МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет)___Автоматизації і комп'ютерних систем
Кафедра _____Інформаційних систем_____

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ _Андрій_Форсюк_	_____ Сергій_Чумаченко
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
«___» _____ 202_р.	«___» _____ 202_р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності_____122 «Комп'ютерні науки»_____
(код та назва спеціальності)
освітньо-професійної програми_Інформаційні управляючі системи та технології
на тему:_Дослідження та розроблення сервіса керування системою закладів громадського харчування

Виконав: здобувач__2__ курсу, групи__ІС-2-3М___

_____Шевчук Олександр Сергійович_____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник _____Горлова Тетяна Михайлівна_____ _____
(прізвище , ім'я та по батькові повністю) (підпис)

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

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

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

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

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

Здобувач_____
(підпис)

Київ - 2022_р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) _Автоматизації і комп'ютерних систем_____
Кафедра ____Інформаційних систем_____
Освітній ступінь __магістр_____
Спеціальність_____122 «Комп'ютерні науки»_____
(код і назва)
Освітньо-професійна програма _Інформаційні управляючі системи та технології_____
_____
(назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри Інформаційних систем

_____Чумаченко С. М.
"_11_" _листопада_____2021 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

_____Шевчука Олександра Сергійович _____
(прізвище, ім'я, по батькові)

1. Тема роботи ___Дослідження та розроблення сервісу керування системою закладів громадського харчування_____

керівник роботи Горлова Тетяна Михайлівна, к.т.н., с.н.с._
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "11"листопада 2021р. №884-КС

2. Строк подання здобувачем роботи_"02" лютого 2022 року_____

3. Вихідні дані до роботи _інформація та статистика про роботу мережі ресторанів, існуючий веб-сайт закладу громадського харчування _____

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)__
1) Розгляд теоретичних основ_____
2) Дослідження діяльності ресторану та напряму діяльності_____
3) Дослідження задачі та методів її вирішення_____
4) Розробка модулю «Веб-сайт ресторану» _____

5. Перелік графічного матеріалу
_Функціональні моделі, схеми, блок-схеми алгоритмів, зображення інтерфейсу користувача_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Горлова Т.М. , доцент	12.11.2021	12.11.2021
2	Горлова Т.М. , доцент	12.11.2021	12.11.2021
3	Горлова Т.М. , доцент	12.11.2021	12.11.2021

7. Дата видачі завдання__“11” листопада 2021 року _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Розгляд теоретичних основ	12.11.2021	виконано
2	Дослідження діяльності мережі ресторанів та напряму діяльності	17.12.2021	виконано
3	Дослідження задачі та методів її вирішення	28.12.2021	виконано
4	Розробка модулю «Веб-сайт ресторану»	15.01.2022	виконано
5	Оформлення роботи	17.01.2022	виконано
6	Оформлення автореферату	19.01.2022	виконано
7	Розробка презентації	20.01.2022	виконано

Здобувач _____ _Шевчук О.С.___
(підпис) (прізвище та ініціали)

Керівник роботи _____ __Горлова Т.М._
(підпис) (прізвище та ініціали)

Анотація

Магістерська робота «Дослідження та розроблення сервіса керування системою закладів громадського харчування», розроблена Шевчуком О.С. та складається з 61 сторінок, 1 таблиці, 23 рисунків, 3 додатків, 2 літературного джерела та 11 інтернет ресурсів, які були використані в роботі.

У роботі досліджено та проаналізовано ресторани “Черноморка”, та їх сервіси, розглянуті переваги та недоліки закладів. Описано методи збільшення рентабельності ресторанів за допомогою сервісів.

Розроблено веб-додаток, який можна використати для будь-якої мережі закладів та розширює можливості як для підприємців, так і для користувачів.

КЛЮЧОВІ СЛОВА: ВЕБ-ДОДАТОК, ЗАКЛАДИ ГРОМАДСЬКОГО ХАРЧУВАННЯ, АНАЛІЗ, РЕНТАБЕЛЬНІСТЬ.

Annotation

This Master's Degree work, a Research and development of service management system of catering establishments, developed by Shevchuk O.S., consists of 61 pages, 1 table, 23 figures, 3 appendices, 2 literature sources and 11 Internet resources that were found in the robots.

This work investigates and analyzes the restaurant "Chornomorka", their website, examines advantages and disadvantages of restaurants. Methods for increasing the profit for restaurants by website are described.

A web application that can be used for any restaurants and add new functionalities for businessmen and clients.

KEY WORDS: WEB APPLICATION, CATERING INSTITUTIONS, ANALYSIS, PROFITABILITY.

ЗМІСТ

Перелік умовних скорочень

Вступ 9

Розділ 1. Системний аналіз ресторану “Черноморка” 13

1.1. Загальна характеристика 13

1.2. Історія підприємства “Черноморка” 14

1.3. Організаційна структура підприємства 15

Розділ 2. Дослідження рентабельності ресторану “Черноморка” 17

2.1. Рентабельність та її види 17

2.2. Види методів збільшення рентабельності ресторану 19

2.3. Збільшення рентабельності ресторану “Черноморка” 214

2.4. Ефективність вжитих методів підвищення рентабельності 28

Розділ 3.Проектування системи 29

3.1. Клієнт 29

3.2. Сервер 32

3.3. API системи 39

3.4 База даних 42

3.5. Admin-клієнт 43

Розділ 4.Інструкція користувача 45

4.1. Інтерфейс користувача 45

4.2. Недоліки та удосконалення 52

Висновки 57

Список використаної літератури 58

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

SPA - Single-page Application

HTTP (Hyper Text Transfer Protocol) - поширений протокол передачі даних

HTTPS (Hyper Text Transfer Protocol Secured) - розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки документів.

API (Application Programming Interface) - код, який дозволяє двом програмним програмам спілкуватися один з одним

REST API (Representational State Transfer) - архітектурний стиль взаємодії компонентів розподіленого додатка в мережі.

SPA (Single page Application) - це додаток, який працює всередині браузера і не потребує перезавантаження сторінки під час використання

API endpoint - це точка, в якій інтерфейс прикладної програми (API) підключається до програмного забезпечення.

CSR (Client Side Rendering) - вид візуалізації, що дозволяє робити веб-сайти повністю виведеними в браузері за допомогою JavaScript.

JS (Javascript) - мова програмування

JSON (Javascript Object Notation) - це легкий формат для зберігання та транспортування даних

npm (Node package manager) - це програмне забезпечення, яке дозволяє встановлювати та керувати залежностями проекту.

yarn (Yet Another Resource Negotiator) - це альтернатива до npm.

Вступ

У наш час ресторанний бізнес — це не просто заклад громадського харчування, а сфера дозвілля, що включає безліч компонентів, не враховуючи які можно легко програти в запеклій конкурентній боротьбі.

Важливо зауважити, що ресторанний бізнес доступний не тільки для великих корпорацій, наприклад Макдональдс, але й для малих, навіть сімейних підприємств, які можуть працювати за франшизою або відкрити свій власний ресторан.

Оскільки, прибуток - це одна з найважливіших складових будь-якої підприємницької діяльності, необхідно оцінювати та аналізувати прибутковість ресторанного бізнесу.

Слід зауважити, що середній показник доходності залежить від різних факторів. Ніхто не може передбачити точний прибуток вашого ресторану. Він змінюватись залежно від розташування, сезонних поставок певних товарів, економічного становища вашого регіону, загальних тенденцій та інших факторів. Для того щоб дізнатись дійсну оцінку рентабельності конкретного ресторанного бізнесу, необхідно провести його детальний системний аналіз, знайти недоліки та переваги.

В даній магістерській роботі досліджено та проаналізовано рентабельність ресторанного бізнесу, визначено недоліки та переваги, визначено шляхи збільшення рентабельності та їх розробка.

Зв’язок роботи з науковими програмами, планами, темами. Наукова робота виконана згідно з планом наукових досліджень на базі кафедри інформаційних систем Національного університету харчових технологій (НУХТ).

Предметом дослідження є рентабельність та дохідність ресторанного бізнесу, методи їх підвищення.

Мета та завдання дослідження. Метою даної магістерської роботи є дослідження та аналіз рентабельності ресторанного бізнесу та розробка сервісу, який покращить показник рентабельності.

Для досягнення мети, необхідно виконати наступні **завдання**:

- дослідження діяльності закладів громадського харчування
- аналіз існуючих сервісів керування системою закладів громадського харчування
- дослідження рентабельності ресторану та визначити методи її підвищення
- розробка бажаного сервісу керування системою закладів громадського харчування, який покращить рентабельність ресторанного бізнесу
- інтеграція існуючих закладів громадського харчування в розроблений сервіс

Методи дослідження. Для оцінювання рентабельності ресторанного бізнесу було використано експрес-аналіз. Реалізація покращення рентабельності закладу - це розроблений веб-додаток, який спростить бізнес процеси як для підприємця, так і для клієнта.

Наукова новизна одержаних результатів. При виконанні поставлених задач було отримано такі наукові результати:

- сформовано оцінки показників рентабельності для ресторанного бізнесу та методи їх підвищення
- розроблено новий підхід бронювання столика у закладах громадського харчування

Наукове значення роботи. Наукове значення магістерської роботи полягає в аналізі та використанні сучасних методів оцінки рентабельності ресторанного бізнесу.

Практичне значення отриманих результатів. Створено веб-сайт, який можно використати для мереж ресторанів “Черноморка”, що значно покращить рентабельність даних закладів.

Особистий внесок здобувача.

- Проведено аналіз та дослідження сучасних сервісів керування закладами громадського харчування
- Розроблено веб-додаток, який можно використати для будь-якої мережі ресторанів та спрощує роботу бізнесу та взаємодію клієнта.

- **Апробація результатів магістерської роботи.**

- Результати досліджень, викладені в магістерській роботі, доповідались та обговорювались на VIII Міжнародній науково-технічній Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», 25-26 листопада 2021 р. [Електронний ресурс] – К: НУХТ, 2021 – с.304.

- Тези: Шевчук О.С., Горлова Т.М. Дослідження та розроблення сервісів із керування системою закладів харчування. [Електронний ресурс] – К: НУХТ, 2021 – с.304. —стор. **286.**

- Режим доступу: https://nuft.edu.ua/naukova-diyalnist/naukovi-konferencii/

Структура та обсяг магістерської роботи. Магістерська робота складається з вступу, 4 розділів, висновку, списку використаних джерел та

додатків. Повний обсяг роботи з 61 сторінок, 1 таблиці, 23 рисунків, 3 додатки, 2 літературного джерела та 11 інтернет ресурсів.

Розділ 1. СИСТЕМНИЙ АНАЛІЗ РЕСТОРАНУ “ЧЕРНОМОРКА”

1.1. Загальна характеристика

Розвиток ресторанного бізнесу йде сьогодні повним ходом, а конкуренція у цій галузі неухильно зростає. Власники ресторанів змушені вигадувати нові маркетингові заходи, акції, презентації, слогани, здатні залучити клієнтів і допомогти в розвитку бізнесу. Існують наступні типи ресторанів:

- ресторани престижного класу
- повсякденні заклади
- сучасні ресторани (модерн)
- сімейний ресторан
- ресторан з елементами фаст-фуда або ресторан швидкого харчування
- фаст-фуд
- кафе
- буфети
- поп-ап ресторани (на винос)

В роботі розглядається ресторан «Черноморка» - це заклад, який можна віднести до сучасних або сімейних ресторанів, який є повно сервісним, загальнодоступним та з обслуговуючим персоналом. Атмосфера в ресторанах даного типу може сильно змінюватись в залежності від бренду та передбачуваної клієнтської бази, але більшість з них поділяють загальні якості:

- позиції у меню за помірною ціною

- столовий сервіз
- проста атмосфера
- унікальний декор

Ресторан «Черноморка» було засновано в 2013р. в місті Одеса. Мережа закладу поширилась, тепер «Черноморку» можна зустріти у Києві, Харкові та Вінниці.

1.2. Історія підприємства “Черноморка”

Перший заклад був відкритий у жовтні 2014 року. “Заклад був невеликий – на 65 кв.м. Спочатку це була звичайна крамниця, де покупці приходили забирати свіжу рибу. Невдовзі було вирішено поєднати лавку із рестораном.

У грудні 2014 року "Чорноморка" відкрила другу лавку. Незважаючи на те, що заклад був у дворах і знайти його було не так просто, потік був більшим, 10 столиків не вистачало. На старті щомісячний дохід досягав (після оплати всіх витрат) 70 000 грн. на місяць.

Далі "Чорноморка" перебралася на вулицю Преображенську. Це одна з найуспішніших крамниць.

Найлегшим відкриттям ресторану було на Подолі. Він коштував 3 млн грн, і з перших днів роботи у ресторані була повна посадка. За цією лавкою-рестораном пішла крапка на "Столичному ринку". Пробувала "Чорноморка" працювати у Львові на Шуварі. Але справа не пішла, бо львівський формат не передбачав приготування їжі. Натомість одеський формат на “Міському ринку їжі” підійшов, і у 2017 році з'явилася одеська філія, у 2019 році з'явилася “Селедечна” – проект, який став володарем срібної пальмової гілки.

На сьогоднішній день, мережа ресторанів «Черноморка» налічує понад 16 окремих закладів у різних містах, що підпорядковується приватному підприємцеві. У компанії працює понад 700 співробітників.

1.3. Організаційна структура підприємства

Одночасно усіма ресторанами важко керувати, тому кожен заклад має окремого директора для управління. На підприємстві є директор та його заступник, головний бухгалтер та керівник відділу постачання продуктів. Заступник директора займається відділом по роботі з клієнтами та обробки замовлень. Також йому підпорядковуються менеджери, кухари та офіціанти. Головний бухгалтер очолює відділ бухгалтерії. За вчасне перевезення, отримання та перевірку продуктів відповідає керівник відділу постачання, якому підпорядковується декілька вантажників.

Кожен працівник закладу має свої певні завдання та обов'язки, з якими він зустрічається кожен день та виконує.

Обов'язки директора:

- вивчення спроси клієнтів на кожну продукцію ресторану
- вивчення нових трендів продукції
- контролювання трудових та фінансових ресурсів закладу
- отримання певних ліцензій та сертифікацій, таких як якості продуктів та здійснення діяльності у сфері послуг харчування
- організація та планування діяльності закладу

Обов'язки керівника відділу бухгалтерії:

- ведення бухгалтерського обліку
- дотримання зазначених правил документообігу
- дотримання норм бухгалтерського обліку згідно законодавства
- оновлення та структуризація інформації у документах

Обов'язки керівника відділу по роботі з клієнтами:

- формування меню, розуміння складу та приготування страв
- швидке та ввічливе обслуговування клієнта
- пропозиція та оцінка кваліфікації персоналу обслуговування

Обов’язки керівника відділу постачання продуктів:

- замовлення продукції у постачальників
- введення накладної щодо продукції на кухні
- контроль обліку інформації щодо наявної продукції
- контроль обліку зіпсованої продукції

Було визначено, що в ресторані та по всій мережі ресторанів потрібно провести оцінку рентабельності та знайти шлях для підвищення цього показника.

Розділ 2.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ РЕНТАБЕЛЬНОСТІ РЕСТОРАНУ “ЧЕРНОМОРКА”

2.1. Рентабельність та її види

Рентабельність (дохідний, прибутковий) - відносний показник економічної ефективності. Рентабельність комплексно відбиває рівень ефективності використання фінансових, матеріальних та трудових ресурсів.

Коефіцієнт рентабельності рахують як відношення прибутку до активів або ресурсів, що її формують. Даний коефіцієнт може виражатися як у прибутку на одиницю вкладених коштів, і у прибутку, яку несе у собі кожна отримана грошова одиниця. Показник рентабельності вимірюють у відсотках.

Загальна формула розрахунку коефіцієнта:

$$\text{Крп} = \frac{\text{прибуток від продажу}}{\text{виручка від реалізації}} \times 100\%$$

Формула розрахунку за даними бухгалтерського балансу:

$$\text{Крп} = \frac{\text{стр.050}}{\text{стр.010}} \times 100\% \quad \text{Крп} = \frac{\text{стр.050}}{\text{стр.010}} \times 100\%$$

, де стр.050 та стр. 010 - це звіти про прибутки та збитки.

Розглянемо різні види рентабельності:

Рентабельність продажів. Це коефіцієнт рентабельності, який показує частку прибутку в кожному зароблені гривні (будь-якій іншій валюті).

Зазвичай розраховується як відношення операційного прибутку (прибутку до оподаткування) за певний період до вираженого в грошових коштах обсягу продажу за той же період.

Рентабельність продаж = (операційний прибуток / обсяг продажів)

Рентабельність продажів є індикатором цінової політики підприємства міста і його можливості контролювати витрати. Відмінності у конкурентних стратегіях та продуктових лінійках викликають значну різноманітність значень рентабельності продажів у різних компаніях. Часто використовується з метою оцінки операційної ефективності компаній. Однак слід враховувати, що при рівних значеннях показників виручки, операційних витрат та прибутку до оподаткування у двох різних фірм рентабельність продажів може сильно різнитися, внаслідок впливу обсягів процентних виплат на величину чистого прибутку.

Рентабельність продажів (ROS) - відношення операційного прибутку (прибутку від продажів) (EBIT) компанії до її виручки (Sales).

ROS = EBIT/SALES= операційний прибуток/Виручка * 100 %

Рентабельність продукції. Це прибуток від продукції / собівартість реалізованої продукції

Рентабельність активів. Це чистий прибуток / середня вартість активів

Рентабельність необоротних активів. Це чистий прибуток / середня вартість необоротних активів

Рентабельність власного капіталу. Це чистий прибуток / середня вартість власного капіталу.

2.2. Види методів збільшення рентабельності ресторану

Ресторанний бізнес - одна з тих сфер, які найбільше постраждали від карантинних обмежень та падіння попиту. Мережа ресторанів “Черноморка” не виняток. Проблема також у тому, що рівень доходів населення впав, купівельна спроможність відвідувачів закладів громадського харчування перебуває в низькому рівні.

Прибутковість ресторанів, що розташовані поруч один з одним, може відрізнятися в рази. І найчастіше справа не в площі закладу чи кількості вкладених у нього грошей, а у грамотному підході до обслуговування клієнтів, навчанні співробітників та створенні загального позитивного іміджу.

Розглянемо наступні загальні методи збільшення рентабельності ресторанного бізнесу:

1. Підвищити середній чек

Відмінність ресторану від магазину чи супермаркету в тому, що людина приходить сюди із чітко оформленою потребою поїсти чи випити. Користуючись цим нюансом, можна повністю задовольнити бажання клієнта в їжі, одночасно заробивши добрий прибуток.

Ресторан “Черноморка” витримує ідеальну межу у ціні середнього чеку, як для підприємця так і для клієнтів. Збільшення чеку може призвести до занадто завищених цін у закладі.

2. Оптимізація ресторанного меню

Хороше меню продає страви саме собою. Його структура повинна враховувати психологічні особливості поведінки клієнта в такий спосіб, щоб збільшувати середній чек без зусиль персоналу. Наприклад, у правому

верхньому кутку слід розміщувати найбільш прибуткові страви, тому що в цю зону людина наглядає найчастіше.

Меню ресторану “Черноморка” гарне складене та ненавантажене з вказанням акцій та спеціальних пропозицій. Страви розбиті по категоріям.

3. Використання технік допродажу

Відвідуючи кафе, людина вже готується розлучитися із певною сумою грошей. І завдання офіціанта – зробити так, щоб відвідувач зробив це добровільно та залишився задоволеним.

Найпростішим способом стимулювати людину на витрати – запропонувати додаткову корисну страву до вже зробленого замовлення. Наприклад: «Можливо, принести келих пива до замовлених Вами «курячих крилець?». Таким чином, людина відчує щиру турботу офіціанта та зможе побачити свою приховану потребу.

Якщо після вживання основної страви клієнт не поспішає йти, варто підійти і запропонувати десерт. Можливо, у людини залишилося ще місце у шлунку та гроші в гаманці, які він готовий витратити.

У ресторані “Черноморка” клієнти коштують 1-2 страви на людину максимум, так як заклад має профіль морепродуктів, тому десерти і інші “закуски” відсутні.

4. «Вигідні» спецпропозиції

Клієнти кафе, що особливо засиділися, рідко використовують логіку при виборі страв, але всім їм хочеться спробувати щось незвичайне. Для таких випадків меню можна передбачити спецпропозиції, які принесуть прибуток без додаткових витрат.

Ресторан “Черноморка” активно використовує спеціальні пропозиції, акції та бонуси. Офіціанти повідомляють про нові з них.

5. Мотивація офіціантів

Рівень середнього чека в ресторані залежить від умінь офіціанта запропонувати страви і зробити допродажу. Щоб співробітники використовували свої навички, можна стимулювати їх до цього такими способами:

- запропонувати процент від продажу. Скупитися спочатку не потрібно – офіціант повинен відчути різницю в доходах за свої старання.
- проводити щотижневі/щомісячні змагання у колективі, закріплюючи найбільш прибуткові столики за переможцями або видаючи премії.
- просити офіціантів вести облік чайових «для себе», щоб вони переконувалися, що рівень «добавки» залежить від середнього чека.

У ресторані “Черноморка” завжди працюють привітні та професіональні офіціанти.

6. Продавати алкоголь за рахунок закусок

Одним із способів збільшення середнього чека для кафе, орієнтованих на вживання алкогольних напоїв, є складання розширеного меню закусок та його постійне модифікування.

Ці страви дозволяють зробити вживання алкоголю приємнішим, у компанії, що відпочиває, створюється атмосфера спілкування та експериментів. За наявності безлічі закусок є можливість, що клієнти замовляють на пробу відразу кілька позицій. Вартість цих страв нижча, але через високу швидкість приготування вони практично не завантажують

кухню. А експерименти зі складом страв дозволять зберегти інтерес до закладу постійних клієнтів.

Ресторан “Черноморка” не спеціалізується на випивках, тому “закуски до пива” відсутні в меню.

7. Припинення крадіжок

Розкрадання у громадському харчуванні не викорінити, але мінімізувати можна. Проблема полягає не тільки в менталітеті наших людей, а й у специфіці сфери — співробітник кухні може зробити будь-яку страву гіршою за якістю або меншою за порцією. Виявити несумлінний персонал багато в чому допоможе онлайн-каса.

Автоматичний облік товарів та грошей дозволяє перевірити склад та касу у будь-який момент часу. Об'єднавши відеоспостереження з функціями онлайн-кас, можна контролювати дії обслуговуючого персоналу у віддаленому режимі. Наприклад, через особистий кабінет власник вибирає всі чеки зі скасуванням товару. На кожному з них зазначена дата та час, орієнтуючись на які можна переглянути відповідний проміжок на відеозаписі, та визначити факт махінації.

У ресторані “Черноморка” є касовий апарат та камери відеоспостереження. Тому ймовірність крадіжок зменшена.

8. Служба доставки

В сучасних реаліях карантинних обмежень, клієнти все більше і більше надають перевагу доставки їжі на дом через сервіси доставки такі як “Glovo”, “Raketa ” та інші. Це стало дуже вигідною та прибільною послугою. За статистикою у великих містах приблизно 65% людей користуються сервісами доставки.

Ресторан “Черноморка” не має такої послуги. Додавання її збільшить рентабельність ресторану.

9. Онлайн замовлення та веб-сайт

Зараз клієнти вважають за краще оформляти такі замовлення онлайн. Отже, наявність веб-сайту закладу, здатного виконувати цю функцію, може в рази збільшити дохід.

Багато власників ресторанів стурбовані тими складнощами та витратами, з якими їм доведеться зіткнутися під час створення та подальшого використання сайту.

«А чи потрібен взагалі моєму закладу сайт?» - таке питання ставить собі, мабуть, кожен власник ресторану чи кафе. Якщо ви орієнтуєтесь на високий рівень закладу або просто прагнете вищих досягнень, то відповідь на це питання має бути, безумовно, позитивною. Сайт ресторану – це його головна візитка, книга відгуків та рекомендація. Забуваючи про це, ресторани часто забувають і про сам сайт, закінчуючи роботу з новим ресурсом після його створення та публікації в мережі.

Слід ставитися до сайту як до найкращого продавця та не забувати, що сайт – це головний продавець, адміністратор та представник в Інтернеті (працюючий 24 години на добу 7 днів на тиждень 365 днів на рік)!

2.3. Збільшення рентабельності ресторану “Черноморка”

Розглянувши всі методи збільшення рентабельності ресторанного бізнеса, можна зробити висновок, що ресторан “Черноморка” по більшості параметрам виглядає дуже добре та слідує найкращим практикам цього складного бізнесу. Єдиними місцями покращення залишились сервіс доставки їжі та веб-сайт. Розглянемо існуючий веб-додаток закладу.

Рисунок 2.1 Головна сторінка ресторану “Черноморка”

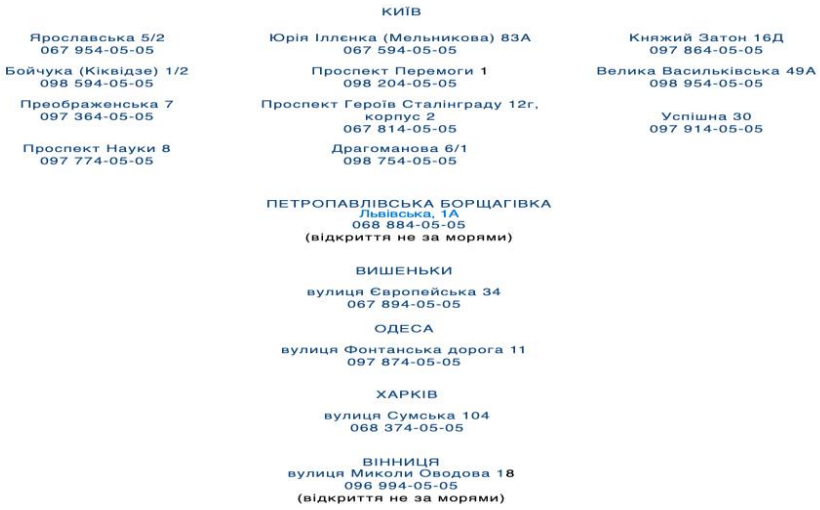


Рисунок 2.2 Головна сторінка ресторану “Черноморка”

Розглянемо вкладку меню та завантажимо PDF документ з стравами.

Рисунок 2.3 Меню ресторану “Черноморка”

Щоб зробити замовлення користувач повинен:

- зайди на сервіс закладу
- перейти в розділ меню (PDF документ) та обрати страви
- запам'ятати страви
- повернутись на головну сторінку
- знайти розділ контакти
- зателефонувати замовити страви та забронювати

Перерахуємо переваги веб-сайту “Черноморка”:

- простий та ненавантажений непотрібною інформацією
- має посилання на соціальні мережі та мобільні додатки
- вказані спеціальні пропозиції та акції
- вказані всі локації та контакти закладів мережі
- місце на певну годину та кількість людей

Перерахуємо недоліки веб-сайту “Черноморка”:

- меню створено як PDF документ без фото страв з дуже не зручним розширенням та малим шрифтом, через це важко орієнтуватись з смартфона
- інформація одноманітна, не розділена по різним сторінкам, не виділяється та важко читабельна
- на сайті немає можливості побачити розташування ресторанів в інтегрованій у веб-сайт google maps
- немає можливості зробити доставку їжі

Закладу “Черноморка” дійсно краще приділити більше уваги їх веб-сайту. Проаналізуємо статистику користування веб-сайтами:

- 77% клієнтів відвідують веб-сайт ресторану, перш ніж вони замовляють або бронюють
- 68% клієнтів були розчаровані рестораном через веб-сайт
- 62% клієнтів були розчаровані від замовлення доставки або вивезення через веб-сайт
- 30% клієнтів покидають веб-сайт, якщо меню нечитабельне
- 33% клієнтів покидають веб-сайт, якщо на ньому важко переключатись
- 33% клієнтів покидають веб-сайт, якщо він виглядає застарілим
- 45% клієнтів оцінюють ресторан по фото страв
- 36% клієнтів відмовляються від відвідування ресторану через фото страв

Розглянемо статистику клієнтів за віком:

- 59% підлітків у віці від 18 до 24 років і 55% 25-34-річних спеціально шукають фотографії їжі на веб-сайті.
- 50% відвідувачів 18-24 років і 44% 25-34-річних, які кажуть, що веб-сайт ресторану відлякував їх від відвідування ресторану, кажуть, що це тому, що фотографія їжі не була привабливою.

- 49% 18-24-річних і 41% 25-34-річних, які замовляють їжу або доставку, стверджують, що веб-сайт ресторану перешкоджає їм замовляти їжу.

Згідно статистики, 77% клієнтів знайомляться зі веб-сайтом перш ніж відвідати заклад. Приблизно 40% оцінюють веб-сайт по фото страв. Тобто ресторанна мережа “Чорноморка” з 10 клієнтів втрачає 3 з них - це 30% перших клієнтів лише через відсутність фото страв.

Веб-сайту “Черноморка” не вистачає наступного функціоналу:

- Можливість перегляду розміщення столиків у закладі
- Бронювання столика без телефонного дзвінка на певну годину та кількість людей
- Вибір страв та перегляд їх інгредієнтів з можливістю вилучення деяких з них
- Реєстрація користувача та створення власного кабінету
- Оцінювання страв та минулих замовлень
- Оплата замовлення
- Вибір закладу
- Доставка їжі
- Адмін сервіс для управління основним сервісом

2.4. Ефективність вжитих методів підвищення рентабельності

Оскільки, наші методи не збільшують середній чек закладу, а збільшують загальну кількість перших клієнтів та лояльність постійних, прорахуємо у процентному співвідношенні на скільки теоретично збільшиться кількість відвідувачів у закладі.

Перші клієнти. 77% клієнтів дивляться веб-сайт закладу перед тим як в ньому перший раз побувати. З них 40% надають перевагу іншим закладам, якщо немає чітких та розбірливих зображень страв та інгредієнтів до них. Тобто, 0,77*0.4 = 0.3. Веб-сайт збільшує кількість перших клієнтів на 30%.

Постійні клієнти. Постійні клієнти з радістю відвідують заклад раді страв, але інколи хочеться скуштувати улюблені страви і удома. Згідно статистики у великих містах 65% людей користуються сервісами доставки. Тобто лояльність, постійних клієнтів збільшиться на 65% через додавання можливості доставки.

Розділ 3. Проектування системи

3.1 Клієнт

Для того щоб створити SPA-додаток, написаний за допомогою React.js бібліотеки, використано набір інструментів Create React App.

Create React App - зручне середовище для вивчення фреймворка React. Цей прилад налаштовує нові можливості Javascript, оптимізовує додаток для продакшена.

Для того щоб створити проект, виконано команду:

```
npx create-react-app my-app
```

Для розробки клієнтської частини веб-додатку необхідні наступні популярні пакети:

- axios - Promise HTTP-клієнт для браузера та node.js
- redux - контейнер стану для додатків JavaScript.
- react-router - предоставляє основний функціонал маршрутизації для React Router
- crypto-js - Javascript бібліотека для шифрування
- lodash - сучасна бібліотека утиліт JavaScript, яка забезпечує модульність та продуктивність
- semantic-ui - це фреймворк інтерфейсу, призначений для тематизації

У веб-додатку здійснення HTTP-запитів для отримання або збереження даних реалізовано за допомогою бібліотеки axios. Наступний фрагмент коду створює екземпляр HTTP-клієнта, налаштовує за умовчанням та встановлює

інтерсептори (дозволяють запускати або змінювати запит до того, як запит та досягне місця призначення) для запиту та відповіді:

```
import axios from 'axios';
import { store } from 'redux/store/configureStore';
const instance = axios.create({
 headers: { 'Content-Type': 'application/json' },
 baseURL: 'https://eatngo.herokuapp.com/',
 responseType: 'json'
});
instance.interceptors.request.use(
 (config) => {
   const AUTH_TOKEN = JSON.stringify(store.getState().auth);
   if (AUTH_TOKEN) {
     config.headers['Authorization'] = AUTH_TOKEN;
   }
   return config;
 }, error => Promise.reject(error)
);
instance.interceptors.response.use(
 response => response,
 ({ response = { status: 500 } }) => {
   const { status } = response;
   const redirect = redirects[status];
   if (redirect) document.location = redirect;
   return Promise.reject(response);
 }
);
```

Управління станом додатку реалізовано за допомогою бібліотеки redux, redux-thunk для здійснення HTTP-запитів у redux та redux-persist для збереження об’єкта стану Redux у сховищі Persistent. Наступний фрагмент коду:

- створює конфігурацію для persistent-сховища
- створює головний редьюсер, який задає, як змінюється стан програми у відповідь на дії, надіслані в сховище
- створює сховище
- створює persistent-сховище

```
import { createStore, applyMiddleware, compose } from 'redux';

import { persistStore, persistReducer } from 'redux-persist';

import logger from 'redux-logger';

import thunk from 'redux-thunk';

import storage from 'redux-persist/lib/storage';

import autoMergeLevel2 from 'redux-persist/lib/stateReconciler/autoMergeLevel2';

import rootReducer from 'redux/reducers';

const persistConfig = {

key: 'root',

storage: storage,

whitelist: ['auth'],

stateReconciler: autoMergeLevel2

};

const pReducer = persistReducer(persistConfig, rootReducer);

export const storeFactory = () => createStore(pReducer, composeEnhancers(applyMiddleware(thunk), applyMiddleware(logger)));

export const store = storeFactory();
```

```
export const persistor = persistStore(store);
```

У веб-додатку реалізована авторизація та реєстрація. Кожен користувач хоче бути впевненим, що його персональні дані залишаються засекреченими. Для цих цілей, потрібно шифрувати дані користувача такі як пароль. Для цього використано бібліотеку шифрування crypto-js. Наступний фрагмент коду шифрує дані на основі стандарту кодування двійкових даних за допомогою тільки 64 символів ASCII, коду автентифікації повідомлення на основі хеша та секретного слова “secret”:

```
import CryptoJS from 'crypto-js';

export const crypting = (value) => {

   return CryptoJS.enc.Base64.stringify(CryptoJS.HmacSHA256(value, 'secret'));

};
```

3.2 Сервер

На серверній частині веб-сайту використано невеликий, open-source та гнучкий фреймворк веб-додатків Node.js - Express.js. Даний фреймворк як і React предоставляє зручний інструмент для швидкого створення "каркасу" веб-додатка. Для того щоб скористатись ним створити проект необхідно виконати наступні команди:

- `npm install express-generator -g`
- `express my-app`

Для розробки серверної частини веб-додатку необхідні наступні популярні пакети:

- express

- mongoose - інструмент моделювання об'єктів MongoDB, призначений для роботи в асинхронному середовищі
- nodemailer - предоставляє функціонал для надсилання електронних повідомлень на пошту на Node.js
- crypto-js - Javascript бібліотека для шифрування

Відправлені запити з клієнтського додатку повинні бути оброблені маршрутизацією. Маршрутизація визначає, як додаток відповідає на клієнтський запит до конкретної адреси (URI). Маршрутизація розбиває API endpoints на певні роути такі як: /users, /dishes, /places, /offers, /order та інші.

Кожен з цих роутів має додаткові шляхи або параметризовані адреси та відповідні обробники. У наступному фрагменті коду наведений приклад user-route та його обробників на різні параметри та HTTP-методи:

```
const user = require('../controllers/usersController')

module.exports = app => {

 app

   .route('/registration')

   .post(user.userRegistration);

 app

   .route('/authenticate')

   .post(user.authenticate);

 app

   .route('/users')

   .get(user.getUserInfo)

   .put(user.updateUserInfo)

 app

   .route('/verify/:code')

   .get(user.verification)
```

```
 app
   .route('/forgotPassword')
   .post(user.sendEmailToResetPassw)
 app
   .route('/reset')
   .post(user.resetPassword)
};
```

Усі розділені маршрути комбінуються в один корінний:

```
const routeSpecialOffers = require('./specialOffersRouters');
const routesOrder = require('./orderRoutes');
const routesTables = require('./tablesRoutes');
const routesUsers = require('./usersRoutes');
const routesUserOrders = require('./userOrdersRoutes');
const routesAdmin = require('./adminRoutes');
const routesDishes = require('./dishesRoutes');
const routesPlaces = require('./placesRoutes');
module.exports = app => {
   routeSpecialOffers(app);
   routesOrder(app);
   routesTables(app);
   routesUserOrders(app);
   routesDishes(app);
   routesPlaces(app);
   routesUsers(app);
   routesAdmin(app);
}
```

та у головному файлі index.js налаштовують маршрутизацію:

```
const routes = require('./src/routes/setUpRoutes');

const app = express();

routes(app);
```

На кожен маршрутний роут є відповідний обробник. Наступний фрагмент коду містить усі контролери до /user маршруту:

```
const mongoose = require('mongoose');

const User = mongoose.model('User');

const userService = require('../services/userService');

const queryWrapper = require('../utils/queryWrapper');

exports.getUserInfo = async (req, res) => {

 queryWrapper(req, res, userService.getUserInfo, userService.errHandler);

};

exports.updateUserInfo = async (req, res) => {

 queryWrapper(req, res, userService.updateUserInfo, userService.errHandler);

};

exports.userRegistration = async (req, res) => {

 queryWrapper(req, res, () => userService.registerNewUser(User, req),
userService.errorHandler);

};

exports.authenticate = async (req, res) => {

 queryWrapper(req, res, () => userService.authUser(req), userService.errorHandler);

};

exports.verification = async (req, res) => {

 queryWrapper(req, res, () => userService.verificationEmail(req, res));

};
```

```
exports.sendEmailToResetPassw = async (req, res) => {

 queryWrapper(req, res, () => userService.sendEmailToResetPassw(req), userService.errorHandler);

};

exports.resetPassword = async (req, res) => {

 queryWrapper(req, res, () => userService.resetPassword(req, res), userService.errorHandler);

};
```

Для зручної роботи з нереляційною базу даних MongoDB використовується бібліотека Mongoose, яка управляє зв'язками між даними, забезпечує перевірку схеми і використовується для перекладу між об'єктами в коді та представлення цих об'єктів у базі даних. До кожної сущності у базі даних є відповідна схема. Наступний фрагмент коду демонструє схему документа користувача:

```
const mongoose = require('mongoose');

const { Schema } = mongoose;

const usersSchema = new Schema(

 {

   firstName: {

     type: String,

     default: ""

   },

   lastName: {

     type: String,

     default: ""

   },

   email: {

     type: String,
```

```
      required: 'email cannot be blank'
    },
    confirm: {
      type: Boolean,
      default: false
    },
    password: {
      type: String,
      required: 'password cannot be blank'
    },
    phoneNumber : {
      type: String,
      default: ""
    },
    userImage: {
      type: String,
      default: ""
    }
  },
  { collection: 'users' }
);
module.exports = mongoose.model('User', usersSchema);
```

Оскільки у користувача є можливість авторизуватись та реєструватись на клієнтській частині, слід обробляти, зашифровувати і розшифровувати дані. Для цих цілей використовується така ж бібліотека шифрування як і на клієнтській частині - crypto-js. Наступний фрагмент коду створює функції шифрування та розшифрування даних:

```
const CryptoJS = require("crypto-js");

exports.secret = 'newSecret';

exports.deCryptPassword = (password, secret = secret) => {

   const bytes  = CryptoJS.AES.decrypt(password.toString(), secret);

   const plainText = bytes.toString(CryptoJS.enc.Utf8);

   return plainText;

};

exports.enCryptPassword = (password, secret = secret) => {

   const cipherText = CryptoJS.AES.encrypt(password, secret);

   return cipherText;

};
```

Приклад їх використання у авторизації адміністратора:

```
authentication = async adminData => {

   const admin = await dbService.getOneElementByField(Admin, {name: adminData.name});

   if(!admin) throw new Error('Admin name or password is incorrect');

   const decryptedPassword = cryptor.deCryptPassword(admin.password, cryptor.secret);

   if(decryptedPassword === adminData.password){

       return true;

   } else {

       throw new Error('Admin name or password is incorrect');

   }

}
```

Також слідує підключитись до бази даних, наступний фрагмент коду виконує це:

```
exports.connectToDB = () => {
```

```
 return new Promise((resolve, reject) => {
   mongoose.connect(url, { useNewUrlParser: true, useFindAndModify: false, dbName: dbName }, (err) => {
     err ? reject(err) : resolve();
   });
 })
};
```

де url - адреса до сервісу бази даних, а dbName - назва бази даних.

3.3 API серверу

Розглянемо основні API endpoints нашого серверу:

Таблиця 3.1 API endpoints

url	method	description
/dishes	GET	отримати усі страви
/dishes	POST	створити страву
/dishes/:dishId	GET	отримати окрему страву
/dishes/:dishId	PUT	оновити окрему страву
/dishes/:dishId	DELETE	видалити окрему страву
/dishes/:filter/:count/:category	GET	отримати список відфільтрованих страв
/places	GET	отримати всі ресторани
/places	POST	створити ресторан
/places/:placeId	GET	отримати окремий ресторан
/places/:placeId	PUT	оновити окремий ресторан
/places/:placeId	DELETE	видалити окремий ресторан
/offers	GET	отримати усі спеціальні акції

/offers	POST	створити акцію
/offers/:specialOffersId	GET	отримати окрему акцію
/offers/:specialOffersId	PUT	оновити окрему акцію
/offers/:specialOffersId	DELETE	видалити окрему акцію
/tables	GET	отримати список всіх столів
/tables/:restId	GET	отримати список столів окремого ресторану
/registration	POST	зареєструвати користувача
/authenticate	POST	аутентифікувати користувача
/users	GET	отримати інформацію користувача
/users	PUT	оновити інформацію користувача
/verify/:code	GET	верифікувати користувача
/forgotPassword	POST	відправити повідомлення на пошту для відновлення паролю
/reset	POST	скинути пароль користувача
/myorders	POST	отримати замовлення користувача
/rating	POST	додати оцінку користувача на страву
/order	POST	створити замовлення
/order	GET	отримати усі замовлення
/order/:orderId	PUT	оновити замовлення

З даним API повинні вміти працювати клієнти. Список API-endpoints може легко масштабуватись та включати нові.

3.4 Бази даних

Базу даних потрібно розмістити на окремому сервері. Для цих цілей використано безкоштовний хостинг баз даних на https://cloud.mongodb.com/. Дана система надає безкоштовний керований сервер, оптимізований для роботи бази даних. Підключення до бази даних описані на серверній частині веб-додатку.

Створено єдину базу даних, яка містить декілька колекцій:

- users
- places (ресторани)
- dishes
- specialOffers
- orders
- tables

Кожна з колекцій містить відповідні документи. Розглянемо вид user-документа:

1. _id - автоматично створюваний ідентифікатор для документа
2. firstName - ім'я користувача
3. lastName - прізвище користувача
4. confirm - чи верифікований користувач
5. phoneNumber - номер телефону користувача
6. userImage - посилання на фото користувача, якщо він завантажив його у персональному кабінеті
7. email - електронна пошта користувача
8. password - зашифрований пароль користувача

Вид документів в базі даних повністю співпадає з Mongoose-схемами, які використовуються на серверній частині.

3.5 Адмін-клієнт

Для керування та встановлення пакунків та бібліотек на Admin-клієнті використано yarn (як на стороні звичайного клієнта). Angular предоставляє альтернативний інструмент до Create React App, за допомогою якого можно легко створювати SPA-додаток - Angular CLI.

Щоб створити каркас проекту, слід виконати наступні команди:

- `npm install -g @angular/cli`
- `ng new my-first-project`

Для розробки Admin-клієнтської частини веб-додатку необхідні наступні популярні пакети:

- @angular/material - інфраструктура компонентів інтерфейсу та компонент Material Design для Angular веб-додатків.
- crypto-js - Javascript бібліотека для шифрування
- @angular/router - предоставляє основний функціонал маршрутизації для Angular

На відміну від звичайного клієнта, Admin-клієнт здійснює HTTP-запити для отримання або збереження даних реалізовано за допомогою бібліотеки нативного HTTP-клієнта. Наступний фрагмент коду містить функцію, яка робить запит на сервер, щоб аутентифікувати адміністратора:

```
import {HttpClient} from '@angular/common/http';

authenticateAdmin(name, password) {

   const admin = {

     name,
```

```
        password
      };
      return this.http.post(`${this.url}admin`, admin);
    }
```

де this.http містить екземпляр HTTP-клієнта, а this.url - адреса сервера.

Шифрування інформації адміністратора відбувається так як на звичайному клієнті:

```
import CryptoJS from 'crypto-js';

crypting = (value) => {
   return CryptoJS.enc.Base64.stringify(CryptoJS.HmacSHA256(value, 'secret'));
 }
```

4. Інструкція користувача

4.1 Інтерфейс користувача

Щоб відкрити веб-додаток, необхідно перейти на адресу у браузері, де розміщений клієнтська частина. Переходячи за адресою, користувач бачить головну сторінку веб-сайту, яка складається з:

1) шапки сайту, яка містить навігацію по додатку у розділи такі як: меню, усі ресторани, замовлення, акції, авторизація та особистий кабінет

Рисунок 4.1 Шапка сайту

2) рекламного банера, на якому знаходяться спеціальні пропозиції від закладів та додаткова навігація по веб-сайту

Рисунок 4.2 Рекламний баннер сайту

3) секції найпопулярніших страв з короткою інформацією (назва, ціна, рейтинг, фото) про них та кнопки “Show More” за допомогою якої здійснюється редирект на сторінку страв.

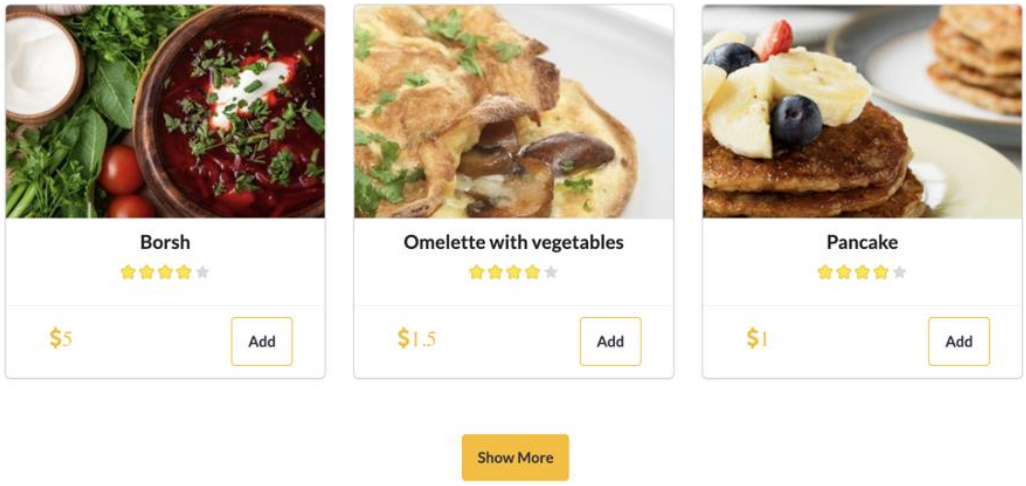


Рисунок 4.3 Найпопулярніші страви

4) секції ресторанів з короткою інформацію (адреса, фото, робочі години) про них та кнопки “View on map” за допомогою якої здійснюється редирект на сторінку ресторанів.

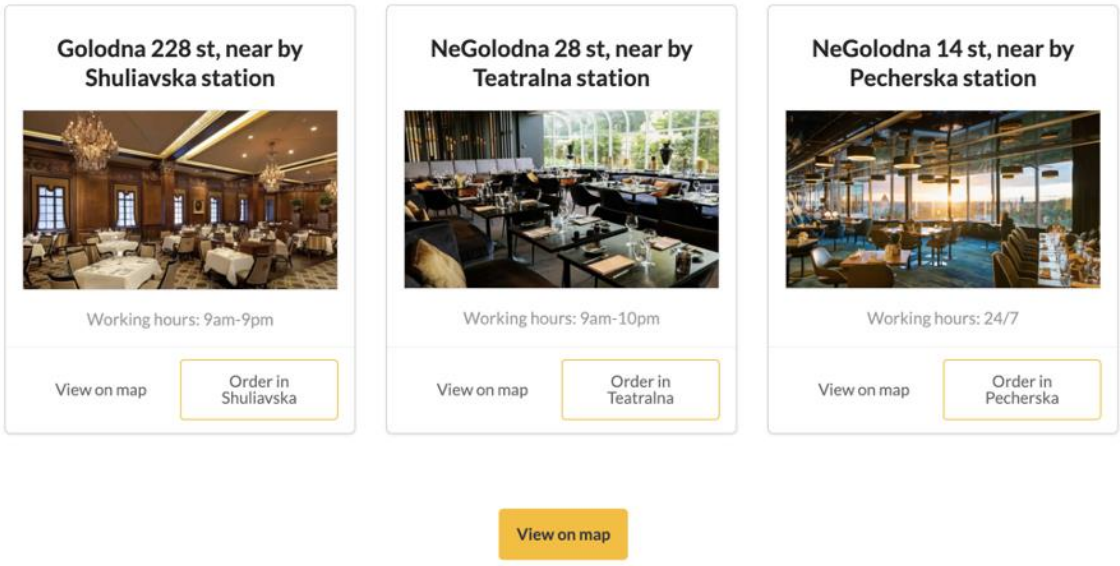


Рисунок 4.4 Ресторани

Для здійснення замовлення, користувач може натиснути “Book a table” у шапці сайту або у секції ресторанів на головній сторінці обрати один з них та натиснути кнопку “Order in ...”. У першому випадку, процес замовлення

почнеться з обиранням ресторану. Для того щоб перейти на наступний крок, слід натиснути “Next”.

Рисунок 4.5 Переключатель кроків замовлення

Наступним кроком являється вибір дати, години, кількості відвідувачів та столика ресторану.

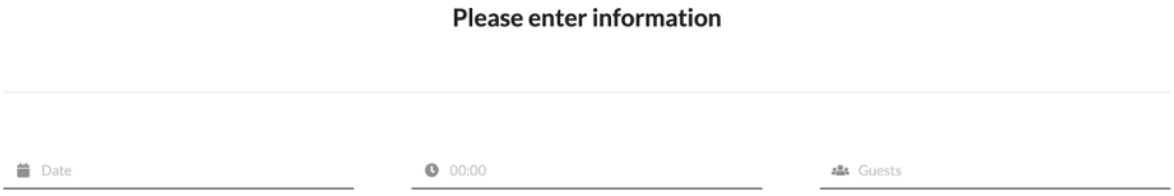


Рисунок 4.6 Вибір дати, години, кількості відвідувачів

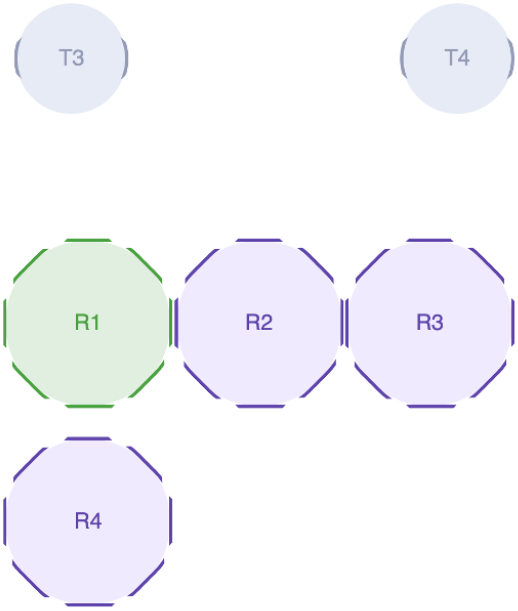


Рисунок 4.7 Вибір столика ресторану

Наступним кроком являється вибір страв. Користувач бачить вже раніше замовлені ним страви, оскільки жодної страви не було замовлено, слід натиснути кнопку “Add dishes!", після чого він буде пере направлений на сторінку страв. На цій сторінці, користувач обирає страви або видаляє їх. Також для зручності пошуку необхідних страв реалізовано їх фільтрування за абеткою та ціною, роздрібненість по категоріям. Щоб завершити процес вибору страв, слід натиснути іконку “Корзина” в лівому верхньому куті.

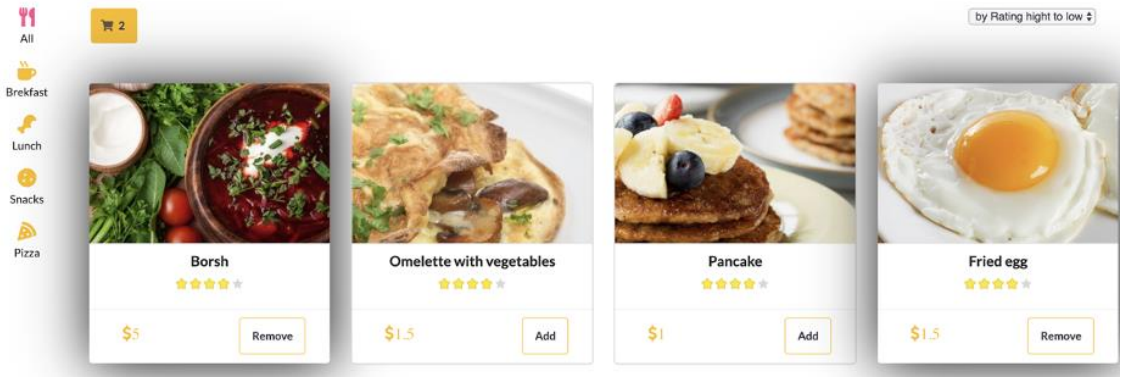


Рисунок 4.8 Сторінка страв

Повернувшись до замовлення на крок з меню, користувач бачить замовлені щойно ним страви, які він може редагувати, змінюючи їхню кількість та видалити.

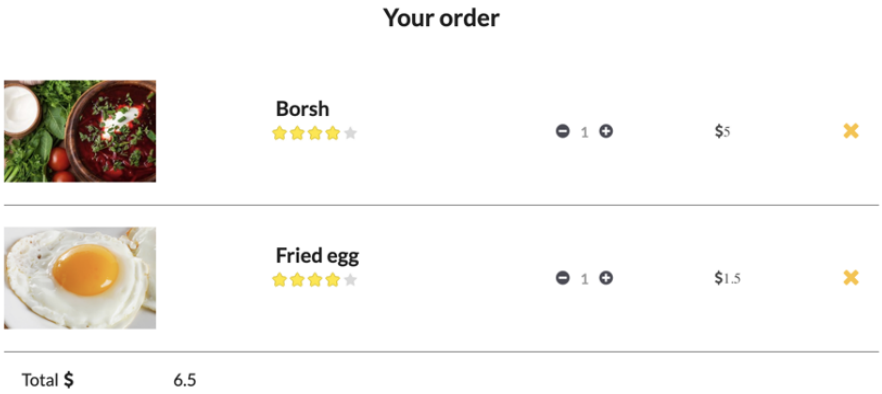


Рисунок 4.9 Крок замовлення страв

Наступний крок для клієнта - чек з усією інформацією про замовлення (час, дата, замовленні страви, ціна) та оплата.

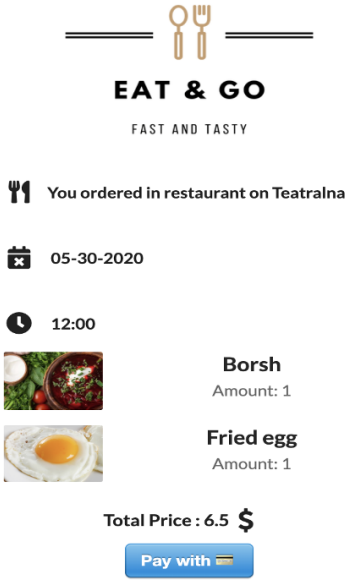


Рисунок 4.10 Чек замовлення

Щоб підтвердити замовлення, потрібно оплатити його, натиснувши “Pay with”. Для здійснення оплати з'являється нове модальне вікно, в якому заповнюється дані карти та здійснюється оплата за допомогою кнопки “Pay”.

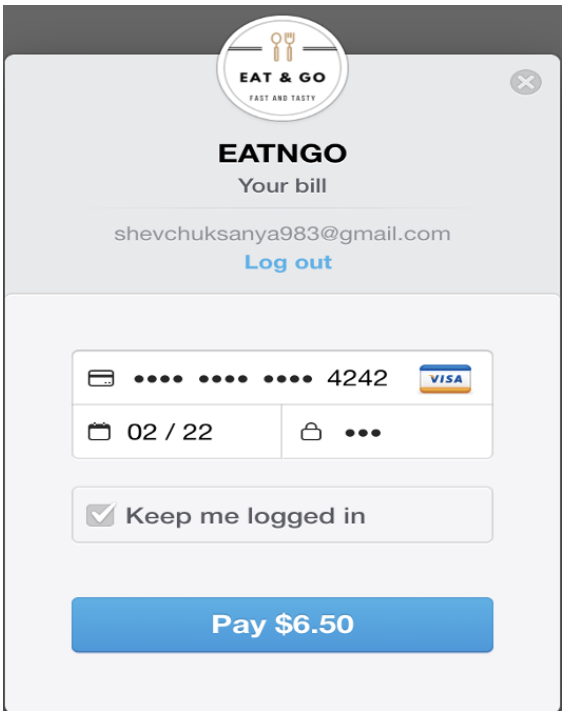


Рисунок 4.11 Форма оплати

Дане замовлення було виконано не зареєстрованим користувачем, що позбавляє його додаткових можливостей сервісу. Зареєстрований користувач має можливість створення власного кабінету з інформацією про себе та спеціальні знижки, які пропонують заклади. Для реєстрації та авторизації слід натиснути “Log-in” у шапці сайту, що перенаправить на Log-in сторінку.

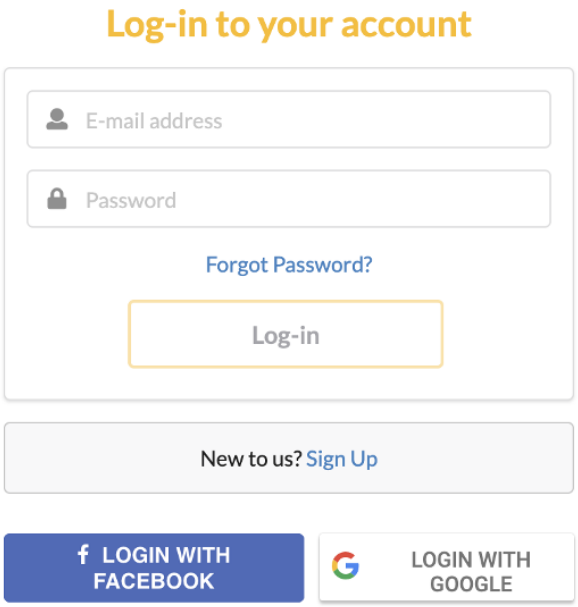


Рисунок 4.12 Log-in форма

Оскільки користувач не має зареєстрованого аккаунта, він може авторизуватись через відому соціальну мережу “Facebook”, або “Google” пошту, або створити новий аккаунт натиснувши кнопку “Sign Up”. Зареєструємо нового користувача.

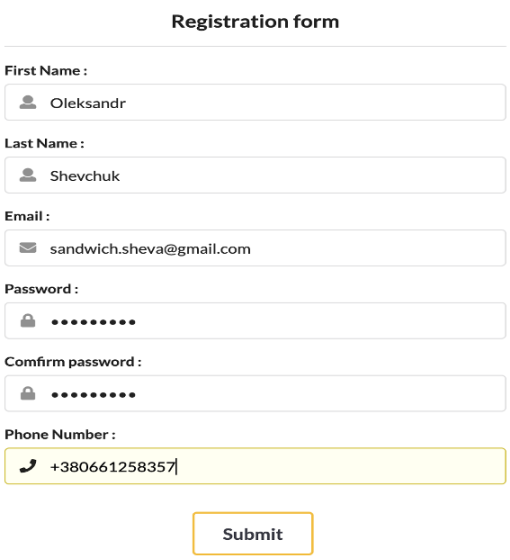


Рисунок 4.13 Sign-up форма

Після успішної реєстрації, створюється особистий кабінет користувача, який можна знайти у шапці веб-сайту.

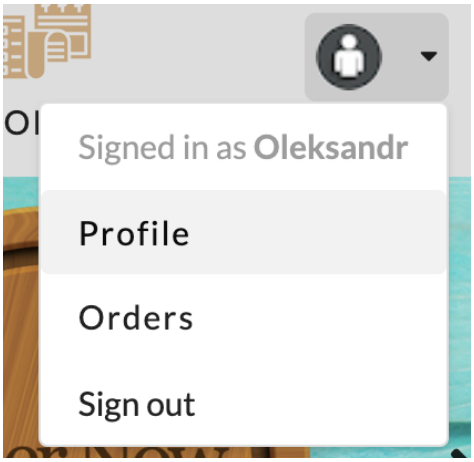


Рисунок 4.14 Юзер дропдаун

Особистий кабінет користувача виглядає наступним чином.

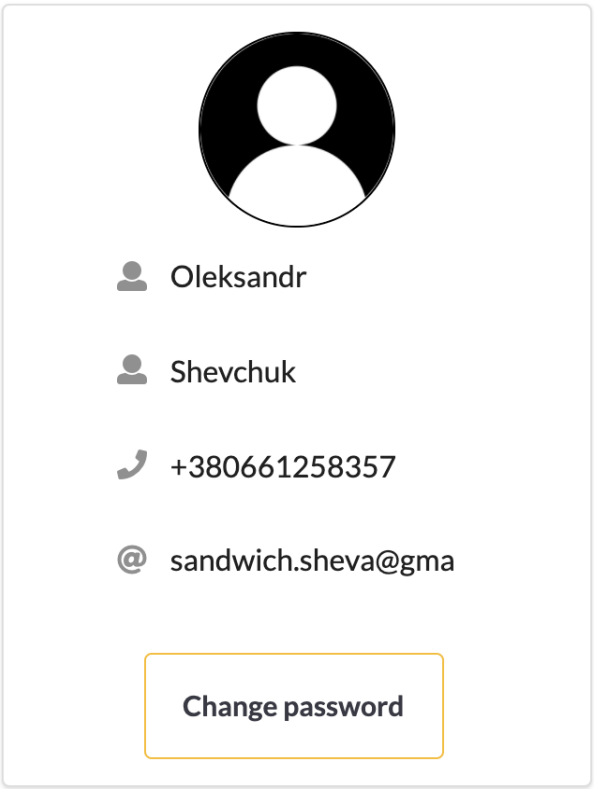


Рисунок 4.15 Особистий кабінет користувача

В особистому кабінеті користувач бачить свою коротку інформацію (ім'я, прізвище, номер телефону та пошта), яку він може змінити разом із паролем.

4.2 Інтерфейс адміністратора

Щоб відкрити веб-додаток для адміністратора, необхідно перейти на адресу у браузері, де розміщений адмін-клієнтська частина. Переходячи за адресою, користувач бачить Log-in форму.

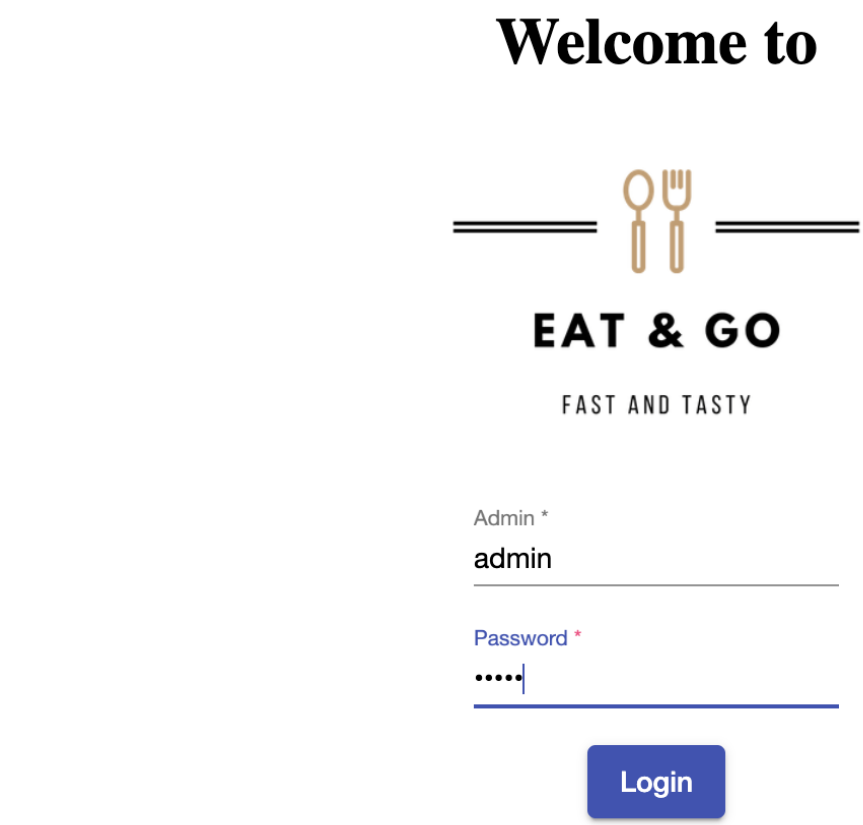


Рисунок 4.16 Admin Log-in форма

Після коректного ведення логіна та пароля адміністратора, користувача перенаправляє на головну сторінку, в якій є лише шапка веб-сайту.

Рисунок 4.16 Admin шапка веб-сайту

Розглянемо редагування меню страв. Адміністратор має можливість додавати, видаляти та редагувати страви.

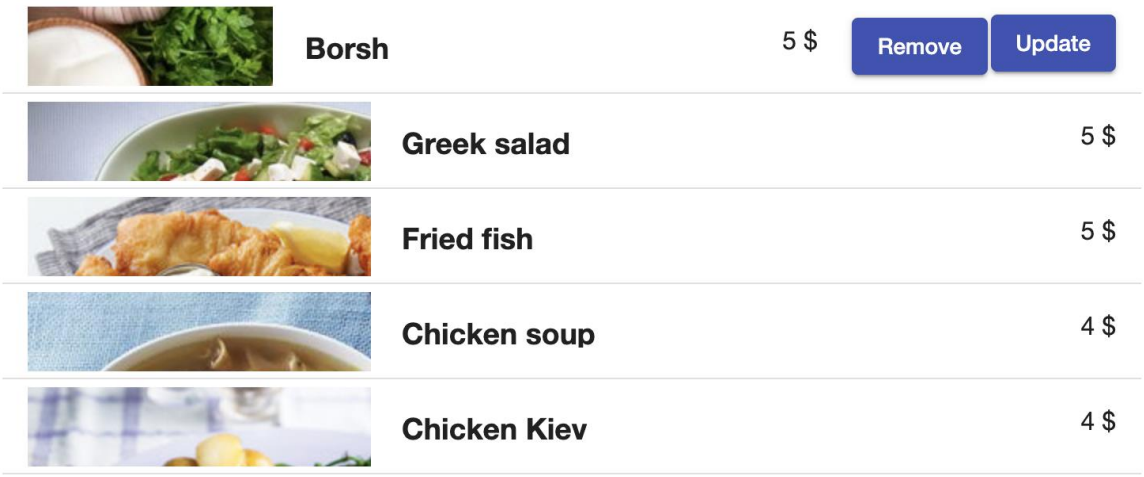


Рисунок 4.17 Список страв

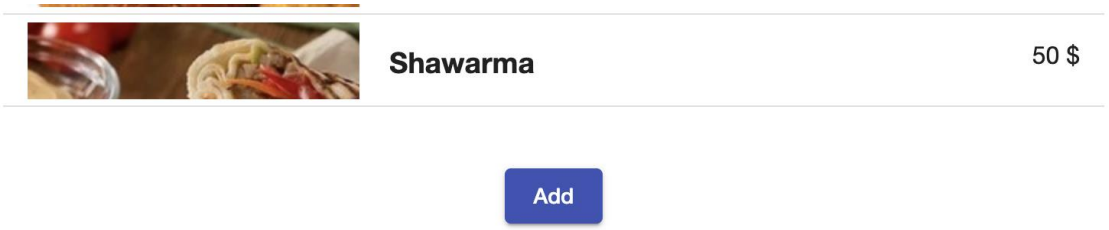


Рисунок 4.18 Кнопка для додавання страв

При додаванні або редагуванні страви, адміністратор повинен заповнити інформацію про страву.

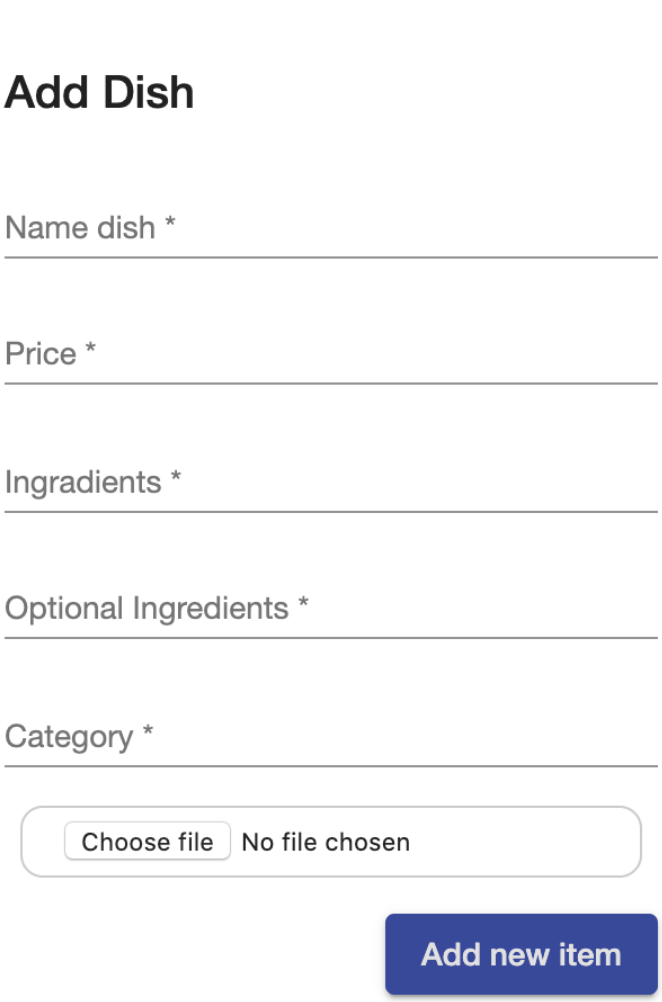


Рисунок 4.19 Форма додавання страв

Такі ж самі операції адміністратор може виконувати над спеціальними акціями ресторанів.

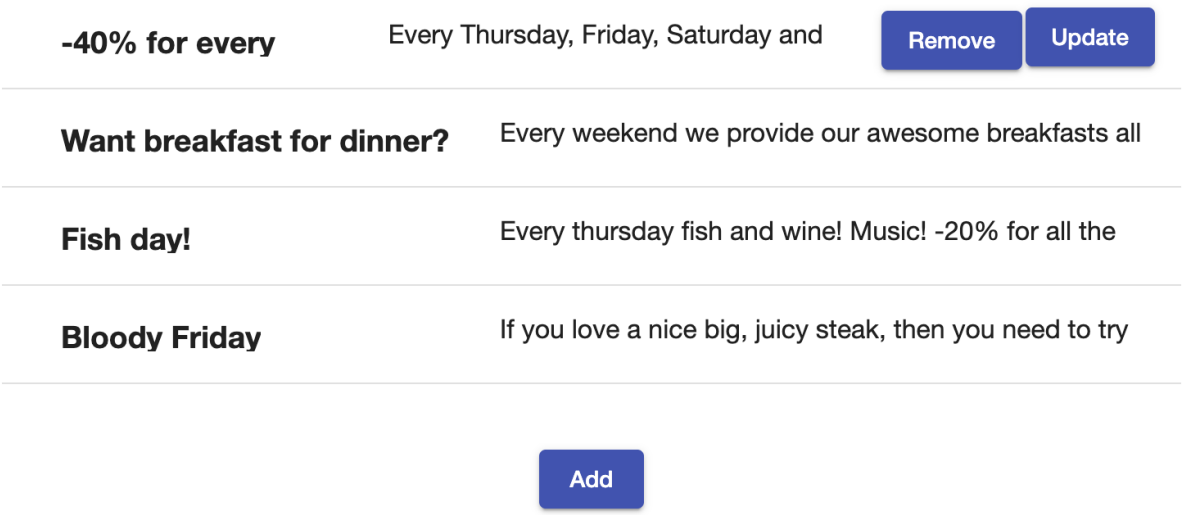


Рисунок 4.20 Список спеціальних акцій

ВИСНОВКИ

В магістерській роботі було досліджено та проаналізовано рентабельність мережі ресторанів «Черноморка»:

- розглянуто та досліджено поняття рентабельності, її види та шляхи підвищення
- розглянуто структури мережі ресторанів “Черноморка”
- обрані методи підвищення рентабельності роботи закладу “Черноморка”
- виконана розробка сервісу, який підвищує рентабельність ресторану.

Створено веб-додаток відповідно до сучасних стандартів зі зручним, конструктивним та зрозумілим дизайном, який задовольняє потреби мережі ресторанів “Черноморка” та може бути використаний в будь-якій іншій мережі ресторанів.

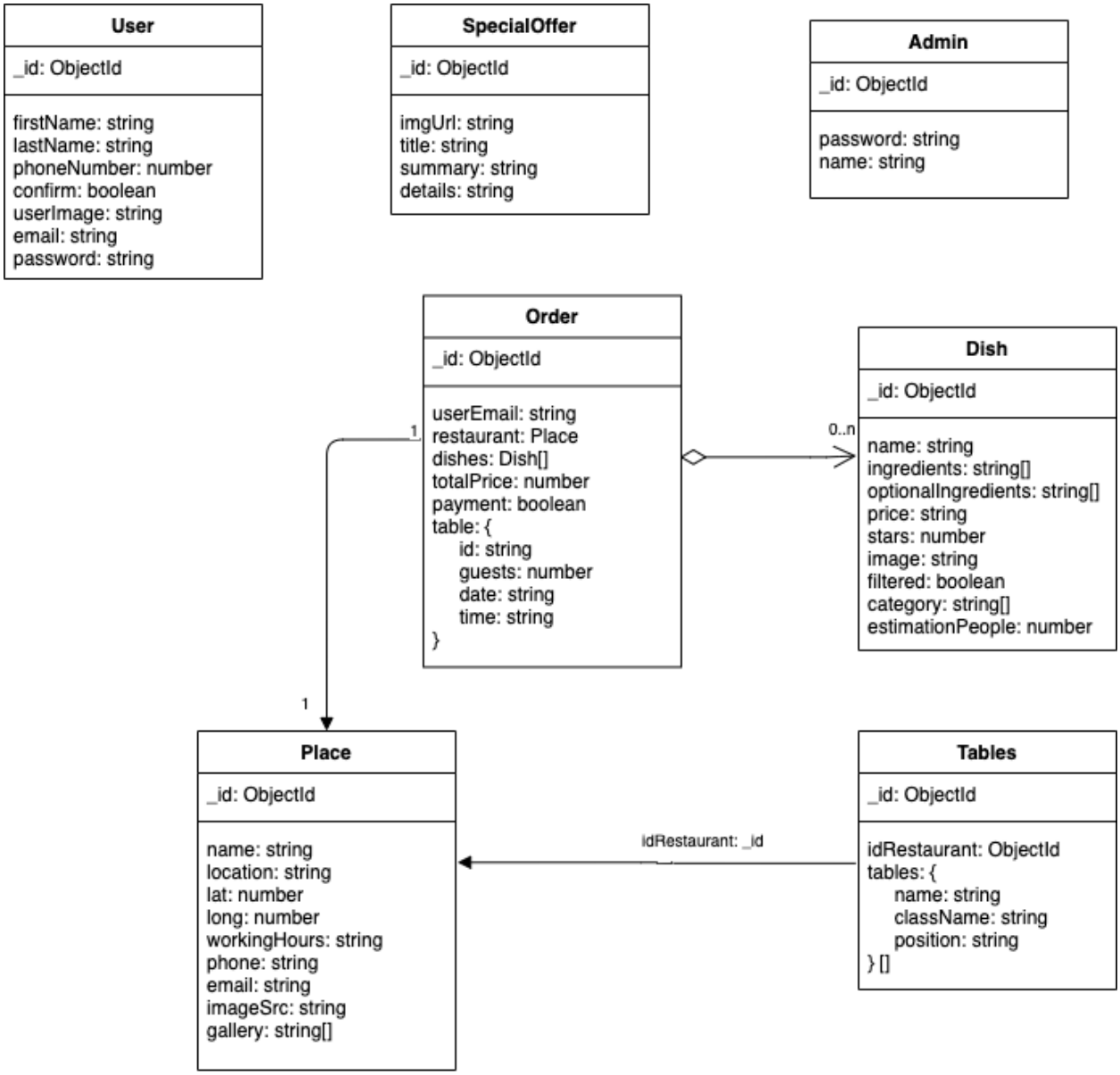
Результати проведені в магістерській роботі можуть бути успішно використані в мережі ресторанів, до якої входить ресторан «Черноморка».

Список використаної літератури

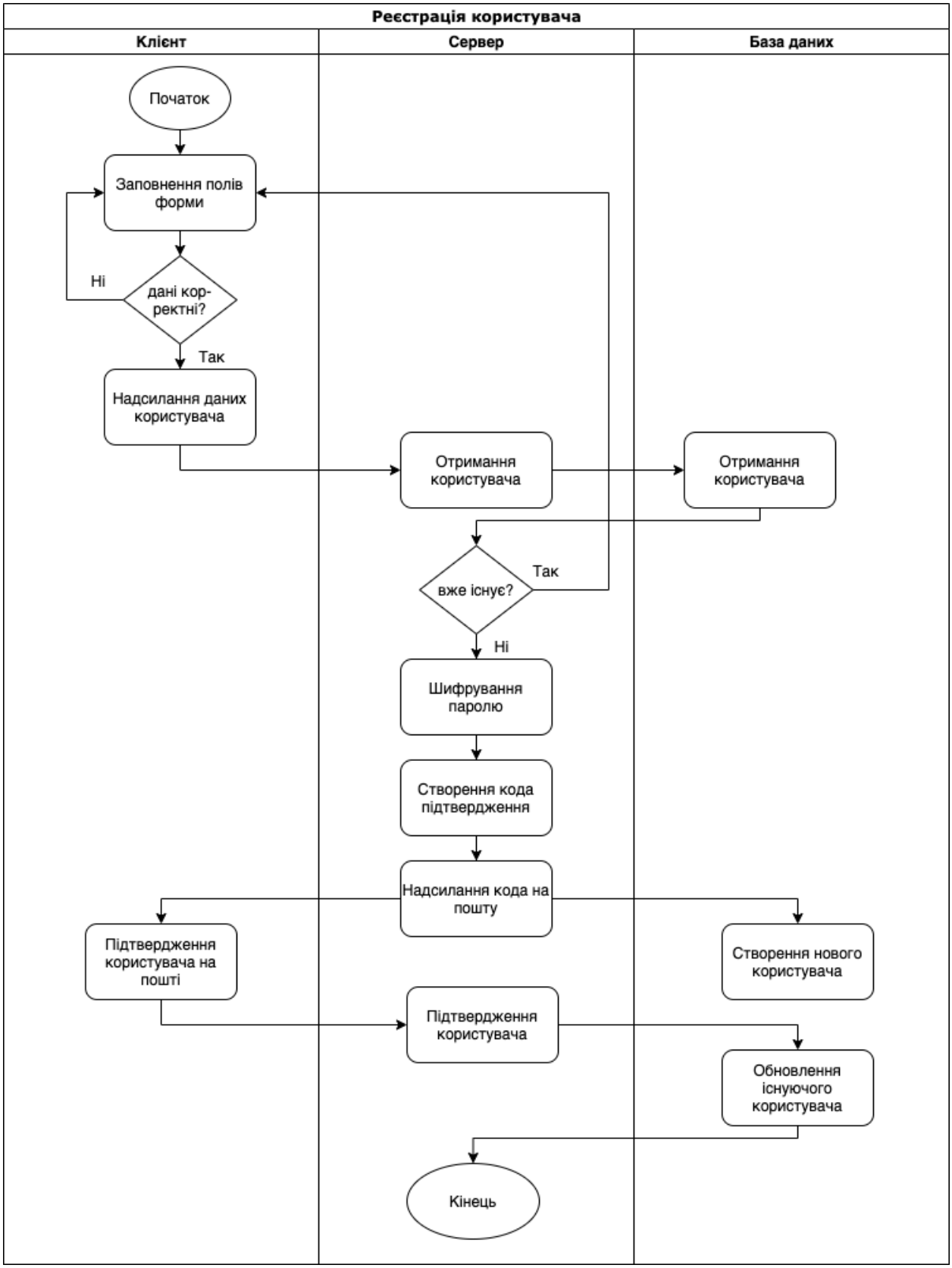
1) 100+ INTERNET STATISTICS AND FACTS FOR 2020 https://www.websitehostingrating.com/internet-statistics-facts/
2) Сучасний підручник Javascript https://learn.javascript.ru/
3) Understanding ECMAScript 6: The Definitive Guide for JavaScript Developers / Nicholas C. Zakas, 2016
4) Javascript | MDN https://developer.mozilla.org/ru/docs/Web/JavaScript
5) Javascript tutorial - W3Schools https://www.w3schools.com/js/
6) Express документація https://expressjs.com/ru/
7) Онлайн посібник з MongoDB https://metanit.com/nosql/mongodb/
8) React.js документація https://ru.reactjs.org/
9) Angular.js документація https://angular.io/
10) Архітектурний підхід проектування веб-додатків Rest API https://dataart.ru/news/podhody-k-proektirovaniyu-restful-api/
11) Менеджер пакунків Yarn https://classic.yarnpkg.com/en/docs/
12) Стаття про SPA https://wezom.com.ua/blog/chto-takoe-spa-prilozheniya
13) Отношения классов - от UML к коду https://habr.com/ru/post/15004

ДОДАТКИ

ДОДАТОК А СХЕМА МОДЕЛІ ДАНИХ



ДОДАТОК Б СХЕМА АЛГОРИТМУ РЕЄСТРАЦІЇ



ДОДАТОК В АЛГОРИТМ ВСТАНОВЛЕННЯ НОВОГО ПАРОЛЯ