МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Факультет автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»
Директор інституту(декан факультету)
(підпис) Андрій ФОРСЮК (ім'я та Прізвище)
«07» 02 2025 р.

«До захисту допущено»
Завідувач кафедри
(підпис) Сергій ГРИБКОВ (ім'я та Прізвище)
«07» 02 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки»
(код та назва спеціальності)
освітньо-професійної програми«Комп'ютерні науки»

на тему: Розроблення веб-порталу з обслуговування клієнтів мережі ресторанів "Bubbly"

Виконав: здобувач 5 курсу, групи ЗКН5-2
Воронова Марина Олександрівна (підпис)
(прізвище, ім'я, по батькові повністю)

Керівник Грама Михайло Петрович (підпис)
(прізвище , ім'я та по батькові повністю)

Консультанти
(ім'я та прізвище) (підпис)
(ім'я та прізвище) (підпис)
(ім'я та прізвище) (підпис)

Рецензент (ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач (підпис)

Київ - 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Факультет автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Андрій ФОРСЮК	_____ Сергій ГРИБКОВ
(підпис) (ім'я та Прізвище)	(підпис) (ім'я та Прізвище)
«___» _____ 2025 р.	«___» _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки»
(код та назва спеціальності)
освітньо-професійної програми«Комп'ютерні науки»

на тему: Розроблення веб-порталу з обслуговування клієнтів мережі ресторанів "Bubbly"

Виконав: здобувач 5 курсу, групи ЗКН5-2

Воронова Марина Олександрівна
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник Грама Михайло Петрович
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач_____
(підпис)

Київ - 2025 р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Факультет автоматизації і комп'ютерних систем

Кафедра _Інформаційних технологій, штучного інтелекту і кібербезпеки

Освітній ступінь _Бакалавр_____

Спеціальність_122 «Комп'ютерні науки»_____

(код і назва)

Освітньо-професійна програма __«Комп'ютерні науки»___

(назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Грибков С.В
«_07__»_жовтня_ 2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

_____Воронова Марина Олександрівна_____

(прізвище, ім'я, по батькові)

1. Тема роботи __Розроблення веб-порталу з обслуговування клієнтів мережі_ ресторанів "Bubbly"_____

керівник роботи __Грама Михайло Петровий, PhD, старший викладач,

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

затверджені наказом закладу вищої освіти від «_07_ » жовтня 2024 р. № 886-кс_

2. Строк подання здобувачем роботи: _03.02.2025__

3. Вихідні дані до роботи:

_Предметом дослідження є проєкт застосунку підтримки виробничого процесу у галузі масового харчування шляхом обміну даними між користувачем і сервером у реальному часі.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):

Основним завданням стало створення динамічного ресурсу, що забезпечує взаємодію користувачів із системою через модулі заявок, управління контентом і повідомлень

5. Перелік графічного матеріалу:

Дипломна робота містить: 78 листів друкованого тексту, 14 рисунків, 2 таблиці, 2 додатки і налічує 37 джерел використаної літератури.

6. Консультанти розділів роботи:

Розділ	**Прізвище, ініціали та посада консультанта**	**Підпис, дата**	
		завдання видав	**завдання прийняв**
1	Грама М.П.		
2	Грама М.П.		
3	Грама М.П		

7.Дата видачі завдання:__ _07 жовтня 2024 року_

КАЛЕНДАРНИЙ ПЛАН

№	**Назва етапів виконання кваліфікаційної роботи**	Строк виконання етапів роботи	Примітка
1	Ознайомлення з вимогами дипломної роботи	28.10.2024	
2	Вибір і уточнення теми дипломної роботи	28.10.2024	
3	Аналіз наукових джерел	15.11.2024	
4	Написання тексту дипломної роботи	23.12.2024	

Здобувач _____ Воронова М,О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Грама М.П
(підпис) (прізвище та ініціали)
Підпис, дата завдання

Анотація

Дипломна робота за темою: Розроблення веб-порталу з обслуговування клієнтів мережі ресторанів «Bubbly»

Виконала студентка: Марина Олександрівна Воронова

Дипломна робота містить: 83 листи друкованого тексту, 17 рисунків, 2 таблиці, 2 додатки і налічує 31 джерел використаної літератури.

Мета дослідження: спроектувати веб-застосунок підтримки діяльності закладу масового харчування Bubbly.

Об’єктом дослідження є технології комбінованої розробки веб-застосунків. Предметом дослідження є проект застосунку підтримки виробничого процесу у галузі масового харчування шляхом обміну даними між користувачем і сервером у реальному часі.

У рамках даної роботи було розроблено веб-застосунок для автоматизації процесів у сфері масового харчування. Основним завданням стало створення динамічного ресурсу, що забезпечує взаємодію користувачів із системою через модулі заявок, управління контентом і повідомлень.

Веб-застосунок побудовано з використанням сучасних веб-технологій, зокрема HTML5, PHP, JavaScript, а також фреймворка jQuery для реалізації асинхронного обміну даними за допомогою Ajax. Для зберігання і управління даними була розроблена реляційна база даних на основі MySQL, що забезпечує надійність і швидкодію системи.

В процесі роботи реалізовано функціонал реєстрації та авторизації користувачів, управління сесіями, відправки, редагування та видалення заявок, а також оновлення статусів користувачів. Особливу увагу приділено забезпеченню кросбраузерної сумісності та оптимізації роботи системи, що дозволяє застосунку коректно функціонувати на різних платформах і пристроях.

Результатом роботи є повноцінний веб-застосунок, що відповідає поставленим вимогам, демонструє високу ефективність та адаптивність, і може бути використаний для вирішення аналогічних задач у сфері автоматизації бізнес-процесів.

Ключові слова: веб-застосунок, масове харчування, HTML5, PHP, JavaScript, jQuery, Ajax, MySQL, база даних, сесійне управління, реєстрація користувачів, динамічний контент.

Annotation

Thesis Title: Development of a Website for the Bubbly Restaurant Network
Author: Marina Oleksandrivna Voronova

The thesis consists of 83 pages of printed text, 17 figures, 2 tables, 2 appendices, and references 31 sources of literature.

The Aim: To design a web application to support the activities of the mass catering establishment Bubbly.

Object of Study: Technologies for combined web application development. Subject of Study: A project of an application to support the production process in the field of mass catering through real-time data exchange between the user and the server.

This thesis includes the development of a web application aimed at automating processes in the mass catering sector. The primary task was to create a dynamic resource that ensures user interaction with the system through modules for requests, content management, and messaging.

The web application was developed using modern web technologies, including HTML5, PHP, JavaScript, and the jQuery framework for asynchronous data exchange using Ajax. A relational database based on MySQL was designed to ensure the system's reliability and high performance.

The implemented functionality includes user registration and authorization, session management, submission, editing, and deletion of requests, as well as user status updates. Particular attention was paid to ensuring cross-browser compatibility and system optimization, allowing the application to function correctly across various platforms and devices.

The result of this work is a fully functional web application that meets the specified requirements, demonstrates high efficiency and adaptability, and can be used to address similar tasks in the field of business process automation.

Keywords: web application, mass catering, HTML5, PHP, JavaScript, jQuery, Ajax, MySQL, database, session management, user registration, dynamic content.

ЗМІСТ

-

ВСТУП

Актуальність теми роботи. Завдяки розвитку цифрових технологій і широкому доступу до Інтернету, веб-застосунки стають дедалі популярнішими у різних галузях, включаючи сферу обслуговування. Вони пропонують зручні та гнучкі рішення для оптимізації бізнес-процесів, навчання персоналу, обміну інформацією та покращення взаємодії з клієнтами. Створення веб-застосунку для мережі ресторанів дозволяє організувати доступ до інтерактивних навчальних матеріалів для співробітників, автоматизувати процеси замовлення, контролювати складські залишки та спрощувати управління закладом.

Такий веб-застосунок може стати ефективним інструментом для підвищення професійного рівня персоналу. Він забезпечує можливість інтеграції онлайн-курсів, відеоуроків з приготування страв, тестів для перевірки знань та інструкцій з обслуговування клієнтів. Це сприяє швидкому освоєнню нових навичок, підвищенню кваліфікації працівників та формуванню уніфікованих стандартів роботи.

Окрім навчальної функції, веб-застосунок може підтримувати взаємодію між працівниками різних закладів мережі. Наприклад, це може бути обмін досвідом щодо популярних страв, спільне вирішення логістичних задач або обговорення сучасних технологій обслуговування. Це сприятиме створенню професійної спільноти, яка підтримає високі стандарти якості у всій мережі ресторанів.

Сучасна розробка веб-ресурсів здійснюється з використанням таких технологій, як HTML5, Ajax, PHP і Java. Ці інструменти дозволяють реалізувати динамічний обмін даними з базою даних, наприклад MySQL, що є важливим для автоматизації процесів обліку замовлень, управління меню та аналізу відгуків клієнтів. Особливо актуальним є використання Ajax, який забезпечує оновлення даних без необхідності перезавантаження сторінки, що значно покращує взаємодію користувачів із системою.

Перевага Ajax полягає у простоті програмування та наявності готових фреймворків, які полегшують роботу розробників. Це дозволяє створювати веб-застосунки, що поєднують функціональність, зручність і доступність.

У межах курсового проекту було розроблено інформаційну систему для мережі ресторанів Bubbly, яка включає модулі замовлень, управління персоналом, оновлення меню та аналіз відгуків клієнтів. Основна мета цієї системи – автоматизація ключових бізнес-процесів і підвищення ефективності роботи мережі, забезпечуючи відповідність сучасним вимогам цифрових рішень у сфері громадського харчування.

Мета дослідження: спроектувати веб-застосунок підтримки діяльності закладу масового харчування Bubbly.

Для досягнення цієї мети потрібно вирішити наступні задачі:

- провести опис задачі курсового проектування;
- виконати аналіз технологій та існуючих рішень;
- описати інформаційне забезпечення проектованого веб-застосунку;
- виконати розробку алгоритмів рішення функціональної задачі;
- розробити програмне забезпечення веб-застосунку;
- розробити інструкцію користувача системи.

Об’єктом дослідження є технології комбінованої розробки веб-застосунків.

Предметом дослідження є проект застосунку підтримки виробничого процесу у галузі масового харчування шляхом обміну даними між користувачем і сервером у реальному часі.

У роботі використовувались такі **методи дослідження**: пошук і аналіз наявної методичної та наукової літератури, порівняння, проектування, теоретичне моделювання, аналіз документації та результатів попередніх досліджень з проблеми, що вивчалась.

Джерельна база дослідження включала науково-методичну літературу, методичні посібники, наукові статті, періодичні видання та роботи програмістів і розробників веб-додатків.

Робота має **теоретичну і практичну цінність** через наявність зібраного та систематизованого теоретичного матеріалу по дослідженню. Вона також включає розроблені власні рішення та код, що поглиблюють розуміння та розвиток області дослідження в порівнянні з попередніми роботами в цій галузі.

Структура роботи. Робота складається з 81 листа друкованого тексту, 17 рисунків, 2 таблиць, 2 додатків і налічує 37 джерел використаної літератури.

РОЗДІЛ 1. СИСТЕМНИЙ АНАЛІЗ РОЗРОБКИ САЙТУ ДЛЯ МЕРЕЖІ РЕСТОРАНІВ «BUBBLY»

1.1 Опис задачі проектування інформаційної системи для мережі ресторанів

Сучасні інформаційні технології надають широкі можливості для створення ефективних систем обміну даними в Інтернеті, що спрощують роботу розробників і підвищують зручність для користувачів. Інформаційна система для мережі ресторанів базується на використанні передових веб-технологій, які дозволяють ефективно обробляти великі обсяги даних через інтуїтивно зрозумілий інтерфейс.

Для аналізу сучасних підходів і визначення актуальності розробки було проведено дослідження інформаційних джерел за допомогою таких сервісів, як Google.com і Wikipedia.org. Вивчення доступної інформації підтвердило, що технологія Ajax (Asynchronous JavaScript and XML) є актуальною і широко використовується у розробці сучасних веб-застосунків, включаючи ті, що стосуються сфери обслуговування [3].

Ajax дозволяє здійснювати асинхронний обмін даними між клієнтом і сервером, що значно підвищує зручність використання веб-застосунків. У порівнянні з класичною моделлю, де кожна взаємодія користувача призводить до повного перезавантаження сторінки, Ajax забезпечує оновлення лише змінених елементів, зберігаючи решту сторінки без змін. Це особливо важливо для інформаційних систем ресторанів, де швидкість і ефективність обробки даних відіграють ключову роль.

Використання Ajax передбачає застосування суміжних технологій, таких як DHTML для динамічної зміни змісту сторінки, XMLHttpRequest для звернення до сервера, JSON для передачі даних, а також динамічне завантаження JavaScript-коду. Це дозволяє створювати інтерактивні веб-інтерфейси, що забезпечують високу якість взаємодії між користувачем і системою.

Ajax став популярним завдяки впровадженню цієї технології у проектах Google, таких як Google Maps і Google Suggest. Ці приклади демонструють переваги асинхронного підходу, зокрема швидкість і зручність у роботі з великими обсягами даних [25, с. 122].

Основні переваги та недоліки технології Ajax у проектуванні системи

Переваги:

1. Динамічне оновлення контенту без перезавантаження сторінки.
2. Підвищення швидкості обробки запитів та зручності користувача.
3. Гнучкість у створенні інтерактивних веб-застосунків.

Недоліки:

1. Відсутність повної інтеграції зі стандартними інструментами браузера, такими як кнопка «Назад».
2. Проблеми з використанням закладок, які потребують впровадження спеціальних механізмів (перманентних посилань).
3. Ускладнена індексація сторінок пошуковими системами через динамічний контент.
4. Складність інтеграції Ajax у проєкт, особливо за умов великої кількості запитів.
5. Підвищені вимоги до сумісності браузерів із JavaScript.
6. Проблеми відстеження відвідуваності, оскільки класичні системи аналітики можуть не враховувати асинхронні запити [25, с. 123].

Для успішного впровадження Ajax у проектування інформаційної системи для мережі ресторанів важливо враховувати ці аспекти та використовувати сучасні підходи для подолання можливих обмежень. Це дозволить створити зручний, швидкий і функціональний веб-застосунок, який відповідатиме потребам ресторанного бізнесу.

1.2 Аналіз технологій та існуючих рішень

У проектуванні веб-ресурсу для мережі ресторанів інтерактивність інтерфейсу відіграє ключову роль. Висока інтерактивність забезпечується використанням сучасних технологій, мультимедійних елементів, шаблонів проектування та оптимізації взаємодії користувача з веб-інтерфейсом. Однією з провідних технологій, що дозволяють досягти цього, є Ajax.

Класичні моделі взаємодії між клієнтом і сервером передбачали відправлення всього вмісту сторінки на сервер для її оновлення, що призводило до значного збільшення використання трафіку. Крім того, користувачеві доводилося чекати повного завантаження сторінки, а в разі помилок – повторно вводити дані у форму. Використання Ajax змінює підхід до обміну даними. Завдяки асинхронним HTTP-запитам дані обробляються у фоновому режимі, і оновлюються лише змінені частини веб-сторінки, що значно зменшує витрати трафіку та час завантаження.

Переваги технології Ajax:

1. Динамічне оновлення контенту: Користувач може залишатись на сторінці, поки необхідні дані передаються та обробляються на сервері, без перезавантаження всієї сторінки.
2. Економія трафіку: Оновлюються лише ті елементи сторінки, які були змінені, що значно знижує обсяг переданих даних.
3. Покращення користувацького досвіду: Інтерактивність інтерфейсу дозволяє виконувати дії швидше, реагуючи на запити в реальному часі. Наприклад, у веб-формах користувач може отримувати миттєві підказки або обмеження щодо введених даних.

Приклади використання Ajax. У системах бронювання чи управління замовленнями, які часто застосовуються в ресторанному бізнесі, Ajax дозволяє швидко перевіряти доступність дат або часу для резервування. Наприклад, якщо користувач намагається забронювати столик, система автоматично перевіряє

доступність вибраного часу та відображає результати без перезавантаження сторінки.

Сучасні веб-застосунки, побудовані із застосуванням Ajax, дозволяють створювати зручні та ефективні інтерфейси. Наприклад:

1. Форми з автоматичними підказками: Скрипти, що відправляють запити на сервер, дозволяють миттєво надавати користувачам актуальну інформацію або коригувати введені дані.
2. Динамічні сторінки: Лише необхідні частини сторінки оновлюються, залишаючи решту статичною. Це полегшує роботу серверів і покращує загальну швидкість взаємодії.

Технологія Ajax сприяє оптимізації серверного навантаження, що особливо актуально для ресторанних мереж із великою кількістю користувачів. Відвідувачі можуть отримувати актуальну інформацію про меню, наявність страв або доступні акції у режимі реального часу. Це не лише підвищує ефективність роботи веб-сайту, але й створює позитивний досвід для клієнтів.

Використання Ajax у проектуванні інформаційних систем для ресторанів дозволяє реалізувати сучасні, зручні та економічно ефективні рішення. Ця технологія забезпечує інтерактивність, швидкість та зменшення трафіку, що робить її незамінною для побудови конкурентоспроможних веб-ресурсів [2].

Прикладами платформ для підтримки навчальних процесів можуть бути наступні:

1. Сайт ««Чотири чебуреки Prosecco Bar»».

Веб-сайт мережі ресторанів «Чотири Чебуреки Prosecco Bar» (https: // center. 4che. bar/) виконує кілька ключових функцій, спрямованих на покращення обслуговування клієнтів та оптимізацію бізнес-процесів:

1. Онлайн-меню: Користувачі можуть ознайомитися з асортиментом страв, включаючи детальні описи та ціни. Це дозволяє клієнтам заздалегідь вибрати бажані позиції.

2. Система замовлень: Сайт надає можливість оформити замовлення онлайн з опціями доставки або самовивозу. Це спрощує процес замовлення та підвищує зручність для клієнтів.

3. Бронювання столиків: Функція резервування столиків дозволяє відвідувачам планувати свій візит заздалегідь, забезпечуючи комфортне перебування в закладі.

4. Інформація про заклад: Розділ «Про нас» містить відомості про концепцію ресторану, використання свіжих сезонних продуктів та інноваційні підходи до приготування страв, що підкреслює унікальність закладу.

5. Контактна інформація: Сайт надає актуальні контактні дані, включаючи адреси, номери телефонів та електронну пошту, що сприяє швидкій комунікації з клієнтами.

6. Франшиза: Розділ, присвячений можливостям франшизи, містить інформацію для потенційних партнерів, зацікавлених у співпраці та розвитку бізнесу під брендом «Чотири Чебуреки Prosecco Bar».

Загалом, функціонал сайту спрямований на забезпечення зручності для клієнтів, підвищення ефективності обслуговування та підтримку розвитку мережі ресторанів (Рис. 1.1). [14]

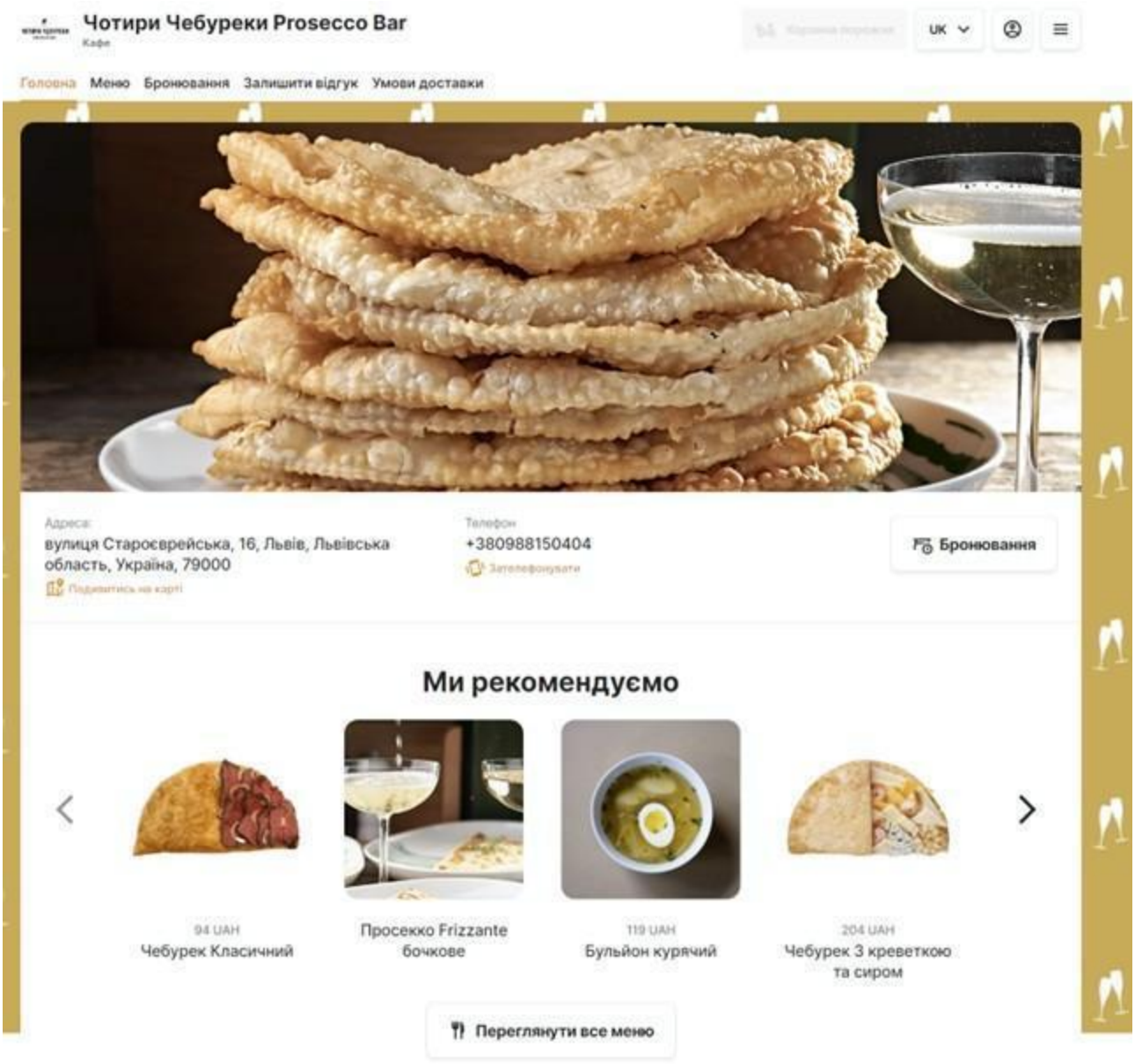

Рисунок 1.1 – Стрічка сайту «Чотири Чебуреки Prosecco Bar»

Послідовність оформлення замовлення на сайті ресторану, наприклад, «Чотири Чебуреки Prosecco Bar», може виглядати так:

1. Перехід на сайт. Користувач відвідує сайт через браузер за посиланням (наприклад, https: // center. 4che. bar).

2. Перегляд меню. На головній сторінці або в розділі меню відображаються доступні страви та напої. Користувач може ознайомитися з асортиментом, переглянути фотографії страв, їх описи, склад та ціни.

3. Вибір страв. Користувач додає обрані позиції в кошик, натискаючи кнопку «Додати» або аналогічну біля кожної страви. Кількість порцій можна змінювати у кошику.

4. Перехід до кошика. Після вибору страв користувач переходить до кошика. У кошику відображається перелік замовлених позицій із зазначенням кількості, ціни кожної страви та загальної вартості замовлення.

5. Вибір способу отримання. Користувач обирає з-поміж доступних варіантів:

– Самовивіз: вказується адреса ресторану, де клієнт забере замовлення.
– Доставка: клієнт вводить свою адресу для доставки [14].

6. Заповнення даних. Користувач заповнює форму з особистими даними:
Ім’я

– Контактний номер телефону.
– Адреса доставки (якщо обрано доставку).
– Додаткові побажання (за потреби).

7. Вибір способу оплати

– Користувач обирає спосіб оплати:
– Онлайн-оплата (банківською картою через платіжну систему).
– Оплата готівкою під час отримання замовлення.

8. Підтвердження замовлення. Користувач перевіряє всі дані, натискає кнопку «Підтвердити замовлення» або аналогічну. Після цього система формує замовлення.

9. Отримання підтвердження. Після оформлення замовлення на екрані з’являється підтвердження із зазначенням:

– Номера замовлення.
– Орієнтовного часу готовності або доставки.
– Контактів для уточнення деталей.

Користувач може отримати підтвердження на електронну пошту або SMS.

10. Обробка замовлення. Замовлення передається в ресторан для приготування. У разі доставки кур’єр отримує інформацію про адресу клієнта.

11. Отримання замовлення. Клієнт забирає замовлення самостійно або приймає доставку. У разі онлайн-оплати чек можна отримати в електронному вигляді.

Ця послідовність забезпечує простоту та зручність для користувача, а також ефективність для персоналу ресторану [14].

2. Веб-сайт ресторану «Хуторець на Дніпрі» (https: // khutoretsnadnipri. choiceqr.com/)[4]

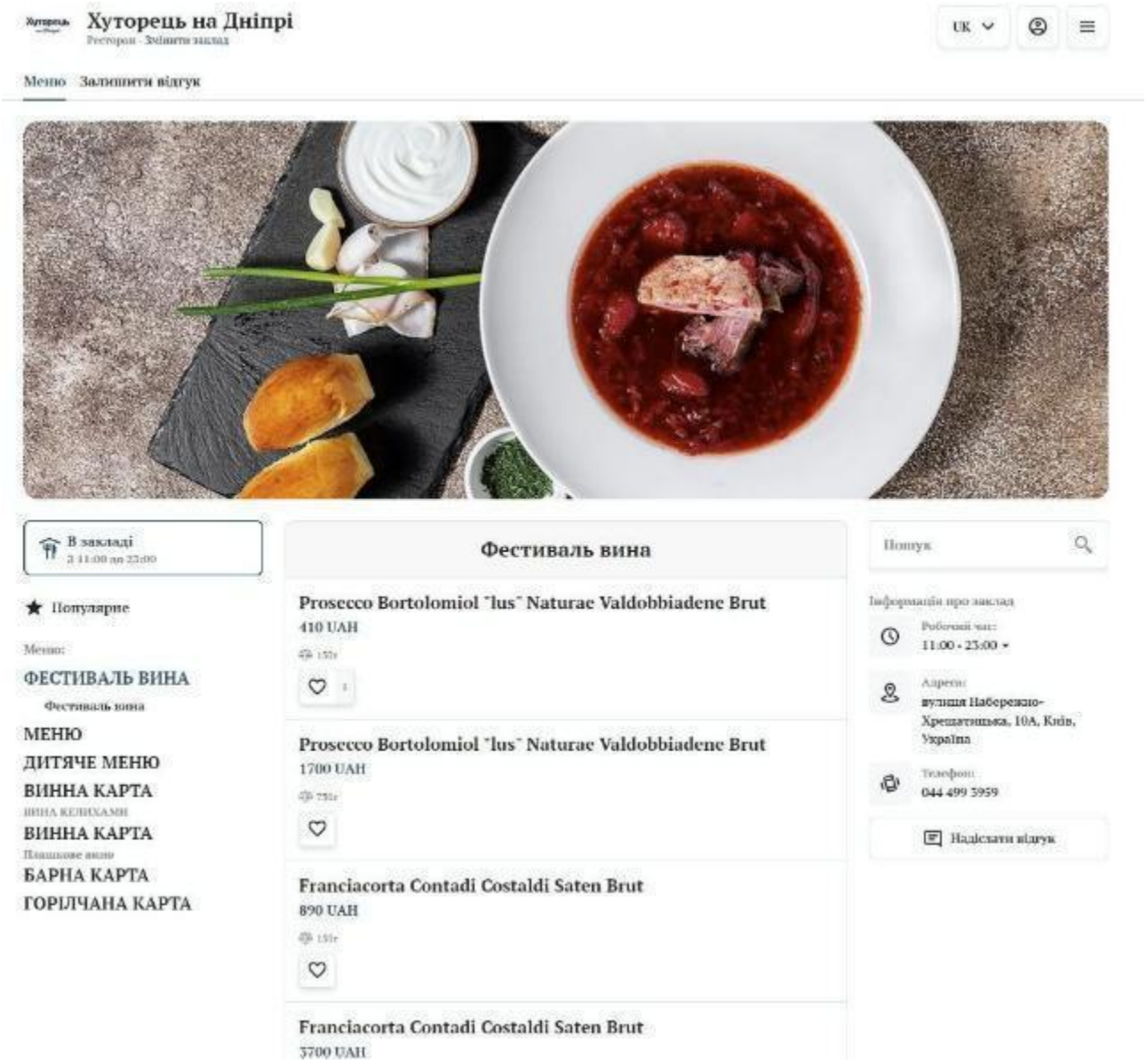

Рисунок 1.2 – Веб-сайт «Хуторець на Дніпрі»

Веб-сайт ресторану «Хуторець на Дніпрі» надає користувачам можливість ознайомитися з меню закладу. На головній сторінці представлено різноманітні розділи меню, такі як:

1. Карта сала: різноманітні види сала, включаючи «Генеральське», сало в гострій аджиці, з зеленню, в червоному перці та паприці, підчеревина в'ялена та копчена.
2. Закуски: оселедець під шубою із запечених овочів, холодець із півня з буряковим хроном, ікра щуки з грінками, тартар з яловичини з маринованими білими грибами та картопляним гратеном, фермерські сири з Прикарпаття та інші.
3. Салати: олів'є з копченою індичкою та яловичим язиком, овочевий салат з домашньою бринзою, салат із запечених овочів з телятиною.
4. Перші страви: борщ червоний з телятиною, зелений борщ з домашньою куркою, уха з окунем та судаком, білий борщ з копченим ребром, червоний борщ з печеним ребром мангалиці.
5. Вареники: з картоплею в грибному соусі, з м'ясом, з окунем, ікрою форелі та мусом з пастернаку, з м'ясом зайця в карпатських травах, з капустою та цибулею-порей, з вишнею та солодким сметанковим соусом.
6. Гарячі страви: смажені карасі, кров'янка з чебрецем, деруни з закарпатськими білими грибами, домашні голубці із копченим м'ясом в листях молодої капусти, котлети по-київськи з овочевим салатом, вогняні реберця.
7. Риба та м'ясо на відкритому вогні: український рібай, закарпатська форель, стерлядь, стейк з червоної форелі з мідіями та морською спаржею, соковитий курячий шашлик, шашлик зі свинного ошийка, каре телятини, медальйони з яловичини, перепілка з морквяним пюре та фундуком, каре косулі з пюре з пастернаку.
8. Овочі: кукурудза з витриманим сиром, картопля з шавлією, картопля з підчеревиною на мангалі, запечені овочі (помідори, перець, цибуля, баклажан, кабачок).

9. Солодке: фірмовий «Київський торт», медовик з морською сіллю, маковик з киселем, фірмове «Хуторець», морозиво ванільне та шоколадне, сорбети (обліпиха, вишня, ревень) [4].

Кожна страва супроводжується описом та вказівкою ваги порції. Зверніть увагу, що сайт не надає можливості онлайн-замовлення або доставки; його основна функція – ознайомлення гостей з меню ресторану.

Послідовність бронювання столика на прикладі сайту ресторану:

1. Перехід на сторінку бронювання. Користувач переходить на сайт ресторану та відкриває розділ, присвячений бронюванню столиків. На деяких сайтах це може бути кнопка «Забронювати столик» або окремий підрозділ меню.
2. Вибір дати та часу. На сторінці бронювання відображається форма, де користувач обирає:

– Дату відвідування.

Бажаний час (із доступного діапазону годин роботи ресторану).

3. Вибір кількості гостей. Користувач зазначає, на скільки осіб потрібен столик. Це може вплинути на доступність столиків певного розміру.

4. Заповнення контактної інформації. Форма запиту може включати наступні поля:

– Ім’я.
– Номер телефону.
– Електронна пошта (опційно).
– Додаткові побажання (наприклад, розташування біля вікна або спеціальні потреби).

5. Підтвердження бронювання. Користувач натискає кнопку «Забронювати». Система обробляє запит і перевіряє доступність столиків на вказану дату та час.

6. Отримання підтвердження. Після успішної перевірки бронювання користувач отримує підтвердження:

– На екрані (наприклад, номер бронювання або повідомлення «Ваш столик заброньовано»).
– У SMS або на електронну пошту (якщо вказано).

7. Зворотний зв’язок. Адміністратори ресторану можуть зв’язатися з користувачем для уточнення деталей або підтвердження бронювання за вказаним номером телефону.

8. Прибуття до ресторану. У визначений час клієнт приходить до ресторану, повідомляє адміністратору своє ім’я або номер бронювання, після чого його проводять до зарезервованого столика.

Додаткові опції:

1. На деяких сайтах є функція вибору конкретного столика за схемою залу.
2. У разі недоступності вказаного часу система може запропонувати альтернативні варіанти [4].

Аналіз функціоналу сайтів «Чотири Чебуреки Prosecco Bar» та «Хуторець на Дніпрі» показав їх сильні сторони, які можуть бути інтегровані в розробку сайту для мережі ресторанів Bubbly. Обидва приклади демонструють сучасний підхід до презентації інформації та забезпечення інтерактивної взаємодії з клієнтами.

Основні переваги, релевантні для нашого проєкту:

1. Інтерактивне онлайн-меню:

– Сайт «Хуторець на Дніпрі» пропонує детальне меню з описами страв, інгредієнтів та ваги порцій, що сприяє підвищенню інтересу клієнтів.
– «Чотири Чебуреки Prosecco Bar» доповнює це інтерактивним підходом із можливістю додавання страв у кошик та формування замовлення.
– Для Bubbly це стане основою, оскільки клієнти матимуть змогу переглядати меню, отримувати інформацію про страви та напої, а також швидко робити вибір.

2. Система замовлень і бронювання:

– У «Чотири Чебуреки Prosecco Bar» реалізовано можливість оформлення замовлення онлайн із вибором доставки або самовивозу, що зручно для користувачів.
– У «Хуторці» акцент зроблено на ознайомленні з меню, проте можливість додати модуль бронювання дозволила б спростити процес планування відвідування ресторану.
– Для мережі Bubbly варто інтегрувати функцію бронювання столиків із вибором часу, кількості гостей та додатковими побажаннями, адже це підвищить рівень сервісу.

3. Простий і зрозумілий інтерфейс:

– Обидва сайти мають інтуїтивно зрозумілу навігацію, що важливо для клієнтів. Легкість пошуку інформації сприяє позитивному досвіду використання.
– Bubbly може успадкувати цей підхід, додаючи сучасний дизайн, що відповідає бренду, та швидкодіючу систему відгуку.

4. Динамічність та інтерактивність:

– Використання технології Ajax, як на зразках, дозволяє створювати швидкі та інтерактивні сторінки, що важливо для користувачів, які цінують час.
– Для Bubbly технології Ajax забезпечать плавний обмін даними з сервером без перезавантаження сторінки, наприклад, при оновленні кошика чи перегляді доступних столиків для бронювання.

Ці приклади є наочними взірцями, як сучасні веб-рішення можуть ефективно підвищувати якість обслуговування клієнтів у ресторанній сфері. Функції інтерактивного меню, онлайн-замовлення та бронювання столиків безпосередньо впливають на задоволення потреб користувачів і сприяють автоматизації бізнес-процесів.

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЕКТУВАННЯ САЙТУ ДЛЯ МЕРЕЖІ РЕСТОРАНІВ «BUBBLY»

2.1. Загальні положення

Об’єктом проектування є інформаційна система для мережі ресторанів, яка забезпечує обмін даними між користувачем і сервером у мережі Інтернет. Її функціональність орієнтована на підтримку ключових бізнес-процесів, таких як управління замовленнями, бронювання столиків, оновлення меню та аналіз відгуків клієнтів. Використання технології Ajax у розробці дозволяє забезпечити асинхронний обмін даними, що значно покращує зручність користувачів і підвищує ефективність роботи системи.

Проектування сайту для мережі ресторанів Bubbly базується на використанні сучасних технологій, які сприяють створенню функціональних, інтерактивних і простих у використанні веб-ресурсів. Технологія Ajax широко використовується у провідних компаніях, таких як Google і Facebook, що підтверджує її актуальність і ефективність. Її застосування дозволяє оновлювати лише змінені частини сторінки без повного перезавантаження, що зменшує затримки, економить трафік і покращує взаємодію з клієнтами.

Система для Bubbly включає модулі для обробки замовлень, управління персоналом, новин та інформування клієнтів. Ключова перевага такого підходу – інтеграція різних блоків, які синхронно взаємодіють між собою, забезпечуючи стабільну і зручну роботу системи. Це робить технологію Ajax та інші сучасні інструменти надзвичайно актуальними для реалізації поставлених задач.

2.2. Призначення і цілі створення системи

Метою створення інформаційної системи для мережі ресторанів «Bubbly» є автоматизація бізнес-процесів, підвищення ефективності взаємодії з клієнтами та забезпечення оперативного управління ресторанною мережею. Система має забезпечити:

– інтерактивний доступ до меню закладів;

- можливість оформлення онлайн-замовлень;
- резервування столиків;
- управління замовленнями та їх обробку;
- збір і аналіз відгуків клієнтів;
- централізоване оновлення інформації про страви, акції та новини.

Основною метою розробки є створення зручного та сучасного інструменту для ефективного управління мережею ресторанів, який підвищить рівень задоволеності клієнтів та оптимізує операційну діяльність персоналу.

2.3. Характеристика об’єкта автоматизації

Об’єктом автоматизації є мережева структура ресторанів «Bubbly», яка включає кілька закладів з централізованим управлінням. Основні характеристики об’єкта:

Кількість закладів: від 3 до 10 ресторанів;

- кількість користувачів системи: адміністративний персонал, кухарі, офіціанти та клієнти;
- основні бізнес-процеси: прийом і обробка замовлень, бронювання столиків, управління складськими залишками, формування звітності;
- інфраструктура: ресторани оснащені комп’ютерами з доступом до Інтернету та POS-терміналами.

Об’єктом автоматизації є мережева структура ресторанів «Bubbly», яка складається з кількох закладів, що функціонують під єдиним брендом і мають спільну систему управління. Кожен заклад мережі виконує повний спектр операцій, пов’язаних із обслуговуванням клієнтів, включаючи прийом замовлень, бронювання столиків, управління меню, роботу з відгуками відвідувачів, а також координацію роботи персоналу.

Мережа ресторанів характеризується широким асортиментом страв і напоїв, а також високим рівнем сервісу, що вимагає швидкої та точної обробки інформації, взаємодії між співробітниками та клієнтами і підтримки актуальності даних у режимі реального часу. Заклади мережі оснащені сучасним

обладнанням, таким як POS-термінали, планшети для прийому замовлень, а також мають стабільний доступ до Інтернету, що є основою для впровадження інформаційної системи.

Особливістю об’єкта є необхідність забезпечення інтеграції між усіма локаціями мережі, включаючи центральний офіс, який відповідає за стратегічне управління, розробку маркетингових кампаній і загальну координацію. Ресторани також мають взаємодіяти між собою, що дозволить ефективно керувати запасами продуктів, розподіляти ресурси та забезпечувати єдині стандарти обслуговування.

Таким чином, автоматизація об’єкта спрямована на інтеграцію ключових бізнес-процесів, підвищення продуктивності персоналу, оптимізацію взаємодії з клієнтами та забезпечення конкурентоспроможності мережі ресторанів «Bubbly» у сучасному цифровому середовищі.

2.4. Вимоги до системи

Розробка інформаційної системи для мережі ресторанів «Bubbly» вимагає дотримання певних функціональних, технічних та експлуатаційних характеристик, що забезпечать її ефективність, надійність та відповідність сучасним стандартам. Нижче представлено розширений опис вимог до системи.

Функціональні вимоги.

1. Система повинна забезпечувати весь спектр функцій, необхідних для управління ресторанною мережею, а саме:

– Інтерактивний перегляд меню. Клієнти повинні мати доступ до актуального меню ресторану з детальними описами страв, фотографіями, інформацією про інгредієнти, алергени, вагу та ціни. Меню має бути адаптоване до різних мовних налаштувань для зручності клієнтів.
– Оформлення онлайн-замовлень. Веб-сайт повинен дозволяти користувачам вибирати страви, додавати їх до кошика, налаштовувати кількість порцій та оформляти замовлення з вибором способу доставки (самовивіз або доставка до клієнта).

– Бронювання столиків. Система має включати модуль бронювання столиків із зазначенням дати, часу, кількості гостей та спеціальних побажань. Повідомлення про підтвердження бронювання повинно надходити клієнту в реальному часі.
– Управління складськими залишками. Система повинна автоматично обробляти дані про використання продуктів, оновлювати інформацію про запаси на складі, а також формувати замовлення на постачання.
– Збір, обробка і аналіз відгуків клієнтів. Важливо передбачити механізм збору зворотного зв'язку, що дозволить клієнтам залишати відгуки, оцінки та коментарі. Аналіз цих даних має сприяти покращенню якості обслуговування.
– Автоматизація звітності. Система повинна генерувати звіти про продажі, відвідуваність, популярність страв, витрати та інші ключові показники, які є важливими для керівництва ресторанної мережі.

2. Технічні вимоги. Система повинна відповідати сучасним стандартам розробки програмного забезпечення та бути технічно досконалою:

– Використання сучасних технологій. Для реалізації функціоналу необхідно застосовувати такі технології, як HTML5, CSS3, PHP, MySQL, JavaScript з використанням бібліотек Ajax і jQuery. Це дозволить створити динамічний і функціональний веб-ресурс.
– Сумісність із основними веб-браузерами. Сайт має коректно працювати в сучасних браузерах, таких як Google Chrome, Mozilla Firefox, Microsoft Edge, Safari тощо, забезпечуючи однаковий досвід для користувачів.
– Інтеграція з мобільними пристроями. Система повинна мати адаптивний дизайн, що дозволить комфортно працювати з нею на різних пристроях, включаючи смартфони, планшети та комп'ютери.
– Швидке завантаження сторінок. Час завантаження сторінки не повинен перевищувати 2 секунд навіть за умов високого навантаження. Це забезпечить зручність для користувачів і сприятиме їхньому задоволенню.

– Захист персональних даних. Система повинна відповідати вимогам безпеки, зокрема забезпечувати шифрування даних, захист від несанкціонованого доступу та дотримання законодавства про конфіденційність.

3. Експлуатаційні вимоги.Для забезпечення стабільної роботи система має відповідати таким експлуатаційним параметрам:

– Можливість одночасної роботи великої кількості користувачів. Система повинна підтримувати до 200 одночасних з’єднань без втрати продуктивності, що є критичним для пікових періодів у ресторанах.
– Резервування даних. Дані повинні зберігатися на сервері з можливістю автоматичного резервного копіювання для запобігання їх втраті у разі технічних збоїв.
– Цілодобова доступність. Система повинна працювати у режимі 24 / 7, забезпечуючи безперебійний доступ до функцій як для персоналу, так і для клієнтів.

Ці вимоги є основою для створення ефективної інформаційної системи, яка відповідатиме потребам мережі ресторанів «Bubbly», сприятиме оптимізації роботи закладів і покращенню якості обслуговування клієнтів.

2.5. Склад і зміст робіт по створенню системи

Необхідність визначення складу і змісту робіт по створенню системи полягає в тому, щоб чітко окреслити всі етапи процесу розробки, забезпечити послідовність виконання завдань та узгодженість між учасниками проєкту. Це дозволяє ефективно розподілити ресурси, уникнути помилок під час реалізації, врахувати специфіку бізнес-процесів і забезпечити функціональність системи відповідно до вимог. Такий підхід сприяє досягненню поставлених цілей, оптимізації витрат часу і ресурсів, а також гарантує якісне впровадження та подальшу підтримку системи.

1. Аналіз вимог:

– Збір інформації про бізнес-процеси мережі ресторанів.
– Аналіз конкурентних рішень.

2. Проектування системи:

– Розробка архітектури системи.
– Визначення структури бази даних.
– Створення прототипів інтерфейсу.

3. Розробка:

– Програмування функціональних модулів.
– Інтеграція Ajax для асинхронного обміну даними.
– Тестування системи.

4. Впровадження:

– Інсталяція системи в ресторанах мережі.
– Навчання персоналу.

5. Підтримка та розвиток:

– Оновлення системи.
– Усунення помилок.

2.6. Порядок контролю і приймання системи

Необхідність визначення порядку контролю і приймання системи полягає в забезпеченні відповідності розробленого програмного забезпечення заданим вимогам і критеріям якості. Це дозволяє виявити та усунути можливі недоліки, перевірити функціональність, продуктивність і надійність системи перед її впровадженням. Чітко встановлений порядок контролю та приймання гарантує, що система відповідає очікуванням користувачів, є безпечною, стабільною і готовою до ефективного використання в реальних умовах.

1. Проведення модульного тестування кожного функціонального блоку.
2. Інтеграційне тестування для перевірки взаємодії модулів.

3. Стрес-тестування для оцінки продуктивності системи під навантаженням.

4. Приймальне тестування за участю замовника.

5. Оформлення акта прийому-передачі системи після успішного завершення тестування.

2.7. Вимоги до складу і змісту робіт із підготовки до введення системи в дію

Вимоги до складу і змісту робіт із підготовки до введення системи в дію включають забезпечення налаштування програмного забезпечення, інтеграції системи з існуючою інфраструктурою мережі ресторанів, проведення тестування в реальних умовах експлуатації, навчання персоналу роботі з новою системою та підготовку супровідної документації. Це забезпечує плавний перехід до використання системи, мінімізує ризики збоїв і забезпечує ефективну експлуатацію з перших днів впровадження.

1. Підготовка інфраструктури:

– Перевірка обладнання та підключення до мережі Інтернет.
– Налаштування серверного середовища.

2. Навчання персоналу:

– Проведення тренінгів для адміністративного та обслуговуючого персоналу.
– Надання інструкцій користувачам.

3. Проведення пробної експлуатації:

– Тестовий запуск системи в обмеженому режимі.
– Виявлення та усунення недоліків.

2.8. Вимоги до документації

Вимоги до документації включають створення технічної документації, яка містить опис архітектури системи, інструкції з її встановлення та налаштування,

а також керівництво користувача для роботи з функціоналом. Звітна документація повинна включати акти приймання системи та протоколи тестування, які підтверджують її відповідність встановленим вимогам і готовність до впровадження.

1. Технічна документація:

– Опис архітектури системи.
– Інструкції з встановлення та налаштування.
– Керівництво користувача.

2. Звітна документація:

– Акт приймання системи.
– Протоколи тестування.

2.9. Джерела розробки

Джерела розробки включають аналіз бізнес-процесів мережі ресторанів «Bubbly», вивчення конкурентних рішень у ресторанній сфері, рекомендації щодо сучасних технологій веб-розробки, технічні специфікації доступного обладнання, а також відгуки потенційних клієнтів і співробітників.

1. Аналіз бізнес-процесів мережі ресторанів «Bubbly».
2. Вивчення конкурентних рішень у сфері ресторанного бізнесу.
3. Рекомендації з використання сучасних технологій веб-розробки.
4. Технічні специфікації обладнання, доступного в ресторанах.
5. Відгуки потенційних клієнтів і співробітників мережі.

РОЗДІЛ 3. ОПИС КОМПЛЕКСУ ЗАДАЧ ПРОЕКТУВАННЯ САЙТУ ДЛЯ МЕРЕЖІ РЕСТОРАНІВ «BUBBLY»

3.1. Інформаційне забезпечення проектованої інформаційної системи мережі ресторанів

Розробка веб-застосунку для мережі ресторанів починається з формування чіткої інформаційної моделі, яка визначає основні вимоги до контенту та функціональності. В технічному завданні для проектування системи необхідно деталізувати всі аспекти інформаційного наповнення та завдань, які має вирішувати веб-застосунок. Структура і зміст веб-застосунку повинні максимально відповідати функціональній моделі ресторанного сервісу, забезпечуючи логічну організацію даних і зручність використання для клієнтів.

Ефективність офіційної веб-сторінки оцінюється за двома основними критеріями: змістом та дизайном. Проте ще до створення візуальної складової необхідно визначити, яку саме інформацію слід розмістити на ресурсі. Це передбачає ретельний аналіз, щоб визначити обсяги, типи та формат даних, які будуть представлені на веб-сайті. Для сайту мережі ресторанів важливо передбачити такі елементи, як меню, акційні пропозиції, можливість бронювання столиків і перегляду локацій.

Веб-сайт має виконувати функцію зручного інструменту взаємодії з клієнтами. На головній сторінці повинні бути розміщені посилання на основні розділи: меню, акції, бронювання, відгуки тощо. Усі ці компоненти формують загальну структуру сайту, яка дозволяє ефективно організувати інформацію і уникнути проблеми «інформаційного перевантаження» [11].

Основними причинами розділення інформації на кілька веб-сторінок є спрощення доступу до даних для користувачів та технічні переваги. Завдяки розподілу інформації на логічно структуровані сторінки користувачі можуть швидше знаходити потрібні дані, що демонструє професіоналізм і повагу до клієнтів. Технічна причина розподілу полягає в оптимізації графічного та мультимедійного контенту. Надмірне навантаження головної сторінки великою

кількістю елементів може негативно вплинути на швидкість завантаження та зручність перегляду.

Ще однією перевагою сегментації веб-сайту є спрощення оновлення та підтримки. Добре організована структура дозволяє швидко оновлювати контент без ризику пошкодження інших частин сайту. Для цього важливо ретельно спланувати загальну структуру сайту, з урахуванням зв’язків між розділами та особливостей контенту.

При проектуванні сайту розглядається кілька можливих структур. Наприклад, стандартна модель організації включає головну сторінку з посиланнями на інші розділи. Ці розділи, у свою чергу, можуть мати зворотні посилання на основну сторінку. Такий підхід є найпоширенішим завдяки своїй простоті та ефективності, що дозволяє оптимізувати взаємодію користувачів із сайтом (Рис .31.).

Отже, розподіл інформації між кількома сторінками та ретельне проектування структури веб-застосунку є ключовими факторами успішної реалізації системи для мережі ресторанів «Bubbly». Це забезпечує зручний доступ до основних функцій, підвищує ефективність взаємодії користувачів із ресурсом і сприяє загальному розвитку ресторанного бізнесу [27].

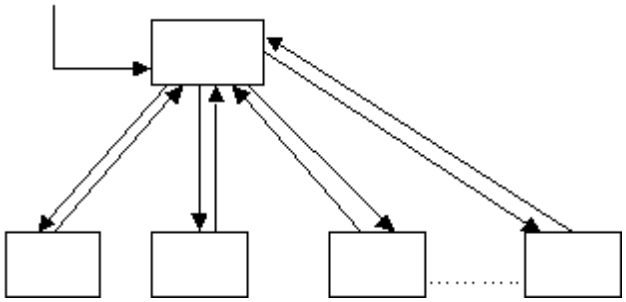

Рисунок 3.1 – Стандартний спосіб організації веб- сайту

У моделі «Каскад» структура посилань у документах організована так, що існує лише один лінійний шлях для переходу між сторінками веб-застосунку. Кожна сторінка включає посилання виключно на наступну в послідовності, створюючи чіткий порядок навігації. На рисунку 3.2 наведено приклад каскадної структури веб-сайту (Рис. 3.2) [27].

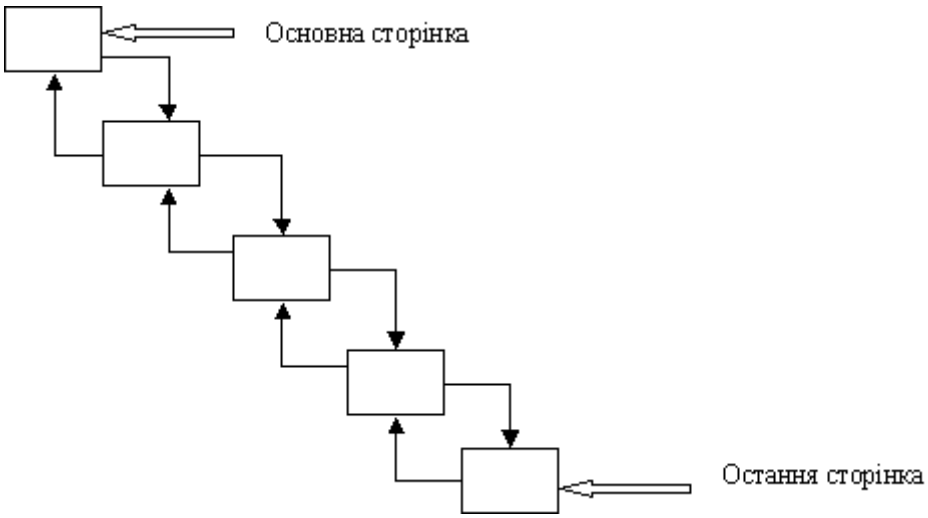

Рисунок 3.2 – Каскадний спосіб організації веб- сайту

При каскадній організації сторінок користувачі можуть переміщуватися між ними лише в одному напрямку – вперед або назад, що забезпечує послідовний і контрольований доступ до інформації.

У моделі «Хмарочос» доступ до певних сторінок можливий лише за умови дотримання правильного маршруту. Це схоже на пересування у великій будівлі, де для досягнення потрібного поверху необхідно пройти через визначені рівні або коридори. Структура сайту за цією моделлю нагадує план хмарочоса, в якому кожна сторінка служить своєрідною кімнатою, доступ до якої вимагає проходження через інші. Схематичне зображення цієї моделі наведено на рисунку 3.3 і відображає її лабіринтоподібну організацію [27].

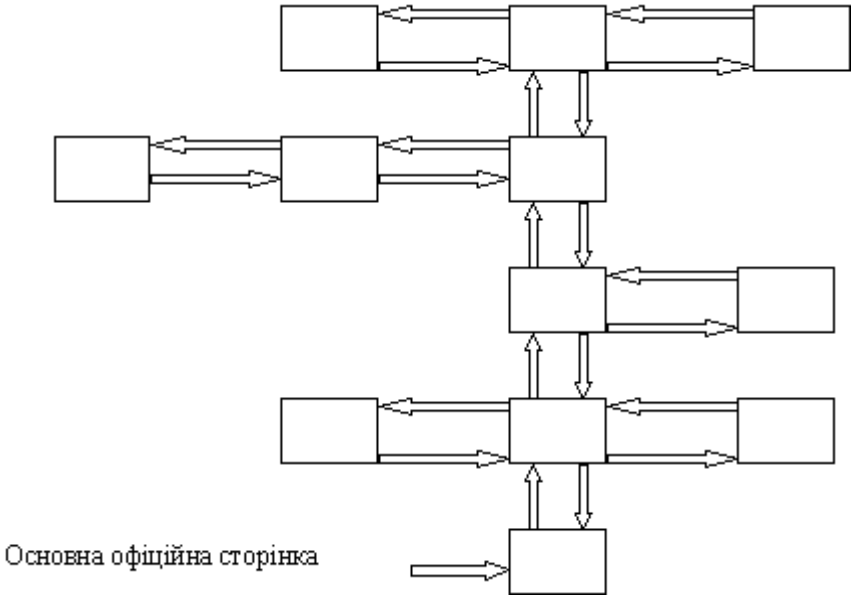

Рисунок 3.3 – Схема організації веб- сайту «Хмарочос»

У моделі «Павутина» кожна сторінка веб-застосунку містить посилання на інші сторінки, що дозволяє користувачам вільно переміщуватися між різними частинами ресурсу. Така організація забезпечує високу гнучкість і відкритість у навігації, створюючи зручні умови для пошуку інформації. Проте, за відсутності контролю над кількістю посилань і їх розташуванням, структура може стати надмірно складною, перетворюючись на навігаційний лабіринт, що ускладнює використання сайту. Модель «Павутина» є популярною для веб-застосунків, де посилання використовуються нечасто, або в тих випадках, коли потрібно створити враження відкритості та необмеженості ресурсу. На рисунку 3.4 подано приклад такої структури, де кожна сторінка містить зв’язки з іншими сторінками, забезпечуючи універсальну взаємодію між різними елементами веб-застосунку [27].

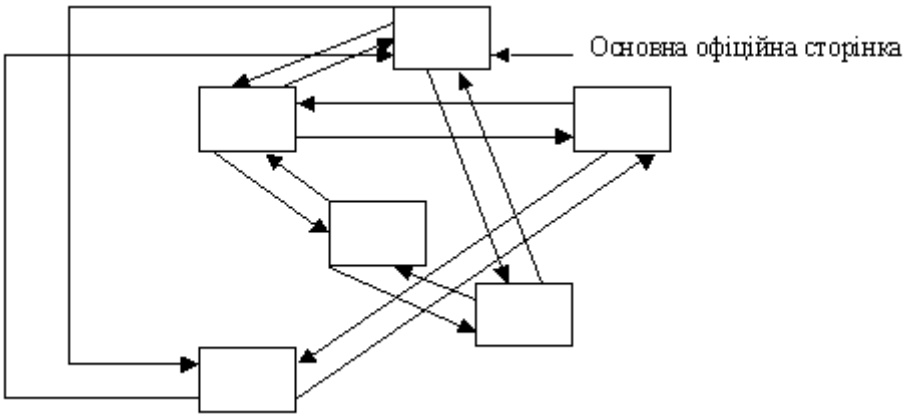

Рисунок 3.4 – Схема організації веб-застосунку «Павутина»

Для організації веб-застосунку мережі ресторанів «Bubbly» доцільно використовувати гібридний підхід, який поєднує елементи стандартного методу навігації та моделі «Павутина». Цей підхід дозволяє користувачам швидко отримати доступ до будь-якого розділу сайту з головної сторінки, зберігаючи при цьому можливість переходів між взаємопов’язаними розділами. Така структура поєднує простоту і зручність стандартного підходу з відкритістю та гнучкістю, яку забезпечує модель «Павутина». Використання моделі «Хмарочос» у випадку цього проекту вважається недоцільним, оскільки вона може створити плутанину в навігації, що негативно вплине на користувацький досвід.

При проектуванні веб-застосунку для мережі ресторанів важливо ретельно обґрунтувати назви та структуру інформаційних розділів, адаптуючи їх до основних завдань системи. Це включає визначення змісту ключових сторінок, таких як меню, система бронювання столиків, інформація про акції та відгуки клієнтів. Уся структура сайту має бути детально описана у технічному завданні, яке затверджується перед початком розробки [19, с. 103].

Окрім того, веб-застосунок має підтримувати багатомовність для забезпечення комфортного доступу до інформації користувачам із різних регіонів. Для цього головна сторінка повинна містити зручний перемикач мов, який може бути реалізований у вигляді графічних зображень, наприклад, у форматі GIF. Такий підхід дозволяє уникнути проблем із відображенням текстових надписів у середовищах із обмеженою підтримкою певних кодувань, наприклад, для користувачів з Європи чи Америки. Цей функціонал забезпечить комфортну роботу із сайтом широкому колу відвідувачів, підвищуючи його доступність і конкурентоспроможність [11, 12, 15].

Основу веб-застосунку для мережі ресторанів «Bubbly» становить користувач, який отримує доступ до ресурсу за вказаною адресою та взаємодіє з ним через інтуїтивний інтерфейс. Користувач має можливість авторизуватися або зареєструватися в системі. У процесі реєстрації створюється персональний профіль, а після успішної авторизації користувач потрапляє на власну профільну сторінку. На цій сторінці передбачено широкий спектр функцій, таких як перегляд меню, оформлення замовлень, бронювання столиків та управління персональними даними. Після завершення роботи користувач має можливість безпечно вийти зі свого профілю.

Ключовою технологією, що забезпечує динамічну взаємодію між користувачем і веб-застосунком, є Ajax. Ця технологія дозволяє асинхронно обмінюватися даними між клієнтом і сервером, завдяки чому інформація оновлюється автоматично без перезавантаження сторінки. Використання Ajax включає такі механізми, як XHR (XMLHttpRequest), динамічне створення фреймів та тегів, а також інструменти DHTML для оновлення сторінок у

реальному часі. Наприклад, у нашому застосунку це забезпечить миттєве відображення змін у статусах замовлень, відгуках клієнтів та наявності столиків.

Завдяки асинхронному обміну даними Ajax значно покращує користувацький досвід, оскільки дозволяє виконувати дії, наприклад, перегляд меню або редагування замовлення, без затримок, викликаних перезавантаженням сторінки. Такий підхід створює зручні та функціональні веб-інтерфейси, особливо на сторінках, де необхідна активна взаємодія з користувачем [12].

Інтеграція Ajax у веб-застосунок «Bubbly» дозволяє автоматично оновлювати дані для користувачів, що суттєво підвищує швидкість і ефективність роботи системи. Наприклад, зміни у статусах бронювань або замовлень будуть відображатися миттєво, без необхідності оновлення всієї сторінки. Такий підхід особливо актуальний для систем, які обслуговують великий потік користувачів у реальному часі.

Структура інтерфейсу нашого веб-застосунку базується на об’єктній моделі документа (DOM). Кожен елемент веб-сторінки представлений як окремий блок, що може динамічно оновлюватися та змінювати свою структуру. Це досягається за допомогою HTML для створення структури сторінки, CSS для стилізації та Ajax для інтерактивності. Використання DOM разом із технологіями Ajax дозволить динамічно оновлювати контент, забезпечуючи користувачам швидкий і безперервний доступ до всіх функцій веб-застосунку. Така реалізація сприятиме підвищенню зручності роботи з ресурсом та його популярності серед клієнтів.

3.2. Схеми інформаційних потоків

Робота веб-застосунку для мережі ресторанів «Bubbly», реалізованого з використанням технології Ajax, передбачає кілька ключових етапів взаємодії між користувачем, браузером і сервером:

1. Користувач відкриває веб-сторінку застосунку та взаємодіє з її елементами, наприклад, обирає страву з меню або оформлює замовлення.

2. У відповідь на дії користувача браузер надсилає асинхронний запит на сервер, використовуючи можливості Ajax.
3. Сервер обробляє запит і повертає лише ту частину даних, яка потребує оновлення, наприклад, статус замовлення чи актуальне меню.

Ця модель роботи дозволяє уникнути повного перезавантаження сторінки, значно покращуючи користувацький досвід завдяки швидкому оновленню інформації в реальному часі.

Для ілюстрації функціонування веб-застосунку з використанням Ajax створено схему, що наочно демонструє процеси взаємодії між клієнтом і сервером. На схемі зображено, як відбуваються запити та отримання відповідей у фоновому режимі, без втручання в основний контент сторінки. Цей підхід забезпечує плавну роботу системи та підвищує її ефективність.

Рисунок 3.5, наведений нижче, демонструє принцип роботи веб-застосунку, зосереджуючи увагу на механізмі передачі та обробки даних через Ajax-запити. Така архітектура дозволяє реалізувати зручний і швидкий сервіс, що відповідає сучасним вимогам галузі масового харчування.

Розробка веб-застосунку для мережі ресторанів «Bubbly» починається зі створення інформаційної моделі, яка забезпечить ефективну взаємодію користувача з веб-ресурсом. Основою роботи системи є відправлення запитів користувачем, обробка цих запитів сервером та оновлення необхідних частин веб-сторінки без повного перезавантаження. Такий підхід реалізується за допомогою технології Ajax у поєднанні з сучасними інструментами, такими як HTML, CSS, PHP, MySQL та бібліотека JavaScript jQuery.

Запити користувача спершу обробляються за допомогою JavaScript, який передає їх методом HTTP на сервер. Сервер повертає лише змінені дані, які оновлюються у моделі DOM. Це дозволяє користувачу взаємодіяти із системою без затримок, з мінімальним навантаженням на браузер [9,13].

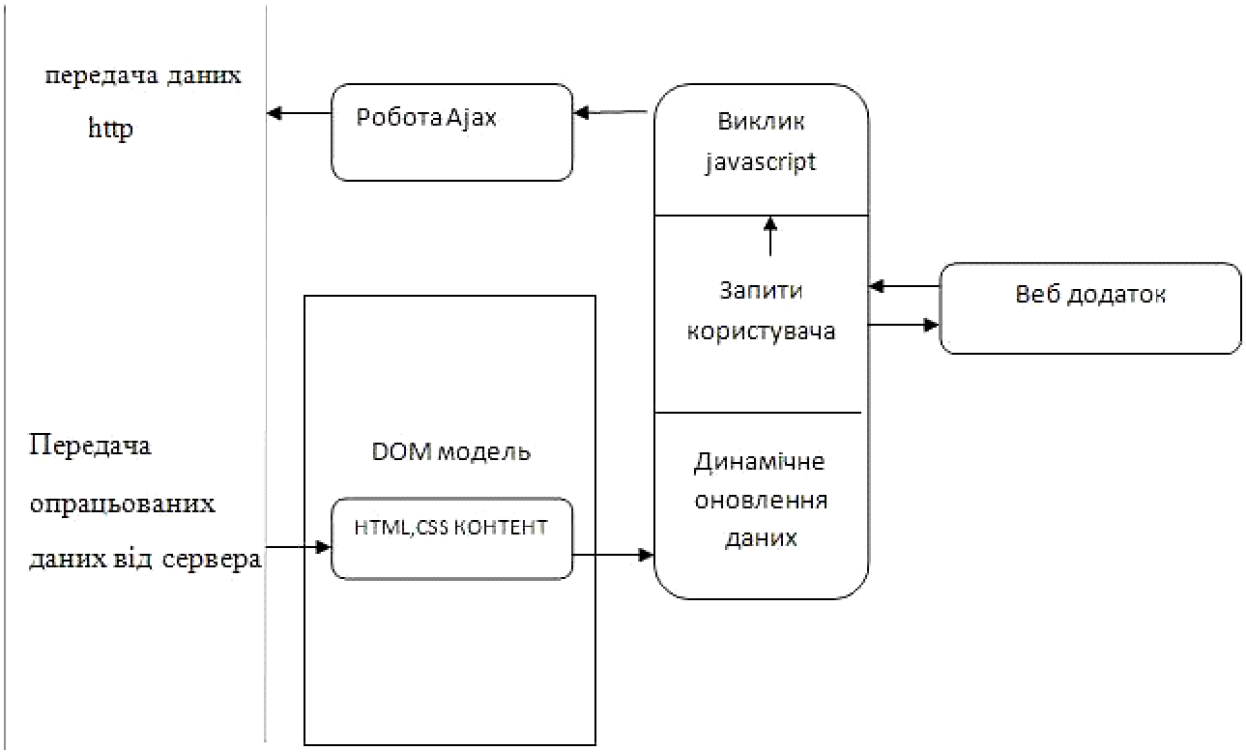

Рисунок 3.5 – Робота системи з використанням Ajax

Для управління даними користувачів та заявками буде використовуватись реляційна база даних MySQL. Структура бази включатиме дві основні таблиці: «user» та «message». Таблиця «user» міститиме дані про користувачів, включаючи їх ідентифікатори, статуси та аутентифікаційну інформацію. Таблиця «message» відповідатиме за збереження заявок із зазначенням відправника, отримувача, тексту заявки та її статусу. Між цими таблицями буде реалізований реляційний зв’язок, що дозволить забезпечити ефективний обмін даними.

Верстка веб-сторінок здійснюватиметься за допомогою HTML, який створює структуру сторінок, та CSS, що відповідає за їх візуальне оформлення. HTML-документи включатимуть три основні частини: декларацію типу документа, заголовок із метаінформацією та тіло сторінки, де розміщуватиметься основний контент. Блокова модель верстки забезпечить динамічність, інтерактивність та можливість інтеграції мультимедійного контенту [11, с. 15].

PHP використовуватиметься для створення динамічних компонентів, які включатимуть реєстрацію, авторизацію, підтримку сесій користувача, вихід із

системи, а також управління заявками. Усі ці функції будуть реалізовані як окремі модулі, інтегровані у загальну структуру веб-застосунку.

Ajax стане ключовою технологією для динамічної взаємодії користувача із системою. За допомогою бібліотеки jQuery будуть реалізовані:

– автоматичне оновлення індикаторів статусу користувачів;
– відправка, редагування та видалення заявок у режимі реального часу;
– підтримка оновлення документів без перезавантаження сторінки.

Функціонал веб-застосунку буде побудований таким чином, щоб користувачі могли легко працювати із системою навіть за одночасної обробки численних запитів. Індикатори документів автоматично оновлюватимуться завдяки проміжним запитам до сервера, показуючи актуальний стан заявок. Водночас статус користувача в системі буде динамічно відображатися як «онлайн» чи «офлайн».

Завдяки технології Ajax користувачі зможуть взаємодіяти із системою без затримок, що сприятиме підвищенню якості обслуговування та ефективності роботи мережі ресторанів. Усі функції, такі як реєстрація, авторизація, управління заявками, будуть інтегровані у модульну структуру, яка дозволить зручно управляти кодом і швидко вносити зміни чи оновлення у функціонал системи [13].

3.3. Схеми бази даних

Під час розробки веб-застосунку для підтримки виробничого процесу в мережі ресторанів «Bubbly», властивості об’єктів, таких як користувачі, замовлення або меню, зберігаються у базі даних, а їх функціональна поведінка реалізується через методи-програмні функції, які реагують на події, ініційовані користувачами. Це дозволяє забезпечити інтерактивність та ефективність взаємодії користувача із системою.

Інформаційна модель бази даних, яка проектується для цього веб-застосунку, має бути орієнтованою на розв’язання специфічних завдань ресторанної галузі, залишаючись при цьому незалежною від конкретної бази

даних, операційної системи чи апаратного забезпечення. Це забезпечує адаптивність системи до різних умов експлуатації.

Основною вимогою до інформаційної моделі є адекватне відображення предметної області ресторанного бізнесу. Інформаційна модель має бути інтегрованим описом усіх компонентів системи, відображати потреби користувачів і забезпечувати консистентність даних. Така модель включає інформаційні потоки, сутності, зв'язки між ними та може бути представлена у вигляді ER-діаграм або реляційної схеми.

Структура інформаційної моделі орієнтована на виявлення і інтеграцію всіх інформаційних вимог, включаючи зв'язки між об'єктами, незалежно від їх змісту чи способу зберігання. Вона повинна бути масштабованою, що дозволить у майбутньому додавати нові дані без значних змін у її структурі.

Основними компонентами інформаційної моделі є сутності, їх атрибути, ідентифікатори (ключі) та зв'язки. Наприклад, сутністю може бути «Користувач», «Замовлення» або «Страви». Атрибути – це характеристики сутності, такі як ім'я користувача, статус замовлення чи ціна страви. Кожен об'єкт ідентифікується за унікальним ключем, що дозволяє однозначно знайти його у базі даних.

Зв'язки між сутностями дозволяють відображати взаємодії між різними компонентами системи. Наприклад, зв'язок між сутністю «Користувач» і «Замовлення» описує, які замовлення належать конкретному клієнту. Використання зв'язків «один-до-одного», «один-до-багатьох» чи «багато-до-багатьох» дозволяє проектувати систему для складних сценаріїв взаємодії.

Для моделювання інформаційної системи використовується мова ER-діаграм. У цій нотації сутності зображуються прямокутниками, зв'язки – ромбами, а атрибути – овалами. Наприклад, сутність «Меню» може мати атрибути «Назва страви», «Ціна» та «Категорія», а зв'язки з іншими сутностями, такими як «Замовлення», вказуватимуть, які страви включені до замовлення.

Сутності класифікуються на стрижневі, асоціативні та характеристичні. Стрижневі сутності, такі як «Користувач» чи «Замовлення», є незалежними і

виконують основні функції у системі. Асоціативні сутності відображають зв’язки між кількома об’єктами, наприклад, замовленням і його деталями. Характеристичні сутності уточнюють чи описують інші об’єкти, наприклад, статус замовлення.

Цей підхід до проектування бази даних забезпечує зручність у використанні, легкість у масштабуванні та адаптацію до нових функціональних вимог, що є критично важливим для розробки інтерактивного веб-застосунку в сфері ресторанного бізнесу [16, с. 53]. Елементи розширеної мови ER-Діаграм зображені на рис. 3.6:[5]

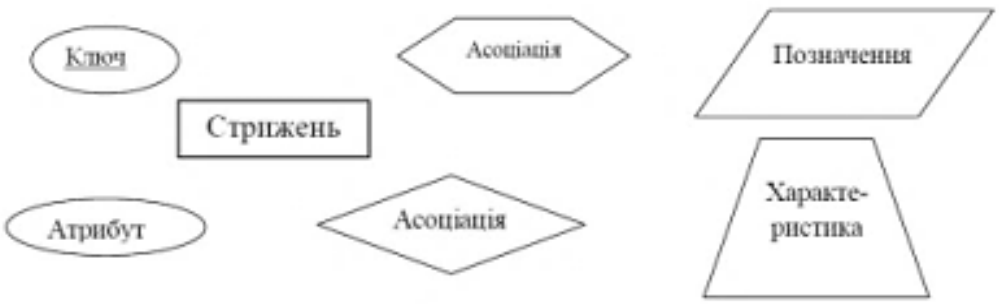

Рисунок 3.6 – Елементи розширеної мови ER-Діаграм

Позначення й характеристики не є повністю незалежними сутностями, оскільки вони припускають наявність деякої іншої сутності, яка буде «позначатися» або «характеризуватися». Однак вони все-таки являють собою окремі випадки сутності й можуть, звичайно, мати властивості, можуть брати участь в асоціаціях, позначеннях і мати свої власні (більш низького рівня) характеристики [10].

Для визначення переліку й структури збережених даних треба зібрати інформацію про реальні й потенційні застосування, а також про користувачів бази даних, а при побудові інфологічної моделі слід опікуватися лише про надійність зберігання цих даних, не звертаючи уваги на застосування та користувачів, для яких створюється база даних [16, с. 46–48].

У нашому проекті моделювання бази даних і розробка програмного коду орієнтовані на забезпечення функціональної взаємодії між користувачами, що працюють у мережі ресторанів.

Моделювання бази даних

1. Структура бази даних:

Таблиця `users`. Використовується для зберігання даних про користувачів, таких як їх унікальний ідентифікатор (`id`), ім’я користувача (`username`), зашифрований пароль (`password`) і статус (`status`). Ідентифікатор є первинним ключем, що гарантує унікальність записів.

```sql
CREATE TABLE `users` (
`id` INT (5) NOT NULL AUTO_INCREMENT,
`username` VARCHAR (40) NOT NULL UNIQUE,
`password` VARCHAR (255) NOT NULL,
`status` TINYINT (1) NOT NULL DEFAULT 0,
PRIMARY KEY (`id`)
) ENGINE=InnoDB;
```

Використання хешування паролів (наприклад, через `password_hash`) забезпечує безпеку зберігання даних.

Таблиця `messages`. Призначена для зберігання повідомлень між користувачами. Поля містять інформацію про відправника (`from_user`), отримувача (`to_user`), текст повідомлення (`text`), статус повідомлення (`status`), а також мітки для видалення з обох сторін (`delete_to` та `delete_from`).

```sql
CREATE TABLE `messages` (
`id` INT (5) NOT NULL AUTO_INCREMENT,
`to_user` INT (5) NOT NULL,
`from_user` INT (5) NOT NULL,
`text` VARCHAR (255) NOT NULL,
`status` TINYINT (1) NOT NULL DEFAULT 0,
`date` DATETIME DEFAULT CURRENT_TIMESTAMP,
`delete_to` TINYINT (1) DEFAULT 0,
`delete_from` TINYINT (1) DEFAULT 0,
PRIMARY KEY (`id`)
) ENGINE=InnoDB;
```

Ця структура дозволяє гнучко керувати статусом повідомлень та організовувати взаємодію між користувачами.

2. Особливості моделювання:

– Реляційні зв’язки дозволяють асоціювати повідомлення з конкретними користувачами через поля `to_user` та `from_user`.
– Атрибути `delete_to` і `delete_from` надають можливість зберігати інформацію про те, чи було повідомлення видалено з боку конкретного користувача.

Нами запропонована наступна послідовність програмної реалізації:

1. Реєстрація користувачів. Код у файлі `add_user.php` відповідає за додавання нових користувачів до бази даних. Функція `registerUser` перевіряє довжину і формат даних, а потім зберігає зашифрований пароль у базі.

```php
if (empty($name) || empty($password)) {
die («Не введені обов’язкові поля.»);
}
$hashedPassword = password_hash($password, PASSWORD_BCRYPT);
registerUser($name, $hashedPassword);
```

2. Надсилання повідомлень. У файлі `send_message.php` реалізовано функціонал надсилання повідомлень. Використовується підготовлений SQL-запит із параметрами для уникнення SQL-ін’єкцій.

```php
$stmt = $conn->prepare («INSERT INTO messages (to_user, from_user,
    text) VALUES (?,?,?)»);
```

3. Видалення повідомлень. Код у `delete_message.php` дозволяє позначати повідомлення на видалення з боку конкретного користувача. Реалізація передбачає оновлення відповідних полів у таблиці `messages`.

4. JavaScript для взаємодії. Скрипт на боці клієнта забезпечує асинхронну взаємодію з сервером за допомогою `fetch` API, що значно підвищує зручність користувача:

```javascript
fetch («send_message.php», {
method: «POST»,
headers: { «Content-Type»: «application / x-www-form-urlencoded»
     },
body: `text=${encodeURIComponent (text)}&to=${encodeURIComponent
     (to)}`
})
```

Запропонований підхід має ряд особливостей:

1. Хешування паролів та використання підготовлених запитів захищає від поширених атак, таких як SQL-ін’єкції.
2. Використання асинхронних запитів через JavaScript дозволяє зменшити затримки під час взаємодії із системою.
3. Реляційна структура бази даних та чітко визначені залежності між сутностями забезпечують можливість розширення функціоналу.

Ця реалізація відображає гнучкий та адаптивний підхід до моделювання даних і програмування, який відповідає завданням підтримки виробничого процесу в мережі ресторанів «Bubbly».

3.4. Модель бази даних

Логічна модель спроектованої бази даних є моделлю, яка відображає логічні зв’язки між елементами даних незалежно від їх змісту та середовища зберігання. Ця модель побудована на основі інформаційних одиниць, які використовуються у конкретній базі даних, для якої проектується модель. Процес створення логічної моделі називається логічним проектуванням. Опис логічної структури бази даних мовою самої бази називається схемою.

Хоча логічне проектування бази даних є абстрактною логічною структурою, його впливають можливості фізичної організації даних, які залежать від конкретної бази. Тому знання особливостей фізичної організації даних є корисним при проектуванні логічної структури.

У рамках цієї курсової роботи, логічна модель бази даних представлена на рисунку 3.7. Вона відображає структуру бази даних, що допомагає забезпечити більш точну побудову та зрозумілість бази даних [5].

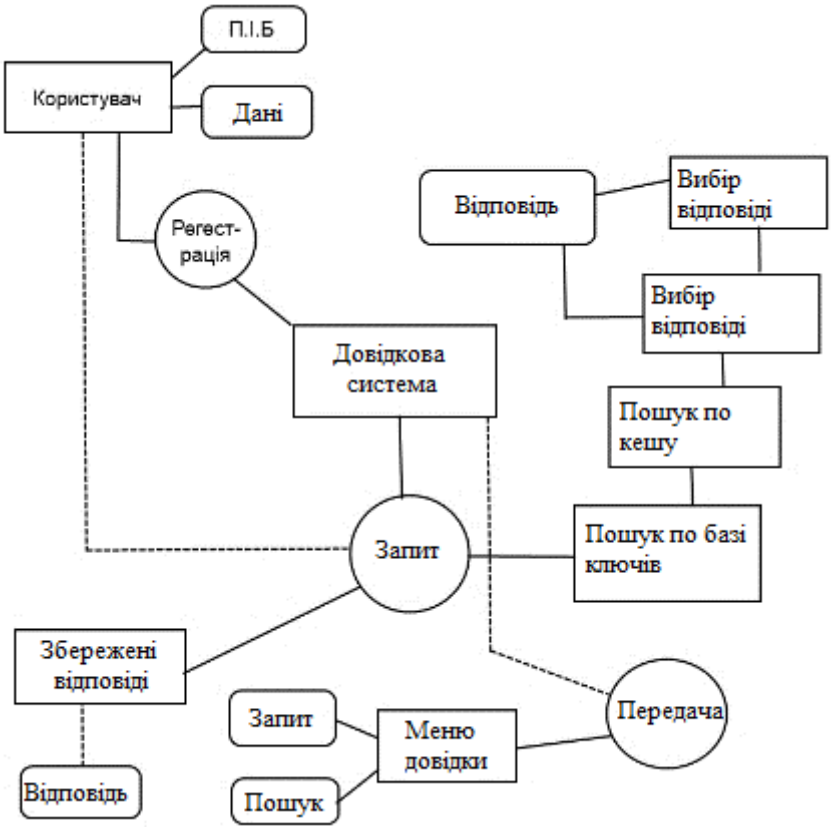

Рисунок 3.7 – Даталогічна модель бази даних проектованої системи

Побудуємо ER-діаграму. При проведенні зв’язку між сутностями первинний ключ мігрує в дочірню сутність. Дані таблиці будуть мати наступній вигляд в ER-діаграмі.

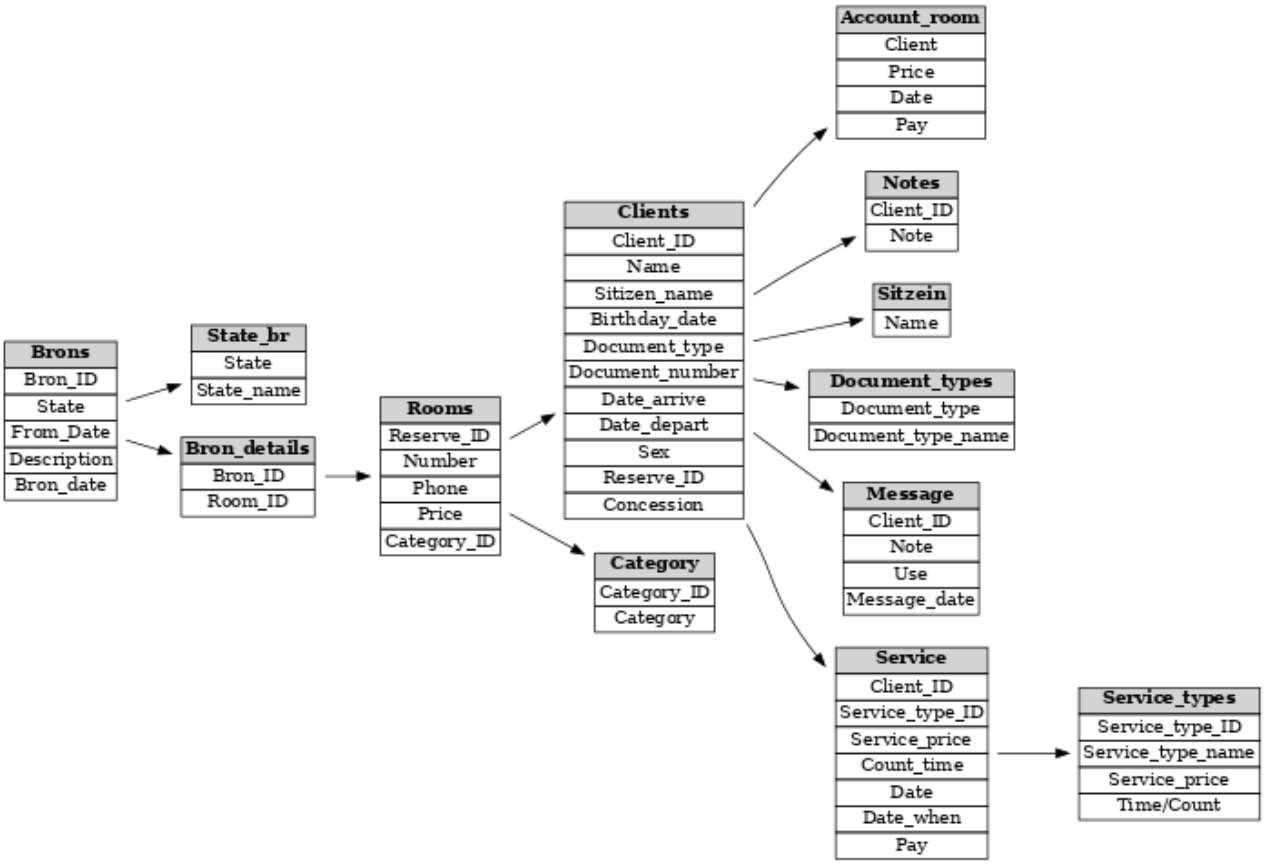

Рисунок 3.8 – ER-діаграма бази даних проектованого веб-застосунку

3.5. Розробка алгоритмів рішення функціональної задачі

Система працює за алгоритмом, який відображає його функціональну діяльність на рівні схеми. Фактично, робота системи не має остаточної кінцевої точки, оскільки процес роботи не зупиняється. Однак, можна вважати, що система досягає кінцевої точки, коли пристрій перестає працювати повністю або вийде з ладу. Система аналізує сигнали та виконує функції відповідно до запрограмованих реакцій на команди, враховуючи права на виконання відповідних операцій.

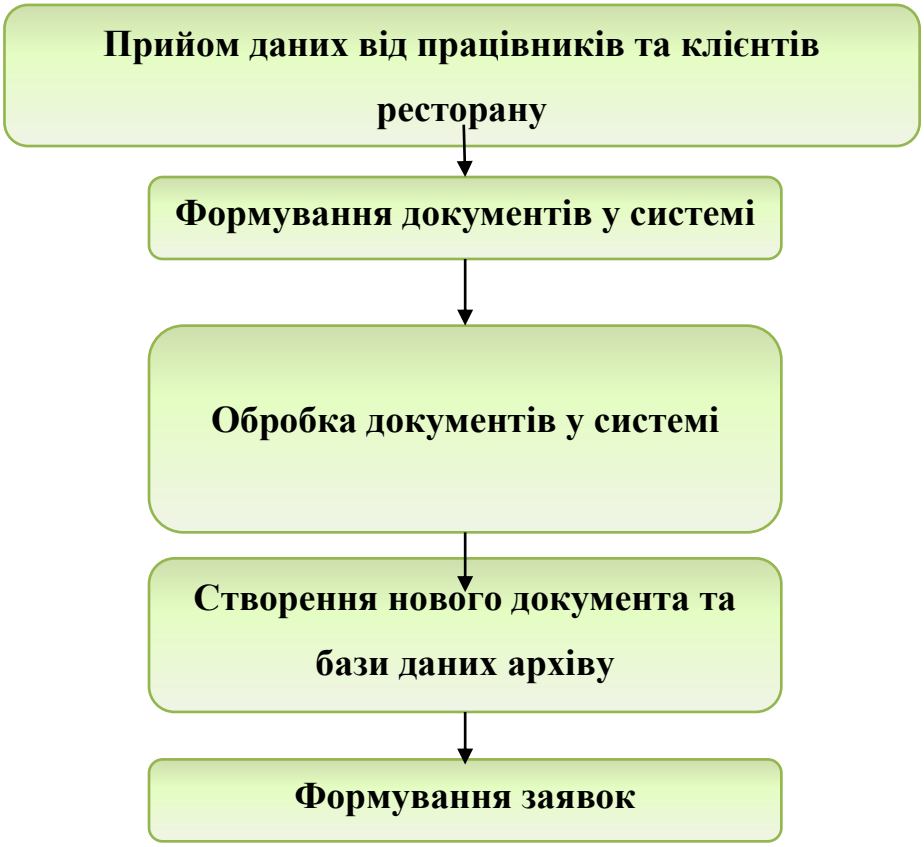

Рисунок 3.9 – Загальний алгоритм роботи користувача в системі

Алгоритм роботи користувача включає наступні етапи:

– Прийом даних від працівників та клієнтів ресторану.
– Формування документів у системі.
– Обробка документів у системі.
– Створення нового документа та бази даних архіву.
– Формування заявок.

3.6. Опис структури програми

Усі JavaScript-файли, що відповідають за динамічну взаємодію з користувачем, будуть зберігатися в окремій папці `JavaScript`. Ця папка міститиме скрипти, необхідні для роботи веб-застосунку, включаючи обробку подій, взаємодію з сервером через Ajax та оновлення інтерфейсу. Для роботи з технологією Ajax передбачено використання бібліотеки jQuery, яка буде розміщена у папці `jquery` [1].

Файл `add_user.php` відповідатиме за реєстрацію нових користувачів у системі, включаючи перевірку валідності введених даних, таких як ім’я користувача та пароль. Для управління сесіями користувачів, авторизації, збереження документів у базі даних і взаємодії з нею буде використовуватися файл `data_valid.php`, який міститиме необхідні функції для підключення до бази, перевірки користувачів та обробки їхніх даних.

Функціонал видалення документів користувача реалізується у файлі `delete.php`, тоді як файл `function.php` відповідатиме за підключення основних функціональних компонентів системи. Початковий вхід до веб-застосунку здійснюватиметься через файл `login.php`, який дозволить користувачам зареєструватися або авторизуватися.

Для виходу із системи передбачено файл `logout.php`, який завершуватиме сесію користувача. Головна сторінка веб-додатку, де користувач взаємодіє із системою, генеруватиметься файлом `member.php`. Цей файл забезпечує основний функціонал, зокрема обмін повідомленнями між користувачем і сервером, а також перевірку валідності даних під час входу [1].

Відображення всіх повідомлень користувача здійснюється через файл `messages.php`, який зчитує дані з бази та відображає їх у вигляді зручного інтерфейсу. Файл `messages_user.php` відповідає за вивід заявок конкретного користувача з бази даних, а файл `output.php` містить функції для створення динамічних елементів веб-сторінки, таких як форми реєстрації чи повідомлення.

Реєстрація нового користувача реалізується через файл `register_new.php`, який створює новий запис у базі даних. Відправка повідомлень користувачами виконується через файл `send.php`, а безпосередній запис заявки в базу даних – через файл `send_message.php`. Для стилізації інтерфейсу веб-застосунку передбачено файл `style. css`.

Для тестування підключення до бази даних та перевірки її функціоналу використовуються файуктура бази даних дозволяє чітко організувати збереження і обробку інформації, забезпечуючи її консистентність та швидкий доступ. Реляційні зв’язки між таблицями, такі як асоціація повідомлень з

конкретними користувачамли `test.php` та `test_db.php`. Виведення списку зареєстрованих або активних користувачів здійснюється через файл `update.php`, який оновлює інформацію про користувачів у реальному часі [1].

Структура бази даних. Назва бази даних – `bubbly`. У базі даних передбачено дві основні таблиці:

1. `users` – для збереження інформації про профілі користувачів (імена, паролі, статуси).

2. `messages` – для збереження всіх повідомлень між користувачами, включаючи текст повідомлення, статус, мітки видалення тощо.

Така стри, значно спрощують управління даними.

Загальна структура веб-застосунку розроблена з урахуванням модульності, що дозволяє легко масштабувати або доповнювати функціонал у майбутньому. Такий підхід забезпечує ефективну роботу системи навіть за умови високого навантаження.

Таблиця 3.1. Структура таблиці message бази даних MySql

Назва поля	Тип даних	Розмір	Не пусте	Автоінкремент / Інше значення
id	int	5	not null	auto_increment
to_mes	Int	5	Not null	-
From_mes	Int	5	Not null	-
text	Varchar	50	Not null	-
Status	int	2	Not null	-
data	Varcahr	50	null	Default null
Delete_to	int	10	Not null	-
Delete_from	int	10	Not null	-

Таблиця `messages` розроблена для збереження інформації про повідомлення між користувачами. Вона складається з декількох полів, кожне з яких виконує окрему функцію в системі. Це забезпечує структуроване зберігання даних, що дозволяє ефективно керувати інформацією про повідомлення. Ось опис кожного поля:

1. Поле `id`. Поле призначене для унікальної ідентифікації кожного повідомлення. Тип даних `INT` з автоматичним збільшенням (`AUTO_INCREMENT`) гарантує, що кожне повідомлення має унікальний індекс. Це основний ключ таблиці, що використовується для швидкого доступу до конкретного повідомлення.

2. Поле `to_user`. Це поле зберігає ідентифікатор користувача, якому адресоване повідомлення. Тип даних `INT` (5 цифр) використовується для зберігання числового значення, яке відповідає унікальному ідентифікатору користувача в системі. Поле є обов'язковим, оскільки кожне повідомлення має мати отримувача.

3. Поле `from_user`. Це поле зберігає ідентифікатор користувача, який відправив повідомлення. Воно також має тип даних `INT` (5 цифр) і є обов'язковим для заповнення. Це забезпечує можливість відстежувати автора кожного повідомлення.

4. Поле `text`. Поле зберігає текст повідомлення. Тип даних `VARCHAR` (255 символів) дозволяє зберігати як короткі, так і довші повідомлення. Поле обов'язкове для заповнення, оскільки повідомлення без тексту не мають сенсу.

5. Поле `status`. Це поле використовується для позначення статусу повідомлення. Значення `0` означає, що повідомлення ще не прочитане, а `1` вказує на те, що повідомлення було прочитано. Тип даних `TINYINT` (1 символ) обрано для збереження невеликого обсягу даних.

6. Поле `date`. Поле зберігає дату та час відправлення повідомлення. Тип даних `DATETIME` дозволяє точно визначати, коли було створено повідомлення. За замовчуванням поле заповнюється поточним часом створення запису (`DEFAULT CURRENT_TIMESTAMP`).

7. Поле `delete_to`. Поле зберігає ідентифікатор користувача, який першим видалив повідомлення. Тип даних `TINYINT` дозволяє зберігати значення, що визначають статус видалення. Якщо інший користувач також видаляє це повідомлення, воно видаляється з бази даних.

8. Поле `delete_from`. Поле зберігає ідентифікатор користувача, який видалив повідомлення другим. Аналогічно до `delete_to`, це поле визначає, чи видалене повідомлення остаточно з обох сторін. Тип даних – `TINYINT`.

Моделювання таблиці `users`. Таблиця `users` призначена для збереження профільної інформації про кожного користувача. Вона забезпечує основу для функціонування системи користувачів. Ось опис її полів:

1. Поле `id`. Унікальний ідентифікатор користувача. Поле має тип даних `INT` з автоматичним збільшенням (`AUTO_INCREMENT`) і використовується як основний ключ таблиці.

2. Поле `username`. Поле зберігає ім’я користувача, яке є унікальним для кожного профілю. Тип даних `VARCHAR` (40 символів) забезпечує достатній простір для створення унікальних імен.

3. Поле `password`. Поле зберігає хеш пароля користувача. Тип даних `VARCHAR` (255 символів) дозволяє зберігати безпечні хеші, створені за допомогою сучасних алгоритмів хешування, таких як `BCRYPT`.

4. Поле `status`. Це поле використовується для позначення статусу користувача в системі. Значення `1` означає, що користувач перебуває онлайн, а `0` – офлайн. Тип даних `TINYINT` обрано для оптимізації зберігання.

Таблиці `users` та `messages` взаємопов’язані через поля `to_user` і `from_user`. Це дозволяє ефективно обробляти повідомлення, зберігаючи інформацію про відправників і отримувачів. Такий підхід забезпечує консистентність даних та спрощує їх обробку у запитах до бази.

Запропонована модель має переваги:

1. Гнучкість: Легке додавання нових полів чи функціоналу без зміни структури існуючих даних.

2. Ефективність: Оптимізація для роботи з великими обсягами даних завдяки реляційній моделі.

3. Масштабованість: Модель підтримує розширення функціоналу, наприклад, впровадження нових статусів повідомлень або додаткових атрибутів користувача.

Така структура бази даних є базисом для роботи веб-застосунку, забезпечуючи його стабільність, функціональність і надійність.

Таблиця 3.2. Структура таблиці register бази даних MySql

Назва поля	Тип даних	Розмір	Не пусте	Автоінкремент / Інше значення
id	int	5	not null	auto_increment
username	varchar	40	Not null	-
pasword	varchar	40	Not null	-
status	int	5	Not null	-

Поле `id` виконує роль унікального ідентифікатора користувачів у системі. Воно автоматично створює унікальне значення для кожного нового запису, забезпечуючи чітку ідентифікацію користувача в базі даних. Тип даних цього поля – `INT`, що є цілочисельним, з максимальним розміром у 5 цифр. Поле є автоінкрементованим (`AUTO_INCREMENT`) і обов'язковим для заповнення, тобто не може бути порожнім (`NOT NULL`).

Поле `username` використовується для збереження імені користувача, яке він вводить під час реєстрації або авторизації. Тип даних `VARCHAR` дозволяє зберігати текст довжиною до 40 символів. Поле є унікальним (`UNIQUE`) для запобігання дублюванню імен у системі, а також обов'язковим для заповнення.

Поле `password` відповідає за збереження паролів користувачів. Для цього також використовується тип даних `VARCHAR`, який дозволяє зберігати хешовані паролі. Довжина поля може сягати 255 символів, щоб забезпечити сумісність із сучасними алгоритмами хешування, такими як `BCRYPT`. Це поле також є обов'язковим.

Поле `status` призначене для зберігання статусу користувача в системі. Значення `1` вказує на те, що користувач онлайн, тоді як `0` означає, що користувач офлайн. Тип даних `TINYINT` обрано через його ефективність для зберігання невеликих числових значень. Це поле використовується для відображення активності користувачів.

База даних налаштована на використання кодування `utf8`, що забезпечує сумісність із міжнародними стандартами. Це дозволяє системі підтримувати різноманітні мови та символи, забезпечуючи коректне відображення даних у сучасних веб-браузерах.

Опишемо код запуску системи. Основний файл `login.php` відповідає за ініціалізацію роботи веб-застосунку, дозволяючи користувачам авторизуватися або зареєструватися. Код цього файлу простий і слугує для підключення основних функціональних компонентів системи. Ось приклад коду:

```
``php
<?php
require_once('function.php');
html_header();
form();
html_footer();
?>
```
```

У цьому файлі реалізовано підключення до головного функціонального файлу системи `function.php`, який забезпечує базові операції, такі як обробка форм та виведення основних елементів сторінки. Функція `html_header()` відповідає за формування верхньої частини веб-сторінки, тоді як `html_footer()` додає нижню частину.

Ключовим елементом цього файлу є функція `form()`, яка забезпечує відображення форми авторизації або реєстрації користувача, залежно від стану користувача в системі.

Переваги такої моделі:

1. Унікальність: Поле `id` гарантує унікальність кожного запису, що спрощує ідентифікацію користувачів.

2. Гнучкість: Поле `username` дозволяє користувачам обирати зручні для них імена, які перевіряються на унікальність.

3. Безпека: Використання хешованих паролів у полі `password` забезпечує надійність збереження конфіденційних даних.

4. Міжнародна сумісність: Кодування `utf8` дозволяє системі підтримувати користувачів із різних регіонів.
```

Завдяки чіткій структурі та належному розподілу функцій, проектована система є ефективною платформою для підтримки виробничих процесів у галузі масового харчування.

3.7. Розробка програмного забезпечення

Програмна реалізація веб-застосунку для системи підтримки роботи ресторану є ключовим етапом розробки, який забезпечує інтеграцію сучасних технологій для вирішення задач автоматизації та покращення взаємодії з користувачами. У нашому випадку, створення сайту для мережі ресторанів Bubbly передбачає використання таких технологій:

1. PHP для серверного програмування та обробки запитів користувачів.
2. JavaScript для створення інтерактивних елементів і забезпечення динамічності інтерфейсу.
3. HTML для опису структури веб-сторінок.
4. jQuery, як бібліотека JavaScript, для реалізації технології Ajax і спрощення роботи з DOM.
5. CSS для стилізації елементів веб-сторінок [1].

HTML (HyperText Markup Language) використовується для створення структури веб-сторінок, зокрема форм, таблиць, списків, посилань і мультимедійних елементів. Це базова мова для опису форматування, яка працює у тісному поєднанні з іншими технологіями, такими як CSS та JavaScript [12, с 122–175].

Кожен HTML-документ містить три основні частини:

1. Декларація типу документа (`<!DOCTYPE html>`), яка вказує браузеру, що документ використовує HTML5.

2. Заголовкова частина (`<head>`), яка містить метадані сторінки, посилання на стилі та скрипти.

3. Основна частина (`<body>`), де розташовані всі елементи, які відображаються у браузері.

Приклад базового HTML-документа для нашого проєкту:

```html
<!DOCTYPE html>
<html>
<head>
<title>Bubbly Restaurant</title>
<meta charset=«UTF – 8»>
<link rel=«stylesheet» href=«styles. css»>
</head>
<body>
<header>
<h1>Ласкаво просимо до ресторану Bubbly!</h1>
</header>
<main>
<p>Ознайомтесь з нашим меню та забронюйте столик.</p>
</main>
</body>
</html>
```

Для відображення даних, наприклад меню або замовлень, використовуються таблиці. Таблиця в HTML має структуру з рядків (`<tr>`) та клітинок (`<td>` для даних і `<th>` для заголовків). Наприклад:

```html
<table border=«1»>
<tr>
<th>Назва страви</th>
<th>Ціна</th>
</tr>
<tr>
<td>Піца Маргарита</td>
<td>120 грн</td>
</tr>
<tr>
<td>Салат Цезар</td>
<td>100 грн</td>
</tr>
</table>
```

Цей підхід дозволяє зручно представляти дані меню у вигляді таблиці, яка легко адаптується до стилізації за допомогою CSS.

Технологія Ajax використовується для оновлення даних на сторінці без її повного перезавантаження. Це забезпечує зручність користувачів, особливо при бронюванні столиків або оформленні замовлень [13].

Приклад JavaScript для відправки запиту на сервер:

```javascript
document. querySelector("#reserveButton»). addEventListener
     («click», function () {
const tableId = document. querySelector("#tableId»). value;

fetch («reserve_table.php», {
method: «POST»,
headers: { «Content-Type»: «application / x-www-form-urlencoded»
     },
body: `tableId=${encodeURIComponent (tableId)}`
})
.then (response => response. text())
.then (data => alert (data))
.catch (error => console. error («Помилка:», error));
});
```

Розробимо таку структуру файлів веб-застосунку:

1. Папка `JavaScript`: містить усі JavaScript-файли, які використовуються на сторінках сайту.

2. Папка `jquery`: містить бібліотеку jQuery, яка полегшує інтеграцію Ajax і роботу з DOM.

3. Файл `add_user.php`: відповідає за реєстрацію користувачів у системі, включаючи перевірку введених даних.

4. Файл `delete_message.php`: забезпечує видалення повідомлень або замовлень користувачів.

5. Файл `function.php`: містить функції для роботи з базою даних, авторизації та обробки даних.

6. Файл `login.php`: забезпечує вхід до системи або реєстрацію користувача.

7. Файл `styles. css`: відповідає за стилізацію всіх елементів сайту.

Отже, проєктована система для ресторану буде ефективною, зручною у використанні та легко адаптованою до потреб клієнтів.

Інтерактивність та динамічність у веб-додатках є ключовими елементами сучасного веб-розроблення, особливо у нашому проекті системи для ресторану. Одним із важливих аспектів є робота з графічними елементами, зокрема із зображеннями, їх атрибутами, інтерактивними картами та можливостями, які

надають списки та гіперпосилання. Розглянемо ці елементи в контексті нашої системи.

Зображення є важливим елементом інтерфейсу, який сприяє кращому сприйняттю інформації користувачами. Проте можуть виникати ситуації, коли зображення не завантажуються через повільне з'єднання з Інтернетом або налаштування користувача, які блокують графічний контент. Для таких випадків використовується атрибут `alt`, який дозволяє показувати текстовий опис зображення, щоб користувач міг зрозуміти, що мало бути показано [23].

Приклад коду:

```
```html
<img src=«menu_item. jpg» alt=«Зображення страви» title=«Назва
    страви» width=«200» height=«200»>
```
```

Атрибут `alt` також допомагає підвищити доступність сайту для користувачів із порушеннями зору, дозволяючи екранним рідерам читати опис зображення.

HTML дозволяє створювати інтерактивні області на зображеннях за допомогою тегів `<map>` і `<area>`. Наприклад, у нашому проекті інтерактивна карта може використовуватися для візуалізації розташування столиків у залі ресторану. Користувач зможе натискати на певний столик, щоб отримати додаткову інформацію або забронювати його.

```
```html
<img src=«restaurant_map. jpg» usemap="#tableMap» alt=«Карта
    залу»>
<map name=«tableMap»>
<area shape=«rect» coords=«34,44,270,350» alt=«Столик 1»
    href=«table1. html»>
<area shape=«circle» coords=«400,300,50» alt=«Столик 2»
    href=«table2. html»>
</map>
```
```

Цей код дозволяє зробити інтерактивну карту залу, де кожен столик є посиланням.

Списки допомагають структурувати інформацію, наприклад, меню страв або доступні акції. HTML надає три типи списків:

1. Упорядковані списки (`<ol>`):

```html
<ol>
<li>Супи</li>
<li>Салати</li>
<li>Десерти</li>
</ol>
```

2. Невпорядковані списки (`<ul>`):

```html
<ul>
<li>Салат Цезар</li>
<li>Грецький салат</li>
</ul>
```

3. Списки означень (`<dl>`):

```html
<dl>
<dt>Салат Цезар</dt>
<dd>Класичний салат з куркою та пармезаном</dd>
</dl>
```

Списки спрощують навігацію та допомагають користувачам швидко знаходити потрібну інформацію.

Гіперпосилання створюються за допомогою тегу `<a>`. Наприклад, для переходу на сторінку бронювання столика:

```html
<a href=«reserve_table. html»>Забронювати столик</a>
```

Гіперпосилання можна використовувати як для переходу між сторінками сайту, так і для навігації в межах однієї сторінки за допомогою якорів:

```html
<a href="#menu»>Перейти до меню</a>
<h2 id=«menu»>Меню ресторану</h2>
```

Додаткові функції:

1. Атрибут `target` дозволяє налаштовувати, де буде відкриватися сторінка:

```
- `_blank`: у новій вкладці.
- `_self`: у тій самій вкладці.
- `_parent` або `_top`: у контексті фреймів.
```

2. Зображення як посилання:

```
```html
<a href=«menu. html»><img src=«menu_preview. jpg» alt=«Попередній
    перегляд меню»></a>
```
```

Отже, проведена робота із зображеннями, списками та гіперпосиланнями є невід'ємною частиною функціональності веб-додатків, особливо таких, як наш сайт для ресторану. Вони забезпечують користувачам зручність у навігації та взаємодії із системою, допомагають створити зрозумілий інтерфейс і полегшують доступ до необхідної інформації [23].

Проектування графіки на веб-сторінці для ресторану є невід'ємною частиною створення привабливого і функціонального інтерфейсу. Графічні елементи не лише покращують візуальне сприйняття, а й створюють атмосферу закладу, допомагаючи користувачам краще орієнтуватися в меню, акціях і можливостях сайту.

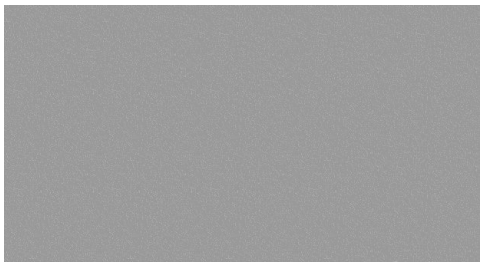

Рисунок 3.10 – Фоновий малюнок

Фонові зображення додають веб-сторінці естетичності. Наприклад, у нашій системі можна використовувати фон, що відображає стиль інтер'єру ресторану. Зображення фонового малюнка задається через атрибут `background` у тегу `<body>`:

```
```html
```
```

```
<body background=«images / restaurant_bg. jpg»>
```

Це зображення буде автоматично повторюватися по горизонталі та вертикалі, якщо його розмір менший за вікно браузера. Для унікального розташування фону можна використовувати CSS-стилі:

```css
body {
background-image: url('images / restaurant_bg. jpg');
background-repeat: no-repeat;
background-size: cover;
}
```

Зображення меню або страв можна вставити за допомогою тега `<img>`. Наприклад, фотографії окремих позицій меню:

```html
<img src=«images / dish_1. jpg» alt=«Салат Цезар» width=«300»
     height=«200»>
```

Використання атрибутів `width` і `height` дозволяє браузеру резервувати простір для зображення, поки воно завантажується, щоб текст завантажувався паралельно з картинкою. Це особливо важливо для оптимізації роботи сайту [23].

Якщо зображення недоступне, користувач побачить текст, заданий у атрибуті `alt`. Це важливо для доступності сайту:

```html
<img src=«images / dish_2. jpg» alt=«Паста Карбонара» width=«300»
     height=«200»>
```

За потреби можна додати рамку до зображення через атрибут `border`:

```html
<img src=«images / dish_3. jpg» alt=«Десерт Тірамісу» width=«300»
     height=«200» border=«1»>
```

Рамка допомагає виділити окремі елементи або створити декоративний ефект.

Для створення інтерактивності, наприклад, вибору столиків у залі ресторану, використовують карти зображень:

```html
<img src=«images / hall_map. jpg» usemap="#hallMap» alt=«Карта
    залу»>
<map name=«hallMap»>
<area shape=«rect» coords=«10,10,100,100» href=«table1. html»
    alt=«Столик 1»>
<area shape=«circle» coords=«200,200,50» href=«table2. html»
    alt=«Столик 2»>
</map>
```

Цей код дозволяє користувачеві натискати на зображення для вибору столика, що значно покращує взаємодію із системою.

Для текстових блоків, які супроводжують зображення, використовують атрибут `align`, що задає розташування зображення відносно тексту:

- `align=«left»` – текст обтікає зображення з правого боку.
- `align=«right»` – текст обтікає зображення з лівого боку.
- `align=«top»`, `align=«middle»`, `align=«bottom»` – розташування зображення у рядку тексту.

```html
<img src=«images / logo. jpg» alt=«Логотип ресторану» align=«left»
    hspace=«10» vspace=«10»>
<p>Наш ресторан пропонує вишукані страви...</p>
```

Сучасні браузери дозволяють вставляти відео через HTML5-тег `<video>`:

```html
<video width=«320» height=«240» controls>
<source src=«videos / restaurant_tour. mp4» type=«video / mp4»>
Ваш браузер не підтримує відео.
</video>
```

Це ідеально підходить для презентації інтер’єру ресторану або спеціальних заходів.

Отже, проектування графіки для веб-додатку ресторану має на меті створення зручного, привабливого та функціонального інтерфейсу.

Використання фонів, зображень, інтерактивних елементів та відео сприяє покращенню взаємодії користувачів із системою та відображає стиль і атмосферу ресторану [20, 23].

Наступним важливим етапом у створенні веб-додатку для ресторану є наповнення ресурсу контентом. Контент – це основа, яка формує враження про сайт і забезпечує його функціональність. Він включає текстову інформацію, зображення та мультимедійні матеріали. Основною метою наповнення є залучення аудиторії, спрощення взаємодії з ресурсом та підвищення видимості у пошукових системах.

Контент має бути:

– Ясним і зрозумілим для цільової аудиторії. Текст має враховувати специфіку ресторанної тематики, акцентуючи увагу на меню, акціях та особливостях закладу.
– Унікальним – тексти та зображення не повинні бути скопійовані з інших джерел. Це важливо не лише для відповідності закону про авторські права, а й для покращення позицій сайту в пошукових системах 011, с. 78].
– Оптимізованим для SEO – контент має містити ключові слова, які допоможуть користувачам швидше знайти сайт за запитами, пов’язаними з рестораном.

Унікальний контент, такий як оригінальні фотографії страв, відеопрезентації закладу або історія ресторану, дозволяє створити позитивне враження у відвідувачів і покращує індексацію сайту пошуковими системами 011, с. 79]. Наприклад, фотографії можна використовувати для ілюстрації меню, а відео – для демонстрації процесу приготування популярних страв. Це сприяє залученню користувачів і їхньому поверненню до ресурсу.

Частиною контенту може бути матеріал, створений користувачами сайту: відгуки, коментарі до страв або рекомендації. Наприклад, функція оцінки страв та залишення коментарів допоможе залучити нових відвідувачів і сприятиме покращенню взаємодії з постійними клієнтами.

Контент має бути адаптованим під пошукові запити, пов’язані з ресторанною тематикою, наприклад: **«кращі страви дня», «бронювання столика онлайн», «меню ресторану Bubbly».** Це допоможе залучати органічний трафік і скоротить витрати на рекламні кампанії.

Наповнення сайту передбачає:

1. Головна сторінка – коротка інформація про ресторан, акції та посилання на інші розділи.

2. Меню – детальний опис страв із фотографіями та цінами.

3. Сторінка бронювання – інтерактивна форма для вибору столика.

4. Розділ відгуків – коментарі від клієнтів.

5. Контакти – адреса, телефон, карта розташування.

Окрім текстового та графічного контенту, важливим елементом веб-додатку є база даних. У базі даних зберігатиметься інформація про меню, користувачів та їхні замовлення. Наприклад:

- Таблиця users для обліку даних клієнтів.
- Таблиця menu_items для зберігання інформації про страви.
- Таблиця reservations для обробки бронювань.

Отже, наповнення контентом є важливим етапом розробки веб-додатку для ресторану. Унікальний, якісний і оптимізований контент дозволяє не лише привернути увагу користувачів, але й підвищити видимість сайту у пошукових системах. Технічне завдання передбачає створення сучасного ресурсу з інтерактивними елементами, мультимедіа та функціоналом бази даних, який забезпечить ефективну взаємодію з клієнтами 011, с. 80].

Побудова файлової структури веб-додатку для ресторану є важливим етапом у реалізації проєкту. Вона дозволяє чітко розподілити програмний код за його функціональним призначенням, полегшуючи розробку, тестування та обслуговування системи. Впровадження цього підходу сприяє модульності та забезпечує багаторазове використання коду.

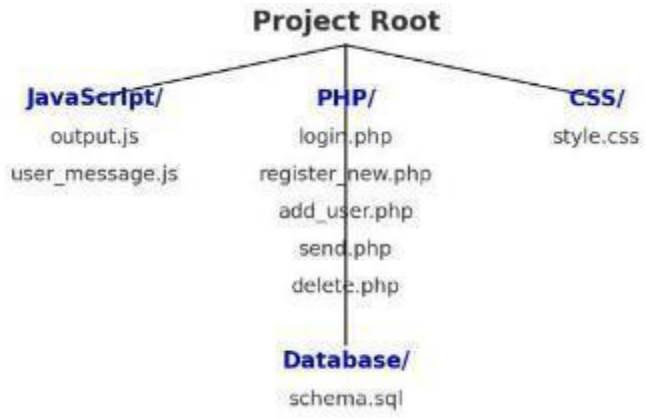

Рисунок 3.11 – Файлова структура проєкту

Для реалізації веб-додатку було застосовано функціональний підхід. Усі файли організовані за їх функціональними ролями:

– `login.php` – стартовий файл для авторизації користувачів.
– `register.php` – файл для реєстрації нових користувачів.
– `functions.php` – головний файл, що містить основні функції для роботи з базою даних та іншими компонентами.
– `send.php` – файл для відправки замовлень.
– `delete.php` – файл для видалення замовлень.
– `update.php` – відповідає за оновлення статусів користувачів та індикаторів замовлень.
– `logout.php` – забезпечує функціонал виходу з системи.
– Папка `javascript` – містить скрипти для інтерактивності сторінок, зокрема, `output.js` для роботи з відправкою, редагуванням та видаленням даних.
– Папка `css` – стилі для оформлення веб-сторінок.

Авторизація користувача є початковою точкою взаємодії із системою. Після запуску додатку користувач потрапляє на сторінку `login.php`, яка відображає форму входу. Код для авторизації виглядає наступним чином:

```
```php
function login_form() {
?>
<form action=«member.php» method=«post»>
<p>Ім’я користувача:</p><input type=«text» name=«username»>
<p>Пароль:</p><input type=«password» name=«password»>
```
```

```
<input type=«submit» value=«Увійти»>
</form>
<a href=«register.php»>Зареєструватися</a>
<?php
}
```

У разі, якщо користувач ще не зареєстрований, він перенаправляється на сторінку `register.php`. Тут виконується перевірка введених даних, а також їх збереження у базу даних.

Система підтримує функції відправки, редагування та видалення замовлень:

1. Відправка замовлень: реалізується через файл `send.php`. Користувач вводить деталі замовлення, які асинхронно надсилаються на сервер за допомогою технології Ajax.

2. Видалення замовлень: здійснюється у файлі `delete.php`. Якщо замовлення видалене обома сторонами, воно видаляється з бази даних.

3. Редагування замовлень: при натисканні кнопки «Редагувати» викликається подія, яка дозволяє змінити існуючий запис у полі введення.

Система оновлює статуси користувачів і замовлень у реальному часі. Файл `update.php` взаємодіє з базою даних, відображаючи актуальну інформацію про статуси. Наприклад, статус користувача змінюється на «онлайн», якщо він авторизувався.

Для забезпечення конфіденційності та безпеки після завершення роботи користувача із системою відбувається вихід із профілю. Файл `logout.php` завершує сесію, видаляючи всі дані, пов’язані з користувачем:

```php
<?php
require_once('functions.php');
session_start();
session_destroy();
echo «Ви успішно вийшли з системи.»;
?>
```

Використання технології Ajax дозволяє виконувати операції у фоновому режимі без перезавантаження сторінки. Це покращує взаємодію з користувачем, забезпечуючи швидкість і зручність роботи.

Отже, побудова чіткої файлової структури і впровадження функціонального підходу дозволяють створити ефективний, гнучкий та зручний веб-додаток для ресторану. Така організація файлів і модулів сприяє спрощенню розробки, тестування та подальшого розширення функціоналу системи 011, с. 78].

3.8. Тестування системи

Перевірка функціонування системи управління рестораном охоплює тестування окремих функціональних модулів та всієї системи в цілому. Для нашого веб-застосунку важливо провести комплексне тестування з урахуванням таких аспектів:

1. Тестування підключення до бази даних. Рекомендується перевірити стабільність підключення до бази даних та його здатність відновлюватися у разі збою. Доцільно оцінити коректність роботи функціоналу для взаємодії з базою даних, включаючи операції запису, оновлення та видалення даних.

2. Тестування технології Ajax. Необхідно провести перевірку роботи асинхронних запитів, які забезпечують динамічну взаємодію між клієнтом і сервером. Важливо впевнитися, що всі Ajax-запити обробляються швидко та без помилок навіть за умов підвищеного навантаження.

3. Перевірка роботи сесій користувачів. Слід переконатися у правильності створення, зберігання та видалення сесій. Сесії повинні забезпечувати збереження авторизаційних даних та стану користувача протягом усього сеансу роботи з системою.

4. Тестування роботи таблиць бази даних. Доцільно перевірити виконання операцій з даними, зокрема додавання нових користувачів, створення замовлень, редагування або видалення інформації. Рекомендується впевнитись у належному збереженні внесених змін у базі даних.

5. Перевірка кросплатформеності та адаптивності інтерфейсу. Варто оцінити роботу системи на різних пристроях, платформах і веб-браузерах. Особливу увагу слід приділити правильному відображенню інтерфейсу та адаптивності дизайну на екранах із різними розмірами.

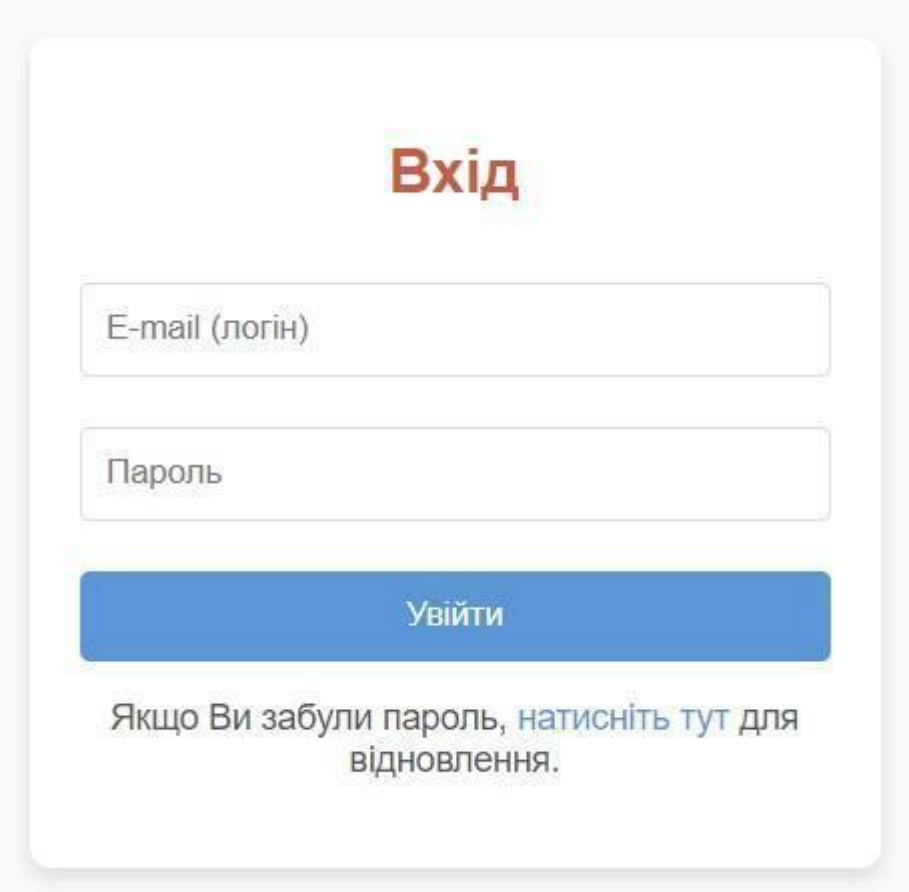

Рисунок 3.12 – Форма входу в адміністративну частину сайту

Тестування механізму бронювання столика охоплювало:

– Перевірку коректного підключення зовнішніх файлів CSS.
– Роботу бази даних для завантаження спеціальних пропозицій.
– Взаємодію форми бронювання з серверною логікою.
– Відповідність дизайну заданому макету.

Сторінка бронювання столика містить:

1. Головне меню:

– Посилання на розділи меню, бронювання та авторизації.
– Перевірено роботу всіх посилань, щоб переконатися, що вони ведуть на відповідні сторінки.

2. Секція спеціальних пропозицій:

– Динамічне завантаження даних з бази даних (MySQL) для відображення спеціальних пропозицій.
– Тестування включало перевірку з’єднання з базою даних, коректного отримання даних та їхнього відображення.

3. Форма бронювання столиків:

– Інтерактивна форма для введення дати, часу та кількості гостей.
– Перевірено коректність роботи форми, включаючи валідацію введених даних (наприклад, обмеження кількості гостей).

4. Дизайн та адаптивність:

– Код містить CSS для стилізації сторінки.
– Тестування включало перевірку відображення сторінки на різних пристроях, браузерах і розмірах екрана.

5. Функціональність футера:

– Інформація про авторські права у футері.
– Перевірено правильність відображення та функціональність футера.

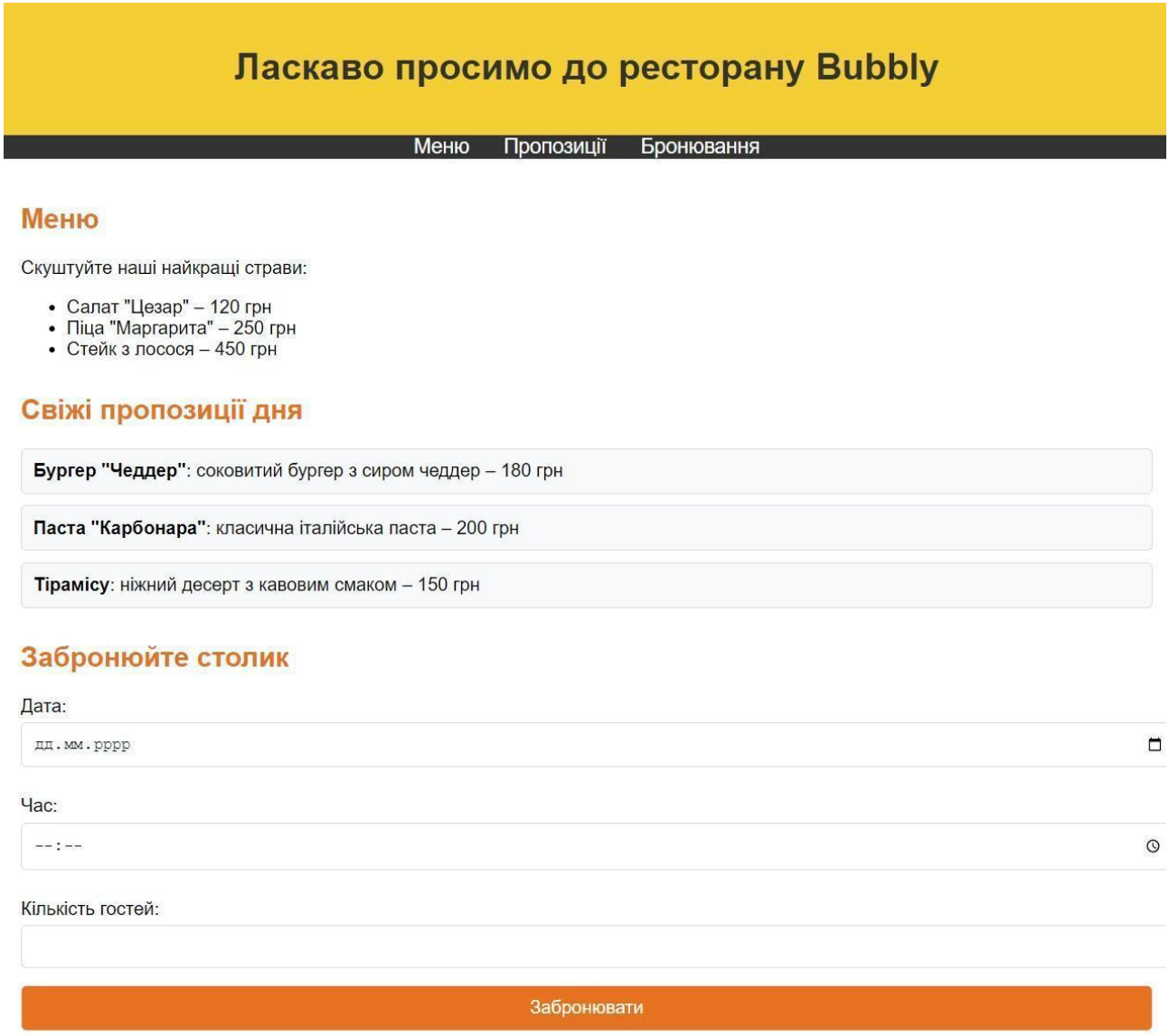

Рисунок 3.13 – Сторінка бронювання столика на сайті

Сторінка бронювання столика ресторану Bubbly містить інтерактивну форму, що дозволяє клієнтам швидко та зручно забронювати столик онлайн. У верхній частині сторінки розташовано навігаційне меню для переходу між основними розділами сайту: «Меню», «Пропозиції» та «Бронювання».

Основний контент сторінки включає такі блоки:

1. Меню: перелік найкращих страв із зазначенням назв і цін.
2. Свіжі пропозиції дня: спеціальні страви, що змінюються щодня, з коротким описом і цінами.

3. Форма бронювання: дозволяє вибрати дату, час і кількість гостей для бронювання. Для зручності користувачів реалізовано випадаючі списки та поля введення.

Сторінка виконана у сучасному стилі, з яскравим заголовком та акцентами на ключових елементах для полегшення взаємодії з користувачем.

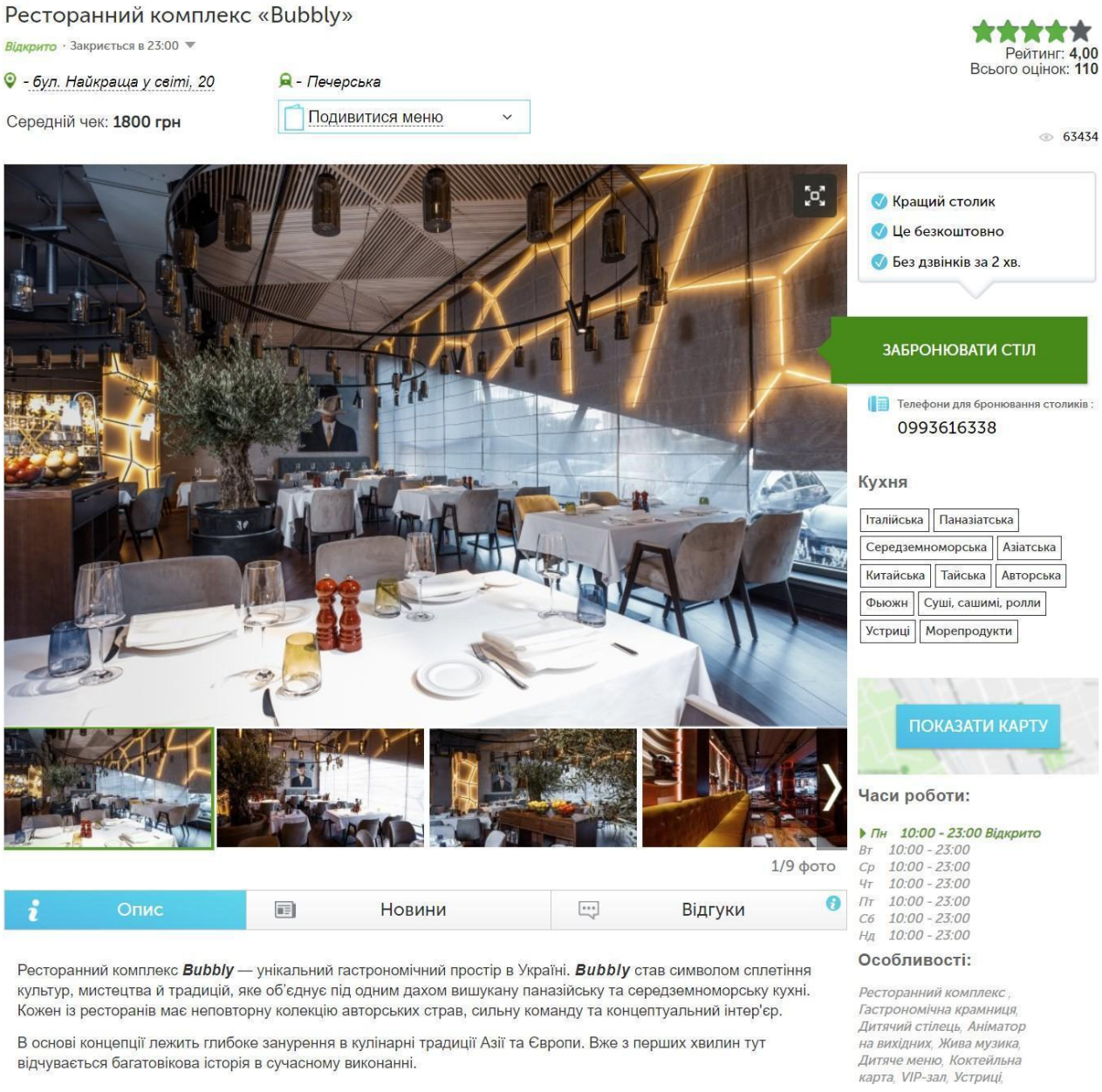

Рисунок 3.14 – Сторінка послуг сайту

На сторінці послуг сайту представлено ключові послуги ресторанного комплексу «Bubbly». Основний акцент зроблено на елегантному інтер’єрі закладу, зображення якого займає центральну частину сторінки. Праворуч розташовано функціональні блоки:

1. Блок бронювання столика: містить контактний телефон, кнопку для онлайн-бронювання та інформацію про зручності (швидкість бронювання, безкоштовність послуги тощо).
2. Кухні ресторану: відображаються різноманітні стилі кухні, що пропонує заклад, серед яких італійська, азійська, авторська тощо.
3. Графік роботи: наведено години роботи закладу для кожного дня тижня.

У нижній частині сторінки розміщено вкладки для переходу до опису, новин і відгуків. Опис акцентує увагу на унікальності концепції закладу та його кулінарних традиціях. Загалом сторінка відображає поєднання функціональності, зручності навігації та естетичної привабливості.

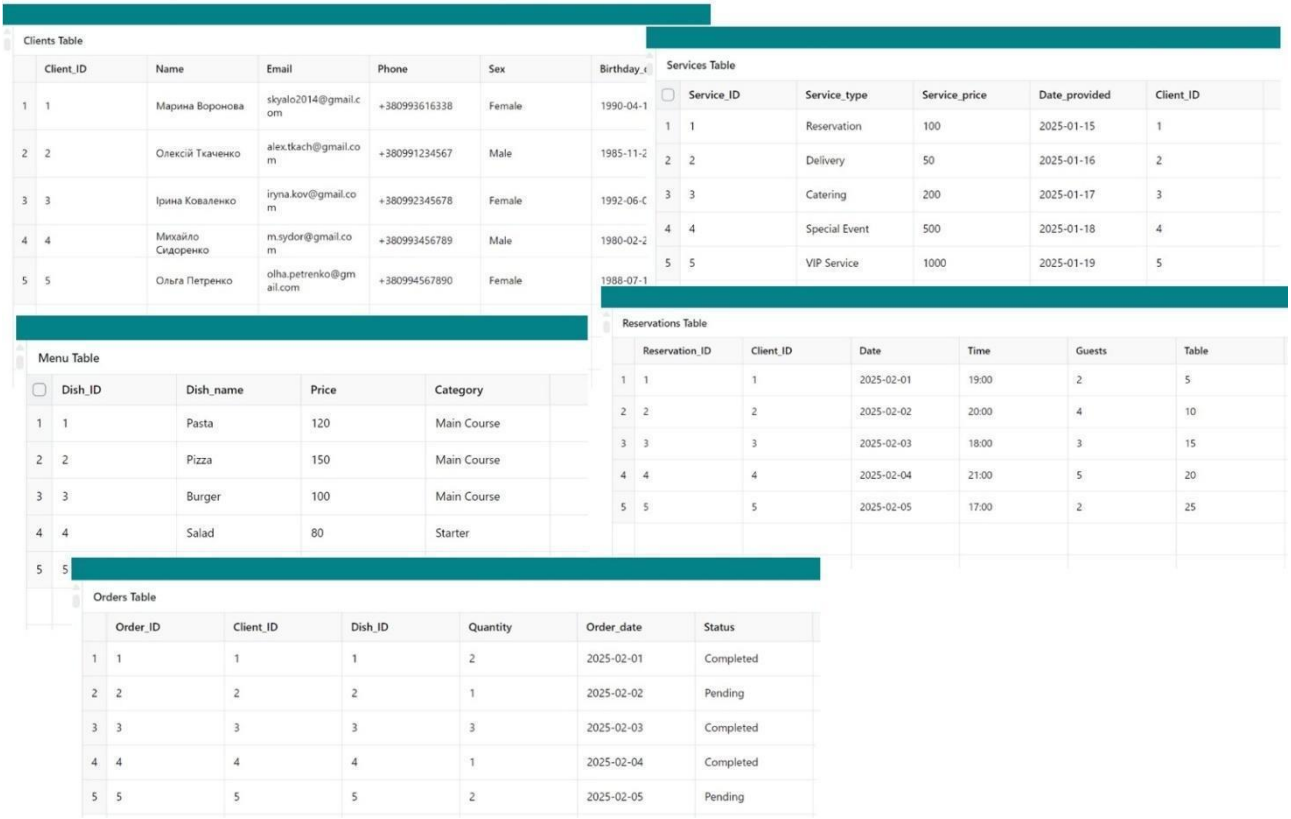

Рисунок 3.15 – Сторінки згенерованих таблиць бази даних

Згенеровані таблиці бази даних для ресторану «Bubbly» забезпечують функціонування системи обслуговування клієнтів. Їхній зміст наведено нижче:

1. Clients Table зберігає інформацію про клієнтів, включаючи імена, email-адреси, номери телефонів, стать, дату народження:

– Клієнт №1: Марина Воронова, skyalo2014@gmail.com, +380993616338.

2. Menu Table містить перелік страв із зазначенням назви, ціни та категорії:

– Паста (120 грн) – Основна страва.
– Салат (80 грн) – Закуски.

3. Services Table представляє послуги, які надаються клієнтам, включаючи тип послуги, ціну, дату надання та ідентифікатор клієнта:

– Резервація (100 грн), дата: 2025-01-15, клієнт: Марина Воронова.

4. Reservations Table відображає інформацію про бронювання столиків: клієнт, дата, час, кількість гостей і номер столика.

– Клієнт №1 забронював стіл №5 на 2025-02-01 о 19:00 для 2 гостей.

5. Orders Table зберігає замовлення клієнтів із зазначенням страви, кількості, дати замовлення та статусу виконання.

– Замовлення №1: Клієнт №1 замовив пасту (1 порція) 2025-02-01, статус – виконано.

Сформовані нами таблиці формують основу для управління клієнтами, бронюванням столиків, обробкою замовлень та наданням додаткових послуг у ресторані «Bubbly».

Результати проведеного тестування:

1. Тестування Ajax. Було відправлено тестові запити на сервер із передачами параметрів. Всі запити оброблялися швидко, без затримок навіть при значному навантаженні. Технологія Ajax забезпечує високу швидкість роботи та не створює надмірного навантаження на сервер.

2. Реєстрація та авторизація. Для кожного нового користувача створюється унікальна сесія. Під час тестування реєстрації користувачів усі сесії генерувалися без затримок, що демонструє надійність алгоритму автентифікації. Автоматичне присвоєння сесій підтверджує коректну інтеграцію з базою даних.

3. Стабільність роботи сесій. Перевірка показала, що сесії стабільно зберігають авторизаційні дані користувачів і забезпечують безперебійну роботу під час переходу між сторінками.

4. Функціонування бази даних. Операції з таблицями, такими як створення, оновлення, перегляд та видалення даних, виконувалися коректно. Жодних втрат або некоректного відображення інформації не було зафіксовано.

Отже, проведене тестування підтвердило стабільність роботи системи управління рестораном. Усі основні модулі, включаючи реєстрацію, авторизацію, динамічну взаємодію через Ajax, а також управління даними у базі, функціонують без збоїв. Це створює передумови для подальшого впровадження проєкту та його ефективного використання в умовах реальної експлуатації.

3.9. Заходи щодо забезпечення безпеки використання застосунку

Формулювання апаратних рішень для забезпечення коректного функціонування системи управління рестораном має базуватися на програмних і апаратних характеристиках, необхідних для надійної роботи веб-застосунку. Основний акцент робиться на налаштуваннях веб-сервера, який є ключовою складовою роботи системи.

Основні вимоги до апаратного та програмного забезпечення:

1. Операційна система: Найчастіше використовуються серверні ОС сімейства Unix, наприклад, Ubuntu або FreeBSD. У нашому випадку доцільно обрати FreeBSD версії 9, яка забезпечує стабільність і безпеку роботи серверного середовища.

2. Веб-сервер: Для роботи веб-застосунку використовується Apache версії 2.2.14, який підтримує інтеграцію з мовою програмування PHP та сервером баз даних MySQL. Apache забезпечує обробку HTTP-запитів і відповідає за передачу даних між користувачем і сервером.

3. Інтерпретатор PHP: Застосунок розроблений мовою PHP, тому сервер повинен містити інтерпретатор PHP версії 5.3.27, який дозволяє виконувати серверний код.

4. Сервер баз даних: Для збереження даних використовується MySQL версії 5.2.9, який забезпечує швидкий доступ до інформації про замовлення, користувачів та інші дані.

5. Апаратні характеристики веб-сервера:

- Процесор: Intel Xeon Quad-Core E6–1225 v2 (6.2 ГГц).
- Оперативна пам’ять: 8 ГБ DDR5 ECC із частотою 8200 МГц.
- Жорсткий диск: 2×500 ГБ SATA II RAID Edition для забезпечення швидкого доступу до даних.
- Материнська плата: SuperMicro X7DAE (RTL) Dual LGA771.
- Блок живлення: Seasonic SS – 800TFX 800W для стабільної роботи.

6. Швидкість передачі даних: Сервер повинен підтримувати пропускну здатність 1000 Мбіт/с, щоб забезпечити швидке завантаження сторінок навіть при значному навантаженні.

Забезпечення надійності та безпеки:

6. Синхронізація роботи бази даних: Для забезпечення актуальності даних і зменшення затримок при обробці запитів.
7. Сесії користувачів: Забезпечують авторизований доступ до профілів і дозволяють відстежувати активність користувачів на платформі.

Веб-застосунок для управління рестораном може функціонувати ефективно за умови правильної конфігурації серверного середовища. Веб-сервер повинен бути оптимально налаштований для обробки запитів користувачів, зберігання даних у базі MySQL і підтримки інтерактивних функцій через PHP та Ajax. Виконання цих вимог гарантує швидке завантаження сторінок і безперебійну роботу навіть за умов великої кількості одночасних користувачів. Використання сучасних технологій та апаратних компонентів створює основу для безпечної, стабільної та високопродуктивної роботи системи.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ

Дія електричного струму на живу тканину на відміну від дії інших матеріальних факторів (пари, небезпеки ураження струмом, випромінювання тощо) носить своєрідний і різнобічний характер. Проходячи через організм людини, електричний струм справляє термічне, електричне та механічне (динамічне) дії, які є звичайними фізико-хімічними процесами, притаманними як живий, так і неживої матерії. Одночасно електричний струм виробляє і біологічну дію, яке є специфічним процесом, властивим лише живій тканині.

Термічна дія струму проявляється в опіках окремих ділянок тіла, нагріванні до високої температури кровоносних судин, нервів, серця, мозку та інших органів, що знаходяться на шляху струму, що викликає в них серйозні функціональні розлади.

Електролітична дія струму виражається в розкладанні органічної рідини, в тому числі і крові, що супроводжується значними порушеннями їх фізико-хімічного складу.

Електроустановки і струмопроводи промислової частоти є основними джерелами електромагнітних полів промислового діапазону. Інтенсивність полів залежить від величини протікаючого струму. Згідно з останніми даними, деякі пристрої є потенційними джерелами ризику від електромагнітних полів промислової частоти, такі як електроплити, електрогрилі, праски та холодильники з робочим компресором. Також мікрохвильові печі, телевізори будь-яких модифікацій та радіотелефони є джерелами підвищеної небезпеки від електромагнітного випромінювання.

Електромагнітне поле є особливою формою матерії. Будь-яка заряджена частка оточена своїм електромагнітним полем, яке взаємодіє з нею. Проте електромагнітне поле може існувати і відокремлено від заряджених часток, у формі фотонного випромінювання, що рухається з великою швидкістю, або у формі електромагнітних хвиль. Біологічний вплив електромагнітних хвиль залежить від їх частоти, інтенсивності, тривалості та умов опромінювання.

Виділяють термічний (тепловий) вплив, а також функціональні та морфологічні зміни, що виникають.

Електромагнітні хвилі (ЕМХ) можуть спричиняти нагрівання тканин і органів людини, що веде до їх змін і пошкоджень. Теплова дія ЕМХ проявляється у загальному підвищенні температури тіла або місцевому нагріванні тканин. Органи зі слабкою терморегуляцією, такі як мозок, очі, органи кишкового та сечостатевого тракту, особливо вразливі до нагрівання. ЕМХ з довжиною хвилі 1–20 см може шкодити очам, спричиняючи катаракту (помутніння кришталика) і втрату зору.

Морфологічні зміни включають зміни в будові та зовнішньому вигляді тканин і органів людини, такі як опіки, некрози, крововиливи та зміни в структурі клітин. Ці зміни спостерігаються в периферичних тканинах, центральній нервовій системі та серцево-судинній системі, що призводить до порушень регуляторних функцій, нервових зв’язків і змін у самій структурі клітин. Внаслідок цього можуть виникати знижений кров’яний тиск (гіпотонія), сповільнення серцевого ритму (брадикардія) та інші порушення.

Функціональні зміни, спричинені електромагнітним полем (ЕМП), можуть виявлятись у вигляді головного болю, порушень сну, підвищеної втомлюваності, роздратованості, пітливості, випадання волосся, болів у серцевій області та зниження статевої потенції. Оцінка впливу ЕМП на організм людини здійснюється шляхом вимірювання поглиненої тілом електромагнітної енергії, яка виражається величиною W (ват) або питомою енергією, яку поглинає організм – Wп (ват на кілограм маси).

Наприклад, для оцінки можливого впливу електромагнітного поля від радіотелефонів визначають потужність електромагнітних полів, що поглинаються одним кілограмом мозку – параметр SAR (специфічна поглинальна швидкість). Основні моделі телефонів мають значення SAR 0,2 Вт/кг і нижче. Основні заходи для захисту від ЕМП включають обмеження тривалості впливу, віддалення від джерела випромінювання, екранування джерел випромінювання, зменшення випромінювання на самому джерелі,

визначення зон випромінювання, екранування робочих місць та використання засобів індивідуального захисту.

Електротравматизм – це явище, яке характеризується сукупністю електротравм, які виникають та повторюються в аналогічних виробничих, побутових умовах та ситуаціях. Осередок, джерело електротравматизму чи інша тимчасова або навіть постійна ситуація при експлуатації електроустановок, коли мають місце аналогічні випадки ураження людини струмом. Проходячи через тіло людини, електричний струм справляє термічну, електричну та механічну (динамічну) дію. Ці фізико-хімічні процеси притаманні живій та неживій матерії. Одночасно електричний струм здійснює і біологічну дію, яка є специфічним процесом, властивим лише живій тканин. Різноманітність впливу електричного струму на організм людини призводить до електротравм, які умовно поділяються на два види:

– місцеві електротравми, які означають місцеве ушкодження організму;
– загальні електротравми електричні удари), коли уражається (або виникає загроза ураження) весь організм внаслідок порушення нормальної діяльності життєво важливих органів та систем.

Зі статистичними даними орієнтовний розподіл нещасних випадків внаслідок дії електричного струму у промисловості за вказаними видами травм має такий вигляд: місцеві електротравми – 20%; електричні удари – 25%; змішані травми, (одночасно місцеві електричні травми) – 55%.

Види електротравм. Електричні опіки – Залежно від умов виникнення опіки поділяються на контактні, дугові і змішані. Контактні струмові опіки більш вірогідні в установках порівняно невеликої напруги – 1... 2 кВ і спричиняються тепловою дією струму. Для місць контакту тіла зі струмопровідними неізольованими елементами електроустановки характерним є велика щільність струму і підвищений опір – за рахунок опору шкіри. Тому в місцях контакту виділяється значна кількість тепла, що і призводить до опіку. Контактні опіки охоплюють прилеглі до місця контакту ділянки шкіри і тканин.

Електричні знаки (знаки струму або електричні мітки) спостерігаються у вигляді різко окреслених плям сірого чи блідо-жовтого кольору на поверхні тіла людини в місці контакту зі струмовідними елементами. Зазвичай знаки мають круглу чи овальну форму, або форму струмовідного елемента, до якого доторкнулася людина, розмірами до 10мм з поглибленням у центрі. Іноді електричні знаки можуть мати форму мікроблискавки, яка контрастно спостерігається на поверхні тіла.

Металізація шкіри – це проникнення у верхні шари шкіри дрібних часток металу, який розплавився під дією електричної дуги на дрібні частки металлу, які мають високу температуру, але малий запас теплоти. Тому вони нездатні проникати через одяг і небезпечні для відкритих ділянок тіла. На ураженій ділянці тіла при цьому відчувається біль від опіку за рахунок тепла, занесеного в шкіру металом, і напруження шкіри від присутності в ній сторонньої твердої речовини (часток металу).

Електроофтальмія – запалення зовнішніх оболонок очей, спричинене надмірною дією ультрафіолетового випромінювання електричної дуги. Електроофтальмія зазвичай розвивається через 2–6 годин після опромінення (залежно від інтенсивності опромінення) і проявляється у формі почервоніння і запалення шкіри та слизових оболонок повік, сльозоточінні, гнійних виділеннях, світлоболях і світлобоязні. Тривалість захворювання 3... 5 днів.

Механічні ушкодження, пов’язані з дією електричного струму на організм людини, спричиняються непередбачуваним судомним скороченням м’язів у результаті подразнювальної дії струму. Внаслідок таких судомних скорочень м’язів можливі розриви сухожиль, шкіри, кровоносних судин, нервових тканин, вивихи суглобів, переломи кісток.

Під час виконання маніпуляцій з участю апарату штучного кровообігу (у перев’язувальній, операційній, палаті тощо), стіл накривають шерстяною ковдрою, 3–4 шарами прогумованої тканини, а також простирадлом таких розмірів, щоб краї його звисали з усіх кінців стола. За наявності у зазначених випадках підлоги з плитки місце персоналу має бути покрите ізолювальним

матеріалом площею не менше 1 м². Забороняється працювати на апаратах у разі їх несправності, вимкненого електричного блокування дверей чи наявності вм’ятин на корпусі апарата.

Надзвичайно велике значення має освітлення операційного поля. До нього ставляться такі вимоги: достатня освітленість, мінімальний засліплювальний вплив джерел світла і блискучість, рівномірність освітлення, відсутність різних і глибоких тіней, відсутність нагрівання. Для освітлення операційної використовують природне (денне) й штучне освітлення. Денне світло найбільш рівномірно розподіляється в операційній над операційним столом. У разі додаткового освітлення бічних поверхонь операційного поля використовують пересувні та переносні лампи. Освітлення дублюється також аварійною мережею з живленням від акумуляторів з автоматичним вмиканням.

Особливості мікроклімату операційної полягають у тому, що під час операції підвищуються температура та вологість повітря, відбувається забруднення його наркотичними газами та бактеріальною флорою, накопичується статична електрика. Усе це несприятливо впливає на здоров’я хворих і персоналу, особливо за наявності повітряно-крапельної інфекції, яка може спричинити гнійно-запальні ускладнення. Температура в операційній має бути в межах 22–23 °С. Іноді, наприклад при опіках, тривалому оперуванні на відкритій черевній порожнині, необхідно дотримуватись температурного рівня 25–30 °С. Опалення здійснюється електроплитами, вмонтованими в нижню частину стіни, з автоматичним регулюванням постійної температури.

Так загальні вимоги охороні праці до роботи з біологічним матеріалом передбачають обов’язкове виконання персоналом наступних дій:

1. До самостійної роботи, при якій можливий контакт з кров’ю та іншими біологічними рідинами пацієнтів, допускаються особи старше 18 років, які не мають медичних протипоказань, навчені безпечним методам роботи, що пройшли вступний та первинний на робочому місці інструктажі з охорони праці, стажування та перевірку знань вимог охорони праці.

2. При роботі персоналу слід керуватися принципом, що всі пацієнти потенційно інфіковані.
3. При виконанні робіт з кров'ю та іншими біологічними рідинами пацієнтів можливі механічні пошкодження шкіри: – колоті рани при необережному поводженні зі шприцами та іншими колючими інструментами (предметами); – порізи кистей рук (при відкриванні пляшок, флаконів, пробірок з кров'ю або сироваткою; при роботі з контамінованих ВІЛ-інструментами).
4. Персонал повинен виконувати роботу в технологічному одязі, передбаченої галузевими нормами: халат бавовняний, медична шапочка, медичні рукавички, надіті поверх рукавів медичного халата.
5. Для проведення інвазійних процедур рекомендовано надягати дві пари рукавичок, водонепроникний халат і фартух.
6. При загрозі розбризкування крові та інших біологічних рідин роботи слід виконувати в масках, захисних окулярах, при необхідності, використовувати захисні екрани, клейончасті фартухи.

При аварії під час роботи з інфекційним матеріалом (биття посуду, розприскування зі шприцу або піпетки, або при зараженні (розтині) тварин, а також в усіх випадках, що ведуть до забруднення заразним матеріалом навколишніх предметів, одягу або відкритих частин тіла працівників), персонал, який при цьому присутній, зобов'язаний негайно провести знезараження приміщення, обладнання і предметів, що могли бути інфіковані, а також провести самознезараження [2].

ВИСНОВКИ

1. У процесі проєктування та розробки кваліфікаційної роботи були засвоєні основи та принципи створення веб-застосунків для автоматизації виробничих процесів у галузі масового харчування. Використання системного аналізу та декомпозиції задач дозволило ефективно структурувати проєкт і розробити функціонал, який відповідає потребам користувачів. Інтеграція сучасних технологій, таких як HTML5, PHP, JavaScript, Ajax та фреймворк jQuery, забезпечила створення динамічного веб-застосунку, що відображає актуальні тенденції у веб-розробці.

2. У рамках виконання кваліфікаційної роботи було реалізовано:

- Інтеграцію Ajax-технології у веб-ресурс для забезпечення асинхронного обміну даними між клієнтом і сервером, що дозволило зменшити час завантаження сторінок і покращити загальну взаємодію користувача з системою.
- Розробку функціональних частин веб-застосунку, включно з модулями реєстрації, авторизації, управління сесіями, обробки заявок та управління даними користувачів.
- Освоєння методів декомпозиції складних завдань, що дало змогу розподілити програмний код на окремі модулі для забезпечення його масштабованості та повторного використання.
- Роботу з реляційною базою даних MySQL, включаючи проєктування структури таблиць, створення запитів для додавання, видалення та редагування даних, а також забезпечення цілісності даних.
- Забезпечення кросбраузерної сумісності веб-ресурсу, що гарантує коректне відображення та роботу застосунку на різних пристроях і в різних веб-браузерах.

3. У результаті розроблено повнофункціональний веб-застосунок, що виконує функції підтримки виробничих процесів у сфері масового харчування. Основними складовими веб-застосунку є:

- Модуль заявок, який дозволяє користувачам взаємодіяти з системою, створювати та обробляти заявки в асинхронному режимі.
- Система управління контентом, що включає модулі новин, завдань і функції взаємодії між користувачами.
- База даних MySQL, яка використовується для зберігання інформації про користувачів, заявки та інші дані, забезпечуючи швидкий доступ і надійність.

4. В процесі розробки була врахована важливість апаратного забезпечення. Було проаналізовано необхідні апаратні та програмні ресурси для роботи веб-застосунку, включаючи:

- Операційну систему FreeBSD для роботи веб-сервера.
- Веб-сервер Apache версії 2.2.14 для обробки HTTP-запитів.
- PHP-інтерпретатор і сервер бази даних MySQL для підтримки серверного функціоналу.

5. Крім розробки основного функціоналу, були впроваджені механізми тестування працездатності системи, зокрема:

- Тестування коректності з’єднання з базою даних.
- Перевірка роботи Ajax-технології для асинхронного обміну даними.
- Тестування роботи модулів управління сесіями, авторизації та реєстрації.

6. Проєкт забезпечив практичні навички з програмування динамічних веб-функцій за допомогою PHP та JavaScript, інтеграції баз даних, розробки кросбраузерних інтерфейсів та використання сучасних технологій веб-розробки. Здобутий досвід може бути застосований для розробки подібних систем в інших галузях.

Таким чином, розроблений веб-застосунок відповідає поставленим вимогам, забезпечує стабільну роботу та може бути легко адаптований до інших сфер. Результати проєкту підтвердили ефективність застосованих методів і технологій, а також створили платформу для подальшого вдосконалення системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Аксьонов С. PHP у створенні інтернет-ресурсів. Україна, Київ: Ґенеза, 2014. 322 с.
2. Барі Борд. Програмування на HTML для початківців, 2-е видання.Україна, Київ: Сісман, 2012. 314 с.
3. Веб-програмування. Вікіпедія. URL: http: // uk. wikipedia. org / wiki / Веб-програмування (дата звернення: 05.01.2025).
4. Веб-сайт «Хуторець на Дніпрі». URL: https: // khutoretsnadnipri. choiceqr.com (дата звернення: 05.01.2025).
5. Вінер А. Бази даних. Київ: Наука, 2018. 422 с.
6. Гусев, Д. Н. Особливості та завдання створення інформаційної системи / Д. Н. Гусев. Міжнародні операції. Україна, Київ 2018. № 5. С. 70–84.
7. Інтернет. Вікіпедія. URL: http: // uk. wikipedia.org / wiki / Інтернет (дата звернення: 05.01.2025).
8. Кей С. Хорстман. HTML 4. Вступний курс. Україна, Київ: Ґенеза, 2013. ISBN 978–5–8459–1900–7.
9. Кей С. Хорстманн, Корнел Г. HTML–5. Бібліотека професіонала, том 1. Основи. 12-е видання.Україна, Київ: Ґенеза, 2021. 512 с.
10. Кравчук Г. Т. Інформаційні системи і технології в ресторанній сфері: [Підручник для студентів вищих навчальних закладів] / Кравчук Г. Т., Шевчук Т. В., Коновал У. М. Україна, Львів: Львівський банківський інститут НБУ, 2002. 135 с.
11. Посібник для програміста WEB. Вільямс. Україна, Київ: Каліпсо, 2014. 433 с.
12. Савчук Т. О. Організація баз даних і знань. Україна, Вінниця: ВДТУ, 2017. 312 с.
13. Степанов Ю. Л. Розробка додатків баз даних для СУБД з використанням Ajax. 212 с. Україна, Вінниця: ВДТУ, 2019

14. Стрічка сайту «Чотири Чебуреки Prosecco Bar». URL: https: // center. 4che. bar (дата звернення: 05.01.2025).
15. Теорія систем і системний аналіз в управлінні організаціями: Довідник / Під ред. В. Н. Волкової. Україна, Київ: Фінанси і статистика, 2018. 1024 с.
16. Фред Лонг та ін. Посібник для програміста WEB. Україна, Київ: Каліпсо, 2014. 712 с.
17. Arachnophilia. URL: http://www.arachnoid.com/arachnophilia (дата звернення: 05.01.2025).
18. Beginning HTML. HTML Development by Giulio Zambon. Brazil, 2019. 512 p.
19. Bruce Eckel. Thinking in HTML (4th Edition). USA, Prentice Hall PTR, 2016. 512 p.
20. Bruno Emaus (University of Amsterdam). (2017) Guidelines for primary school teachers for integration of ICT in their lessons. URL: http: // www. ecolenet. nl / projects / guidelines_primary. html (дата звернення: 05.01.2025).
21. HTML, 2nd Edition by Bryan Basham, Kathy Sierra, Bert Bates. USA, O'Reilly Media 2018. 912 p.
22. HTML. Вікіпедія. URL: http: // uk. wikipedia.org / wiki / HTML (дата звернення: 05.01.2025).
23. HTML. Вікіпедія. URL: http: // uk. wikipedia.org / wiki / HTML (дата звернення: 05.01.2025).
24. James Gosling; Bill Joy, Guy Steele, Gilad Bracha, USA, O'Reilly Media 2021. 422 p.
25. Murach's Java Servlets and HTML (2nd Edition) by J. Murach, A. Stelmann. 729 p. Mike Murach & Associates Inc, USA, 2008 p.
26. Technology in Schools: What the Research Says. Metiri Group – Commissioned by Cisco Systems. 1992–2006. 18 p. URL: http: // www. cisco.com / web / strategy / docs / education / TechnologyinSchoolsReport. pdf (дата звернення: 05.01.2025).
27. TextPad. URL: www. textrad.com (дата звернення: 05.01.2025).

28. The HTML Language Specification, Third Edition. Addison-Wesley, 2014. 245 p.
29. The Lewisham Education and Culture Web. URL: http: // ess. lewisham. gov. uk / talent / pricor / module4. html (дата звернення: 05.01.2025).
30. Web Programming with Eclipse by David A Turner, Jinseok Chae Joshua Bloch. Effective HTML (3nd Edition). Prentice Hall PTR, USA, 2012. 148 p.

ДОДАТКИ

Додаток А

Лістинг сторінки бронювання столика у веб-застосунку

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
     scale=1.0">
    <meta name="description" content="Bubbly – Перегляньте меню,
     забронюйте столик або зробіть замовлення онлайн.">
    <meta name="keywords" content="ресторан, їжа, бронювання,
     онлайн-замовлення, Bubbly">
    <meta name="author" content="Команда Bubbly">
    <title>Bubbly – Головна</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        header {
            background-color: #ffcc00;
            color: #333;
            padding: 20px;
            text-align: center;
        }

        nav ul {
            list-style-type: none;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            background-color: #333;
        }

        nav ul li {
            margin: 0 15px;
        }

        nav ul li a {
            color: white;
            text-decoration: none;
            font-size: 18px;
        }
```

```
nav ul li a:hover {
    text-decoration: underline;
}

main {
    padding: 20px;
}

section {
    margin-bottom: 30px;
}

h2 {
    color: #ff6600;
}

.special-offers ul {
    list-style-type: none;
    padding: 0;
}

.special-offers ul li {
    background-color: #f9f9f9;
    margin: 10px 0;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
}

form label {
    display: block;
    margin: 10px 0 5px;
}

form input, form button {
    width: 100%;
    padding: 10px;
    margin-bottom: 15px;
    border: 1px solid #ddd;
    border-radius: 5px;
}

form button {
    background-color: #ff6600;
    color: white;
    font-size: 16px;
    cursor: pointer;
}

form button:hover {
    background-color: #e65c00;
}
```

```
        footer {
            background-color: #333;
            color: white;
            text-align: center;
            padding: 10px 0;
            position: fixed;
            width: 100%;
            bottom: 0;
        }
    </style>
</head>
<body>
    <header>
        <h1>Ласкаво просимо до ресторану Bubbly</h1>
    </header>

    <nav>
        <ul>
            <li><a href="#menu">Меню</a></li>
            <li><a href="#specials">Пропозиції</a></li>
            <li><a href="#reservation">Бронювання</a></li>
        </ul>
    </nav>

    <main>
        <section id="menu">
            <h2>Меню</h2>
            <p>Скуштуйте наші найкращі страви:</p>
            <ul>
                <li>Салат "Цезар" - 120 грн</li>
                <li>Піца "Маргарита" - 250 грн</li>
                <li>Стейк з лосося - 450 грн</li>
            </ul>
        </section>

        <section id="specials" class="special-offers">
            <h2>Свіжі пропозиції дня</h2>
            <ul>
                <li><strong>Бургер "Чеддер"</strong>: соковитий
    бургер з сиром чеддер - 180 грн</li>
                <li><strong>Паста "Карбонара"</strong>: класична
    італійська паста - 200 грн</li>
                <li><strong>Тірамісу</strong>: ніжний десерт з
    кавовим смаком - 150 грн</li>
            </ul>
        </section>

        <section id="reservation">
            <h2>Забронюйте столик</h2>
            <form action="reserve.php" method="POST">
                <label for="date">Дата:</label>
                <input type="date" id="date" name="date" required>
```

```
                <label for="time">Час:</label>
                <input type="time" id="time" name="time" required>

                <label for="guests">Кількість гостей:</label>
                <input type="number" id="guests" name="guests"
     min="1" max="20" required>

                <button type="submit">Забронювати</button>
            </form>
        </section>
    </main>

    <footer>
        <p>&copy; 2025 Ресторан Bubbly. Усі права захищено.</p>
    </footer>
</body>
</html>
```

Додаток Б

Лістинг програмних модулів системи

Програмний код створення таблиць бази даних:

```
CREATE DATABASE bubbly;
USE bubbly;

-- Таблиця Clients
CREATE TABLE `clients` (
    `Client_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Name` VARCHAR(50) NOT NULL,
    `Surname` VARCHAR(50) NOT NULL,
    `Birthday_date` DATE NOT NULL,
    `Document_type` INT(5) NOT NULL,
    `Document_number` VARCHAR(50) NOT NULL,
    `Date_arrive` DATE,
    `Date_depart` DATE,
    `Sex` CHAR(1),
    `Reserve_ID` INT(5),
    `Email` VARCHAR(100),
    `Phone` VARCHAR(15),
    `Address` TEXT,
    PRIMARY KEY (`Client_ID`),
    FOREIGN KEY (`Document_type`) REFERENCES `document_types`
     (`Document_type_ID`),
    FOREIGN KEY (`Reserve_ID`) REFERENCES `rooms` (`Room_ID`)
) ENGINE=InnoDB;

-- Таблиця Document_types
CREATE TABLE `document_types` (
    `Document_type_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Document_type_name` VARCHAR(50) NOT NULL,
    `Description` TEXT,
    PRIMARY KEY (`Document_type_ID`)
) ENGINE=InnoDB;

-- Таблиця Bron_details
CREATE TABLE `bron_details` (
    `Bron_ID` INT(5) NOT NULL,
    `Room_ID` INT(5) NOT NULL,
    `Price_per_night` DECIMAL(10, 2) NOT NULL,
    `Total_price` DECIMAL(10, 2) NOT NULL,
    `Number_of_nights` INT(3) NOT NULL,
    PRIMARY KEY (`Bron_ID`, `Room_ID`),
    FOREIGN KEY (`Bron_ID`) REFERENCES `brons` (`Bron_ID`),
    FOREIGN KEY (`Room_ID`) REFERENCES `rooms` (`Room_ID`)
) ENGINE=InnoDB;

-- Таблиця Rooms
CREATE TABLE `rooms` (
```

```
    `Room_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Reserve_ID` INT(5),
    `Number` VARCHAR(10) NOT NULL,
    `Phone` VARCHAR(15),
    `Price` DECIMAL(10, 2) NOT NULL,
    `Category_ID` INT(5),
    `Description` TEXT,
    `Max_occupancy` INT(2),
    PRIMARY KEY (`Room_ID`),
    FOREIGN KEY (`Category_ID`) REFERENCES `category`
     (`Category_ID`)
) ENGINE=InnoDB;

-- Таблиця Category
CREATE TABLE `category` (
    `Category_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Category_name` VARCHAR(50) NOT NULL,
    `Description` TEXT,
    PRIMARY KEY (`Category_ID`)
) ENGINE=InnoDB;

-- Таблиця Service
CREATE TABLE `service` (
    `Service_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Client_ID` INT(5) NOT NULL,
    `Service_type_ID` INT(5) NOT NULL,
    `Count_time` INT(5),
    `Date` DATE,
    `Date_when` DATE,
    `Pay` DECIMAL(10, 2),
    `Total_cost` DECIMAL(10, 2),
    PRIMARY KEY (`Service_ID`),
    FOREIGN KEY (`Client_ID`) REFERENCES `clients` (`Client_ID`),
    FOREIGN KEY (`Service_type_ID`) REFERENCES `service_types`
     (`Service_type_ID`)
) ENGINE=InnoDB;

-- Таблиця Service_types
CREATE TABLE `service_types` (
    `Service_type_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Service_type_name` VARCHAR(50) NOT NULL,
    `Service_price` DECIMAL(10, 2),
    `Time_count` INT(5),
    `Description` TEXT,
    PRIMARY KEY (`Service_type_ID`)
) ENGINE=InnoDB;

-- Таблиця Notes
CREATE TABLE `notes` (
    `Note_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Client_ID` INT(5) NOT NULL,
    `Note` TEXT,
    `Date` DATETIME DEFAULT CURRENT_TIMESTAMP,
```

```
    PRIMARY KEY (`Note_ID`),
    FOREIGN KEY (`Client_ID`) REFERENCES `clients` (`Client_ID`)
) ENGINE=InnoDB;

-- Таблиця Message
CREATE TABLE `message` (
    `Message_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Client_ID` INT(5) NOT NULL,
    `Note` TEXT,
    `Use` TINYINT(1) DEFAULT 0,
    `Message_date` DATETIME DEFAULT CURRENT_TIMESTAMP,
    `Sender` VARCHAR(100),
    `Receiver` VARCHAR(100),
    PRIMARY KEY (`Message_ID`),
    FOREIGN KEY (`Client_ID`) REFERENCES `clients` (`Client_ID`)
) ENGINE=InnoDB;

-- Таблиця State_br
CREATE TABLE `state_br` (
    `State_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `State_name` VARCHAR(50) NOT NULL,
    `Description` TEXT,
    PRIMARY KEY (`State_ID`)
) ENGINE=InnoDB;

-- Таблиця Brons
CREATE TABLE `brons` (
    `Bron_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `State` INT(5) NOT NULL,
    `From_date` DATE NOT NULL,
    `To_date` DATE NOT NULL,
    `Description` TEXT,
    `Bron_date` DATE NOT NULL,
    PRIMARY KEY (`Bron_ID`),
    FOREIGN KEY (`State`) REFERENCES `state_br` (`State_ID`)
) ENGINE=InnoDB;

-- Таблиця Account_room
CREATE TABLE `account_room` (
    `Account_ID` INT(5) NOT NULL AUTO_INCREMENT,
    `Client_ID` INT(5) NOT NULL,
    `Price` DECIMAL(10, 2) NOT NULL,
    `Date` DATE NOT NULL,
    `Pay` DECIMAL(10, 2),
    `Total_balance` DECIMAL(10, 2),
    PRIMARY KEY (`Account_ID`),
    FOREIGN KEY (`Client_ID`) REFERENCES `clients` (`Client_ID`)
) ENGINE=InnoDB;```
```

Код файлу `add_user.php`:

```
```php
<?php
```
```

```
require_once('functions.php');

$name = trim($_POST['name']);
$password = trim($_POST['password']);

if (empty($name) || empty($password)) {
die («Не введені обов’язкові поля.»);
}

if (strlen($name) > 40) {
die («Ім’я користувача занадто довге. Максимум 40 символів.»);
}

if (strlen($password) < 6) {
die («Пароль занадто короткий. Мінімум 6 символів.»);
}

// Хешування паролю
$hashedPassword = password_hash($password, PASSWORD_BCRYPT);

registerUser($name, $hashedPassword);
echo «Користувача успішно зареєстровано!»;
?>
```

Файл `functions.php`:

```
```php
<?php
function connectDatabase() {
$host = 'localhost';
$username = 'your_username';
$password = 'your_password';
$database = 'bubbly';

$conn = new mysqli($host, $username, $password, $database);

if ($conn->connect_error) {
die («Помилка підключення до бази даних:». $conn->connect_error);
}
return $conn;
}

function registerUser($username, $password) {
$conn = connectDatabase();

$stmt = $conn->prepare («INSERT INTO users (username, password)
     VALUES (?,?)»);
$stmt->bind_param («ss», $username, $password);

if ($stmt->execute()) {
echo «Користувач зареєстрований успішно!»;
} else {
```
```

```
echo «Помилка:». $stmt->error;
}

$stmt->close();
$conn->close();
}
?>
```

Файл `send_message.php`:

```php
<?php
require_once('functions.php');

$to = intval($_POST['to']);
$from = intval($_POST['from']);
$text = trim($_POST['text']);

if (empty($text)) {
die («Текст повідомлення не може бути порожнім.»);
}

$conn = connectDatabase();
$stmt = $conn->prepare («INSERT INTO messages (to_user, from_user,
     text) VALUES (?,?,?)»);
$stmt->bind_param («iis», $to, $from, $text);

if ($stmt->execute()) {
echo «Повідомлення успішно надіслано.»;
} else {
echo «Помилка:». $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

Файл `delete_message.php`:

```php
<?php
require_once('functions.php');

$messageId = intval($_POST['id']);
$userId = intval($_POST['user_id']);

$conn = connectDatabase();
$stmt = $conn->prepare («UPDATE messages SET delete_to = ? WHERE
     id = ? AND delete_to = 0»);
$stmt->bind_param («ii», $userId, $messageId);
```

```
if ($stmt->execute()) {
echo «Повідомлення помічено на видалення.»;
} else {
echo «Помилка:». $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

JavaScript для взаємодії з повідомленнями:

```javascript
document. querySelector("#sendButton»). addEventListener («click», 
     function () {
const text = document. querySelector("#messageInput»). value;
const to = document. querySelector("#recipientId»). value;

if (!text) {
alert («Текст повідомлення не може бути порожнім.»);
return;
}

fetch («send_message.php», {
method: «POST»,
headers: { «Content-Type»: «application / x-www-form-urlencoded» 
     },
body: `text=${encodeURIComponent (text)}&to=${encodeURIComponent 
     (to)}`
})
.then (response => response. text())
.then (data => alert (data))
.catch (err => console. error («Помилка:», err));
});
```

Додаток В

Форма входу в адміністративну частину сайту

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
     scale=1.0">
    <title>Форма входу</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f9f9f9;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
        }
        .login-container {
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
            width: 300px;
            text-align: center;
        }
        .login-container h2 {
            color: #e74c3c;
            margin-bottom: 20px;
        }
        .login-container input[type="text"],
        .login-container input[type="password"] {
            width: 100%;
            padding: 10px;
            margin: 10px 0;
            border: 1px solid #ddd;
            border-radius: 4px;
            box-sizing: border-box;
        }
        .login-container button {
            background-color: #3498db;
            color: #fff;
            border: none;
            padding: 10px 15px;
            border-radius: 4px;
            cursor: pointer;
            width: 100%;
            margin-top: 10px;
        }
```

```
        .login-container button:hover {
            background-color: #2980b9;
        }
        .login-container p {
            font-size: 14px;
            color: #555;
            margin-top: 15px;
        }
        .login-container a {
            color: #3498db;
            text-decoration: none;
        }
        .login-container a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
    <div class="login-container">
        <h2>Вхід</h2>
        <form action="/login" method="POST">
            <input type="text" name="email" placeholder="E-mail
     (логін)" required>
            <input type="password" name="password"
     placeholder="Пароль" required>
            <button type="submit">Увійти</button>
        </form>
        <p>
            Якщо Ви забули пароль, <a href="/forgot-
     password">натисніть тут</a> для відновлення.
        </p>
    </div>
</body>
</html>
```

Додаток Г

Лістинг сторінки послуг сайту

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
     scale=1.0">
    <title>Bubbly - Меню</title>
    <link rel="stylesheet" href="styles.css">
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f9f9f9;
        }
        header {
            background-color: #ffc107;
            color: white;
            padding: 10px;
            text-align: center;
        }
        h1 {
            margin: 0;
            font-size: 2em;
        }
        .menu-container {
            display: grid;
            grid-template-columns: repeat(auto-fit, minmax(300px,
     1fr));
            gap: 20px;
            padding: 20px;
            max-width: 1200px;
            margin: 0 auto;
        }
        .menu-item {
            background-color: white;
            border-radius: 10px;
            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
            overflow: hidden;
            transition: transform 0.3s;
        }
        .menu-item:hover {
            transform: scale(1.05);
        }
        .menu-item img {
            width: 100%;
            height: 200px;
            object-fit: cover;
```

```
        }
        .menu-item h3 {
            margin: 10px;
            font-size: 1.5em;
        }
        .menu-item p {
            margin: 10px;
            font-size: 0.9em;
            color: #555;
        }
        .menu-item .price {
            font-weight: bold;
            color: #28a745;
            margin: 10px;
        }
        footer {
            background-color: #333;
            color: white;
            text-align: center;
            padding: 10px;
            margin-top: 20px;
        }
    </style>
</head>
<body>
<header>
    <h1>Меню ресторану Bubbly</h1>
</header>

<main>
    <div class="menu-container">
        <div class="menu-item">
            <img
     src="https://via.placeholder.com/300x200?text=Салат+Цезар"
     alt="Салат Цезар">
            <h3>Салат Цезар</h3>
            <p>Класичний салат з куркою гриль, пармезаном,
     сухариками та фірмовим соусом.</p>
            <p class="price">Ціна: 120 грн</p>
        </div>
        <div class="menu-item">
            <img
     src="https://via.placeholder.com/300x200?text=Піца+Маргарита"
     alt="Піца Маргарита">
            <h3>Піца Маргарита</h3>
            <p>Соковита піца з томатним соусом, сиром моцарела та
     базиліком.</p>
            <p class="price">Ціна: 150 грн</p>
        </div>
        <div class="menu-item">
            <img
     src="https://via.placeholder.com/300x200?text=Стейк+Рібай"
     alt="Стейк Рібай">
```

```
            <h3>Стейк Рібай</h3>
            <p>Соковитий стейк зі спеціями, обсмажений до
        ідеального стану.</p>
            <p class="price">Ціна: 350 грн</p>
        </div>
        <div class="menu-item">
            <img
        src="https://via.placeholder.com/300x200?text=Суші+Сет"
        alt="Суші Сет">
            <h3>Суші Сет</h3>
            <p>Асортимент найпопулярніших ролів з якісних
        інгредієнтів.</p>
            <p class="price">Ціна: 400 грн</p>
        </div>
        <div class="menu-item">
            <img
        src="https://via.placeholder.com/300x200?text=Кава+Капучино"
        alt="Кава Капучино">
            <h3>Кава Капучино</h3>
            <p>Ароматна кава з пишною молочною пінкою.</p>
            <p class="price">Ціна: 50 грн</p>
        </div>
        <div class="menu-item">
            <img
        src="https://via.placeholder.com/300x200?text=Десерт+Тірамісу
        " alt="Десерт Тірамісу">
            <h3>Десерт Тірамісу</h3>
            <p>Ніжний десерт з маскарпоне, просочений кавою та
        посипаний какао.</p>
            <p class="price">Ціна: 100 грн</p>
        </div>
    </div>
</main>

<footer>
    <p>&copy; 2025 Ресторан Bubbly. Усі права захищено.</p>
</footer>
</body>
</html>
```