

МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І
СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА
на тему:

РОЗРОБКА ВЕБ-ДИЗАЙНУ ТА СТВОРЕННЯ ВЕБ-САЙТУ РЕСТОРАНУ

Здобувача вищої освіти першого (бакалаврського) рівня,
4 курсу, групи КБ-19-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології (за галузями)»



(підпис)

Чижевського Максима
Андрійовича

(прізвище, ім'я, по батькові)

Науковий керівник



(підпис)

Грищенко Світлана Миколаївна,
к.пед.н., ст. досл.

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Керівник освітньо-
професійної програми



(підпис)

Федотова - Півень Ірина
Миколаївна, к.т.н, доцент

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Допущено до захисту за рішенням кафедри
протокол від «26» травня 2023 № 13

Завідувач кафедри



(підпис)

Філоненко Михайло Миколайович,
к.ф.-м.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Реєстрація № 7

05.06.2023

(дата)



(підпис відповідальної особи)

Результати перевірки

до захисту

(до захисту, на доопрацювання)

29.05.2023

(дата)



(підпис)

Результат захисту

5 (відмінно)

(оцінка)

15.06.2023

(дата)



(підпис)

Голова комісії:



(підпис)

В.М. Франчук

Члени комісії:



(підпис)

М.М. Філоненко



(підпис)

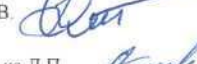
О.В. Редич



(підпис)

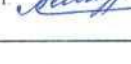
С.В. Котух

БЛАНК ОЦІНЮВАННЯ ДИПЛОМНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії, секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	27		15.06.2023 Франчук В.М.  Філоненко М.М. 
2. Дотримання вимог щодо оформлення	від 0 до 10	7		Редич О.В. 
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	3		Котух Є.В.  Суховерхівка Л.П. 
4. Рецензент	від 0 до 5	5		
РАЗОМ БАЛІВ	від 0 до 50	42		

Оцінювання кваліфікаційної роботи науковим керівником - 50 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підписи)
92	A	5 (відмінно)	15.06.2023	Франчук В.М.  Філоненко М.М.  Редич О.В.  Котух Є.В.  Суховерхівка Л.П. 

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	
50-59	E	3 (задовільно)
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
 Кафедра комп'ютерних та інформаційних технологій і систем
 Спеціальність 122 «Комп'ютерні науки»
 Освітня програма «Інформаційні управляючі системи і технології (за галузями)»

Затверджую

Завідувач кафедри

 Філоненко М.М.
 « 16 » 11 2022

Завдання на кваліфікаційну роботу здобувачу вищої освіти

Чижевському Максиму Андрійовичу

(прізвище, ім'я, по батькові)

Тема випускної кваліфікаційної (дипломної) роботи

Розробка веб-дизайну та створення веб-сайту ресторану

Затверджена наказом ректора від «15» листопада 2023 №1205 (зі змінами наказ від «04» квітня 2023 № 361)

Строк здачі закінченої роботи «28» травня 2023 року

Цільова установка застосування набутих знань, умінь в розробці дизайну та веб-сайту ресторану

Мета роботи розробка веб-сайту для закладу ресторанного господарства.

Об'єкт дослідження проєктування та розробка веб-сайту з використанням фреймворку React.

Предмет дослідження розробка дизайну та веб-сайту ресторану

Назва дослідження Веб-сайт ресторану Anywhere pizza

Вихідні дані до роботи методи та підходи до розробки та створення веб-сайту

Перелік графічного матеріалу головна сторінка веб-сайту ресторану, сторінка вибору піци, сторінка вибору їжі, сторінка замовлення веб-сайту, архітектура ресторану піцерії, схема бази даних веб-сайту, базова структура веб-сайту

Зміст випускної кваліфікаційної (дипломної) роботи

МІСТ

СТУПІ

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.

1. Історія та сучасний стан ресторанного бізнесу в Україні.

2. Аналіз обраних інструментів розробки веб-сайту.

РОЗДІЛ 2. ТЕХНОЛОГІЯ ПРОЄКТУВАННЯ ВЕБ-САЙТУ.

1. Етапи створення веб-сайту та проєктування дизайну сайту.

2. Архітектура та база даних веб-сайту ресторанна.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-САЙТУ

ДІСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	За фактом
1	2	3	
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.11.2022	15.11.20
2	I розділ (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	01.02.2023	01.02.20
3	II розділ (аналіз експериментальної частини про методики досліджень)	03.04.2023	04.04.20
4	III розділ (проведені теоретичні і (або) експериментальні дослідження та проєктні рішення)	10.05.2023	09.05.20
5	Висновки	15.05.2022	15.05.20
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	26.05.2023	26.05.20
7	Отримання відгуку керівника	06.06.2023	06.06.20
8	Отримання зовнішньої рецензії	08.06.2023	08.06.20

8. Дата видачі завдання «16» листопада 2022

9. Керівник випускної кваліфікаційної

(дипломної) роботи


 (підпис)
Грищенко С.М.

(прізвище, ініціали)

10. Керівник освітньо-

професійної програми


 (підпис)
Федотова-Півень І.М.

(прізвище, ініціали)

11. Завдання до виконання отримав

здобувач вищої освіти


 (підпис)
Чижевський М.А.

(прізвище, ініціали)

АНОТАЦІЯ

Розробка веб-дизайну та створення веб-сайту ресторану

Мета роботи полягає в розробці веб-сайту для закладу ресторанного господарства.

Завданнями роботи є:

- Проведення аналізу предметної області,
- Розробку привабливого дизайну,
- Створення функціонального веб-сайту,
- Впровадження адаптивного дизайну.

Предметом дослідження є розробка дизайну та веб-сайту ресторану.

Об'єктом дослідження є проєктування та розробка веб-сайту з використанням фреймворку React.

Базою дослідження виступає веб-сайт ресторану Anywhere pizza.

За результатами дослідження сформульовані методика та етапи реалізації веб-сайту.

Одержані результати можуть бути використані при оптимізації бізнес процесів ресторану та залученню нових клієнтів завдяки використанню веб-сайту.

Кваліфікаційна робота містить 60 сторінок, 0 таблиць, 7 рисунків, список використаних джерел 40 найменувань, 2 додатка.

Ключові слова: ресторан, піцерія, онлайн-замовлення, бізнес, користувацький інтерфейс, дизайн, функціональність.

Рік виконання кваліфікаційної роботи 2023

Рік захисту роботи 2023

ABSTRACT

Development of web design and creation of a restaurant website

The aim of this work is to develop a website for a restaurant establishment.

The tasks of the work are as follows:

- Conducting analysis of the subject area.
- Development an attractive design.
- Creation of a functional website.
- Implementation of responsive design.

The subject of the research is the design and development of a restaurant website.

The object of the research is the planning and development of a website using the React framework.

The research is based on he Anywhere Pizza restaurant website.

The results of the research include the methodology and stages of website implementation.

The obtained results can be used to optimize the restaurant's business processes and attract new customers through the use of the website.

The qualification work consists of 61 pages, 0 tables, 7 images, a list of 40 references, and 2 appendices.

*Keywords*restaurant, pizzeria, online ordering, business, user interface, design, functionality.

Year of qualification work: 2023

Year of work defense: 2023

ЗМІСТ

ВСТУП	9
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	12
1.1. Історія та сучасний стан ресторанного бізнесу в Україні	12
1.2 Аналіз обраних інструментів розробки веб-сайту	20
РОЗДІЛ 2. ТЕХНОЛОГІЯ ПРОЄКТУВАННЯ ВЕБ-САЙТУ	29
2.1. Етапи створення веб-сайту та проєктування дизайну сайту	29
2.2 Архітектура та база даних веб-сайту ресторанна	39
РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-САЙТУ	49
ВИСНОВКИ	54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	56
ДОДАТКИ	61

ВСТУП

Актуальність теми. В наш час склалося так, що комп'ютерна техніка стала невід'ємною частиною сучасного повсякдення. Тепер вона знаходить своє застосування майже у всіх сферах людської діяльності.

З кожним роком у країні зростає кількість ресторанних господарств, зростає попит на надані ними послуги. В умовах жорсткої конкуренції ресторанної справи веб-сайт для ресторанного господарства – це можливість заявити про себе, розповісти про свої конкурентні переваги; послуги, що надаються; а також, це відмінний засіб для залучення широкої клієнтської аудиторії.

У сучасних реаліях наявність сайту ресторану вже не елемент розкоші, а інструмент для збільшення прибутку закладу. Що стосується ресторанного бізнесу, веб-сайт закладу ресторанного господарства дає перевагу перед конкурентами, у яких його немає, або які неефективно використовують веб-сайт для розвитку і просування бізнесу [3].

Створення веб-сайту для ресторану піцерії є важливим елементом маркетингової стратегії, оскільки це дозволяє залучити нових клієнтів та зберегти існуючих. Крім того, веб-сайт може допомогти встановити зв'язок з клієнтами, вести промо-акції, повідомляти про нові страви, що допоможе залучити більше клієнтів. Також веб-сайт дозволяє ресторану піцерії бути в курсі останніх тенденцій у галузі та відстежувати дії конкурентів, що також допоможе вести більш успішний бізнес [15].

Різноманітні інноваційні підходи особливостей розвитку ресторанного бізнесу відображено у працях як вітчизняних авторів Алексєєнка В. С. [3], Лисецького Ю. М [8], Турчиняка М. К., Форися О. Г. [21], так і зарубіжних Hutsulyak O. [33], Oleynik P., Nikolenko O., Yuzefova S. [34], Zemlina Y., Lifirenko O. [40].

Метою кваліфікаційної роботи є розробка веб-сайту для закладу ресторанного господарства.

Для досягнення мети необхідно вирішити такі завдання, а саме:

- Проведення аналізу предметної області, де було проаналізовано ресторанний бізнес та визначено актуальність створення веб-сайту,
- Розробку привабливого дизайну, де в рамках дослідження були вивчені сучасні тенденції у дизайні веб-сайтів ресторану та було розроблено привабливий дизайн, який відображає сучасність та атмосферу ресторану;
- Створення функціонального веб-сайту, де була проведена розробка та реалізація веб-сайту, що включає основну логіку та функціональні елементи, такі як замовлення їжі та перегляд різних видів страв
- Впровадження адаптивного дизайну, де була забезпечена адаптивність веб-сайту, що дозволяє користувачам швидко переглядати та використовувати сайт на різних пристроях, а саме: планшети, комп'ютери та мобільні пристрої;

Об'єкт дослідження – проектування та розробка веб-сайту з використанням фреймворку React.

База дослідження – веб-сайт ресторану Anywhere pizza

Предмет дослідження – розробка дизайну та веб-сайту ресторану.

Методи дослідження – теоретичні – аналіз, узагальнення, систематизація законодавчої бази, матеріалів конференцій, Інтернет-ресурсів для виділення теоретичних засад дослідження; емпіричні – аналіз досвіду роботи в даній галузі, для констатування стану розв'язання проблеми.

Науково-практична важливість отриманих результатів полягає в тому, що розробка веб-сайту ресторану може допомогти покращити ефективність ресторанного бізнесу. Веб-сайт може залучити нових клієнтів та збільшити продажі, а також забезпечити зручний спосіб для клієнтів замовляти їжу онлайн.

Практичне значення отриманих результатів. Отримані результати кваліфікаційного проєкту з розробки веб-сайту ресторану піцерії мають велике

практичне значення для бізнесу. Створення ефективного веб-сайту ресторану піцерії може допомогти залучати нових клієнтів, збільшувати обсяг продажів та зміцнювати бренд компанії. Веб-сайт може бути корисним для ресторанних піцерій та інших закладів громадського харчування, які прагнуть розширити свою онлайн присутність та покращити користувацький досвід своїх клієнтів.

Апробація результатів роботи. Результати дослідження висвітлено в конференції «Цифрова трансформація фінансової системи України та Країн V-4 в умовах євроінтеграції» (м. Дубляни, 26 квітня 2023 р.).

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Історія та сучасний стан ресторанного бізнесу в Україні

Ресторанний бізнес є одним з найрозповсюдженіших видів малого бізнесу, тому для нього є характерною постійна конкурентна боротьба за залучення споживачів і їх утримування як постійних клієнтів. В умовах несприятливої економічної кон'юнктури багато підприємств ресторанного господарства опиняються у важкій ситуації внаслідок зменшення попиту споживачів і одночасного зростання витрат, зокрема, витрат на оплату праці, оренду приміщень, закупку сировини. В результаті конкуренція загострюється і проблема підтримання конкурентоспроможності набуває особливої важливості. Напрямами розв'язання цієї проблеми можуть бути, зокрема, вдалі організаційні рішення, впроваджені інновації, ефективні маркетингові технології.

За даними Міжнародної асоціації готелів і ресторанів у світі існує більше 8 мільйонів підприємств ресторанного господарства, в яких працює 48 млн осіб [10]. Згідно з результатами досліджень щільність закладів ресторанного господарства напряму залежить від рівня доходів населення. Так, якщо в середньому один заклад ресторанного харчування припадає на 477 мешканців, то у країнах з низькими доходами населення, цей показник збільшується до 618 осіб, а у країнах з середніми і високими доходами – зменшується до 268 осіб [12].

Місткість українського ресторанного ринку становить близько 30 млрд грн [18]. Ресторанне господарство України налічує зараз приблизно 5 тисяч підприємств, кожне з яких має в середньому дещо більше двох закладів харчування, проте усі ці заклади складають менше половини усіх закладів громадського харчування в країні (цей відсоток коливався в останні роки на рівні 42–45 %) [12]. Решта закладів належать до підприємств інших видів економічної діяльності (кафе у складі підприємств, установ, ресторани при готелях). Загальна кількість закладів ресторанного господарства за період з 2003 р. змінювалася від

20,6 до 30,6 тисячі, середня кількість посадкових місць в розрахунку на один заклад – від 58 до 72 [12], причому щорічні коливання цих величин свідчать про досить інтенсивні процеси створення нових і злиття (поглинання), реструктуризації й ліквідації діючих закладів. Зокрема саме ці процеси імовірно створюють певну невпорядкованість статистичних даних про галузь ресторанного господарства.

У структурі підприємств ресторанного господарства за останні 15–20 років суттєво зросла частка ресторанів з 9,7 % у 2001 р. до теперішнього рівня у майже 20 відсотків. Натомість скоротилася приблизно на 5 % частка їдалень, причому це скорочення відбулося у першому десятиріччі XXI століття, а за останні 10 років їх частка залишається стабільною на рівні 9–10 %. Стабільною на рівні 15–16 % є частка барів. Дещо скоротилася частка кафе з 60 до 55 %. [12]. За іншими даними у структурі підприємств ресторанного господарства 46 % закладів типу кафе і ресторани, 40 % закладів категорії Fast Food і 14 % припадає на бари, паби та нічні клуби [18]. Розбіжність даних пояснюється неоднозначністю існуючих класифікацій, розмитістю критеріїв віднесення конкретного закладу до тієї чи іншої категорії, нечіткістю самопозиціонування окремих закладів, наприклад, заклади Fast Food можуть одночасно вважатися і їдальнею, і кафе, і рестораном низькоцінової категорії.

Широко розповсюджена в країнах світу сучасна класифікація форматів закладів ресторанного господарства, яка є актуальною сьогодні і для України:

□ street food (вуличний фаст-фуд) – продаж зі стаціонарного павільйону або пересувного автопричепу безпосередньо на людних вулицях міст, переважно зі спеціалізацією на монопродукті, який призначений для споживання на ходу (правило однієї руки – споживач повинен мати можливість утримувати їжу однією рукою);

□ fast food – підприємства швидкого харчування, яке здебільшого відбувається у формі самообслуговування після попередньої оплати, страви готуються з напівфабрикатів високого ступеня готовності;

□ fast casual (швидкий і демократичний) – проміжна ланка між фаст-фудом і демократичним рестораном з обмеженим спектром послуг, з меншою кількістю заморожених інгредієнтів і напівфабрикатів, ніж в ресторанах швидкого харчування, клієнт самостійно обирає закуски з числа виставлених, вказує на гарячі страви, які йому видаються працівниками кухні, розраховується на касі. Подібним є формат free flow (вільний рух), який передбачає вільне переміщення гостей торговим залом із можливістю самостійного вибору страв, що готуються і формуються в їх присутності;

□ quick service restaurant (ресторани швидкого обслуговування) – відрізняються обслуговуванням офіціантами за столиками, займають середнє положення між елітним рестораном і фаст-фудом;

□ casual dining (демократичний обід) – ресторан з обслуговуванням через офіціанта, не орієнтований на пріоритет швидкого обслуговування, але з нижчими цінами, ніж в елітних ресторанах;

□ fine dining (вишуканий обід) – ресторани преміум класу.

Ніша закладів преміум-класу є практично заповненою, тому, на думку експертів [25], суттєвого зростання у цьому сегменті не буде. Попри те, що закладів середнього цінового рівня кількісно більше і конкуренція у цій категорії є більш вираженою, саме таким закладам прогнозуються вищі темпи зростання.

Загострення конкурентної боротьби між підприємствами примушує їх постійно удосконалювати процеси виробництва, маркетингу, реалізації продукції й обслуговування споживачів, впроваджувати науково-технічні інновації, нові форми організації виробництва й обслуговування, підвищувати якість роботи,

продуктивність праці та внаслідок усього цього підвищувати свою конкурентоспроможність.

Останнім часом намітилася стійка тенденція переміщення послуг з організації споживання продукції та обслуговування споживачів із залів закладів ресторанного господарства до робочих місць (офіси, установи); місць відпочинку; місць святкування ювілеїв та інших офіційних та неофіційних святкових подій; до домівки. Ця послуга має назву у міжнародній індустрії гостинності "catering". Послугу з "кейтерингу" здійснюють не тільки заклади ресторанного бізнесу (ресторани, кафе, бари), а також і фірми, створені на базах ресторанів.

Крім того, особливість сьогоденної ситуації полягає у тому, що розвиток ресторанної справи відбувається в комплексі із розвитком транспортного вузла - з/д вокзалу чи автовокзалу, аеропорту. Разом з тим навколо них розбудовується інфраструктура, що включає в себе готелі, магазини, численні кафе та ресторани, автомати з продажу чаю-кави.

Але спалах COVID-19 на початку 2020 року, що перетворився на масштабну пандемію, та подальші карантинні обмеження суттєво вдарили майже по всіх сферах бізнесу як в Україні, так і у світі. За даними Світового банку, глобальна економіка впала на 4,3%, пандемія призвела до маси жертв і хворих, а мільйони людей уввергнула у бідність.

Чи не найбільше від локдауну постраждав ресторанний бізнес. За даними аналітичного центру «Ресторани України», протягом «карантинного» 2020-го кількість ресторанів та кафе в Україні зменшилась майже на 4 тисячі закладів.

На початок 2021 року працювало 14,7 тисячі ресторанів, кафе та барів, у порівнянні з початком 2020 року, коли в Україні працювало 18,6 тисячі закладів. Відповідно об'єм ресторанного ринку скоротився за 2020 рік майже на 30% і склав за 2020 рік 14,1 мільярда гривень. За 2020 рік ресторанний ринок втратив майже 6 мільярдів гривень, як зазначають експерти [15].

Також у 2022 році почалося повномасштабне російське вторгнення, яке надзвичайно вплинуло на українську економіку. Крім очевидних втрат на тимчасово окупованих територіях і в зонах бойових дій, а також пошкоджень інфраструктури, багато підприємств, зокрема ресторанів, припинили роботу або були змушені звільнити персонал. Згідно з даними Української ресторанно-керуючої компанії «Постер ПОС», у перші дні вторгнення в Україні продовжували працювати 19% закладів харчування. У липні ця кількість зросла до 74%. Кафе, кав'ярні, бари дають 82% довоєнного товарообігу. Проте українські інституції прагнуть не лише звести кінці з кінцями та відновити роботу, а й допомогти тим, хто цього потребує, у цей важкий час.

З перших днів воєнного стану українські ресторани та кав'ярні адаптувалися до нової реальності. Заклади, які не могли запропонувати повноцінне харчування, надсилали зарезервовані запаси своїх продуктів нужденним. Багато ресторанів, такі як McDonald's – мережа швидкого харчування, призупинили роботу в Україні з міркувань безпеки після початку вторгнення, розподіляючи продукти зі своїх складів між гуманітарними штабами. Завдяки волонтерським організаціям компанії розпочали роботу допомагаючи постраждалим від війни українцям у партнерстві з фондом «Дім Рональда Макдональда». Наприклад, відправляючи ліки українським лікарням. Водночас українці чекають на відновлення роботи цих мереж. Український офіс McDonald's вже провів переговори з Міністерством закордонних справ України, та відкрився в багатьох областях країни [16].

За словами експертів, Україна після війни, стане туристичною дестинацією номер 1, і відповідно, туристичним клондайком, у тому числі в сенсі гастротуризму. Українська кухня шляхом інтересу до України – вже стає популярною в усьому світі. В Україну зайдуть іноземні інвестори – найбільші ресторанні компанії. Українці, які повернуться з-за кордону і там працювали, привезуть із собою нові навички й знання. Українські ресторани – точно потраплять в поле зору міжнародних ресторанних рейтингів. З'являться нові

економічні можливості від дії угоди про асоційоване членство в ЄС. Україна отримає мільярдні інвестиції на відновлення, в тому числі й для сфери гостинності [6].

Сучасний заклад ресторанного бізнесу стає одним з основних елементів дозвілля. Організація подібного процесу вкрай складна за своїм змістом та наповненням і вимагає: дотримання санітарних і технологічних норм, контроль за стилем і культурою поведінки обслуговуючого персоналу, за обліковим процесом, аналіз транзакцій, облік надходження продуктів, формування вартості страв і напівфабрикатів, процедури списання продуктів. Вимога автоматизації всіх цих процесів впливає, перш за все, з необхідності врахування великої кількості деталей, які узагальнимо до: неможливості наочного контролю керівниками кожного технологічного процесу за умов диверсифікації та відокремленості технологічних процесів; необхідності швидкого прийняття рішень і високої мобільності людських кадрів [23].

Зручність автоматизації та інформатизації процесів на підприємстві громадського харчування очевидна не тільки з точки зору «ведення справ», але і з позицій клієнтів тому, що інформаційні системи дозволяють більш оперативно здійснювати розрахунки з відвідувачами, черговість обслуговування, забезпеченість пропонованого меню усіма необхідними інгредієнтами.

Програми для автоматизації ресторану повинні бути багатофункціональними системами для підвищення прибутковості та зниження витрат підприємства, легкими у впровадженні та простими в обслуговуванні.

Використання функції бронювання столиків через додатки мобільних телефонів та інших електронних девайсів, що є основною темою нашого дослідження та розробки, стає все більш популярним, хоча на заході використання такої функції є популярним понад 15 років. Використання мобільних додатків в ресторанному бізнесі є прикладом автоматизованих систем управління (АСУ). У більшості випадків ресторатори вважають АСУ додатковими

капіталовкладеннями, що є стратегічною помилкою власників, адже капіталовкладення в АСУ – це інвестиції у майбутнє закладу, його розвиток, концепцію, персонал, що в сукупності дозволяє не лише полегшити управління підприємством, але й підвищити його конкурентоспроможність[1].

Заклади ресторанного бізнесу стикаються з проблемами, які заважають їм працювати з максимальною віддачею. Проте ці проблеми вирішуються з мінімальними витратами із впровадженням АСУ. Таким чином основні проблеми та їх негативні наслідки спричиняють певну втрату клієнтів.

Сучасні інформаційні системи, розроблені спеціально для підприємств громадського харчування, дозволяють значно спростити, оптимізувати і прискорити цілий ряд рутинних повсякденних, специфічних для цього бізнесу операцій. Визначальним фактором в ході автоматизації підприємства громадського харчування є комплексний підхід, який визначається правильною організацією бізнес-процесів, сучасним високотехнологічним устаткуванням і надійністю контрольних функцій програмного забезпечення. Цей симбіоз забезпечує ріст прибутковості ресторану, лояльність клієнтів закладу та захист інвестицій, вкладених у підприємство [2].

Інформаційні системи беруть на себе процедуру формування страви на основі набору інгредієнтів і схеми закупівлі продуктів. Автоматизується ведення списку страв з урахуванням нормативів витрат продуктів, сезонних норм закладок продуктів в страви.

Також сучасні програми автоматично визначають витрати інгредієнтів по кожній страві, списують потрібну кількість продуктів і розраховують собівартість страв. І, нарешті, інформаційні системи значно полегшують і роблять більш суворим ведення обліку продуктів і страв на кількох кухнях і точках реалізації, формують меню для залу і преїскурант барної продукції. Шляхом автоматизації спрощується процес руху продукції, наприклад, організація надходження продуктів безпосередньо на кухню або місце реалізації.

Автоматизація підприємства ресторанного господарства в цей час є потребою і необхідністю, через те, що вона набагато спрощує процес роботи і допомагає звести до мінімуму збитки і зайві витрати. В результаті автоматизації підвищується якість сервісу і престиж закладу; забезпечується швидка обробка замовлень і обслуговування відвідувачів; поліпшується узгодженість і контроль за роботою персоналу; ведеться облік руху матеріальних цінностей і руху товарів; скорочуються тимчасові витрати на пошук і аналіз інформації.

Важливість та ефективність використання АСУ пояснюється насамперед можливістю формування єдиної бази даних, інформація в якій постійно систематизується та оновлюється, й на основі якої керівник може приймати економічно обґрунтовані управлінські рішення.

У кваліфікаційній роботі робимо особливий акцент ресторану на піцу.

Піца – одна з найпопулярніших страв у світі. При правильному підході до організації бізнесу, чіткому розумінні, як вести облік в ресторані, заклад може приносити досить високий дохід. Але навіть саме грамотне адміністрування та управління не знижує вплив людського фактора, не дозволяє контролювати «кожну копійку» і «кожен грам». Краще рішення для оперативного розвитку та успішної конкуренції - автоматизована програма для піцерії. Вона забезпечує максимально повний, об'єктивний і точний контроль бізнес-процесів.

Доля піци в Україні склалася вдало: любов до неї скорила усі віки і соціальні верстви населення. «Піцерійний бізнес» популярний, а, отже, вигідний. Перші піцерії почали з'являтися в Україні на початку дев'яностих і швидко стали одним із найбільш популярних форматів громадського харчування. Піца «прижилася» і полюбилася буквально усім: зараз це просто безпрограшна пропозиція. Практично нескінченна кількість варіацій використовуваних топінгов (начинок) перетворює піцу в продукт, що ніколи не приїдається: ресторатори (або самі відвідувачі) завжди вільні винаходити нові види піци [11].

Крім піци клієнтові нашого ресторану буде запропоновано широкий вибір інших продуктів: різних салатів, напоїв, десертів тощо. На думку багатьох фахівців, у піцерії доречно відводити на частку піци до 40% меню. Піцерія — дуже гнучкий формат, здатний розміщати в собі блюда різних кухонь. Обов'язкові в меню піцерії також спиртні напої.

Ефект від розробки веб-сайту ресторану може бути поділений на дві складові: фінансову та якісну. Відповідно основні характеристики фінансової складової – це реальна економія і прибуток, а також: відсутність зловживань працівників, повний контроль за вибором клієнтів, велика кількість страв, введення продуманої політики знижок. Натомість якісна складова, крім підвищення якості роботи піцерії, в кінцевому підсумку також збільшує прибуток.

1.2 Аналіз обраних інструментів розробки веб-сайту

У сучасному світі, де веб-ресурси відіграють важливу роль у бізнесі, потреба у створенні веб-сайтів для підприємств, в тому числі ресторанів і піцерій, стає все більш актуальною. Від веб-сайту залежить не тільки відображення асортименту та цін на продукцію, а й можливість замовлення онлайн, розміщення відгуків, отримання актуальної інформації про ресторан.

Для розробки веб-сайту ресторану піцерії було визначено низку інструментів, які можна використовувати для досягнення мети проекту. Серед цих інструментів:

«HTML – це мова розмітки, яка визначає структуру вашого вмісту. HTML складається з серії елементів, які ви використовуєте для включення або обгортання різних частин вмісту, щоб він виглядав певним чином або діяв певним чином» [32].

HTML (Hypertext Markup Language) є основою веб-розробки і забезпечує структуру та контент веб-сторінок. Ось деякі з плюсів використання HTML:

1. Простота вивчення: HTML є дуже простою мовою програмування, що робить її доступною для вивчення для будь-якого користувача, навіть без досвіду в програмуванні.
2. Стандартизація: HTML має стандартизовану структуру та синтаксис, що забезпечує сумісність з різними браузерами і платформами.
3. Легкість редагування: HTML-документи можна легко редагувати з використанням будь-якого текстового редактора.
4. SEO-оптимізація: HTML може допомогти з покращенням рангу веб-сторінок в пошукових системах завдяки відповідному використанню метатегів і ключових слів.
5. Підтримка мультимедіа: HTML дозволяє вставляти мультимедіа елементи, такі як зображення, аудіо і відео, на веб-сторінки.
6. Крос-платформовість: HTML дозволяє розробляти веб-сторінки для будь-якої платформи, що дозволяє їм бути доступними для широкої аудиторії користувачів.
7. Відкритий стандарт: HTML є відкритим стандартом, що означає, що він вільно доступний для використання та редагування, що забезпечує більшу відкритість та переносимість для розробників

Незважаючи на багато переваг, HTML також має свої недоліки. Ось кілька з них:

1. Обмежена функціональність: HTML забезпечує структуру та контент веб-сторінок, але не надає складних функцій, таких як динамічне оновлення сторінок, обробка форм, та інші.
2. Роздібнена розробка: HTML часто використовується в поєднанні з іншими технологіями, такими як CSS та JavaScript, що може збільшити складність розробки веб-сторінок.
3. Конкуренція з іншими технологіями: HTML не є єдиною технологією, що використовується для розробки веб-сторінок. Існують інші технології, такі як

Flash та Silverlight, які можуть надавати більшу функціональність, але їх використання стає менш популярним.

4. Безпека: HTML може бути вразливим до атак, таких як внедрення коду та скімінг, які можуть призвести до крадіжки даних користувачів.

5. Відсутність розширеності: HTML не надає розширеності на рівні користувача, що означає, що користувачі не можуть додавати свої власні функції або модифікувати веб-сторінки, які вони відвідують [32].

CSS розшифровується як каскадні таблиці стилів. Це мова таблиць стилів, яка використовується для опису вигляду та форматування документа, написаного мовою розмітки. Він надає додаткову функцію до HTML. Зазвичай він використовується з HTML для зміни стилю веб-сторінок та інтерфейсу користувача. Його також можна використовувати з будь-якими видами XML-документів, включаючи простий XML, SVG і XUL.

CSS використовується разом із HTML і JavaScript на більшості веб-сайтів для створення інтерфейсів користувача для веб-додатків і інтерфейсів користувача для багатьох мобільних додатків [38].

CSS має кілька переваг, серед яких:

- Розділення контенту та оформлення: CSS дозволяє розділити контент веб-сторінки від її оформлення, що полегшує підтримку та зміну вигляду сторінки без зміни її змісту.
- Можливість використовувати спільні стилі: CSS дозволяє використовувати спільні стилі для кількох елементів, що дозволяє ефективно використовувати код та полегшує підтримку веб-сторінок.
- Багатофункціональність: CSS дозволяє використовувати різні типи стилів, такі як кольори, розміри, шрифти, фонові зображення та інші, що дозволяє створювати багатофункціональні та красиві веб-сторінки.

- Скорочення часу розробки: CSS дозволяє ефективно використовувати код та виконувати швидкі зміни вигляду сторінки, що скорочує час розробки веб-сторінок.

- Кросбраузерна сумісність: CSS дозволяє створювати веб-сторінки, які можуть бути відображені в різних браузерах та на різних пристроях з різними розмірами екранів.

Незважаючи на багато переваг CSS, вона також має кілька недоліків, серед яких:

- Складність: CSS може бути складною для вивчення та розуміння, особливо для початківців. Він має велику кількість правил та властивостей, що потребує часу та практики для вивчення та використання.

- Браузерна несумісність: різні браузери можуть інтерпретувати CSS-код по-різному, що може призводити до проблем з відображенням веб-сторінок в різних браузерах.

- Швидкість завантаження: CSS-файли можуть бути великими та повільно завантажуватися, що може вплинути на швидкість завантаження веб-сторінок.

«JavaScript – це мова сценаріїв, яка використовується для створення та керування динамічним вмістом веб-сайту, тобто будь-чим, що переміщується, оновлюється чи іншим чином змінюється на вашому екрані без необхідності перезавантажувати веб-сторінку вручну» [27]. JavaScript має такі функції, як:

- анімована графіка
- фото слайд-шоу
- пропозиції автозаповнення тексту
- інтерактивні форми

JavaScript має кілька переваг, серед яких:

- Клієнтська сторона: JavaScript виконується на клієнтській стороні, що означає, що код виконується на комп'ютері користувача, а не на сервері. Це дозволяє створювати більш динамічні та інтерактивні веб-сайти без потреби перезавантаження сторінки.

- Переносимість: JavaScript є стандартом веб-розробки та підтримується всіма сучасними браузерами. Це означає, що код, написаний на JavaScript, можна використовувати на будь-якій платформі.

- Багатофункціональність: JavaScript може використовуватися для розробки різноманітних функцій, таких як перевірка форм, анімації, інтерактивних карт та багато іншого.

- Взаємодія зі сторонніми API: JavaScript може використовувати API, надані різними сторонніми сервісами, такими як Google Maps, YouTube та багато інших.

- Простота використання: JavaScript має простий та зрозумілий синтаксис, що дозволяє швидко та ефективно розробляти веб-додатки та веб-сайти.

- Розширюваність: JavaScript має велику кількість бібліотек та фреймворків, що дозволяє розширювати можливості мови та швидко розробляти веб-додатки.

Незважаючи на переваги, JavaScript має кілька недоліків:

- Безпека: JavaScript може створювати потенційні проблеми з безпекою, такі як злам веб-сайту або віддалене виконання коду, якщо його не застосовувати правильно.

- Підтримка старих браузерів: Старі версії браузерів можуть не підтримувати останні функції JavaScript, що може створити проблеми з переносимістю коду на різні платформи.

- Швидкість: JavaScript інтерпретується, а не компілюється, що може знизити швидкість виконання коду в порівнянні з іншими мовами програмування.

- Залежність від клієнта: Код JavaScript виконується на стороні клієнта, тому залежить від того, який браузер та яку версію використовує користувач. Це може створити проблеми з переносимістю коду та забезпеченням сумісності.

- Перевантаження веб-сторінок: JavaScript може використовувати значні ресурси браузера, що може призвести до перевантаження веб-сторінок та зниження їх продуктивності.

React є бібліотекою JavaScript, розробленою компанією Facebook. Вона використовується для створення швидких користувацьких інтерфейсів для веб-сайтів і програм, зокрема Instagram.com. Основною концепцією React є віртуальний DOM, який складається з дерева компонентів JavaScript, що імітує дерево DOM. Це дозволяє виконувати мінімальну кількість маніпуляцій з DOM для підтримки компонентів React у поточному стані.

React є частиною мови JavaScript, що дає безліч переваг. Використання React дозволяє легко масштабувати продукти, використовувати єдину мову на стороні сервера/клієнта/мобільного пристрою, забезпечує виняткову продуктивність і має шаблони робочого процесу для зручної командної роботи. Код інтерфейсу користувача є читабельним та легко підтримується. Багато провідних світових компаній використовують React та інші технології JavaScript у своїх продуктах, що визначає ринок (наприклад, Instagram, Reddit і Facebook).

React.js є популярною бібліотекою для веб-розробки, оскільки вона пропонує оптимізований та ефективний інтерфейс розробки та мову кодування. Завдяки спрощеному API та високопродуктивним можливостям, React.js надає зручний та швидкий робочий процес розробки. Крім того, компоненти та концепції React.js дуже прості для розуміння, що сприяє швидкому освоєнню технології [33].

У порівнянні з іншими фреймворками, такими як Vue та Angular, React.js не навантажує HTML-код додатковими атрибутами, що сприяє створенню чистішого

та більш читабельного коду. Зокрема, з використанням JSX у JavaScript, React.js дозволяє створювати більш організований та зрозумілий код.

Найбільшою перевагою використання React.js є можливість повторного використання коду на різних платформах. Зокрема, React.js дозволяє створювати компоненти та елементи веб-дизайну, що можуть використовуватися на різних веб-сайтах та платформах. Окрім того, React.js дозволяє розробникам зосередитися на створенні окремих компонентів, що сприяє ефективній та зручній роботі з бібліотекою.

Додатково, слід зазначити значний внесок великої та давно створеної спільноти. Розширена екосистема React настільки обширна, що дозволяє розробникам створювати рішення для різних платформ, включаючи настільні комп'ютери та мобільні додатки, створювати статичні веб-сайти, обробляти серверний рендеринг та інтегрувати передові технологічні концепції, такі як віртуальна реальність та 360-огляди, з веб-рішеннями. Все це можливо завдяки простій та зрозумілій філософії веб-розробки React.

Використання React забезпечує неймовірно швидку візуалізацію, оскільки він віртуалізує та зберігає DOM у пам'яті. Всі зміни перегляду легко відображаються у віртуальному DOM, а спеціалізований алгоритм відмінностей зіставляє колишні та існуючі віртуальні стани DOM, щоб обчислити найефективніший спосіб застосування нових змін, не потребуючи занадто багато оновлень. Мінімальна кількість оновлень вводиться для досягнення найшвидшого часу читання/запису, що відзначається загальним підвищенням продуктивності.

Зміни DOM уповільнюють системи, а завдяки віртуалізації DOM ці зміни мінімізуються й оптимізуються інтелектуальним способом. Усі маніпуляції з віртуальним DOM відбуваються «за лаштунками», тобто внутрішньо й автономно, що також дозволяє значно заощаджувати споживання апаратних ресурсів (наприклад, потужність ЦП і батарея в мобільних пристроях, які мають підказувати, коли також використовувати React.js).

Хоча React має багато переваг, він також має деякі недоліки:

- Високий навчальний поріг: React має дещо високий поріг входження. Для розуміння React необхідно мати базові знання JavaScript, HTML та CSS. Також, React використовує специфічні концепції, такі як віртуальний DOM, що може забути складним для початківців.
- Залежність від сторонніх бібліотек: Для розробки великих додатків на React може знадобитися використання багатьох сторонніх бібліотек. Це може збільшити кількість завантажуваного коду та знизити продуктивність додатка.
- Складність документування: React дозволяє розробникам писати код більш зручним способом, але це може створити складність при документуванні коду для інших розробників.
- Потреба в конфігурації: React не має вбудованого механізму для роботи з бекендом, а також не має вбудованих інструментів для роботи з різними форматами даних, такими як JSON або XML. Тому, для розробки повноцінного додатка на React, може знадобитися конфігурація додаткових інструментів та бібліотек.
- Роздутість проектів: React дозволяє розробляти дуже складні проекти з великою кількістю функцій, але це може призвести до роздутості проектів, збільшення їх складності та важкості їх підтримки.

Програмне забезпечення JetBrains WebStorm - це інструмент для розробки веб-сайтів та редагування HTML, CSS та коду JavaScript. Рішення забезпечує швидку навігацію по файлах і генерує повідомлення про виникаючі проблеми в коді в режимі реального часу. JetBrains WebStorm дозволяє додавати розмітку для HTML-документів або елементів SQL безпосередньо на JavaScript. JetBrains WebStorm розгортає та синхронізує проекти через FTP [31].

Використовуючи потужність HTML/XHTML та XML-коду, WebStorm автоматично завершує стилі, посилання, атрибути та інші елементи коду. При роботі з CSS код для класів, HTML-номерів, ключових слів і так далі

завершується. WebStorm пропонує автоматичні рішення таких проблем, як вибір формату, властивостей, класів, посилань на файли та інших атрибутів CSS. Рішення дозволяє використовувати потужність інструменту кодування дзен для макета HTML, відображає дії тегів на веб-сторінці. WebStorm завершує код JavaScript для ключових слів, міток, змінних, параметрів і функцій DOM, а також підтримує специфічні функції популярних браузерів. Реалізовані в рішенні функції рефакторингу JavaScript дозволяють трансформувати структуру вашого коду і файлів, а також .js.

WebStorm забезпечує налагодження коду JavaScript і надає широкий спектр функцій: пошук точки зупину в HTML і JavaScript, налаштування параметрів точки зупину, тестування синтаксису коду в режимі реального часу і т.д. Продукт підтримує платформи jQuery, YUI, Prototype, DoJo, MooTools, Qooxdoo і Bindows. WebStorm забезпечує інтегровану перевірку тексту для тегів, послідовності коду, орфографічних помилок тощо. WebStorm дозволяє редагувати файли і автоматично синхронізувати їх на вимогу при роботі або зберіганні їх віддалено.

Продукт підтримує функцію контролю версій і попередніх версій коду і записує всі внесені дії і зміни. Створюючи історію в WebStorm, ви можете відновити вирази коду, блоки і навіть цілі файли [29].

Розглянувши низку технологій та інструментів, які можна використовувати для створення сучасного веб-сайту ресторану піцерії. Було досліджено React як популярний інструмент для розробки веб-додатків, що дозволяє створювати динамічні та масштабовані веб-сайти. Також було розглянуто інструменти для розробки веб-інтерфейсів, такі як HTML та CSS, а також IDE WebStorm для покращення продуктивності розробки.

Загалом, використання вищезгаданих технологій та інструментів може допомогти забезпечити створення зручного та привабливого веб-сайту ресторану піцерії, який буде зручним для користувачів та матиме високу продуктивність та швидкість роботи. Однак, при виборі конкретних технологій та інструментів слід

враховувати вимоги та потреби конкретного проекту, а також навчитися ефективно використовувати обрані інструменти для досягнення бажаних результатів.

РОЗДІЛ 2. ТЕХНОЛОГІЯ ПРОЄКТУВАННЯ ВЕБ-САЙТУ

2.1. Етапи створення веб-сайту та проєктування дизайну сайту

Веб-розробка є складним процесом, який включає кілька етапів та ідей. Основними етапами є аналіз, проєктування, розробка, тестування та перевірка, розгортання та обслуговування. Метою цього процесу є досягнення спільної мети та спрощення робочого процесу для отримання кращих результатів.

Етап аналізу та збору інформації є важливим для розуміння цілей веб-сайту та визначення найкращого способу їх досягнення. Веб-розробники співпрацюють з клієнтами, щоб зрозуміти їхні потреби та визначити найкращий спосіб їх задоволення. Це передбачає проведення досліджень користувачів, аналіз веб-сайтів конкурентів, створення персонажів користувачів, технічне обґрунтування та розробку карт сайтів і каркасів.

Метою фази аналізу є отримання глибокого розуміння проблеми, яку веб-сайт намагається вирішити, для знаходження найкращого рішення. Ретельний аналіз вимог до веб-сайту дозволяє веб-розробникам переконатися, що кінцевий продукт ефективний та відповідає потребам користувачів. Чітке розуміння цілей та вимог є необхідним для розробки стратегії досягнення мети веб-сайту.

Етап проєктування процесу веб-розробки є ключовим, оскільки він безпосередньо впливає на UI/UX веб-сайту. Особливо важливо, щоб дизайн вашого веб-сайту був адаптивним, інтуїтивно зрозумілим та простим у навігації. Веб-дизайнери відповідають за створення оптимізованого дизайну веб-сайту, що відповідає вимогам клієнта.

На цьому етапі розробляється структура веб-сайту, а також його візуальні компоненти та вміст, включаючи макети діяльності, навігаційні панелі, зображення та відео, щоб забезпечити користувачам багатофункціональний аудіовізуальний досвід. Досвідчені дизайнери, які орієнтовані на останні тенденції

веб-розробки, можуть допомогти вам створити ефективний дизайн за допомогою найсучасніших фреймворків, бібліотек та компонентів дизайну.

Веб-дизайнери повинні мати високий рівень володіння зовнішніми технологіями, зокрема HTML/CSS, та відповідати вимогам кількох браузерів та веб-стандартів.

Фаза розробки та кодування процесу веб-розробки є періодом написання фактичного коду веб-сайту з використанням різних мов програмування залежно від функціональності, необхідної для веб-сайту. В цій фазі веб-розробник використовує конструкції, каркаси та інші елементи з попередніх етапів та перетворює їх на робочий веб-сайт [29].

Етап кодування може включати різноманітні завдання, від написання коду до налаштування функцій на стороні сервера. У деяких випадках весь сайт може бути написаний з нуля, але розробники зазвичай використовують вже чинні інструменти або фреймворки, щоб прискорити процес. Етап розробки також передбачає розгортання плагінів сайту, додаткових функцій та інтеграцію засобів безпеки. Після завершення кодування веб-сайт проходить тестування, щоб переконатися, що все функціонує належним чином.

Після завершення етапів дизайну та розробки веб-сайту настає час комплексного тестування з метою виявлення та виправлення помилок перед його запуском. Це один з найбільш критичних та недооцінених етапів веб-розробки, який вимагає використання різноманітних методів та прийомів тестування для переконання у належній роботі та оптимальному рівні веб-сайту.

Етап тестування можна розділити на дві основні категорії:

- Функціональне тестування: спрямоване на перевірку виконання веб-сайтом завдань, поставлених перед ним.
- Нефункціональне тестування: оцінює продуктивність, безпеку та зручність веб-сайту.

Під час розробки веб-сайту тестувальники виконують серію тестів, включаючи модульні, інтеграційні та системні тести, щоб переконатися, що веб-сайт не містить помилок. Після запуску веб-сайту важливо також слідкувати за його роботою та розв'язувати будь-які проблеми, що виникають. Здійснюючи ретельне тестування на кожному етапі розробки, розробники можуть створювати високоякісні веб-сайти, які відповідають потребам користувачів.

Фаза розгортання веб-розробки є процесом запуску веб-сайту або програми. Цей етап зазвичай включає перенесення коду з сервера розробки або проміжного сервера на сервер виробництва. Хоча етап розгортання може відрізнятися в залежності від розміру та складності проєкту, він, як правило, включає такі завдання, як налаштування середовища, стиснення файлів, налаштування кешування та налаштування балансування навантаження.

Процес розгортання може бути досить складним, тому перед початком важливо мати надійний план. Оскільки веб-сайт запускається на останньому етапі, дуже важливо переконатися, що всі необхідні умови є на місці та розробники готові до різних сценаріїв. Після того, як сайт або програма запрацювали, їх необхідно відстежувати та підтримувати, щоб гарантувати, що вони залишаються доступними та продуктивними.

Обслуговування веб-сайту є необхідною складовою процесу веб-розробки з метою забезпечення оптимальної продуктивності та неперервного оновлення з урахуванням змінних вимог. Етап обслуговування містить в собі систематичний моніторинг продуктивності веб-сайту та забезпечення регулярних оновлень для розширення функцій та покращення їх роботи. Також, періодичні оновлення є необхідними для впровадження надійних протоколів безпеки та оптимізації взаємодії з користувачем.

Обслуговування веб-сайту може включати різноманітні завдання, такі як оновлення програмного забезпечення, резервне копіювання даних, моніторинг веб-трафіку та перевірка на наявність вразливостей у безпеці. Регулярно

виконуючи технічне обслуговування, можна забезпечити, що ваш веб-сайт завжди є актуальним, оновленим, доступним та оптимальним для користувачів. Це важлива складова веб-розробки, тому необхідно відвести достатньо часу та ресурсів для обслуговування, як частини загальної стратегії веб-розробки [17].

Етапи розробки веб-сайту можуть змінюватися в залежності від проєкту, проте загалом, їх можна розділити на кілька основних етапів, таких як аналіз, дизайн, розробка та тестування. Для розробки веб-сайту ресторану виділено декілька етапів, які будуть розроблятися під час кваліфікаційної роботи:

1. Аналіз: цей етап передбачає дослідження та аналіз вимог готелю та його клієнтів. На цьому етапі необхідно з'ясувати, яку мету веб-сайту має виконувати, яку має бути цільова аудиторія та які функціональні вимоги мають бути враховані. Також важливо дослідити конкурентів та аналізувати їхні веб-сайти.

2. Дизайн: на цьому етапі створюється дизайн веб-сайту. Візуальний дизайн та архітектура інформації мають бути розроблені з урахуванням мети веб-сайту та його аудиторії. Дизайн повинен бути привабливим, зручним та легким у використанні для користувачів.

3. Розробка: цей етап передбачає програмування веб-сайту на базі вимог, що були визначені на етапі аналізу. Розробники повинні створити функціональні модулі та додатки для веб-сайту та настроїти серверне забезпечення. Розробка веб-сайту має бути проведена відповідно до сучасних стандартів та забезпечувати максимальну продуктивність та швидкість роботи.

4. Тестування: Після завершення розробки вмісту та візуальних елементів веб-сайту, настає час перейти до етапу тестування. Основною метою даного етапу є перевірка кожної сторінки веб-сайту з метою забезпечення коректної роботи всіх посилань та відображення веб-сайту в різних браузерах. Важливою деталлю є також порядок слів у заголовках та описах, що має великий вплив на ефективність веб-сторінки у різних пошукових системах.

Усі етапи розробки веб-сайту повинні бути проведені відповідно до сучасних вимог і стандартів розробки, щоб забезпечити високу якість та ефективність веб-сайту.

Веб-дизайн є процесом планування, концептуалізації та розміщення вмісту в Інтернеті, включаючи веб-програми, мобільні програми та дизайн інтерфейсу користувача. Сучасна розробка веб-сайту перетинає межі естетики та включає загальну функціональність веб-сайту [26].

Важливість дизайну полягає в тому, що люди потребують лише 0,05 секунди, щоб сформувати думку про ваш веб-сайт. Більшість враження про сайт формується на основі його дизайну. Крім того, дизайн веб-сайту має вплив на конверсії, довіру та, в цілому, успіх сайту. Оптимізація дизайну веб-сайту є важливою для досягнення максимального успіху.

Необхідно також звернути увагу на те, що 88% споживачів після негативного досвіду не повернуться на веб-сайт. Зокрема, останнє дослідження показало, що 77% агентств вважають поганий дизайн веб-сайту найбільшою слабкістю своїх клієнтів.

Для успішного створення інформативного та зручного для користування сайту, веб-дизайнер повинен мати глибокі знання загальних принципів програмування, вміти створювати та оформлювати різноманітні об'єкти інформаційного середовища Інтернету, а також мати експертизу в галузі маркетингу та пошукової оптимізації [31].

Головною метою веб-дизайну є:

1. Унікальність. В умовах високої конкуренції особливо важливо, щоб дизайн сайту був оригінальним та унікальним. Це дозволяє виділятися серед подібних ресурсів та забезпечувати високу прибутковість комерційних проєктів. Оригінальність сайту забезпечується за допомогою використання оригінальних графічних та кольорових рішень, а також індивідуальних творчих та художніх прийомів.

2. Залучення уваги. Важливою метою дизайну сайту є привернення уваги максимальної кількості користувачів та формування цільової аудиторії сайту. Для досягнення цієї мети використовуються індивідуально спроектовані графічні елементи, які допомагають привернути увагу користувачів та збільшити популярність ресурсу. Проте надмірне захоплення деталями та використання однотипних текстів негативно впливає на сприйняття сайту.

3. Донесення інформації є одним із ключових аспектів веб-дизайну, оскільки він спрямований на правильне інформування користувачів про розміщену на сайті інформацію. Ефективне комбінування контенту та ілюстрацій, які допомагають зрозуміти та легко організувати інформацію для зручності сприйняття та прочитання, є ключовим елементом в цьому процесі. Для досягнення цієї мети, дизайнер повинен мати велику ерудицію в роботі зі шрифтами, міжрядковими інтервалами, довжиною рядків, вибором кольору та іншими аспектами дизайну [36].

Своєю чергою, зручність користування є важливим фактором в успіху веб-сайту. Вбудована навігаційна система має вирішальне значення для залучення цільової аудиторії на сторінки сайту, і повинна бути розроблена з врахуванням вимог користувачів та їх поведінки на сайті. При проектуванні навігації, дизайнер повинен дотримуватися стандартів і забезпечувати зручний та логічний доступ до всіх розділів та функцій сайту, що своєю чергою забезпечує високу якість веб-сайту та задоволення його відвідувачів.

На головній сторінці веб-сайту розміщено яскравий банер зі зображенням піци та запрошенням скористатися послугою доставки. На наступному блоку можна знайти найпопулярніші пропозиції з можливістю вибрати піцу за доступними цінами (Рис. 2.1).

Подальші блоки на головній сторінці містять інформацію про послуги, що надає компанія, а також про її історію та місію. Крім того, на сторінці присутній

блок зі зворотними зв'язками клієнтів, де можна ознайомитися з думками задоволених покупців та залишити відгук самостійно.

У верхній частині сторінки розміщені посилання на соціальні мережі компанії, а також блоки "Популярні піци", "Найвищі оцінки клієнтів" та "Новинки в меню". З метою зручності користувачів, присутня можливість вибору піци за фільтрами - вага, склад, ціна тощо.

У нижній частині сторінки веб-сайту доступна детальна інформація про технічні питання, зокрема про замовлення, оплату послуг, умови доставки та гарантії на продукцію.

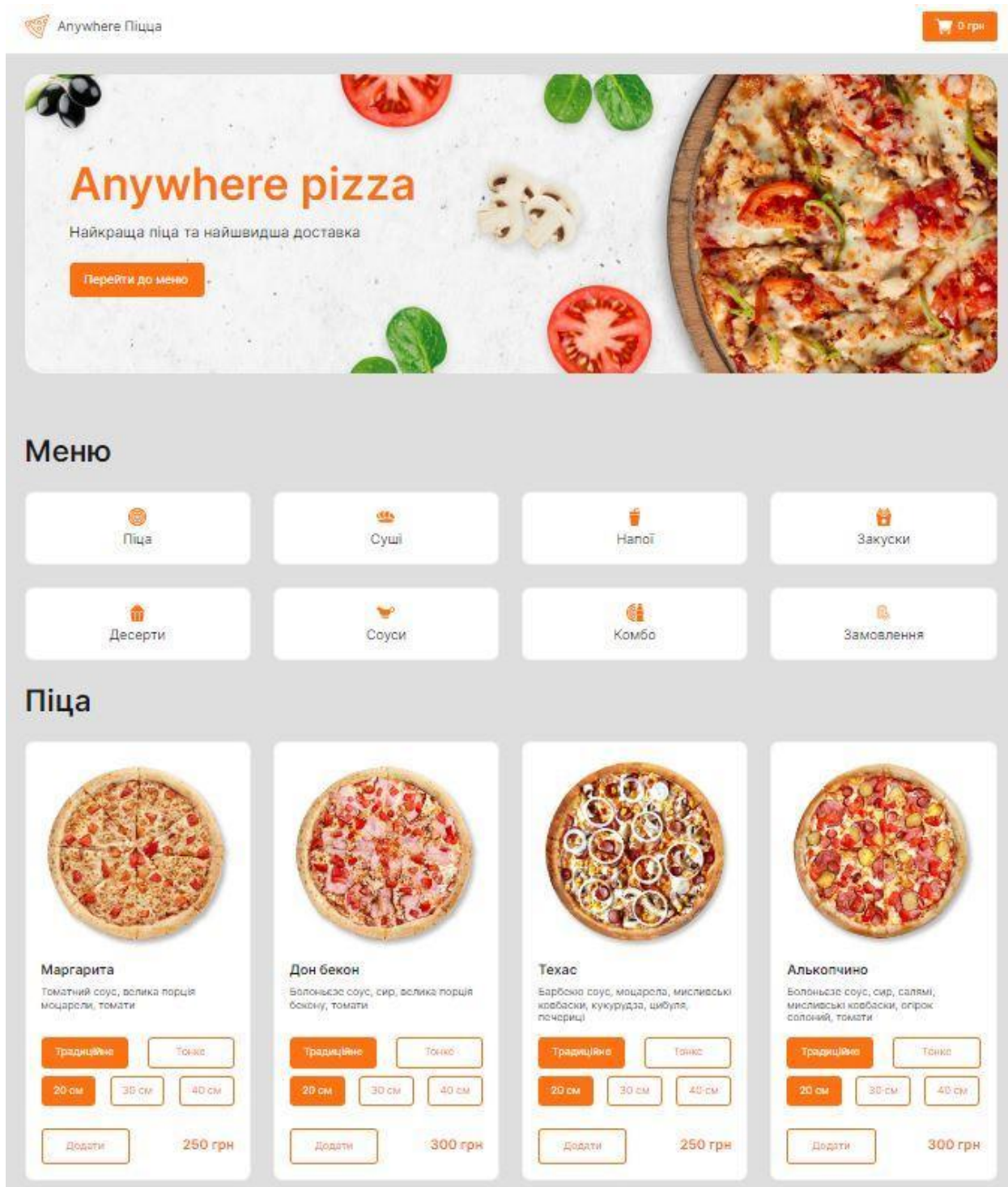


Рис. 2.1 Головна сторінка веб-сайту ресторану піцерії

На сторінці вибору їжі (Наприклад суші) присутній блок із зображеннями та описом суші. Кожне зображення містить назву та склад страви, що відображається

під нею. Під блоком знаходиться кнопка «Замовити», яка спрямовує користувача на сторінку замовлення.

Для зручності користувачів в розділі суші присутні фільтри за складом, вагою та ціною, що значно спрощує процес вибору необхідної страви. Додатково, на сторінці можна знайти опис доставки та способів оплати, а також інформацію про акції та знижки (Рис 2.2).

У верхній частині сторінки присутні посилання на інші розділи сайту та соціальні мережі компанії.

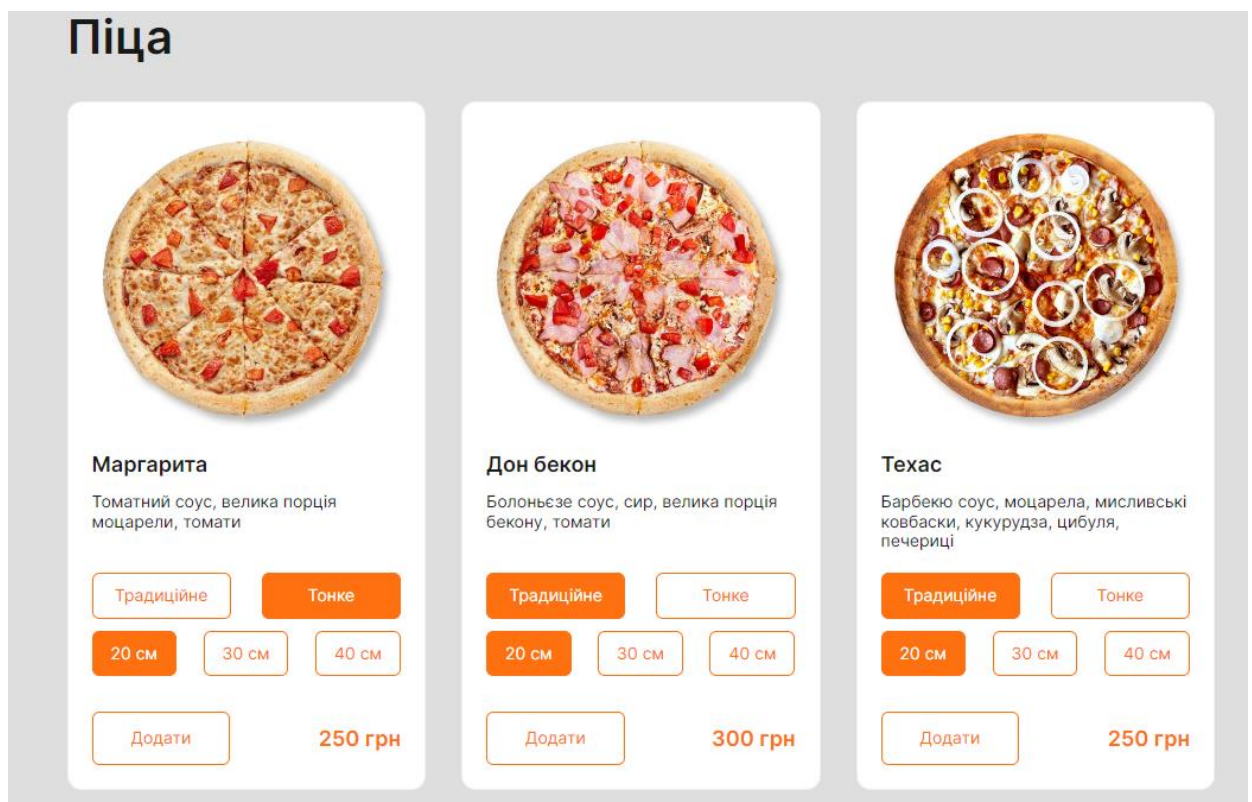


Рис. 2.2. Сторінка вибору піци

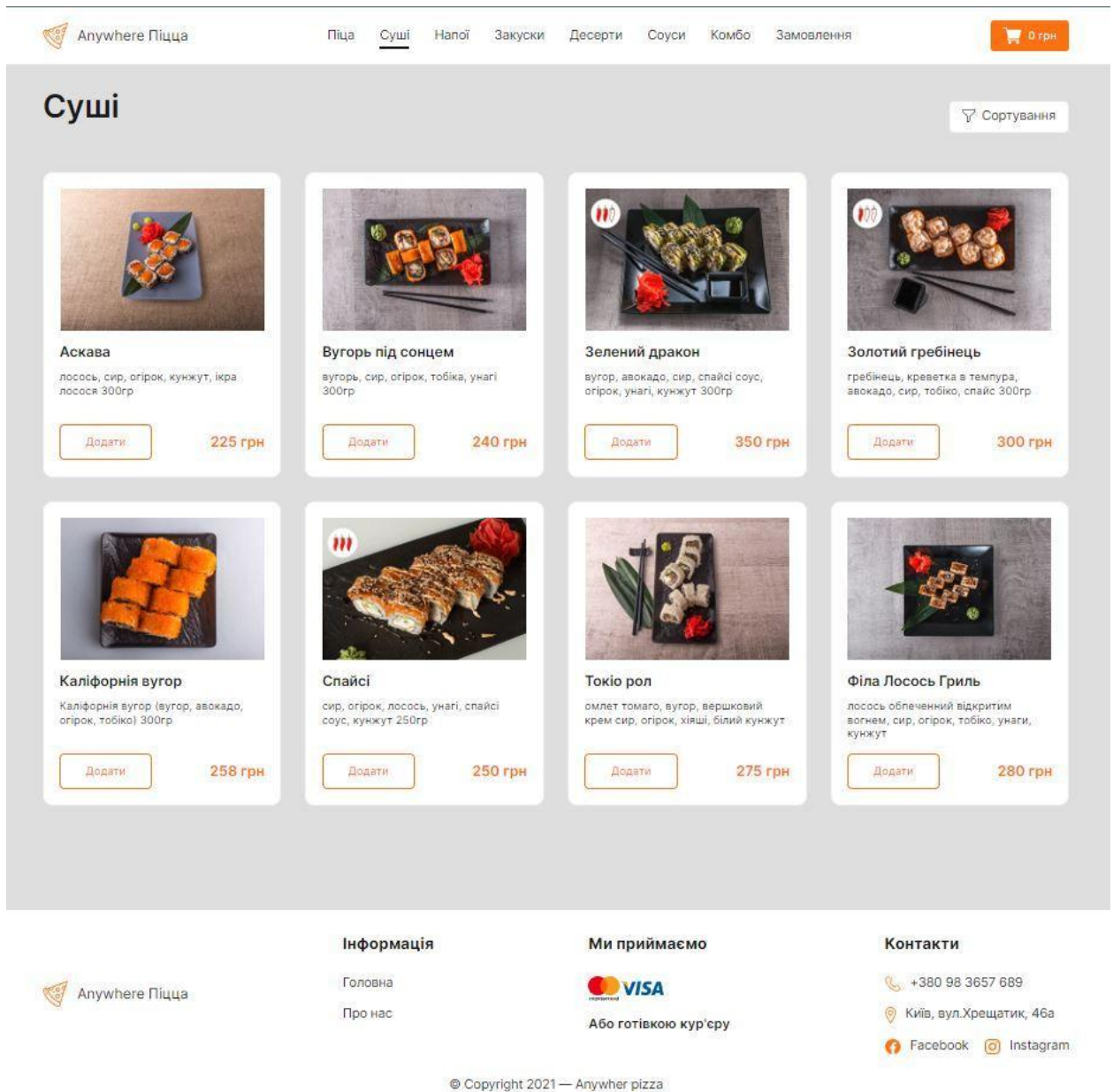


Рис. 2.3 Сторінка вибору їжі

Сторінка замовлення є дуже важливою, оскільки саме тут відвідувач може обрати та замовити піцу. На цій сторінці користувач може перевірити список обраних товарів та їх кількість, а також перевірити загальну вартість замовлення (Рис. 2.4).

Крім того, на сторінці замовлення є можливість ввести промокод, який знизить вартість замовлення, або ввести коментар до замовлення. Клієнт може

обрати метод оплати та спосіб доставки, заповнивши необхідні дані для цього. Після цього користувач повинен натиснути на кнопку «Оформити замовлення», щоб підтвердити свій вибір.

У разі успішного оформлення замовлення, на сторінці з'явиться повідомлення про успішну оплату та доставку замовлення. Клієнт також отримає підтвердження замовлення на вказану електронну пошту.

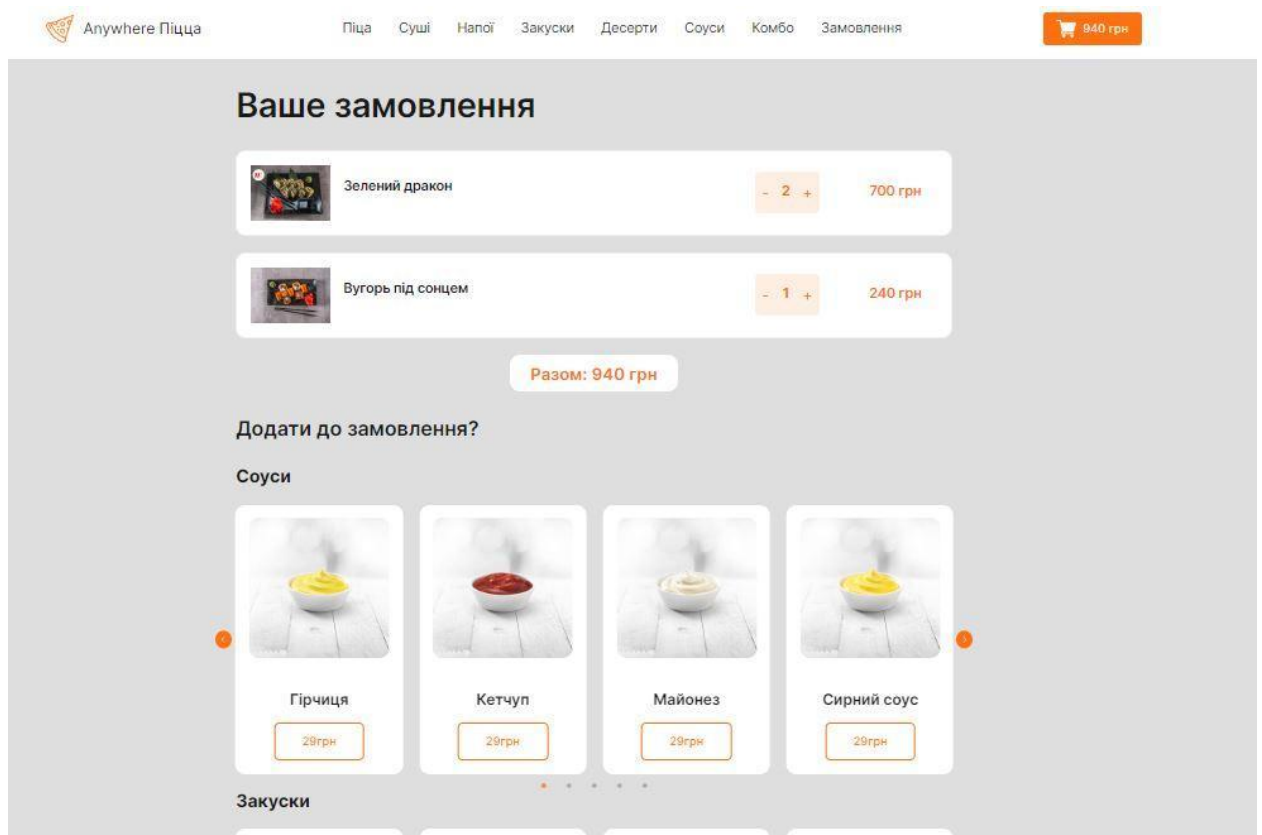


Рис. 2.4 Сторінка замовлення веб-сайту

Дизайн сайту ресторану піцерії є привабливим та зручним для користувачів. Він має яскраві та чіткі елементи, які дозволяють швидко знайти необхідну інформацію та зробити замовлення піци. Структура сайту розроблена логічно та зручно, що полегшує навігацію користувачів. Крім того, на сайті присутні функції фільтрації, що дозволяють користувачам швидко знайти потрібну інформацію, та зручна форма замовлення піци.

Загалом, дизайн веб-сайту піцерії відповідає сучасним стандартам та потребам користувачів, дозволяє залучити цільову аудиторію та зробити покупки в зручний спосіб.

2.2 Архітектура та база даних веб-сайту ресторанна

Розробка архітектури веб-сайту є важливим етапом у створенні нового онлайн-проєкту. Цей процес містить в собі розробку зовнішньої та внутрішньої структури ресурсу та встановлення зв'язків між всіма його елементами згідно з вимогами пошукових систем, правилами зручності користування та потребами цільової аудиторії.

Ресурси з добре продуманою структурою мають більші шанси на успіх, оскільки їх легше індексувати та ранжувати пошуковими системами. Крім того, на таких сайтах користувачі можуть легше орієнтуватися та проводять на них більше часу, що збільшує їхню вірогідність виконання цільових дій, таких як реєстрація, замовлення товарів та послуг або підписки на розсилки. Таким чином, якісна та логічна архітектура веб-сайту не тільки сприяє ефективному просуванню проєкту в інтернеті, а й допомагає залучати нових клієнтів та збільшувати прибуток.

Процес проєктування архітектури веб-сайту є складним та багатоетапним. Вартість та терміни послуг визначаються індивідуально та залежать від складності та особливостей конкретного веб-проєкту. Фактори, які впливають на вартість та терміни проєктування, включають створення структури з нуля або оптимізацію вже існуючої, цілі та бюджет клієнта, а також потребу в розробці додаткових функцій та інтерфейсів [29].

При проєктуванні архітектури сайту діє команда кваліфікованих фахівців, що складається зі спеціалістів у галузі SEO, UX, програмування та верстки. SEO-експерт займається складанням структури сайту та ієрархії сторінок для подальшої розробки та просування ресурсу. UX-дизайнер відповідає за зовнішнє відображення сторінок та проєктування дизайну. Програміст відповідає за

технічну реалізацію веб-розробки, а верстальник забезпечує «збирання» елементів онлайн-ресурсу та розробку адаптивної верстки.

Архітектурний шаблон є узагальненим розв'язанням поширеної проблеми в архітектурі програмного забезпечення в конкретному контексті. Шаблон розглядається як вирішення завдання у певному контексті. Часто розробники не повністю розуміють різницю між архітектурними шаблонами, а іноді взагалі мають мало знань про них.

Багаторівнева архітектура (також відома як «n-рівнева») є найпоширенішим архітектурним шаблоном. Її добре знають більшість архітекторів, проєктувальників та розробників. Немає обмежень на кількість або тип рівнів, проте в більшості випадків така архітектура складається з чотирьох рівнів: рівень представлення даних, рівень бізнес-логіки, рівень зберігання даних та рівень бази даних.

Цей підхід рекомендований для невеликих та простих додатків та веб-сайтів. Також цей шаблон добре підходить, якщо бюджет і час обмежені.

Однією з переваг багаторівневої архітектури є те, що вона дозволяє розділити програмне забезпечення на сутності, звані рівнями. Кожен рівень - група модулів, що надають взаємопов'язаний набір сервісів. Їх застосування має бути односпрямованим. Рівні повністю поділяють програмне забезпечення, при цьому кожна частина доступна через публічний інтерфейс.

Наявність рівнів в багаторівневій архітектурі може знизити продуктивність, що робить її неоптимальним варіантом для високопродуктивних застосунків. Проходження кількох рівнів архітектури для виконання бізнес-запиту може бути неефективним, тим самим збільшуючи загальну вартість і ускладнюючи систему [24].

З іншого боку, багаторівнева архітектура є добрим варіантом для невеликих та простих застосунків і веб-сайтів, особливо коли бюджет та час є обмеженими ресурсами. Така архітектура дозволяє розділити програмне забезпечення на різні

сутності, які відомі як рівні. Кожен рівень складається з групи модулів, що надають взаємопов'язаний набір сервісів. Використання цих сервісів повинно бути односпрямованим. Кожен рівень повністю розділяє програмне забезпечення, а кожна частина стає доступною через публічний інтерфейс.

Один з аспектів застосування багаторівневої архітектури полягає в тому, що вона дозволяє розробникам розділити програмне забезпечення на сутності, звані рівнями, що спрощує процес його розробки та підтримки.

Інший аспект використання багаторівневої архітектури полягає в тому, що вона дозволяє розділити програмне забезпечення на рівні, кожен з яких має свою визначену роль і відповідальність. Наприклад, рівень представлення даних відповідає за обробку інтерфейсу, який призначений для користувача. Такий розподіл завдань в багаторівневій архітектурі спрощує створення ефективних ролей та відповідальності.

Для веб-сайту ресторану піцерії розроблено базову багаторівневу архітектуру, яка включає чотири рівні: База даних, де зберігатиметься база даних ресторану, рівень збереження даних, рівень бізнес-логіки та рівень презентації (Рис 2.5).

Шар збереження даних (Persistence layer) - це компонент багаторівневої архітектури програмного забезпечення, який відповідає за збереження та отримання даних з бази даних або іншого сховища даних. Для веб-сайту ресторану піцерії шар збереження даних включає класи та методи для роботи з даними про меню, замовлення, користувачів та іншу інформацію, необхідну для функціонування сайту.

Бізнес-шар (Business layer) - це компонент багаторівневої архітектури програмного забезпечення, який відповідає за реалізацію бізнес-логіки. Для веб-сайту ресторану піцерії, цей шар включає класи та методи для обробки замовлень, розрахунку вартості замовлення, оплати та інших бізнес-процесів, необхідних для

функціонування сайту. А саме клас OrderService, який містить методи для обробки замовлень, Cart, який містить методи для роботи кошика тощо.

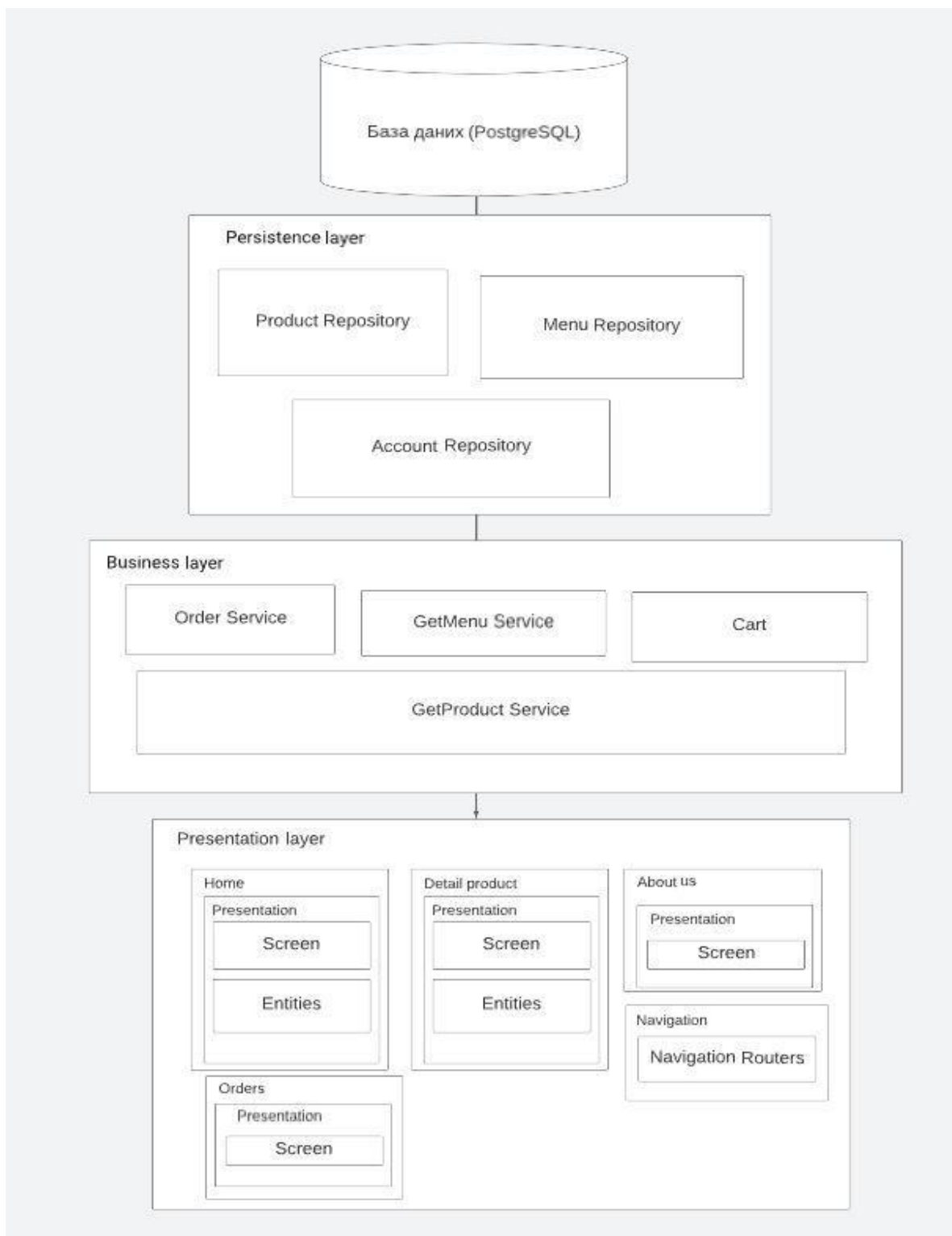


Рис. 2.5. Базова архітектура ресторану піцерії

Шар представлення (Presentation layer) – це компонент багаторівневої архітектури програмного забезпечення, який відповідає за взаємодію з користувачем. Для веб-сайту ресторану піцерії, цей шар включає класи та методи для обробки запитів користувача, формування веб-сторінок та іншої логіки, пов'язаної з взаємодією з користувачем. В ньому містяться основні сторінки веб-сайту, навігація та логіка показу на екран.

Багаторівнева архітектура є вдалим вибором для веб-сайту ресторану піцерії, оскільки вона забезпечує розділення програмного забезпечення на окремі рівні, кожен з яких має свої функції та відповідальність. Такий підхід дозволяє розподілити завдання між розробниками та спростити процес розробки та підтримки програмного забезпечення.

Застосування багаторівневої архітектури дає можливість створити гнучке та легко підтримуване програмне забезпечення для веб-сайту ресторану піцерії. Це забезпечує більш ефективне та швидке розв'язання проблем, що виникають у процесі розробки та підтримки веб-сайту. Крім того, така архітектура забезпечує більшу складність та можливість розширення веб-сайту під час його розвитку.

База даних є структурованим набором інформації, що може бути доступним для перегляду та модифікації. Вона є невидимим фундаментом багатьох застосунків, що забезпечують функціонування веб-сайтів, систем обробки даних та інших програмних засобів.

Зазвичай набір даних підпорядковується певним правилам, які забезпечують коректність та цілісність інформації. Наприклад, у випадку видалення користувача з бази даних, відповідні правила передбачають видалення всіх пов'язаних параметрів, таких як адреса електронної пошти, телефонний номер, інформація про рік народження та інші. З цієї причини реляційні бази даних надають надійний та структурований підхід до зберігання даних [24].

Реляційна база даних - це тип бази даних, яка будується на основі реляційної моделі. Тобто, має табличний спосіб вистави даних. Реляційна база даних є сукупністю елементів даних, організованих у вигляді набору формально описаних таблиць, з яких дані можуть бути доступними або повторно зібрані багатьма різними способами без необхідності реорганізації таблиць бази даних [37].

Основна особливість – надійність та незмінність даних, низький ризик втрати інформації. При оновленні даних їхня цілісність гарантується, вони замінюються у одній таблиці.

Реляційні бази даних, на відміну від нереляційних, відповідають ACID – це вимоги до транзакційних систем. Відповідність до них гарантує цілісність та збереження даних, а також передбачуваність роботи бази даних.

Проектування бази даних веб-сайту піцерії відбулося в додатку LucidChart. Це один з найбільш широко використовуваних інструментів проектування баз даних, який допоможе швидко створити діаграми баз даних в Інтернет, за допомогою інструментів спільного проектування баз даних. Вона надає допомогу командам у створенні правильно функціонуючої бази даних. Це забезпечує ефективні можливості моделювання баз даних, гнучке ціноутворення, а також зручно для дизайнерів. LucidChart працює на основі хмари. Ця функція не вимагає завантажень, а веб-служба дозволяє користувачам співпрацювати для роботи над проєктами в режимі реального часу. Це запобігає простоті у використанні, підтримуючи різні функції імпорту файлів та інші.

Нормалізація бази даних, відома також як нормалізація SQL, виконує процес групування пов'язаних даних у єдину таблицю. Атрибутивні дані або опосередковано пов'язані дані знаходяться в різних таблицях, які пов'язані між собою логічним взаємозв'язком між батьківськими та дочірніми таблицями.

Нормалізація бази даних надає кілька основних переваг:

- **Покращена узгодженість даних:** Нормалізація дозволяє уникнути дублювання даних шляхом зберігання їх тільки в одному місці. Це сприяє збереженню консистентності та точності даних.
- **Зручне групування даних:** Нормалізація допомагає групувати подібні або пов'язані дані за однією і тією ж схемою. Це полегшує управління та аналіз даних, а також сприяє зрозумілості та організації інформації.
- **Покращений пошук:** Нормалізація сприяє швидкому пошуку даних, оскільки можна створювати індекси більш ефективно. Це особливо корисно для систем обробки транзакцій (OLTP), де швидкий доступ до даних має вирішальне значення.

Проте, нормалізація баз даних має також свої обмеження:

Складні запити: У разі потреби знайти пов'язані дані, як-от інформацію про товар чи працівника, потрібно об'єднувати більше ніж одну таблицю. Це може призводити до затримок у витягуванні даних та ускладнення запитів

Отже, нормалізація не є ідеальним варіантом для систем аналітичної обробки даних (OLAP), де вимагається швидкий доступ до великих обсягів даних та складні операції аналізу.

Усвідомлюючи ці аспекти, слід розуміти такі терміни:

Суб'єкт: Сутність, яка представляє реальний об'єкт, в якому зберігаються дані, пов'язані з цим об'єктом. Наприклад, працівники, кафедри, студенти тощо є прикладами сутностей.

Атрибути: Характеристики сутності, що надають інформацію про неї. У випадку таблиць, атрибути представлені стовпцями, які містять відповідні дані про сутність.

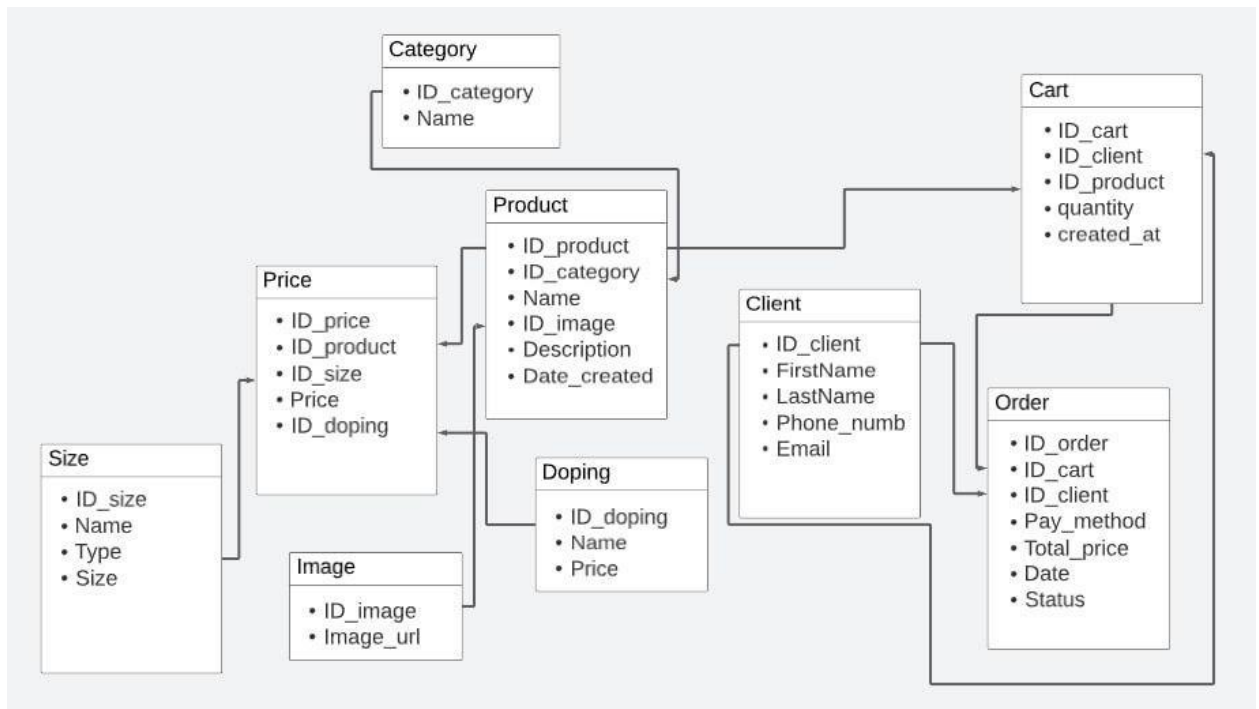


Рис. 2.6. Схема бази даних веб-сайту піцерії

Таблиця "Продукт" містить інформацію про всі продукти, які пропонуються в піцерії. Кожен рядок цієї таблиці відповідає одному продукту, а кожен стовпець буде містити певну інформацію про цей продукт.

Кожен продукт має унікальний ідентифікатор (id), який може бути використаний для посилання на цей продукт в інших таблицях. Також зберігається назва продукту, короткий опис, вартість у валюті піцерії, список складових, ідентифікатор категорії і шлях до зображення продукту на сервері.

Кожна категорія продуктів має свій унікальний ідентифікатор, який може бути використаний для посилання на цю категорію в таблиці "Категорія". Таким чином, в поле "Категорія" ми зберігаємо ідентифікатор категорії, до якої відноситься продукт.

Таблиця "Кошик" містить інформацію про замовлені клієнтом продукти, які ще не були оформлені як замовлення. Кожен рядок таблиці "Кошик" відповідає

одному продукту, який знаходиться в кошику, а кожен стовпець містить інформацію про цей продукт у кошику.

Кожен продукт у кошику має свій унікальний ідентифікатор (id), що дозволяє посилатись на цей продукт у інших таблицях.

В поле "Кількість" зберігається кількість продуктів, які знаходяться в кошику. Це дозволяє користувачу додавати більше однієї одиниці продукту у кошик, якщо він цього бажає.

Таблиця "Замовлення" містить інформацію про замовлені користувачами продукти та їх статус, який вказує на етап обробки замовлення. Кожен рядок таблиці "Замовлення" відповідає одному замовленню, а кожен стовпець містить певну інформацію про це замовлення. Також зберігається ідентифікатор користувача, який зробив замовлення, дату та час оформлення замовлення, загальну суму замовлення (включаючи доставку) та адресу, за якою необхідно доставити замовлення.

В полі "Статус" зберігається ідентифікатор поточного статусу замовлення. Наприклад, "очікує на оплату", "в обробці", "відправлено" або "доставлено".

Таблиця "Категорії" містить інформацію про категорії продуктів, які пропонуються в піцерії. Ця таблиця допоможе клієнтам легше знаходити продукти, які вони шукають, а також дозволить організувати продукти в логічні групи.

У цій таблиці зберігається ідентифікатор категорії, який є унікальним для кожної категорії продукту в базі даних, а також назва категорії, що дозволяє ідентифікувати продукти, які входять до цієї категорії.

Таблиця "Клієнт" містить інформацію про клієнтів, які роблять замовлення на веб-сайті піцерії. Ця таблиця містить інформацію про кожного клієнта, таку як його ім'я, електронну пошту, телефон, адресу доставки та інші контактні дані.

База даних веб-сайту піцерії створена в СУБД MySQL. MySQL – система управління базами даних вільного реляційного типу.

З основних плюсів mysql:

- Система має подвійне ліцензування, вона поширюється за умовами ліцензії GPL, але, якщо виникають проблеми або просто питання щодо даної ліцензії, то в гру вступає ліцензія від Oracle.
- Система інтегрується з великою кількістю платформ: Linux, macOS і багатьма версіями Windows, а також з багатьма іншими. Широкий асортимент мов програмування: API, C, C++, Java, Python, PHP, Компонентний Паскаль, Ейфель та інші.

З мінусів:

У багатобайтових кодуваннях, таких, як, наприклад, для Юнікод, mysql повинна переходити на одnobайтове кодування. Це все тому, що система некоректно вводить висловлювання в багатобайтових випадках [20].

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-САЙТУ

Реалізація веб-сайту зазвичай розпочинається зі створення загальної структури проекту та встановлення необхідних залежностей. Даний веб-сайт розроблявся в середовищі WebStorm, що дозволяє швидко і зручно створити структуру проекту.

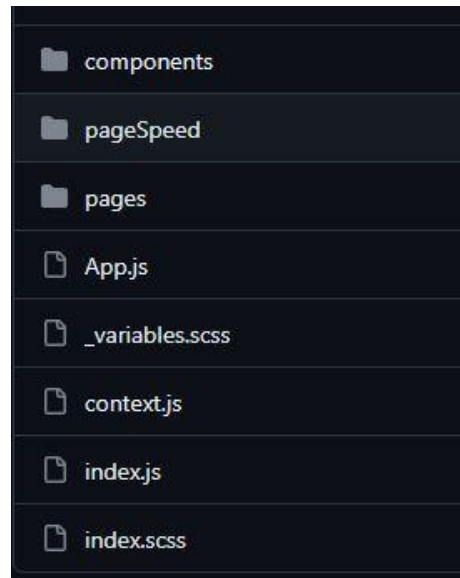


Рис. 3.1. Базова структура проекту

Після успішного створення структури проекту слід розпочинати створення компонентів.

У React компоненти використовуються для побудови ієрархії візуальних елементів і логіки вашого додатку. Вони дозволяють розділити великі та складні інтерфейси на менші, самодостатні частини, які можна повторно використовувати.

При побудові додатку ви можете створювати власні компоненти та вкладати їх один в одного, утворюючи ієрархію компонентів. Це дозволяє розділити логіку та візуальну представленість на невеликі та зрозумілі частини. Компоненти можна повторно використовувати у різних частинах додатку, що сприяє підтримці та масштабованості коду.

Реактивність React дозволяє автоматично оновлювати компоненти при зміні стану, що полегшує реагування на дії користувача та зміни даних у додатку.

Загальною структурою коду є компонент App, який є кореневим компонентом додатка. Він містить всі інші компоненти та забезпечує передачу станів та функцій через контекст (Додаток А, Рис.3.3).

Цей код є частиною програмної реалізації веб-сайту на React. Він включає в себе використання різних компонентів, станів, ефектів та іншого функціоналу. Основна логіка коду полягає в налаштуванні стану, отриманні даних з сервера та їх обробці, а також відображенні відповідних сторінок на основі маршрутизації. Код також використовує бібліотеки react-router-dom для налаштування маршрутів і навігації, а також axios для здійснення запитів до сервера.

Для налаштування навігації використовується компонент Nav (Додаток А, Рис. 3.4). Цей код є компонентом Nav, відповідальним за відображення навігаційного меню на веб-сторінці. Він використовує контекст AnywherePizzaContext для доступу до потрібних станів та функцій.

Компонент Nav отримує два параметри: navType і flexType. navType використовується для задання класу навігаційного меню, а flexType використовується для задання класу, що визначає тип розташування елементів меню.

У компоненті використовується цикл map, щоб перебрати елементи navLinks і створити відповідні елементи списку . Кожен елемент містить посилання <Link>, яке спрацьовує при кліку на нього. Параметри посилання встановлюються на основі даних з navLinks. Клас navType та navType--active встановлюються в залежності від того, чи посилання активне або ні.

При кліку на посилання виконуються деякі дії, такі як встановлення активного посилання, закриття мобільного меню тощо. Всі ці дії залежать від станів та функцій, які отримуються з контексту AnywherePizzaContext.

На основі переданих параметрів компонент створює відповідні класи для елементів меню, що дозволяє налаштовувати вигляд навігаційного меню за допомогою CSS стилів (Додаток А, Рис.3.4).

Цей код є файлом стилів для компонента Nav. Він використовує SASS для опису стилів навігаційного меню. Файл містить два блока стилів: `.nav` і `.nav-mobile`, які відповідають за стилізацію навігаційного меню для різних розмірів екрану.

У блоку `.nav` задані стилі для навігаційного меню на десктопних пристроях. Він має властивість `display: none` для розмірів екрану менше 992 пікселів, що означає, що на мобільних пристроях меню буде приховано.

У блоку `.nav-mobile` задані стилі для навігаційного меню на мобільних пристроях. Він має відступи (`margin-top` та `margin-left`) для розташування меню на певному місці на екрані. Є також медіа-запити для різних розмірів екрану та висоти, які змінюють властивості меню для досягнення бажаного вигляду.

Також з основних компонентів є створення сторінок. Веб-сайт ресторану піцерії складається з 11 сторінок сайту (Додаток А, Рис. 3.6).

Цей код представляє компонент `PizzaPage`, який використовується для відображення сторінки з піццею.

У компоненті використовуються хук `useContext` та `useEffect` з пакету `React`. Хуки – це функції, за допомогою яких можна «зачепитися» за стан та методи життєвого циклу `React` з функційних компонентів. `UseContext` використовується для отримання значень з контексту `AnywherePizzaContext`. З контексту витягуються змінні `pizza` та `sortProductsPizza`.

У хуці `useEffect` встановлено, що при монтуванні компонента відбувається прокручування вгору сторінки за допомогою `window.scrollTo(0, 0)`. Це означає, що після завантаження сторінки користувач буде перенаправлений на верхню частину сторінки.

Після створення всіх компонентів і сторінок, відбувається розгортання веб-сайту на сервер. Для даного дипломного проєкту було обрано сервер компанії GitHub. GitHub Pages – це служба розміщення статичних сайтів, яка бере файли HTML, CSS і JavaScript прямо зі сховища на GitHub, за бажанням запускає файли через процес створення та публікує веб-сайт.

GitHub Pages має кілька переваг:

1. Безкоштовна хостингова платформа: GitHub Pages надає безкоштовне хостингове рішення для статичних веб-сайтів. Можна розміщувати веб-сайт безкоштовно без необхідності власного сервера або платити за хостинг.
2. Легка налаштування: Розгортання веб-сайту на GitHub Pages просте і швидке. Для цього просто створюється репозиторій на GitHub, додаються файли, тоді сайт автоматично стає доступним за вказаною URL-адресою.
3. Інтеграція з Git: GitHub Pages легко інтегрується з системою контролю версій Git, що дозволяє зручно відстежувати та керувати змінами в веб-сайті. Можна використовувати функціонал Git для управління версіями, створення гілок (branches) та злиття змін.
4. Можливість використовувати своє доменне ім'я: Можна налаштувати GitHub Pages для використання свого власного доменного імені для веб-сайту. Це дозволяє мати професійний вигляд, якщо ви володієте власним доменом.

Існує три типи сайтів GitHub Pages: проєкт, користувач і організація. Сайти проєктів пов'язані з певним проєктом, розміщеним на GitHub, наприклад бібліотекою JavaScript або колекцією рецептів. Сайти користувачів і організацій підключені до певного облікового запису на GitHub.com [30].

Щоб опублікувати сайт піцерії, потрібно створити репозиторій, який належить особистому обліковому запису під назвою <username> github.io.

Щоб зробити замовлення на сайті, користувач може натиснути на кнопку перейти до меню, або вибрати категорії в швидкому доступі.

Після того як користувач вибрав категорію, він переходить на сторінку з їжею (наприклад піцею). Він має можливість обрати яку саме піцу він хоче, з типами тіста, а також має можливість вибрати розмір. Після цього користувач може перейти в корзину, або продовжити замовлення та вибрати ще якесь блюдо (Додаток А, Рис. 3.9).

Далі користувач переходить на сторінку замовлення, де має можливість переглянути своє замовлення та додати ще щось до замовлення, наприклад соус (Додаток А, Рис. 3.10).

Після цього користувачу потрібно заповнити ключові дані про себе, та вибрати тип доставки. Надається можливість доставки кур'єром, де потрібно вказати адресу користувача або самовивіз, коли користувач сам приходить до ресторану і забирає замовлення (Додаток А, Рис. 3.11).

Після успішного оформлення замовлення, користувачу демонструється форма підтвердження, де він має можливість натиснути кнопку Замовлення, і переглянути своє замовлення (Додаток А, Рис. 3.12-3.13).

Використання веб-сайту піцерії дозволяє користувачам отримати зручний та ефективний спосіб замовлення піци та доставки онлайн. Завдяки простому та зрозумілому інтерфейсу, користувачі можуть легко переглядати меню, вибрати піцу за своїми уподобаннями та оформлювати замовлення всього за кілька кроків. Також, користувачам надається можливість вибрати зручний спосіб доставки, введення адреси доставки та зазначення часу доставки. Остання сторінка підтвердження дозволяє користувачам переглянути своє замовлення, зробити остаточні зміни та підтвердити його.

Цей веб-сайт допомагає забезпечити зручну та швидку процедуру замовлення піци для користувачів та дозволяє піцерії ефективніше керувати замовленнями та покращити якість обслуговування своїх клієнтів.

ВИСНОВКИ

У кваліфікаційній роботі на тему "Розробка веб-дизайну та веб-сайту ресторану" було успішно реалізовано створення функціонального та привабливого веб-сайту для ресторану. Проєкт включав в себе розробку інтуїтивно зрозумілого інтерфейсу, де користувачі можуть з легкістю переглядати меню, замовляти піцу, вибирати додаткові послуги та здійснювати оплату.

В першому розділі було досліджено предметну область. Аналізуючи історію та сучасний стан ресторанного бізнесу в Україні, стає очевидним, що створення веб-сайту для піцерії є необхідним та актуальним кроком для розвитку цього бізнесу. Ресторанний сектор в Україні переживає значний розквіт, зростання конкуренції та зміну попиту клієнтів. Завдяки веб-сайту, піцерія може ефективно привертати нових клієнтів, поліпшувати обслуговування та підтримувати зв'язок зі своїми поточними клієнтами.

Особливості бізнес-процесів ресторанного бізнесу, такі як замовлення їжі онлайн, оптимізація доставки та самовивозу, електронна оплата та маркетингові акції, легко впроваджуються за допомогою веб-сайту. Це дає піцерії зручність, ефективність та конкурентну перевагу на ринку.

Аналіз обраних інструментів розробки додатку показав, що створення веб-сайту піцерії можливе з використанням сучасних технологій та платформ, таких як HTML, CSS, JavaScript та фреймворка React. Ці інструменти дозволяють створити привабливий, функціональний та зручний веб-сайт, який задовольнить потреби як клієнтів, так і власників піцерії.

В другому розділі було проаналізовано етапи створення веб-сайту, розроблено дизайн та базову архітектуру веб-сайту.

Етапи створення веб-сайту ресторану, такі як розробка архітектури, створення дизайну та програмування, є ключовими для створення функціонального та привабливого веб-сайту. Кожен з цих етапів вимагає уважного

планування та виконання з метою створення зручного та ефективного інтерфейсу для користувачів.

Дизайн веб-сайту піцерії має велике значення, оскільки він впливає на перше враження клієнтів, візуальну привабливість та легкість використання. Було розроблено дизайн згідно найкращих практик створення дизайну, таких як мінімалістичність, зручна навігація, привабливий інтерфейс тощо.

Архітектура веб-сайту піцерії має забезпечити зручну навігацію, логічну структуру та ефективне розміщення контенту. Організація розділів, меню, пошукових функцій та форм бронювання допомагає користувачам легко знайти необхідну інформацію та виконати потрібні дії.

В останньому розділі вже було спроектовано базу даних та заповнено її. Моделювання моделі бази даних є ключовим етапом в розробці веб-сайту ресторану. Визначення структури бази даних, таблиць, взаємозв'язків та полів дозволяє зберігати та організовувати всю необхідну інформацію про меню, клієнтів, замовлення та інші аспекти ресторанної діяльності.

Також було розроблено сам веб-сайт піцерії, використовуючи сучасні технології розробки. Програмування, використання веб-технологій та інтеграція з базою даних дозволили створити динамічний та інтерактивний веб-сайт, який забезпечує зручну навігацію, функціональність пошуку та замовлення.

В результаті виконання кваліфікаційної роботи було отримано ключові навички для розробки веб-дизайну та веб-сайту, що дозволить бути більш професійним в подальшій діяльності розробника програмного забезпечення.

Сайт ресторану можна переглянути за посиланням:

<https://xjlott4eh9l.github.io/anywhere-pizza/>

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автоматизація піцерії - програма обліку в піцерії PalmaBox. *PalmaBoxHoreca*. URL: <https://horeca.palmabox.com/avtomatizatsiya-pizerii-programma/> (дата звернення: 12.04.2023).
2. АйТі Кафе, автоматизація кафе та ресторану за 1\$. URL: <http://itkafe.com/index-en.php> (дата звернення: 10.04.2023).
3. Алексеєнко В. С. Розвиток ресторанного бізнесу в Україні : thesis. 2019. URL: <https://er.knutd.edu.ua/handle/123456789/14133> (дата звернення: 12.05.2023).
4. Бізнес-план з проекту: відкриття піцерії-ресторану швидкого. *Портал фінанси-кредит*. URL: <https://economics.studio/biznes-planirovanie/biznes-plan-proektu-vidkrittya-pitseriji-75142.html> (дата звернення: 22.04.2023).
5. Генерація SQL-запиту засобами MySQL-сервера. *Dou.ua*. URL: <https://dou.ua/lenta/articles/sql-query-mysql-server/>.
6. Кознова Олександра. Як відновлюється та адаптується український бізнес: результати дослідження | Think brave. *Think brave* | *Останні новини бізнесу України*. URL: https://biz.ligazakon.net/news/214925_yak-vdnovlyutsya-ta-adaptutsya-ukranskiy-bznes-rezultati-doslidzhennya (дата звернення: 01.05.2023).
7. Кращі системи управління базами даних. *Онлайн навчання IT спеціальностям* | *IT курси онлайн* | *IT STEP*. URL: https://cloud.itstep.org/blog_3/5-best-database-management-systems-in-2022 (дата звернення: 17.04.2023).
8. Лисецький Ю. М. Інформаційні технології в управлінні та обробці інформації : монографія. Київ : LAM&K, 2018. 268 с.
9. Найважливіші архітектурні шаблони, які необхідно знати. *DevZone*. URL: <https://devzone.org.ua/post/nayvazhlivishi-arkhitekturni-shablони-yaki-neobkhidno-znati> (дата звернення: 26.04.2023).

10. Обозна А. О., Шабельник Н. М., Федотов І. А. Аналіз сучасного ринку ресторанних послуг та ефективність їх застосування на підприємстві. 13-те вид. Інвестиції: практика та досвід, 2019. - 213 с.
11. Особливості розвитку ресторанного господарства в Україні. *Все про туризм: туристична бібліотека*. URL: https://tourlib.net/statti_ukr/yashyna.htm (дата звернення: 29.03.2023).
12. Офіційний сайт статистики України. *Держана служба статистики України*. URL: <https://www.ukrstat.gov.ua/> (дата звернення: 12.05.2023).
13. Офіційний сайт MySQL. URL: <https://www.mysql.com/> (дата звернення: 12.05.2023).
14. Пильщиков В. А. Економічне обґрунтування проєкту створення кафе «Піцерія» : thesis. 2021. URL: <http://dspace.oneu.edu.ua/jspui/handle/123456789/5061> (дата звернення: 05.04.2023).
15. Поворознюк І. Інноваційні технології в ресторанному бізнесі. *Економіка та суспільство*. 2021. № 30. URL: <https://doi.org/10.32782/2524-0072/2021-30-17> (дата звернення: 27.04.2023).
16. Прищепа Я. Бізнес під час пандемії: з чого почати та як вижити. *Суспільні новини*. URL: <https://suspilne.media/139155-biznes-pid-cas-pandemii-z-cogo-pocati-ta-ak-viziti> (дата звернення: 16.04.2023).
17. Про затвердження Технічного завдання на створення Автоматизованої інформаційної системи "Офіційний веб-сайт Центральної виборчої комісії" : Постанова Центр. вибор. коміс. від 16.08.2018 р. № 123. URL: <https://zakon.rada.gov.ua/laws/show/v0123359-18#Text> (дата звернення: 13.04.2023).
18. Розвиток ресторанного господарства в Україні: організаційні, інноваційні і маркетингові чинники підвищення конкурентоспроможності підприємств. *Електронний архів Полтавського університету економіки і торгівлі*.

URL: <http://dspace.puet.edu.ua/bitstream/123456789/10448/1/M.%20Ю.%20Барна,%20ОЛ.%20К.%20Гліненко,%20Ю.%20А.%20Дайновський.pdf> (дата звернення: 12.04.2023).

19. СУБД (Системи управління базами даних) - що це, види та функції. *Highload.today* - *медиа* для *разработчиков*. URL: <https://highload.today/uk/subd-yaki-buvayut-yak-vibrati/> (дата звернення: 12.04.2023).

20. Точна різниця між SQL та NoSQL (знайте, коли використовувати NoSQL та SQL) - Інший. *Огляди, Ігри, Розваги, Може 2023*. URL: <https://uk.myservername.com/sql-vs-nosql-exact-differences> (дата звернення: 29.04.2023).

21. Турчиняк М. К., Форись О. Г. Розвиток підприємств ресторанного господарства в західному регіоні України. *Індустрія туризму і гостинності в Центральній та Східній Європі*. 2021. № 3. С. 68–73. URL: <https://doi.org/10.36477/tourismhospcee-3-10> (дата звернення: 25.04.2023).

22. Управління ресурсами проекту. *Pidru4niki*. URL: https://pidruchniki.com/67963/menedzhment/upravlinnya_resursami_proektu (дата звернення: 19.04.2023).

23. Чумак М. А., Chumak M. A. Визначення системи забезпечення конкурентоспроможності підприємства ресторанного господарства (на прикладі ТОВ “Україна” ресторан): *master's thesis*. 2021. URL: <http://elartu.tntu.edu.ua/handle/lib/36529> (дата звернення: 18.04.2023).

24. 10 найкращих інструментів проектування баз даних для побудови складних моделей даних - Інший. *Огляди, Ігри, Розваги, Може 2023*. URL: https://uk.myservername.com/top-10-database-design-tools-build-complex-data-models#1_Visual_Paradigm_ERD_Tools (дата звернення: 12.04.2023).

25. 2023 буде роком виживання для ресторанного бізнесу - Ольга Насонова. *PostEat* – *Профільне гастрономічне видання* - *PostEat*.

URL: <https://posteat.ua/news/2023-bude-rokom-vizhivannya-dlya-restorannogo-biznesu-olga-nasonova/> (дата звернення: 22.03.2023).

26. 7 steps of web design development process | infolio. *Infolio*. URL: <https://www.infolio.co/use-cases/design-team-management/7-steps-of-web-design-development-process-template> (date of access: 12.04.2023).

27. An Introduction to JavaScript. *The Modern JavaScript Tutorial*. URL: <https://javascript.info/intro> (date of access: 17.04.2023).

28. Celentano D. How do QSR, casual dining, and fast casual restaurants differ?. *LiveAbout*. URL: <https://www.thebalancesmb.com/types-of-restaurant-formats-1326193> (date of access: 12.05.2023).

29. Chawre H. Understanding web development process: full guide in 6 steps. *Hire the World's Most Deeply Vetted Remote Developers | Turing*. URL: <https://www.turing.com/resources/web-development-process> (date of access: 12.05.2023).

30. Creating a GitHub Pages site - GitHub Docs. *GitHub Docs*. URL: <https://docs.github.com/en/pages/getting-started-with-github-pages/creating-a-github-pages-site> (date of access: 18.04.2023).

31. How to design a web application: software architecture 101. *Educative: Interactive Courses for Software Developers*. URL: <https://www.educative.io/blog/how-to-design-a-web-application-software-architecture-101> (date of access: 02.04.2023).

32. HTML basics - Learn web development | MDN. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics (date of access: 12.05.2023).

33. Hutsulyak O. Why use react for web development: 10 reasons to apply. *Blog | TechMagic*. URL: <https://www.techmagic.co/blog/why-we-use-react-js-in-the-development/> (date of access: 12.04.2023).

34. Oleynik P., Nikolenko O., Yuzefova S. Information system for fast food. 2nd ed. Restaurants Engineering and Technology, 2015. 191 с.
35. Sullivan D. Project management in manufacturing using IDEF0. *International journal of project management*. 1991. Vol. 9, no. 3. P. 162–168. URL: [https://doi.org/10.1016/0263-7863\(91\)90041-s](https://doi.org/10.1016/0263-7863(91)90041-s) (date of access: 11.04.2023).
36. Pizza websites - 20+ best pizza web design ideas 2023 | 99designs. 99designs. URL: <https://99designs.com/inspiration/websites/pizza> (date of access: 03.04.2023).
37. SQL чи nosql - ось в чому питання - альтернативна наука віталія сема. *Альтернативна Наука Віталія Сема*. URL: <https://alternativescience.net/programming/242-sql-chy-nosql-os-v-chomu-pytannya/> (дата звернення: 12.04.2023).
38. What is CSS, how does it work and what is it used for? - skillcrush. *Skillcrush*. URL: <https://skillcrush.com/blog/css/> (date of access: 24.04.2023).
39. What is web design? The ultimate guide to website design [2020] - pagecloud blog - web design & small business tips for your website. *Pagecloud.com*. URL: <https://www.pagecloud.com/blog/web-design-guide> (date of access: 24.04.2023).
40. Zemlina Y., Lifirenko O. Trends in business hotel development in Ukraine. *Restaurant and hotel consulting. innovations*. 2019. Vol. 2, no. 1. P. 121–131. URL: <https://doi.org/10.31866/2616-7468.2.1.2019.170430> (date of access: 29.03.2023).

ДОДАТКИ

Додаток А

```

6    //pages
7    import Home from './pages/Home';
8    import PizzaPage from './pages/PizzaPage';
9    import SushiPage from './pages/SushiPage';
10   import DrinksPage from './pages/DrinksPage';
11   import SnacksPage from './pages/SnacksPage';
12   import SaucesPage from './pages/SaucesPage';
13   import DessertsPage from './pages/DessertsPage';
14   import ComboPage from './pages/ComboPage';
15   import CartPage from './pages/CartPage';
16   import OrdersPage from './pages/OrdersPage';
17   import AboutPage from './pages/AboutPage';
18
19   //components
20   import Header from './components/Header/Header';
21   import SideCart from './components/SideCart/SideCart';
22   import Footer from './components/Footer/Footer';
23   import CartAdded from './components/CartAdded/CartAdded';
24   import ScrollTop from './components/ScrollTop/ScrollTop';
25
26   function App() {
27     //products
28     const [pizza, setPizza] = useState([]);
29     const [sushi, setSushi] = useState([]);
30     const [drinks, setDrinks] = useState([]);
31     const [snacks, setSnacks] = useState([]);
32     const [sauces, setSauces] = useState([]);
33     const [desserts, setDesserts] = useState([]);
34     const [combo, setCombo] = useState([]);
35
36     const [orders, setOrders] = useState(JSON.parse(window.localStorage.getItem('orders'))) || [];
37     const [cartItems, setCartItems] = useState(JSON.parse(window.localStorage.getItem('cartItems'))) || [];
38     const [ordersId, setOrdersId] = useState(JSON.parse(window.localStorage.getItem('ordersId'))) || 1;
39     const [isMobileMenuOpen, setIsMobileMenuOpen] = useState(false);
40     const [isCartOpen, setIsCartOpen] = useState(false);
41     const [force, setForce] = useState(1);
42     const [displayNav, setDisplayNav] = useState(false);

```

Рис. 3.3 Частина структури коду базового компоненту App

```

1  import './Nav.scss';
2  import { useContext } from 'react';
3  import { Link } from 'react-router-dom';
4  import { AnywherePizzaContext } from '../context.js';
5
6
7  function Nav({ navType, flexType }) {
8      const { setDisplayNav, navLinkActive, setNavLinkActive, navLinks, setIsMobileMenuOpen } = useContext(AnywherePizzaContext);
9
10     return (
11         <nav className={navType}>
12             <ul className={navType + flexType}>
13                 {navLinks.map(link => {
14                     const isActive = navLinkActive === link.title;
15                     return (
16                         <li className={` ${navType}__item` } key={link.name}>
17                             <Link
18                                 className={isActive ? ` ${navType}__link ${navType}__link--active` : ` ${navType}__link`}
19                                 onClick={() => {
20                                     setDisplayNav(true);
21                                     setNavLinkActive(link.title);
22                                     setIsMobileMenuOpen(false);
23                                 }}
24                                 to={link.href}>
25                                 {link.name}
26                             </Link>
27                         </li>
28                     )
29                 })}
30             </ul>
31         </nav>
32     )
33 }
34
35 export default Nav;

```

Рис. 3.4 Код компоненту Nav

```

1  @import '../variables.scss';
2
3  .nav {
4      @media(max-width: 991px) {
5          display: none;
6      }
7      &__list {
8          display: flex;
9      }
10     &__list-column {
11         display: flex;
12         flex-direction: column;
13         .nav__item {
14             font-size: 16px;
15             margin-bottom: 30px;
16         }
17     }
18     &__item {
19         font-size: 16px;
20     }
21     &__item:not(:last-child) {
22         margin-right: 32px;
23         @media(max-width: 1170px) {
24             margin-right: 20px;
25         }
26     }
27     &__link {
28         padding-bottom: 5px;
29         transition: border-bottom 0.1s ease-in-out;
30         &--active {
31             border-bottom: 3px solid $black-color;
32         }
33         &:hover {
34             border-bottom: 3px solid $black-color;
35         }
36     }
37 }
38

```

Рис. 3.5. Стилізація компоненту Nav за допомогою SCSS

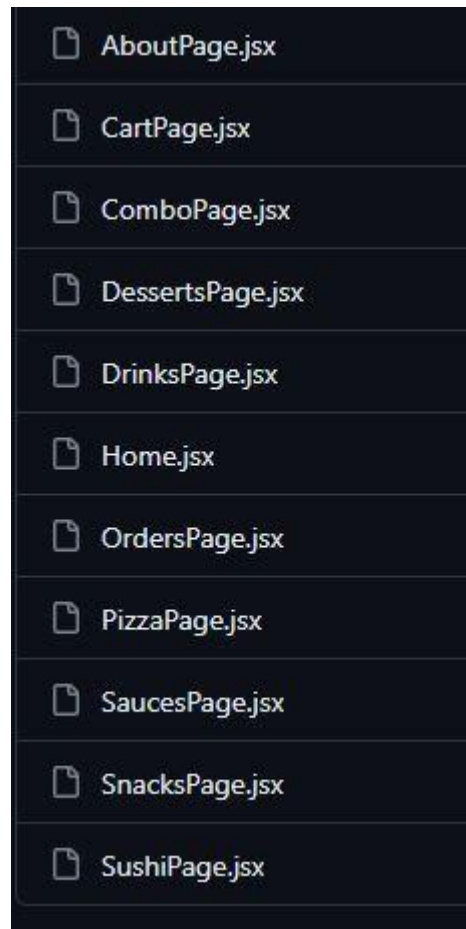


Рис. 3.6 Сторінки веб-сайту піцерії

```

1  import { useContext, useEffect } from "react";
2  import Catalog from "../components/Catalog/Catalog";
3  import { AnywherePizzaContext } from "../context";
4
5  export default function PizzaPage() {
6      const { pizza, sortProductsPizza } = useContext(AnywherePizzaContext);
7
8      useEffect(() => {
9          window.scrollTo(0, 0)
10     }, [])
11
12     return (
13         <Catalog
14             title="Піцца"
15             data={pizza}
16             sortFunction={sortProductsPizza}
17         />
18     )
19 }

```

Рис. 3.7 Код сторінки Піца в React

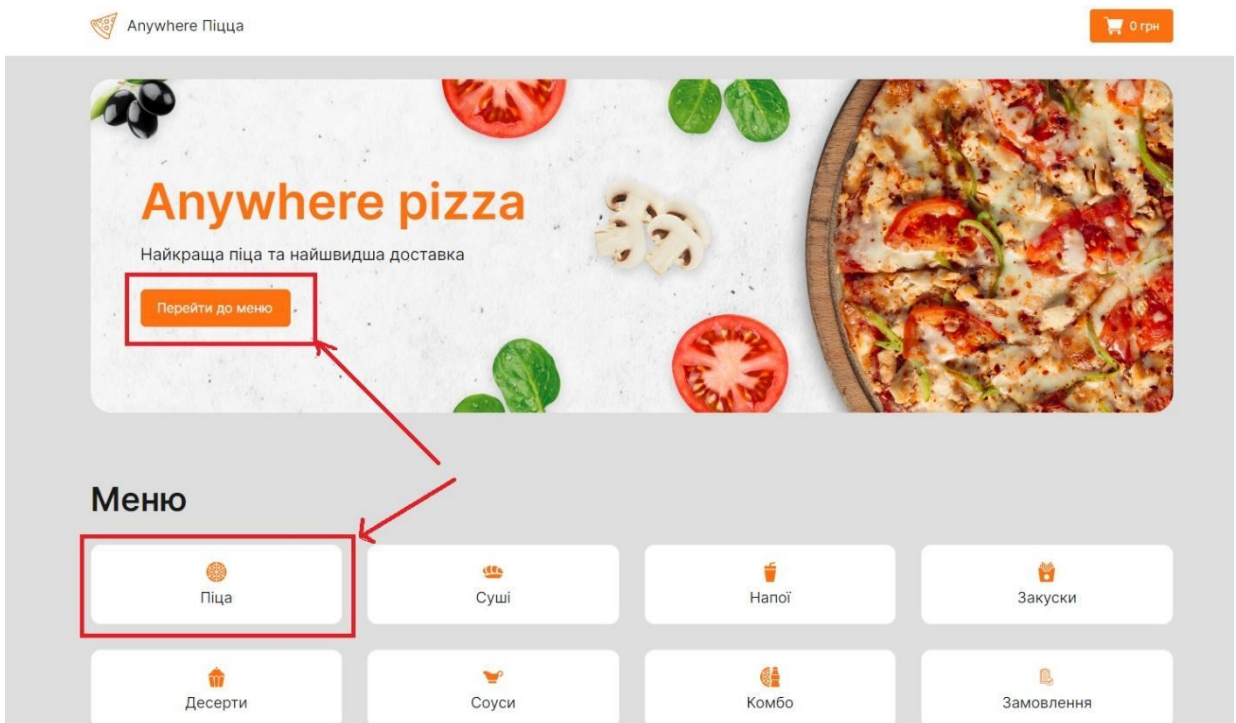


Рис. 3.8 Головна сторінка замовлення

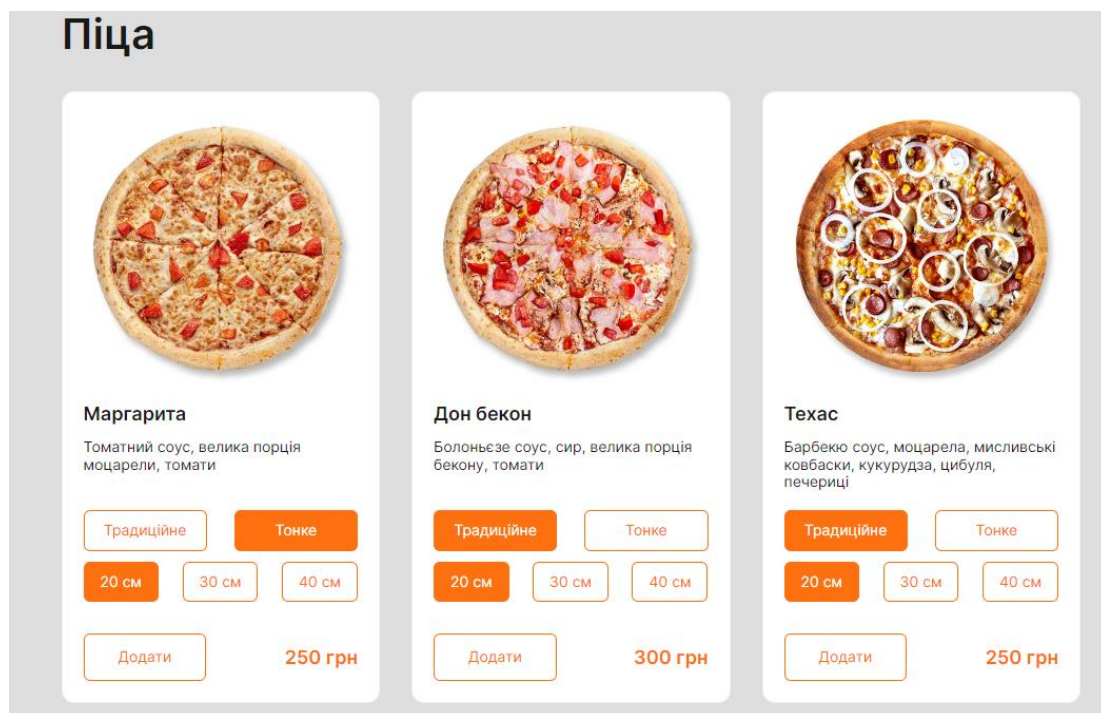


Рис. 3.9. Сторінка вибору піци

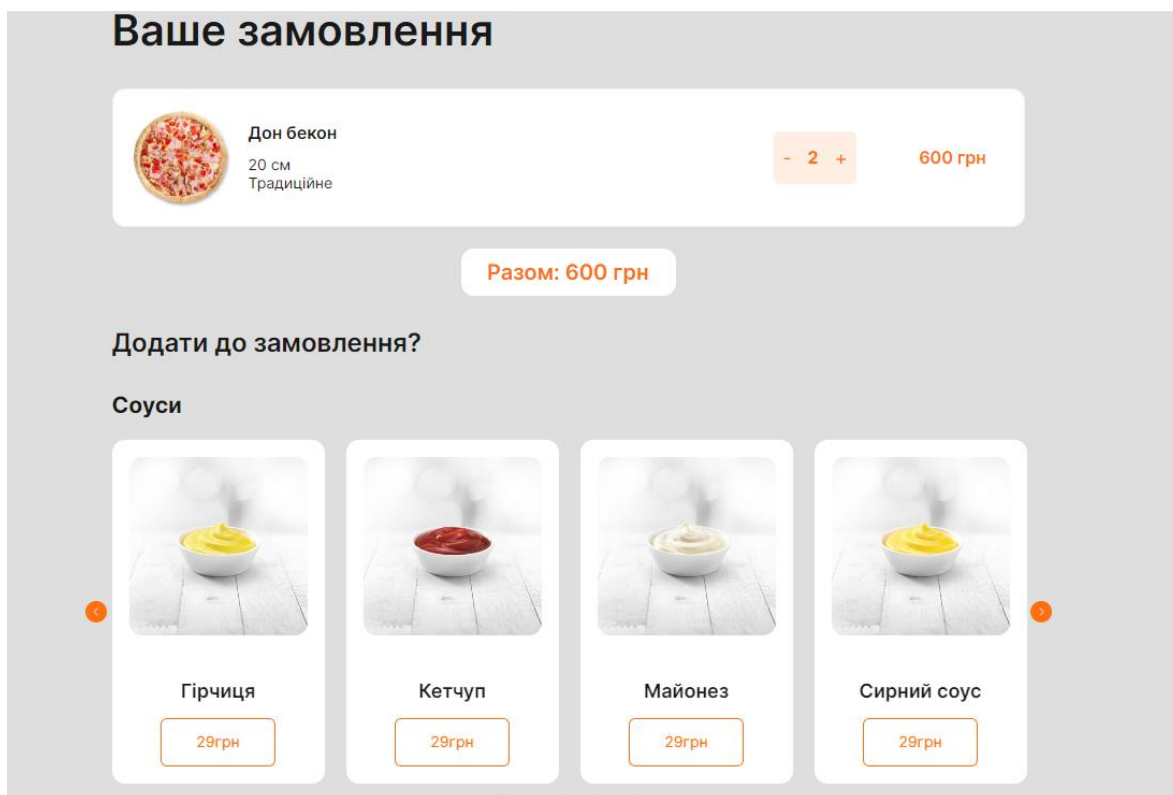


Рис. 3.10. Сторінка замовлення

Про вас

Ім'я

Телефон

Пошта

Доставка

Вулиця*

Будинок*

Доставка **Самовивіз**

Разом: 629 грн

Оформити замовлення

Рис. 3.11 Форма заповнення даних для доставки

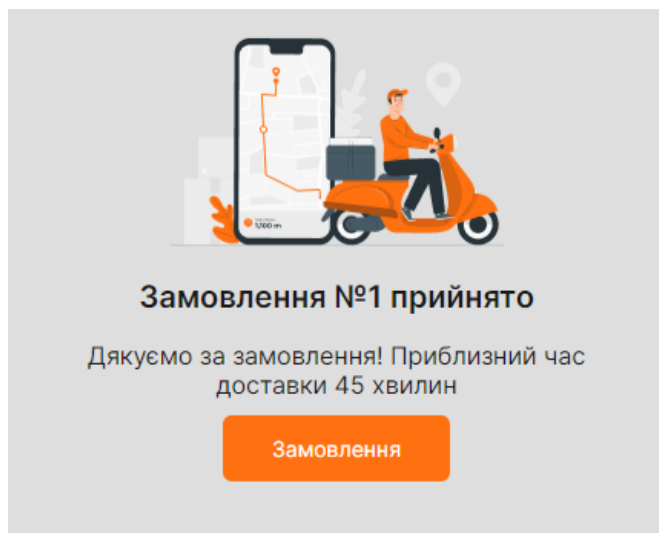


Рис. 3.12 Форма підтвердження замовлення

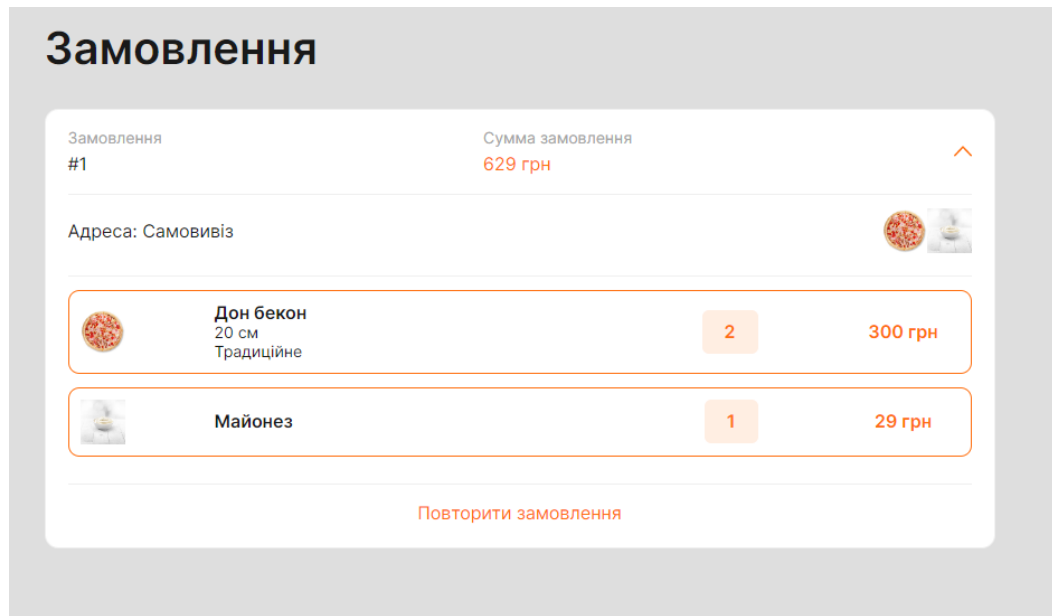


Рис. 3.13. Форма перегляду замовлення

Додаток Б

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-
scale=1.0, user-scalable=0" />
    <meta name="copyright" content="Anywhere Піцца" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />

```

```

<link rel="preconnect" href="https://fonts.googleapis.com" />
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
<link
href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&display=swa
p" rel="stylesheet" />
<title>Anywhere pizza</title>
</head>
<body>
<div id="root"></div>
</body>
</html>
{
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
import './BurgerBtn.scss';
import { useContext } from 'react';

```

```

import { AnywherePizzaContext } from '../context';

export default function BurgerBtn() {
  const { isMobileMenuOpen, setIsMobileMenuOpen } =
    useContext(AnywherePizzaContext);
  return (
    <button
      className={isMobileMenuOpen ? 'burger-btn burger-btn--active' : 'burger-btn'}
      onClick={() => setIsMobileMenuOpen(prev => !prev)}
    >
      <span className='burger-btn__item'></span>
      <span className='burger-btn__item'></span>
      <span className='burger-btn__item'></span>
    </button>
  )
}

@import '../variables.scss';

.burger-btn {
  display: none;
  position: absolute;
  left: 50%;
  transform: translate(-50%);
  z-index: 100;
  padding: 2px;
  @media(max-width: 991px) {
    display: block;
  }
}

```

```

&--active {
  .burger-btn__item:nth-child(2) {
    opacity: 0;
  }
  .burger-btn__item:nth-child(1) {
    transform: translateY(8px) rotate(45deg);
  }
  .burger-btn__item:nth-child(3) {
    transform: translateY(-8px) rotate(-45deg);
  }
}
&__item {
  display: block;
  width: 28px;
  height: 3px;
  margin: 5px auto;
  background-color: $primary-color;
  transition: all 0.5s ease-in-out;
}
}
import './Card.scss';
import { useState, useContext } from 'react';
import { AnywherePizzaContext } from '../context';

function Card({id, imgUrl, title, compound, price}) {
  const { onCartAdded, cartItems } = useContext(AnywherePizzaContext);

  function onCart() {

```

```

    onCartAdded({id, title, imgUrl, compound, price, counter: 1});
  }

function getProductCounter() {
  const product = cartItems.find(cartObj => cartObj.id === id);
  if(product) {
    return product.counter;
  }
}

return (
  <div className="card">
    <div className='card__content'>
      <img src={imgUrl} alt={title} />
      <h3 className="card__title">{title}</h3>
      <p className="card__setup">{compound}</p>
    </div>
    <div className='card__pick'>
      <button className={getProductCounter() ? 'btn btn--active' : 'btn'}
onClick={onCart}>
        {
          getProductCounter() ? (
            <div>Додано - <span
className='card__counter'>{getProductCounter()}</span></div>
            ) : 'Додати'
          }
        </button>
        <span className="card__price">{price} грн</span>

```

```

        </div>
    </div>
)
}

```

```

function PizzaCard({id, imgUrl, title, compound, prices, sizes}) {
    const { onCartAddedPizza, cartItems } = useContext(AnywherePizzaContext);
    const doughTypes = ['Традиційне', 'Тонке'];
    const [type, setType] = useState(doughTypes[0]);
    const [activeSize, setActiveSize] = useState(sizes[0]);
    const [price, setPrice] = useState(prices[0]);

    function onCart() {
        onCartAddedPizza({id, title, imgUrl, compound, price, counter: 1, activeSize,
type});
    }

    function getProductCounter() {
        const initialValue = 0;
        const productsInCart = cartItems
            .filter(product => product.id === id)
            .reduce((summ, nextProduct) => summ + nextProduct.counter, initialValue);

        if(productsInCart) {
            return productsInCart;
        }
    }
}

```

```

function sizeHandler(size) {
  setActiveSize(size);

  switch(size) {
    case 20:
      setPrice(prices[0])
      break
    case 30:
      setPrice(prices[1])
      break
    case 40:
      setPrice(prices[2])
      break
    default:
      console.log('hello')
      break
  }
}

return (
  <div className="card">
    <div className='card__content'>
      <img src={imageUrl} alt={title} />
      <h3 className="card__title">{title}</h3>
      <p className="card__setup">{compound}</p>
      {sizes &&
        <>
          <div className='card__types'>

```

```

    {doughtypes.map(doughtype => {
      const isActive = doughtype === type;
      return (
        <button
          key={doughtype}
          className={isActive ? 'card__btn card__btn--active
card__btn--type' : 'card__btn card__btn--type'}
          onClick={() => setType(doughtype)}>
            {doughtype}
          </button>
        )
      )}}
</div>
<div className='card__sizes'>
  {sizes.map(size => {
    const isActive = size === activeSize;
    return (
      <button
        key={size}
        className={isActive ? 'card__btn card__btn--active' :
'card__btn'}
        onClick={() => sizeHandler(size)}>
          {size} cm
        </button>
      )
    )}}
</div>
</>

```

```

    }
  </div>
  <div className='card__pick'>
    <button className={getProductCounter() ? 'btn btn--active' : 'btn'}
onClick={onCart}>
      {
        getProductCounter() ? (
          <div>Додано - <span
className='card__counter'>{getProductCounter()}</span></div>
        ) : 'Додати'
      }
    </button>
    <span className="card__price">{price} грн</span>
  </div>
</div>
)
}

```

```

export {Card, PizzaCard};
@import '../variables.scss';

```

```

.card {
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  max-width: 350px;
  background-color: $white-color;
  border: 1px solid $gray-line-color;

```

```

border-radius: 12px;
transition: transform 0.3s ease-in-out;
@media(max-width: 991px) {
    margin: 0 auto;
}
@media(max-width: 770px) {
    width: 500px;
    max-width: 70%;
}
@media(max-width: 550px) {
    max-width: 350px;
    width: 350px;
}
@media(max-width: 420px) {
    width: 300px;
}
@media(max-width: 350px) {
    width: 220px;
}

&:hover {
    transform: translateY(-10px);
    @media(max-width: 991px) {
        transform: none;
    }
}

&__content {
    padding: 20px;

```

```

padding-bottom: 0;
@media(max-width: 991px) {
    text-align: center;
}
& img {
    margin: 0 auto;
    margin-bottom: 16px;
    width: 258px;
    height: auto;
    @media(max-width: 991px) {
        margin: 0 auto;
        margin-bottom: 16px;
    }
    @media(max-width: 450px) {
        width: 238px;
    }
    @media(max-width: 350px) {
        width: 150px;
    }
}
}
&__title {
    font-weight: 600;
    font-size: 18px;
    margin-bottom: 12px;
}
&__setup {
    font-weight: 400;

```

```

font-size: 14px;
margin-bottom: 16px;
max-width: 300px;
min-height: 52px;
@media(max-width: 770px) {
    margin: 0 auto;
    margin-bottom: 16px;
}
@media(max-width: 420px) {
    max-width: 200px;
    margin: 0 auto;
    margin-bottom: 16px;
}
}
&__price {
    font-weight: 600;
    font-size: 18px;
    color: $primary-color;
    @media(max-width: 350px) {
        font-size: 14px;
    }
}
&__pick {
    padding-left: 20px;
    padding-right: 20px;
    padding-bottom: 20px;
    display: flex;
    justify-content: space-between;

```

```

    align-items: center;
}
&__counter {
    display: inline-block;
    text-align: center;
    padding-top: 1px;
    padding-right: 1px;
    min-width: 20px;
    height: 20px;
    border-radius: 50%;
    color: $primary-color;
    background-color: $white-color;
}
&__sizes {
    display: flex;
    justify-content: space-between;
    margin-bottom: 30px;
    @media(max-width: 770px) {
        max-width: 80%;
        margin-left: auto;
        margin-right: auto;
    }
    @media(max-width: 420px) {
        max-width: 100%;
    }
}
&__types {
    display: flex;

```

```

    justify-content: space-between;
    margin-bottom: 10px;
}
&__btn {
    padding: 10px 15px;
    color: $primary-color;
    background-color: $white-color;
    border: 1px solid $primary-color;
    border-radius: 6px;
    transition: color, background-color 0.3s linear;
    &--active {
        color: $white-color;
        background-color: $primary-color;
    }
    &--type {
        width: 45%;
    }
    @media(min-width: 992px) {
        &:hover {
            color: $white-color;
            background-color: $primary-color;
        }
    }
}
}

import './CardPlaceholder.scss';

function PizzaPlaceholder() {

```

```

return (
  <div className="pizzaPlaceholder">
    <div className='pizzaPlaceholder__round'></div>
    <div className='pizzaPlaceholder__title'></div>
    <div className='pizzaPlaceholder__compound'>
      <span></span>
      <span></span>
      <span></span>
    </div>
    <div className='pizzaPlaceholder__types'>
      <span></span>
      <span></span>
    </div>
    <div className='pizzaPlaceholder__sizes'>
      <span></span>
      <span></span>
      <span></span>
    </div>
    <div className='pizzaPlaceholder__bottom'>
      <div className='pizzaPlaceholder__btn'></div>
      <span></span>
    </div>
  </div>
)
}

```

```

function CardPlaceholder() {
  return (

```

```

    <div className="cardPlaceholder">
      <div className="cardPlaceholder__img"></div>
      <div className='pizzaPlaceholder__title'></div>
      <div className='cardPlaceholder__compound'></div>
      <div className='pizzaPlaceholder__bottom'>
        <div className='pizzaPlaceholder__btn'></div>
        <span></span>
      </div>
    </div>
  )
}

```

```
export { PizzaPlaceholder, CardPlaceholder}
```

```
@import '../variables.scss';
```

```

.pizzaPlaceholder {
  max-width: 350px;
  border: 1px solid $gray-line-color;
  border-radius: 12px;
  padding: 20px;
  background-color: $white-color;
  @media(max-width: 991px) {
    margin: 0 auto;
  }
  @media(max-width: 770px) {
    width: 500px;
    max-width: 70%;
  }
}

```

```

@media(max-width: 550px) {
    max-width: 350px;
    width: 350px;
}
@media(max-width: 420px) {
    width: 300px;
}
@media(max-width: 350px) {
    width: 220px;
}
@keyframes loading {
    0% {
        background-color:$gray-text-color;
    }
    50% {
        background-color: rgba($gray-text-color, 0.5);
    }
    100% {
        background-color:$gray-text-color;
    }
}
&__round {
    width: 256px;
    height: 256px;
    margin: 0 auto;
    border-radius: 50%;
    background-color: $gray-text-color;
    margin-bottom: 16px;
}

```

```

transition: background-color 0.3s linear;
animation: loading 2s infinite linear;
@media(max-width: 450px) {
    width: 238px;
    height: 238px;
}
}
&__title {
    width: 100%;
    height: 22px;
    margin-bottom: 12px;
    background-color: $gray-text-color;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
}
&__compound {
    margin-bottom: 16px;
    span {
        display: block;
        width: 100%;
        height: 10px;
        background-color: $gray-text-color;
        transition: background-color 0.3s linear;
        animation: loading 2s infinite linear;
        &:not(:last-child) {
            margin-bottom: 2px;
        }
    }
}

```

```

}
&__types {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 16px;
  span {
    display: block;
    width: 45%;
    height: 38px;
    border-radius: 6px;
    background-color: $gray-text-color;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
  }
}
&__sizes {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 16px;
  span {
    width: 70px;
    height: 30px;
    border-radius: 6px;
    background-color: $gray-text-color;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
  }
}

```

```

    }
}
&__bottom {
    display: flex;
    justify-content: space-between;
    align-items: center;
    span {
        width: 70px;
        height: 22px;
        background-color: $gray-text-color;
        transition: background-color 0.3s linear;
        animation: loading 2s infinite linear;
    }
}
&__btn {
    width: 117px;
    height: 45px;
    background-color: $gray-text-color;
    border-radius: 6px;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
}
}

```

```

.cardPlaceholder {
    max-width: 350px;
    border: 1px solid $gray-line-color;
    border-radius: 12px;
}

```

```
padding: 20px;
background-color: $white-color;
@media(max-width: 991px) {
    margin: 0 auto;
}
@media(max-width: 770px) {
    width: 500px;
    max-width: 70%;
}
@media(max-width: 550px) {
    max-width: 350px;
    width: 350px;
}
@media(max-width: 420px) {
    width: 300px;
}
@media(max-width: 350px) {
    width: 220px;
}
@keyframes loading {
    0% {
        background-color:$gray-text-color;
    }
    50% {
        background-color: rgba($gray-text-color, 0.5);
    }
    100% {
        background-color:$gray-text-color;
    }
}
```

```

    }
  }
  &__img {
    width: 258px;
    height: 258px;
    margin: 0 auto;
    margin-bottom: 12px;
    border-radius: 6px;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
  }
  &__compound {
    width: 100%;
    height: 40px;
    margin-bottom: 16px;
    transition: background-color 0.3s linear;
    animation: loading 2s infinite linear;
  }
}

import './CartAdded.scss';
import { useContext } from 'react';
import { AnywherePizzaContext } from '../context';

function CartAdded() {
  const { addedToCart } = useContext(AnywherePizzaContext)
  return <div className={addedToCart ? 'cart-added cart-added--active' : 'cart-
added'}>Товар додано в корзину</div>
}

```

```
export default CartAdded;

@import '../variables.scss';

.cart-added {
  width: 100%;
  max-width: 250px;
  border-radius: 10px;
  padding: 10px 20px;
  font-size: 16px;
  text-align: center;
  position: fixed;
  top: 100px;
  left: 85%;
  transform: translate(-50%, 0);
  color: $primary-color;
  background-color: $primary-light-color;
  opacity: 0;
  transition: opacity 0.3s ease-in-out;
  z-index: 5;
  @media(max-width: 900px) {
    left: 80%;
  }
  @media(max-width: 770px) {
    max-width: 210px;
    left: 69%;
    font-size: 14px;
  }
}
```

```

@media(max-height: 900px) {
  top: 90px;
}
@media(max-width: 600px) {
  left: 50%;
  top: 100px;
}
@media(max-width: 500px) {
  font-size: 14px;
}
&--active {
  opacity: 1;
}
}

import './CartItem.scss';
import { useContext } from 'react';
import { AnywherePizzaContext } from '../../context';

function CartItem({ id, imgSrc, title, price, counter, activeSize, type }) {
  const { onClickMinus, onClickPlus } = useContext(AnywherePizzaContext);

  return (
    <div className="cart-item">
      <div className='cart-item__info'>
        <img className='cart-item__img' src={imgSrc} alt={title} />
        <div className='cart-item__description'>
          <h2 className='cart-item__title'>{title}</h2>
          {activeSize} && <p className='cart-item__size'>{activeSize} cm</p>
        </div>
      </div>
    </div>
  );
}

```

```

        {type && <p className='cart-item__size'>{type}</p>}
      </div>
    </div>
    <div className='cart-item__summary'>
      <div className='cart-item__counter'>
        <button className="cart-item__btn" onClick={() => onClickMinus({ id,
imgSrc, title, price, counter, activeSize, type })} >-</button>
        <h4 className="cart-item__amount">{counter}</h4>
        <button className="cart-item__btn" onClick={() => onClickPlus({ id,
imgSrc, title, price, counter, activeSize, type })} >+</button>
      </div>
      <span className="cart-item__price">{price * counter} рpH</span>
    </div>
  </div>
)
}

```