки**

Джерело: авторська розробка

Дана форма по замовчування прихована і на сайті вона у звичайному для користувача вигляді не відображається. Для її виклику, користувач має натиснуту на будь-яку кнопку, наприклад «Замовити дзвінок», «Виклик майстра» тощо.

Для цього, на кожній кнопці, яка по кліку по ній повинна викликати форму, було виділено окремий css клас «.btn-lg», який використовується у js сценаріх як «тригер» для виклику події .click(), яка «звертається» до форми по класу «.popup-form» і відображає її з короткою анімаційною затримкою (рис. 3.32).

```
$('.btn-lg').click(function(event) {
    event.preventDefault();
    $('.popup-form').bPopup({
        opacity: 0.65,
        closeClass: 'icon-close'
    });
    return false;
})
```

Рисунок 3.32. – **Фрагмент лістингу розмітки футера WEB-сторінки**

Джерело: авторська розробка

Після збірки проєкту, його файлова структура буде мати наступний вигляд (рис. 3.33):

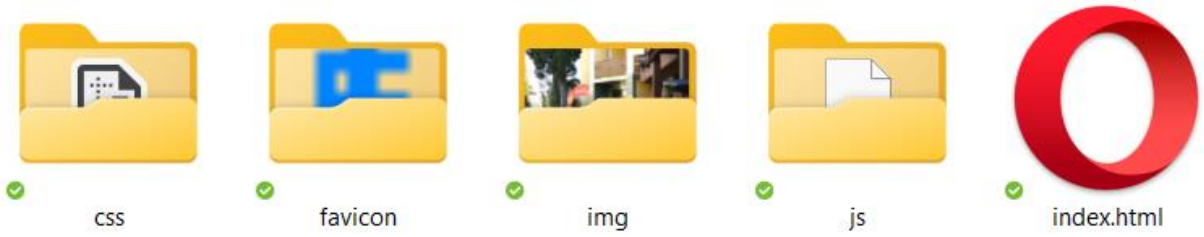


Рисунок 3.33 – **Вихідна файлова структура WEB-сайту**

Джерело: авторська розробка

Файл index.html є основним документом, який по замовчуванню відкривається браузером; в каталозі css зберігаються оптимізовані файли стилів; в каталозі img – зображення для перегляду; в каталозі js – мінімізований файл script.js для виконання сценаріїв, зокрема для виклику «плаваючої» форми зв'язку. Ієрархічна структура кінцевого каталогу проєкту build наведена рис. 3.34.

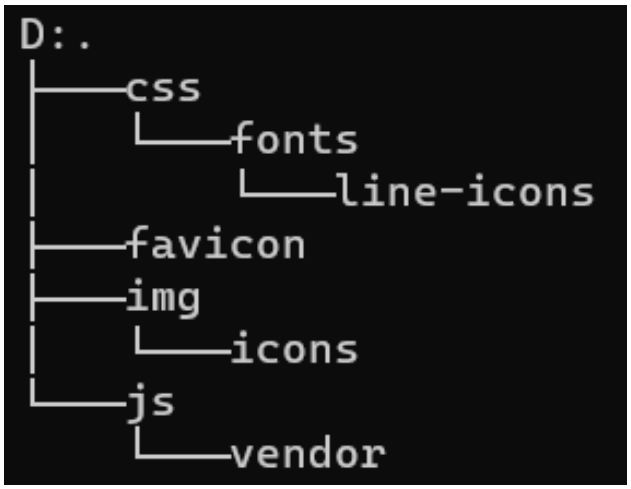


Рисунок 3.34. – **Ієрархічна структура каталогу build**

Джерело: авторська розробка

Останнім етапом розробки WEB-сайту є завантаження каталогу з файлами проєкту на хостинг для можливості WEB-перегляду усім користувачам мережі Інтернет, адже на даний момент, перегляд сайту можливий лише на локальній робочій машині розробника.

Для цього, нам потрібно завантажити файли на сервер, використовуючи FTP менеджер на прикладі FileZilla. Щоб розмістити сайт через панель

управління, необхідно завантажити архів з файлами сайту в кореневій каталог. В якості хостингу ми використовуємо популярний український сервіс надання хостингових послуг https://www.zzz.com.ua/.

Для підключення до серверу необхідно у FileZilla вести реєстраційні дані від хостингу, створити кореневий каталог та завантажити архів із сайтом, як зображено на рис. 3.35.

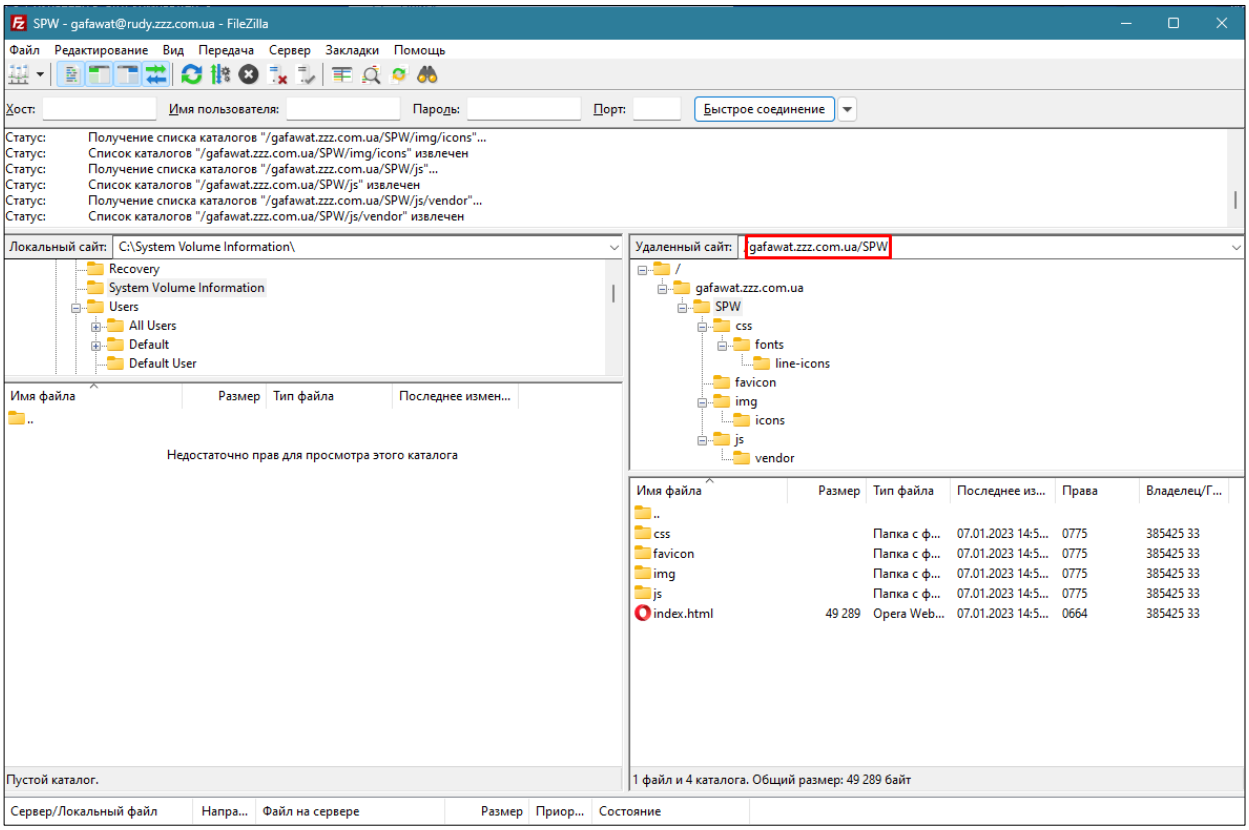


Рисунок 3.35. – **Завантаження односторінкового WEB-сайту на хостинг**

Джерело: авторська розробка

Розроблений односторінковий WEB-сайт завантажений на хостинг та доступний для перегляду. Сайт відображається коректно на усіх пристроях, тобто має адаптивний дизайн.

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

Розгортання діяльності в мережі Інтернет дає підприємствам можливість підвищити конкурентоспроможність товарів, розширити ринки збуту, підвищити пізнаваність бренду, знаходити нових постачальників, посередників та споживачів тощо. Сьогодні для підприємств, незалежно від специфіки їх діяльності, WEB-сайт – є вкрай ефективним та багатоцільовим маркетинговим інструментом, який при мінімальній витратах на його розробку та запуск згенерує не лише потік користувачів та потенційних клієнтів, а високі рівні прибутків.

Розробка односторінкового WEB-сайту – це багатоетапний та складний процес, який вимагає від розробника не лише творчих, але й аналітичних навичок. Реалізація WEB-проєкту передбачає детального планування та реалізації великого обсягу робіт, незалежно від розмірів WEB-сайту. Щоб сайт ефективно виконував покладені на нього функції, він повинен містити цікавий та неперевантажений зайвою інформацією контент, швидко та коректно відображатися на пристроях з різним розширенням екранів та під керуванням різних операційних систем. Для запуску успішного інтернет-ресурсу необхідно скласти план розробки, сформувати етапи створення веб-сторінок та дотримуватись наміченого плану.

В результаті виконання дипломної роботи нами було розроблено та завантажено на хостинг односторінковий WEB-сайт для підтримки діяльності підприємств. В ході розробки сайту, було поставлено та виконано наступні завдання, зокрема проаналізовано предметну область дослідження; досліджено теоретичні засади розробки WEB-сайтів; обрати та описати інструменти розробки WEB-сайтів; здійснити верстку WEB-сайту; завантажити WEB-сайт на хостинг.

Ґрунтуючись на проведеному дослідженні, у першій частині дипломної роботи було з’ясовано, що під WEB-сайтом розуміється сукупність веб-сторінок, оформлених в одному стилі та об’єднаних спільною концепцією; наведено класифікацію WEB-сайтів, згідно якої сайти за технологією їх створення

поділяються на статичні, динамічна та Flash-сайти; наведено загальну структуру односторінкових WEB-сайтів; визначено важливість створення та активного використання сайту підприємства для підвищення репутації підприємства та збільшення кількості замовлень; основні правила розробки успішного односторінкового WEB-сайту; переваги та недоліки посадкових сторінок; порівняльну структуру односторінкових WEB-сайтів та традиційних інтернет-магазинів, а також їх воронок продажів; узагальнено етапи створення WEB-сайту; найбільш часто використовувані підходи до організації зв'язків між сторінками WEB-сайту, серед яких лінійна, блокова, стандартна та деревоподібна (ієрархічна), графічно представлено їх структуру; здійснено огляд сайтів-аналогів, де було виділено ключові переваги та особливості кожного, а також сформовано зведену характеристику аналізованих односторінкових WEB-сайтів; наведено результати порівняння відповідності структури сайтів стандартам W3C.

В рамках другої частини дипломної роботи було досліджено підходи, інструменти та технології розробки сучасних односторінкових WEB-сайтів. За результатами дослідження було обрано та описано інструменти розробки WEB-сайтів, зокрема у якості середовища розробки односторінкового WEB-сайту було обрано Visual Studio Code, як одного із найбільш зручних та швидких програмних середовищ для розробки проєктів будь-якої складності, що підтверджується опитуванням користувачів WEB-спільноти Stack Overflow. Продемонстровано процес вставлення середовища розробки Visual Studio Code, принципи встановлення розширень через інтерфейс Visual Studio Code Marketplace, а також наведено інструменти розгортання проєктів у хмарному середовищі.

Для налаштування середовища розробки під власні потреби, було встановлено додаткові розширення Visual Studio Code, серед яких Pug/Jade Snippets та Pug beautify для розширеної роботи з файлами препроцесора PUG гіпертекстової розмітки HTML; SCSS Formatter та SCSS IntelliSense для розширеного форматування SCSS файлів; JSHint та JS Refactor для роботи з

файлами JavaScript. Розглянуто сучасні технології WEB-розробки та графічно перестановлено структуру GULP-проєкту, HTML-тегів, CSS-правил, PUG-розмітки та SCSS-конструкцій. Наведено структуру найпростішої WEB-сторінки та принцип під'єднання JavaScript сценаріїв.

Описано та графічно представлено принципи функціонування клієнт-серверної архітектури. Наведено структуру дво- , три- та багаторівневої клієнт-серверної архітектури додатків та WEB-сайтів, а також схематично зображено модель «тонкого» та «товстого» клієнта архітектури.

Проведене дослідження дозволило у третій частині здійснити програмну реалізацію односторінкового WEB-сайту. Цьому передувало ґрунтоване моделювання роботи WEB-сайту, де було наведено контекстну діаграму роботи односторінкового WEB-сайту та здійснена подальша її декомпозиція. За допомогою уніфікованої мови моделювання було розроблено та графічно представлено діаграми варіантів використання WEB-сайту, послідовності перегляду інформації, послідовності процесу авторизації в системі, послідовності процесу перегляду заявок від користувачів, послідовності процесу внесення змін та станів роботи односторінкового WEB-сайту.

За допомогою встановлення менеджера пакетів Node.js було розгорнуто програмне середовище та сформовано початкову структуру проєкту. Для цього було налаштовано файли-конфігуратори менеджера пакетів npm та систему керування пакетами bower. Для підтримки препроцесорів PUG та SASS було встановлено інструмент автоматизації GULP та відповідні пакети залежності.

На основі проведених досліджень було розроблено та завантажено на хостинг власний односторінковий WEB-сайт, який складається із восьми секцій, серед яких головний екран, «Наші послуги», «Індивідуальні рішення», «Готові рішення», «Калькулятор», «Наші переваги», «Статистика» та футер. Сайт має адаптивний дизайн, що забезпечує його коректне функціонування на різних пристроях користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Карпій О.П., Виноградська Ю.А. Веб-сайт підприємства як засіб управління маркетинговою діяльністю. *Наукові записки Львівського університету бізнесу та права*. 2022. С. 213-219. DOI: 10.5281/zenodo.7299732

2. Комарницький І.М., Бублик М.І., Мужилівський М.Д. Розробка концепції оцінки якості веб-сайтів як умови розвитку електронного підприємництва. *Вісник ЖДТУ. Економічні науки*. 2008. № 1 (43). С. 177-189.

3. Кінас І.О. Інтернет–маркетинг та його роль у формуванні маркетингової стратегії підприємства. *Ефективна економіка*. 2021. № 4. URL: http://www.economy.nayka.com.ua/?op=1&z=8812 (дата звернення: 20.01.2023). DOI: 10.32702/2307-2105-2021.4.103

4. Лісовська А.Ю., Калита А.А. Контент веб-сайтів і їхня структура. *Молодий вчений.* № 10 (74). С. 166-170. DOI: 10.32839/2304-5809/2019-10-74-39

5. «Why You (Yes, You) Need to Create More Landing Pages. URL: http://surl.li/gdplb (дата звернення: 20.01.2023).

6. Шафалюк О.К. Методологічні проблеми і можливості розвитку Інтернет-маркетингу. *Маркетинг і цифрові технології.* 2017. Т. 1 № 1. С. 107-126. DOI: 10.15276/mdt.1.1.2017.7

7. Ілляшенко С. М. Сучасні тенденції застосування інтернет-технологій у маркетингу/ *Маркетинг і менеджмент інновацій.* 2011. № 4 (2). С. 64-74.

8. Бурило Ю.П. Веб-сайт як інформаційний ресурс та об’єкт права інтелектуальної власності. *Науковий вісник Ужгородського національного університету.* 2015. С. 67-70

9. Вовк О.Б. Методи та засоби підвищення життєздатності веб-сайту як інформаційного продукту : автореф. дис. ... канд. техн. наук : 05.13.06. Львів, 2013. 20 с.

10. Компанєєтс М.О. Принципи проектування ефективних веб-сайтів. *Молодий вчений.* № 9 (24). С. 106-109.

11. Кільченко А.В., Поповський О.І., Матросова Н.М. Базові поняття і

терміни веб-технологій. Київ, 2014. 49 с.

12. Курси англійської у Вінниці. URL: https://p12.com.ua/vinnytsia

13. IT-курси Wezom Academy. URL: https://wezom.academy/ua/

14. Health and Food Kyiv – сервіс правильного харчування. URL: https://health-food.com.ua/ua/

15. Інструмент для прискорення прогресу в SEO. URL: https://serpstat.com

16. The W3C Markup Validation Service. URL: https://validator.w3.org (дата звернення: 08.02.2023)

17. Лебеденко М.С., Лученко І.В. Веб-ресурс як ефективний інструмент маркетингової комунікації. *Вісник Хмельницького національного університету. Економічні науки.* 2011. № 2. Т. 1. С. 178-182.

18. Клушин Ю.С., Захарчин Ю.Б. Підвищення швидкості роботи веб-додатків. *Комп'ютерні системи та мережі.* 2020. № 1 (20). С. 33-43. DOI: 10.23939/csn2020.01.033

19. Developer Survey 2021. «Programming, scripting, and markup languages». URL: https://insights.stackoverflow.com/survey/2021#overview

20. Visual Studio Code – Code Editing. Redefined. URL: https://code.visualstudio.com

21. Visual Studio Code – Code Editing. Redefined. URL: https://azure.microsoft.com/ru-ru/products/visual-studio-code/

22. Попхадзе О.А. Розгляд перспективної концепції побудови композитних вебдодатків. *Системи обробки інформації*. 2016. Вип. 5. С. 137-141.

23. Харенко О.О. Дослідження поняття веб-технологій і ролі front-end розробки в формуванні сучасного веб-простору. *Сучасні електромеханічні та інформаційні систем.* 2021. С. 95-101.

24. Кривошеев А.А., Терещенко Т.М. Ускорение работы с HTML и CSS посредством использования плагинов Zen coding (emmet). *Вісник Східноукраїнського національного університету імені Володимира Даля*. 2014. № 6 (213). С. 168-171.

25. Sass: Syntactically Awesome Style Sheets. URL: https://sass-lang.com

26. Щербакова М.Є. Функціональні особливості JavaScript-додатків *Вісник Східноукраїнського національного університету імені Володимира Даля*. 2014. № 10. С. 142-146.

27. Посвістак В.С., Мірошниченко Д.В., Демківська Т.І. Архітектура програмного забезпечення для системи конвертації форматів на базі Telegram Bot API. URL: https://er.knutd.edu.ua/handle/123456789/16433.

28. Петренко О.М., Клименко С.В., Поляков Г.О. Клієнт-серверна система для безпечного обміну приватними повідомленнями із застосуванням криптографії з відкритим ключем. *Будівництво, матеріалознавство, машинобудування.* 2017. С. 177-182.

29. Балик Н.Р., Мандзюк В.І. Сучасні клієнт-серверні технології та їх застосування при вивченні систем управління базами даних. *Науковий часопис НПУ імені М.П.Драгоманова. Серія 2. Комп'ютерно-орієнтовані системи навчання*. 2007. № 5 (12). С. 151-154.

30. Касім А.М. Склад і структура клієнт-серверної системи моделювання поведінки навігаційно-моніторингових комплексів змінного призначення. *Математичні машини і системи*. 2016. № 4. С. 54-67.

31. Риндич Є.В., Коняшин В.В., Зайцев С.В., Усов Я.Ю. Особливості створення мережевої системи виявлення вторгнень у комп'ютерні системи. *Математичні машини і системи.* 2018. № 3. С. 89-96.

32. Олешко Т.І., Бахорчук В.О. Моделювання бізнес-процесів за допомогою діаграми випадків використання. *Вчені записки Таврійського національного університету імені В.І.Вернадського, серія «Економіка і управління».* 2020. Т. 31(70). № 3. С. 178-184.

33. Новицький О.В. Розширення UML специфікації для моделювання семантичних об'єктів. *Проблеми програмування.* 2016. № 2-3. С. 211-219.

34. Заїка А.В., Філенко М.І., Остапченко А.С., Григорова Т.А. Моделювання архітектурних рішень підтримки мультисайтовості для організації

інформаційних систем. *Вісник Кременчуцького національного університету імені Михайла Остроградського.* 2015. Вип. 3 (92). С. 54-59.

35. Полотай О.І. Використання діаграми класів UML для запровадження освітніх ІТ-проектів у ВНЗ. *Торгівля, комерція, підприємництво*. 2011. Вип. 13. С. 111-115.

36. Великодний С.С., Бурлаченко Ж.В., Зайцева-Великодна С.С. Розробка архітектури програмного засобу для управління мережевим плануванням реінжинірингу програмного проекту. *Сучасний стан наукових досліджень та технологій в промисловості.* 2019. № 2 (8). С. 25-35.

37. Великодний С.С. Метод представлення оцінки реінжинірингу програмних систем за допомогою проектних коефіцієнтів. *Сучасний стан наукових досліджень та технологій в промисловості.* 2019. № 1 (7). С. 34-42.