МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ
(підпис)

2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерні науки,

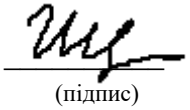
освітньо-професійної програми «Інформатика»

на тему: **«АДАПТИВНИЙ ВЕБ-САЙТ ДЛЯ СЕРВІСУ ДОСТАВКИ ЇЖІ»**

здобувача групи ІНз-01с Макушенка Олександра

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.


(підпис) Олександр МАКУШЕНКО

Керівник доцент,
кандидат фізико-математичних наук Сергій ШАПОВАЛОВ 
(підпис)

Суми – 2024

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«Затверджую»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерні науки, освітньо-професійної програми «Інформатика»

здобувача групи ІНз-01с Макушенка Олександра

1. Тема роботи: «Адаптивний веб-сайт для сервісу доставки їжі»
затверджую наказом по СумДУ від *«22» квітня 2024 р. №0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 09 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз проблеми та актуальності розробки систем управління взаємовідносинами з клієнтами, конкурентів та цільової аудиторії, постановка й формування завдань дослідження. 2) Огляд та вибір програмних засобів для інформаційних систем. 3) Розроблення адаптивного веб-сайту для сервісу доставки їжі. 4) Аналіз отриманих результатів.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____

(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми та актуальності розробки систем управління взаємовідносинами з клієнтами, конкурентів та цільової аудиторії, постановка й формування завдань дослідження*		
2	*Огляд та вибір програмних засобів для інформаційних систем.*		
3	*Розроблення адаптивного веб-сайту для сервісу доставки їжі*		
4	*Аналіз отриманих результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ Керівник _____

(підпис) (підпис)

АНОТАЦІЯ

Записка: 53 стор., 28 рис., 1 додаток, 24 використаних джерел.

Обґрунтування актуальності теми роботи – тема кваліфікаційної роботи є актуальною, оскільки зростає популярність сервісів доставки їжі. Мобільні пристрої стають основними для замовлень. Конкуренція в цій галузі вимагає вдосконалення сервісу. Забезпечення доступності та зручності для всіх користувачів важливо. Технологічний розвиток потребує постійного оновлення веб-сайту.

Об’єкт дослідження – сервіси доставки їжі.

Предмет дослідження – методи і моделі веб-сервісів доставки їжі.

Мета роботи – розробка конкурентоспроможного сервісу доставки їжі на ринку, надаючи інноваційний та ефективний веб-сайт за допомогою мов програмування та їх фреймворків.

Методи дослідження – мови програмування Mern-стеку (ReactJS, ExpressJS, MongoDB, NodeJS).

Результати – розроблено сервіс доставки їжі, який надає змогу користувачам ефективно надавати пропозиції щодо обслуговування, замовляти послуги компанії, розраховувати вартість, а менеджерам контролювати процеси і взаємодіяти з користувачами. Проведено тестування розробки на тестових даних клієнтів.

ВЕБ-СЕРВІС, ДОСТАВКА ЇЖІ,
REACTJS, NODEJS, MONGODB, EXPRESSJS

ЗМІСТ

ВСТУП 5
1 АНАЛІТИЧНИЙ ОГЛЯД 6
1.1 Аналіз предметної області 6
1.2 Аналіз конкурентів-аналогів 8
1.3 Постановка задачі 15
2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ 17
2.1 Вибір засобів для реалізації мети 17
2.2 Проєктування бази даних 20
3 ПРАКТИЧНА РЕАЛІЗАЦІЯ 21
3.1 Інформаційна модель та структура веб-сайту 21
3.2 Налаштування бібліотек та модулів 28
3.3 Визначення архітектури та структури додатку 29
3.4 Налаштування взаємодії з базою даних та API 32
3.5 Створення та налаштування інтерактивного бота..**Ошибка! Закладка не определена.**
3.6 Механізми авторизації та реєстрації 36
3.7 Кабінет користувача 36
3.8 Кабінет менеджера..... **Ошибка! Закладка не определена.**
3.9 Тестування інформаційної системи 37
3.10 Алгоритм роботи системи 45
ВИСНОВКИ 46
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 47
ДОДАТОК А МОДЕЛІ БАЗИ ДАНИХ 49

ВСТУП

Актуальність. Тема кваліфікаційної роботи є актуальною, оскільки присвячена веб-сервісам доставки, які в сучасних реаліях стрімко набирають популярності.

Об’єкт дослідження – сервіси доставки їжі.

Предмет дослідження – методи і моделі веб-сервісів доставки їжі.

Гіпотеза. Гіпотеза. Застосування адаптивного дизайну та оптимізації функціоналу веб-сайту для сервісу доставки їжі дозволить покращити користувацький досвід, забезпечити зручність замовлення їжі на різних пристроях і платформах, та, в результаті, збільшити кількість замовлень та задоволення клієнтів, знизивши відсоток відмов..

Новизна. Розроблене в цій роботі програмне рішення відзначається інноваційним підходом до покращення функціональності сервісу доставки їжі.. Крім того, програмне рішення також дозволяє автоматизувати внутрішні процеси підприємства, спрощуючи аналіз та візуалізацію статистичних даних, що призводить до більшої ефективності та задоволення як клієнтів, так і співробітників організації.

Структура. Дана робота організована у такий спосіб: у вступі розглядається актуальність теми; далі проводиться аналіз сучасного стану ринку сервісів доставки їжі та особливостей користувацького досвіду; формулюються конкретні завдання дослідження; обгрунтовується вибір необхідних інструментів та технологій для розробки адаптивного веб-сайту; подається детальний опис програмного забезпечення розробленого в ході дослідження; висновки систематизують отримані результати; у розділі "Список використаних джерел" перераховуються джерела, використані для підготовки роботи, та додаються необхідні додатки.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Аналіз предметної області

1. Адаптивність веб-сайту

Адаптивний веб-сайт – це ключовий аспект, що дозволяє користувачам комфортно взаємодіяти зі сервісом на різних пристроях. За словами Г. Барона та Д. Міллса, в їхній книзі "Адаптивний веб-дизайн: зручний і корисний", такий дизайн робить сайт доступним та зручним як на комп'ютерах, так і на мобільних пристроях. Використання принципів прогресивного покращення дозволяє створювати багаторівневі та гнучкі інтерфейси, що відповідають різним екранам та можливостям користувачів. [1]

2. Сервіс доставки їжі

Оптимізація процесів замовлення та доставки їжі вимагає ретельного розгляду інтерфейсу та логіки організації. Згідно зі словами С. Макгаврена у книзі "JavaScript та jQuery: розробка веб-сайтів", важливо розробляти відстеження замовлення та оповіщення користувачів для підвищення якості обслуговування. Використання JavaScript та jQuery може значно полегшити інтерактивність та зручність користування веб-сайтом. [2]

3. Чат з підтримкою штучного інтелекту

Впровадження штучного інтелекту у чат на веб-сайті може суттєво полегшити обслуговування користувачів. За словами П. Норвіга в книзі "Майстерність Python для штучного інтелекту", системи можуть автоматизувати відповіді на типові запитання та вдосконалюватися на основі попередніх інтеракцій [3]. Це може покращити ефективність обслуговування та знизити час очікування відповідей.

Безпека та конфіденційність

Оскільки веб-сайт обробляє особисті дані та інформацію про замовлення, безпека є критичною. Заходи для захисту даних та транзакцій важливі для забезпечення довіри користувачів. У книзі "Захист веб-додатків" М. Зеліц

розглядає методи та стратегії забезпечення безпеки веб-додатків, що є ключовим аспектом у веб-розробці [5]. Реалізація таких заходів є важливою для успішної експлуатації веб-сайту доставки їжі.

Підходи до адаптивного веб-дизайну:

1. Гнучкі сітки (Fluid Grids):

- Опис: Використовуючи гнучкі сітки, розміри елементів веб-сторінки вказуються відсотковими значеннями, а не фіксованими пікселями. Це дозволяє їм адаптуватися до різних розмірів екранів.

- Переваги:

- Забезпечує пропорційне розміщення елементів на різних пристроях.

- Відповідає на розширення чи стискання екрану.

2. Гнучкі зображення та медіа (Flexible Images & Media):

- Опис: Використання CSS та HTML5 для створення гнучких зображень та медіа-контенту, які можуть адаптуватися до розмірів екранів.

- Переваги:

- Зменшує розмір зображень при необхідності, що полегшує завантаження на мобільних пристроях.

- Зберігає якість візуального вмісту на різних пристроях.

3. Медіа-запити (Media Queries):

- Опис: CSS3 медіа-запити дозволяють застосовувати різні стилі в залежності від параметрів пристрою та екрану (наприклад, розмір екрану, роздільна здатність, орієнтація).

- Переваги:

- Забезпечує можливість створювати стилізацію елементів, яка відповідає конкретним характеристикам пристрою.

- Покращує досвід використання на різних пристроях.

4. Прогресивне розширення (Progressive Enhancement):

- Опис: Прогресивне розширення практикує побудову базового функціоналу для всіх користувачів, а потім додавання додаткових можливостей для пристроїв, які це підтримують.

- Переваги:

- Забезпечує доступ до важливого контенту для всіх користувачів, незалежно від їхніх можливостей пристрою.

- Підвищує швидкодію та ефективність сайту на більш потужних пристроях.

Комбінація цих підходів створює ефективний адаптивний веб-дизайн, що дозволяє створювати веб-сайти, які оптимально працюють на різних пристроях та забезпечують зручний та естетичний користувацький досвід. Рекомендована література, зокрема "Адаптивний веб-дизайн: зручний і корисний," надає поглиблене розуміння цих підходів та їхню практичну реалізацію.

1.2 Аналіз конкурентів-аналогів

Найпопулярніші конкуренти це Glovo (рис. 1.1) та Bolt Food (рис. 1.2), найголовнішою відмінністю є використання штучного інтелекту для покращення комунікації з клієнтами.

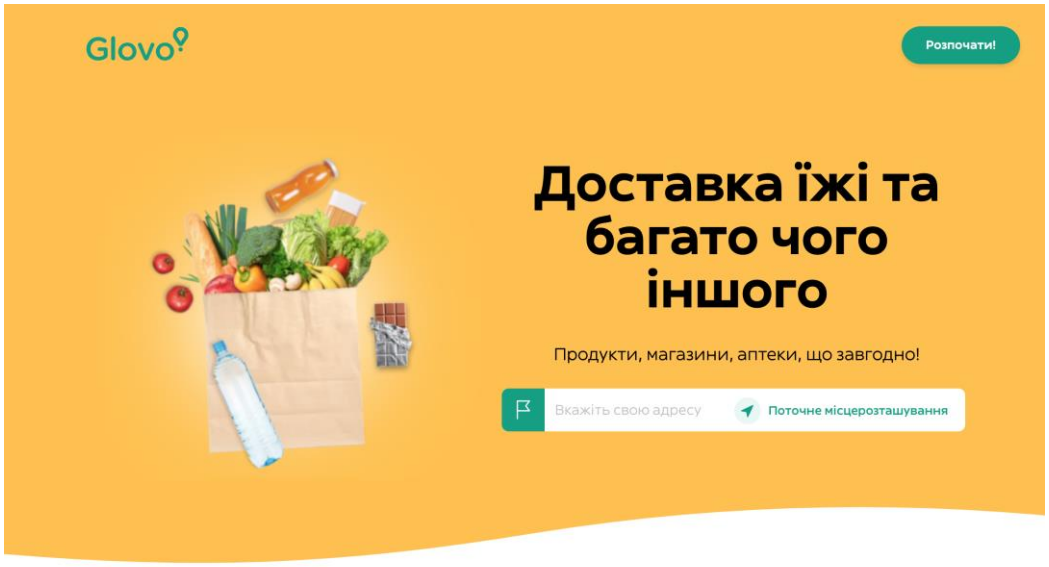


Рисунок 1.1 – Початкова сторінка сайту Glovo

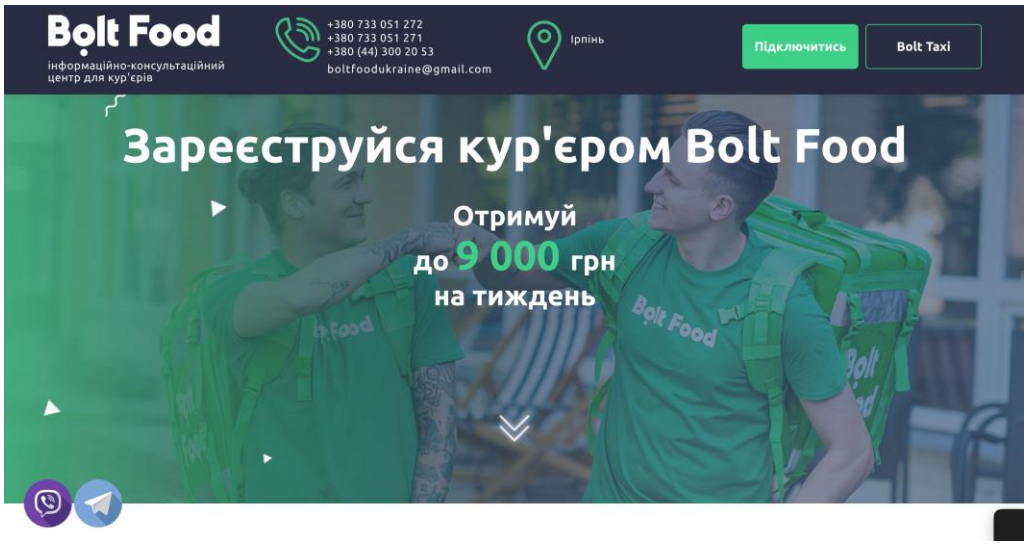


Рисунок 1.2 – Початкова сторінка сайту Bolt Food

Чат з підтримкою штучного інтелекту:

1. Автоматизована Обробка Запитань:

- Наш сервіс:

- Використання штучного інтелекту для автоматизованої обробки запитань користувачів. Чат-бот може визначати типи запитань та надавати відповіді на них без втручання оператора підтримки.

- Конкуренти:

- Glovo та Bolt використовують традиційні підходи до обслуговування клієнтів, включаючи людських операторів.

2. Персоналізовані Рекомендації та Пропозиції:

- Наш сервіс:

- Інтеграція штучного інтелекту для аналізу історії замовлень та вподобань користувачів для надання персоналізованих рекомендацій страв та ресторанів.

- Конкуренти:

- Більшість конкурентів обмежуються стандартними рекомендаціями без глибокого аналізу індивідуальних вподобань.

3. Здатність Приймати Замовлення через Чат:

- Наш сервіс:

- Можливість користувачів робити замовлення за допомогою чат-бота. Штучний інтелект розпізнає замовлення та взаємодіє з користувачем для уточнення деталей.

- Конкуренти:

- Замовлення через конкурентів, зазвичай, виконуються через стандартні форми та інтерфейси.

4. Аналіз Скарг та Відгуків:

- Наш сервіс:

- Використання аналітики штучного інтелекту для обробки та аналізу великої кількості відгуків та скарг, що дозволяє розпізнавати та вирішувати проблеми ефективно.

- Конкуренти:

- Конкуренти можуть використовувати людські ресурси для обробки відгуків, що може бути менш ефективним та витратним.

5. Навчання Чат-Бота за Допомогою Машинного Навчання:

- Наш сервіс:

- Використання машинного навчання для постійного вдосконалення чат-бота. Система навчається на основі нових даних та виправлень в реальному часі.

- Конкуренти:

- Традиційні чат-боти можуть бути менш гнучкими та не так швидко адаптуватися до змін у питаннях користувачів.

Використання штучного інтелекту в чат-боті нашого сервісу дозволяє створити високоінтелектуальну систему обслуговування клієнтів. У порівнянні з конкурентами, які в основному використовують традиційні методи підтримки, наш сервіс може забезпечити персоналізований та ефективний досвід для користувачів, що може значно підвищити рівень задоволення клієнтів та відмінити проблеми швидко.

Безпека та Конфіденційність

1. Захист Даних Користувачів:

- Наш сервіс:

- Застосування сучасних технологій шифрування для захисту особистих даних користувачів, включаючи інформацію про платіжні картки, адреси та історію замовлень.

- Конкуренти:

- Багато конкурентів також використовують шифрування, але рівень його складності та ефективності може варіюватися.

2. Аутентифікація та Авторизація:

- Наш сервіс:

- Впровадження двофакторної аутентифікації для підвищення рівня безпеки у випадку втрати пароля або неправомірного доступу.

- Конкуренти:

- Багато конкурентів також використовують аутентифікацію, але не завжди впроваджують двофакторну.

3. Моніторинг та Виявлення Погроз:

- Наш сервіс:

- Використання систем моніторингу та аналізу поведінки для виявлення надто активних або підозрілих активностей, що може свідчити про атаки або порушення безпеки.

- Конкуренти:

- Конкуренти також можуть мати системи моніторингу, але рівень їхньої автоматизації та точність може варіюватися.

4. Регулярні Аудити та Тестування на Проникнення:

- Наш сервіс:

- Проведення регулярних аудитів безпеки та тестування на проникнення для ідентифікації та усунення можливих слабких місць у системі.

- Конкуренти:

- Багато конкурентів також проводять аудити, але їх регулярність та глибина може відрізнятися.

5. Система Резервного Копіювання та Відновлення Даних:

- Наш сервіс:

- Застосування системи резервного копіювання для захисту даних від втрати або пошкодження, а також можливість швидкого відновлення в разі аварії.

- Конкуренти:

- Багато конкурентів також мають системи резервного копіювання, але не всі вони можуть бути так же ефективними.

Наш сервіс активно використовує передові методи та технології для забезпечення безпеки та конфіденційності даних користувачів. Порівняно з конкурентами, які також надають певний рівень безпеки, наш підхід орієнтований на використання передових методів шифрування, систем моніторингу та тестування на проникнення, що може додатково підвищити рівень захисту. У такому світлі наш сервіс може визначатися як високоякісний та сучасний в аспекті безпеки і конфіденційності.

1. Оптимізація Вибору Їжі та Рекомендації:

- Наш сервіс:

- Використання штучного інтелекту для аналізу історії замовлень та особистих уподобань користувача для надання персоналізованих рекомендацій щодо страв та ресторанів.

- Glovo:

- Засновані на алгоритмах рекомендації та акційні пропозиції, але без високого рівня персоналізації.

2. Прогнозування Часу Доставки:

- Наш сервіс:

- Використання аналітики та штучного інтелекту для прогнозування точного часу доставки з урахуванням трафіку, погодних умов та інших факторів.

- Bolt Food:

- Базовий прогноз часу доставки без використання складних аналітичних моделей.

3. Управління Запасами та Стратегія Ціноутворення:

- Наш сервіс:

- Використання штучного інтелекту для оптимізації стратегії ціноутворення та управління запасами на основі аналізу попиту та трендів.

- Glovo:

- Традиційні методи управління запасами та фіксована стратегія ціноутворення.

4. Користувацький Інтерфейс та Відповіді на Запитання:

- Наш сервіс:

- Інтеграція чат-бота з штучним інтелектом для відповіді на питання користувачів, прийому замовлень та надання допомоги.

- Bolt Food:

- Традиційні методи підтримки клієнтів, включаючи відділ служби підтримки.

Таблиця 1.1 Порівняльна таблиця конкурентів-аналогів

Параметр	**Bolt Food**	**Glovo**
Переваги		
Зручний та простий процес замовлення	✓	✓
Широкий вибір ресторанів та кухонь	✓	✓
Швидка доставка з урахуванням термінів	✓	✓
Ефективна підтримка користувачів	✓	✓
Недоліки		
Можливі проблеми з точністю доставки	✓	✓
Обмежена доступність в окремих місцях	✓	✓

Можливість затримок у доставці		✓
Можливі проблеми з точністю замовлень		✓
Використання Штучного Інтелекту		
Автоматизована обробка запитань користувачів	Немає	Немає
Персоналізовані рекомендації та пропозиції	Немає	Немає
Здатність приймати замовлення через чат	Немає	Немає
Аналіз скарг та відгуків	Немає	Немає
Навчання чат-бота за допомогою машинного навчання	Немає	Немає
Безпека та Конфіденційність		
Захист даних користувачів	Шифрування даних	Шифрування даних
Аутентифікація та авторизація	Двофакторна аутентифікація	Двофакторна аутентифікація
Моніторинг та виявлення загроз	Моніторинг активностей	Моніторинг активностей
Регулярні аудити та тестування на проникнення	Регулярні аудити	Регулярні аудити
Система резервного копіювання та відновлення даних	Резервне копіювання	Резервне копіювання

Наш сервіс відзначається використанням штучного інтелекту для створення індивідуалізованого та ефективного досвіду для користувачів. У порівнянні з Glovo та Bolt, які дотримуються традиційних методів управління та обслуговування, наш сервіс виходить за межі та створює інтелектуальну систему, яка враховує багато аспектів клієнтського досвіду. Застосування штучного інтелекту у всіх аспектах бізнесу, від рекомендацій їжі до управління запасами та відповіді на питання, позиціонує наш сервіс як інноваційного лідера в галузі доставки їжі.

1.3 Постановка задачі

На основі проведеного аналізу вже відомих рішень та існуючих джерел інформації поставимо наступну задачу – розробити та впровадити адаптивний веб-сайт для сервісу доставки їжі, який використовує чат з підтримкою штучного інтелекту.

В основу такого провадження покладемо створення інноваційного та ефективного інструменту для замовлення їжі, забезпечення персоналізованого досвіду користувача та підвищення конкурентоспроможності на ринку сервісів доставки їжі. Виконання такої задачі можливе при рішенні наступних звдань:

1. Розробка Адаптивного Веб-Сайту:

- Створення інтуїтивно зрозумілого та адаптивного інтерфейсу веб-сайту для замовлення їжі на різних пристроях (планшети, смартфони, настільні комп'ютери).

2. Онлайн-замовлення:

- Реалізація функції онлайн-замовлення їжі.

3. Безпека та Конфіденційність:

- Впровадження заходів для забезпечення безпеки особистих даних користувачів, включаючи шифрування, двофакторну аутентифікацію та системи моніторингу безпеки.

Очікувані Результати:

1. Функціональний та Ефективний Веб-Сайт:

- Створення веб-сайту, який задовольняє стандарти адаптивного дизайну та забезпечує ефективну роботу сервісу.

2. Безпека та Конфіденційність:

- Впровадження ефективних заходів для захисту даних та конфіденційності користувачів.

3. Зручна та Інтуїтивно Зрозуміла Система Оплати:

- Розробка системи оплати, яка забезпечить зручність та безпеку операцій.

Оцінка та Критерії Успішності:

1. Користувацький Досвід:

- Оцінка задоволення користувачів з використання веб-сайту та чат-бота.

2. Безпека та Конфіденційність:

- Перевірка ефективності заходів безпеки та ступеня захищеності особистих даних.

3. Інтеграція та Функціональність:

- Оцінка якості роботи різних функціональних модулів та їхньої інтеграції.

4. Інноваційність та Персоналізація:

- Оцінка рівня інновацій та ефективності персоналізації в роботі чат-бота та системи рекомендацій.

Завдання для Виконання:

1. Дослідження:

- Провести аналіз конкурентів та вивчення сучасних технологій у галузі доставки їжі та чат-ботів.

2. Проектування та Розробка:

- Спроектувати структуру веб-сайту та чат-бота, розробити їхні функціональні елементи та інтеграцію.

3. Впровадження Безпеки:

- Застосувати заходи для забезпечення безпеки даних, включаючи шифрування та двофакторну аутентифікацію.

4. Тестування та Відладка:

- Провести тестування всіх функціональностей, виявити та усунути помилки та недоліки.

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ

2.1 Вибір засобів для реалізації мети

1. Фреймворк для Розробки Веб-Сайту: Django

Опис:

Django - це високорівневий фреймворк для розробки веб-сайтів на мові програмування Python. Він надає готові рішення для багатьох аспектів веб-розробки, включаючи моделі баз даних, систему аутентифікації та структуру URL. [6][7]

Переваги:

- Зручність та Швидкість Розробки: Django пропонує швидке розгортання завдяки готовим компонентам та зручній структурі.

- Безпека: Включає вбудовані заходи безпеки, що полегшує розробку безпечних додатків.

- Адаптивність: Підтримує розробку адаптивних веб-сайтів, що важливо для зручного відображення на різних пристроях.

2. Інструменти для Розробки Чат-Бота: Rasa

Опис:

Rasa - це фреймворк для розробки чат-ботів з відкритим вихідним кодом. Він використовується для розуміння намірів користувачів та обробки їхніх повідомлень. [8][9]

Переваги:

- Відкритий Код: Дозволяє повний контроль та налагодження чат-бота відповідно до вимог проекту.

- Машинне Навчання: Використовує машинне навчання для покращення розпізнавання намірів та витягу інформації з повідомлень.

- Співпраця з Python: Легко інтегрується з іншими інструментами на основі Python, що полегшує використання в екосистемі Django.

3. База Даних: MongoDB

Опис:

MongoDB - це не реляційна система керування базами даних. Вона підтримує та забезпечує розширені можливості для роботи з даними.[12][13]

Переваги:

- Надійність та Стабільність: MongoDB відома своєю стабільністю та надійністю, що важливо для збереження цілісності даних.

- Підтримка JSON-типу: Забезпечує підтримку JSON-типу, що полегшує взаємодію з даними у форматі JSON.

5. Фреймворк для Адаптивного Дизайну: Tailwind

Опис:

Tailwind - це фреймворк для розробки адаптивних та естетичних веб-сайтів, який містить готові CSS-стилі та компоненти для різних елементів. [14][15]

Переваги:

- Готові Компоненти: Tailwind має багато готових компонентів, які роб

лять розробку швидкою та забезпечують єдність дизайну.

- Адаптивність: Дозволяє легко створювати адаптивні веб-сайти, що автоматично адаптуються під різні пристрої та розміри екранів.

Використання обраних інструментів забезпечить розробку високоякісного, ефективного та інноваційного веб-сайту для сервісу доставки їжі з чатом та штучним інтелектом. Кожен інструмент відіграє свою роль у створенні зручного та функціонального продукту, забезпечуючи його стабільність, безпеку та високу продуктивність.

Порівняння Обраних Фреймворків із Іншими

1. Фреймворк для Розробки Веб-Сайту: Express

Порівняння:

- Django vs Express: Обидва фреймворки мають широкий функціонал, але Express надає готові рішення для багатьох завдань, що полегшує розробку. Django є меншим та гнучким, ініціюючи власну структуру.

Обрано:

- Express обрано через вбудовані можливості, швидкість розробки та стандартизацію.

4. База Даних: MongoDB

Порівняння:

- MongoDB vs MySQL: Обидва стабільні та надійні, але PostgreSQL має більше розширені можливості, зокрема, підтримку JSON-типу.

Обрано:

- MongoDB обрано через його розширені можливості та підтримку JSON.

5. Фреймворк для Адаптивного Дизайну: Tailwind

Порівняння:

- Tailwind vs Foundation: Обидва популярні, але Tailwind частіше використовується завдяки великій спільноті та великому набору готових компонентів.

Обрано:

- Tailwind обрано через його готові компоненти та широке використання в галузі веб-розробки.

Висновок:

Обрані фреймворки були вибрані через їхню ефективність, гнучкість та здатність вирішувати конкретні вимоги проекту. Кожен фреймворк має свої унікальні переваги у порівнянні з конкурентами, і їх вибір базується на потребах та цілях проекту з розробки адаптивного веб-сайту для сервісу доставки їжі.

2.2 Проєктування бази даних

База даних є невід'ємною складовою будь-якого проєкту. Вона може зберігати не тільки дані про клієнтів, а й дані про послуги, продукцію тощо. Існують різні системи керування базами даних, такі як: MySQL, PostgreSQL, Oracle, Microsoft SQL Server, SQLite. Але всі вони відрізняються своїм функціоналом та предметною областю застосування. З попередніх досліджень ми вияснили, що будемо використовувати нереляційну базу даних – MongoDB.

Перед початком роботи потрібно спроєктувати моделі, які потім будемо застосовувати у своєму проєкті (рис. 2.2).

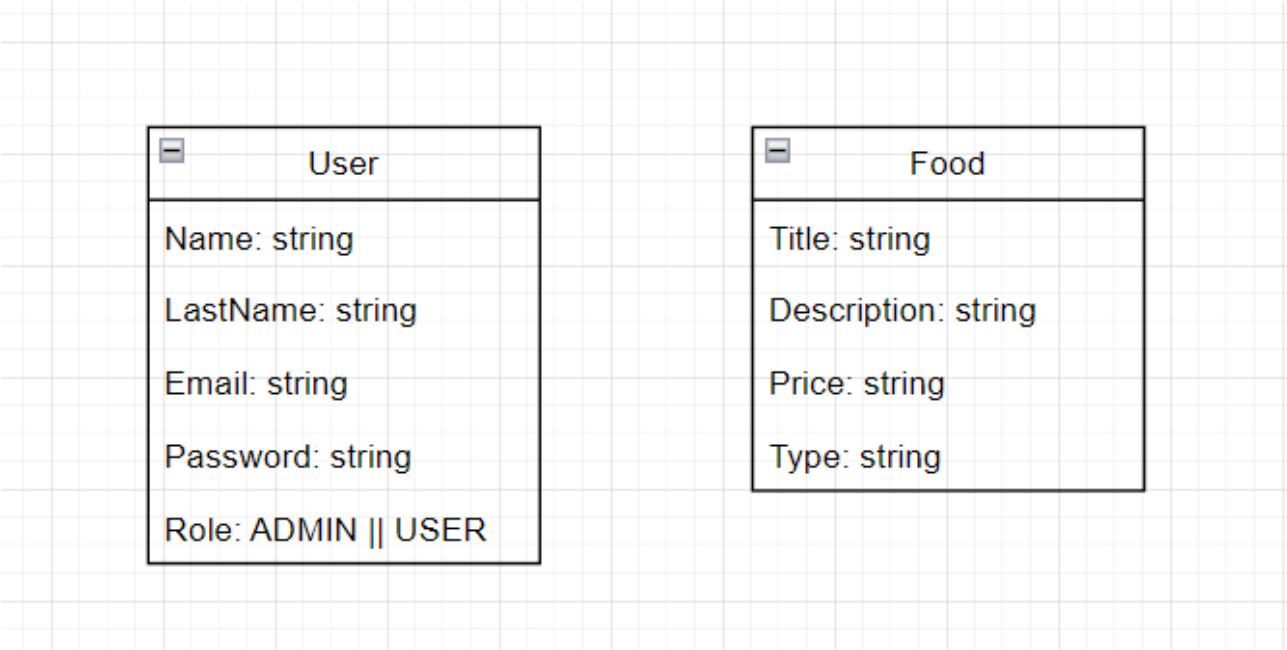


Рисунок 2.2 – Спроєктована схема бази даних

Як бачимо з рисунку, ми маємо 2 моделі, з якими будемо працювати у серверній та клієнтській частині.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Інформаційна модель та структура веб-сайту

Веб-сайт – це комплекс веб-сторінок, зображень та анімацій, електронних файлів та різних документів, баз даних, які розміщені у Всесвітній мережі Інтернет. Це структурована кореляційна інформаційна платформа, що має деяке оформлення та власний інтерфейс функціонування з даними та інформацією.

Створення інформаційної моделі веб-сайту це перше, що необхідно виконати при написанні веб-сайту будь-якого типу. Інформаційна модель складається з основних елементів веб-сторінки. Розміщення та порядок розділів веб-сторінки відображає структура веб-сайту [17].

Розрізняють два типи структури веб-сайту: внутрішня і зовнішня. До внутрішньої структури входить розроблення бізнес логіки веб сайту, категорій, послуг тощо. Веб сайт для підприємства «СумЕнерго» має ієрархічну структуру, так як існує багато розділів, підрозділів. Такий тип структури є найоптимальнішим для CRM систем, так як користувач завжди має вибір та можливість перейти в будь-який розділ, переглянути інформацію як з головної сторінки так і з будь-якої іншої [18]. Така внутрішня структура показана на рисунку 3.1.

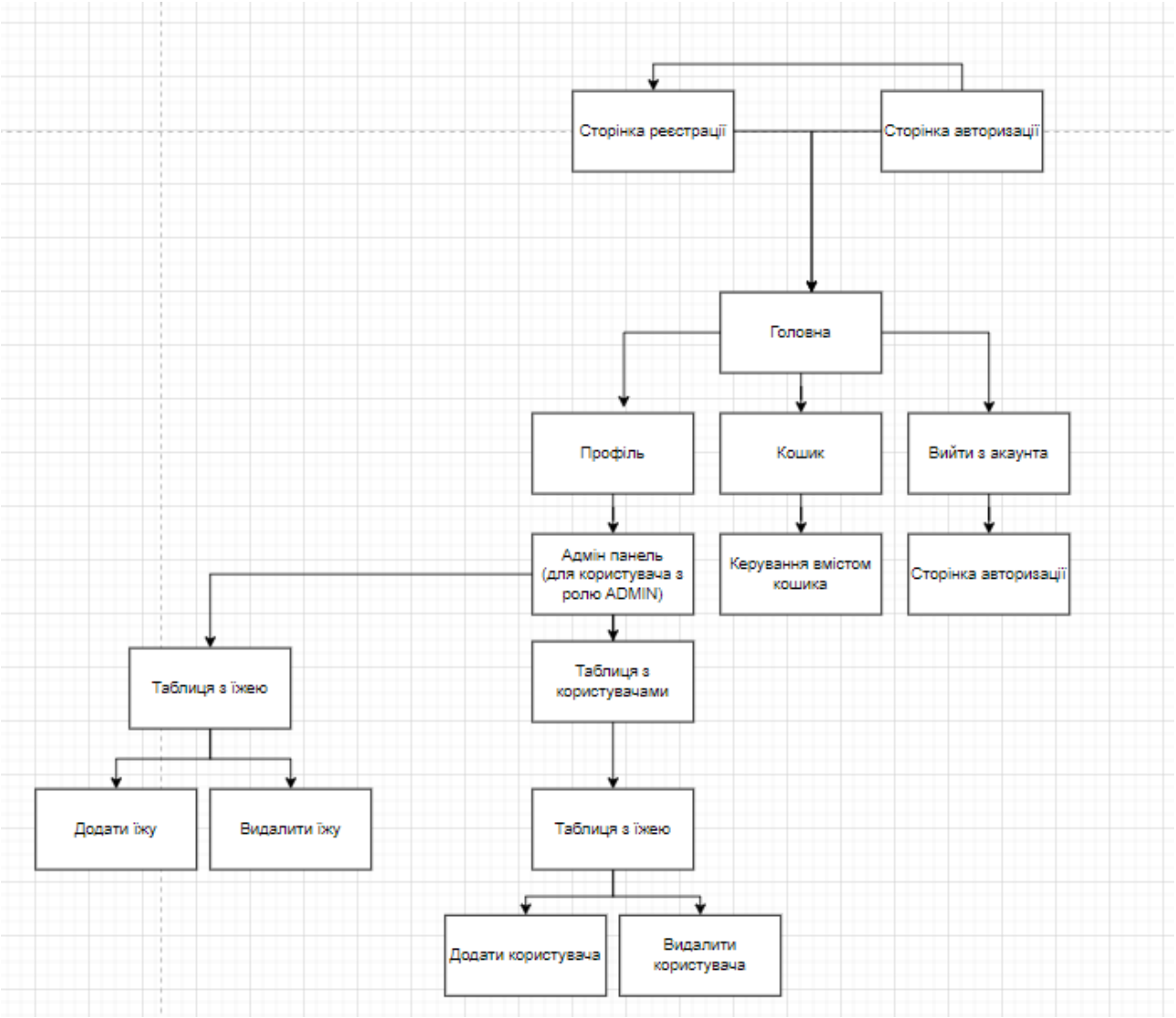


Рисунок 3.1 – Внутрішня структура сайту

Вибір правильних кольорів для дизайну сайту є також вирішальним значенням для успіху в Інтернеті. Кольори можуть бути найпотужнішим інструментом для отримання реакції від цільової аудиторії. Ви можете використовувати кольори, щоб викликати у відвідувачів емоції або відповісти на заклик до дії на нашому сайті. Колір допомагає нам обробляти і зберігати зображення більш ефективно, ніж безбарвні (чорно-білі) зображення [19]. Так як наша цільова аудиторія переважно повнолітні та люди старшого віку, потрібно використати кольори, які б означали надійність. Щодо палітри кольорів використаних в дизайні, в проєкті використано наступні кольори (рис. 3.2):

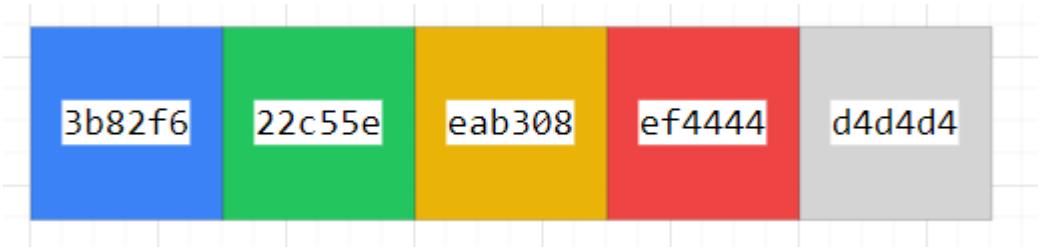


Рисунок 3.2 – Використані кольори на сайті

За психологією кольорів саме синій колір та його відтінки передають надійність. Вважається, що саме цей колір викликає у людей довіру до постачальників послуг та вони можуть бути більш схильні до покупки або замовлення будь-яких послуг, оскільки вони сприймають сайт, як надійний та професійний. Також синій колір асоціюється з оптимізмом. Коли люди бачать певні відтінки синього кольору, вони з більшою ймовірністю відчувають піднесений настрій. Такий спосіб також може допомогти людям відчути оптимізм щодо продукції або послуг, які сайт постачає і підвищує ймовірність замовлення або реєстрації [20].

Зовнішня структура веб сайту передбачає модель сайту та відповідає його дизайну. Вона визначає розміщення даних на веб сторінці при певному виборі пункту меню сайту. Зовнішня структура створюється для полегшення користування веб-сайтом користувачами та повторює навігацію сайтом. Також розроблення такої структури дає змогу веб програмісту краще орієнтуватися при створенні веб ресурсу. Таким чином, цей вид структурування сайту відповідає за розташування елементів на веб-сторінці.

Основними елементами зовнішньої структури є:

– Шапка сайту. Складається з логотипу або назви сайту. Також там може розміщуватися додаткові посилання такі як, інформація про компанію, меню, контактні дані та інше;
– Меню. Засіб для переходу до каталогу товарі або інформації;
– Центральна частина або контент. Містить основні елементи сторінки, які доступні користувачу, такі як послуги, різна інформація у вигляді тексту, онлайн підтримка, зображення або відео;

Важливими чинниками для успішної структури сайту є зручність користування, інтуїтивно зрозуміле розташування елементів на веб-сторінці, яскравість дизайну та інше. Саме такої тенденції дотримано в ході розробки зовнішньої структури сайту. Інформаційна система розділена на наступні блоки:

- Головна сторінка, якою переважно можуть користуватися гості, які тільки зайшли на сайт (рис. 3.3). Вона включає в себе:
  1) сторінку з актуальною інформацією про їжу (рис 3.4);
  2) навагаційне меню (рис. 3.5).
- Особистий кабінет клієнта. В кабінеті представлена інформація про користувача та, якщо, користувач має роль ADMIN буде наявна кнопка переходу на адмін панель (рис. 3.6).
  3)
- Адмін панель. Представлена основна інформація про користувачі, які вже зареєстровані в системі (рис. 3.7). Також адмін панель включає в собі сторінку з їжею, яка наявна на веб-сайті (рис. 3.8):

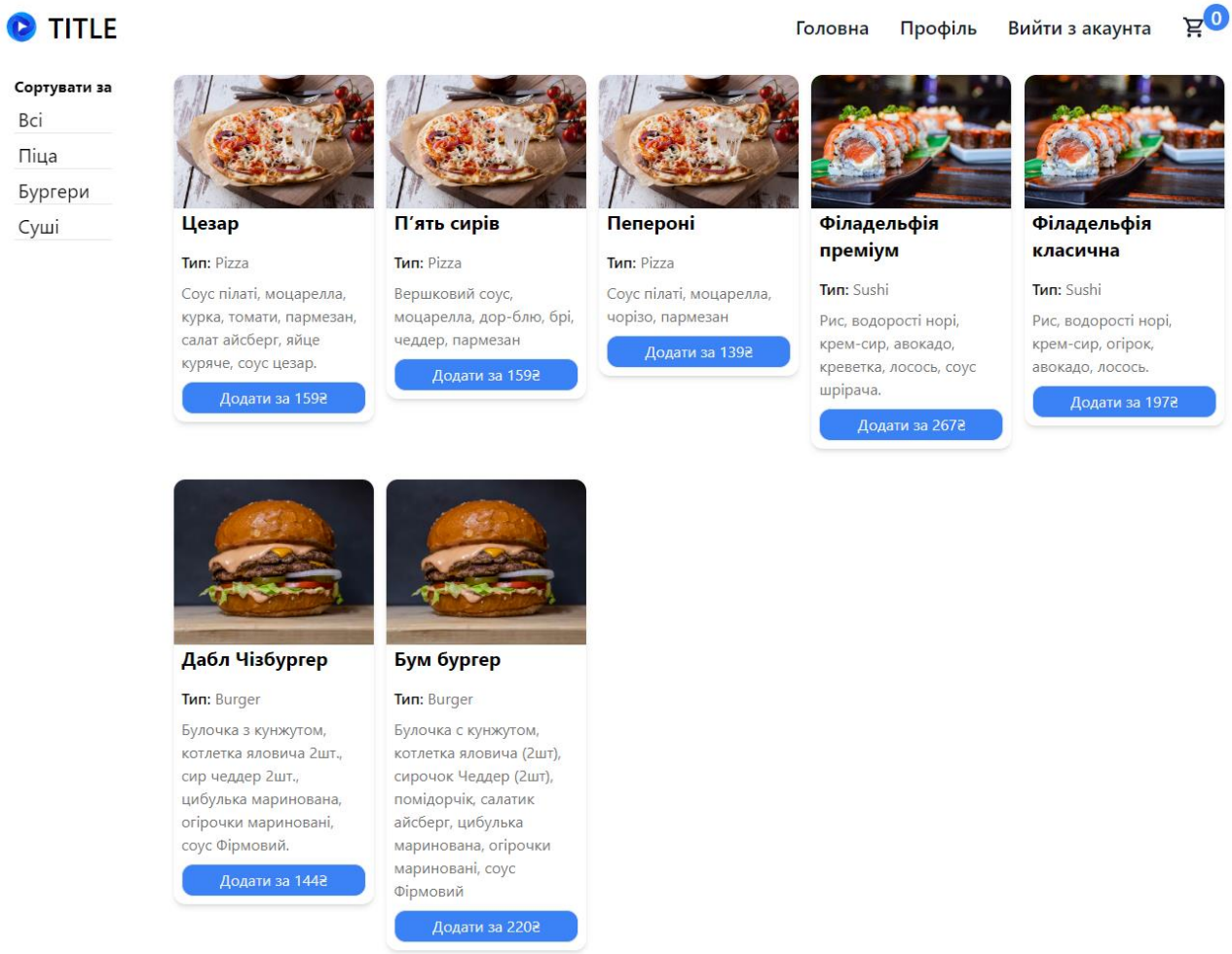


Рисунок 3.3 – Головна сторінка сайту

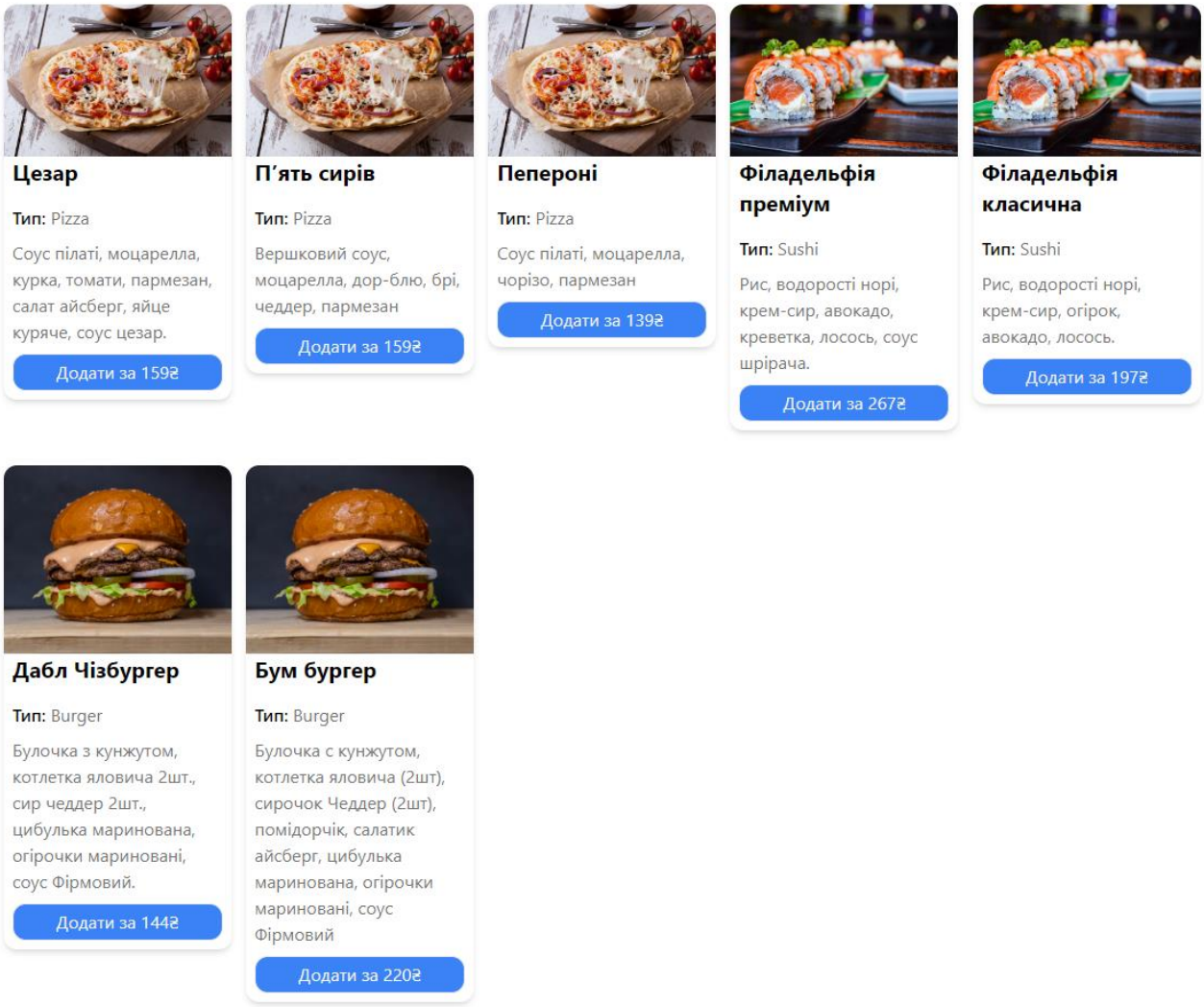


Рисунок 3.4 – Актуальна інформація про їжу

Рисунок 3.5 – Навігаційне меню

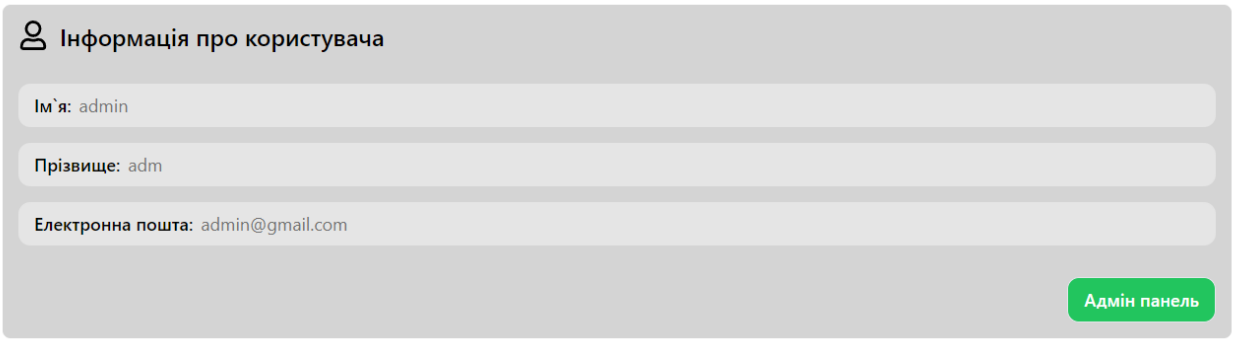
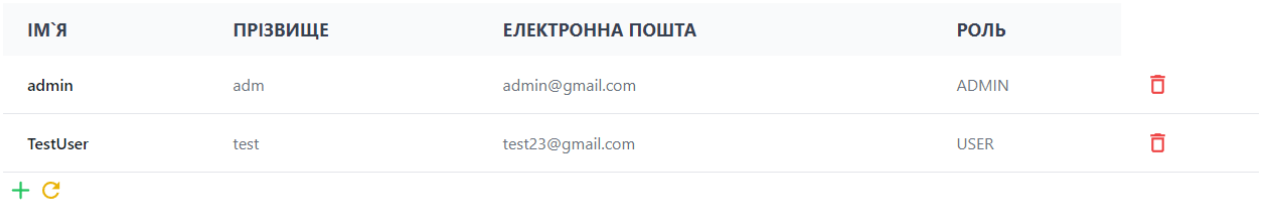
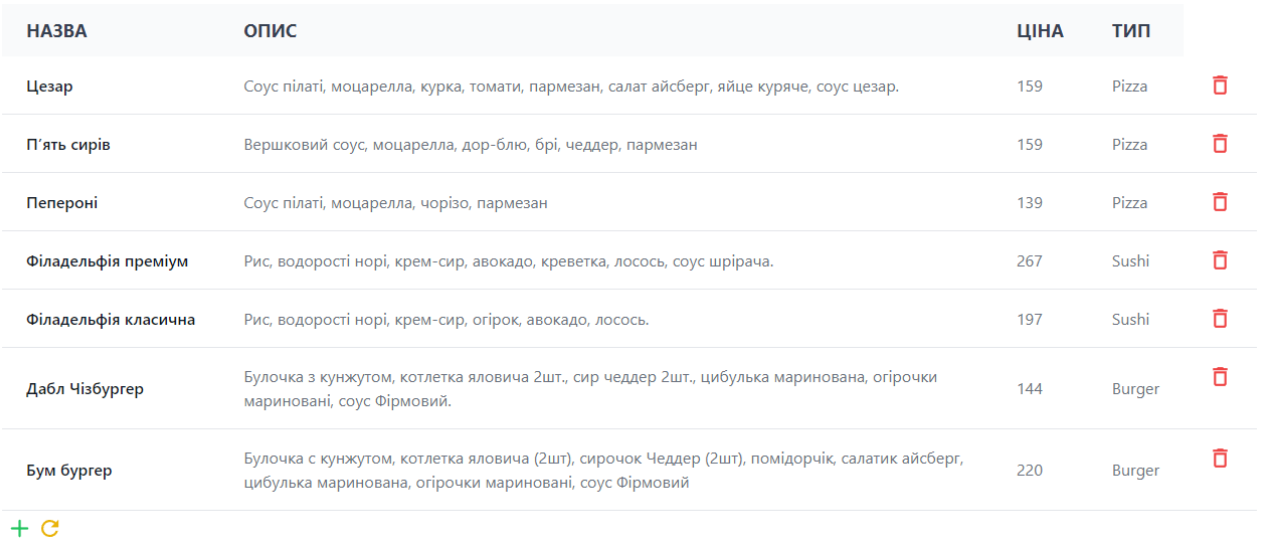


Рисунок 3.6 – Головна сторінка особистого кабінету



ІМ`Я	ПРІЗВИЩЕ	ЕЛЕКТРОННА ПОШТА	РОЛЬ
admin	adm	admin@gmail.com	ADMIN
TestUser	test	test23@gmail.com	USER

Рисунок 3.7 – Інформація про користувачів



НАЗВА	ОПИС	ЦІНА	ТИП
Цезар	Соус пілаті, моцарелла, курка, томати, пармезан, салат айсберг, яйце куряче, соус цезар.	159	Pizza
П'ять сирів	Вершковий соус, моцарелла, дор-блю, брі, чеддер, пармезан	159	Pizza
Пепероні	Соус пілаті, моцарелла, чорізо, пармезан	139	Pizza
Філадельфія преміум	Рис, водорості норі, крем-сир, авокадо, креветка, лосось, соус шрірача.	267	Sushi
Філадельфія класична	Рис, водорості норі, крем-сир, огірок, авокадо, лосось.	197	Sushi
Дабл Чізбургер	Булочка з кунжутом, котлетка яловича 2шт., сир чеддер 2шт., цибулька маринована, огірочки мариновані, соус Фірмовий.	144	Burger
Бум бургер	Булочка с кунжутом, котлетка яловича (2шт), сирочок Чеддер (2шт), помідорчік, салатик айсберг, цибулька маринована, огірочки мариновані, соус Фірмовий	220	Burger

Рисунок 3.8 – Інформація про їжу

Для користувачів така структура веб-сайту та надана інформація та навігація сайтом буде абсолютно зрозумілою на візуальному та інтуїтивному рівні. Ефективність якісно структурованої CRM системи пояснюється запорукою успішної оптимізації, полегшенням просування веб-сайту, високою відвідуваністю користувачами та стабільною роботою.

3.2 Налаштування бібліотек та модулів

Для досягнення швидкодії та надійності роботи системи доцільно використовувати необхідні програмні бібліотеки та модулі, які можливо знайти у реєстрі програмного забезпечення NPM [21]. До переваг використання таких пакетів входять:

- Прискорення роботи при розробці програмного забезпечення;
- Керування декількома версіями коду та залежностями;
- Оновлення додатку, так як оновлюється базовий код;
- Прискорення швидкодії застосунку.

Для підключення пакетів та бібліотек використовують консольну команду:

```
npm install <назва бібліотеку/модулю/пакету>
```

У проєкті використано важливі бібліотеки, які надають додаткові функції та можливості для реалізації роботи з базою даних, валідації даних, взаємодії клієнта з серверною частиною, роботи зі стилями, розробки клієнтської та серверної частини, роботи з датами, надання безпеки та налаштування автентифікації. До таких бібліотек відносяться: Express, BcryptJS, JsonWebToken, Cors, Mongoose, Axios, React (React-dom, react-router-dom, React), tailwindCSS та React-redux. Зберегти та налаштувати дані пакети можливо у конфігураційному файлі package.json (Додаток А – серверна частина, Б – клієнтська частина), який зберігає основну інформацію про додаток, а саме опис того, як взаємодіяти з додатком та запускати його [22]. Для застосування пакетів в програмному коді використовується команда:

```
import <назва компоненту> from «<назва пакету>»
```

3.3 Визначення архітектури та структури додатку

Гарно спроєктована архітектура папок впливає на ефективність, обслуговування та швидкодію продукту. Такий спосіб забезпечує зрозумілу і логічну організацію файлів та компонентів системи, полегшує розробку сумісно з розробниками та забезпечує навігацію та управління продуктом. Правильна структура модулів та каталогів також допомагає забезпечити чистоту коду, дотримання стандартів програмування, що впливає на якість та підтримку продукту протягом його життєвого циклу.

Так, в нашому проєкті використовуються різні структури папок для серверної (рис. 3.9) та клієнтської (рис. 3.10) частини.

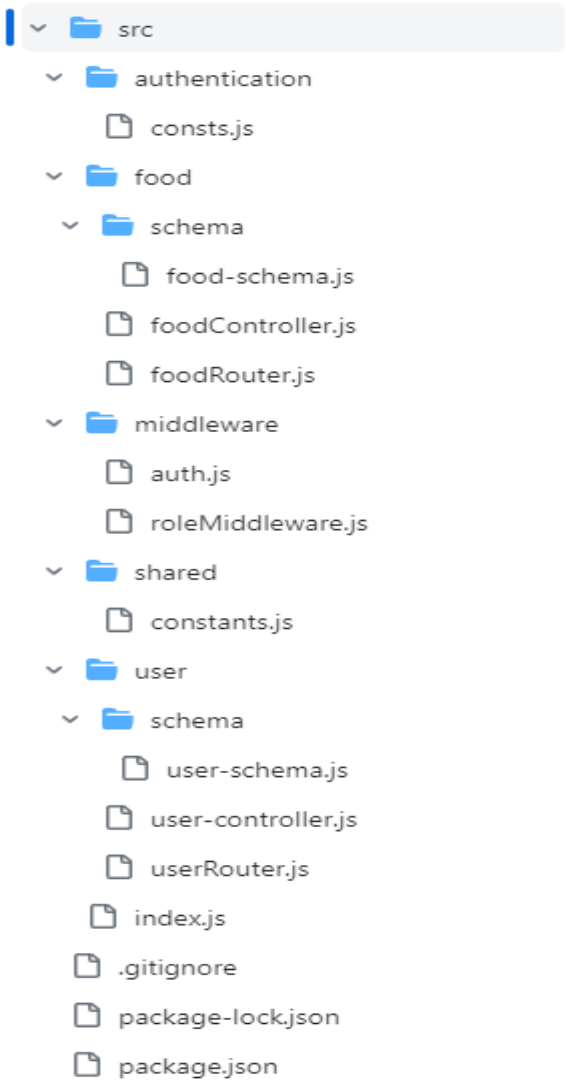


Рисунок 3.9 – Структура папок серверної частини

Назви папок відповідають за свої обов’язки та функції в програмній реалзіації:

- authentication – містить файл який зберігає доступні ролі користувача;
- food – містить файли з ендпоінтами та напрямки виконання API запитів до food;
- middleware – містить файли для роботи з автентифікацією користувача на стороні серверу;
- shared – містить файл з паролями до БД у зашифрованому вигляді;
- user – містить файли з ендпоінтами та напрямки виконання API запитів до user.

В основному файлі index.js відбувається підключення модулів та використання файлів з перелічених папок для належної роботи з сервером.

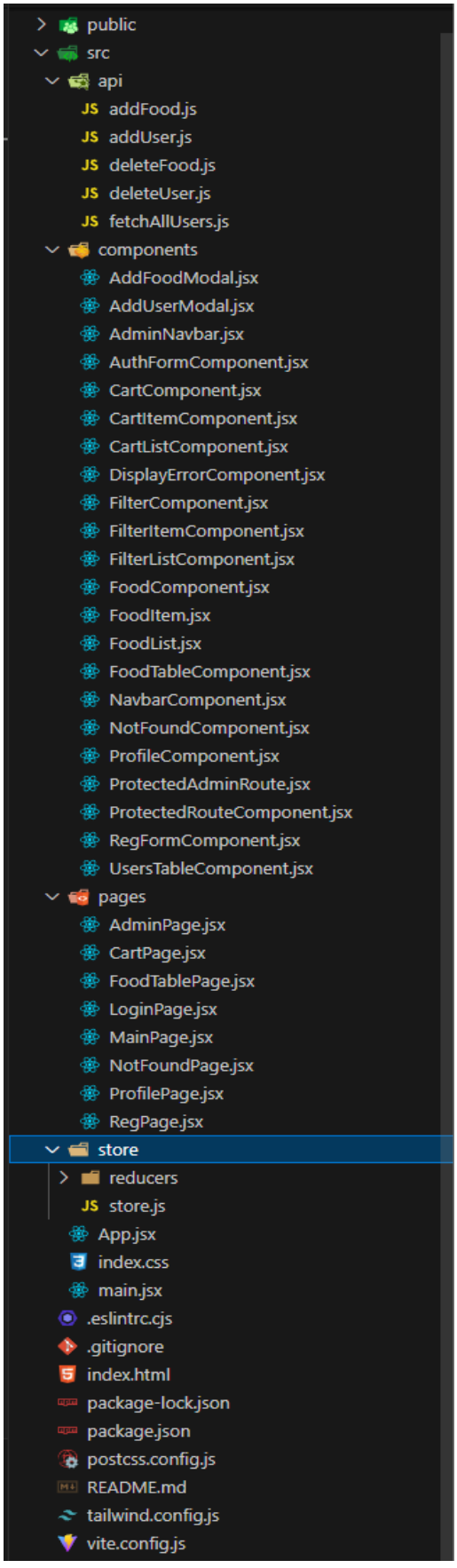


Рисунок 3.10 – Структура папок клієнтської частини

В клієнтській частині структура папок включає в собі папки компонентів, сторінок, роботу з отриманням даних з серверу та конфігураційний файл для перевірки автентифікації користувача та сумісної роботи серверу та клієнту.

3.4 Налаштування взаємодії з базою даних та API

Для забезпечення коректної роботи з базою даних, необхідно налаштувати конфігураційний файл з назвою бази даних, портом та хостом до якого потрібно підключитись. Ці конфігураційні дані зберігаються у папці cfg і мають наступний вигляд (рис. 3.11).

```
export const JWT_SECRET = "D0VFuKR";
export const DB_PWD_ENCODED = "R1gzMUZEOHpya2diTTBvRQ==";
export const DB_STRING = `mongodb+srv://olexandr:${Buffer.from(
  DB_PWD_ENCODED,
  "base64"
).toString(
  "utf-8"
)}@cluster0.rgbncix.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0`;
export const PORT = 5000;
```

Рисунок 3.11 – Конфігураційний файл з даними для підключення до БД

З’єднання з базою даних відбувається в основному файлі серверу і виглядає наступним чином (рис. 3.12).

```
async function start() {
  try {
    await mongoose.connect(DB_STRING);

    app.listen(CONNECTION_PORT, () =>
      console.log(`Server started on ${CONNECTION_PORT}`)
    );
  } catch (error) {
    console.error(error);
  }
}

start();
```

Рисунок 3.12 – Підключення до БД

Після підключення до БД також потрібно створити моделі за концептуальною моделлю бази даних, контролери для роботи з цими моделями та роути (напрямки) (Додаток Б) за якими будуть відбуватися взаємодія з даними на стороні серверу та клієнту.

Мій проєкт містить 2 моделі (Додаток А):

– Модель користувача – Додаток А (user-schema.js). Містить інформацію про споживача: прізвище, ім’я, адреса електронної пошти, пароль та роль;
– Модель їжі – Додаток А (food-schema.js). Містить наступну інформацію: назва, опис, ціна та тип;

Після створення моделі потрібно налаштувати контролери та напрямки для застосування API (Додаток Б).

Також потрібно забезпечити автентифікацію користувача з використанням JWT Tokena. JWT – відкритий стандарт, який визначає компактний і автономний спосіб безпечної передачі інформації між сторонами у вигляді об’єкта JSON [23]. Токен використовується при авторизації призначаючи користувача секретний токен, який має строк придатності 30 діб. Для перевірки чи автентифікований користувач чи ні – використовується наступний модуль:

```
import jwt from "jsonwebtoken";
import { JWT_SECRET } from "../shared/constants.js";

export function authMiddleware(req, res, next) {
  if (req.method === "OPTIONS") {
    next();
  }

  try {
    const authHeader = req.headers.authorization;

    if (!authHeader) {
      return res.status(401).json({ message: "Unauthorized" });
    }
    const token = authHeader.split(" ")[1];

    if (!token) {
      return res.status(401).json({ message: "Unauthorized" });
```

```
    }
    const decodedData = jwt.verify(token, JWT_SECRET);

    req.user = decodedData;
    next();
  } catch (error) {
    console.error(error);
    return res.status(401).json({ message: error });
  }
}
```

Таким чином відбувається налаштування роботи з базою даних.

Щоб налаштувати правильне отримання даних з серверної частини на клієнтській потрібно налаштувати axios - HTTP-бібліотека на основі promise, яка дозволяє розробникам надсилати запити до власного або стороннього сервера для отримання даних [24]. Відповідні функції описуються наступним кодом:

```
import axios from "axios";

const instance = axios.create({
  baseURL: "http://localhost:5000/",
});

instance.interceptors.request.use(
  async (config) => {
    const token = sessionStorage.getItem("token");

    if (token) {
      config.headers.Authorization = "Bearer " + token;
    }

    return config;
  },
  (error) => {
    if (axios.isAxiosError(error)) {
      if (error.status === 401) {
        return Promise.reject(error.response?.data);
      }
    } else {
      Promise.reject("Error occured, try again later");
    }
  }
);
```

Однієї http бібліотеки для роботи з даними не вистачить, тому використовуються компоненти Redux – фрагменти, частини коду Redux, які відносяться до певного набору даних і дій в межах стану сховища (Додаток В).

Для відображення даних на сторінках також використовуються хуки UseSelector, useEffect та useDispatch [25,26]. За допомогою хуків ми можемо отримати дані про користувача, його роль, дані про показники, послуги тощо, шляхом використання функцій та станів фрагментів Redux. На приклад ми можемо перевірити чи авторизований користувач адмін та направити його на потрібну сторінку (використані функції та стани фрагменту loginSlice.js Додатку Г):

```
const dispatch = useDispatch();
  const error = useSelector((state) => state.auth.error);

  const handleAuth = async (event) => {
    event.preventDefault();

    const formData = new FormData(event.currentTarget);
    const email = formData.get("email");
    const password = formData.get("password");

    const userAuthInfo = {
      email,
      password,
    };

    try {
      const response = await dispatch(authentification(userAuthInfo));
      const token = response.payload.token;

      if (token) {
        nav("/main");
      } else {
        console.error("Authentication failed");
      }
    } catch (error) {
      console.error(error.message || "Registration failed!");
    }
  };
```

Таким чином дані з серверу використовуються в клієнтській частині.

3.5 Механізми авторизації та реєстрації

Важливим компонентом в розробці інформаційних систем є авторизація та реєстрація. В момент реєстрації пароль шифрується за допомогою алгоритму Bcrypt та зберігається в зашифрованому вигляді в таблиці бази даних (Додаток Б – user-controller.js, registration). В стані авторизації пароль розшифровується за допомогою функцій дефишровки того ж алгоритму та звіряється з введеним користувачем (Додаток Б – user-controller.js, login). Користувачу також присвоюється токен авторизації з терміном зберігання 24 години. Такий спосіб допоможе користувачу залишатись авторизованим в системі. В такий спосіб забезпечується безпека та конфіденційність даних користувача в момент використання системи.

3.6 Кабінет користувача

В рамках розробки нашої системи кабінет користувача є невід'ємною складовою. Кабінет є основним компонентом системи і надає користувачам широкий спектр функцій та можливостей.

В особистому кабінеті користувач може переглянути загальну інформацію та, якщо, користувач має роль ADMIN він зможе перейти до адмін (рис. 3.13).

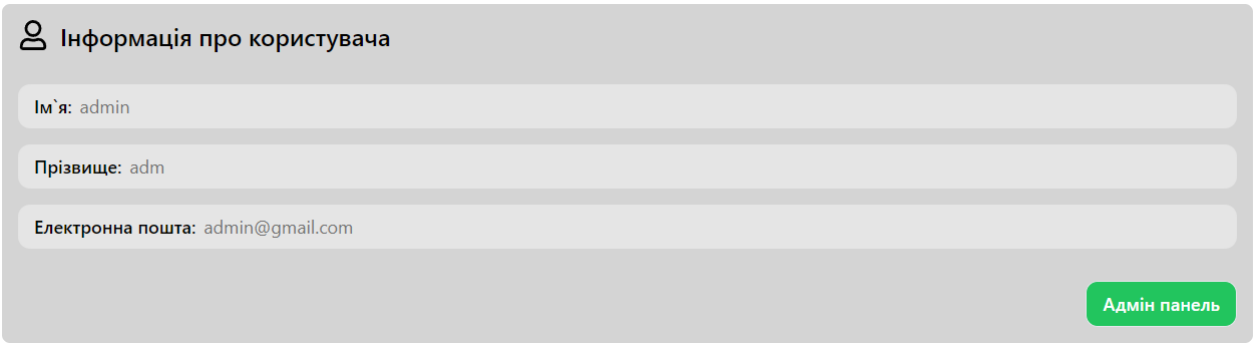


Рисунок 3.13 – Кабінет користувача

3.7 Адмін панель

Адмін панель є також невід'ємною складовою розробки інформаційної системи для підприємства. Адмін може не виходячи з дому ефективно контролювати користувачів та інформацію про їжу.

В кабінеті менеджер може переглядати актуальних користувачів системи (Додаток Б – userController.js – getUsers) на головній сторінці. На «вкладці Їжа» адмін може переглядати наявну їжу на веб-сайті (Додаток Б – serviceController.js – getService) та керувати нею: видаляти (Додаток Б – food-controller.js – deleteFood) та додавати (Додаток Б – food-controller.js – createFood). Після додавання або видалення, таблицю треба оновити, кнопка наяна під таблицею.

3.8 Тестування інформаційної системи

Перед релізом будь-якої інформаційної системи або додатку потрібно провести відповідно тестування системи аби передати замовнику вже готовий продукт і зберегти рейтинг компанії в майбутньому.

Тестування буде проводитись шляхом White-box тестування. Специфікацією такого тестування є те, що його проводять розробники, яким відома внутрішня структура та реалізація системи. Дані для тестування вибираються, ґрунтуючись на знанні коду, який будемо тестувати. Також ми знаємо яким повинен бути результат тестування [28].

В нашій системі ми будемо тестувати наступні компоненти: авторизація, реєстрація, додавання до кошика товарів, видалення товарів з кошика, наявність адмін панелі для користувачів з роллю ADMIN.

Перевірка на валідацію даних є важливим аспектом при розробці будь-якого ресурсу. В нашому програмному коді перевірка на валідацію на всіх формах відбувається на стороні сервера:

```
router.post(
  "/registration",
  [
    check("name").trim().notEmpty(),
    check("lastName").trim().notEmpty(),
    check("email").trim().notEmpty().isEmail(),
    check("password").trim().isLength({ min: 5, max: 10 }),
  ],
  userController.registration
);
```

Поля форми реєструються за допомогою функції register та в разі помилки зберігають повідомлення в змінній errors (рис. 3.14).

```
{error && (
  <div className="text-center">
    <DisplayErrorComponent errorMessage={error} />
  </div>
)}
```

Рисунок 3.14 – Відображення помилки, якщо вона наявна

Наступним компонентом є авторизація. При введенні неіснуючої пошти користувач отримує повідомлення «Не правильна пошта або пароль» (рис. 3.15). В разі введення некоректного паролю або пошти виводиться повідомлення «Некоректний формат пошти», як це було на формі реєстрації.

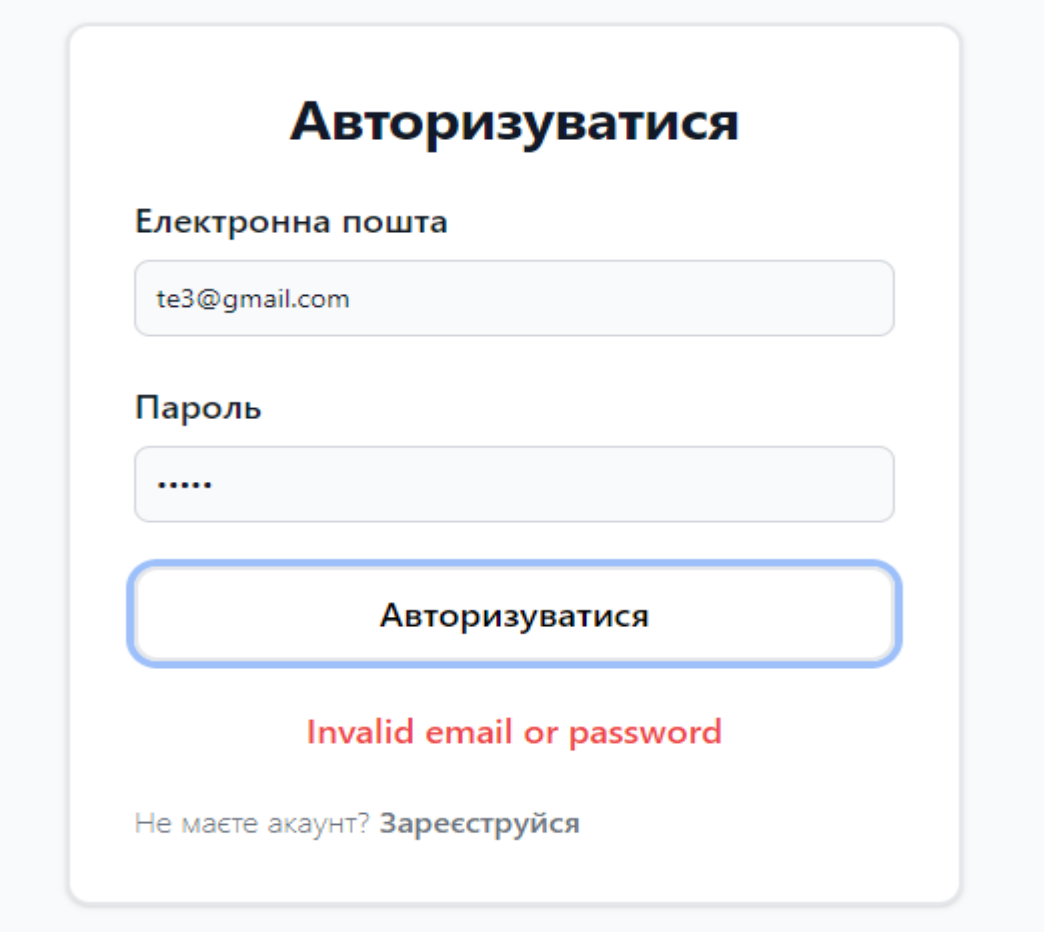


Рисунок 3.15 – Авторизація з неіснуючою поштою

В разі введення неправильного паролю користувач отримує теж саме повідомлення(рис. 3.16)

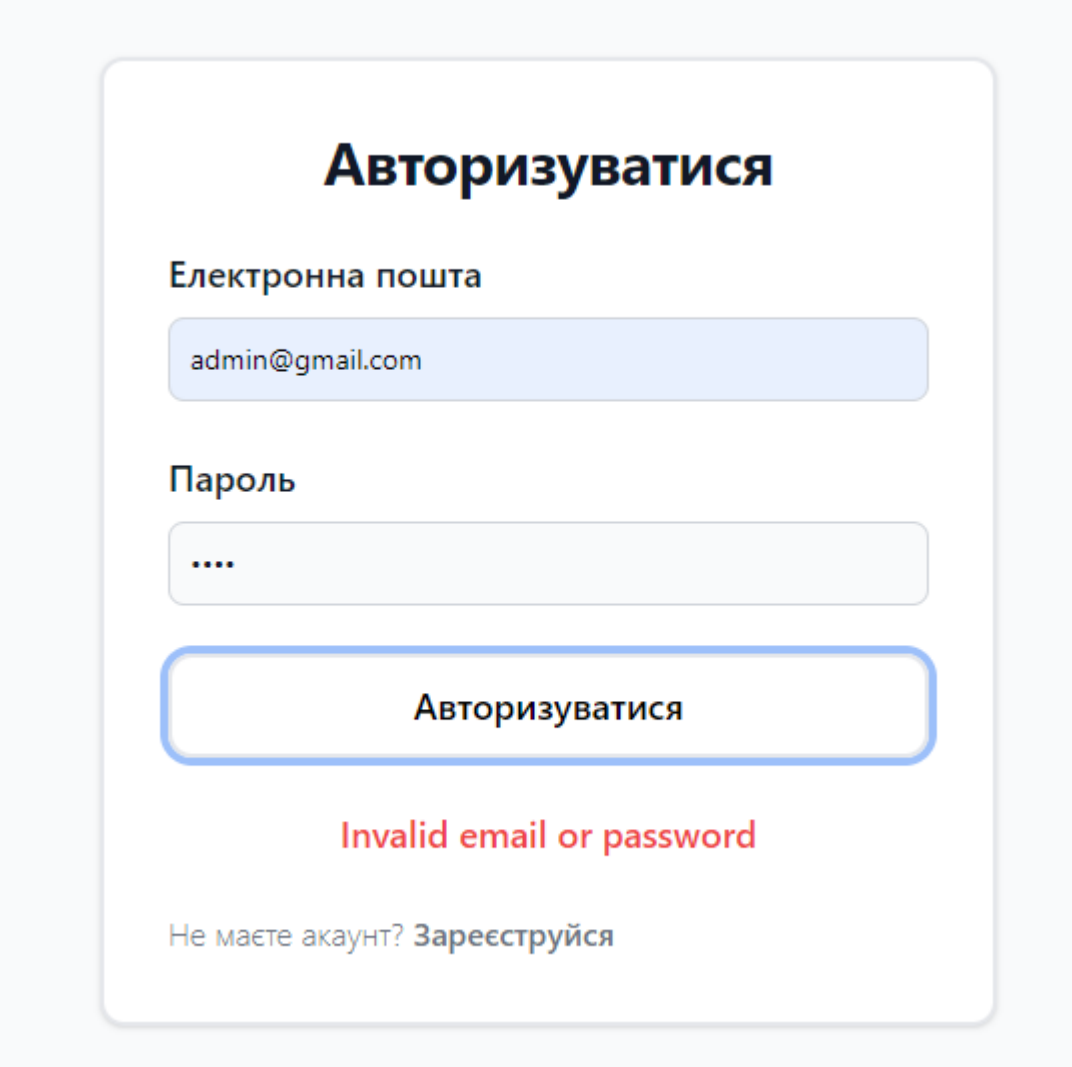


Рисунок 3.16 – Авторизація з невалідним паролем

Наступним компонентом реалізації є модуль додавання та видалення з кошика. Для цього було реалізовано функціонал у redux

```
const initialState = {
  cartItems: [],
};

const cartReducer = (state = initialState, action) => {
  switch (action.type) {
    case "ADD_TO_CART":
      return {
        ...state,
        cartItems: [...state.cartItems, action.payload],
      };
    case "REMOVE_FROM_CART":
      return {
        ...state,
        cartItems: state.cartItems.filter((item) => item.id !== action.payload),
```

```
      };
    case "CLEAR_ALL":
      return {
        ...state,
        cartItems: action.payload,
      };
    default:
      return state;
  }
};

export const addToCart = (item) => ({
  type: "ADD_TO_CART",
  payload: item,
});

export const removeFromCart = (itemId) => ({
  type: "REMOVE_FROM_CART",
  payload: itemId,
});

export const clearAll = (item) => ({
  type: "CLEAR_ALL",
  payload: item,
});

export default cartReducer;
```

Протестуємо даний компонент програми. Додавання до кошика (рис. 3.17). Видалення з кошика (рис. 3.18)

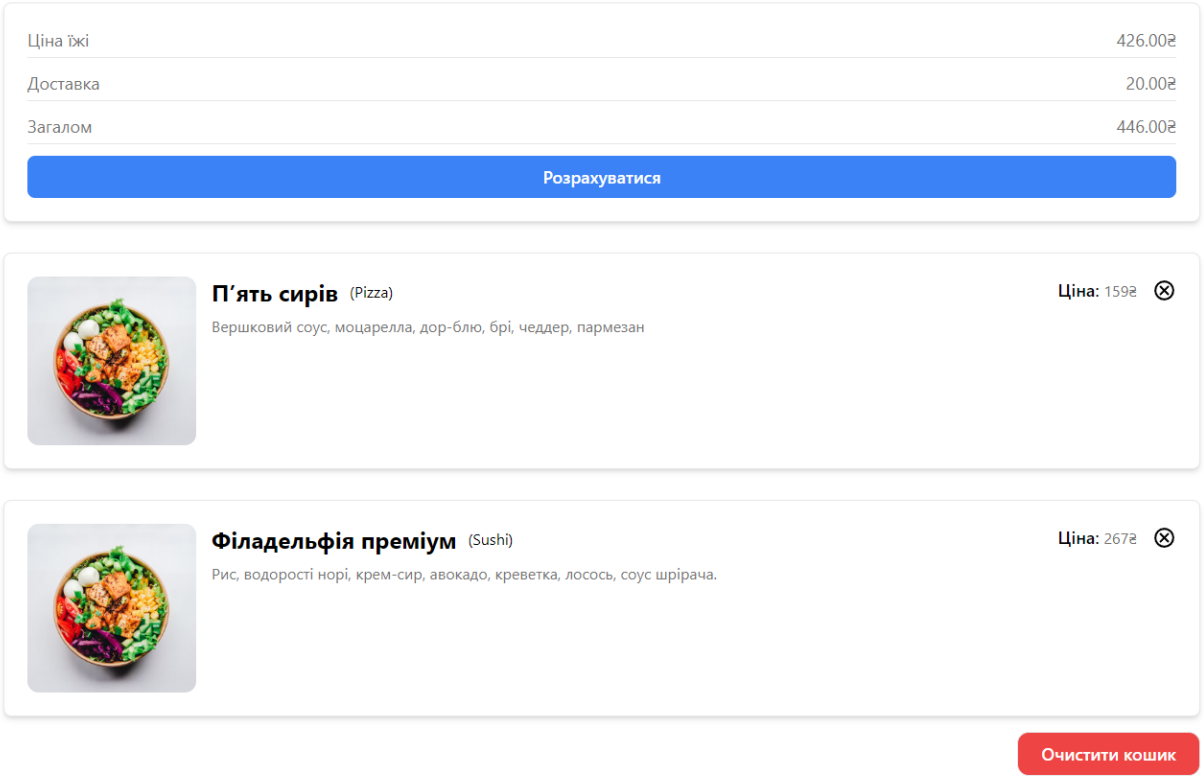


Рисунок 3.17 – Додавання товарів

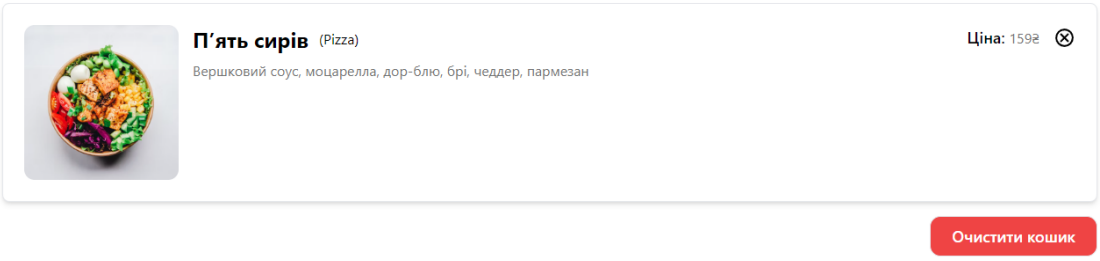


Рисунок 3.18 – Видалення товарів

Наступним і останнім модулем для тестування є модуль модуль адмін панелі. Адмін панель повинна бути скрита від звичайних користувачів, тому для цього було реалізовано наступний модуль програми. Після реєстрації користувачу надсилається token JWT який вже наявний у БД та розшифрований. Якщо роль дорівнює ADMIN, то відображаємо кнопку для переходу на адмін панель.

```
{userInfo.role === "ADMIN" && (
          <div className="flex items-end justify-end">
            <div className="mt-8 bg-green-500 inline-flex items-center gap-2 py-2
px-4 rounded-xl border hover:border-black transition cursor-pointer">
              <button
                onClick={() => nav("/admin")}
                className="text-white text-lg font-semibold"
              >
                Адмін панель
              </button>
            </div>
          </div>
        )}
```

В разі переходу на адмін панель, буде відображено таблиці користувачів та їжі. Також, можна виконати наступні дії: додати, видалити та оновити таблицю (рис. 3.19 – 3.20).

ІМ`Я	ПРІЗВИЩЕ	ЕЛЕКТРОННА ПОШТА	РОЛЬ
admin	adm	admin@gmail.com	ADMIN
TestUser	test	test23@gmail.com	USER

Рисунок 3.19 – Таблиця користувачів

НАЗВА	ОПИС	ЦІНА	ТИП
Цезар	Соус пілаті, моцарелла, курка, томати, пармезан, салат айсберг, яйце куряче, соус цезар.	159	Pizza
П'ять сирів	Вершковий соус, моцарелла, дор-блю, брі, чеддер, пармезан	159	Pizza
Пепероні	Соус пілаті, моцарелла, чорізо, пармезан	139	Pizza
Філадельфія преміум	Рис, водорості норі, крем-сир, авокадо, креветка, лосось, соус шрірача.	267	Sushi
Філадельфія класична	Рис, водорості норі, крем-сир, огірок, авокадо, лосось.	197	Sushi
Дабл Чізбургер	Булочка з кунжутом, котлетка яловича 2шт., сир чеддер 2шт., цибулька маринована, огірочки мариновані, соус Фірмовий.	144	Burger
Бум бургер	Булочка с кунжутом, котлетка яловича (2шт), сирочок Чеддер (2шт), помідорчік, салатик айсберг, цибулька маринована, огірочки мариновані, соус Фірмовий	220	Burger

Рисунок 3.20 – Таблиця їжі

3.9 Алгоритм роботи системи

Гість

Клієнт, який ще не має облікового запису відразу перенаправляється до сторінки реєстрації. Для доступу до послуг користувач повинен бути зареєстрований.

Авторизований користувач

Користувач, який має обліковий запис може переглядати усі сторінки, включаючи: головну, сторінку, сторінку профілю та кошик. Має можливість додавати товари до кошику, видаляти товари з кошику та розрахуватися.

Адмін

Адмін як і авторизований користувач має доступ до всіх сторінок веб-системи та до сторінки адмін панелі. Він може керувати вмістом веб-сайту, а саме додавати та видаляти їжу, та користувачів.

.

ВИСНОВКИ

Всі поставлені завдання кваліфікаційної роботи спрямовані на розробку адаптивного веб-сайту для сервісу доставки їжі виконані в повному обємі.

Для досягнення цієї мети були обрані ключові технологічні компоненти, які взаємодіють між собою для створення інноваційного та ефективного продукту.

1. Веб-Сайт:

- Використано фреймворк Express та React для швидкої та зручної розробки веб-сайту з адаптивним дизайном.

3. База Даних та Забезпечення Безпеки:

- Обрано MongoDB як систему керування базами даних для забезпечення надійності та підтримки JSON-типу.

- Застосовано заходи забезпечення, включаючи шифрування, для захисту особистих даних користувачів.

4. Адаптивний Дизайн:

- Tailwind використано для розробки адаптивного дизайну, що забезпечує однаково зручний перегляд веб-сайту на різних пристроях.

5. Тестування та Безпека:

- Проведено комплексне тестування для виявлення та усунення помилок, а також забезпечено високий рівень безпеки та конфіденційності даних.

В цілому, кваліфікаційна робота ставить перед собою амбіційні завдання і вдається в їх реалізації за допомогою сучасних технологій та інноваційних підходів. Розроблений продукт має потенціал поліпшити якість обслуговування в галузі доставки їжі та надає користувачам зручний та інтелектуальний інструмент для взаємодії з сервісом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Г. Барон, Д. Міллс. "Адаптивний веб-дизайн: зручний і корисний." (Adaptive Web Design: Crafting Rich Experiences with Progressive Enhancement)
2. С. Макгаврен. "JavaScript та jQuery: розробка веб-сайтів." (JavaScript and jQuery: The Missing Manual)
3. П. Норвіг. "Майстерність Python для штучного інтелекту." (Python Machine Learning)
4. Й. Лау. "Веб-розробка з Flask." (Flask Web Development)
5. М. Зеліц. "Захист веб-додатків." (Web Application Defender's Cookbook)
6. Django. Django Project. URL: https://docs.djangoproject.com/
7. W. Vincent. (2018). "Django for Beginners." ISBN-13: 978-1983172669.
8. Rasa: Developer Documentation Portal. Rasa. URL: https://rasa.com/docs/
9. D. Brown. (2016). "Chatbot Development: A Beginner's Guide." ISBN-13: 978-1539398030.
10. TensorFlow. TensorFlow. URL: https://www.tensorflow.org/
11. A. Geron. (2019). "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow." ISBN-13: 978-1492032649.
12. PostgreSQL: Documentation. PostgreSQL: The world's most advanced open source database. URL: https://www.postgresql.org/docs/
13. R. Ramadurgam. (2019). "PostgreSQL High Performance Cookbook." ISBN-13: 978-1838983507.
14. Bootstrap. The most popular HTML, CSS, and JS library in the world. URL: https://getbootstrap.com/docs/
15. J. Gamble, D. Turton. (2014). "Bootstrap Site Blueprints." ISBN-13: 978-1782164524.

16. M. Foust. (2020). "Python Testing with pytest." ISBN-13: 978-1801072636.
17. M. McMillan. (2019). "Serious Python: Black-Belt Advice on Deployment, Scalability, Testing, and More." ISBN-13: 978-1593278786.
18. A. J. Menezes, P. C. van Oorschot, S. A. Vanstone. (1996). "Handbook of Applied Cryptography." ISBN-13: 978-0849385230.
19. S. Krug. (2014). "Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability." ISBN-13: 978-0321965516.
20. J. Nielsen, H. Loranger. (2006). "Prioritizing Web Usability." ISBN-13: 978-0321350312.
21. R. S. Pressman, B. R. Maxim. (2014). "Software Engineering: A Practitioner's Approach." ISBN-13: 978-0078022128.
22. D. Beazley. (2009). "Python Essential Reference." ISBN-13: 978-0672329784.
23. A. Dewey. (2015). "Learning Linux Security and Hardening." ISBN-13: 978-1785285518.
24. B. Kernighan, R. Pike. (1999). "The Practice of Programming." ISBN-13: 978-0201615869.

ДОДАТОК А
МОДЕЛЬ БАЗИ ДАНИХ

```
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
  },
  lastName: {
    type: String,
    required: true,
  },
  email: {
    type: String,
    required: true,
  },
  password: {
    type: String,
    required: true,
  },
  role: {
    type: String,
    enum: [Role.Admin, Role.User],
  },
});
```

```
const foodSchema = new mongoose.Schema({
  title: {
    type: String,
    required: true,
  },
  description: {
    type: String,
    required: true,
  },
  price: {
    type: Number,
    required: true,
  },
  type: {
    type: String,
    required: true,
  },
});
```