МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____Аліна САВЧЕНКО

"____"_____2021 р.

ДИПЛОМНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИЦІ ОСВІТНЬОГО СТУПЕНЯ
"МАГІСТРА"

ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: "Веб-застосунок «Резюме-Портфоліо» на основі JavaScript"

Виконавиця: Тультова Маргарита Сергіївна

Керівник: к. т. н., доцент Савченко Аліна Станіславівна

Нормоконтролер: _____ Ігор РАЙЧЕВ

Київ – 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра Комп'ютерних інформаційних технологій

Галузь знань, спеціальність, освітньо-професійна програма:12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології".

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____Аліна САВЧЕНКО
«____»_____2021р.

ЗАВДАННЯ

на виконання дипломної роботи студентки

Тультова Маргарита Сергіївна
(прізвище, ім'я, по батькові)

1. Тема роботи: «Веб-застосунок «Резюме-Портфоліо» на основі JavaScript» затверджена наказом . ректора від 12.10.2021 за № 2228/ст.

2.Термін виконання роботи: з 12.10.2021 по 31.12.2021.

3.Вихідні дані до роботи: теоретичні відомості та основи проектування інформаційних систем, опис використаного програмного забезпечення для створення веб-застосунку, опис використаних мов прогамування JavaScript, HTML та CSS.

4.Зміст пояснювальної записки: вступ, актуальність та постановка задачі для розробки, аналіз можливих методів та засобів для розробки пропонованого продукту, розробка та експлуатація продукту, висновки.

5. Перелік обов'язкового ілюстративного матеріалу: слайди, презентація.

6. Календарний план-графік

№ п/п	Завдання	Термін виконання	Підпис керівника
1.	Аналіз літератури та джерел за темою дипломного проєкту	12.10.2021 – 15.10.2021	
2.	Розробка та затвердження плану дипломного проєкту	16.10.2021 – 19.10.2021	
3.	Проведення консультацій з науковим керівником, щодо створення першого розділу	20.10.2021 – 24.10.2021	
4.	Аналітичний огляд та постановка задачі	25.10.2021 – 31.10.2021	
5.	Написання Розділу 1 дипломної роботи.	01.11.2021 – 07.11.2021	
6.	Розробка та реалізація програмного веб-застосунку.Написання Розділу 2 дипломної роботи.	08.11.2021 – 17.11.2021	
7.	Написання Розділу 3 дипломної роботи. Завершення створення пояснювальної записки дипломної роботи.	18.11.2021 – 01.12.2021	
8.	Оформлення та друк пояснювальної записки.	02.12.2021 – 11.12.2021	
9.	Створення презентації, доповіді та підготовка до захисту дипломної роботи	12.12.2021 – 20.12.2021	

7. Дата видачі завдання: 12.10.2021р.

Керівник дипломної роботи_____ Аліна САВЧЕНКО
(підпис керівника)

Завдання прийняла до виконання_____ Маргарита ТУЛЬТОВА
(підпис випускниці)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи “Веб-застосунок Резюме-портфоліо” складається із вступу, трьох розділів, загальних висновків, списку використаних джерел і містить 80 сторінок тексту, 37 рисунків та 1 таблицю.

Список використаних джерел містить 10 найменувань.

Метою роботи є розробка веб-застосунку, для представлення навичок роботи для майбутніх роботодавців. Також оцінка специфіки діяльності Резюме-портфоліо, його відмінності від звичайного портфоліо та резюме.

Предметом дослідження є розробка веб-застосунку використовуючи мову програмування Java Script на платформі .Nodejs.

Об’єктом дослідження є процес відображення представлення персональної інформації в інформаційному середовищі.

Ключові слова:ВЕБ ДОДАТОК, РЕЗЮМЕ-ПОРТФОЛІО, JAVASCRIPT, REACT, NODE.JS, CSS.

ЗМІСТ

ВСТУП

Останнім часом зростає потреба в «самопрезентації», яка б демонструвала сильні сторони людини. Цей матеріал може стати в нагоді при вступі до навчального закладу, працевлаштуванні, отриманні договору про співпрацю з сучасним цифровим проектом.

Одного навику буде недостатньо, щоб роботодавець повністю оцінив кандидата. Потрібно вміти правильно представити свою кандидатуру та продемонструвати найкращу роботу. У цьому обов’язково допоможе портфоліо – складання зразків робіт, сайтів, брендингу, добре висвітлених послуг компанії (фізична особа) або спеціаліста (модель, фотограф, дизайнер, програміст). «Портфоліо» — це спосіб групування власних досягнень, які повністю зосереджують увагу замовника на професійних навичках.

Кар'єрні портфоліо допомагають документувати освіту, зразки роботи та навички. Люди використовують кар’єрні портфоліо, щоб подати заявку на роботу, вступити в коледж або навчальні програми. Вони більш глибокі, ніж резюме, яке використовується для узагальнення вищесказаного на одній або двох сторінках. Кар’єрні портфоліо є доказом навичок, здібностей та потенціалу людини в майбутньому.

Кар'єрні портфоліо допомагають влаштуватися на роботу або вступити до вищих навчальних закладів. Кар'єрне портфоліо має бути особистим і містити важливу інформацію. Пункти, які мають бути включені, включають (але не обмежуються) особисту інформацію, оцінки, зразки роботи, діловий портрет, а також нагороди та подяки. Кар'єрні портфоліо часто зберігаються в простій папці з трьома кільцями або в Інтернеті як електронне портфоліо та часто оновлюються. Кар'єрне портфоліо використовується як інструмент маркетингу для продажу себе для особистого просування. У деяких галузях роботодавці або приймальні комісії зазвичай вимагають кар’єрного портфоліо, тому буде розумною ідеєю мати оновлене портфоліо під рукою.

РОЗДІЛ 1
АКТУАЛЬНІСТЬ ТА ПОСТАНОВКА ЗАДАЧІ ДЛЯ РОЗРОБКИ

1.1. Аналіз предметної області

У 21 столітті веб-технології почали зачіпати навіть портфоліо, особливо на цифровому ринку праці.

Все більше і більше тих, хто шукаючи роботу, створюють веб-сайти з власними брендами, щоб підтвердити та відзначити свої навички, досягнення та досвід. Різні відомі веб-сайти, такі як LinkedIn чи навіть Twitter, стали популярними в основному через можливість такої послуги на веб-сайті, як розміщення порт фоліо для клієнта. Ці веб-сервіси надають користувачам інструменти для включення всіх форм цифрових медіа на свої веб-сайтах, включаючи документи, зображення, відео та аудіофайли.

Стрічка з резюме та демонстраційні стрічки є різновидом портфоліо. Їх використовують часто в мистецтві, наприклад, музиканти, актори, художники і навіть журналісти. Творчі професіонали також шукають веб-сайти портфоліо, як засіб представити свою роботу більш професійно та елегантно.

1.2. Постановка задачі

Сучасні роботодавці все частіше вказують на наявність портфоліо в числі обов'язкових вимог до кандидата. Для певних професії резюме без атрибутивних даних не розглядаються. Тому потрібно розуміти, кому не обійтися без портфоліо та як його створити самостійно. Часто самостійно створити портфоліо нескладно, зовсім необов'язково користуватися послугами спеціалістів. Хоча іноді без нього не обійтись. Наприклад, якщо ви модель, вам знадобляться якісні професійні фото, які ви замовляєте у фотографа. Якщо є всі необхідні комплектуючі, головне дотримуватися порядку формування.

Всі документи розташовані в хронологічному порядку їх створення – так ви продемонструєте своє професійне зростання, здатність до розвитку.Інший спосіб розміщення - за жанром, напрямом, стилем. Якщо ви вирішили вибрати цей спосіб, то на початку та в кінці розмістіть найбільш вигідну роботу. Психологія сприйняття така, що перше та останнє слово має вирішальне значення. Складання резюме залежить від специфіки професії, але у будь-якому разі необхідно включати лише найкращі роботи.

Бажано не обмежуватися одним напрямком, а продемонструвати своє вміння працювати у різних обличчях. Різним людям це завжди подобається більше. Але врахуйте специфіку вакансії, не розбігайтеся, якщо спеціальність вузька, а компанія шукає людину, яка добре знається на тій чи іншій дієті. Головне – не намагайтеся обдурити, не дивіться чужі роботи за свої.

1.3. Структура портфоліо

Портфоліо означає в буквальному перекладі портфель з документацією. Але якщо у вимогах роботодавця зазначено цей пункт, він означає систематизовану демонстрацію виконаних робіт. Простіше кажучи, це приклади ваших робіт, які покажуть, чи ви достатньо компетентні і кваліфіковані, щоб виконувати те чи інше

завдання. Мета портфоліо – додати реалістичність портрету претендента,навички. Вміст портфоліо може включати різні елементи:

- відеофайли;
- фотографії;
- аудіозаписи;
- публікації;
- посилання на ті чи інші роботи.

Все залежить від діяльності. Портфоліо потрібно далеко не всім людям, хто шукає роботу. Портфоліо може бути представлене у двох видах.

Папір. Це роздруковані роботи будь-якого виду, збережені в папці. Сьогодні цей різновид стає рідкісним, тому що набагато простіше докласти посилання на електронний файл. З іншого боку, паперове портфоліо дозволить вам не залежати від жодних збоїв, ви завжди будете «у всеозброєнні».

Електронне. Найбільш поширене портфоліо в цифровому форматі, яке може розміщуватися на ПК, на біржах, блогах, віртуальних дисках для зберігання інформації. Якщо портфоліо відкрите і перебуває у широкому доступі, фахівця можуть помітити і запропонувати роботу, спираючись на враження з його робіт.

Багато людей плутає резюме та портфоліо, насправді між цими поняттями є суттєва різниця. Це не взаємозамінні речі. Портфоліо - це паперовий або електронний документ, який доповнює резюме, але не замінює його. Подібності немає, головна відмінність у тому, що одного портфоліо замало, щоб влаштуватися на роботу. Резюме роботодавця може бути достатньо, щоб оцінити ваш досвід та здібності. Портфоліо, окрім інформації про людину, наочно демонструє її талант, досвід, попередні досягнення. Тому необхідно не тільки якісне, повне та резюме, а й доповнювати його маючи порт фоліо.

1.3.1. Портфоліо в IT

Пошук першої роботи в IT – це окрема робота. Зазвичай новачки відразу губляться після фрази «будь ласка, надішліть портфоліо до резюме». Незрозуміло, що роботодавець хоче бачити, як це має виглядати, де брати проєкти, які потрібно показувати, а які ні.

Загалом портфоліо є зручним інструментом для обох сторін. Для спеціаліста це підтвердження його практичних навичок та підтвердження досвіду. Починаючи з молодших +/середніх позицій, це дозволяє пасивно шукати роботу і знімає деякі питання з тестових завдань. Портфоліо допомагає роботодавцю переконатися, що він отримує не просто теоретика, який знає всі інструменти, але ніколи не використовував їх на практиці, а людину, яка вже вміє щось робити своїми руками. Виявляється, що портфоліо виключає тих, хто прагне до IT, просто прочитавши кілька статей.

Портфоліо може багато розповісти про професійні якості кандидата. Наприклад, ось на що звертає увагу роботодавець, наймаючи розробника. Скільки часу витрачається на розробку та кодування. Погано, коли входиш у портфоліо початківця розробника, а проектів мало і остання активність була вісім-дев’ять місяців тому. Якість коду. Це має бути зрозуміло для всіх членів команди.

Увага до дрібниць. За проектами в портфоліо можна припустити, чи зможе кандидат підготувати документацію, чи зможе пояснити, що було зроблено в проекті, і передати власні знання.

Для яких напрямків в IT-портфоліо потрібно. Особливо це стосується розробників, спеціалістів з обробки даних та інженерів ML, аналітиків, дизайнерів, менеджерів з продуктів. Але яким би зручним портфоліо не було для оцінювання кандидатів, не всі спеціальності мають можливість його надати.

Наприклад, нелегко подумати про презентацію проєктів тестувальнику. Так, ви можете детально написати про об’єкт тестування та спроектувати знайдені баги. Але роботодавцю все одно потрібно якось оцінити важливість і правильність виявлених недоліків, а це досить складно зробити, коли товар незнайомий. Тому при відборі

кандидатів у тестувальники роботодавець здебільшого орієнтується на тестове завдання та відповіді на співбесіді.

Схожа історія з системними адміністраторами, інженерами DevOps і всією сферою експлуатації та моніторингу. У таких випадках в резюме зазвичай вказується основна інформація про попередній досвід: опис завершених проектів, приблизний розмір сервісів і підтримуваних платформ, перелік розроблених технологій.

Для початківця фахівця портфоліо служить підтвердженням практичних навичок, тому його головною складовою є проекти. Важливо, щоб вони включали обгрунтування того, які інструменти вибрано і чому, цілі, показники оцінки та очікувану функціональність були прописані. Неможливо оцінити проект, який знаходиться у вакуумі. Тому в роботі обов'язково потрібен контекст і висновок про те, наскільки ефективно вдалося вирішити поставлене завдання.

Єдиного шаблону для оформлення портфоліо не існує. Зазвичай їх завантажують на онлайн-платформи, які вже підібрані для цільової аудиторії та сформували професійну спільноту:

Головне питання для новачка в IT – як заповнити портфоліо. Зазвичай на вакансії, навіть для стартових посад, потрібно мати вже два-три завершених проекти. І якщо дизайнерам простіше знайти проекти, то для розробників-початківців частіше виникає питання, де шукати проекти та ідеї для них, а фріланс підходить далеко не завжди.

Один з робочих варіантів — спробувати автоматизувати деякі ваші щоденні дії. Наприклад, створіть калькулятор витрат із виводом статистичних даних або програму зі списком справ. Або напишіть чат-бота. Боти можуть бути найрізноманітнішими: прогноз погоди, генератор паролів або пошук найближчих кафе.

Якщо у вас вже є досвід розробки і ви хочете чогось серйознішого, то можете взяти участь у проектах з відкритим кодом. Найкраще зосередитися на розробці програмного забезпечення, яким ви постійно користуєтеся, оскільки знайомство з програмним забезпеченням допомагає зрозуміти його особливості.

Коли в скарбничці вже накопичилася пара проектів і почався активний етап пошуку роботи, з’являється ще одне джерело – тестові завдання. Їх присутність може добре диверсифікувати ваше портфоліо. Об’ємне тестове завдання можна завантажити на ваш GitHub з відредагованим описом і без вказівки конкретної компанії. Немає потреби окремо вказувати, що це тест. Тут важлива ваша реалізація.

Не варто додавати все до свого портфоліо – це може мати негативні наслідки і навіть зіпсувати враження про кандидата. Повністю студентські проекти краще не розміщувати. Десятки коротких домашніх завдань із пройдених курсів не дадуть потенційному роботодавцю додаткової інформації. Він давно знайомий з основними інструментами і чекає від вас чогось більш цікавого. Крім того, перший контакт з портфоліо часто буває коротким. Тому серед великої кількості невиграшних робочих місць дійсно якісні проекти можуть просто загубитися.З тих же причин не варто публікувати незавершені проекти, навіть якщо вони здаються дуже крутими.

Надмірні коментарі в коді — ще один момент, на який варто звернути увагу. Іноді після перевірки студенти забувають видалити отримані коментарі і разом з ними завантажити проекти на GitHub. В результаті залишені коментарі відволікають від самого коду і створюють враження, що людина не обробив зворотний зв’язок або не сприйняв портфоліо всерйоз. Найголовніше, не додавайте до свого портфоліо роботи інших людей.

Зрозумійте структуру веб-сервісів для розміщення портфоліо.Створіть свій перший проект і спробуйте розмістити його на обраному вами сайті.Візьміть участь у проекті з відкритим кодом, щоб урізноманітнити своє портфоліо.Заповніть опис завантажених проектів. У розробці це чудово підходить для тренування навику написання документації.Покажіть свою роботу комусь у спільноті. У багатьох спеціалізованих групах можна скинути посилання на проект і попросити коментар.Коли у вас вже накопичилося чотири-п’ять крутих проектів, можна створити сайт-візитку. Це посилить враження від вашого портфоліо.

1.4. Персональний сайт

Персональний веб-сайт - один з найважливіших активів, які ви можете отримати. Це не тільки дає вам повний контроль над своїм ім'ям та особистим брендом, але також підвищує вашу помітність та спрощує для роботодавців або навіть клієнтів можливість знайти вас, коли вони шукають певного таланту або навички.

Ви можете використовувати індивідуальний сайт, якщо ви закінчили коледж та шукаєте роботу. Ви також можете використовувати його, якщо намагаєтеся проникнути у світ фрілансу та хочете залучити потенційних клієнтів. Але як ви маєте намір його створити?

Один із найпростіших способів створити особистий сайт – це робота з професійними HTML-шаблонами сайтів. Ще краще взяти шаблон резюме у форматі HTML, розроблений спеціально для створення особистого веб-сайту. Це сучасний формат для онлайн-дизайну резюме на веб-сторінці. Іншими словами, це особистий сайт у вигляді резюме або портфоліо, який містить досвід роботи, освіту, навички, мови, досягнення, можливо, приклади роботи і навіть рекомендації.

За допомогою резюме на сайті ви зможете не тільки виділитися на тлі інших кандидатів, але й запам’ятатися роботодавцю. Це, безумовно, надійний спосіб отримати роботу своєї мрії.

А найголовніше, що написати резюме в такому форматі можна дуже швидко, просто і безкоштовно. У цій статті ми покажемо вам, як скласти резюме для успішної кар’єри.

Рекрутери витрачають в середньому 6 секунд на перегляд свого резюме. Якщо резюме кандидата викликало інтерес, з вами зв’яжуться. Але найчастіше їх навіть не повідомляють. Тому сайт резюме – найкраща альтернатива звичайному файлу Word або PDF. «Коли я дивлюся на резюме, перше, на що я дивлюся, це його структура та форматування. Якщо це проблема, я довго не розглядатиму це резюме. ”- Джеймі Хіченс, старший менеджер із залучення талантів. Згідно з дослідженням rabota.ua, середня тривалість перегляду резюме за розділами виглядає так (рис.1.1):

Секция резюме	Время просмотра
Личная информация	до 3 секунд
Название должности	1 секунда
Карьерные цели	до 5-8 секунд
Ключевая информация	до 12-15 секунд
Последнее или нынешнее место работы	15-17 секунд
Предыдущее место работы	8-12 секунд
Более раннее место работы	5-8 секунд
Четвертое место работы и далее	3-6 секунд
Образование	до 3-5 секунд
Языки	3-5 секунд
Курсы, тренинги, сертификаты	5-8 секунд
Дополнительная информация	7-10 секунд

Рис.1.1. Статистика перегляду сайту резюме

1.5. Односторінковий сайт та лендинг

Лендінг — це посадкова сторінка, яка спонукає людину вчинити цільову дію: купити продукт, оформити замовлення, забронювати квиток на захід, взяти участь у вебінарі тощо. Сайт має комерційний характер — залучити трафік, отримати контактні дані користувачів та збільшити прибуток.

Односторінковий сайт — це ресурс, який складається з єдиної сторінки та закріплений за одним URL.

Мета: залучення та інформування користувачів. Не кожен односторінок є лендінгом.

Односторінники ділять на кілька видів, давайте зупинимося на них докладніше. Оскільки ми вже розглянули поняття «лендінг», його у списку не буде.

Сайт візитка. Інформаційний ресурс, на якому розміщують інформацію про компанію та її діяльність. Іншими словами, це перенесена з офлайну в онлайн-візитівка, тільки з більш докладним описом.

Організація зможе будь-якої миті відправити посилання на сайт і допомогти людині ознайомитися зі своїми можливостями. Зазвичай на подібних односторінниках є вся інформація про фірму, її контактні та реєстраційні дані, список переваг та варіанти продукції.

Якщо користувача зацікавила інформація, він може зателефонувати або написати електронний лист. Також на сайті часто є форма зворотного зв'язку, куди людина може додати свої контактні дані. Подібними форматами користуються фахівці у певній галузі — наприклад психологи або digital-агентства.

Портфоліо. Сайт, на якому автор розміщує свої роботи, щоб показати свій професіоналізм. Наприклад, фотограф показує свої варіанти зйомки, а програміст у такий спосіб може позначити свої навички — працював із C#, написав фрагмент коду для проекту.

Сайт-опитувач, на якому користувачеві пропонується відповісти на кілька запитань та залишити свої контактні дані. Наприкінці на нього завжди чекає подарунок — чек-лист, безкоштовний доступ до програми, VIP-підписка тощо. Це потрібно для того, щоб привернути увагу людини та зібрати її контакти.

Інформація може використовуватись для збирання цільової аудиторії та складання портретів клієнтів. Інформаційна сторінка. Ресурс, на якому міститься інформація про захід, подію, новий товар або послугу. Сайт створений для того, щоб підігріти інтерес аудиторії.

Висновки до розділу 1

Враховуючи наведену вище статистику, можна прийти до висновку, що найкращим варіантом є сайт резюме. Дійсно, на веб-сторінці ви можете чітко структурувати всю інформацію, яку ви хочете повідомити роботодавцю про себе.

Крім того, онлайн-резюме може легко стати креативним і технічно підкованим кандидатом. Саме ці якості дуже високо цінуються при відборі на роботу в 2021 році.Нарешті, за допомогою сайту персонального резюме ви можете створити свій особистий бренд в Інтернеті. Роботодавці чи клієнти зможуть самостійно знайти ваше онлайн-резюме чи портфоліо, шукаючи спеціаліста у вашому профілі.

Так ви зможете не тільки знайти високооплачувану роботу, а й розпочати власний бізнес – спочатку працювати фрілансером, а з часом можна відкрити власне агентство.

Цей розділ обґрунтовує актуальність розробки веб-сайту вашого портфоліо. Доведено, що ваш сайт є хорошим інструментом для розвитку фахівця та його просування в цій сфері. Клієнт у пошуках виконавця свого проекту зазвичай заходить в пошукову систему і вводить ключові запити на кшталт «дизайнер Україна», «брендинг», «зробити презентацію». Розроблений сайт може потрапити в топ і привернути увагу потенційних клієнтів. Для того, щоб з’явитися в пошукових системах і залучити роботодавців, сайт дизайнера повинен виділятися серед конкурентів.

РОЗДІЛ 2
АНАЛІЗ МОЖЛИВИХ МЕТОДІВ ТА ЗАСОБІВ ДЛЯ РОЗРОБКИ ПРОПОНОВАНОГО ПРОДУКТУ

2.1. Веб- додаток

Веб-додаток – це клієнт-серверний додаток, де відбувається взаємодія між клієнтом та веб сервером через браузер. Веб-додатки розвивались та перетворювались на складні відповідно до часу. Ця складність може бути з точки зору динамічного характеру, використання мультимедіа, продуктивності або багатьох інших способів. Традиційні програми були простими, менш складними, статичний вміст, обмежене використання та рівень безпеки був мінімальним. Не було сфери своєчасного оновлення, мали обмежений функціонал та інтерактивність. Розширені веб-програми постачаються з динамічним вмістом, містять велику інформацію, просту в інтеграції, розклад та планування.

Логіка веб-додатку розподілена між клієнтом та сервером, на сервері зазвичай здійснюється зберігання даних, а по мережі проходить обмін інформацією. Перевагою такого підходу є те, що клієнти не залежать від конкретної операційної системи користувача.

Веб-додатки дозволяють відвідувачам швидко та легко знаходити потрібну їм інформацію на веб-сайтах з великим обсягом інформації. Даний вид веб-додатків дозволяє здійснювати пошук у вмісті, впорядковувати вміст і переміщатися по ньому зручним для відвідувачів способом. Веб-додаток дозволяє зберігати дані безпосередньо в базі даних, а також отримувати дані і формувати звіти на основі отриманих даних для аналізу.

2.1.1. Структура веб додатку

Зазвичай веб-додатки складаються як мінімум з трьох основних компонентів (рис.2.1):

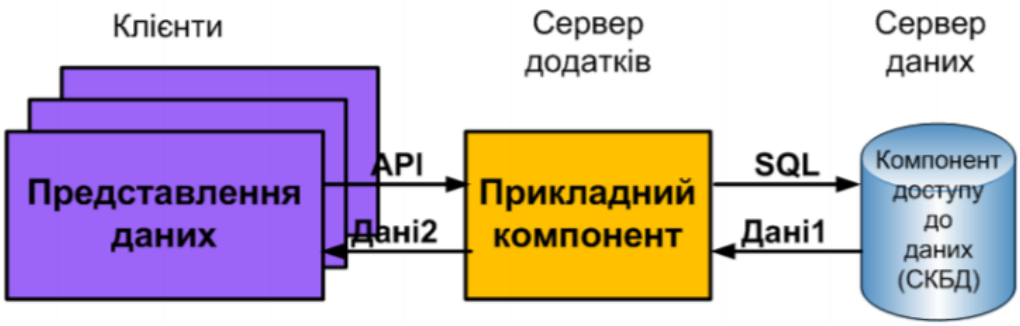

Рис.2.1. Компоненти веб додатку

1) Клієнтська частина веб додатку - це графічний інтерфейс. Це та частина, яку можна побачити на сторінці. Графічний інтерфейс відображається в браузері. Користувач взаємодіє з веб-додатком саме через браузер, клацаючи на посилання. клієнтський компонент розроблений у CSS, HTML та JS. Оскільки він існує у веб-браузері користувача, немає необхідності в налаштуваннях операційної системи чи пристроїв.

2) Серверна частина веб-додатку - це програма або скрипт на сервері, яка оброблює запити користувача (точніше, запити браузера). Запит відправляється кожен раз, як тільки користувач переходить за посиланням в браузері. Далі відбувається обробка сервером цього запиту, викликаючи деякий скрипт, який формує веб-сторінку, описану мовою HTML, а потім відсилає клієнту по мережі. Браузер одразу відображає отриманий результат у вигляді веб-сторінки.

3) База даних (БД, або система управління базами даних, СУБД) - програмне забезпечення на сервері, що займається зберіганням даних і їх видачею в потрібний момент. База даних розташовується на сервері.

2.1.2. Архітектура веб- додатку

Архітектура веб-додатків описує взаємодію між додатками, базами даних та системами проміжного програмного забезпечення в Інтернеті (рис.2.2). Це забезпечує одночасну роботу декількох додатків. Як тільки користувач натискає кнопку переходу після введення URL-адреси в адресному рядку веб-браузера, він запитує саме цю веб-адресу. Сервер надсилає файли в браузер як відповідь на зроблений запит. Потім браузер виконує ці файли, щоб показати потрібну сторінку.Нарешті, користувач може взаємодіяти з веб-сайтом.

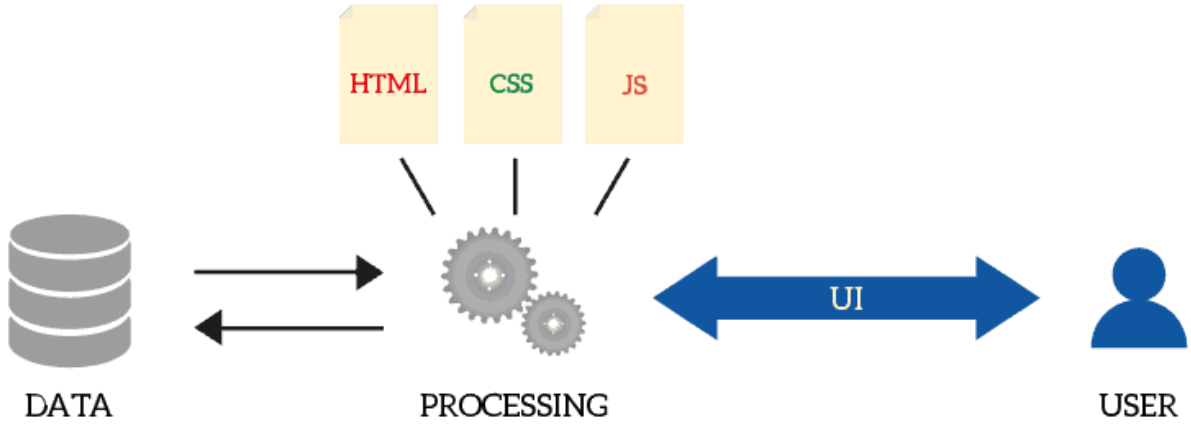

Рис.2.2. Архутектура веб-додатку

Архітектура веб-додатків є незамінною у сучасному світі, оскільки значна частина глобального мережевого трафіку, а також більшість додатків та пристроїв використовують використання веб-комунікацій. Архітектура веб-додатків має стосуватися не лише ефективності, але й надійності, масштабованості, безпеки та надійності.

У будь-якому типовому веб-додатку є два різних коди (підпрограми), що працюють поруч. це: код на стороні клієнта - код, який знаходиться в браузері і відповідає на деякий ввід користувача. Код на стороні сервера - код, який знаходиться на сервері і відповідає на HTTP-запити.

Тип архітектури веб-додатків залежить від розподілу логіки програми між сторонами клієнта та сервера.Існує три основні типи архітектури веб-додатків. Кожен з них пояснюється так:

Програми на одній сторінці - замість завантаження абсолютно нових сторінок із сервера щоразу для дії користувача, веб-додатки з однією сторінкою забезпечують динамічну взаємодію шляхом надання оновленого контенту на поточну сторінку.Це епоха мінімалізму, коли більш популярна односторінкова веб-програма. Найбільш затребувані програми включають лише необхідні елементи вмісту.Це пропонує більше інтерактивного користувальницького досвіду, що дозволяє веб-додатку на одній сторінці та користувачеві мати більш динамічну взаємодію.

Мікросервіси - це невеликі та легкі сервіси, які виконують єдиний функціонал. Рамка архітектури мікросервісів має ряд переваг, що дозволяє розробникам не тільки підвищити продуктивність, але й прискорити весь процес розгортання.Компоненти, що складають додаток, створений за допомогою архітектури мікросервісів, не залежать безпосередньо один від одного. Таким чином, їх не потрібно створювати за допомогою однієї мови програмування.Виконання єдиного та специфічного функціоналу за допомогою Microservices Architecture Framework дозволяє розробникам швидше і з більшою ефективністю розгортати програми.Оскільки різні компоненти розробляються різними мовами кодування, більша гнучкість у виборі технології вибору.

Отже, розробники, що працюють з архітектурою мікросервісів, можуть вільно підібрати набір технологій. Це робить розробку програми простішою та швидшою.

Безсерверна архітектура - У цьому типі архітектури веб-додатків розробник додатків звертається до сторонніх постачальників послуг хмарної інфраструктури для аутсорсингу сервера, а також управління інфраструктурою.Перевага такого підходу полягає в тому, що він дозволяє програмам виконувати логіку коду, не турбуючись із завданнями, пов'язаними з інфраструктурою.Архітектура без сервера найкраще, коли компанія-розробник не хоче керувати або підтримувати сервери, а також обладнання, для якого розробили веб-додаток.Це дозволяє програмам виконувати без співвідношення завдань, пов’язаних з інфраструктурою, коли

розробникам не потрібно керувати задніми серверами, працюючи над сторонніми інфраструктурами.Спосіб планування цієї взаємодії визначає стійкість, ефективність та безпеку майбутнього веб-додатка.

Отже, архітектура веб-додатків повинна включати всі підкомпоненти, а також обміни зовнішніх додатків для всього програмного забезпечення у вищезгаданому випадку, який є веб-сайтом.

2.3. JavaScript

JavaScript — мова підготовки сценаріїв, що дозволяє зробити Web-сторінки більш інтерактивними та функціональними.

JavaScript був створений, щоб "оживити веб-сторінки".Програми цією мовою називаються скриптами. Їх можна записати в HTML веб-сторінки та запустити автоматично під час завантаження сторінки.Сценарії надаються та виконуються, як звичайний текст. Для запуску їм не потрібна спеціальна підготовка чи компіляція.

Області застосування JavaScript в клієнтському програмуванні дуже широкі, а велика кількість версій мови з різними функціональними можливостями дозволяє дуже гнучко підходити до вирішення завдань, що стоять перед розробником. Сьогодні JavaScript може виконуватись не тільки в браузері, але і на сервері або фактично на будь-якому пристрої, який має спеціальну програму під назвою механізм JavaScript.

Можливості JavaScript значною мірою залежать від середовища, в якому він працює. Наприклад, Node.js, підтримує функції, які дозволяють JavaScript читати / записувати довільні файли, виконувати мережеві запити тощо. В браузері JavaScript можна робити все, що стосується маніпуляції веб-сторінками, взаємодії з користувачем та веб-сервером.

Мова JavaScript містить типи даних: Undefined (даних нема), Null (нульовий), String (рядковий), Number (числовий), Boolean (логічний) і Object (об'єктний), зображені на (рис.2.3). Це відносно незначна кількість типів дозволяє, тим не менше, створювати повноцінні сценарії для виконання багатьох функцій.

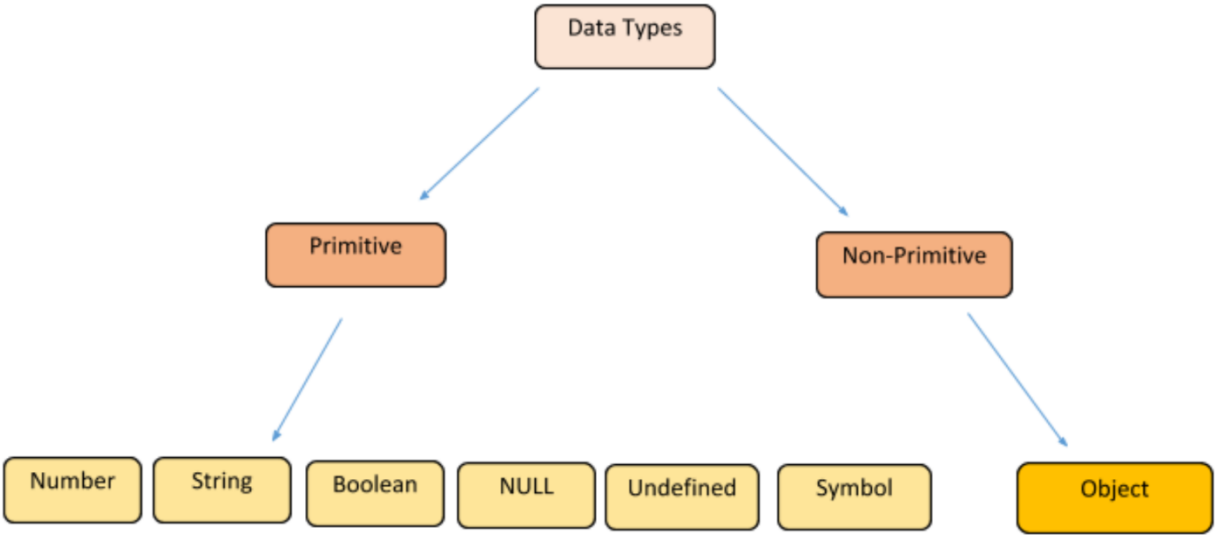

Рис.2.3. Типи даних JavaScript

При розробці веб-сайту використовувались такі типи даних, як: String (рядковий) і Boolean (логічний).

2.3.1. Унікальність JavaScript

Є щонайменше три чудові речі щодо JavaScript:

- Повна інтеграція з HTML / CSS.
- Прості речі робляться просто.
- Підтримка всіх основних браузерів та включена за замовчуванням.

JavaScript - єдина технологія браузера, яка поєднує ці три речі.

Двигуни JavaScript спочатку використовувались лише у веб-браузерах, але тепер вони вбудовані в деякі сервери, як правило, через Node.js. Вони також вбудовані в різноманітні програми, створені за допомогою таких систем, як Electron та Cordova.

Для забезпечення високого рівня абстракції при розробці веб-додатків використовуються бібліотеки JavaScript. Вони являють собою набір багаторазово використовуваних об'єктів і функцій. Серед відомих JavaScript бібліотек можна

відзначити React.js, Vue.js, Ember.js, AngularJS, Dojo, jQuery, і Node.js. Angular Framework був створений Google, аналогічно, Facebook створив структуру React для свого веб-сайту і пізніше випустив її як відкритий код; інші сайти, включаючи Twitter, зараз використовують його.

Як і будь-який інший мову програмування, JavaScript використовує змінні для зберігання даних певного типу. Реалізація JavaScript є прикладом мови вільного використання типів. У ньому не обов'язково ставити тип змінної. Її тип залежить від типу збережених у ній даних, причому при зміні типу даних змінюється і тип змінної.

JavaScript - мова, керований подіями. HTML-елементи, подібні кнопок, списків або текстових полів, вдосконалені з метою підтримки обробників подій. Більшість написаних на JavaScript кодів якраз і виявляються пов'язаними з тими чи іншими подіями.Гіпертекстова інформаційна система складається з безлічі інформаційних вузлів, безлічі гіпертекстових зв'язків, визначених на цих вузлах і інструментах маніпулювання вузлами і зв'язками. Основна ідея JavaScript полягає в можливості зміни значень атрибутів HTML-контейнерів і властивостей середовища відображення в процесі перегляду HTML-сторінки користувачем. При цьому перезавантаження сторінки не відбувається.На практиці це виражається в тому, що можна, наприклад, змінити колір фону сторінки або інтегровану в документ картинку, відкрити нове вікно або видати попередження.

2.3.2. Таблиці стилів, Рівні CSS

CSS (Cascading Style Sheets) використовується для стилювання та розміщення веб-сторінок - наприклад, для зміни шрифту, кольору, розміру та інтервалу вашого вмісту, розділення його на кілька стовпців або додавання анімації та інших декоративних функцій.

CSS стала революцією у світі веб-дизайну. Конкретні переваги CSS включають:

- управління компонуванням багатьох документів одночасно через єдиний аркуш стилів;
- більш точний контроль компонування;
- застосування різного макета для кожного типу носіїв;
- велика кількість вдосконалених і складних методик.

HTML можна широко використовувати для додавання макетів на веб-сайти. Але Сьогодні CSS підтримується всіма браузерами.HTML використовується для структурування вмісту. CSS використовується для форматування структури вмісту. Коли мережа стала популярною, дизайнери почали шукати альтернативи для додавання макетів до онлайн-документа. Щоб впоратися з попитом, виробники браузерів (на той час Netscape та Microsoft) винайшли нові теги HTML, такі як <font>, відмінні від оригінальних тегів HTML тим, що вони визначають макет, а не структуру.Це призвело до ситуацій, коли оригінальні теги структури, такі як <table>, широко використовувались для розмітки сторінок замість додавання структури до тексту.

Багато інших типів нових тегів, наприклад <blink>, підтримувалися лише певними типами браузерів. CSS дозволяє автору (та користувачеві) визначати, як повинні відображатися документи HTML, або в HTML-документі, або в окремому файлі CSS. Переваг цього багато, і одне з найважливіших полягає в тому, що вам не потрібно постійно писати FONT FACE, FONT COLOR, BGCOLOR тощо у всіх документах HTML. Натомість ви можете мати всю інформацію про макет у кількох

CSS-файлах, а змінивши один із них, ви можете змінити зовнішній вигляд багатьох HTML-документів. Це також робить розмітку HTML легшою для запису та простішим обслуговуванням сторінок.

Останньою перевагою є те, що він стає менш завантажуваним, щоб швидше завантажуватись. CSS також має суто графічні переваги, оскільки пропонує можливості, які графічно недоступні через HTML. У вас є набагато більше можливостей вирішити, де розмістити елементи, інтервали між елементами, розміри шрифту, кольори тла, обрамлення (не FRAME) тощо.Для цього є і деякі переваги для користувача. А саме, користувач повинен мати можливість вказати власні набори стилів, які застосовуватимуться на додаток до тих, що вказані автором. Якщо автор зробив свою роботу належним чином, документ повинен протистояти йому, не будучи нерозбірливим.

Остання, і, можливо, найважливіша перевага - це те, що набір стилів «витончено деградує». Тобто, якщо у вас є веб-браузер, який не підтримує набори стилів, сторінки, які використовують набори стилів, все одно будуть в ньому читатими, тому що сторінка написана у простому HTML. Це дозволяє реалізовувати CSS без шкоди читачам із застарілим програмним забезпеченням, яке не підтримує CSS. Це також означає, що використання CSS не поширюється на читачів із не графічними веб-браузерами.

Набори стилів CSS складаються зі списку правил стилю. Правило стилю має дві частини: селектор, що вказує, де воно застосовується, та список декларацій, що розповідає про форматування. Селектор може застосовуватися до таких промов, як один конкретний елемент, всі елементи H1 або всі EM-елементи всередині елемента A, загорнута елемент STYLE.Синтаксис у зовнішньому файлі CSS і в елементі STYLE точно такий же, тобто. всіх правил стилю, але в іншому випадку набір стилів повинен складатися тільки з правил стилю та коментарів CSS. Правило стилю виглядає приблизно таке: селектор {декларація; декларація; і т.д. ...} Селектор вказує, до яких елементів застосовуються декларації, оформлені у дужках.

2.4. Поняття Node

Node- це платформа, яка створена для розробки веб – додатків, серверів додатків та просто для програмування.

В основі Node лежить автономна віртуальна машина JavaScript з розширеннями, що робить її придатною для програмування загального призначення з упором на розробку серверів додатків. Node.js дозволяє розробникам використовувати JavaScript для написання інструментів командного рядка та для сценаріїв на стороні сервера - запуску скриптів на стороні сервера для створення динамічного вмісту веб-сторінки до відправки сторінки у веб-браузер користувача. Отже, Node.js представляє парадигму "JavaScript скрізь", що об'єднує розробку веб-додатків навколо однієї мови програмування, а не різних мов для скриптів на сервері та клієнтах.

У Node не вбудована ні об'єктна модель документа (DOM), ні будь-які ще можливості браузера. Саме мова JavaScript в поєднанні з асинхронним введенням / виведенням робить Node потужною платформою для розробки додатків.Крім вбудованого вміння виконувати код на JavaScript, включені до складу дистрибутива модулі надають і інші можливості:

- утиліти командного рядка;
- функції управління процесами для спостереження за дочірніми процесами;
- об'єкт Buffer для роботи з двійковими даними;
- механізм для роботи з сокетомUDP з повним комплектом зворотних викликів у відповідь на події;
- пошук в системі DNS;
- засоби для створення серверів і клієнтів протоколів HTTP і HTTPS, побудовані на основі бібліотеки TCP-сокетів;
- засоби доступу до файлової системи.

2.4.1. Новизна використовувати Node

Мова JavaScript дуже популярна завдяки присутності її будь-якому веб-браузері. Вона ні в чому не поступається іншим мовам, але при цьому підтримує багато сучасних уявлення про те, якою має бути мова програмування.

Один з основних недоліків JavaScript - Глобальний Об'єкт. Всі змінні верхнього рівня «звалюються» до Глобального Об'єкту, і при використанні одночасно декількох модулів це може призвести до неконтрольованого хаосу. Оскільки веб-додатки зазвичай складаються з безлічі об'єктів, можливо, створювалися різними організаціями, то може виникнути побоювання, ніби програмування для Node те саме ходіння по мінному полю. Насправді в Node використовується система організації модулів CommonJS, а це означає, що локальні змінні деякого модуля так і будуть локальними в ньому, нехай навіть виглядають як глобальні. Таке чітке розмежування між модулями вирішує проблему Глобального Об'єкту.

Використання єдиної мови програмування на сервері і на клієнті давно вже було мрією розробників для веб. З появою Node ми, нарешті, зможемо реалізувати мрію - зробити JavaScript мовою, яка використовується по обидва боки веб - на стороні клієнта і сервера.

У єдиної мови є кілька потенційних плюсів:

– одні й ті ж програмісти можуть працювати над обома сторонами додатки;

– загальний для клієнта і сервера формат даних;

– загальний програмний інструментарій;

– загальні для клієнта і сервера кошти тестування і контролю якості;

– на обох сторонах веб-додатки можна використовувати загальні шаблони.

Мова JavaScript не має багатопотокових функцій. Тому робочі потоки Node.js поводяться інакше, ніж традиційна багатопотокова робота в багатьох інших мовах високого рівня.

У Node.js відповідальність працівника полягає в тому, щоб виконати фрагмент коду (робочий сценарій), наданий батьківським працівником. Сценарій Worker тоді працюватиме ізольовано від інших працівників із можливістю передавати повідомлення між ним і батьківським працівником. Робочий сценарій може бути окремим файлом або сценарієм у текстовому форматі, який можна оцінити. У нашому прикладі ми вказали __filename як робочий сценарій, оскільки і батьківський, і дочірній робочі коди знаходяться в одному сценарії, визначеному властивістю isMainThread.

Кожен працівник підключений до свого батьківського працівника через канал повідомлень. Дочірній працівник може писати в канал повідомлень за допомогою функції parentPort.postMessage, а батьківський працівник може писати в канал повідомлень, викликавши функцію worker.postMessage() у робочому екземплярі, що зображено на (рис.2.4).

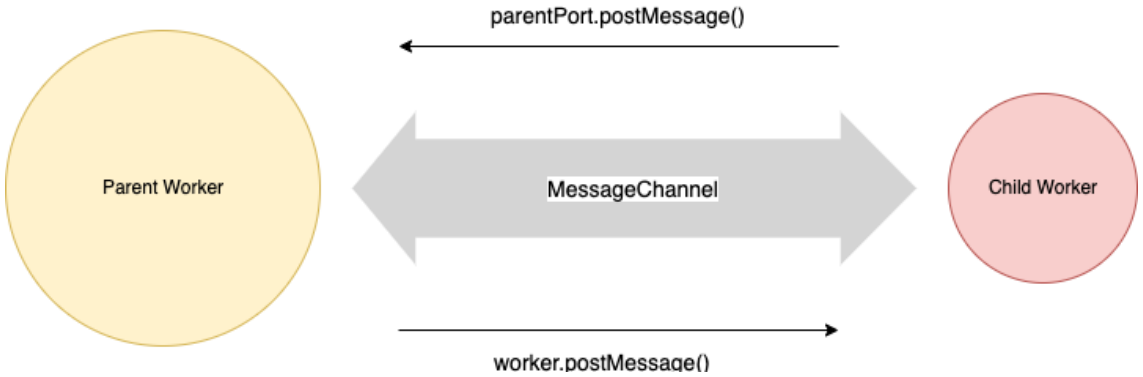

Рис. 2.4. Канал повідомлень між батьками та дочірніми

Як працівники Node.js працюють параллельно, якщо JavaScript не забезпечує паралельність відразу, як два працівники Node.js можуть працювати паралельно? Відповідь: V8 Isolate

Ізолят V8 — це незалежний екземпляр середовища виконання chromeV8, який має власну купу JS і чергу мікрозавдань. Це дозволяє кожному працівнику Node.js виконувати свій код JavaScript повністю ізольовано від інших працівників.

Недоліком цього є те, що працівники не можуть безпосередньо отримати доступ до купи один одного.

Завдяки цьому кожен працівник матиме власну копію циклу подій libuv, який не залежить від циклів подій іншого працівника та батьківського працівника. Створення екземпляра нового обробника та забезпечення зв’язку між батьківським сценарієм JS і робочим сценарієм JS встановлюється реалізацією робочого C++. На момент написання цієї статті це реалізовано в worker.cc.

Реалізація Worker доступна для сценаріїв JavaScript для користувача за допомогою модуля worker_threads. Ця реалізація JS розділена на два скрипти, які я хотів би назвати так:

Сценарій ініціалізації працівника — відповідає за створення екземпляра робочого екземпляра та налаштування початкового зв’язку між батьками й дочірньою роботою, щоб уможливити передачу метаданих працівника від батьківського до дочірнього працівника.

WorkerExecutionscript—Виконує робочий JS-скрипт користувача з наданими користувачем worker Data та іншими метаданими, наданими батьківським працівником. (Рис.2.5.) пояснює це набагато більш зрозуміло.

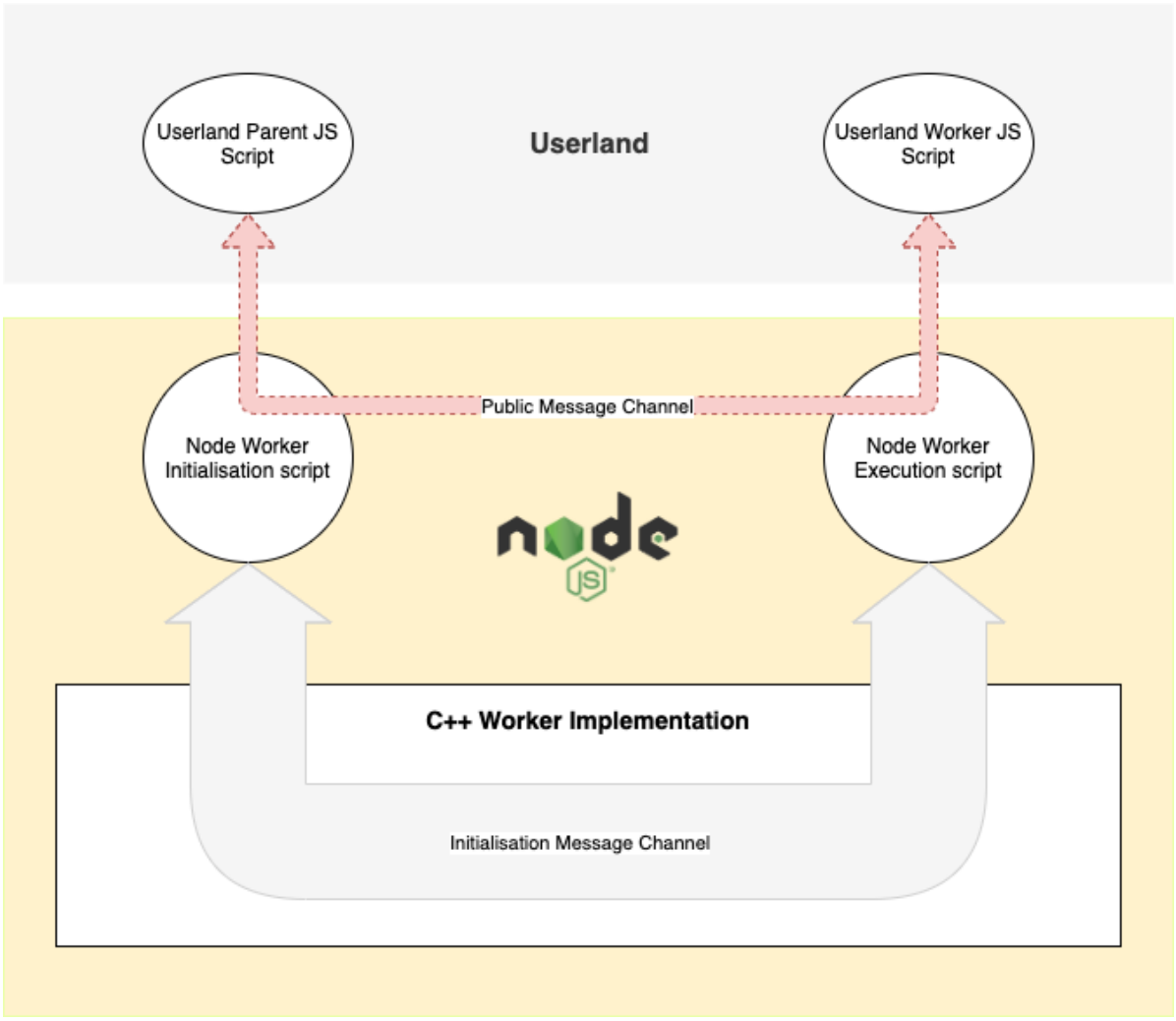

Рис.2.5. Внутрішні органи впровадження Worker

Тепер ми розуміємо, як працюють робочі потоки Node.js. Розуміння того, як вони працюють, допомагає нам досягти найкращої продуктивності за допомогою робочих потоків. При написанні більш складних додатків, ніж наш worker-simple.js, нам потрібно пам’ятати про дві основні проблеми з робочими потоками.

Навіть незважаючи на те, що робочі потоки легші, ніж реальні процеси, робочі породжування потребують серйозної роботи і можуть бути дорогими, якщо їх виконувати часто.

Використовувати робочі потоки для розпаралелювання операцій введення-виводу нерентабельно, оскільки використання власних механізмів введення-виведення Node.js набагато швидше, ніж запуск робочого потоку з нуля лише для

цього. Щоб подолати перше занепокоєння, нам потрібно реалізувати «Об’єднання робочих потоків».

2.4.2. Об’єднання робочих потоків в Node.js

Потік робочих потоків Node.js — це група запущених робочих потоків, які доступні для використання для вхідних завдань. Коли надходить нове завдання, його можна передати доступному працівнику через канал повідомлень «батько-дочірня». Після того, як працівник виконає завдання, він може передати результати назад батьківському працівнику через той самий канал повідомлень.

Після правильного впровадження об’єднання потоків може значно покращити продуктивність, оскільки зменшує додаткові витрати на створення нових потоків. Варто також зазначити, що створення великої кількості потоків також неефективно, оскільки кількість паралельних потоків, які можна ефективно запускати, завжди обмежена апаратним забезпеченням.

На наступному графіку представлено порівняння продуктивності трьох серверів Node.js, (рис.2.6.) які приймають рядок і повертають хеш Bcrypt з 12 раундами солі. Це три різні сервери:

- Сервер без багатопоточності
- Сервер з багатопоточністю, але без об’єднання потоків
- Сервер з пулом потоків з 4 потоків

Як видно на перший погляд, використання пулу потоків має значно меншу вартість у міру збільшення робочого навантаження.

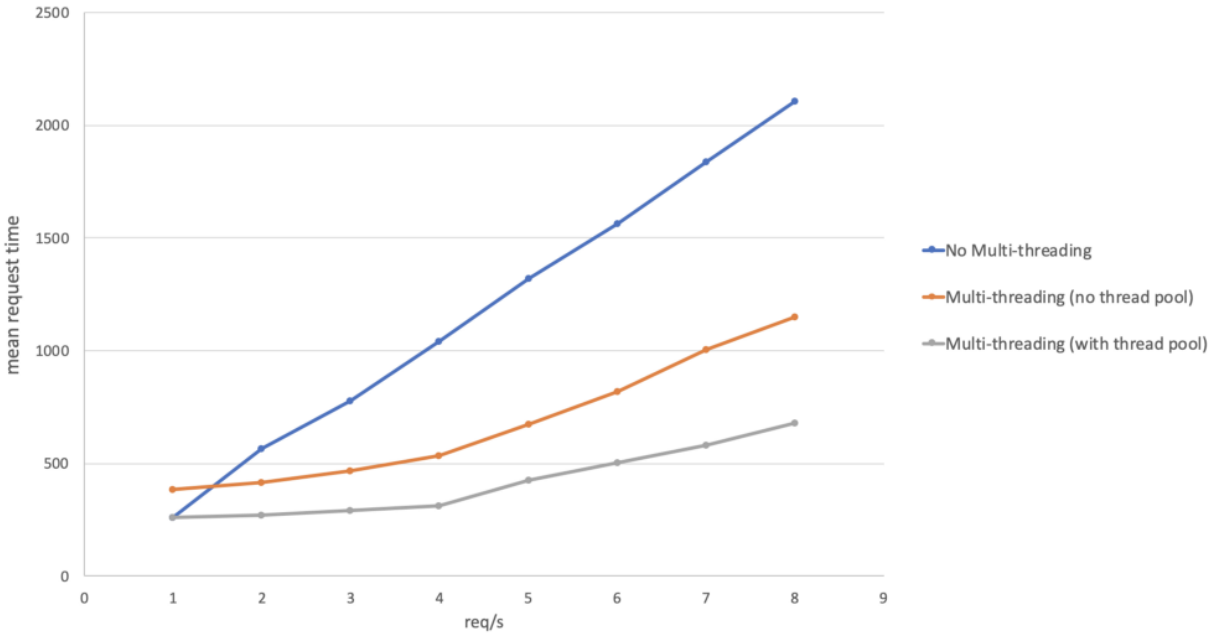

Рис.2.6. Порівняння серверів

2.5. React

React (також відомий як React.js або ReactJS) — це бібліотека JavaScript для побудови користувальницьких інтерфейсів. Він використовується з іншими бібліотеками для візуалізації у певні середовища. Замість того, щоб безпосередньо маніпулювати DOM браузера, React створює віртуальний DOM в пам'яті, де виконує всі необхідні маніпуляції, перш ніж вносити зміни в DOM браузера(рис.2.7).

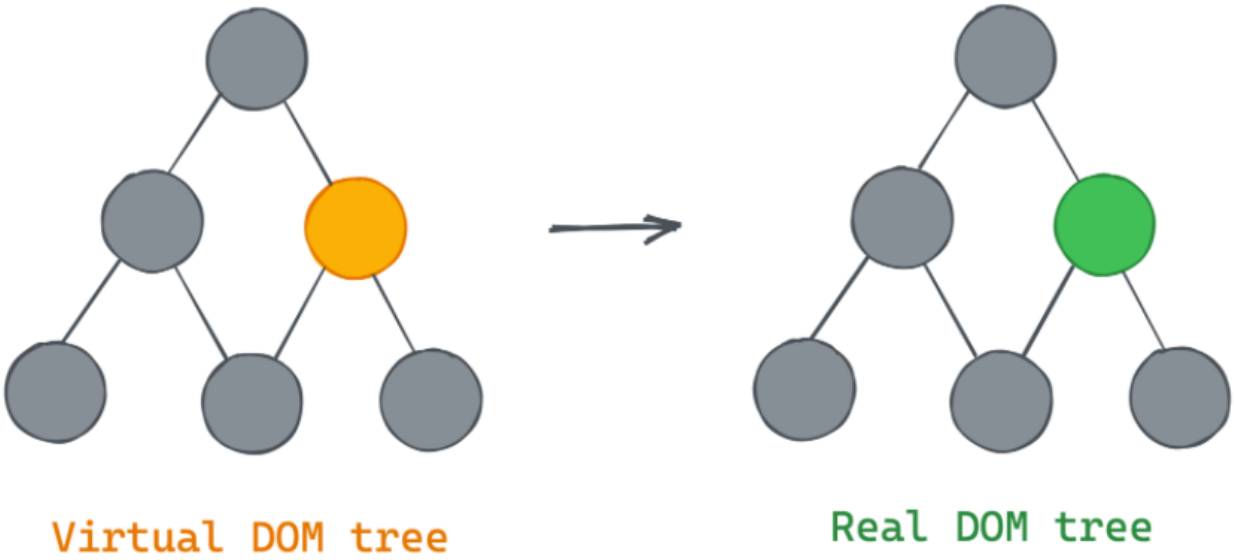

Рис.2.7 Створення віртуального DOM

Віртуальний DOM (Document Object Model) — найважливіша частина веб-додатків — вона представляє структуру веб-сторінки документа, переважно HTML.

DOM представляє документ з логічним деревом. Кожна гілка дерева закінчується вузлом, і кожен вузол містить об’єкти. Методи DOM надають програмний доступ до дерева. — MDN

Маніпулювання DOM широко використовується сьогодні, оскільки сучасні програми вимагають багато змін стану, анімації, ефектів тощо. Уявіть, що у вас є програма і щоразу, коли відбувається зміна стану, ви хочете оновити лише ті частини вашого дерева DOM, які залежать від цих змін. Ви не хочете повторно відображати весь свій інтерфейс користувача з нуля, тому що це буде дорого коштувати з точки зору продуктивності, а UX буде поганим.

React працює з функцією під назвою віртуальний DOM, віртуальним представленням реального дерева DOM. Це просто деревовидна структура даних простих об’єктів JavaScript, яка синхронізується в пам’яті. Відтворення віртуальної DOM відбувається швидше, тому що вона ніколи не буде представлена користувачеві, вона буде жити лише в пам’яті.

Віртуальне дерево DOM імітує ту саму структуру дерева, що й справжнє дерево DOM, навіть виділяє той самий вузол.

Коли ваш додаток React завантажується, React створює копію вашого справжнього дерева DOM. Щоразу, коли в будь-якій частині вашої програми відбувається зміна стану, замість того, щоб повторно відображати все справжнє дерево DOM, React спочатку оновлює свою віртуальну DOM оновленим станом. Далі React порівнює свій віртуальний DOM з вашим реальним деревом DOM, щоб дізнатися, що потрібно змінити. Потім він змінить ваше справжнє дерево DOM, але змінить лише ті елементи, які потрібно змінити, нічого більше. Віртуальний DOM є однією з функцій, які роблять фреймворк React настільки швидким і надійним

Щоб використовувати React у виробництві, вам потрібні NPM та Node.js. React стосується лише надання даних у DOM, тому створення React додатків зазвичай вимагає використання додаткових бібліотек.Код реагування складається з об'єктів, званих компонентами. Компоненти можуть бути передані певному елементу в DOM за допомогою бібліотеки React DOM. Під час візуалізації компонента можна передати значення, відомі як "реквізит":

Два основних способи декларування компонентів у React - це через функціональні компоненти та компоненти на основі класу. Компоненти на основі класу оголошуються за допомогою класів ES6. Вони також відомі як "стаціонарні" компоненти, оскільки їх стан може містити значення у всьому компоненті і може бути переданий дочірнім компонентам через реквізити: Віртуальна DOM

Ще одна примітна особливість - використання віртуальної моделі об'єкта документа або віртуальної DOM. React створює кеш даних структури пам’яті, обчислює отримані відмінності, а потім ефективно оновлює відображений DOM браузера. Цей процес називається примиренням. Це дозволяє програмісту писати код так, ніби вся сторінка відображається при кожній зміні, в той час як бібліотеки React надають лише підкомпоненти, які фактично змінюються. Це вибіркове візуалізація забезпечує значне підвищення продуктивності. Це економить зусилля для перерахунку стилю CSS, компонування сторінки та візуалізації для всієї сторінки.

Основна мета React - мінімізувати помилки, які виникають, коли розробники створюють інтерфейси користувача. Це відбувається завдяки використанню компонентів - автономних логічних фрагментів коду, які описують частину інтерфейсу користувача. На відміну від інших фреймворків, охоплених у цьому модулі, React не застосовує суворих правил щодо конвенцій коду чи організації файлів. Це дозволяє командам встановлювати конвенції, які найкраще працюють для них, і приймати React будь-яким способом, який вони хотіли б. React може обробляти одну кнопку, кілька фрагментів інтерфейсу або весь користувальницький інтерфейс програми.

2.5.1. Порівняння React з іншими фреймворками

Наведено статистику завантажень фреймворків за 2 роки згідно з даними з NPMtrendsЗа наведеними даними наведеними на діаграмі (рис.2.8) і співвідношення скачування на GitHub (рис.2.9) видно, що більшість розробників готові працювати з React і Angular. The State of JavaScript хоч і з застереженням, але вважають, що популярність Angular падає.

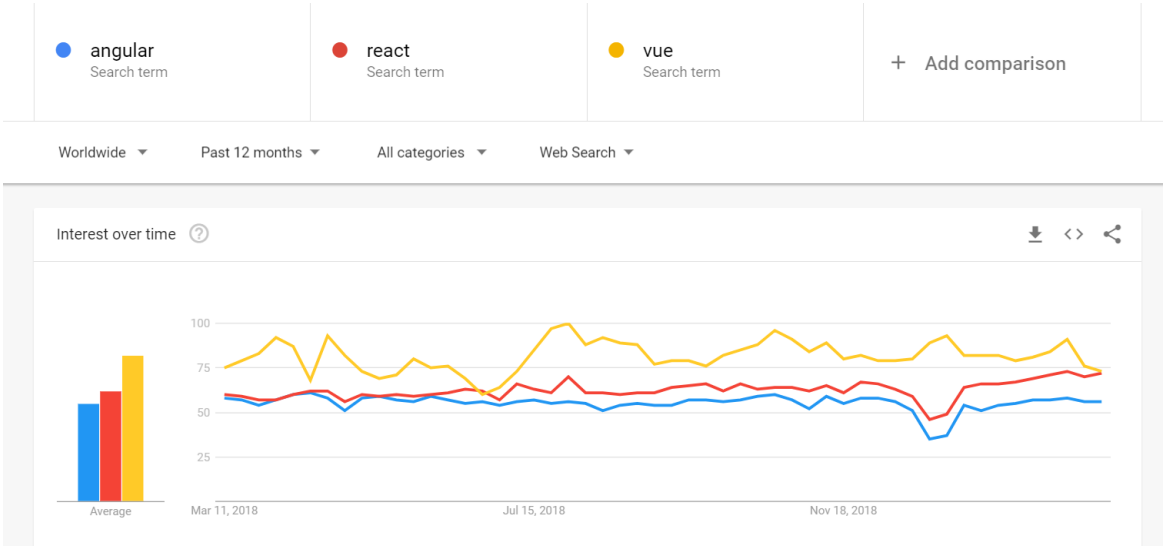

Рис.2.8. Статистика використання

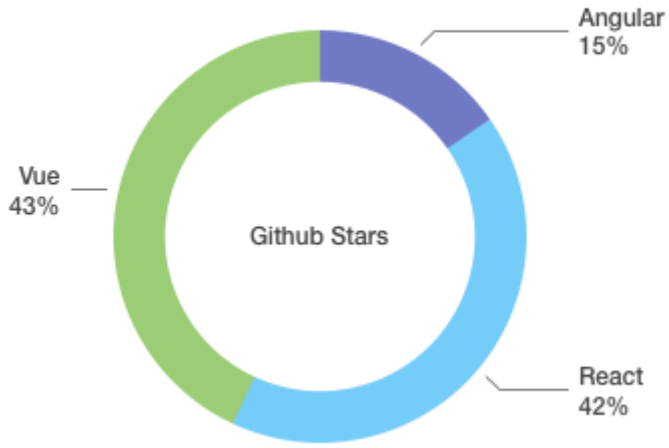

Рис.2.9. Співвідношення скачування на GitHub

React утримує пальму першості серед улюблених бібліотек розробників. Згідно з даними вище, він обійшов Angular за всіма пунктами. При тому, що останній розчаровує користувачів, і на StackOverflow його шанують менше, ніж React. Vue не дуже популярний, хоча спільноті подобається цей фреймворк, тому варто за ним стежити.

Висновки до розділу 2

Сучасна розробка веб-додатків замінила старі застарілі фреймворки та основні компоненти. Важливі параметри залежать від вибору архітектури - швидкості веб-додатка, надійності та безпеки та способу реагування на неї. Майбутнє розширення не тільки збільшить попит, але й неминуче включатиме такі вимоги, як сумісність та потреба у підвищенні надійності. Надійність, швидкість реагування, безпека тощо веб-додатків значною мірою визначаються моделлю та типом архітектури веб-додатків, які ви вибираєте.

Велика кількість веб-сторінок створюється за допомогою скриптів, а сайти без них здаються нудними і нудними. Яким би цікавим і пізнавальним не був вміст сайту, деякі відвідувачі відразу захочуть його закрити від незацікавленості. Звичайно, немає сенсу заперечувати важливість текстового вмісту для будь-якої веб-сторінки. Тому створення веб-додатка за допомогою JavaScript не тільки покращить подання матеріалу, але і зробить сторінку більш запам’ятовується, а головне, зручною для постійного використання. Чим цікавішим і креативнішим буде сайт, чим більше на ньому інформації про фільми, телешоу і навіть акторів, тим популярнішим він стане. Дійсно, за дизайном сайту можна сказати і про сам контент, який він надасть.

Виходячи з усього написаного вище, ви повинні освоїти React в 2020 році. Він використовується в компаніях, він розвивається і залишається в тренді. Також варто звернути увагу на сильне співтовариство, хорошу документацію та багато ресурсів для навчання та роботи. Виходячи з наведеного вище опису, можна зробити висновок, що фреймворк React дуже гнучкий і зручний і має значну кількість можливостей для створення сучасних веб-додатків.

РОЗДІЛ 3.
РОЗРОБКА ТА ЕКСПЛУАТАЦІЯ ПРОДУКТУ

3.1 Створення React компонентів

Щоб додати React-компонент до вже існуючої HTML сторінки нам не має необхідності встановлювати складні інструменти або що-небудь інше. Спчатку ми додаємо DOM-контейнер до HTML структури. Потім Відкриваємо HTML сторінку, яку потрібно відредагувати та додаємо пустий тег, щоб показати де потрібно відобразити компонентиReact.

* *
<divclassName="movieinfo-text">
<h1> {movie.title}</h1>

Ми надали цьому -yidHTML-атрибут, який є унікальним. Це дозволить нам знайти його пізніше за допомогою JavaScript-коду та відобразити React-компонент всередині нього. Далі ми додаємо теги скриптів

Ми беремо три теги<script>до HTML-сторінки перед закриваючим тегом</body>:

<script src="https://unpkg.com/react@16/umd/react.development.js"
src="https://react-dom@16/umd/react-dom.development.js" crossorigin></script>
<! -- React-компонент. -->
<script src=".js"></script>
</body>

Перші два теги завантажують React. Третій завантажує код компонента. Створюється React-компонент Створюємо файл з назвою.js в директорії з HTML-сторінкою. Відкриваємо початковий код та вставте його у файл, який створили раніше. Цей код описує компонент React. Ці два рядки коду шукають , який ми додали до нашого HTML-файлу в першому кроці і потім відображає React-компонент.

3.2 Встановлення програмного забезпечення

Фреймворк React пропонує розробнику повну архітектуру та структуру веб-додатку. У якості серверної технології React пропонує використовувати платформу Node.js засновану на JavaScript.

Встановити це можна використовуючи інсталятор з офіційного сайту https://nodejs.org/. Інші залежності можливо встановлювати в автоматичному режимі за допомогою використання вбудованого менеджера пакетів npm.

3.2.1. Структура каталогів та файли

Файлова структура веб-додатку складається з різних компонентів (рис.3.1)

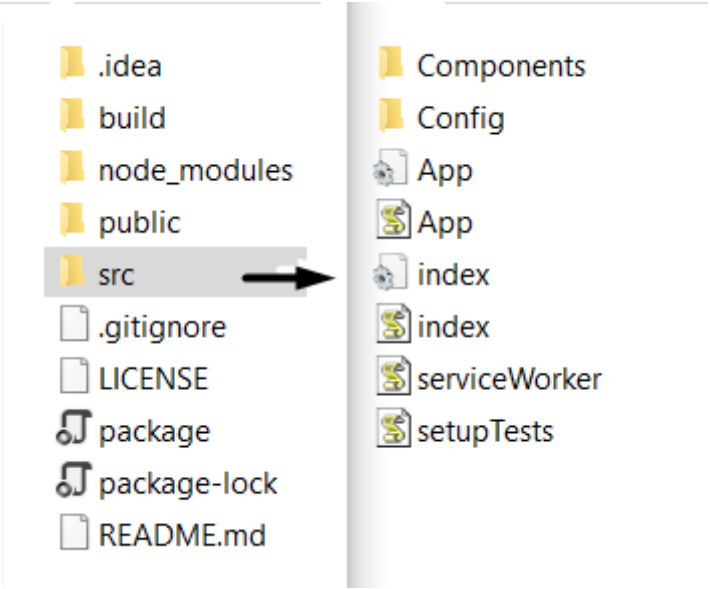

Рис.3.1. Файлова структура прикладу

Файли, що відповідають за роботу серверної частини:

«package.json» — файл пакетного менеджера npm, який містить перелік та версії залежностей які необхідні для роботи прикладу, а також деякі команди для збірки та запуску серверу;

«package-lock.json» — автоматично генерується для будь-яких операцій, де npm модифікує або дерево node_modules, або package.json. Він описує точне дерево, яке було сформовано таким чином, що наступні установки можуть генерувати однакові дерева, незалежно від проміжних оновлень залежності.

«asset-manifest.json» — він не містить інформації про те, які файли є частинами одного пакету;

«manifest.json» — Визначає базові метадані про розширення, такі як ім'я та версія. Також можна визначити деякі аспекти функціоналу (такі, як фонові скрипти, дії браузера).

Файли відповідні за роботу клієнтської частини:

«index.html» - головна сторінка веб-додатку яка містить лише основні теги та скрипти які асинхронно завантажать потрібні ресурси для повного відображення додатку;

«main.ts» - містить код який завантажує інфраструктуру фреймворка React сервіси та запускає додаток з головний компонентом;

«AppComponent». За це відповідає функція «bootstrap»;

«styles.css» - містить загальні спільні стилі необхідні для роботи додатку.

Файл «manifest.json» - це єдиний файл, який обов'язково повинен бути у кожному розширенні.

Використовуючи manifest.json, ви визначаєте базові метадані про розширення, такі як імена та версії. Так само можна визначити деякі аспекти функціоналу.

manifest.json підтримує такий список полів (рис.3.2)

- author
- background
- browser_action
- browser_specific_settings
- chrome_settings_overrides
- chrome_url_overrides
- commands
- content_scripts
- content_security_policy
- default_locale
- description
- developer
- devtools_page
- dictionaries
- externally_connectable
- homepage_url
- icons
- incognito
- name
- offline_enabled
- omnibox
- optional_permissions
- options_page
- options_ui
- page_action
- permissions
- protocol_handlers
- short_name
- sidebar_action
- storage
- theme
- theme_experiment
- user_scripts
- version
- version_name
- web_accessible_resources

Рис.3.2. Поля які підтримує manifest.json

3.2.2. Збірка та запуск

Управління Node.js та npm відбувається через термінал. Отже, перш за все необхідно відкрити командний рядок та перейти до теки з проектом.Для першого запуску, або після внесення змін у файл «package.json», Необхідно запустити процес відновлення залежностей. В ході данного процесу завантажуються необходні версії всіх потрібних залежностей. Для виконання процесу відновлення необхідно спочатку за допомогою команди cd перейти у саме папку. Далі перейшовши в папку ввести команду npm run build. Коли ваш додаток буде готовий для розгортання, запустіть команду npm run build, це створить оптимізовану версію вашого додатку у папці build.

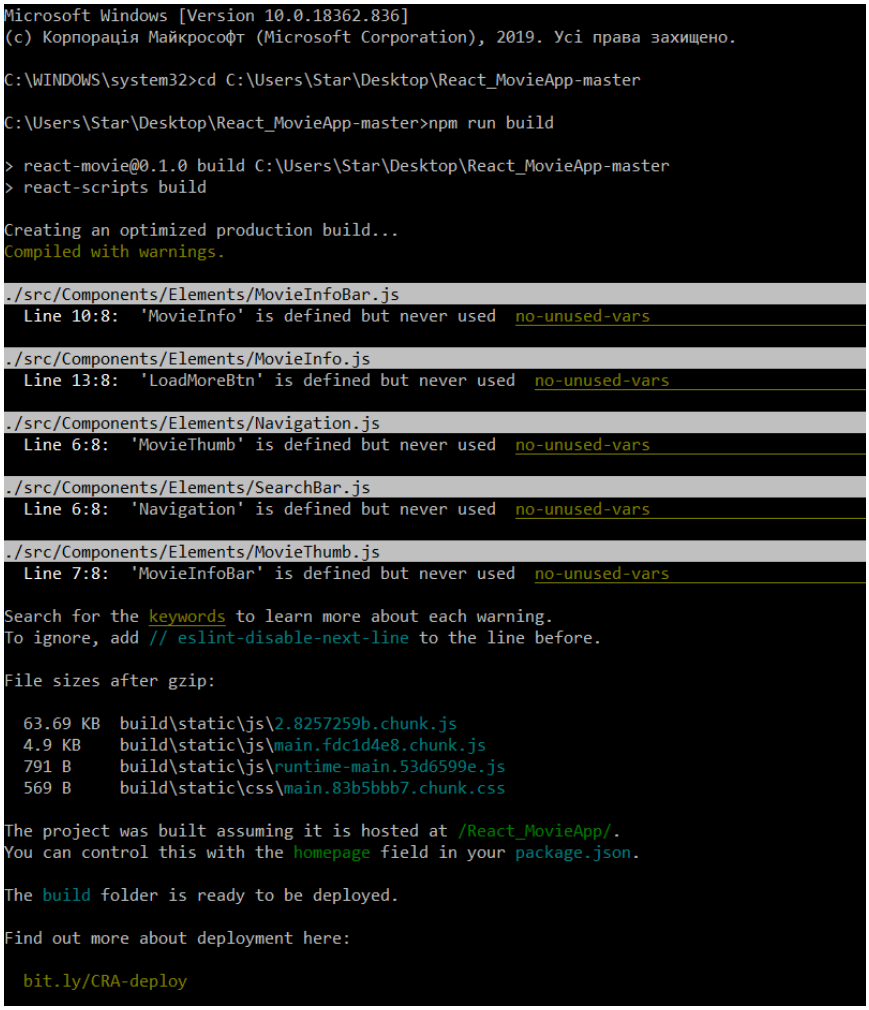

Рис.3.3. Результат роботи команда «npmrunbuild»

Щоб запустити сервер, в консолі слід ввести команду «npm start» (рис.3.4).

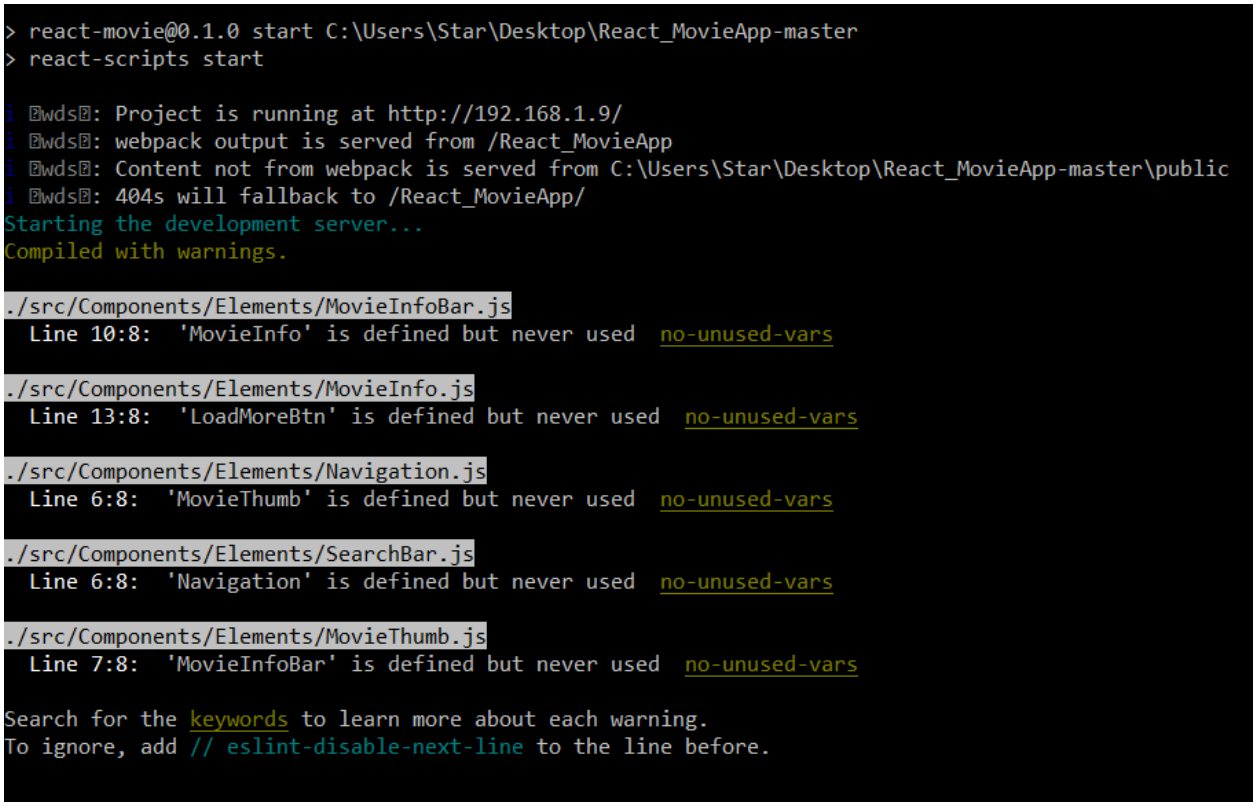

Рис. 3.4. Вивід консолі при успішному запуску серверу

Після успішного запуску сервера веб-сайт буде доступним за адресами які виведені у консолі. Веб додаток містить такі об’єкти як: блоки текстів, посилання, графічні зображення, галерея, адаптивне навігаційне меню на основі списків та інші об’єкти. Оформлення зовнішнього вигляду (дизайн) Веб-сайту виконано за допомогою технології CSS3.

Структура веб-сайту складається з шапки сайту, головної центральної частини, нижньої частини - підвалу та debug-зони. Шапка сайту та нижня частина на кожній сторінці сайту є однаковими, а от центральна частина різна. Для написання коду було використано стандарт HTML5.

<!doctypehtml>

<html>

<head>

<metacharset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

Оскільки веб — сайт є CMS системою, то основний його функціонал побудований на формах. За допомогою них передаються дані в контролери, де вони валідуються та переходять до бази даних. Елементи побудови форм, такі як кнопки, текстові поля, меню, яке прокручується та інші являються інтерфейсом для отримання інформації від користувача.

Форми – це спеціально обмежені області сторінки сайту, в які користувачеві сайту пропонується ввести будь-яку інформацію або вибрати будь-яку дію із запропонованих. Ці елементи керування фактично не обробляють дані. Дані форми обробляються скриптами (скриптами, серверними додатками), які взаємодіють з формами.

До основних елементів створення форм належать:

form – основний елемент форми, які визначає її межі;

input – створює різноманітні елементи управління;

button – створює звичайну кнопку вводу;

textarea – створює багато строкове поле для вводу тексту;

select – створює меню, що розкривається (прокручуваний список);

option – оформлює пункт в елементі управління select.

optgroup – визначає групу пунктів

label – Напис. Зв’язує інформацію з елементами управління;

fieldset – групує між собою зв’язані елементи управління та надписи;

legend – створює заголовок для групи полів (fieldset).

Елемент <form>…</form>

В форму може бути доданий будь-який текст та елементи управління (флажки, перемикачі, поля вводу, створення меню, перегляд тексту та ін.), таблиці, графічні зображення та мультимедія. Не допускається тільки створення вкладених форм.

*<form action = "URL_сценарія" /*Визначає URL-сценарій для обробки форми*/*

*method = "get | post" /*Визначає метод відправки форми на сервер*/*

*enctype = "application/x-www-form-urlencoded".> /*Спосіб кодування даних*/*

<form action="<?= ADMINPATH ?>/teachers/edit" method="post">

Крім того, щоб дані відправлялись із елементів форми, у елемента має бути обов’язково задане ім’я за допомогою атрибута name.

Тег <input> використовується для створення текстових полів, різних кнопок, перемикачів флажків. Хоча елемент <input> не обов’язково розміщувати всередині контейнера <form>, який визначає форму, але якщо введені користувачем дані повинні бути відправлені на сервер, де їх оброблює серверна програма, то вказувати <form> обов’язково.

Зазвичай елемент textarea використовується для заповнення поля текстовою інформацією. Однак, завдяки тому, що в ньому можна зберігати html код, то його перероблено під візуальний редактор контенту, який можна використовувати для заповнення будь якої текстової інформації, списків, зображень чи посилань. Завдаки скрипту який підключає візуальний редактор адміністратору сайту зручніше заповнювати дані та він може робити це, навіть не знаючи html розмітки.

Оператор — це інструкція даної мови програмування, якою задається певний крок процесу обробки інформації.Оператори служать для керування потоком команд в JavaScript. Один об'єкт може бути розбитий на кілька рядків, або, навпаки в одному рядку може буть декілька операторів.

По-перше, блоки операторів, такі як визначення функцій, повинні бути укладені у фігурні дужки. По-друге, крапка з комою служить роздільником окремих операторів. Оператор this використовується для створення посилань на властивості об’єктів, створення посилань на властивості в функціях і на поля форми (рис.3.5).

Рис.3.5. Перевірка елементів на відкриття

Оператор Function було використано для створення власних функцій і об'єктів. Функції бувають іменовані та анонімні. За допомогою React можна мінімізувати створення іменних функцій (рис.3.6).

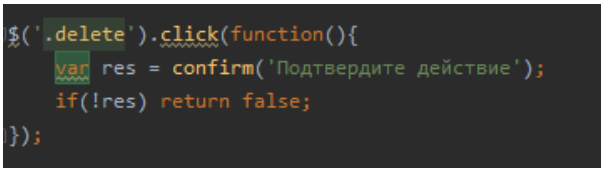

Рис. 3.6. Анонімна функція

Джерела CSS-стилів

Стильові правила можуть застосовуватися до документів трьома способами:

- У вигляді включених стильових інструкцій до окремого елементу;

- У вигляді елементів <style>, розміщених у верхній частині (секції заголовка) документа;

- У вигляді зовнішніх файлів, на які може посилатися документ чи які можуть бути імпортовані в документ (зовнішні CSS).

Включені CSS - це стильові інструкції, які додаються у всередину будь-якого елемента розмітки за допомогою атрибуту style.

Впроваджені CSS - це стильові інструкції, які додаються в секцію заголовка документа за допомогою елемента <style> (рис.3.7)

```
<style>
  *,
  html {
    margin: 0;
    box-sizing: border-box;

    /* font-size: 1em !important; */
    /* color: #000 !important; */
    font-family: Arial !important;
  }
</style>
```

Рис. 3.7. Стилі через тег

Зовнішні CSS забезпечують збір всіх правил в окремий текстовий файл (з розширенням .css) і створення посилання на цей файл в секції заголовка документа.

3.3. Характеристика використаних технічних і програмних засобів

Було проведено порівняльну характеристику таких текстових редакторів, як: Notepad++, Sublime Text, Visual Studio Code. Проаналізувавши їх, було обрано для написання HTML-коду та CSS-коду Visual Studio Code тому що:

- Мало важить, не навантажує систему;
- Перевіряє правопис;
- Підтримка роботи з великою кількістю файлів в одному вікні;
- Підтримка таких операційних систем, як: Windows, Mac OSi Linux;
- Є безкоштовним;
- Підтримка роботи з терміналом;
- Веб-сайт підтримує більше 80% сучасних браузерів.

Елементи побудови форм, такі як кнопки, перемикачі, текстові поля, меню, яке прокручується та інші являються інтерфейсом для отримання інформації від користувача.

Тег <button> створює на веб-сторінці кнопки і за своєю дією нагадує результат, який отримується за допомогою тега <button> з атрибутом type="button | reset | submit".

Поле <textarea> представляє собою елемент форми для створення області, в яку можна вводити декілька рядків тексту. На відміну від тега <input> в текстовому полі допустимо робити перенесення рядків, які зберігаються при відправленні даних на сервер.

soft – це довгий текст, який самостійно не поміщується в поле по ширині і буде автоматично перенесений на новий рядок, але на сервер буде передаватись як один рядок.

hard - слова в полі переносяться механічно, щоб вони помістились в розмір області, і при відправленні на сервер місця автоматичного переносу зберігаються.

off - перенесення рядків відключені. При введені довгого тексту без переносів, він буде друкуватися в один рядок, при цьому буде відображатися полоса прокручення. (значення за замовчуванням).

Тег <select> дозволяє створити елемент інтерфейсу у вигляді списку, який розкривається. Кожний пункт списку описується дескриптором <option>. Значенням пункту списку являється текст, розміщений справа від <option>. Ширина списку, що розкривається визначається довжиною найдовшого тексту елемента <option>.

Синтаксис

<select name="ім'я_поля" [size="число"] [multiple="multiple"]>
<option [selected="selected"] [value="значення1"]>текстелемента 1
<option [selected="selected"] [value="значення2"]>текстелемента 2
<option [selected="selected"] [value="значениеN"]>текстэлемента N
</select>

3.4. Файлова структура папок web-сайту

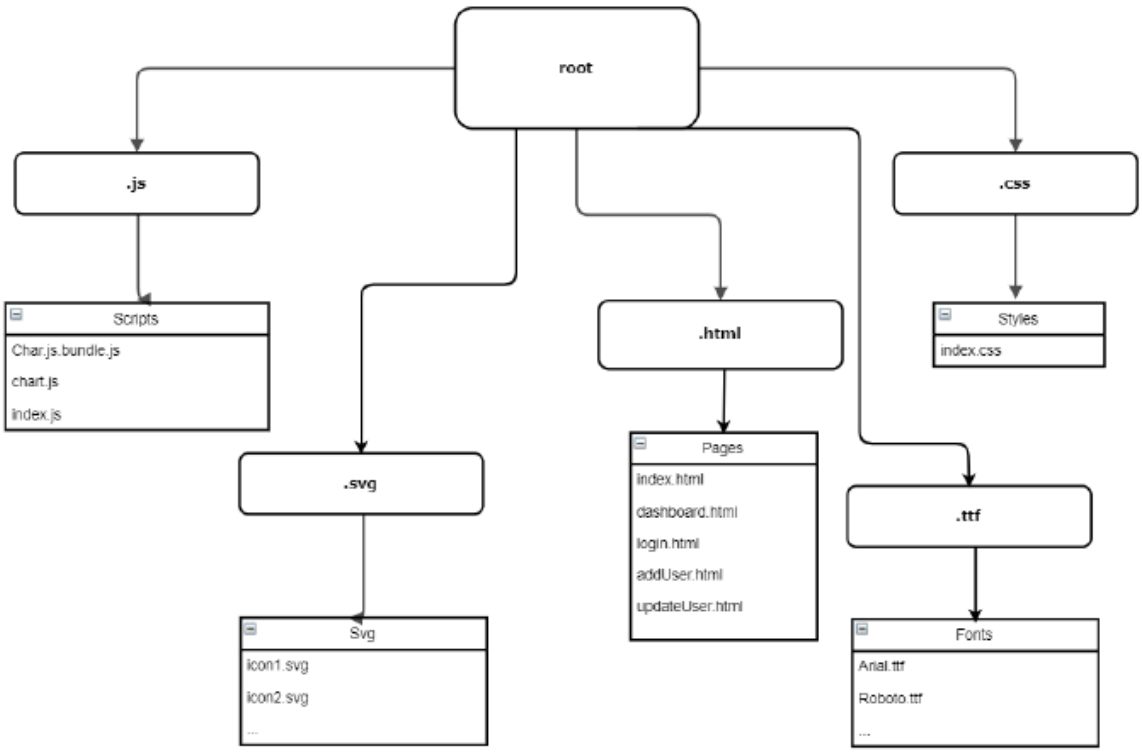

Рис.3.8. Файлова структура папок

Існують такі папки, як: Pages (html-сторінки які використовуються), Fonts (шрифти), Styles (css-стилі для оформлення web-сторінки), Scripts (рис.3.8) (сценарії JavaScript), Images (SVG зображення, які використовуються на сайті).За допомогою допоміжних скриптів (Chart.bundle.js), імпортованими у нашу сторінкуми маємо можливість динамічно стилізувати та оформляти наші графіки. SVG-зображеннябуло обрано, так як векторна графіка, на відміну від растрової має властивість бути гнучкоюі гарно відображатися на різних екранах. Додаткові шрифти додають унікальності нашому сайту та естетичну красу. CSS-стулі включають у себе media-querie, які роблять сайтадаптивним, за допомогою них сайтом можна користуватися під різними екранами.

3.4.1. Характеристика основних об’єктів сторінок веб-застосунку

У Web-застосунку реалізовані такі об’єкти, як: блоки тексту, посилання, зображення, списки, таблиці, аудіо, відео, навігаційна карта, слайдер, вбудований pdf-файл, google-карта,кнопки та інше.Для навігації та загальної інформації користувача використовується тег header (рис.3.9).

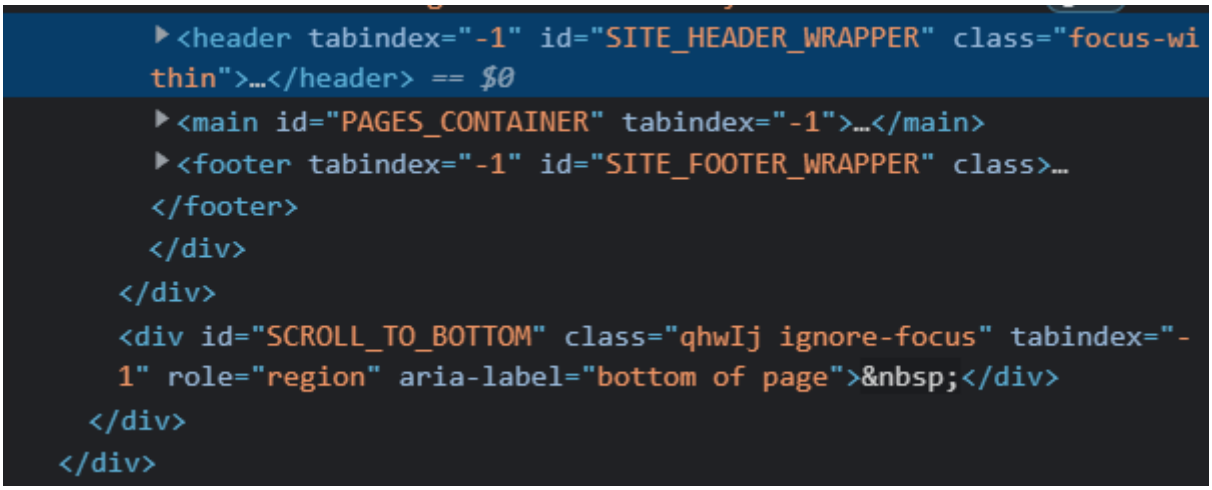

Рис.3.9.Тег header

Для вставки тексту, в переважній більшості використовувався контейнер , тег <p> (рис.3.10) та заголовки <h1>…<h6>.

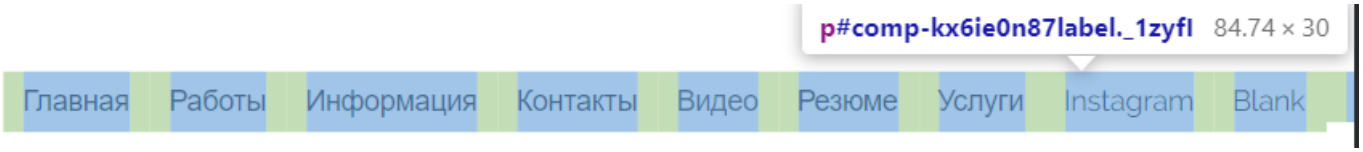

Рис.3.10. Тег <p>

При використанні кожного елемента важливо знати, які елементи можуть розташовуватися всередині нього, і всередині яких елементів може бути він сам. Так, взаємне розташування елементів HTML, HEAD, TITLE, BODY має бути стандартним на будь-якій сторінці, у тих випадках, коли не використовуються

кадри. Якщо сторінка є документом планування кадрів, то замість елемента BODY використовується елемент FRAMESET.

Існують групи елементів, що використовуються спільно. До них відносяться елементи для створення таблиць, списків, кадрів. У цьому випадку порядок вкладення елементів визначається логікою створення об'єкта на сторінці. Таблиці та кадри часто використовуються для того, щоб розмістити деталі сторінки (малюнки, текст та ін.) у певному порядку. Наприклад, маючи малюнок усередині комірки таблиці, можна домогтися певного його положення.

Багато елементів, які використовуються для форматування тексту, допускають найрізноманітніші варіанти вкладення. І самі вони обов'язково повинні розташовуватися всередині певних елементів.
Наприклад, є два абзаци (перший у зеленій рамці) та таблиця:

<P style = "border: 3px solid green"> Текст абзацу 1</p>
<TABLE>...</table>
<P>Текст абзацу 2</p>

Таблиця у разі – незалежний елемент. Її можна, наприклад, вирівнювати незалежно від решти тексту. Можна використовувати інший код:

<Pstyle = "border: 3pxsolidgreen ">Текстабзацу 1
<TABLE>...</table>
<P>Текстабзацу 2</p>

Зник кінцевий тег першого абзацу. Тепер таблиця є частиною першого абзацу, і зелена рамка охоплюватиме таблицю та текст. І навпаки, елемент P може перебувати всередині таблиці: наприклад, один елемент клітини TD може містити кілька абзаців P. Правила синтаксису поширюються і використання стартового і кінцевого тегів, атрибутів і вмісту елемента. Не можна плутати поняття «елемент» та «тег». Елемент – це контейнер, що містить атрибути всередині стартового тега та

корисну інформацію між стартовим та кінцевим тегами. Тег – це конструкція, укладена в кутові дужки та використовується для позначення області впливу елемента.

Деякі елементи не мають кінцевого тегу. Елемент BR, що позначає кінець рядка, не потрібен кінцевий тег. Деякі елементи можуть бути використані з кінцевим тегом і без нього. Елемент абзацу P може мати кінцевий тег , але якщо тег не заданий, ознакою закінчення дії елемента служить наступний елемент, який може логічно визначити кінець поточного абзацу: інший елемент P, елемент малюнка IMG, елемент списку UL, елемент таблиці TABLE та ін. Важливим правилом, яке немає винятків, є розміщення атрибутів елемента всередині початкового тега.

Для створення посилань було використано тег <a> (рис.3.11) з атрибутом href та його значенням у вигляді адреси, куди необхідно перейти за посиланням. Наприклад, для створення кнопкиз посиланням було реалізовано наступний код:

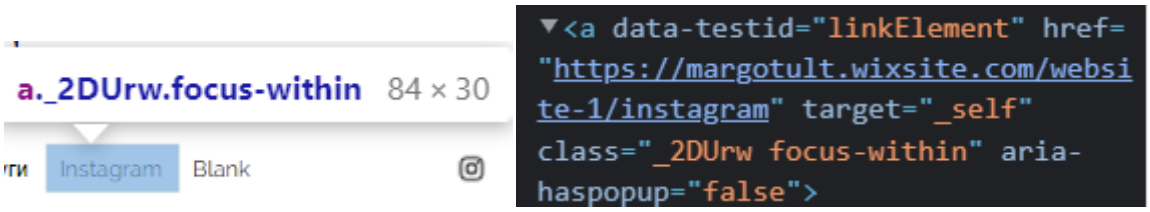

Рис. 3.11. Тег <a>

HTML-документ може бути дуже великим, і в цьому випадку необхідно мати можливість швидкого переміщення до потрібного розділу документа. Для цього можна використовувати механізм гіперпосилань. Необхідно також у потрібних місцях тексту розставити відповідні позначки. Шаблон для створення міток такий:
<A name = “>Довільний текст

В даному випадку даному рядку документа присвоюється ім'я, і, отже, в іншій частині документа або навіть на іншому документі може бути створене гіперпосилання, що веде до цієї точки. Для переходу всередині документа можна використовувати таку конструкцію:

<P>Перехід до <A href = “#мітка”>мітки</p>

Декілька подібних рядків можуть утворити своєрідне зміст Web-сторінки, яке можна розмістити на початку і в кінці документа.

Гіпертекстові посилання серед інших елементів тексту виділяються кольором та підкресленням;мишачий курсор на засланні змінює свою форму і перетворюється на вказівний перст;для переходу за посиланням необхідно клацнути по ньому мишкою;для повернення із посилання необхідно використовувати навігаційну кнопку браузера "Назад" ("Back").

Якщо посилання утворюють вкладений ланцюжок, то кнопки "Назад" ("Back") і "Вперед"("Forward") можна використовувати для руху пройденим посилальним шляхом в обидві сторони. Вони працюють як традиційні операції "відкатка" та "накатка" в більшості прикладних програм.

У наведеному нижче прикладі використовується ланцюжок вкладених посилань. Здійсніть невелику подорож. Спочатку, клацаючи посилання, дістаньтеся до тексту, в якому посилань вже немає. Потім, користуючись навігаційними кнопками браузера, "прогуляйтеся" пройденим шляхом вперед і назад.

Для вставки зображень було використано тег <img> з атрибутами src, alt, title, width. Атрибус src є головним, так як саме він вказує місце розташування зображення. Атрибут alt є бажаним атрибутом, адже, у випадку, якщо по будь-яким причинам зображення не відобразиться, то має бути підпис на цьому місці, який описує, що повинно бути зображеним.

Атрибут title теж є бажаним, щоб користувач при наведенні на зображення міг дізнатись, щоце. При написанні ширини зображення висота буде підлаштовуватись автоматично, щобзберегти пропорції зображення. Приклад коду зображен (Рис.3.12)

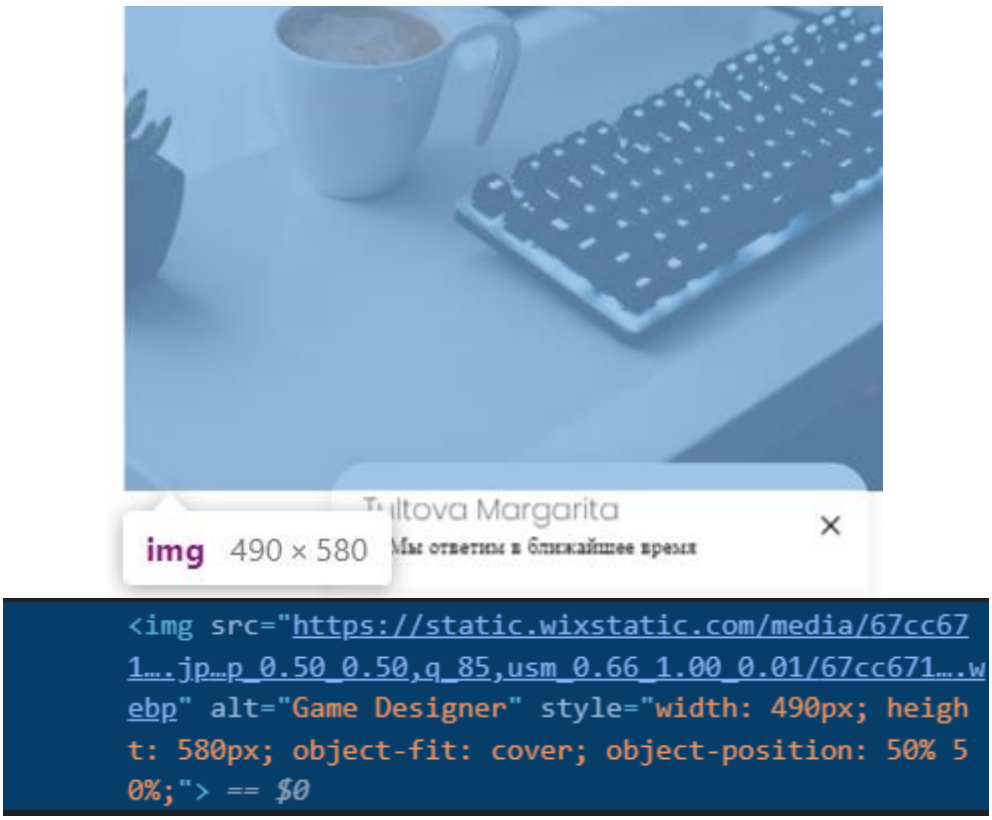

Рис. 3.12. Використання img

Картинка у тексті. Зображення можна помістити в документ майже так, як текстовий символ. Щоб вставити цей незвичайний знак, потрібно скористатися командою <IMG>. Команда не має парного тега, що закриває, але має багато атрибутів. Атрибут src = ім'я файлу

Найголовнішим атрибутом команди є атрибут src, з якого можна задати ім'я файлу з картинкою. Наприклад, <IMG src=img.gif> змусить браузер відобразити на екрані графічний файл img.gif з поточного каталогу.

Зазвичай графічні файли не змішують з HTML-текстами, а поміщають на окремий каталог pic, що є підкаталогом для каталогу з програмами (html-файлами). Тоді команда виведення графіки матиме вигляд: <IMG src=./pic/img.gif>

Атрибут alt = "Текст напису".

Якщо браузер не знаходить картинки у вказаному місці на диску, він замість неї малює на екрані маленький прямокутник і вписує напис, який заданий атрибутом alt:<IMG src=monstr.jpg alt="Страшний звір">

Рекомендується використовувати атрибут alt завжди. Навіть якщо зображення знайдено браузером, завдання напису не буде зайвим: варто користувачеві зупинити курсор на картинці, як напис з'явиться в маленькому вікні і повідомить додаткову інформацію. Перевірте це на наступному малюнку:

Для вставки списків було використано такі теги, як: ol, ul, dl (в залежності від того, якийтип списку був використаний. Тег <li> вказує кожен елемент списку.

Для вставки ненумерованого списку використовувалась така конструкція (Рис.3.13):

```
<ul>
  <li><b>id:</b><span>ab985603-65eb-bd75-27e6-dc761c1b9101</span></li>
  <li><b>created:</b><span>02.14.2020 15:53</span></li>
  <li><b>msisdn:</b><span>9998</span></li>
</ul>
```

Рис.3.13 Конструкція тегу

3.5. Джерела CSS-стилів

Стильові правила можуть застосовуватися до документів трьома способами: У вигляді включених стильових інструкцій до окремого елементу. У вигляді елементів <style>, розміщених у верхній частині (секції заголовка) документа (впроваджені CSS). У вигляді зовнішніх файлів, на які може посилатися документ або які можуть бути імпортовані в документ (зовнішні CSS).

Включені CSS - це стильові інструкції, які додаються у всередину будь-якого елементарозмітки за допомогою атрибуту style (рис. 3.14).

```
div class="cb_stats_info_labels">
  <h3 style="text-align: center;">UNDELIVERED</h3>
  <span>Value</span>
</div>
<div class="cb_stats_info_svg">
  <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24" fill="blac
</div>
/div>
```

Рис.3.14 Атрибут style

Зовнішні CSS забезпечують збір всіх правил в окремий текстовий файл (з розширенням.css) і створення посилання на цей файл в секції заголовка документа. Існує два способипосилання в документі на зовнішні CSS: за допомогою елемента link в секцію заголовка документа.

Наприклад: <link rel="stylesheet" href="./another.css">. Можна вказувати декілька елементів link.

У даній дипломній роботі використовуються зовнішні CSS-стилі. Це рішення через значну комфортність, швидкість розробки та перевикористанням стилів.

3.6. Створення форми

Форма повинна містити різні поля для ввода даних:однострочні поля для ввода тексту; поле ввода пароля; поле ввода e_mail;флажки і перимикачі; меню; кнопки різного призначення (з графічним зображенням, пересилки даних,ініціалізації полей форми та ін.).

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initialscale=
1.0">
```

<title>Document</title>

</head>

<body>

<form method="POST" enctype="application/x-www-formurlencoded"

action="http://localhost:9595/">

<label for="email">

EMAIL

<input name="email" id="email" type="email">

</label>

*
*

<label for="password">

Обов'язковим є застосування елементи специфікації HTML5.

HTML5 — це остання версія мови розмітки гіпертексту, коду, який описує веб-сторінки. Фактично це три види коду: HTML, який надає структуру; Каскадні таблиці стилів (CSS), які піклуються про презентацію; і JavaScript, завдяки якому все відбувається.

HTML5 був розроблений для надання майже всього, що ви хотіли б робити в Інтернеті, не вимагаючи додаткового програмного забезпечення, такого як плагіни для браузера. Він виконує все: від анімації до програм, музики до фільмів, а також може використовуватися для створення неймовірно складних програм, які запускаються у вашому браузері.

HTML5 не є власністю, тому вам не потрібно платити роялті за його використання. Він також є кросплатформним, а це означає, що йому не важливо, чи використовуєте ви планшет чи смартфон, нетбук, ноутбук чи ультрабук чи SmartTV: якщо ваш браузер підтримує HTML5, він повинен працювати бездоганно. Неминуче, це трохи складніше.

Ми пройшли довгий шлях, оскільки HTML ледве міг обробляти простий макет сторінки. HTML5 можна використовувати для написання веб-програм, які все ще

працюють, коли ви не підключені до мережі; повідомляти веб-сайтам, де ви фізично перебуваєте; обробляти відео високої чіткості; і надати надзвичайну графіку.

HTML5 є стандартом, який розвивається, тому говорити про те, коли він буде закінчений, трохи оманливо. Важливо те, що функції HTML, такі як вищезгадане геолокацію, веб-програми, відео та графіка, можна використовувати зараз, якщо ваш браузер їх підтримує.Усі відомі браузери – InternetExplorer, Edge, Firefox, Chrome, Safari та Opera, MobileSafari та браузер Android – підтримують HTML5, але не всі вони підтримують однакові речі. Firefox, як правило, підтримує найширший вибір функцій HTML5, з Chrome і Safari незабаром, але, як ми вже сказали, HTML5 є стандартом, що розвивається, і останні версії кожного браузера більш ніж охоплюють основи. Якщо ви хочете отримати більш детальну інформацію про підтримку браузера, відмінний Caniuse.com надає детальну розбивку того, що підтримує.

Стандарт HTML5 підтримує відео, але, на жаль, ніхто не може домовитися, який(и) формат(и) підтримувати, а це означає, що різні браузери підтримують різні відеоформати HTML5.

Є три основних: OggTheora, який підтримується всіма панелями браузера InternetExplorer (підтримка Safari вимагає ручного встановлення); H.264, який підтримується всіма, крім Firefox; і VP8/WebM, який підтримується всім (хоча Safari та IE вимагають ручного встановлення).

У певному сенсі це вже є: пристрої iOS не запускають Flash, і багато відеосайтів або перейшли з Flash на відео HTML5, або принаймні пропонують HTML5 як варіант. Однак, оскільки HTML5 не включає технологію керування цифровими правами (DRM) для запобігання копіювання, багато власників вмісту віддають перевагу власним, дружнім DRM форматам, таким як Flash або Silverlight. Наприклад, британський відеосайт LoveFilm скидає Flash, але замість HTML5 він переходить на Silverlight.Хоча Adobe оголосила, що припинить розробку FlashPlayer для мобільних пристроїв, Flash також використовується для значно більше, ніж просто показ відео, тому нікуди не поспішає.

Заповнюючи поля форми даними, зробіть аналіз послідовності символів (за правилом URL-кодування), що передаються з різних полів форми. Послідовність даних, що передаються, можна побачити в полі введення адреси браузера при методі передачі GET після натискання кнопки Submit форми. Пропрацюйте (проаналізуйте) якнайбільше варіантів стану полів при їх заповненні та вихідного коду форми, включаючи застосування різних мов (англійська, українська та ін.).

Кодування даних виконується для того, щоб вони не були спотворені або втрачені у процесі передачі. Кодування даних форми (URL кодування) відбувається перед надсиланням на сервер після натискання кнопки Submit.

Незалежно від методу передачі (get або post) інформація з форми тип кодування “application/x-www-form-urlencoded” передбачає виконання операцій:

1) З кожного елемента форми дані передаються парами
ім'я_поля = значення,
де имя_поля – значення атрибута name елемента форми
значення – значення поля (корисні дані), яке вводить користувач визначено стандартним атрибутом value.

2) об'єднання пар у ланцюжок послідовності, розділених символомамперсанд (&)

имя_поля1=значение1&имя_поля2=значение2&…,

Приклад

```
<form method = "get" action = "http://stat.kar.ne/cgi-bin/guestbook.pl”>
<input name = “first” type=text /><br/>
<input name = “nickname” type=text /><br/>
<input type=submit /><input type=reset />
</form>
<form action = "http://stat.kar.net/cgi-bin/guestbook.pl”>
<input name = “first” type=”text” value=“Josephine”/><br/>
<input name = “mail” type=”text” value=“Josie@ukr.net”/><br/>
<input type=”submit” /><input type=”reset” />
</form>
```

3.7. Структура головної сторінки веб-застосунк

Структура сайту складається з шапки в якій розміщено: Назву сайту, та іконки переходу, при цьому сайт створений, як одно сторінковий, тому при виборі іконки, сайт не вантажить нову сторінку, а просто опускається до низу. Сторінка сайту розпочинається з вибору розділу (Рис.3.15), а потім не потрібно чекати завантаження певного розділу, адже сайт просто опуститься донизу.

Мета меню– забезпечити зручну та зрозумілу навігацію по блоках сайту.

Меню можна закріпити над усіма блоками односторінкового сайту або сховати збоку або зверху.

Структура меню:

- Логотип або назва компанії.
- Основні розділи сайту — опис, відгуки, інформація про компанію, контактні дані, акції тощо.
- Посилання на соцмережі.
- Телефон організації.
- Email.
- Кнопка зворотного зв'язку.

Рис.3.15 Головна сторінка сайту

Блок «Вартості»Мета — познайомитись з клієнтом і дізнтись його цілі. (рис.3.16).

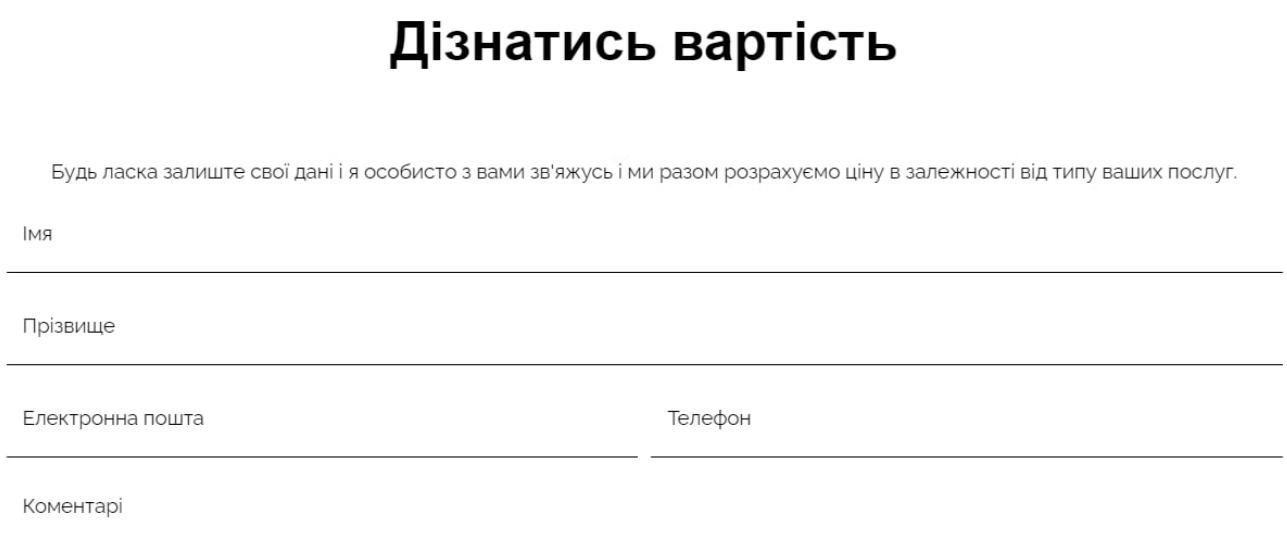

Рис.3.16. Блок «Вартості»

Блок «Відгуки»Мета – ще один показник довіри до компанії. Через відгуки нові користувачі можуть судити про якість товару та обслуговування клієнтів, а також про те, чи сподобався продукт покупцю.

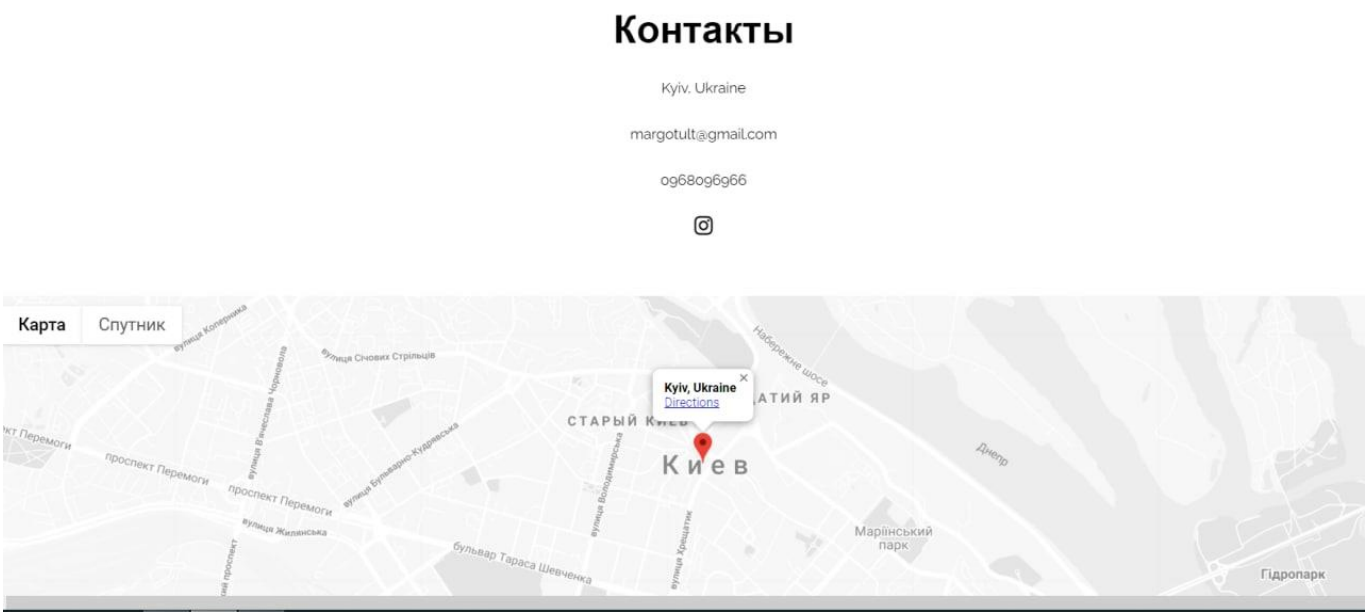

Рис. 3.17. Розділ «Контакти»

На даному сайті є Блок «Контакти» (рис.3.17) з номером телефону, мейлом та точною адресою з посиланням на карти, телефони для зв'язку та посилання на соцмережі. Мета — контактна інформація, яка допомагає користувачеві зв'язатися з компанією та знайти адресу на карті. Також є розділ «Біографія» (рис.3.18), де можна описати короткі відомості про себе.

Рис.3.18. Розділ «Біографія»

Також на сайті доступна кнопка «Напишите нам» рис.3.19, яка є посиланням на спливаюче вікно рис.3.20. Після спливання даного вікна користувач може на пряму зв’язатись з творцем сайту надавши відповідь на кілька пунктів у спливаючумо вікні рис.3.21. Одразу 2 сповіщання отримає власник даного сайту. Перше сповіщання прийде на сайті в розділі сповіщання рис.3.22, а друге аналогічно продублюється на меіл пошту рис.3.23. Форма відкритого спілкування в чаті з клієнтом представлена рис.3.24.

Рис. 3.19. Кнопка зворотнього зв’язку

Рис.3.20. Спливаюче діалогове вікно

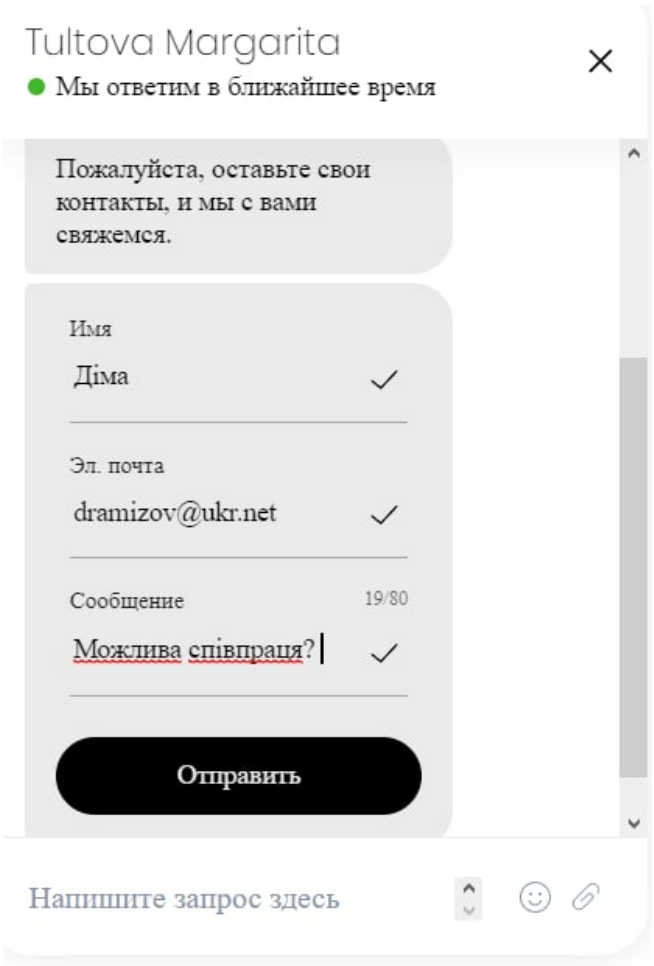

Рис. 3.21. Форма відповіді у спливаючому вікні

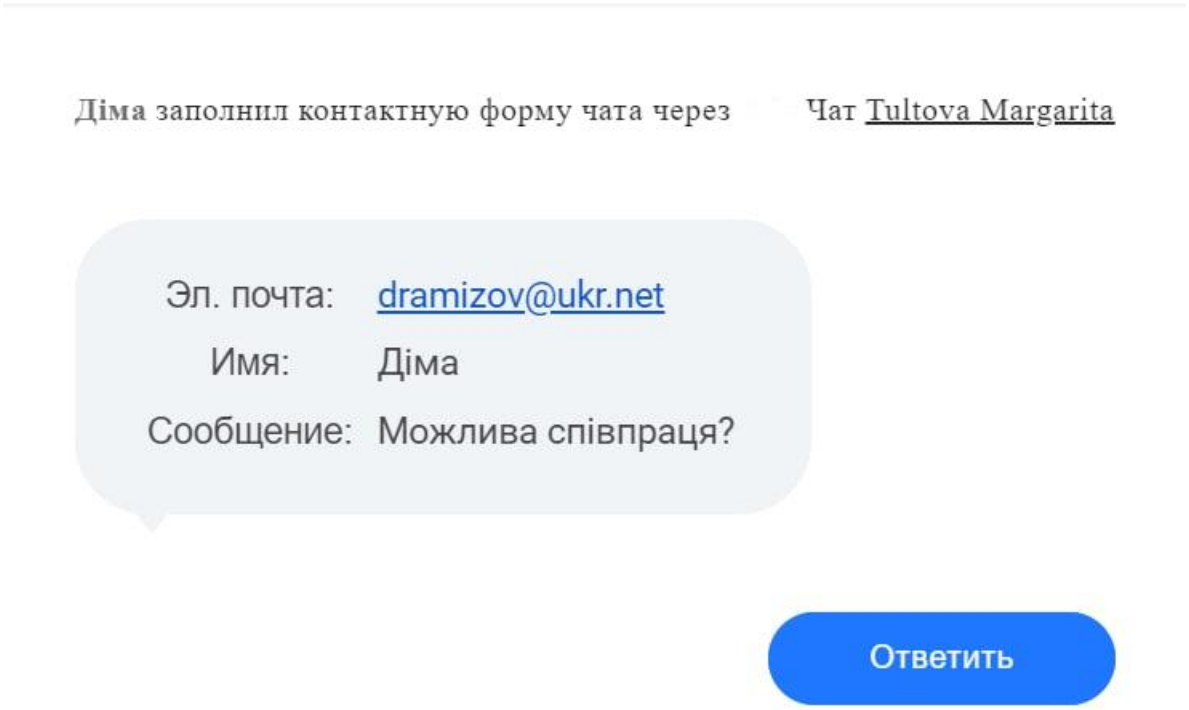

Рис.3.22. Сповіщання на сайті

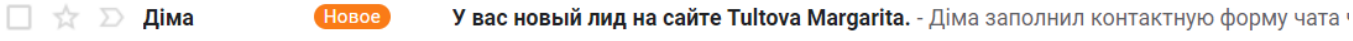

Рис.3.23. Сповіщання на пошті

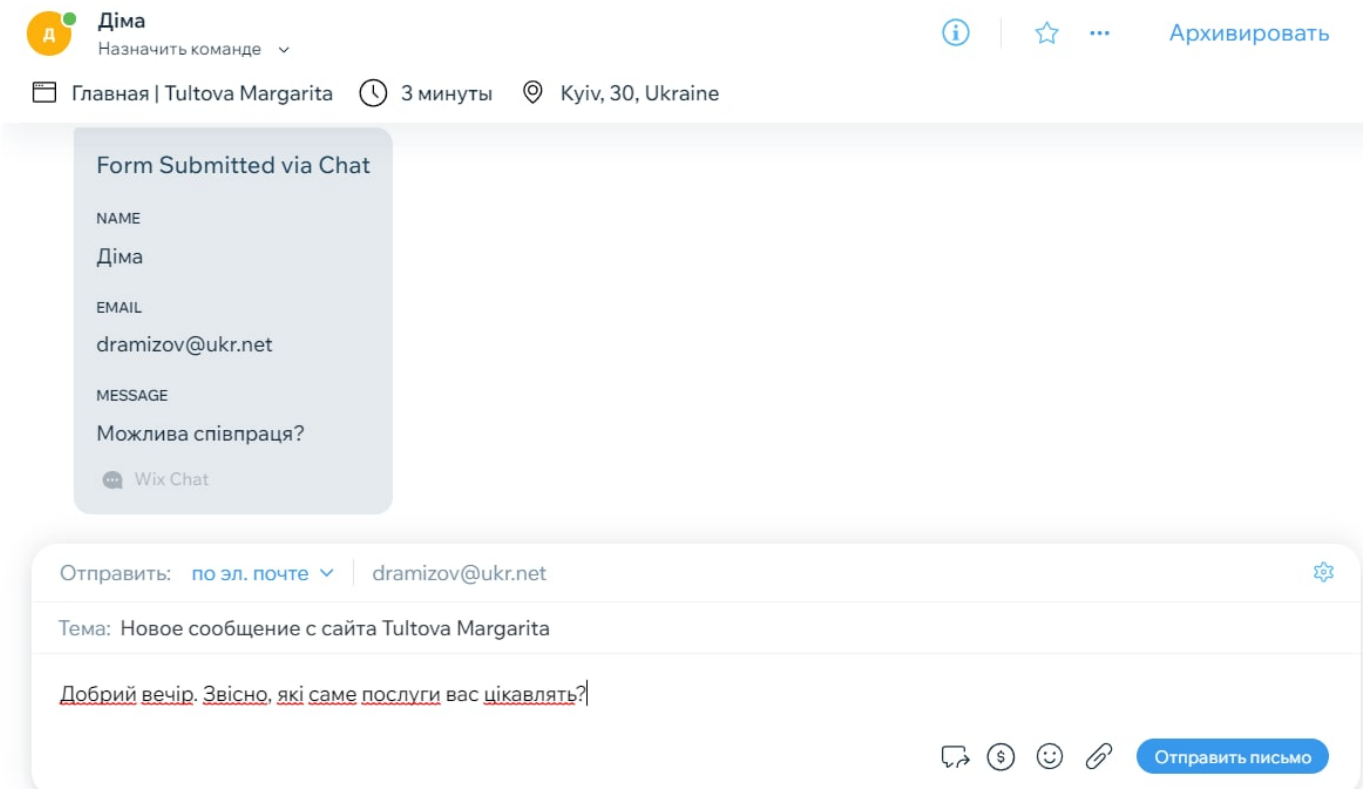

Рис.3.24. Надання відповіді

Аналогічно, власник веб за стосунку може надати відповідь у двох варіантах, на сайті в чаті чи надавши відповідь в електронному листі, про що клієнт отримає сповіщання. Для кожного виду односторінника є додаткові блоки:

Даний сайт є Резюме Портфоліо, яке по своїй суті є розробкою веб за стосунку, яке одразу дає роботодавцям оцінити ваші навички та побачити рівень ваших вмінь. На даному сайті можна розмістити ссилки на всі свої роботи, також можна створити блок з «Відгуками» від клієнтів з якими вже була співпраця.

Далі для його створення потрібно було

Вибирати доменне ім'я.

Оплатити хостинг.

Налаштувати безпечне підключення.

Перевірити швидкість завантаження сторінки.

Підключити послуги для просування — лічильники для відстеження метрик, рекламні кабінети, інтергації з CRM-системою та платіжними системами, а також соцмережі та месенджери. Розібратися у юридичних деталях

3.8. Переваги веб за стосунку над шаблонами

Фішка сайту-односторінника у структурі та форматі використовуваних блоків. Такі сайти не передбачають тривалого розгойдування з плавним навіюванням зацікавленості, проводячи відвідувача через блог, форум, опис виробництва, емоцій автора або чогось ще. Усього 1 сторінка, і вона має бути ефективною – показати лише те, що допоможе швидко продати.

Можна довго перебирати конструктори, які дозволяють створювати односторінкові сайти, проте дані конструктори мають багато мінусів, один з головних яких це щомісячна оплата.

В багатьох шаблонах немає вбудованих A/B тестів, виронок продажу, аналізу трафіку за джерелами – загалом, потрібно користуватися сторонніми сервісами для збору/аналізу статистики по конверсії; Немає вбудованої системи повідомлень про продаж та інші події, що компенсується підключенням CRM.

Реалізація деяких функцій перенесена до додатків для полегшення сторінки версії інтерфейсу системи, з коробки не все знаходиться на поверхні - наприклад, синхронізація з платіжними системами, конструктори форм, відео, фоторедактори та інші модулі вимагатимуть установки з бібліотеки додатків;

Для створення макетів сторінок з нуля необхідна навичка - редактор потужний, але елементарним його не назвеш, доведеться наловчитися з ним працювати.

Неможливо реалізувати мультимовність сторінок;

Немає багатосторінкових шаблонів, хоча такі сайти можна створювати;

Обмежена область додавання кнопок соціальних мереж (футер);

Немає блогового модуля, який знадобився б при оформленні багатосторінкового сайту;

На молодшому тарифі недоступна значна частина можливостей системи. Мінімалізм набору можливостей з коробки, багато що доведеться реалізувати через установку плагінів;

Велика кількість тем і розширень породжує багато неякісних варіантів, необхідно бути уважними під час виборів і тих, та інших;

Двигун серйозно навантажує сервер, бажано використовувати хороший хостинг, який зазвичай коштує трохи дорожче за прохідні варіанти.

Чим відрізняється лендинг від односторінника. Лендінг — це сайт, причому не обов'язково односторінковий, який завжди переслідує цільову дію: продаж, підписка, збір інформації або ще щось. Цілком і повністю комерційний тип сайтів, що переслідує досягнення цілей, які прямо чи опосередковано допоможуть отримати прибуток. Має характерну структуру, що розкриває в динамічному темпі особливості, як правило, одного продукту, послуги або товару, що неодноразово закликає до дії (купити, замовити, підписатися, зв'язатися, отримати, завантажити, залишити заявку тощо). Це гострий інструмент, мета якого – викликати яскравий спалах зацікавленості у потенційного клієнта та спонукати до бажаної дії, тут і зараз, відразу ж, без жодного розгойдування та варіантів на кшталт «ну, я ще подумаю».

Односторінковий сайт не обов'язково є лендингом – це може бути візитівка, портфоліо, лонгрід, квіз чи резюме, зрештою. Так, ці типи сторінок теж можуть мати комерційний характер, але відрізняються структурою, подачею, ступенем напористості в підштовхуванні відвідувача до досягнення вашої мети. Хоча так, більшість лендингів є односторінниками і просувають одну послугу або товар, активно. Звідси і розхоже помилки, що зливають сенс лендингу та односторінника в єдине ціле. Бувають лендинги-виключення – кілька товарів (2-5) і кілька сторінок, кожна з яких дає більш розгорнуту інформацію про продукт, але формат при цьому зберігається: структура, що продає, акцент на відміні до здійснення дії і, як правило, відсутність навігації.

У більшості випадків односторінковий лендинг ефективніший за багатосторінковий. Оптимально зробити окремі сторінки під кожен продукт, але бувають ситуації, коли потрібно кілька сторінок для досягнення цілей. Є плюси: такі сайти простіше просувати за запитами, тому що вони містять більше текстової інформації, можна додати більше пошукових ключових фраз, також є запас інформаційного обсягу для якісного переконання, виклику стійкої зацікавленості у клієнта.

Загалом варіантів маса. Лендінг - не завжди односторінник. І навпаки. Проте ці поняття часто збігаються за змістом. Важливу роль відіграють фактори поведінки користувачів. Ваш лендинг має бути цікавим за змістом, привабливим настільки, щоб люди затримувалися на ньому, а не тікали після кількох секунд. Додайте відео, інтерактивні елементи, слайдери та все з того, що може втягнути клієнта у вивчення сторінки, захопити, дати привід затриматися довше, та й прийняти рішення про покупку на піку зацікавленості.

Велику роль відіграють блоки переваг, відгуки та контакти. Чим переконливіше, детальніше та якісніше все це буде оформлено, тим більше шансів продовжити перебування людини на сторінці та переконати у корисності угоди, викликати довіру до вашої компанії та офферу. Все це працює в сукупності - чим якісніша сторінка, тим краще вона продає і просувається, одне не відокремлене від іншого.

Важливу роль відіграють кнопки соцмереж та зовнішні посилання на вашу сторінку, частину яких можна легко проставити через сайти відгуків, безкоштовних каталогів, вакансій, дошки оголошень та інше. Чи не переборщите - нарощуйте посилальне поступово. Потрібно отримати в такий спосіб 100-150 посилань за 3-4 місяці – робота технічно нескладна, хоч одноманітна. Оптимізуємо лендинг під поведінку трафіку.

Отже, трафік пішов на сторінку, ви мониторите конверсію, порівнюєте результати. Можна спробувати поекспериментувати зі структурою та окремими елементами лендингу, особливо при низькій або нульовій конверсії. Якщо все йде добре, нічого не чіпайте - не варто ламати те, що вже працює.

Оптимізація має на увазі правку дизайну (кольори, шрифти), переміщення блоків місцями, правок текстового контенту (особливо важливу роль відіграють заголовки та тези переваг офферу), можна також змінити колір кнопок CTA, змінити їх розташування, кількість та інше в такому дусі. Не вносите за раз велику кількість змін – спробуйте потроху оптимізувати та порівнюйте результати. Так ви знайдете оптимально працюючу комбінацію.

Вартість створення односторінкового сайту

Собівартість односторінника залежить від того, яким шляхом створення ви підете, варіантів кілька:

Створити у конструкторі;

Створити на CMS, орендувавши хостинг;

Замовити у студії чи фрілансера.

Серед CMS найбільш простою точкою входу в розробку є WordPress, все інше складніше. Двигун безкоштовний, тому собівартість сайту залежить від ціни хостингу та домену до нього. Ціни у хостерів різні, найбільш якісним і офіційно рекомендованим самими ж розробниками двигуна є Bluehost, річна оренда якого обійдеться в $35.4. Домен та SSL йдуть у подарунок. Установка CMS автоматична, в комплекті йде оптимізація під двигун, тому з налаштуванням морочитися не доведеться - рівень комфорту приблизно такий же, що пропонують конструктори з поправкою на особливості освоєння WP.

Замовлення готового односторінника веб-студії залежить від розцінок. Найважливіше у технічній оптимізації лендингу — швидкість завантаження. На це впливає хостинг, швидкість роботи двигуна та вага контенту всередині. Основна вага зосереджена на картинках. Їх потрібно стискати з мінімальною втратою як, наприклад, безкоштовний сервіс Tinypng. Це важливо. Пошуковики та відвідувачі чуйно реагують на швидкість завантаження сторінок, довше 3-4 секунд – і все, ви втрачаєте прихильність і тих, і інших приблизно в 70% випадків.

Основну роль просуванні односторінників грає контекстна реклама. Саме від бюджету, який ви готові вкласти в неї, залежить успіх сторінки за умови нормальної якості оффера та самого лендінгу.

3.9. Тестування

Тестування — це процес аналізупрограмного засобу, спрямований на виявлення відмінностей між його реально існуючими і необхідними властивостями (дефект) і на оцінку властивостей програмного засобу. При реалізації проекту потрібно враховувати, які інструментальні засоби будуть використовуватися для

пошуку і для документування знайдених дефектів. Чим більша увага приділяється тестування програмного продукту, тим вище буде його якість. Перед впровадженням програмного продукту в експлуатацію необхідно провести його тестування. Тестування є одним з етапів життєвого циклу програмного засобу, спрямованим на підвищення якісних характеристик. Обрана була стратегія тестування «білий ящик», тобто заздалегідь були визначені передбачувані помилки і некоректні дані, що перевіряють систему на адекватність поведінки.

Тестування базується на тестових процедурах з конкретними вхідними даними, початковими умовами і очікуваним результатом, розробленими для певної мети, такої, як перевірка окремої програми або верифікація відповідності на певну вимогу. Тестові процедури можуть перевіряти різні аспекти функціонування програми — від правильної роботи окремої функції до виконання бізнес-вимог.

Тестування програми починається з реєстрації нового користувача. При переході по посиланню на додаток, користувач бачить форму входу, яка містить поля для номера телефону і пароля. При натисканні на кнопку «Увійти» відбувається перевірка коректності введених облікових даних і або виробляється вхід в додаток, або виводиться повідомлення «Невірний логін або пароль». При натисканні на кнопку «Реєстрація» користувачеві відкривається сторінка з полем для номера телефону і посилання у вигляді «Я згоден з умовами угоди». Після згоди з умовами, додаток повертає користувача на сторінку для входу, а на введений номер телефону приходить смс з паролем. При реєстрації на сайті необхідно заповнити всі поля, а також необхідно вирішити, якщо поля будуть не заповнені відобразиться помилка.

При оформленні підписки необхідно заповнити всі поля. Якщо поля будуть не заповнені, то стається збій даних і необхідно буде заново заповнити дані. При успішній реєстрації, з'явиться повідомлення відповідно до малюнком, після чого Вам за вказаною адресою (E-mail) прийде пароль від аккаунта.

Мобільний дизайн Його також називають респонсивним – елементи спочатку заточені під смартфони. Контент на сайті є динамічним і при зміні масштабу кожен

блок адаптується під будь-який шаблон - сторінка буде коректно відображатися на всіх видах пристроїв.

Веб-сайт є адаптивним, тому вдало відображається на будь яких екранах, в тому числі на мобільних пристроях (рис.3.25).

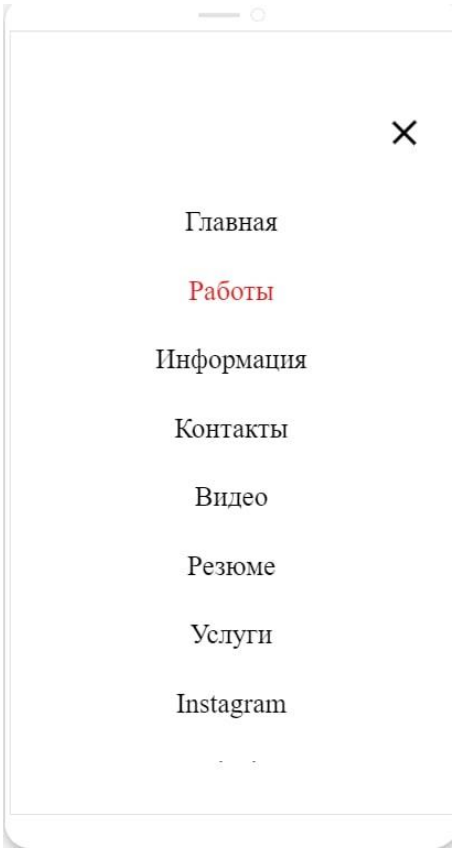

Рис. 3.25. Версія веб-додатку для телефону

Проста навігація. Не ускладнюйте меню сайту найскладнішими назвами. Робіть прості заголовки розділів – до двох слів.

Багато картинок знижує швидкість завантаження сторінки. Щоб уникнути цього, ви можете додати одне велике зображення, яке оптимізується так, щоб збільшити швидкість завантаження односторінника.

У наш час на рахунку кожна секунда - якщо ваш сайт повільно підвантажує інформацію, користувач закриє сторінку і ніколи не повернеться до вас.

Використовуйте лише якісні зображення. Найкраще додавати власний контент, а стокові знімки включати лише на старті проекту.

Мінімалізм у дизайні Він проявляється у всьому: прості шрифти, від трьох до п'яти кольорів та лаконічне оформлення кнопок та блоків.

Сумісність сайту з різними браузерами і різними настройками браузерів вироблялося на останніх версіях браузера, відповідно до таблиці 3.1.

Таблиця 3.1

Сумісність з браузерами

Браузер	Оцінка
Google Chrome	Позитивно. Всі ефекти працюють відмінно.
Mozilla Firefox	Не використовувати
Opera	Позитивно. Всі ефекти працюють відмінно.
Windows Internet Explorer	Не використовувати

Для правильності роботи сайту, необхідно використовувати GoooglеChrome версі 32.19.100.1.

Наповнення бази даних здійснювалося ручним введенням.Тестові набори були засновані на виявленні коректного введення даних. Надійність можна також визначити експериментально шляхом підрахунку кількості зависань чи інших збоїв системи за певний період часу, також експериментально можна визначити середній час відновлення при збоях.

Зручність ручного введення можна оцінити шляхом визначення часу, витраченого користувачами на заповнення основних форм.

Також необхідно протестувати програмну систему на продуктивність, наскільки вона буде відповідати висунутим на початку проекту вимогам (рис 3.26).

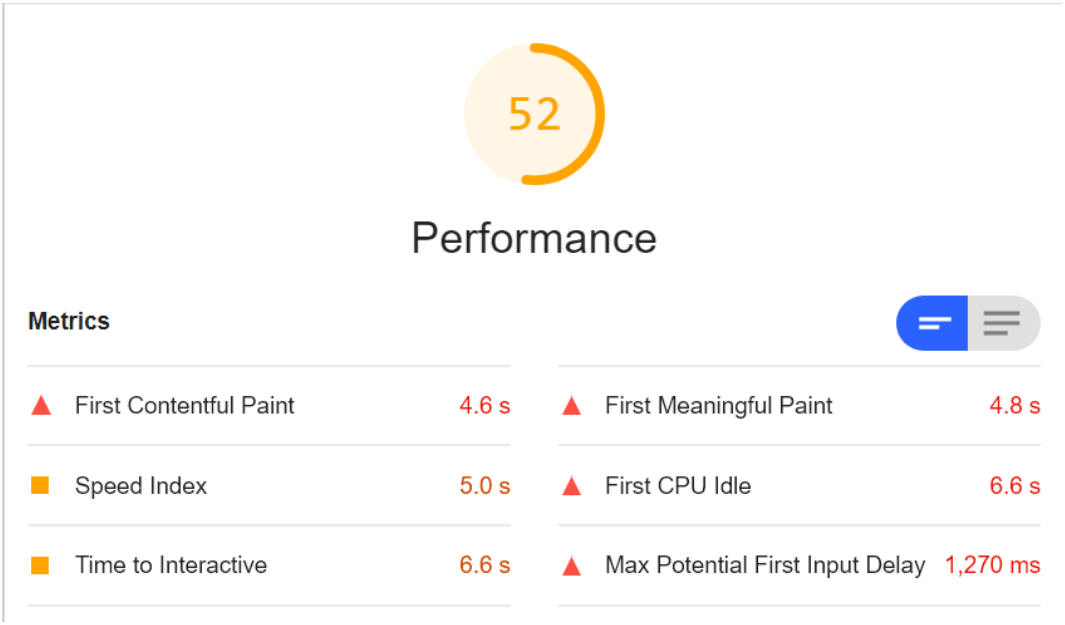

Рис. 3.26. Тестування продуктивності

При тестуванні сайту була перевірена працездатність всіх посилань, розташованих на його сторінках. В результаті тестування битих посилань виявлено не було.При заповненні бази даних час на формування запиту становить менше 1,27 секунди.Експериментально встановлено середню швидкість завантаження сторінки при низкоскоростном каналі становить 2,5 секунди.При тестуванні системи збоїв не виявлено.

Висновки до розділу 3

В цьому розділі, булорозроблено робочий приклад веб-додатку, на якому було розглянуто особливості розробки веб-додатку за допомогою фреймворку React. У розділі були досліджені аспекти розробки з використанням фреймворку React. Також було розглянуто основні можливості фреймворка та способи їх використання. Описані структура та структурні одиниці додатку, а також необхідне для його роботи програмне забезпечення та процес створення запуску додатку.Виходячи з проведеного опису, можна заключити, що фреймворк React є дуже гнучкий йта зручним та має значну кількість можливостей для створення сучасних веб-додатків.

Було закріплено навички використання основних компонентів веб-технології та веб-дизайну (мови HTML, CSS, мови сценаріїв JavaScript), інструментальних засобів створення веб-ресурсів, підготовки базових елементів веб-документів (тексту, форм та іншого) для створення динамічних веб-сайтів.

Односторінкові сайти існують не лише у форматі лендінгів. Проте найчастіше їх використовують саме у ролі посадкових сторінок для вирішення різних завдань: збору підписок, заявок, інформації, продажу товарів, послуг та інших цілей. Найкраще працюють сторінки, підпорядковані одній-єдиній меті – без розпилення на спроби продати кілька товарів, підписати на щось, а потім ще й змусити заповнити анкету. Одна сторінка має вирішувати одне конкретне завдання. Це класичний, добре працюючий підхід.

Просувати односторінники потрібно контекстною рекламою, будьте готові хоч трохи вкластися в неї. Найкраще для вирішення завдання підходять uKit та Wix – потужні конструктори з помірною вартістю, здатні зацікавити і новачків, і досвідчених онлайн-манімейкерів. LPgenerator та Mottor – потужні конструктори з повним профільним набором можливостей із коробки. Tilda цікава дизайнерам насамперед. Маркетологам меншою мірою, оскільки, хоч і позиціонується як конструктор односторінкових сайтів, але все ж таки містить недостатню кількість інструментів для підвищення ефективності лендингів.

В цілому, простіше та дешевше створювати лендинги самостійно. Це нескладний тип веб-сайту. Якщо сумніваєтеся у своїх силах, просто оберіть готовий шаблон і замініть у ньому контент на свій – отримайте сторінку з грамотною структурою без зайвого клопоту. Створюйте сторінку для цільової аудиторії — пам'ятайте про зручність користування та прості слова.Мінімалістичний дизайн краще сприймається користувачами.

ВИСНОВКИ

В процесі підготовки даної дипломної роботи, в першому розділі було детально досліджено принципи роботи портфоліо, одно сторінкових сайтів. Було розглянуто питання актуальності його створення, переваги та недоліки.

В другому розділі було досліджено типові особливості архітектури веб-додатку. Також розглянуто фреймворк React, та причина його використання в дипломній роботі. В результаті розгляду в розділі фреймворків та технологій, можна дійти висновку про зростання інтересу до розробки веб-сайтів даного типу в останні роки. Нові фреймворки з’являються кожні кілька років й значно спрощують процес розробки, зменшуючи об’єми коду, та здійснюючи його структурний розподіл, що полегшує адаптацію відомих рішень з інших платформ.

На основі порівняння декількох популярних фреймворків, React був обраний як найбільш сучасний та перспективний. Було детально досліджений повний процес створення веб-додатку з використанням обраного фреймворку React. Також була розглянута переважна більшість основних можливостей фреймворка та досліджено способи їх використання. Детально описана структура застосунку, процес його створення й запуску. Процес розробки з використанням даного фреймворку був зручним та комфортним для розробки.

В третьому розділі набуті в ході даної роботи знання концепції, архітектури та практичні навички роботи з фреймворком React були використанні в розробці інтерфейсу у вигляді веб-застоснкуу для дипломного проекту.

В результаті виконання роботи було створено веб-додаток який являє собою веб-застосунок «Резюме Портфоліо».

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Веб - застосунок [Електронний ресурс] режим доступу: URL: http://uk.wikipedia.org/wiki/Веб-застосунок. (Дата звернення 26.11.2021 р). – Назва з екрану.
2. Структура веб-приложения [Електронний ресурс] режим доступу:URL http://labaka.ru/likbez/struktura-veb-prilozheniya, свободный. (Дата звернтання 27.11.2021). – Назва з екрану.
3. Розробка-Web-додатки з використанням JavaScript каркаса Node.js [Електронний ресурс] режим доступу:https://ukrbukva.net/page,2,88554-Razrabotka-Web-prilozheniya-s-ispol-zovaniem-JavaScript-karkasa-Node-js.html. (Дата звернення 28.11.2021 р). – Назва з екрану.
4. Фреймворк — что это такое? Определение, значение, перевод [Електронний ресурс] – режим доступу: URL https://chto-eto-takoe.ru/framework(Дата звернення 11.11.2021). – Назва з екрану.
5. Что такое фреймворк [Електронний ресурс] режим доступу: URL http://www.dbhelp.ru/what-is-framework/page(Дата звернення 01.12.2021). – Назва з екрану.
6. Система автоматичного контролю продукції на основі RFID міток з оптимізацією швидкості сканування. [Електронний ресурс] режим доступу: URL https://revolution.allbest.ru/manufacture/00757073_1.html(Дата звернення 01.12.2021). – Назва з екрану.
7. Адаптивне ціле фонове зображення за допомогою CSS. Що таке веб-додатки і динамічні веб-сторінки Скорочена запис CSS. [Електронний ресурс] режим доступу: URL https://sukachoff.ru/uk/virusy/adaptivnoe-celoe-fonovoe-izobrazhenie-s-pomoshchyu-css-chto-takoe-veb-prilozheniya-i-dinamicheskie-veb-stra/(Дата звернення 01.12.2021). – Назва з екрану.

8. Веб додаток [Електронний ресурс] – режим доступу: URLhttps://znaimo.com.ua/Веб-додаток.(Дата звернення 11.11.2021). – Назва з екрану.
9. Що таке портфоліо в резюме? Приклади портфоліо на роботу. Чим відрізняється від резюме? Як скласти самостійно? [Електронний ресурс] – режим доступу: URLhttp://monaco.te.ua/scho-take-portfol-o-v-rezyume-prikladi-portfol-o-na-robotu-chim-v-dr-znya-t-sya-v-d-rezyume-yak-sklasti-samost-yno/ (Дата звернення 11.11.2021). – Назва з екрану.
10. Using Web Workers ? [Електронний ресурс] – режим доступу: URL https://developer.mozilla.org/enUS/docs/Web/API/Web_Workers_API/Using_web_workers (Дата звернення 11.11.2021). – Назва з екрану.