Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

ВЕРСТКА САЙТУ З ГОТОВОГО ДИЗАЙН-МАКЕТУ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ JQUERY
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-18-1
Іщенко А.О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Вечірська І.Д.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)
Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 Комп'ютерні науки
(код і повна назва)
Тип програми освітньо-професійна
Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Іщенку Артуру Олександровичу
(прізвище, ім'я, по батькові)
1. Тема роботи Верстка сайту з готового дизайн-макету з використанням фреймворку jQuery.

затверджена наказом по університету від «16» травня 2022 року № 541Ст
2. Термін подання студентом роботи до екзаменаційної комісії 30 травня 2022 р.
3. Вихідні дані до роботи: алгоритм написання коду HTML , алгоритм написання коду CSS, приклади створених сайтів з подібною структурою.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд основних методів Web програмування.
2. Аналіз основних засобів HTML.
3. Дослідження каскадних таблиць стилів.
4. Дослідження методів, інструментарію CSS Flex.
5. Розробка архітектури сайту.
6. Програмна реалізація сайту з використанням HTML та CSS.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри), Актуальність проблеми написання сайту з використанням Java Script.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Вечірська І.Д.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на до атестаційної роботи	18.04.2022	
2	Аналіз завдання, аналіз літератури	18.04.22-25.04.22	
3	Огляд існуючих програм для написання коду сайту	26.04.22-29.04.22	
4	Огляд існуючих алгоритмів написання сайту	30.04.22-02.05.22	
5	Програмна реалізація	03.05.22-10.05.22	
6	Тестування застосунку	11.05.22-15.05.22	
7	Оформлення пояснювальної записки	18.05.22-27.05.22	
8	Перевірка на плагіат	03.06.22	
9	Рецензування	03.06.22	
10	Підготовка презентації та доповіді	27.05.22-12.06.22	
11	Попередній захист атестаційної роботи	13.06.22	

Дата видачі завдання 18 квітня 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Вечірська І.Д.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 33 с., 10 рис., 31 джерело.

ВЕБ-САЙТ, ДИЗАЙН, МАКЕТ, ВЕРСТКА, ПРОГРАМУВАННЯ, ОПТИМІЗАЦІЯ.

Об’єктом роботи є послідовність написання сайту.

Метою роботи є аналітичний огляд літератури на тему, аналіз об'єкта дослідження та виявлення його особливостей, проектування та реалізація сайту, виклад виконаних у роботі практичних розробок.

Використано методи html, css, jquery, prepros. Проведено дослідження з метою виявлення більш практичного та актуального методу написання коду для сайту

У результаті роботи здійснена програмна реалізація досліджуваної теми.

WEBSITE, DESIGN, LAYOUT, LAYOUT, PROGRAMMING, OPTIMIZATION.

The object of the work is the sequence of writing the site.

The purpose of the work is an analytical review of the literature on the topic, analysis of the object of study and identification of its features, design and implementation of the site, presentation of practical developments.

Methods html, css, jquery, prepros are used. A study was conducted to identify a more practical and relevant method of writing code for the site

As a result of the work the program realization of the researched topic is carried out.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

HTML — мова розмітки гіпертексту (HyperText Markup Language)

CSS — Каскадні таблиці стилів (Cascading Style Sheets)

JS — динамічна, об'єктно-орієнтована прототипна мова програмування (JavaScript)

ВСТУП

Верстальник використовують HTML і CSS для створення веб-сторінок, тобто сторінок в інтернеті. За даними We Are Social, у січні 2020 року у світі налічувалося 4,54 млрд користувачів інтернету. Це 59% від загального населення планети. Тобто потенціал зростання інтернет-користувачів обчислюється мільярдами людей.

Ці дані дозволяють припустити, що інтернет в найближчому майбутньому зростатиме. Тобто з'являтимуться нові сайти, а для створення цих сайтів знадобиться HTML та CSS. Очевидно, ці мови розвиватимуться, а їхні можливості зростатимуть.

Веб-розробники, як фронтендери, так і бекендери, так чи інакше стикаються з версткою під час роботи. Не кожному веб-програмісту, особливо бекенд-розробнику, потрібно вміти верстати сторінки "з точністю до пікселя". Але без розуміння принципів HTML та CSS працювати у веб-розробці практично неможливо.

Краще за все розбиратися в якомусь питанні не тільки за допомогою здобуття теоретичних знань, а й використовувати їх на практиці.

Метою даної кваліфікаційної роботи є розробка функціонального та сучасного сайту. Відповідно до наявних сайтів-конкурентів та цільової аудиторії було опрацьовано матеріал, проведено науково обґрунтований вибір художнього оформлення сайту та його реалізація, проведено активну роботу зі створення функціональності та привабливості сайту.

Актуальність роботи полягає у збільшенні потоку клієнтів за допомогою розміщення сайту в інтернеті.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ WEB ПРОГРАМУВАННЯ

1.1 Аналіз основних засобів HTML

Equation Section (Next) (не видаляти зробити білим) (рядок)

HTML (від англ. HyperText Markup Language - "мова розмітки гіпертексту") - мова розмітки документів www. Браузер бере файл із розширенням html і відображається у вигляді документа, у зручній для людини формі.

Перша версія HTML була розроблена в 1989 Тімом Бенерс-Лі для популярного в минулому браузера Mosaic. Але на той час ні для мови, ні для браузера не знайшлося гідного застосування. У 1993 році з'явився HTML+, і ця версія також залишилася практично непоміченою. Початок широкого використання гіпертексту дала версія 2.0, яка з'явилася в червні 1994 року. Це був рік зростання популярності WWW у всьому світі. Елементи, включені у версію 2, здебільшого використовуються до цього дня.

У версії 3.0 HTML, яка з'явилася через рік, була реалізована можливість промальовування математичних символів (знаків інтервалу, нескінченності, дробу, дужок і тощо) за допомогою елементів мови. Під цю версію було розроблено браузер Arena. Але цей проект виявився тупиковим і не набув подальшого поширення.

1996 року з'явився HTML 3.2. Це було новаторське рішення, у специфікацію мови були введені фрейми, які стали тепер дуже популярними у розробників WEB-сторінок. Навіть зараз, на основі цієї специфікації, можна реалізувати цікаві дизайнерські рішення. Майже всі сучасні браузери підтримують версію 3.2, тому автори WEB-сторінок впевнені у працездатності всіх елементів.

Поряд із офіційними специфікаціями мови, які розроблялися організацією W3C (W3 Консорціум), компанії-виробники браузерів

створювали власні елементи (розширення). Згодом деякі з цих елементів після отримання загального визнання включилися до специфікації наступних версій мови. Але новаторське рішення - фрейми, що не були включені до специфікації 3.2. Але браузери підтримували кадри і багато книг, присвячених HTML, містили опис кадрів без згадки про те, що це нестандартні елементи. Згодом, фрейми стали стандартом де-факто. У версії 4 вони вже були включені на повній основі.

І навпаки, елементи APPLET і SCRIPT, необхідні розширення HTML іншими програмними кодами версії 3.2, не зіграли тієї ролі, яку мали зіграти. Це тим, що браузери різних версій по-різному інтерпретували програми різними мовами JAVA, JAVASCKRIPT, Visual Basic (VBScript). В результаті не вдалося отримати достатньо надійний працюючий код, і ці мови використовувалися любителями HTML в основному для експериментів.

Офіційна специфікація HTML 4 (Dynamic HTML) з'явилася 1997 року. У цей час вже було очевидно, що розвиток гіпертексту буде здійснюватися за рахунок скрипт - програмування. Це виявилося дещо ефективнішим, ніж вводити в мову все нові елементи. Браузери (Netscape Navigator 4, Microsoft Internet Explorer 4 та ін.), що з'явилися на той час, вже досить надійно інтерпретували програмний код (був встановлений певний рівень стандартизації). Проте проблеми розробників ще залишилися. Як приклад можна відзначити, що багато скриптів починаються з визначення версії браузера, щоб потім використовувати той чи інший фрагмент коду. Очевидно, що на програміста лягає обов'язок тестування сторінок на всіх популярних зараз браузерах.

В результаті використання всіх можливостей Dynamic HTML стало долею програмістів досить великих організацій, де є умови для розробки складних програм і всебічного їх тестування. Творцям особистих WEB-сторінок часом доводиться шукати компроміс між надійністю та новаторством, щоб отримати досить грамотний HTML-код.

Вже досить багато часу витрачено, щоб навчитися за допомогою HTML і браузера виводити на екран тексти.

Але це просто тексти. Використовуючи потужний текстовий процесор, такий, як Microsoft Word, зверстати оголошення, лист брошуру та невелику книгу можна набагато швидше, і результат буде кращої якості.

При цьому не треба писати програми, вникати у всі тонкощі численних тегів та їх атрибутів. Працюючи у Worde, практично не треба думати ні про що, крім змісту власного твору. Різноманітний оформлювальний інструментарій має інтуїтивно-зрозумілий інтерфейс та численну бібліотеку шрифтів, ліній, значків, рамочок, орнаментів, фігур, картинок та інших корисних "штучок", які роблять текст на екрані та папері приємним для ока.

Головних причин популярності HTML є три. Ось вони в порядку зростання ваги.

HTML-програмування дуже просте. У ньому немає традиційних алгоритмічних структур, таких як розвилка, цикл, процедура. Воно лінійне у своїй основі.

Багато авторів підручників навіть соромляться називати HTML-тексти програмами. Роботу HTML-проектувальника називають розміткою тексту. Насправді, у цій назві міститься методична помилка. Адже вона (назва) передбачає такий порядок роботи:

береться звичайний текст

і розмічається, тобто на нього накладається сітка з тегів так, щоб браузер зміг показати рядки тексту на екрані.

Однак, хороші гіпертекстові документи при такому підході отримати набагато складніше, ніж у випадку, коли структура HTML-документа планується ще до написання текстів. Адже гіпертекст влаштований принципово інакше.

Переносимість. Якщо працювати в Word, створюється документ для власного споживання. Немає проблем і тоді, коли надсилається робота у світ

як друковану копію. Якщо партнер, видавець або колега просить надіслати електронну версію - починаються проблеми.

Щоб документ з'явився на екрані у абонента, необхідно, щоб його комп'ютер та операційна система (платформа) дозволяли запустити улюблений редактор.

Для того, щоб документ завантажився в Word, дуже бажано збіг версій цього продукту у на різних пристроях.

Щоб текст на екрані не виглядав як давньонорвезький манускрипт, необхідно, щоб партнер мав на своєму комп'ютері шрифти, які були використані при створенні тексту.

Але навіть якщо всі ці умови виконані, все одно мало шансів на те, що ваш інший чоловік побачить текст неспотвореним (картинки вилазять з рамок, межі сторінок пливуть текстом). Це відбувається тому, що Word налаштований по-різному.

Після того, як по телефону (або електронній пошті) редактори будуть налаштовані однаково, текст все одно може виглядати по-різному вже через причини, відомі тільки фірмі Microsoft.

Все, що говорилося, називається непереносимістю продукту. Не в тому сенсі, що продукт когось не любить, а в тому, що різним користувачам неможливо або вкрай важко відображати у себе те, що зроблено в іншому місці.

HTML-документ є переносним продуктом. Це означає, що авторська праця буде легко доступна величезному числу користувачів, незалежно від марки комп'ютера та типу операційної системи. Саме ця властивість HTML-документів, а також відносно малий розмір, дозволили з успіхом використовувати HTML-технологію для підготовки WWW-сторінок в Інтернеті.

Перенесення HTML-документа досягається за рахунок того, що пересилається не екранне зображення, а програма. А зображення будує

браузер, виконуючи команди цієї програми. Щоправда різні браузери можуть працювати трохи по-різному.

HTML-документ – це гіпертекст.

Традиційний текст має лінійну структуру. І хоча користувач може читати будь-які його сторінки та рядки, гортаючи документ на екрані або паперовій книзі, автор припускає, що текст читається по порядку. Спочатку перша сторінка, потім друга і таке інше.

Перший етюд до гіпертексту. Деяке порушення лінійності звичайної книги вносять виноски, посилання інші сторінки і посилання іншу літературу.

Передбачається, що користувач може перервати лінійне читання в місці посилання, подивитися іншу частину тексту або навіть зовсім інший текст, а потім продовжити читання з місця переривання.

Другий етюд до гіпертексту. З-поміж маси лінійної текстової продукції виділяються словники, довідники, енциклопедії. Фізично текст влаштований лінійно: за сторінкою 10 обов'язково слідує сторінка 11, а за сотою - сто перша. Однак, передбачається, що читач звертається до книги для довідки, а не читає її всю поспіль від кірки до кірки.

Для спрощення навігації користувача в таких книгах передбачаються докладні змісти, алфавітні та тематичні покажчики. Розділи словника чи довідника мають систему розвинених перехресних посилань.

Третій етюд до гіпертексту. У вік бурхливих комп'ютерних технологій якось не хочеться копатися у великому словнику. Чому б цю рутину не доручити "залізному" другу з "м'якою" душею? Вженемо словник у "залізо" і налаштуємо "душу". Клацнув мишкою за потрібним словом - отримав результат: потрібний розділ книги на екрані.

Останній етюд до гіпертексту. Можливості комп'ютера народжують принципово нову ідею: чому б спочатку не проектувати текст для можливості читання не по-порядку, а по контексту. Такий підхід дозволяє

різним користувачам переглядати текст орієнтуючись на свої смаки, рівень володіння темою та поставлені завдання.

Структура комп'ютерної книги стає суттєво нелінійною, вона навіть перестає бути ієрархічною, а швидше нагадує сплутану рибальську мережу або порцію спагетті, перекинуту на підлогу.

Для навігації такою мережею передбачається простий спосіб: ті фрагменти документа, які мають переходи на інші його частини, якимось чином виділені. Просте інтерфейсне вплив на такому посиланні (натискання на Enter або мишачий клацання) перекидають користувача в інший інформаційний вузол.

Подивився, повернувся назад, або, не повертаючись, продовжив подорож за новим засланням.

1.1.1 Різновиди тегів HTML

Мова HTML складається з тегів. Кожен тег має певний сенс і призначення (параграф, малюнок, таблиця). Коли необхідно створити список або вставити таблицю, треба «розмітити» елемент, «обертаючи» його в певний тег:

– some text - парный тег;

–
 - непарный тег;

Основні теги:

– <!doctype html> необхідний, щоб браузер розумів, як слід інтерпретувати поточну веб-сторінку, оскільки HTML існує у кількох версіях різних за синтаксисом;

– <head></head> призначений для зберігання інших елементів, метою яких є допомогти браузеру в роботі з даним;

– <title>Title</title> - заголовок сторінки, який відображається в закладках браузерів;

– <body></body> призначений для вмісту веб-сторінки (контенту), що відображається у браузері.

Найголовнішим є тег – призначений для розміщення елементів (текст, зображення, посилання та ін) на html сторінці.

Сам по собі (без атрибутів та стилів CSS) тег ніяк не впливає на елементи HTML сторінок.

Теги, що роблять текст заголовками:

– <h1>заголовок першого рівня</h1>;

– <h2>заголовок другого рівня</h2>;

– <h3>заголовок третього рівня</h3>;

– <h4>заголовок четвертого рівня</h4>;

– <h5>заголовок п'ятого рівня</h5>;

– <h6>заголовок шостого рівня</h6>;

Теги   виділяють важливі фрагменти тексту.

Теги   задають жирність тексту.

1.2 Каскадні таблиці стилів

CSS (Cascading Style Sheets) – каскадні таблиці стилів.

CSS, як і будь-яка мова, має синтаксис. У ньому немає елементів, параметрів, тегів. У ньому є правила – CSS складається з селектора та блоку оголошення стилів.

Сам блок оголошення стилів складається з властивостей та їх значень, розділених крапкою з комою.

Вперше каскадні таблиці стилів CSS було реалізовано у браузері Internet Explorer 3.0. Однак у той час розвиток CSS перебував у зародковому стані, тому правила складання стильових шаблонів були дуже розрізненими.

З моменту свого виникнення структура CSS була кілька разів переглянута, до неї були додані нові елементи та прибрані (видозмінені)

старі. Існують три рівні CSS, що визначаються наявністю завершеної редакції структури. Це: CSS 1 (перший рівень структури стильових шаблонів, остаточно затверджений 11 січня 1999 року), CSS 2 (другий рівень стильових конструкцій, початок обговорення якого датується травнем 1998 року) та CSS 3 (третій рівень стильового оформлення електронних документів, прийнятий до обговорення 2 травня 2001 року, на момент написання книги перебував у стадії опрацювання).

На завершення розмови про рівні CSS слід додати, що перехід від одного рівня до іншого, в основному, супроводжувався деякими видозмінами в структурі та правилах стильового оформлення, технологічними доповненнями, а також спробами систематизувати застосування CSS.

Саме третій рівень (CSS 3) позиціонується розробниками як єдиної системи представлення стилів в електронному документі, заснованої на використанні спеціальних модулів.

1.2.1 Селектори

Як селектор CSS можуть виступати:

Елементи HTML. Перевизначення стилю для конкретного елемента сторінки:

BODY { color: orange; }.

У цьому випадку весь текст у межах розділу body буде помаранчевим. При додаванні, наприклад, таблиці – призначення стильового шаблону пропаде для тексту всередині клітинок.

Використання класів дозволяє перевизначати стиль як конкретного елемента, так будь-якого елемента, якому присвоєний даний клас. Найменування класу починається з точки і зазвичай пишеться малими літерами (допускається використання латинських літер та цифр, але

наявність спеціальних символів, нижніх підкреслень та інших нестандартних елементів не рекомендується).

.red { color: red; }

У цьому випадку будь-який елемент HTML, що дозволяє змінювати колір, відображатиметься червоним, якщо йому присвоїти клас .red:

<FONT CLASS="red">ТеКСТ червоним кольором</ГОИТ>

АБО

<HR CLASS="red">

Якщо ми доповнимо селектор класу найменуванням конкретного HTML-елемента, то дія стильового правила поширюватиметься лише на цей елемент:

HR.red { color: red; }

При вказівці класів стильового шаблону слід уважно стежити за тим, чи підтримує HTML-елемент типеревизначення стилю. Наприклад, запис виду:

HR {text-align: justify; };

буде безглуздою, тому що горизонтальний роздільник відноситься до галузі структурного форматування і не може містити текст, який, згідно з стильовим правилом, слід розтягнути по ширині.

Запис ідентифікатора починається з символу «#» і закінчується найменуванням:

fblack { background-color: black; }.

Наприклад, надавши даний ідентифікатор тегу, отримуємо осередок таблиці, залитий чорним кольором:

<TD 1П="blac1<">Комірка чорного кольору</ТО>

Порівнявши функції селектора класу та ідентифікатора, можна поставитися цілком закономірним питанням - чим же відрізняються ці селектори? Справді, формат визначення селектора обох типів аналогічний структурі та присвоєнню HTML-елементам. Однак, селектор ідентифікатора часто застосовується для завдання унікального імені елементу, який задіяний

у програмному сценарії (скрипті). На відміну від нього, селектор класу обмежується в основному застосуванням у стильових шаблонах.

Насамкінець необхідно звернути особливу увагу на неможливість поєднання селекторів різних типів. Не можна одночасно перевизначити стиль для стандартного елемента HTML і для нього, але за конкретним класом/ідентифікатором.

Розрізняють такі селектори:

- Класи (. myclass);
- ID (#myid);
- Елементи (div).

Стандарт CSS визначає пріоритети, в порядку яких застосовуються правила стилів, якщо для якогось елемента підходять властивості кількох правил одночасно (або, поодинокі, в одному правилі є однойменні властивості). Це називається «каскадом», у якому для властивостей розраховуються пріоритети чи «ваги», що робить результати передбачуваними.

Пріоритети розраховуються від більшого до меншого:

- якість задана з допомогою !important;
- стиль прописаний інлайново в тезі в HTML через атрибут style="";
- кількість ідентифікаторів (#id) у селекторі;
- кількість класів (. class) і псевдокласів (: pseudoclass) у селекторі;
- кількість імен тегів у селекторі.

1.3 Дослідження методів інструментарію CSS Flex

Протягом тривалого часу використовували таблиці, float-елементи, inline-block та інші CSS властивості, щоб надати блокам потрібне розташування. Прості речі як вертикальне центрування здійснювалися досить

складно. Створення ж макету на основі рідких сіток взагалі вважається верхом професіоналізму, ось чому широкого поширення набули CSS-фреймворки на основі сіток.

Вирішення всіх цих проблем є Flexbox(рис. 1.1).

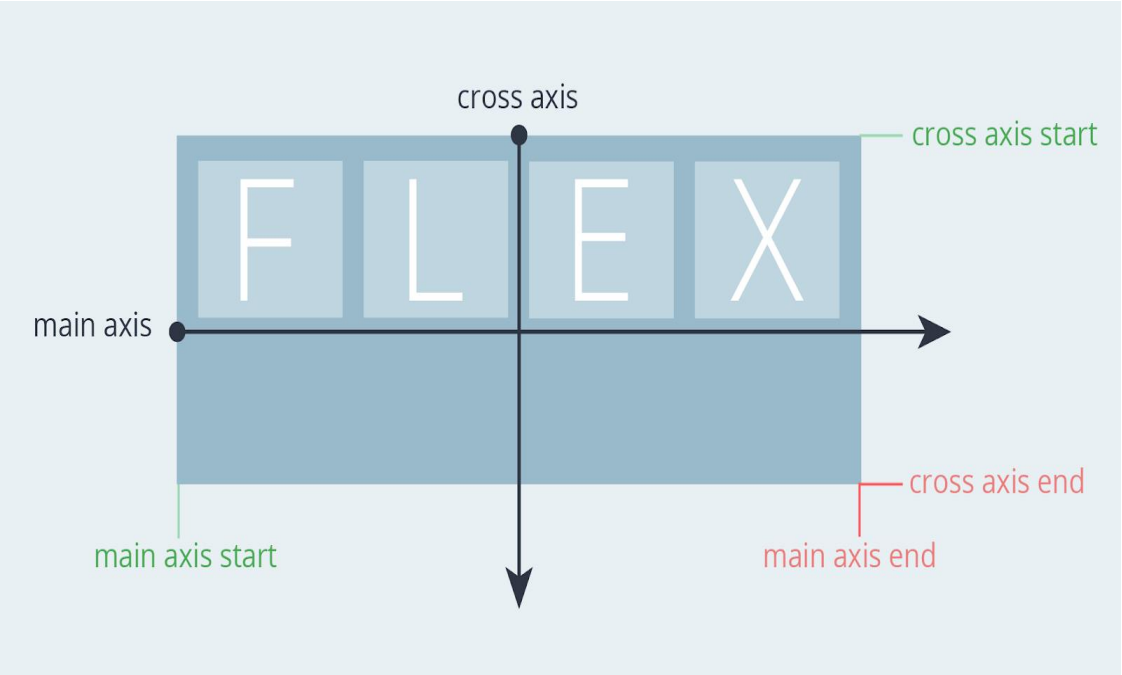

Рисунок 1.1 – Розташування елементів Flexbox

Flexbox покликана кардинально змінити ситуацію на краще при вирішенні величезної кількості завдань. Flexbox дозволяє контролювати розмір, порядок та вирівнювання елементів по кількох осях, розподіл вільного місця між елементами та багато іншого.

Всі блоки дуже легко робляться "гумовим", що вже випливає з назви "flex". Елементи можуть стискатися та розтягуватися за заданими правилами, займаючи потрібний простір.

Є можливість вирівнювання по вертикалі та горизонталі, базової лінії тексту.

Розташування елементів у HTML не має вирішального значення. Його можна змінити в CSS. Це особливо важливо для деяких аспектів responsive верстки.

1.3.1 Що таке Flexbox та його переваги

Елементи можуть автоматично вишиковуватися в кілька рядків/стовпців, займаючи все надане місце.

Багато мов у світі використовують написання праворуч наліво rtl (right-to-left), на відміну від звичного ltr (left-to-right). Flexbox пристосований для цього. У ньому є поняття початку та кінця, а не права та ліва. Тобто, у браузерах з локаллю rtl всі елементи будуть автоматично розміщені у реверсному порядку.

Синтаксис CSS Flex дуже простий і швидко освоюється.

Все це можливо лише в останніх версіях браузерів, тому не завжди можна буде використовувати флекс у роботі.

Підтримка браузерами останньої специфікації flexbox:

- Chrome 29+;
- Firefox 28+;
- Internet Explorer 11+;
- Opera 17+;
- Safari 6.1+ (з префіксом -webkit-).

1.3.2 Flex макет. Головна та поперечна вісь

Flex-макет(рис.1.2) складається з батьківського контейнера, вказаного як flex-container, та його дочірніх елементів, які називаються flex-items.

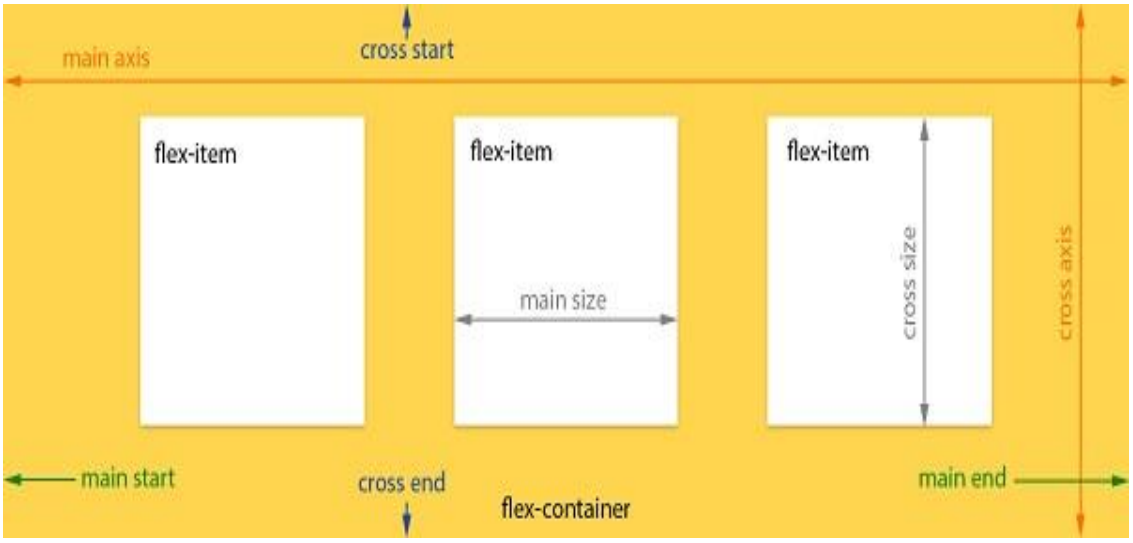

Рисунок 1.2 – Flex-макет

Розташовуються елементи у flexbox за допомогою таких команд:

– main-axis - головна вісь, вздовж якої розташовуються flex-елементи. Зверніть увагу, вона необов'язково має бути горизонтальною, все залежить від якості justify-content;

– main-start main-end - flex-елементи розміщуються у контейнері від позиції main-start до позиції main-end;

– main size – ширина або висота flex-елемента в залежності від обраної основної величини. Основна величина може бути або завширшки, або заввишки елемента;

– cross axis – поперечна вісь, перпендикулярна до головної. Її напрямок залежить від напрямку головної осі;

– cross-start | cross-end - flex-рядки заповнюються елементами та розміщуються у контейнері від позиції cross-start і до позиції cross-end;

– cross size - ширина чи висота flex-елемента залежно від обраної розмірності дорівнює цій величині. Ця властивість збігається з width або height елементом залежно від вибраної розмірності(рис.1.3).

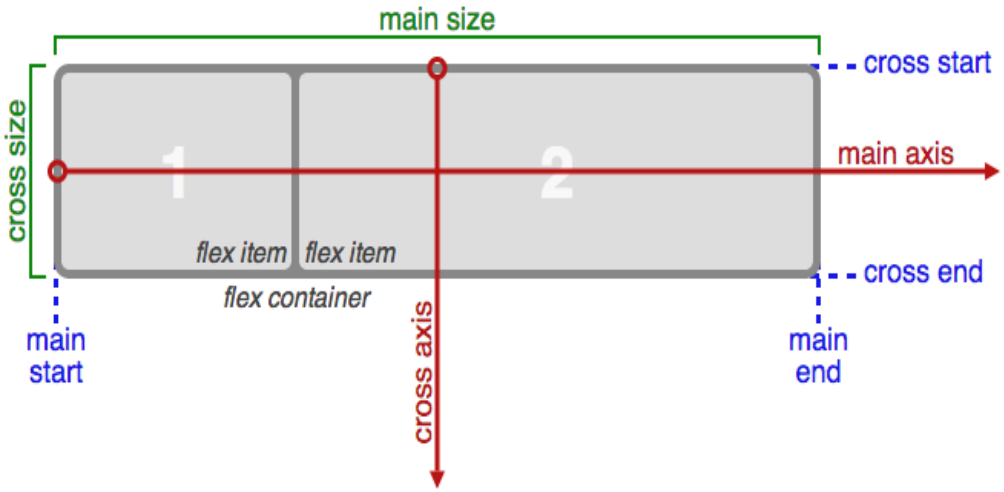

Рисунок 1.3 – розташування елементів у flexbox

1.4 Дослідження мови JavaScript

JavaScript спочатку створювався для того, щоб зробити web-сторінки «живими». Програми цією мовою називаються скриптами. У браузері вони підключаються безпосередньо до HTML і, як тільки завантажується сторінка - відразу виконуються.

Коли створювалася мова JavaScript, у неї спочатку була інша назва: LiveScript. Але тоді була дуже популярна мова Java, і маркетологи вирішили, що подібна назва зробить нову мову більш популярною.

Планувалося, що JavaScript буде таким собі «молодшим братом» Java. Однак, історія розпорядилася по-своєму, JavaScript сильно виріс, і зараз це абсолютно незалежна мова, зі своєю специфікацією, яка називається ECMAScript, і до Java не має жодного відношення.

Для виконання програм, не важливо якою мовою, існують два способи: «компіляція» та «інтерпретація».

Компіляція – це коли вихідний код програми, з допомогою спеціального інструменту, інший програми, що називається «компілятор», перетворюється на іншу мову, зазвичай – в машинний код. Цей машинний код потім поширюється та запускається. При цьому вихідний код програми залишається у розробника.

Інтерпретація – це коли вихідний код програми отримує інший інструмент, який називають інтерпретатор, і виконує його як є. При цьому поширюється саме вихідний код (скрипт). Цей підхід застосовується у браузерах для JavaScript.

Сучасні інтерпретатори перед виконанням перетворюють JavaScript на машинний код або близько до нього, оптимізують, а вже потім виконують. І навіть під час виконання намагаються оптимізувати. Тому JavaScript працює дуже швидко.

Сучасний JS – це «безпечна» мова програмування загального призначення. Він не надає низькорівневих засобів роботи з пам'яттю,

процесором, оскільки спочатку був орієнтований на браузери, в яких це не потрібно. Інші можливості – залежать від оточення, у якому запущено JS. У браузері JS вміє робити все, що відноситься до маніпуляції зі сторінкою, взаємодії з відвідувачем і, певною мірою, із сервером:

– Створювати нові HTML-теги, видаляти існуючі, змінювати стилі елементів, ховати, показувати елементи тощо;

– Реагувати на дії відвідувача, обробляти кліки миші, переміщення курсору, натискання клавіатури тощо;

– Надсилати запити на сервер та завантажувати дані без перезавантаження сторінки (ця технологія називається "AJAX").

JavaScript - швидка і потужна мова, але браузер накладає на його виконання деякі обмеження.

Це зроблено для безпеки користувачів, щоб зловмисник не міг за допомогою JavaScript отримати особисті дані або нашкодити комп'ютеру користувача.

Цих обмежень немає там, де JavaScript використовується поза браузером, наприклад, на сервері. Крім того, сучасні браузери надають свої механізми по встановленню плагінів і розширень, які мають розширені можливості, але вимагають спеціальних дій щодо встановлення від користувача.

Більшість можливостей JavaScript у браузері обмежено поточним вікном та сторінкою.

1.4.1 Програмована мова стилів LESS

CSS. Він простий та зрозумілий. Це рушійна сила Інтернету, але він занадто обмежений і важко керувати. Цю мову можна зробити більш корисною, використовуючи динамічний CSS за допомогою LESS.

І тоді замість використання #FF9F94 для отримання темно-персикового кольору просто зберегти значення цього кольору в змінній для подальшого використання. Щоб перефарбувати сайт досить буде змінити значення змінної всього в одному місці і все.

Це все можливе з використанням Less.

LESS – це надбудова над CSS. Це означає, що будь-який CSS код – це валідний LESS, але додаткові елементи LESS не працюватимуть у простому CSS. Це чудово, тому що існуючий CSS вже є працездатним кодом LESS, що зменшує поріг входження в нову технологію.

LESS додає багато потрібних динамічних властивостей у CSS. Він вводить змінні, операції, function-like елементи та домішки. Можливість писати таблиці стилів модульно позбавить вас багатьох клопотів.

Існують два способи використання LESS. Можна створити LESS файл і конвертувати його за допомогою Javascript на льоту або скомпілювати його заздалегідь і використовувати CSS файл, що вийшов.

Використання LESS та Javascript файлів.

Для початку потрібно завантажити з сайту LESS Javascript файл та прив'язати його до сторінки як будь-який інший js скрипт:

<script src="less.js" type="text/javascript"></script>

Потім створити файл з розширенням .less та прив'язати його за допомогою такого коду:

<link rel="stylesheet/less" type="text/css" href="style.less">

Переконайтися, що прикріплено файл LESS перед JS. Тепер файл LESS буде працювати також як і звичайний CSS.

Змінні в LESS працюють так само як у PHP, JS та у більшості інших мов програмування. Можна використовувати їх для зберігання значення, а потім використовувати змінні замість самого значення кожного разу, коли це потрібно.

@header-font: Georgia; h1, h2, h3, h4 { font-family: @header-font; } .large { font-family:@header-font; }

У прикладі вище оголошено змінну @header-font і записано туди значення Georgia. Тепер можна використовувати цю змінну завжди, коли необхідно встановити Georgia шрифт. Якщо буде вирішено, що Trebuchet MS краще підходить для заголовків, то не потрібно буде переглядати весь файл, можна просто змінити значення змінної.

Область видимості змінних визначає місця, де вони доступні. Якщо визначити змінну на самому початку файлу LESS, то вона буде доступна для будь-якого коду написаного після.

Також можна визначати змінну всередині правила CSS. У цьому випадку змінні не будуть доступні поза цим правилом, вони можуть бути використані локально:

a { @color: #ff9900; color:@color; }

button { background: @color; }.

У цьому прикладі LESS не буде конвертовано через помилку, color не визначена для використання всередині елемента button. Якщо змінна оголошена поза елементом і всередині іншого елемента, вона буде доступна лише локально.

1.4.2 Скриптова метамова SASS

SASS (Syntactically Awesome Stylesheets) - це скриптова метамова, яка компілюється в звичайні CSS-стилі. Всі, хто стикається з CSS розміром більше 500 рядків, мають справу з головним болем на тему того, як його спростити. На жаль, з часів розробки стандартів каскадних стилів їхня структура кардинально не змінювалася. Вимоги до верстки — ускладнилися в рази. Якщо колись 50-70 рядків стилів могли оформити простий сайт, сьогодні такого обсягу вистачить хіба що на header.

Розширення файлів SASS можуть бути .sass і .scss — це залежить від вибраного синтаксису. Браузер, втім, не розуміє жодного з них, тому для

порозуміння потрібно використовувати компілятор. Його завдання – привести SASS у зрозумілий класичний CSS, який буде розпізнаний будь-яким браузером.

Роль компілятора може виконувати серверний js або програма, встановлена у вас на робочій машині та моніторить зміни в робочих файлах.

Sass дозволяє використовувати функції недоступні в самому CSS, наприклад змінні, вкладеності, міксини, успадкування та інші приємні речі, що повертають зручність написання CSS. Як тільки починається користування Sass, препроцесор обробляє Sass-файл і зберігає його як простий CSS-файл, який можна використовувати на будь-якому сайті.

1.4.3 Бібліотека JQUERY

jQuery - бібліотека JavaScript, тобто, збірник класів та/або функцій мовою JavaScript. Бібліотека фокусується на взаємодії JavaScript та HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елемента сітки, звертатися до атрибутів та вмісту елементів, маніпулювати ними.

Перш за все, щоб працювати з плагінами анімації за допомогою JQUERY ми повинні підключити цей самий JQUERY.

Є два способи підключення бібліотеки JQUERY - онлайн та локальне.

Підключення бібліотеки JQUERY онлайн:

Знаходимо в гуглі jquery link google і в перших посиланнях знаходимо потрібну онлайн бібліотеку:

<script src="//ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

Але краще використовувати бібліотеку JQUERY локально. Тому робимо інший запит у гуглі: jquery, і переходимо на сайт самого jquery, де на першій же сторінці є кнопка Download Jquery. Відкриється сторінка зі

списком різних збірок цієї бібліотеки. Необхідно лише Download the compressed jQuery, після завантаження цієї бібліотеки, необхідно перенести її в папку JS проекту і підключити її в head так само, як підключений файл MAIN.JS.

1.5 Постановка задачі

Об’єктом роботи є послідовність написання сайту.

Метою роботи є аналітичний огляд літератури на тему, аналіз об'єкта дослідження та виявлення його особливостей, проектування та реалізація сайту, виклад виконаних у роботі практичних розробок.

Для досягнення мети необхідно вирішити такі завдання:

- провести аналіз існуючих методів написання сайтів;
- розробити алгоритм написання коду використовуючи HTML та CSS;
- реалізувати сайт на базі наявних знанн.

2 ДИЗАЙН САЙТУ

2.1 Композиційна побудова сторінок

Початковим етапом розробки html-шаблону сайту є створення макету сторінки. Це дозволить уникнути помилок у постановці композиції. Проаналізувавши цілі та завдання сайту було вирішено, що макет сайту повинен являти собою поєднання наступних елементів:

- header сторінки;
- верхній горизонтальний елемент (включає "Пошук");
- ліва навігація (головне меню);
- основна частина (контент сторінки);
- права навігація;
- footer сторінки.

Блок основного контенту розташований у центральній зоні. У цьому блоці буде розташована основна інформація з того чи іншого розділу.

У header також розташований блок опитування який також продубльований у footer для зручності клієнта.

Сформований даним чином макет сайту дозволяє судити про можливість його використання як основу для дизайну сторінки. Елементи закомпоновані добре, положення кожного елемента у загальній структурі відповідають їх «важливості» та призначенню. При побудові макета враховано закони зорового сприйняття та правила юзабіліті. Далі можна розпочинати розробку дизайну головної сторінки сайту.

2.2 Колірне рішення

Один із найважливіших елементів дизайну, що впливає на сприйняття, це колір. За допомогою різних кольорів можна створити затишок, викликати відчуття страху або радість. У дизайні інтерфейсу корисно використати натуральні кольори. Колірні комбінації, що зустрічаються в природі, мають найбільшу здатність виділятися (допомагаючи створити веб-сайт, що більш запам'ятовується), спрямовувати (дозволяючи користувачам зосереджуватися на взаємодії з елементами сайту), захоплювати (роблячи загальне розташування елементів сторінки більш зручним і привабливим), і надихати (пропонуючи нові ідеї у виборі кольорів).

Мабуть, жоден інший елемент дизайну не впливає на наше відчуття простору, як колір. Колір впливає на фізіологічні процеси людини та її психологічний стан. Знаючи особливості кожного кольору, можна сформувати певний образ, викликати певні емоції та асоціації.

Створення для нашого проекту відчуття комфорту дуже важливе. Живі тони фіолетового кольору задають основне тло вмісту сторінки. Цей колір добре збалансований із кольором навігації та «футера» сторінки. Фіолетовий колір дає контрастність за допомогою чого ми можемо підкреслити потрібні нам деталі сайту.

Як основний колір сторінки заданий білий. Цей колір використовували як підкладку для всього інформаційного контенту. Цей колір має м'який вплив, розширює простір, полегшує сприйняття, дає впевненість.

Завдання колірного оформлення полягало в розробці такого поєднання відтінків, щоб підкреслити певні елементи сайту, але в той же час мали яскравість, помірність, своєрідний колорит. У процесі вибору колірного рішення було реалізовано завдання створення яскравого, але водночас заспокоюючого дизайну.

2.3 Вибір шрифтів

Шрифти є: одним із найважливіших елементів дизайну сайту; елемент корпоративного стилю компанії; носієм інформації; елементами концентрації уваги користувача. Вони грають величезну роль і багато в чому визначають якість сайту.

Як елемент дизайну шрифти сайту повинні становити єдине ціле з дизайном сайту як за розміром, так і за гарнітурою, і за кольором. Як носій інформації шрифти визначають комфортність читання матеріалів сайту та втому користувача.

Простий текст є основою передачі відвідувачам. Ігнорування такого, здавалося б, незначного фактора, як вибір шрифту для сайту, не тільки призведе до неминучого зниження прибутку, який має приносити сайт, але може зробити сайт збитковим, а у самому найгіршому випадку ви можете отримати позов до суду за плагіат у використанні платних шрифтів без підписки. Враховуючи важливість правильного використання шрифтів, було сформовано основні правила, яких необхідно буде дотримуватись при створенні сторінок сайту:

– розмір основного шрифту документа – не менше 10 px;

– забороняється застосування згладжування та ефектів для шрифтів в основному тексті та заголовках сайту, а також у тексті розміром менше 14 px;

– забороняється використовувати ефекти прозорості для основного тексту сайту;

– колір тексту вказується явно у властивостях шрифту, а не через ефекти шару.

Для основного тексту сайту, заголовків та посилань необхідно використовувати безпечні шрифти. Безпечні шрифти - шрифти, які однаково (або практично однаково) виглядають у різних браузерах та на різних платформах. Використання цих шрифтів забезпечує коректне відображення сайту будь-якого відвідувача.

Виходячи із загального дизайну сайту, розташування його окремих елементів, характеру та кількості викладеного на сайті матеріалу, обрана шрифтова композиція розглядається найбільш оптимальною для нашого проекту.

2.3.1 Підключення унікальних шрифтів які є в бібліотеці Google

Для початку потрібно в проекті створити окрему папку font на рівні з папкою images , у цій папці треба створити CSS файл із назвою font-face.

Тепер цей файл підключається в head проекту, за допомогою дописання в head рядка:

<link rel="stylesheet" type="text/css" href="font/font-face.css" />.

Далі, треба дізнатися чи є шрифт у списку гугл-фонтів, написавши в запиті назву шрифту необхідно додати google font. Далі залежить від результатів пошуку. Якщо шрифт є в Google списку:

Наприклад, потрібен шрифт «open sans» https://fonts.google.com/specimen/Open+Sans(рис2.1).

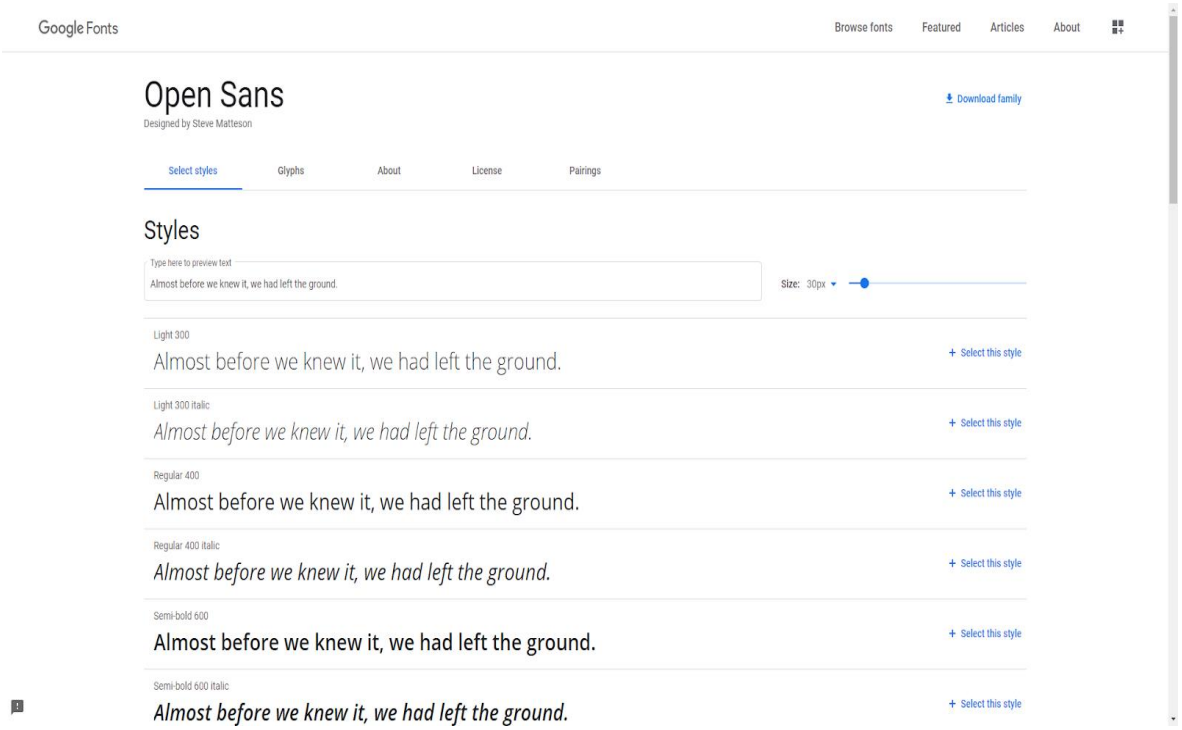

Рисунок 2.1 – панель вибру шрифту.

Праворуч від шрифту є посилання Select this font при натисканні внизу екрана з'являється ще одне посилання з чорним тлом.

Після відкриється вікно для вибору необхідних накреслень, стилі та жирність шрифту. Над кожним рядком написано формат, наприклад: Light 300. Якщо він потрібен за макетом, то треба обрати Select this style після чого відкриється бічна панель налаштування(рис 2.2).

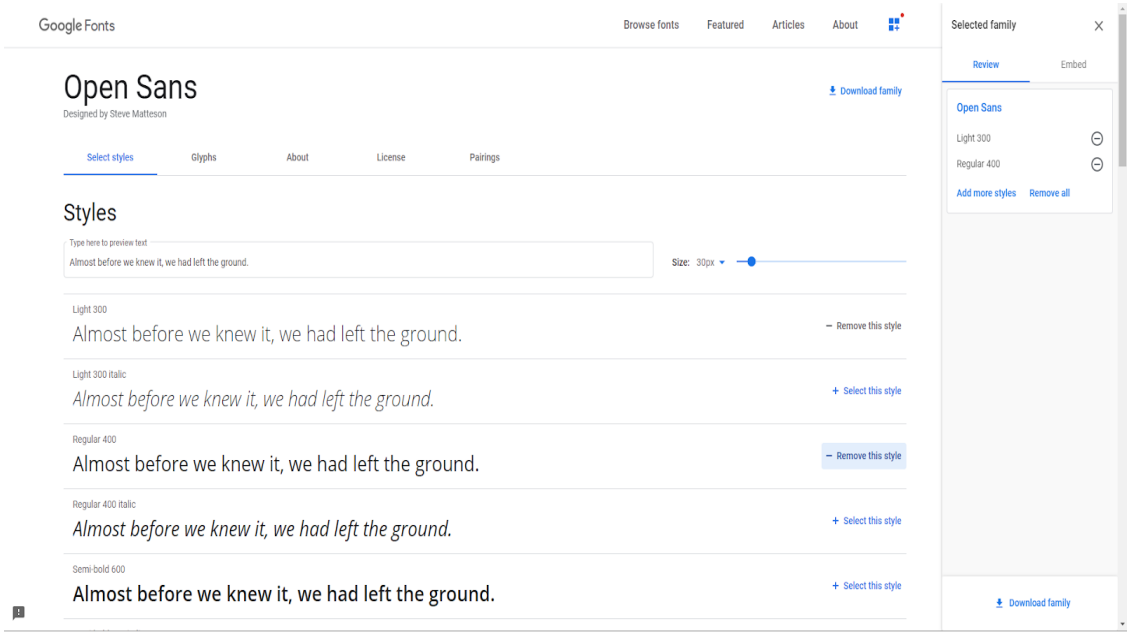

Рисунок 2.2 - бічна панель налаштування.

У бічній панелі є вибрані стилі шрифту. При натисканні на Embed відкриється друга вкладка налаштувань(рис 2.3).

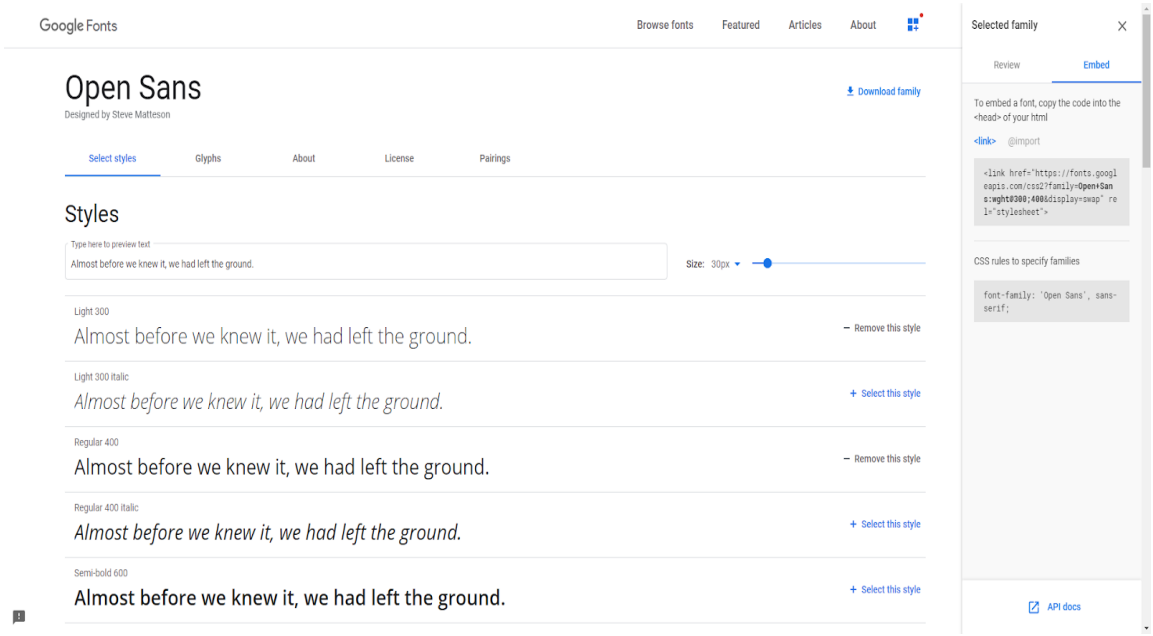

Рисунок 2.3 – друга вкладка налаштувань.

У вкладці Embed знаходиться тег Link який треба скопіювати в проект. Head частина HTML наприкінці всіх тегів Link, щоб шрифт підключався після основних стилів проекту.

Далі стиль font-family треба скопіювати та вставити його у стилі до необхідних об'єктів.

2.3.2 Підключення унікальних шрифтів яких немає в бібліотеці Google

У гуглі треба знайти необхідний шрифт у форматі OTF та завантажити його.

Для прикладу був використаний шрифт Helvetica.

Використавши запит Helvetica download otf, у першому посиланні відкриється сайт http://font.in.ua/family/Helvetica-neue де є список шрифту з різними стилями та жирністю. Наприклад потрібен перший у списку шрифт, тиснемо на посилання(рис.2.4)

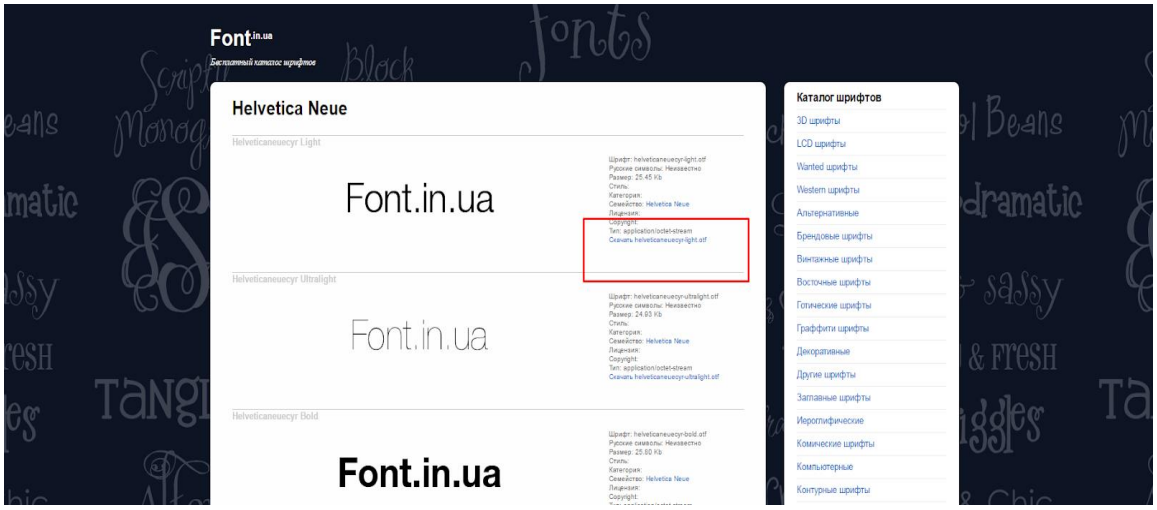

Рисунок 2.4 – посилання на шрифт.

Цей файл OTF, TTF переноситься в папку font. Далі треба відкрити сайт :

https://www.fontsquirrel.com/tools/webfont-generator або

https://transfonter.org.

Треба натиснути кнопку Upload і погодитися з угодою користування авторськими правами на шрифти(рис. 2.5).

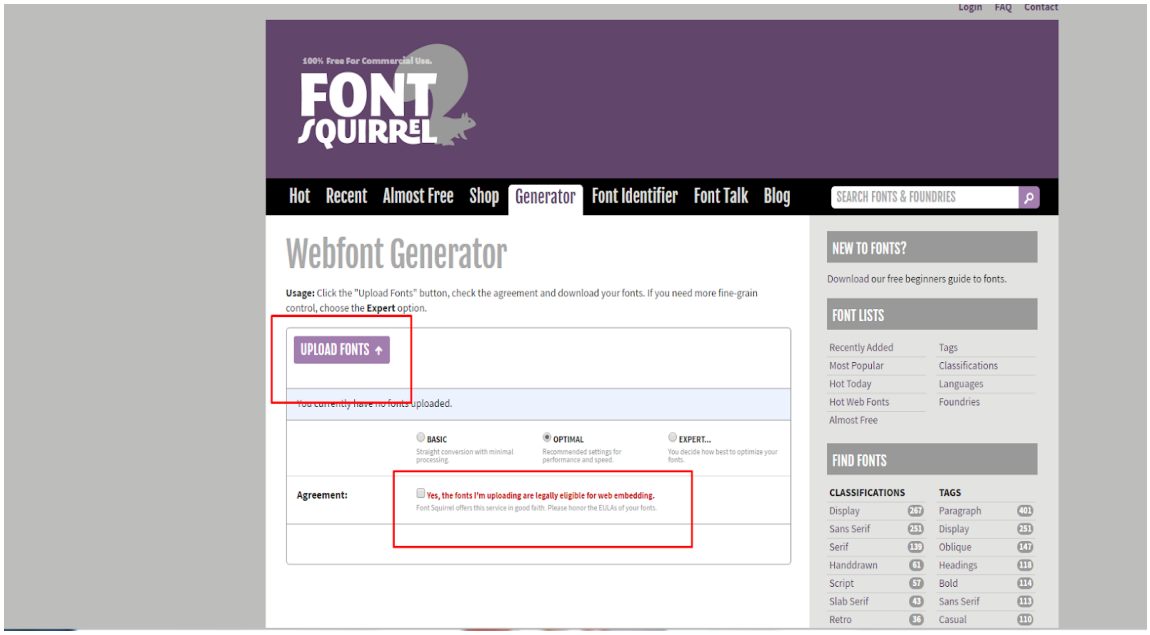

Рисунок 2.5 – завантаження web шрифту

Після обробки файлу з'являється кнопка download , при натисканні завантажується вже web шрифт з яким можна працювати. Треба перенести всі файли в папку font а з файлу stylessheet перенести всі записи в файл font-face. Треба відкрити файл, тому що треба відредагувати шлях (URL) шрифту(рис. 2.6).

```
@font-face {
  font-family: 'helveticaneuecyrlight';
  src: url('helveticaneuecyr-light-webfont.woff2') format('woff2'),
       url('helveticaneuecyr-light-webfont.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}

@font-face {
  font-family: 'helveticaneuecyrlight';
  src: url('../font/helveticaneuecyr-light-webfont.woff2') format('woff2'),
       url('../font/helveticaneuecyr-light-webfont.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}
```

Рисунок 2.6 – код що підлягає редагуванню

3 ДОДАВАННЯ ПРОГРАМНОГО КОМПОНЕНТА

3.1 Обґрунтування вибору середовища програмної реалізації

Головна особливість Atom - багаті можливості настроювання. Редактор можна настроїти на свій смак. Спочатку в нього вбудовані файл-менеджер, просунуті функції пошуку та заміни, різноманітні курсори, опції згортання коду, ясний інтерфейс, можливість імпорту правил та тем із TextMate.

Десктопна програма Atom має повний доступ до файлової системи, природні для операційної системи меню та панель команд. При цьому вона ідеально пристосована для веб-програмування: можна додавати власні функції для редагування CSS, HTML та JavaScript. Потрібно також відзначити інтеграцію з Node.js, включаючи запуск веб-сервера прямо з редактора. Архітектура програми проста і зрозуміла кожному: можна замінити будь-який пакет своїм власним та загрузити його в центральний репозиторій, щоб ним скористався будь-хто.

Також варто відмітити, що за допомогою Atom користатися Git дуже просто тому що Atom є розробкою GitHub.

Однак стосовно програми для компіляції кожен може обрати сам, суттєво на процес роботи це не вплине.

3.2 Програмна реалізація

Розробка HTML-структури відбувається у кілька етапів:

- Розробка HTML архітектури(рис. 3.1);
- Стилізація об'єктів та HTML елементів;
- Анімація окремих частин структури.

HTML

```
<!DOCTYPE html>
<html>
<head>
<meta charset=utf-8 />
<title>Flexbox examples</title>
</head>
<body>
  <div class="flex-container">
    <div class="flex-item">
      Curabitur ac vestibulum mi
    </div>
    <div class="flex-item">
      In viverra dapibus
    </div>
    <div class="flex-item">
      Fusce tincidunt diam et
      </div>
    <div class="flex-item">
      Nulla in dui vel est
    </div>
    <div class="flex-item">
      at diam in lobortis
    </div>
  </div>

</body>
</html>
```

CSS

```
body {
  padding: 20px;
  background: white;
  }
.flex-container {
  padding: 10px;
  background: gold;
  border-radius: 10px;
  }
.flex-item {
  margin: 10px;
  padding: 5px;
  background: tomato;
  border-radius: 5px;
  border: 1px solid #FFF;
  }
```

Рисунок 3.1 – розробка HTML архітектури.

3.2.1 Реалізація взаємодії користувача з адміністратором

Загальний недолік багатьох інформаційних ресурсів у тому, що вони лише надають інформацію, але не виводять користувача на поведінковий рівень. Це значно знижує можливість досягнення цілей сайту та його ефективність. Результатом взаємодії з сайтом має стати ухвалення користувачем певного рішення. Ідеальним буде, якщо це рішення співпаде із завданнями творця інформаційного ресурсу.

Тому було відтворено невеликий опитувальник який зустрічає користувача на початку сайту та в кінці. За рахунок того, що один і той опитувальник користувач може спочатку обрати цікаві йому пункти а після вивчення сайту змінити щось в своєму виборі або залишити все як і було (рис. 3.2).

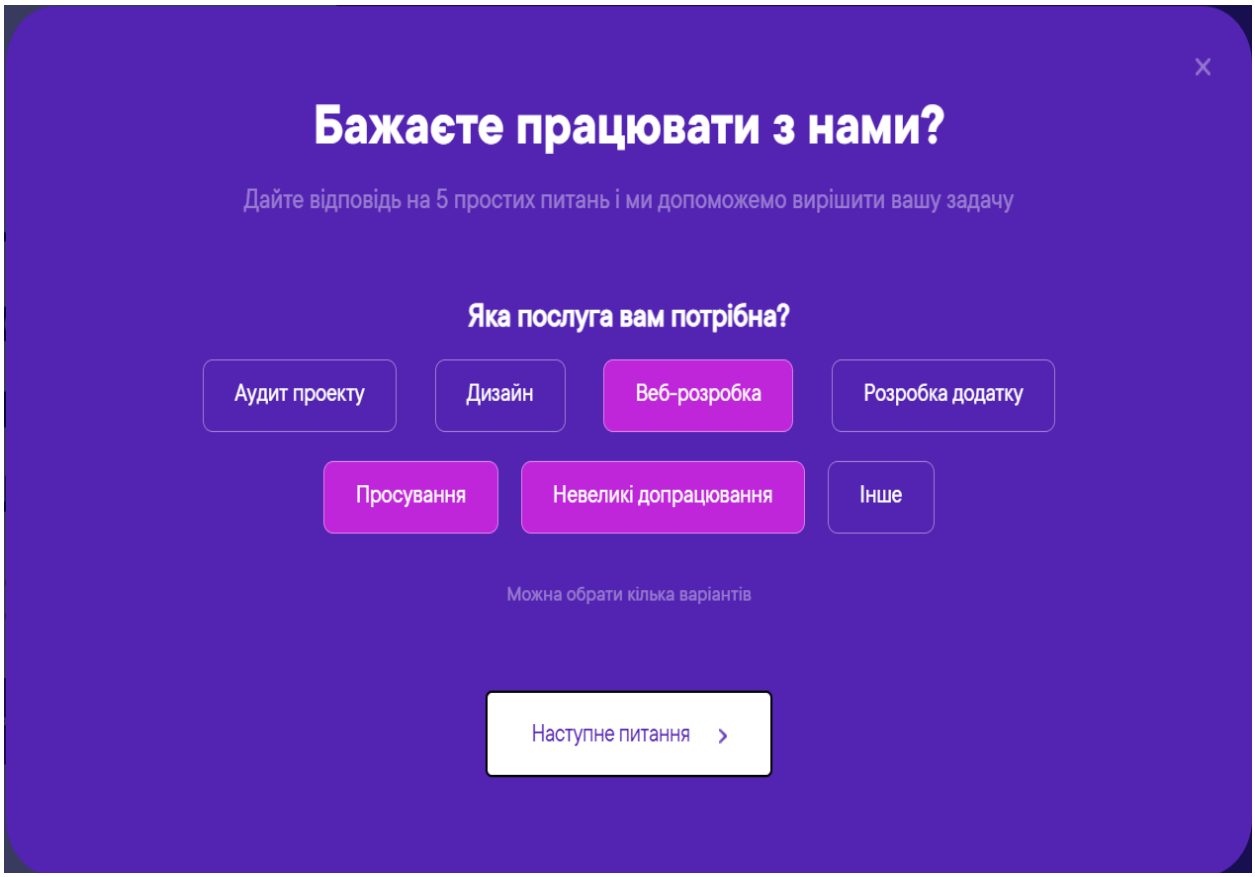

Рисунок 3.2 – опитувальник.

3.3 Тестування розробленої моделі

Під час тестування помилок у роботі сайту знайдено не було, однак була виявлена необхідність змінити код який відповідає за опитувальник, бо при натисканні на цікавлячий пункт він не змінював колір, як це задумав дизайнер. Тому для кожного пункту було додано checkbox який знаходиться над кожним пунктом, має такий самий розмір але має функцію display:none(не є видимим об’єктом) однак зберігає всі свої можливості. Тобто при натисканні на пункт фактично читач натискає на невидимий який в свою чергу дає сигнал браузеру змінити колір кнопки.

Також у користувача немає можливості якось порушити працездатність сайту бо всі елементи які можуть це зробити є доступними тільки для адміністратора сайту.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений алгоритм для верстки сайту. Для реалізації було обране середовище Atom 1.60.0. Це обумовлено тим, що Atom є більш функціональним та простішим за рахунок того що ця програма допомагає у написанні коду та виявленні помилок. Однак через це Atom є однією з найбільш вимогливих до потужності комп’ютера програм.

Виходячи з основних завдань, було проведено детальну роботу над структурою сайту, макетами web-сторінок, навігацією по сайту. Макети заповнені вмістом, у них впроваджено графічне наповнення та додаткові компоненти, що забезпечують реалізацію функціональних можливостей сайту.

В результаті виконання роботи було створено функціональний сайт, що містить докладну інформацію про компанію, розташовану за розділами, галерею зображень, форму надсилання повідомлень, пошук сайту, опитування користувачів. Забезпечено можливість керувати сайтом за допомогою системи керування вмістом.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Лямец В.И., Успаленко В.И. Основы общей теории систем и системный анализ/ Харків: “БУРУН и К”, Київ ООО “КНТ”, 2015. – 304с.

2. К. Джамс, К. Кінг Ефективний самовчитель з креативного Web-дизайну / Київ : ‘‘ДіаСофтЮП’’, 2005. - 672 с.

3. Карасьова Е.В., І.М. Чумаченко. Photoshop CS2/ ‘‘NT Press’’, 2006. – 408 с.

4. Клименко Р. Веб-мастеринг : изучаем HTML5, CSS3, JavaScript, PHP, CMS, AJAX, SEO / 2013. – 508с.

5. Дж. Н. Роббінс Web-дизайн: довідник/ ТОВ «Видавництво «Ексмо», 2008. – 216 с.

6. Цеслів О.В. WEB-програмування : навч. посібнник / О.В. Цеслів ; М-во освіти і науки, молоді та спорту України, Нац. техн. ун-т України “Київ. політехн. ін-т”. – Київ : НТУУ “КПІ”, 2011. – 296 с.

7. Бернерс-Лі Заснування павутини = Weaving the web. The original design and ultimate destiny of the world wide web : З чого починалася і до чого прийде Всесвітня мережа / Тім Бернерс-Лі разом з Марком Фічетті; пер. з англ. А. Іщенка. – Київ : Києво-Могилянська академія, 2007. – 208с.

8. Лабберс П. HTML 5 для профессионалов = Pro HTML 5 Programming : мощные инструменты для разработки современных веб-приложений / Питер Лабберс, Брайан Олберс, Фрэнк Салим ; [пер. с англ. и ред. А.Г. Гузикевича ; предисл. Пола Айриша]. – Москва [и др.] : Вильямс, 2011. – 272 с.

9. Зандстра М. PHP. Объекты, шаблоны и методики программирования = PHP. Objekts, patterns, and practice / Мэтт Зандстра ; [пер. с англ. и ред. С.Н. Тригуб]. – 2-е изд. – Москва: Вильямс, 2010. – 478 с.

10. Шмідт Я. Нова мережа: ознаки, практики і наслідки веб 2.0 = Das Neue Nets Markmale, Praktiken und Folgen des Web 2.0 : посібник для вузів

/ Ян Шмідт ; [пер. з нім. В. Климченко ; за заг. ред. В. Іванова]. – Київ : Академія Української Преси, Центр Вільної Преси, 2013. – 283 с.

11. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript = Learning PHP, MySQL and JavaScript / Робин Никсон ; [пер. с англ. Н. Вильчинского]. – Санкт-Петербург 2013. – 496 с.

12. Куленко М.Я. Основи графічного дизайну : підручник для студентів вищих навч. закладів / Михайло Куленко; МОНУ; Київський нац. ун-т будівництва і архітектури. – 2-ге вид., виправл. та доп. – Київ : Кондор, 2007. – 492с.

13. Дронов В.А. JavaScript и AJAX в Web-дизайне / Владимир Дронов. – 2-е изд., перераб. и доп. – Санкт-Петербург : БХВ-Петербург, 2008. –724 с.

14. Клифтон Ян. Проектирование пользовательского интерфейса в Android / Мовчан Д. А. —ДМК Пресс, 2017. — 452 с.

15.

16. Макнейл П. Веб-дизайн. Книга идей веб-разработчика / П. Макнейл. —Питер, 2017. — 480 с.

17.Д. С. Макфарланд Новая большая книга CSS / Санкт-Петербур, 2018 – 720 с.

18. What is front-end development. URL: https://www.freecodecamp.org/news/front-end-developer-what-is-front-end-development-explained-in-plain-english/ (дата звернення 24.04.2022).

19. Front-end development technologies concepts. URL: https://www.altexsoft.com/blog/front-end-development-technologies-concepts/ (дата звернення 24.04.2022).

20. Beginners guide for back-end development. URL: https://www.upwork.com/resources/beginners-guide-back-end-development (дата звернення 24.04.2022).

21. Back-end architecture. URL: https://www.codecademy.com/article/back-end-architecture (дата звернення 24.04.2022).

22. Development tools for web developers. URL: https://www.geeksforgeeks.org/12-backend-development-tools-for-web-developers/ (дата звернення 24.04.2022).

23. Який back-end потрібний вашому вебсайту? URL: https://goldwebsolutions.com/uk/blog/yakij-backend-potribnij-vashomu-veb-sajtu/ (дата звернення 24.04.2022).

24. Visual Studio Code. URL: https://uk.wikipedia.org/wiki/Visual_Studio_Code (дата звернення 24.04.2022).

25. XAMPP. URL: https://uk.wikipedia.org/wiki/XAMPP (дата звернення 24.04.2022).

26. Git. URL: https://uk.wikipedia.org/wiki/Git (дата звернення 24.04.2022).

27. Atom. URL: https://uk.wikipedia.org/wiki/Atom (дата звернення 24.04.2022).

28. HTML. URL: https://uk.wikipedia.org/wiki/HTML (дата звернення 24.04.2022).

29. CSS. URL: https://uk.wikipedia.org/wiki/CSS (дата звернення 24.04.2022).

30. JavaScript. URL: https://uk.wikipedia.org/wiki/JavaScript (дата звернення 24.04.2022).

31. GitHub. URL: https://github.com/ (дата звернення 24.04.2022).