НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки Кафедра обчислювальної техніки

«На правах рукопису»
УДК_____

До захисту допущено: Завідувач кафедри
_____Сергій СТІРЕНКО
«_ __»_____2023 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Комп'ютерні системи та мережі» зі спеціальності 123 «Комп'ютерна інженерія»

на тему: «Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу»

Виконав:
студент ІІ курсу, групи ІО-21мп
Борозенець Денис Романович _____

Керівник:
доцент каф. ОТ, к.т.н.
Ткаченко Валентина Василівна _____

Консультант з нормконтролю:
проф. каф. ОТ д.т.н.
Кулаков Юрій Олексійович _____

Рецензент:
проф. каф. ІПІ, д.т.н, професор,
Стеценко Інна В’ячеславівна
_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань. Студент (-ка)_____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет *Інформатики та обчислювальної техніки*
(повна назва)

Кафедра *Обчислювальної техніки*
(повна назва)

Рівень вищої освіти – другий (магістерський)
Спеціальність *123. Комп'ютерна інженерія*
(код і назва)

Освітньо-професійна програма *Комп'ютерні системи та мережі*
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій СТІРЕНКО
(підпис)
«_____» _____ 2023 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Борозенцю Денису Романовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу»,

науковий керівник дисертації доцент каф. ОТ, к.т.н. Ткаченко В.В.,

затверджені наказом по університету від «07» листопада 2023 р. № 5168-с

2. Строк подання студентом дисертації _____

3. Об'єкт дослідження процес проєктування клієнтської частини та підтримки платформ-агрегаторів для ресторанного бізнесу.

4. Предмет дослідження засоби, інструменти, технології та концепції для реалізації гнучких платформ-агрегаторів для ресторанного бізнесу.

5. Перелік завдань, які потрібно розробити: проаналізувати існуючі інструменти та технології розробки клієнтського інтерфейсу, розробити стратегію для оптимізації й підвищення продуктивності клієнтської складової платформи-агрегатора в ресторанному бізнесі, створити основні елементи функціональності, проаналізувати результати тестування, сформулювати рекомендації щодо подальшого використання та розвитку системи

6. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1.	доцент каф. ОТ, к.т.н. Ткаченко В.В.		
2.	доцент каф. ОТ, к.т.н. Ткаченко В.В.		
3.	доцент каф. ОТ, к.т.н. Ткаченко В.В.		
4.	доцент каф. ОТ, к.т.н. Ткаченко В.В.		
5.	доцент каф. ОТ, к.т.н. Ткаченко В.В.		

7. Дата видачі завдання 01.09.2023

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Затвердження теми роботи	1.09.23	
2	Вивчення та аналіз завдання	1.09.23-7.09.23	
3	Дослідження існуючих рішень	8.09.23-21.09.23	
4	Адаптація обраних алгоритмів до розроблюваної системи	22.09.23-10.10.23	
5	Розробка архітектури та загальної структури програми	11.10.23-20.10.23	
7	Програмна реалізація	20.10.23-20.11.23	
8	Тестування рішення та аналіз ефективності.	21.11.23-05.11.23	
9	Оформлення пояснювальної записки	06.11.23-20.12.2023	
10	Захист	18.01.2024	

Студент ___Денис БОРОЗЕНЕЦЬ_____ _____
(підпис)

Науковий керівник дисертації ___Валентина ТКАЧЕНКО____ _____
(підпис)

РЕФЕРАТ

на магістерську дисертацію

виконану на тему: Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу

студентом: Борозенцем Денисом Романовичем

Робота складається із вступу та чотирьох розділів. Загальний обсяг роботи: 92 аркуша основного тексту, 26 ілюстрацій, 12 таблиць. При підготовці використовувалася література з 38 джерел.

Актуальність теми. Актуальність теми дисертації визначається тим що у сучасному світі все більше людей планують свій відпочинок та розваги. Зарання обирають місце перебування, а також місця, в яких хочуть побувати для отримання нових вражень або щоб спробувати щось незвичне і нове для себе. З появою Всесвітньої мережі Інтернет стало значно легше отримати доступ до інформації про різноманітні заклади, які пропонують велику кількість послуг для відпочинку.

Однак, на сьогоднішній день, не існує єдиного комфортного джерела, яке б містило всю інформацію про різні заклади і допомагало користувачам, в залежності від їх потреб, знайти найкращий варіант для своєї відпустки. У зв'язку з цим, створення застосунку, що допомагає обрати кращий заклад для відпочинку і надає повну інформацію про нього, має велику актуальність.

Мета і завдання дослідження. Метою даного дослідження є розробка концепцій, що забезпечуватимуть гнучкість при масштабуванні користувацького програмного забезпечення для платформ-агрегаторів для ресторанного бізнесу.

Ціллю дослідження магістерської дисертації є розробка клієнтської частини для платформи-агрегатора, що забезпечує актуальні вимоги щодо високої продуктивності та спрощення масштабування клієнтського програмного забезпечення.

Для досягнення поставленої мети магістерської дисертації були сформовані наступні завдання дослідження:

1. Проаналізувати відомі платформи-агрегатори, а також відомі інструменти та технології для реалізації легко-масштабованих та високо-продуктивних платформ-агрегаторів для ресторанного бізнесу з метою визначення проблематики і постановки завдань дослідження.

2. Розробити концепцію архітектури та функціонування клієнтської частини платформи агрегатора для ресторанного бізнесу для реалізації високопродуктивної та масштабованої платформи агрегатора.

3. Розробити архітектуру клієнтського програмного забезпечення для реалізації високопродуктивної та масштабованої платформи-агрегатора.

4. На основі розробленої архітектури розбити користувацький інтерфейс платформи-агрегатора для ресторанного бізнесу.

Об’єкт дослідження. Об’єктом дослідження є процеси проєктування та розробки клієнтського програмного забезпечення платформ-агрегаторів для ресторанного бізнесу, що обумовлюють проблему підвищення продуктивності функціонування та забезпечення гнучкості масштабування програмного забезпечення.

Предмет дослідження. Архітектурна та функціональна концепція та програмні засоби для реалізації високопродуктивної та масштабованої платформи-агрегатора для ресторанного бізнесу.

Методи дослідження. У магістерській дисертації використано методи синтезу, аналізу, моделювання, тестування. Це дозволяє детально проаналізувати задачу, що поставлена і сформувати коректну концепцію розробки.

Наукова новизна одержаних результатів. Удосконалено архітектурну та функціональну концепцію реалізації клієнтського програмного забезпечення платформи-агрегатора для ресторанного бізнесу, за рахунок використання компонентного підходу для формування односторінкового користувацького інтерфейсу, що дозволило підвищити продуктивність, за

рахунок зменшення загальної об’єктної моделі документа та спростити масштабованість програмного забезпечення клієнтської частини платформи-агрегатора.

Практичне значення одержаних результатів. Практичними результатами магістерської дисертації є розроблене та реалізована клієнтське програмне забезпечення платформи-агрегатора для ресторанного бізнесу. Розроблене програмне забезпечення може бути використано для реалізації високопродуктивних та масштабованих застосунків для керування ресторанним бізнесом. Платформа-агрегатор може бути застосована, як потенціальний інструмент для кардинального підвищення кількості відвідувачів будь-якого закладу, на основі різноманітних переваг, що надає розроблена платформа-агрегатор.

Особистий внесок магістранта. Магістерська дисертація є науковою роботою, що виконана самостійно. В даній роботі відображено авторський підхід та власне отримані теоретичні і практичні результати, пов'язані з вирішенням задач побудови архітектури інформаційної системи. У магістерській роботі використовувалися сучасні методи для досягнення поставлених цілей, що безпосередньо призвело до отримання практичних результатів. Мета і завдання дослідження були сформульовані спільно з науковим керівником.

Ключові слова: платформа-агрегатор, клієнтське програмне забезпечення, Figma, JavaScript, React, ресторанний бізнес.

ABSTRACT

for the master's thesis

completed on the topic: Client-Side
Software Tailored for the Restaurant Business Platform Aggregator

by student: Denys Romanovych Borozenets

The work consists of an introduction and four chapters. The total volume of the work is 92 pages of main text, 26 illustrations, 12 tables. Literature from 38 sources was used in the preparation.

Relevance of the topic. The relevance of the dissertation topic is determined by the fact that in the modern world, an increasing number of people plan their leisure and entertainment in advance. They carefully choose their place of stay, as well as places they want to visit to gain new experiences or try something unusual and new for themselves. With the advent of the World Wide Web, it has become much easier to access information about various establishments that offer a wide range of leisure services.

However, to date, there is no single convenient source that contains all the information about different establishments and helps users find the best option for their vacation, depending on their needs. In this regard, the creation of an application that assists in selecting the best leisure venue and provides comprehensive information about it is highly relevant.

Purpose and objectives of the research. The purpose of this research is to develop concepts that will provide flexibility when scaling custom software for aggregator platforms for the restaurant business.

The goal of the master's thesis research is the development of the client part for the aggregator platform, which meets the current requirements for high performance and simplifying the scaling of the client software.

To achieve the set goal of the master's thesis, the following research tasks were formed:

Research objectives:

1. To analyze known aggregator platforms, as well as known tools and technologies for the implementation of easily scalable and highly productive aggregator platforms for the restaurant business in order to determine the issues and set research objectives.

2. Develop the concept of architecture and operation of the client side of the aggregator platform for the restaurant business to implement a high-performance and scalable aggregator platform.

3. Develop a client software architecture to implement a high-performance and scalable aggregator platform.

4. Based on the developed architecture, break down the user interface of the aggregator platform for the restaurant business.

Research object. The object of research is the processes of designing and developing client software of aggregator platforms for the restaurant business, which determine the problem of increasing the productivity of operation and ensuring the flexibility of software scaling.

Subject of research. Architectural and functional concept and software tools for implementing a high-performance and scalable aggregator platform for the restaurant business.

Research methods. In the master's dissertation, methods of synthesis, analysis, modeling, and testing have been employed. This allows for a detailed analysis of the posed task and the formulation of a sound development concept.

Scientific novelty of the obtained results. The architectural and functional concept of the implementation of the client software of the aggregator platform for the restaurant business has been improved, due to the use of a component approach to the formation of a one-page user interface, which made it possible to increase productivity, due to the reduction of the overall object model of the document and simplify the scalability of the software of the client part of the platform- aggregator

Practical significance of the obtained results. The practical results of the master's thesis are the developed and implemented client software of the aggregator platform for the restaurant business. The developed software can be used to

implement high-performance and scalable restaurant business management applications. The aggregator platform can be used as a potential tool for dramatically increasing the number of visitors to any establishment, based on the various advantages provided by the developed aggregator platform.

Personal contribution of the master's student. The master's dissertation is an independent scientific work. This work reflects the author's approach and the theoretical and practical results obtained in solving problems related to the construction of the information system architecture. Modern methods were employed in the master's thesis to achieve the set objectives, leading directly to practical results. The research goals and tasks were formulated collaboratively with the academic supervisor.

Key words: aggregator platform, client software, Figma, JavaScript, React, restaurant business.

Пояснювальна записка
до магістерської дисертації

на тему: «Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу»

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API (Application Programming Interface) – прикладний програмний інтерфейс

HTTP (Hypertext Transfer Protocol) – протокол передачі гіпертексту

NoSQL (Not Only SQL) – тип баз даних, який зберігає інформацію у форматі, відмінному від традиційних таблиць SQL.

GPS (Global Positioning System) – система глобального позиціювання

MVC (Model-View-Controller) – модель-представлення-контролер

SDLC (Software Development Life Cycle) – життєвий цикл програмного забезпечення

SRC (Software Requirement Specification) – специфікація вимог до програмного забезпечення

DDS (Design Document Specification) – специфікація проєктування

SQL (Structured Query Language) – мова структурованих запитів

OS (Operating System) – операційна система

GIF (Graphics Interchange Format) – формат обміну зображеннями

UI (User interface) - дизайн інтерфейсу користувача

HTML (HyperText Markup Language) – стандартизована мова розмітки документів

CSS (Cascading Style Sheets) – спеціальна мова стилю сторінок

XML (EXtensible Markup Language) - розширювана мова розмітки

PWA (Progressive Web Apps) - вебзастосунок, який є гібридом звичайної вебсторінки та мобільного застосунку

DOM (Document Object Model) – об’єктна модель документа

ML (Machine Learning) – машинне навчання

ПЗ (Програмне забезпечення) – software

ПП (Програмний продукт) – software product

ВСТУП

В сучасному світі, де технологічний прогрес дуже швидко змінюється, ефективність та зручність клієнтських інтерфейсів систем є вирішальними факторами успіху в будь-якій галузі, не виключенням є і галузь обслуговування громадського харчування. З впровадженням новітніх технологій, таких як штучний інтелект, інтернет речей та блокчейн, на ринку з'явилися різноманітні можливості для вдосконалення та розширення функціональності цих систем. Дана магістерська робота присвячена дослідженню та порівнянню різноманітних технологій, що використовуються в розробці клієнтського інтерфейсу веб-додатків. Метою дослідження є з'ясування переваг та недоліків кожної з технологій з метою вибору оптимального рішення для забезпечення найвищого рівня зручності та задоволення потреб клієнтів.

У першому розділі було описано доцільність та актуальність розробки платформи-агрегатора для ресторанного бізнесу. Було проаналізовано системи, що на теперішній час існують і успішно функціонують, наведено переваги та недоліки кожної з них.

У другому розділі увага була зосереджена на описі необхідних технологій, що потрібні для формування архітектури системи, її дизайну і безпосередньо розробки клієнтського інтерфейсу. Було проаналізовано різні інструменти для розробки, мови програмування тощо.

У третьому розділі основну увагу було призначено для планування, проєктування та реалізації розробленої системи платформи-агрегатора для ресторанного бізнесу. Було сформовано необхідний перелік вимог, розроблено архітектуру платформи-агрегатора та реалізовано основні компоненти.

У четвертому розділі висвітлено розробку стартап-проекту. Розробка стартап-проєкту ґрунтується на результатах дослідження попередніх розділів.

Основна увага даного дослідження першочергово приділена детальному аналізу сучасних інструментів та технологій, які активно використовуються для розробки користувацького інтерфейсу, з метою обрання найбільш швидких, легких та ефективних рішень.

Метою даної магістерської дисертації є не лише вирішення теоретичної частини проблеми, а в той же час виконати і практичну реалізацію платформи-агретгатора, що в подальшому може знайти використання у ресторанному бізнесі.

Результатом дослідження є розроблена платформа-агрегатор для ресторанного бізнесу, яка акумулюватиме велику кількість закладів разом з детальною інформацією та описом, що спрятиме кращій комунікації закладу і відвідувачів.

РОЗДІЛ 1
АНАЛІЗ ТА ОГЛЯД СУЧАСНИХ РІШЕНЬ ТА РЕАЛІЗАЦІЇ СИСТЕМ ТИПУ АГРЕГАТОР РЕСТОРАНІВ.

1.1 Актуальність та доцільність розробки платформи-агрегатора

Наш час з повною впевненістю можна вважати епохою розквіту інформаційних систем та технологій. Дана сфера стала важливим аспектом нашого життя. Особливо це стало помітним в останні роки. Щодня з’являється все більше засобів та можливостей задовольнити власні потреби або вирішити різні питання не виходячи з власної зони комфорту. Хоч зараз і спостерігається тенденція великої кількості компаній на повернення робітників до офісів, можна впевнено сказати, що так як було колись, вже не буде, адже люди зрозуміли переваги віддаленої можливості роботи та більш гнучкого робочого дня. Онлайн-оплата різноманітних товарів, купівля речей побуту, електроніки, одягу, можливість замовити продукти або вже готову страву виявились на диво зручними.

У сучасному світі все більше людей планують свій відпочинок та розваги. Зарання обирають місце перебування, а також місця, в яких хочуть побувати для отримання нових вражень або щоб спробувати щось незвичне і нове для себе. З появою Всесвітньої мережі Інтернет стало значно легше отримати доступ до інформації про різноманітні заклади, які пропонують велику кількість послуг для відпочинку. Однак, на сьогоднішній день, не існує єдиного комфортного джерела, яке б містило всю інформацію про різні заклади і допомагало користувачам, в залежності від їх потреб, знайти найкращий варіант для своєї відпустки. У зв'язку з цим, створення мобільного додатку, що допомагає обрати кращий заклад для відпочинку і надає повну інформацію про нього, має велику актуальність.

На підставі вищесказаного обґрунтування доцільності розробки платформи-агрегатора складається в наступному:

– Постійний попит

Їжа є невід'ємною частиною людського життя. Разом з цим, гастрономічна індустрія є, напевне, однією з найбільш стабільних та постійних галузей сучасного життя. Завжди буде існувати попит на кафе, ресторани тощо. Комфортний засіб для пошуку та вибору їжі буде завжди актуальним для користувачів.

- Конкуренція

  У великих містах та розвинених регіонах завжди наявна конкуренція серед закладів. За рахунок додатку, кафе та ресторани матимуть можливість залучити нових клієнтів та відвідувачів.

- Мінливість вподобань користувачів

  Смаки та вподобання користувачів завжди змінюються. Саме тому додаток може надати рекомендації в обранні відповідного ресторану, що задовольняють теперішні смаки або ж обмеження зв'язані з дієтичним харчуванням.

- Висока вірогідність масштабування

  У разі успішної концепції побудови додатку, його можна буде масштабувати. Тобто спочатку можна орієнтуватись на невеличке місто або регіон, і потім по трохи збільшувати масштаб.

- Розвиток гастрономічного туризму

  Гастрономічний туризм є доволі популярним відпочинком серед туристів. Даний додаток може сприяти розвитку даного виду туризму, завдяки привертанню нових туристів, і разом з цим, підвищує популярність регіону.

- Дослідження та аналітика

  Додаток матиме змогу надавати різноманітні дані про відвідування ресторанів та кафе, популярність певних кухонь або страв. Подібна інформація може бути корисною для різного роду бізнесів і досліджень в даній галузі.

Далі визначимо актуальні причини для розробки платформи-агрегатора, які складаються в наступному:

- Доступність та зручність інформації

  Створення додатку для обрання закладу відпочинку ставить на меті реалізувати процес пошуку інформації доволі легким та зручним для користувача. Подібний додаток має містити базу даних з великою кількістю закладів, що буде мати різноманітну інформацію про розташування, послуги, рейтинги та ціни. Завдяки цьому можна буде легко знайти потрібну інформацію та прийняти зважене рішення щодо місця відпочинку.

- Велика і постійно зростаюча популярність гастрономічних вражень

  З кожним днем все більше людей цікавляться різноманітною їжею, різними кухнями народів тощо. І відповідно до цього ресторани з різноманітною кухнею і великою кількістю страв мають більшу популярність. Розробка додатку, що надає користувачам можливість знайти ресторан за їх гастрономічними вподобаннями, може здійснити попит.

- Економія часу

  З допомогою мобільного додатку не потрібно витрачати багато часу на пошук великої кількості різноманітних ресурсів для збору інформації про різні місця відпочинку. Додаток надасть зручну систему фільтрування для швидкого пошуку необхідного закладу, що в свою чергу дозволить оперативно знайти відповідне місце відпочинку в залежності від уподобань та вимог.

- Інтерактивність

  Завдяки можливості перегляду місця розташування ресторанів на мапі, можна швидко зорієнтуватись та обрати найбільш швидкий або комфортний варіант

- Мобільність

  Користувачі відчувають необхідність у комфортному інструменті для пошуку місця відпочинку або ресторану. Великою мірою ця необхідність проявляється під час подорожі, відрядження або просто

в новому місті. Додаток, що функціонує на мобільному пристрої, може надати легкий доступ до інформації.

– Надійність

  Додаток може стати доволі надійним джерелом інформації для користувачів. База даних, що має в наявності різні заклади для відпочинку, може постійно оновлюватись та перевірятись, для забезпечення актуальності інформації. Окрім цього, користувачі матимуть змогу залишати свої відгуки про заклади, що в свою чергу допоможе іншим користувачам мати об'єктивну інформацію та оцінку про якість послуг.

– Розвиток бізнесу

  Додаток може бути гарною можливістю для ресторанів, для того щоб просувати свої страви та послуги, за рахунок даної платформи. Це в свою чергу посприяє залученню нових клієнтів, що надалі збільшуватиме клієнтську базу а разом з цим і прибуток.

1.2 Огляд відомих реалізацій систем агрегаторів для ресторанів

Існування систем агрегаторів ресторанів та кафе стає надзвичайно важливим аспектом в галузі гастрономії на сьогоднішній день. Подібні системи мають значний вплив, як на зручність обрання ресторану або кафе відповідно власних побажань, так і на можливість швидкого отримання страв. Окрім цього, подібні системи можуть оптимізувати використання різних ресурсів, пов’язаних з гастрономією. За рахунок подібної системи кожен заклад має змогу рекламувати власні страви та послуги, що в свою чергу призведе до збільшення кількості клієнтів, а разом з цим і до збільшення прибутку і розвитку самого закладу.

Саме тому розробка та реалізація ефективної та комфортної системи пошуку ресторанів та кафе є доволі важливим інструментом для задоволення потреб користувачів, а також оптимізації гастрономічного бізнесу в даній галузі.

Наведемо аналіз основних функціональних особливостей відомих систем агрегаторів.

Система Yelp

Yelp Inc. являється американською компанією, що займається розробкою веб-сайту Yelp.com та мобільним додатком Yelp. Дані сервіси створені для публікації різноманітних відгуків, що зібрані від великої кількості користувачів про бізнеси. Окрім цього, дана компанія керує службою бронювання столиків в ресторанах та кафе під назвою Yelp Guest Manager. Головний офіс компанії розташовується в місті Сан-Франциско, штат Каліфорнія. Дата заснування компанії датується 2004 роком. Її заснували колишні співробітники компанії PayPal Джеремі Стопельман та Рассел Сімонс.

В роботі [1] представлені наступні характерні особливості.

Yelp це платформа, створена для відгуків, яка має списки місцевих закладів. Ресторани, кафе, бари і тд. мають особисті сторінки, на яких будь який користувач здатен прочитати про цей заклад, дізнатися про перелік послуг і графік роботи, а також залишити відгук. Цікавою особливістю даної платформи є можливість завантажувати відео та зображення, які пов’язані з обраним закладом.

На підставі огляду характерних особливостей системи Yelp, можна визначити наступні функціональні та нефункціональні вимоги [2].

Функціональні вимоги:

- Власник бізнесу або ж клієнт в обов’язковій формі повинні мати змогу безперешкодно створювати сторінки для власного бізнесу.
- Користувачі платформою повинні мати змогу знаходити визначені місця або об’єкти з допомогою розташування або координат GPS(Global Positioning System – система глобального позиціювання) і разом з цим знаходити найближчі місця та заклади відносно їх локації.

- Обов’язково користувачі повинні мати змогу залишати оцінку та відгуки про заклади, в яких вони побували і послуги, якими їм вдалося скористатись.
- Користувачі повинні мати змогу додавати відео та фото, які пов’язані з місцем відпочинку.

Нефункціональні вимоги:

- Платформа повинна бути легкодоступна для користувачів.
- Легка масштабованість для більшого обсягу регіонів.
- Користувачі повинні мати змогу отримувати будь яку інформацію якнайшвидше

Платформа Yelp має наступні переваги [3]:

- Достатньо велика база даних про ресторани, кафе та загалом гастрономічні заклади.
- Має можливість пошуку відносно категорій та швидку фільтрацію.
- Відгуки даної платформи зазвичай є надійними і точними.
- Створення і керування власною сторінкою є безкоштовним.

Недоліки:

- Існує залежність від рейтингу та відгуків користувачів, які все ж таки можуть бути суб’єктивними. За статистикою 16% відгуків є фальшивими.
- Реклама закладів має змогу спотворювати результати пошуку.
- Для того щоб користуватись всіма функціями, які надає платформа, потрібно витрачати кошти.

Система OpenTable

OpenTable – це онлайн платформа, що створена для бронювання столиків у кафе, ресторанах, барах та інших гастрономічних місцях. Основною метою створення даної платформи було облегшення процесу бронювання

столиків. Окрім цього дана система орієнтована ще й на покращення взаємодії між клієнтом та рестораном.

OpenTable була створена в 1998 році в місті Сан-Франциско, штат Каліфорнія. Першочергово система почала функціонувати в Сан-Франциско, а згодом поширила свою діяльність на весь світ [4].

На підставі огляду характерних особливостей системи OpenTable, можна визначити наступні особливості та можливості даної системи [5].

Основні особливості та можливості даної системи:

- Бронювання столиків

  Користувачі мають змогу легко знаходити різноманітні заклади за місцезнаходженням і відповідно до дати і часу резервувати бажаний столик в обраному закладі.

- Можливість ставити оцінку

  Користувачі мають змогу залишити відгук після відвідування закладу і разом з цим ставити оцінку. Це в свою чергу впливає на вибір інших користувачів, які бачитимуть оцінки.

- Інтерфейс

  Доволі зручний інтерфейс, що спрощує користувачам процедуру знаходження та бронювання столика.

- Мобільність

  Додатки, орієнтовані на 2 системи Android та IOS, а також є веб-версія, що в свою чергу дозволяє здійснити бронювання з будь-якого девайса.

- Рекомендації та фільтри

  Дана система також має змогу надавати рекомендації щодо обрання ресторанів та меню, разом з цим кожен користувач може відфільтрувати ресторани за місцем, кухнею, ціною тощо.

Система має наступні переваги [6, 7]:

- Надає користувачам абсолютно безкоштовну систему для резервації онлайн, тобто в реальному часі.
- Приймає участь і допомагає закладам легко заповнювати столики, залучати нових клієнтів, та перетворювати їх в постійних.
- Допомагає закладам відслідковувати вподобання відвідувачів, для покращення меню або обслуговування.
- Система допомагає зекономити час протягом обрання закладу і резервування столика в даному закладі.

Недоліки [8, 9]:

- Дана система має можливість купівлі підписки, яка надаватиме додаткові функції, але разом з цим може бути достатньо дорогою адже плата проводиться щомісячно.
- Так як не кожен ресторан користується послугами системи OpenTable, користувач не може бачити абсолютно всі заклади, які функціонують, а тільки ті, які є учасниками системи.
- Технології, з допомогою яких реалізована дана система, можуть інколи відмовляти. Це в свою чергу призводить до того, що кілька користувачів можуть зарезервувати один і той самий столик.
- Стягується певна плата від закладу за кожного користувача, який зарезервував столик за допомогою системи.

Система Zomato

Zomato являється індійською мультинаціональною компанією, що займається агрегуванням ресторанів і разом з цим надає послуги з доставки їжі. Заснована компанія у 2008 році, її засновниками є Діпіндер Гойял та Панкандж Чадда. Це загалом комплексна платформа, що надає клієнтам можливість замовляти їжу з різноманітних ресторанів, а також допомагає власникам просувати власний бізнес [10].

На підставі огляду характерних особливостей системи Zomato, можна визначити наступні можливості даної системи [11, 12]:

– Інформація про ресторани

  Zomato надає повну та точну інформацію щодо ресторанів, це і відгуки та рейтинги, і меню, що в свою чергу допомагає клієнтам приймати рішення щодо вибору закладу харчування.

– Система маркетингу

  Дана платформа дозволяє власникам закладів рекламувати власний гастрономічний бізнес, особливо, якщо власник новачок в даній сфері. Окрім цього, власники бізнесу мають змогу розміщувати банери на власному веб-сайті, щоб значно підвищити так звану видимість.

– Система POS

  Zomato Base це доволі проста в використанні хмарна POS-система, що фактично є серверною частиною для закладів.

– Система керування столиками

  Zomato Book це одне з небагатьох рішень, щодо керування столиками та замовлення.

Система має наступні переваги [10, 13]:

– Дана платформа надає своєчасну доставку напряму до клієнта, які не мають можливості сидіти в закладі.
– Надає точну інформацію про заклад, повне меню, відгуки і рейтинги.
– Платформа надає можливість резервування столика онлайн за допомогою веб-сайту або додатку, що є доволі зручною функцією для користувачів.
– Надає дуже великий та різноманітний вибір кухонь для вдоволення різних вподобань та смаків.

Недоліки [10, 14]:

- Так як система доставки все ж таки залежить від партнерів, тобто сторонніх компаній, інколи можуть виникати затримки та сторонні проблеми.
- Платформа стягує плату з закладу за кожного користувача, який зробив замовлення або бронювання столика.
- Інформація щодо закладу інколи може бути недостатньо об’єктивною, адже ґрунтується на відгуках та рейтингах, які можуть бути суб’єктивним.

Система TripAdvisor

TripAdvisor являється американською компанією, що займається функціонуванням різноманітних туристичних агентств, створенням порівнюючих сайтів для шопінгу та додатків під мобільні пристрострої [15].

На підставі огляду характерних особливостей системи TripAdvisor, можна визначити наступні можливості даної системи [15, 16, 17]:

- Агрегує оцінки та відгуки користувачів про місця, помешкання, ресторани, кафе, бари тощо.
- Звертає основну увагу на зміст, що створений безпосередньо користувачами, що вміщують в собі рейтинги, відгуки, а також фото. Саме сукупність всіх цих факторі дозволяє TripAdvisor надавати доволі точну інформацію про напрямки туризму.
- Додатковими можливостями даної платформи є можливість бронювання різноманітних місць відпочинку.
- Надає можливість користувачам просувати свої заклади і послуги завдяки певній кількості рекламних рішень.
- Завдяки наявності мобільного додатку, у клієнтів є можливість мати доступ до додатку та інформації в дорозі. А це в свою чергу дозволяє оперативно бронювати заклади під час подорожі.

Система має наступні переваги [18]:

- Є однією з найпопулярніших платформ для організації подорожі.
- Надає детальну інформацію щодо місць та закладів під час подорожі. В тому числі надається і сам путівник, який допомагає зорієнтуватись на місці.
- Можливість просування закладу за рахунок рекламних рішень.
- Мобільний додаток, що дозволяє користуватись даною платформою під час подорожі.

Недоліки [19, 20]:

- Завжди існує вірогідність неправдивих відгуків, які були написані шахраями суто з особистих мотивів або ж робітниками закладу, які самі себе вихваляють.
- Якщо у власника бізнесу є бажання інтегрувати зміст з платформи TripAdvisor на власний сайт, потрібно укласти угоду щоб отримати доступ до повноцінних, а не обмежених продуктів, яка коштує чималих коштів.

1.3 Необхідний набір функцій для систем агрегаторів ресторанів

При розробці кожної системи або додатку, першочергово складається певний набір функцій, які потрібно реалізувати, щоб задовольнити потреби користувача. Саме тому виокремлення необхідного набору функціональності відіграє значну роль при реалізації системи. Платформа повинна мати наступний функціонал [21, 22]:

- Реєстрація та авторизація користувачів

  У користувача повинна бути можливість створити власний обліковий запис, з можливістю авторизації за рахунок соціальних мереж або за рахунок електронної пошти.

- Фільтрування та пошук

  Фільтр є невід’ємною частиною подібного додатку, щоб локалізувати побажання користувача, це може бути фільтр по стравам або

категоріям. Наприклад гострі/не гострі страви, заклади без вживання м’яса тощо. Також дуже важливою є функція пошуку ресторанів та кафе за різноманітними категоріями. Для прикладу тип кухні, рейтинг, місце знаходження та багато іншого.

- Інформація про ресторани

  Однією з найважливіших частин функціоналу подібного додатку є інформація про заклад. Це має бути повна інформація, що включає до себе меню страв і бару, графік роботи, контактні дані для уточнення замовлення або бронювання, адреса розташування і, якщо є можливість і час, то розташування потрібно інтегрувати на карті і показати користувачу для легшого орієнтування на місці.

- Бронювання столика

  Так як далеко не у всіх подібних додатків є можливість дистанційно забронювати столик, то дана функція хоч і не є першочергова, але може стати дуже актуальною і затребуваною. Підтвердження бронювання має приходити на електронну пошту.

- Відгуки, оцінки та рейтинги

  Статистично на рішення користувача піти до того чи іншого закладу дуже сильно впливає оцінка. Якщо рейтинги у ресторана високі, то і вірогідність того, що погляд користувача впаде саме на цей заклад значно вища. Тому можливість залишати відгуки, ставити оцінку і разом з цим переглядати відгуки інших користувачів є доволі важливою функцією.

- Інтегрована карта

  Інтегрована карта з відображенням усіх ресторанів поблизу та легким і швидким переходом до навігації до обраного закладу є дуже потрібною функцією, особливо якщо рішення приймається швидко, на прогулянці.

- Інформування та сповіщення

Доволі важливо інформувати користувача про оновлення меню, додавання цікавих страв, або сповіщати про цікаві пропозиції або знижки. Також, у разі бронювання столика, нагадування про бронь кардинально знизила б кількість пропущених відвідувань.

- Збір даних та аналітика

  В системі має бути можливість збору різноманітних даних. Це може бути статистика про популярність тієї чи іншої кухні за місяць або ж страви. Також доволі важливим є збір даних про найбільш популярні ресторани. Саме подібна інформація і показує тенденції вподобань користувачів, відносно чого вже відштовхуються власники бізнесів, надаючи послуги, які є найбільш актуальними.

- Реклама

  Можливість реклами та партнерства є доволі важливою для будь якої системи і користувача, який буде користуватись даною системою. Для ресторанів мати змогу розміщувати рекламу і просувати власні послуги є необхідним, адже від цього напряму залежить популярність закладу, а отже і кількість відвідувачів.

1.4 Можливі складнощі використання та реалізації систем агрегаторів ресторанів

Система типу агрегатор ресторанів здатна мати дуже багато переваг, що сприяють зручності для клієнтів, але разом з цим може стикатись з деякими складнощами, які варто враховувати при розробці подібної системи. Ось певні складнощі, що можуть виникати при користування схожими системами:

- Залежність від інтернет-з’єднання

  Комфорт використання подібної системи напряму залежить від наявності постійного інтернет-з’єднання, що на жаль на сьогоднішній день реалізовано не в будь якій точці нашої планети.

- Неактуальна інформація

Часом, інформація про заклади, наприклад меню, графік роботи, адреса, стає застарілою, тобто неактуальною, що в будь якому випадку призведе до незручностей для користувачів.

– Конкуренція

  Заклади, що використовують подібні системи можуть зіштовхнутись з великою конкуренцією, яка в свою чергу ставитиме під питання рівень привабливості бізнесу.

– Плата за партнерство

  Деякі системи можуть просити плату від закладів за кожного клієнта, що обрав даний заклад завдяки додатку, або за можливість рекламування свого ресторану тощо. Це формує певне фінансове навантаження, особливо для малих бізнесів, які тільки починають розвиватись.

Окрім складнощів користування подібними системами, реалізація схожої системи також може бути доволі складним завданням через певну кількість технічних та бізнес-обмежень. Ось деякі з них:

– Технічні проблеми

  В першу чергу системи типу агрегатор ресторанів потребують чіткої, легко масштабованої та ефективної бази даних, яка вміщатиме доволі великий обсяг інформації як про заклади, так і про клієнтів. Також важливим аспектом є розробка легкого та зрозумілого інтерфейсу, від якого у користувачів не виникатиме складнощів і зайвих запитань. На окремому рівні важливості стоїть безпека даних та конфіденційність, яку система має обов’язково забезпечити для користувачів.

– Юридичні обмеження

  Обов’язково потрібно дотримуватись законів та правил, що пов’язані з обробкою персональних даних користувачів та умовами використання різних послуг.

– Маркетинг

Розробка унікальної конкурентної переваги через наявність доволі сильної конкуренції з вже існуючими системами. Окрім цього дуже важливим є формування ефективної стратегії маркетингу, що допомагатиме просуванню системи.

– Технічна підтримка

  Постійне оновлення системи та покращення, що враховують наявність нових технологій і вимог користувачів. Окрім цього дуже важливим аспектом є забезпечення технічною підтримкою для користувачів, яка дозволить швидко реагувати на різні проблеми.

– Фінансові складнощі

  Для реалізації подібних систем потрібна достатня фінансова основа, в першу чергу для розробки платформи, її маркетингу і підтримки.

1.5 Формування та постановка завдання магістерської дисертації

Завданням даної магістерської роботи є розробка додатку-агрегатора ресторанів, що реалізуватиметься шляхом вирішення певної сукупності проблем та викликів, пов’язаних з великою кількістю інформації про заклади та необхідністю створення зручного та легкого інтерфейсу, що стане ефективним інструментом для кожного користувача. Основними завданнями є:

– Зібрати та систематизувати дані про ресторани

  Необхідно створити базу даних закладів, що включатиме дані про розташування, графік роботи, тип кухні.

– Розробити комфортний інтерфейс

  Сформувати мінімалістичний та комфортний інтерфейс для мобільного пристрою, що в свою чергу дозволить легко та швидко знайти необхідний ресторан.

– Реалізувати розроблений інтерфейс

  Реалізувати в середовищі розробки той дизайн інтерфейсу, що було розроблено засобами web-дизайну.

- Реалізувати можливість фільтрації та пошуку

  Реалізувати механізм пошуку закладів за назвою ресторану та за типом кухні.

- Розробити зручний інтерфейс для перегляду детальної інформації

  Забезпечити здатність перегляду основної інформації про заклад. Це має бути адреса розташування, графік роботи, меню та ціни страв.

- Забезпечити конфіденційність даних

  Забезпечити конфіденційність особистої інформації користувачів.

- Виконати тестування

  Виконати тестування основного функціоналу додатка, з фокусуванням та виявленням помилок саме у клієнтській частині.

ВИСНОВКИ ДО РОЗДІЛУ 1

Наш час впевнено можна вважати епохою розквіту інформаційних систем та технологій. Дана сфера стала важливим аспектом нашого життя. Особливо це стало помітним в останні роки.

В даному розділі було оглянуто поняття системи типу агрегатор ресторанів. Сформовано доцільність розробки та актуальні причини реалізації подібного додатку на сьогоднішній день. Окрім цього було розглянуто системи такого типу, які активно й успішно функціонують на сьогоднішній день в різних країнах. Також, відносно розглянутих систем, було сформовано певний набір функціональності, який має бути присутній в системі для комфортного користування користувачами.

В результаті дослідження було виокремлено можливі складнощі реалізації та використання платформ-агрегаторів ресторанів. Описано та проаналізовано кожну проблему, яка може заважати реалізації.

Було встановлено що реалізація платформи-агрегатора ресторанів є доволі актуальною в наш час, і потребує пошуку нових підходів і послуг, які зацікавлять користувачів.

На підставі аналізу проблематики процесів проєктування та розробки клієнтського програмного забезпечення платформ-агрегаторів для ресторанного бізнесу визначено необхідність вдосконалення архітектурної та функціональної концепції та програмних засобів для реалізації високопродуктивної та масштабованої платформи-агрегатора, що забезпечуватимуть гнучкість при масштабуванні користувацького програмного забезпечення.

Для досягнення чого в магістерській дисертації поставлені наступні завдання:

– Проаналізувати відомі платформи-агрегатори, а також відомі інструменти та технології для реалізації легко-масштабованих та

високо-продуктивних платформ-агрегаторів для ресторанного бізнесу з метою визначення проблематики і постановки завдань дослідження.

- Розробити концепцію архітектури та функціонування клієнтської частини платформи агрегатора для ресторанного бізнесу для реалізації високопродуктивної та масштабованої платформи агрегатора.
- Розробити архітектуру клієнтського програмного забезпечення для реалізації високопродуктивної та масштабованої платформи-агрегатора.
- На основі розробленої архітектури розбити користувацький інтерфейс платформи-агрегатора для ресторанного бізнесу.

РОЗДІЛ 2
ОБГРУНТУВАНЯ ВИКОРИСТАННЯ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ТА ПРОЄКТУВАННЯ СИСТЕМИ ТИПУ АГРЕГАТОР РЕСТОРАНІВ

2.1 Реалізація життєвого циклу розробки програмного забезпечення

У даній магістерській дисертації для покрокової реалізації платформи-агрегатора було використано життєвий цикл розробки програмного забезпечення (ПЗ) (Software Development Life Cycle) [23]. Це методологія з чітко окресленими процесами задля створення якісного програмного забезпечення. Якщо говорити чіткіше, то дана методологія формується завдяки наступним фазам розробки ПЗ:

- Аналіз вимог
- Планування
- Проєктування ПЗ
- Розробка ПЗ
- Тестування
- Впровадження

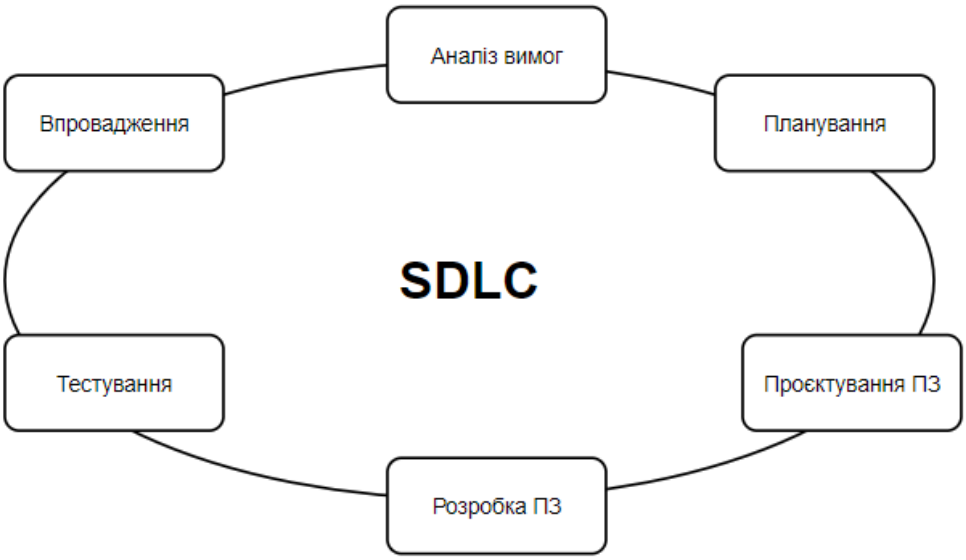

Рис.2.1 – Життєвий цикл розробки програмного забезпечення для платформи-агрегатора [23]

Життєвий цикл розробки ПЗ представляє собою процес, який, в силу своєї структурованості і етапності, дозволяє реалізувати ПЗ дуже високої якості, при цьому використовуючи значно меншу кількість ресурсів і коштів протягом короткого періоду часу. Використання SDLC передбачає добре структурований та прописаний потік етапів, які допомагають команді розробників значно швидше створювати високоефективне ПЗ. Завершальним етапом є тестування, після якого ПЗ загалом готове до впровадження в продакшн, тобто іншими словами система готова до використання. Як вже було сказано вище, SDLC має шість етапів. На сьогоднішній день найбільш популярними серед великих IT компаній є наступні моделі: водоспадна модель, гнучка модель або Agile, спіральна модель.

SDLC значно знижує вартість розробки, і разом з цим покращує якість та зменшує витрати часу. Дана методологія прибирає однотипні пастки проєктів під час розробки ПЗ.

Використовуючи дану методологію, в першу чергу потрібно виконати оцінку вже існуючих систем і платформ на предмет можливих і найбільш частих проблем, дефектів або недоліків. Наступним етапом варто визначити необхідні вимоги до майбутньої системи. Після цього йде власне створення ПЗ, яке проходить через фази аналізу, планування, безпосереднього проєктування та розробки, тестування програмного продукту та реліз. Окрім цього, передбачення недоліків, на зразок поганої зворотної комунікації з замовником, дає можливість SDLC уникнути виправлень програмного продукту після впровадження. Саме це і грає суттєву роль в збереженні коштів, що виділені на розробку системи, адже статистично можна побачити, що саме на виправлення помилок йде величезна кількість ресурсів та коштів.

Дана методологія приділяє дуже велику увагу тестуванню. Так як SDLC являється повторюваною методологією, тобто постійно виконується повторювання етапів перерахованих вище на кожній фазі розробки, необхідно тримати якість коду на високому рівні протягом кожного етапу. Чимала кількість компаній має тенденцію витрачати мало ресурсів на тестування і

детальну перевірку програмного продукту, але саме ця перевірка економить дуже багато коштів і часу розробників, що з високою вірогідністю підуть на переробку дефектів, які можуть виникнути в разі відсутності гарного тестування.

Зазвичай життєвий цикл розробки ПЗ формується з наступних етапів [24]:

– Аналіз вимог

  Першочергово потрібно проаналізувати вимоги до програмного забезпечення. Це саме те що хоче бачити і отримати замовник. Даний етап є одним з найважливіших кроків у SDLC, так би мовити фундамент всього процесу розробки системи. Ним займаються старші учасники команди, окрім цього, в даному етапі приймає участь замовник, тобто клієнт, відділи маркетингу та продажу. Надалі ця інформація застосовується у плануванні головного підходу до проєкту та виконуються певні дослідження, в яких визначається степінь ефективності продукту у різних сферах: економічних, технічних та операційних.

– Планування та визначення вимог

  Даний етап займається плануванням вимог щодо забезпечення якості ПЗ та розглядаються можливі ризики пов’язані з проєктом. В результаті дослідження з технічної сторони, визначають різні підходи розробки, що необхідно буде використовувати для успішної реалізації системи. Хоч всіх ризиків не уникнути, але завдяки цьому етапу їх можна мінімізувати.

  Після завершення етапу аналізу вимог, наступною дією йде коректне і чітке визначення та, обов’язково, документування вимог програмного продукту, які мають отримати затвердження зі сторони аналітиків та замовника. Подібна дія робиться за рахунок документу SRS(Software Requirement Specification). Специфікація вимог до ПЗ

містить абсолютно всі вимоги до програмного продукту, що будуть реалізовуватись протягом всього життєвого циклу.

– Проєктування архітектури ПЗ

  SRS являється нотаткою, такий собі записник для архітекторів системи, що будуть займатись розробкою архітектури продукту. На основі вимог, що були описані у SRS, в більшості випадків пропонується не один, а декілька підходів до проєктування архітектури програмного продукту, які в свою чергу також документуються. Такий документ називається DDS(Design Document Specification) - Специфікація проєктування.

  Даний документ обов'язково розглядається зацікавленими сторонами, і за рахунок різних параметрів, наприклад надійність, оцінка ризиків, обмеження ресурсів і часу, модульність дизайну, обирається найбільш кращий підхід до розробки проєкту.

– Розробка ПЗ

  Даний етап SDLC фактично представляє собою процес розробки і формування програмного продукту. Код системи генерується відповідно до DDS. У випадку, якщо проєктування сформовано доволі докладно і організаційно, то написання коду не викличе особливих проблем.

  Розробники в свою чергу повинні притримуватись правил написання коду, які визначені компанією, включно з інструментами програмування, наприклад середовища розробки, компілятори або інтерпретатори тощо. Мова програмування обирається відносно того, на чому спеціалізується компанія, яка взялась розробляти систему і відносно типу розроблюваного ПЗ.

– Тестування

  Даний етап є фактично підмножиною абсолютно всіх етапів, адже в усіх моделях SDLC, що є найбільш популярними на теперішній час, тестування включено в кожний етап. З іншої сторони даний етап

більшою мірою відноситься до тестування програмного продукту, де відстежуються, виправляються й надалі знову тестуються різноманітні дефекти до тих пір, поки система не досягне стандарту, що визначений у SRS.

– Впровадження

Як тільки система повністю протестована і готова до впровадження, її на офіційному рівні запускають на відповідному ринку. Інколи може бути таке, що впровадження програмного продукту відбувається в декілька етапів, що прописано стратегією замовника. В першу чергу можуть випустити бета-версію для обмеженої кількості людей, для того щоб протестувати, а вже потім повністю випустити систему такою яка вона є або з запропонованими покращеннями.

Будь-яка модель SDLC надає унікальний процес розробки ПЗ для різних вимог що стосуються команди розробників так і безпосередньо проєкту. На вибір моделі в будь-якому разі впливає в першу чергу саме специфікація проєкту та бажані, а отже очікувані результати. Для прикладу модель «водоспад» обирається в тому випадку, якщо у команди розробників не має прямого доступу до замовника або ж зворотний зв’язок доволі обмежений. Це можуть буди різноманітні причини, наприклад різниця часу, якщо замовник знаходиться в одній частині планети, а розробники в іншій, зрозуміло що будуть виникати доволі серйозні проблеми комунікації, які в обов’язковому порядку потрібно виправляти. Але в наші дні модель «Agile», завдяки своїй гнучкості, набирає все більшої популярності, вона дозволяє підлаштовуватись під зміни вимог впродовж всього періоду розробки ПЗ. Основними принципами даної моделі є адаптивність, командна робота, час, ефективний розвиток та тестування, а основними елементами виокремлюють командну роботу та час. «Agile» ділить проєкт на окремі частини, які мають свої часові проміжки, протягом яких повинна виконуватись певна кількість задач. Вона надає перевагу адаптивності, комунікації та співпраці. В кінцевому результаті це забезпечує високоякісний продукт, що відповідає

вимогам замовника. Саме тому і було обрано модель «Agile», яка є найбільш комфортною моделлю для реалізації поставлених завдань.

2.2 Обґрунтування вибору та аналіз сучасних технологій, застосованих для розробки клієнтської частини системи

Далі оглянемо засоби для розробки дизайну систем.

Першою ланкою в розробці клієнтської частини додатку стоїть розробка дизайну. Наразі існує чимала кількість інструментів, що допомагають швидко і комфортно це робити. У дизайнерській спільноті наступні інструменти мають високу популярність та широко використовуються для створення як мобільних додатків, так і веб-додатків. Для того, щоб обрати найбільш комфортний інструмент для розробки дизайну клієнтської частини було проаналізовано такі системи як Figma, Sketch, Adobe Photoshop та InVision.

Система Figma

Figma є одним з найпопулярніших інструментом для дизайну. Існує як веб-версія так і повноцінний додаток з однаковим функціоналом. Однією з особливостей даної системи є можливість спільної розробки в реальному часі і спільного редагування. Це одна з нових платформ, адже була випущена в 2016 році, що по міркам технологій є незначною датою. Figma доступна для різних операційних систем, це і Windows, macOS, Android і iOS.

Дана платформа надає різні інструменти векторного графічного редактора а також прототипування. Окрім цього Figma дозволяє перевикористовувати макети і створювати спільні бібліотеки, в яких стандартизуються компоненти та змінні, це в свою чергу дозволяє заощадити час на створення і зберегти послідовність виконаних дій. Завдяки наявності мобільного додатку для Android та iOS, завжди існує можливість переглядати та редагувати прототипи дизайну в реальному часі як на мобільних пристроях, так і на планшетах. А завдяки технології FigJam, розробники мають змогу

разом працювати над одним і тим самим проєктом в команді в режимі реального часу [25].

На підставі огляду характерних особливостей системи Figma, можна визначити наступні переваги та недоліки [26].

Переваги:

– Легкий і інтуїтивний інтерфейс

  Інтуїтивний інтерфейс являється доволі важливим аспектом у користування будь-яким засобом розробки, адже від цього залежить швидкість і легкість розробки. Саме тому однією з переваг Figma є її інтерфейс. Завдяки цьому користувачу не потрібні роки досвіду або відповідний ступінь щоб створювати якісний дизайн для додатків.

– Можливість одночасної спільної роботи

  Дана платформа дозволяє кільком розробникам працювати над одним файлом в один і той самий період часу. Це дуже велика перевага на фоні інших інструментів для реалізації дизайну, які в більшості випадків дозволяють працювати над макетом лише одному розробнику. Подібний колективний підхід означає, що команди мають змогу працювати значно ефективніше, а редагування і внесення змін можна впроваджувати значно швидше. Окрім цього, можливість спільної роботи дозволяє учасникам команди слідкувати за розробкою один одного і за необхідності вказувати на дефекти або помилки. Отже, розробка стає швидшою, продуктивнішою і ефективнішою.

– Обмін файлами

  Коли відбувається розробка веб-дизайну разом з іншими розробниками, дуже важливо мати змогу легко обмінюватись файлами, щоб кожен хто приймає в цьому участь, міг отримати найсвіжішу версію. За рахунок Figma у користувача є можливість поділитись посиланням на свій документ, і ті, у кого буде посилання, зможуть переглядати або навіть редагувати даний документ. Це дуже сильно економить час.

– Хмарний інструмент для розробки

Однією з головних переваг Figma є те, що дана платформа базується на хмарі, це в свою чергу надає можливість отримати доступ до платформи з будь-якого місця планети за умови підключення до Інтернету. Це дуже велика перевага для будь-якого дизайнера, які постійно потребують співпрацювати з учасниками команди, що розташовані в різних частинах земної кулі.

Недоліки:

– Не можна користуватись без підключення до мережі Інтернет

Одним з головних недоліків є те, що Figma не можна користуватись без підключення до мережі Інтернет. Це викликає дуже великі проблеми для дизайнерів, що працюють у віддалених районах, де немає доступу до мережі або ж зв’язок доволі слабкий. Хоча дана платформа і надає можливість працювати в режимі офлайн, але він значно обмеженіший і дуже багато функцій просто недоступні. Тому даний недолік є однією з найбільших проблем.

– Відсутність глобальних кольорів

Ще одним недоліком є відсутність глобальних кольорів. Це може створювати певні незручності, особливо в тих випадках, коли відбувається робота над дизайном з декількома кольорами і потрібно мати змогу легко їх міняти. Глобальні кольори допомагали б ділитись колірними палітрами з іншими дизайнерами. Хоч в даній платформі можна обходити даний недолік, створивши власну палітру і використовувати її, але б було дуже великою перевагою, якби даний функціонал був в нативному вигляді.

– Параметри пошуку недоступні для локальних компонентів

У Figma у користувача є можливість шукати лише за назвою або типом шару, на противагу у Photoshop і Sketch користувач має змогу шукати за рахунок ключових слів, кольору або ж місцем у дизайні, що розроблюється. Це означає, що якщо користувач хоче знайти

конкретний елемент у документі, це може зайняти доволі багато часу. Іншим недоліком є те, що в даній платформі немає великої кількості плагінів та ресурсів, які створені спільнотою, на відміну від інших систем для розробки дизайну. На кінець, Figma це на сьогоднішній день одна з найновіших програмних рішень, саме тому ще немає достатньої досконалості, як у значно старших програм.

– Потрібний належний обсяг оперативної пам’яті

  Для плавної роботи даної платформи все ж таки потрібна певна кількість оперативної пам’яті і за можливості відеокарта. В іншому випадку можуть виникнути проблеми у користувачів, чиї девайси уже застаріли. Ще однією проблемою є те, що Figma інколи може некоректно працювати так як платформа нова і все ж вимагає доопрацювань. Хоча команда розробників Figma працює над вдосконалення і розвиток ПЗ, деякі дефекти все ж таки потребують нових рішень.

В підсумку, Figma — це дуже потужний інструмент веб-дизайну, що пропонує користувачам чимало переваг. Інтуїтивно зрозумілий інтерфейс і набір функцій роблять дану платформу ідеальним вибором для будь-якого дизайнера, що потребує універсальний і зручний інструмент. Разом з цим Figma має деякі дефекти, наприклад відсутність можливості працювати в режимі офлайн або відсутність глобальних кольорів. Незважаючи на це, в будь-якому разі це потужний і дуже ефективний інструмент, який варто перевірити будь-якому дизайнеру.

Система Sketch

Sketch це редактор векторної графіки для MacOS, що має застосування в цифровому дизайні, так як він не має функцій дизайну для друку. Додаток побачив світ в 2010 році і має доволі велику аудиторію. Основними функціями Sketch є недеструктивне редагування векторів, створення прототипів, ідеальна

точність пікселів. Дана система має змогу синхронізації з сотнями плагінів, експорту попередніх налаштувань і коду.

На підставі огляду характерних особливостей системи Sketch, можна визначити наступні переваги та недоліки [27].

Переваги:

- Дана платформа створена для цифрового дизайну з векторними і точними до пікселів інструментами.
- Дуже велика бібліотека плагінів і різноманітних компонентів для розробки проєктів. Дуже велика спільнота розробників.
- Sketch має можливість застосовувати розумні макети, які автоматично масштабують елементи у межах дизайну.
- Надає можливість проєктування та розробки в єдиному робочому просторі.
- Легке і комфортне експортування коду та різних налаштувань.
- За допомогою Sketch можливо створювати чималу кількість речей, від gif-зображень до макетів та прототипів.

Але разом з перевагами завжди існують і недоліки:

- Найбільшим недоліком Sketch є його одноплатформенність, так як він призначений лише для користувачів систем MacOS. Дана система також не працює в браузері.
- Це доволі важка платформа, що потребує чимало місця на диску та оперативної пам’яті.
- Відсутня сітка повторів.

  Також потрібно додатково встановлювати плагіни для того щоб отримати більш адаптивні інструменти для дизайну.

Система Adobe Photoshop

Adobe Photoshop являється класичним графічним редактором, який на сьогоднішній день залишається доволі популярним серед дизайнерів. Він

широко застосовується для створення складних графічних макетів та дизайну. В загальному це ПЗ для редагування зображень та ретушування фото, яке можна використовувати як на операційній системі Windows так і MacOS. Photoshop надає можливість користувачам покращувати, створювати або якимось чином редагувати ілюстрації та зображення. Існує декілька версій Photoshop: Photoshop CC, Photoshop Elements, Photoshop Lightroom та Photoshop Express, а також версія Photoshop для iOS з обмеженим функціоналом [28].

На підставі огляду характерних особливостей системи Adobe Photoshop, можна визначити наступні переваги та недоліки [29].

Переваги:

- Один з найбільш професійних засобів редагування

  Photoshop являється одним з найпрофесійніших інструментів редагування, які доступні на сьогоднішній день. Він підходить як для простого використання так і для дуже складного графічного проєктування. Дана система має в собі практично всі інструменти, які тільки можуть знадобитись.

- Доступний на всіх платформах

  Adobe Photoshop доступний практично під будь-яку платформу, включаючи MacOS, Windows, Android, iOS, iPad OS тощо. Таким чином немає значення яку операційну систему ви використовуєте, адже Photoshop можна встановити на будь-яку. Звичайно інструменти та певна функціональність можуть різнитись відповідно до операційної системи. Наприклад версії для мобільних пристроїв не мають всіх інструментів, що доступні для ПК або MacOS.

- Підтримка практично всіх форматів

  Дана платформа надає можливість редагувати майже всі формати ілюстрацій. Якщо ж виникла така ситуація, що формат не підтримується системою, то існують спеціальні плагіни, що реалізують сумісність.

– Можливість редагування відео та GIF

  Дану платформу можна використовувати і для звичайного редагування відео або ж GIF (Graphics Interchange Format – формат обміну зображеннями).

Незважаючи на те, що Photoshop є дуже потужною платформою, в ньому є певні недоліки:

– Ціна

  Однією з проблем даної платформи є те, що вона не безкоштовна. Користувачу, потрібно витрачати кошти щомісяця або ж щороку, що є доволі витратно для звичайного користувача.

– Складність в освоєнні

  Якщо користувач тільки починає користуватись та освоювати Photoshop, в нього на початку можуть виникнути певні складнощі. Дана платформа має дуже велику кількість інструментів і величезну кількість функцій, що може доволі сильно заплутати на початку.

– Підтримка векторної графіки

  Вбудована сумісність Photoshop з векторною графікою до сих пір не є ідеальною. Навіть за умови, що користувач здатен розширити функціонал за рахунок додаткових плагінів, це все одно не гарантує що йому вистачить можливостей для редагування абсолютно всіх векторних форматів.

– Вимогливість до ресурсів

  Якщо користувач має середньої потужності девайс, це може призвести до проблем з продуктивністю, затримок під час запуску, проблем з відтворенням графіки або невеликих зображень. За рахунок величезної кількості інструментів, що додано до системи, ПЗ є дуже ресурсозатратним.

Система InVision

InVision являється платформою для розробки дизайну цифрових продуктів. Дана платформа сприяє співпраці в двосторонньому порядку між різними зацікавленими сторонами протягом всіх етапів робочого процесу при розробці дизайну інтерфейсу користувача, від самого початку, тобто задуму, до розробки. Нею користуються численні компанії, що входять до списку Fortune 100, але ПЗ також призначене для використання меншими командами.

Як і більшість інструментів для дизайну інтерфейсу, InVision пропонує дизайнерам створювати шаблони навігації користувацького інтерфейсу, каркаси та прототипи дизайну і планувати маршрути користувача. Використовуючи InVision, різні розробники мають змогу писати відгуки для поліпшення планів, дизайнів та презентацій.

На підставі огляду характерних особливостей системи InVision, можна визначити наступні функціональні можливості [30].

- Чимала кількість інструментів для створення прототипів інтерфейсу без коду. Це готові шаблони, малювання від руки, різноманітна анімація.
- Обмін дизайном і можливість презентування: зустрічі в реальному часі та дошка для планування задач.
- Організація дизайну та співпраця: різні ментальні карти та маркери, які потрібні щоб вказати на того, хто проєктує.
- Є можливість надавати відгуки і коментувати певні частини розробки.
- Є механізм керування проєктами для UI (User interface – дизайн інтерфейсу користувача) дизайнерів.
- Інструменти для тестування та дослідження вподобань користувачів.
- Механізми інтеграції: експорт макетів до іншого ПЗ.

Дана платформа має наступні переваги:

- Простота та доступність

InVision доволі простий і інтуїтивно зрозумілий у використанні для користувачів різного технічного рівня, з функціями коментування, щоб облегшити можливість залишити відгук про певні частини дизайну.

– Масштабованість

  ПЗ розроблено так, щоб задовольнити потреби всіх користувачів, від звичайних фрілансерів до великих компаній з сотнями і тисячами співробітників. Це відображається в гнучкій вартості інструментів InVision.

– Безкоштовний для початківців

  Якщо потреби користувача обмежені, тобто дана платформа використовуватиметься для навчальних цілей, то за умови не більше 10 користувачів, що будуть переглядати та редагувати документи, система надасть безкоштовну модель підписки. З обмеженням в кількості можливих створюваних документів але з усім наявним функціоналом.

– Чудово підходить для співпраці

  Дана система спрощує процес надсилання pdf-файлів та скріншотів розроблюваних макетів. InVision дозволяє користувачам, розробникам та іншим сторонам легко переглядати та коментувати актуальні прототипи на будь-якому пристрої.

– Багатофункціональні інтерактивні прототипи

  В InVision прототипи можна легко анімувати, окрім цього їх також можна використовувати для демонстрацій в реальному часі і тестувати перед впровадженням продукту. Це стосується як веб-версій так і мобільних додатків.

– Історія версій

  InVision зберігає записи про минулі макети, щоб у користувача була повна історія різних версій дизайну програмного продукту.

– Інтеграція

  InVision можна інтегрувати з Slack, Sketch, Atlassian і Adobe.

Недоліки:

– Дорогий для великих команд

  У разі використання даного ПЗ у комерційних цілях, за нього потрібно буде платити. Необхідність платити певну суму за кожного розробника може обійтись дуже дорого компаніям.

– Дефекти інтеграції

  Були випадки коли користувачі повідомляли про проблеми інтеграції даної платформи з іншими системними рішеннями, наприклад Sketch.

– Обмежені можливості співпраці

  InVision дозволяє розробникам залишати відгук в документі, але не надає можливості редагувати документ в реальному часі, як це роблять інші програмні рішення для спільної роботи, для прикладу Figma.

Наступним кроком оглянемо технології програмування для Front-end розробки

Front-end розробка включає в себе створення клієнтської частини як мобільних так і веб-додатків, яка взаємодіє напряму з користувачем. Це включає розробку інтерфейсу користувача, що охоплює відображення контенту, взаємодію з користувачем. До основних мов та технологій, що використовуються у front-end розробці, належать:

– HTML (HyperText Markup Language)

  Використовується для створення структури веб-сторінок, абзаців, зображень та інших елементів.

– CSS (Cascading Style Sheets)

  Використовується для дизайну сторінок, в тому числі кольори, шрифти, та інші аспекти вигляду системи.

– JavaScript

Найпопулярніша мова розробки ПЗ для взаємодії з користувачем в браузері. Основне застосування в створенні динамічного контенту, валідації форм, анімації тощо.

Гіпертекстова мова розмітки HTML

HTML є стандартною мовою розмітки. Веб-розробники використовують дану технологію для створення основної структури веб-сторінки, включно з макетом, текстом і зображеннями.

HTML надає набір тегів і елементів, які визначають спосіб організації та відображення вмісту на веб-сторінці. Користувачі використовують ці теги та деталі, щоб пояснити, що означає вміст, щоб людям було легше зрозуміти його.

Дана технологія також має свої переваги та недоліки [31]. В першу чергу розглянемо переваги:

- Сумісність

  HTML має дуже широку підтримку браузерів, що робить її універсальною мовою для веб-вмісту. Такі популярні веб-браузери, як Chrome, Firefox, Internet Explorer та Safari, відповідають всім стандартам HTML, тому їхні веб-сторінки виглядають однаково на різних платформах.

- Рентабельність

  HTML безкоштовна. Розробникам не потрібно купувати ліцензії або додаткове ПЗ для створення вмісту HTML.

- Легка для вивчення

  HTML дуже проста і легка для вивчення. Навіть школярі мають змогу безперешкодно використовувати дану технологію для написання базової веб-сторінки.

- Гнучкість

  Вільний синтаксис HTML надає розробникам гнучкість. Він дозволяє експериментувати та адаптуватись до різних типів вмісту. Ця

універсальність має особливе значення для створення різноманітних веб-досвідів.

– Швидкість

HTML забезпечує швидке завантаження веб-сторінок. Скорочуючи час завантаження, HTML економить час розробників.

– Збереження даних

HTML розширює свої можливості щодо збереження даних за допомогою такої технології, як XML (EXtensible Markup Language - розширювана мова розмітки). Це в свою сергу забезпечує структуроване зберігання даних у веб-документах, що відповідно робить їх легкодоступними для пошуку та використання програмами.

Недоліки:

– Статичність

HTML є статичною мовою. В її функціонал входить визначення структури та відображення контенту, але сама по собі дана мова не може створювати динамічних взаємодій. Для динамічного функціоналу розробники використовують чисту мову програмування JavaScript або фреймворки JavaScript, які доволі ефективно доповнюють статичну природу HTML.

– Складність структури

Створення та підтримка структури HTML-сторінок доволі складна, в більшості випадків для великомасштабних проєктів. Оскільки веб-сторінки з процесом розробки стають все складнішими, керування вкладеними елементами також стає складним завданням. Ця складність може призвести до великої кількості помилок.

– Слабка безпека

Власне HTML не забезпечує надійного функціоналу безпеки. Дана мова не здатна захистити від різних дефектів веб-сайту, як-от написання міжсайтових скриптів (XSS) або використання ін’єкцій SQL(Structured Query Language – мова структурованих запитів). Ці

проблеми безпеки можуть призвести до втрати даних. Щоб захистити веб-додаток, необхідно вжити додаткових заходів, наприклад перевірки на стороні сервера.

– Довжина коду

  Створення звичайної веб-сторінки за допомогою чистого HTML може призвести до великої кількості рядків коду. Особливо, якщо розробник кодує складні структури. В свою чергу це призводить до повторення коду, проблем з читабельністю і збільшення часу завантаження сторінки.

Каскадна таблиця стилів CSS

CSS надає веб-дизайнерам контроль над тим, яким чином веб-сторінка спілкується з браузерами, включно з форматуванням та відображенням HTML коду. Мова дозволяє розробникам регулювати різноманітні елементи стилю та функціональності, наприклад макет, колір, шрифти.

Розглянемо наступні переваги та недоліки CSS [32].

CSS має наступні переваги:

– За допомогою CSS, користувач повинен вказати повторюваний стиль для елемента лише один раз і може використовувати його кілька разів, оскільки CSS автоматично застосовуватиме необхідні стилі.
– Одна інструкція може контролювати кілька областей, тобто за допомогою CSS стиль застосується на різних сторінках сайту.

Доволі просто мова, тому не потребує великих зусиль для вивчення.

Недоліки:

– Інколи відбувається така ситуація що в одному браузері всі стилі працюють, а в іншому ні. Це призводить до того, що розробники завжди повинні перевіряти як відображається сайт в різних браузерах.
– Завжди існує дефіцит безпеки, тому що мова не надає ніяких засобів для захисту.

Мова програмування JavaScript

JavaScript — це універсальна мова програмування, яка в більшості випадків використовується для створення інтерактивних і динамічних функцій для веб-додатків. Першочергово розроблений для додавання інтерактивності статичним HTML-сторінкам. Наразі JavaScript перетворився на фундаментальний інструмент для веб-розробки.

Відмінною рисою даної мови від інших популярних мов, які в більшості випадків використовуються для сценаріїв на стороні сервера, JavaScript в основному використовується у веб-браузерах для поліпшення взаємодії з інтерфейсом користувача та створення ефективних адаптивних веб-сторінок. Це дозволяє розробникам додавати наступні функції: анімація, оновлення в реальному часі та взаємодія з користувачем, при цьому не вимагаючи від користувачів перезавантажувати всю веб-сторінку.

Як і в будь-якої мови програмування, у JavaScript є свої переваги та недоліки [33]. Оглянемо наступні переваги:

- Швидкість та ефективність

  Користувач можете вільно обирати будь-яку платформу для розміщення файлу JavaScript, тобто з розширенням .js, оскільки він завжди автоматично виконується для перегляду користувачем. Окрім цього, за короткий проміжок часу можна передати велику кількість даних. Ця функціональність дозволяє дуже швидко та ефективно працювати, а також надає потужність для створення якісного та насиченого інтерфейсу.

- Простота

  Цілий веб-сайт (front-end розробка + back-end розробка) можна створити лише за допомогою однієї мови – JavaScript. Він забезпечує мінімальну складність та має доволі простий синтаксис.

- Постійні оновлення

  JavaScript постійно розвивається, на це вказують регулярні оновлення та широкий спектр версій. Організація, яка відповідальна за

стандартизацію JavaScript, ECMA International, регулярно та систематично публікує нові видання специфікації ECMAScript. Ці оновлення надають нові функції, покращують синтаксис і оптимізують продуктивність.

– Універсальний

  JavaScript може змінювати свої ролі від інтерпретованої до об’єктно-орієнтованої мови програмування, процедурної, а також клієнтської мови сценаріїв для веб-сторінок. JavaScript дуже відомий своїми багатими інтерфейсами, що дозволяють розробникам використовувати величезну кількість функціоналу та інтерактивних елементів до веб-застосунків.

– Можливість асинхронного програмування

  Модель асинхронного програмування JavaScript значно полегшує виконання неблокуючих операцій, завдяки цьому дана мова дуже добре підходить для завдань, що передбачають отримання даних із серверів, бази даних, обробку введених користувачем даних і керування одночасними процесами. Дана можливість сприяє підвищенню продуктивності програми.

Недоліки:

– Ризик безпеки на стороні клієнта

  Існує дуже багато варіантів того, як JavaScript відображається у браузері. Через це розробникам доволі важко писати кросбраузерний код. Це в свою чергу створює простір для взлому внутрішніх програм і їх модифікації. Модифікована версія разом із шкідливим кодом може бути виконана в системі клієнта, відповідно надаючи контроль над системою хакеру.

– Читабельність коду

  Будь-хто з користувачів може переглядати код у JavaScript, і це створює дуже великий ризик для безпеки, оскільки полегшує доступ

до коду та в результаті цього його зміну. Це також сповільнює виконання коду, оскільки JS має відстежувати весь видимий код.

– Різне тлумачення в різних браузерах

  Однією з головних проблем JavaScript є його різна інтерпретація різними веб-браузерами. Хоч і існують загальноприйняті стандарти для JavaScript (наприклад специфікації ECMAScript), браузери дуже часто реалізують ці стандарти по-своєму. Це зазвичай призводить до неузгодженості в тому, як сценарій буде себе вести в різних браузерах.

– Єдине успадкування

  JavaScript використовує таку модель успадкування, що є на основі прототипу, тобто дозволяє об’єктам успадковувати властивості і поведінку від інших об’єктів. Однак дана мова програмування підтримує лише одне успадкування, це означає, що об’єкт може успадкувати тільки один об’єкт-прототип.

Насамкінець оглянемо фреймворки та бібліотеки, що застосовуються для швидкої розробки клієнтської частини.

Фреймворк та бібліотека — це два різних поняття, що використовуються у розробці програмного забезпечення для опису засобів та інструментів, які облегшують розробку ПЗ. Хоч багато хто вважає ці два визначення схожими, вони мають різницю в першу чергу в своєму призначенні та підходах до використання.

– Фреймворк

  Фреймворком можна назвати комплексний набір правил, стандартів та інструментів, які в свою чергу визначають структуру програми та, окрім цього, взаємодію основних її компонентів.

  Основним призначенням фреймворку є визначення базової архітектури додатка, що розроблюється та надання загальних рекомендацій щодо вірної взаємодії з певними компонентами. Фреймворк визначає так званий "скелет" додатка, порядок викликів та подій, а також надає структуру для роботи.

– Бібліотека

Бібліотекою являється набір підпрограм, які можна застосовувати для різноманітної кількості завдань, наприклад: обробка зображень, робота з мережею, з файловою системою або з базами даних.

Призначенням бібліотеки є надання готових функцій або класів, які можна легко і без зайвих зусиль використовувати у власному коді. Вона загалом розширює можливості мови програмування, пропонуючи вже реалізовані та готові рішення для конкретних задач.

Фреймворк та бібліотека мають наступну різницю:

– Контроль щодо виконання програми

Фреймворк – в більшості випадків має доволі великий контроль над тим, як програма себе веде та виконується. Розробнику ПЗ необхідно слідувати встановленим правилам та структурі, що прописана фреймворком.

Бібліотека – протягом використання розробник має значно більший вибір, як використовувати бібліотеку у власному коді. Він може використовувати функції бібліотеки в обраному порядку в будь-якому місці ПЗ.

– Керування потоком виконання програми

Фреймворк - загалом надає власний потік виконання та точки входу, які контролюються власне фреймворком.

Бібліотека - зазвичай використовується розробником ПЗ у визначених місцях власного коду за його вибором.

– Обернення керування

Фреймворк - визначає базову структуру та обов’язковим кроком від розробника є реалізація конкретних деталей.

Бібліотека – розробник сам вирішує, де і як використовувати функції бібліотеки.

Обидва підходи мають свої як переваги так і недоліки, і вибір між ними в першу чергу залежить від конкретних вимог проекту.

Найбільш популярними фреймворками для розробки клієнтської частини додатку є Angular та Vue.js.

Фреймворк Angular

Angular — це фреймворк із відкритим кодом, що був створений і підтримуваний компанією Google для створення та реалізації високодинамічних інтерфейсів розроблюваних веб-додатків. Хоч даний фреймворк і був представлений у вересні 2016 року, він має старішу версію, яка відома під назвою AngularJS. AngularJS фактично був попередником нової версії, але ці технології кардинально відрізняються.

Відносно цього Angular не варто розглядати як більш сучасне оновлення AngularJS, оскільки його розробники повністю його переписали.

На підставі огляду характерних особливостей фреймворку Angular, можна визначити наступні переваги та недоліки [34].

Angular має наступні переваги:

– Компонентна архітектура

  Компонентна архітектура є одним із найбільш важливих плюсів Angular. Кожна частина інтерфейсу розроблюваної програми, разом із основною функцією формує фактично окремий компонент. Для прикладу, інтерфейс користувача Facebook містить наступні компоненти: стрічка новин, чат, історії тощо. Усі компоненти самодостатні, хоча й відносяться до одного веб-рішення. Їхні API (Application Programming Interface – прикладний програмний інтерфейс), структури та методи, можуть відрізнятися, але в той же час вони можуть доволі легко «спілкуватися» один з одним. Для розробників подібна архітектура справді є дуже комфортною. Це дозволяє оновлювати незалежні частини(компоненти) програми, і не боятись, що інший функціонал припинить роботу через зміни. На кінець подібна форма архітектури покращує читабельність коду, що робить його значно легшим для розуміння новим розробникам.

– Мобільно-орієнтована філософія

  За допомогою Angular розробники мають змогу створювати доволі легкі веб-рішення, що доволі швидко завантажуються, а це в свою чергу ідеально підходить для смартфонів. Спеціальна технологія, яка називається «відкладеним» завантаженням, великою мірою покращує продуктивність веб-додатку, завантажуючи різноманітні компоненти у браузері тоді, коли відбувається відповідний маршрут.

– Розширена перевірка типу

  Спрощене кодування є доволі великою перевагою Angular відносно інших фреймворків. Angular написаний на TypeScript, а не на JavaScript. Перевагою TypeScript є те, що вона пропонує більш розширені можливості виявлення помилок. Вони в свою чергу дозволяють розробникам виявляти типові помилки та виправляти їх на етапі написання коду. Таке вдосконалення є дуже корисним у випадку розробки складних довгострокових проєктів, тому що воно допомагає команді розробників значно скоротити час, що необхідний для забезпечення якості та налагодження.

– Двостороннє зв'язування даних

  Однією з основних переваг Angular є наявність синтаксису двостороннього зв'язування даних. За рахунок цього дані пов'язані з представленням, тож будь-яка зміна даних в моделі буде негайно відображена в представленні і навпаки. Це дозволяє розробникам ПЗ працювати значно швидше, оскільки синхронізація не вимагає додаткового програмування. Окрім зменшення часу розробки, двостороннє зв'язування даних дозволяє писати більш чистий код і досягати кращої продуктивності.

– Асинхронне програмування

  Асинхронне програмування має дуже важливе значення для сучасних додатків, оскільки з допомогою нього покращується швидкість

операцій і запобігаються зависання системи. Подібна модель дозволяє виконувати кілька команд одночасно.

– Екосистема

  Даний фреймворк доповнюється довгим списком ресурсів для розробників: інструменти, доповнення, плагіни тощо. Подібні інструменти допомагають розробникам ПЗ створювати більш складні веб-рішення, але за коротший період часу, автоматично генерувати документацію тощо.

Окрім переваг, Angular має і недоліки:

– Крива навчання

  Доволі крута крива навчання означає, що розробникам JavaScript на початку може бути складно швидко вивчити дану структуру, оскільки Angular написаний на TypeScript. Отже, розробникам ПЗ, які ніколи раніше не працювали з Angular, потрібно буде витратити деякий час, щоб вивчити всі аспекти. З цієї причини процес адаптації може бути тривалим.

– Часті випуски нових версій

  Команда розробників Google намагається підтримувати актуальність даного фреймворку, тому випускає нові версії Angular два рази на рік. На жаль, перевага наявності сучасної технології йде поряд з необхідністю часто оновлювати рішення, що були реалізовані на основі Angular. Загалом це не викликає значних проблем, але іноді велика кількість версій Angular може бути причиною того, що додаток перестане працювати і інженерам DevOps потрібно буде витратити чимало зусиль для того щоб полагодити застосунок.

– Багато шаблонного коду

  Програма потребує доволі багато шаблонного коду. Якщо користувач хоче створити додаток з використанням Angular у своїй системі, йому в будь-якому разі доведеться встановити кілька речей, і навіть самий

простий додаток «Hello World» буде містити багато непотрібного коду.

Отже, Angular варто використовувати якщо необхідно розроблювати прогресивні веб-програми(PWA - Progressive Web Apps), динамічні веб-сторінки, мобільні додатки. Також даний фреймворк буде дуже корисним для корпоративних веб-додатків. Так як великим підприємствам та установам необхідне велике програмне забезпечення для того, щоб постійно забезпечувати обслуговування баз даних та інших комплексних функцій.

Фреймворк Vue.js

Vue.js на сьогоднішній день є одним із найпопулярніших і найпрогресивніших фреймворків JavaScript, що доступні на ринку. Він призначений для створення та реалізації інтерфейсів користувача на основі вже згаданих вище JavaScript, CSS і HTML. Даний фреймворк пропонує декілька підходів - декларативний і компонентний, це дозволяє розробникам ПЗ створювати швидкий та інтерактивний інтерфейс як для легких і простих, так і для складних програмних продуктів. Окрім того, Vue.js дозволяє створювати односторінкові додатки.

Наразі front-end розробники, що створюють веб-сайти або веб-додатки, як правило, працюють із фреймворками Javascript, Vue.js один з них. Офіційні бібліотеки Vue доволі легко інтегрувати з масою інших бібліотек. Це в свою чергу прискорює весь процес розробки. На підставі огляду характерних особливостей фреймворку Vue.js, можна визначити наступні переваги та недоліки [35].

Даний фреймворк має наступні переваги :

– Простота

  Vue.js складається з однофайлових компонентів. Кожен компонент містити в собі усі типи кодів, як-от HTML, CSS і JavaScript, усе в одному файлі. За рахунок цього розробка ПЗ відбувається з найменшими зусиллями, просто використовуючи пару рядків коду.

– Легкий у вивченні

Загальна ідея під час розробки даного фреймворку полягала в тому, аби дозволити розробникам ПЗ виконувати меншу кількість рядків коду, але досягати при цьому високих результатів. Більшість розробників ПЗ погоджуються з тим, що користувач не повинен бути професіоналом, щоб зрозуміти дану технологію. Все, що потрібно, це мати знання основ, якими є HTML, CSS та JavaScript. Будь який популярний редактори коду, такий як VSCode, WebStorm, Atom або Sublime text, підтримують Vue, що в свою чергу полегшує використання цього фреймворку.

– Легка інтеграція

Розробники мають змогу легко інтегрувати Vue.js в інші фреймворки та бібліотеки. Для прикладу, можна легко інтегрувати Vue з React або Angular і налаштувати проєкт, що розроблюється відповідно до власних вимог.

– Продуктивність та відтворення віртуального DOM

Однією з цікавих особливостей Vue.js є продуктивність віртуального DOM (Document Object Model – об’єктна модель документа) та відтворення. DOM це модель, яка представляє сторінки HTML разом з усіма стилями, вмістом та схожими системними елементами, як об'єкти, що зберігаються у вигляді деревовидної структури, а протягом завантаження сторінки їх генерує браузер. Поки користувач взаємодіє зі сторінкою, об'єкти змінюють свій власний стан, і браузер відповідає як за оновлення інформації так і за відтворення її на екрані.

– Двостороннє зв’язування

Даний функціонал Vue.js успадкував від Angular. Він в основному описує зв'язок, що відбувається між оновленням даних моделі та представленням (UI), зі зв'язаними компонентами, що містять дані, які можна оновлювати час від часу.

Недоліки:

- Мовний бар’єр

  Спочатку даний фреймворк був створений розробниками ПЗ в Китаї і користується дуже великою популярністю саме там. Тому різноманітні обговорення на форумах, описи плагінів або будь-які інструкції розробляються в першу чергу китайською мовою, що означає, що користувачу спочатку потрібно отримати переклад для продовження роботи або щоб отримати відповідь на своє питння. Це інколи може виступати мовним бар'єром.

- Відсутність узгодженого підходу розробки ПЗ

  Оскільки Vue.js - це спільнота, яку фінансують та підтримують розробники, вона не має значних фінансових ресурсів або достатньо сильної підтримки великими компаніями. Технологія не є надійною та не достатньо добре підтримується, якщо виникає необхідність швидкого рішення багів, які виникають в процесі користування.

Зрештою, Vue.js — це невеликий фреймворк, який має дуже великий потенціал. Vue.js пропонує користувачам зручність та розуміння простоти фреймворку. Іншими перевагами фреймворку є швидкість та простота.

Найбільш популярною бібліотекою для розробки клієнтської частини додатку є React.

Бібліотека React

React являється бібліотекою JavaScript для побудови користувацького інтерфейсу, що дозволяє розробникам побудувати складні компоненти інтерфейсу шляхом декларативного підходу тобто з малих частин складати великий проєкт. Хоча React - це переважно засіб для розробки веб-додатків, універсальність React поширюється і на розробку мобільних додатків за допомогою такої технології як React Native. Дана потужна бібліотека з відкритим кодом дозволяє розробляти програмні продукти для Android, iOS та Windows, демонструючи при цьому гнучкість бібліотеки у розробці під різні

платформи. На підставі огляду характерних особливостей бібліотеки React, можна визначити наступні переваги та недоліки [36].

Переваги даної бібліотеки:

- Компонентна архітектура

  React пропонує легкі, незалежні та з можливістю повторного використання компоненти, що дозволяють розробникам оновлювати частини веб-сторінки, при цьому не впливаючи на інші — для забезпечення слабкої зв'язності та спільного функціоналу. Звісно, подібний концепт є не тільки в React; наприклад, Angular також використовує компоненти як основні фундаментальні будівельні блоки. Але велика спільнота React, підтримка компанією Meta та менш крута "крива навчання" роблять його одним з найкращих інструментів для розробки клієнтської частини додатку.

- Розширена можливість налаштувань протягом розробки

  React є достатньо гнучким, це виявляється, коли йде мова про створення індивідуальних додатків, що повинні бути адаптовані до конкретних потреб бізнесу. Зокрема, особливості архітектури на основі компонентів дозволяють збирати складні структур в додатках. Подібний рівень гнучкості підкреслює, чому саме React продовжує залишатися пріоритетним вибором для розробників ПЗ, які прагнуть створити універсальні та надійні веб-додатки.

- Постійні оновлення та розвиток

  React дозволяє інтегрувати чат-боти, підтримані ML (Machine Learning – машинне навчання), а також функції рекомендацій. Крім того, WebAssembly може бути корисним для використання ML-додатків, написаних на Rust, Python або C++, в межах нативного додатка.

- Redux для управління станом

  У односторінкових додатках, також відомих як SPA(Single-page application), де кілька компонентів розташовані на одній сторінці,

керування станом та міжкомпонентною комунікацією може швидко стати доволі складним завданням — саме в таких випадках Redux для React є дуже ефективним. Дана технологія є невід'ємною частиною React, виконує роль "менеджера", який забезпечує послідовний та точний потік даних між розроблюваними компонентами, централізує керування станом та сприяє незалежності компонентів, разом з цим значно покращуючи стійкість даних та користувацький досвід.

Хоч React надає численні переваги розробникам ПЗ різного рівня кваліфікації, він не позбавлений і відповідних недоліків:

– Складні концепції та високорівневі патерни

  React має декілька високорівневих концепцій і патернів, що можуть бути складними для початківців. Розуміння синтаксису JSX, компонентів, різних властивостей, керування станом, методів життєвого циклу та спеціальних так званих хуків потребує міцного розуміння основ JavaScript.

– Складність інтеграції

  React дуже часто використовується в поєднанні з іншими інструментами та технологіями, наприклад Redux, React Router та різноманітні посередники, і чітке розуміння того, яким чином інтегрувати ці технології з даним фреймворком, може бути складним завданням для новачків.

– Бар'єр для розробників, що не знайомі з JavaScript

  Велика залежність React від мови програмування JavaScript може стати бар'єром для розробників ПЗ, які не володіють JavaScript на високому рівні. Хоч JavaScript є універсальною та широко використовуваною мовою, розробники, які приходять з інших програмних середовищ, можуть зіштовхнутись зі складнощами адаптації до парадигм JavaScript та відповідно його використання в React.

– Не повноцінний фреймворк

React в більшості випадків відповідає за частину "Представлення" у архітектурі MVC(Model-View-Vontroller), також відомій як Модель-Представлення-Контролер. Для "Моделі" та "Контролера" необхідні додаткові бібліотеки, в свою чергу може призвести до менш структурованого та потенційно доволі хаотичного коду в порівнянні з повнофункціональними фреймворками, наприклад Angular.

Разом з тим, як простір веб-розробки продовжує розвиватися, гнучкість і достатньо міцна екосистема React розташовують його у вигідне положення. Це загалом буде дозволяти розробникам легко впроваджувати новітні функції в свої додатки. Однак, не дивлячись на те, що React надає багато переваг для розробників, в нього є і недоліки.

Фреймворки та бібліотеки, що були розглянуті вище дозволяють розробникам швидко та ефективно створювати сучасні та ефективні користувацькі інтерфейси. Вибір між ними залежить від конкретних вимог та вподобань розробника.

2.3 Обгрунтування обраного стеку технологій для проєктування системи типу агрегатор ресторанів

Після розглянутих технологій для розробки клієнтської частини додатку було обрано такий інструмент для розробки дизайну як Figma, також обрано мову програмування JavaScript, мови розмітки та стилів відповідно HTML та CSS та фреймворк React.

Було розглянуто декілька популярних інструментів для розробки дизайну, таких як Figma, Sketch, Adobe Photoshop та InVision, з метою вибору найбільш комфортного для потреб розроблюваного додатку. Обравши Figma, увагу було зосереджено на кількох ключових аспектах, які роблять даний інструмент відмінним вибором. Розробка веб-додатку тою чи іншою мірою включає в себе спільну роботу, можливо, з іншими фахівцями в майбутньому. Можливість працювати в режимі реального часу надає змогу всім учасникам бачити оновлення миттєво та ефективно реагувати на ці оновлення. Окрім

цього Figma надає доволі зручні інструменти для створення прототипів, що є дуже важливим аспектом при розробці дизайну. Це дозволяє не тільки створювати дизайн, а й взаємодіяти з ним, покроково тестувати та вносити зміни. Також Figma відзначається інтуїтивним і легким інтерфейсом, що дозволяє швидко освоїти інструмент та зосередитися на розробці. Хоча інші інструменти мають свої переваги, Figma найкраще відповідає конкретним потребам додатку, завдяки своїм функціональним можливостям та можливостями спільної роботи в реальному часі.

Було обрано JavaScript, як основну мову прграмування для розробки клієнтської частини веб-додатку через наступні причини. JavaScript має дуже велику та активну спільноту розробників. Це гарантує наявність великої кількості документації, чималу кількість корисних ресурсів і своєчасну підтримку від інших фахівців у даній галузі. Також дана мова програмування підтримує різноманітні фреймворки та бібліотеки, що дозволяють значно прискорити процес розробки і покращити користувацький досвід. Дуже важливою функціональністю є можливість ефективно обробляти асинхронні події, що робить JavaScript чудовим вибором для розробки веб-додатків. Інтеграція JavaScript у веб-додаток дозволяє використовувати широкий спектр сучасних технологій та забезпечує зручний та ефективний процес розробки.

Оглянувши та проаналізувавши різні фреймворки та бібліотеки було прийнято рішення використовувати бібліотеку React через врахування наступних факторів. React є доволі ефективним та продуктивним за рахунок використання Virtual DOM, дана технологія надає фреймворку можливість оптимізувати продуктивність додатку, що є важливим аспектом при розробці веб-додатків. Компонентна архітектура React дозволяє реалізовувати модульний та легко керуючий код, що є дуже важливо при роботі з клієнтською частиною веб-додатку. Наявність великої спільноти розробників та широка екосистема здатна допомогти у вирішенні проблем та швидкому освоєнні нових принципів та концепцій.

ВИСНОВКИ ДО РОЗДІЛУ 2

У ході написання другого розділу був розглянутий життєвий цикл розробки будь-якого програмного забезпечення разом з чітко окресленими процесами, що дозволяють створити якісне програмне забезпечення, а також описані всі необхідні етапи створення системи й використані методології, що застосовуються командами розробників по всьому світу протягом багатьох років. Деякі моделі використовуються менше, деякі більше, але всі вони мають важливе значення для гнучкості розробки в залежності від вимог замовника.

Також були оглянуті й проаналізовані різні технології та інструменти для ефективної розробки клієнтської частини веб-застосунків. В першу чергу проведено докладне вивчення найпопулярніших інструментів для розробки дизайну, які активно використовуються на сьогоднішній день. Надалі було оглянуто та проаналізовано необхідні мови програмування для реалізації якісного та ефективного клієнтського інтерфейсу. Окрім цього, було обрано найбільш ефективні технології програмування для розробки, а також проаналізовано найпопулярніші фреймворки для цієї мови, що надають додатковий функціонал для полегшення та прискорення процесу створення програмного продукту.

РОЗДІЛ 3
РОЗРОБКА КЛІЄНТСЬКОЇ ЧАСТИНИ ВЕБ-ДОДАТКУ ПЛАТФОРМИ-АГРЕГАТОРА РЕСТОРАНІВ

3.1 Розробка архітектури платформи-агрегатора

Вибір архітектури ПЗ є одним з найважливіших етапів розробки будь-якого ПП. Архітектура ПЗ визначає те яким чином структурована програмна частина системи, і як саме її компоненти мають змогу ефективно взаємодіяти, обмінюватися інформацією та розширюватися незалежно один від одного. Відповідна, вірно обрана, архітектура ПП забезпечує швидку та ефективну роботу всієї системи, не впливаючи на інші компоненти.

3.1.1 Клієнт-серверна архітектура

В ході аналізу різноманітних архітектурних рішень, для розробки даної системи було застосовано клієнт-серверну архітектуру. На сьогоднішній день, подібна архітектура являється однією з найпопулярніших. Особливістю даного підходу є поділ системи на 3 основні частини: клієнт, сервер, база даних. Кожен компонент виконує свої конкретні завдання і взаємодіє один з одним через мережу.

Основними характеристиками клієнт-серверної архітектури є [37]:

- Клієнт

  Користувацький інтерфейс: клієнтська частина відповідає за відображення інформації та взаємодію з користувачем через графічний інтерфейс.

  Обробка даних: клієнт обробляє локальні обчислення та взаємодіє з користувачем.

- Сервер:

  Збереження даних: сервер забезпечує централізоване збереження та керування даними, що можуть бути спільно використані клієнтами.

Бізнес-логіка: виконання бізнес-логіки, обробка запитів та забезпечення взаємодії з базою даних.

– Мережа

Передача даних: клієнти та сервери обмінюються даними через мережу, яка може бути Інтернетом, локальною мережею або іншими засобами передачі даних.

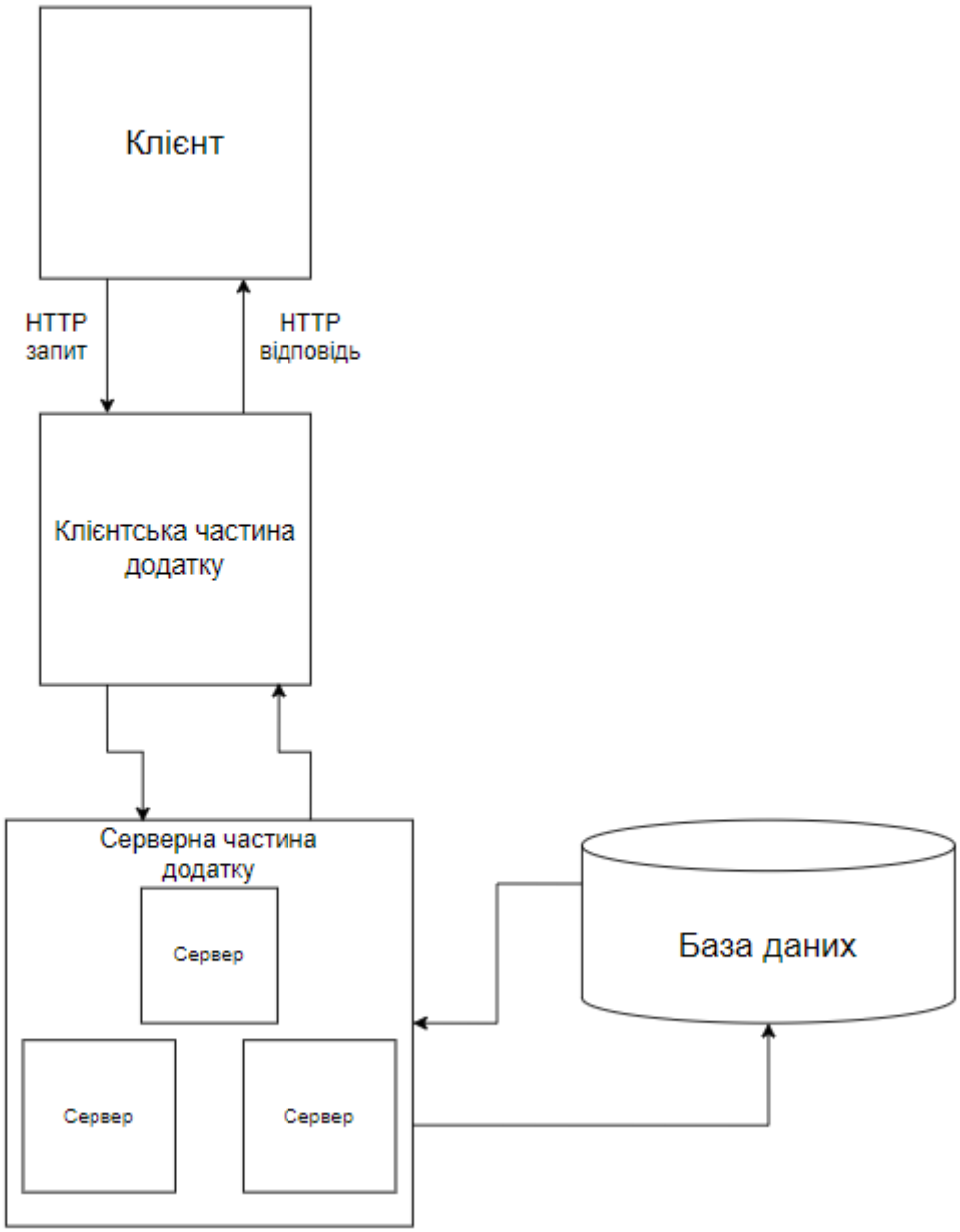

Рис. 3.1 Клієнт-серверна архітектура

Розроблювана система представляє собою веб-додаток, саме тому клієнтом в даній архітектурі виступатиме веб-браузер. В більшості випадків клієнт представляє собою користувацький інтерфейс, його основним завданням є надання зручних механізмів для взаємодії потенційного користувача з системою. В даній системі взаємодія між сервером та клієнтом забезпечується за рахунок HTTP (Hypertext Transfer Protocol – протокол передачі гіпертексту) протоколу.

Подібний підхід поділу веб-застосунку на декілька частин надає незалежність компонентам системи, що в свою чергу дозволяє значно краще масштабувати систему та керувати.

3.1.2 Патерн проєктування MVC

MVC (Model-View-Controller) – це патерн проєктування [38], який використовується для побудови ПЗ, де обов'язки розділені між трьома основними компонентами:

- Модель (Model)

  Відповідає за надання та обробку даних, які використовуються в додатку. Виконує операції над даними, наприклад: збереження, видалення, оновлення та отримання. Модель не залежить від інших компонентів і не знає про їх існування.

- Представлення (View)

  Відповідає за відображення даних та користувацький інтерфейс. Отримує дані від моделі та безпосередньо відображає їх користувачеві. Не містить ніякої бізнес-логіки, а лише відображає дані, що передаються.

- Контролер (Controller)

  Відповідає за обробку інформації, що ввів користувач та керування потоком даних між моделлю та представленням. Забезпечує взаємодію із представленням та моделлю, але сам не містить бізнес-логіки.

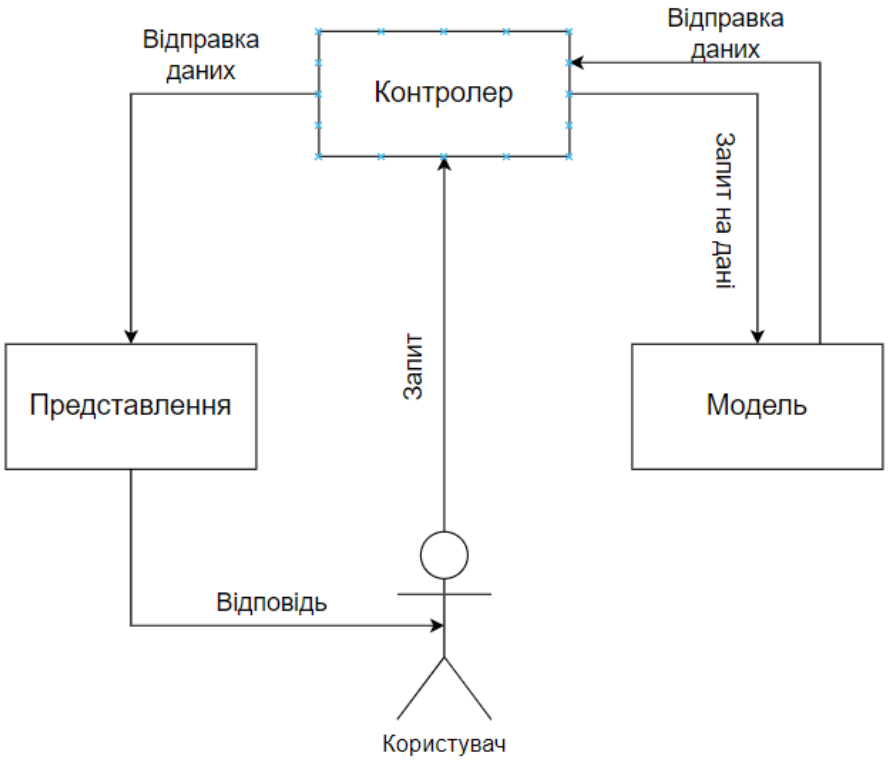

Рис. 3.1 Патерн проєктування Модель-Представлення-Контролер

3.2 Розробка користувацького інтерфейсу платформи-агрегатора

3.2.1 Розробка компонентів для реєстрації та входу в застосунок

В ході розробки структури користувацького інтерфейсу заздалегідь необхідно розробити шаблони та виконати детальний аналіз кожного з користувачів. Обов'язково потрібно врахувати відмінності функціоналу кожної з ролей користувача системи.

Перед тим як дозволити ввійти та виконувати певні дії, інтерфейс повинен ідентифікувати користувача в веб-застосунку. Автентифікація та авторизація користувача відбувається за рахунок введення необхідної інформації для входу. В даному випадку даними для входу є адреса електронної пошти та пароль. Після введення, дані обробляються відповідними API. В свою чергу API надсилають відповідь, щодо користувача. Навіть якщо користувач не авторизований, він має певний обмежений доступ до застосунку, він може лише переглядати головну сторінку з повним

переліком ресторанів та закладів. Якщо користувач зареєстрований та авторизований, то відповідно до своєї ролі, матиме певний функціонал.

На Рис.3.1 зображено форму для автентифікації та авторизації користувача. На сторінці в верхньому лівому кутку вікна розташовано назву та логотип застосунку. По центру вікна розташовуються поля для введення необхідних для входу даних. Основними кольорами додатку було обрано білий, відтінки сірого та червоного. Власне сторінка для входу була реалізована мінімалістично, аби бути достатньо інтуїтивно зрозумілою для нового користувача. Лише після введення електронної адреси та паролю, у користувача буде можливість повноцінно користуватись функціоналом системи.

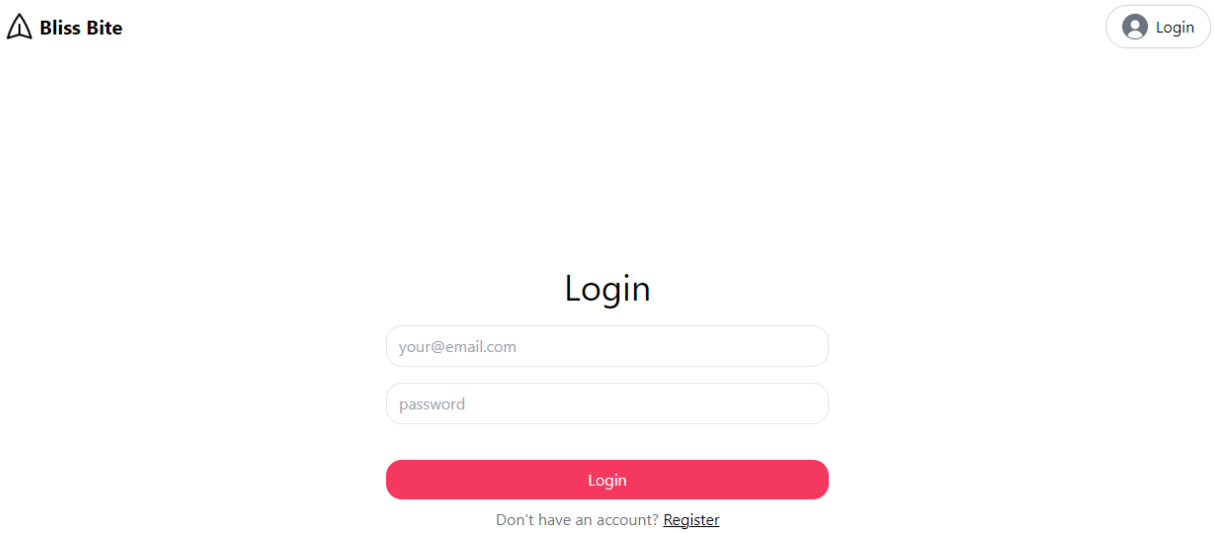

Рис. 3.1 Форма для входу користувача

У випадку, якщо користувач не зареєстрований та не має облікового запису, у нього є можливість перейти по посиланню, що знаходиться під кнопкою «Login» та виконати всі необхідні дії для того щоб успішно зареєструватись. Після заповнення всіх полів відповідними даними, користувач повинен обрати чи буде він займатись адмініструванням закладу, або ж бути звичайним клієнтом. Насамкінець необхідно натиснути на кнопку «Register», завдяки якій відбудеться збереження всіх даних нового користувача в системі, і в нього з’явиться можливість ввійти в додаток. Для

переходу на сторінку входу в застосунок, на сторінці реєстрації передбачено посилання під кнопкою «Register».

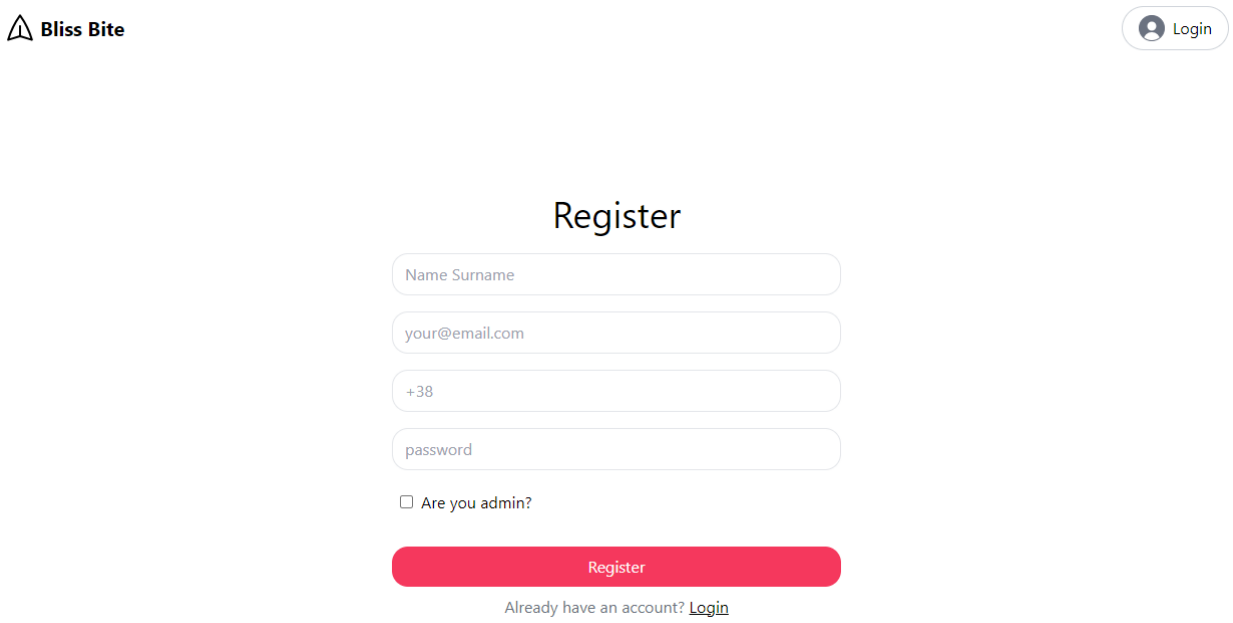

Рис. 3.2 Форма реєстрації нового користувача

Для реалізації сторінки входу та сторінки реєстрації було створено два окремих компоненти LoginPage.jsx та RegisterPage.jsx.

```
return (
    <div className="mt-4 grow flex items-center justify-around">
        <div>
            <h1 className="text-4xl text-center mb-4">Login</h1>
            <form className="max-w-md mx-auto" onSubmit={handleLoginSubmit}>
                <input
                    type="email"
                    placeholder="your@email.com"
                    value={email}
                    onChange={(ev : ChangeEvent<HTMLInputElement> ) : void  => setEmail(ev.target.value)}
                />
                <input
                    type="password"
                    placeholder="password"
                    value={password}
                    onChange={(ev : ChangeEvent<HTMLInputElement> ) : void  => setPassword(ev.target.value)}
                />
                <button className="primary mt-6">Login</button>
                <div className="text-center py-2 text-gray-500">
                    Don't have an account?{' '}
                    <Link to={'/register'} className="underline text-black">
                        Register
                    </Link>
                </div>
            </form>
        </div>
    </div>
);
}
```

Рис. 3.3 Компонент сторінки для входу в застосунок LoginPage.jsx

3.2.2 Розробка компонентів для користувача з роллю «Адміністратор»

Основними функціями адміністратора є повне управління закладом. Саме він відповідає за додавання повної інформації про заклад. В його повноваження також входить можливість реагувати та відповідати на коментарі клієнтів, що відвідали той заклад, яким він керує.

Щоб переконатись в тому що автентифікація та авторизація пройшли успішно, можна поглянути в верхній правий куток вікна застосунку. Там буде зображено ім’я користувача, що було ведено при реєстрації.

На Рис.3.3 зображена основна сторінка адміністратора для керування закладами. В першу чергу він має змогу дивитись список всіх закладів, якими йому доручено керувати. Це може бути як один заклад, так і п’ять. Все залежить від власника ресторанного бізнесу.

Кнопка «Add new place» дозволяє створити новий заклад і додати всю необхідну інформацію. На картці самого закладу зображено три кнопки. «Menu» дозволяє перейти на компонент керування стравами закладу. «Edit» забезпечує внесення коректив та виправлень у вже створений заклад, «Delete» видаляє заклад з списку та з бази даних.

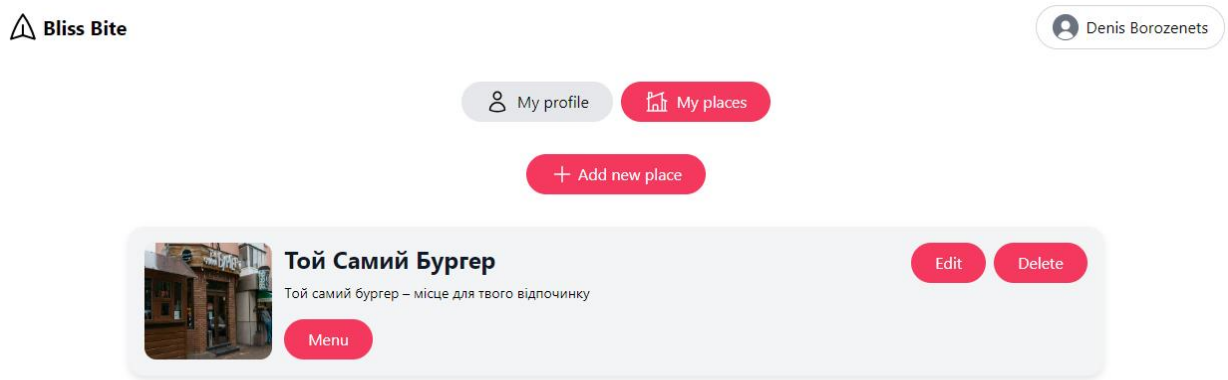

Рис. 3.3 Основна сторінка адміністратора

Після натиснення кнопки «Add new place» адміністратор переходить на іншу сторінку для того щоб додати новий заклад. Умовно її можна поділити

на декілька частин. Сама ж сторінка створена з різних менших частин, тобто компонентів.

1. Першочергово адміністратор повинен вести назву закладу.
2. Наступне поле вимагає введення унікального ідентифікатора для закладу, який можна отримати перейшовши на Google карти, та знайшовши заклад на мапі.
3. Потім йде покрокове введення контактних даних закладу: повна адреса, номер телефону, електронна адреса та, за наявності, веб-сайт.

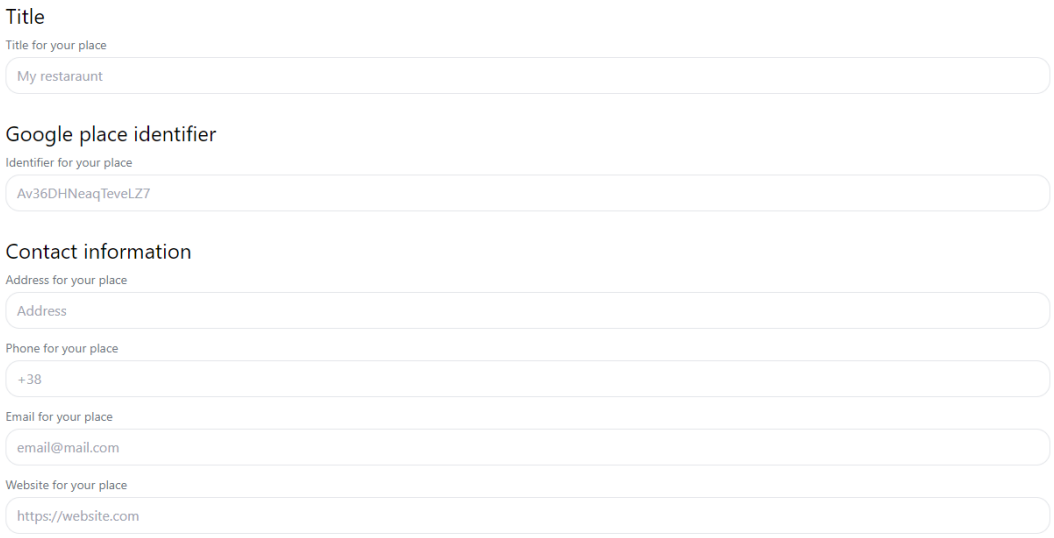

Рис. 3.4 Перша частина для заповнення даних закладу

Друга частина вимагає від адміністратора задати графік роботи закладу, цінову категорію та опис. В даному випадку існує чотири цінові категорії від 1 до 4, відповідно 1 - заклад з найбільш лояльними та низькими цінами на страви, а 4 – це дуже дорогий заклад з достатньо високою ціновою категорією. Опис є дуже важливою частиною заповнення даних, в якому необхідно розповісти про інтер’єр або ж особливості закладу.

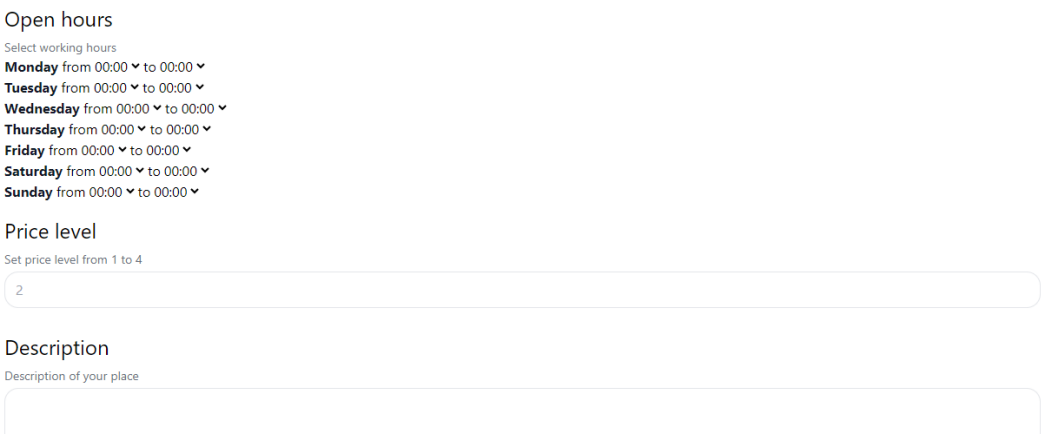

Рис. 3.5 Друга частина для заповнення даних закладу

Остання частина відповідає за загальну інформацію про заклад. Для кращого сприйняття, було прийнято рішення зробити даний функціонал так званими етикетками.

Спочатку адміністратор повинен обрати тип кухні, який заклад пропонує клієнту. Наступним йде можливості сервісу. Адже не кожен ресторан пропонує доставку своїм клієнтам або можливість самому забрати страву.

Насамкінець зображена кнопка для того щоб додати фото закладу, адже естетика дуже важлива при виборі місця відпочинку і для багатьох клієнтів вона має вирішальне значення.

Після того як адміністратор перевірив коректність введених даних, він повинен натиснути кнопку «Save», після чого його перекине на попередню сторінку і список буде доповнено новим закладом.

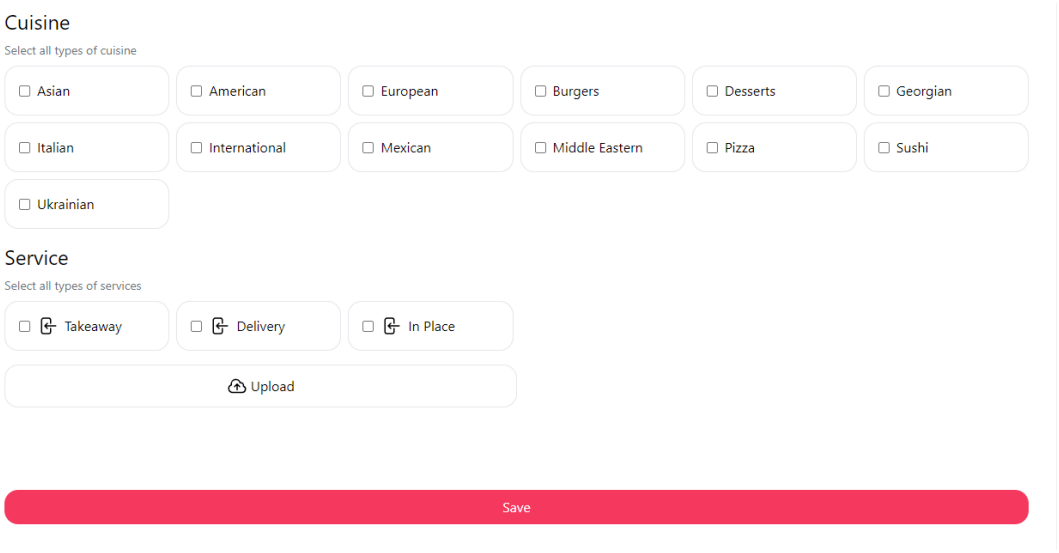

Рис. 3.6 Остання частина для заповнення даних закладу разом з кнопкою збереження

Так як сторінка для додавання нового закладу реалізована за рахунок компонентного принципу, на Рис.3.7 зображено один з компонентів під назвою Perks.jsx, що реалізує формування етикеток для комфортного вибору типу кухні та сервісу.

```
    return (
        <div className="grid gap-2 mt-2 grid-cols-2 md:grid-cols-3 lg:grid-cols-6">
            {items.map((item, index) => (
                <label key={item} className="border p-4 flex rounded-2xl gap-2 items-center cursor-pointer">
                    <input
                        type="checkbox"
                        checked={selected.includes(item)}
                        name={item}
                        onChange={handleChbClick}
                    />
                    {images && images[index]}
                    <span>{item}</span>
                </label>
            ))}
        </div>
    );
}
```

Рис. 3.7 Компонент для формування етикеток для типу кухні і сервісу

Наступним кроком після додавання закладу, є необхідність сформувати меню. Це є дуже важливим етапом, адже клієнту важливо бачити перелік страв, і, можливо, навіть зарання обрати щось, що б він хотів спробувати в залежності від різноманітності страв і їх ціни.

Для того щоб це зробити, адміністратор повинен на основній сторінці обрати необхідну картку закладу та натиснути «Menu». На Рис.3.8 зображено сторінку з вже доданими картками страв для кращого візуального сприйняття.

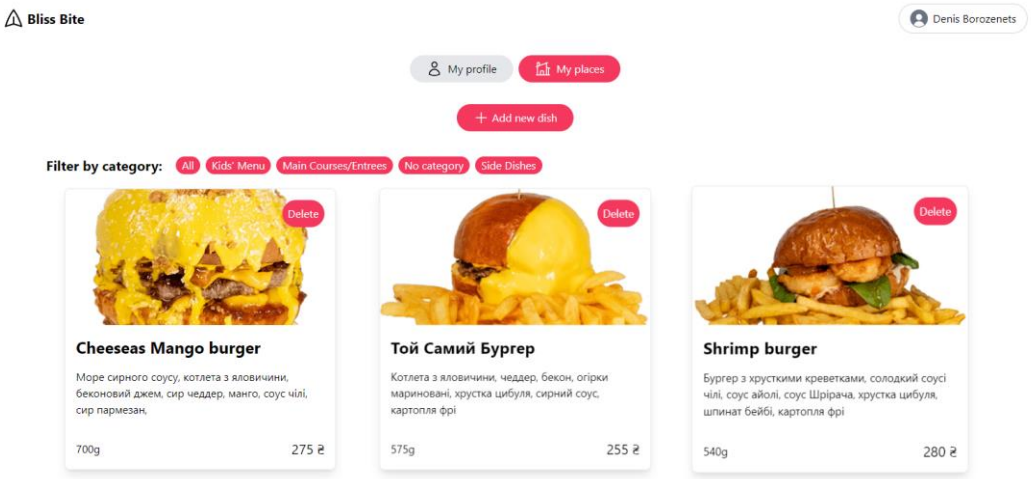

Рис. 3.8 Сторінка для формування меню

Щоб додати страву до меню, адміністратор повинен натиснути кнопку «Add new dish». Після цього відкриється модальне вікно, що зображено на Рис.3.9. Далі необхідно заповнити даними всі поля відповідно до назви кожного поля. У кожної страви є своя категорія, яку також потрібно обирати. Так як страв може бути дуже велика кількість, необхідно реалізувати спосіб легкого та ефективного фільтрування даних. Саме з цих причин і було додано подібний функціонал.

Останнє, що повинен виконати адміністратор, це обрати фото страви та додати його до картки.

Після всіх виконаних дій необхідно натиснути кнопку «Save» та зберегти картку в загальний список страв.

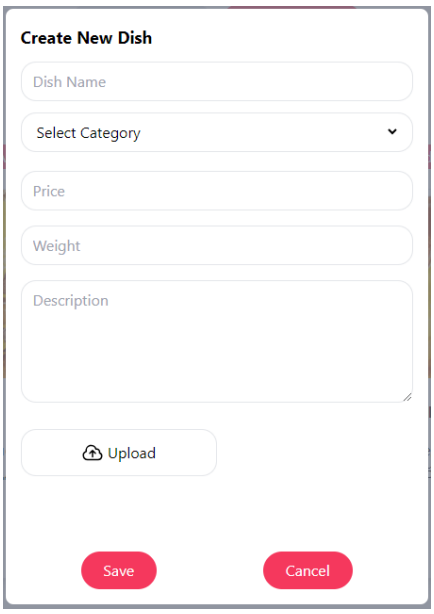

Рис. 3.9 Модальне вікно для додавання страви

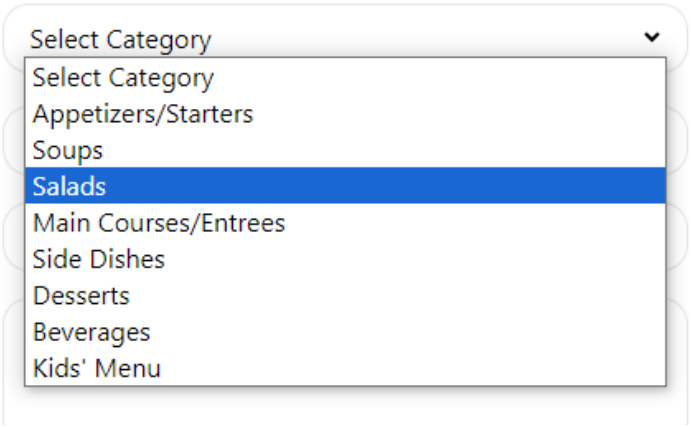

Рис. 3.10 Випадаючий список вибору категорії

Після того, як страву було збережено, модальне вікно автоматично закриється, а сторінка перезавантажиться. Для більшого комфорту користувача, було реалізовано адаптивність, яка реагує на зміни розміру екрану, зберігаючи при цьому естетичний вигляд. Дану реалізацію можна спостерігати на Рис. 3.11.

Також було додано фільтр категорій, для більш комфортної роботи з додатком. Якщо адміністратор не обирає ні одну з категорій, за замовчуванням будуть відображатись всі страви. У випадку коли потрібно знайти страви певної категорії, необхідно натиснути кнопку, що відображає відповідно

категорію. Після натискання дана кнопка змінить стиль, що символізуватиме відпрацювання фільтру. Для того щоб відмінити фільтр, потрібно ще раз натиснути на ту ж саму кнопку.

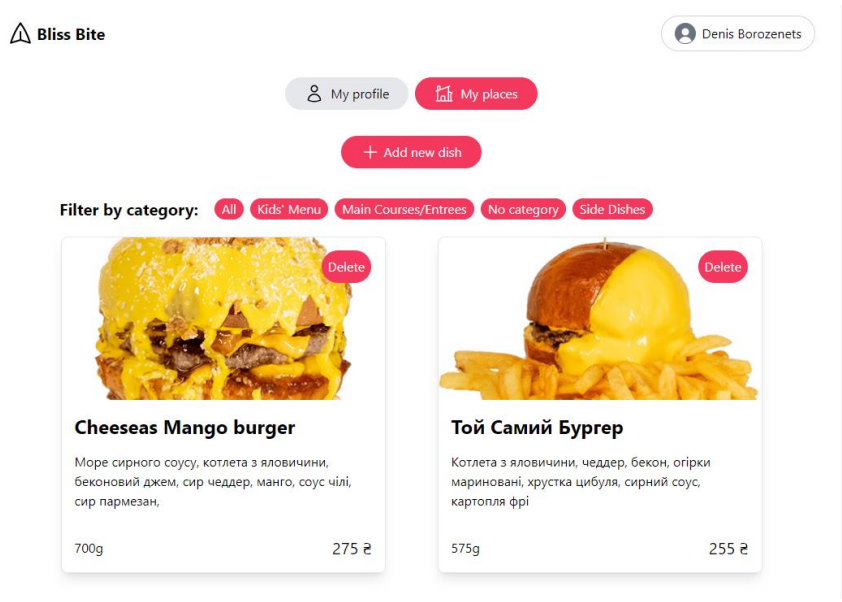

Рис. 3.11 Реалізація адаптивності при зменшенні розмірів екрану

У випадку, якщо потрібно внести певні корективи в дані про страву, необхідно просто ще раз натисну на карту страви і також відкриється модальне вікно з вже заповненою раніше інформацією. Адміністратор повинен внести корективи у відповідне поле і натиснути кнопку «Save».

Якщо необхідно повністю видалити страву, необхідно натиснути кнопку «Delete» в верхньому правому кутку картки страви.

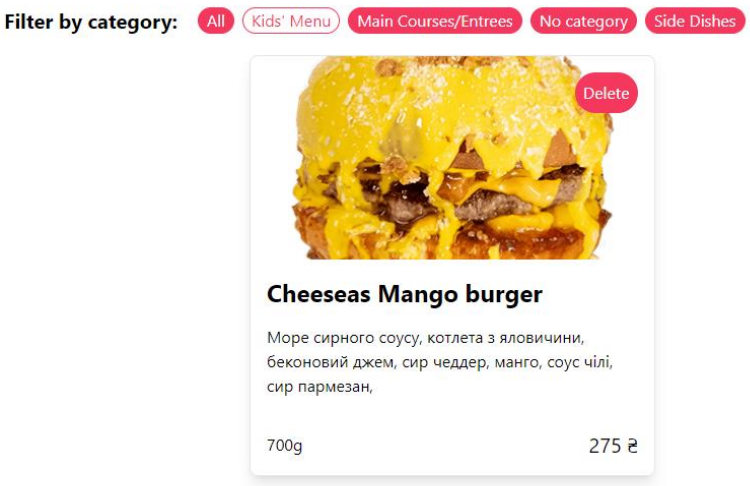

Рис. 3.12 Реалізація фільтру

Ще одним обов’язком адміністратора закладу є необхідність відповідати на коментарі клієнтів. У звичайного користувача є можливість залишити відгук про заклад, який він відвідав, позитивний чи негативний, та поставити оцінку. Подібний функціонал було розроблено для покращення зв’язку між закладом та клієнтом. Окрім цього, коментарі відкриті іншим користувачам, і вони безпосередньо впливатимуть на вибір майбутніх клієнтів. Позитивні відгуки приваблюватимуть відвідувачів, а от негативні відповідно відлякуватимуть.

Для того щоб відповісти на коментар, адміністратору потрібно натиснути стрілочку справа. Після цього відкриється місце для написання тексту. Як тільки відповідь написана, потрібно натиснути кнопку «Submit Reply»

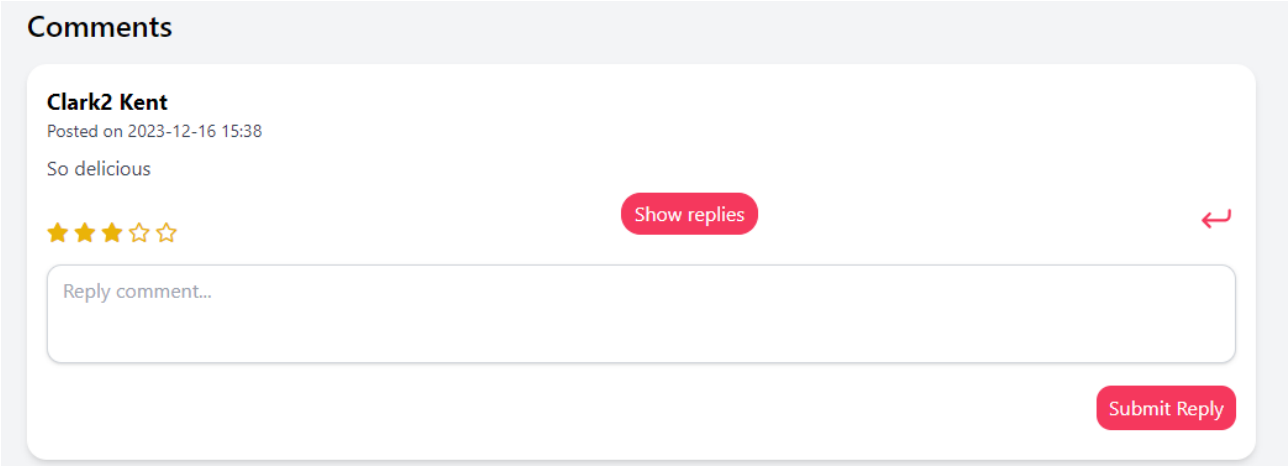

Рис. 3.12 Механізм відповіді на коментар

3.2.3 Розробка компоненту керування обліковим записом

У користувача з роллю «Адміністратор» та у звичайного авторизованого користувача є можливість змінити власні дані. Все що потрібно, це натиснути в верхньому правому кутку на іконку з власним ім’ям. Після цього відкриється вкладка під назвою «My profile».

Будь який користувач має змогу змінити електронну адресу, ім’я та прізвище, мобільний телефон та пароль. Якщо всі корективи внесені,

необхідно натиснути кнопку «Save changes» і всі нові дані будуть збережені в базі даних.

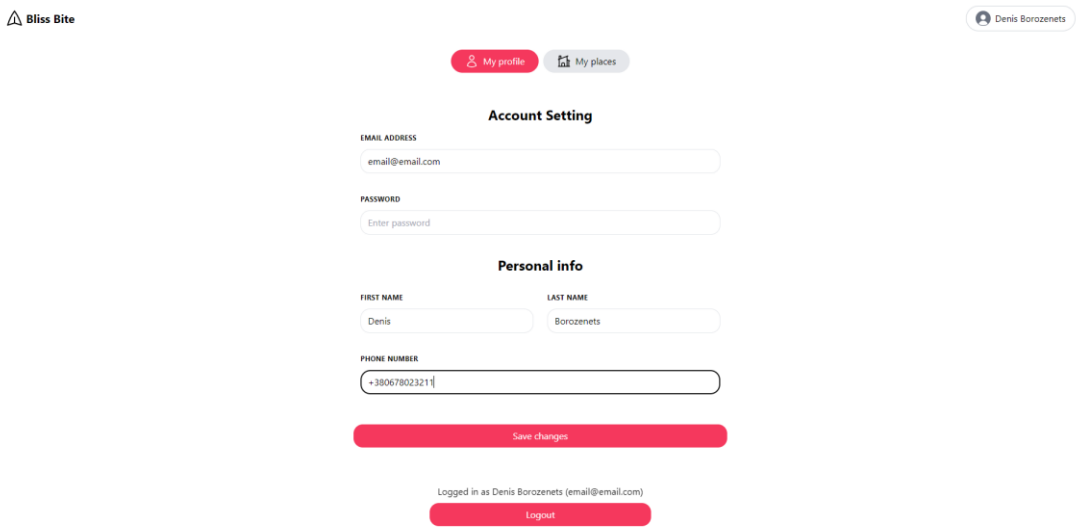

Рис. 3.12 Сторінка керування обліковим записом

Компонент, що відповідає за відображення сторінки керування обліковим записом називається ProfilePage.jsx. Його зображено на Рис.3.13. Даний компонент вміщує в собі ще два інших компоненти <AccountNav /> та <UserProfileComponent>, які і формують дану сторінку. Саме таким чином і реалізується компонентний підхід, який надається бібліотекою React.

```
    return (
        <div>
            <AccountNav/>
            <UserProfileComponent/>
            <div className="text-center max-w-lg mx-auto">
                Logged in as {userData.name} ({userData.email}) <br/>
                <button onClick={logout} className="primary max-w-sm mt-2">
                    Logout
                </button>
            </div>
        </div>
    );
}
```

Рис. 3.13 Компонент для відображення сторінки керування обліковим записом

Розробка компонентів для звичайного користувача

У авторизованого звичайного користувача є доступ до всіх закладів головної сторінки. Першочергово він має змогу ознайомитись з короткою інформацією. На кожній картці закладу відображено наступне: фото закладу або логотип, рейтинг, що формується за рахунок відгуків клієнтів, які відвідали даний заклад, тип кухні, цінова категорія, місце розташування на карті та посилання на меню, яке заклад пропонує.

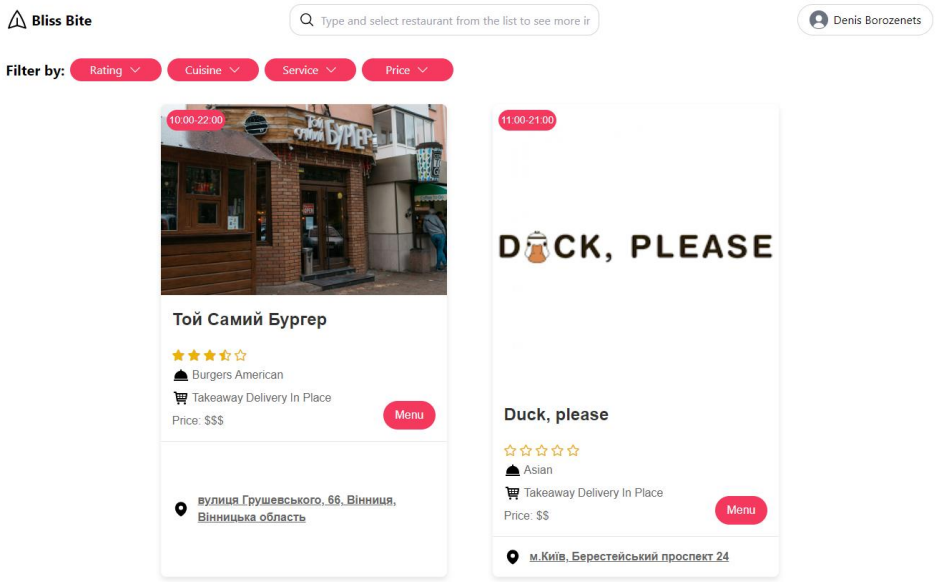

Рис. 3.14 Сторінка відображення всіх закладів

На головній сторінці може бути велика кількість різноманітних закладів, зі своє концепцією та особливостями. Саме тому розроблено фільтр, який може обробити запит за наступними особливостями: рейтинг закладу, тип кухні, послуги сервісу та цінова категорія. Завдяки чекбоксам, користувачу дозволено обирати декілька варіантів фільтру. Це може бути корисним якщо клієнту необхідно знайти заклад, який пропонує азіатську та європейську кухні тощо.

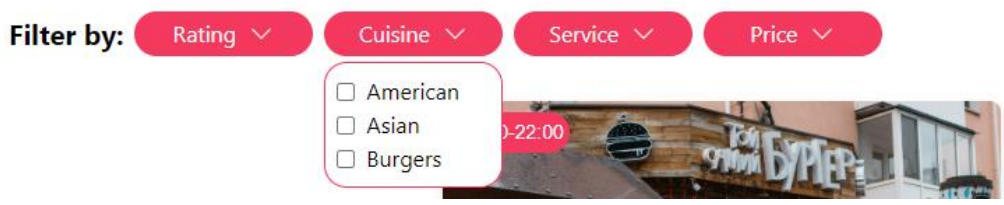

Рис. 3.15 Реалізація фільтру закладів

Якщо виникла ситуація, коли користувач знає назву закладу, але не знає його особливостей, та будь-якої інформації, то в такому випадку можна скористатись пошуковим рядком. Необхідно лише почати вводити заклад і результат пошуку надасть очікуваний результат. Надалі користувач може натиснути на назву закладу і перейти на сторінку детальної інформації.

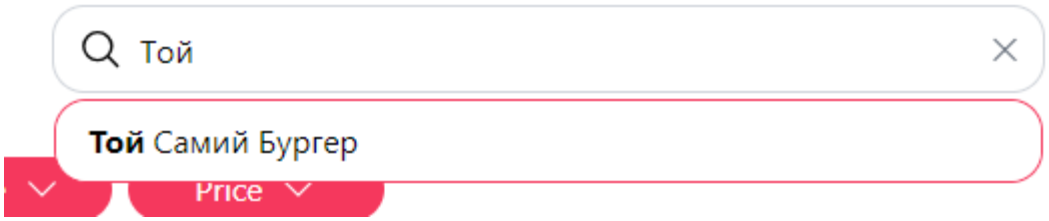

Рис. 3.16 Рядок пошуку закладів

Перейшовши на сторінку детальної інформації закладу користувач може отримати всі необхідні дані.

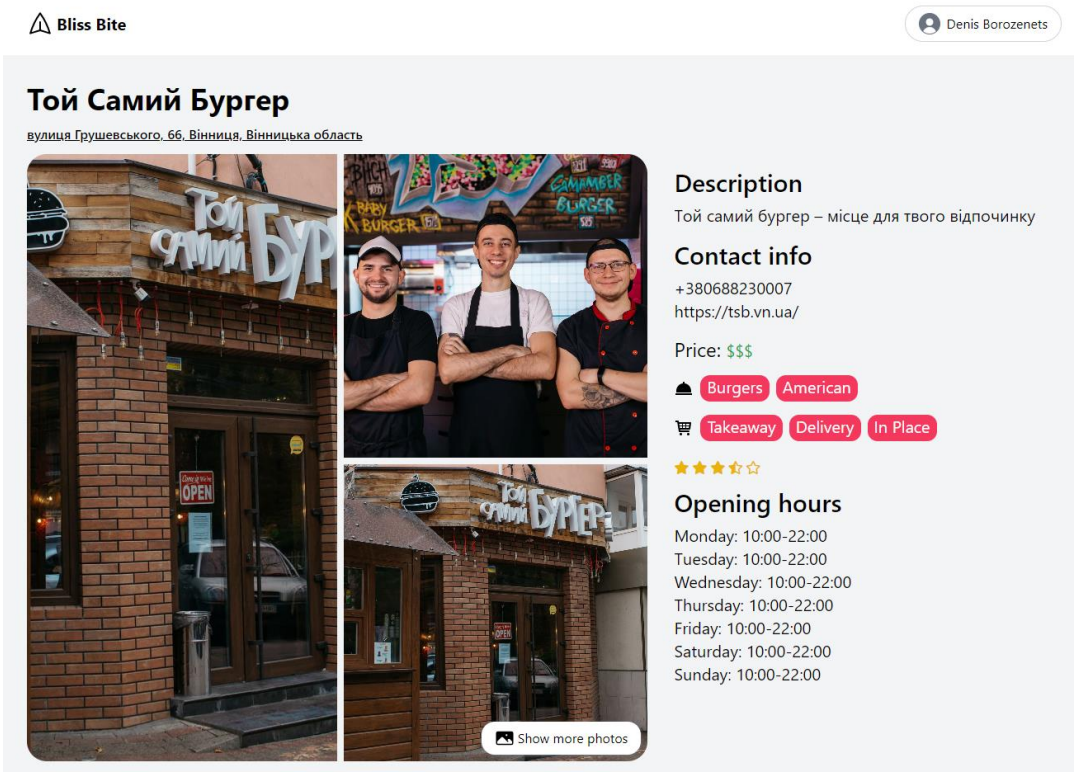

Рис. 3.17 Сторінка детальної інформації про заклад

Якщо користувач має бажання переглянути всі фото закладу, він може натиснути кнопку «Show more photos». Після цього для користувача будуть доступні всі фото закладу, які були додані адміністратором.

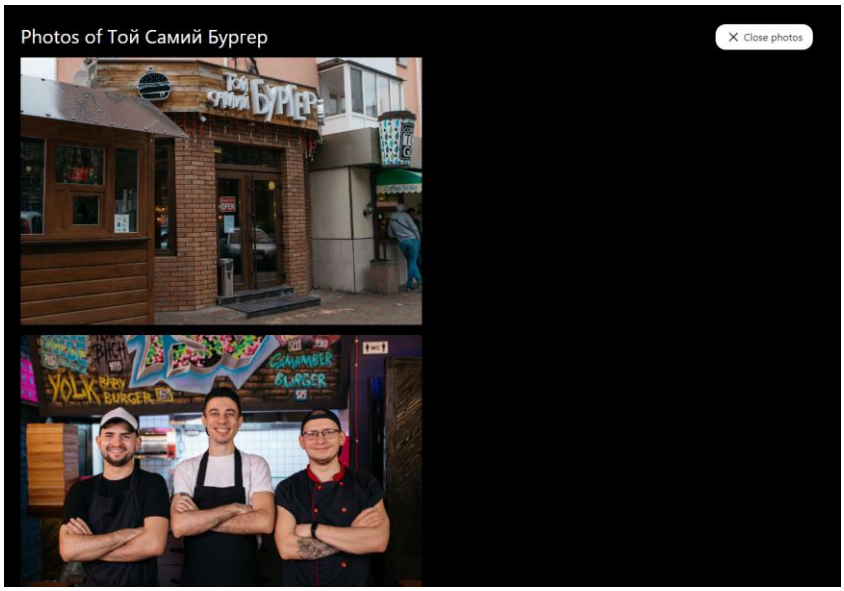

Рис. 3.18 Сторінка детальної інформації про заклад

Окрім цього додано посилання місця розташування закладу разом з міткою, щоб відвідувач міг зорієнтуватись на карті і зрозуміти яким чином він може потрапити до закладу.

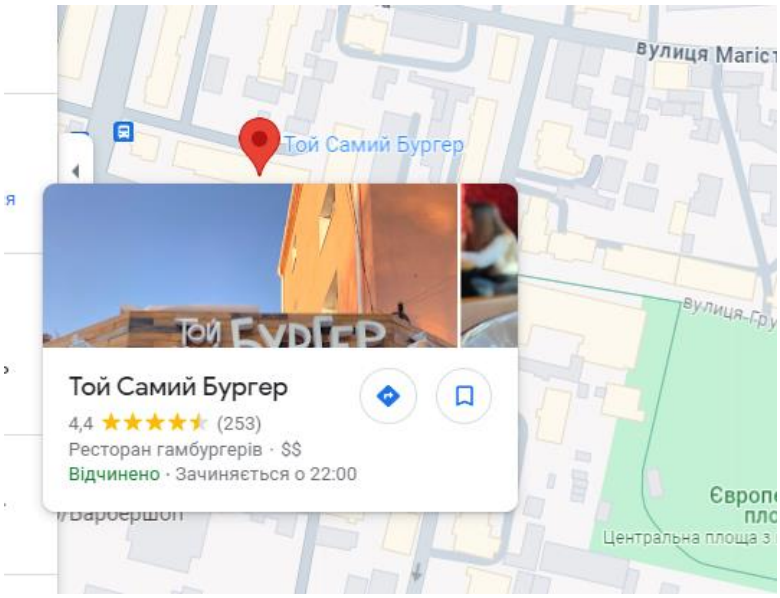

Рис. 3.19 Перехід на мітку закладу на google карті

Насамкінець у користувача є можливість залишити коментар про заклад, який він відвідав, та поставити оцінку, яка буде враховуватись в загальний рейтинг і відображатись на картці закладу на основній сторінці.

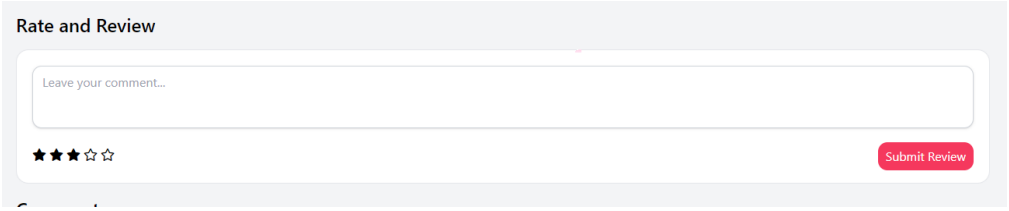

Рис. 3.20 Компонент для коментування

ВИСНОВКИ ДО РОЗДІЛУ 3

У ході написання третього розділу було обрано клієнт-серверну архітектуру та патерн проєктування MVC для розроблюваної платформи-агрегатора для ресторанного бізнесу.

Перераховано ключові моменти, які безпосередньо вливають на вибір архітектури системи та патерну проєктування.

Спроєктовано та створено архітектуру платформи-агрегатора відповідно з визначеними технологіями і визначеними задачами для досягнення цілей роботи.

Сформовано структуру користувацького інтерфейсу для різних ролей платформи-агрегатора. Окрім цього створено структуру директорій клієнтської частини системи.

Розроблено та представлено всі компоненти системи для різних частин платформи-агрегатора, відповідно до того, відбулася автентифікація та авторизація чи ні та відповідно до того, яку роль має користувач і яким чином може взаємодіяти з інтерфейсом.

Реалізована архітектура платформи-агрегатора для ресторанного бізнесу дозволяє створювати новий функціонал за будь-яких вимог бізнес-процесів. Також дозволено масштабування системи, яке не потребує втручання до основної структури системи.

РОЗДІЛ 4
РОЗРОБКА СТАРТАП-ПРОЕКТУ

4.1 Інформаційна картка проєкту

У таблиці 4.1 продемонстровано інформаційну картку проєкту.

Таблиця 4.1

Інформаційна картка проєкту

№	Характеристика	Зміст характеристики
1	Назва проєкту	Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу
2	Автори проєкту	Ткаченко Валентина Василівна, Борозенець Денис Романович
3	Коротка анотація	Клієнтське програмне забезпечення для платформи-агрегатора для ресторанного бізнесу представляє собою комплексне рішення, яке полегшує керування замовленнями, спрощує інвентаризацією та покращує взаємодією з клієнтами. Завдяки новітнім технологіям та сучасним інструментам розробки клієнтської частини, а також передбаченої інтеграції з іншими платформами та модулями, ресторани та інші заклади можуть ефективніше керувати власними ресурсами та оптимізувати процеси обслуговування, що в свою чергу призведе до підвищення задоволеності клієнтів.
4	Термін реалізації	Шість кварталів (18 місяців)
5	Необхідні фінансові ресурси	Зарплати співробітникам (щомісячно) протягом етапу розробки: - Розробник (4 особи) = 3000$ кожен; - Бізнес-аналітик = 2.500$; - Проєкт-менеджер = 2.000$; - Тестувальник (2 особи) = 2.700$; Всього щомісячно = 21.900$; Інвестиції в обладнання та програмне забезпечення = 18.000$; Ліцензії = $500; Маркетинг = 10.000$. Зберігання даних, хмарні сервіси = $500 щомісяця (після запуску проєкту).

№	Характеристика	Зміст характеристики
6	Необхідні матеріальні ресурси	- Комп'ютери з високою обчислювальною потужністю для розробників, тестувальників та бізнес-аналітика для розробки та тестування алгоритмів. - Звичайні комп’ютери для проект-менеджера та маркетолога
7	Необхідні людино-ресурси	◦ 4 Розробники (два, які спеціалізуються на розробці користувацького програмного забезпечення, інші - на розробці серверної частини та роботи з базою даних); • 1 Бізнес-аналітик; • 1 Проєкт-менеджер; • 2 Тестувальники (один спеціалізується на тестуванні користувацького програмного забезпечення, інший - на серверній частині та базі даних).
8	Опис проблеми, яку вирішує проєкт	Проєкт вирішує проблеми керування замовленнями, обліком інвентаризації, оптимізації позицій в меню та взаємодією з клієнтами за рахунок можливості коментувати свої враження та залишати відгук з оцінкою кожному закладу. Він оптимізує робочі процеси, підвищує ефективність обслуговування, забезпечує інтеграцію з різними платформами та покращує загальну задоволеність клієнтів.
9	Головні цілі та завдання	Головною метою проєкту є створення клієнтського програмного забезпечення для платформи-агрегатора, що ефективно керує ресторанним бізнесом та оптимізує робочі процеси, що в свою чергу підвищує задоволеність клієнтів. 1. Розробка достатньо надійної платформи для керування замовленнями, інвентаризацією та швидкої взаємодії з клієнтами. 2. Інтеграція платформи з різними модулями, системами та сервісами для забезпечення швидкої та ефективної взаємодії і обміну даними. 3. Тестування системи для оцінки її ефективності та надійності. 4. Аналіз отриманих даних та забезпечення зворотного зв'язку з користувачами для виявлення можливості наступних покращень та оптимізації функціоналу системи зокрема інтерфейсу користувача.

Продовження таблиці 4.1

№	Характеристика	Зміст характеристики
10	Очікувані результати	1. Перша версія ПЗ завершена: розроблено та протестовано клієнтське програмне забезпечення, що вміщує в собі основні функції для взаємодії клієнта з додатком. 2. Ефективне керування ресурсами: ПЗ дозволяє ресторанам оптимізувати керування замовленнями, та, відповідно, взаємодією з клієнтами, що в свою чергу підвищує їхню продуктивність. 3. Підвищення рівня задоволеності клієнта: вдосконалення якості надання послуг та взаємодії з клієнтами веде до позитивної реакції кожного клієнта на заклад, що призводить до росту лояльності та задоволення клієнтів. 4. Зворотний зв'язок та обробка отриманих даних: збір, обробка та аналіз отриманих даних від користувачів для того щоб ефективно та швидко виявити потреби користувачів для можливості майбутнього вдосконалення продукту. 5. Масштабування: розробка ПЗ з використанням таких технологій, що дають можливість легкого масштабування розроблюваного додатку та його легкої адаптації під різноманітні параметри будь-якого типу ресторану. 6. Збільшення рівня конкурентоспроможності закладу: підвищення рівня конкурентоспроможності будь-якого закладу завдяки ефективному керуванню та сучасним інструментам і технологіям. 7. Формування та використання маркетингових стратегій: розробка стратегій для можливості рекламування ПЗ серед потенційних майбутніх користувачів.

4.2 Склад команди стартап-проєкту

У таблиці 4.2 наведено склад команди розробки.

Таблиця 4.2

Склад команди розробки

№	Роль	Спеціальність	Кількість
1	Розробник	Розробник серверної частини	2 люд.
2	Розробник	Розробка користувацького інтерфейсу	2 люд.
№	Роль	Спеціальність	Кількість
3	Бізнес -аналітик	Бізнес-аналітик	1 люд.
4	Менеджер	Project Manager	1 люд.
5	Тестувальник	Manual	2 люд.
Разом			8 людей

Команда для розробки та реалізації стартап-проєкту має формуватися з ролей, оглянутих далі:

- Розробник клієнтської частини або користувацького інтерфейсу(UI)

  Головними завданням даного розробника є створення достатньо естетичного та комфортного для сприйняття дизайну, який має відповідати вимогам функціональності та повинен відображати бренд або стиль програмного продукту. Розробка ефективної системи навігації, щоб користувач міг швидко та легко знаходити потрібні функції та інформацію. Забезпечення комфорту та легкості взаємодії користувача з додатком, обов’язково враховуючи введення даних, взаємодію з різними кнопками та елементами сторінки.

- Розробник серверної частини

  Головними завданням даного розробника є розробка та реалізація логіки, яка регулює взаємодію програми з даними та виконує бізнес-логіку додатку. Створення серверного ПЗ для обробки запитів, що йдуть безпосередньо від клієнтської частини. Забезпечення безпеки даних та взаємодії між клієнтом та сервером, разом з тим включаючи захист від атак та неналежного використання. Розробка коду для того щоб взаємодіяти з базою даних(БД), забезпечення ефективного та швидкого зберігання і отримання даних.

- Бізнес-аналітик

Головними завданням бізнес-аналітика є взаємодія із зацікавленими сторонами, збір та аналіз бізнес-вимог для визначення необхідних вимог для досягнення цілей бізнесу. Створення бізнес-моделі, що відображає ключові елементи бізнес-процесів. Використання аналітичних методів для аналізу даних та вивчення трендів, що можуть вплинути на бізнес.

- Project Manager(Менеджер проєкту)

  Головними завданням менеджер проєкту є розробка детального плану проєкту, включаючи формування завдань, ресурсів, термінів та розподіл обов'язків між учасниками команди. Відбір, навчання та управління командою проєкту та розподіл завдань. Забезпечення ефективного та вірного використання бюджету та ресурсів, що були визначені для проєкту. Постійний моніторинг прогресу протягом проєкту та звітність, включаючи виявлення ризиків, що можуть з’явитись та прийняття вчасних коригувань.

- Manual Tester(Тестувальник)

  Головними завданням тестувальника є розробка покрокових детальних сценаріїв тестування для покриття різноманітних аспектів ПП. Запуск тест-кейсів і ручне виконання функціональних, інтеграційних та інших різновидів тестів. Запис помилок та інших непрацюючих моментів, які виявлені під час тестування, та їх детальний опис.

Таблиця 4.3

Важливі фактори проєкту

№	Фактор	Вага важливості (1-10 балів)
1	Створення та просування ідеї	7
2	Формування бізнес-плану	9
3	Компетентність співробітників	9
4	Виконання обов’язків	10
5	Співпраця та комунікація	8

Завершальним етапом для оцінювання діяльності співробітників є аналіз індивідуального внеску кожного з учасників команди проєкту. Всі зібрані та проаналізовані дані зображено в таблиці, що оцінює вклад кожного учасника команди. За рахунок використання цих даних, маємо змогу побудувати таблицю для візуального представлення вкладу кожного учасника проєкту. Дана таблиця сформує чітке бачення того, скільки кожен учасник команди виконав завдань і який загальний вплив мав на проєкт.

Таблиця 4.4

Оцінка вкладу кожного учасника команди

Фактор	Розробка користувацького інтерфейсу	Розробка серверного програмного забезпечення	Бізнес-аналітик	Project Manager	Тестувальник	Разом
Створення та продвигання ідеї	35	30	10	70	15	
Формування бізнес-плану	20	25	30	90	15	
Компетентність співробітників	80	80	70	95	60	
Виконання обов’язків	100	100	90	100	80	
Співпраця та комунікація	60	70	65	85	50	
Разом	295	305	265	440	220	
Відсоток	20.4%	21.1%	18.3%	30.5%	15.2%	100 %

Проаналізувавши результати, можна зробити висновок, що хоч кожен член команди і є дуже важливим, але основний вклад все ж таки лежить на розробниках та менеджері проєкту.

4.3 Формулювання основної ідеї проєкту

Основною ідеєю проєкту є створення новітнього ПЗ для платформи-агрегатора, спеціально пристосованого до потреб ресторанного бізнесу. Проєкт має на меті докорінно змінити спосіб управління повсякденною діяльністю ресторанів, від виконання замовлень до управління запасами продуктів та взаємодії з клієнтами. Основна ідея полягає у створенні зрозумілого, і в одночас дуже потужного інструменту, який дозволить ресторанному бізнесу значно підвищити ефективність, і разом з цим зменшити зайві витрати та підвищити загальний рівень задоволеності клієнтів.

ПЗ даного проєкту має на меті інтегрувати найбільш сучасні інструменти та технології, для того щоб було легше всього оптимізувати різні необхідні аспекти ведення ресторанного бізнесу. Прекрасним прикладом є можливість збору та аналізу інформації, в першу чергу, отриманої з замовлень. Не є секретом, що в різні періоди або пори року вподобання клієнтів змінюються, взимку хочеться випити гарячої та запашної кави, а влітку відповідно чогось прохолодного та освіжаючого. Аналіз подібної інформації дозволить закладам передбачити тенденції попиту, і відповідно до цього буде виконуватись оптимізація використання ресурсів, більша закупівля необхідних продуктів, і менша вже не актуальних на той момент. Окрім цього наявний функціонал для спілкування з клієнтами дозволить краще розуміти побажання цих самих клієнтів і вчасно реагувати на проблеми, від яких ніхто не застрахований.

Ще однією особливістю додатку є розробка з врахуванням всіх вимог для гнучкості та масштабованості. Даний функціонал необхідний для більшої універсальності. Тобто дане ПЗ повинно ефективно використовуватись закладами різних розмірів і типів, від маленької кав’ярні до великого французького ресторану. Також передбачено розробку легкого та інтуїтивно зрозумілого користувацького інтерфейсу. Доволі важливий аспект будь-якого додатку, адже це мінімізує ресурси та час, які необхідно виділити

для того щоб навчити новий персонал користуватись системою, а, отже, новий персонал швидше приступить до виконання своїх обов’язків.

Однією з найважливіших частин при розробці будь-якого проєкту, є забезпечення достатньо високої планки у забезпеченні безпеки даних користувачів. Система включатиме новітні інструменти захисту конфіденційної інформації.

Реалізація даного програмного продукту впливатиме на різні частини ресторанного бізнесу. Ефективність роботи закладів зросте, це в свою чергу збільшить рівень задоволеності відвідувачів, а даний аспект безпосередньо впливає на бажання клієнта ще раз відвідати заклад. Саме тому це матиме значний вплив на ринок ресторанного бізнесу.

Але варто розуміти, що немає ідеальних проєктів без вад та недоліків. Саме тому було сформовано таблицю, що зображує позитивні та негативні аспекти розроблюваного додатку.

Висновки про переваги та недоліки винесені у таблицю 4.5.

Таблиця 4.5

Порівняльна таблиця переваг та недоліків проєкту

№	Переваги	Недоліки
1	Автоматизація керування рестораном	Висока вартість розробки
2	Оптимізація інвентаризації	Потреба в спеціалізованому персоналі
3	Підвищення рівня задоволеності клієнтів	Складність інтеграції з існуючими системами
4	Інтеграція новітніх технологій	
5	Підвищення конкурентоспроможності персоналу	
6	Підвищення продуктивності персоналу	
7	Покращення управління ризиками	
Сума	5	3

Для того щоб розробка та реалізація проєкту мали успіх, необхідно провести детальний аналіз, що повинен розкрити не тільки переваги ПП, але й

недоліки. В першу чергу це має стосуватись технічної частини та економічної, адже це безпосередньо впливає на кількість ресурсів, які потрібно буде витратити. Особливо важливо знаходити саме недоліки і слабкі місця розроблюваного додатку, та, окрім того, що їх потрібно знаходити, необхідно вчасно розробляти шляхи вирішення тих чи інших проблем, особливо, якщо дана проблема буде напряму впливати на результативність додатку.

Таблиця 4.6

Аналіз характеристик з урахуванням аналогів

№	Характеристика	Власна розробка	Аналоги			W	N	S
			1	2	3			
1	Новітні функції	+						+
2	Комфорт для клієнтів	+			+		+	
3	Маркетинг	+	+	+	+		+	
4	Аналітика	+			+		+	
5	Програми лояльності							+
6	Комісії та витрати		+	+		+		
7	Залежність від платформи				+		+	

Даний проєкт характеризується низкою новітніх функцій, що власне випереджають конкурентів та приваблюють клієнтів. Дана платформа вирізняється не лише тим, що надає можливість легкого доступу до різних закладів, але й має привабливий та інтуїтивно зрозумілий інтерфейс для користувачів.

Окрім переваг, існують і недоліки, з якими зіштовхується будь-який проєкт. Комісії та доволі високі витрати можуть бути сильною перешкодою, в першу чергу для середнього та малого бізнесів. Додатково присутній ризик того, що заклад буде залежати від платформи. Це в свою чергу може призвести до втрати впливу на різні аспекти ресторанного бізнесу, основними і одними з найбільш важливими є керування клієнтською базою та

ціноутворення. Подібні проблеми повинні бути детально проаналізовані, для того щоб їх вплив на проєкт можна було мінімізувати.

4.4 Аналіз стеку технологій розроблюваного проєкту

Розроблюваний проєкт платформи-агрегатора ресторанів використовує MERN стек. До нього входять наступні технології: MongoDB, Express.js, React, Node.js. Аналіз стеку технологій має на меті вивчити чи сумісні дані технології. Окрім цього, потрібно проаналізувати переваги та, якщо такі будуть, недоліки використання цих технологій, в першу чергу чи вони буду ефективними інструментами для розробки програмного продукту, та чи забезпечать подальше комфортне масштабування проєкту, якщо стане така необхідність.

MongoDB:

- Гнучкість та масштабованість: MongoDB - це NoSQL (Not Only SQL) база даних, що дозволяє зберігати дані у форматі документів, подібна система збереження даних надає гнучкість та легкість масштабування.
- Швидкість розробки: схема бази даних може легко змінюватися без перерви в роботі системи, що безпосередньо сприяє швидкій розробці та адаптації.

Подібна система збереження даних та гнучкість є доволі важливим фактором для того щоб використовувати саме цю БД для платформи-агрегатора ресторану.

Express.js:

- Легкість та простота: Express.js – це достатньо легкий та мінімалістичний веб-фреймворк, який значно прискорює розробку серверної частини додатка.
- Маршрутизація: Express надає дуже простий та водночас потужний механізм маршрутизації, який спрощує керування маршрутами.

Подібна простота у використанні і легкість робить даний фреймворк чудовим вибором для розробки платформи-агрегатора.

React:

- Компонентний підхід: React базується на компонентах, що значно полегшує побудову інтерфейсу та керування станом додатка.
- Віртуальний DOM: Застосування віртуального DOM забезпечує ефективне виконання оновлення інтерфейсу без перезавантаження всієї сторінки, що зберігає дуже багато ресурсів.

Компонентний підхід дозволяє швидко створювати сторінки та різноманітні користувацькі компоненти. А величезна спільнота забезпечує швидкими відповідями та кодовою базою у разі виникнення проблем. Самі ці речі і роблять React чудовим вибором для розробки стартап-проєкту.

Node.js:

- Асинхронний ввод/вивід: Node.js використовує асинхронний підхід, це дозволяє доволі ефективно обробляти багато одночасних з'єднань, безпосередньо покращує продуктивність.
- Спільна мова програмування: Node.js використовує JavaScript, як мову ядро. А використання єдиної мови для серверної та клієнтської частини додатка облегшує взаємодію та зменшує проблеми сумісності.

React є бібліотекою JavaScript, тобто використовує JavaScript, як мову ядро. Це означає що у React та Node.js чудова сумісність, а відповідно і взаємодія.

В підсумку можна сказати, що головною перевагою MERN стеку є те, що весь стек використовує одну мову програмування (JavaScript/TypeScript) та сприяє швидкій розробці, масштабованості, а також легкості управління та розширенню. Ще однією перевагою є відкритий код кожної з розглянутих технологій, це в свою чергу значно знижує затрати, що виділяються під розробку ПЗ, а це є дуже важливим аспектом для будь-якого стартапу. Технології даного стеку одні з найновіших, і разом з цим набрали значну популярність серед розробників. На сьогоднішній день існує величезна кількість ресурсів, пов’язаних з даними технологіями, а також активна та

постійно зростаюча спільнота, що надає чудову підтримку, а це є безперечно дуже позитивним фактором для того щоб використовувати даний стек для стартапу.

Окрім позитивних моментів, завжди існують і негативні. Хоча MERN стек є дуже популярним та ефективним для розробки веб-додатків, у кожної технології в цьому стеку є свої проблеми та обмеження.

MongoDB

– Недостатня підтримка для складних транзакцій: MongoDB, як будь-яка NoSQL БД, не надає повноцінної підтримки для складних транзакцій, що може бути важливим для деяких видів додатків.

Express.js

– Брак вбудованих функцій для певних завдань: Express.js є мінімалістичним фреймворком, тому для деяких завдань може бути потрібно використання додаткових бібліотек.

React

– SEO-проблеми(Search Engine Optimization): сторінки можуть завантажуватися динамічно за рахунок JavaScript, що може призводити до проблем із індексацією пошуковими системами.

Node.js

– Синхронність та однопотоковість: в асинхронному середовищі даної технології, використання блокуючих операцій може призводити до затримок, і лише один потік вірогідніше всього може стати буттям для деяких значних застосунків.

Отже, розглянемо основні технології, що будуть застосовуватись у проєкті та занесемо їх до таблиці 4.7.

Таблиця 4.7

Аналіз технологічного стеку MERN

№	Технологія	Версія	Сфера використання	Доступність
1	MongoDB	6.0	БД NoSQL, веб-додатки, біг дата, обробка великих обсягів даних	Вільно доступна (Open Source)

№	Технологія	Версія	Сфера використання	Доступність
2	Express.js	4.18.2	Розробка веб-додатків, API, серверна технологія для веб-додатків	Вільно доступна (Open Source)
3	React.js	18.2.0	Розробка веб-додатків, користувацьких інтерфейсів, мобільні додатки	Вільно доступна (Open Source)
4	Node.js	16.16.0	Розробка серверної частини додатків, мікросервіси	Вільно доступна (Open Source)

4.5 Оцінка потенційних можливостей ринку з метою впровадження нового стартап-проєкту

Оцінка потенційних можливостей ринку потребує детального аналізу та вивчення багатьох аспектів. Основними є унікальність та спроможність конкурувати з іншими подібними системами на ринку. Подібна оцінка виконана та представлена в таблицях. Результати аналізу представлені у таблицях 4.8, 4.9 та 4.10.

Таблиця 4.8

Дослідження аспектів ринку для з'ясування його характеристик і особливостей

№	Характеристика	Значення
1	Кількість аналогів на ринку	3 компанії
2	Обсяг запланованої реклами	300 000 щомісячно (1.70$ за 1000 показів = 510$)
3	Якісна оцінка динаміки ринку	Продовження розвитку онлайн-платформ для пошуку та бронювання закладів. Споживачі все більше розраховують на цифрові засоби для пошуку ресторанів, читання відгуків та бронювання.
4	Обмеження для входу на ринок	Середній рівень: необхідні інвестиції в технології та розробку
5	Вимоги та стандарти	- Заохочення аудиторії - Актуальність даних - Наявність адаптивного та інтуїтивного користувацького інтерфейсу - Забезпечення конфеденційності та безпеки даних

№	Характеристика	Значення
6	Середня рентабельність в галузі	50-70% річних, враховуючи стратегію монетизації

Стартап-проєкт повинен залучити широке коло людей, що охоплюють різні демографічні групи. Серед них - любителі їжі, працюючі люди, сім'ї, люди, які піклуються про власне здоров'я, мандрівники та туристи, корпоративні клієнти та організатори заходів.

Таблиця 4.9

Аналіз загроз ринку

№	Фактор	Опис загрози	Відповідна реакція компанії
1	Конкуренція	Висока конкуренція відносно платформ, що є вже встановленими у користувачів може утруднити залучення нових користувачів	Розробка унікальних пропозиції, програм лояльності, фокус на нішевих ринках, інновації в технологіях та нововведення в маркетингу
2	Зміна споживчих трендів	Динамічна і постійна зміна вподобань споживачів може зробити послуги платформи неактуальними	Гнучкість та швидкість адаптації до нових вподобань, збір та аналіз даних користувачів
3	Проблеми з безпекою даних	Ризики, що пов'язані з конфіденційністю, збереженням та захистом персональних даних користувачів	Впровадження сучасних стандартів кібербезпеки, постійні оновлення безпеки, прозора політика конфіденційності
4	Залежність від відгуків	Негативні відгуки та оцінки можуть мати великий вплив на репутацію платформи, а це впливатиме на бажання користувачів використовувати платформу	Активне керування репутацією, система відгуків з можливістю відповіді від ресторанів, моніторинг відгуків

№	Фактор	Опис загрози	Відповідна реакція компанії
5	Законодавчі зміни	Зміни в законодавстві можуть потребувати додаткових витрат або корегування діяльності	Гнучкість у веденні бізнесу, безперервний юридичний моніторинг, пристосування до нових норм

Таблиця 4.10

Аналіз можливостей ринку

№	Фактор	Опис можливості	Відповідна реакція компанії
1	Розвиток мобільних технологій	Зростання використання мобільних пристроїв створює можливості для мобільного маркетингу	Розробка зручного мобільного додатку, оптимізація під використання на мобільному пристрої
2	Потреба онлайн-бронювання	Зростаючий інтерес до онлайн-бронювання столиків та замовлень їжі	Впровадження легкого інтерфейсу для бронювання, інтеграція з системами оплати
3	Локалізація	Популярність місцевих закладів, ресторанів та кафе серед місцевого населення та туристів	Надання інформації про місцеві заклади, та традиційні особливості їжі
4	Індивідуальні переваги	Зростаюча потреба в здоровій, вегетаріанській або спеціалізованій їжі	Фільтри та рекомендації за типом харчування, застосування дієтичних альтернатив
5	Соціальні медіа	Використання соціальних мереж та медіа для реклами, відгуків та просування	Активність у соціальних мережах, маркетинг

Даний покроковий аналіз надає реальну оцінку можливостей розроблюваного проєкту. Він точно визначає і ризики додатку, і можливості. Окрім цього забезпечує активне реагування на ринкові коливання та допомагає визначити стратегію подальшого розвитку відносно зовнішніх чинників.

Аналіз ринку проводиться для того, щоб надати детальну картину ринкової позиції та конкурентоспроможності компаній, що розвиваються. Результати аналізу допоможуть сформулювати чітку стратегію розвитку.

Аналіз сегментів ринку представлений у таблиці 4.11.

Аналіз конкурентних переваг зображений у таблиці 4.12.

Таблиця 4.11

Сегментний аналіз ринку

Характеристика конкурентного середовища	Сфера впливу характеристики	Дії компанії для забезпечення конкурентоспроможності
Великі гравці	Перевага бренду, що є загальновизнаним та ресурсах	Розробка унікальної пропозиції, звернення уваги на інновації та якісне обслуговування
Цінова конкуренція	Рівень цін, доступність для різних шарів ринку	Гнучкі цінові стратегії, програми лояльності, акції та знижки
Різноманітність послуг	Вибір ресторанів, варіативність харчування	Надання варіативності, фільтрів за типом кухні, стилем закладів
Технологічні інновації	Легкість використання, персоналізація, ефективність	Впровадження AI для персоналізації, мобільний додаток, інтеграція з соцмережами
Користувацький досвід	Інтуїтивність інтерфейсу, підтримка користувачів	Вдосконалення дизайну користувацького інтерфейсу, ефективна служба підтримки

Таблиця 4.12

Оцінка переваг у порівнянні з конкурентами

№	Конкурентна перевага	Бали (1-20)	Оцінка ефективності
1	Інноваційний інтерфейс користувача	18	Високий рівень комфорту та залучення користувачів
2	Широка база даних ресторанів	17	Ефективне реагування на забезпечення різноманітних вподобань користувачів

№	Конкурентна перевага	Бали (1-20)	Оцінка ефективності
3	Персоналізовані рекомендації	15	Покращення користувацького досвіду через AI та аналітику даних.
4	Інтеграція з соціальними мережами		Підвищення пізнаваності бренду та залучення нових користувачів
5	Сильна програма лояльності		Залучення та утримання користувачів

ВИСНОВОК ДО РОЗДІЛУ 4

У ході написання четвертого розділу було виконано покроковий аналіз основних аспектів стртап-проєкту. Розробка платформи-агрегатора для ресторанного бізнесу вміщує в собі різноманітні підступи та критерії, охоплюючи при цьому стратегічне планування, технічне підґрунтя та розуміння особливостей ринку.

Було пропрацьовано та проаналізовано конкурентну спроможність розроблюваного стартап-проєкту, разом з існуючими на сьогоднішній день існуючими рішеннями ринку, та їх змогою задовільнити вимоги користувачів. Вибором технологічного стеку став MERN за рахунок своєї гнучкості, легкої масштабованості та широкої технічної спільноти. Також було оцінено основні аспекти конкурентної спроможності, що дозволило виокремити позитивні та негативні сторони стартап-проєкту.

Ґрунтовне розуміння потенціальних потреб користувачів та особливостей ринку є головним фактором при розробці стартап-проєкту.

Реалізовано модель виробничого плану, що застосовується для детальної організації всіх кроків реалізації проєкту, починаючи з початкових дослідницьких етапів та закінчуючи етапом впровадження кінцевого продукту на ринок.

ВИСНОВКИ ТА РЕЗУЛЬТАТИ

У ході написання магістерської дисертації було проведено всебічне та структуроване дослідження з розробки платформи-агрегатора для ресторанного бізнесу. Основну увагу було сфокусовано на рішеннях, що стосуються клієнтської частини. Дисертація поєднує велику кількість аспектів, починаючи з теоретичного аналізу новітніх технологій та інструментів, закінчуючи формуванням архітектури та реалізацією власне платформи-агрегатора.

У першому розділі було проаналізовано та описано актуальність та доцільність розробки платформи-агрегатора. Окрім цього було проведено огляд вже існуючих на сьогоднішніх день систем. Детально описано можливості при використанні кожної з них. Сформовано та виокремлено переваги та недоліки користування існуючими рішеннями.

У другому розділі було описано життєвий цикл розробки електронної системи. Оглянуто та проаналізовано сучасні інструменти та технології для розробки клієнтської частини застосунку. Виконано порівняльний аналіз, та відповідно вибір найбільш вдалих для розроблюваної платформи-агрегатора технологій, які будуть забезпечувати ефективну дієспроможність.

У третьому розділі було сформовано та реалізовано архітектурні рішення для розроблюваної системи. Обрано патер проєктування системи. Розроблено клієнтську частину платформи-агрегатора, в основі якої знаходяться новітні програмні засоби та рішення. Описано можливості та функціонал реалізованої платформи-агрегатора.

У четвретому розділі покроково розроблено та проаналізовано головні аспекти стартап-проєкту, включно з ринковою стратегією, конкурентною спроможністю, планом реалізації, що в свою чергу створює сильне підґрунтя для того щоб впровадження системи на ринку мало успіх.

Підсумовуючи, дана магістерська дисертація представляє чималий потенціал розробленої системи, як новітнього рішення для ринку

ресторанного бізнесу, а також комплексний та інноваційний підхід розробки клієнтських рішень. Результатом є гнучке, масштабоване клієнтське рішення, що великою мірою збільшує якість та ефективність взаємодії з користувачами платформ-агрегаторів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Yelp. [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Yelp

2. Yelp System Design. [Електронний ресурс] – Режим доступу до ресурсу: https://www.systemdesigntutorial.com/blog/yelp

3. Yelp Pros & Cons - What You Need to Know About Yelp for Business. [Електронний ресурс] – Режим доступу до ресурсу: https://perfectlyoptimized.com/yelp-pros-cons/

4. OpenTable. [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/OpenTable

5. How OpenTable works for restaurants. [Електронний ресурс] – Режим доступу до ресурсу: https://www.opentable.com/blog/how-opentable-works-for-restaurants/

6. OpenTable. [Електронний ресурс] – Режим доступу до ресурсу: https://www.trustradius.com/products/opentable/reviews?qs=pros-and-cons#overview

7. The Pros and Cons of OpenTable. [Електронний ресурс] – Режим доступу до ресурсу: https://www.fsrmagazine.com/marketing-promotions/pros-and-cons-opentable

8. OpenTable Review 2022 (ratings, features & more). [Електронний ресурс] – Режим доступу до ресурсу: https://www.perfectvenue.com/post/opentable-review

9. The Pros and Cons of OpenTable. [Електронний ресурс] – Режим доступу до ресурсу: https://www.positivelyindy.com/the-pros-and-cons-of-opentable/

10. What is Zomato & how does it work? [Електронний ресурс] – Режим доступу до ресурсу: https://www.appsrhino.com/blogs/what-is-zomato-and-how-does-it-work-everything-you-need-to-know

11. Zomato. [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Zomato

12. Management Information Sytem. [Електронний ресурс] – Режим доступу до ресурсу: https://www.scribd.com/document/530187611/MANAGEMENT-INFORMATION-SYTEM

13. 17 Astounding Facts About Zomato. [Електронний ресурс] – Режим доступу до ресурсу: https://facts.net/science/technology/17-astounding-facts-about-zomato/

14. What is Zomato and Why Should Your Restaurant Use It? [Електронний ресурс] – Режим доступу до ресурсу: https://wpbusinessreviews.com/what-is-zomato-and-why-should-your-restaurant-use-it/

15. Tripadvisor. [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Tripadvisor

16. Investor FAQs. [Електронний ресурс] – Режим доступу до ресурсу: https://ir.tripadvisor.com/investor-faqs

17. How the site works. [Електронний ресурс] – Режим доступу до ресурсу: https://www.tripadvisor.com/pages/service_en.html

18. TripAdvisor Review. [Електронний ресурс] – Режим доступу до ресурсу: https://techboomers.com/t/tripadvisor-review

19. The Pros and Cons of TripAdvisor. [Електронний ресурс] – Режим доступу до ресурсу: https://www.sodhatravel.com/blog/2013/08/the-pros-and-cons-of-tripadvisor

20. TRIPADVISOR FOR MARKETING VS YOUR OWN USER REVIEW SYSTEM. [Електронний ресурс] – Режим доступу до ресурсу: https://www.phocuswire.com/TripAdvisor-for-marketing-VS-your-own-user-review-system

21. Must-Have Features Of a Food Ordering and Delivery System. [Електронний ресурс] – Режим доступу до ресурсу: https://networkon.io/resources/blog/must-have-features-of-a-food-ordering-system/

22. Food Aggregator or Own Platform: Which Should Restaurants Choose for Online Ordering. [Електронний ресурс] – Режим доступу до ресурсу: https://www.heyinnovations.com/post/food-aggregators

23. What Is SDLC? Understand the Software Development Life Cycle. [Електронний ресурс] – Режим доступу до ресурсу: https://stackify.com/what-is-sdlc/

24. SDLC – Overview. [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/sdlc/sdlc_overview.htm

25. Figma(software). [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Figma_(software)

26. Figma Software Review | Advantages and Disadvantages [Електронний ресурс] – Режим доступу до ресурсу: https://blog.talentgarden.com/en/blog/design/figma-software-review-advantages-and-disadvantages

27. Adobe XD vs. Sketch: Pros + Cons. [Електронний ресурс] – Режим доступу до ресурсу: https://designshack.net/articles/software/adobe-xd-vs-sketch/

28. What is Photoshop. [Електронний ресурс] – Режим доступу до ресурсу: https://www.agitraining.com/adobe/photoshop/classes/what-is-photoshop

29. Advantages and Disadvantages of Adobe Photoshop. [Електронний ресурс] – Режим доступу до ресурсу: https://thetechhacker.com/2020/12/12/advantages-and-disadvantages-of-photoshop/

30. 2021 Guide to InVision: Cost, Pros & Cons. [Електронний ресурс] – Режим доступу до ресурсу: https://www.nickelled.com/blog/invision-review/

31. Advantages and Disadvantages of HTML. [Електронний ресурс] – Режим доступу до ресурсу: https://ellow.io/advantages-and-disadvantages-of-html/

32. Advantages and Disadvantages of CSS [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-css/

33. Advantages And Disadvantages Of JavaScript That You Need To Know! [Електронний ресурс] – Режим доступу до ресурсу: https://unstop.com/blog/advantages-and-disadvantages-of-javascript

34. Angular Development: Pros and Cons. [Електронний ресурс] – Режим доступу до ресурсу: https://exoft.net/angular-pros-and-cons/

35. What is Vue js? The Pros and Cons of Vue.JS [Електронний ресурс] – Режим доступу до ресурсу: https://www.tatvasoft.com/outsourcing/2021/10/what-is-vue-js-and-its-benefits.html

36. The Pros and Cons of Using React Today. [Електронний ресурс] – Режим доступу до ресурсу: https://thenewstack.io/the-pros-and-cons-of-using-react-today/

37. What is Client Server Architecture? [Електронний ресурс] – Режим доступу до ресурсу: https://intellipaat.com/blog/what-is-client-server-architecture/

38. MVC Design Pattern [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/mvc-design-pattern/