Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем та програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Бакалавр
(назва освітнього ступеня)

на тему: Розробка веб-застосунку персоналізованих програм лояльності для кафе з використанням технології: Next.js

Виконав(ла): студент(ка) 4 курсу, групи СП-41
спеціальності 121 «Інженерія програмного забезпечення»
(шифр і назва спеціальності)

	(підпис)	Вигонний О.А (прізвище та ініціали)
Керівник	(підпис)	Мудрик І.Я (прізвище та ініціали)
Нормоконтроль	(підпис)	Стоянов Ю.М (прізвище та ініціали)
Завідувач кафедри	(підпис)	Петрик М.Р (прізвище та ініціали)
Рецензент	(підпис)	Марценко С.В (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Петрик М.Р
(підпис) (прізвище та ініціали)
« » 2025 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Вигонному Олександру Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка веб-застосунку персоналізованих програм лояльності для кафе з використанням технології: Next.js

Керівник роботи Мудрик Іван Ярославович, доктор філософії, старший викладач кафедри П.І
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «___» _____ 20___ року № _____

2. Термін подання студентом завершеної роботи _____

3. Вихідні дані до роботи Технічна документація для Next.js, React, Vercel, JavaScript

4. Зміст роботи (перелік питань, які потрібно розробити)
Дослідження існуючих рішень на ринку, їхня порівняльна характеристика, мотивація вибору напрямку розробки, обгрунтування вибраної методології, визначення функціональних та не-Функціональних вимог, моделювання взаємодії між користувачем та системою, проєктування архітектури, структурне планування системи, розробка серверної та клієнтської частини, тестування системи, значення адаптації в трудовому процесі, шляхи збереження працездатності та підвищення продуктивності праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Діаграма варіантів використання, діаграма класів.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності,	Лазарюк В.В., ктн., доц., доц. каф. МТ		
основи охорони праці			

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Процес вибору та затвердження теми		
2.	Аналіз предметної області		
3.	Дослідження існуючих рішень на ринку		
4.	Розробка архітектури, її структурне планування,		
	проєктування головних класів системи.		
5.	Розробка серверної частини		
6.	Розробка клієнтської частини		
7.	Проведення тестування системи		
8.	Безпека життєдіяльності, основи охорони праці		
9.	Оформлення презентації та доповіді до неї		
10.	Перевірка роботи на антиплагіат		
11.	Нормоконтроль		
12.	Попередній захист кваліфікаційної роботи		
13.	Захист кваліфікаційної роботи		

Студент _____ *Вигонний О.А*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Мудрик І.Я*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота бакалавра. Тернопільський національний технічний університет імені Івана Пулюя, каферда програмної інженерії, спеціальність 121 «Інеженерія програмного забезпечення». ТНТУ, 2025, Сторінок 75, таблиць 1, рисунків 45, презентація.

Тема: Розробка веб-застосунку персоналізованих програм лояльності для кафе з використанням технології: Next.js

У цій бакалаврській кваліфікаційній роботі представлено створення веб-застосунку, призначеного для впровадження персоналізованої програми лояльності у закладах типу кафе. Вибір веб-платформи як основи розробки зумовлений її універсальним доступом, зручністю у користуванні, а також перспективами для подальшого масштабування та розвитку. Під час виконання проєкту було вивчено рішення конкурентів, визначено сильні сторони їхніх систем і проаналізовано недоліки, які стали орієнтирами для уникнення типових помилок. На основі отриманих висновків було створено концепцію архітектури сайту та розроблено відповідну базу даних. Для реалізації інтерфейсу було використано фреймворк Next.js, що дало змогу забезпечити високу швидкодію та інтерактивність користувацького досвіду.Ключові слова: веб-сайт, проєктування, архітектура веб-сайту, Next.js

Ключові слова: веб-застосунок, програмна архітектура, системи лояльності, Next.js, переваги використання next.js та mongodb.

Прикладна цінність результатів: розроблена програма лояльності надає інструменти, котрі збільшуватимуть середній чек у кафе та стимулюватимуть клієнтів повертатись. Впровадження програми лояльності сприятиме підвищенню вже існуючих клієнтів, та залучатиме нову постійну аудиторію, чим збільшуватиме заробіток кафе.

ABSTRACT

Bachelor's Qualification Work. Ternopil Ivan Puluj National Technical University, Department of Software Engineering, Specialty 121 "Software Engineering". TNTU, 2025, Pages: 75, Tables: 1, Figures: 45, Presentation.

Title: Development of a Web Application for Personalized Loyalty Programs for Cafes Using Next.js Technology.

This bachelor's qualification work presents the development of a web application designed to implement personalized loyalty programs for cafes. The choice of a web-based platform was motivated by its broad accessibility, ease of use, and scalability potential. During the project, an analysis of existing competitors was conducted to identify the strengths of their solutions and common shortcomings, which helped avoid typical design flaws. Based on these findings, the system architecture and a corresponding database were designed. The user interface was implemented using the Next.js framework, which enabled high responsiveness and improved user experience, along with enhanced performance.

Keywords: web application, software architecture, loyalty systems, Next.js, benefits of using next,js and mongodb.

Practical value of the results: The developed loyalty program provides tools that will increase the average check in the cafe and encourage customers to return. The implementation of the loyalty program will contribute to increasing the retention of existing customers and attracting new regular clientele, thereby increasing the cafe's revenue.

ЗМІСТ

ВСТУП

Останнє десятиріччя докорінно змінило принцип роботи усіх сфер діяльності, сфера громадського харчування не стала винятком. Кафе, кав'ярні та невеликі заклади харчування дедалі активніше впроваджували сучасні інформаційні технології з метою підвищення ефективності бізнесу та покращення взаємодії з клієнтами. В умовах високої конкуренції та швидкої зміни уподобань споживачів, особливого значення набувала здатність закладу не лише залучати нових відвідувачів, а й утримувати постійних. Одним з найкращих та ефективніших методів досягнення цієї мети стали програми лояльності.

Програми лояльності дозволяли налагодити стабільний зв'язок між закладом і споживачем, стимулювати повторні покупки, підвищувати рівень довіри до бренду та створювати додаткову цінність для клієнта. Проте традиційні форми реалізації таких програм — у вигляді пластикових або паперових карток — з часом втрачали свою актуальність. Користувачі чекали на більш зручні, персоналізовані та інтерактивні рішення, інтегровані у цифрове середовище, з яким вони звикли взаємодіяти щодня.

У зв'язку з цим зростала потреба у розробці програм лояльності у вигляді веб-застосунків, які були б доступними на будь-яких пристроях, незалежно від операційної системи, не вимагали встановлення додаткового програмного забезпечення та дозволяли бізнесу швидко адаптуватися до змін. Веб-застосунки надавали змогу не лише автоматизувати рутинні процеси (наприклад, нарахування балів або видачу бонусів), але й відкривали широкі можливості для збору та аналізу даних, необхідних для прийняття ефективних DD рішень.

Особливу увагу у межах даного дослідження було приділено персоналізації — створенню індивідуального досвіду для кожного користувача на основі його історії взаємодії, уподобань та поведінкових даних. Такий підхід дозволяв значно підвищити ефективність програми лояльності, зробивши її не просто бонусною системою, а повноцінним інструментом маркетингової стратегії закладу.

Для реалізації цього підходу доцільним стало використання сучасного фреймворку Next.js, який поєднував гнучкість JavaScript-розробки з перевагами серверного рендерингу, забезпечуючи високу продуктивність, зручну маршрутизацію, швидке завантаження сторінок і добрий користувацький досвід. Цей фреймворк надавав можливість створити веб-застосунок, який легко масштабувався, мав чітку архітектуру і був готовий до впровадження у реальне середовище кафе.

Актуальність даної теми полягала в необхідності розробки ефективного, сучасного та персоналізованого інструменту для підвищення лояльності клієнтів у закладах громадського харчування. Такий інструмент повинен був відповідати очікуванням як бізнесу, так і кінцевого користувача — бути швидким, інтуїтивним, доступним і функціональним.

Таким чином, розробка веб-застосунку для програми лояльності з використанням Next.js розглядалася як актуальне інженерне завдання, що об’єднувало в собі елементи програмної архітектури, веб-інтерфейсів, UX-дизайну, аналізу даних та прикладного програмування.

1 АНАЛІЗ ПРЕДЕТНОЇ ОБЛАСТІ

1.1 Дослідження існуючих рішень на ринку

Серед сучасних рішень для організації програм лояльності особливу увагу привертає сервіс Loyallyst [1].

Ця платформа створена з урахуванням специфіки закладів громадського харчування та пропонує комплексний підхід до побудови клієнтських програм. Вона дозволяє створювати гнучкі бонусні системи, накопичувальні або фіксовані знижки, формувати та керувати базою клієнтів. Завдяки інтеграції з Telegram Loyallyst надає можливість гостям кафе легко переглядати свої бонуси, отримувати індивідуальні пропозиції й бути постійно на зв’язку із закладом. Інтерфейс адміністратора дозволяє відслідковувати статистику відвідуваності та активності користувачів, запускати акції, розсилки та формувати базову аналітику. Водночас сервіс орієнтований на малі та середні підприємства, забезпечуючи їм зручний і доступний інструмент підвищення клієнтської лояльності.

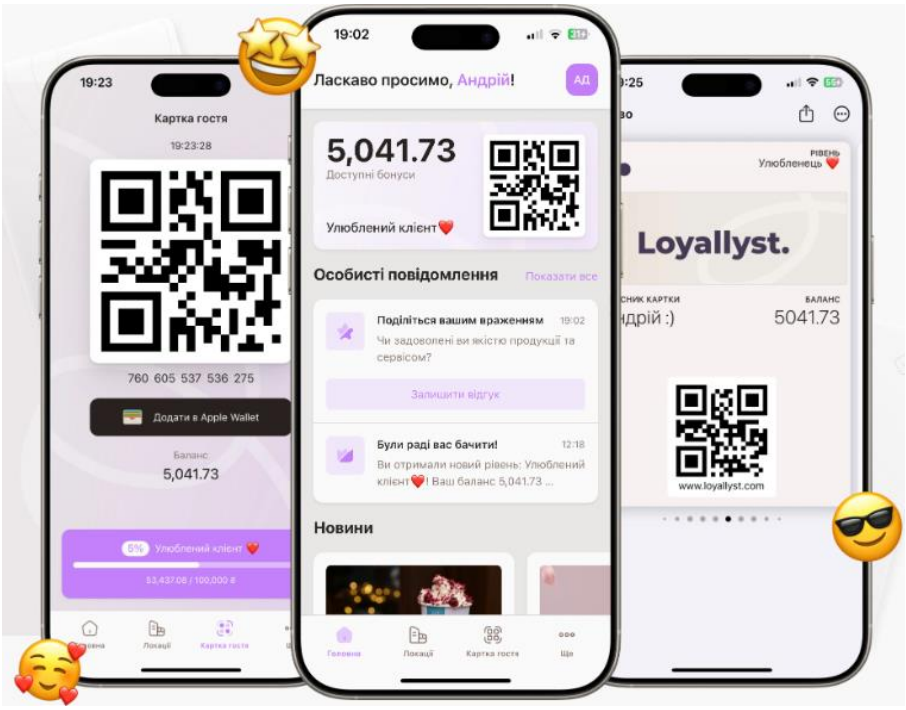

Рис. 1.1 – Вигляд сервісу Loyallyst.

Ще одним популярним рішенням на ринку є міжнародна CRM-платформа Eber, яка орієнтується на побудову довготривалих відносин з клієнтами через багаторівневі програми лояльності. Система надає можливість впроваджувати різні моделі взаємодії: нарахування балів за покупки, видачу цифрових купонів, використання штамп-карт, введення рівнів клієнтів (наприклад, «срібний», «золотий»), а також дарувати привітальні або святкові бонуси. Значну увагу Eber приділяє аналітичній частині — власник закладу має змогу бачити детальну поведінку клієнтів, частоту візитів, середній чек, реакцію на акції тощо. Крім того, платформа підтримує автоматичні email і SMS-розсилки залежно від дій користувача. Важливою перевагою є можливість інтеграції з касовими системами, мобільними додатками та навіть зовнішніми CRM, що дозволяє легко масштабувати рішення під потреби середнього та великого бізнесу [2].

Рис. 1.2 – Вигляд сервісу Eber.

Для малого бізнесу та нових закладів хорошим прикладом доступного рішення є українська система BonusQR.

Цей сервіс дозволяє запускати програму лояльності без складних налаштувань та інтеграцій. Основний принцип роботи базується на використанні QR-кодів, які клієнти можуть сканувати зі стікерів, меню або чеків. Після

сканування користувач може накопичувати бонуси, отримувати миттєві знижки або брати участь в акціях. Для закладу передбачено простий онлайн-кабінет, де можна переглядати статистику використання кодів, налаштовувати акції, проводити розіграші та комунікувати з клієнтами. BonusQR — це зручний стартовий інструмент для тих, хто хоче швидко почати роботу з лояльністю без складних технічних витрат, водночас маючи базові можливості для аналізу клієнтської активності [3].

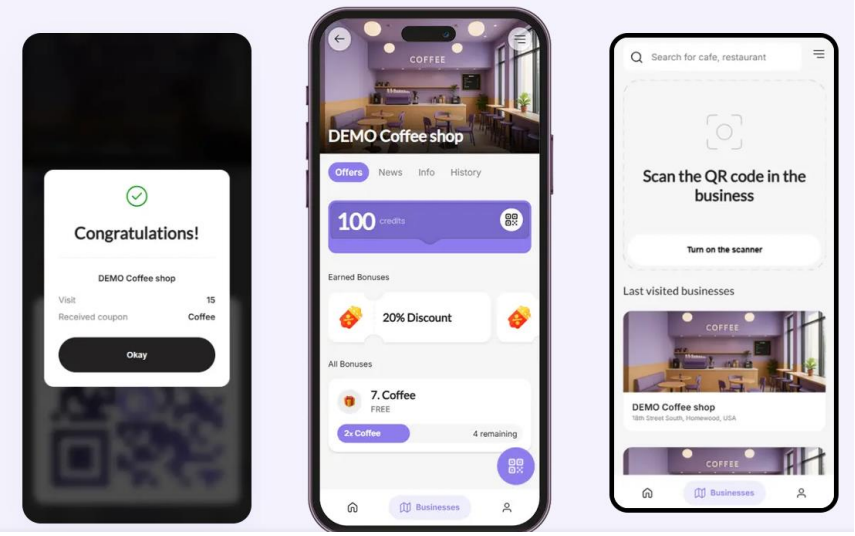

Рис. 1.3 – Вигляд сервісу BonusQR

1.2 Порівняльна характеристика існуючих рішень

Перед створенням ефективного рішення для програми лояльності важливо проаналізувати наявні продукти, які вже працюють на ринку. У цьому розділі ми зосередимось на трьох популярних системах — Loyallyst, Eber та BonusQR. Кожна з них має свої особливості, сильні сторони та певні обмеження, що дає змогу оцінити, які підходи є дієвими, а які можуть стати перешкодою для зручної та результативної взаємодії з клієнтами. Такий аналіз дозволить сформувати чітке бачення щодо функціональності, зручності використання та масштабу

майбутнього рішення, щоб створити продукт, який не просто конкурує з ринком, а навіть перевершує очікування користувачів.

Розпочнемо зі застосунку Loyallyst, а саме з його переваг:

- Розроблений спеціально з урахуванням потреб закладів громадського харчування, що робить інтерфейс та функціонал максимально релевантними для закладів харчування.
- Дозволяє створювати різноманітні програми: накопичувальні бонуси, фіксовані або відсоткові знижки, одноразові акції.
- Має інтеграцію з Telegram, чим значно спрощує комунікацію з клієнтами, дає можливість надсилати персоналізовані пропозиції та дозволяє клієнтам легко переглядати свої бонуси.
- Інтерфейс адміністратора інтуїтивно зрозумілий — можна переглядати статистику, запускати розсилки, формувати аналітичні звіти без додаткового навчання.
- Сервіс адаптований до потреб малого та середнього бізнесу: не потребує великих вкладень і складної інтеграції з іншими системами.

Недоліки:

- Обмежена функціональність у порівнянні з потужнішими міжнародними платформами.
- Відсутність розширених можливостей персоналізації та автоматизованих сценаріїв (наприклад, поведінкових тригерів).
- Не підтримує глибоку інтеграцію з POS-системами та сторонніми CRM, що може бути недоліком при масштабуванні.

Loyallyst — це зручне та ефективне рішення для невеликих кафе чи ресторанів, які прагнуть швидко налагодити програму лояльності без зайвих складнощів. Платформа ідеально підходить для бізнесів, котрі хочуть ефективну та просту програму лояльності [1].

Наступним на черзі – застосунок Eber, розпочнемо з переваг:

- Один із найпотужніших інструментів на ринку CRM-рішень, що дозволяє впроваджувати багаторівневі програми лояльності: бали за покупки, електронні купони, віртуальні штамп-карти, персоналізовані бонуси до свят чи днів народжень.
- Широкі аналітичні можливості: можна відстежувати детальну поведінку клієнтів, аналізувати їхню активність, періодичність покупок, середній чек, ефективність акцій.
- Автоматизовані email і SMS-розсилки з тригерами, що базуються на діях або бездіяльності клієнтів.
- Гнучка інтеграція з POS-системами, мобільними додатками, зовнішніми CRM-платформами та іншими бізнес-системами.
- Підтримка масштабованості — підходить як для локального бізнесу, так і для мережі закладів.

Недоліки:

- Складніша система налаштувань та адміністрування, що вимагає часу на вивчення та налаштування.
- Висока вартість у порівнянні з базовими рішеннями — може бути недоцільною для малого бізнесу.
- Надлишковий функціонал для простих сценаріїв взаємодії з клієнтами.

Eber — ідеальне рішення для середнього та великого бізнесу, який потребує глибокого аналізу клієнтської поведінки, персоналізованих взаємодій і масштабування. Платформа забезпечує високий рівень автоматизації та контролю, проте потребує відповідного ресурсу для впровадження й обслуговування [2].

Останній застосунок для аналізу – BonusQR, розпочинаємо з переваг:

- Дуже простий і доступний інструмент, що не вимагає технічної підготовки або глибоких знань.
- Основний функціонал побудований на використанні QR-кодів — клієнт сканує код зі стікера, меню або чеку, після чого отримує бонуси або бере участь в акціях.

- Онлайн-кабінет дає змогу налаштовувати акції, переглядати базову статистику, проводити розіграші й комунікувати з аудиторією.
- Швидкий запуск — сервіс підходить для нових закладів або тим, хто щойно починає роботу з програмами лояльності.
- Низька вартість і мінімальні вимоги до інтеграції з іншими системами.

Недоліки:

- Мінімальний функціонал у плані персоналізації, автоматизації та аналітики.
- Немає підтримки глибокої інтеграції з POS-системами або CRM.
- Не підходить для бізнесів, які потребують комплексного підходу до побудови клієнтських відносин.

BonusQR — чудове рішення для малого бізнесу, який хоче протестувати програму лояльності без значних витрат і технічних бар'єрів. Це ідеальний стартовий варіант, однак для подальшого зростання можливостей може бути недостатньо [3].

1.3 Мотивація вибору напряму розробки

У сучасному динамічному світі бізнесу особливе значення набуває здатність компаній ефективно взаємодіяти зі своїми клієнтами, формуючи довготривалі й лояльні відносини. Для закладів сфери HoReCa, зокрема кафе, які часто працюють в умовах високої конкуренції та змінного потоку клієнтів, це питання є не лише важливим, а критичним для виживання та розвитку бізнесу. Саме тому все більше власників закладів звертають увагу на цифрові інструменти лояльності — зокрема, персоналізовані програми, які дозволяють збирати та використовувати дані про клієнтів для надання їм індивідуального досвіду.

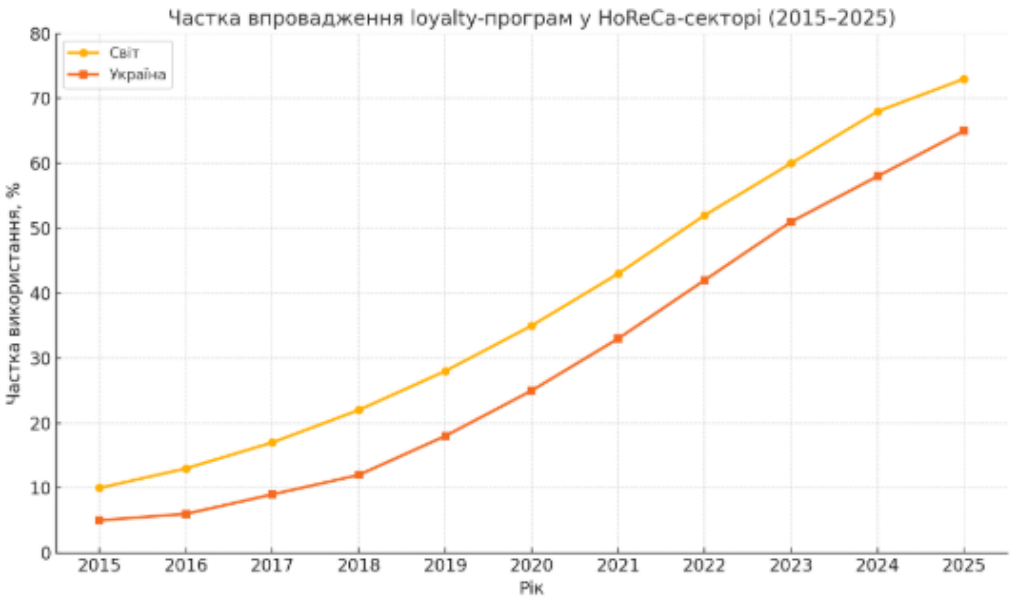

Рис. 1.4 – Частка впровадження програм лояльності у HoReCa 2015-2025

Ідея програм лояльності у бізнесі бере початок ще з 1929 року, коли компанія Sperry & Hutchinson започаткувала видачу “Green Stamps” — купонів, які можна було обміняти на товари. Відтоді концепція бонусів і знижок трансформувалась від паперових накопичувальних карток до повноцінних CRM-систем. Починаючи з 1990-х років, зі зростанням обчислювальних можливостей і доступністю інтернету, відбувся перехід до електронних програм лояльності. Серед піонерів сучасних рішень — такі компанії, як Starbucks, яка ще у 2009 році запустила власну мобільну програму лояльності з бонусами, що прив’язувались до особистого кабінету користувача.

З розвитком хмарних технологій, веб-фреймворків і підходів до клієнтського маркетингу, відбувся значний стрибок у якості цифрових рішень. Сьогодні програми лояльності — це не просто система знижок. Це складна система взаємодії, що поєднує в собі елементи аналітики, штучного інтелекту, автоматизованого маркетингу та персоналізації. На цьому етапі на перший план виходить здатність бізнесу формувати унікальний клієнтський досвід.

Враховуючи зростаючі очікування клієнтів щодо персоналізованого підходу та мобільного доступу, розробка веб-застосунку для програми лояльності є актуальним і виправданим напрямом. Веб-технології дозволяють забезпечити

кросплатформність, доступність, зручність і швидке масштабування. У цьому контексті використання фреймворку Next.js відкриває додаткові переваги.

Next.js, як прогресивний фреймворк для React, активно розвивається з 2016 року, коли він був вперше представлений компанією Vercel. З того часу він став одним із провідних інструментів у світі frontend-розробки, завдяки підтримці серверного рендерингу (SSR), генерації статичних сторінок (SSG), автоматичному поділу коду, вбудованому маршрутизаційному механізму та високій продуктивності. Його архітектура дозволяє легко адаптувати застосунок до великої кількості клієнтів, оптимізувати швидкість завантаження та покращити SEO — усе це є критичним для залучення і втримання користувачів [4].

Рис. 1.5 – Переваги технології Next.js

Крім технологічних аспектів, варто врахувати й соціально-економічні передумови. У 2020–2023 роках, після пандемії COVID-19, сфера громадського харчування зіткнулася з необхідністю переосмислення комунікації з клієнтами. Обмеження, дистанція, зменшення потоку спонтанних відвідувань змусили бізнеси зосередитись на повторних візитах та лояльній аудиторії. Саме у цей період попит на digital-інструменти взаємодії з гостями виріс у кілька разів. Зокрема, за даними McKinsey, у 2021 році понад 78% малих і середніх бізнесів у

сфері обслуговування почали або запланували цифрову трансформацію процесів взаємодії з клієнтами.

На українському ринку вже існують певні рішення, проте більшість із них або надто складні для впровадження в малих закладах, або не враховують специфіки національного користувача, ну або ж банально не забезпечують достатнього рівня персоналізації. У зв’язку з цим з’явилась потреба у розробці власного, більш гнучкого, адаптивного та доступного інструменту.

Таким чином, вибір напряму розробки веб-застосунку персоналізованої програми лояльності з використанням Next.js є обґрунтованим як з практичного, так і з технологічного погляду. Він відповідає актуальним тенденціям цифрової трансформації у сфері кафе та ресторанного бізнесу, забезпечує гнучкість, масштабованість, зручність і можливість глибокої взаємодії з користувачем. Такий підхід сприяє не лише покращенню сервісу, а й формуванню довготривалих, емоційно забарвлених зв’язків із клієнтами — що є ключовим чинником конкурентоспроможності у XXI столітті.

1.4 Обгрунтування вибраної методології створення системи

Під час розробки веб-застосунку персоналізованої програми лояльності для кафе одним із ключових аспектів є вибір ефективної методології розробки, яка дозволяє не лише забезпечити стабільність процесу, а й адаптуватися до змін у вимогах та масштабах проєкту. У цій роботі обрано методологію Feature-Driven Development (FDD), яка належить до гнучких (Agile) підходів та зосереджена на реалізації програмного забезпечення через чітко визначені функціональні елементи, які несуть користувацьку цінність.

Методологія FDD була створена у 1997 році Джеффом Де Лукою у співпраці з Пітером Кодером у межах масштабного проєкту для банківського сектору в Сінгапурі. З того часу вона була успішно застосована в численних

проєктах, де було потрібно зберігати баланс між суворою структурованістю та гнучкістю реалізації. Особливо доцільним її використання є у випадках, коли розробка ведеться невеликою або середньою командою, але потребує поступової, контрольованої розробки з можливістю масштабування.

У межах цієї кваліфікаційної роботи розглядається побудова системи лояльності, що передбачає впровадження низки послідовних функціональних компонентів — зокрема, реєстрацію клієнтів, нарахування бонусів, облік знижок, формування статистики, керування акціями тощо. Така структура задач ідеально відповідає принципам FDD, де кожна окрема функція системи виступає як завершена одиниця розробки.

Методологія FDD включає п’ять основних етапів:

• Розробка загальної моделі (Develop Overall Model) — створення загальної уявної структури системи, що охоплює її логіку, основні об’єкти, сценарії взаємодії та доменну модель.

• Формування списку функцій (Build Feature List) — складання переліку функцій, що реалізують очікувану поведінку системи. Наприклад, «створити нового клієнта», «переглянути його бонусний баланс», «відобразити статистику за останній місяць».

• Планування по функціях (Plan by Feature) — розподіл відповідальності за реалізацію кожної функції, визначення етапів та черговості виконання.

• Проєктування по функціях (Design by Feature) — створення дизайну та архітектури для кожної функції окремо, з урахуванням її взаємозв’язків із загальною структурою системи.

• Реалізація функції (Build by Feature) — безпосереднє програмування функціональності, її модульне тестування та інтеграція в основний продукт.

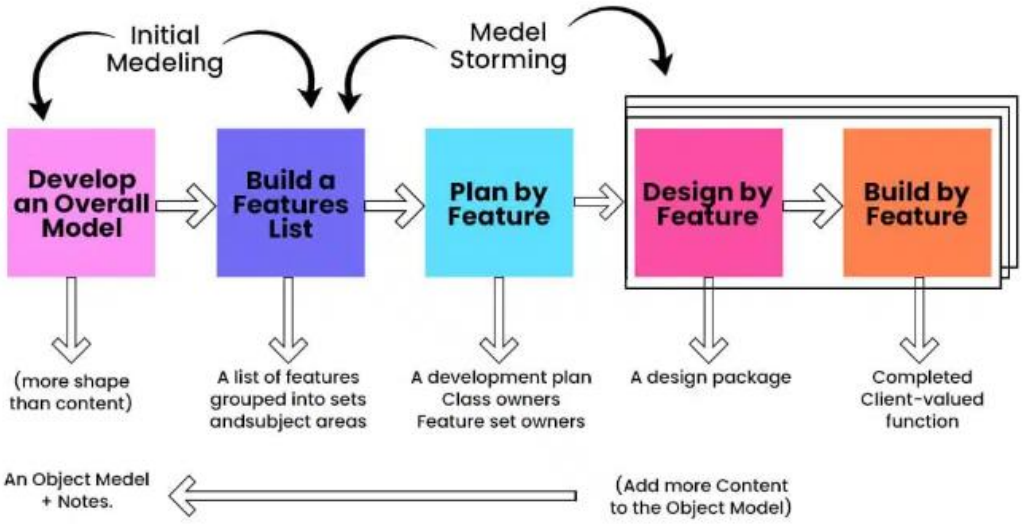

Рис. 1.6 – Принцип роботи методології Feature-Driven Development (FDD)

Застосування FDD у цьому проєкті зумовлене кількома ключовими перевагами:

• Чітке структурування розробки дозволяє уникнути неузгодженості між компонентами системи.

• Можливість послідовного тестування й інтеграції функцій гарантує вищу стабільність і зменшує ризики накопичення помилок.

• Орієнтація на готові до використання функції дозволяє отримувати відчутний результат на кожному етапі, що є критично важливим у розробці MVP (мінімально життєздатного продукту).

• Простота масштабування: додавання нових функцій або модифікація вже реалізованих не вимагає зміни загальної архітектури системи.

• Прозорість процесу для замовника або зацікавленої сторони: функціональна карта системи є чіткою та зрозумілою, що дозволяє контролювати прогрес та коригувати пріоритети.

Обрана методологія дозволяє ефективно розвивати систему відповідно до змін у бізнес-потребах, а також забезпечити високу якість розробки та стійкість до навантажень у майбутньому. Особливо це важливо в контексті веб-застосунку, який передбачає активну взаємодію з кінцевими користувачами, швидкий відгук

інтерфейсу, а також можливість масштабування з розвитком закладу або мережі кафе.

Отже, Feature-Driven Development цілком відповідає специфіці даного проєкту і є обґрунтованим вибором для досягнення ефективної, поетапної та контрольованої розробки сучасного веб-застосунку системи лояльності [5].

1.5 Визначення функціональних та нефункціональних вимог

У минулому розділі було проведено аналіз існуючих рішень у сфері програм лояльності для кафе, зокрема таких сервісів, як Loyallyst, Eder та BonusQR. Детальне вивчення функціоналу, сильних і слабких сторін цих систем дозволило краще зрозуміти їх підходи до реалізації клієнтської взаємодії. Виходячи з цього аналізу, а також з урахуванням актуальних запитів ринку та особливостей роботи HoReCa-сектору, у цьому розділі буде сформовано перелік вимог до створюваної веб-платформи персоналізованих програм лояльності. Окрім цього, розглянемо принцип роботи програми лояльності, її зміст та можливості для кафе. Застосунок BonusCafe – це платформа для ефективного керування програмами лояльності в кафе та ресторанах. Вона дозволяє створювати персоналізовані програми, у котрих клієнти можуть отримати бонуси за здійснення покупок у кафе, маючи згодом змогу обміняти їх на знижки або ж пропозиції від закладу. Система включає в себе багаторівневу лояльність, що стимулює клієнта повертатись у цей заклад та повторно здійснювати покупки, аби згодом отримати з цього вигоду.

Власники бізнесу мають повний контроль над клієнтською базою, аналітикою поведінки клієнтів та можливість створювати різні пропозиції та акції. Платформа надає змогу отримати розширену аналітику ефективності програми, провести повну персоналізацію під бренд закладу та автоматизацію усіх процесів, пов’язаних з нарахуваннями балів та застосуванням знижок.

Програма лояльності допомагає закладам не лише утримати свою вже наявну постійну аудиторію, а й залучати нову. Особливо корисним таке введення буде для закладів, котрі лише недавно розпочали свою роботу, оскільки у них відсутня постійна аудиторія відвідувачів. Закладам, котрі уже певний час перебувають на ринку, програма лояльності також допоможе підняти середній чек та дохід взагальному.

До функціональних вимог проєкту варто віднести наступні:

1. Управління клієнтами:

   Система дозволяє реєструвати нових клієнтів, переглядати їхню інформацію, відстежувати історію відвідувань та проводити їхній аналіз для формування висновків з метою покращення кількості відвідувань та загального враження клієнтів.

2. Система бонусів:

   Передбачена система нарахування бонусів за відвідування та їх списання при використанні. Власник може встановлювати умови нарахування бонусів і переглядати історію усіх бонусних операцій.

3. Рівні лояльності:

   Функціонал включає у себе налаштування рівнів лояльності з особливими перевагами для кожного з них, встановлення мети для їх досягнення та автоматичне підвищення або ж пониження цих рівнів.

4. Цифрова картка клієнта:

   Система має можливість налаштовувати дизайн цифрових карт, відображати основу інформацію про клієнта та змогу проведення інтеграції з Apple Wallet та Google Pay.

5. Аналітика:

   Цей розділ забезпечує усіх необхідні інструменти для проведення детальної аналітики клієнтів: частоти відвідувань, аналіз середнього чеку, статистику використання бонусів.

До нефункціональних вимог віднесемо:

1. Продуктивність:

   Система повинна забезпечувати швидкий час відгуку(не більше 2-х секунд) та завантаження сторінки(не більше 3-х секунд), одночасно підтримуючи понад 1000 користувачів завдяки своїй оптимізованій роботі з базою даних.

2. Безпека:

   Пріоритетом є надійних захист: шифрування даних користувачів, захист від SQL ін’єкцій та XSS атак, впровадження двофакторної автентифікації та постійне проведення резервного копіювання даних.

3. Надійність:

   Система повинна забезпечувати високу доступність, автоматичне відновлення після збоїв, ефективно обробляти великі навантаження та гарантувати захист від втрати даних.

4. Масштабованість:

   Архітектура системи передбачає горизонтальне масштабування, підтримку зростання бази користувачів, можливість додавання нових функцій та адаптацію до збільшення загального навантаження.

5. Зручність використання:

   Інтерфейс системи повинен бути інтуїтивно зрозумілим, адаптивний дизайн забезпечить коректне відображення на всіх пристроях, зі змогою вибору різних мов.

6. Сумісність:

   Система працюватиме в усіх сучасних браузерах, підтримувати мобільні платформи iOS та Android, а також мати інтеграцію з популярними POS-системами та платіжними системами.

7. Підтримка:

   Для забезпечення ефективної роботи та розвитку системи буде надаватись документація для користувачів та технічна документація, впроваджена система відстеження помилок, з регулярними оновленнями застосунку.

Отже, чітке розуміння та змога розмежування функціональних вимог від нефункціональних є важливим етапом для розробки веб-застосунку.

Функціональні вимоги, які визначають конкретні функції у системі, та нефункціональні вимоги, котрі визначають, як саме система повинна працювати.

1.6 Моделювання взаємодії між користувачем та системою

Для кращого розуміння того, як актори повинні взаємодіяти з програмою лояльності, варто описати основні сценарії їхньої роботи. У цьому розділі розглянемо моделювання цих взаємодій у двох акторів:

• Власник кафе:

Власник кафе отримує повний контроль над програмою лояльності, починаючи з реєстрації у системі та детального налаштування параметрів: відсотка нарахування бонусів та визначення рівнів лояльності з їхніми перевагами. Він має змогу переглядати списки усіх клієнтів та їхні профілі, глибоко аналізувати статистику відвідувань та створювати персоналізовані пропозиції й акції, щоб максимально залучити свою аудиторію. Окрім того, власник може налаштовувати зовнішній вигляд цифрової картки, генерувати QR-коди для ідентифікації закладу, переглядати звіти про ефективність програми та керувати правами доступу співробітників. Для повної інтеграції передбачено налаштування зв'язку з POS-системами, а також можливість безпосередньо надсилати повідомлення клієнтам про актуальні акції.

• Клієнт кафе:

Клієнт кафе має змогу приєднатись до програми лояльності, зареєструвавшись через мобільний додаток, де відразу отримує унікальний QR-код для швидкої ідентифікації. У додатку він бачить свій поточний баланс бонусів, відстежує рівень лояльності та прогрес до наступного. Також доступна повна історія усіх його візитів та покупок, а накопичені бонуси можна використовувати для отримання знижок. Для максимальної зручності цифрову

картку легко додати до Apple Wallet або Google Pay. Клієнти завжди будуть в курсі актуальних пропозицій завдяки push-повідомленням, матимуть доступ до усіх акцій, зможуть ділитися програмою лояльності з друзями та залишати відгуки про заклад.

Таким чином, на основі визначених акторів, упорядкуємо сценарії взаємодії у форматі таблиці, аби зобразити способи використання системи у більш зручному форматі.

Таблиця 1.5.1 – Варіанти використання системи

Номер	Актор	Найменування	Опис
ВК1	Власник кафе	Реєстрація кафе	Власник кафе може зареєструвати своє кафе в системі вказавши назву, адресу та основі параметри закладу
ВК2	Власник кафе	Вхід в систему	Власник кафе може увійти в систему для управління програмо лояльності свого закладу
ВК3	Власник кафе	Налаштування програми лояльності	Власник кафе може налаштувати відсоток нарахування бонусів, створити рівні лояльності та встановити правила програми
ВК4	Власник кафе	Перегляд клієнтів	Власник кафе може переглядати список усіх клієнтів та бачити детальну інформацію з історією їхніх відвідувань та замовлень
ВК5	Власник кафе	Аналіз статистики	Власник кафе може переглядати детальну аналітику відвідувань, популярних товарів та ефективності програми лояльності

Продовження таблиці 1.5.1

ВК6	Власник кафе	Створення акцій	Власник кафе може створювати спеціальні пропозиції та акції для залучення клієнтів
ВК7	Власник кафе	Налаштування картки	Власник кафе може налаштовувати зовнішній вигляд цифрової картки
ВК8	Власник кафе	Керування співробітниками	Власник кафе може додавати співробітників та налаштовувати їхні права доступу до системи
КК1	Клієнт кафе	Реєстрація в програмі	Клієнт кафе може зареєструватись в програмі лояльності через мобільний додаток
КК2	Клієнт кафе	Вхід в додаток	Клієнт кафе може увійти в мобільний додаток для перегляду свого профілю
КК3	Клієнт кафе	Перегляд балансу бонусів	Клієнт кафе може перевірити поточний баланс бонусів
КК4	Клієнт кафе	Відстеження рівня лояльності	Клієнт кафе може бачити свій поточний рівень лояльності та прогрес до наступного рівня
КК5	Клієнт кафе	Перегляд історії відвідувань	Клієнт кафе може переглянути історію своїх візитів та покупок
КК6	Клієнт кафе	Використання бонусів	Клієнт кафе може використати накопичені бонуси для отримання знижок або безкоштовних товарів
КК7	Клієнт кафе	Додавання картки в гаманець	Клієнт кафе може додати цифрову картку лояльності до Apple Wallet або Google Pay

Продовження таблиці 1.5.1

КК8	Клієнт кафе	Отримання повідомлень	Клієнт кафе може отримувати push-повідомлення про спеціальні пропозиції та акції
КК9	Клієнт кафе	Перегляд акцій	Клієнт кафе може переглядати доступні акції та пропозиції

Для доступнішої візуалізації зв’язків взаємодії між користувачами та системою на основі даних із Таблиці 1 побудуємо UML-діаграму варіантів використання. Такий тип діаграм є базовим інструментом для визначення вимог до майбутнього програмного забезпечення, адже дає змогу описати функціональність системи з точки зору її взаємодії з зовнішнім середовищем. Важливо те, що діаграма не деталізує технічну реалізацію функцій, а зосереджується на сценаріях користування системою. Її головне призначення — показати, як саме різні типи користувачів (актори) взаємодіють із функціональними можливостями системи (прецедентами), що дозволяє сформувати чітке уявлення про поведінку розроблюваного застосунку [6].

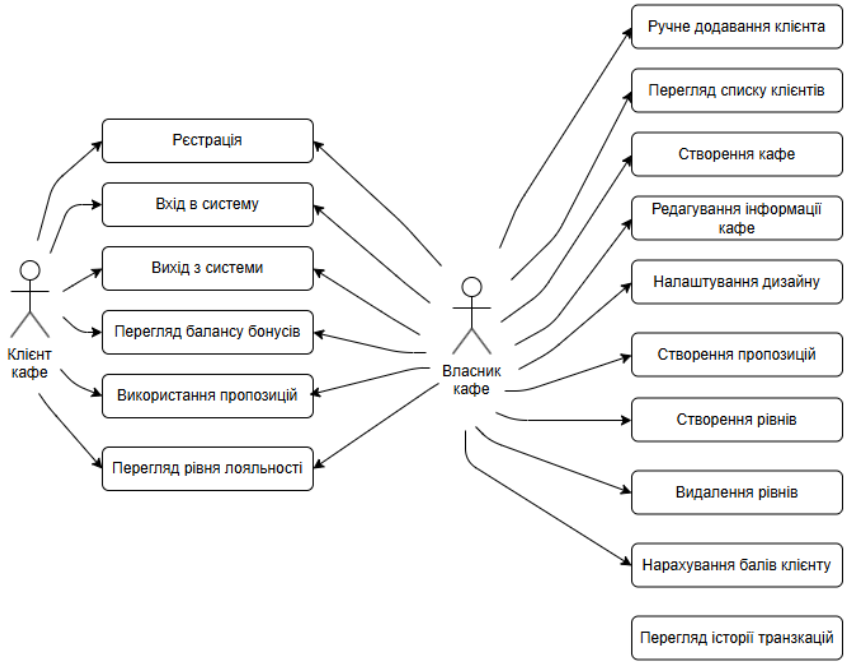

Рис. 1.7 – Діаграма варіантів використання

Отже, у цьому розділі було здійснено комплексне дослідження предметної області з метою виявлення актуальних потреб користувачів та обґрунтування доцільності розробки системи лояльності для кафе. Було проаналізовано існуючі рішення на ринку, виконано їх порівняльну характеристику, що дозволило визначити ключові переваги та недоліки конкурентів на ринку. На основі отриманих даних сформовано мотивацію вибору напряму розробки, яка базується на потребі у більш доступному, адаптованому під малий бізнес інструменті з простим інтерфейсом і можливістю швидкого впровадження.

Обґрунтовано вибір гнучкої методології створення системи, що дозволяє поступове вдосконалення продукту відповідно до змін ринку та зворотного зв’язку від користувачів. Визначено чіткі функціональні та нефункціональні вимоги до майбутньої системи, які відповідають цілям проєкту. На завершення було побудовано моделі взаємодії користувачів із системою, що надало візуальне уявлення про основні процеси та їх учасників.

2 РОЗРОБКА АРХІТЕКТУРИ ТА ПРОЄКУТВАННЯ СИСТЕМИ

2.1 Проєктування архітектури

Архітектура програмної системи — це стратегічний каркас, на якому базується вся подальша розробка. Вона визначає, як взаємодіятимуть між собою компоненти системи, які технології будуть застосовані, як забезпечуватиметься безпека, продуктивність та масштабованість програмного продукту. Етап проєктування архітектури є критично важливим, адже помилки, допущені на цьому рівні, можуть призвести до значних труднощів у реалізації та підтримці проєкту.

У межах цього підпункту буде здійснено проєктування архітектури інформаційної системи BonusCafe з урахуванням раніше визначених вимог. Детально розглядатиметься вибір архітектурного стилю, визначення ключових компонентів, їх ролей та взаємодії, а також обґрунтування обраних технологічних рішень. Враховано також специфіку предметної області, що потребує гнучкості, швидкої обробки даних та зручності використання для кінцевих користувачів — власників закладів, працівників і клієнтів. Усі ці аспекти дозволять побудувати ефективну, надійну та масштабовану систему, яка відповідатиме сучасним стандартам розробки.

2.2 Структурне планування системи

Для системи BonusCafe було обрано трирівневу клієнт-серверну архітектуру з використанням мікросервісного підходу. Такий підхід дозволяє чітко розділити систему на три основні частини: інтерфейс користувача (презентаційний рівень), обробку бізнес-логіки (логічний рівень) та роботу з базами даних (рівень даних).

Це забезпечує зручність у розробці, масштабованість і стабільність роботи системи [9].

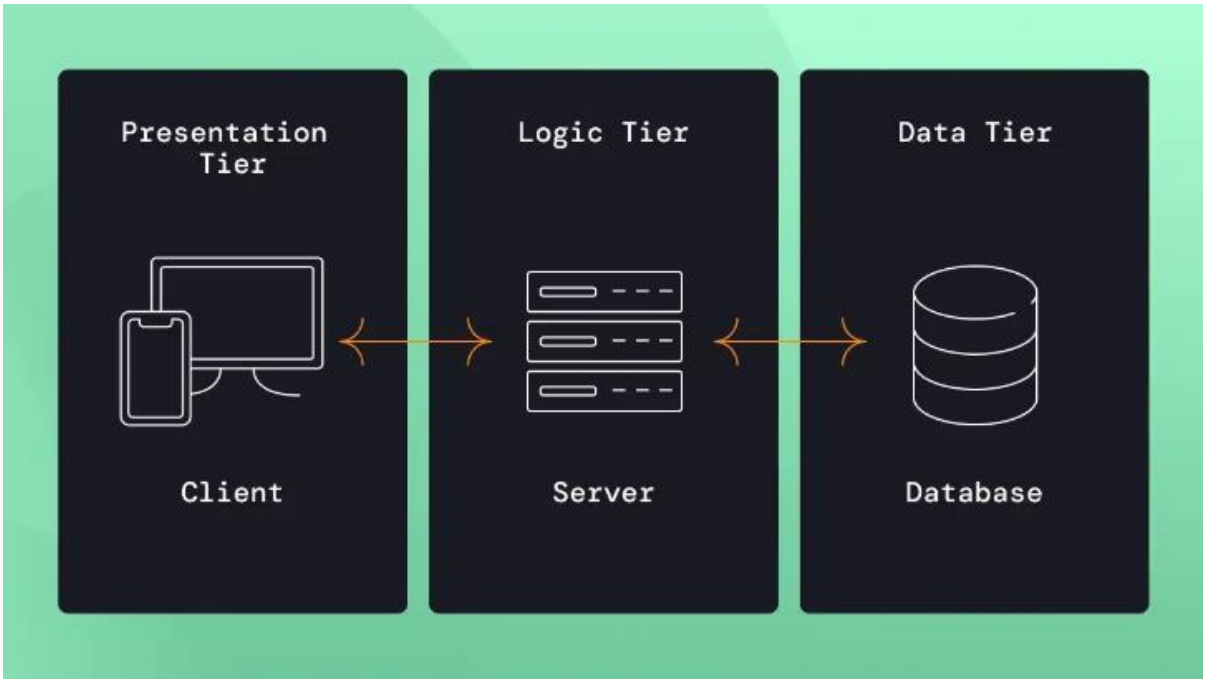

Рис. 2.2.1– Зображення принципу роботи трьохрівневої архітектури

Презентаційний рівень відповідає за все, що бачить і з чим взаємодіє користувач. Він побудований за допомогою Next.js 14 з новим маршрутизатором App Router, що дозволяє створювати сучасні веб-додатки. React 18 і TypeScript використовуються для для створення зручного інтерфейсу з чітко визначеними типами, цим знижуючи ймовірність помилок. Tailwind CSS використовується для швидкого оформлення компонентів, а Shadcn/ui надає вже готові, зручні та настроювані елементи інтерфейсу [4].

Для керування станом інтерфейсу використовується бібліотека Zustand — легка альтернатива Redux. Для обробки форм застосовуються React Hook Form і Zod, що забезпечує зручну перевірку даних, які вводить користувач. Вся структура інтерфейсу побудована за логічним принципом: маршрути згруповані у директорії відповідно до функціоналу.

У системі є три основні типи клієнтів: власники кафе (вони мають повнофункціональну панель управління), співробітники (спрощений інтерфейс для роботи).

Логічний рівень реалізований за мікросервісною архітектурою — система складається з окремих сервісів, кожен з яких відповідає за певну функцію. Наприклад:

- Authentication Service — відповідає за вхід у систему, генерацію токенів (JWT, refresh) та двофакторну перевірку.
- User Management Service — керує обліковими записами, правами доступу, ролями.
- Loyalty Program Service — обробляє правила бонусної програми, рівні лояльності, акції.
- Transaction Service — відповідає за облік бонусних транзакцій, аудит.
- Analytics Service — збирає і аналізує дані про користувачів, ефективність системи.
- Notification Service — надсилає push-, email- і SMS-повідомлення.
- QR Code Service — створює та обробляє QR-коди.

Усі ці сервіси взаємодіють із API Gateway — центральною точкою входу для клієнтів. Вона забезпечує правильну маршрутизацію запитів, авторизацію, обмеження частоти запитів, логування й моніторинг [8].

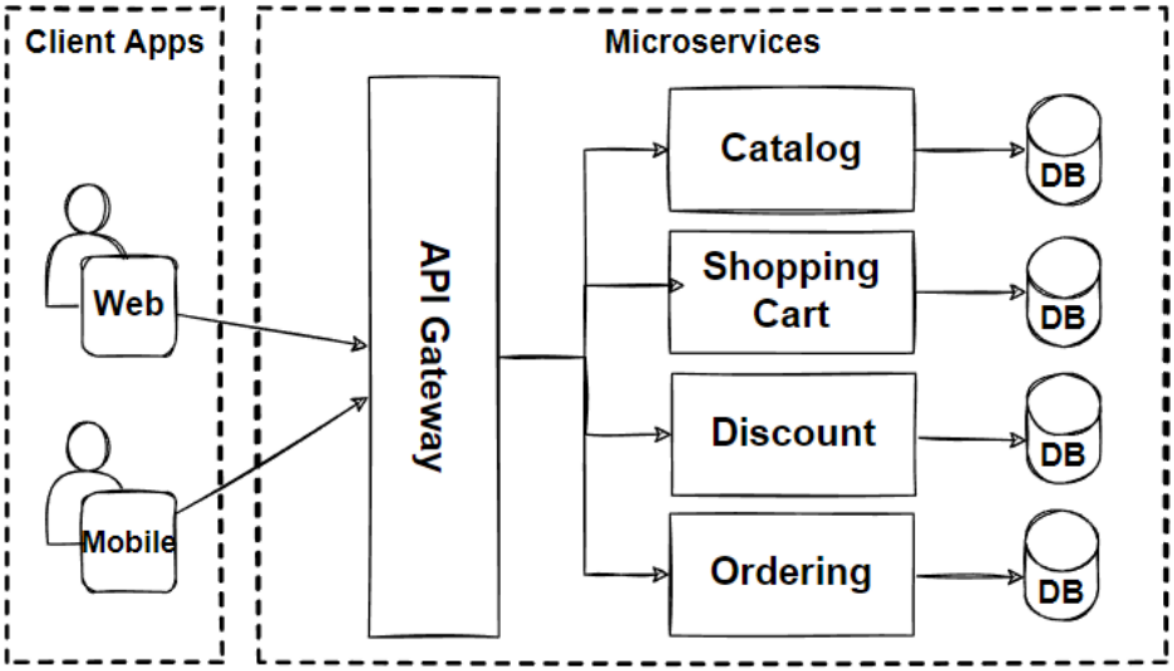

Рис. 2.2.2 – Зображення принципу роботи мікросервісної архітектури

Для зберігання основних даних (користувачі, кафе, транзакції тощо) використовується база даних PostgreSQL. Часто використовувані дані кешуються у Redis, що пришвидшує роботу. Для аналітики виділена окрема база, оптимізована для швидкого читання великих обсягів інформації. Медіафайли зберігаються у файловому сховищі.

Переваги такої архітектури:

- Масштабованість: Кожен сервіс можна масштабувати окремо. Наприклад, якщо потрібно більше ресурсів для аналітики — масштабують лише Analytics Service.
- Надійність: Якщо один сервіс вийде з ладу, інші продовжать працювати.
- Гнучкість: Для кожного сервісу можна використовувати інші мови чи фреймворки, що підходять саме для його задач.
- Зручність підтримки: Кожен сервіс має чітко визначені задачі, тому новим розробникам легко орієнтуватися.
- Безпека: API Gateway фільтрує запити, кожен сервіс може мати власні засоби безпеки, чутливі дані можна ізолювати.
- Тестування: Оскільки сервіси незалежні, їх можна тестувати окремо.
- Моніторинг і логування: Забезпечуються як на рівні Gateway, так і всередині кожного сервісу.

Інтеграція між сервісами:

Сервіси спілкуються між собою або напряму через REST API (якщо потрібна швидка відповідь), або через систему повідомлень (message broker), якщо відповідь може бути з затримкою. Події, наприклад "бонуси нараховано", публікуються у вигляді повідомлень, на які підписуються інші сервіси. Такий підхід (event-driven) робить систему більш гнучкою.

Технології:

- Frontend: Next.js із серверним рендерингом для швидкого завантаження та SEO, TypeScript для безпечного коду.

- Backend: Node.js (з Express або Fastify), Python (з FastAPI), Go — залежно від задач сервісу.
- Бази даних: PostgreSQL (основна), Redis (кеш), окрема аналітична база для звітів.

2.3 Проєктування головних класів системи

На етапі проєктування увага приділяється визначенню основних сутностей системи та взаємозв'язків між ними. Класи формують структуру внутрішньої логіки додатка, відображаючи основні бізнес-об'єкти, такі як користувачі, транзакції, бонусні програми, кафе тощо. Їх правильне проєктування забезпечує зрозумілу, гнучку та масштабовану архітектуру. У цьому підпункті описано ключові класи, їхні атрибути, методи, а також принципи наслідування та взаємодії між ними.

Клас Cafe представляє заклад у системі і містить всю інформацію про кафе, включаючи назву, адресу, контактну інформацію, години роботи, опис та зображення. Клас також зберігає налаштування програми лояльності, такі як відсоток нарахування бонусів, мінімальна сума для нарахування та спеціальні правила. Методи класу включають оновлення інформації про кафе, управління налаштуваннями лояльності, додавання співробітників та генерацію звітів. Cafe служить як центральна сутність, навколо якої організована вся програма лояльності. Розглянемо його атрибути та методи:

Атрибути:

- id: UUID — унікальний код кав'ярні у системі.
- name: String — назва кав'ярні.
- description: string – опис закладу.
- address: String — фізична адреса закладу.
- phone: String — контактний телефон.

- email: String — контактна електронна адреса.
- ownerId: UUID — ідентифікатор власника кав’ярні (посилання на клас CafeOwner).
- createdAt: Date — дата і час створення запису про кав’ярню у системі.
- isActive: Boolean — статус активності кав’ярні (чи працює вона зараз).

Методи:

- addEmployee(employee: Employee): void — додає співробітника до кав’ярні, що дозволяє йому працювати та взаємодіяти з клієнтами в межах цього закладу.
- removeEmployee(employeeId: UUID): void — видаляє співробітника з кав’ярні (наприклад, якщо він звільнився).
- updateSettings(settings: CafeSettings): void — оновлює налаштування кав’ярні, що можуть включати інформацію про робочі години, контакти, правила обслуговування тощо.

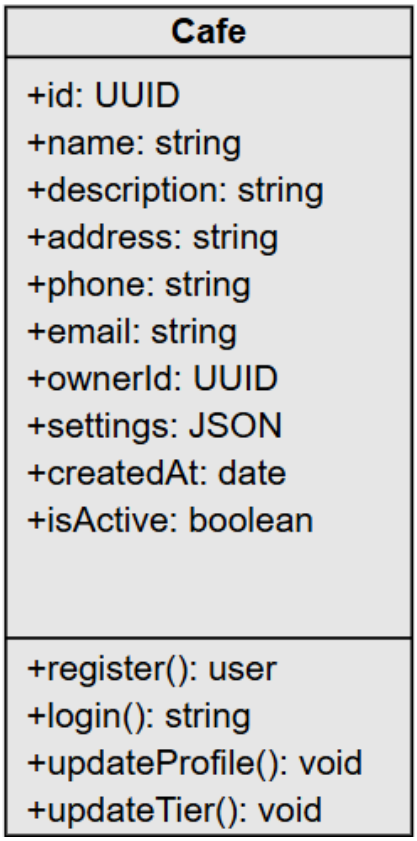

Рис. 2.3.1– Клас Cafe

Клас Transaction є центральним елементом системи BonusCafe, що відповідає за всі операції з бонусними балами між користувачами та кафе. Цей клас реалізує основну бізнес-логіку програми лояльності, забезпечуючи нарахування, списання та відстеження всіх транзакцій. Transaction містить повну інформацію про кожну операцію та надає методи для аналізу фінансових потоків і поведінки клієнтів.

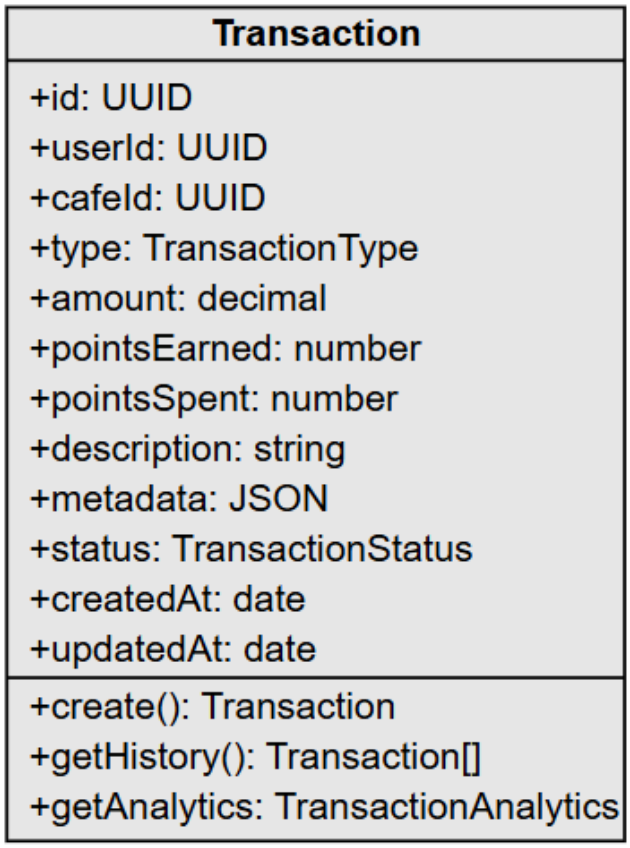

Рис. 2.3.2 – Transaction

Клас User є базовим класом для всіх типів користувачів системи. Він містить основні атрибути, такі як унікальний ідентифікатор, електронна пошта, пароль, повне ім'я, номер телефону, дата реєстрації та статус активності. Цей клас також включає методи для аутентифікації користувача, оновлення профілю, зміни пароля та валідації електронної пошти. User служить як батьківський клас для спеціалізованих типів користувачів і забезпечує базову функціональність управління обліковими записами. Розглянемо його поля:

Атрибути:

- id: string - унікальний код клієнта у системі.
- email: string - електронна пошта клієнта.
- password: string - хешований пароль.
- fullName: string - повне ім'я клієнта.
- phone: string - номер телефону клієнта.
- createdAt: Date - дата створення акаунту.
- updatedAt: Date - дата останніх змін.
- isActive: boolean - статус активності облікового запису
- lastLoginAt: Date - дата останнього входу в систему

Методи:

- authenticate(password: string): boolean - перевірка пароля при вході
- updateProfile(data: UserProfileData): void - оновлення профілю користувача
- changePassword(oldPassword: string, newPassword: string): boolean - зміна пароля
- validateEmail(): boolean - валідація формату електронної пошти
- deactivate(): void - деактивація облікового запису
- getLastActivity(): Date - отримання дати останньої активності

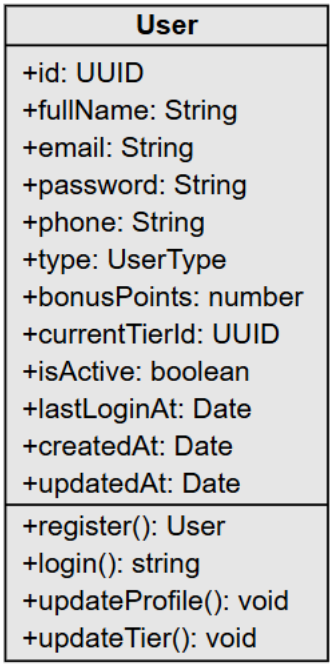

Рис. 2.3.4 – Клас User

Ці класи формують основу архітектури системи BonusCafe і забезпечують всю необхідну функціональність для ефективної роботи програми лояльності. Кожен клас має чітко визначені відповідальності та взаємодіє з іншими класами через добре продумані інтерфейси, що забезпечує модульність, підтримуваність та масштабованість системи.

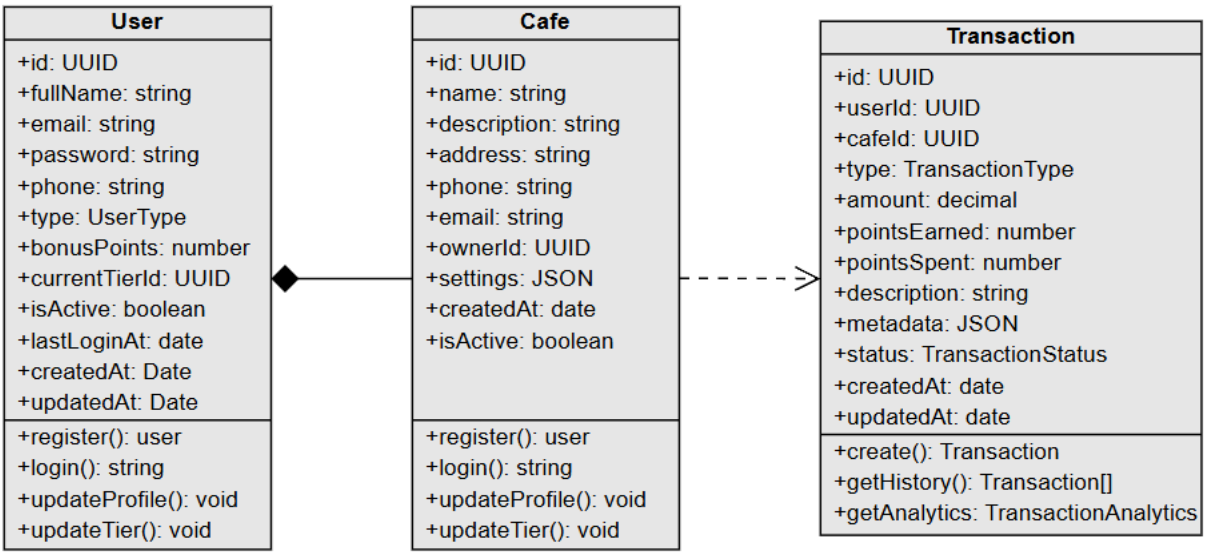

Рис. 2.3.5 – Зв’язки між основними класами

Було створено діаграму класів, у котрій наведено класи та зображено зв’язки між ними. Переглянути повну діаграму класів можна у додатку Б, Рис.Б.2.

2.4 Розробка серверної частини

Цей розділ присвячений проєктуванню та розробці бекенд-частини веб-застосунку персоналізованих програм лояльності. Серверна частина відповідає за збереження та обробку даних, реалізацію бізнес-логіки програми лояльності, взаємодію з базою даних та надання API для клієнтської частини.

2.4.1 Вибір технологічного стеку для бекенду

Node.js з Express.js було обрано як основну технологію для серверної частини з наступних причин:

1. Єдина мова програмування: Використання JavaScript як на фронтенді (Next.js), так і на бекенді забезпечує консистентність розробки та дозволяє команді працювати з одним технологічним стеком.
2. Асинхронність: Node.js ідеально підходить для обробки великої кількості одночасних запитів, що критично важливо для системи лояльності з багатьма користувачами.
3. Велика екосистема: NPM надає доступ до величезної кількості готових модулів для швидкої розробки (автентифікація, валідація, робота з базами даних).
4. Швидкість розробки: Express.js забезпечує мінімалістичний та гнучкий підхід до створення веб-серверів з можливістю швидкого прототипування.
5. Масштабованість: Можливість легкого горизонтального масштабування через мікросервісну архітектуру [10].

База даних:

PostgreSQL обрано як основну базу даних з наступних міркувань:

1. Реляційна структура: Система лояльності оперує структурованими даними (користувачі, транзакції, бали), що вимагають високої цілісності та консистентності.
2. ACID-властивості: Критично важливо для фінансових операцій з балами лояльності, де кожна транзакція повинна бути атомарною.
3. Продуктивність: PostgreSQL забезпечує високу продуктивність для складних запитів та аналітики.
4. JSON-підтримка: Можливість зберігання JSON-даних для гнучких налаштувань програм лояльності.

5. Розширюваність: Підтримка індексів, тригерів та збережених процедур для оптимізації.

2.4.2 Проєктування бази даних та моделей даних

Основні сутності та їх атрибути:

Модель «User» зберігає основну інформацію про всіх зареєстрованих користувачів системи, які можуть бути як звичайними клієнтами кафе, так власниками кафе.

```
module.exports = (sequelize, DataTypes) => {
  const User = sequelize.define('User', {
    id: {…},
    fullName: {…},
    email: {…},
    password: {…},
    phone: {…},
    type: {…},
    bonusPoints: {…},
    currentTierId: {…},
    isActive: {…},
    lastLoginAt: {…}
  }, {…});

  User.associate = function(models) {
    User.belongsTo(models.LoyaltyTier, {…});
    User.hasMany(models.Transaction, {…});
    User.hasMany(models.Cafe, {…});
    User.belongsToMany(models.Offer, {…});
  };

  return User;
};
```

Рис. 2.4.1 – Реалізація моделі «Користувач»

У моделі "Кафе" зберігається інформація про кожне кафе, яке бере участь у програмі лояльності. Кожне кафе пов'язане з конкретним користувачем, який є його власником або адміністратором.

Рис. 2.4.2 – Реалізація моделі «Кафе»

Ця таблиця містить дані про різні програми лояльності, які можуть бути створені для кожного кафе. Вона визначає базові правила нарахування та списання балів.

Рис. 2.4.3 – Реалізація моделі «Програма лояльності»

Таблиця "Рівні лояльності" дозволяє визначити ієрархічні рівні в рамках кожної програми лояльності, пропонуючи різні переваги або бонуси залежно від кількості накопичених балів.

```
module.exports = (sequelize, DataTypes) => {
  const LoyaltyTier = sequelize.define('LoyaltyTier', {
    id: {…},
    name: {…},
    minPoints: {…},
    maxPoints: {…},
    color: {…},
    loyaltyProgramId: {…},
    benefits: {…},
    order: {…}
  }, {…});

  LoyaltyTier.associate = function(models) {
    LoyaltyTier.belongsTo(models.LoyaltyProgram, {…});
    LoyaltyTier.hasMany(models.User, {…});
  };

  return LoyaltyTier;
};
```

Рис. 2.4.4 – Реалізація моделі «Рівні лояльності»

Таблиця "Транзакції" веде повну історію всіх операцій з балами лояльності, включаючи нарахування, списання, закінчення терміну дії та ручні коригування, пов'язані з конкретним членством.

Рис. 2.4.5 – Реалізація моделі «Транзакції»

Ця таблиця зберігає інформацію про різні акції та спеціальні пропозиції, які кафе можуть надавати своїм учасникам програм лояльності. Пропозиції можуть вимагати списання балів або надавати знижки/безкоштовні товари.

Рис. 2.4.6 - Реалізація моделі «Пропозиції»

Таблиця "Використання пропозицій" фіксує кожен випадок активації або використання конкретної пропозиції клієнтом, пов'язуючи її з відповідним членством.

Рис. 2.4.7 – Реалізація моделі «Використання пропозицій»

Взаємозв'язки між сутностями:

1. Користувач → Кафе: Зв’язок «один до багатьох» (1:N). Один користувач(власник) може керувати кількома закладами. Дозволяє централізовано керувати мережею закладів.
2. Кафе → Програма лояльності: Зв’язок «один до одного» (1:1). Кожен заклад має власну програму лояльності. Забезпечує індивідуальне налаштування бонусних пропозицій та акцій для кожного закладу.
3. Програма лояльності → Рівні лояльності: Зв’язок «один до багатьох» (1:N). Кожна програма лояльності здатна мати кілька рівнів. Створює ієрархію лояльності, мотивуючи клієнта знову повертатись.
4. Користувач → Членство: Зв’язок «багато до багатьох» (N:M) реалізований через сутність «Членство». Один користувач(клієнт) може бути членом багатьох різних програм лояльності. Дозволяє клієнту приєднуватися до різни програм лояльності зберігаючи унікальні дані для кожної з них.

5. Членство → Транзакції: Зв’язок «один до багатьох» (1:N). Кожне членство клієнта пов’язане з багатьма транзакціями. Забезпечує облік історії взаємодії клієнта з програмою лояльності.
6. Пропозиція → Використання: Зв’язок «один до багатьох» (1:N). Кожна пропозиція може бути використана декілька разів. Дозволяє відстежити ефективність тієї чи інакшої пропозиції.
7. Пропозиція → Використання: Зв'язок "один до багатьох" (1:N). Одна програма лояльності може мати декілька пропозицій або акцій. Дає власнику можливість створювати різні маркетингові кампанії.

2.4.3 Реалізація бізнес-логіки та основних функцій

Контролер автентифікації є першою точкою взаємодії для нових користувачів. Він відповідає за безпечну реєстрацію, включаючи перевірку унікальності даних та хешування паролів. Особливістю є автоматичне створення додаткових сутностей для бізнес-користувачів, що забезпечує готовність системи до їх функціонування одразу після реєстрації.

```
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');
const { User, Cafe, LoyaltyProgram, LoyaltyTier } = require('../models');

const generateToken = (userId) => {
  return jwt.sign({ userId }, process.env.JWT_SECRET, {...});
};

const register = async (req, res) => {...};

const login = async (req, res) => {...};

const getProfile = async (req, res) => {
  try {...} catch (error) {
    console.error('Get profile error:', error);
    res.status(500).json({
      error: 'Failed to get profile',
      message: 'Помилка отримання профілю'
    });
  }
};

module.exports = {
  register,
  login,
  getProfile
};
```

Рис. 2.4.8 – Реалізація контролера автентифікаціх

Контролер транзакцій є центральним елементом системи лояльності. Він не лише фіксує всі фінансові операції, але й автоматично розраховує бали, що нараховуються клієнту, базуючись на налаштуваннях програми лояльності. Після успішної транзакції він також оновлює баланс балів користувача та його рівень лояльності.

Рис. 2.4.9 – Реалізація контролера транзакцій

Ця функція призначена для аналізу поведінки клієнтів на основі їхніх транзакцій. Вона обробляє дані для виявлення ключових показників, таких як час відвідувань, середній чек, частота покупок та активність використання бонусів. На основі цих показників генеруються індивідуальні висновки, які допомагають бізнесу краще розуміти своїх клієнтів та адаптувати маркетингові стратегії.

Рис. 2.4.10 – Реалізація аналітики користувача

Цей розділ відповідає за повний життєвий цикл управління програмою лояльності та її рівнями. Це включає отримання детальної інформації про програму, її оновлення, а також створення, зміну та видалення окремих рівнів лояльності. Важливою частиною є перевірка прав власності та валідація даних, що забезпечує цілісність та безпеку системи.

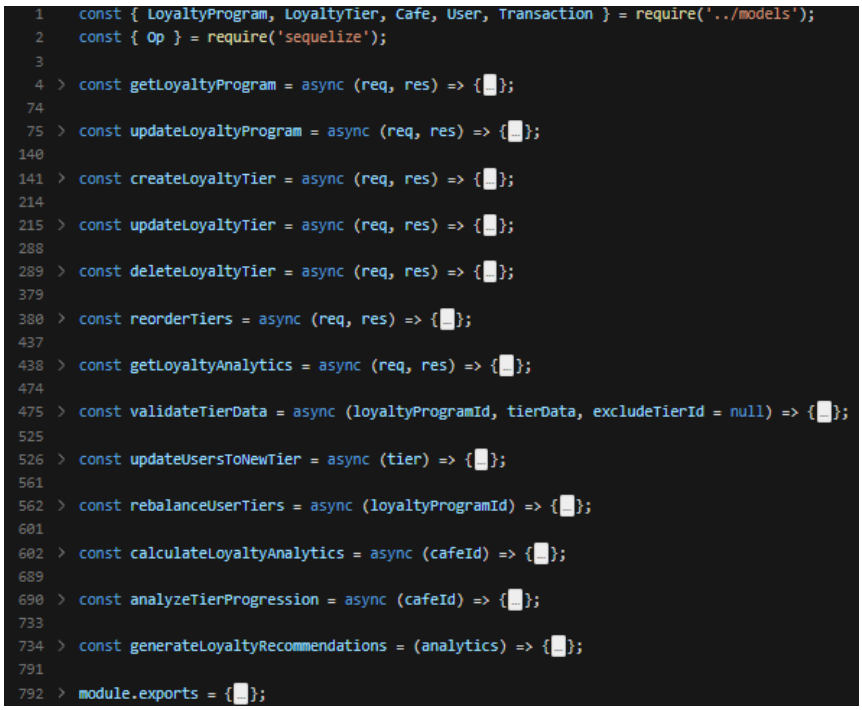

Рис. 2.4.11 – Реалізація управління програмою лояльності

2.4.4 Механізми безпеки та валідації

Автентифікація є першим кроком у забезпеченні безпеки, перевіряючи справжність користувача. Даний механізм використовує JWT (JSON Web Tokens) для перевірки ідентичності користувача на кожен запит. Це дозволяє уникнути повторної передачі облікових даних і забезпечує безперервний та безпечний доступ до ресурсів.

```
const authenticateToken = async (req, res, next) => {
  const authHeader = req.headers['authorization'];
  const token = authHeader && authHeader.split(' ')[1];

  if (!token) {…}

  try {…} catch (error) {
    return res.status(403).json({
      error: 'Invalid token',
      message: 'Недійсний токен'
    });
  }
};
```

Рис. 2.4.12 – Реалізація механізму автентифікації та авторизації

Авторизація встановлює, які дії дозволені автентифікованому користувачу. Завдяки ролям користувачів (business та client), система чітко розмежовує доступ до функціоналу, гарантуючи, що кожен тип користувача має доступ лише до тих ресурсів, які йому призначені. Це запобігає несанкціонованому використанню бізнес-функцій клієнтами та навпаки.

```
const requireBusinessUser = (req, res, next) => {
  if (req.user.type !== 'business') {
    return res.status(403).json({
      error: 'Business account required',
      message: 'Потрібен бізнес-акаунт'
    });
  }
  next();
};

const requireClientUser = (req, res, next) => {
  if (req.user.type !== 'client') {
    return res.status(403).json({
      error: 'Client account required',
      message: 'Потрібен клієнтський акаунт'
    });
  }
  next();
};
```

Рис. 2.4.13 – Реалізація мехінізму розмежування

Валідація даних є критично важливою для підтримки цілісності бази даних та запобігання помилкам, або зловмисним ін'єкціям. Система застосовує багатошаровий підхід до валідації, використовуючи різні інструменти на різних рівнях, щоб забезпечити, що всі вхідні дані відповідають очікуваним форматам та бізнес-правилам.

Joi використовується для валідації вхідних даних на рівні API-запитів. Це дозволяє гнучко визначати схеми для різних типів даних (наприклад, для реєстрації користувачів, створення транзакцій або рівнів лояльності), включаючи перевірку типів, довжини, форматів (наприклад, email, телефон, колір) та бізнес-логіки (наприклад, maxPoints повинно бути більше minPoints).

```
1   const Joi = require('joi');
2
3   const validateRequest = (schema) => {…};
16
17  // Validation schemas
18  const schemas = {
19    register: Joi.object({…}),
26
27    login: Joi.object({…}),
31
32    createCafe: Joi.object({…}),
46
47    createTransaction: Joi.object({…}),
63
64    createOffer: Joi.object({…}),
82
83    createLoyaltyTier: Joi.object({…}).custom((value, helpers) => {
90      if (value.maxPoints <= value.minPoints) {…}
95      return value;
96    }),
97
98    updateLoyaltyProgram: Joi.object({…}),
104
105   createCustomer: Joi.object({…}),
110
111   updateCustomerPoints: Joi.object({…})
116 };
117
118 module.exports = {
119   validateRequest,
120   schemas
121 };
```

Рис. 2.4.14 – Реалізація валідації

Крім автентифікації та валідації даних, серверне оточення також має бути захищеним від різноманітних атак та загроз. Цей розділ показує, як використовуються стандартні бібліотеки та конфігурації для підвищення загального рівня безпеки застосунку.

Інтеграція Helmet забезпечує базовий захист від поширених веб-уразливостей (наприклад, XSS, CSRF), встановлюючи відповідні HTTP-заголовки. Налаштування CORS контролює доступ з інших доменів, запобігаючи небажаним міждоменним запитам. Обмеження частоти запитів (rate limiting) захищає від атак типу "відмова в обслуговуванні" (DDoS), обмежуючи кількість запитів з однієї IP-адреси за певний період.

```
const limiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 хвилин
  max: 100, // ліміт кожного IP по 100 запитів на пристрій
  message: {
    error: 'Too many requests',
    message: 'Забагато запитів з цієї IP адреси'
  }
});
```

Рис. 2.4.15 – Налаштування безпеки

2.5 Розробка клієнтської частини

Клієнтська частина веб-застосунку BonusCafe є основним інтерфейсом взаємодії користувача із системою лояльності. Її розробка була зосереджена на створенні інтуїтивно зрозумілого, високопродуктивного та безпечного середовища, що дозволяє користувачам легко отримувати доступ до інформації та ефективно керувати функціоналом.

2.5.1 Вибір технологічного стеку

Серед основних використаних технологій було обрано наступні:

Next.js 14 з App Router: Обраний як фреймворк для React, Next.js забезпечує оптимізовану продуктивність завдяки Server Side Rendering та Static Site

Generation, що критично для швидкого завантаження контенту та покращення Time to Content. Його файлова маршрутизація спрощує організацію проєкту, а вбудована SEO оптимізація важлива для публічних сторінок. Нативна TypeScript інтеграція та активна підтримка спільноти додатково прискорюють розробку.

TypeScript: Використання TypeScript дозволило досягти значного зменшення кількості помилок на етапі розробки завдяки строгій типізації. Це підвищило продуктивність розробки та забезпечило масштабованість кодової бази, роблячи її легшою для підтримки та розуміння.

Tailwind CSS: Обраний за свій utility-first підхід, що значно прискорює стилізацію компонентів та забезпечує консистентність дизайну завдяки вбудованій дизайн-системі. Його адаптивні breakpoints спрощують створення responsive дизайну, а оптимізація автоматично видаляє невикористані стилі, зменшуючи кінцевий розмір CSS.

Shadcn/ui: Ця бібліотека надає професійно розроблені, доступні UI-компоненти, що базуються на Radix UI. Це гарантує високу якість та доступність компонентів, дозволяючи при цьому розробнику повний контроль над їхньою кастомізацією, оскільки компоненти інтегруються безпосередньо в проєкт. Повна TypeScript підтримка гармонійно вписується в загальну філософію проєкту.

2.5.2 Реалізація основного функціоналу

Опис реалізації основного функціоналу починається з процесу входу та реєстрації.

Форми входу та реєстрації використовують React Hook Form для управління станом форми та Zod для валідації вхідних даних на клієнті. При успішній валідації дані надсилаються на бекенд API. Після отримання відповіді та токена авторизації, стан автентифікації оновлюється через Zustand store, і користувач

перенаправляється на відповідну панель (клієнтська або бізнес-панель) залежно від його ролі.

```tsx
"use client";

import { useState } from "react";
import { useRouter } from "next/navigation";
import { Eye, EyeOff, Loader2 } from "lucide-react";
import { z } from "zod";
import { useForm } from "react-hook-form";
import { zodResolver } from "@hookform/resolvers/zod";

import { Button } from "@/components/ui/button";
import {
  Form,
  FormControl,
  FormField,
  FormItem,
  FormLabel,
  FormMessage,
} from "@/components/ui/form";
import { Input } from "@/components/ui/input";
import { Checkbox } from "@/components/ui/checkbox";
import { useToast } from "@/hooks/use-toast";
import { useAuth } from "@/lib/auth";

const formSchema = z.object({
  email: z.string().email({ message: "Введіть коректну email адресу" }),
  password: z.string().min(6, { message: "Пароль має містити не менше 6 символів" }),
  rememberMe: z.boolean().optional(),
});

export function LoginForm() {…}
```

Рис. 2.4.16 – Реалізація процесу входу/реєстрації

Панель клієнта з бонусами:

Ця сторінка відображає основні показники лояльності: поточну кількість бонусних балів, поточний та наступний рівні лояльності, прогрес до наступного рівня. Використовуються Shadcn/ui компоненти для привабливого та зрозумілого представлення даних.

```
const loyaltyTiers = [
  { name: "Бронзовий", color: "#CD7F32", icon: Award, minPoints: 0, maxPoints: 999 },
  { name: "Срібний", color: "#C0C0C0", icon: Star, minPoints: 1000, maxPoints: 2499 },
  { name: "Золотий", color: "#FFD700", icon: Crown, minPoints: 2500, maxPoints: 4999 },
  { name: "Платиновий", color: "#E5E4E2", icon: Trophy, minPoints: 5000, maxPoints: 9999 }
];

export default function ClientDashboardPage() {
  const { toast } = useToast();
  const [customerData, setCustomerData] = useState<CustomerData>({
    id: "1",
    name: "Іван Петренко",
    email: "ivan@example.com",
    bonusPoints: 3200,
    currentTier: "Золотий",
    tierColor: "#FFD700",
    nextTier: "Платиновий",
    pointsToNextTier: 1800,
    totalSpent: 15600,
    visitCount: 42,
    joinedAt: new Date(2024, 0, 15)
  });

```

Рис. 2.4.17 – Реалізація панелі з бонусами

Клієнти можуть використовувати доступні пропозиції, що вимагають списання бонусних балів. Перед списанням виконується клієнтська валідація на достатність балів, а потім імітується створення транзакції на бекенді. Для надання зворотного зв'язку використовуються toast сповіщення.

```
const handleUseOffer = (offer: Offer) => {
  if (customerData.bonusPoints < offer.bonusCost) {
    toast({
      title: "Недостатньо бонусів",
      description: `Для використання цієї пропозиції потрібно ${offer.bonusCost} бонусів, а у вас ${customerData.bonusPoints}`,
      variant: "destructive"
    });
    return;
  }

  setSelectedOffer(offer);
  setIsConfirmDialogOpen(true);
};
```

Рис. 2.4.18 – Реалізація використання пропозицій

Для кожного клієнта доступна детальна аналітика, що включає інсайти про його поведінку (наприклад, переважний час відвідувань, середній чек). Ці інсайти генеруються на клієнті на основі отриманих даних про транзакції, що забезпечує швидке відображення.

```
87    // Аналіз поведінки клієнта
88  > const getCustomerInsights = () => {…};
133
134   // Підготовка даних для графіків
135   const visitsByDay = DAYS.map(day => ({
136     name: day,
137     visits: visitHistory.filter(visit =>
138       format(visit.date, 'EEEEEE', { locale: uk }).toUpperCase() === day.toUpperCase()
139     ).length
140   }));
```

Рис. 2.4.19 – Реалізація аналітики клієнта

2.5.3 Інтерфейс користувача

Застосунок використовує принципи сучасної дизайн-системи, базуючись на Tailwind CSS та Shadcn/ui.

Для типографії використовується шрифт Inter з підтримкою кирилиці, що забезпечує чітке та сучасне відображення тексту. Визначена ієрархія заголовків за допомогою класів Tailwind CSS гарантує візуальну послідовність та зручність читання.

```
// Використання Inter шрифту з підтримкою кирилиці
const inter = Inter({ subsets: ['latin', 'cyrillic'] });

// Ієрархія заголовків
h1: "text-4xl md:text-5xl lg:text-6xl font-bold"
h2: "text-3xl md:text-4xl font-bold"
h3: "text-2xl font-bold"
h4: "text-xl font-semibold"
```

Рис. 2.4.20 – Типографія

Система використовує toast сповіщення (ненав'язливі повідомлення, що з'являються та зникають), щоб надавати користувачеві негайний зворотний зв'язок про успішність операцій, виникнення помилок або важливі попередження.

```
const { toast } = useToast();

const handleSuccess = () => {
  toast({
    title: "Успішно!",
    description: "Операцію виконано успішно",
    duration: 3000,
  });
};

const handleError = () => {
  toast({
    title: "Помилка",
    description: "Щось пішло не так. Спробуйте ще раз.",
    variant: "destructive",
    duration: 5000,
  });
};
```

Рис. 2.4.21 – Оформлення сповіщень

Бібліотека Framer Motion використовується для додавання плавних анімацій та мікро-взаємодій (наприклад, при наведенні курсору на кнопки, переходи між елементами). Це робить інтерфейс більш "живим" та інтуїтивним, покращуючи загальне сприйняття.

```
22        {/* Літаючі іконки кави */}
23        <div className="absolute inset-0 overflow-hidden z-0">
24          {[...Array(6)].map((_, i) => (
25            <motion.div
26              key={i}
27              className="absolute text-amber-500/20 dark:text-amber-500/10"
28 >            initial={{ … }}
34 v          animate={{
35 >            x: [ … ],
40 >            y: [ … ],
45 v            rotate: [
46                Math.random() * 360,
47                Math.random() * 360,
48                Math.random() * 360
49              ],
50              opacity: [0.3, 0.5, 0.3]
51            }}
52 >          transition={{ … }}
57 >          style={{ … }}
```

Рис. 2.4.22 – Мікро-взаємодії та анімації

2.6 Тестування системи

Наступним, також важливим етапом буде тестування нашого веб-застосунку. Проведемо декілька тест-кейсів для перевірки застосунку на помилки.

1. Тест кейс №1

Summary: Реєстрація в системі

Preconditions: Користувач зайшов на сайт застосунку

Steps to reproduce:

1) Увійти на сайт застосунку
2) Натиснути кнопку «Реєстрація»
3) Ввести свої дані
4) Обрати тип облікового запису
5) Погодитись з умовами
6) Зареєструватись

Expected result: Користувач створить свій особистий акаунт.

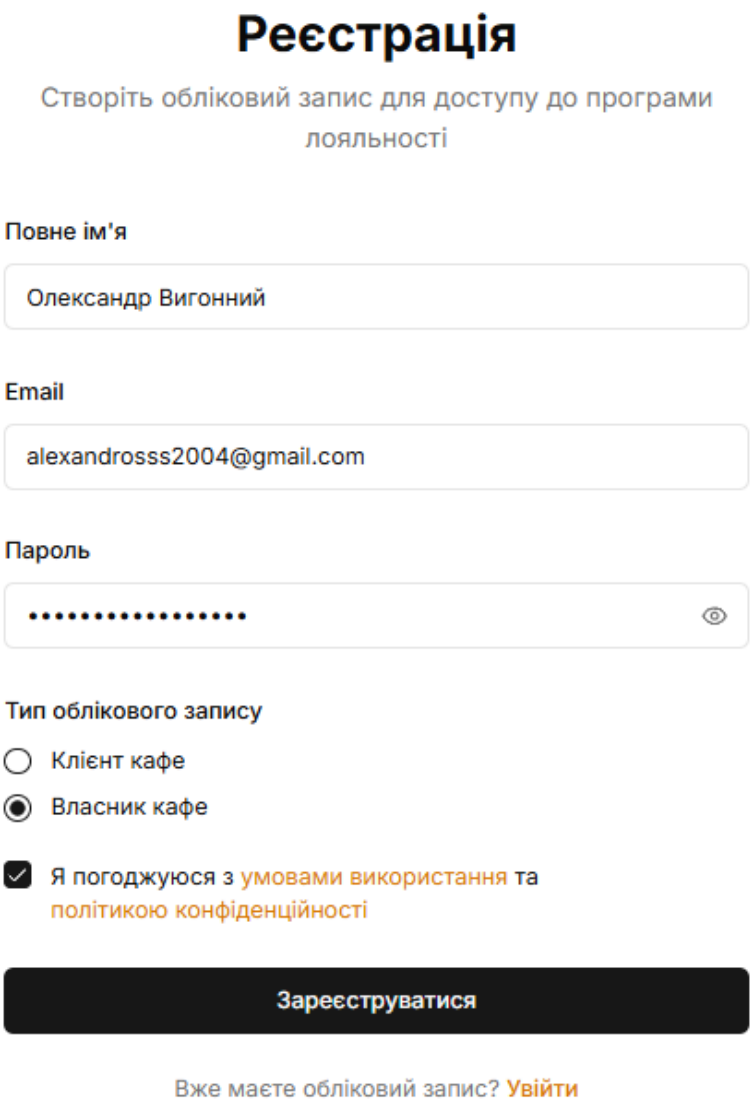

Рис. 2.6.1 – Реєстрація у застосунку

2. Тест кейс №2

Summary: Додавання клієнта

Preconditions: Користувач авторизований як власник кафе.

Steps to reproduce:

1) Увійти в додаток як власник кафе
2) Натиснути кнопку «Клієнти»
3) Натиснути кнопку «Додати клієнта»
4) Ввести інформацію про клієнта
5) Підтвердити

Expected result: Новий клієнт, інформацію про якого увів власник, з’явиться у списку клієнтів.

Ім'я	Email	Телефон	Бонуси	Відвідування
Іван Петренко	ivan@example.com	+380501234567	150	5
Марія Коваленко	maria@example.com	+380671234567	300	8
Олег Тест	testingclient@gmail.com	+380655438893	0	0

Рис. 2.6.2 – Спроба додати нового клієнта

3. Тест кейс №3

Summary: Налаштування рівнів лояльності

Preconditions: Користувач авторизований в системі як власник кафе

Steps to reproduce:

1) Увійти в додаток як власник кафе
2) Зайти у панель керування
3) Редагувати рівень лояльності встановивши його рамки
4) Зберегти налаштування

Expected result: Доданий рівень збережеться за автоматичо застосується до усіх клієнтів.

Налаштування рівнів лояльності

Назва рівня	Мінімум балів	Максимум балів	Колір
Бронзовий	0	999	#CD7F32

Рис. 2.6.3 – Спроба додати новий рівень лояльності

4. Тест кейс №4

Summary: Редагувати картку лояльності в гаманці клієнта

Preconditions: Користувач авторизований в системі як власник кафе

Steps to reproduce:

1) Увійти в додаток як власник кафе
2) Увійти в панель керування
3) Перейти у розділ «Картка в гаманці»
4) Змінити інформацію та зовнішній вигляд картки клієнта
5) Натиснути кнопку «Зберегти налаштування»

Expected result: Картка клієнта зміниться відповідно до змін, які вніс власник кафе.

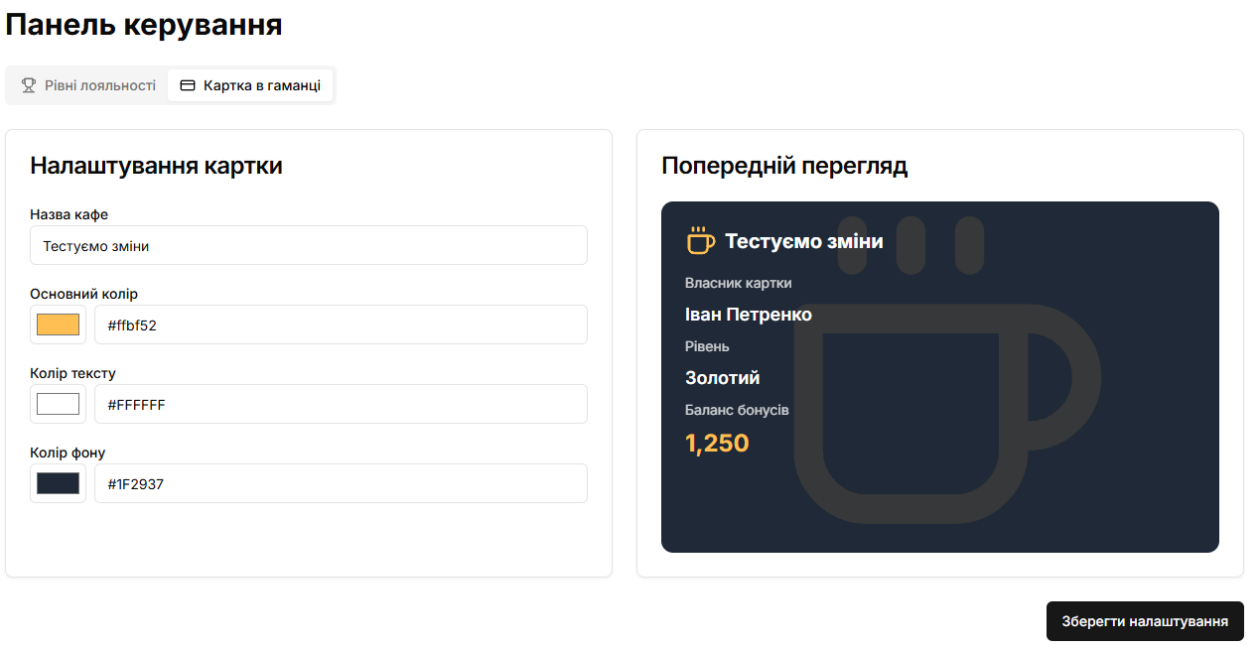

Рис. 2.6.4 – Спроба змінити інформацію про картку клієнта

5. Тест кейс №5

Summary: Додаткове нарахування клієнту бонусів

Preconditions: Користувач авторизований в системі як власник кафе

Steps to reproduce:

1) Увійти в систему як власник кафе
2) Перейти у розділ «Клієнти»
3) Натиснути кнопку «Бали» навпроти клієнта, якому хочемо нарахувати або списпати бали
4) Підтвердити операцію

Expected result: Клієнту додадуться/віднімуться бали після проведеної операції.

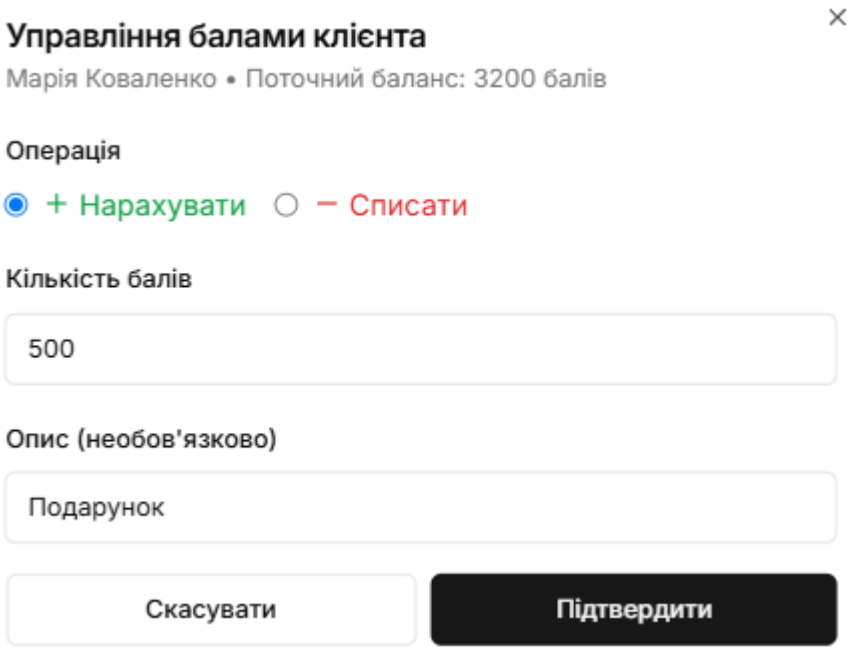

Рис. 2.6.5 – Нарахування балів клієнту

Проведене тестування системи за попередніми умовами підтвердило коректність у роботі системи. Усі основі функції успішно пройшли перевірку, це свідчить про якісну реалізацію поставлених вимог та про змогу проєкту переходити на наступні етапи.

3 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

3.1 Значення адаптації в трудовому процесі

Адаптація в трудовому процесі — це процес поступового пристосування працівника до умов праці, виробничого середовища, колективу, корпоративної культури та вимог посадових обов’язків. Основна мета адаптації — забезпечення ефективного входження нових працівників у трудову діяльність, зниження виробничих ризиків та підвищення рівня безпеки і продуктивності праці.

Особливо важливою є професійна адаптація, що передбачає засвоєння працівником знань та навичок, необхідних для безпечного виконання роботи, а також розуміння правил охорони праці, виробничої дисципліни та корпоративних стандартів. Процес адаптації стосується як новоприйнятих працівників, так і працівників, які переходять на нове робоче місце або ж змінюють свої обов’язки.

Відповідно до статті 153 Кодексу законів про працю України, роботодавець зобов’язаний створити працівникам належні умови для оволодіння професією, а також для безпечного виконання роботи. Крім того, згідно зі статтею 18 Закону України «Про охорону праці», роботодавець повинен організувати проходження працівниками навчання і перевірки знань з питань охорони праці, що є невід’ємною частиною адаптаційного процесу [17].

Також адаптація узгоджується з принципами управління ризиками, визначеними в ДСТУ EN ISO 45001:2019 – «Системи управління охороною здоров’я та безпекою праці. Вимоги та настанови щодо застосування», де зазначено, що організація має враховувати індивідуальні характеристики працівника, зокрема досвід, компетентність і стан здоров’я, у процесах планування безпечної трудової діяльності [17].

До заходів адаптації належать:

- Проведення первинного інструктажу з охорони праці;
- Призначення наставника для новачка;
- Поетапне введення в трудові процеси;

- Психофізіологічна підтримка (створення прийнятної атмосфери);
- Оцінювання ефективності адаптації за допомогою анкетування, співбесід тощо.

Згідно з ДСТУ ISO 10015:2006 — «Менеджмент якості. Настанови з навчання персоналу», ефективна адаптація є частиною системи професійного розвитку працівників та підвищення якості виконання роботи [17].

Недостатній рівень адаптації може призвести до:

- Підвищеної тривожності та стресу;
- Помилок під час виконання роботи;
- Порушень вимог безпеки;
- Зниження продуктивності праці;
- Зростання плинності кадрів.

Таким чином, адаптація є ключовим елементом організації безпечного та ефективного трудового процесу. Її належне впровадження сприяє формуванню відповідального ставлення до безпеки, зменшенню виробничих травм та забезпеченню стабільної інтеграції працівника у професійне середовище.

3.2 Шляхи збереження працездатності та підвищення продуктивності праці

Збереження працездатності працівників і підвищення продуктивності праці є фундаментальними для підприємства, впливаючи на якість завдань, безпеку, задоволеність персоналу та ефективність виробництва. В умовах зростання значення людського капіталу ці аспекти критично важливі.

Працездатність — це комплексна характеристика фізичних, психологічних, емоційних та інтелектуальних можливостей працівника, що забезпечують оптимальне виконання професійних функцій протягом робочого періоду. Згідно з Наказом МОЗ України від 21.05.2007 № 246, працездатність тісно пов'язана зі

станом здоров'я та умовами праці. Її зниження може бути наслідком втоми, професійних захворювань або стресів.

Розрізняють декілька видів працездатності для ефективного управління трудовими ресурсами.

Повна працездатність — здатність виконувати обов'язки без обмежень.

Часткова працездатність — зниження можливості виконувати певні роботи з обмеженнями, що може бути тимчасовим станом; законодавство передбачає переведення на легшу роботу, гнучкий графік.

Тимчасова непрацездатність — втрата працездатності на обмежений період через хворобу, травму, оформлюється листком непрацездатності (лікарняним) і регулюється Законом України «Про загальнообов'язкове державне соціальне страхування».

Стійка (постійна) непрацездатність — значна або повна втрата працездатності на тривалий час/постійно, що призводить до встановлення групи інвалідності, регулюється Законом України «Про основи соціальної захищеності осіб з інвалідністю в Україні».

Тимчасова непрацездатність — це втрата працездатності на обмежений період через хворобу, травму або інші обставини, що оформлюється листком непрацездатності (лікарняним) і регулюється Законом України «Про загальнообов'язкове державне соціальне страхування».

Стійка (постійна) непрацездатність — це значна або повна втрата працездатності на тривалий час або постійно, що призводить до встановлення групи інвалідності, що регулюється Законом України «Про основи соціальної захищеності осіб з інвалідністю в Україні».

Працездатність є багатогранною і включає взаємопов'язані компоненти. Фізична складова охоплює витривалість, силу та координацію, залежить від стану організму. Психологічна складова включає увагу, пам'ять, мислення, емоційну стабільність; здатність до концентрації є ключовою, а перевтома знижує продуктивність. Інтелектуальна складова відображає здатність до аналізу інформації, прийняття рішень та навчання. Емоційна складова характеризує

саморегуляцію, управління емоціями та мотивацію; хронічне емоційне виснаження може призвести до професійного вигорання.

Збереження працездатності — це системний підхід, спрямований на підтримку оптимального функціонального стану організму працівника, що дозволяє уникнути професійних захворювань, перевтоми та емоційного вигорання.

Шляхи збереження та відновлення працездатності працівників:

• Раціональний режим праці та відпочинку. Правильне чергування роботи з відпочинком є базовим. Згідно статей 50–53 КЗпП України, робочий час — до 40 год/тиждень, перерва — від 30 хв. Додаткові перерви передбачені для інтенсивних/небезпечних умов. Дотримання 8-годинного робочого дня, регулярні перерви та достатній сон критичні для відновлення. Зміна завдань, гнучкий графік та додаткові вихідні допомагають уникнути монотонності та знизити стрес [17].

• Організація оптимальних умов праці. Фізичне середовище впливає на функціональний стан працівника. Параметри регулюються ДСН та ДСТУ. Мікроклімат: оптимальна температура (20-24°C), вологість, вентиляція критичні; відхилення, згідно ДСН 3.3.6.042-99, викликають втому. Освітлення: за ДБН В.2.5-28:2018, освітленість (300-500 лк для офісів) має бути оптимальною, інакше втомлюються очі. Шум та вібрація: тривалий вплив призводить до втоми, зниження слуху; ДСН 3.3.6.039-99 встановлює допустимі рівні [18, 19].

• Психологічна підтримка. Сприятливий соціально-психологічний клімат ключовий для морального стану та емоційної стабільності. Конфлікти та відсутність підтримки призводять до стресів і вигорання. Засоби: тренінги стресостійкості, відкриті обговорення, прозора комунікація, командні заходи. Важливо, щоб працівники відчували свою цінність, запобігаючи виснаженню.

• Здоровий спосіб життя та медичне обслуговування. Здоровий спосіб життя (харчування, фіз. активність, сон) безпосередньо впливає на працездатність, підвищуючи стійкість до стресів. Компанії можуть підтримувати це через спортивні програми. Медичне обслуговування невід'ємне: згідно ст. 17 Закону України «Про охорону праці» та Наказу МОЗ №246, обов'язкові медогляди для

працівників у шкідливих умовах дозволяють своєчасно виявляти ризики та запобігати профзахворюванням.

• Профілактика професійного вигорання. Хронічне емоційне виснаження часто призводить до вигорання. Профілактичні заходи: ротація завдань, гнучкий графік, додаткові вихідні, психологічні консультації. Оптимізація робочого навантаження також знижує ризик вигорання.

Шляхи підвищення продуктивності праці:

Підвищення продуктивності праці — стратегічне завдання для підприємства, що прагне до конкурентоспроможності, і вимагає застосування різноманітних методів.

• Мотивація персоналу. Мотивація є рушійною силою. Вона може бути матеріальною (зарплата, премії) та нематеріальною (похвала, кар'єрний ріст). Стаття 143 КЗпП України передбачає заохочення. Важливо, щоб системи мотивації були прозорими, справедливими, прив'язаними до результатів, створюючи можливості для зростання [17, 18].

• Підвищення кваліфікації та розвиток компетенцій. Регулярне навчання, розвиток навичок та розширення знань є основою адаптації. Згідно ДСТУ ISO 10015:2006, навчання — ключовий інструмент підтримки якості. Форми: семінари, вебінари, менторство. Інвестиції в розвиток персоналу є довгостроковими в зростання продуктивності, включаючи навчання з охорони праці [2].

• Автоматизація та механізація. Використання сучасної техніки та ПЗ скорочує час, зменшує фізнавантаження та підвищує точність. Механізація та автоматизація звільняють людину від важких операцій, підвищуючи безпеку та ефективність. Наприклад, впровадження CRM-систем або робототехніки збільшує ефективність, мінімізуючи людські помилки [18].

• Раціоналізація робочого місця. Організація робочого простору за принципами ергономіки зменшує втомлюваність, підвищує комфорт та ефективність. Це включає регульовані меблі, оптимальне розміщення інструментів та освітлення [18]. Добре організоване місце сприяє кращій

концентрації та знижує ризик профзахворювань. Системи управління якістю (ISO 9001, ISO 45001) допомагають систематизувати процеси та підвищувати результативність.

• Колективна взаємодія та командна робота. Ефективна командна робота, де кожен знає свою роль, підвищує продуктивність. Спільне планування, зворотний зв'язок, взаємодопомога та обмін досвідом є ключовими. Злагоджені команди швидше реагують на зміни та ефективніше вирішують проблеми.

ВИСНОВКИ

У цій кваліфікаційній роботі було описано процеси проєктування, реалізації та тестування веб-застосунку для персоналізованих програм лояльності, призначеного для кафе з використанням технології Next.js.

Було проведено глибоке дослідження ринку, існуючих рішень для програм лояльності, а також їх порівняльна характеристика. Це дозволило виявити переваги та недоліки наявних на ринку пропозицій. Мотивація вибору напряму розробки полягала у створенні гнучкої, адаптивної та легко інтегрованої системи, яка б відповідала сучасним вимогам бізнесу та користувачів. Було обґрунтовано вибір методології створення системи, а також детально визначено функціональні та нефункціональні вимоги, що стало основою для подальшого проєктування. Проведено моделювання взаємодії між користувачем та системою, представлене за допомогою діаграми варіантів використання, що забезпечило чітке розуміння функціональних можливостей застосунку.

Розроблено архітектуру системи, яка включає як клієнтську, так і серверну частини, забезпечуючи високу продуктивність та масштабованість. Було виконано детальне структурне планування системи, проєктування головних класів та моделей даних. Вибір сучасного технологічного стеку для бекенду, а також ретельне проєктування бази даних та моделей даних, забезпечили стабільність та безпеку збереження інформації. Реалізовано бізнес-логіку та основні функції, також впроваджено механізми безпеки та валідації, що гарантує коректну роботу та захист даних. Для клієнтської частини було обрано технологічний стек, який дозволив створити інтуїтивно зрозумілий та функціональний інтерфейс користувача. Важливим етапом стало тестування системи, що дозволило виявити та усунути можливі недоліки, забезпечивши високу якість кінцевого продукту.

Також було розглянуто важливі аспекти, що стосуються адаптації в трудовому процесі, збереження працездатності та підвищення продуктивності роботи.

Загалом, розроблений веб-застосунок є комплексним рішенням, яке дозволяє кафе ефективно керувати програмами лояльності, залучати нових клієнтів та утримувати існуючих. Система забезпечує персоналізований підхід до кожного клієнта, що сприяє підвищенню їх лояльності та створенню позитивного враження. Реалізований функціонал, включаючи нарахування та списання балів, управління пропозиціями, аналітику та звітність, надає власникам кафе зручний інструмент для оптимізації бізнес-процесів. Спроєктований веб-застосунок підійде як для менших кафе, так і для більших закладів, оскільки перекриває усі необхідні потреби.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Офіційна сторінка Loyallyst [Електронний ресурс] – Режим доступу: https://www.loyallyst.com

2. Офіційна сторінка Eber [Електронний ресурс] – Режим доступу: https://eber.co

3. Офіційна сторінка BonusQR [Електронний ресурс] – Режим доступу: https://bonusqr.com

4. Next.js Docs [Електронний ресурс] – Режим доступу: https://nextjs.org/docs/app/getting-started

5. Feature Driven Development (FDD) [Електронний ресурс] – Режим доступу: https://www.productplan.com/glossary/feature-driven-development/

6. Architecting for success: how to choose the right architecture pattern [Електронний ресурс] – Режим доступу: https://www.redpanda.com/blog/how-to-choose-right-architecture-pattern

7. Способи генерації сторінок: CSR, SSR, SSG, ISR [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/41585/

8. Про мікросервісну архітектуру [Електронний ресурс] – Режим доступу: https://foxminded.ua/mikroservisna-arkhitektura/

9. Three-Level Architecture [Електронний ресурс] – Режим доступу: https://www.sciencedirect.com/topics/computer-science/three-level-architecture

10. Express web framework (Node.js/JavaScript) – Режим доступу: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Express_Nodejs

11. What Does a Backend Developer Do? [Електронний ресурс] – Режим доступу: https://careerfoundry.com/en/blog/web-development/backend-developer-guide/

12. Next.js Documentation. Features[Електронний ресурс] – Режим доступу: https://nextjs.org/docs/getting-started/features

13. Addy Osmani. The Cost Of Javascript [Електронний ресурс] – Режим доступу: https://medium.com/dev-channel/the-cost-of-javascript-84009f51e99e

14. MongoDB Documentation. Key Features [Електронний ресурс] – Режим доступу: https://docs.mongodb.com/manual/core/key-features/

15. Martin Fowler. Schema Last [Електронний ресурс] – Режим доступу: https://martinfowler.com/bliki/SchemaLast.html

16. Bachynskyi, M., Petryk, M., Brevus, V., Mudryk, I., Glova, B. 3D-hybrid mathematical model for analysis of abnormal neurological movements for the purposes of diagnosis and treatment of limb tremor [Електронний ресурс] - 3rd International Workshop on Information Technologies: Theoretical and Applied Problems, ITTAP 2023. Ternopil. 22–23 November 2023. CEUR Workshop Proceedings, 2023, 183–196. – Режим доступу: https://ceur-ws.org/Vol-3628/paper20.pdf

17. Закон України «Про охорону праці» від 14.10.1992 № 2694-XII [Електронний ресурс] / Верховна Рада України. – 2025. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12#Text

18. Курс «Безпека життєдіяльності та основи охорони праці» [Електронний ресурс] / Електронне навчання ТНТУ. – 2025. – Режим доступу: https://dl.tntu.edu.ua/index.php

19. Безпека життєдіяльності та охорона праці : навчально-методичний комплекс [Електронний ресурс] / М.М. Сакун, І.В. Москалюк, В.Ф. Нагорнюк. – Одеса : «ОДАУ» 2017. – С. 37-41, 85-95, 231-235, 209-211, 314-323 – Режим доступу: http://lib.osau.edu.ua/jspui/bitstream/123456789/1593/1/1-Посібник%20БЖД%20та%20ООП.pdf

20. Дистанційний курс «Кваліфікаційні роботи бакалаврів» сайту дистанційного навчання ТНТУ [Електронний ресурс]. – Режим доступу: https://dl.tntu.edu.ua/bounce.php?course=5329

ДОДАТКИ

ДОДАТОК А

Тези для Міжнародна науково-технічна конференція

«Фундаментальні та прикладні проблеми сучаних технологій»

УДК 004.4
О. Вигонний, І. Мудрик
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ПЕРЕВАГИ ВИКОРИСТАННЯ NEXT.JS ТА MONGODB В КОНТЕКСТІ РОЗРОБКИ ВЕБ-ЗАСТОСУНКІВ ПЕРСОНАЛІЗОВАНИХ ПРОГРАМ ЛОЯЛЬНОСТІ

O. Vyhonnyi, I. Mudryk

ADVANTAGES OF USING NEXT.JS AND MONGODB IN THE CONTEXT OF DEVELOPING WEB APPLICATIONS FOR PERSONALIZED LOYALTY PROGRAMS

Через високу конкурентність у середовищі закладів харчування, побудова ефективних програм лояльності є критично важливою для залучення та утримання клієнтів. Персоналізація цих програм дозволяє створити унікальний досвід для кожного відвідувача, що підвищує його задоволеність та сприяє формуванню довгострокових відносин. Розробка веб-застосунку, який забезпечує гнучке впровадження та управління персоналізованими програмами лояльності, вимагає вибору технологій, що можуть надати необхідну продуктивність, масштабованість та зручність розробки.

Використання фреймворку Next.js надає значні переваги при розробці веб-застосунків для персоналізованих програм лояльності [1]. Завдяки можливостям Server-Side Rendering (SSR) та Static Site Generation (SSG), Next.js забезпечує високу швидкість завантаження сторінок, що є важливим для позитивного користувацького досвіду, особливо при відображенні динамічного контенту, пов'язаного з персональними пропозиціями [2]. Компонентна архітектура React, на якій базується Next.js, дозволяє ефективно будувати складний користувацький інтерфейс з багаторівневими програмами лояльності та детальними профілями клієнтів.

У контексті зберігання та управління великими обсягами даних, необхідних для персоналізації (історія покупок, переваги, бали), документо-орієнтована база даних MongoDB є доцільним вибором [3]. Її гнучка схема дозволяє легко адаптувати структуру даних до різноманітних атрибутів клієнтів та типів бонусів, які можуть змінюватися в рамках персоналізованих програм. Масштабованість MongoDB забезпечує обробку зростаючої кількості користувачів та транзакцій без суттєвої втрати продуктивності [4].

Поєднання Next.js та MongoDB дозволяє створити потужний та гнучкий веб-застосунок. Next.js забезпечує швидкий та динамічний фронтенд, здатний ефективно відображати персоналізовану інформацію, тоді як MongoDB надає надійну та масштабовану основу для зберігання даних, необхідних для реалізації алгоритмів персоналізації та управління програмами лояльності. Буде показано, як саме можливості Next.js та переваги MongoDB можуть бути використані для побудови ефективного веб-застосунку, що підтримує персоналізовані програми лояльності, включаючи систему накопичення балів та багаторівневі програми. Застосування обраних технологій сприяє оптимізації розробки, підвищенню продуктивності та забезпеченню гнучкості рішення для потреб бізнесу.

Таким чином, застосування сучасних підходів до веб-розробки на основі Next.js та MongoDB дозволяє створити ефективний та гнучкий веб-застосунок для персоналізованих програм лояльності кафе. Таке рішення є важливим компонентом сучасних інформаційних систем для закладів харчування [5], сприяючи оптимізації бізнес-процесів, підвищенню лояльності клієнтів та зміцненню позицій на ринку.

Література:

1. Next.js Documentation. Features. URL: https://nextjs.org/docs/getting-started/features
2. Addy Osmani. The Cost Of Javascript. URL: https://medium.com/@addyosmani/the-cost-of-javascript-84009f51e99e
3. MongoDB Documentation. Key Features. URL: https://docs.mongodb.com/manual/core/key-features/
4. Martin Fowler. Schema Last. URL: https://martinfowler.com/bliki/SchemaLast.html
5. Bachynskyi, M., Petryk, M., Brevus, V., Mudryk, I., Glova, B. 3D-hybrid mathematical model for analysis of abnormal neurological movements for the purposes of diagnosis and treatment of limb tremor. 3rd International Workshop on Information Technologies: Theoretical and Applied Problems, ITTAP 2023. Ternopil. 22–23 November 2023. CEUR Workshop Proceedings, 2023, 183–196.

ДОДАТОК Б - Діаграми

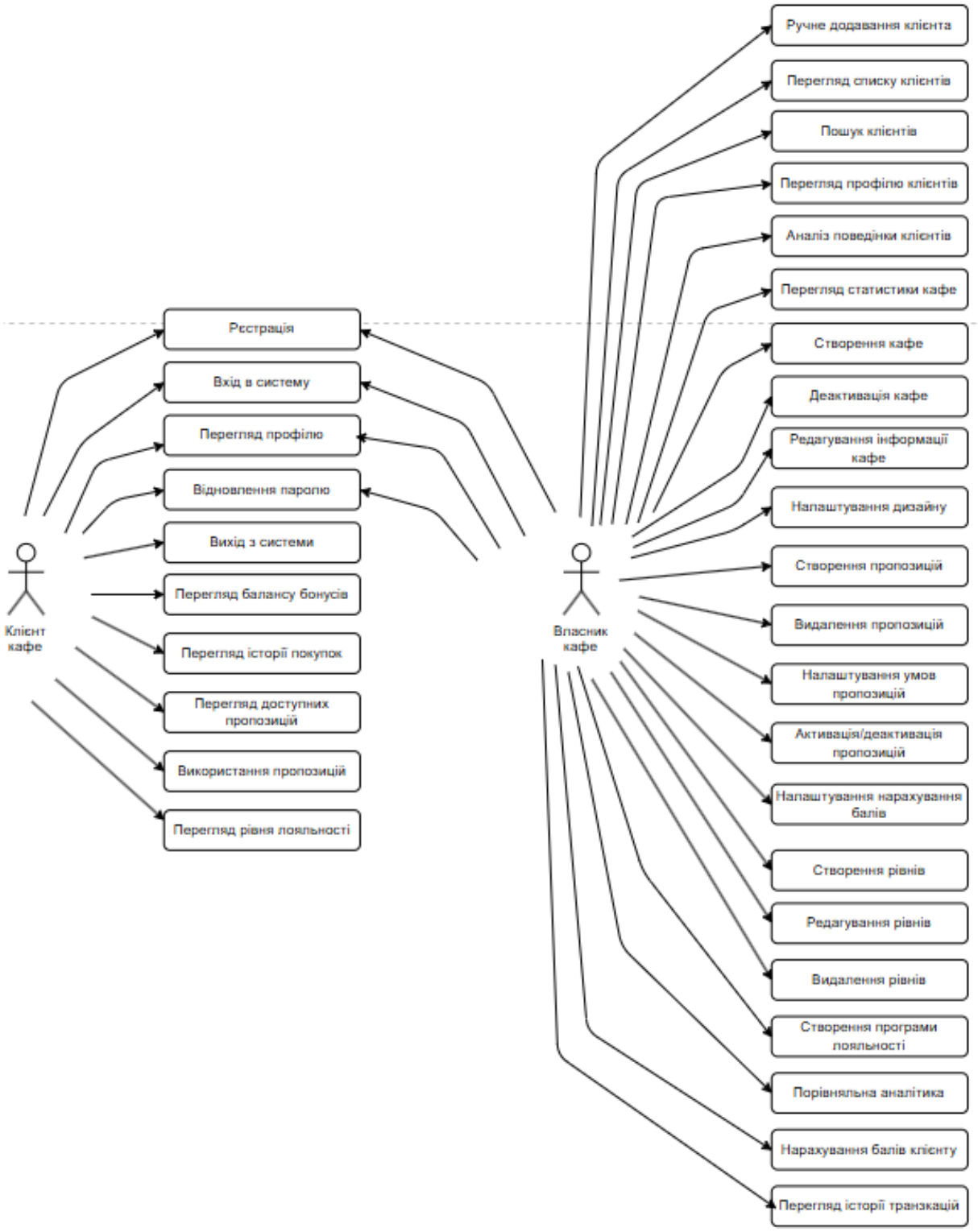

Рис. Б.1 – Діаграма варіантів використання

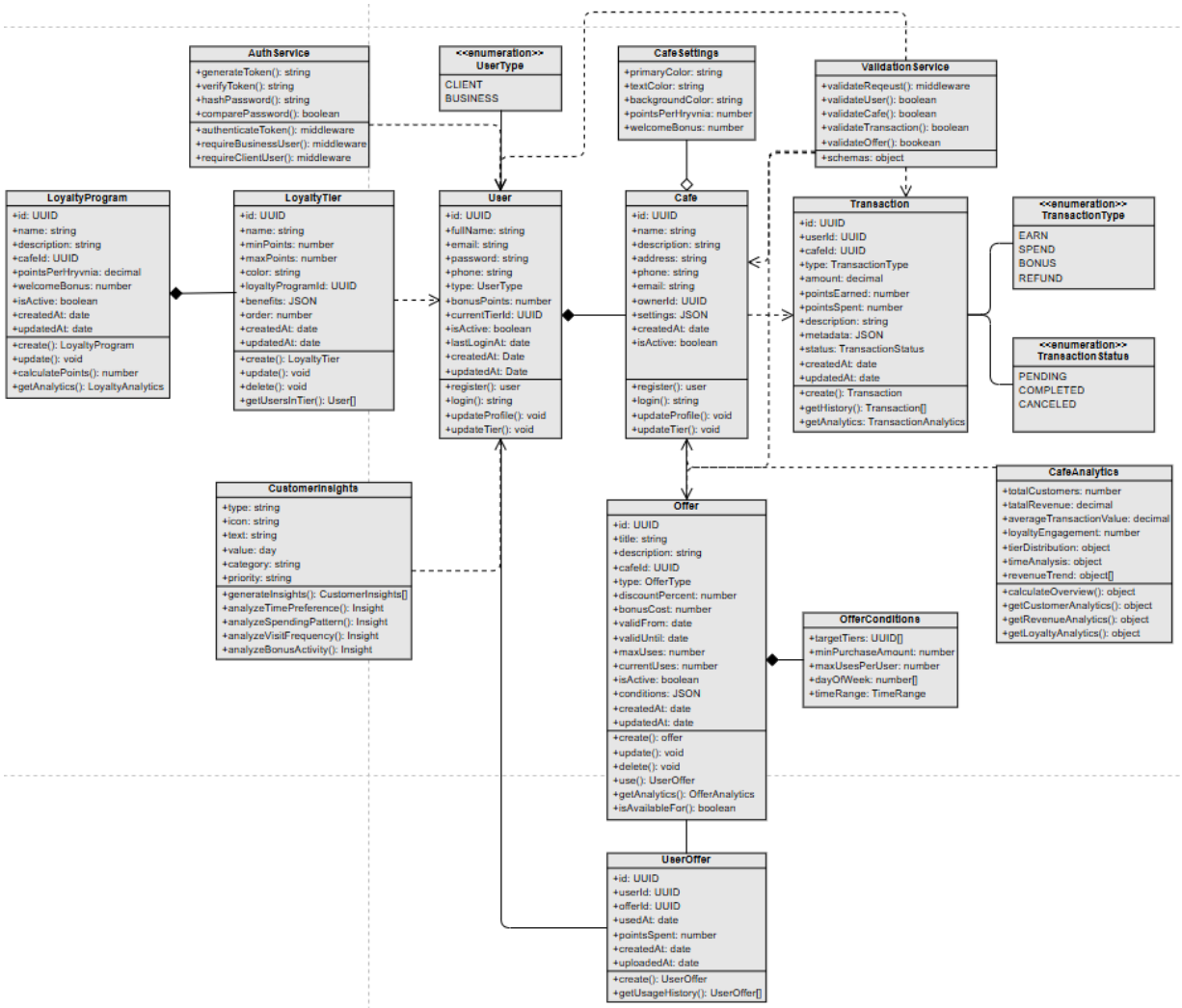

Рис. Б.2 – Діаграма класів

ДОДАТОК В – Диск з роботою