КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет автоматизації інформаційних технологій

Кафедра інформаційних технологій

(назва випускової кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ **БАКАЛАВР**

на тему:

Web-додаток для автоматизації бізнес-процесів ресторану

ДАРНОПИХ Дмитро Миколайович

Київ 2025 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

інформаційних технологій

(кафедра)

ЗАТВЕРДЖУЮ
Завідувачка кафедри ІТ
д.т.н., професор Гончаренко Т.А.
_____
„___” _____2025 року

КВАЛІФІКАЦІЙНА РОБОТА ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

на тему:

«Web-додаток для автоматизації бізнес-процесів ресторану»

Я як здобувач вищої освіти КНУБА розумію і підтримую політику закладу з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Здобувач
Дарнопих Дмитро Миколайович
122 «Комп’ютерні науки»
(спеціальність)
Інформаційні управляючі системи і технології
(освітня програма)
Групи КН-21-1
Керівник Рябчун Ю.В.
(прізвище та ініціали)
Доктор філософії
(вчене звання, науковий ступінь)
Рецензент к.т.н., доц. Баліна О.І.
(Прізвище та ініціали)

Ідентичність підтверджую

Київ, 2025 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет:	Автоматизації інформаційних технологій
Випускова кафедра:	Інформаційних технологій
Освітній ступінь:	Бакалавр
Спеціальність:	Комп’ютерні науки
Освітня програма:	Інформаційні управляючі системи та технології

ЗАТВЕРДЖУЮ

Завідувач кафедри ІТ

Тетяна ГОНЧАРЕНКО

_____

„___”_____2025 року

З А В Д А Н Н Я

ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

	Дарнопих Дмитро Миколайович

1. Тема роботи Web-додаток для автоматизації бізнес-процесів ресторану затверджена наказом ректора КНУБА № 235/23/25 від «14» лютого 2025 року

2. Керівник роботи	Рябчун Юлія Володимирівна, PhD

3. Строк подання Здобувачем роботи до захисту 28/05/2025.

4. Зміст пояснювальної записки за розділами:

Р.1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

Р.2 ПРОЕКТУВАННЯ WEB-ДОДАТКУ

Р.3 РЕАЛІЗАЦІЯ WEB-ДОДАТКУ

Р.4 ОЦІНКА ЕФЕКТИВНОСТІ ТА ПЕРСПЕКТИВИ РОЗВИТКУ

5. Графічний матеріал за розділами:

Р.1 рисунків 8, таблиць 3

Р.2 рисунків 8, таблиць 5, діаграм 2

Р.3 рисунків 22

Р.4 таблиць 1

6. Календарний план виконання роботи:

Види робіт та їх зміст	Дата виконання
Розділ 1	05.03.2025
Розділ 2	06.04.2025
Розділ 3	21.04.2025
Розділ 4	23.05.2025
Остаточне оформлення роботи	28.05.2025
Направлення роботи для перевірки на плагіат	28.05.2025
Попередній захист роботи на випусковій кафедрі	29.05.2025
Направлення роботи на рецензування	29.05.2025

7. Консультанти розділів атестаційної випускної роботи

Розділ	Прізвище, ініціали та посада консультанта	Перевірив	
		дата	підпис
Розділ 1	Рябчун Ю.В., доцент каф.ІТ	05.03.2025	
Розділ 2	Рябчун Ю.В., доцент каф.І	06.04.2025	
Розділ 3	Рябчун Ю.В., доцент каф.ІТ	21.04.2025	
Розділ 4	Рябчун Ю.В., доцент каф.ІТ	23.05.2025	

8. Дата видачі завдання січень 2025 р.

Зав. кафедри			Гончаренко Т.А.
	(підпис)		(прізвище та ініціали)
Керівники			Рябчун Ю.В.
	(підпис)		(прізвище та ініціали)
Здобувач			Дарнопих Д.М.
	(підпис)		(прізвище та ініціали)

АНОТАЦІЯ

Дарнопих Д.М. Web-додаток для автоматизації бізнес-процесів ресторану.

Кваліфікаційна випускна робота бакалавра за спеціальністю 122 «Комп'ютерні науки», освітня програма «Інформаційні управляючі системи та технології». – Київський національний університет будівництва та архітектури. – Київ, 2025.

Дана робота присвячена розробці веб-додатку для автоматизації бізнес-процесів ресторану з використанням технологій штучного інтелекту. У роботі реалізовані функціональні можливості системи, такі як оформлення замовлень, адміністрування меню, реєстрація користувачів, бронювання столиків. Також приділено увагу інтеграції алгоритмів штучного інтелекту для персоналізації рекомендацій клієнтам.

У ході дослідження розглянуто сучасні технології веб-розробки, архітектурні підходи до побудови веб-систем та методи інтеграції з базами даних. Експериментальна частина оцінює продуктивність веб-додатку, ефективність роботи алгоритму штучного інтелекту, зручність використання та вплив автоматизації на підвищення ефективності бізнес-процесів.

Робота викладена на *110* аркушах, містить *1* додатків, *9* таблиць, *38* рисунків, *2* діаграм, список використаної літератури із *25* найменувань.

Ключові слова: веб-додаток, автоматизація, бізнес-процеси, штучний інтелект, база даних, UI/UX, API, HTML, CSS, Bootstrap, WordPress, WooCommerce, JavaScript, SQLite, Python, Flask.

SUMMARY

Darnopykh D. M. Web application for restaurant business process automation. Bachelor's thesis in the specialty: 122 "Computer Science," specialization: "Information Management Systems and Technologies." – Kyiv National University of Construction and Architecture. – Kyiv, 2024.

This thesis is dedicated to the development of a web application for automating restaurant business processes using artificial intelligence technologies. The system implements functionalities such as food ordering, menu management, user registration, table reservations. Special attention is given to the integration of AI algorithms for generating personalized customer recommendations.

The research examines modern web development technologies, architectural approaches to building web systems, and methods of database integration. The experimental part evaluates the performance of the web application, the effectiveness of the AI algorithm, user-friendliness, and the impact of automation on business process efficiency.

ЗМІСТ

Оглавление

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

REST API (Representational State Transfer Application Programming Interface) - Інтерфейс програмування застосунків на основі передачі стану представлення.

UI (User Interface) - інтерфейс користувача

UX (User Experience) - досвід користувача

ШІ - штучний інтелект;

БД – база даних;

UML (Unified Modelling Language) – універсальна мова моделювання;

ERD (entity-relationship diagram, ER-діаграма) – діаграма сутність-зв'язок;

SQLite – компактна вбудована реляційна база даних;

ВСТУП

У сучасному світі розвиток інформаційних технологій суттєво змінює підходи до ведення бізнесу, зокрема у сфері ресторанної індустрії. Зростаюча конкуренція, високі вимоги до якості обслуговування клієнтів і необхідність оптимізації внутрішніх процесів спонукають власників закладів шукати ефективні рішення для управління своїм бізнесом. Автоматизація бізнес-процесів стає ключовим фактором підвищення продуктивності, зниження певних витрат та покращення взаємодії з клієнтами.

В останні роки особливу увагу привертають технології штучного інтелекту (ШІ), які відкривають нові можливості для бізнесу, дозволяючи не лише автоматизувати рутинні завдання, але й аналізувати великі обсяги даних для прийняття стратегічних рішень. Одним з прикладів інтеграції ШІ у веб-додатки для ресторанного бізнесу є персоналізація пропозицій користувачам на основі асоціативних зв’язків, знайдених в їхньої історії замовлення. У випадку, якщо користувач новий, йому можна надати загальні рекомендації із асоціативними правилами, які були знайдені за допомогою аналізу всіх фіскальних чеків закладу.

Попри наявність різних систем для автоматизації роботи ресторанів, багато з них обмежуються базовою функціональністю, що не дозволяє глибоко аналізувати дані чи адаптуватися до швидких змін ринку. Використання штучного інтелекту стає інноваційним підходом, здатним забезпечити конкурентні переваги завдяки гнучкості, масштабованості та здатності до самонавчання.

Дана робота присвячена розробці веб-додатку для ресторанного бізнесу із використанням технологій штучного інтелекту.

Основна мета додатку — автоматизація ключових бізнес-процесів ресторану з одночасною інтеграцією персоналізації обслуговування клієнтів.

Метою даної роботи є розробка веб-додатку для ресторанного бізнесу, що використовує штучний інтелект для персоналізації пропозицій клієнтам, збільшуючи середній чек.

Завдання дослідження:

1. Аналіз існуючих рішень для автоматизації ресторанного бізнесу та визначення їхніх обмежень.
2. Дослідження можливостей застосування технологій ШІ у сфері ресторанного бізнесу.
3. Визначення функціональних вимог до веб-додатку з інтеграцією ШІ.
4. Проєктування архітектури веб-додатку та розробка його основних модулів.
5. Реалізація алгоритмів ШІ для надання персоналізованих пропозицій користувачам.
6. Тестування веб-додатку та аналіз його ефективності в умовах моделювання реальних бізнес-процесів.
7. Формування рекомендацій щодо подальшого вдосконалення додатку з використанням ШІ.

Об’єкт дослідження – процес автоматизації бізнес-процесів у ресторанному бізнесі.

Предмет дослідження – веб-додаток для ресторанного бізнесу із використанням ШІ та його вплив на ефективність управління бізнесом.

Методи дослідження – аналіз літературних джерел, дослідження сучасних технологій ШІ, проєктування програмного забезпечення, розробка алгоритмів, моделювання бізнес-процесів, тестування програмного продукту та аналіз результатів.

Розділ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

У даному розділі проведено аналіз предметної області, що стосується розробки веб-додатку для автоматизації бізнес-процесів ресторану з елементами технологій штучного інтелекту. Виконано аналіз існуючих рішень, визначено їхні переваги та недоліки. На основі отриманих даних сформульовано основні проблеми та вимоги до програмного забезпечення, а також визначено завдання, які необхідно вирішити за допомогою розробленого веб-додатку. Усі ці етапи є ключовими в процесі розробки програмного забезпечення, оскільки вони визначають подальші кроки та спрямовують робочий процес на досягнення успішного результату.

Ресторанний бізнес є однією з найбільш динамічних галузей, що постійно змінюється під впливом економічних, соціальних та технологічних факторів. Останні 5 років стали особливо непростими для цієї сфери, спочатку через пандемію COVID-19, потім через повномасштабне вторгнення в Україну, що суттєво змінили умови функціонування закладів громадського харчування країни.

1.1 Постановка та аналіз проблеми

Пандемія COVID-19 спричинила значне падіння доходів ресторанної індустрії через введення карантинних обмежень, закриття фізичних закладів, зменшення кількості відвідувачів та необхідність дотримання суворих санітарно-гігієнічних норм. У цих умовах набули популярності "кухні-привиди" (ghost kitchens) — заклади, що працюють виключно на доставку без фізичних точок обслуговування клієнтів [1]. Це дозволило рестораторам знизити витрати на оренду та обслуговування приміщень, одночасно задовольняючи підвищений попит на доставку їжі. За даними дослідження, проведеного компанією "Ресторанний консалтинг", такі формати стали особливо затребуваними в посткарантинний період [1]. Крім того, ресторани змушені були інвестувати в цифрові технології для підтримки роботи: створення власних додатків, QR-меню,

впровадження безконтактних способів оплати та співпраця з платформами доставки, щоб забезпечити безперебійне обслуговування клієнтів [2]. На рисунку 1.1 наведено приблизний обсяг ринку доставки їжі в усьому світі з 2022 до 2027 рр.

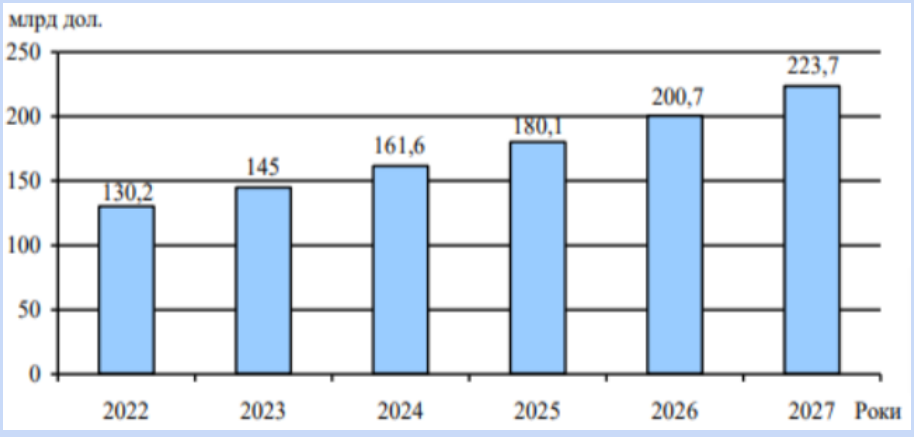

Рисунок 1.1. Приблизний обсяг ринку доставки їжі
в усьому світі з 2022 до 2027 рр [3]

Оскільки ринок доставки їжі демонструє стабільне зростання, веб-додаток повинен підтримувати інтеграцію з популярними платформами (Glovo, Uber Eats, DoorDash), або мати вбудовані алгоритми для доставки [2].

Повномасштабне вторгнення в Україну у 2022 році створило додаткові виклики для ресторанного бізнесу, серед яких — економічна нестабільність, проблеми з логістикою, енергетичні обмеження та зниження купівельної спроможності населення. У цих умовах особливо важливою стала оптимізація операційних процесів, зниження витрат і підвищення ефективності бізнесу.

Враховуючи ці виклики, впровадження цифрових рішень, зокрема веб-додатків із використанням технологій ШІ, стає стратегічно важливим для ресторанної індустрії. Такі рішення дозволяють автоматизувати управлінські процеси, покращити обслуговування клієнтів, оптимізувати логістику та прогнозування попиту, а також забезпечити ефективний моніторинг фінансових показників.

Таким чином, розробка веб-додатку для ресторанного бізнесу із використанням ШІ є актуальним завданням, яке сприятиме подоланню сучасних викликів і забезпечить сталий розвиток галузі в умовах нестабільності.

1.2 Аналіз сучасних рішень автоматизації в ресторанному бізнесі

Для ефективної розробки веб-додатку важливо провести детальний аналіз існуючих рішень на ринку. Це дозволить глибше зрозуміти сильні та слабкі сторони конкурентів, виявити невикористані можливості та визначити потенційні напрямки для вдосконалення власного продукту. Дослідження досвіду інших платформ дає змогу зрозуміти, які фактори є ключовими для залучення та утримання користувачів: зручність інтерфейсу, оперативність обробки замовлень, широта функціоналу чи гнучкість налаштувань. Водночас аналіз недоліків конкурентних рішень допомагає виявити слабкі місця, які можна використати як основу для створення власних конкурентних переваг. У результаті такий підхід сприятиме розробці продукту, який краще відповідатиме очікуванням цільової аудиторії та зможе ефективно конкурувати на ринку.

1.2.1 Підсистема онлайн замовлень та бронювань

Однією з найважливіших сфер діджиталізації ресторанного бізнесу є організація можливості замовлення страв онлайн. Це не лише підвищує комфорт клієнта, але й дозволяє закладу оптимізувати внутрішні процеси, контролювати потоки замовлень, здійснювати аналіз продажів, запускати персоналізовані пропозиції та програми лояльності.

На сучасному ринку існує [4] два основних підходи до впровадження можливості онлайн-замовлень. Перший підхід - розробка та використання власного веб-додатку. Це індивідуальне рішення, створене спеціально для потреб конкретного бізнесу. Такий підхід дозволяє повністю контролювати клієнтський досвід, інтегрувати функції лояльності, проводити маркетингові кампанії та володіти базою даних клієнтів.

Переваги:

- Повний контроль над клієнтським досвідом та даними.
- Можливість впровадження інновацій, таких як ШІ для персоналізації.
- Зниження витрат на комісії стороннім сервісам.

Недоліки:

- Високі початкові інвестиції у розробку та підтримку платформи.
- Необхідність самостійного управління логістикою та маркетингом.

Приклади ресторанів:

- BEEF meat & wine має власний сайт з можливістю онлайн-замовлень та бронювання столиків.
- Lucky Restaurant Vinotheque використовує власну цифрову платформу для замовлень та взаємодії з клієнтами.
- 36ПО львівський ресторан з інтерактивним сайтом, що дозволяє клієнтам ознайомитися з меню та зробити замовлення.

Другий підхід - партнерство зі сторонніми службами доставки, такими як Glovo, Bolt Food (рис.1.2), які вже мають популярні платформи для онлайн-замовлень, багатомільйонну аудиторію та напрацьовану логістику.

Рисунок 1.2. Фрагменти Glovo та BoltFood

У цьому випадку ресторан інтегрує своє меню до додатку служби доставки, користуючись її інфраструктурою, аудиторією та системою платежів. Проте, така

співпраця часто супроводжується високими комісіями та обмеженим контролем над клієнтським досвідом (табл.1.1).

Таблиця 1.1 - Порівняння моделей онлайн-замовлення

Ресторан	**Модель взаємодії**	**Власний сайт**	**Співпраця з агрегаторами**
3B Cafe	Співпраця з агрегаторами (Glovo)	Ні	Так
Kitai	Співпраця з агрегаторами (Glovo)	Ні	Так
BEEF meat & wine	Власна цифрова інфраструктура	Так	Мінімальна
Lucky Restaurant Vinotheque	Власна цифрова інфраструктура	Так	Мінімальна
36ПО	Власна цифрова інфраструктура	Так	Ні

Переваги:

- Швидкий вихід на ринок без значних інвестицій у цифрову інфраструктуру.
- Доступ до великої бази клієнтів агрегатора.
- Мінімальні витрати на маркетинг та логістику.

Недоліки:

- Високі комісії (до 30% від вартості замовлення).
- Обмежений контроль над клієнтськими даними та досвідом.
- Залежність від політики та умов агрегатора.

Приклади ресторанів:

- Kitai, 3B Cafe, Torisho, Menya Musashi. Ці заклади співпрацюють з Glovo через хмарні кухні, не маючи власних сайтів [5].

Обидва підходи мають свої переваги та недоліки. У виборі між ними бізнес має враховувати багато факторів: масштаб закладу, фінансові можливості, стратегічні цілі, маркетингову політику, тип клієнтів, територіальне охоплення, конкурентне середовище тощо.

У результаті, вибір між створенням власного цифрового рішення та партнерством зі службами доставки — це стратегічне рішення, яке визначає не лише зручність обслуговування клієнта, але й загальну бізнес-модель ресторану. Саме тому в межах цього розділу було здійснено порівняльний аналіз обох підходів, визначено їхні сильні й слабкі сторони.

Тепер ми також можемо сформувати критерії, за якими той чи інший варіант може бути найбільш доцільним для конкретного ресторанного бізнесу (табл.1.2).

Таблиця 1.2 - Критерії вибору моделі взаємодії ресторану з клієнтом

№	Критерій	Пояснення	Кому більше підходить
1	**Розмір бізнесу**	Великі та середні ресторани мають більше ресурсів для створення та підтримки власного сайту/додатку.	Власна інфраструктура
2	**Фінансові ресурси**	Створення цифрової платформи вимагає інвестицій у розробку, підтримку, маркетинг.	Власна інфраструктура при наявності ресурсів
3	**Кількість клієнтів / географія роботи**	Якщо ресторан обслуговує велику територію або працює в кількох містах — агрегатори забезпечують логістику та охоплення.	Агрегатори доставки
4	**Залежність від повторних замовлень / лояльність**	Якщо ресторан робить ставку на постійних клієнтів — важливо мати прямий зв'язок і персоналізацію через власну платформу.	Власна інфраструктура

5	**Цінова політика ресторану**	Заклади середнього та високого сегменту часто не хочуть платити комісію агрегаторам або втрачати контроль над ціноутворенням.	Власна інфраструктура
6	**Наявність логістики (доставка)**	Якщо ресторан має власну доставку — йому вигідніше мати свій сайт. Якщо доставки немає — агрегатори стають критично необхідними.	Відповідно: власна або агрегатори
7	**Рівень цифрової грамотності персоналу**	Якщо команда не готова працювати з CRM, чат-ботами, оновленням сайту — простіше використовувати зовнішній сервіс.	Агрегатори доставки
8	**Бренд та маркетингова стратегія**	Якщо ресторан позиціонує себе як унікальний бренд і хоче будувати особисту цифрову присутність — власна платформа необхідна.	Власна інфраструктура
9	**Контроль над даними клієнтів**	Агрегатори не передають ресторану повні дані про замовників. Це обмежує персоналізацію та аналітику.	Власна інфраструктура
10	**Швидкість виходу на ринок**	Партнерство з агрегаторами дозволяє миттєво стартувати без довгих етапів розробки.	Агрегатори доставки

У контексті практичного застосування наведених вище критеріїв доцільно розглянути типові сценарії, з якими може зіткнутися ресторанний бізнес при

прийнятті рішення щодо впровадження тієї чи іншої моделі онлайн-взаємодії з клієнтами.

Якщо ресторан функціонує у великому місті, має значну клієнтську базу, орієнтований на отримання повторних замовлень, володіє власною логістичною інфраструктурою (службою доставки) та володіє необхідними фінансовими ресурсами для запуску і підтримки цифрової платформи, — оптимальним вибором у такому випадку є створення власного веб-додатку або мобільного застосунку. Такий підхід забезпечує більший контроль над взаємодією з клієнтом, персоналізацію послуг, збір аналітичних даних, а також дозволяє знизити довгострокові операційні витрати, пов’язані з комісіями агрегаторів.

Натомість, у випадку, коли ресторан є новоствореним, працює з обмеженим бюджетом, не має налагоджених процесів доставки та бракує внутрішніх ресурсів для самостійного цифрового просування, — доцільним буде партнерство з агрегаторами доставки (Glovo, Bolt Food, тощо). Таке рішення дозволяє швидко інтегруватися в ринок онлайн-доставки, скористатися вже наявною клієнтською аудиторією агрегаторів і уникнути значних початкових витрат.

1.2.2 Підсистеми на основі штучного інтелекту

Програмне забезпечення на основі ШІ може аналізувати величезні обсяги даних клієнтів, надаючи цінну інформацію про тенденції та вподобання клієнтів. Це дозволяє ресторанам персоналізувати досвід клієнтів, оптимізувати розробку меню та приймати рішення на основі даних для покращення обслуговування гостей. Крім того, інструменти штучного інтелекту можуть допомогти в управлінні запасами, автоматизації маркетингових заходів і навіть створенні описів меню, допомагаючи ресторанам підтримувати сильну присутність в Інтернеті та залучати нових клієнтів.

1.2.2.1 SevenRooms

SevenRooms (рис.1.3) – це провідна платформа для управління взаємодією з гостями, розроблена спеціально для ресторанів, готелів та закладів гостинності. Вона допомагає закладам покращити відносини з клієнтами, оптимізувати операційні процеси та збільшити дохід [6].

Рисунок 1.3. Фрагмент SevenRooms

Сильні сторони:

- Управління бронюваннями - система дозволяє ефективно керувати бронюваннями, зменшуючи кількість пропущених можливостей та покращуючи заповнюваність закладу.
- Профілі гостей - збір та аналіз даних про клієнтів дозволяє персоналізувати обслуговування та підвищити лояльність гостей.
- Автоматизація маркетингу - інструменти для створення таргетованих маркетингових кампаній допомагають залучати нових клієнтів та утримувати постійних.

Слабкі сторони:

- Вартість впровадження: Для невеликих закладів вартість впровадження та обслуговування платформи може бути високою.
- Складність інтеграції: Інтеграція з існуючими системами управління може вимагати додаткових ресурсів та часу.

1.2.2.2 Restoke

Restoke (рис.1.4) — це платформа на основі ШІ, розроблена для допомоги ресторанам у ефективному управлінні витратами та оптимізації операційних

процесів. Вона збирає дані про закупівлі та операційні витрати, надаючи реальні рекомендації щодо зниження витрат та покращення ефективності [7].

Рисунок 1.4. Фрагмент Restoke

Сильні сторони:

- Зниження витрат: ресторани, що використовують Restoke, повідомляють про економію в середньому $8000 на тиждень завдяки оптимізації витрат.
- Інтеграція з існуючими системами: платформа легко інтегрується з наявними системами управління рестораном, автоматизуючи процеси від контролю витрат на продукти до управління персоналом.

Слабкі сторони:

- Обмежена географія: наразі Restoke активно використовується в Австралії, Новій Зеландії, Сінгапурі та США, що може обмежувати доступність для закладів в інших регіонах.
- Залежність від якості даних: ефективність рекомендацій платформи залежить від точності та повноти введених даних про операційні процеси.

1.2.2.3 Bronze

Bronze (рис.1.5) — стартап, що спеціалізується на оптимізації операцій у ресторанній індустрії за допомогою ШІ [8]. Їхній продукт, Bronze Tracker, використовує комп'ютерний зір для мінімізації помилок у замовленнях на доставку, зберігаючи візуальні докази для запобігання шахрайству.

Рисунок 1.5. Bronze

Сильні сторони:

- Зменшення помилок: Bronze Tracker допомагає знизити кількість помилок при обробці замовлень, покращуючи якість обслуговування клієнтів.
- Розширення на міжнародні ринки: після успішного впровадження в Іспанії, компанія розпочала експансію в Центральну Америку, що свідчить про ефективність її рішень.

Слабкі сторони:

- Обмежена функціональність: поточний продукт зосереджений переважно на обробці замовлень на доставку, що може не покривати всі потреби ресторану.
- Необхідність додаткових інвестицій: для розширення функціоналу та присутності на ринку компанія потребує додаткових фінансових вливань.

1.2.2.4 AI-голоси для ресторанів

У великих містах США, таких як Нью-Йорк, Маямі, Атланта та Сан-Франциско, ресторани почали впроваджувати AI-голоси для обробки дзвінків клієнтів. Ці системи здатні відповідати на запитання щодо наявності місць, дрес-коду, політик закладу та навіть керувати бронюваннями.

Slang.ai [9] — це інноваційний віртуальний телефонний агент на основі ШІ, створений спеціально для ресторанної індустрії. Функціонуючи як цифровий консьєрж, ця інтелектуальна система цілодобово відповідає на дзвінки, керує бронюванням і надає миттєві відповіді на запити клієнтів. Шляхом автоматизації цих важливих завдань, що виконуються безпосередньо перед закладом, Slang.ai допомагає ресторанам підвищити рівень задоволеності клієнтів, підвищити операційну ефективність і отримати додаткові можливості для отримання прибутку, які інакше могли б бути втрачені.

Однією з видатних особливостей Slang.ai є простота налаштування та налаштування. Ресторани можуть швидко налаштувати платформу відповідно до

голосу свого бренду та бажаної подорожі клієнтів. Примітно, що передовий ШІ Slang.ai може розпізнавати різні акценти та залучати абонентів будь-якого віку, забезпечуючи безперебійну та задовільну роботу для кожного клієнта. Здатність системи навчатися та розвиватися на основі реальних взаємодій з клієнтами дозволяє їй постійно вдосконалювати свою продуктивність, пропонуючи нові відповіді та пристосовуючись до конкретних потреб ресторану з часом.

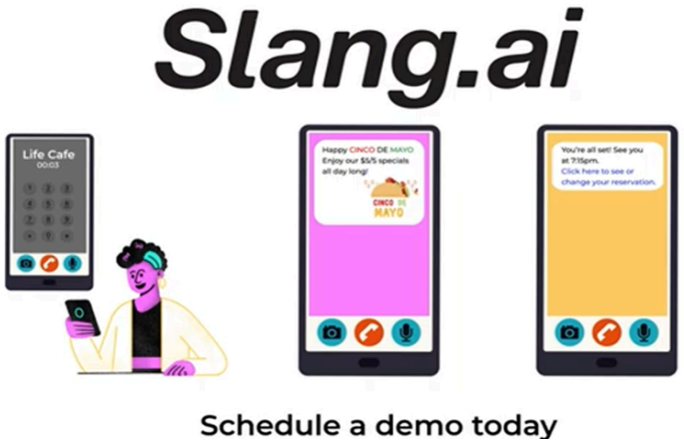

Рисунок 1.6. Фрагмент Slang.ai

Popmenu [10] — це комплексна технологічна платформа для ресторанів, яка використовує потужність штучного інтелекту для вирішення критичних завдань індустрії гостинності. Будучи цифровим центром керування для понад 10,000 XNUMX незалежних ресторанів і гостинних груп, Popmenu пропонує набір рішень на основі штучного інтелекту, призначених для підвищення ефективності, залучення гостей і стимулювання зростання доходів. Платформа об’єднує цифровий маркетинг, онлайн-замовлення та локальні технології, забезпечуючи цілісний підхід до управління рестораном.

Нещодавно Popmenu розширив свої можливості штучного інтелекту, щоб подолати основні перешкоди галузі, такі як нестача робочої сили та втрачені можливості отримання прибутку. Функції платформи, керовані штучним інтелектом, автоматизують різні аспекти роботи ресторану, від створення персоналізованого маркетингового контенту до керування дзвінками клієнтів і

надання поглибленої аналітики. Цей комплексний підхід дозволяє операторам ресторанів оптимізувати свої процеси, приймати рішення на основі даних і зосередитися на наданні виняткових страв.

«Маі» від Encounter AI [11] — це складний помічник із ШІ, спеціально розроблений для секторів ресторанів і роздрібної торгівлі. Це хмарне рішення розроблено для підвищення точності, обслуговування та швидкості, миттєво реагуючи на запити клієнтів через різні канали, включаючи проїзд, кіоск і планшети за столом. Автоматизуючи цю взаємодію, Маі дозволяє персоналу зосередитися на важливіших завданнях, оптимізуючи розподіл ресурсів і підвищуючи прибутковість.

Однією з видатних особливостей Маі є його здатність навчатися та постійно вдосконалюватись за допомогою розмовного штучного інтелекту та машинного навчання. Це дозволяє системі уточнювати своє розуміння пунктів меню, уподобань замовлення та взаємодії з клієнтами з часом. Інклюзивність Маі є ще однією ключовою перевагою, оскільки платформа відповідає розділу 508 щодо доступності та задовольняє когнітивні та дієтичні потреби. Це забезпечує безперебійний і персоналізований досвід для всіх клієнтів, незалежно від їхніх конкретних вимог.

Сильні сторони подібних систем:

- Зменшення навантаження на персонал: AI-голоси дозволяють знизити кількість дзвінків, які обробляє персонал, що особливо корисно під час пікових годин.
- Покращення доступності: Клієнти можуть отримувати відповіді на свої запитання в будь-який час, навіть коли ресторан закритий або персонал зайнятий.

Слабкі сторони:

- Обмеження в розумінні: AI-голоси можуть мати труднощі з обробкою нестандартних або складних запитів, що може призвести до незадоволення клієнтів.
- Проблеми з затримкою: Деякі користувачі відзначають затримки у відповідях або невідповідність відповідей на специфічні запитання.

1.2.3 Адміністрування веб-сайта

У сучасному ресторанному бізнесі надзвичайно важливою є можливість оперативного оновлення веб-сайту — зокрема, редагування меню, цін, опису страв, розкладу роботи та акційних пропозицій. Часто власники або менеджери ресторану не володіють глибокими знаннями у сфері веб-розробки, тому постає потреба в таких програмних рішеннях, які дозволяють самостійно здійснювати базове адміністрування сайту без залучення IT-фахівців.

З цією метою використовуються системи керування контентом (CMS), спеціалізовані SaaS-рішення для ресторанного бізнесу, а також конструктори сайтів з інтуїтивно зрозумілим інтерфейсом.

1.2.3.1 WordPress

WordPress (рис.1.7) є однією з найпопулярніших систем управління контентом у світі. Завдяки широкому вибору шаблонів і плагінів, цю платформу можна адаптувати під потреби ресторанного бізнесу [12].

Рисунок 1.7. Фрагмент WordPress

Переваги:

- Простий та зручний інтерфейс адміністративної панелі.

- Можливість використання спеціалізованих плагінів (наприклад, Five Star Restaurant Menu, WPCafe, RestroPress).
- Підтримка мультимовності, інтеграція з платіжними системами та службами доставки.
- Велика спільнота та наявність документації.

Недоліки:

- Висока залежність від сторонніх плагінів, що може впливати на безпеку.
- Складніше забезпечити високу швидкість завантаження без додаткової оптимізації.

1.2.3.2 Конструктор сайтів Wix (рис.1.8)

Це рішення з орієнтацією на користувачів без досвіду в програмуванні. Вони дозволяють швидко створити веб-сайт із функціоналом для замовлень, меню та зворотного зв'язку [13].

Рисунок 1.8. Фрагмент Wix

Переваги:

- Вбудовані шаблони для ресторанів із адаптивною версткою.
- Автоматична оптимізація для мобільних пристроїв.
- Простота редагування — достатньо базових навичок користувача ПК.

Недоліки:

- Обмежені можливості гнучкої кастомізації та інтеграції нестандартних функцій.
- Залежність від платформи (неможливість перенести сайт на інший хостинг).
- Обмежений контроль над SEO-оптимізацією та швидкодією.

1.2.3.3 SaaS-платформи для ресторанного бізнесу MenuDrive, GloriaFood, UpMenu

Ці сервіси спеціально створені для ресторанів і поєднують у собі сайт, систему замовлень, меню, CRM та маркетингові інструменти [14].

Переваги:

- Швидке створення повнофункціонального ресторанного веб-сайту.
- Панель керування дозволяє змінювати меню, ціни, зображення страв у кілька кліків.
- Підтримка інтеграції з POS-системами, службами доставки та платіжними шлюзами.
- Можливість запуску мобільного додатку без розробки з нуля.

Недоліки:

- Щомісячна абонплата, яка може бути неприйнятною для малих закладів.
- Обмежена кастомізація інтерфейсу (оскільки шаблони типові).
- Деякі сервіси мають географічну прив’язаність (не всі підтримують українську мову або локальні платіжні системи).

Таблиця 1.3 - Порівняння інструментів

Критерій	**WordPress**	**Wix / Tilda / Squarespace**	**MenuDrive / GloriaFood / UpMenu**
Простота для користувача	Середня	Висока	Висока
Можливість редагування меню	Так (через плагіни)	Так	Так (через CRM)
Необхідність програміста	Інколи (на старті)	Ні	Ні
Гнучкість дизайну та кастомізації	Висока	Середня	Низька
Інтеграції з	Через плагіни	Частково	Вбудовано

платіжними системами			
Витрати (на місяць)	$10–50 (хостинг + плагіни)	$15–30	$30–100+
Локалізація українською	Так	Так	Частково

Незалежно від обраного рішення, важливо передбачити можливість швидкої адаптації сайту до змін у бізнес-процесах та забезпечити зручний інтерфейс для самостійної роботи менеджера або адміністратора ресторану.

1.3 Вимоги та особливості проєктування системи

Вимоги та особливості проєктування системи є ключовим етапом у розробці будь-якого інформаційного проєкту. Ці вимоги та особливості визначають параметри та характеристики системи, спрямовані на задоволення потреб користувачів та вирішення конкретних завдань.

1.3.1 Функціональні вимоги

- Реєстрація та авторизація користувачів.
- Можливість створення та редагування профілю користувача.
- Управління замовленнями (прийом, обробка, зміна статусу, історія замовлень).
- Створення меню, оновлення позицій страв, налаштування акцій та знижок.
- Система бронювання столиків онлайн.
- Підсистема персоналізованих пропозицій для клієнтів, що працює на основі аналізу попередніх замовлень, уподобань та поведінки користувача за допомогою штучного інтелекту.

1.3.2 Безпека та конфіденційність

- Захищена реєстрація та вхід у систему з використанням надійних методів аутентифікації.

- Захист персональних даних клієнтів та інформації про замовлення.
- Контроль доступу персоналу до різних рівнів системи.

1.3.4 Інтерфейс

- Інтуїтивно зрозумілий та зручний інтерфейс для адміністраторів, персоналу та клієнтів.
- Адаптивний дизайн для коректної роботи на комп'ютерах, планшетах та смартфонах.

1.3.5 Інновації та технології

- Використання технологій штучного інтелекту для створення персоналізованих пропозицій клієнтам на основі їхніх уподобань, історії замовлень та поведінкових факторів.
- Інтеграція системи з платіжними сервісами для швидкої та безпечної оплати онлайн.

1.3.6 Мультиверсійність

Розробка веб-додатку, який буде доступний через браузери на різних операційних системах (Windows, macOS, Android, iOS).

1.3.7 Розширюваність

Проєктування системи з урахуванням можливості додавання нових функцій та інтеграції з іншими системами у майбутньому.

1.3.8 Адаптивність

Забезпечення адаптивного дизайну для зручного використання на різних розмірах екранів (ПК, планшети, смартфони).

1.3.9 Висока продуктивність

Оптимізація системи для швидкої обробки замовлень і роботи з великою кількістю клієнтів у години пік.

1.3.10 Підтримка міжнародних користувачів

Забезпечення можливості використання додатку різними мовами для роботи як на локальному, так і на міжнародному ринку.

1.3.11 Моніторинг та підтримка

Реалізація системи моніторингу стану додатку, виявлення збоїв та оперативне реагування на проблеми.

1.3.12 Регулярні оновлення

Розробка плану регулярних оновлень для покращення функціоналу, виправлення помилок та вдосконалення системи безпеки.

Отже, існуючі рішення на основі ШІ пропонують широкий спектр можливостей для автоматизації бізнес-процесів ресторанів. Платформи на кшталт **SevenRooms** дозволяють покращити управління бронюваннями та персоналізувати обслуговування гостей. **Restoke** оптимізує витрати та допомагає контролювати фінансові потоки, що критично важливо для прибутковості ресторанного бізнесу. **Bronze** демонструє ефективність застосування комп'ютерного зору для зменшення кількості помилок при виконанні замовлень, особливо у сфері доставки. Впровадження **AI-голосів** для обробки дзвінків допомагає оптимізувати роботу персоналу та покращити комунікацію з клієнтами. Разом із тим, кожне з цих рішень має свої обмеження, зокрема високі витрати на впровадження, залежність від якості даних, складність інтеграції з іншими

системами та можливі проблеми у взаємодії з клієнтами. Це свідчить про необхідність комплексного підходу до автоматизації бізнес-процесів ресторану, який враховує як фінансові можливості закладу, так і специфіку його роботи.

Таким чином, впровадження сучасних технологій, зокрема систем на базі ШІ, є перспективним напрямом розвитку ресторанної індустрії. Вони дозволяють підвищити операційну ефективність, покращити якість обслуговування клієнтів та забезпечити конкурентні переваги на ринку. Однак, для досягнення максимального результату, важливо враховувати індивідуальні особливості кожного закладу та обирати ті інструменти, які найбільше відповідають його потребам.

1.4 Формулювання задачі

У межах даного дослідження передбачається створення веб-додатку, призначеного для автоматизації ключових бізнес-процесів ресторанного закладу. Особливу увагу приділено розробці інтегрованої підсистеми персоналізованих пропозицій, реалізованої з використанням технологій штучного інтелекту. Основною метою є підвищення ефективності внутрішніх операцій та покращення якості обслуговування клієнтів.

Вхідні дані:

- Інформація для реєстрації та авторизації користувачів і персоналу.
- Дані про меню, складові страв та їхню вартість.
- Замовлення клієнтів (історія покупок).
- Інформація про бронювання столиків.

Вихідні дані:

- Дані реєстрації;
- Дані авторизації;
- Персоналізовані пропозиції для клієнтів на основі історії замовлень.
- Список поточних замовлень та їх статус.

Функції системи:

- Реєстрація та авторизація користувачів і персоналу.
- Управління меню та ціноутворенням.
- Оформлення замовлень клієнтами.
- Опрацювання замовлень персоналом.
- Бронювання столиків з вибором місця.
- Генерація персоналізованих пропозицій за допомогою ШІ.

Розділ 2. ПРОЄКТУВАННЯ WEB-ДОДАТКУ

Проєктування веб-додатку для автоматизації бізнес-процесів ресторану є ключовим етапом розробки, який забезпечує структуроване бачення системи та визначає її архітектуру. Основна мета — створення ефективного рішення для оптимізації процесів замовлення страв, бронювання столиків та формування персоналізованих пропозицій на основі штучного інтелекту.

Для реалізації цієї мети було обрано об'єктно-орієнтований підхід до проєктування, що дозволяє моделювати систему як набір взаємодіючих об’єктів із чітко визначеними властивостями. Такий підхід забезпечує масштабованість, гнучкість і зручність у супроводі програмного забезпечення [15].

У процесі проєктування активно використовувались засоби мови моделювання UML. Вона є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. Діаграми дозволили візуалізувати структуру системи, сценарії взаємодії користувачів з підсистемами, а також забезпечити логічну узгодженість між функціональними вимогами та архітектурними рішеннями.

2.1 Визначення функціональних та нефункціональних вимог

Функціональні вимоги є фундаментальною складовою процесу розробки програмного забезпечення, адже вони визначають основну поведінку системи, що відповідає очікуванням користувачів та замовників [16]. У контексті розробки

веб-додатку для автоматизації бізнес-процесів ресторану були визначені три ключові підсистеми:

- Замовлення страв;
- Бронювання столиків;
- Персоналізовані пропозиції на основі штучного інтелекту.

Вимоги для підсистеми замовлення страв:

- Відображення меню ресторану з категоріями, назвами, описами, фото та цінами;
- Можливість додавання страв до кошика;
- Оформлення замовлення з вибором способу оплати;
- Сторінка підтвердження замовлення;
- Інтеграція з CMS WordPress та плагіном WooCommerce для гнучкого адміністрування меню.

Вимоги для підсистеми бронювання столиків:

- Форма для введення даних: кількість гостей, дата, час, ім’я, контактний телефон;
- Перевірка доступності столиків на обрану дату та час;
- Підтвердження бронювання з відповідним повідомленням;
- Збереження записів у базі даних SQLite через серверний застосунок на Flask.

Вимоги для підсистеми персоналізованих пропозицій (ШІ):

- Обробка даних історії замовлень користувачів;
- Аналіз уподобань клієнтів за допомогою алгоритмів машинного навчання;
- Виведення персоналізованих рекомендацій страв на головній сторінці або під час оформлення замовлення..

Нефункціональні вимоги є не менш важливою складовою процесу розробки програмного забезпечення, оскільки вони визначають якісні характеристики системи, що впливають на її зручність, надійність, продуктивність

та масштабованість [17]. На відміну від функціональних, які описують, що саме повинна робити система, нефункціональні вимоги визначають, як саме вона повинна це робити, щоб забезпечити ефективну та стабільну роботу веб-додатку в реальних умовах експлуатації.

У контексті розробки веб-додатку для автоматизації бізнес-процесів ресторану були враховані такі нефункціональні вимоги:

- Інтуітивність інтерфейсу
- Сумісність з популярними веб-браузерами
- Безпека збереження даних
- Доступність сервісу в режимі 24/7

Ці вимоги стали базою для подальшого проектування архітектури системи та моделювання її логіки.

2.2 Вибір методології проєктування

Для проєктування веб-додатку було обрано об’єктно-орієнтовану методологію, яка дозволяє ефективно моделювати складні системи та забезпечувати гнучкість при розширенні функціоналу. Основна мета системи — оптимізація роботи ресторану шляхом автоматизації ключових процесів: замовлення страв, бронювання столиків і формування персоналізованих пропозицій для клієнтів за допомогою інструментів штучного інтелекту.

Об’єктно-орієнтований підхід дозволяє розглядати кожен компонент як окремий об’єкт із власними атрибутами та методами. Наприклад, сутності *замовлення*, *столик* та *користувач* моделюються як класи, що полегшує управління даними та обробку бізнес-логіки.

Система персоналізації розроблена з використанням алгоритмів машинного навчання, які аналізують історію замовлень та уподобання клієнтів для формування точних рекомендацій. Це сприяє підвищенню якості обслуговування й середнього чека.

Для візуалізації структури та логіки взаємодії між компонентами використовуються діаграми UML: варіантів використання, класів та послідовностей. Це забезпечує чітке розуміння вимог до системи, її архітектури та функціональності.

2.3 Вибір архітектури

Проектування ефективного веб-додатку для автоматизації бізнес-процесів ресторану вимагало гнучкого підходу до побудови архітектури. Зважаючи на необхідність обробки різних типів бізнес-логіки — замовлення страв, бронювання столиків та персоналізованих рекомендацій — було вирішено використовувати модульну мікросервісну архітектуру.

2.3.1 Обґрунтування вибору

Мікросервісна архітектура дозволяє розділити систему на незалежні сервіси, кожен із яких виконує чітко визначені функції. Це забезпечує такі переваги:

- Кожен сервіс можна масштабувати окремо залежно від навантаження (наприклад, сервіс замовлень у години пік);
- Гнучкість у розробці та тестуванні - підсистеми можуть розроблятися паралельно, навіть різними командами або мовами програмування;
- Надійність - збій одного модуля не зупиняє роботу всієї системи;
- Простота супроводу - кожен сервіс має власний життєвий цикл, що спрощує оновлення або заміну частин системи.

2.3.2 Способи взаємодії між модулями

Обмін даними між сервісами реалізовано через REST API. Це забезпечує стандартизований і зручний механізм взаємодії. Для автентифікації та авторизації використовується єдина система ідентифікації, що дозволяє зберігати сесії, перевіряти ролі користувачів і обмежувати доступ до функціоналу.

2.3.3 Додаткові міркування

- Розподіл відповідальностей дозволяє легко оновлювати або змінювати окремі частини без впливу на всю систему.
- Можливість хостингу окремих сервісів на різних серверах чи хмарних платформах забезпечує більшу гнучкість у розгортанні.
- Використання кешування та черг повідомлень (під час масштабування проєкту) може покращити продуктивність і стабільність системи.

2.4 Моделювання системи

Для забезпечення ефективного проєктування, аналізу та візуалізації структури майбутньої системи було використано формальні методи моделювання, зокрема мову UML (Unified Modeling Language) та ER-діаграми (Entity-Relationship). Вони дозволили чітко представити ключові компоненти веб-додатку, їхню взаємодію, логіку обробки запитів користувачів і структуру зберігання даних [18].

UML-діаграми, зокрема діаграми варіантів використання, класів та послідовностей, дали змогу описати логіку бізнес-процесів, взаємодію користувачів із системою, а також внутрішню організацію програмних компонентів. ER-діаграми допомогли спроєктувати структуру баз даних підсистем, які не залежать від CMS, та встановити логічні зв’язки між сутностями, що зберігаються.

Такий підхід сприяв системному розумінню взаємозв'язків між модулями, покращив точність формулювання вимог до функціоналу і спростив подальшу реалізацію програмного забезпечення.

2.4.1 База даних

(ER-діаграма. Таблиці, атрибути, зв’язки)

Проєктування бази даних є ключовим етапом моделювання інформаційної системи, оскільки саме структура збереження даних забезпечує цілісність,

узгодженість та ефективний доступ до інформації між підсистемами. Частина структури реалізується автоматично CMS WordPress із плагіном WooCommerce, тоді як інші модулі мають власну реалізацію баз даних. У розробленому веб-додатку ресторану використовуються дві додаткові підсистеми: підсистема бронювання столиків (реалізована з використанням Flask і SQLite) та підсистема персоналізованих рекомендацій на основі ШІ (Python). Для цих підсистем спроектована відповідна логічна модель бази даних у вигляді ER-діаграми та описано сутності з атрибутами й зв’язками.

2.4.1.1 Підсистема бронювання столиків

Бронювання реалізовано як REST-сервіс, який зберігає дані у локальній базі даних SQLite. Такий вибір обґрунтований простотою розгортання, легкістю інтеграції з Python та відсутністю потреби в окремому сервері бази даних.

Таблиця 2.1 - Структура таблиці “Reservations”

Ідентифікатор	Тип дпних	Опис
id	(INTEGER, PRIMARY KEY)	унікальний ідентифікатор бронювання
user_name	(TEXT)	ім’я користувача
user_email	(TEXT)	електронна пошта користувача
table_number	(INTEGER)	номер столика
reservation_date	(DATE)	дата бронювання
reservation_time	(TIME)	час бронювання
guests_count	(INTEGER)	кількість гостей
status	(TEXT)	статус бронювання (активне, скасоване, підтверджене)

2.4.1.2 Підсистема персоналізованих пропозицій на основі ШІ

Для генерації рекомендацій аналізуються історичні замовлення, що експортуються з WooCommerce (WordPress). Вони зберігаються у формі таблиць,

що містять агреговані та очищені дані. Сам ML-модуль працює на основі цих записів для пошуку асоціативних правил між товарами.

Таблиця 2.2 - Структура таблиці “Orders”

Ідентифікатор	Тип дпних	Опис
order_id	(INTEGER, PRIMARY KEY)	унікальний ідентифікатор бронювання
user_id	(INTEGER)	ідентифікатор користувача
order_date	(DATETIME)	дата та час замовлення

Таблиця 2.3 - Структура таблиці “Order_items”

Ідентифікатор	Тип дпних	Опис
id	(INTEGER, PRIMARY KEY)	унікальний ідентифікатор бронювання
order_id	(INTEGER, FOREIGN KEY → orders.order_id)	ідентифікатор замовлення
item_name	(TEXT)	назва страви
item_id	(INTEGER)	внутрішній ідентифікатор страви

Таблиця 2.4 - Структура таблиці “Recommendations”

Ідентифікатор	Тип дпних	Опис
id	(INTEGER, PRIMARY KEY)	унікальний ідентифікатор бронювання
user_id	(INTEGER)	ідентифікатор користувача
recommended_items	(TEXT)	список рекомендованих страв у JSON/CSV-форматікорис тувача

generated_at	(DATETIME)	час генерації рекомендацій

2.4.1.3 Особливості проєктування

- Нормалізація даних дозволяє уникнути дублювання інформації, зберігаючи її структуровано.
- Зв’язки один-до-багатьох між orders і order_items є критично важливими для обчислення закономірностей.
- Таблиця рекомендацій створюється автоматично після обробки даних ML-модулем для кешування результатів та повторного використання.

Проєктована база даних є гнучкою, легко маштабується та підтримує інтеграцію між підсистемами. Завдяки розділенню на логічні компоненти (бронювання, обробка замовлень, рекомендації) забезпечено високий рівень модульності, що спрощує подальше супроводження та масштабування веб-додатку ресторану.

2.4.2 Підпроцеси

У моделюванні інформаційної системи важливу роль відіграє формалізація логіки взаємодії користувачів із системою та послідовності виконання внутрішніх операцій. Для цього застосовуються діаграми з мови моделювання UML (Unified Modeling Language), зокрема діаграми варіантів використання та діаграми послідовності, які дозволяють візуально представити функціональні підпроцеси в рамках інтегрованого веб-додатку ресторану.

2.4.2.1 Діаграма варіантів використання (Use Case Diagram)

Цей тип діаграми описує основні сценарії взаємодії користувачів із системою [19], окреслюючи функціональність, доступну для різних акторів (рис.2.1). У рамках веб-додатку передбачено чотири ключові варіанти використання:

- «Реєстрація та вхід у систему» — сценарій що працює на базі WordPress.
- «Перегляд меню» — сценарій реалізований за допомогою бібліотеки Bootstrap.
- «Перегляд рекомендацій на основі ШІ» — сценарій, що відображає інтеграцію AI-модуля з WordPress.
- «Бронювання столиків» — сценарій, реалізований через сервер Flask із базою SQLite.
- «Замовлення страв та онлайн-оплата» — базовий сценарій через плагін WooCommerce на платформі WordPress.
- «Адмініструвати меню» — функція доступна лише адміністратору для керування асортиментом страв.

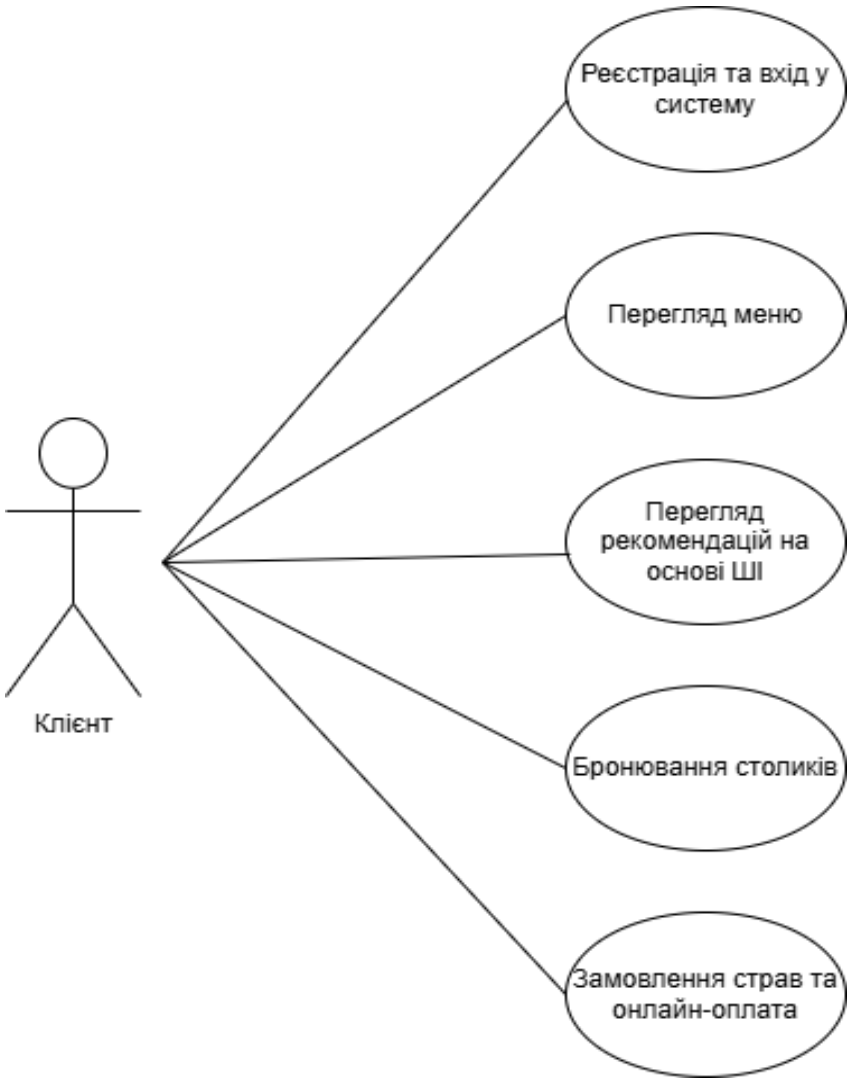

Рисунок 2.1. Діаграма варіантів використання

2.4.2.2 Діаграма послідовності (Sequence Diagram)

Ця діаграма демонструє часову послідовність взаємодії між об'єктами системи в рамках конкретного процесу. Для системи автоматизації ресторану розглянуто два ключові сценарії.

Перший сценарій - реєстрація на базі WordPress:

- Користувач ініціює відкриття форми.
- Заповнює всі поля форми.
- Натискає кнопку "Зареєструватись" для надсилання даних у БД WordPress.

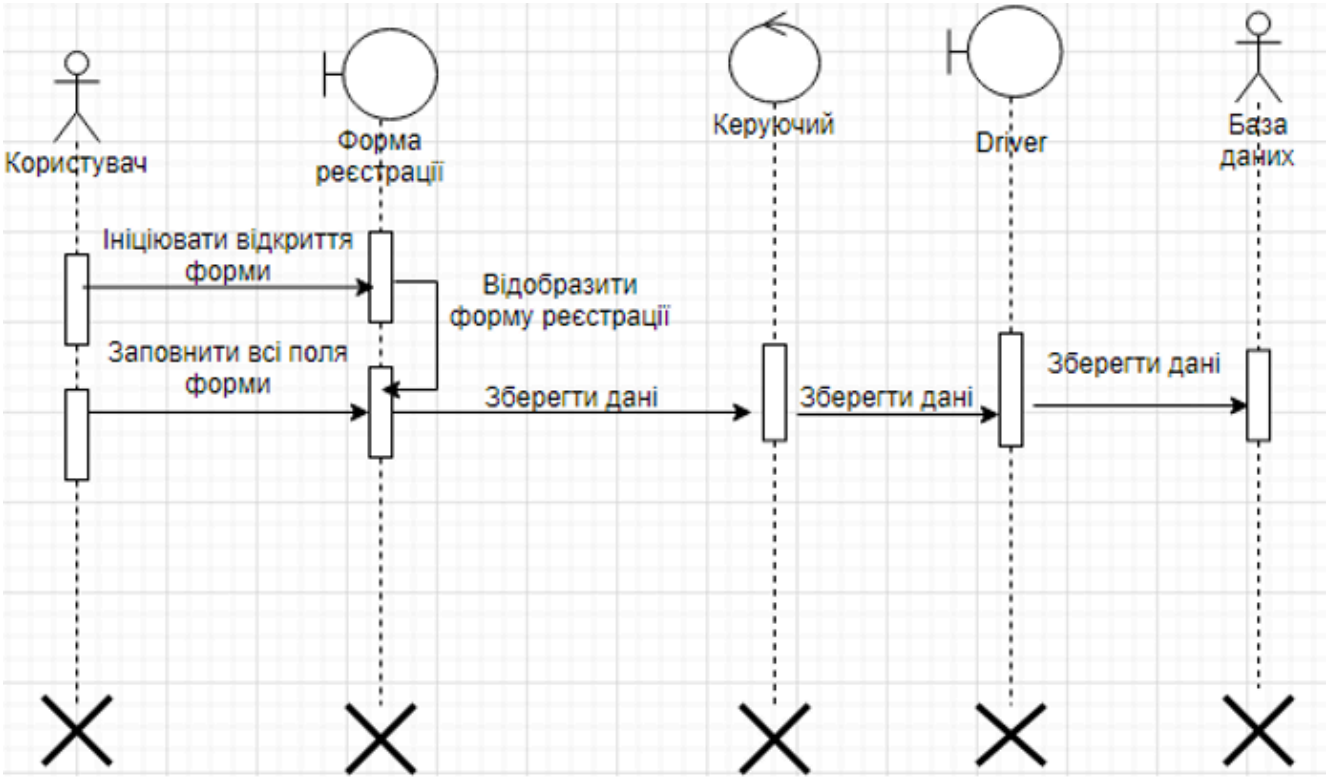

Рисунок 2.2 Діаграма послідовності для процесу реєстрації

Другий сценарій - обробка замовлення страви через WooCommerce:

- Користувач обирає страву з меню та додає її в кошик.
- Дані передаються до модуля WooCommerce для формування замовлення.
- Здійснюється оплата через платіжну систему..
- Платіжна система підтверджує або відхиляє операцію.

- Користувач отримує повідомлення про статус замовлення через інтерфейс WordPress
- Система формує чек, який у подальшому передається до AI-модуля.

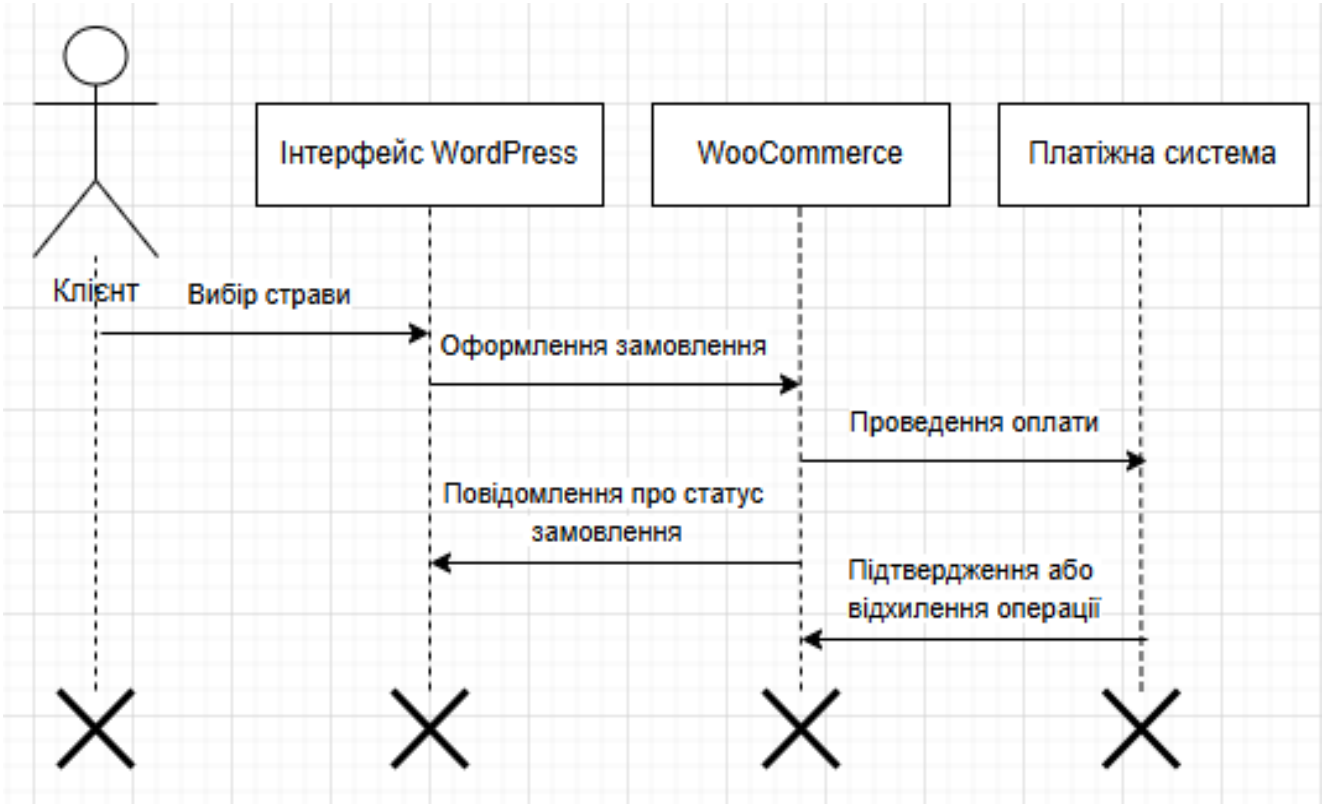

Рисунок 2.3 Діаграма послідовності для процесу обробки замовлення

Третій сценарій - обробка бронювання через Flask:

- Користувач заповнює форму бронювання.
- Запит надходить на сервер Flask.
- Flask взаємодіє з базою SQLite.
- Збереження результату та підтвердження користувачу.

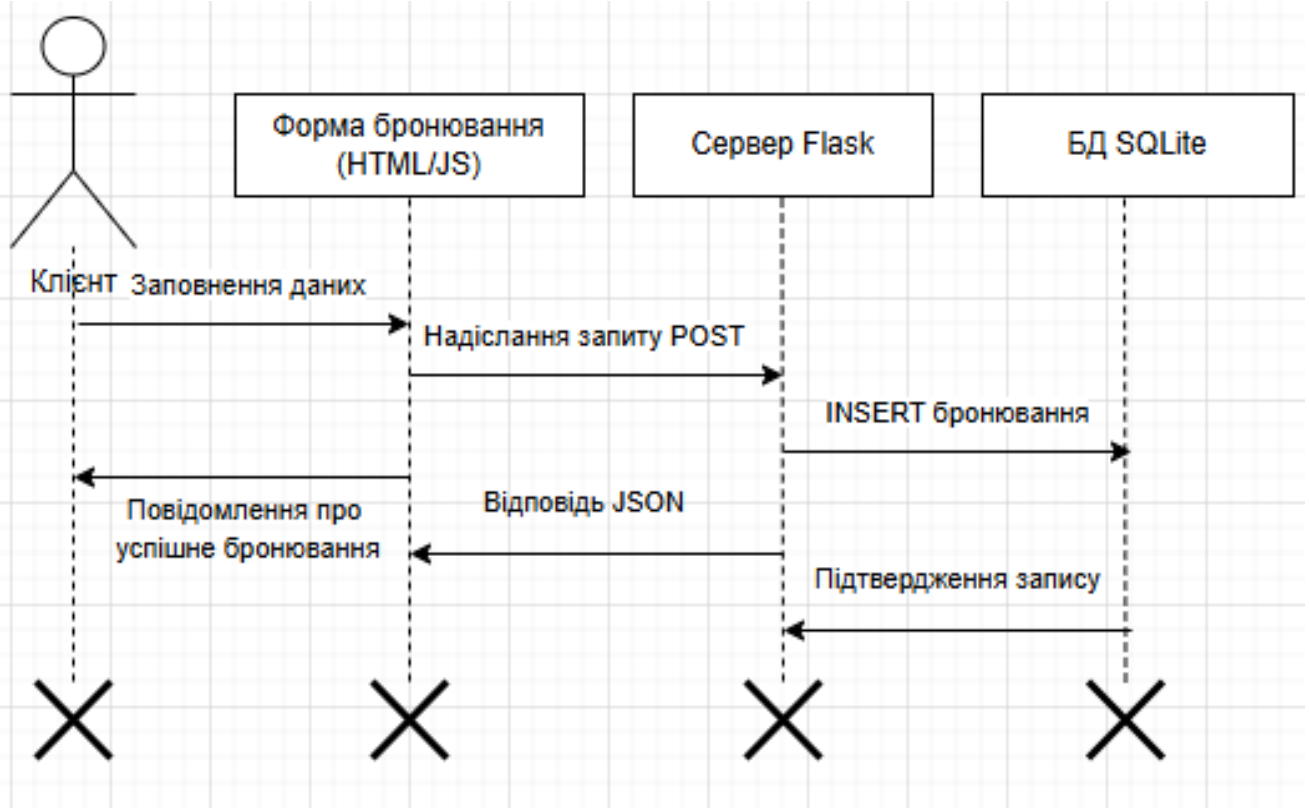

Рисунок 2.4 Діаграма послідовності для процесу бронювання столиків

Застосування UML-діаграм у процесі моделювання дозволяє:

- Формалізувати підпроцеси системи.
- Визначити взаємодію між компонентами.
- Чітко розмежувати функціональність між користувачами та технічними модулями.

Таким чином, підпроцеси оформлення замовлення, бронювання столиків і персоналізації рекомендацій мають чітку логічну структуру та можуть бути ефективно реалізовані у вигляді узгодженої системи.

2.5 Інтерфейс користувача

Інтерфейс користувача є одним із ключових компонентів будь-якої інформаційної системи, оскільки визначає зручність, швидкість та ефективність взаємодії кінцевого користувача з функціоналом системи. У межах розробки веб-додатку для автоматизації бізнес-процесів ресторану інтерфейс створено з

використанням фреймворку Bootstrap, який забезпечує адаптивність, кросбраузерність і швидкість розробки.

Bootstrap — це фронтенд-фреймворк з відкритим кодом, розроблений компанією Twitter [20]. Він надає набір готових компонентів для розмітки, стилізації та JavaScript-функціональності, що дозволяє швидко і якісно створювати сучасні веб-інтерфейси. Основними перевагами Bootstrap є:

- Адаптивна сітка (grid system), яка забезпечує коректне відображення елементів на різних пристроях;
- Компонентність (набір готових елементів: кнопки, форми, модальні вікна);
- Широка документація та підтримка спільноти, що спрощує процес розробки;
- Інтеграція з JavaScript-бібліотеками без необхідності додаткової конфігурації.

Серед альтернатив Bootstrap варто згадати:

- Tailwind CSS — утилітарний фреймворк, що дозволяє створювати унікальні інтерфейси без надмірної стилізації;
- Foundation — потужний, проте менш популярний фреймворк для створення адаптивних веб-інтерфейсів;
- Bulma — легкий фреймворк на базі Flexbox;
- Materialize — фреймворк, заснований на принципах Material Design від Google.

У межах даного проєкту вибір Bootstrap зумовлений його простотою, готовими рішеннями для адаптивності та широким спектром використаних компонентів, що дозволило пришвидшити розробку без втрати якості.

2.5.1 UX/UI логіка та шлях користувача

UX/UI-дизайн розроблено з орієнтацією на інтуїтивну навігацію та мінімізацію кількості кроків до виконання основних дій. Шлях користувача передбачає наступну послідовність:

- Головна сторінка - привітальне повідомлення. та перехід до меню або бронювання столика.

- Сторінка замовлення страв (WooCommerce) - перегляд меню, додавання страв до кошика, оформлення замовлення.
- Сторінка бронювання столика (Flask) - вибір одного з наявних варіантів. Підтвердження та збереження заявки в SQLite.
- Сторінка персональних рекомендацій (AI-модуль) - перегляд запропонованих страв на основі попередніх замовлень.
- Адміністративна панель - редагування меню, перегляд бронювань.

2.5.2 Прототип інтерфейсу

Прототип інтерфейсу є важливим етапом у процесі розробки програмного забезпечення, оскільки дозволяє на ранній стадії візуалізувати зовнішній вигляд та логіку взаємодії користувача із системою. Створення прототипу дає можливість виявити потенційні проблеми юзабіліті, оцінити зручність навігації, а також узгодити бачення кінцевого продукту між замовником, розробниками та користувачами.

Прототип було створено за допомогою інструмента “Wireframe.cc”, який забезпечує зручне середовище для візуалізації інтерфейсних рішень. Його функціональні можливості сприяють зосередженій роботі над структурою користувацького інтерфейсу та ефективному втіленню дизайнерських ідей. [21].

Рисунок 2.5 Прототип головної сторінки

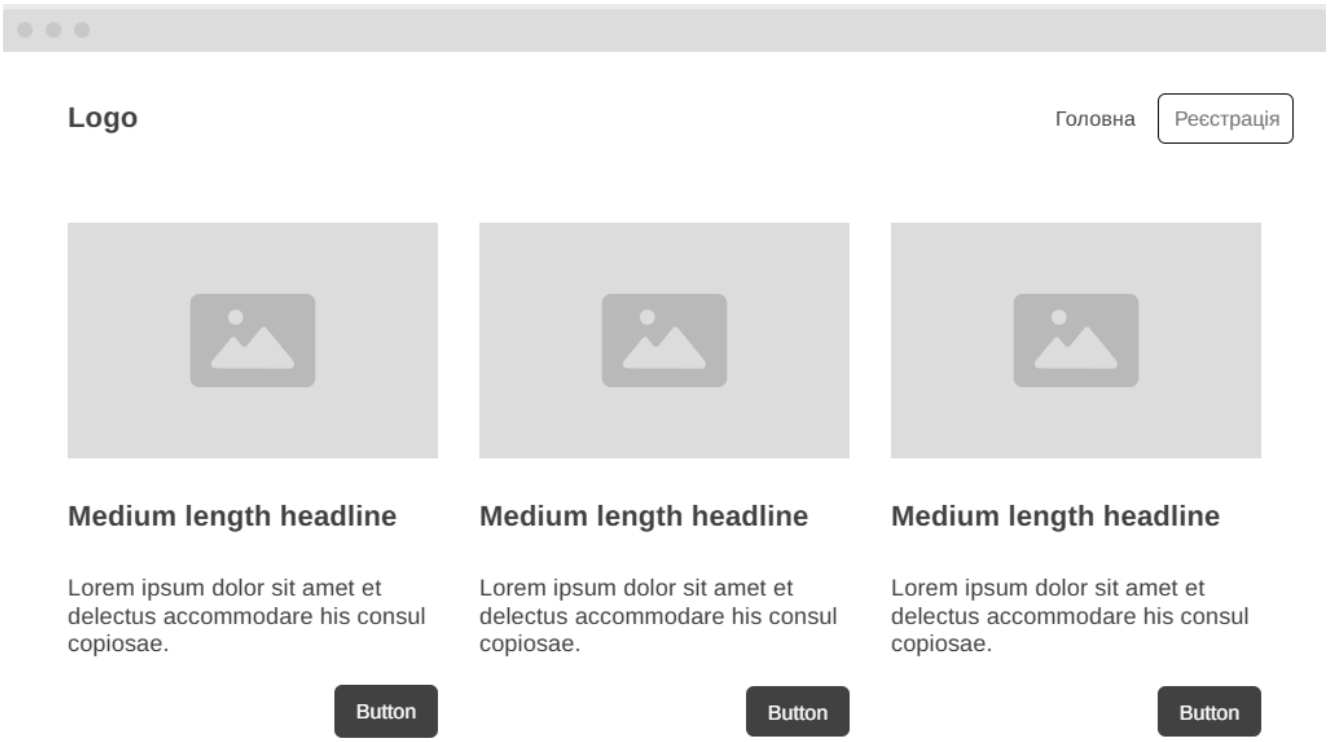

Рисунок 2.6 Прототип сторінки меню

Рисунок 2.7 Прототип сторінки бронювання столиків

2.6 Вибір технологій

2.6.1 Системи управління контентом

У сучасному цифровому світі створення й підтримка веб-ресурсів є невіддільною частиною інформаційної інфраструктури організацій та окремих користувачів. Ефективний інструмент для реалізації цих задач — системи управління контентом (Content Management Systems, CMS).

Система управління контентом (CMS) — це програмне забезпечення, яке дозволяє користувачам створювати, редагувати, публікувати й організовувати цифровий контент без необхідності безпосередньо писати програмний код [22]. Основна мета CMS — спростити процес розробки та обслуговування веб-сайтів, зробивши його доступним для осіб без технічної освіти.

CMS включає в себе інтерфейс для взаємодії з контентом, систему адміністрування, модулі для управління структурою сайту, а також засоби безпеки та контролю доступу. Системи управління контентом широко використовуються у створенні блогів, корпоративних порталів, інтернет-магазинів, освітніх платформ та новинних ресурсів.

Серед найпопулярніших CMS — WordPress, Joomla, Drupal, Magento та інші, кожна з яких має свої особливості, орієнтовані на різні потреби користувачів.

WordPress — це система управління контентом з відкритим кодом, яка з моменту свого заснування у 2003 році еволюціонувала з простої блог-платформи у потужний інструмент для створення веб-ресурсів будь-якої складності. Станом на сьогодні WordPress є найпопулярнішою CMS у світі, забезпечуючи роботу понад 40% усіх веб-сайтів.

Ключові переваги WordPress порівняно з іншими CMS:

- Інтуїтивно зрозумілий інтерфейс дозволяє користувачам без досвіду програмування самостійно редагувати сторінки, публікувати контент та управляти медіафайлами.

- Завдяки десяткам тисяч плагінів і тем, WordPress можна адаптувати до будь-яких функціональних потреб — від особистих блогів до великих електронних комерційних платформ.
- Широке міжнародне співтовариство розробників і користувачів забезпечує постійне вдосконалення платформи, своєчасне оновлення безпеки та наявність численних ресурсів для навчання.
- WordPress має вбудовані механізми, які полегшують оптимізацію сайтів для пошукових систем SEO, що є важливою умовою успішного просування в Інтернеті.
- Наявність безкоштовних шаблонів, плагінів і хостингів робить WordPress доступним навіть для невеликих компаній або приватних осіб з обмеженим бюджетом.
- WordPress підтримує інтеграцію з численними зовнішніми сервісами, такими як соціальні мережі, платіжні системи, CRM тощо, що дозволяє створювати повноцінні цифрові екосистеми.

Загалом можна сказати що системи управління контентом відіграють важливу роль у демократизації цифрового простору, надаючи широкому колу користувачів інструменти для створення веб-ресурсів. WordPress, завдяки своїй доступності, гнучкості та потужному функціоналу, заслужено вважається однією з найкращих CMS. Хоча кожна система має свої переваги у специфічних контекстах використання, WordPress демонструє універсальність, яка робить її оптимальним вибором для більшості проєктів.

CMS WordPress у поєднанні з плагіном WooCommerce слугує основою для реалізації клієнтської частини веб-додатку, забезпечуючи наявність зручної адміністративної панелі. За допомогою цієї системи управління контентом адміністратор отримує можливість редагування вмісту сайту, зокрема керування переліком страв, їх описами, зображеннями та цінами, а також обробки замовлень клієнтів.

WordPress має власну базу даних, тому створювати свою для адміністрування немає необхідності. На таблиці 2.5 наведено [23] огляд усіх таблиць БД, створених під час стандартної інсталяції WordPress. Далі наведено конкретну інформацію про вміст кожної таблиці.

Таблиця 2.5 - Огляд таблиць [23]

Назва таблиці	Опис	Відповідні області інтерфейсу користувача WordPress
wp_commentmeta	Кожен коментар містить інформацію, яка називається метаданими , і вона зберігається у файлі wp_commentmeta .	■ Адміністрація > Коментарі > Коментарі
wp_comments	Коментарі в WordPress зберігаються в таблиці wp_comments .	■ Адміністрація > Коментарі > Коментарі
wp_links	Файл wp_links містить інформацію, пов'язану з посиланнями , введеними у функцію «Посилання» WordPress. *(Ця функція застаріла, але її можна повторно ввімкнути за допомогою плагіна «Менеджер посилань» .)*	■ Адміністрування > Посилання > Додати нове ■ Адміністрація > Посилання > Посилання
wp_options	Параметри , встановлені на панелі Адміністрування > Параметри, зберігаються в таблиці wp_options . Див. Довідку з параметрів для option_namезначень за замовчуванням.	■ Адміністрування > Налаштування > Загальні ■ Адміністрування > Налаштування > Запис ■ Адміністрування > Налаштування > Читання ■ Адміністрування > Налаштування > Обговорення ■ Адміністрування > Налаштування > Конфіденційність ■ Адміністрування > Налаштування > Постійні посилання

		■ Адміністрування > Зовнішній вигляд > Віджети
wp_postmeta	Кожна публікація містить інформацію, яка називається метаданими, і вона зберігається у файлі wp_postmeta . Деякі плагіни можуть додавати власну інформацію до цієї таблиці.	■ Адміністрування > Дописи > Додати новий ■ Адміністрування > Сторінки > Додати нове
wp_posts	Основою даних WordPress є записи (posts) . Вони зберігаються в таблиці wp_posts . Також у цій таблиці зберігаються сторінки та елементи меню навігації.	■ Адміністрування > Дописи > Додати новий ■ Адміністрація > Публікації > Публікації ■ Адміністрування > Сторінки > Додати нове ■ Адміністрування > Сторінки > Сторінки ■ Адміністрування > Медіа > Додати нове ■ Адміністрація > Медіа > Бібліотека ■ Адміністрування > Зовнішній вигляд > Меню
wp_terms	Категорії для публікацій і посилань, а також теги для публікацій знаходяться в таблиці wp_terms .	■ Адміністрація > Дописи > Теги дописів ■ Адміністрація > Дописи > Категорії
wp_termmeta	Кожен термін містить інформацію, яка називається метаданими, і вона зберігається у wp_termmeta .	■ Адміністрування > Посилання > Категорії посилань
wp_term_relationships	Дописи пов’язані з категоріями та тегами з таблиці wp_terms , і цей зв’язок зберігається в таблиці wp_term_relationships . Зв’язок посилань з відповідними категоріями також зберігається в цій таблиці.	■ Адміністрування > Дописи > Додати новий ■ Адміністрація > Публікації > Публікації ■ Адміністрування >

		■ Сторінки > Додати нове ■ Адміністрування > Сторінка > Сторінки
wp_term_taxonomy	У цій таблиці описано таксономію (категорію , посилання або тег) для записів у таблиці wp_terms .	
wp_usermeta	Кожен користувач має інформацію, яка називається метаданими , і вона зберігається у wp_usermeta .	■ Адміністрування > Користувачі
wp_users	Список користувачів зберігається в таблиці wp_users .	■ Адміністрування > Користувачі

На рис.2.1 наведено візуальний огляд бази даних WordPress та зв'язків між таблицями, створеними під час стандартної інсталяції WordPress. Огляд таблиць нижче містить додаткові відомості про таблиці та стовпці.

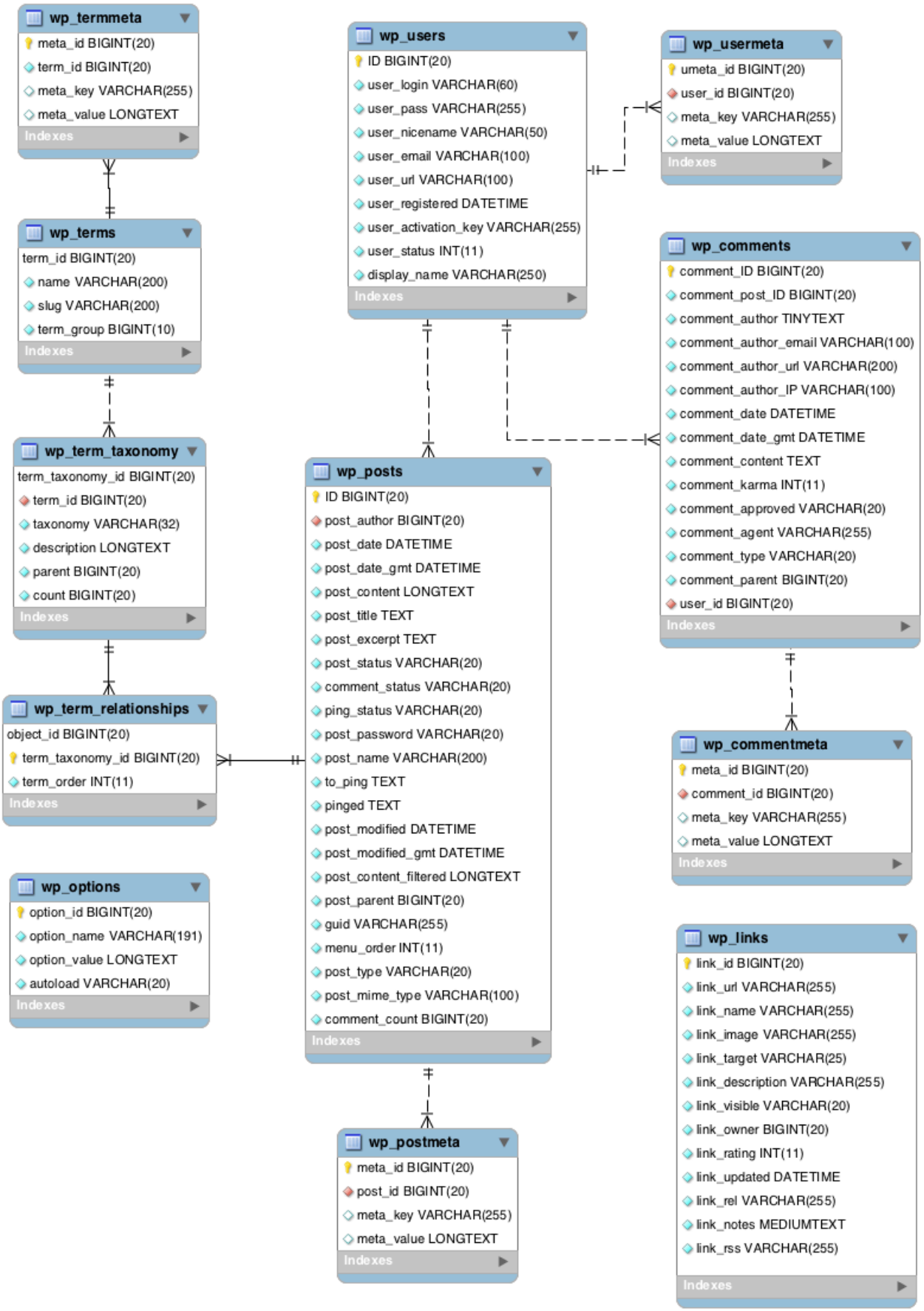

Рисунок 2.8. - Діаграма БД WordPress [23]

2.6.2 Плагіни WordPress

WordPress дозволяє розширювати свою функціональність за допомогою плагінів — спеціальних програмних модулів, що додають або змінюють можливості базової платформи. Однією з ключових сфер застосування плагінів є електронна комерція, зокрема — організація системи оформлення замовлень.

Плагіни для обробки замовлень у WordPress дозволяють перетворити звичайний сайт на повноцінний інтернет-магазин. Вони забезпечують такі функції, як додавання товарів у кошик, обробка платежів, управління доставкою, інтеграція з CRM-системами, а також автоматичне формування рахунків-фактур і повідомлень.

Серед великої кількості плагінів для електронної комерції найвідомішими є:

- WooCommerce - є найпопулярнішим плагіном для електронної комерції у світі. Його розроблено спеціально для WordPress, і він дозволяє створювати магазини як для фізичних, так і для цифрових товарів. WooCommerce підтримує налаштування доставки, податків, купонів, кількох платіжних систем, аналітики та інших ключових функцій для онлайн-продажу.
- Easy Digital Downloads - орієнтований виключно на продаж цифрових продуктів (наприклад, електронних книг, програмного забезпечення, ліцензій тощо). Його перевага — простий інтерфейс і оптимізація під цифрові завантаження, однак він поступається WooCommerce у багатофункціональності.
- WP Simple Pay - плагін призначений для швидкої інтеграції платіжної системи Stripe. Хоча він не є повноцінною e-commerce платформою, WP Simple Pay підходить для сайтів, де не потрібен складний кошик або каталог товарів.
- CartFlows - не є самостійною e-commerce платформою, а доповненням до WooCommerce, яке дозволяє оптимізувати процес оформлення замовлення, додаючи воронки продажів, шаблони сторінок оформлення та функції A/B-тестування.

- Shopify Buy Button - дає змогу інтегрувати товари з Shopify у сайт на WordPress. Незважаючи на зручність, використання такого плагіна передбачає плату за сервіс Shopify і не є повністю нативним рішенням для WordPress.

WooCommerce заслужено вважається найкращим плагіном для організації замовлень у WordPress з низки причин:

- Плагін розроблений з урахуванням архітектури WordPress, що забезпечує високу сумісність і стабільність.
- Базова версія WooCommerce є безкоштовною, а додаткові функції (рівні членства, бронювання, підписки тощо) можна підключити через платні розширення, що дозволяє масштабувати бізнес поступово.
- WooCommerce легко налаштовується під специфіку конкретного бізнесу. Наявність великої кількості тем і плагінів дозволяє створювати унікальні інтерфейси замовлення.
- На відміну від EDD, WooCommerce універсальний — підходить для фізичних товарів, послуг, підписок, цифрових завантажень і навіть бронювання.
- WooCommerce має активну спільноту користувачів і розробників, що сприяє постійному вдосконаленню функціоналу та швидкому вирішенню технічних питань.
- Плагін сумісний з найкращими SEO-інструментами WordPress і має вбудовану аналітику продажів, що важливо для прийняття бізнес-рішень.

Плагіни WordPress відіграють критичну роль у розвитку електронної комерції, забезпечуючи функціональність, необхідну для повноцінної взаємодії з клієнтами. Серед доступних рішень WooCommerce є беззаперечним лідером, що обумовлено його гнучкістю, багатофункціональністю, інтеграцією з WordPress та широкими можливостями масштабування. Таким чином, WooCommerce є оптимальним вибором для більшості підприємств, які прагнуть реалізувати підсистему електронної комерції, забезпечуючи облік замовлень, обробку кошика та налаштування способів оплати через веб-платформу.

2.6.3 Підсистема бронювання столиків

У межах інтегрованого веб-застосунку для автоматизації бізнес-процесів ресторану ключову роль відіграє підсистема бронювання столиків. Її функціональне призначення полягає в організації процесу попереднього резервування місць у залі закладу, що сприяє покращенню клієнтського сервісу, ефективнішому управлінню ресурсами та оптимізації внутрішньої логістики ресторану.

Підсистема реалізована з використанням мікрофреймворку Flask, написаного мовою програмування Python [25]. Цей вибір зумовлений легкістю налаштування, мінімалістичністю та високою гнучкістю Flask, що дозволяє швидко створювати RESTful-сервіси без надмірної складності. Flask є оптимальним рішенням для проєктів малого та середнього масштабу, де потрібне чітке розділення логіки між серверною і клієнтською частинами.

У якості системи управління базами даних обрано SQLite — легковагову реляційну СУБД, яка не потребує розгортання окремого серверного середовища. Цей підхід забезпечує локальне зберігання даних у вигляді звичайного файлу, що значно спрощує розгортання додатка та знижує вимоги до інфраструктури. SQLite є особливо доцільним вибором у випадках, коли обсяги даних є помірними, а система не передбачає одночасної роботи великої кількості користувачів.

Основу серверної логіки підсистеми становить REST API, розроблений на базі Flask. REST-архітектура дозволяє організувати взаємодію між клієнтською частиною (наприклад, веб-інтерфейсом або мобільним додатком) та серверною логікою через HTTP-запити. Це забезпечує модульність, масштабованість та простоту інтеграції підсистеми у ширший програмний комплекс.

REST API забезпечує такі основні функції:

- Створення нового бронювання - користувач може зарезервувати столик, вказавши дату, час, кількість осіб та контактну інформацію.
- Перевірка доступності столиків система дозволяє перевірити наявність вільних місць у заданий проміжок часу.

- Зберігання та обробка інформації - усі операції супроводжуються записом до бази даних SQLite, що дозволяє зберігати історію бронювань, уникати конфліктів та забезпечувати достовірність обліку.

Поєднання Flask і SQLite має низку суттєвих переваг у контексті реалізації підсистеми бронювання:

- Простота розгортання та обслуговування - рішення не потребує встановлення окремого серверного ПЗ (наприклад, PostgreSQL чи MySQL), що робить його ідеальним для малих підприємств.
- Швидкодія при невеликому навантаженні - для локалізованих рішень із помірним числом запитів така архітектура є ефективною і достатньо продуктивною.
- Масштабованість API - RESTful-підхід дозволяє в майбутньому легко розширити або замінити клієнтський інтерфейс без зміни серверної логіки.
- Безпека та контроль доступу - за допомогою Flask-розширень можна реалізувати автентифікацію, авторизацію та захист від типових веб-загроз (наприклад, CSRF або SQL-ін’єкцій).

Підсистема бронювання столиків, розроблена із застосуванням Flask і SQLite, демонструє ефективну реалізацію ключового функціоналу з використанням мінімалістичного, але потужного технологічного стеку. Обрана архітектура забезпечує баланс між функціональністю, простотою реалізації та інфраструктурною економічністю, що робить її доцільною для застосування в невеликих ресторанних закладах та стартапах, орієнтованих на швидке впровадження цифрових рішень.

2.6.4 Персоналізовані пропозиції на основі ШІ

У сучасних веб-додатках, орієнтованих на взаємодію з кінцевими користувачами, персоналізація вмісту на основі аналізу поведінки споживача є важливим чинником підвищення лояльності клієнтів, збільшення обсягу продажів та удосконалення користувацького досвіду. У контексті ресторанного веб-додатку,

інтеграція інтелектуального модуля персоналізованих рекомендацій доповнює функціонал замовлення страв (на базі WooCommerce) і бронювання столиків (реалізованого за допомогою Flask + SQLite), формуючи єдину цифрову екосистему.

Підсистема персоналізованих пропозицій інтегрується у веб-додаток, що функціонує на базі CMS WordPress, розширеної плагіном WooCommerce для управління онлайн-замовленнями. Ключовою особливістю є автоматизоване передавання інформації про замовлені страви — чеків — до окремого інтелектуального модуля, який реалізований на мові програмування Python з використанням бібліотек машинного навчання (наприклад, scikit-learn, pandas, NumPy). Отримані дані зберігаються у локальній базі даних або передаються через REST API до модуля Flask, який також забезпечує логіку підсистеми бронювання.

Система отримує дані про замовлення у форматі JSON через API або пряме зчитування з бази WooCommerce (що функціонує на MySQL/MariaDB). Кожен чек містить інформацію про:

- перелік замовлених страв,
- час замовлення,
- загальну суму,
- ідентифікатор користувача (якщо він авторизований).

Ці дані проходять етапи попередньої обробки, включаючи:

- очищення від повторюваних або неповних записів,
- агрегацію за користувачами та категоріями товарів,
- перетворення у формат, придатний для навчання моделі.

На основі зібраної інформації застосовується одна з класичних моделей рекомендаційних систем — зокрема, асоціативні правила (алгоритм Apriori) або методи колаборативної фільтрації, які дозволяють виявити найбільш частотні поєднання страв у замовленнях. Це дає змогу реалізувати функціонал типу *«З цим також замовляють»*, де система, наприклад, при виборі піци може автоматично

рекомендувати напої, десерти або закуски, які найчастіше зустрічаються в аналогічних кошиках інших користувачів.

Результати обчислень передаються назад до WordPress через REST API або за допомогою механізмів, які дозволяє WooCommerce (наприклад, гачки — *hooks*). Усі рекомендації динамічно відображаються на сторінці товару, сторінці оформлення замовлення або у вигляді персоналізованих банерів, що підвищує ймовірність перехресних продажів (*cross-selling*).

Інтеграція також передбачає можливість локального кешування результатів або застосування простих систем ранжування (наприклад, на основі популярності товарів за останній період) для забезпечення високої швидкодії при великій кількості одночасних запитів.

Інтеграція штучного інтелекту у веб-додаток ресторанного спрямування забезпечує такі переваги:

- Підвищення конверсії: релевантні рекомендації стимулюють додаткові замовлення, збільшуючи середній чек.
- Адаптація до вподобань клієнта: система враховує історію замовлень, що дозволяє формувати індивідуальний профіль кожного користувача.
- Автоматичне оновлення даних: завдяки постійній обробці нових замовлень модель самооновлюється та адаптується до змін у попиті.
- Гнучкість у масштабуванні: технології Python та REST API дозволяють масштабувати систему до більших обсягів даних або розгортати її у хмарних середовищах при потребі.

Підсистема персоналізованих рекомендацій на основі ШІ є ключовим елементом сучасної цифрової стратегії для закладів харчування. Вона поєднує можливості WooCommerce як платформи для замовлень, гнучкість серверної логіки Flask, зручність локального зберігання у SQLite та потужність інструментів машинного навчання Python. Це дозволяє реалізувати цілісне, інтелектуальне та користувачоцентричне рішення, що сприяє ефективнішому обслуговуванню клієнтів та підвищенню економічних показників ресторану.

Діаграма 2.2 - Обмін даними

1. Користувач робить замовлення → WooCommerce
2. WooCommerce створює чек (order) → зберігає в MySQL
3. WordPress плагін/скрипт витягує чек → надсилає POST-запит до Flask API
4. Flask API → передає дані до ML-модуля для обробки
5. ML-модуль → формує список рекомендованих товарів (ID або назви)
6. Flask API → повертає результат до WordPress
7. WordPress відображає "З цим також замовляють"

2.6.5 Додаткові інструменти

До складу додаткових інструментів розробки входять HTML, CSS та JavaScript, які використовуються для створення та налаштування зовнішнього вигляду інтерфейсу. Для тестування REST API застосовується програмне середовище Postman, а для керування версіями програмного коду використовується система контролю версій Git.

Цей набір технологій забезпечує масштабованість, простоту у впровадженні та ефективність у розгортанні веб-рішення для ресторанного бізнесу.

У цьому розділі було здійснено всебічне проєктування веб-додатку для автоматизації бізнес-процесів ресторану з урахуванням сучасних технологій та принципів системного моделювання. Зокрема, було визначено структуру та функціональні підсистеми системи: оформлення замовлень за допомогою CMS WordPress і плагіну WooCommerce, бронювання столиків на основі Flask і SQLite, а також модуль персоналізованих рекомендацій, що інтегрує елементи штучного інтелекту.

Проведено обґрунтування вибору технологій, зокрема переваги WordPress як універсальної CMS із розширюваною екосистемою, ефективність Flask у реалізації REST API для малих застосунків, та Bootstrap як зручного фреймворку для побудови адаптивного інтерфейсу. Визначено ключові компоненти бази даних

та реалізовано ER-діаграму, що описує логіку зберігання і зв’язків між сутностями.

Також було змодельовано основні підпроцеси системи з використанням UML-діаграм варіантів використання та послідовності, що відображають взаємодію користувачів із системою та обробку основних запитів. Особливу увагу приділено логіці передачі даних між модулями, інтеграції між WordPress, Flask і модулем персоналізованих пропозицій на основі даних про попередні замовлення.

Таким чином, запропонована архітектура є гнучкою, масштабованою і здатною до розширення, що забезпечує основу для побудови ефективної автоматизованої системи у сфері ресторанного бізнесу.

Розділ 3. РЕАЛІЗАЦІЯ WEB-ДОДАТКУ

На цьому етапі роботи розглядається практична реалізація розробленого веб-додатку для автоматизації бізнес-процесів ресторану. Основною метою реалізації є створення інтегрованої системи, що охоплює ключові аспекти обслуговування клієнтів — від оформлення онлайн-замовлень і бронювання столиків до генерації персоналізованих рекомендацій. У процесі розробки було використано сучасні веб-технології, такі як WordPress із плагіном WooCommerce для організації меню та замовлень, Flask для реалізації серверної логіки бронювання, а також Python-бібліотеки для побудови інтелектуальної системи рекомендацій.

У даному розділі детально описано структуру проєкту, основні компоненти, взаємозв’язки між підсистемами, а також реалізаційні особливості кожного модуля.

3.1 Опис структури проєкту

Проєкт веб-додатку для автоматизації бізнес-процесів ресторану реалізовано як комплексне рішення, що об'єднує три функціональні підсистеми, кожна з яких виконує специфічні завдання в межах єдиної архітектури. З метою забезпечення гнучкості, масштабованості та підтримуваності, система побудована за модульним принципом і структурована у вигляді трьох взаємодіючих компонентів: підсистеми оформлення замовлень (на базі WordPress та WooCommerce), підсистеми бронювання столиків (реалізованої з використанням Flask), а також підсистеми персоналізованих пропозицій, що базується на технологіях штучного інтелекту. Нижче подано опис основної структури проєкту, ключових директорій і файлів кожної з підсистем.

3.1.1 Підсистема оформлення замовлень (WordPress + WooCommerce)

Цей компонент відповідає за створення інтерактивного інтерфейсу для клієнтів, які здійснюють онлайн-замовлення страв. Його структура включає такі основні елементи:

- wp-content/ – директорія, що містить усі користувацькі ресурси сайту.
- themes/restaurant-theme/ – власна тема, створена для дизайну ресторанного веб-інтерфейсу.
- functions.php – файл з описом функціоналу теми, зокрема інтеграції із WooCommerce.
- template-parts/ – підкаталог з шаблонами для відображення категорій меню, картки товару тощо.
- plugins/custom-order-handler/ – плагін, розроблений для обробки специфічних замовлень (наприклад, страв на замовлення).
- custom-order-handler.php – основний файл плагіна із реєстрацією хуків.
- woocommerce/ – каталог із шаблонами WooCommerce, адаптованими до стилю ресторану.

3.1.2 Підсистема бронювання столиків (Flask + SQLite)

Підсистема реалізована як окремий веб-сервіс на Flask, що забезпечує створення, перегляд і управління бронюваннями.

- reservation_app/ – коренева директорія підсистеми бронювання.
- app.py – головний виконуваний файл застосунку Flask, що містить маршрутизацію запитів.
- templates/ – HTML-шаблони інтерфейсу користувача.
- booking_form.html – форма для здійснення бронювання.
- admin_view.html – шаблон для адміністративного перегляду бронювань.
- static/ – статичні ресурси (CSS, JavaScript).
- database/ – підкаталог для зберігання бази даних.
- reservations.db – файл SQLite з інформацією про всі активні та минулі бронювання.

- models.py – опис моделі даних.
- utils.py – допоміжні функції для перевірки доступності столиків, форматування дат тощо.

3.1.3 Підсистема персоналізованих пропозицій (Python + AI)

Даний модуль формує рекомендації для користувачів на основі їх попередніх замовлень та поведінкових патернів.

- recommendation_system/ – директорія системи рекомендацій.
- train_model.py – скрипт для тренування моделі машинного навчання на історичних даних користувачів.
- predict.py – модуль, який виконує генерацію персоналізованих пропозицій у режимі реального часу.
- data/ – вхідні та вихідні набори даних.
- user_orders.csv – історія замовлень користувачів.
- recommendations.json – результати роботи моделі.
- model/ – збережені моделі у форматі .pkl або інші формати, залежно від фреймворку (наприклад, scikit-learn або TensorFlow).
- api.py – реалізація REST API для інтеграції з фронтендом WordPress через AJAX-запити.

3.1.4 Інтеграційний рівень

Для узгодження роботи всіх підсистем використовуються REST API та відповідні точки обміну даними. Зокрема:

- Використання Flask RESTful API дозволяє WordPress надсилати запити до сервісу бронювання та отримувати підтвердження.
- Модуль рекомендацій також надає ендпоінт для отримання персоналізованих пропозицій на основі ідентифікатора користувача.

Таким чином, проєкт має чітко структуровану архітектуру, де кожна з підсистем ізольована у власному середовищі виконання, що забезпечує легкість

супроводу, гнучке масштабування та можливість незалежного розвитку кожного компонента.

3.2 Реалізація підсистем

Інформаційна система ресторану розроблена як модульна архітектура з трьома основними підсистемами: система замовлень, система бронювання столиків, підсистема персоналізованих пропозицій. Кожна підсистема реалізована з урахуванням специфіки її функціонального призначення та орієнтована на забезпечення зручності для кінцевого користувача й ефективності для адміністратора закладу.

3.2.1 Система замовлень (WordPress + WooCommerce)

Підсистема онлайн-замовлень реалізована на базі CMS WordPress із використанням плагіна WooCommerce. Основна мета цього модуля — надати клієнтам можливість ознайомитися з меню, обрати страви та здійснити замовлення безпосередньо на веб-сайті ресторану.

Ключові компоненти та процеси:

- Меню страв реалізовано як каталог товарів WooCommerce. Кожна страва — це окремий товар з описом, ціною, категорією (наприклад, перші страви, десерти, напої) та зображенням.
- Кошик і оформлення замовлення використовують вбудовані шаблони WooCommerce, адаптовані під ресторанний стиль.
- Обробка замовлення: після оформлення, замовлення автоматично фіксується у базі даних WordPress та надсилається на email адміністратора. Для цього використовуються стандартні хуки плагіна, зокрема woocommerce_thankyou для запуску логіки після оплати.
- Приклад сценарію: користувач обирає страви, додає їх до кошика, вводить контактні дані й надсилає замовлення. На екран виводиться повідомлення про

успішне оформлення, а адміністратор отримує повідомлення про нове замовлення.

3.2.2 Підсистема бронювання столиків (Flask + SQLite)

Ця підсистема дозволяє клієнтам забронювати столик у ресторані, вказавши зручну дату, час та кількість осіб. Реалізована за допомогою мікрофреймворку Flask та СУБД SQLite.

Основна логіка роботи:

- Форма бронювання реалізована як HTML-сторінка з обробкою запитів через метод POST.
- Перевірка доступності столика: серверна логіка перевіряє, чи існують вільні місця на обрану дату та час. Якщо місця є, заявка записується в базу.
- Збереження даних: кожне бронювання зберігається в таблиці reservations бази даних SQLite із полями: id, name, date, time, guests, contact_info.
- Адміністративна панель дозволяє персоналу переглядати поточні бронювання.
- Приклад сценарію: користувач заповнює форму на сторінці /reserve, надсилає її, і система перевіряє наявність вільного столика. У разі успіху виводиться підтвердження бронювання.

3.2.3 Підсистема персоналізованих пропозицій (AI + Python)

Інтелектуальна підсистема персоналізованих пропозицій покликана підвищити лояльність клієнтів, аналізуючи їхні попередні замовлення та генеруючи індивідуальні рекомендації.

Технічна реалізація:

- Модель машинного навчання тренується на основі даних про історію замовлень користувачів (user_id, product_id, order_date).

- Алгоритм використовує колаборативну фільтрацію або кластеризацію за інтересами клієнтів.
- Інтерфейс API: модуль на Flask надає ендпоінт /recommend/<user_id>, який повертає JSON-список рекомендованих страв.
- Інтеграція з WordPress: через AJAX-запити система витягує персоналізовані пропозиції та відображає їх на головній сторінці користувача.
- Приклад сценарію: користувач, який часто замовляє вегетаріанські страви, отримує рекомендації на основі цих вподобань — наприклад, нові позиції зі схожими інгредієнтами.
-

3.3 Інтеграція з базою даних

Інтеграція веб-додатку з базою даних є ключовим аспектом реалізації, оскільки забезпечує збереження, доступ і обробку інформації, необхідної для функціонування усіх підсистем. У рамках реалізації було використано різні типи баз даних, відповідно до технологій, що лежать в основі кожної з підсистем.

3.3.1 База даних системи бронювання (Flask + SQLite)

Для підсистеми бронювання столиків було обрано реляційну базу даних SQLite як легку, вбудовану та ефективну СУБД для невеликих веб-сервісів. З'єднання з БД у Flask встановлюється за допомогою бібліотеки sqlite3.

```
SELECT * FROM reservations WHERE date = '2025-05-26' AND time = '19:00';
```

Рисунок 3.1 - Приклад SQL-запиту на отримання бронювань за певну дату

```
import sqlite3
def get_db_connection():
conn = sqlite3.connect('database.db')
conn.row_factory = sqlite3.Row
return conn
```

Рисунок 3.2 - Приклад з’єднання з БД у Flask

```
CREATE TABLE reservations (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    date TEXT NOT NULL,
    time TEXT NOT NULL,
    guests INTEGER NOT NULL,
    contact_info TEXT NOT NULL
);
```

Рисунок 3.3 - Структура таблиці SQL для бронювання

```
conn = get_db_connection()
conn.execute(
    'INSERT INTO reservations (name, date, time, guests, contact_info) VALUES (?, ?, ?,
    (user_name, date, time, guests, contact)
)
conn.commit()
conn.close()
```

Рисунок 3.4 - Приклад SQL-запиту на додавання нового бронювання

3.3.2 База даних підсистеми персоналізованих пропозицій

У цій підсистемі база даних зберігає історію замовлень користувачів, яка надалі використовується для навчання моделі персоналізованих рекомендацій.

Інтеграція з Python (через sqlite3 або SQLAlchemy) дозволяє витягувати дані безпосередньо перед обробкою алгоритмами машинного навчання. З’єднання аналогічне попередньому прикладу, з додатковим перетворенням у формат Pandas DataFrame для подальшої обробки.

```
CREATE TABLE order_history (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER NOT NULL,
    product_id INTEGER NOT NULL,
    order_date TEXT NOT NULL
);
```

Рисунок 3.5 - Структура таблиці історії замовлень

```
SELECT product_id FROM order_history WHERE user_id = 7 ORDER BY order_date DESC;
```

Рисунок 3.6 - Приклад запиту для отримання історії замовлень користувача

3.3.3 База даних WordPress/WooCommerce

WordPress і WooCommerce використовують власну реляційну СУБД — зазвичай MySQL або MariaDB. Доступ до неї забезпечується через REST API або безпосередньо через SQL, проте перевага надається API для збереження сумісності та безпеки.

Основні таблиці:

- wp_users — інформація про користувачів;
- wp_posts — замовлення оформлені через WooCommerce (тип shop_order);
- wp_postmeta — додаткові метадані замовлень;
- wp_woocommerce_order_items — склад замовлення (товари);
- wp_woocommerce_order_itemmeta — метаінформація товарів у замовленні.

```
GET https://example.com/wp-json/wc/v3/orders?customer=7
```

Рисунок 3.7 - Приклад коду інтеграції з Flask/AI-підсистемою

```sql
SELECT p.ID, p.post_date, pm.meta_value AS total
FROM wp_posts p
JOIN wp_postmeta pm ON p.ID = pm.post_id
WHERE p.post_type = 'shop_order'
AND p.post_status = 'wc-completed'
AND pm.meta_key = '_order_total'
AND p.post_author = 7
ORDER BY p.post_date DESC
LIMIT 5;
```

Рисунок 3.8 - Приклад SQL-запиту на отримання останніх замовлень користувача

3.3.4 Захист даних і безпека

Для забезпечення безпеки інтеграції реалізовано наступні заходи:

- усі з’єднання із зовнішніми базами або API здійснюються через HTTPS;
- дані, що передаються між підсистемами, шифруються або маркуються токенами авторизації;
- запити до баз даних параметризуються, що мінімізує ризик SQL-ін’єкцій.

У результаті реалізована надійна й гнучка модель взаємодії з базами даних для кожної підсистеми веб-додатку. Такий підхід забезпечує збереження цілісності даних, масштабованість архітектури та ефективність обробки інформації на всіх етапах функціонування системи.

3.4 Реалізація інтерфейсу користувача

Інтерфейс користувача (UI, від англ. User Interface) відіграє ключову роль у забезпеченні ефективної взаємодії кінцевого користувача з веб-додатком. Основною метою розробки UI у межах цього проєкту було створення інтуїтивно зрозумілого, адаптивного та візуально привабливого середовища для виконання основних бізнес-процесів ресторану: оформлення замовлень, бронювання столиків, перегляду персоналізованих пропозицій.

3.4.1 Використані технології

Інтерфейс веб-додатку реалізовано за допомогою класичних веб-технологій:

- **HTML5** — для побудови семантичної структури веб-сторінок;
- **CSS3** — для оформлення зовнішнього вигляду елементів;
- **JavaScript** — для реалізації клієнтської логіки, динамічних ефектів та обробки подій;
- **Bootstrap 5** — для прискореної адаптивної верстки та використання готових UI-компонентів. Використання Bootstrap дозволило забезпечити адаптивність інтерфейсу до різних розмірів екрану, що є важливим чинником з огляду на потребу підтримки мобільних користувачів.

3.4.2 Структура інтерфейсу скріни

Інтерфейс поділено на тематичні модулі відповідно до підсистем:

- **Головна сторінка —** містить загальну інформацію про ресторан, меню, кнопки навігації до підсистем;

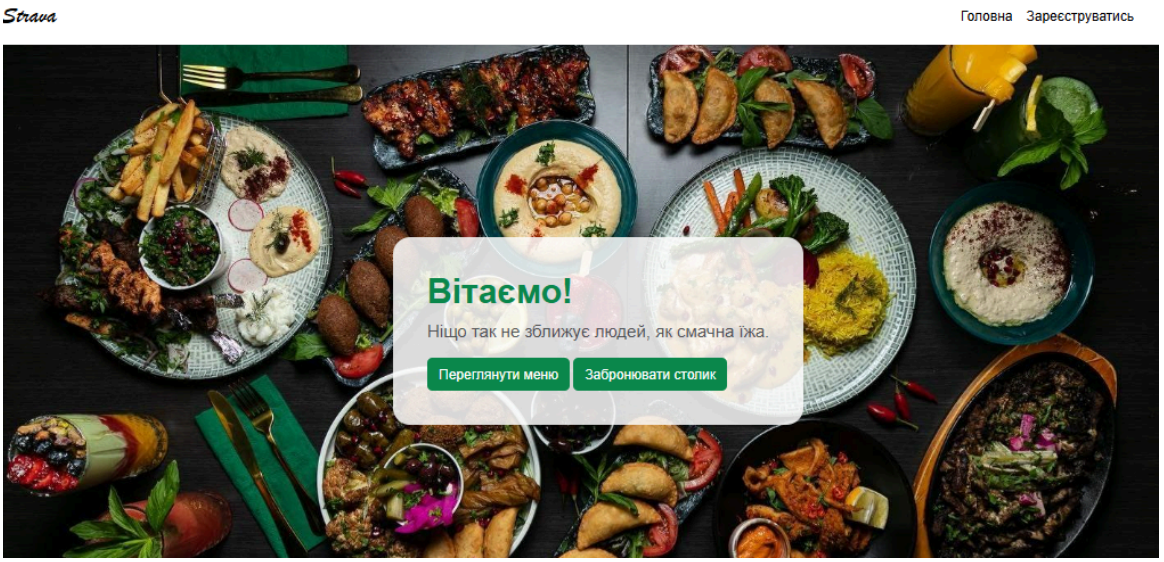

Рисунок 3.9 - Головна сторінка

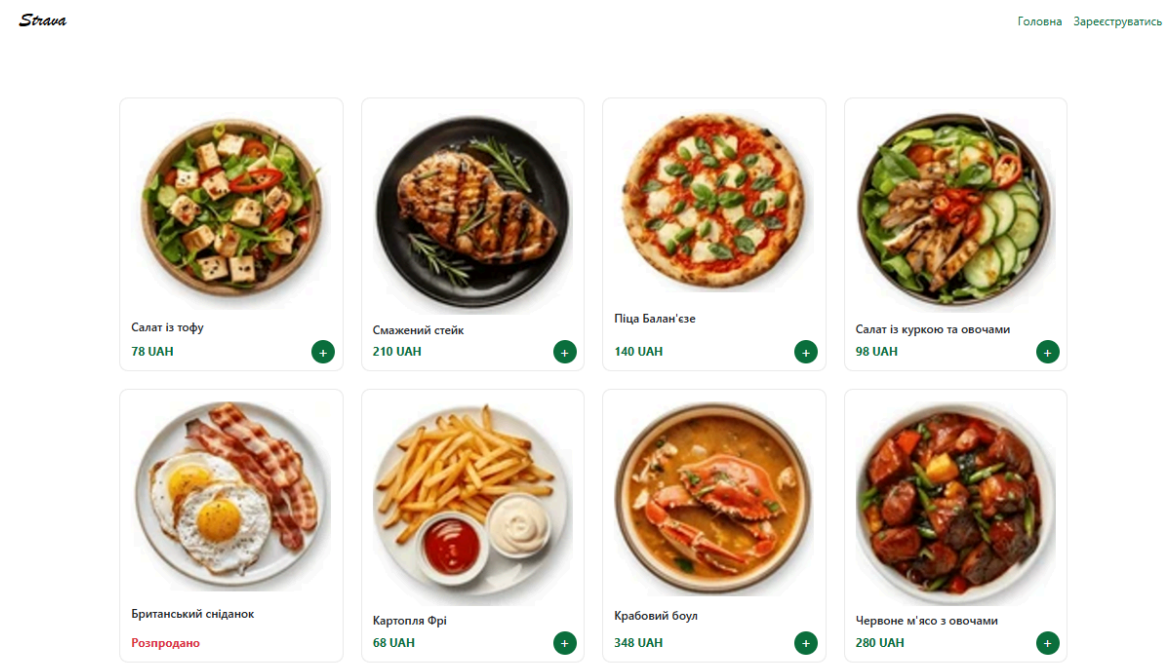

Рисунок 3.10 - Меню ресторану

– **Сторінка замовлень** (WordPress/WooCommerce) — реалізується через стандартний WooCommerce шаблон, модифікований за допомогою CSS для узгодження з загальним дизайном додатку. Та **секція для персоналізованих пропозицій** — забезпечує відображення згенерованих рекомендацій на основі замовлень користувача.

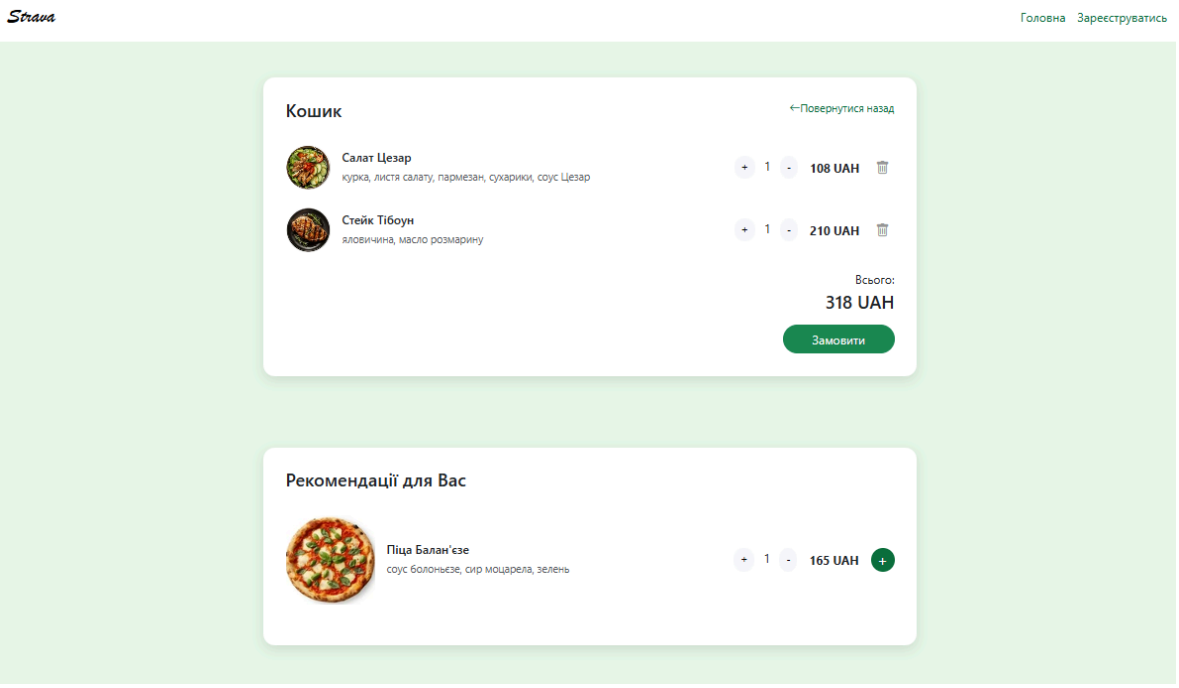

Рисунок 3.11 - Оформлення замовлення

- **Форма бронювання столиків** — реалізована у Flask за допомогою HTML-форм, з обробкою введення на стороні клієнта через JavaScript;

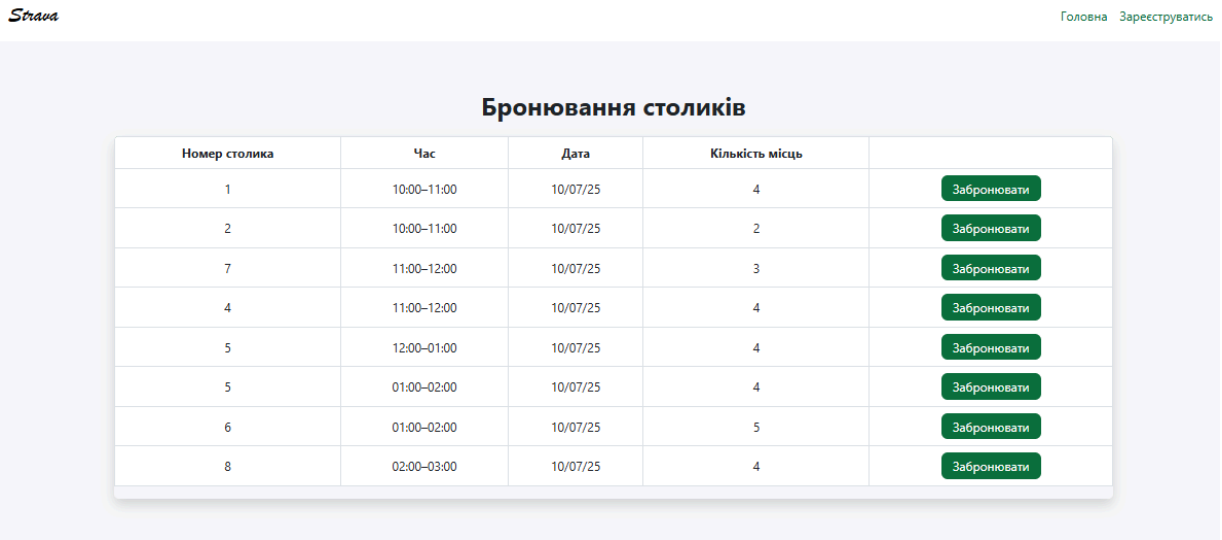

Рисунок 3.12 - Сторінка бронювання

3.4.3 Особливості взаємодії з користувачем

Користувацький досвід було оптимізовано з урахуванням принципів UX-дизайну:

- **Валідація форм** - усі форми (наприклад, бронювання) містять як HTML-валидацію (required, type="email", тощо), так і JavaScript-перевірки, які попереджають користувача про помилки введення до надсилання запиту.
- **Зворотній зв'язок** - після оформлення замовлення чи бронювання користувач отримує повідомлення про успішну дію (наприклад, модальне вікно або сповіщення на сторінці).
- **Динамічне завантаження контенту** - сторінка персоналізованих рекомендацій використовує AJAX-запити для отримання даних без перезавантаження сторінки, що покращує зручність використання.

- **Адаптивність** - усі основні елементи UI (меню, кнопки, таблиці) адаптуються до ширини екрана завдяки класам Bootstrap (container-fluid, col-sm, d-flex, тощо).

3.4.4 Приклади реалізації

index.html × style.css

```
<form action="/reserve" method="post" class="p-4 shadow-sm bg-light rounded">
  <div class="mb-3">
    <label for="name" class="form-label">Ім'я</label>
    <input type="text" class="form-control" id="name" name="name" required>
  </div>
  <div class="mb-3">
    <label for="date" class="form-label">Дата</label>
    <input type="date" class="form-control" id="date" name="date" required>
  </div>
  <div class="mb-3">
    <label for="guests" class="form-label">Кількість гостей</label>
    <input type="number" class="form-control" id="guests" name="guests" min="1" required>
  </div>
  <button type="submit" class="btn btn-primary">Забронювати</button>
</form>
```

Рисунок 3.13 - Фрагмент коду для форми бронювання (HTML + Bootstrap)

index.js ×

```
1  document.querySelector( selectors: "form").addEventListener( type: "submit",
2      const guests = document.getElementById( elementId: "guests").value;
3      if (guests <= 0) {
4          alert("Кількість гостей має бути більшою за 0.");
5          event.preventDefault();
6      }
7  });
```

Рисунок 3.14 - Фрагмент коду для JS-перевірки форми (JavaScript)

3.4.5 Роль інтерфейсу в забезпеченні злагодженої роботи підсистем

Інтерфейс відіграє роль інтеграційного середовища для трьох підсистем, надаючи користувачеві єдиний доступ до всіх функцій системи. Візуальна

уніфікація стилю сторінок (незалежно від того, чи йдеться про сторінки Flask, WordPress чи AI-підсистему) забезпечує цілісне сприйняття додатку.

Таким чином, реалізація інтерфейсу користувача у веб-додатку базується на сучасних веб-технологіях і практиках UX/UI-дизайну, забезпечуючи зручність, швидкість та ефективність взаємодії користувача з системою.

3.5 API та серверна логіка

Для забезпечення ефективної взаємодії між підсистемами веб-додатку, а також реалізації логіки обробки даних на стороні сервера, у системі застосовується REST API (Representational State Transfer Application Programming Interface). Він забезпечує стандартизовану архітектуру обміну даними між клієнтською частиною (інтерфейсом користувача) та серверною логікою, а також між окремими підсистемами додатку.

3.5.1 Архітектура REST API

REST API побудовано згідно з принципами RESTful-архітектури:

- використання HTTP-методів: GET, POST, PUT, DELETE;
- ідентифікація ресурсів за URI;
- обмін даними у форматі JSON;
- клієнт-серверна модель;
- відсутність стану на стороні сервера (stateless).

API реалізовано з використанням фреймворку Flask (Python), що дозволяє швидко створювати ендпоїнти, обробляти запити та повертати відповіді у форматі JSON.

3.5.2 Серверна логіка

Серверна частина виконує низку критично важливих функцій:

- зберігання, обробка та фільтрація даних;
- перевірка автентичності та авторизація користувачів;

- формування відповідей згідно з бізнес-логікою;
- взаємодія з базою даних та передача даних клієнтській частині через API.

Особливістю реалізації є модульна побудова серверної логіки для кожної з підсистем: бронювання, рекомендацій, авторизації тощо.

3.5.3 Основні ендпоїнти

Нижче наведено приклади основних ендпоїнтів, які забезпечують функціональність додатку - авторизація (аутентифікація користувача з перевіркою логіна й пароля), бронювання столиків (перегляд вільних місць та бронювання) та перегляд персоналізованих пропозицій.

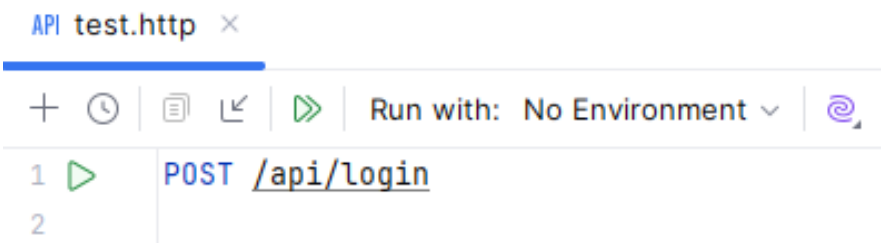

Рисунок 3.15 - POST запит для авторизації

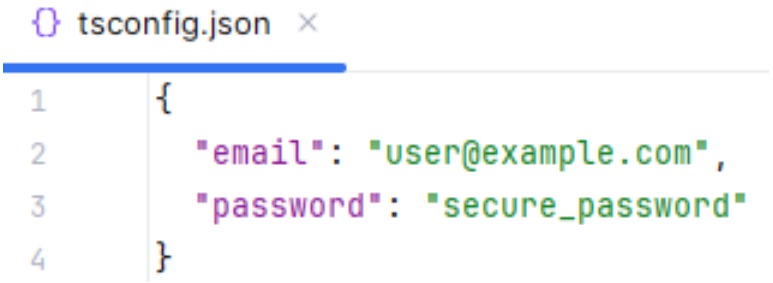

Рисунок 3.16 - Тіло запиту (JSON) для авторизації

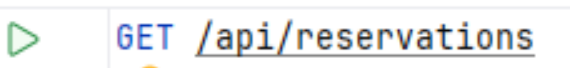

Рисунок 3.17 - Фрагмент коду для отримання усіх бронювань

Рисунок 3.18 - Фрагмент коду для створення нового бронювання

```
{
  "name": "Іван",
  "date": "2025-05-30",
  "guests": 3,
  "phone": "+380991234567"
}
```

Рисунок 3.19 - Тіло запиту (JSON) для бронювання

Рисунок 3.20 - Фрагмент коду для отримання пропозицій

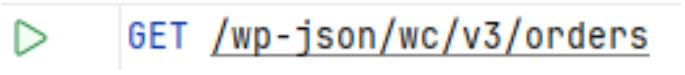

Рисунок 3.21 - Фрагмент коду для синхронізації з WooCommerce

3.5.4 Обробка помилок та валідація

Кожен ендпоїнт містить механізми перевірки вхідних даних та обробки можливих помилок. Наприклад, у випадку спроби створити бронювання на минулу дату сервер поверне помилку 400:

```
{
  "error": "Дата бронювання не може бути в минулому"
}
```

Рисунок 3.22 - Відповідь (JSON)

3.5.5 Безпека API

Для захисту REST API реалізовано такі заходи:

- перевірка автентичності через токени (JWT);
- обмеження доступу до захищених маршрутів;
- використання HTTPS-протоколу для передачі даних;
- санітизація вхідних даних з метою запобігання SQL-ін'єкціям та XSS-атакам.

Таким чином, REST API забезпечує злагоджену роботу усіх компонентів системи, дозволяючи обмінюватися даними між підсистемами та клієнтським інтерфейсом. Серверна логіка, реалізована у Flask, виконує критичні функції бізнес-обробки, валідації та безпечної взаємодії з базами даних і зовнішніми сервісами.

3.6 Безпека та авторизація

У сучасних веб-додатках питання безпеки та контролю доступу є надзвичайно актуальними, оскільки порушення конфіденційності чи несанкціонований доступ можуть призвести до витоку персональних даних користувачів або компрометації бізнес-логіки. У рамках розробленого веб-додатку для автоматизації бізнес-процесів ресторану реалізовано комплексний підхід до забезпечення безпеки та системи авторизації користувачів.

3.6.1 Авторизація через WordPress

Система автентифікації користувачів реалізована з використанням вбудованого механізму авторизації платформи WordPress, яка є основою для підсистеми оформлення замовлень. WordPress забезпечує надійний рівень безпеки за рахунок перевіреної системи управління користувачами, що включає:

- хешування паролів з використанням bcrypt;
- захист від brute-force атак (через плагіни типу Limit Login Attempts);
- можливість розмежування прав доступу (ролі: адміністратор, клієнт, менеджер тощо).

Користувачі проходять автентифікацію через стандартну форму входу WordPress. Після авторизації сеанс зберігається за допомогою cookie-файлів або токенів доступу (при роботі з REST API). Додатково застосовуються CAPTCHA або двофакторна автентифікація (2FA) за потреби.

3.6.2 Реалізація захисту даних

Передача всіх даних між клієнтом та сервером здійснюється через захищене з'єднання HTTPS, що запобігає перехопленню інформації (наприклад, логінів і паролів) при передачі через мережу.

Також на рівні бази даних реалізовано принцип мінімально необхідного доступу. Кожна підсистема взаємодіє з БД лише в межах своїх дозволів, що зменшує потенційні ризики у разі компрометації одного з компонентів.

3.6.3 Валідація даних

Для запобігання атакам, пов’язаним із впровадженням шкідливого коду (SQL-ін’єкції, XSS), на всіх рівнях додатку реалізовано валідацію та санітизацію вхідних даних:

- на клієнтському рівні – за допомогою JavaScript (перевірка формату, довжини, типів полів);
- на серверному рівні – за допомогою вбудованих функцій Flask або WordPress, зокрема sanitize_text_field(), esc_html(), filter_input() тощо.

Використання параметризованих запитів або ORM (Object-Relational Mapping) гарантує безпечну взаємодію з базою даних.

3.6.4 Контроль доступу та права користувачів

У системі передбачено рольову модель доступу, що дозволяє обмежувати або надавати права на виконання певних дій. Наприклад:

- звичайні користувачі можуть переглядати меню, оформлювати замовлення, бронювати столики;

- адміністратори мають доступ до аналітики, редагування меню, обробки замовлень;
- модуль персоналізованих пропозицій доступний лише авторизованим користувачам з історією замовлень.

Перевірка прав виконується під час кожного запиту до серверної частини або при зверненні до REST API.

3.6.5 Захист REST API

REST API, реалізоване у Flask, захищено за допомогою авторизаційних токенів (JWT). Після успішного логіну клієнту повертається токен, який додається до заголовку всіх наступних запитів. Усі захищені ендпоїнти перевіряють дійсність токена перед обробкою запиту.

Таким чином, у межах розробки веб-додатку реалізовано багаторівневу систему безпеки, що охоплює автентифікацію користувачів, шифрування переданих даних, контроль доступу до ресурсів та захист серверної логіки. Поєднання можливостей платформи WordPress та кастомних механізмів безпеки у Flask забезпечує високий рівень надійності веб-додатку.

3.7 Тестування додатку

Тестування є невіддільною частиною процесу розробки програмного забезпечення, що забезпечує виявлення помилок, підвищення надійності та відповідність функціоналу вимогам. Для веб-додатку з автоматизації бізнес-процесів ресторану було проведено комплексне тестування з метою перевірки коректності роботи трьох основних підсистем: оформлення замовлень, бронювання столиків та персоналізованих пропозицій. Окрема увага була приділена також перевірці авторизації та взаємодії з базою даних.

3.7.1 Види тестування

У рамках проєкту застосовувалися такі основні види тестування:

- **Функціональне тестування**

  Перевірялося виконання ключових функцій додатку згідно з технічними вимогами. Тестувалися сценарії створення та обробки замовлень, резервування столиків, авторизації користувачів, отримання персоналізованих пропозицій тощо. Усі функції були протестовані як з боку адміністратора, так і з боку користувача.

- **Тестування інтерфейсу користувача (UI-тестування)**

  Оцінювалася коректність відображення елементів інтерфейсу, їх взаємодія, адаптивність та зручність використання. Перевірялися форми входу, замовлень, календар бронювання, елементи меню тощо. Тестування проводилося вручну в різних браузерах (Google Chrome, Mozilla Firefox, Microsoft Edge).

- **Тестування API (інтеграційне тестування)**

  Для підсистеми бронювання на Flask та підсистеми персоналізованих пропозицій проводилось тестування REST API. Тестувалися коректність відповіді серверу, обробка помилкових запитів, час відповіді, а також поведінка системи при відсутності авторизаційного токена.

- **Тестування безпеки**

  Було перевірено стійкість форм до SQL-ін’єкцій та XSS-атак, а також коректність обробки вхідних даних. Застосовувались інструменти автоматичної валідації та ручні сценарії з навмисно некоректними даними.

- **Кросбраузерне та адаптивне тестування**

  Перевірялася коректність відображення інтерфейсу на різних пристроях (десктоп, планшет, смартфон) та в різних браузерах. Використовувалися інструменти інспекції елементів (DevTools), а також середовище BrowserStack.

3.7.2 Інструменти тестування

Для реалізації тестування використовувалися як ручні, так і автоматизовані засоби:

- **Postman** — для тестування REST API, перевірки відповіді серверу, авторизації через токени, роботи з параметрами запиту.
- **Selenium WebDriver** — для автоматизації UI-тестів: тестування сценаріїв входу, оформлення замовлень, взаємодії з формами.
- **pytest** — для написання юніт-тестів у підсистемі на Flask. Перевірялася логіка обробки запитів, функції маршрутизації, взаємодія з базою даних.
- **WordPress Site Health & Query Monitor** — для тестування навантаження, запитів до бази, виявлення помилок у плагінах.
- **HTML/CSS validators** — для перевірки валідності коду інтерфейсу.

3.7.3 Результати тестування

У результаті тестування було виявлено та усунуто декілька логічних помилок, зокрема:

- неправильне відображення помилок при некоректному бронюванні (виправлено валідацію дати);
- відсутність перевірки токена у деяких API-запитах (додано JWT-перевірку);
- дублювання персоналізованих пропозицій для одного користувача (оптимізовано логіку обробки історії замовлень).

Усі ключові функціональні сценарії пройшли перевірку, що дозволяє вважати систему працездатною, стабільною та готовою до подальшого розгортання або впровадження.

У даному розділі було детально розглянуто процес реалізації веб-додатку для автоматизації бізнес-процесів ресторану, який охоплює три ключові підсистеми: оформлення замовлень (на базі WordPress і WooCommerce), бронювання столиків (на Flask), а також модуль персоналізованих пропозицій, реалізований із використанням технологій штучного інтелекту. Проведено опис

загальної структури проєкту, наведено приклади організації файлів, директорій та модулів для кожної підсистеми.

Розкрито логіку функціонування кожного функціонального блоку: від авторизації користувачів до обробки запитів REST API. Показано способи взаємодії з базою даних, зокрема механізми підключення, структуру таблиць, приклади SQL-запитів та заходи з безпеки доступу. Особливу увагу приділено реалізації користувацького інтерфейсу з використанням HTML, CSS, JavaScript і фреймворку Bootstrap, що забезпечує адаптивність та зручність у користуванні.

Також у межах цього розділу було висвітлено реалізацію серверної логіки, побудову API та захист усіх компонентів системи. Окремим етапом проведено тестування системи, що включає функціональні, інтеграційні, UI- та безпекові перевірки, з використанням сучасних інструментів, таких як Postman, Selenium та pytest.

Таким чином, розділ демонструє цілісну та технічно обґрунтовану реалізацію веб-додатку, яка відповідає вимогам до сучасних інформаційних систем, забезпечує автоматизацію ключових бізнес-процесів ресторану та є готовою до практичного застосування.

Розділ 4. ОЦІНКА ЕФЕКТИВНОСТІ ТА ПЕРСПЕКТИВИ РОЗВИТКУ

Успішне впровадження інформаційних технологій у сферу ресторанного бізнесу вимагає не лише розробки та реалізації функціонального програмного забезпечення, але й подальшої оцінки його впливу на діяльність закладу. Даний розділ присвячений аналізу ефективності створеного веб-додатку та розгляду перспектив його розвитку в умовах цифровізації обслуговування.

Оцінювання результатів автоматизації є важливою складовою проєктного циклу, оскільки дозволяє виявити сильні сторони системи, визначити її вплив на ключові бізнес-показники та сформулювати напрямки вдосконалення. У межах цього розділу буде представлено порівняння стану до та після впровадження веб-додатку, визначено основні переваги використання розробленої системи, а також окреслено можливі шляхи її подальшого розвитку з урахуванням новітніх технологій та змін у потребах користувачів.

4.1 Порівняння до та після автоматизації

Автоматизація бізнес-процесів у сфері ресторанного обслуговування є важливою умовою підвищення ефективності роботи закладу, поліпшення клієнтського досвіду та оптимізації витрат. Розроблений веб-додаток, що складається з трьох інтегрованих підсистем — оформлення замовлень (на основі WordPress та WooCommerce), бронювання столиків (реалізовано за допомогою Flask) та персоналізованих пропозицій на основі технологій штучного інтелекту, дозволяє суттєво трансформувати ключові аспекти взаємодії клієнтів із рестораном.

До впровадження веб-додатку основні процеси обслуговування здійснювалися вручну або із застосуванням фрагментованих інструментів, не пов’язаних в єдину систему. Оформлення замовлень вимагало фізичної присутності клієнта або телефонного дзвінка, що часто призводило до помилок у передачі інформації, затримок обслуговування та зростання навантаження на

персонал. Процес бронювання столиків також здійснювався вручну, що не дозволяло контролювати завантаженість залу в реальному часі та ускладнювало управління резерваціями. Відсутність персоналізованих пропозицій обмежувала можливості індивідуального підходу до клієнтів і знижувала рівень їх задоволеності.

Після інтеграції веб-додатку спостерігається якісне покращення функціонування ресторану. Онлайн-оформлення замовлень через WooCommerce дозволило скоротити час взаємодії клієнта з персоналом, зменшити кількість помилок під час прийому замовлень та створити зручний інтерфейс для перегляду меню. Підсистема бронювання, реалізована на Python (Flask), забезпечує динамічне керування наявністю вільних столиків, автоматичну обробку резервацій і надсилання підтверджень клієнтам, що значно знижує адміністративне навантаження.

Окрема перевага — впровадження підсистеми персоналізованих пропозицій із використанням методів штучного інтелекту. Завдяки аналізу історії замовлень та поведінкових моделей користувачів, система формує індивідуальні рекомендації, що сприяє підвищенню рівня лояльності клієнтів та зростанню середнього чека.

Узагальнюючи, автоматизація бізнес-процесів ресторану за допомогою розробленого веб-додатку дозволила досягти суттєвих покращень у трьох основних напрямах: швидкість обслуговування, зменшення помилок та підвищення якості взаємодії з клієнтами. Таблиця 4.1 ілюструє ключові зміни, що відбулися після впровадження системи.

Таблиця 4.1 - До та після автоматизації

Критерій	**До автоматизації**	**Після автоматизації**
Оформлення замовлень	Телефон/усно, вручну	Онлайн через WooCommerce
Помилки у замовленнях	Часті (через людський фактор)	Зведені до мінімуму (автоматичне підтвердження)

Бронювання столиків	Телефон/вручну	Онлайн через Flask-додаток
Управління завантаженістю залу	Вручну, без синхронізації	Автоматизоване, в режимі реального часу
Індивідуальні пропозиції	Відсутні	Формуються на основі ШІ-аналізу
Залучення клієнтів	Пасивне	Активне через персоналізовані пропозиції
Час обслуговування	Від 10–15 хв на замовлення	3–5 хв через онлайн-систему

Таким чином, розроблений веб-додаток не лише автоматизує окремі процеси, а й створює єдину інтегровану систему управління взаємодією з клієнтами, що закладає основу для подальшого цифрового розвитку ресторану.

4.2 Переваги використання розробленого додатку

Впровадження розробленого веб-додатку забезпечило ресторанному закладу низку стратегічних та операційних переваг, що значно підвищили ефективність його функціонування. Завдяки комплексному підходу до автоматизації бізнес-процесів, система сприяє оптимізації внутрішньої роботи персоналу, покращенню клієнтського досвіду та зміцненню конкурентоспроможності закладу на ринку послуг громадського харчування.

По-перше, інтеграція підсистеми онлайн-замовлень на основі CMS WordPress у поєднанні з плагіном WooCommerce дозволила клієнтам здійснювати вибір страв, переглядати їх склад, ціни та здійснювати замовлення дистанційно. Це значно підвищило зручність та швидкість взаємодії, знизивши потребу у залученні персоналу для обробки кожного запиту. Автоматичне формування чеків, синхронізація з наявністю страв та можливість оплати онлайн мінімізують помилки та спрощують бухгалтерський облік.

По-друге, підсистема бронювання столиків, реалізована з використанням фреймворку Flask, забезпечує гнучке та динамічне управління розсадженням клієнтів. Користувачі отримали можливість самостійно обирати дату, час та кількість місць для бронювання, що знижує навантаження на адміністраторів та мінімізує кількість конфліктних ситуацій, пов’язаних із дублюванням або неправильним розподілом місць. Система також може надсилати автоматичні підтвердження та нагадування, що додатково підвищує рівень організованості та зручності.

По-третє, впровадження інтелектуального модуля персоналізованих пропозицій є важливою інноваційною складовою, яка відрізняє розроблений додаток від типових систем автоматизації. За допомогою аналізу даних про попередні замовлення, уподобання клієнтів та поведінкові патерни, система формує індивідуальні рекомендації щодо страв, спеціальних пропозицій або акцій. Такий підхід сприяє глибшому залученню клієнтів, формуванню довгострокової лояльності та підвищенню середнього чека.

Окремо слід зазначити ще кілька важливих переваг:

- **Масштабованість**: архітектура додатку дозволяє легко адаптувати систему до потреб інших ресторанів чи розширювати її функціонал;
- **Уніфікованість**: всі три підсистеми інтегруються в єдину платформу, що забезпечує цілісне управління бізнес-процесами;
- **Зменшення витрат**: скорочення витрат на персонал, зменшення кількості помилок та підвищення точності обліку позитивно впливають на фінансові показники закладу;
- **Аналітична підтримка**: дані, що збираються через додаток, можуть бути використані для аналізу попиту, планування закупівель та маркетингових стратегій.

Таким чином, розроблений веб-додаток є не лише інструментом автоматизації, але й засобом стратегічного управління, що забезпечує

багатовимірні переваги для якості обслуговування, внутрішньої організації та довгострокового розвитку ресторанного бізнесу.

4.3 Можливі напрями подальшого розвитку

Хоча розроблений веб-додаток уже надає повноцінне рішення для автоматизації ключових бізнес-процесів ресторану, існує низка перспективних напрямів, які можуть забезпечити його подальший розвиток та розширення функціональних можливостей. Подальше удосконалення системи спрямоване не лише на оптимізацію внутрішніх операцій, але й на поглиблення аналітичного потенціалу, підвищення рівня персоналізації та розширення каналів взаємодії з клієнтами.

Одним із першочергових напрямів розвитку є інтеграція мобільного додатку як доповнення до веб-версії. Мобільна платформа забезпечить ще вищу доступність для користувачів, дозволяючи здійснювати замовлення, бронювання та перегляд персоналізованих пропозицій у зручному форматі з будь-якого пристрою. Крім того, мобільний додаток може підтримувати push-сповіщення, які сприятимуть більш оперативному інформуванню клієнтів про акції, зміну меню чи підтвердження замовлень.

Другим перспективним кроком є впровадження модуля аналітики для менеджменту, який дозволить адміністрації ресторану відстежувати ключові бізнес-індикатори у реальному часі: обсяги продажів, найпопулярніші страви, часову динаміку відвідувань тощо. Побудова аналітичних панелей (dashboard) на основі зібраних даних сприятиме ухваленню більш обґрунтованих управлінських рішень, ефективнішому плануванню закупівель та оптимізації графіків персоналу.

У сфері штучного інтелекту доцільним є розширення персоналізованого функціоналу за рахунок додавання прогнозних моделей поведінки клієнтів. Наприклад, система може не лише рекомендувати страви на основі минулого досвіду, але й прогнозувати ймовірність повторного візиту, обирати найкращий

час для надсилання промоцій чи формувати динамічні знижки на основі активності клієнта.

Окремим напрямом розвитку є забезпечення багатомовної підтримки інтерфейсу, що є особливо актуальним для туристично привабливих регіонів. Можливість взаємодії з додатком різними мовами сприятиме зростанню клієнтської бази та підвищенню рівня задоволеності іноземних відвідувачів.

Нарешті, розвиток системи в частині автоматизації внутрішніх процесів ресторану — таких як управління запасами, контроль за графіками персоналу, моніторинг постачань — дозволить перетворити веб-додаток на повноцінну ERP-платформу, що охоплює як клієнтську, так і операційну частину діяльності ресторану.

Таким чином, розроблений веб-додаток має великий потенціал для масштабування та вдосконалення. Орієнтація на мобільність, аналітику, персоналізацію та інтеграцію із зовнішніми сервісами сприятиме трансформації додатку в багатофункціональну екосистему, здатну підтримувати гнучке та ефективне управління ресторанним бізнесом в умовах сучасного цифрового середовища.

4.4 Оцінка економічної ефективності

Оцінка економічної ефективності розробленого веб-додатку є важливим етапом аналізу його практичного впровадження у діяльність ресторанного бізнесу. Вона дозволяє не лише кількісно визначити отриманий економічний ефект, але й оцінити доцільність подальших інвестицій у розвиток цифрової інфраструктури закладу. Основними критеріями, що використовуються для такої оцінки, є зменшення витрат, зростання прибутку, підвищення продуктивності праці, а також скорочення часу на виконання бізнес-процесів.

Одним із найбільш помітних результатів автоматизації є зниження витрат на оплату праці. Після впровадження підсистеми онлайн-замовлень і бронювання потреба у значній кількості обслуговуючого персоналу (операторів,

адміністраторів, касирів) зменшилася. Наприклад, у випадку середнього ресторану з щоденним потоком замовлень у 80–100 клієнтів, система здатна автоматично обробити до 70% операцій, які раніше виконувалися вручну. Це дозволяє зменшити кількість змін персоналу або скоротити потребу у понаднормовій роботі, що дає змогу економити в середньому 10–15% від місячного фонду оплати праці, що еквівалентно 5–10 тис. грн на місяць.

Додатковий економічний ефект забезпечується зростанням прибутку за рахунок зручності користування сервісом, що стимулює повторні покупки та збільшує середній чек. Інтелектуальний модуль персоналізованих пропозицій формує цільові рекомендації для клієнтів, що сприяє продажу додаткових позицій (десертів, напоїв, акційних страв). За оцінками, середній чек збільшився на 8–12%, що у сукупності з ростом кількості онлайн-замовлень на 20–25% дозволило підвищити місячний виторг у середньому на 15–20 тис. грн.

Ще одним аспектом є зменшення непродуктивних втрат (наприклад, помилок у замовленнях, дублювання бронювань, конфліктів при обслуговуванні). За рахунок точного обліку замовлень, автоматичних підтверджень та інтерфейсної перевірки даних, кількість інцидентів такого характеру зменшилась, що позитивно позначається на репутації закладу та знижує приховані витрати, пов’язані з поверненнями, скаргами або недоотриманим прибутком.

У сукупності, результати впровадження веб-додатку свідчать про відчутне підвищення економічної ефективності, що проявляється у:

- щомісячному зменшенні операційних витрат;
- стабільному зростанні доходу завдяки підвищенню лояльності клієнтів;
- покращенні обліку та внутрішньої організації роботи;
- швидкому поверненні інвестицій у розробку системи (окупність можлива протягом 4–6 місяців).

Таким чином, автоматизація бізнес-процесів за допомогою розробленого веб-додатку забезпечує економічно обґрунтоване та довгостроково вигідне

рішення для розвитку ресторанного бізнесу в умовах високої конкуренції та зростаючої цифрової взаємодії з клієнтами.

4.5 Аналіз ризиків та обмежень

Незважаючи на численні переваги впровадження веб-додатку для автоматизації бізнес-процесів ресторану, важливо провести комплексний аналіз можливих ризиків і обмежень, пов’язаних із його функціонуванням та подальшим розвитком. Такий аналіз дозволяє завчасно виявити потенційні загрози, які можуть вплинути на ефективність системи, надійність її роботи або загальну рентабельність проекту.

Одним з основних ризиків є технічні збої або відмова в роботі підсистем, зокрема, через оновлення плагінів WordPress/WooCommerce, помилки у Flask-сервері або збої в роботі модуля штучного інтелекту. Такі збої можуть призвести до втрати замовлень, порушення логіки бронювання або некоректного формування персоналізованих пропозицій. Для мінімізації цих ризиків доцільним є впровадження системи моніторингу продуктивності та резервного копіювання, а також регулярне тестування і оновлення компонентів.

Збирання та обробка персональних даних клієнтів (імена, номери телефонів, історія замовлень) підвищує вимоги до захисту даних. Потенційні вразливості у веб-додатку можуть стати мішенню для атак (SQL-ін’єкції, міжсайтове скриптування тощо), що загрожує як конфіденційності даних, так і репутації ресторану. Усунення цих ризиків вимагає використання сучасних механізмів автентифікації, шифрування, регулярного аудиту безпеки та відповідності вимогам законодавства щодо захисту персональних даних (наприклад, GDPR).

Одним з обмежень є опір персоналу до змін, особливо серед працівників, які звикли до традиційних підходів у роботі. Недостатній рівень цифрової грамотності може стати бар’єром для ефективного використання системи. Це обмеження може бути усунене шляхом організації навчальних сесій для

працівників та залучення внутрішніх «агентів змін» — персоналу, який підтримує цифрові інновації.

Оскільки додаток функціонує в онлайн-середовищі, нестабільне інтернет-з’єднання може тимчасово унеможливити використання окремих сервісів (наприклад, бронювання або онлайн-замовлення). Для критичних модулів можливо передбачити локальні копії або офлайн-режими, що дозволять мінімізувати ризики втрачених даних при тимчасових перебоях у мережі.

Зі зростанням кількості користувачів або розширенням функціоналу можуть виникати обмеження продуктивності, особливо якщо початково обрані технології не були розраховані на високе навантаження. Заздалегідь передбачене масштабування серверної частини та використання хмарної інфраструктури (наприклад, AWS, Heroku) дозволяє уникнути вузьких місць у продуктивності.

Інтеграція з зовнішніми платформами (служби доставки, платіжні шлюзи, CRM-системи) може ускладнюватися через різні формати API, нестабільність сторонніх сервісів або зміну політик доступу. Для пом’якшення цих обмежень рекомендується застосовувати стандартизовані протоколи взаємодії (REST, GraphQL), а також гнучку архітектуру з мікросервісами.

Таким чином, хоча впровадження веб-додатку відкриває широкі можливості для автоматизації ресторанного бізнесу, на кожному етапі розробки та експлуатації важливо враховувати потенційні ризики. Своєчасна ідентифікація обмежень та застосування превентивних заходів дозволяє не лише уникнути збоїв, але й підвищити загальну стійкість, безпеку й ефективність цифрового рішення.

У результаті проведеної оцінки ефективності розробленого веб-додатку для автоматизації бізнес-процесів ресторану встановлено, що впровадження даної системи значно підвищує загальний рівень цифровізації та оптимізації діяльності закладу. Зіставлення показників до і після автоматизації демонструє зменшення навантаження на персонал, зниження операційних витрат, а також зростання швидкості обслуговування клієнтів. Крім того, система сприяє поліпшенню якості

взаємодії з клієнтами за рахунок персоналізованих рекомендацій на основі алгоритмів штучного інтелекту.

Проведений аналіз підтвердив наявність вагомих переваг від використання додатку: зручність онлайн-замовлень, автоматизація бронювання столиків, підвищення задоволеності клієнтів, а також зростання прибутку через підвищення середнього чеку та збільшення кількості повторних замовлень. Додаток забезпечує прозору та керовану структуру внутрішніх процесів, що дозволяє оперативно приймати управлінські рішення на основі достовірних даних.

Водночас, виявлено певні обмеження та ризики, пов’язані з технічною надійністю, безпекою персональних даних, можливістю масштабування, людським фактором і залежністю від стабільності інтернет-з’єднання. Однак завдяки вчасній діагностиці та впровадженню превентивних заходів, ці ризики можуть бути мінімізовані.

Щодо перспектив розвитку, проєкт має потенціал до масштабування та розширення функціональності, зокрема через інтеграцію з CRM-системами, мобільними застосунками, сервісами доставки та аналітичними платформами для глибшої роботи з клієнтськими даними.

Загалом, веб-додаток зарекомендував себе як ефективний інструмент цифрової трансформації ресторанного бізнесу. Його подальший розвиток у поєднанні з системним управлінням ризиками забезпечить сталу конкурентну перевагу підприємства в умовах сучасного ринку.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було реалізовано веб-додаток для автоматизації бізнес-процесів ресторану з інтеграцією елементів штучного інтелекту. Дослідження охоплювало як технічні, так і прикладні аспекти проєктування та впровадження систем автоматизації в ресторанній сфері. Згідно з поставленими завданнями можна сформулювати такі висновки:

1. Проведено аналіз сучасних систем автоматизації ресторанного бізнесу, виявлено їхні функціональні обмеження, зокрема щодо персоналізації взаємодії з клієнтами. Це підтвердило актуальність розробки нового рішення з можливістю адаптації під індивідуальні потреби користувачів.

2. Досліджено сучасні технології штучного інтелекту та їхнє застосування в ресторанному бізнесі. Виокремлено ті підходи, які найбільше відповідають задачам персоналізованих рекомендацій і покращення користувацького досвіду, зокрема методи класифікації, фільтрації та сегментації користувачів.

3. Визначено функціональні вимоги до веб-додатку, зокрема до трьох основних підсистем: замовлення страв, бронювання столиків та формування персоналізованих пропозицій. Особлива увага приділялася гнучкості та масштабованості системи, можливості інтеграції з іншими сервісами.

4. Розроблено архітектуру веб-додатку, що базується на модульному принципі. Для кожної з трьох підсистем реалізовано відповідні компоненти: підсистема замовлення страв — на базі CMS WordPress з плагіном WooCommerce; підсистема бронювання столиків — на основі фреймворку Flask з використанням SQLite; підсистема персоналізованих рекомендацій — за допомогою алгоритмів класифікації в середовищі Python.

5. Реалізовано та протестовано базові алгоритми штучного інтелекту, спрямовані на формування персоналізованих пропозицій користувачам. У процесі моделювання враховувались параметри попередніх замовлень, поведінкові

шаблони та загальні уподобання клієнтів. Це дозволило забезпечити динамічну адаптацію рекомендацій до потреб окремих користувачів.

6. Проведено функціональне тестування веб-додатку в умовах моделювання бізнес-процесів. Оцінено ефективність його роботи, швидкодію, стабільність та зручність користування. Результати показали, що система коректно обробляє запити, забезпечує взаємодію між підсистемами та дає релевантні персоналізовані пропозиції.

7. Сформовано перелік можливих напрямів подальшого вдосконалення веб-додатку. Зокрема, рекомендується впровадити глибші алгоритми машинного навчання, розширити базу знань про користувачів, додати механізми самооновлення моделей та інтеграцію з мобільними платформами для підвищення зручності користування.

Окрім досягнення основної мети, під час реалізації проєкту було отримано цінний досвід практичного використання сучасних технологій веб-розробки та штучного інтелекту. Зокрема, опанування особливості побудови клієнт-серверних архітектур, використання CMS та фреймворків, а також основи побудови моделей машинного навчання. Розроблений веб-додаток відповідає сучасним вимогам до подібного програмного забезпечення та має потенціал для подальшого комерційного застосування в індустрії ресторанного бізнесу.

Таким чином, поставлені у роботі завдання були повністю виконані, а розроблений веб-додаток підтвердив доцільність застосування технологій ШІ для покращення ефективності й персоналізації сервісів у сфері ресторанного бізнесу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Олійник, О. В., Шестакова, А. В., & Ярмолюк, Д. І. (2023). Напрями цифровізації ресторанного бізнесу. Економіка, управління та адміністрування, (1(103), 15–21. URL: https://doi.org/10.26642/ema-2023-1(103)-15-21 (дата звернення: 20.03.2025)

[2] Market size of the global online food delivery sector 2022–2027 [Electronic resource]. URL: https://cutt.ly/r3aQjCI (дата звернення: 20.03.2025)

[3] Deloitte “The future of restaurants: The new normal and beyond”. URL: https://www2.deloitte.com/us/en/pages/consumer-business/articles/future-of-restaurants-study.html (дата звернення: 20.03.2025)

[4] Toast “Should I Use a Third-Party Delivery Service or Create My Own?”. URL:https://pos.toasttab.com/blog/on-the-line/third-party-delivery-services?srsltid=AfmBOorJBX5l2GGf3T7uP1WMOweyKxX5WnkDQLUYGE5AkOlivMCKHaSR&utm_source=chatgpt.com (дата звернення: 25.03.2025)

[5] Glovo - Menu. URL: https://glovoapp.com/ua/en/kyiv-right-bank/ (дата звернення 25.03.2025)

[6] SevenRooms Website. URL: https://sevenrooms.com/ (дата звернення: 26.05.2025)

[7] Restoke Website. URL: https://www.restoke.ai/ (дата звернення: 26.05.2025)

[8] Bronze Vision Website. URL: https://bronze.vision/en/?lng=en (дата звернення: 05.04.2025)

[9] Slang.ai Website. URL: https://www.slang.ai/ (дата звернення: 05.04.2025)

[10] Unite.ai “7 найкращих інструментів штучного інтелекту для ресторанів”.URL: https://www.unite.ai/uk/%D0%BD%D0%B0%D0%B9%D0%BA%D1%80%D0%B0%D1%89%D1%96-%D1%96%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B8-%D1%88%D1%82%D1%83%D1%87%D0%

BD%D0%BE%D0%B3%D0%BE-%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82%D1%83-%D0%B4%D0%BB%D1%8F-%D1%80%D0%B5%D1%81%D1%82%D0%BE%D1%80%D0%B0%D0%BD%D1%96%D0%B2/ (дата звернення: 05.04.2025)

[11] LinkedIn profile Encounter AI. URL: https://www.linkedin.com/company/encounter-ai (дата звернення 05.04.2025)

[12] LandingList “Чому WordPress є найпопулярнішою CMS у світі?” URL: https://landinglist.com.ua/chomu-wordpress-ye-najpopulyarnishoyu-cms-u-sviti/ (дата звернення: 05.04.2025)

[13] Ukrainian Digital Community - “Конструктори сайтів: які бувають та як обрати”. URL: https://www.slang.ai/ (дата звернення: 20.04.2025)

[14] GloriaFood Website - “Конструктори сайтів: які бувають та як обрати”. URL: https://www.gloriafood.com/ (дата звернення: 20.04.2025)

[15] Wikipedia “Об'єктно-орієнтоване програмування”. URL: https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F (дата звернення: 20.04.2025)

[16] Visure "Функціональні та нефункціональні вимоги". URL: https://visuresolutions.com/uk/%D0%BF%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA-%D0%B7-%D0%B2%D1%96%D0%B4%D1%81%D1%82%D0%B5%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F-%D0%B2%D0%B8%D0%BC%D0%BE%D0%B3%D0%B0%D0%BC%D0%B8/%D1%84%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%96-%D1%82%D0%B0-%D0%BD%D0%B5%D1%84%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%

D1%8C%D0%BD%D1%96-%D0%B2%D0%B8%D0%BC%D0%BE%D0%B3%D0%B8/ (дата звернення: 03.05.2025)

[17] QATestLab - "Нефункціональні вимоги: приклади, типи, підходи". URL: https://training.qatestlab.com/blog/technical-articles/non-functional-requirements-examples-types-approaches/ (дата звернення: 06.05.2025)

[18] Wikipedia “UML”. URL: https://uk.wikipedia.org/wiki/Unified_Modeling_Language (дата звернення 26.05.2025)

[19] Maxym Zosym - "Варіанти використання та сценарії (Use Cases and Scenarios)". URL: https://www.maxzosim.com/use-cases-and-scenarios/ (дата звернення: 10.05.2025)

[20] Motiff - "What is Bootstrap?". URL: https://motiff.com/design-system-wiki/design-systems-overview/bootstrap-design-system-overview (дата звернення: 26.05.2025)

[21] Evernote Design - "Bookmark Links for Designer". URL: https://www.evernote.design/post/wireframecc/ (дата звернення: 10.05.2025)

[22] Hostkoss Blog - "Що таке система управління контентом (CMS)?". URL: https://hostkoss.com/b/uk/cms/ (дата звернення: 20.05.2025)

[23] WordPress Documentation. Database Description. URL: https://codex.wordpress.org/Database_Description (дата звернення: 26.05.2025)

[24] Increo - "Wordpress with WooCommerce". URL: https://increo.no/en/technology/wordpress (дата звернення: 26.05.2025)

[25] JS Communities - "Flask: Легкий у використанні веб-фреймворк та його реальні застосування". URL: https://javascript.org.ua/flask-legkij-u-vikoristanni-veb-frejmvork-ta-jogo-realni-zastosuvannya/ (дата звернення: 26.05.2025)

ДОДАТОК А. Код розробки

Лістинг А1. - Підсистема рекомендацій на основі ШІ

```
import random
```

```
import pandas as pd

# Список страв
dishes = [
    "Борщ", "Вареники", "Піца Маргарита", "Піца Пепероні", "Цезар", "Грецький салат",
    "Карбонара", "Спагеті Болоньєзе", "Том Ям", "Рамен", "Суші рол Філадельфія",
    "Сашимі", "Бургер з куркою", "Бургер з яловичиною", "Картопля фрі", "Стейк",
    "Кебаб", "Шашлик", "Плов", "Долма", "Лате", "Капучино", "Американо",
    "Чай зелений", "Чай чорний", "Морс", "Узвар", "Лимонад", "Тірамісу", "Морозиво"
]

# Параметри
num_rows = 1000
min_items = 2
max_items = 7

# Генерація списку замовлень
orders = []
for _ in range(num_rows):
    order = random.sample(dishes, random.randint(min_items, max_items))
    orders.append(",".join(order))

# Створення DataFrame з одним стовпцем 'items'
df = pd.DataFrame(orders, columns=["items"])

# Запис у CSV (з індексом і правильним розділенням)
df.to_csv("restaurant_orders.csv", index=False)

import warnings
warnings.filterwarnings('ignore')
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules, fpgrowth
from mlxtend.preprocessing import TransactionEncoder
import networkx as nx
import matplotlib.pyplot as plt
import seaborn as sns
import ipywidgets as widgets
from IPython.display import display

data = pd.read_csv("/content/restaurant_orders.csv")  # автоматично візьме заголовки з першого рядка
data.head()
```

```
# Групування даних по транзакціям
transactions = data['items'].str.split(',')
# Перетворення транзакцій
te = TransactionEncoder()
te_ary = te.fit(transactions).transform(transactions)
df_transformed = pd.DataFrame(te_ary, columns=te.columns_)
transactions.head()
# Віджет для вибору алгоритму асоціативних правил
algorithm_widget = widgets.Dropdown(
    options=['Apriori', 'FP-Growth'],
    description='Algorithm:',
)

# Віджет для встановлення мінімальної підтримки — % появи об'єкту у транзакціях
min_support_widget = widgets.FloatSlider(
    value=0.05,
    min=0.01,
    max=1,
    step=0.01,
    description='Min Support:',
)

# Віджет для вибору продукту
product_widget = widgets.Dropdown(options=te.columns_, description='Select Product:')

recommend_button = widgets.Button(description="Recommend")
recommendation_output = widgets.Output()

min_threshold_widget = widgets.FloatSlider(
    value=0.2,
    min=0.1,
    max=1,
    step=0.1,
    description='Min Threshold:',
)

build_button = widgets.Button(description="Build Rules")
output = widgets.Output()

def visualize_rules(rules):
```

```
    G = nx.DiGraph()
    labels_map = {}
    label_counter = 1
    for index, row in rules.iterrows():
        antecedents = tuple(row['antecedents'])
        consequents = tuple(row['consequents'])
        confidence = row['confidence']
        for antecedent in antecedents:
            if antecedent not in labels_map:
                labels_map[antecedent] = str(label_counter)
                label_counter += 1
        for consequent in consequents:
            if consequent not in labels_map:
                labels_map[consequent] = str(label_counter)
                label_counter += 1
        G.add_edge(labels_map[antecedent], labels_map[consequent], confidence=f"{confidence:.2f}")

    pos = nx.spring_layout(G)
    edges = G.edges(data=True)
    nx.draw_networkx_nodes(G, pos, node_size=700)
    nx.draw_networkx_edges(G, pos, edgelist=edges, edge_color='black')
    nx.draw_networkx_labels(G, pos)
    edge_labels = dict(((u, v), d['confidence']) for u, v, d in edges)
    nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels)
    plt.title('Association Rules')
    plt.axis('off')
    plt.show()

    print("Label mapping:")
    for item, label in labels_map.items():
        print(f"{label}: {item}")

def build_rules(button):
    global rules
    with output:
        output.clear_output()
        min_support = min_support_widget.value
        min_threshold = min_threshold_widget.value
        algorithm = algorithm_widget.value

        if algorithm == 'Apriori':
```

```
            frequent_itemsets = apriori(df_transformed, min_support=min_support, use_colnames=True)
        elif algorithm == 'FP-Growth':
            frequent_itemsets = fpgrowth(df_transformed, min_support=min_support, use_colnames=True)

        rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=min_threshold)

        if not rules.empty:
            print(f"Found {len(rules)} rules:")
            print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']])
        else:
            print("No rules found with the given parameters.")

build_button.on_click(build_rules)

display(widgets.VBox([
    algorithm_widget,
    min_support_widget,
    min_threshold_widget,
    build_button,
    output
]))

# Фільтруємо правила за confidence > 0.2
top_rules = rules[rules['confidence'] > 0.2].sort_values(by='confidence', ascending=False).head(10)

# Виведемо топ 10 правил
top_rules[['antecedents', 'consequents', 'confidence', 'lift']]

import matplotlib.pyplot as plt
import networkx as nx

def visualize_rules(rules):
    G = nx.DiGraph()  # Створюємо орієнтований граф
    for index, row in rules.iterrows():
        antecedents = tuple(row['antecedents'])
        consequents = tuple(row['consequents'])
        confidence = row['confidence']

        # Додаємо зв'язки між antecedents і consequents
        for antecedent in antecedents:
            for consequent in consequents:
```

```
            G.add_edge(antecedent, consequent, confidence=f"{confidence:.2f}")

    # Використовуємо Kamada-Kawai layout для гарного розташування
    pos = nx.kamada_kawai_layout(G, weight=None)

    plt.figure(figsize=(12, 12))  # Задаємо розмір графа
    nx.draw_networkx_nodes(G, pos, node_size=700, node_color="skyblue", alpha=0.7)
    nx.draw_networkx_edges(G, pos, edgelist=G.edges(), width=2, alpha=0.6, edge_color="black")
    nx.draw_networkx_labels(G, pos, font_size=10, font_color="black", font_weight="bold")

    # Виводимо етикетки на рёбрах (зв'язки), це буде значення confidence
    edge_labels = {(u, v): f"{d['confidence']}" for u, v, d in G.edges(data=True)}
    nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels, font_size=8, font_color="red")

    plt.title('Top 10 Association Rules', fontsize=15)
    plt.axis('off')
    plt.show()

# Візуалізуємо топ-10 правил
visualize_rules(top_rules)

# Функція для отримання рекомендацій
def get_recommendations(button):
    with recommendation_output:
        recommendation_output.clear_output()
        selected_product = product_widget.value
        recommendations_set = set()
        recommended = rules[rules['antecedents'].apply(lambda x: selected_product in x)]['consequents']

        if not recommended.empty:
            print(f"Recommendations for {selected_product}:")
            for index, item in recommended.items():
                recommendations_set.update(item)
            for item in recommendations_set:
                print(f"- {item}")
        else:
            print(f"No recommendations found for {selected_product}")

recommend_button.on_click(get_recommendations)

display(widgets.VBox([
```

```
    product_widget,
    recommend_button,
    recommendation_output
]))
```

Лістинг А2. - Інтерфейс головної сторінки

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Strava Restaurant</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    .hero {
      background-image: url('4.jpg'); /* заміни на актуальний шлях до зображення */
      background-size: cover;
      background-position: center;
      background-repeat: no-repeat;
      min-height: 100vh;
      display: flex;
      align-items: center;
      color: #fff;
    }
    .hero-content {
      text-align: left;
      margin-left: 130px;
      max-width: 600px;
      background-color: rgba(255, 255, 255, 0.85);
      padding: 40px;
      border-radius: 20px;
    }
    .hero h1 {
      color: #0d8b4e;
      font-weight: bold;
    }
    .btn-green {
```

```
        background-color: #0d8b4e;
        color: white;
      }
      .btn-green:hover {
        background-color: #0b6f3c;
        color: white;
      }
      .navbar-brand {
        font-family: 'Brush Script MT', cursive;
        font-size: 28px;
      }
      .nav-link {
        color: black !important;
      }
  </style>
</head>
<body>

<!-- Navbar -->
<nav class="navbar navbar-expand-lg navbar-light bg-white px-4">
  <a class="navbar-brand" href="#">Strava</a>
  <div class="ms-auto">
    <ul class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link" href="#">Головна</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Зареєструватись</a>
      </li>
    </ul>
  </div>
</nav>

<!-- Hero Section -->
<div class="hero">
  <div class="container">
    <div class="row">
      <div class="col-md-12 d-flex justify-content-start">
        <div class="hero-content">
          <h1>Вітаємо!</h1>
```

```
            <p class="fs-5 text-muted">Ніщо так не зближує людей, як смачна їжа.</p>
            <div>
              <button class="btn btn-green">Переглянути меню</button>
              <button class="btn btn-green">Забронювати столик</button>
            </div>
          </div>
        </div>
      </div>
    </div>
</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Лістинг А3. - Меню ресторану

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Меню ресторану</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <style>
    .menu-card {
      position: relative;
      border: 1px solid #eee;
      border-radius: 12px;
      padding: 15px;
      background-color: #fff;
      height: 100%;
      display: flex;
      flex-direction: column;
      justify-content: space-between;
    }
    .menu-card img {
      border-radius: 10px;
      width: 100%;
      height: auto;
      margin-bottom: 10px;
```

```
    }
    .menu-card .price {
        color: #0b6f3c;
        font-weight: bold;
    }
    .menu-card .add-icon {
        position: absolute;
        bottom: 10px;
        right: 10px;
        background-color: #0b6f3c;
        color: white;
        border-radius: 50%;
        width: 32px;
        height: 32px;
        display: flex;
        align-items: center;
        justify-content: center;
        font-size: 20px;
        cursor: pointer;
    }
    .section-title {
        margin: 30px 0 10px;
    }

    .navbar-brand {
        font-family: 'Brush Script MT', cursive;
        font-size: 28px;
    }
    .nav-link {
        color: #0b6f3c !important;
    }

    .dish-img {
        width: 100%;
        height: 200px;
        object-fit: cover;
        border-radius: 8px;
    }

</style>
```

```
</head>
<body>

<!-- Navbar -->
<nav class="navbar navbar-expand-lg navbar-light bg-white px-4">
  <a class="navbar-brand" href="#">Strava</a>
  <div class="ms-auto">
    <ul class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link" href="#">Головна</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Зареєструватись</a>
      </li>
    </ul>
  </div>
</nav>

<div class="container my-5">


  <div class="row row-cols-1 row-cols-md-4 g-4 mt-2">

    <!-- Страва -->
    <div class="col">
      <div class="menu-card">
        <img src="1.png" alt="Індичка рублена Капрезе">
        <h6>Салат із тофу</h6>
        <span class="price">78 UAH</span>
        <div class="add-icon">+</div>
      </div>
    </div>

    <!-- Страва -->
    <div class="col">
      <div class="menu-card">
        <img src="2.png" alt="Торт зі шпинату">
        <h6>Смажений стейк</h6>
        <span class="price">210 UAH</span>
        <div class="add-icon">+</div>
      </div>
```

```
</div>

<!-- Страва -->
<div class="col">
   <div class="menu-card">
      <img src="3.png" alt="Торт зі шпинату">
      <h6>Піца Балан'єзе</h6>
      <span class="price">140 UAH</span>
      <div class="add-icon">+</div>
   </div>
</div>

<!-- Страва -->
<div class="col">
   <div class="menu-card">
      <img src="4.png" alt="Торт зі шпинату">
      <h6>Салат із куркою та овочами</h6>
      <span class="price">98 UAH</span>
      <div class="add-icon">+</div>
   </div>
</div>

<!-- Страва -->
<div class="col">
   <div class="menu-card">
      <img src="5.png" alt="Торт зі шпинату">
      <h6>Британський сніданок</h6>
      <span class="price text-danger">Розпродано</span>

   </div>
</div>

<!-- Страва -->
<div class="col">
   <div class="menu-card">
      <img src="6.png" alt="Торт зі шпинату">
      <h6>Картопля Фрі</h6>
      <span class="price">68 UAH</span>
      <div class="add-icon">+</div>
   </div>
</div>
```

```
      <!-- Страва -->
      <div class="col">
        <div class="menu-card">
          <img src="7.png" alt="Торт зі шпинату">
          <h6>Крабовий боул</h6>
          <span class="price">348 UAH</span>
          <div class="add-icon">+</div>
        </div>
      </div>

      <!-- Страва -->
      <div class="col">
        <div class="menu-card">
          <img src="8.png" alt="Торт зі шпинату">
          <h6>Червоне м'ясо з овочами</h6>
          <span class="price">280 UAH</span>
          <div class="add-icon">+</div>
        </div>
      </div>

    </div>
  </div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```