МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМЕНІ С.З. ҐЖИЦЬКОГО

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

першого (бакалаврського) рівня вищої освіти

на тему: **“ Розробка інтернет-магазину для продажу продуктів харчування ”**

Виконав: студент гр. Іт-41_____
Спеціальності 126 – «Інформаційні системи та технології»_____
(шифр і назва)

_____Кучкуда Максим Ігорович_____
(Прізвище та ініціали)

Керівник: ____к.е.н., доц. Смолінський В.Б.___
(Прізвище та ініціали)

Рецензенти: _____
(Прізвище та ініціали)

ДУБЛЯНИ-2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Перший (бакалаврський) рівень вищої освіти

126 – «Інформаційні системи та технології»

"ЗАТВЕРДЖУЮ"
Завідувач кафедри_____
д.т.н., проф. А.М. Тригуба
"____"_____2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу студенту

Кучкуда Максим Ігорович

1. Тема роботи: «Розробка інтернет-магазину для продажу продуктів харчування»

Керівник роботи Смолінський Валентин Броніславович, к.е.н., доцент
Затверджені наказом університету 123/к-с від 25.02.2025.

2. Строк подання студентом роботи 16.06.2025 р.

3. Початкові дані до роботи: 1. Вимоги до побудови веб-сайтів. 2. Стандарти і довідкова література. 3. Засоби створення та структура веб-сайту. 4. Інструменти та фреймворки. 5. Методика використання шаблонізаторів.

4. Зміст розрахунково-пояснювальної записки:
1. Аналіз веб-сайтів продажу їжі.
2. Постановка задачі створення веб-сайту.
3. Проектування веб-сайту.
4. Охорона праці та безпека в надзвичайних ситуаціях.
Висновки
Бібліографічний список.
Додатки

5. Перелік презентаційного матеріалу : Тема, автор, керівник кваліфікаційної роботи – *2 слайди;* Аналіз інтернет-магазинів – *1 слайд;* Аналіз-порівняння – *1 слайд*; Методика проектування – *1 слайд*; Структура проектованого сайту – *1 слайд*; Специфікація вимог – *1 слайд*; Програмна реалізація – *1 слайд*; Інструменти реалізації – *2 слайди*; Використання TWIG – *1 слайд*; Структура сайту та елементи коду – *1 слайд;* Тестування та оптимізація сайту – *1 слайд.*

6. Консультанти з розділів:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3	*Смолінський В.Б., доцент кафедри інформаційних технологій*		
4	*Городецький І.М., доцент кафедри інженерної механіки*		

7. Дата видачі завдання 25.02.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Етапи виконання кваліфікаційної роботи	Строк виконання етапів роботи	Приміт-ка
1.	Написання першого розділу та означення головних завдань роботи	*25.02.2025 – 01.03.2025*	
2.	Виконання другого розділу, опис інструментів та формування головних показників для розробки	*01.03.2025 – 01.04.2025*	
3.	Виконання третього розділу, реалізація мети роботи	*01.04.2025 – 01.05.2025*	
4.	Виконання четвертого розділу «Охорона праці та безпека в надзвичайних ситуаціях»	*01.04.2025 – 01.05.2025*	
5.	Завершення оформлення розрахунково-пояснювальної записки та презентаційного матеріалу	*01.05.2025 – 01.06.2025*	
6.	Завершення роботи в цілому	*01.06.2025 – 09.06.2025*	

Студент _____ Кучкуда М.І.
(підпис)

Керівник роботи_____ Смолінський В.Б.
(підпис)

УДК 004.432: 658.87

Кваліфікаційна робота: 51 с. текст. част., 15 рис., 2 табл., 13 слайдів, 20 джерел.

Розробка інтернет-магазину для продажу продуктів харчування. Кучкуда М.І. Кафедра ІТ. – Дубляни, Львівський НУВМБ, 2025.

У кваліфікаційній роботі розглядається процес розробки веб-сайту для продажу їжі. У першому розділі проведено аналіз сучасних веб-сайтів у сфері онлайн-торгівлі продуктами харчування, класифіковано їх типи та сформульовано технічне завдання для реалізації власного проекту.

Другий розділ присвячено вибору та обґрунтуванню технологій створення сайту: мови HTML, стилів CSS, автоматизації за допомогою Gulp, використанню npm-пакетів, JavaScript та шаблонізатора Twig.

У третьому розділі детально описано процес проектування веб-сайту, включаючи створення структури, макету, верстку та реалізацію функціональних можливостей у середовищі Visual Studio Code. Описано тестування, оптимізацію, розгортання веб-сайту та підготовку інструкції для користувача.

У четвертому розділі розглянуто питання охорони праці та безпеки в умовах надзвичайних ситуацій, зокрема електробезпеку та організацію безпечного технологічного процесу в офісі або закладі.

Робота є прикладом повного циклу створення сучасного веб-сайту від аналізу до впровадження з урахуванням безпекових аспектів.

Ключові слова: веб-сайт, онлайн-торгівля, HTML/CSS/JavaScript, проектування, шаблонізатор

ЗМІСТ

ВСТУП

Сьогодні ресторанний бізнес вважається однією з найбільш динамічних та перспективних сфер. Ефективне функціонування будь-якого закладу, зокрема ресторану, залежить від багатьох чинників. Процес управління рестораном є багатогранним і охоплює ведення бухгалтерії, контроль за роботою персоналу, аналіз фінансових операцій, управління складськими запасами, розрахунок вартості страв та інгредієнтів, а також контроль за термінами придатності продуктів і дотриманням санітарно-технічних стандартів.

Через велику кількість аспектів автоматизація цих процесів стає вкрай необхідною. Її впровадження приносить вигоди як для власників, так і для клієнтів: спрощується розрахунок, ефективно діє система лояльності, зменшуються черги, а також забезпечується актуальність меню. У цифрову епоху велике значення набуває онлайн-просування, адже кількість інтернет-користувачів постійно зростає. Наявність сучасного та зручного веб-сайту здатна залучити більше відвідувачів.

Реклама через Інтернет є економічно доцільною, дозволяє оперативно змінювати контент, отримувати зворотний зв’язок, проводити опитування, що дає змогу вдосконалювати якість обслуговування, страв та загальний сервіс. Власний сайт із повним меню також слугує важливим елементом формування позитивного іміджу закладу.

Мета роботи – розробити елементи веб-сайту інтернет-магазину для продажу продуктів харчування бургер-ресторану.

Виходячи з мети, визначено такі завдання:

1 проаналізувати онлайн-сервіси продажу їжі;

2 проаналізувати існуючі програмні рішення;

3 розробити план побудови системи, включаючи вибір технологій та реалізувати їх;

4 розробити інструкцію відвідувача.

РОЗДІЛ 1. АНАЛІЗ ВЕБ-САЙТІВ ПРОДАЖУ ЇЖІ

1.1 Розгляд понять та класифікація сайтів

В Інтернеті можна знайти кілька основних типів веб-сайтів, зокрема: соціальні мережі, інформаційні ресурси, комбіновані платформи, бізнес-сайти та онлайн-сервіси. Сайт-представництво, або сайт-візитка, забезпечує базову, але функціонально розширену присутність у мережі. Він зазвичай включає докладний опис послуг, приклади виконаних проєктів, відгуки клієнтів і форми для зворотного зв’язку [19].

Корпоративний сайт містить глибоку інформацію про діяльність компанії, її продукцію, послуги та новини. У порівнянні з сайтами-візитками, корпоративні платформи пропонують більше можливостей, зокрема пошук, фільтрацію контенту, календарі подій, блоги, форуми тощо. Часто вони інтегруються з внутрішніми системами підприємства — наприклад, ERP, CRM або бухгалтерським програмним забезпеченням. Такі ресурси також можуть мати окремі захищені розділи для співробітників, партнерів або підрядників.

Соціальні мережі створені для формування онлайн-спільнот людей із подібними інтересами або видами діяльності. Вони надають зручні інструменти для комунікації, включаючи внутрішні повідомлення або сервіси миттєвого обміну.

Комерційні сайти спрямовані на підтримку підприємницької діяльності. Їх можна розділити на ті, що допомагають у просуванні офлайн-бізнесу, та ресурси для електронної торгівлі, як-от інтернет-магазини.

Сайт-візитка, зазвичай, має від п’яти до десяти сторінок і подає основну інформацію про компанію або фізичну особу: сферу діяльності, контакти,

реквізити. Він виконує роль цифрової візитної картки для бізнесу або організації.

Цільова аудиторія комерційних веб-ресурсів — поточні та потенційні клієнти. У залежності від цілей та змісту, такі сайти можуть набувати різних форм:

1) інтернет-магазини мають каталог продукції з можливістю оформлення онлайн-замовлень. Вони зазвичай оснащені функціоналом для додавання товарів у "кошик" і подальшого оформлення покупки.

2) промо-сайти створюються з метою просування конкретного товару чи бренду. На таких платформах можна знайти технічні характеристики продукції, її переваги, а тако ж відомості про знижки та акції.

3) бізнес-портали належать, як правило, великим компаніям з широким спектром послуг. Вони складаються з численних сторінок і пропонують користувачам додаткові сервіси — електронну пошту, чати, форуми тощо [19].

Некомерційні сайти найчастіше містять інформаційний контент або онлайн-сервіси. Прикладом є особисті сторінки (домашні сторінки), на яких користувачі діляться контентом про себе — текстами, фотографіями тощо. Проте нині їх частково витіснили блоги та соціальні медіа. Прикладами таких сторінок є фан-сайти, присвячені відомим авторам, фільмам або музичним колективам [19].

У багатьох випадках навіть некомерційні сайти опосередковано приносять прибуток — через розміщення реклами або платні функції. Хоча вони не створювались для заробітку, їх іноді класифікують як умовно некомерційні, що ускладнює чітке розмежування.

Таким чином, класифікація сайтів є доволі складною і може варіюватися залежно від обраних критеріїв. Часто буває непросто визначити категорію, до якої належить певний веб-ресурс. Проте така типологія є корисною для орієнтації серед великої кількості доступних онлайн-ресурсів. Хоча кожен сайт має індивідуальний дизайн, усі вони поділяють спільні функціональні риси..

1.2 Аналіз інтернет-магазинів з харчуванням

Для аналізу обрано декілька сайтів зі схожою тематикою і призначенням.

Сайт «Burger King». Веб-сайт належить популярному ресторану швидкого обслуговування "Burger King", який працює на території Польщі. Ресурс має виразний, добре впізнаваний візуальний стиль, що базується на фірмових кольорах бренду — червоному, жовтому та білому. Такий дизайн повністю відповідає айдентиці "Burger King" і формує динамічну, привітну атмосферу.

Інтерфейс сайту вирізняється зручною структурою та логічною побудовою навігації, що полегшує доступ до основних розділів. Головне меню, яке розташоване у верхній частині сторінки, включає такі категорії, як "Меню", "Спеціальні пропозиції", "Про нас" та інші.

На сайті представлено повне меню "Burger King", де користувачі можуть ознайомитися з асортиментом бургерів, закусок, напоїв та іншої продукції. До кожної страви додається детальний опис, зображення та вартість. Крім того, доступна можливість індивідуалізації замовлення — наприклад, додавання інгредієнтів або вибір розміру порції.

Окремий розділ ресурсу присвячений історії бренду, де можна знайти інформацію про розвиток компанії, її місію, цінності та ключові досягнення.

Рисунок 1.1 – Сторінка меню Burger King

Загалом веб-сайт «Burger King» забезпечує користувачам зручний доступ до актуального меню, спеціальних пропозицій та додаткових сервісів. Він дає змогу оформити онлайн-замовлення та з легкістю спланувати відвідування найближчого ресторану мережі.

Веб-ресурс ресторану ***«Meat and Burger»*** вирізняється сучасним, вишуканим дизайном, у якому переважають темні відтінки, що формують стильну та витончену атмосферу. Візуальне оформлення підкреслює концепцію закладу, орієнтовану на м’ясні страви та бургери. Структура сайту інтуїтивно зрозуміла — головне меню знаходиться у верхній частині сторінки, а на головному екрані розміщені швидкі посилання на ключові розділи, зокрема: «Меню», «Про заклад», «Контакти» тощо.

На сайті представлено повне меню закладу «Meat and Burger», де користувач може ознайомитися з різними видами бургерів, м’ясними стравами, снеками та іншим асортиментом. Кожна позиція містить короткий опис, фотографію та ціну.

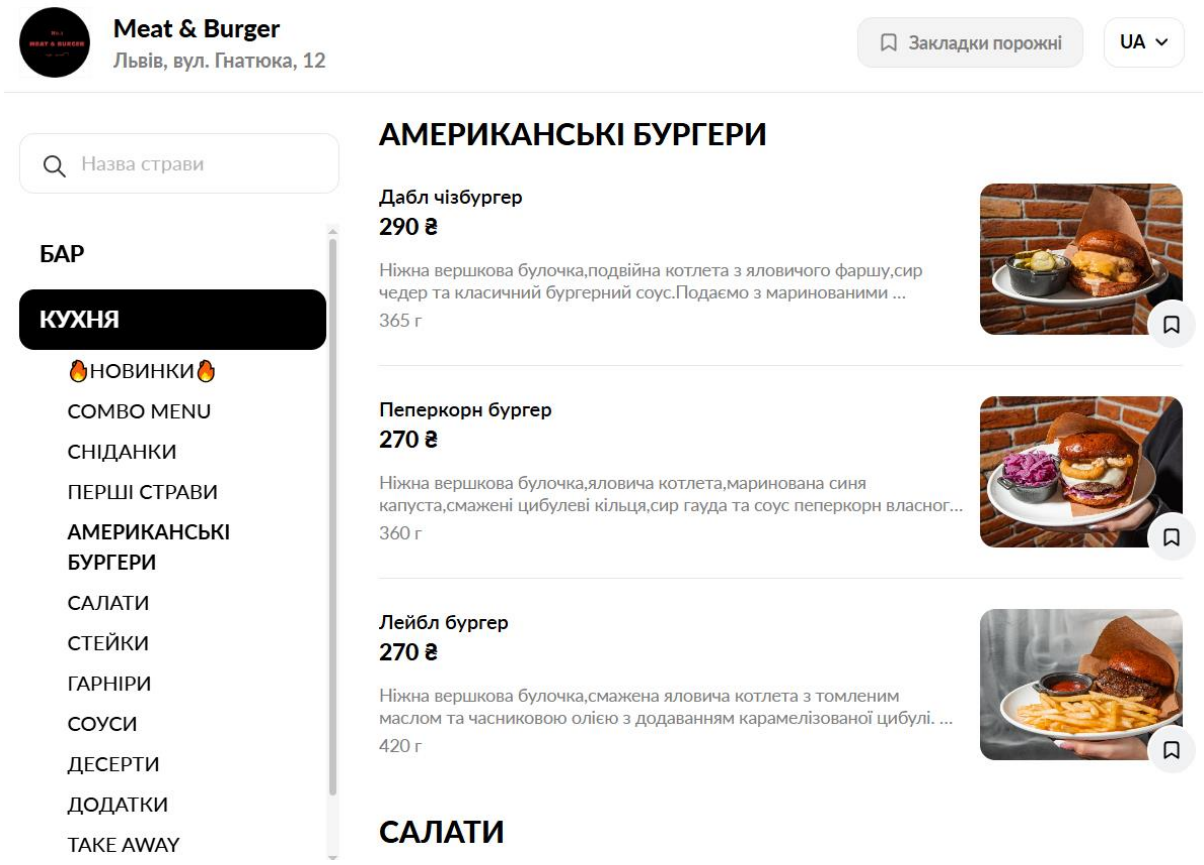

Рисунок 1.2 – Головна сторінка сайту Meat and Burger

Ресурс також надає інформацію про сам ресторан — його концепцію, підхід до якості продуктів, команду та сервіс. Присутній розділ з відгуками клієнтів, а також контактні дані (адреса, телефон, e-mail). Для зручності відвідувачів є форма зворотного зв’язку, яка дозволяє залишити повідомлення або звернення.

Загалом сайт «Meat and Burger» дозволяє легко ознайомитися з меню закладу, дізнатися про його особливості та зв’язатися з представниками ресторану для уточнення деталей чи оформлення замовлення.

Веб-сайт «Pasibus» також має сучасний, естетично привабливий інтерфейс. Колірна палітра створює динамічну, життєрадісну атмосферу, яка відповідає стилю бренду. Дизайн чітко транслює основну ідею закладу — спеціалізацію на бургерах. Основна навігація організована через горизонтальне меню у верхній частині сторінки, яке включає розділи «Бургери», «Напої», «Акції» тощо. Це спрощує доступ до потрібної інформації.

Рисунок 1.3 – Сторінка меню сайту Pasibus

На сайті представлено повний перелік бургерів з можливістю переглянути склад, опис та ціну. Окрім класичних варіантів, доступні також вегетаріанські та веганські бургери, кожен з яких супроводжується фотографією. Веб-ресурс містить детальну інформацію про філософію ресторану, якість і походження

інгредієнтів, а також підходи до приготування страв. Окрім бургерів, представлена інформація про інші елементи меню, наприклад, напої.

Контактна інформація ресторану включає адресу, телефон та посилання на офіційні сторінки у соціальних мережах. Крім того, на сайті доступна форма зворотного зв’язку, яка дозволяє користувачам швидко сконтактувати з представниками закладу. У підсумку, сайт «Pasibus» дозволяє отримати повне уявлення про асортимент бургерів, дізнатися більше про концепцію ресторану та зручно зв’язатися з ним для замовлення або отримання консультації.

Таблиця 1.1 – Порівняння аналогів сайтів

Характеристика	**Pasibus**	**Meat and Burger**	**Burger King**
1	2	3	4
Функціонал	Онлайн-меню, зручна навігація, контактна інформація, форма зворотного зв'язку	Онлайн-меню, контактна інформація, огляди клієнтів	Онлайн-меню, контактна інформація, огляди клієнтів
Дизайн	Сучасний, жвавий, енергійний	Стильний, сучасний	Корпоративний, чіткий
Схожість тематики	Спеціалізація на бургерах, багато варіантів страв	Спеціалізація на м'ясі та бургерах	Широкий асортимент страв
Зворотній зв'язок	Форма зворотного зв'язку, контактна інформація	Контактна форма зворотного зв'язку, контактна інформація	Контактна форма зворотного зв'язку, контактна інформація
Збільшення фото меню	Не вказано	Присутнє	Присутнє
Цінова політика	Ціни вказані в меню, прозора цінова політика	Ціни вказані в меню	Ціни вказані в меню
Інгредієнти	Інформація про склад бургерів, вегетаріанські та веганські опції	Інформація про склад бургерів	Інформація про склад страв
Зручність використання	Зручна навігація, легко знаходити необхідну інформацію	Зручна навігація, легко знаходити необхідну інформацію	Зручна навігація, легко знаходити необхідну інформацію

Після детального аналізу сайтів зі схожою тематикою та функціональним призначенням було ухвалене рішення розробити власний веб-ресурс, який враховуватиме переваги наявних прикладів, але водночас не матиме їхніх недоліків, зазначених у порівняльній таблиці. Запланований сайт поєднає

найкращі функціональні та візуальні рішення з унікальним дизайном і зручною структурою для користувача.

1.3 Технічне завдання кваліфікаційної роботи

Метою роботи є створення сучасного веб-сайту під назвою Urban із використанням актуальних веб-технологій. Основні завдання полягають у проєктуванні, розробці та впровадженні ресурсу, що дозволятиме ефективно представляти ресторанний заклад в онлайн-середовищі.

Веб-сайт "Urban" забезпечуватиме користувачів повною інформацією про ресторан: контактні дані, місцезнаходження, графік роботи, а також розділ із загальним описом закладу та галереєю зображень.

Ключовою функцією сайту стане перегляд актуального меню з можливістю ознайомлення зі складом страв та використаними інгредієнтами. Це дозволить відвідувачам заздалегідь отримати повну інформацію про асортимент і зробити вибір, не виходячи з дому.

Крім цього, сайт буде оснащений багатомовним інтерфейсом — користувачі зможуть перемикати мови залежно від своїх потреб, що забезпечить комфортне користування ресурсом для гостей з різних країн.

Таким чином, сайт Urban стане ефективним інструментом цифрової взаємодії між рестораном і клієнтами, сприятиме підвищенню впізнаваності закладу, залученню нових відвідувачів і покращенню якості обслуговування.

РОЗДІЛ 2. ПОСТАНОВКА ЗАДАЧІ СТВОРЕННЯ ВЕБ-САЙТУ

2.1 Мова розмітки гіпертекст- HTML

HTML (*HyperText Markup Language*) — це мова розмітки, яка використовується для створення структури веб-сторінок. Вона визначає, як має виглядати вміст сторінки, та надає йому семантичне значення, що дозволяє браузерам правильно відображати інформацію користувачам.

Основні характеристики HTML:

1. Теги — основа HTML-документів. Вони визначають елементи сторінки й зазвичай мають парну структуру: відкриваючий і закриваючий тег, що охоплюють вміст.

2. Структурна ієрархія — HTML використовує вкладену систему елементів. Найвищим рівнем структури є тег <*html*>, у межах якого розміщуються основні секції: <*head*> і <*body*>.

3. Елементи контенту — HTML містить набір стандартних тегів для оформлення різних частин контенту. Наприклад, теги <*h1*>–<*h6*> використовуються для заголовків, <*p*> — для абзаців, <*img*> — для вставки зображень тощо.

4. Атрибути — спеціальні властивості, які додаються до елементів для уточнення їхніх параметрів. Наприклад, атрибут *src* у тегі <*img*> визначає шлях до зображення.

5. Гіперпосилання — елемент <*a*> дозволяє створювати посилання на інші сторінки або ресурси в Інтернеті, що забезпечує навігацію між веб-ресурсами.

6. Форми для введення даних — HTML підтримує створення форм за допомогою тегів <*input*>, <*textarea*>, <*select*> тощо, що дозволяє отримувати інформацію від користувачів.

7. Семантичні елементи — HTML також включає теги, які мають значення для структурування контенту з точки зору його призначення.

Наприклад, *<header>*, *<nav>*, *<main>*, *<article>*, *<footer>* розділяють сторінку на логічні блоки.

Типовий HTML-документ містить базові елементи, які формують основу для подальшого оформлення та функціонування веб-сторінки:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Назва сторінки</title>
</head>
<body>
    <!-- Зміст сторінки тут -->
</body>
</html>
```

HTML є фундаментальною мовою у сфері веб-розробки й активно використовується разом із CSS (таблицями стилів) і JavaScript для створення динамічних, інтерактивних веб-додатків. Знання HTML є критично важливим для побудови правильно структурованих і семантично грамотних веб-ресурсів [3].

2.2 Каскадні таблиці стилів – CSS

CSS (*Cascading Style Sheets*) або каскадні таблиці стилів — це мова, що використовується для візуального оформлення веб-сторінок. Вона дозволяє задавати зовнішній вигляд HTML-елементів, зокрема їхнє розташування, кольори, шрифти, фонові зображення та інші стилістичні властивості.

Одна з основних особливостей CSS — каскадність, яка полягає в тому, що стилі можуть застосовуватись із різних джерел: безпосередньо в HTML-елементах, у внутрішньому стилі в межах HTML-документа або у зовнішніх CSS-файлах. При цьому, система пріоритетів визначає, які стилі матимуть перевагу, якщо вони конфліктують.

CSS забезпечує широкі можливості для оформлення сторінки за допомогою різноманітних селекторів і властивостей, що дозволяє розробникам точно керувати візуальним виглядом елементів. Зокрема, можна налаштовувати розміри,

кольори, шрифти, рамки, відступи, вирівнювання, анімації тощо. Такий підхід дає змогу створювати естетично привабливі, адаптивні й однаково коректні сторінки на різних пристроях і в усіх сучасних браузерах [2].

У сучасній веб-розробці CSS застосовується в тісній інтеграції з HTML і JavaScript для створення інтерактивних веб-інтерфейсів. Завдяки можливості централізованого управління стилями, навіть одна зміна в CSS-файлі може вплинути на вигляд усієї сторінки або сайту, що значно спрощує оновлення дизайну та подальшу підтримку.

Sass (*Syntactically Awesome Style Sheets*) та його популярна варіація SCSS (Sassy CSS) — це препроцесорні мови, які компілюються у звичайний CSS. Вони додають до стандартного синтаксису CSS низку розширених можливостей, таких як змінні, вкладені селектори, міксіни, оператори й функції.

Ці інструменти значно полегшують написання складних стилів, дозволяючи структурувати код, уникати повторів і пришвидшити розробку. Використання Sass або SCSS покращує масштабованість та підтримку стилів у великих проєктах, дозволяючи краще організувати стилі за допомогою модульності та повторного використання коду [2].

Рисунок 2.1 – Відмінності препроцесорів

Загалом, Sass і SCSS забезпечують потужний набір інструментів для оптимізації та підтримки CSS-структур. Вони допомагають зменшити

дублювання коду, підвищити його читабельність і спрощують адаптацію дизайну під нові вимоги, що є надзвичайно важливим у сучасній веб-розробці [7].

2.3 Інструмент для автоматизації розробки – Gulp

Gulp — це популярний інструмент для автоматизації типових завдань, які виникають під час створення веб-проєктів. Його головна функція полягає в упорядкуванні та автоматичному виконанні рутинних процесів, що значно пришвидшує розробку та оптимізує робочий процес.

За допомогою Gulp можна реалізовувати різноманітні дії, зокрема: об'єднання та мініфікацію CSS і JavaScript-файлів, оптимізацію зображень, автоматичне оновлення сторінок при зміні коду, запуск тестів тощо [10]. Це дозволяє зменшити кількість ручної роботи та уникнути помилок, які можуть виникати під час повторюваних операцій.

Одна з ключових переваг Gulp полягає в тому, що він дозволяє зосередитись безпосередньо на написанні функціонального коду, оскільки всі допоміжні задачі виконуються автоматично. Завдяки цьому розробники можуть витрачати менше часу на технічне обслуговування проєкту.

Інструмент також підтримує широку екосистему плагінів, що дозволяє легко розширювати його функціонал. Наприклад, з використанням додаткових модулів можна налаштувати автоматичну перевірку якості коду, створення документації, деплой на сервер тощо.

Отже, Gulp є надійним та ефективним інструментом автоматизації, який значно полегшує процес веб-розробки. Його застосування дозволяє підвищити продуктивність команди, скоротити час розробки та підтримувати високу якість кінцевого продукту [11].

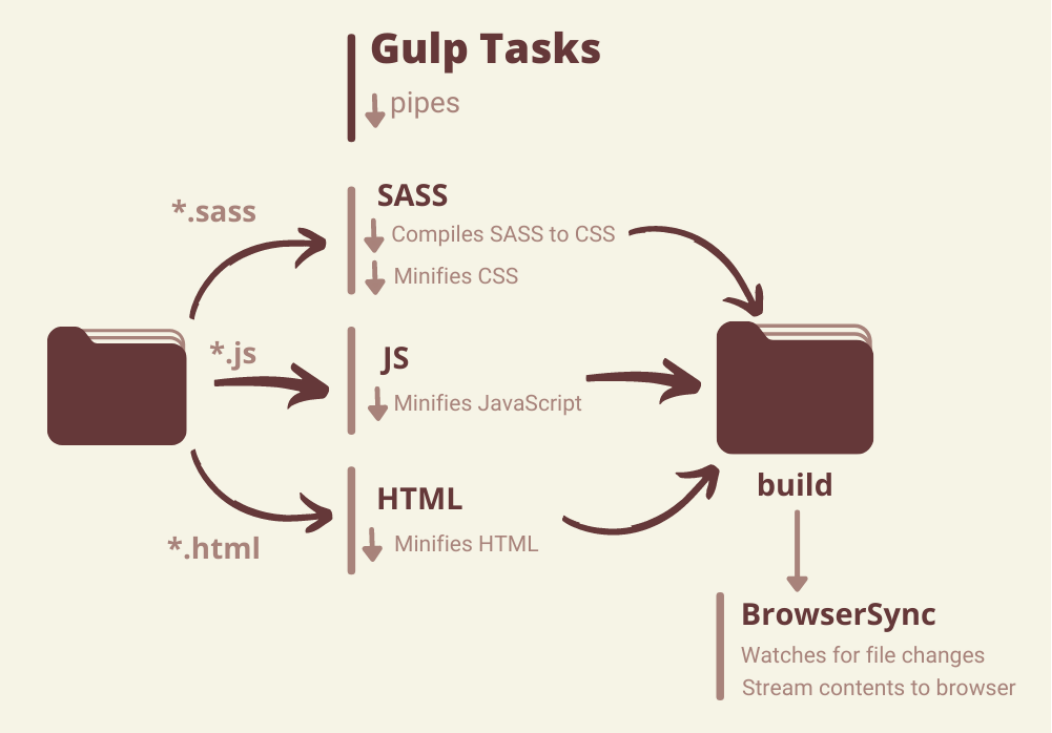

Рисунок 2.2 – Gulp, як інструмент збірки

```
// Підключення необхідних модулів
const gulp = require('gulp');
const sass = require('gulp-sass');
const minifyCSS = require('gulp-clean-css');
const concat = require('gulp-concat');
// Завдання для компіляції SCSS в CSS
gulp.task('compile-scss', function () {
  return gulp.src('src/scss/**/*.scss') // Вихідні файли SCSS
    pipe(sass()) // Компіляція SCSS в CSS
    pipe(gulp.dest('dist/css')); // Збереження скомпільованих CSS-файлів
});
```

У наведеному фрагменті коду спочатку імпортуються необхідні модулі для роботи з Gulp, зокрема gulp, gulp-sass, gulp-clean-css та gulp-concat. Далі створюються два окремі завдання: compile-scss і minify-css.

Завдання compile-scss виконує трансформацію SCSS-файлів у звичайні CSS-файли. Для цього задається шлях до SCSS-джерел, виконується компіляція з використанням gulp-sass, після чого отримані CSS-файли зберігаються у відповідну папку проєкту.

Завдання minify-css поєднує всі CSS-файли в один за допомогою gulp-concat, а потім стискає (мінімізує) цей об’єднаний файл за допомогою gulp-clean-

css. У результаті отриманий компактний файл зберігається у вказану директорію.

На завершення використовується метод gulp.task('default', gulp.series ('compile-scss', 'minify-css')), що створює завдання за замовчуванням. Це дозволяє запускати обидва етапи — компіляцію та мініфікацію — послідовно однією командою gulp.

Цей приклад демонструє базову конфігурацію Gulp для обробки стилів, яку за потреби можна розширити, додавши нові плагіни або завдання залежно від технічних вимог конкретного вебпроєкту.

2.4 Менеджер пакетів та npm-плагіни

Окрім базових модулів, що вбудовані у Node.js, у спільноті створено безліч сторонніх бібліотек, фреймворків і інструментів, які можна легко інтегрувати у власні проєкти. Серед найпопулярніших прикладів – Express, Grunt, Gulp тощо. Ці компоненти поширюються у формі пакетів, які об'єднують певну функціональність і можуть бути повторно використані у різних розробках.

Щоб спростити управління такими пакетами, застосовуються спеціальні інструменти – менеджери пакетів. Основним менеджером для JavaScript-проєктів є npm (Node Package Manager). Він дозволяє зручно встановлювати, оновлювати, видаляти залежності, а також керувати конфігурацією проєкту через файл package.json [11].

У межах даної розробки було застосовано низку npm-плагінів для спрощення роботи, автоматизації процесів та оптимізації фронтенду веб-сайту. Серед них:

1 SCSS – препроцесор для CSS, що додає корисний синтаксис: змінні, вкладеність, міксіни тощо. Використання SCSS робить стилі більш структурованими та легко підтримуваними [13].

2 MinifyJS – інструмент, який зменшує розмір JavaScript-файлів шляхом видалення непотрібних символів (пробілів, коментарів). Це позитивно впливає на

швидкість завантаження сторінок та економію трафіку.

3 Layout – плагін, що полегшує верстку сторінки, дозволяє організовувати блоки, створювати сітки та адаптивний інтерфейс.

4 CriticalCSS – модуль, який виділяє найважливіші стилі, потрібні для першого екрану. Це дозволяє прискорити початкове відображення сторінки, що покращує загальну продуктивність і користувацький досвід.

Завдяки цим модулям вдалося підвищити якість, швидкодію та зручність підтримки сайту, розробленого в межах даного проєкту.

2.5 Динамічна мова програмування JavaScript

JavaScript — це динамічна мова програмування, яка виконується безпосередньо у веб-браузері та дозволяє створювати інтерактивні функції на веб-сторінках. Вона реалізує стандарти ECMAScript і базується на прототипному підході, що походить від мови Self. JavaScript було створено компанією Netscape у співпраці з Sun Microsystems для додавання інтерактивності до HTML-документів.

Основні сфери застосування JavaScript: 1) динамічне оновлення контенту — зміна елементів сторінки без її перезавантаження; 2) перевірка форм — перевірка введених даних перед відправкою на сервер; 3) клієнтські обчислення — виконання логіки на стороні браузера; 4) робота з DOM — зміна структури HTML, подій, стилів тощо.; 5) асинхронні запити — використання AJAX для взаємодії із сервером без оновлення сторінки; 6) збереження даних — доступ до localStorage, sessionStorage, cookies ;7) анімації та ефекти — створення інтерактивних візуальних елементів; 8) завдяки цим можливостям JavaScript став основою сучасної веб-розробки як на клієнтській, так і на серверній стороні (через Node.js) [17].

JavaScript на стороні клієнта. JavaScript, використаний на стороні клієнта, виступає ключовим інструментом для створення інтерактивного та динамічного

користувацького досвіду в сучасних веб-додатках. Він виконується безпосередньо у браузері користувача та дозволяє гнучко керувати елементами веб-сторінки, реагуючи на дії відвідувача в режимі реального часу.

Основні функції клієнтського JavaScript включають:

1. Робота з DOM (Document Object Model). Мова дозволяє змінювати структуру та вміст веб-сторінки без її повторного завантаження. Наприклад, можна динамічно створювати, видаляти або змінювати HTML-елементи, застосовувати стилі, обробляти події, такі як натискання кнопки або введення тексту.

2. Валідація форм. JavaScript використовується для перевірки правильності введених користувачем даних перед їх надсиланням на сервер. Це дозволяє оперативно інформувати користувача про помилки введення та зменшує навантаження на сервер.

3. Асинхронна взаємодія з сервером (AJAX). Завдяки технології AJAX JavaScript здатен виконувати запити до сервера у фоновому режимі, оновлюючи частину вмісту сторінки без повного її перезавантаження. Це забезпечує плавність і зручність взаємодії з веб-інтерфейсом.

4. Робота з локальним сховищем і cookie. JavaScript може читати, записувати та видаляти cookie, а також використовувати localStorage і sessionStorage для збереження даних між сесіями користувача. Це корисно для збереження налаштувань, стану авторизації тощо.

5. Створення анімацій та інтерактивної візуалізації. JavaScript дозволяє реалізовувати анімовані ефекти, динамічні переходи, візуалізацію даних (наприклад, діаграми, графіки) та інші засоби, що покращують естетичне сприйняття сторінки.

JavaScript на стороні клієнта забезпечує основу для побудови інтерактивних веб-додатків, що працюють у браузері без потреби постійного звернення до серверу. Його універсальність і підтримка всіма сучасними браузерами робить цю мову невід’ємною частиною фронтенд-розробки [16].

2.6 Twig інстумент PHP-шаблонізатор коду

Twig – це потужний шаблонізатор для PHP, який дозволяє розділяти логіку програми та її представлення. Це досягається через систему шаблонів, що використовують зрозумілий синтаксис і не потребують вставок PHP-коду безпосередньо у HTML [19].

Переваги використання Twig: 1) зрозумілий, легкий для навчання синтаксис; 2) підтримка умов, циклів, фільтрів та шаблонного спадкування; 3) розділення бізнес-логіки та інтерфейсу; 4) кешування шаблонів для підвищення продуктивності.

Основні етапи використання Twig: 1) інсталяція через Composer — Twig легко додається до проєкту за допомогою менеджера пакетів Composer; 2) організація шаблонів — зазвичай файли розміщуються у директорії templates/; 3) передача даних у шаблон — змінні, масиви, об’єкти передаються з PHP-коду; 4) робота з шаблонною логікою — умовні блоки (if), цикли (for), вставки шаблонів (include) тощо; 5) успадкування шаблонів — дозволяє створити базовий шаблон і розширювати його при потребі; 5) використання фільтрів — Twig підтримує фільтри для форматування значень (наприклад, {{ name|upper }}); 6) кешування — Twig може зберігати скомпільовані шаблони для пришвидшення завантаження сторінок [17].

```
<nav>
    <ul>
      {% for item in menuItems %}
        <li>
          <a href="{{ item.url }}">{{ item.label }}</a>
        </li>
      {% endfor %}
    </ul>
  </nav>
```

У цьому фрагменті використовується цикл for для створення динамічного меню. Змінна menuItems — це масив об'єктів, кожен з яких має url (посилання) та label (назву пункту). Цей підхід дозволяє формувати навігацію з даних, що надходять із бекенду, без жорсткого кодування.

РОЗДІЛ 3. ПРОЕКТУВАННЯ ВЕБ-САЙТУ

3.1 Середовище розробки Visual Studio Code

Visual Studio Code (VS Code) — це безкоштовний, зручний і потужний редактор для написання коду, розроблений компанією Microsoft. Його активно використовують фахівці в галузі веб-розробки та програмування завдяки широкому функціоналу й гнучкості налаштування.

Серед головних особливостей, які пропонує VS Code, варто відзначити [24]:

1) Підсвічування синтаксису та розпізнавання мов: редактор підтримує велику кількість мов програмування і розмітки, що забезпечує зручність читання та редагування коду завдяки синтаксичному виділенню.

2) Система розширень та плагінів: за допомогою численних розширень користувачі можуть адаптувати редактор під власні потреби — додавати підтримку мов, фреймворків, інструментів для автоматизації тощо.

3) Інтеграція з системами контролю версій: VS Code підтримує інтеграцію з Git, що дає змогу ефективно керувати змінами у проєктах та працювати з репозиторіями безпосередньо з редактора.

4) Вбудований термінал: редактор оснащений інтегрованим терміналом, завдяки якому можна виконувати команди напряму в середовищі розробки, що спрощує запуск серверів, виконання скриптів та інші командні операції.

5) Інструменти для налагодження: VS Code дозволяє налагоджувати код безпосередньо всередині редактора — це зручно для пошуку помилок, перегляду змінних і аналізу виконання програми.

6) Можливості спільної роботи: редактор підтримує командну розробку — кілька користувачів можуть одночасно працювати з кодом, залишати коментарі й синхронно вносити зміни.

Усе це робить Visual Studio Code популярним серед розробників по всьому світу. Програма підтримує багато операційних систем і має активну спільноту, яка постійно створює нові розширення й покращує функціональність редактора.

Крім того, VS Code підтримує більшість сучасних мов програмування. Базовий набір включає JavaScript, TypeScript, HTML і CSS, але через VS Code Marketplace можна встановити додаткові модулі для будь-яких інших мов.

3.2 Розробка структури

На цьому етапі розробки визначається логічна структура веб-сайту. Створюється навігаційна схема, формуються основні розділи та підрозділи, встановлюється їх взаємозв'язок та ієрархія.

Ретельне проектування структури є вкрай важливим, адже від цього залежить зручність навігації, спосіб подачі інформації, а також загальний користувацький досвід і ефективність функціонування сайту.

Структура веб-сайту "*Urban*" включатиме такі розділи:

- *про нас:* 1) хто ми такі — інформація про заклад, його історію та цінності; 2) що ми пропонуємо — опис гастрономічної концепції та унікальності страв; 3) категорії страв — представлення основних категорій меню (бургери, піца, барне меню, закуски).

- меню: 1) основне меню — перелік усіх страв з описами; 2) сніданки — різноманітні варіанти сніданків та склад інгредієнтів; 3) обіди — добірка страв для обіду на будь-який смак; 4) барне меню — напої та легкі закуски.

- доставка: сервіси доставки, які дозволяють замовити страви додому або в офіс.

- галерея: 1) фото закладу — візуальна презентація інтер'єру та атмосфери; 2) презентація нових страв — регулярні оновлення з фото новинок та акцій; 3) різні заходи — знімки подій, що проходять у закладі.

- контакти: 1) інформація про заклад — адреса, години роботи, контактні дані та соціальні мережі; 2) місце розташування на карті — інтеграція карти з можливістю прокладання маршруту.

- футер сайту: 1) навігація по сайту — посилання на основні розділи для швидкого переходу; 2) контактні дані — дублювання контактів для зручного доступу з будь-якої сторінки; 3) логотип — візуальний елемент, що підсилює брендинг і впізнаваність.

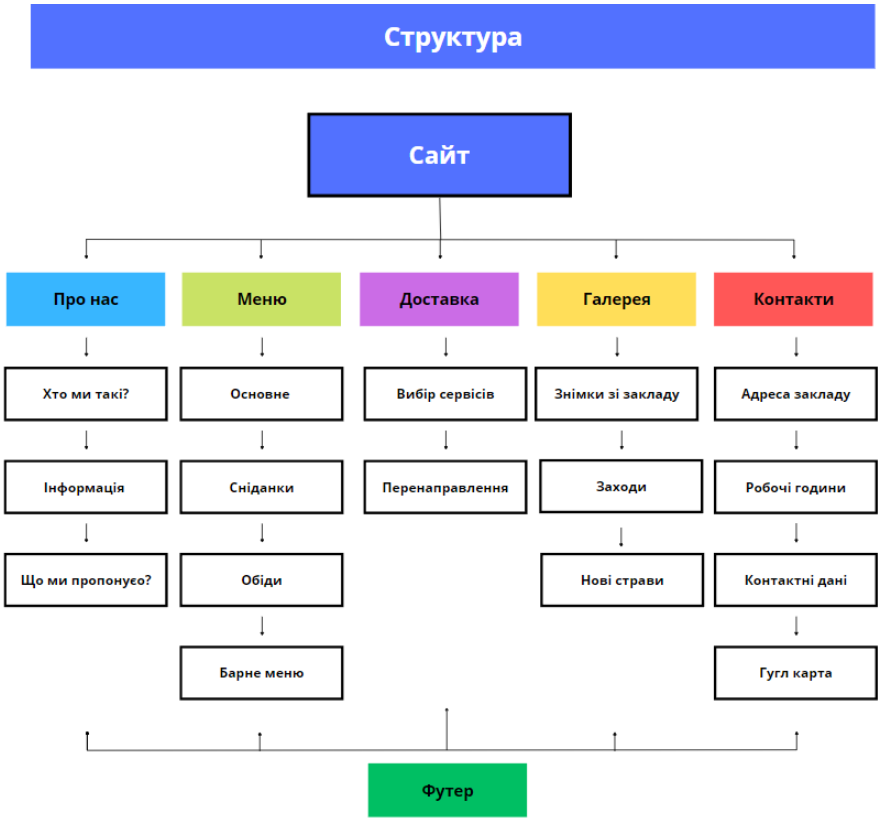

Рисунок 3.1 – Структура сайту

Ця структура сайту сприятиме зручній навігації, дозволяючи користувачам швидко орієнтуватися у вмісті, знаходити потрібну інформацію, переглядати меню та ознайомлюватися з послугами, які ми пропонуємо.

3.3 Результати створення макету веб-сайту

На цьому етапі розробки веб-сайту "Urban" створюється візуальний макет, який допоможе визначити зовнішній вигляд та організацію інтерфейсу сайту. Для створення макету використовується векторний онлайн-сервіс розробки інтерфейсів та прототипування Figma.

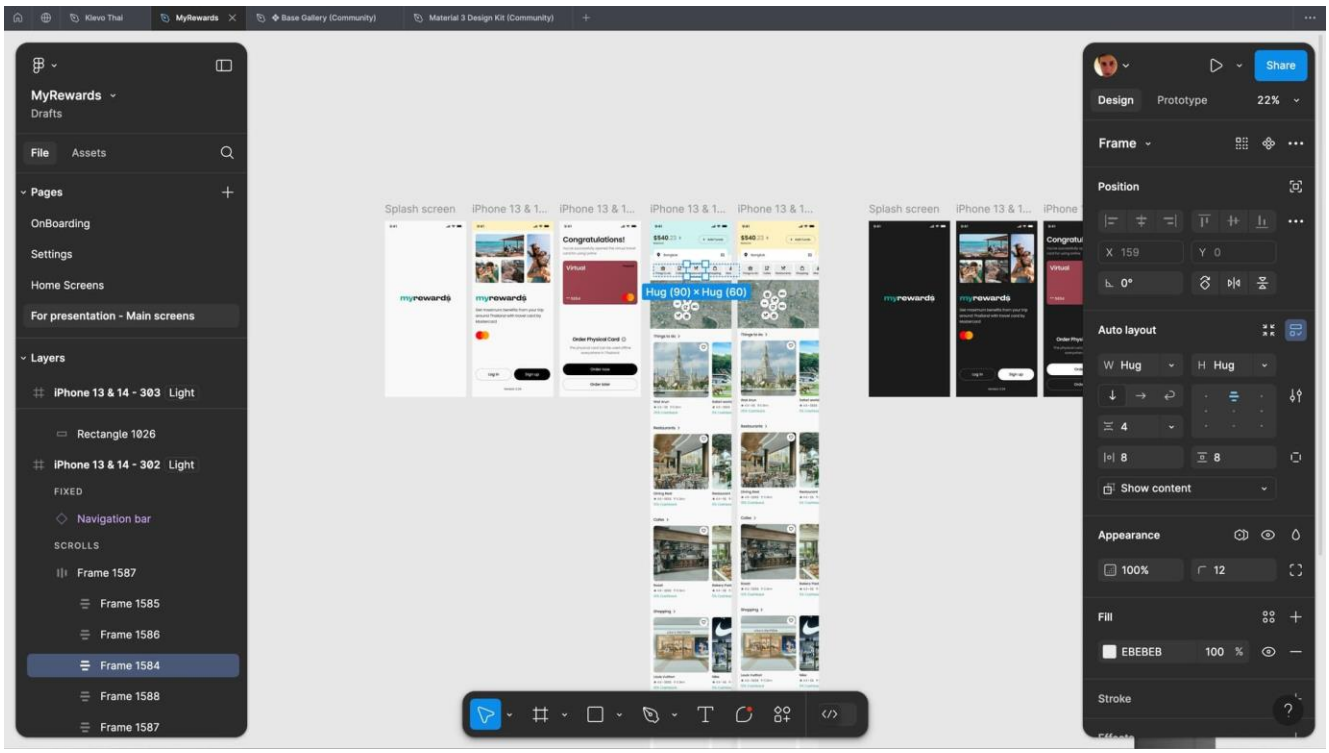

Рисунок 3.2 – Головна сторінка застосунку Figma

Основне завдання розробки макету полягає у візуальному представленні структури та ідеї веб-сайту. На початковому етапі дизайнер аналізує цільову аудиторію, вивчає потреби користувачів, проводить дослідження конкурентів і формує загальну концепцію дизайну, яка буде узгоджена з цілями та філософією бренду "Urban".

Головна сторінка створюється так, щоб мати естетично привабливий вигляд і викликати інтерес у відвідувачів, демонструючи ключову інформацію про заклад і мотивуючи досліджувати інші розділи. Серед основних елементів головної сторінки — логотип "Urban", меню навігації, відеоблок з яскравими візуальними кадрами, а також короткий опис закладу та його послуг.

Щодо внутрішніх сторінок, то для них розробляються макети, які відображають компонування таких елементів, як заголовки, тексти, кнопки, зображення та інші візуальні деталі. Дизайнер визначає палітру кольорів, типографіку та інші стилістичні параметри, що формують цілісний і впізнаваний образ сайту.

Готовий макет подається на розгляд замовнику та команді розробників для

перевірки, обговорення й подальшого коригування. Після погодження цей макет служить базою для наступного етапу — розробки функціоналу та впровадження дизайну в робочий сайт.

Макет відіграє важливу роль у процесі створення сайту "Urban", оскільки дозволяє візуалізувати зовнішній вигляд сторінок і зрозуміти логіку взаємодії користувача з інтерфейсом.

3.4 Реалізація основних функціональних можливостей

Цей етап зосереджений на реалізації основних функціональних можливостей веб-сайту "Urban", зокрема створенні інтерактивних елементів, що забезпечують комфортне використання ресурсу. Основна мета — надати користувачам можливість ефективно взаємодіяти з сайтом.

Серед ключових завдань — створення елементів взаємодії, таких як кнопки, гіперпосилання, меню, слайдери, контактні форми тощо. Розробляється логіка роботи кожного з цих компонентів, програмується їхня поведінка при взаємодії з користувачем.

Функціональна частина є складною та відповідальною, адже потребує уважного опрацювання деталей, знань у сфері програмування та глибокого розуміння запитів аудиторії. Команда розробників тісно співпрацює з дизайнерами та іншими фахівцями, щоб досягти бажаного результату — сучасного, зручного та функціонального сайту.

Функціональні можливості сайту "Urban" включатимуть наступне:

1. Інтерактивні розділи сайту — меню навігації дозволяє переходити між сторінками, такими як "Про нас", "Меню", "Галерея" та інші.

2. Відеоконтент — на сайті буде розміщене відео, яке відображає атмосферу закладу та його концепцію.

3. Класифікація страв — користувачі зможуть обирати окремі категорії страв (наприклад, бургери, піца) та переглядати відповідні пропозиції.

4. Галерея зображень — інтерактивна фотогалерея дасть можливість переглядати фото інтер’єру та подій із можливістю збільшення зображень.

5. Інтеграція з Google Maps — карта із зазначенням місцезнаходження закладу і можливістю прокладання маршруту.

6. Зручна навігація між розділами меню — реалізовано логіку, яка дозволяє користувачам легко переходити між різними частинами меню.

7. Збільшення фото страв — реалізовано функцію перегляду збільшених зображень страв, щоб користувачі могли детально розглянути вміст і зробити вибір перед замовленням.

Ці функції роблять веб-сайт "Urban" не лише візуально привабливим, а й практичним, забезпечуючи простоту у користуванні та позитивний досвід взаємодії.

3.5 Верстка та оформлення веб-сайту

Цей етап є одним із ключових у процесі створення веб-сайту в межах моєї кваліфікаційної роботи. На цьому кроці я використовую HTML для створення структури сторінок, а CSS — для їх візуального оформлення. HTML дозволяє вибудувати логічну ієрархію контенту, організовуючи сторінки за допомогою таких елементів, як заголовки, абзаци, списки, таблиці та посилання.

За допомогою CSS визначаються зовнішні характеристики сторінок: кольорова гама, типографіка, розміщення елементів, відступи, розміри тощо. Таким чином, HTML відповідає за структуру, а CSS — за стилізацію, що разом дозволяє створити привабливий, адаптивний і зручний інтерфейс для користувача.

На цьому етапі важливо врахувати дизайн-макет, підібрану кольорову палітру, відповідні шрифти та візуальні елементи, щоб досягти гармонійного і привабливого вигляду сторінок.

Усі подальші кроки щодо створення веб-сайту "Urban" реалізуються в середовищі Visual Studio Code. Цей редактор забезпечує комфортне програмування, дозволяє зручно зберігати й структурувати файли проекту, а також має інструменти для відлагодження та попереднього перегляду.

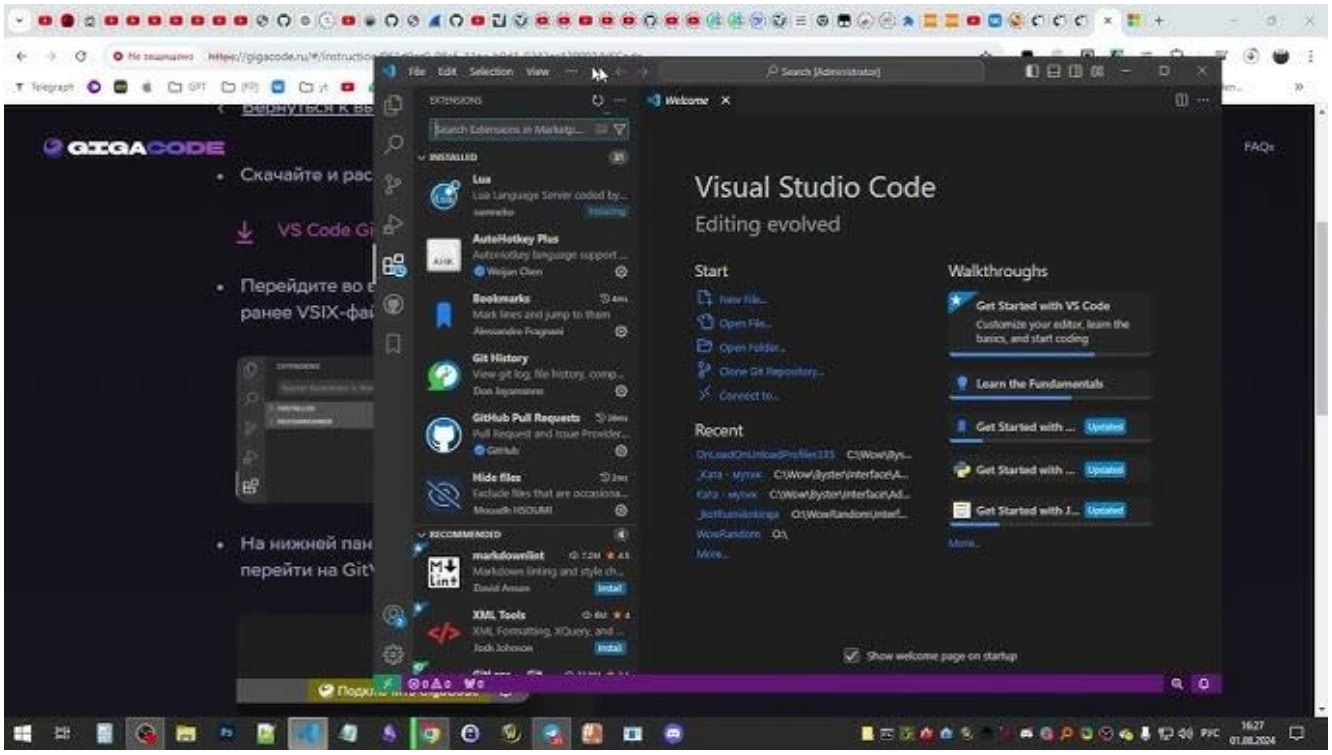

Рисунок 3.3 – Структура проекту в VS Code

Для автоматизації процесів верстки використовується Gulp — інструмент, який допомагає оптимізувати рутинні завдання: компіляцію CSS/JS, зменшення розміру зображень, оптимізацію ресурсів для продакшн-версії. Завдяки Gulp розробка проходить швидше та ефективніше.

Щодо шаблонізації, для веб-проєкту "Urban" обрано Twig — сучасний і гнучкий шаблонізатор, який дозволяє створювати динамічні шаблони сторінок. Використання Twig допомагає централізовано керувати повторюваними елементами сайту — заголовками, меню, списками страв тощо.

Twig підтримує компонентний підхід до розробки, коли сторінка поділяється на окремі, незалежні блоки (компоненти), які мають власну логіку та стилізацію. Кожен із них — хедер, футер, галерея, навігація — розміщується

окремо та підключається до потрібних сторінок. Це забезпечує консистентність інтерфейсу та пришвидшує подальше оновлення або масштабування проєкту.

Таким чином, Twig дозволяє будувати ефективну, гнучку і багаторазово використовувану кодову базу, що полегшує супровід сайту.

Окрему увагу приділено семантичній верстці — вона базується на використанні HTML-тегів відповідно до їхнього призначення. Наприклад, теги <header>, <nav>, <main>, <footer> задають логічну структуру сторінки, а <article>, <section>, <aside> використовуються для оформлення змісту. Заголовки <h1>–<h6> застосовуються відповідно до їхнього рівня значущості.

Семантична верстка покращує розуміння структури сайту як для розробників, так і для пошукових систем, що сприяє SEO-оптимізації й підвищенню доступності [18].

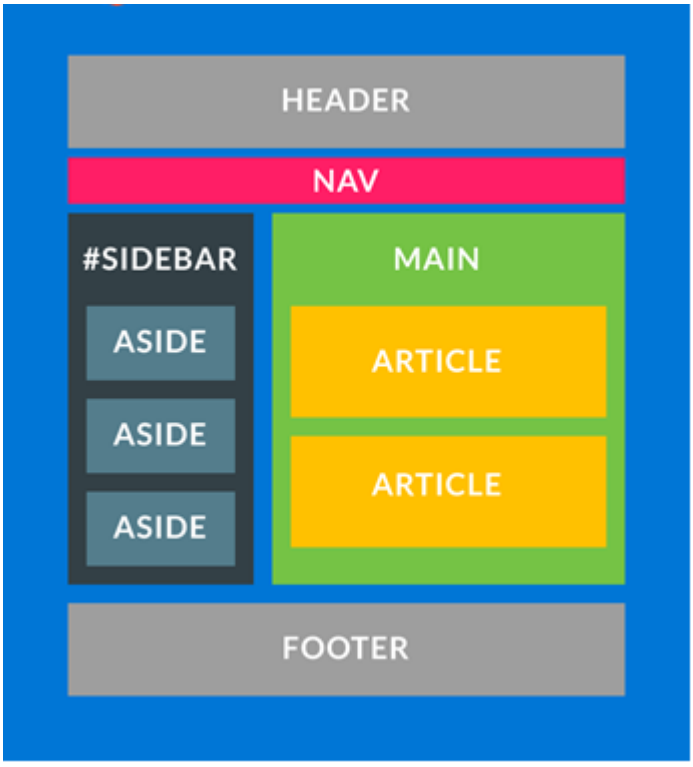

Рисунок 3.4 – Семантична HTML -розмітка

На головній сторінці сайту "Urban" перший блок — це хедер, який містить посилання на соціальні мережі та меню з переходами до розділів сайту. Це дозволяє користувачам швидко знайти потрібну інформацію й взаємодіяти з брендом через соцмережі.

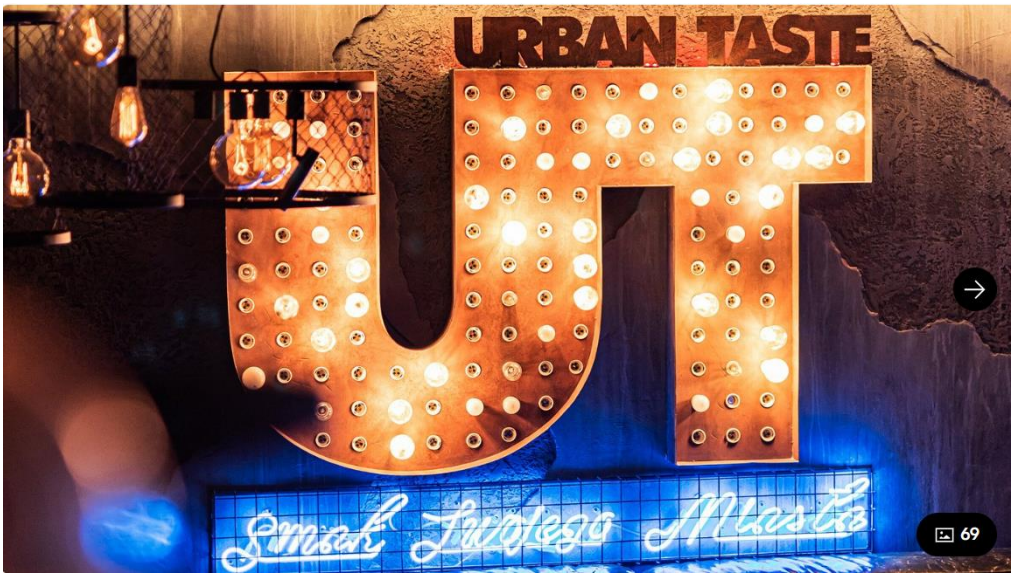

Рисунок 3.5 – Головна сторінка сайту

Нижче за хедером розташовується головний банер, реалізований у форматі відео. Він демонструє атмосферу закладу за допомогою динамічних зображень і привертає увагу користувача з перших секунд, формуючи перше враження про ресторан "Urban".

Далі представлений HTML-код, який використовується для створення хедера:

```
<header class="header">
    <nav class="nav">
      <ul class="nav-list">
        <li class="nav-item"><a href="#" class="nav-link">Про нас</a></li>
        <li class="nav-item"><a href="#" class="nav-link">Меню</a></li>
        <li class="nav-item"><a href="#" class="nav-link">Доставка</a></li>
        <li class="nav-item"><a href="#" class="nav-link">Галерея</a></li>
        <li class="nav-item"><a href="#" class="nav-link">Контакти</a></li>
      </ul>
    </nav>
    <div class="social-icons">
      <a href="#" class="social-icon"><i class="fab fa-facebook"></i></a>
      <a href="#" class="social-icon"><i class="fab fa-instagram"></i></a>
    </div>
  </header>
     CSS код для стилізації хедера:
.header {
  background-color: #000;
  padding: 10px;
  color: #fff;
}
.nav {
```

```
  display: flex;
  justify-content: space-between;
  align-items: center;
}
.nav-list {
  list-style: none;
  display: flex;
}
.nav-item {
  margin-right: 15px;
}
.nav-link {
  color: #fff;
  text-decoration: none;
  font-weight: bold;
  transition: color 0.3s;
}
.social-icons {
  display: flex;
  align-items: center;
}
.social-icon {
  color: #fff;
  margin-left: 10px;
}
```

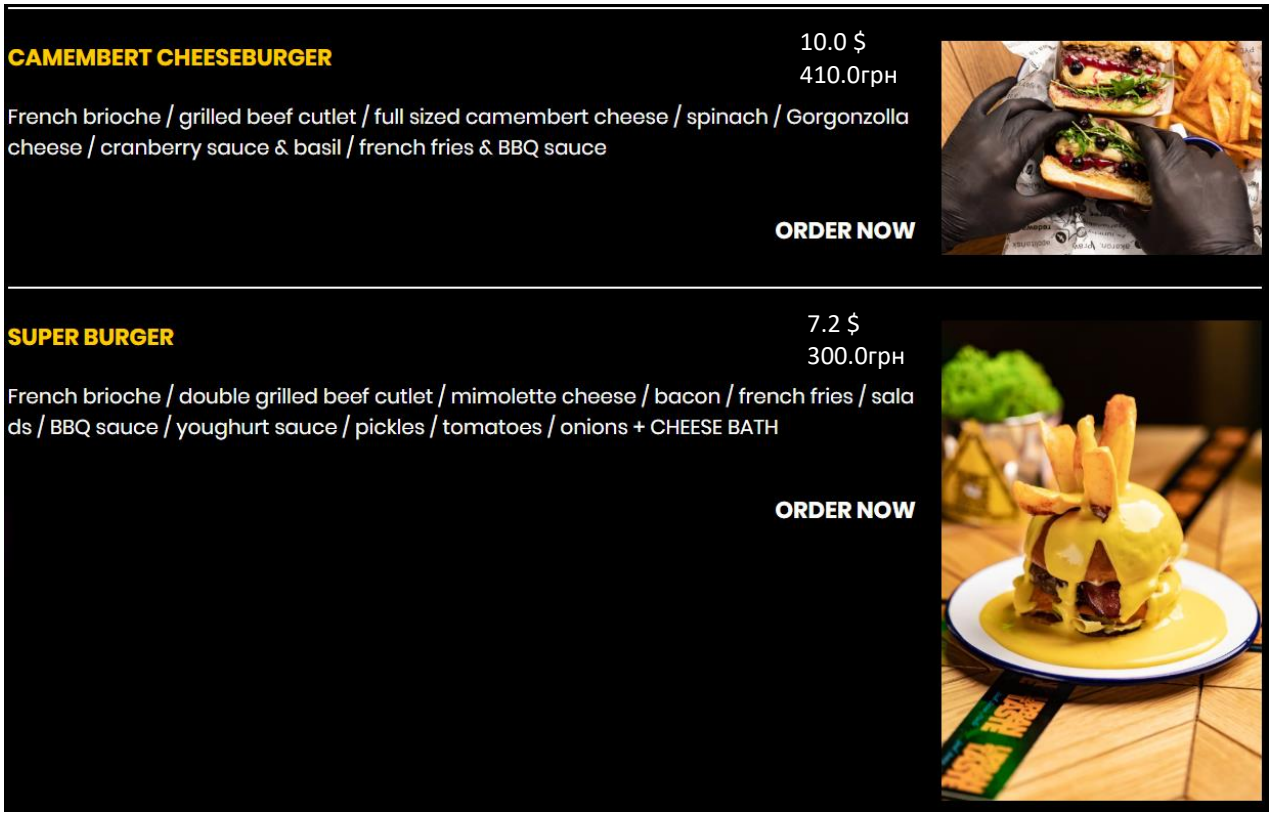

Рисунок 3.6 – Блок меню закладу

Поданий код є демонстраційним прикладом і може відрізнятися від

остаточної версії, яка буде сформована після завершення етапів тестування та оптимізації.

Розмітка меню реалізується за допомогою елементів списку <ul>, де кожен пункт представлений окремим блоком. Кожен з елементів меню включає такі компоненти: зображення, назву страви, її короткий опис, вартість, а також функцію для перегляду зображення у збільшеному форматі.

У наведеному прикладі кожна позиція меню оформлена як елемент списку <li>. Зображення страви розміщується всередині цього елемента за допомогою тегу <img>, де атрибут src вказує на шлях до зображення, а alt — на текстовий опис страви, який відображається у випадку недоступності зображення.

Назва кожної страви розміщується у заголовку <h3>, її опис — у абзаці <p>, а вартість вказується всередині елемента з класом price. Додатково реалізовано посилання <a> з класом zoom, яке дає можливість переглянути зображення у збільшеному вигляді.

Щоб додати нові пункти меню, достатньо повторити структуру елемента <li> з відповідними даними. Такий підхід до верстки дає змогу організувати меню у вигляді структурованого списку з візуальним супроводом, описами та цінами, а також інтерактивною функцією перегляду зображень.

Завдяки використанню шаблонізатора Twig можна автоматизувати створення подібних елементів меню — достатньо реалізувати цикл, який динамічно виводитиме повторювану HTML-структуру на основі масиву даних.

Ось приклад реалізації такої ітерації в Twig:

```
<ul class="menu">
    {% for item in menuItems %}
        <li>
            <img src="{{ item.image }}" alt="{{ item.name }}">
            <h3>{{ item.name }}</h3>
            <p>{{ item.description }}</p>
            <span class="price">Ціна: {{ item.price }} грн</span>
                  </li>
    {% endfor %}
</ul>
```

У даному прикладі змінна menuItems являє собою масив або колекцію об'єктів, що містять інформацію для формування кожного пункту меню. Завдяки

використанню циклу {% for %} у шаблонізаторі Twig, кожен об'єкт item з цієї колекції обробляється по черзі, генеруючи відповідний HTML-фрагмент для кожного елемента меню. Такий підхід дозволяє зручно та швидко повторювати структуру елемента для численних позицій меню на основі вхідних даних.

Дизайн головної сторінки сайту передбачає такі складові:

1. *Блок "Про нас"* — містить текстову інформацію, розміщену у тегу <p>, контактні дані оформлені за допомогою відповідних HTML-елементів (наприклад, <address> для адреси, <a> — для номерів телефонів і email). Для емоційного підсилення можуть використовуватись іконки або графічні символи.

2. *Блок "Меню" або "Категорії"* — включає список категорій страв, таких як бургери, піца, закуски тощо. Кожна категорія має інтерактивні властивості: клікабельність та візуальний ефект збільшення при наведенні курсору, що підкреслює можливість взаємодії.

3. *Блок "Галерея"* — демонструє актуальні фотографії закладу. Галерея автоматично змінює зображення, дозволяючи користувачу переглядати їх без втручання, а також має навігаційні стрілки для ручного перемикання. Код реалізації цієї галереї наведено у додатках.

4. *Блок з контактною інформацією та інтегрованою картою Google* — забезпечує наочне уявлення про розташування закладу, маршрути, режим роботи та контактні дані. Усі контактні елементи є активними посиланнями для швидкої взаємодії, а сама розмітка відповідає сучасним вимогам зручності користування.

Завдяки поєднанню цих блоків головна сторінка веб-сайту має привабливу структуру, забезпечує ефективну навігацію та є інформативною для відвідувачів.

Дизайн адаптовано під різні типи пристроїв, що гарантує коректне відображення вмісту як на комп’ютерах, так і на мобільних гаджетах. Це дозволяє сайту залишатися зручним та функціональним незалежно від розміру екрана користувача.

Фінальна реалізація сторінки з меню передбачає наступну структуру: у лівій частині знаходиться список категорій страв, які є активними. Натиснувши на

певну категорію, сторінка плавно прокручується до відповідного розділу з асортиментом страв.

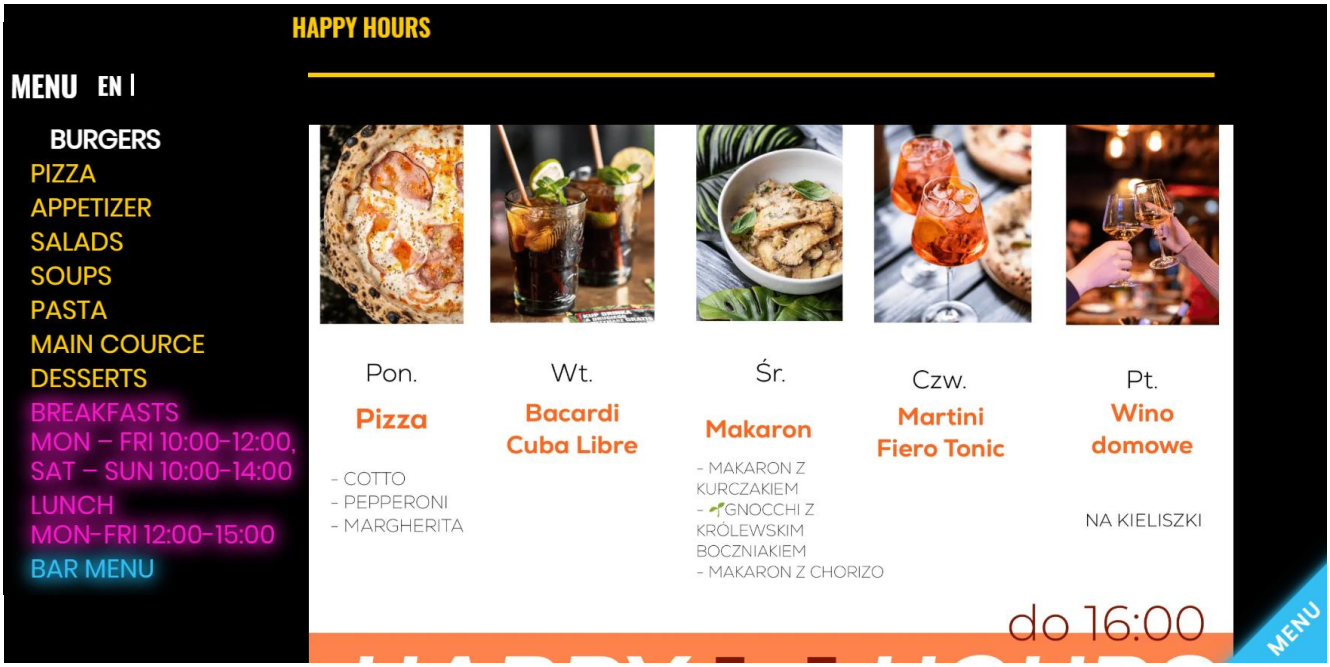

Рисунок 3.7 – Desktop сторінка меню закладу

У правій частині сторінки розміщується безпосередньо вміст меню — з назвами, короткими описами, вартістю та іншими деталями, що полегшують вибір для відвідувача.

Таке компонування дозволяє забезпечити зручний доступ до різних частин меню, логічну організацію сторінки та покращений користувацький досвід. Навігація стає інтуїтивно зрозумілою, а структура сторінки — логічно впорядкованою.

Після реалізації цього етапу веб-сайт набуває повноцінного зовнішнього вигляду. За допомогою HTML і CSS реалізується привабливий, сучасний і функціональний інтерфейс, що відповідає вимогам до проєкту веб-сайту "Urban".

3.6 Тестування та оптимізація структури веб-сайту

У процесі оптимізації веб-сайту були реалізовані заходи, що забезпечили

максимальні показники продуктивності, зручності користування та відповідність сучасним стандартам. Зокрема, було особливу увагу приділено підвищенню ефективності роботи сайту шляхом мінімізації CSS- і JavaScript-файлів, стискання графічних елементів, застосування кешування, а також удосконалення взаємодії з базою даних. Завдяки цьому сторінки завантажуються швидше, а час відповіді системи значно скорочено.

Сайт спроектовано з урахуванням вимог до доступності: він має адаптивний інтерфейс, зручний для перегляду на різних пристроях, та враховує потреби людей з обмеженими можливостями, що покращує загальний користувацький досвід. Також при розробці використовувались сучасні підходи до програмування — зокрема, компонентна структура, шаблонізація за допомогою Twig та впровадження найкращих практик у написанні коду, що зробило проект структурованим, легким для обслуговування й масштабування.

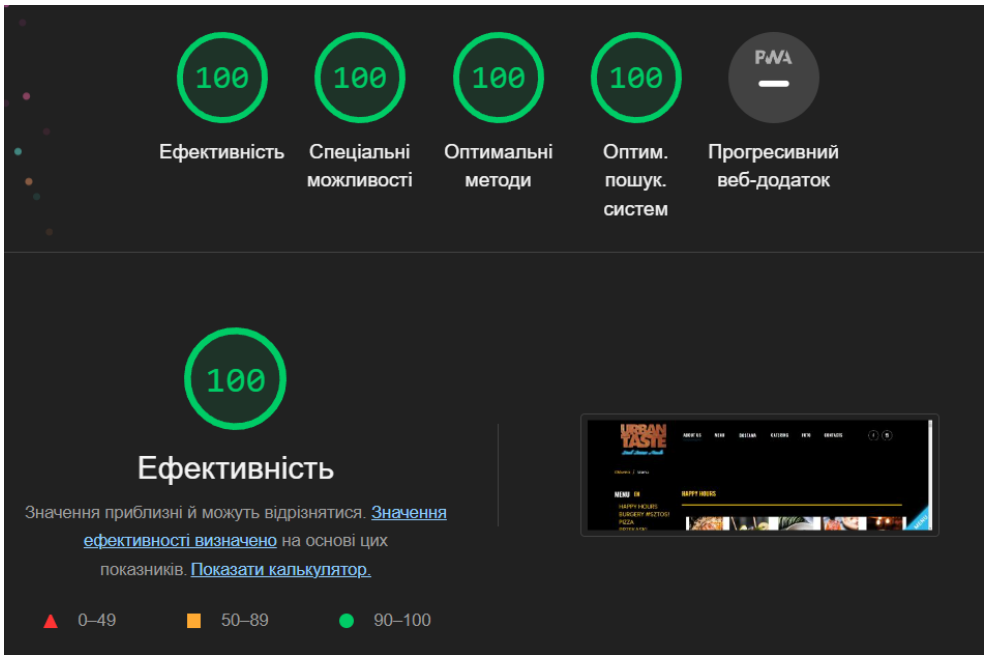

Рисунок 3.8 – Звіт з оптимізації веб-сайту

Окремим етапом стала SEO-оптимізація. Була реалізована семантична верстка, належне використання мета-тегів, оптимізовано заголовки, а також впроваджено релевантні ключові слова. Це дозволило покращити видимість сайту у пошукових системах і посприяло підвищенню позицій у видачі.

У підсумку проведених робіт вдалося забезпечити високий рівень продуктивності, доступності, структурованості коду та відповідність сучасним вимогам пошукової оптимізації, що робить сайт зручним для користувачів і ефективним з технічної точки зору.

3.7 Розгортання та підтримка

Розгортання веб-сайту розпочинається з завантаження файлів проекту (HTML, CSS, JavaScript, зображень тощо) на обраний сервер. Для цього може використовуватись FTP-клієнт або інші засоби перенесення файлів. Далі слід авторизуватись у системі управління контентом (CMS), що використовується в проекті, створити або обрати відповідний сайт, після чого імпортувати завантажені матеріали.

Після перенесення ресурсів у CMS виконується налаштування сайту: створюються сторінки, налаштовується навігація, додаються текстові блоки, зображення, посилання й інші елементи вмісту. Додатково здійснюється реєстрація доменного імені через відповідного реєстратора та підключення хостингу. Обраний тариф має відповідати потребам проекту, після чого налаштовується сервер, база даних і параметри безпеки.

Ці дії дозволяють створити повноцінну присутність проєкту в Інтернеті та відкривають доступ користувачам до сайту через доменне ім’я.

Після завершення етапу розміщення відбувається фінальне розгортання сайту — він стає доступним у мережі. Для цього налаштовується веб-сервер, прив’язується домен до IP-адреси, активуються протоколи безпеки (зокрема, HTTPS), що забезпечує безпечний доступ до ресурсу.

На наступному етапі відбувається підтримка сайту: оновлення вмісту, усунення помилок, моніторинг працездатності, захист від кіберзагроз та обробка запитів користувачів. Використовуються інструменти аналітики для відстеження

активності, виявлення популярних розділів і постійного вдосконалення функціоналу на основі отриманих даних.

Таким чином, процес розгортання й подальшого супроводу забезпечує стабільну роботу сайту, його доступність та актуальність у довготривалій перспективі.

3.8 Інструкція відвідувача

Взаємодія з сайтом починається з головної сторінки, яка відкривається одразу після переходу за адресою. Першим елементом, який бачить користувач, є динамічний відеобанер. Це відео виконує важливу функцію — воно створює перше враження, передаючи атмосферу закладу, його стиль та концепцію.

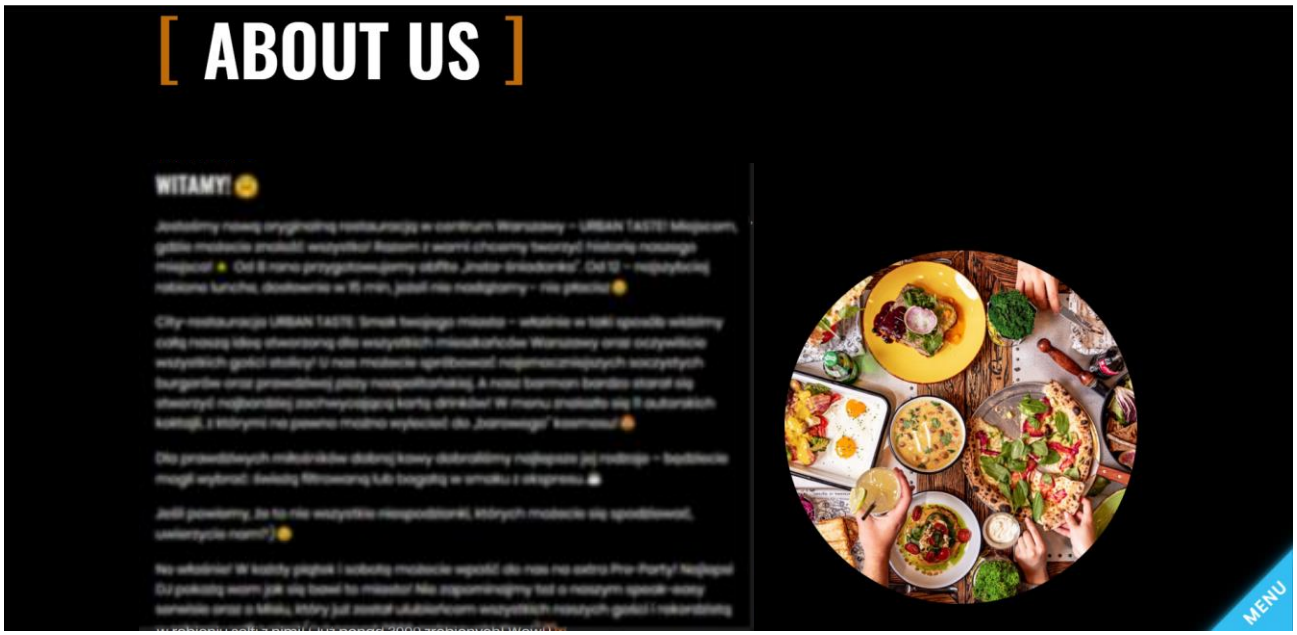

Рисунок 3.9 – Блок про нас на головній сторінці

Під блоком головної сторінки розташовано базову інформацію про заклад: адреса, години роботи, контактні дані. Цей блок дає змогу відвідувачеві швидко зорієнтуватися, де знаходиться заклад і як з ним зв’язатись.

Наступним розділом є категорії страв, що представлені у вигляді клікабельних карток. Кожна картка відповідає певному розділу меню (наприклад,

«Бургери», «Піци», «Напої» тощо). Натискання на категорію здійснює автоматичний перехід на сторінку меню, прокручуючи її до відповідного розділу.

Рисунок 3.10 – Категорії страв

Загальна структура сайту побудована таким чином, щоб забезпечити максимальну зручність навігації. Інтерфейс інтуїтивно зрозумілий: відвідувач легко знаходить потрібну інформацію і може швидко перейти до перегляду страв.

На сторінці меню користувачеві відкривається повний перелік страв з обраної категорії. Кожна позиція містить зображення, назву, опис, склад, а також ціну. При натисканні на фото страви відкривається збільшене зображення, що дозволяє краще роздивитись зовнішній вигляд страви перед оформленням замовлення.

Кнопка «Замовити» відкриває модальне вікно, в якому можна обрати сервіс доставки або резервування. Це значно спрощує процес взаємодії, дозволяючи користувачеві швидко оформити замовлення без необхідності переходу на сторонні ресурси.

Головною метою при створенні сайту було розробити зручну та функціональну онлайн-платформу для перегляду меню. Усі додаткові елементи дизайну, такі як галерея, блок «Контакти», відгуки тощо — слугують

доповненням, яке формує загальне враження про заклад і сприяє емоційному зв’язку з відвідувачем.

Окремий функціонал, який значно підвищує зручність користування сайтом у самому закладі — QR-код, що веде безпосередньо на сторінку меню. Такий підхід має низку переваг:

Швидкий доступ — не потрібно шукати фізичне меню чи запитувати офіціанта.

Актуальність — меню на сайті може оперативно змінюватись без потреби у друці нових версій.

Інформативність — відвідувач бачить повну інформацію про страви: опис, склад, фото.

Екологічність — відмова від паперових меню допомагає зменшити використання ресурсів.

Такий функціонал є простим, але дуже ефективним інструментом взаємодії з користувачами та сучасним рішенням для ресторанного бізнесу.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Аналіз структури та функцій технологічного процесу

Своєчасне виявлення небезпек має вирішальне значення для розробки та впровадження ефективних заходів щодо запобігання нещасним випадкам і травмам. Оскільки не завжди можливо виявити небезпечні умови заздалегідь, а вивчення небезпечних дій може вимагати значного часу для збору відповідних даних, методи виявлення цих небезпек повинні бути відповідним чином диференційовані

На основі аналізу небезпечних умов, присутніх у виробничому процесі, виділяють наступні групи відповідно до їх впливу на працівників:

1. Вплив на обладнання, де оцінюється стан або рівень небезпеки, пов'язаний з використовуваним обладнанням.

2. Створення умов, що сприяють виникненню технологічних помилок у обслуговуючого персоналу під час виробничого процесу.

3. Створення умов і можливостей для проникнення працівників у небезпечні зони.

4. Причини небезпечних дій, що виникають через недостатню підготовку працівників та організацію навчання з охорони праці.

Моделі, що ілюструють формування та виникнення травмонебезпечних та аварійних ситуацій в комп'ютерному залі, представлені у вигляді моделі, що зображає процес їх формування та виникнення [8].

Відповідно до аналізу небезпечних умов, які існують у виробничому процесі виокремлено такі наступні за характером дії на працівника їх групи [8]:

- характеризують стан або рівень небезпеки обладнання, які використовуються.

- сприяють виникненню технологічних помилок обслуговуючого персоналу впродовж виробничого процесу;

- створювати умови та можливість проникнення працівника в небезпечну зону;

- приводять до виникнення небезпечних дій (внаслідок низького рівня професійної підготовки працівників та організації навчання з охорони праці).

Таблиця 4.1 – Моделі становлення та виникнення небезпечних ситуацій

Вид робіт, виробничий підрозділ, робоче місце, виробниче обладнання, склад агрегату	Виробнича небезпека			Можливі наслідки	Заходи запобігання небезпечним ситуаціям
	Небезпечна умова (НУ)	Небезпечна дія (НД)	Небезпечна ситуація (НС)		
Виконання робіт із електрообладнанням	Не вимкнено живлення. Відсутність заземлення.	Нехтування правилами ТБ	Ураження струмом	Травма (Т)	Проведення повторного інструктажу з ТБ. Розробка нових способів захисту. Встановлення заземлення.
НД ↓ НУ → НС → Т					

Моделі формування та виникнення травмонебезпечних і аварійних ситуацій в комп'ютерному кабінеті представлено у вигляді моделі формування та виникнення травмонебезпечних і аварійних ситуацій – табл. 4.1.

4.2 Електробезпека в офісі чи закладі

Промислові приміщення можна освітлювати штучним або природним світлом. Природне світло, якщо воно належним чином організоване, є найбільш сприятливим для людини. Основні вимоги до освітлення включають:

1 Достатня освітленість для швидкого і легкого розпізнавання об'єктів роботи.

2 Рівномірне освітлення без різких тіней.

3 Відсутність сліпучого світла, що може шкодити працівнику.

4 Неприпустимість обмеження рівня освітленості в залежності від часу.

Розрахунок штучного освітлення починається з визначення висоти розташування світильників і їх кількості. Висоту (м) розташування світильників над робочим місцем знаходимо за допомогою такої формули:

$$h_н = H - \left(h_1 + h_2\right), \tag{4.1}$$

де H – висота приміщення, м; h_1 – відстань від підлоги до поверхні освітлення, м; h_2 – віддаль від стелі до світильника, м.

$$h_н = 4,5 - \left(2,2 + 1,5\right) = 0,8 \text{ м}.$$

Якщо світильники розташовані симетрично на вершинах квадратів, їх кількість можна визначити за допомогою наступної формули:

$$n_c = \frac{S_n}{l^2}, \tag{4.2}$$

де l – віддаль між світильниками, м.

Підставивши значення отримаємо:

$$n_c = \frac{36}{9} = 4 \text{ од}.$$

Тоді світловий потік буде становити:

$$F_л = \frac{1,3 \cdot 36 \cdot 300}{4 \cdot 0,25 \cdot 0,545} = 2576,2 \text{ Лк}.$$

З урахуванням світлового потоку 2576,2 Лк, ми обираємо тип лампи – LED-трубка T8 (G13) довжиною 1200 мм, яка замінює люмінесцентну 40 Вт. Як варінат для вибору може бути – Philips CorePro LEDtube 1200mm 16.5W 840, або Osram SubstiTUBE Advanced T8 1200mm 16.8W.

4.3 Забезпечення безпеки в умовах надзвичайних ситуацій

Забезпечення безпеки населення та території в умовах загроз та надзвичайних ситуацій є одним з найважливіших завдань держави. Захист населення передбачає систему масштабних заходів, які реалізуються центральними та місцевими органами виконавчої влади, органами виконавчої влади, управлінням надзвичайних ситуацій та цивільного захисту, підпорядкованими системами та підприємствами, які забезпечують широкий спектр організаційних, технічних, санітарних, гігієнічних, епідеміологічних та інших заходів у сфері запобігання та ліквідації наслідків надзвичайних ситуацій [8].

Загрози життєвим інтересам громадян, держави та суспільства поділяються на зовнішні та внутрішні. Вони можуть виникати внаслідок різноманітних природних катастроф, техногенних аварій та військових конфліктів. Принципи захисту базуються на основних положеннях Женевської конвенції про захист жертв воєнних дій та її Додаткових протоколів з урахуванням специфіки можливих бойових дій та реальних можливостей держави забезпечити матеріальну базу для захисту. Приймаються системні заходи з метою ефективного захисту населення та зменшення збитків у разі виникнення надзвичайних ситуацій [8].

Одним із важливих аспектів є система попередження та інформування населення, яка досягається через попереднє створення та постійну готовність національних, територіальних та об'єктових систем попередження.

ВИСНОВКИ

Головною метою кваліфікаційної роботи була розробка веб-сайту для закладу, який забезпечує користувачам доступ до меню та можливість здійснення замовлень. Після ґрунтовного вивчення теми та її аналізу було визначено ключові аспекти та поточні вимоги до створення ресурсу. Ми також провели порівняльну оцінку конкурентних рішень, виявивши їхні переваги та недоліки.

У ході реалізації роботи було враховано особливості предметної області. Це дозволило оптимізувати навігацію та користувацький інтерфейс, забезпечивши комфортну роботу відвідувачів сайту. В результаті вдалося реалізувати веб-ресурс, що надає повну інформацію про заклад — контактні дані, локацію, графік роботи, опис, меню з цінами та сервіс оформлення онлайн-замовлень.

Для реалізації структури сайту використовувалася мова гіпертекстової розмітки HTML, що дало змогу створити логічну та впорядковану основу ресурсу. Оформлення зовнішнього вигляду було реалізоване за допомогою каскадних таблиць стилів CSS, що дозволило досягти естетичної привабливості та зручного користувацького досвіду.

Одним із основних інструментів, які забезпечили функціональність сайту, стала мова JavaScript. Завдяки її широким можливостям були реалізовані інтерактивні елементи, такі як форма замовлення та модулі для сервісу доставки, що значно підвищило зручність користування ресурсом.

Розробка здійснювалась у середовищі програмування Visual Studio Code, яке забезпечило комфортну роботу над кодом. Для автоматизації рутинних процесів та підвищення ефективності розробки використовувався інструмент Gulp.

Застосування шаблонізаторів, таких як Twig, у розробці HTML-коду веб-сайтів є доцільним та обґрунтованим інженерним рішенням, яке суттєво підвищує ефективність, масштабованість та підтримуваність проєкту. На основі проведеного аналізу можна сформулювати наступні технічні переваги використання шаблонізаторів: 1) розділення логіки та представлення (Separation

of Concerns); 2) повторне використання компонентів (DRY-принцип); 3) спрощення обслуговування та масштабування проєкту; 4) підвищення безпеки; 5) зрозумілий синтаксис і висока читабельність; 6) гнучкість у роботі з динамічними даними.

Оптимізація веб-сайтів для пошукових систем (SEO) є критично важливою складовою ефективної онлайн-присутності будь-якого ресурсу. З позиції програмного інженера, слід зазначити, що якісне впровадження SEO-стратегій безпосередньо впливає не лише на видимість сайту у пошуковій видачі, але й на загальну архітектуру, швидкість роботи, безпеку та структуру даних.

Технічна SEO-оптимізація, зокрема правильна реалізація семантичної верстки, коректне використання метатегів (title, description, alt), адаптивність інтерфейсу, мінімізація ресурсоємних запитів і правильне формування URL-структури, суттєво підвищують ранжування сторінок у результатах пошуку та покращують індексацію ботами.

У результаті виконаної роботи був реалізований повноцінний веб-сайт, який відповідає всім поставленим вимогам. Користувачі можуть швидко знайти необхідну інформацію, ознайомитися з меню, цінами, а також зробити онлайн-замовлення через доступний сервіс доставки.

Загалом, проєкт підтвердив рівень набутих знань та практичних умінь, продемонструвавши нашу здатність до аналізу, проєктування, розробки та впровадження сучасних веб-рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вандер Воф, М. Responsive Web Design. A Book Apart, 2011. 150 с.

2. Дакетт Д. HTML та CSS, Ексмо, 2021, 480 с.

3. Даддаріо, Дж. HTML5 і CSS3 в реальному світі. SitePoint, 2011. 390 с.

4. Джонсон, Е. Розробка веб-сайтів з використанням Node.js, Express та MongoDB. O'Reilly Media, 2014. 352 с.

5. Ерік Фрімен, Елізабет Робсон. Head First. Програмування JavaScript, 2022, 672 с.

6. Макфарланд, Д. CSS: Практичний посібник. O'Reilly Media, 2017. 840 с.

7. Майер, Е. А. CSS: Визначний посібник: Візуальна презентація для вебу. O'Reilly Media, 2017. 1080 с.

8. Винокурова Л.Е., Васильчук М.В., Гаман М. В. Основи охорони праці: Підручник для проф.-техн. навч. закладів, 2001, 192с.

9. Робін Н. Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS та HTML5. 6-те вид, Прес, 2019, 816 с.

10. Фленаган Д. JavaScript. Докладний посібник. Київ: Науковий Світ, 2023. 722 с.

11. Фріман, Е., Робсон, Е. Head First HTML і CSS: Посібник для самостійного навчання. O'Reilly Media, 2012. 768 с.

12. Розробка веб-сайту з використанням HTML: основи та принципи / під заг. ред. О. В. Петренка. Київ: Видавництво "WebCode", 2020. 200 с.

13. CSS у веб-розробці: стилізація та оформлення веб-сайтів / за заг. ред. І. І. Мельникової ; уклад. М. Ю. Ковальова. Харків: 2021. 180 с.

14. Програмування на JavaScript для веб-розробників: практичні аспекти та приклади / за ред. В. П. Кравченка. Львів: 2022. 220 с.

15. Використання шаблонів Twig у веб-розробці: ефективність та зручність / під заг. ред. О. М. Соколової. Одеса: Видавництво "WebDev", 2022. 160 с.

16. Вікіпедія. JavaScript. URL: https://uk.wikipedia.org/wiki/JavaScript (дата звернення: 05.05.2025)

17. Веб-документація MDN. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript (дата звернення: 25.05.2025)

18. DOU. Рейтинг мов програмування 2025. URL: https://dou.ua/lenta/articles/language-rating-2023/ (дата звернення: 05.05.2025)

19. TWIG. Компілюючий обробник шаблонів. URL: https://twig.symfony.com/ (дата звернення: 25.05.2025)

20. Вікіпедія. Редактор початкового коду. URL: https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%BF%D0%BE%D1%87%D0%B0%D1%82%D0%BA%D0%BE%D0%B2%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D1%83 (дата звернення: 20.05.2025)

ДОДАТКИ

Додаток А

Фрагмент коду сторінки меню

```
    var toc_link = document.querySelectorAll(".js-products-nav");
    var toc_list = document.querySelector(".js-products-list");
    if (toc_list) {
      var all_id_list = toc_list.querySelectorAll("a[href*=toc-]");
    }
    var toc_title = document.querySelectorAll(".c-products__right *[id^=toc]");
    var window_height = window.innerHeight;
    var top_marign = 0.01;
    var bottom_margin = 0.71;
    if (toc_list) {
      function toc() {
        [].forEach.call(toc_title, function (e) {
          var target_bound = e.getBoundingClientRect();
          let target_bound_id = e.id.slice(4);
          let index = 0;
          if( target_bound.bottom > window_height * top_marign && target_bound.top <
    window_height * ( 1 - bottom_margin ) ) {
            for (var i = 0; i < toc_link.length; i++) {
              toc_link[i].classList.remove('is-active');
              if (toc_link[i].getAttribute('href').slice(5) == target_bound_id)
                index = i;
            }
            toc_link[index].classList.add('is-active');
          }
        })
      }
      if (window.matchMedia("(min-width: 768px)").matches){
        toc();
        [].forEach.call(all_id_list, function(e) {
              e.addEventListener("click", function() {
                  setTimeout(function () {
                      window.scrollBy(0, -25);
                      for (var i = 0; i < toc_link.length; i++) {
                toc_link[i].classList.remove('is-active');
              }
              e.classList.add('is-active');
                  }, 1)
              });
          });
      }
    }


var frames_count = 20;
var check_is_scrolling = false;

var all_menu_list = document.querySelectorAll('a[href^="#go-"]');
var all_menu_arr = [].slice.call(all_menu_list);
if (all_id_list) {
  var all_id_arr = [].slice.call(all_id_list);
  var all_anchor = all_id_arr.concat(all_menu_arr);
}
else{
  var all_anchor = all_menu_arr;
```

```
}

all_anchor.forEach( function (item) {
    item.addEventListener('click', function(e) {
        e.preventDefault();

        var element_with_id = document.querySelector(item.getAttribute('href'));
        var coord_y = element_with_id.getBoundingClientRect().top;
        var animation_speed = Math.abs(coord_y / 200);

        var previos_cord;
        var current_cord = coord_y;
        var pole = current_cord >= 0 ? true : false;

        if (!check_is_scrolling) {
          check_is_scrolling = true;

          let scroller = setInterval(function() {
              let scrollBy = coord_y / frames_count;

              if (element_with_id.getBoundingClientRect().top > 100 ||
element_with_id.getBoundingClientRect().top < -100) {
                previos_cord = element_with_id.getBoundingClientRect().top;
                  window.scrollBy(0, scrollBy);
                  current_cord = element_with_id.getBoundingClientRect().top;

                  if (pole && current_cord < 0){
                    coord_y =
document.querySelector(item.getAttribute('href')).getBoundingClientRect().top +
window.pageYOffset;
                    setTimeout(() => window.scrollTo(0, coord_y), 5);
                    clearInterval(scroller);
                    check_is_scrolling = false;
                  }
                  else if (!pole && current_cord > 0) {
                    coord_y =
document.querySelector(item.getAttribute('href')).getBoundingClientRect().top +
window.pageYOffset;
                    setTimeout(() => window.scrollTo(0, coord_y), 5);
                    clearInterval(scroller);
                    check_is_scrolling = false;
                  }
                  else if (previos_cord == element_with_id.getBoundingClientRect().top) {
                    coord_y =
document.querySelector(item.getAttribute('href')).getBoundingClientRect().top +
window.pageYOffset;
                    setTimeout(() => window.scrollTo(0, coord_y), 5);
                    clearInterval(scroller);
                    check_is_scrolling = false;
                  }

              } else {
                coord_y =
document.querySelector(item.getAttribute('href')).getBoundingClientRect().top +
window.pageYOffset;
                  setTimeout(() => window.scrollTo(0, coord_y), 5);
                  clearInterval(scroller);
                  check_is_scrolling = false;
              }
          }, animation_speed);
      }
    });});
```