НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії_____
Кафедра комп'ютеризованих систем управління_____
Спеціальність 126 «Інформаційні системи та технології»_____
(шифр, найменування)

Освітньо-професійна програма «Інформаційні системи та технології»_____
Форма навчання денна_____

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____
«_____»_____202 __ р.

ДИПЛОМНИЙ ПРОЄКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ

"БАКАЛАВР"

Тема: Програмна онлайнова система компанії з продажів авіаквитків.

Виконавець: Ярошенко Єгор Валерійович

Керівник: Артамонов Євген Борисович

Консультанти з окремих розділів пояснювальної записки:
_____
_____
_____

Нормоконтролер: _____

Київ 2022

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра інженерії програмного забезпечення

Освітній ступінь бакалавр

Спеціальність 126 Інформаційні системи та технології

Освітньо-професійна програма Інормаційні системи та технології

ЗАТВЕРДЖУЮ

Завідувач кафедри

Зибін С.В.

"___" _____ 2021 р

ЗАВДАННЯ

на виконання дипломного проєкта студента

Ярошенко Єгора Валерійовича

1. Тема проєкту: «Програмна онлайнова система компанії з продажу авіаквитків.»
затверджена наказом ректора від «_____»_____202__ р. №_____

2. Термін виконання проєкту: з 16.05.2022 р. до 19.06.2022 р.

3. Вихідні данні до проєкту: програмний застосунок розроблений за допомогою засіба для створення, редагування та зневадження вебзастосунків Visual Studio Code.

4. Зміст пояснювальної записки:
1) Аналіз предметної області.
2) Вимоги до застосунку: Розробка інтерфейсу веб-програми для покупки авіаквитків.
3) Структура застосунку: складається з технологій Front-end розробки, таких як: HTML&CSS, JQuery, JavaScript, Bootstrap, SASS, Media-Query.
4) Прототип застосунку онлайнової системи для покупки авіаквитків.

5. Перелік слайдів презентації:
1) Тема, виконавець, керівник.
2) Вимоги до застосунку.
3) Структура засобу.

4) Інтерфейс засобу.
5) Висновок.

6. Календарний план-графік.

№ пор	Завдання	Термін виконання	Відмітка про виконання
1.	Ознайомлення з постановкою задачі та вивчення літератури Написання 1 розділу, представлення керівнику		
2.	Написання 2 розділу, представлення керівнику		
3.	Написання 3 розділу, представлення керівнику		
5.	Загальне редагування та друк пояснювальної записки, графічного матеріалу		
6.	Проходження нормо-контролю, перепліт пояснювальної записки.		
7.	Захист дипломного проекту		

7. Дата видачі завдання____ 12.05.2022 *початок проектування*
Керівник: _____Артамонов Євген Борисович.
Завдання прийняв до виконання: _____Ярошенко Єгор Валерійович
Дата:_____12.05.2022

РЕФЕРАТ

Пояснювальна записка до доплимного проєкту “Програмна онлайнова система компанії з продажів авіаквитків”.

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, ПРОГРАМУВАННЯ, ПРОГРАМНИЙ ЗАСІБ.

Об’єкт розробки – веб-застосунок компанії з продажу авіаквитків.

Мета роботи – розробити інтерфейс посадкової сторінки веб-додатка для покупки авіаквитків застосовуючи технології Front-end розробки а також різні бібліотеки технологій Front-end.

ВСТУП

У сучасному світі досить важко жити без доступних нам технологій, оскільки багато сфер життя стрімко переходять в онлайн.

Не обов'язково йти в кафе або ресторан, якщо хочеться їсти, або витрачати час стоячи в черзі за покупкою. Все це можна зробити за допомогою мобільного або комп'ютерного пристрою, скориставшись якоюсь послугою, зробивши замовлення в інтернет-магазині або замовивши їжу через доставку.

На сьогоднішній день існує велика кількість технологій, які полегшують нам життя, завдяки яким ми заощаджуємо свій час, здоров'я та нерви.

Всі інформаційні системи та пристрої мають своє програмне забезпечення і завдяки швидкому розвитку IT, значна кількість дій скоротилася і людина повністю поринула в інформаційний простір.

Сьогодні нам достатньо одного смартфона і вихід в інтернет, щоб мати доступ до важливих для людини технологій, усі можливі витвори мистецтва, музика, кіно, книги та інше.

Всі ми любимо подорожувати та відвідувати інші країни. Плануючи відпочинок, ми починаємо заздалегідь займатися такими речами як бронювати готель, шукати рейс на літак в потрібну нам дату, займатися покупкою квитків та багато іншого.

Найшвидшим транспортом для подорожі доступним нам сьогодні є літак. Існує багато сервісів та програм для покупки квитків та бронювання рейсів. Придбати квитки на літак можна за допомогою смартфона або комп'ютера, прив'язавши онлайн банківську карту.

Для створення складного та добре працюючого додатка покупки авіаквитків потрібна велика кількість фахівців, таких як дизайнер, Фронтенд розробник, Бекенд розробник, Копірайтер та багато інших.

У дипломній роботі я хочу створити власну частину інтерфейсу веб-сайту купівлі авіаквитків, тобто Фронтенд-розробку.

Головне завдання Фронтенд-розробника створити сайт або програму максимально зручною та комфортною для користувача, зменшити швидкість завантаження та зробити його інтерактивним застосовуючи нові технології.

Завдання на дипломну роботу:

1) Аналіз предметної області;

2) Опис використаних технологій для створення застосунку;

3) Розроблення програмних модулів;

РОЗДІЛ 1. Аналіз предметної області.

1.1. Динамічний вміст веб-сторінки.

HTTP та HTML.

HTTP є стандартом взаємодії, регулюючий порядок направлення запитів і набуття результатів — процесу, що відбувається між браузером, запущеним на комп'ютері фінального користувача, і веб-сервером. Завдання сервера полягає в тому, щоб прийняти запит від замовника і спробувати дати на нього змістовний результат, зазвичай передаючи йому потрібну веб-сторінку. Саме тому і застосовується термін «сервер» («обслуговуючий»). Партнером, що взаємодіє з сервером, є замовник, тому це уявлення застосовується як до браузеру, так і до комп'ютера, на якому він працює.

Між замовником і сервером може розташовуватися ряд інших пристроїв, наприклад маршрутизатори, модулі доступу, шлюзи і т. д. Вони виконують різні завдання забезпечення безпомилкового переміщення запитів і результатів між замовником і сервером. Як правило, для надсилання цієї інформації використовується Інтернет.

Зазвичай веб-сервер може обробляти відразу кілька підключень, а за відсутності зв'язку з клієнтом він перебуває у режимі очікування вхідного підключення. При надходженні запиту на підключення сервер підтверджує його отримання надсиланням результату.

Основна послідовність процесу між клієнтом та сервером:

- Вводиться до адресного рядка браузера http://server.com.
- Браузер шукає IP-адресу, відповідну доменному імені server.com
- Браузер надсилає запит на головну сторінку server.com.
- Запит проходить через Інтернет і надходить на веб-сервер server.com.

- Веб-сервер, який отримав запит, шукає веб-сторінку на своєму жорсткому диску.
- Веб-сервер, який отримав запит, шукає веб-сторінку на своєму жорсткому диску.
- Браузер відображає веб-сторінку

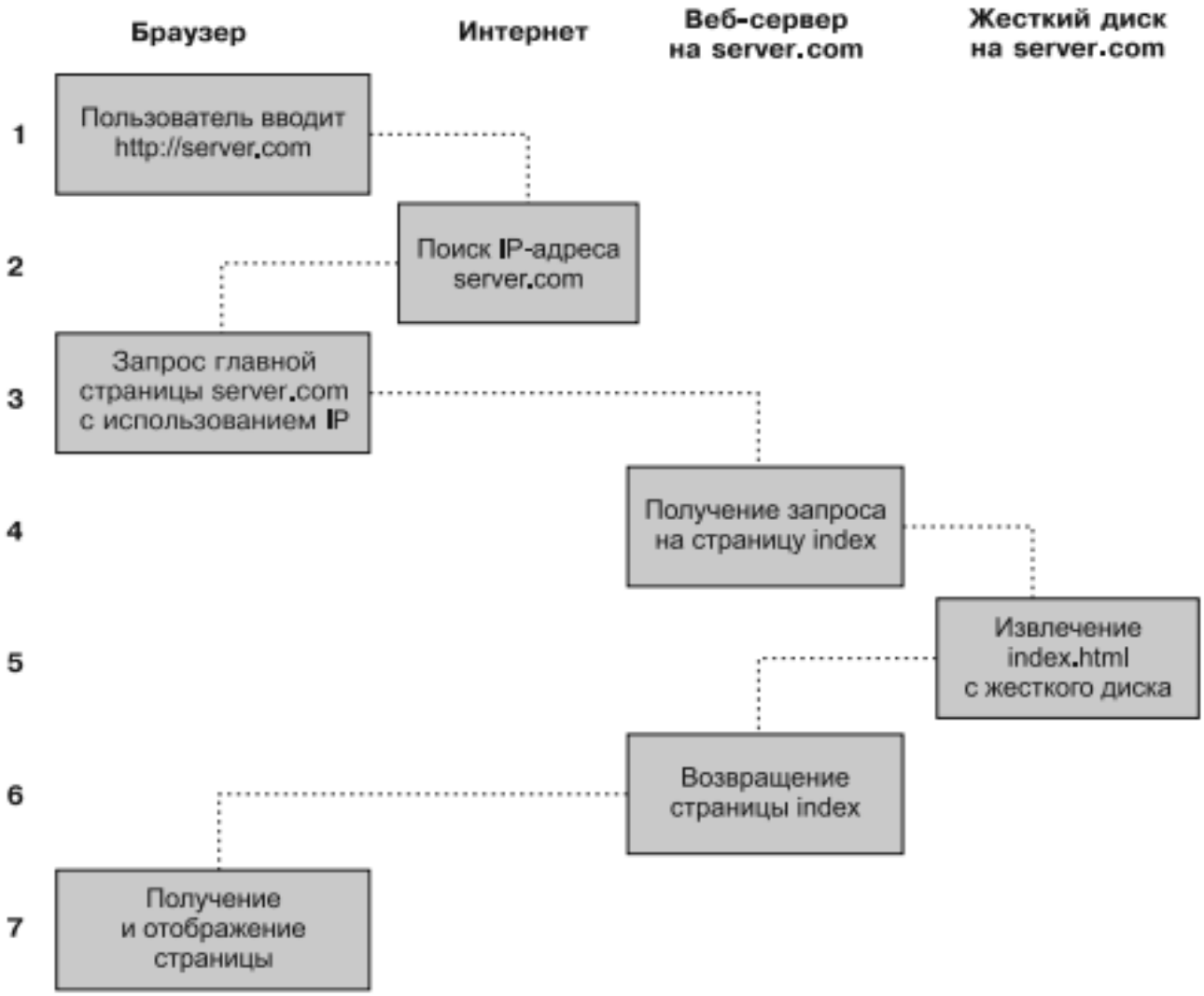


Рис. 1.1 – процес між клієнтом та сервером.

При передачі типової веб-сторінки цей процес здійснюється для кожного об'єкта, що є на ній. На кроці два браузер шукає IP-адресу, що належить доменному імені server.com. Будь-яка машина, підключена до Інтернету, включаючи і ваш комп'ютер, має свою IP-адресу. Але, як правило, доступ до веб-серверів здійснюється за іменами, такими як google.com. Вам, мабуть, відомо, що браузер звертається до допоміжної інтернет-служби, так званої служби доменних імен (DNS), для того щоб виявити пов'язану з сервером IP-

адресу, а потім скористатися ним для зв'язку з комп'ютером . При передачі динамічних веб-сторінок процедура складається з більшої кількості дій, від того, що до неї можуть залучатися як PHP, так і MySQL.

- Ви вводите адресний рядок браузера http://server.com.
- Ваш браузер шукає IP-адресу, що відповідає домену імені server.com.
- Браузер надсилає запит на домашню сторінку server.com.
- Запит проходить через мережу і надходить на веб-сервер server.com.
- Веб-сервер, який отримав запит, шукає веб-сторінку на жорсткому диску.
- Коли головна сторінка розміщена в пам'яті, веб-сервер помічає, що вона представлена файлом, що включає PHP-сценарії, і передає сторінку інтерпретатора PHP.
- Інтерпретатор PHP виконує код PHP.
- Деякі фрагменти коду PHP містять MySQL-інструкції, які інтерпретатор PHP, своєю чергою, передає процесору бази даних MySQL.
- База даних MySQL повертає результати виконання інструкції до інтерпретатора PHP.
- Інтерпретатор PHP повертає веб-серверу результати виконання PHP коду, а також результати, отримані з бази даних MySQL.
- Веб-сервер повертає сторінку клієнту, який видав запит, який відображає цю сторінку на екрані.

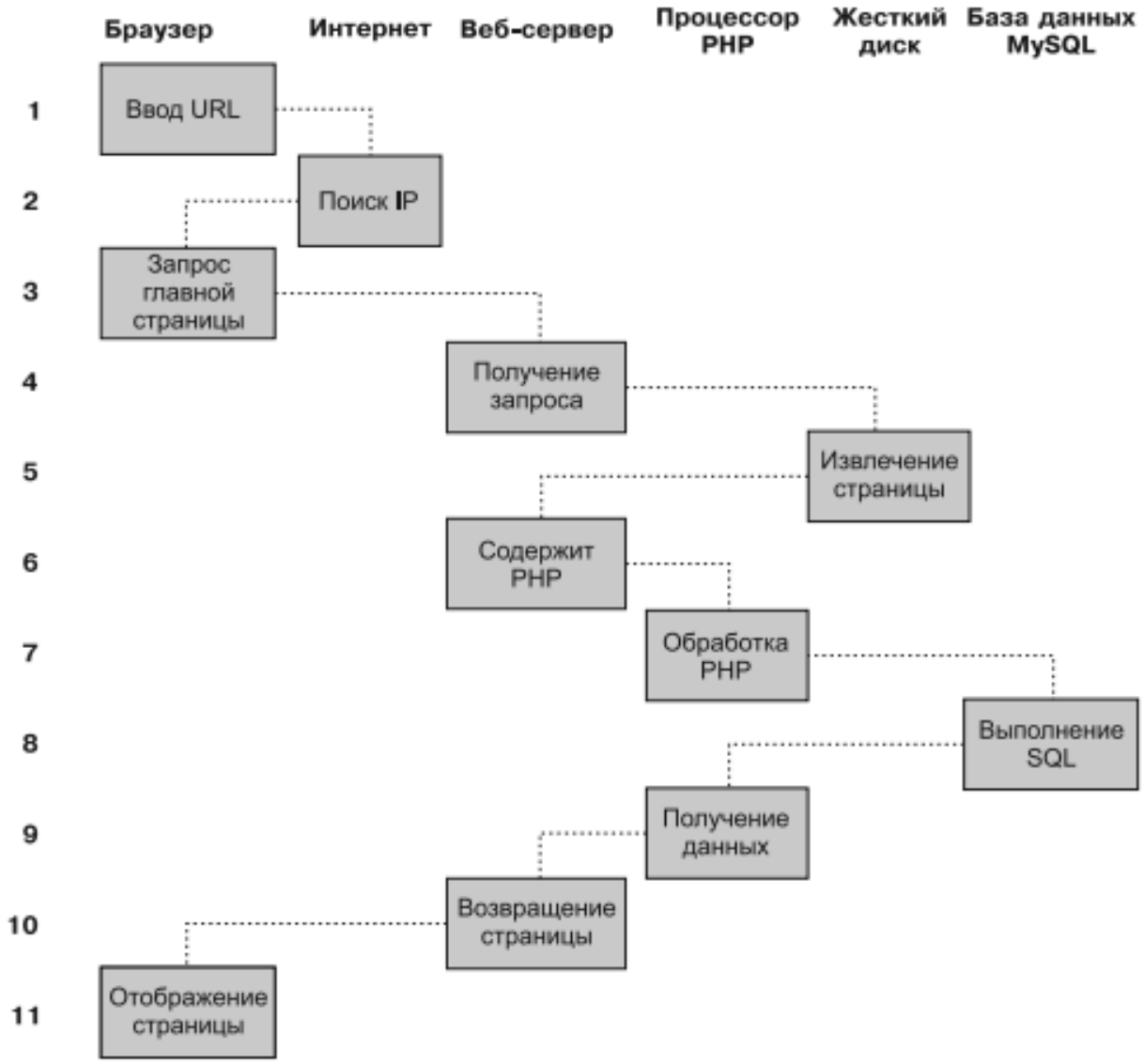


Рис.1.2 – динамічна послідовність процесу між клієнтом та сервером.

1.2. Історія та огляд веб-застосунків.

Глобальна павутина (WWW) була зроблена в 1989 британським комп'ютерником з CERN Тімом Бернерсом-Лі. 30 квітня 1993 року ЦЕРН оголосив, що Глобальне павутиння буде безкоштовним для всіх, що сприяло великому зростанню мережі. До появи протоколу передачі гіпертексту (HTTP) для отримання окремих файлів з сервера застосовувалися інші протоколи, такі як протокол передачі файлів і протокол gopher . Ці протоколи пропонують примітивну схему каталогів .за якими переміщається користувач і де він вибирає файли для завантаження. Документи найчастіше представлялися як примітивних текстових файлів без форматування чи кодувалися у форматах текстового процесора.

Веб-сайти можна використовувати по-різному: індивідуальний сайт, корпоративний сайт організації, урядовий сайт, сайт організації і т. д. Веб-

сайти можуть бути роботою окремих осіб, підприємств або інших організацій і зазвичай призначені на певну тему чи мети. Будь-який сайт може містити посилання на будь-який інший сайт, тому відмінність між окремими сайтами у сприйнятті користувача може бути розмито.

Деякі веб-сайти вимагають реєстрації користувача або підписки для доступу до вмісту. Приклади веб-сайтів з підпискою включають безліч бізнес-сайтів, новинних веб-сайтів, веб-сайтів академічних журналів, ігрових веб-сайтів, веб-сайтів для обміну файлами, дощок оголошень, електронної пошти У мережі Інтернет, веб-сайтів громадських мереж , веб-сайтів, що надають дані про фондовий ринок в режимі реального часу , а також веб-сайти, що надають різні інші послуги.

У той час як «веб-сайт» був початковим написанням (зрідка «веб-сайт» пишеться з великої літери, тому що «веб-сайт» є власним ім'ям стосовно всесвітньої павутини), даний варіант став застосовуватися рідко, і «веб-сайт» став стандартним написанням.

Статичний сайт:

Статичний веб-сайт — це той, веб-сторінки якого зберігаються на сервері у форматі, який надсилається до клієнтського веб-браузера. В основному він закодований мовою гіпертекстової розмітки (HTML); Каскадні таблиці жанрів (CSS) використовуються управління зовнішнім виглядом крім базового HTML. Зображення зазвичай використовуються для бажаного зовнішнього вигляду і як частина основного контенту. Аудіо або відео також можуть вважатися «статичним» контентом, якщо вони відтворюються автоматично або зазвичай не є інтерактивними.

Цей тип веб-сайту зазвичай відображає ту саму інформацію для всіх відвідувачів. Аналогічно роздачі друкованої брошури замовникам або клієнтам, статичний сайт зазвичай надає несуперечливу стандартну інформацію протягом тривалого часу. Правда, власник веб-сайту може періодично вносити оновлення, це ручний процес редагування тексту, фотографій та іншого контенту, який може вимагати базових навичок дизайну веб-сайту та програмного забезпечення. Примітивні форми або маркетингові приклади веб-сайтів, такі як типовий сайт, п'ятисторінковий сайт. Це може включати інформацію про організацію, її продукти та служби у вигляді тексту, фотографій, анімації, аудіо/відео та навігаційних меню.

Статичні веб-сайти можуть використовувати серверні включення (SSI) для комфорту редагування, наприклад, для загального застосування

загального рядка меню на багатьох сторінках. Від того, що ставлення сайту до читача все ще нерухоме, він не вважається динамічним сайтом.

Динамічний веб-сайт:

Динамічний сайт - це той, який часто і механічно змінюється або налаштовується. Динамічні сторінки на стороні сервера генеруються на льоту комп'ютерним кодом, що створює HTML (CSS відповідає за зовнішній вигляд і, таким чином, є статичними файлами). Існує широкий спектр програмних систем, таких як CGI, Java Servlets і Java Server Pages (JSP), Active Server Pages та ColdFusion (CFML), які доступні для виробництва динамічних веб-систем та динамічних сайтів. Різні фреймворки веб-додатків та системи веб-шаблонів доступні для мов програмування універсального призначення, таких як Perl, PHP, Python і Ruby, щоб спростити та прискорити реалізацію важких динамічних веб-сайтів.

Сайт може відображати нинішній стан діалогу між користувачами, стежити за обстановкою, що змінюється, або надавати інформацію якимось чином персоналізованим під вимоги певного користувача. Наприклад, коли запитується основна сторінка новинного сайту, код, що працює на веб-сервері, може комбінувати збережені фрагменти HTML з новинами, отриманими з бази даних або іншого сайту через RSS, щоб створити сторінку, що містить найновішу інформацію. Динамічні сайти можуть бути інтерактивними завдяки застосуванню HTML-форм , збереженню та зчитуванню файлів cookie браузера .або шляхом створення серії сторінок, що відображають попередню історію кліків. Ще одним прикладом динамічного знаходження є випадок, коли роздрібний сайт з базою даних медіапродуктів дозволяє користувачеві вводити пошуковий запит, наприклад, за ключовим словом "Бітлз". В результаті вміст веб-сторінки мимоволі змінить свій колишній вигляд, а потім відобразить список продуктів Beatles, таких як компакт-диски, DVD-диски та книги. Динамічний HTML використовує JavaScript, щоб вказати веб-браузеру, як інтерактивно змінювати вміст сторінки. Один з методів змоделювати певний тип динамічного веб-сайту, цураючись при цьому втрати ефективності при запуску динамічного механізму для будь-якого користувача або для будь-якого з'єднання, це періодично механічно регенерувати велику серію статичних сторінок.

Категорії:

Веб-сайти можна розділити на дві великі категорії - статичні та інтерактивні. Інтерактивні сайти є частиною спільноти сайтів Web 2.0 та

забезпечують взаємодію між власником веб-сайту та його відвідувачами чи користувачами. Статичні сайти надають або збирають інформацію, але не дозволяють безпосередньо взаємодіяти з аудиторією чи користувачами. Деякі сайти носять інформаційний характер, створені ентузіастами або призначені для особистого використання або розваги. Багато веб-сайтів тяжіться заробляти гроші, використовуючи одну або кілька бізнес-моделей, у тому числі:

- Розміщення цікавого контенту та продаж контекстної реклами або через прямий продаж, або через рекламну мережу.
- Електронна комерція: товари або послуги купуються безпосередньо через веб-сайт.
- Рекламні продукти або послуги, доступні у звичайному бізнесі.

1.3. Етапи розробки веб-застосунку.

На сьогоднішній день існує кілька етапів розробки сайту:

- Проектування сайту або веб-додатки (збір та аналіз вимог, розробка технічного завдання, проектування інтерфейсів);
- Розробка креативної концепції сайту;
- створення дизайн-концепції сайту;
- створення макетів сторінок;
- створення мультимедіа-об'єктів;
- Верстка сторінок та шаблонів;
- Програмування (розробка функціональних інструментів) або інтеграція у систему управління вмістом (CMS);
- Оптимізація та розміщення матеріалів сайту;
- Тестування та внесення коригувань;
- Публікація проекту на хостингу;
- Обслуговування сайту або його програмної основи.

Створення технічного завдання (ТЗ)

Складанням технічного завдання можуть займатися проектувальник, аналітик, веб-архітектор, адміністратор проектного плану разом чи окремо. (У разі, коли сайт розробляється фрілансером, технічне завдання може бути складено з боку організації замовника).

Робота із замовником починається із заповнення брифа, в якому замовник викладає свої побажання щодо візуального представлення та структури веб-сайту, вказує на помилки у застарілій версії сайту, наводить приклади сайтів суперників. Виходячи з брифу, адміністратор складає технічне завдання, розглядаючи можливості програмних та дизайнерських засобів. Етап закінчується після заяви технічного завдання замовником. Важливо відразу відзначити, що етапи проектування сайтів залежать від багатьох факторів, таких як обсяг сайту, функціональність, завдання, які має виконувати майбутнє джерело та багато іншого. Втім, є кілька етапів, які в обов'язковому порядку присутні в плануванні будь-якого проектного плану. У результаті документі, де описано технічне завдання, можуть бути такі основні розділи:

- Мета та призначення сайту.
- Аудиторія сайту.
- Технічні характеристики.
- Зміст сайту (структура сайту з докладним описом елементів та функцій кожної сторінки).
- Інтерактивні елементи та послуги (форми зворотного зв'язку, пошук на сайті, форум на сайті).
- Форми (надсилання на пошту, підписки на розсилку, зворотного зв'язку).
- Система керування вмістом (контентом).
- Вимоги до матеріалів.
- Перенесення на хостинг.

Дизайн основних та типових сторінок сайту:

Починається робота з виробництва дизайну, зазвичай, у графічному редакторі. Дизайнер створює один або кілька варіантів дизайну, відповідно до технічного завдання.

При цьому окремо створюється дизайн основної сторінки та дизайни типових сторінок (наприклад: статті, новини, каталог продукції). Власне «дизайн сторінки» є графічним файлом, листковим малюнком, що складається з особливо дрібних картинок-шарів елементів універсального малюнка. При цьому дизайнер повинен розглядати обмеження стандартів HTML (не створювати дизайн, який після цього не зможе бути реалізований стандартними засобами HTML). Виняток становить Flash-дизайн.

Кількість ескізів та порядок їх надання обумовлюється проект-адміністратором. Також менеджер проектного плану здійснює контроль за термінами. У величезних веб-студіях бере участь арт-директор, який контролює якість графіки. Етап також закінчується заявою ескізу замовником.

HTML-верстка

Схвалений дизайн передається HTML-верстальнику, який "нарізає" графічну картинку на окремі малюнки, з яких пізніше складає HTML-сторінку. У результаті створюється код, який можна переглядати за допомогою браузера. А типові сторінки пізніше будуть застосовуватись як зразки.

Програмування:

Далі готові HTML-файли передають програмісту. Програмування веб-сайту може здійснюватися як «з нуля», і за допомогою CMS — системи управління сайтом. Веб-розробники часто називають CMS "движком".

У випадку з CMS потрібно сказати, що сама «CMS» в певному сенсі це готовий сайт, що складається із частин, що замінюються. «Програміст» — у разі правильно назвати його просто експертом з CMS — повинен замінити звичайний зразок, поставлявшийся з CMS, на справжній зразок. Цей справжній зразок він і повинен зробити за допомогою початкового веб-дизайну.

У разі програмування веб-сайту експерту призначаються контрольні точки термінів.

Завершальним етапом розробки веб-сайту є тестування.

Процес тестування може включати найрізноманітніші перевірки: вид сторінки зі збільшеними шрифтами, при різних розмірах вікна браузера, при відсутності флеш-плеєра і багато інших.

Знайдені помилки відправляються на виправлення, доки не будуть усунені. Строки контролює адміністратор проектного плану. Також на цьому етапі залучають до роботи дизайнера, щоб він провів авторський нагляд.

Розміщення веб-сайту в Інтернеті.

Файли сайту розміщують на сервері провайдера (хостингу) і виконують необхідні налаштування. На цьому етапі сайт поки що закритий для відвідувачів.

Наповнення контентом та публікація

Сайт наповнюють вмістом (контентом) - текстами, зображеннями, файлами для скачування і так далі. Іноді тексти складаються експертом студії, рідко контентом займається відповідальна особа з боку замовника. Це вирішується на етапі складання технічного завдання.

У випадку, якщо контент складається представником студії, це відбувається і затверджується паралельно з іншими етапами проектного плану. На будь-якій сторінці є текстові блоки, вони можуть бути типовими (звичайні) і не типовими.

Як правило, нетиповий текстовий блок розміщений на сторінці чотириста чотири. До стандартних текстових блоків відносяться:

- header сайту;
- footer сайту;
- навігаційний ланцюжок.

Основні елементи текстового блоку:

- заголовки 1, 2 та 3 рівнів;
- зображення;
- зображення у тексті;
- галереї;
- текст;
- блок тексту, що містить заголовок;
- нумеровані та ненумеровані списки;
- таблиці;
- файли для скачування;
- відео.

Внутрішня та зовнішня SEO-оптимізація

Внутрішня:

Пов'язана з деякими змінами веб-сайту. SEO-оптимізація починається з визначення семантичного ядра. Тут визначаються такі ключові слова, які залучать особливо зацікавлених відвідувачів, якими виграти конкуренцію простіше. Після цього слова вносяться на сайт. Тексти, посилання, інші теги адаптуються так, щоб пошукові системи могли їх успішно шукати за ключовими словами.

Зовнішня:

Зводиться, зазвичай, до побудови структури вхідних посилань. Це, власне, і є розкрутка сайту. До створення веб-сайту зовнішня SEO-оптимізація не має відношення. SEO-оптимізація систематизується на «білу» та «чорну» (таку, після якої сайт за два тижні потрапляє в топ, а потім у бан пошуковців). Справжня, «біла» SEO-оптимізація, це трудомісткий і довгий процес, ціна якого може в кілька разів перевищувати витрати на реалізацію сайту.

РОЗДІЛ 2. Опис використаних технологій для створення застосунку

(програмна онлайнова система з продажу авіаквитків) Front-end частини.

2.1. Середовище розробки (Visual Studio Code).

Visual Studio Code (відомий як VS Code) - це безкоштовний текстовий редактор з відкритим вихідним кодом від Microsoft. VS Code доступний для Windows, Linux та MacOS. Хоча редактор відносно легкий, він включає кілька потужних функцій, які зробили VS Code одним з найпопулярніших інструментів середовища розробки останнім часом.

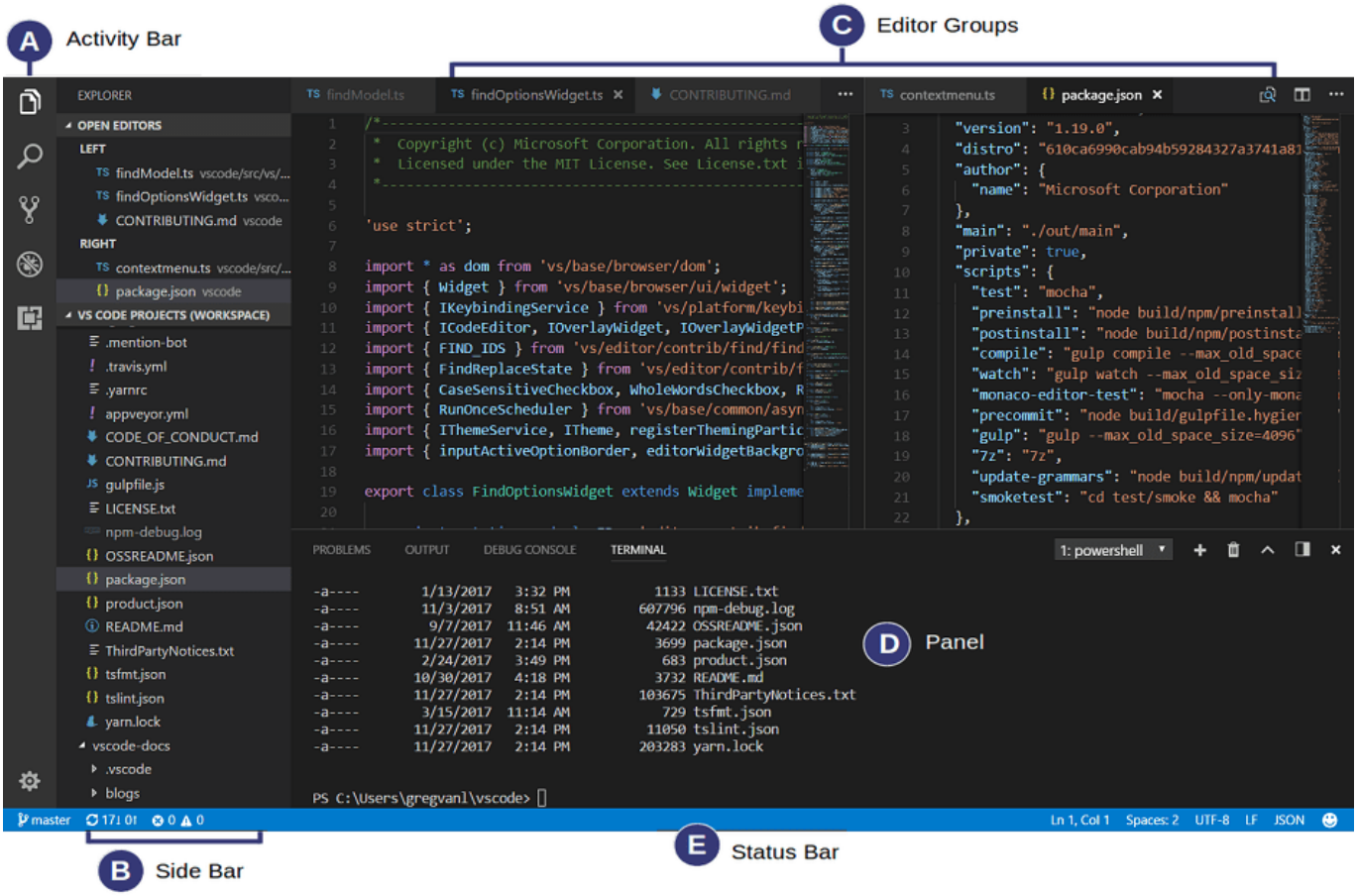
Фунцкії:

VS Code підтримує широкий спектр мов програмування від Java, C++ та Python до CSS, Go та Dockerfile. Більше того, VS Code дозволяє додавати і навіть створювати нові розширення, включаючи лінтери коду, налагоджувачі та підтримку хмарних та веб-розробок.

Інтерфейс користувача VS Code забезпечує більшу взаємодію в порівнянні з іншими текстовими редакторами. Для спрощення взаємодії з користувачем VS Code поділено на п'ять основних областей:

- Панель активності
- Бічна панель
- Групи редакторів
- Панель
- Рядок стану

На зображенні нижче показано, як відображаються ці області:



2.2. HTML & CSS.

HTML - мова гіпертекстової розмітки, надає контенту схему і значення, визначаючи цей контент, наприклад, як заголовки, абзаци чи зображення. CSS або каскадні таблиці жанрів - це мова уявлення, створена для стилізації зовнішнього вигляду контенту із застосуванням, наприклад, шрифтів або кольорів.

Дві мови - HTML і CSS - самостійні один від одного і повинні залишатися такими. CSS не повинен бути написаний всередині HTML-документу та навпаки. Як правило, HTML постійно представляє контент, а CSS постійно представляє зовнішній вигляд цього контенту.

Розуміння загальних термінів HTML

Елементи

Елементи – це позначки, які визначають структуру та вміст об'єктів на сторінці. Деякі з найчастіше використовуваних елементів включають кілька рівнів заголовків (позначених <h1>наскрізні <h6>елементи) і абзаців (позначених як <p>елемент); Список включає елементи <a>, , , , і і багато іншого.

Елементи ідентифікуються за допомогою кутових дужок менше та більше < >, що оточують ім'я елемента. Таким чином, елемент виглядатиме так:

```
1  <div>
```

Теги

Застосування кутових дужок поменше і більше, що оточують елемент, створює так званий тег . Теги найчастіше зустрічаються як пар відкривають і закривають тегов.

Тег, що відкриває, відзначає початок елемента. Він складається із знаку «менше», за яким слідує ім'я елемента, а після цього закінчується знаком «більше»; Наприклад, .

Закриває тег відзначає кінець елемента. Він складається із знаку «менше», за яким слідує коса риса та ім'я елемента, а після цього закінчується знаком «більше»; наприклад, .

Вміст, що знаходиться між тегами, що відкриває і закриває, є вмістом цього елемента. Якірне посилання, наприклад, матиме тег <a>і відкриває тег . Те, що знаходиться між цими двома тегами, і буде вмістом якірного посилання.

Отже, якірні теги виглядатимуть приблизно так:

```
<div>...</div>
```

Атрибути:

Атрибути - це властивості, які використовуються для надання додаткової інформації про елемент. Найбільш поширені атрибути включають ID атрибут, який ідентифікує елемент; атрибут class, що класифікує елемент; атрибут src, що вказує джерело контенту, що вбудовується; та hrefатрибут, який надає гіперпосилання на пов'язаний ресурс.

Атрибути визначаються у тезі, що відкриває, після імені елемента. Зазвичай атрибути включають ім'я та значення. Формат цих атрибутів складається з імені атрибута, за яким слідує знак рівності, а потім значення атрибута в лапках. Наприклад, <a>елемент, що включає hrefатрибут, буде виглядати так:

```
<a href="http://yehoryaroshenko/">NAU</a>
```

Структура HTML-документа:

Всі HTML-документи мають неодмінну схему, яка включає наступні оголошення та елементи: <!DOCTYPE html>, <html>, <head>і <body>.

Оголошення типу документа або <!DOCTYPE html> повідомляє веб-браузери про те, яка версія HTML застосовується, і розміщується на самому початку документа HTML. Від того, що ми будемо застосовувати останню версію HTML, наше оголошення типу документа буде просто <!DOCTYPE html>. Після оголошення типу документа <html>елемент означає початок документа.

Всередині <html>елемента <head>елемент ідентифікує верхню частину документа, включаючи будь-які метадані (що супроводжують інформацію про сторінку). Вміст <head>елемента не відображається на

веб-сторінці. Натомість він може включати найменування документа (яке відображається в рядку заголовка вікна браузера), посилання на будь-які зовнішні файли або будь-які інші придатні метадані.

Цілий видимий контент на веб-сторінці буде всередині <body>елемента. Розбивка нормальної структури HTML-документа виглядає так:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is a web page.</p>
  </body>
</html>
```

Демонстрація структури HTML-документу:

Hello World

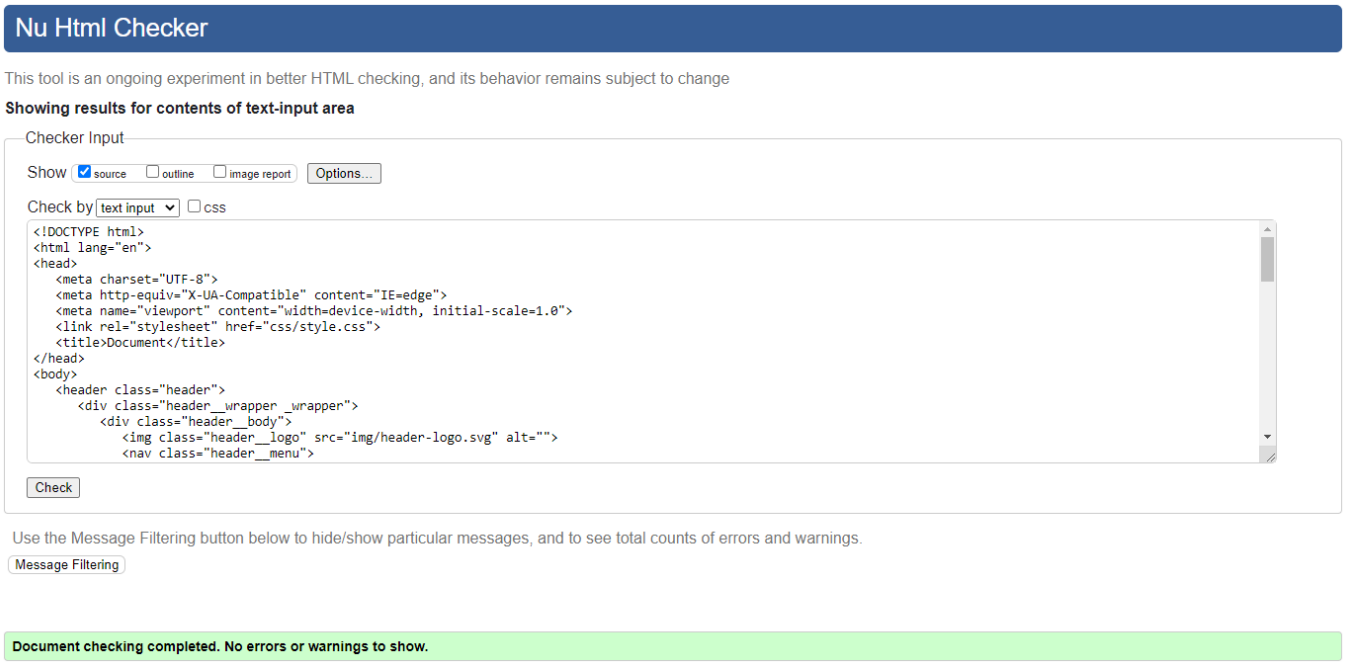
This is a web page.

Перевірка коду:

Як би ми не були обережні при написанні коду, ми неминуче робитимемо помилки. На щастя, при написанні HTML та CSS ми маємо валідатори для перевірки нашої роботи. W3C створив засоби перевірки HTML і CSS, які будуть сканувати код на наявність помилок. Перевірка нашого коду не тільки допомагає правильно відображати його у всіх

браузерах, але й допомагає навчити нас найкращим способом написання коду.

W3C-валідатор:



CSS – це мова, яку ми використовуємо для оформлення веб-сторінки.

- CSS розшифровується як каскадні таблиці стилів.
- CSS описує, як HTML-елементи повинні відображатись на екрані, папері або інших носіях.
- CSS заощаджує багато роботи. Він може одночасно керувати макетом кількох веб-сторінок.
- Зовнішні таблиці стилів зберігаються у файлах CSS.

Селектори:

Коли елементи додаються до веб-сторінки, вони можуть бути оформлені за допомогою CSS. Селектор показує , який саме елемент або елементи в нашому HTML є цільовими і до яких використовуються стилі (такі як колір, розмір та стан). Селектори можуть включати комбінацію різних кваліфікаторів для вибору унікальних елементів, все залежить від того, наскільки певними ми хочемо бути. Наприклад, ми можемо захотіти віддати перевагу будь-якому абзацу на сторінці або віддати перевагу тільки одному певному абзацу на сторінці.

Селектори зазвичай орієнтовані значення ознаки, наприклад значення id або class, чи тип елемента, наприклад <h1>або <p>.

У CSS за селекторами слідують фігурні дужки, які охоплюють жанри, що застосовуються до обраного елементу. Селектор тут націлений на всі елементи.

```
button {
    ...
    ...
    ...
}
```

Характеристики:

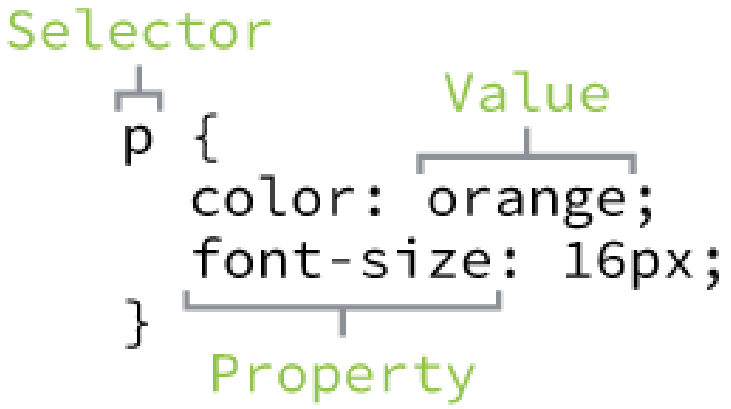
Після вибору елемента якість визначає стилі, які застосовуватимуться до цього елемента. Імена властивостей розташовуються після селектора у фігурних дужках {} і природно перед двокрапкою:.

Ми можемо використовувати безліч властивостей, таких як background, color, font-size, height і width, і часто додаються нові властивості. У подальшому коді ми визначаємо властивості color і font-size для використання до всіх елементів.

```
button {
    font-size: 18px;
    color: black;
}
```

CSS наш комплект правил починається з селектора, за яким відразу ж йдуть фігурні дужки. У цих фігурних дужках знаходяться оголошення, що складаються з пар властивостей та значень. Будь-яке оголошення починається з якості, за яким слідує двокрапка, значення якості і, нарешті, точка з комою.

Загальновизнаною практикою є виділення пар властивостей та значень у фігурних дужках. Як і у випадку з HTML, ці відступи допомагають систематизувати наш код і зробити його розбірливим.



2.3. Методологія БЕМ та препроцесори.

БЕМ - це зовнішній спосіб іменування для організації та найменування класів CSS. Методологія Block, Element, Modifier - це популярна угода про імена класів HTML і CSS. Це допомагає писати чистий CSS, дотримуючись деяких простих правил.

Проекти будь-якого розміру з CSS можуть отримати вигоду з середовища БЕМ, якщо стилі не пишуться безпосередньо всередині вже організованих файлів JavaScript і не використовуються стилізовані компоненти або щось подібне. Щоб доповнити BEM, увімкніть деякі частини SMACSS, які діють як посібник із стилю для угод про імена. БЕМ легко читається та впорядковує стилі, навіть коли проекти дуже великі.

Назва класів ніколи не було легкою частиною CSS. Стороннім розробникам важко зрозуміти, що роблять зазначені розробником класи. Угоди про імена можуть швидко вийти з-під контролю при багаторазовому рефакторинг коду і при роботі в команді.

Деякі з найбільш дратівливих проблем, пов'язаних з ім'ям CSS, коли:

- Випадкове перевизначення класів.
- Важко читати як CSS, і HTML, немає чіткої області впливу і структури в іменуванні.
- Нові члени команди витрачають час вивчення чи спроби знайти вже створені класи.
- Код складно оновлювати, тому що неясно, які аспекти є глобальними, а які належать локальному компоненту.
- Незрозуміло, як поводитися з вкладеністю

БЕМ складається з трьох основних частин:

- **Блок** , який містить усі (елементи) всередині та діє як область.
- **Елемент**, що діє як певна частина компонента.
- **Модифікатор**, який додає додаткові стилі до певного елемента (елементів).

Як показано у фрагменті коду один блок .head містить всі два елементи з іменами .head _ _ eye. Ім'я елемента починається з імені блоку, за яким слідують два символи підкреслення, а після цього йде саме ім'я елемента. Ім'я елемента є просто ім'ям елемента і може мати підкреслення. При застосуванні кількох слів використовуйте «-» дефіс-мінус, щоб розділити їх. Ім'я модифікатора має ім'я блоку на самому початку, два символи підкреслення, ім'я елемента (так само як ім'я елемента), а після цього два дефіси-мінус з ім'ям модифікатора.

```
<div class="head">
  <div class="head__eye head__eye--left"></div>
  <div class="head__eye head__eye--right"></div>
</div>
```

Scss це препроцесор. Це просто програма, яка обробляє вхідні дані та робить висновок, а потім цей висновок використовуватиметься в іншій програмі.

Sass має різні синтаксиси, тепер є два популярних, званих SCSS і SAAS, тому у нас є SCSS, який розшифровується як Scssy CSS, і це надмножина синтаксису CSS3, а SCSS більше схожий на посередника

між Scss та CSS, тоді у нас є справжній Scss код, тому Scss буде просто повноцінним файлом Scss, який буде скомпілюваний у CSS.

Як працює SCSS?

Файл Scss , що містить купу змінних і використовує безліч інших функцій, які використовує Scss звідти, він проходитиме через процесор або компілятор, і цей компілятор буде витрачати CSS, який браузер зможе прочитати, так що все це відбувається за лаштунками, і Scss необхідно скомпілювати CSS, тому що браузер насправді не може читати Scss, тому їх потрібно перетворити назад в CSS.

Переваги використання SCSS:

- SCSS більш виразний

Ми можемо стиснути кілька рядків коду в синтаксисі SASS до меншої кількості рядків SCSS. У SCSS стандартні рядки також можуть бути стиснуті, коли я роблю щось складне, і може бути знову розширено для довідки.

- SCSS дозволяє користувачеві краще писати вбудовану документацію

SASS гнучкий з коментарями, але будь-який хороший розробник віддасть перевагу вбудованій документації, доступній в SCSS. Вбудована документація робить рядки коду зрозумілими.

- Це сприяє правильній вкладеності правил.

Якщо ви використовуєте оператор комою на високому рівні, це збільшує розмір остаточного файлу CSS. Це може призвести до того, що код буде дуже складно виконувати перевизначення правил.

- Інтеграція існуючих інструментів CSS та кодової бази CSS набагато простіше.

Підсвічування синтаксису інструменту CSS, що широко використовується, і підтримується в SCSS. SCSS дозволяє використовувати існуючий код та допомагає покращити його внутрішню структуру без зміни зовнішньої поведінки коду.

Приклад використання SCSS:

```scss
/* Визначити стандартні змінні та значення для веб-сайту */
$bgcolor: lightblue;
$textcolor: darkblue;
$fontsize: 18px;

/* Використовуйте змінні */
body {
  background-color: $bgcolor;
  color: $textcolor;
  font-size: $fontsize;
}
```

Приклад на моєму проекті:

```scss
.header {
    position: fixed;
    width: 100%;
    top: 0;
    left: 0;
    z-index: 50;
    background-color: rgba(255, 255, 255, 0.9);
    &:before {
        content: '';
        position: absolute;
        top: 0;
        left: 0;
        width: 100%;
        height: 100%;
    }
    &__body {
        position: relative;
        z-index: 2;
        display: flex;
        justify-content: space-between;
        align-items: center;
        height: 110px;
    }
}
```

2.4. Bootstrap.

Bootstrap — це безкоштовний CSS-фреймворк із відкритим вихідним кодом, призначений для адаптивної веб-розробки з орієнтацією на мобільні пристрої. Він містить HTML, CSS та шаблони дизайну на основі JavaScript для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу.

Bootstrap - це бібліотека HTML, CSS і JS, спрямована на полегшення розробки інформаційних веб-сторінок (на відміну від веб-додатків). Основна мета додавання його до веб-плану — застосувати до цього плану вибрані Bootstrap кольори, розмір, шрифт та макет. Таким чином, основним фактором є те, чи виявлять відповідальні розробники ці варіанти на свій смак. Після додавання до проекту Bootstrap надає базові визначення жанрів всім елементів HTML .

Підсумком є одноманітний зовнішній вигляд текстів, таблиць та елементів форм у всіх веб-браузерах. Крім того, розробники можуть скористатися класами CSS, визначеними в Bootstrap, для подальшого налаштування зовнішнього вигляду свого вмісту. Наприклад, Bootstrap передбачив ясні та темні таблиці, заголовки сторінок, більш помітні лапки та текст із виділенням.

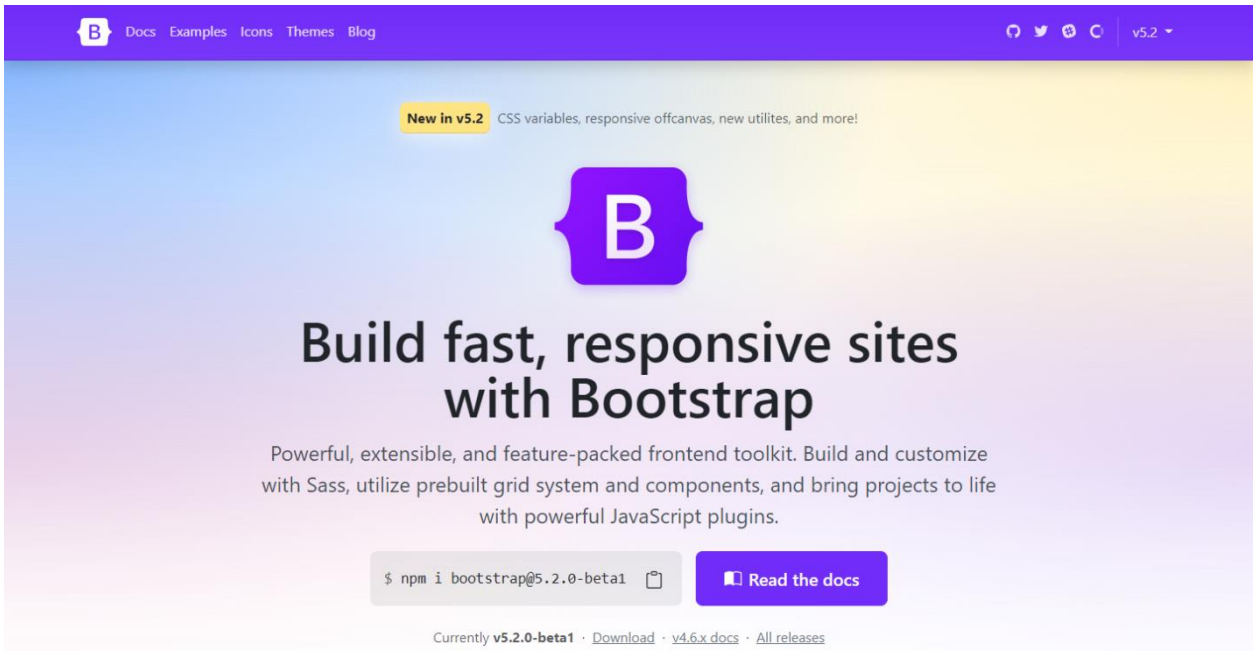
Bootstrap також поставляється з кількома компонентами JavaScript у вигляді плагінів jQuery. Вони надають додаткові елементи інтерфейсу користувача, такі як діалогові вікна, спливаючі підказки і каруселі. Кожен компонент Bootstrap складається з структури HTML, оголошень CSS і, у деяких випадках, супутнього коду JavaScript. Вони також розширюють функціональність деяких існуючих елементів інтерфейсу, включаючи, наприклад, автозаповнення функцію для полів введення.

Особливо помітними компонентами Bootstrap є компоненти макету, оскільки вони впливають всю веб-сторінку. Базовий компонент макету називається «Контейнер», тому що в нього міститься будь-який інший елемент сторінки. Розробники можуть вибирати між контейнером фіксованої ширини і контейнером ширини, що змінюється. Коли остаточний постійно заповнює ширину веб-сторінки, перший використовує одну з п'яти визначених фіксованих ширин, залежно від розміру екрана, що показує сторінку:

- Менш 576 пікселів
- 576–768 пікселів
- 768-992 пікселів
- 992–1200 пікселів

- Більше 1200 пікселів

Таким чином виглядає офіційний сайт Bootstrap, на якому можна завантажити всі необхідні компоненти для розробки веб-програми:



2.5. JavaScript.

Коли справа доходить до розробки інтерфейсу, є три основні компоненти: HTML, CSS та JavaScript. Кожен з них має вирішальне значення для створення веб-сторінки такою, якою вона є. HTML – це структура та вміст сайту, CSS (каскадні таблиці стилів) робить його красивим, і, нарешті, JavaScript – це те, що забезпечує його інтерактивність.

JavaScript - дуже потужний інструмент, який може багато зробити для веб-сайту. По-перше, він підтримує загальну інтерактивність сайту. JavaScript дозволяє створювати багаті компоненти інтерфейсу користувача, такі як слайдери зображень, спливаючі вікна, мегаменю навігації по сайту, перевірки форм, вкладки, акордеони та багато іншого.

Він також може виконувати більш тонкі операції. Наприклад, ви можете клацнути прапорець у формі, і, залежно від вибраного прапорця, він завантажить спливаюче вікно, яке поставить вам інше питання. Це дає сайту додаткову функціональність, яка інакше недоступна лише за допомогою HTML і CSS. JavaScript дозволяє веб-сторінкам реагувати на дії користувачів та динамічно оновлювати себе, і все це без необхідності перезавантаження сторінки для зміни її зовнішнього вигляду.

AJAX:

AJAX означає асинхронний JavaScript та XML. Це в основному дозволяє веб-сторінці взаємодіяти з веб-сервером у фоновому режимі без перезавантаження сторінки.

Візьмемо, наприклад, перелік подій на веб-сторінці. За замовчуванням ці події сортуються за місяцями, останніми, а потім класифікуються за типом. При зміні фільтра «ВСЕ» на «Спільний» у фоновому режимі запускається операція, і сторінка частково перезавантажується з новими елементами, що були запрошені. Сторінка не мерехтить під час перезавантаження, і ми залишаємося на місці, що робить роботу в Інтернеті приємнішою.

Це також можна використовувати у формах для надсилання даних на сервер для збереження; знову ж таки, щоб сторінка не перезавантажувалася. AJAX дозволяє веб-сторінці надсилати та отримувати дані з веб-сервера, не викликаючи оновлення сторінки, забезпечуючи безперешкодний інтерфейс користувача між користувачем і веб-програмою.

Фреймворки:

Фреймворк JavaScript може бути потужним інструментом, який можна використовувати для візуалізації сторінки. Зазвичай вони використовуються лише за наявності складних динамічних взаємодій.

Одним із прикладів цього є багатоетапне заповнення форми. У цьому випадку процес заповнення форми має певні кроки, які виконуються лише на основі раніше введеної інформації. Крім того, певні дані заповнюються для певних вхідних даних, а також попередніх вхідних даних. Виконання цього без фреймворку може бути дуже складним завданням. Речі можуть стати проблематичними, і це може статися швидко.

Використання фреймворку JavaScript допомагає вирішити ці проблеми, щоб ви могли заповнити свою чудову форму та порадувати своїх клієнтів. Хоча їх десятки, найпопулярнішими (на момент написання цієї статті) є Angular від Google, React від Facebook та Vue.js з відкритим вихідним кодом.

Переваги JavaScript:

- **Менш взаємодії з сервером** — можна перевірити введення користувача перед надсиланням сторінки на сервер. Це заощаджує трафік сервера, що означає менше навантаження на ваш сервер.

- **Негайний зворотний зв'язок із відвідувачами** — їм не потрібно чекати на перезавантаження сторінки, щоб побачити, чи не забули вони щось ввести.
- **Підвищена інтерактивність** - можна створювати інтерфейси, які реагують, коли користувач наводить на них курсор миші або активує їх за допомогою клавіатури.
- **Більш багаті інтерфейси**. Можна використовувати JavaScript для включення таких елементів, як компоненти перетягування та повзунки, щоб надати відвідувачам вашого сайту багатий інтерфейс.

2.6. JQuery.

jQuery - це бібліотека JavaScript, призначена для спрощення обходу дерева HTML DOM і маніпулювання ним, а також обробки подій, анімації CSS та Ajax.

Синтаксис jQuery спрощує навігацію за документом, вибір елементів DOM, створення анімації, обробку подій та розробку програм Ajax. jQuery також надає розробникам можливість створювати плагіни поверх бібліотеки JavaScript.

Це дозволяє розробникам створювати абстракції для низькорівневої взаємодії та анімації, розширених ефектів та високорівневих віджетів з підтримкою тем. Модульний підхід до бібліотеки jQuery дозволяє створювати потужні динамічні веб-сторінки та веб-програми.

Бібліотека jQuery зазвичай розповсюджується у вигляді одного файлу JavaScript, який визначає всі його інтерфейси, включаючи DOM, Events та Ajax-функції. Його можна увімкнути на веб-сторінку, пов'язану з локальною копією або однією з безлічі копій, доступних на загальнодоступних серверах.

jQuery має мережу доставки контенту (CDN), розміщену на MaxCDN. Google у службі Google Hosted Libraries та Microsoft також розміщують бібліотеку.

Приклад локального підключення копії бібліотеки (з того ж сервера, на якому розміщено веб-сторінку):

```
<script  src = "jquery-3.5.1.min.js"></script>
```

РОЗДІЛ 3. Розроблення програмних модулів для програмної онлайнової системи компанії з продажу авіаквитків.

3.1. Мова програмування та перелік використованих технологій.

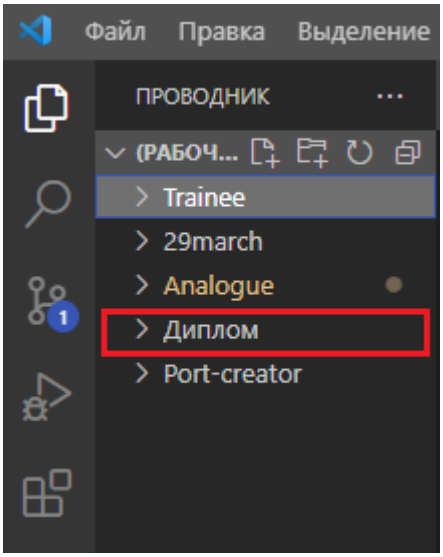
Основними інструментами розробки є мова JS, мова гіпертекстової розмітки HTML, препроцесор SCSS каскадної таблиці стилі CSS та бібліотеки для розробки Bootstrap, JQuery.

Текстова розмітка написана методологією БЕМ, описаної вище в "Розділ 2" (Основні інструменти розробки).

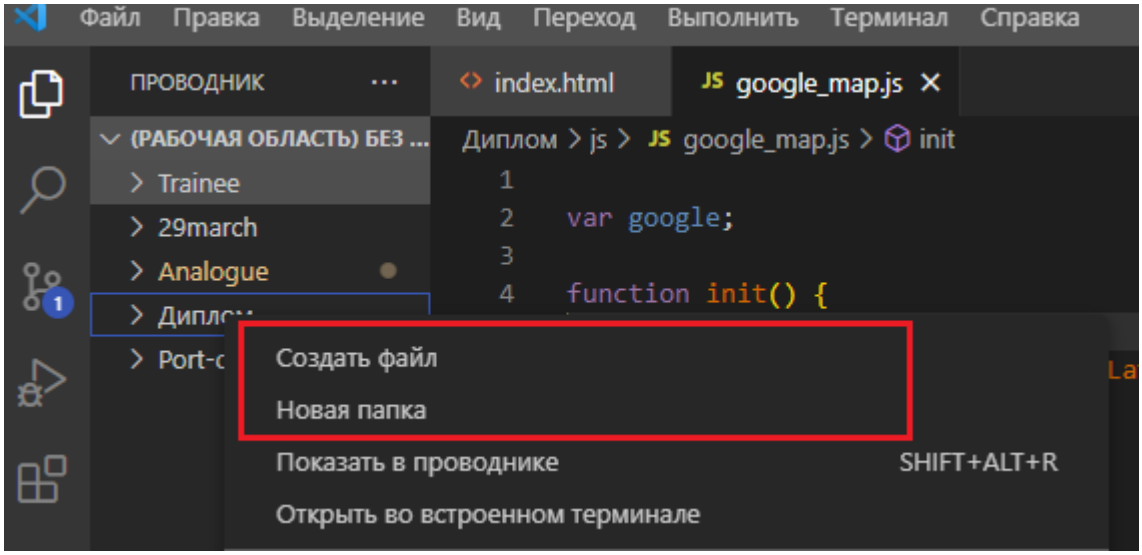
3.2. Створення папок/файлів у редакторі та підключення бібліотек.

Великий плюс редактора VSCode полягає в тому, що можна створювати файли всередині нього, створювати папки та підключати файли між собою.

Загальну папку я назвав досить просто - "Диплом", в яку вкладено інші папки та файли.



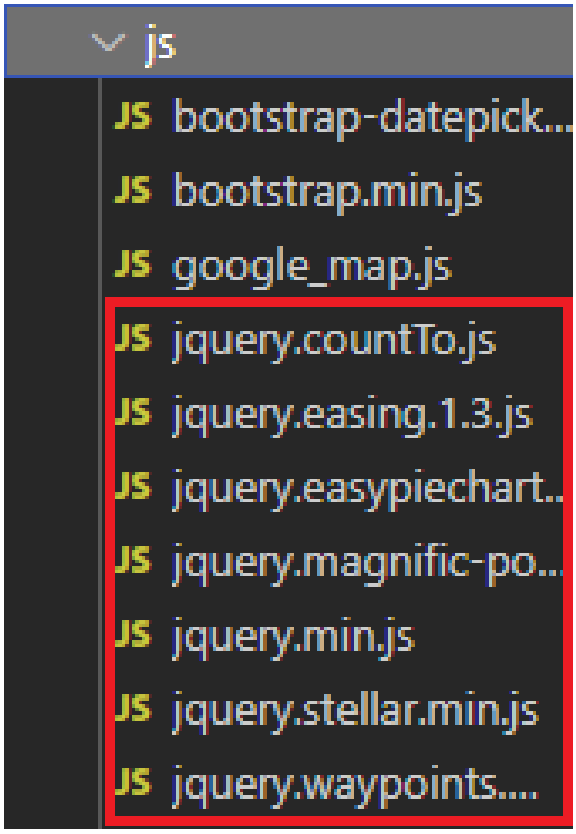
Створення дочірніх папок та файлів:



Йдемо за адресою і тиснемо велику жовто-жовтогарячу кнопку Download jQuery. Переходимо на сторінку, де вибираємо версію jQuery. Бажано вибирати останню із доступних. Позначення compressed – означає, що бібліотека мінімізована, тобто. займаємо мінімум місця, але, на жаль, читати початковий код складно. Після збереження підключаємо файл бібліотеки. Для моєї структури (всі скрипти лежать у папці js), код виглядає так:

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
  <!--Подключаем библиотеку-->
  <script src="js/jquery-2.2.3.min.js"></script>
</head>
<body>

</body>
</html>
```

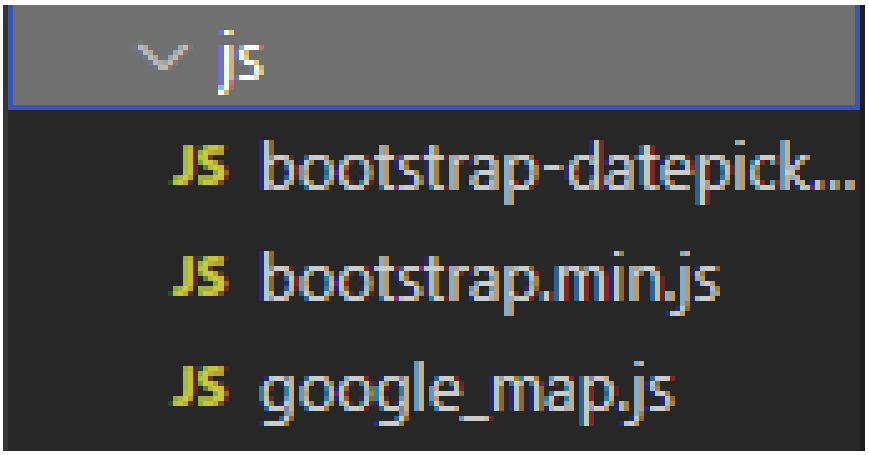
Структура проекту, бібліотеку jQuery завантажили в папку JS:



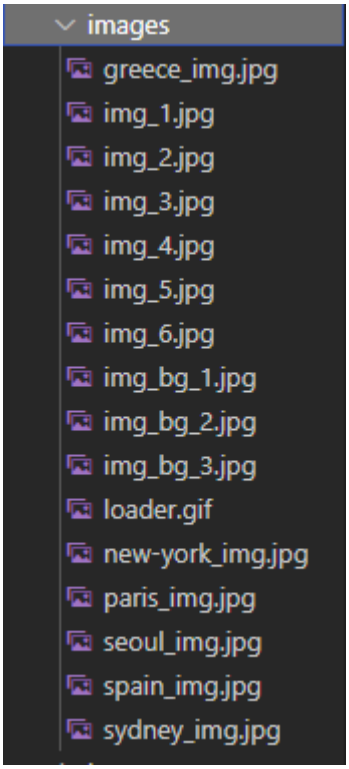
Для того щоб підключити Bootstrap необхідно:

- скачати готове складання з GitHub;
- розпакувати отриманий архів та скопіювати з нього мінімізовані версії файлів («bootstrap.min.css» та «bootstrap.bundle.min.js») у свій проект;
- підключити ці файли до HTML-сторінки.

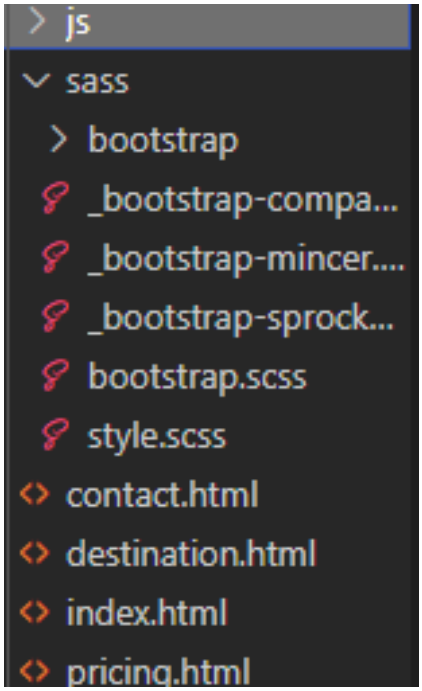
Після цих дій файли успішно підключені та готові до використання



Також для проекту мені потрібно завантажити картинки і додати їх до створеної мною папки "images" - (картинки).



Було створено 4 файли HTML, файли JS та CSS:



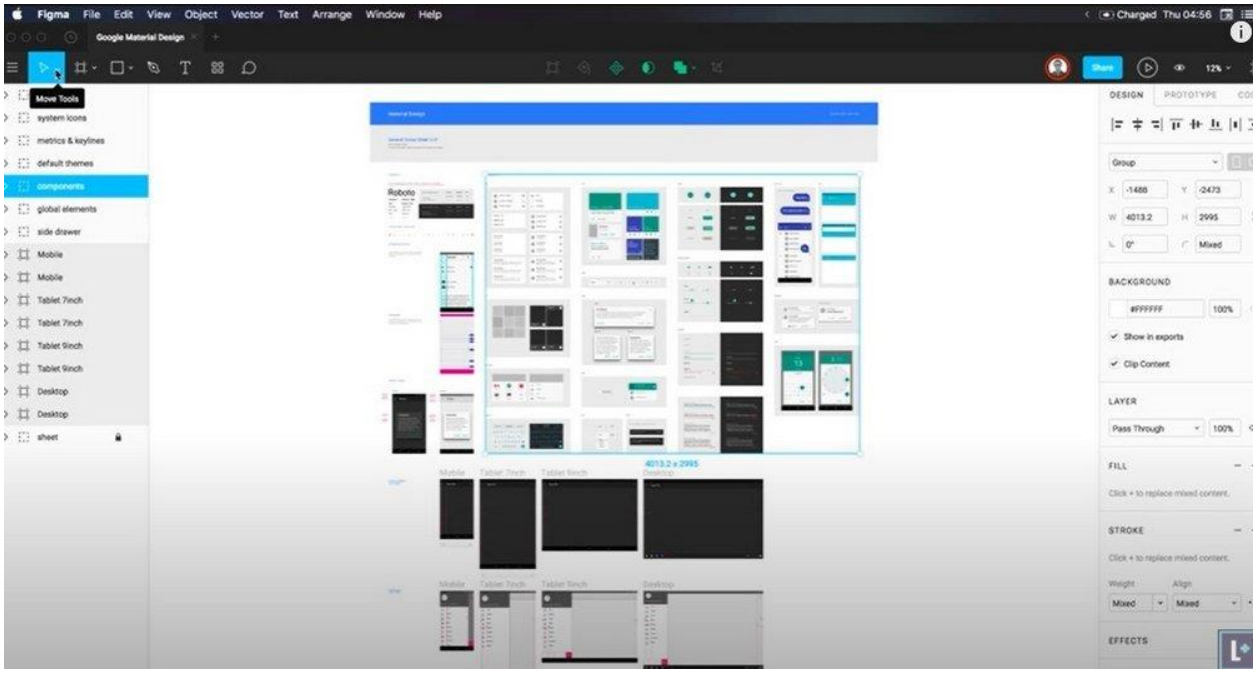
3.3. Розроблення основної частини – інтерфейсу системи.

Дизайн макету програмної онлайнової системи з продажу авіаквитків був взятий з мережі інтернет і використовувався в написанні коду.

Додаток "Figma" дозволив мені скористатися макетом і зручно написати код за прикладом, використовуючи правила відступів, розмірів та багатьох інших властивостей.

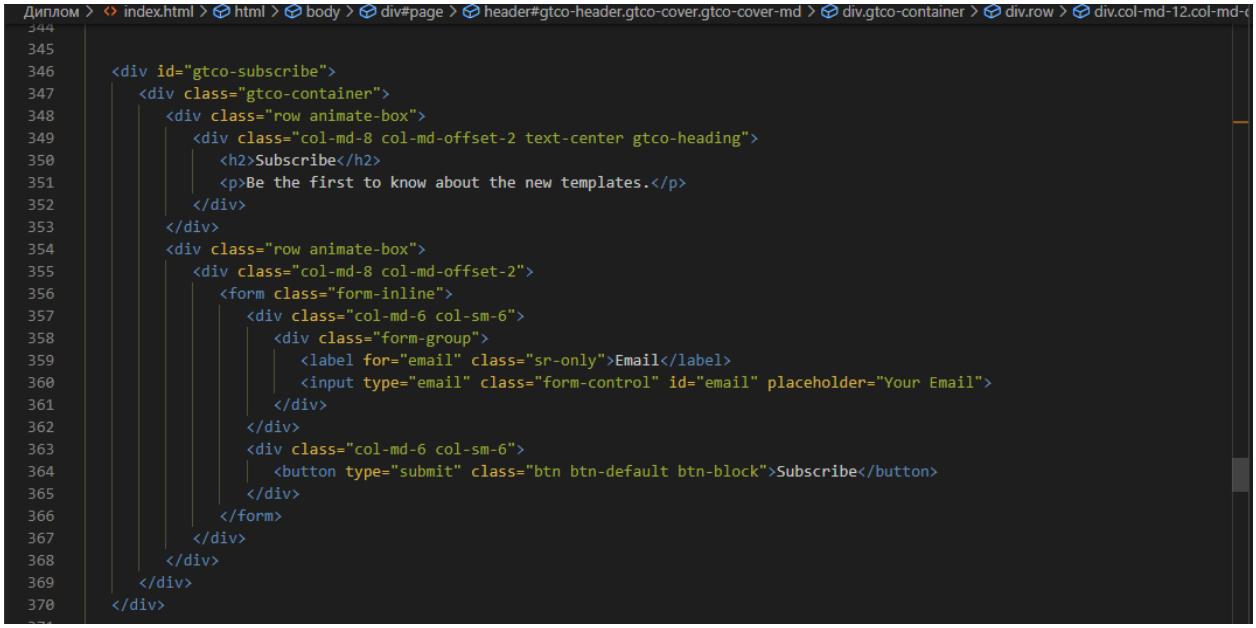
Figma — це веб-додаток для редагування графіки та дизайну інтерфейсу користувача. Можна використовувати його для виконання всіх видів робіт з графічного дизайну, починаючи з веб-сайтів, закінчуючи дизайном інтерфейсів мобільних програм, прототипуванням дизайну, створенням постів у соціальних мережах і всім, що між ними.

Інтерфейс програми:



Завдяки бібліотекам, підключеним до проекту, веб-сайт виходить інтерактивним, адаптивним і чуйним.

Скріншоти фрагментів коду:



(Фрагменти коду HTML файлу)

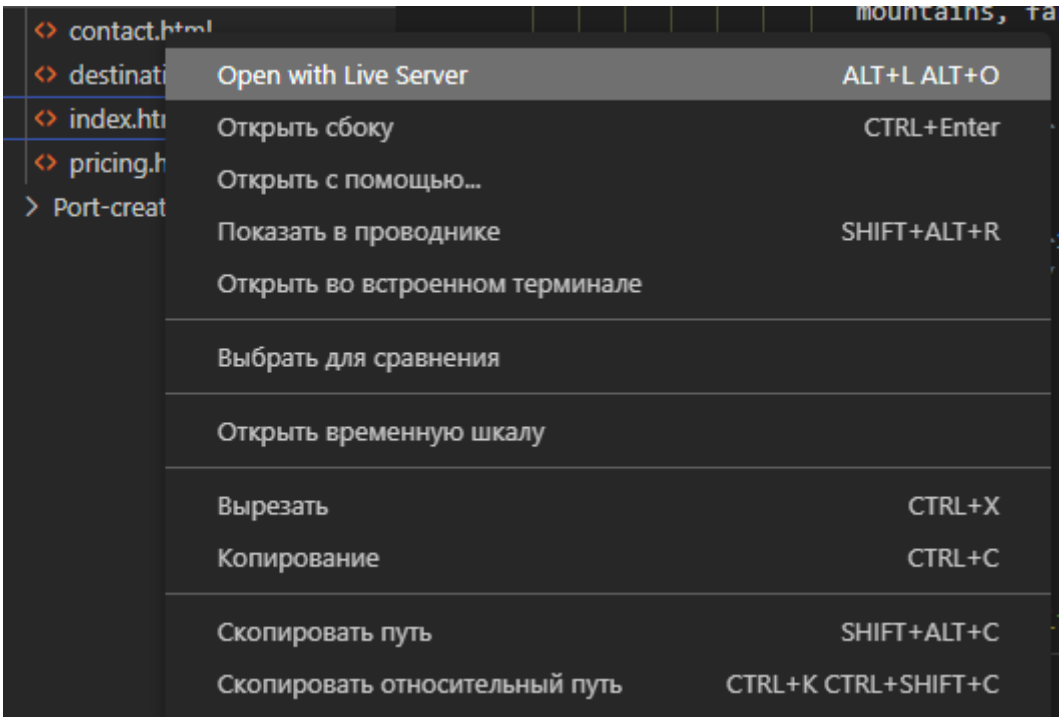
```scss
#gtco-header,
#gtco-counter,
.gtco-bg {
    background-size: cover;
    background-position: top center;
    background-repeat: no-repeat;
    position: relative;
}
.gtco-bg {
    background-position: center center;
    width: 100%;
    float: left;
    position: relative;
}

.gtco-video {
    height: 450px;
    overflow: hidden;
    margin-bottom: 30px;
    @include border-radius(7px);
    &.gtco-video-sm {
        height: 250px;
    }
    a {
        z-index: 1001;
        position: absolute;
        top: 50%;
```

(Фрагменти коду SCSS файлу)

(Фрагмент коду підключеного JQuery файла)

Завдяки плагінам всередині редактора коду VSCode можна полегшити та прискорити написання коду, наприклад завантажувати сторінку в реальному часі при змінах та редагування коду.

Натиснувши правою кнопкою миші HTML-файл у вікні провідника та вибравши "Open with Live Server", увімкнеться стан сервера і сторінка відкриється в браузері за замовчуванням.

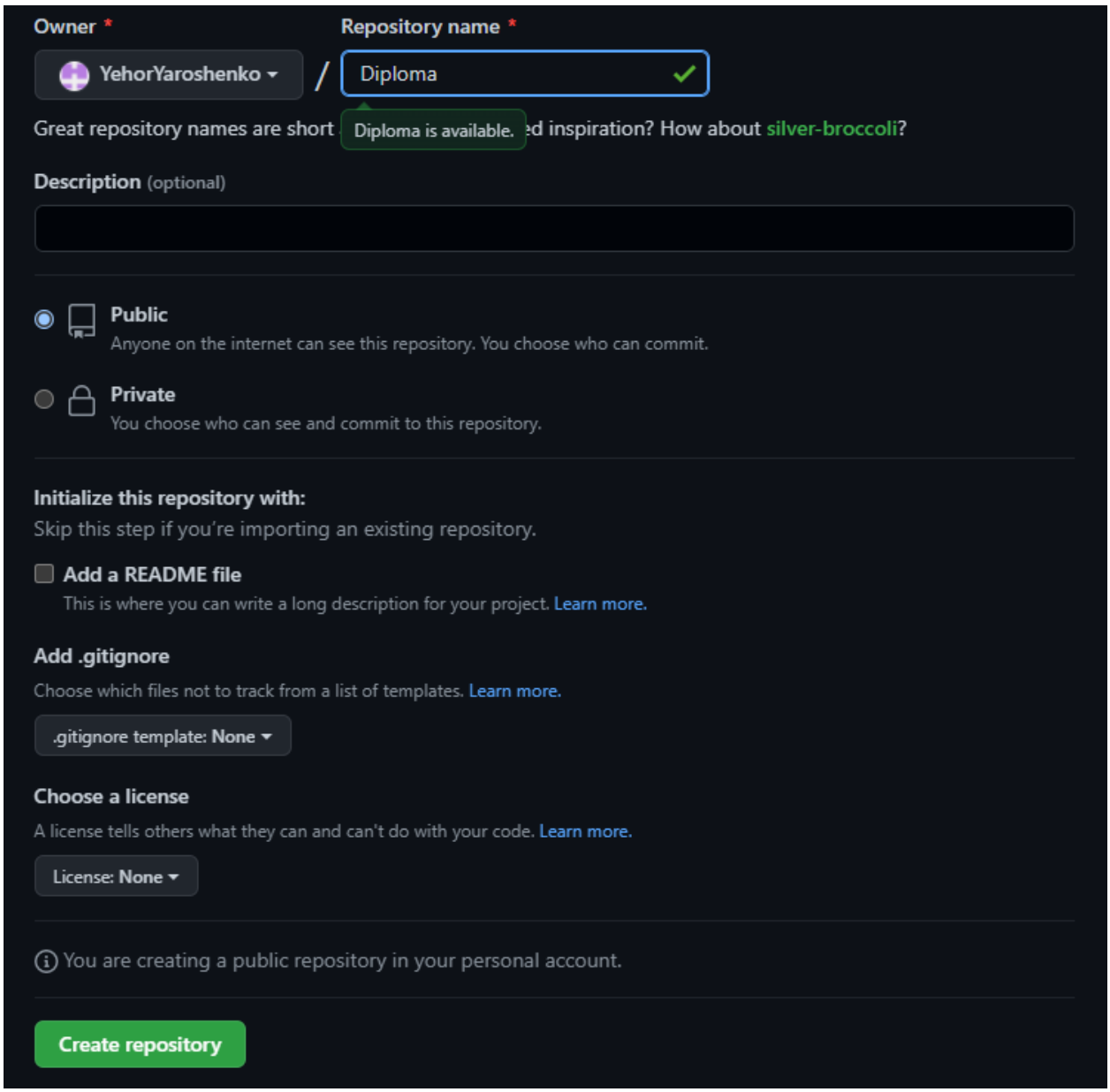


Щоб можна було подивитись код повністю без установки компонентів та завантажування файлів я додам свій проект на безкоштовний хостинг –

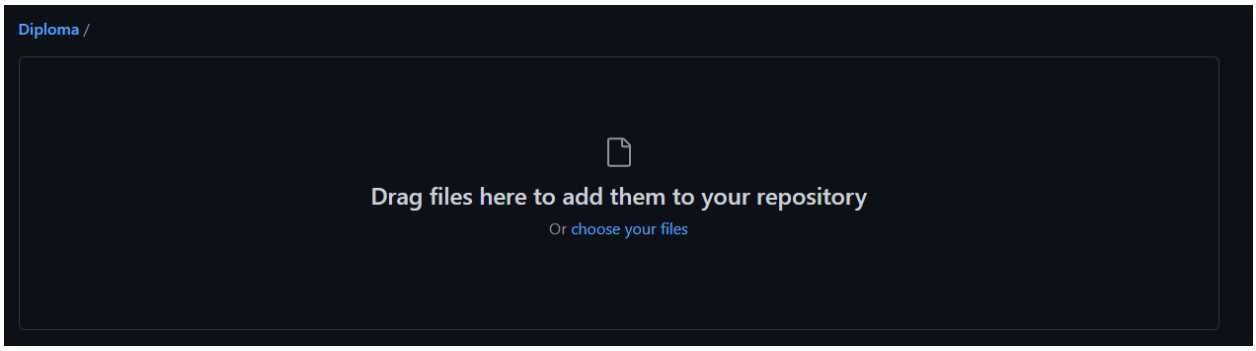
GitHub.

GitHub - провайдер інтернет-хостингу для розробки програмного забезпечення та управління версіями з використанням Git. Він пропонує функції розподіленого контролю версій та управління вихідним кодом (SCM) Git, а також власні функції. Він забезпечує контроль доступу та кілька функцій спільної роботи, таких як відстеження помилок, запити функцій, управління завданнями, безперервна інтеграція та вікі для кожного проекту.

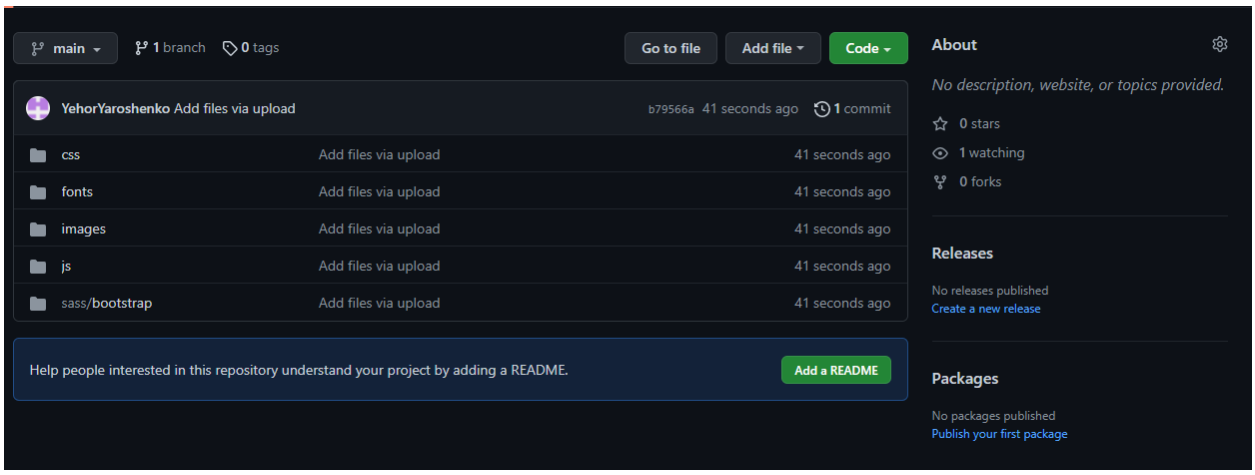
Для додавання файлів на GitHub слід створити репозиторій, я назву його "Diploma".



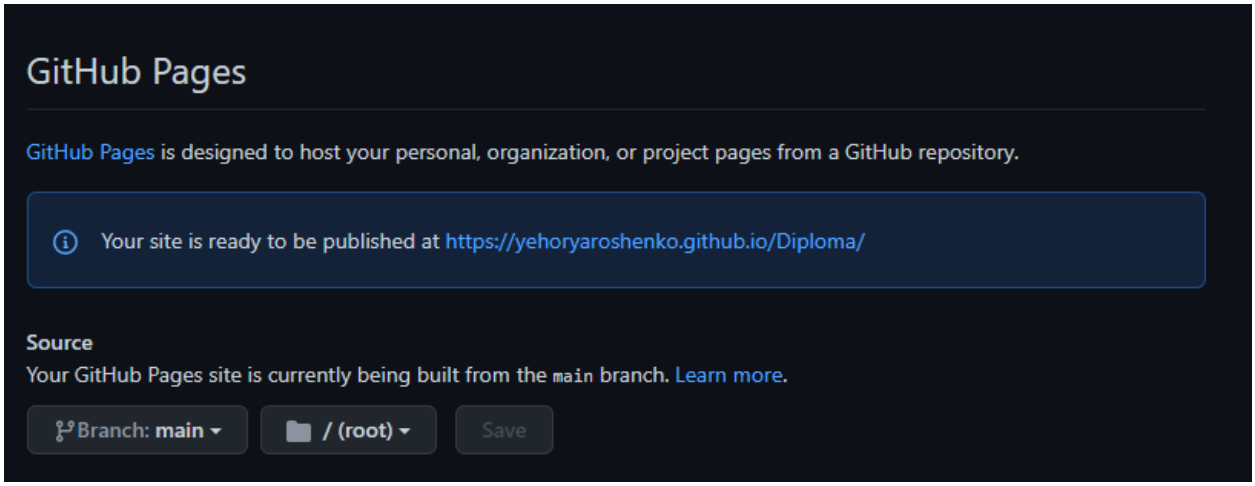
Після створення репозиторію завантажуємо файли.



Файли завантажені.



Після завантаження файлів було створено домен.



Посилання на сайт:

https://yehoryaroshenko.github.io/Diploma/

ВИСНОВКИ

Метою дипломного проекту було розробка інтерфейсу програмної онлайнової системи компанії з продажу авіаквитків, використовуючи веб-технології Front-end розробки.

Для створення цього додатку було проведено аналіз того, як він зможе допомогти та куди його можна впровадити. Можливо щось схоже вже працює і реалізовано в деяких аеропортах або ще на етапі створення наприклад блокчейн-компанія Zamna залучила $5 млн для використання блокчейну та біометричних даних з метою автоматизації перевірки пасажирів в аеропортах ОАЕ. Єдиним незначним минусом додатку є простий інтерфейс, заточений лише на надання практичного функціоналу без задоволення естетичних потреб майбутнього користувача.

Інтерфейс програми був розроблений з метою надання, як працівникам аеропорту, так і користувачу повного та урізаного функціоналу реєстрацій, відповідно, із інтуїтивно зрозумілим інтерфейсом. Користувачі легко можуть додавати свої дані, а працівники аеропорту зможуть їх за потреби виправити чи через непотрібність даних в БД видалити їх(наприклад якщо людина покинула аеропорт).

Створене програмно-технологічне рішення прискорення медичних перевірок пасажирів в аеропортах під час карантинних заходів відповідає усім описаним вимогам та стандартам.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. – К.: НАУ, 2017. – 63 с.

2. *An all-in-one test automation solution* [Електронний ресурс] – Режим доступу до ресурсу: *https://www.katalon.com/*

3. *Beazley D. Python essential reference / Beazley D. – BHV,* 2015. – 734*c.*

4. *Burns D. Selenium* 2 *Testing Tools*: *Beginner's Guide / Burns D. – Birmingham*: *Packt Publishing,* 2012. – 437 *c.*

5. *Kazarian A., Holoschuk R., Kunanets N., Pasichnyk V., Rzheuskiy A. Information Support Of The Virtual Research Community Activities Based On Cloud Computing, in*: *International Conference on Computer Sciences and Information Technologies, CSIT, CSIT,* 2018, *pp*. 199-202.

6. *Top* 10 *Automation Testing Tools in* 2022 [Електронний ресурс] – Режим доступу до ресурсу: *https://www.netsolutions.com/insights/top-10-automation-testing-tools/*

7. Гамма Е., Холм Р., Джонсон Р., Вліссідес Дж. Прийоми об'єктно-орієнтованого проектування. Паттерни проєктування / Гамма Е., Холм Р., Джонсон Р., Вліссідес Дж. – К: Комора, 2021. – 366 с.

8. Грофф, Дж. *SQL*: Повне керівництво [Текст] / Дж. Грофф: пров. з англ. – К.: Видавнича група *BHV,* 2011. – 816 с.

9. Зручна Транспортна Служба [Електронний ресурс] *URL*: *https://utsr.com* (дата звернення 03.11.2022)

10. Клієнт-серверна архітектура [Електронний ресурс] – Режим доступу до ресурсу: *https://uk.wikipedia.org/wiki/*

11. Криспін А. Гнучке тестування: практичне керівництво для тестувальників ПЗ і гнучких команд комманд: пер. з англ. / Л. Криспін.– Х.: «Думка», 2021. – 463 с. – *ISBN* 0-471-46912-2.

12. Куліков С. Тестування програмного забезпечення. Базовий курс/ Святослав Куліков., 2021. – Т. 3 : – 298 с.

13. Огляд протоколу *HTTP* [Електронний ресурс] – Режим доступу до ресурсу: *https://developer.mozilla.org/docs/Web/HTTP/Overview*

14. Пейн Едріан. Керівництво з *CRM*. Шлях до вдосконалення менеджменту клієнтів. – К.: Гревцов Паблышер, 2007. – 384 с.

15. Про Тестінг - Тестування ПЗ [Електронний ресурс] – Режим доступу до ресурсу: *http://www.pro_testing.cxom.ua/*

16. Рахімов, Т.Н., Заїкін, О.А., Рад, Б.Я. Основи побудови АСУ [Текст] / Т.М. Рахімов, О.А. Заїкін, Б.Я. Рад. – Х: "Уроборос", 2015. – 370 с.

17. Фаулер, М., Скотт К. *UML* в короткому викладі. Застосування стандартного мови об’єктного моделювання [Текст] / М. Фаулер, К. Скотт: пров. з англ. – К.: Наукова думка, 2019. – 721 с.