Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій і технічного захисту інформації
(повна назва)

Кафедра Радіотехнологій інформаційно-комунікаційних систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти **другий (магістерський)**

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ТА АНАЛІЗУ ТОВАРУ В МЕРЕЖІ ЗАКЛАДІВ ХАРЧУВАННЯ

(тема)

Виконав:
студент **II** курсу, групи **АПСм-22-1**
Хижняк І.О.
(прізвище, ініціали)
Спеціальність **126 Інформаційні системи та технології**
(код і повна назва спеціальності)
Тип програми **освітньо-професійна**
Освітня програма **Архітектурне проєктування інформаційних систем**
(повна назва освітньої програми)
Керівник **Доцент Сайківська Л. Ф.**

(посада, прізвище, ініціали)

Допускається до захисту
В.о. зав. кафедри РТІКС _____ **Зарудний О.А.**
(підпис) (прізвище, ініціали)

2024 р.

Не містить відомостей заборонених для відкритого публікування

Керівник Сайківська Л.Ф.

Студент Хижняк І.О.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій і технічного захисту інформації

Кафедра Радіотехнологій інформаційно-комунікаційних систем

Рівень вищої освіти другий (магістерський)

Спеціальність 126 інформаційні системи та технології
(код і повна назва)

Тип програми Освітньо-професійна

Освітня програма Архітектурне проєктування інформаційних систем
(повна назва)

ЗАТВЕРДЖУЮ:
В.о. зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Хижняку Івану Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Проєктування інформаційної системи обліку та аналізу товару в мережі закладів харчування
затверджена наказом по університету від «28» 12 2023 р. № 1514 Ст
Термін подання студентом роботи (проекту) 10.01.2024

2. Вихідні дані до роботи (проекту) . розробити інформаційну систему, яка дозволяє поєднати кілька закладів харчування, забезпечити доступ до центрального складу адміністраторів кожного закладу харчування, включеного у мережу.

3. Перелік питань, що потрібно опрацювати у роботі:
Вступ
1 Опис структури ресторанного бізнесу
2 Розробка структури мережі закладів харчування
3 Проєктування архітектури інформаційної системи
Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів)
Слайди презентації

6. Консультанти розділів роботи (проекту)

Найменування Розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Технічний	доц. Сайківська Л.Ф.		

7. Дата видачі завдання 20.10.2023

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи (проекту)	Термін виконання етапів проекту (роботи)	Примітка
1	Аналіз і огляд літературних джерел	25.10.2023	виконано
2	Опрацювання питань про технічні особливості закладів харчування	01.11.2023	виконано
3	Опис структури ресторанного бізнесу	05.11.2023	виконано
4	Створення архітектури інформаційної системи.	10.11.2023	виконано
5	Створення бази даних, та її опис	12.12.2023	виконано
6	Проєктування системи обліку для мережі закладів громадського харчування	20.12.2023	виконано
7	Висновки, оформлення документу	10.01.2024	виконано
8	Підготовка до захисту	10.01.2024	виконано

Студент Хижняк І.О.
(підпис)

Керівник роботи (проекту) _____ доц. Сайківська Л.Ф.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Кваліфікаційна робота магістра складається з пояснювальної записки, що містить 60 сторінок тексту, 11 рисунків, 10 таблиць, 9 джерела посилання і 5 додатки.

АНАЛІЗ ДАНИХ, ОБЛІК ТОВАРІВ, АРХІТЕКТУРА ІНФОРМАЦІЙНОЇ СИСТЕМИ, РЕСТОРАННА СПРАВА В ІС

Об’єктом дослідження є методи і засоби, які використовуються для обліку товару в мережі громадського харчування

Метою даної кваліфікаційної роботи є розробка системи з даними, яка дозволяє автоматизувати процес обліку в мережі. Розроблена структурна схема, архітектура інформаційної системи, створення та опис бази даних з авторизацією адміністратора закладу, для входу в обліковий доступ, та отримання доступу даних. Внесення змін в базі даних, облік товару по всій мережі.

ABSTRACT

The mastr's thesis consists of an explanatory note containing 60 pages of text, 11 figures, 11 tables, 9 references and 5 appendices.

DATA ANALYSIS, GOODS ACCOUNTING, INFORMATION SYSTEM ARCHITECTURE, RESTAURANT BUSINESS IN IS

The object of research is the methods and tools used to account for goods in a catering network

The purpose of this qualification work is to develop a data system that allows you to automate the accounting process in the network. A structural diagram, architecture of the information system, creation and description of the database with the authorization of the institution's administrator to log in to the account access and get access to the data. Making changes in the database, accounting of goods throughout the network.

ПЕРЕЛІК СКОРОЧЕНЬ

ІС - інформаційна система

БД – База даних

СУБД (ще як СКБД) – Система управління (керуванням) базами даних

ЦС – Центральний склад

JSON – JavaScript Object Notation

ЗМІСТ

ВСТУП

Ресторанний бізнес в Україні на сьогоднішній день відіграє важливу роль у підтримці економіки країни. Навіть у тяжких умовах, статистика показує, що кількість зареєстрованих ФОП більше ніж ті, які закриваються, а ті хто на ринку вже не перший рік, розвивають мережі не тільки в своїй області, а й за їх межами. У 2023 р. позитивна динаміка також спостерігається майже в усіх областях Західної та Центральної України: у Закарпатський області у працюючих закладах кількість транзакцій зросла на 17%, у Кіровоградський області — на 16% та у Чернівецький — на 14% у порівняні з попереднім роком.

На підставі аналізу діяльності ресторанних закладів, які успішно відновили свою роботу, були виявлені наступні проблеми, з якими ці підприємства стикалися:

– дефіцит персоналу внаслідок міграції або мобілізації;

– зменшення обігу клієнтів через скорочення їх кількості на середній рівень 50%, порівняно з попереднім періодом;

– підвищення вартості або недоступність деяких продуктів, що призвело до скорочення асортименту страв;

– втрата актуальності ряду форматів, таких як корпоративні вечірки, дні народження, дитячі свята тощо; значний спад попиту на кейтеринг;

– скорочення годин роботи ресторанів через введення комендантської години, що призвело до значних втрат в найбільш прибутковій частині ресторанного бізнесу – вечірній період.

Окрім цього, збільшується обсяг інформації, яку треба прорахувати, та обробити. Але багато підприємців навіть не замислюються про автоматизацію своїх закладів, та поліпшення роботи своїх закладів загалом.

Автоматизація допоможе запобігти проблемам в розрахунках та обліку, який має перевірятись як мінімум раз на місяць, але враховуючи те, що здебільшого розрахунки виконуються людиною, не рідко бувають помилки, або якісь інші недоліки.

У сучасних умовах економічної діяльності, де панує нестабільне середовище та зростаюча конкуренція, висуваються високі вимоги до ефективності та якості управлінських рішень на всіх рівнях підприємства.

Для успішного прийняття рішень необхідне володіння актуальною інформацією щодо стану та тенденцій розвитку бізнесу, використання методів та інструментів інтелектуального аналізу даних. Крім того, зазначається зростання обсягу інформації, яку необхідно враховувати при прийнятті найбільш обґрунтованих рішень.

Мета кваліфікаційної роботи: Підвищення автоматизації процесів у мережі закладів харчування, за допомогою якої зростає швидкість обробки замовлень та спрощується сервіс для клієнтів мережі, в закладах громадського харчування.

1 ОПИС СТРУКТУРИ РЕСТОРАННОГО БІЗНЕСУ

Заклади громадського харчування є прибутковим сектором в економіці, але для правильної побудови треба враховувати багато факторів, серед них інформаційна система відіграє дуже важливу роль, без якої система не буде працювати коректно, особливо для мережі.

Ресторанний бізнес створює умови для досягнення соціальних цілей розвитку туризму. Люди потребують не тільки в насиченні їжею, а й в спілкуванні один з одним. Ресторани – одне з небагатьох місць на Землі, де працюють всі наші органи чуття, викликаючи загальне відчуття задоволення.

Ресторан – це заклад ресторанного господарства з різноманітним асортиментом продукції

До сфери ресторанного бізнесу входять такі типи: ресторан, кафе, бар, їдальня, закусочна. Ієрархічна структура таких закладів представлена на рисунку 1.1. [1]

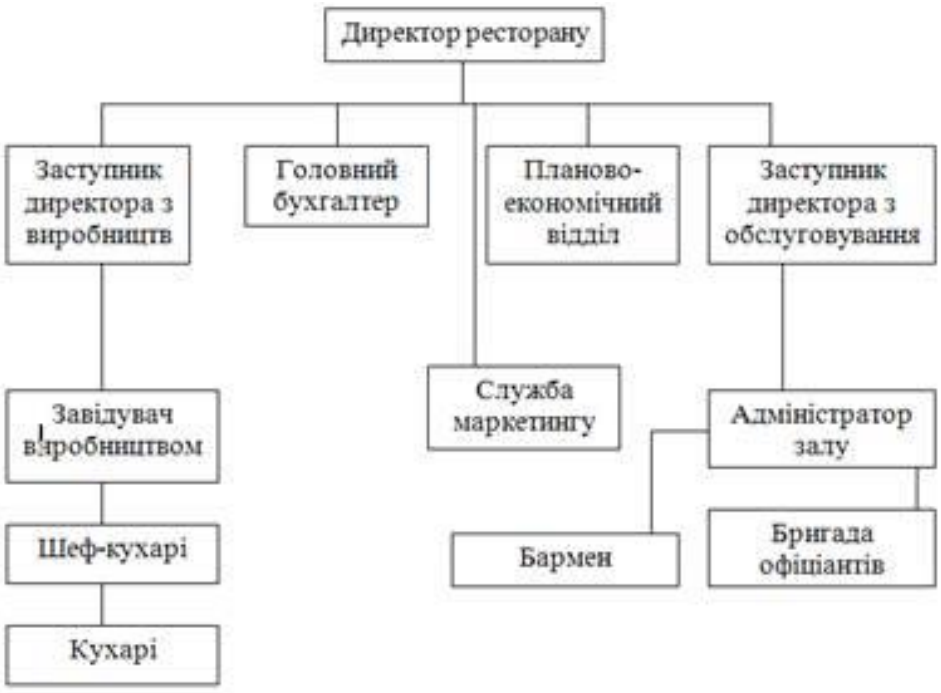

Рисунок 1.1 – Ієрархічна структура ресторану

Представлений рисунок 1.1 показує структуру ресторану. Як можна помітити керує директор(-и), в основному контролюють процес “життєдіяльності” закладу, дисципліну, наявність продукції, тощо, далі ідуть вже окремі підрозділи:

– заступник директора та адміністратор закладу, які стежать за порядком залі, їм підкоряється рядовий склад працвників, офіціанти та бармени;

– бухгалтерія має головного бухгалтера, який відповідальний за всіма звітами, та бухгалтери які підпорядковуються йому;

– планово-економічний займається розрахунком та керівництвом у сфері фінансових ресурсів;

– служба маркетингу відповідає за піар кампанію, рекламу, оголошення про сам заклад, знижки, акції тощо;

– виробництво очолює мережі бренд-шеф, контролює процес роботи харчування в усіх закладах, шеф-кухар контрою закупку, анявність продуктів на складі в пеному закладі.

У ресторанному бізнесі зосереджена значна частина матеріально-технічної бази туристської індустрії.

Успішність діяльності ресторану залежить від багатьох факторів, починаючи від формулювання загальної філософії ведення бізнесу і закінчуючи контролем за тим, як ця філософія реально втілюється в життя.

Враховуючи всі представлені тенденції, важливо відзначити, що операції в сучасному ресторанному та кавовому бізнесі є складним процесом, який вимагає особливого підходу. Заклади цього типу можуть виживати в умовах кризи лише за умови розробки відповідної креативної стратегії, яка висвітлюватиме їх унікальні особливості та виділятиме серед конкурентів. Вибір креативної стратегії буде визначати кроки у рекламній кампанії. Важливо зауважити, що креативна стратегія повинна відображатися не лише у рекламі, але й в самій атмосфері закладу. Створення успішного бізнесу передбачає аналіз сильних і слабких сторін конкретної стратегії. Щодо розвитку ресторану чи кав'ярні, рішення про те, які пріоритети варто реалізувати в першу чергу,

передбачає диференціацію продукту закладу. Це може включати додавання до меню нових кухонь.

Але в складних умовах, власникам доводиться економити практично на всьому, щоб протриматись, та не закрити заклад, тому скорочення персоналу це стала вимушеною мірою. Персонал скорочений, але потреба в праці нікуди не поділася. І, як результат, ті хто залишився на роботі навантажені більше. Тому потреба в автоматизації тільки зростає. У зв'язку з цим необхідно приділяти належну увагу організації харчування як працівників, так і покупців, що користуються послугами цих підприємств.

1.1 Опис процесу діяльності закладів харчування

Виробничо-торгівельна структура закладів ресторанного господарства – це склад усіх його підрозділів із указівкою зв’язків між ними.

На виробничу структуру закладу ресторану впливають такі фактори:

1. асортимент продукції, що випускаються, ступінь їх ґатунку;
2. обсяг виробництва та реалізації;
3. потужність (місткість залів);
4. рівень та форми спеціалізації й кооперування.

Виробнича структура підприємств ресторанного господарства може характеризуватися як технологічна, коли окремі цехи спеціалізуються на виконанні певної частини технологічного процесу, тобто створюються за принципом технологічної однорідності. [2]

При цеховій структурі основним виробничим підрозділом є - цех.

За всіма цехами відповідальним виступає саме шеф-кухар, який відповідає за наявність певних продуктів та дотримання санітарно-гігієнічних норм, включно: термін зберігання продукту, температура в різних умовах. Він наділяється певною виробничо-господарською самостійністю, одержує єдине завдання, яке регламентує обсяг виконуваних робіт.

У цеху здійснюється оперативний облік. Цехова структура застосовується на заготівельних підприємствах ресторанного господарства. Основним документом, що визначає взаємини та обов'язки постачальників та закладів ресторанного господарства, які виступають у ролі споживачів, є накладна.

На рис. 1.2 представлена USE CASE діаграма, яка відображає відношення між акторами (персонал закладу та клієнт) та прецеденти.

Діаграма станів ілюструє роботу системи, яка очікує наступного клієнта. У разі появи нового клієнта система виводить меню, приймає замовлення та одночасно контролює час очікування. Після цього можуть виникнути два стани: завершення оформлення замовлення та запис даних у базу даних. Після завершення оформлення замовлення система переходить до стану виконання замовлення.

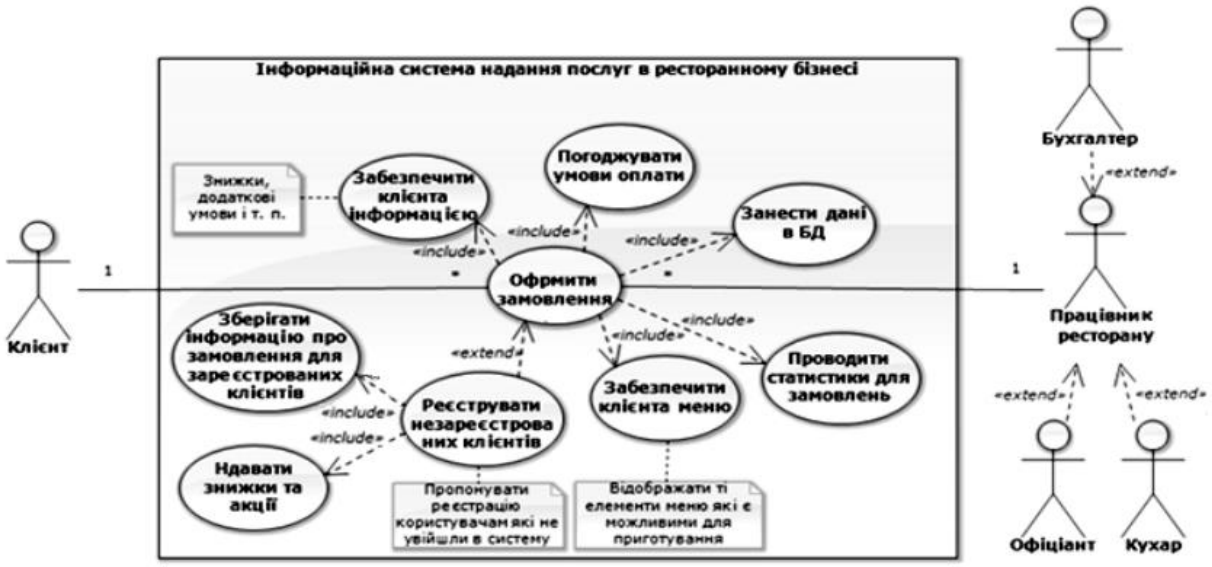

Рисунок 1.2 – Діаграма надання послуг в ресторані

Іншими можливими станами є внесення даних до бази даних, завершення виконання замовлення та повідомлення клієнта про початок виконання замовлення. Після завершення виконання замовлення система вносить дані до бази даних, повідомляє клієнта про завершення виконання та очікує завершення замовлення. Потім відбуваються переходи до станів внесення даних до бази даних та завершення замовлення.

На рис. 1.3 відображається технологічна карта, за якою персонал обробляє замовлення, всі продукти які представлені на рисунку розраховані на певну кількість страв (зазвичай на одну), як тільки вибивається страва в терміналі та замовлення передано на кухню, всі позиції зі стовпчиків “Норма кг, ціна, сума”, автоматично списуються та вираховуються з загальної кількості продуктів на складі, що дає можливість для більш чіткого моніторингу обліку товарів.

ТОВ «Кафе «Мрія»
(заклад, підприємство)

Код за ЗКУД 0903102

КАЛЬКУЛЯЦІЙНА КАРТА № 15
«12» липня 2017 року

Найменування страви *Салат із білокачанної капусти та яблук*

Номер за збірником рецептур *1.11*

	Порядковий номер калькуляції і дата її затвердження	№ 1 12 липня 2017 року		
№ з/п	Найменування продуктів	Норма, кг	Ціна, грн	Сума, грн
1	Капуста білокачанна свіжа	8,14	10,00	81,40
2	Яблука свіжі	2,27	22,00	49,94
3	Цибуля ріпчаста	1,19	12,00	14,28
4	Цукор	0,20	17,50	3,50
5	Оцет 9 %	0,30	8,00	2,40
6	Сметана	1,50	40,00	60,00
7	Сіль	0,20	3,50	0,70
Загальна вартість продуктів за первісною вартістю, грн				212,22
Націнка (200 %), грн				424,44
Загальна вартість набору продуктів на 100 страв, грн				636,66
Ціна продажу однієї страви, грн				6,37 (у тому числі ПДВ — 1,06)
Вихід у готовому вигляді однієї страви, г				100

Завідувач виробництва *Юхно* Юхно А. Є.
Калькуляцію склав *Дяченко* Дяченко Н. Г.
ЗАТВЕРДЖЕНО
Керівник закладу (підприємства) *Орлов* Орлов О. В.

Рисунок 1.3 – Технологічна(калькуляційна) карта страви

Діаграма станів, представлена на рис. 1.4 відображає систему, яка очікує наступного клієнта. При появі нового клієнта система відображає меню, приймає замовлення та одночасно перевіряє час очікування. Після цього можливі два стани: завершення оформлення замовлення та внесення даних до бази даних.

Після завершення оформлення замовлення система переходить до стану виконання замовлення. Іншими станами є внесення даних до бази даних, завершення виконання замовлення та повідомлення клієнта про початок виконання замовлення.

Після завершення виконання замовлення система вносить дані до бази даних, повідомляє клієнта про завершення виконання та очікує завершення замовлення.

Потім відбуваються переходи до станів внесення даних до бази даних та завершення замовлення.

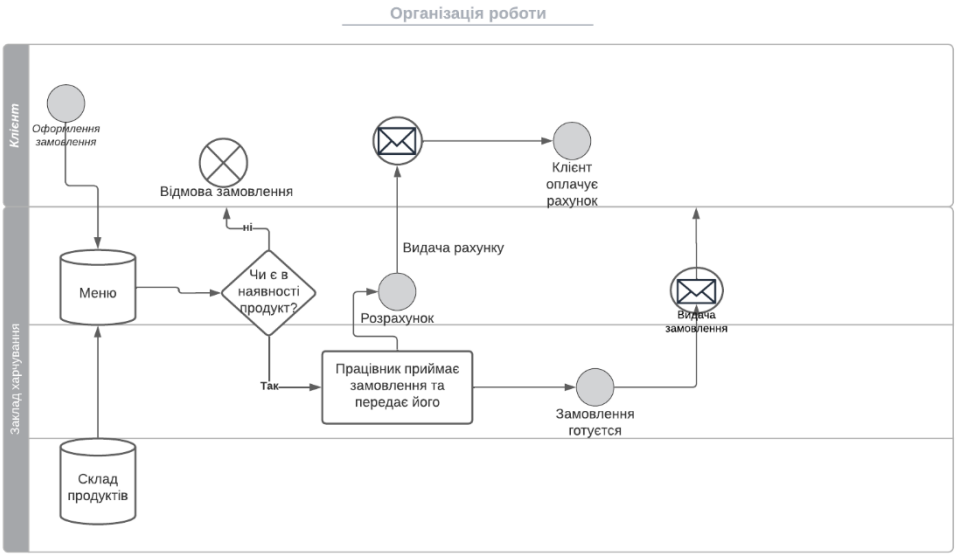

Рисунок 1.4 – BPMN діаграма бізнес процесу обслуговування клієнта

Заклади громадського харчування можуть пережити період кризи лише за умови розробки відповідної креативної стратегії, яка визначатиме їх унікальні особливості і виділятиме серед конкурентів. Вибір креативної стратегії буде визначати подальші кроки у рекламній кампанії. Важливо підкреслити, що креативна стратегія повинна відображатись не лише у рекламі, але й у самій атмосфері закладу. Успішне створення бізнесу передбачає аналіз сильних та слабких сторін обраної стратегії. Щодо розвитку ресторану чи кав'ярні, рішення про те, які пріоритети слід реалізувати в першу чергу, передбачає диференціацію продукту закладу.

Отже, за допомогою такої автоматизації можна отримати систему яка автоматично визначає витрати інгредієнтів по кожній страві і розраховує собівартість. Тому інформаційні системи значно полегшують і роблять більш суворим ведення обліку товарів, дозволяють зменшити випадки недостачі продуктів через неуважність при їх обліку та формують меню продукції в автоматичному режимі. Завдяки автоматизації спрощується процес обліку та коректне відображення діючого меню в закладі.

1.2 Переваги та недоліки автоматизації закладів громадського харчування

Ресторан, так само як і будь-яке підприємство, має приносити прибуток, і для цього всі процеси в закладі повинні бути організовані та керуватися ефективно. Автоматизація багатьох аспектів може значно полегшити роботу як персоналу ресторану, так і самого підприємства громадського харчування. Через необхідність постійно враховувати рух грошових коштів, облік продуктів, замовлень та потоку клієнтів, автоматизація стає необхідним інструментом для грамотного управління ресторанним бізнесом.

Проте важливо розуміти, що впровадження автоматизації ресторанного бізнесу само по собі не гарантує миттєвого прибутку. Система автоматизації є лише зручним інструментом в руках керівника, який повинен грамотно використовувати її можливості.

Загалом, автоматизація повинна сприяти зменшенню витрат ресторану, оптимізації робочих процесів і підвищенню прибутковості підприємства. Контроль над персоналом, економія та продуктивність праці є кількісними аспектами, які можна виміряти і оцінити, що дозволяє підтримувати ефективність системи. Крім того, автоматизація дозволяє ефективно контролювати персонал, що є важливим в галузі громадського харчування, де проблеми з крадіжками та зловживаннями можуть суттєво впливати на прибуток закладу.

Автоматизація є конкретним рішенням для вирішення проблем у будь-якій компанії. Питання про вартість автоматизації відкриває можливість розгляду таких аспектів, як вартість помилок при прийманні товарів на складі та під час інвентаризації, вартість недобросовісних дій персоналу та вартість незадоволених клієнтів, які стикаються з низькою якістю обслуговування та затримками.

Автоматизація дозволяє ефективно економити до 20% прибутку, уникати крадіжок і забезпечувати більший контроль над процесами. І в умовах економічної нестабільності, автоматизація стає ще більш актуальною, прискорюючи відновлення витрат та оптимізуючи робочі процеси.

1.3 Огляд існуючих систем

На сучасному ринку праці представлені різноманітні системи автоматизації.

Розглянемо пару прикладів існуючих сервісів обліку:

1. Dilovod - Український онлайн-сервіс ведення обліку та подання звітності для підприємців. Облік коштів, взаєморозрахунків, закупівлі, продажу, склад, простий кадровий облік та розрахунок зарплати, здавання звітності. [3]

2. POS Sector – Дуже відома програма для автоматизації в сфері обслуговування, може так само вести облік товару, формування замовлень, Офіціант набирає замовлення та відправляє його на кухню (або кухні) та бар. Кухня отримує замовлення у вигляді роздрукованого на термопринтері чека, або на Кухонний Монітор, моніторинг кількості замовлень і таке інше. [4]

На сьогоднішній день, ці системи користуються великою популярністю в закладах, їм надають перевагу більшість засновників, бо скорочують витрати, та дають можливість для більш зручного керування товарів, але не для мережі закладів, всі ці системи об’єднує одна проблема, вони локальні в кожному

закладі. Тому тема розробки інформаційної системи (ІС) для мережі закладів харчування є актуальною.

Представлена кваліфікаційна робота присвячена розробці ІС обліку та аналізу товару мережі закладів харчування, яка полегшує не тільки моніторинг ресурсів, а й дає змогу скоротити витрати.

2 РОЗРОБКА СТРУКТУРИ МЕРЕЖІ ЗАКЛАДІВ ХАРЧУВАННЯ

При розгляді процесу автоматизації за основу взято ІС центрального (загального) складу (ЦС), за допомогою якого буде проводитись облік з усієї мережі, перевірка стан складу в будь якому закладі та моніторинг ресурсів.

Зазвичай автоматизація діяльності ресторанних закладів здійснюється за допомогою створення комп'ютерної системи управління. Взаємодія між цими двома сферами, пристроями та програмним забезпеченням, дозволяє сформувати єдину базу даних підприємства. Ця база даних накопичує всю оперативну інформацію щодо діяльності закладу, що надає можливість створення єдиної систематизованої та оновлюваної інформаційної бази. На основі цієї бази даних керівник/адміністратор може приймати обґрунтовані рішення. На рисунку 2.1 зображена діаграма потоку даних, де представлений процес роботи та потоку даних з ЦС. До ЦС надходить запит на отримання позицій для певного ресторану, після чого співробітник складу перевіряє наявність продуктів, та створює накладу, в якій вказує зазначену кількість яка вказана в запиті.

На відміну від традиційного управління товарами, ці системи охоплюють всі сфери та бізнес-процеси компанії. Центральною фігурою стає не лише управління матеріальними ресурсами, але й фінансами, бухгалтерським обліком, кадровим управлінням, продажами, маркетингом, дослідженнями та іншими аспектами діяльності компанії. Функціональні відмінності, які раніше були характерні для логістики, фінансового обліку та контролінгу, зараз усуваються завдяки комплексній системі. Всі ці сфери взаємодіють між собою та користуються однією та самою базою даних.

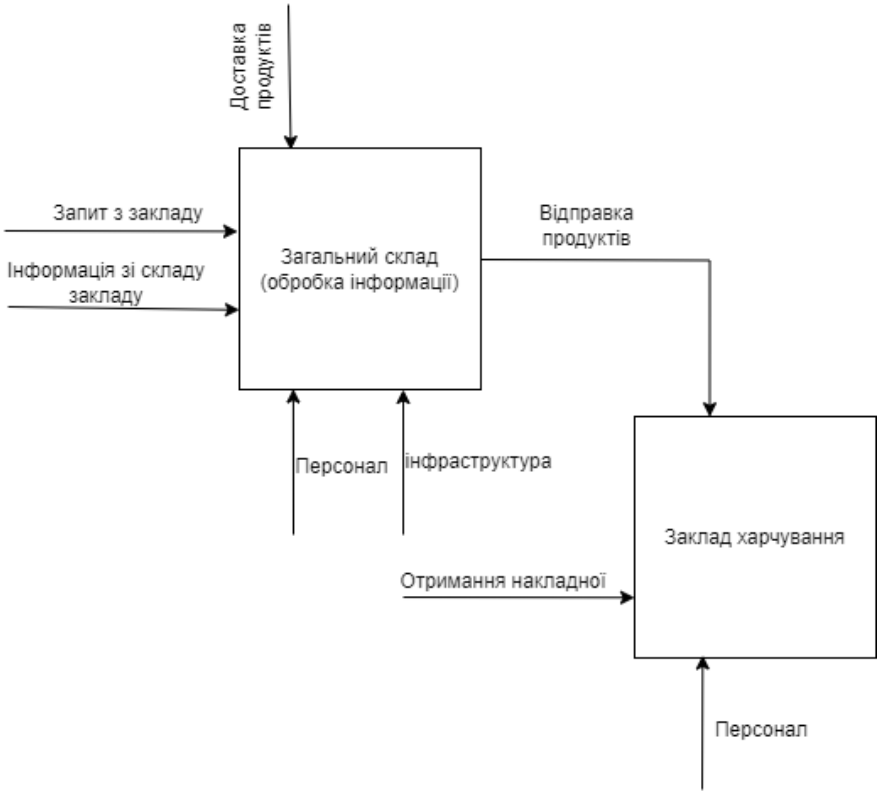

Рисунок 2.1 – Діаграма потоку даних бізнес процесу ЦС

Управління матеріальними ресурсами або планування ресурсів підприємства – це стратегічне планування, управління та контроль за всіма рухами матеріалів в компанії, а також між компанією та іншими бізнес-суб'єктами, такими як клієнти і постачальники. Концепція матеріалів охоплює широкий спектр, включаючи закупівлю товарів для перепродажу, сировину та напівфабрикати, які необхідні для виробництва. Ці матеріали потрібно закуповувати, зберігати і відправляти так, щоб завжди були наявні в достатній кількості, з необхідною якістю, в необхідний час.

На рисунку 2.2 представлена діаграма прецедентів розробляємої ІС, яка описує взаємодію між адміністратором ЦС та складу з одним із закладів харчування мережі.

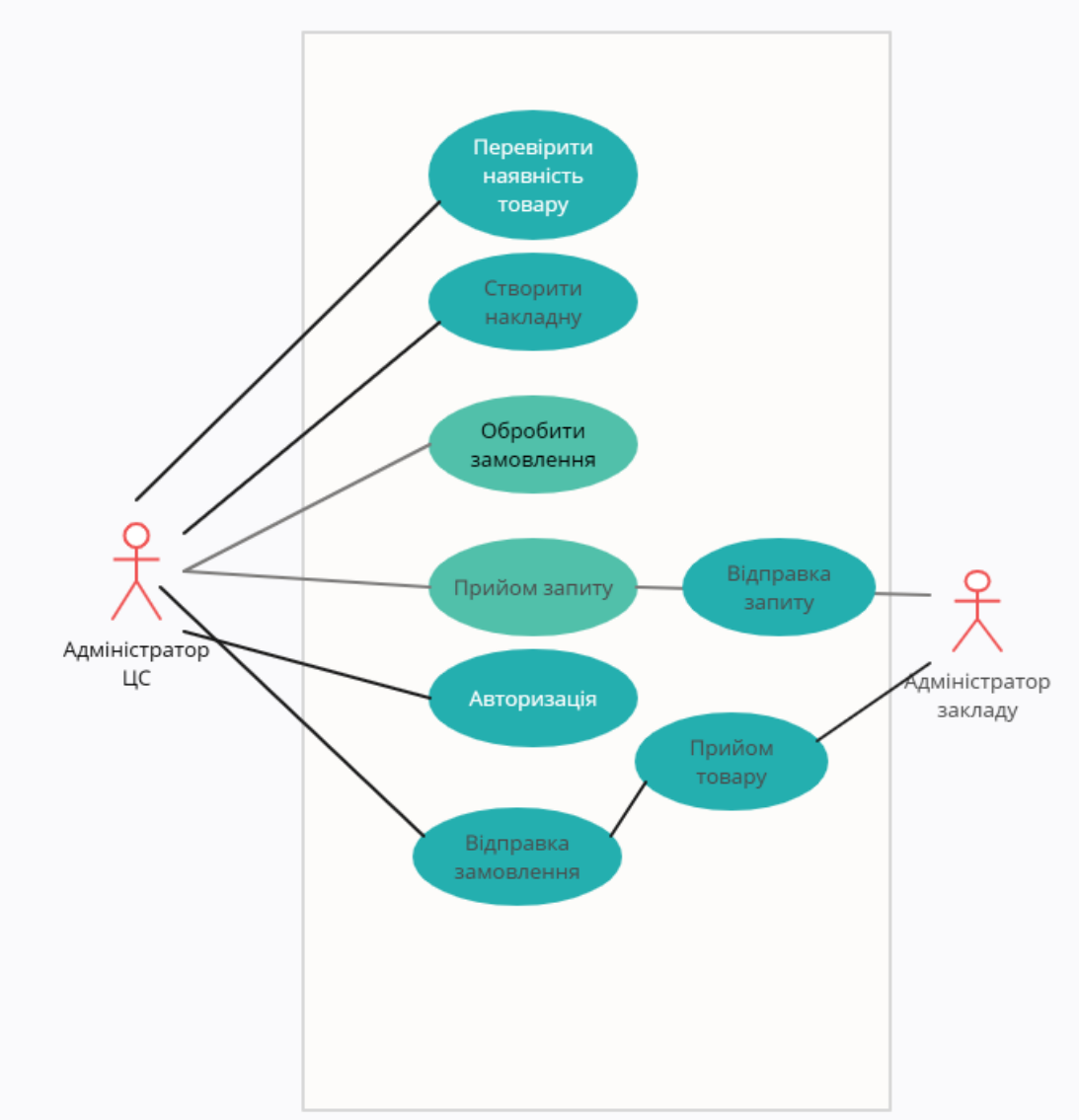

Рисунок 2.2 – Діаграма прецедентів ЦС

Можна побачити на представленому рисунку 2.2, що адміністратор саме закладу харчування робить запит на отримання певних продуктів, далі адміністратор центрального складу приймає запит, обробляє його, створює накладну за якою будет створений лист призначення (який саме заклад), відправити його до закладу, де вже адміністратор закладу його приймає.

2.3 Вибір мережевої архітектури

На сучасному етапі розвитку засобів опрацювання даних найпоширенішими є архітектури: Клієнт-сервер, та Мережева модель OSI.

Мережева модель OSI (Open systems interconnection basic reference model – Базова Еталонна Модель Взаємодії Відкритих Систем (БЕМВВС)) – це мережева модель стеку мережевих протоколів OSI/ISO .За допомогою даної моделі різні мережеві пристрої можуть з’єднуватися один з одним. Модель визначає різні рівні взаємодії систем. Кожен рівень виконує певні функції притакій взаємодії.

У літературі найчастіше зустрічається опис рівнів моделі OSI з 7-го рівня, відомий як прикладний. На цьому рівні розташовані програми, які призначені для взаємодії з користувачем та звертання до мережі. Модель OSI завершується 1 рівнем – фізичним, де встановлені стандарти, які незалежні виробники використовують для середовищ передачі даних. [3]

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних застосунків і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

– набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;

– набір клієнтів, які використовують сервіси, що надаються серверами;

– мережа, яка забезпечує взаємодію між клієнтами та серверами;

– сервери є незалежними один від одного.

Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів. [4]

При проектуванні ІС обліку та аналізу товару для закладів харчування обрана клієнт-серверна архітектура, у рамках якої деяка прикладна програма

може взаємодіяти з іншими такими програмами у рамках мережі шляхом обміну даними через сервер баз даних. [3]

На рисунку 2.1 представлена схема роботи дворівневої архітектури моделі “клієнт-сервер”.

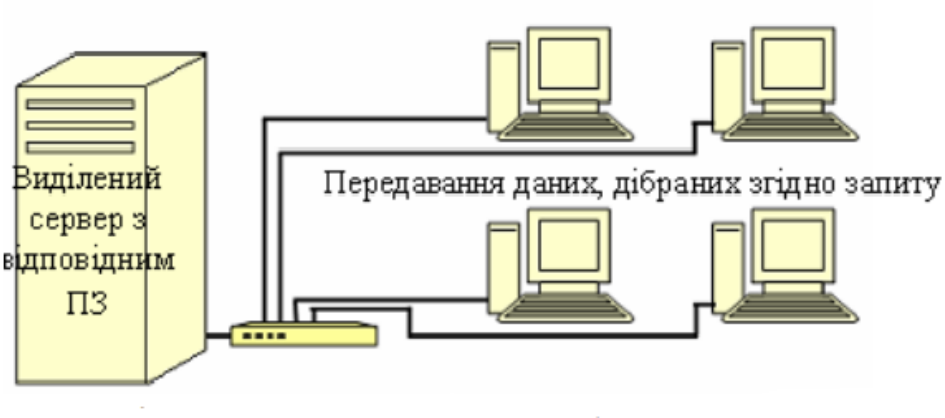

Рисунок 2.3 – Дворівнева архітектура “клієнт-сервер”

Модель клієнт-сервер використовується для розподілу завдань у мережі і приносить користувачам безліч переваг:

– централізоване адміністрування: сервер розташований в центрі мережі, що дає йому можливість здійснювати доступ до всіх ресурсів користувачів, таких як центральна база даних. Технічне обслуговування також полегшене: оновлення програмного забезпечення виконується лише на серверах, тому зазвичай клієнти цього навіть не помічають;

– економія ресурсів: оскільки дані зберігаються централізовано та завжди доступні, немає потреби у дублюванні цих самих даних на різних клієнтських комп'ютерах;

– підвищена безпека доступу: централізоване зберігання даних спрощує контроль за доступом. Перед тим як отримати доступ до конкретних даних, клієнти повинні пройти автентифікацію на сервері;

– розширювана мережа: додавання чи видалення клієнтів можливе без впливу на роботу мережі та не потребує серйозних змін. Крім того, кількість клієнтів може бути розширена практично необмежено;

– незалежність від місця розташування: централізоване зберігання даних вивільняє користувачів від обмежень щодо конкретного місцезнаходження, надаючи програмі високу гнучкість.

В архітектурі клієнт-сервер, коли клієнтський комп'ютер висилає запит на передачу даних на сервер через Інтернет, сервер приймає запитані дані, обробляє їх і повертає запитані пакети даних клієнту. Особливістю є те, що серверний комп’ютер може керувати численними клієнтами одночасно. Крім того, один клієнт може підключатися до численних серверів одночасно, при цьому кожен сервер може надавати різний набір послуг для даного клієнта.

2.4 Проектування бази даних

База даних є складовою більшості систем і зазвичай інтегрується в процес розробки програмного забезпечення.

Створення бази даних - це багатоетапний процес, що включає кілька фаз [9]:

– Зовнішня фаза: Визначення інформаційної структури, опис даних та структурування інформаційних вимог користувачів. На цьому етапі створюється неформальний опис технічної проблеми.
– Концептуальна фаза: Встановлення семантичної моделі. Мета концептуального дизайну - формалізований опис розглянутих фактів.

Існують різні підходи до створення загальної картини, але найбільш відомою є модель відносин (ER-модель). На таких діаграмах сутності зображуються прямокутниками з визначеними полями всередині, а зв'язки між сутностями позначаються лініями та кількісними мітками.

2.4.1 ER – Модель. На початковому етапі проектування бази даних, визначається, яка інформація про предметну область повинна бути включена до

бази даних. Для задоволення інформаційних потреб конкретного кола користувачів розробляється інформаційна система з базою даних, фокусована на сфері ресторанного бізнесу.

На цьому етапі виникає питання створення семантичної моделі домену, яка відображає необхідну інформацію про обрану предметну область. Однією з широко використовуваних моделей на етапі інфологічного проектування є модель сутності/відносин, також відома як "ER-модель", запропонована Пітером Ченом у 1976 році.

Моделювання предметної області базується на використанні графічних діаграм, які включають невелику кількість різнорідних компонентів. Модель "сутність-зв'язок" не визначає операції над даними і обмежується описом їхньої логічної структури. При проектуванні сутностей важливо дотримуватися принципу відкритості/закритості (OCP), який був сформульований Бертраном Меєром у 1988 році і визначає, що програмні сутності повинні бути відкриті для розширення і закриті для зміни.

Кожна сутність характеризується множиною атрибутів, що описують властивості всіх членів набору сутностей і утворюють кортежі, які задають екземпляри сутності. Для зберігання даних про клієнта використовуються атрибути, такі як ПІБ, контактна інформація та дані для авторизації.

Кожна сутність має свій унікальний ідентифікатор “id”, який однозначно визначає екземпляр класу. Після опису всіх необхідних атрибутів сутностей визначають зв'язки між ними, що відображає взаємодію між "Адміністратором закладу", "Закладом" та "Столиком".

У закладі може працювати кілька адміністраторів, і той же адміністратор може керувати менеджментом декількох закладів у мережі, тому встановлення зв'язку багато-до-багатьох є обов'язковим із обох сторін.

Адміістратор повинен бути прив'язаний до конкретного закладу, і це може бути тільки один заклад. Зворотнє відношення також справедливе: заклад може мати багато столиків, проте обов'язково має хоча б один, що призводить до встановлення зв'язку один-до-багатьох між цими сутностями обидві сторони.

На рисунку 2.4 представлена ER діаграма бази даних ЦС, через який проходить весь поток даних та моніторинг ресурсів усієї мережі.

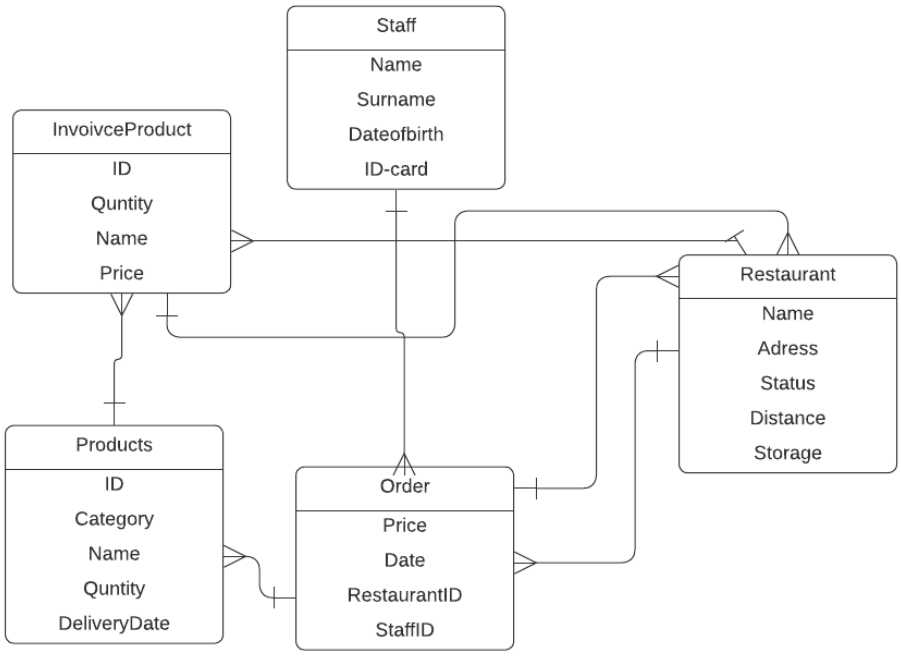

Рисунок 2.4 – ER діаграма бази даних ЦС

У ER діаграмі бази даних ЦС використовуються такі сутності:

– Staff – це персонал складу, який має такі значення, як Ім’я та Прізвище робітника, дата народження, його паспорт або ID-card;

– Order Здійснюється запит на склад з інших точок мережі, вони мають інформацію про ціну, дату замовлення, ресторан, який заповнив замовлення та номер адміна, який надіслав запит;

– Products – Містить інформацію про категорію продукту, його назву, якість (зіпсоване чи ні), дата привозу;

– InoviceProduct – Має вказану кількість товару, яка перевозиться, найменування товару, ціну собівартості продукту, не ринкову

– Restaurant - має назву ресторану, адресу, статус (відчинено/зачинено), дистанцію між ЦС та самим закладом та інформацію про склад в закладі.

2.1.2 Реляційна модель. Наступним етапом є даталогічне програмування, що означає перетворення ER-моделі в логічну модель даних, яка містить всю інформацію про предметну область. Реляційна модель є структурованим поданням даних на логічному рівні і повинна відповідати вимогам обраної системи керування базами даних (СКБД). Кожна СКБД накладає певні обмеження на реляційну модель, тому необхідно вибрати СКБД та вивчити її особливості.

Однією з важливих властивостей є тип системи за моделлю організації даних, такі як ієрархічні, мережеві та реляційні бази даних. У цій роботі використовується реляційна база даних, представлена у вигляді взаємопов'язаних таблиць.

Кількість обмежень СКБД, таких як кількість атрибутів, записів у таблиці тощо, є іншим важливим чинником. Обрана система повинна відповідати вимогам предметної області.

При переході від ER-моделі до реляційної моделі необхідно дотримуватися певних правил. Для кожної сутності створюється відповідна реляція, де вказуються атрибути, їх типи та обов'язковість зберігання в базі даних. PRIMARY KEY нової реляції визначається як первинний ключ вихідної сутності. Важливо враховувати, що не всі атрибути з моделі "сутність-зв'язок" повинні зберігатися в базі даних, оскільки деякі можуть обчислюватися або випливати з інших значень. Розглянемо перетворення сутності "Клієнт" у реляцію "User".

2.2 Обґрунтування вибору MySQL для оперування даними

Для виконання різноманітних операцій з даними частіше всього використовується база даних MySQL. Переваги використання MySQL:

– Висока продуктивність: Інноваційна організація механізму зберігання дозволяє фахівцям налаштовувати сервер MySQL індивідуально для кожного конкретного додатка. Чи буде це високоефективна система обробки

транзакцій чи масштабний веб-сайт, який обслуговує мільярди запитів щодня, MySQL може відповісти на найвибагливіші вимоги щодо продуктивності будь-якої системи. Завдяки швидким утилітам завантаження, ефективним кешам пам'яті, повнотекстовим індексам та іншим механізмам, які підвищують продуктивність, MySQL забезпечує необхідні інструменти для успішного функціонування в сучасних бізнес-системах.

– Висока доступність Надійність та постійна доступність є ключовими характеристиками MySQL, як на неї розраховують клієнти, бажаючі забезпечити неперервну роботу. MySQL пропонує різноманітні варіанти високої доступності: від швидких конфігурацій реплікації до спеціалізованих кластерних серверів, які гарантують миттєве відновлення, а також від сторонніх постачальників, що пропонують унікальні рішення з високою доступністю для сервера баз даних MySQL.

– Надійна підтримка транзакцій MySQL володіє одним із найпотужніших на ринку механізмів транзакційних баз даних. Його функціонал включає повну підтримку транзакцій ACID (атомарність, консистентність, ізольованість, довговічність), необмежене блокування на рівні рядків та підтримку транзакцій з декількома версіями, де читачі не блокують авторів і навпаки. Повна цілісність даних забезпечується завдяки посилальній цілісності, спеціалізованим рівням ізоляції транзакцій та миттєвим виявленням взаємного блокування.

Сильні сторони Web та сховища даних: MySQL визнано як стандарт для веб-сайтів з великим обсягом трафіку завдяки ефективній системі запитів, швидкій можливості вставки даних та потужній підтримці спеціалізованих веб-функцій. Ці переваги також застосовуються в області сховища даних, де MySQL може масштабуватися до терабайтних обсягів, чи то на окремих серверах, чи в масштабованих архітектурах. Інші функції, такі як таблиці основної пам'яті, B-дерева та хеш-індекси, а також стислі таблиці архівів, що зменшують вимоги до сховища на вісімдесят відсотків, роблять MySQL привабливим для веб-розробників та бізнес-аналітиків.

– Потужний захист даних Захист даних – це основне завдання для фахівців з баз даних, і MySQL пропонує вражаючі функції безпеки для забезпечення повноцінного захисту. Щодо автентифікації бази даних, MySQL використовує потужні механізми, які дозволяють доступ до сервера баз даних лише авторизованим користувачам, і є можливість блокування користувачів на рівні клієнтської машини. Підтримка SSH та SSL забезпечує безпечні та зашифровані з'єднання. Наявна гранульована структура об'єктів дозволяє користувачам бачити лише ті дані, до яких у них є доступ, а потужні функції шифрування та дешифрування даних забезпечують захист конфіденційної інформації від несанкціонованого доступу. Утиліти для резервного копіювання та відновлення, які надаються як через MySQL, так і від сторонніх постачальників програмного забезпечення, дозволяють забезпечити повне логічне та фізичне резервне копіювання.

– Комплексний розвиток додатків MySQL є найпопулярнішою у світі відкритою базою даних, оскільки вона надає вичерпну підтримку для різноманітних потреб у розробці додатків. База даних підтримує збереження процедур, тригерів, функцій, виглядів, курсорів та іншого. Для вбудованих програм існують бібліотеки плагінів для інтеграції MySQL практично в будь-яку програму. Також MySQL пропонує різні роз'єми та драйвери (ODBC, JDBC та інші), що дозволяють програмам використовувати MySQL як уподобану систему управління даними.

– Легкість управління MySQL вражає швидкістю встановлення – середній час від завантаження програмного забезпечення до завершення встановлення становить менше п'ятнадцяти хвилин, незалежно від операційної системи (Microsoft Windows, Linux, Macintosh або UNIX). Після встановлення автоматичні функції, такі як автоматичне розширення простору, автоматичний перезапуск та динамічні зміни конфігурації, значно полегшують життя адміністраторів баз даних. Крім того, MySQL пропонує повний комплект графічних інструментів управління та міграції, що дозволяють керувати,

виправляти неполадки та контролювати роботу багатьох серверів MySQL з однієї робочої станції. Вибір MySQL для цього проекту обумовлено його популярністю, відкритим кодом, надійністю та простотою управління. [6]

2.3 Підключення API

API ("Application-Programming-Interface" - Інтерфейс для прикладного програмування) надає іншим програмам можливість взаємодії з програмною системою. Він дозволяє розробникам маніпулювати обладнанням, таким як монітор або даними на жорсткому диску, не звертаючись до них напряму. Операційна система виступає як інтерфейс, який отримує запити від програм через визначені бібліотеки і передає їх апаратному забезпеченню. [13]

API отримав широке використання завдяки веб-сервісам. Це дозволяє розробникам використовувати надані інтерфейси для динамічної інтеграції контенту у свої програми. Таким чином, API-інтерфейси служать для обміну та подальшої обробки даних та контенту між різними веб-сайтами, програмами та постачальниками контенту. Крім того, вони дають можливість третім сторонам отримувати доступ до раніше обмежених пулів даних та груп користувачів.

Під час програмування API забезпечують взаємодію з даними, які передаються між різними компонентами програми, такими як модулі та програми.

У складних програмах API стають необхідною складовою. У таких випадках окремі частини програми, включаючи модулі із реальним кодом, взаємодіють виключно через API.

Таким чином, API перетворюють програмне забезпечення на машинну мову, роблячи його зрозумілим для різних компонентів. Завдяки програмним модулям і відповідним API-інтерфейсам обслуговування складних програм стає простішим. API використовують передані дані для перевірки коректності роботи модуля, сприяючи виявленню та виправленню помилок. Крім того, API-інтерфейси функціонують як постачальники даних у сфері програмного

забезпечення, що дозволяє передавати контент між різними веб-сайтами та програмами.

Під час роботи з даними використовуємо API, за допомогою якого можно інегрувати базу даних для ЦС з якокось конкретного закладу, та моніторити наявність страв, продукції на складі в режимі реального часу, причому як адміністратор ЦС так і адміністратор закладу зможуть не просто прегеглядати інформацію, а також редагувати її за необхідністтю, залишаючи лог файли для відствежування дій конкретним адміністратором.

2.4 Технічне забезпечення ЦС

Організація технічного забезпечення автоматизованого програмного рішення вимагає належного встановлення відповідного програмного та апаратного обладнання. Програмне забезпечення включає спеціальні програмні продукти та відповідну документацію для їх використання.

До загальносистемного програмного забезпечення належать програми, які орієнтовані на користувачів і призначені для розв'язання звичайних завдань обробки інформації. Ці програми розширюють функціональні можливості комп'ютерів та використовуються для контролю та управління процесом обробки даних.

Апаратне забезпечення включає пристрої, які формують конфігурацію комп'ютера. Виділяють внутрішні та зовнішні пристрої. Взаємодія між окремими вузлами та блоками відбувається за допомогою апаратно-логічних пристроїв, відомих як апаратні інтерфейси.

На підприємстві вже використовуються комп'ютер. Рекомендовано використовувати операційну систему Windows 10 або новішу для комп'ютерів працівників чи користувачів. Системні вимоги до ПК для роботи з системою представлені у табл. 2.1.

Таблиця 2.1 – Системні вимоги до ПК для роботи з системою

Назва компонента	Системні характеристики
Процесор	Intel core i3 – 5500
ОЗУ	8 GB
Пам’ять на диску	512 gb SSD
Операційна система	Windows 10 +
Монітор	IPS 60 ghz

2.5 Обмеження доступу користувачів

Одним з підходів до авторизації та аутентифікації в ASP.NET Core є механізм аутентифікації та авторизації за допомогою веб-стандарту, який визначає спосіб передачі даних про користувача у форматі JSON у зашифрованому вигляді - JWT-токенів.

JWT-токен складається з трьох частин:

1. Header – об’єкт JSON, який містить інформацію про тип токена та алгоритм його шифрування;

2. Payload – об’єкт JSON, який містить дані, потрібні для авторизації користувача;

3. Signature – рядок, який створюється за допомогою секретного коду, Headera та Payload. Цей рядок служить для верифікації токена [5]

Деякі співробітники закладів харчування (адміністратор складу, тощо) мають розширені дозволи, що надають їм доступ до додаткових видів діяльності чи можливість переглядати розширену інформацію.

Для зберігання сесії та ідентифікації користувача у системі використовується підхід, заснований на Bearer token. Сесія є контекстом, що встановлюється між користувачем і сервером під час взаємодії, дозволяючи зберігати стан інформації про користувача протягом його візиту.

Bearer token - це формат токена авторизації, який передається в заголовку або тілі запиту. Цей токен містить інформацію про авторизованого користувача,

таку як його ідентифікатор, ролі або дозволи, а також інші відповідні дані. Токен підписується цифровим ключем для гарантії його цілісності.

Однією з переваг використання Bearer token є те, що вся необхідна інформація про користувача знаходиться безпосередньо в токені, уникнення необхідності додаткових запитів до бази даних. Це забезпечує швидкий та ефективний доступ до даних про користувача без додаткового навантаження на сервер. [5]

Крім того, використання Bearer token гарантує безпеку авторизації, оскільки токен підписується цифровим ключем, і тільки сервер, що має відповідний ключ, може перевірити та розшифрувати токен. Це запобігає можливості редагування або підміни токену користувачем, забезпечуючи, що лише авторизований користувач може мати доступ до власних особистих даних та виконувати авторизовані дії на сайті.

Під час процедури аутентифікації, коли користувач успішно увійшов до системи, використовуючи свої облікові дані, йому повертається веб-токен у форматі JSON. Оскільки токени є чутливою інформацією, важливо виявляти особливу обережність для запобігання проблемам безпеки. Загалом, важливо не зберігати токени довше, ніж це необхідно.

У деяких випадках це може бути механізм авторизації без збереження стану. Захищені маршрути сервера перевірять наявність допустимого JWT в Authorization заголовку, і якщо він присутній, користувачеві буде дозволено доступ до захищених ресурсів. Якщо JWT містить необхідні дані, потреба запитувати базу даних для певних операцій може бути зменшена, хоча це не завжди так. [5]

На рисунку 2.5 показано діаграму використання JWT для доступу до API або ресурсів.

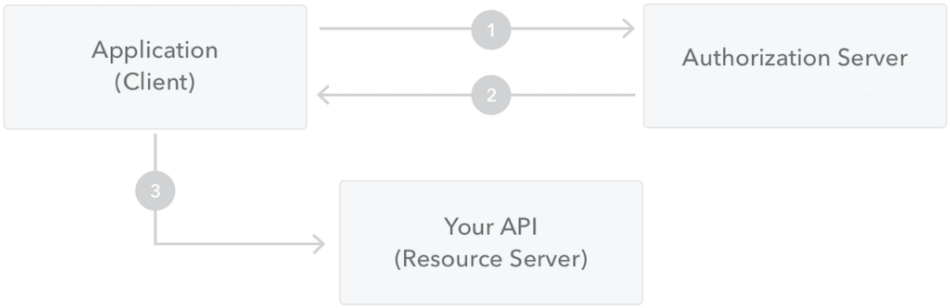

Рисунок 2.5 – Діаграма авторизації з використанням JWT

Розроблена інформаційна система використовує різноманітні методи та засоби для контролю продукції. Вона призначена для автоматизації операцій обліку продуктів харчування у галузі громадського харчування. Запропоновано впроваджувати інформаційну систему в мережах закладів харчування, таких як кафе, ресторани, заклади швидкого харчування тощо. Функціонал інформаційної системи у ресторанному бізнесі включає:

1. облік продуктів складу, включаючи додавання, відображення та видалення продуктів, контроль за списанням та відображенням кількості продуктів;
2. облік персоналу, з можливістю пошуку та внесення інформації про працівників;
3. створення та редагування рецептів страв, напоїв і меню, включаючи додавання, видалення та редагування елементів;
4. забезпечення системи звітності;
5. забезпечення терміналів офіціанта, кухні та клієнта для оптимізації взаємодії та забезпечення комфорту.

Впровадження інформаційної системи в ресторанний бізнес сприятиме полегшенню обліку товару на складі, контролю зіпсованих продуктів, що в свою чергу сприятиме підвищенню прибутковості та продуктивності.

3 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1.1 Розробка функціональних вимог до ІС

На першому етапі проектування архітектури ІС необхідно визначити функціональні вимоги до системи. Вони визначають, як система повинна виконуватися взаємодія у проєктуємій системі. Це включає обчислення даних, обмін і керування інформацією та інші конкретні функції, які система повинна виконувати.

Оскільки у системі передбачено декілька типів користувачів, необхідно визначити функціональні вимоги для кожного окремо.

Авторизація адміністратора в системі здійснюється через введення електронної пошти та пароля.

При реєстрації потрібно вказати особисті дані, ім’я, унікальна електронна пошта, телефонний номер та підтвердити пароль. Неавторизовані користувачі можуть тільки переглядати та фільтрувати інформацію про заклади.

Система забезпечує сортування та фільтрацію закладів за різними критеріями, такими як ціна, рейтинг та популярність.

Адміністратори можуть здійснювати пошук за назвою, переглядати заклади поблизу та отримувати список схожих закладів.

– адміністратор закладу може створювати та редагувати інформацію про заклад.

– адміністратор може приховувати заклад, якщо він припиняє роботу;

– персонал закладу може переглядати інформацію про бронювання та створювати нові записи для обліку завантаженості;

– адміністратор системи має можливість створювати нові облікові записи для менеджерів закладів, які запитують доступ до системи.

Ці функціональні вимоги визначають конкретні дії, які користувачі та адміністратори можуть виконувати в системі.

3.1.2 Розробка нефункціональних вимог

Нефункціональні вимоги представляють собою SRP-документ, який визначає якість атрибутів програмного забезпечення, таких як надійність, зручність користування, стійкість до збоїв та швидкість реагування. Функціональні вимоги описують, що система має робити, а нефункціональні вимоги вказують, як це має виконуватися. Ці аспекти часто розглядаються як невід'ємні характеристики, які визначають рівень свободи для розробників чи користувачів. Нефункціональні вимоги є складними для тестування, і тому їх оцінюють суб'єктивно.

Деякі конкретні нефункціональні вимоги можна сформулювати так:

– Адміністратор може зареєструвати лише один профіль за однією електронною адресою та номером телефону, тобто ці дані повинні бути унікальними.

– Пароль повинен містити не менше 6 символів і зберігатися у базі даних у зашифрованому вигляді.

– При неправильному введенні даних під час реєстрації або авторизації користувач отримує відповідне повідомлення.

– Адміністратор може переглядати заклади на головній сторінці або сторінці розширеного пошуку, шукати заклад за іменем, сортувати та фільтрувати їх. Якщо закладів із заданими параметрами немає, то жоден заклад не відображається.

– Неавторизований користувач, який намагається перейти на сторінку, буде перенаправлений на сторінку входу.

– Авторизовані користувачі можуть бачити лише свої бронювання.

– Шеф-кухар який контролює процес роботи та наявність товарів на складі, також відповідальний за переррахунок на самому складі.

– Адміністратори можуть редагувати та приховувати лише ті заклади, які зареєстровані у їхньому профілі.

3.2 Стейкхолдери системи

Стейкхолдер - це будь-хто, хто зацікавлений у вашому проекті або з ким вам потрібно співпрацювати, щоб завершити проект. Розуміння ваших стейкхолдерів та їхніх поглядів є ключовим для успіху вашого проекту і зазвичай досягається шляхом проведення інтерв'ю із зацікавленими сторонами. [14]. Головною ціллю стейкхолдінгу – залучити найбільше зацікавлених в підтримці сторін в розробці системи, це економить час та ресурси та кількість роботи зменшується в рази. На рисунку 3.1 зображені стейкхолдери інформаційної системи.

Рисунок 3.1 – Стейкхолдери проєкту ІС

У даному випадку стейкхолдерами є:

Замовник – фізична особа яка зацікавлена в контролі обліку товарів по всій мережі.

Посередники – комерційні організації які предоставляють послуги перевезення продуктів з ЦС та по всім закладам в мшсті.

IT Спеціалісти – спеціалісти вузького напрвлення, для підтримки функціонування системи, виправлення помилок, проблем. Користувач – він же адміністратор центрального складу, який моніторить наявніть того чи іншого продукту, та обробка замовлення як тільки виникає потреба.

3.3.2 Атрибути якості

Основною ідеєю інженерних методів у програмуванні є прагнення до підвищення якості програмного забезпечення. Для досягнення цієї мети визначено методи формулювання вимог до якості програмного забезпечення, розроблені підходи до вибору та удосконалення моделей для метричного аналізу показників якості програмного забезпечення, а також розроблені методи кількісного вимірювання показників якості на різних етапах життєвого циклу програмного забезпечення. [15]. На рисунку 3.2 показані основні атрибути якості, які важливі для ІС обліку та аналізу в мережі закладів харчування. Вони включають:

1. Функіональність – Оцінюється здатність програмного забезпечення ефективно вирішувати завдання, що відповідають визначеним та очікуваним потребам користувача при визначених умовах його використання.
2. Надійність – Можливість програмного забезпечення виконувати визначені завдання протягом визначеного часового інтервалу або при вказаній кількості операцій.
3. Зручність використання- можливість легкого розуміння, вивчення, використання і привабливість ПЗ для користувача.
4. Ефективність – здатність ПЗ забезпечувати необхідний рівень продуктивності згідно з виділеними ресурсами, часом та іншими зазначеними умовами.
5. Зручність супроводу – легкість, з якою ПЗ може аналізуватися, тестуватися, змінюватися для виправлення дефектів

6. Портативність - характеризує ПЗ з точки зору легкості його перенесення з одного оточення (software/hardware) в інше.

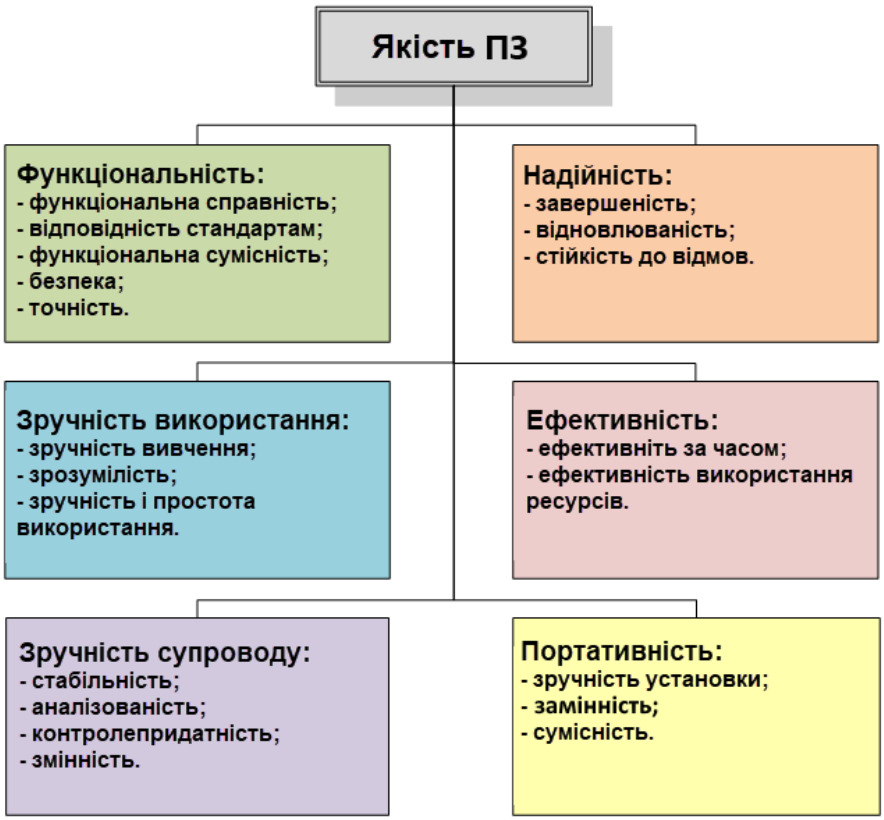

Рисунок 3.2 – атрибути якості програмного забезпечення

Їх важливість для представленої розробки представлена у таблиці 3.1.

Таблиця 3.1 – Атрибути якості ІС обліку та товарів мережі закладів харчування

Атрибут	Важливість	Пріорітет
Функціональність	M	L
Надійність	H	H
Зручність використання	M	L
Ефективність	H	M
Зручність супроводу	L	M
Портативність	L	L
Легенда: L – low M – medium, H - High		

3.4 Розробка клієнт-серверної архітектури системи

Основною метою архітектури програмного забезпечення є зменшення трудовитрат на створення та супровід системи.

Щоб досягти цієї мети, програмне забезпечення повинно бути гнучким, тобто легко змінюваним. У такому випадку складність повинна бути пропорційною масштабам змін, а не їхній формі.

Головне завдання архітектури ІС - підтримка життєвого циклу системи. Ефективна архітектура робить систему простою для освоєння, легкою для розробки, супроводу і розгортання. Кінцева мета полягає в мінімізації витрат протягом терміну служби системи та максимізації продуктивності програміста.

У цьому проекті існує розподіл на бекенд та фронтенд. Архітектура веб-застосунку буде мати чітку структуру побудови системи.

Бекенд - це програмно-апаратна частина сервісу, набір інструментів, які забезпечують бізнес-логіку застосунку. Це серверна сторона, де отримують запити від клієнта та формують відповідь на основі інформації, отриманої з бази даних.

У файлі server.js визначено основні кінцеві точки API (точки входу API), такі як '/api/users', '/api/places' та '/api/bookings'. Для кожної кінцевої точки створено власний роутер, де здійснюється підключення до бази даних. Ці роутери виконують SQL-запити відповідно до запиту клієнта та надсилають відповідь у вигляді об'єкту або масиву об'єктів. Сортування, фільтрування та спеціалізований вибір даних здійснюються за допомогою SQL, на відміну від методу, коли всі дані отримуються з таблиці, а потім обробляються за допомогою функцій певної мови програмування. Наприклад, для відображення всіх закладів типу "ресторан" сервер отримує запит від адмыныстратора, що містить набір фільтрів, включаючи фільтр за типом закладу.

Ми отримуємо лише ті заклади, які потрібно відобразити на сторінці, аналогічно процесу авторизації. У системі не зберігається в момент звернення список усіх користувачів та їхніх даних, що є небезпечним з погляду безпеки.

Здійснюється запит до бази даних для пошуку клієнта з такими даними, і при успішному запиті отримуємо усю інформацію про цього користувача.

Отже, клієнт завжди звертається на певну кінцеву точку, де здійснюється SQL-запит, який задовольняє саме той запит користувача. Таким чином, можна сказати, що більша частина бізнес-логіки реалізовується функціями мови програмування структурованих засобів.

У проекті є два package.json: один для React та інший для nodejs API. Краще мати абсолютно різні модулі Node.js для кожного з них, щоб уникнути проблем зі злиттям або іншими конфліктами модулів веб-вузлів та серверів.

Всі ресурси, створені для клієнтської частини програми, повинні знаходитись у каталозі /src/.... Файл /src/App.js є основним контейнером додатку і служить точкою входу.

Деякі сторінки складаються з декількох частин, реалізованих у каталозі /src/components. Відповідно до функціональних вимог на головній сторінці потрібно відображати усі наявні заклади. Кожен заклад представлений у вигляді карточки з головним зображенням, назвою, адресою та рейтингом. Тому було доцільно виділити цей елемент як окремий компонент Place.js. Також на головній сторінці (HomePage) використовуються інші компоненти: анімація завантаження (LoadingBox.js) та кастомне повідомлення (MessageBox.js).

Однією із широко використовуваних практик у розробці архітектури додатків на React є використання Redux. Redux є контейнером для управління станом застосунку і в багатьох відношеннях схожий на Flux. Redux не прив'язаний безпосередньо до React.js і може також використовуватися з іншими бібліотеками та фреймворками JavaScript.

Основні аспекти Redux включають:

– Сховище (store): Зберігає стан застосунку.

– Дії (actions): Представляють собою набір інструкцій, які надходять з компонентів до сховища, вказуючи, що конкретно потрібно виконати. Метод dispatch() використовується для передачі цих інструкцій у сховище.

– Творці дій (action creators): Функції, які створюють дії.

– Редюсер (reducer): Функція (або кілька функцій), яка приймає дію і відповідно до неї змінює стан сховища.

Схема взаємодії елементів архітектури Redux виглядає так: від елементів View (компонентів React) надсилається дія, яку отримує функція редюсера, що відповідає за оновлення стану сховища. Після цього компоненти React використовують оновлений стан.

Обробка дій користувача відбувається через обробники дій, які визначені у папці /src/actions. Там відбувається відправка GET та POST запитів на сервер із відповідними параметрами. У цьому проекті наявні три редюсера: placeReducer, userReducer, та bookingReducer, які відповідають за оновлення сховища у файлі store.js.

3.5 Опис бази даних одного закладу харчування в мережі.

На рисунку 3.3 представлена ER- модель бази даних для конкретного закладу мережі.

Розглянемо атрибути кожної сутності у вигляді окремих таблиць, де стовпці відображатимуть атрибути таблиці та їх характеристики.

CentralStorage – центральний склад, де зберігаються всі компоненти, необхідні для роботи в інших закладах, зберігає інформацію про переміщення між закладами, обробляє запити на доставку з інших закладів в мережі (табл. 3.2).

Таблиця 3.2 – Опис атрибуту Центральний склад

Adress	Адреса самого складу
Phone	Номер телефону
Email	Електронна пошта
InStock	Наявність продуктів
Desription	Опис зберігання
Positions	Продукти які зберігаються

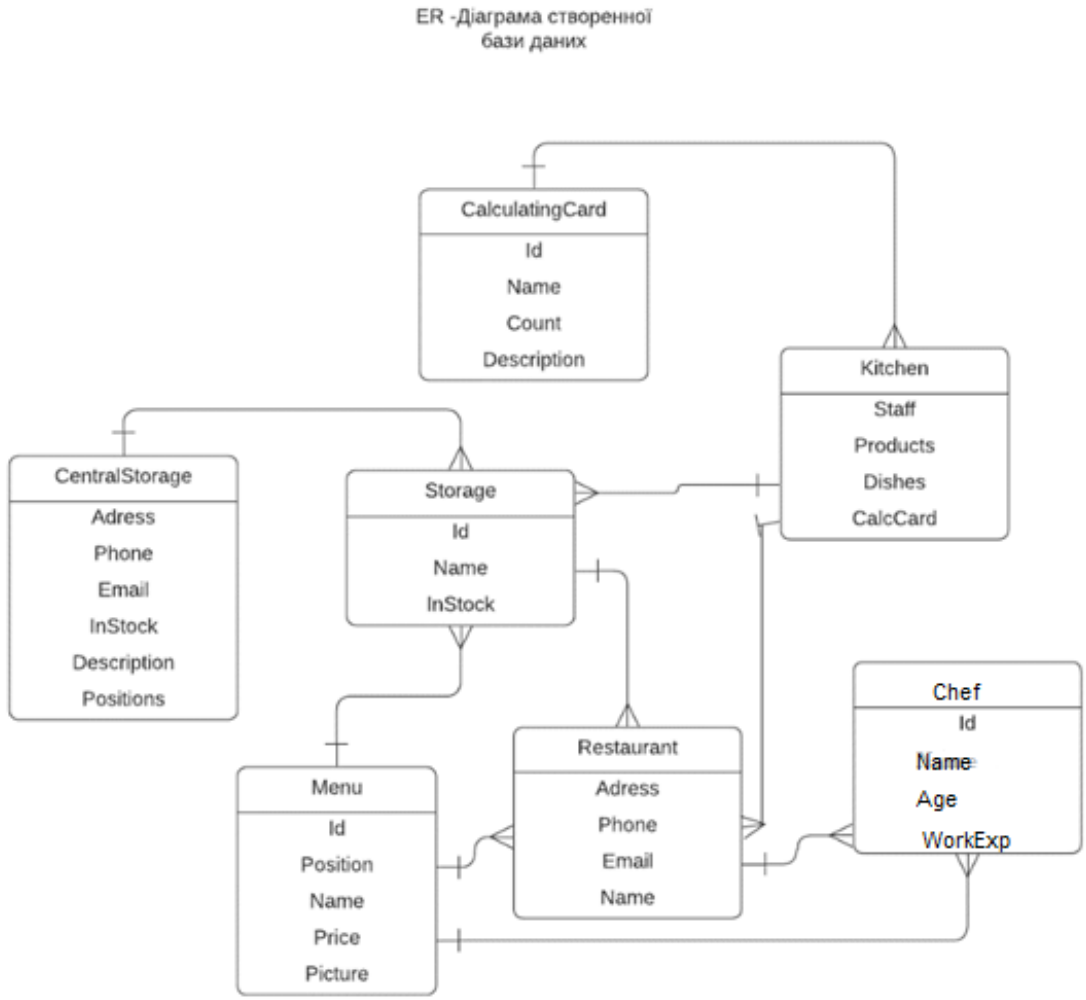

Рисунок 3.3 – ER модель бази даних одного закладу харчування

Storage - склад, який знаходиться в самому закладі, має назву продукту та його кількість, має інформацію про поточний стан продуктів на складі (кількість), відображає інформацію в режимі реального часу (табл. 3.3).

Таблиця 3.3 – Опис атрибуту склад

Id	Айді складу прив’язаного до певного закладу
Name	Найменування складу
InStock	Наявніть продуктів
Count	Кількість продуктів на складі

Menu –, має свій ID, позицію страви наприклад, може бути вид кухні (європейська, національна тощо), назву страви, її ціну та опис в якому будуть

основні інгредієнти, порівнює наявність страв на складі, якщо певного інгрдієнта не вистачає, позиція буде недоступна (табл. 3.4).

Таблиця 3.4 – Опис атрибуту Меню

Id	Унікальний ID страви
Position	Позиція страви в меню
Name	Назва страви
Price	Ціна
Picture	Фотографія самої страви

Restaurant – сам ресторан, як одиниця мережі; має свою адресу, номер телефону для зв'язку та електронну пошту, і Name – назва точки, прив'язана адреса допомогає чітко орієнтуватися по закладах, для прискорення відправки товару з ЦС (табл. 3.5).

Таблиця 3.5– Опис атрибуту Ресторан

Adress	Адреса конкретного закладу
Phone	Номер телефону
Email	Електронна пошта
Name	Назва ресторану

Kitchen – Кухня має свій персонал який зазначається як "Staff", має продукти, які бере зі складу, страви які готує та видає, та технічну карту за якою працює персонал (табл. 3.6).

Таблиця 3.6 – Опис атрибуту Кухня

Staff	Персонал який відповідає за видачу
Product	Продукти, які там є
Dishes	Страви, які виготовляються там
CalcCard	Тех. Карта за якою відбувається процес приготування

CalculatingCard - Калькуляційна карта має інформацію про страву, її технологію приготування, назви інгрідієнтів, та порційних розрахунок вказаний в графі “Count” (табл. 3.7).

Таблиця 3.7 – Опис атрибуту Калькуляційна карта

id	Унікальний айді карти
Name	Назва
Count	Розрахунок позицій
Description	Опис та технологія приготування

Chef – шеф кухар закладу, відповідальний за наявність продукті на складі, контроль та організація роботи на кухні, відповідальний за доставку від складу закладу до кухні, в базу вносятся дані про його Ім’я, Вік та стаж роботи. (рисунок 3.8)

Таблиця 3.8 – Опис атрибуту Шеф-кухаря

Id	Унікальний айді працівника
Name	Ім’я
Age	Вік співробітника
WorkExp	Стаж роботи

3.6 Розробка інтерфейсу користувача

Користувацький інтерфейс – це засіб для взаємодії між користувачем та програмним забезпеченням. У розробці програм для кінцевих користувачів важливе значення повинно бути відведено інтерфейсу, оскільки він визначає зручність та простоту використання, що напряму впливає на задоволеність користувачів.

Мапа сайту, створена на цьому етапі, відображає ієрархічну структуру сайту та надає пріоритети, лінкування та мітки сторінок.

Розробка графічного інтерфейсу для цієї системи пройшла кілька етапів:

– розробка інформаційної архітектури;

– проектування каркасу веб-сайту;

– розробка прототипу;

– перевірка ергономічності.

Відповідно до розробленої архітектури були обрані конкретні технології для кожного рівня.

React Native - це фреймворк для мобільних додатків з відкритим кодом, розроблений Facebook. Він призначений для створення додатків для платформ Android, iOS, Web та UWP, дозволяючи розробникам використовувати React разом із можливостями рідної платформи.

Основні можливості:

Write Once and Use Everywhere: На сьогоднішній день ця особливість є ключовою концепцією, яка підтримує ідею написання коду в React Native, який практично можна використовувати на кожній мобільній платформі, включаючи IOS, Android, Windows і інші. Завдяки цьому не потрібно писати окремий код для кожної платформи, збільшуючи ефективність розробки.

Мова програмування: Вибір мови програмування є ключовим фактором для розробників. React Native використовує одну з найпопулярніших і широко використовуваних мов програмування - JavaScript. JavaScript вже є однією з основних технологій Всесвітньої павутини (WWW), разом з HTML та CSS. Ця мова є простою для вивчення і використання, що робить її доступною для багатьох розробників, особливо тих, хто має досвід у веб-розробці.

Фокус користувацького інтерфейсу в React Native вкладений у сам дизайн. Цей фреймворк вражає відмінною реакцією та найвищими можливостями рендерингу на сучасному рівні.

Час розробки в React Native значно скорочений у порівнянні з іншими підходами:

– Підтримка сторонніх бібліотек завжди є плюсом, дозволяючи розробникам вільно вибирати інструменти. Це може бути і перевагою, і недоліком одночасно.

– Використання NPM для встановлення є простим і зрозумілим процесом, зокрема для тих, хто вже має досвід з JavaScript або NPM.

– В розробці React Native використовується орієнтація на GPU замість CPU, що сприяє покращеній продуктивності у порівнянні з програмами, орієнтованими на процесор.

– Функція Live Reload дозволяє одночасно змінювати код та переглядати його модифікації, що спрощує розробку.

– Мова програмування React Native, якою є JavaScript, забезпечує широкий доступ та легкість вивчення, особливо для тих, хто вже працював у сфері веб-розробки.

У порівнянні з іншими фреймворками, такими як Ionic, React Native використовує JavaScript разом із можливостями HTML та CSS, тоді як Dart є мовою, яку використовує Flutter, і вона менш популярна серед розробників.

3.7 Використання редактора коду

Для реалізації технічної частини, було використано такий редактор як Visual studio code. Visual Studio Code — це спрощений редактор коду з підтримкою таких операцій розробки, як налагодження, виконання завдань і керування версіями. Він має на меті надати саме ті інструменти, які потрібні розробнику для швидкого циклу створення коду та налагодження, і залишає складніші робочі процеси для повнофункціональнішої IDE. Основні переваги використання саме цього редактора:
Ціна: редактор абсолютно безкоштовний, не протребує якихось платних підписок, чі покупка ключів активації.

- **Крос-платформний:** з відкритим вихідним кодом і крос-платформний редактор, який працює на Windows, Linux і MacOS, так що ви можете працювати незалежно від платформи, на якій засновано ваш пристрій.
- **Intelli-Sense:** Він може виявити, якщо який-небудь фрагмент коду залишився неповним. Також загальні синтаксиси змінних та оголошення

змінних складаються автоматично. Наприклад: Якщо в програмі використовується певна змінна, і користувач забув її заявити, Intelli-sense оголосить її для користувача.

3.8 Реалізація проекту

Під час першого використання сервісу можна зареєструватися для того, щоб отримати можливість переглядати доступність товару та робити бронювання накладних. З метою обмеження доступу до конкретної інформації (наприклад, перегляд деталей закладу або заповнення форми бронювання) проводиться перевірка, чи користувач є авторизованим та чи він єдиний авторизований користувач на даному пристрої. Цю перевірку здійснюється за допомогою інформації станів Redux, збереженої у сховищі, та унікального користувацького токена.

Після успішної авторизації користувача перенаправляється на головну сторінку додатку (HomePage), де представлені всі доступні заклади у вигляді карток із зображенням, назвою, адресою, рейтингом та середньою ціною за один обід. Для отримання цієї інформації відправляється запит до відповідної API-адреси 'api/places', і після оновлення локального сховища оновлюється вміст сторінки. Після цього необхідна інформація відображається за допомогою React-компонента "Place".

Такий самий потік даних відбувається на сторінках кожного закладу і розширеного пошуку. Різниця полягає в тому, що для відображення використовуються різні сторінки, які відправляють запити на різні кінцеві точки. На сервері ці запити обробляються, виконуючи SQL-запити для отримання відповідної інформації з бази даних.

Наприклад, для отримання даних про всі заклади на головній сторінці ми відправляємо запит без параметрів. Але для отримання закладів, відсортованих та відфільтрованих за одним або декількома значеннями, ми виконуємо GET-запит наступного вигляду:

/api/places?name=${name}&category=${category}&min=${min}&max=${max}&rating=${rating}

Усі ці параметри запитуються і використовуються у SQL-запиті.

Наразі користувач може сортувати за рейтингом, середньою ціною у порядку зростання або спадання, а також використовувати фільтрацію за типом закладу, середньою ціною за один прийом їжі, середнім рейтингом та доступністю на певну дату.

Для адміністратора/директора закладу існує аналогічний функціонал з деякими відмінностями: він може здійснювати та переглядати бронювання лише для закладів, які входять до його мережі, зокрема для реєстрації бронювань, здійснених телефоном або особисто. Крім того, адміністратор має можливість змінювати інформацію про заклади та приховувати їх. У цій системі, якщо заклад стає неактивним з певних причин, керівництво може не видаляти його, а лише приховувати його для зберігання всієї документації та забезпечення цілісності даних.

Кожен елемент системи функціонує як запущений сервер на різних портах локальної мережі:

Веб-сервер працює на порті 3000.

Панель адміністратора використовує порт 4000.

Під час розробки та тестування продукту кожен сервер запущено локально на комп'ютері, на якому відбувається розробка. Кожен сервер доступний за посиланням:

http://localhost:3000,

http://localhost:4000,

а також http://localhost:8081 для різних частин проекту.

Отже, всі запити між компонентами системи відбуваються в локальній мережі, за винятком запитів до бази даних, яка розташована в "хмарі".

Взаємодія між серверами під час розробки була налаштована за допомогою методу проксі у файлі package.json. Веб-сервер панелі адміністратора має конфігураційний файл проксі, який спрямовує запити до

бекенд-сервера, і в конфігурації бекенд-сервера встановлено відповідний проксі навпаки. [6]

App.js є головним виконавчим файлом, який включає конфігурації сервера, код для підключення до бази даних та основні параметри маршрутизації, які обробляють вхідні запити до серверу. Маршрутизація визначає кінцеві точки програми (URI), які реагують на HTTP-запити від клієнтів.

– Основні типи HTTP-запитів включають:

– GET: Цей метод вимагає представлення конкретного ресурсу, і запити, що використовують GET, призначені лише для отримання даних.

– HEAD: Запитує заголовки, які повертаються, якщо вказаний ресурс буде запрошено методом HTTP GET. Це може бути використано перед завантаженням великого ресурсу для збереження пропускної здатності.

– POST: Цей метод використовується для надсилання об'єкта на вказаний ресурс і часто викликає зміну стану або побічні ефекти на сервері.

– PUT: Метод HTTP PUT створює новий ресурс або замінює представлення цільового ресурсу на корисне навантаження запиту.

– DELETE: Метод DELETE видаляє вказаний ресурс.

– CONNECT: Метод CONNECT встановлює тунель до сервера, ідентифікованого цільовим ресурсом.

– OPTIONS: Метод OPTIONS використовується для опису параметрів зв'язку для цільового ресурсу.

– TRACE: Метод TRACE виконує тест зворотного зв’язку повідомленням по шляху до цільового ресурсу.

– PATCH: Метод PATCH використовується для застосування часткових модифікацій до ресурсу.

У Express JS, методи маршрутизації задають функцію зворотного виклику, іноді її називають "функцією обробника". Ця функція викликається, коли програма отримує запит на вказаний маршрут та метод HTTP, тобто коли виявляється відповідність, вона викликає вказану функцію обробника.

3.8 Імплементація панелі адміністратора.

Адміністративна панель є працюючим сервером React, який складається з компонентів. Дані для кожного компонента отримуються за допомогою методу GET HTTP у форматі JSON і зберігаються у локальному стані компонента. Після цього ці дані виводяться у зручному та зрозумілому для користувача форматі.

Перед тим як розпочинати роботу, розглянемо рисунок 3.2 - в системі директори повинні забезпечувати чітку особисту відповідальність працівників за кожен початковий документ і достовірність інформації, що міститься в цьому документі. Цього можна досягти за допомогою виконання наступних вимог:

– Автоматизація документу, що означає введення в нього спеціальної інформації, яка одночасно ідентифікує особу, яка виконала, затвердила або зареєструвала явище господарської діяльності підприємства.

– Обмеження доступу до засобів реєстрації, щоб уникнути можливості реєстрації даних особами, які не мають відповідних повноважень.

– Виявлення кожного випадку несанкціонованого використання засобів реєстрації.

Якщо прийшов новий адміністратор, в такому випадку реєстрація буде через унікальний код, тобто просто так в системи не разеєструатись, та неможливо абсолютно кожному заходити та міняти данні.

Вигляд панелі авторизації адміністратора представлена на рисунку 3.4.

Для зміни чи оновлення даних у базі даних використовується POST HTTP запит. Файл Index.js містить усі компоненти сервера у віртуальному DOM. Файл App.js, який є основним виконавчим файлом, включає в себе підключення модулів для роутінгу та історії переміщень по сторінках, а також розташовує самі компоненти роутінгу та статичний елемент лівого меню.

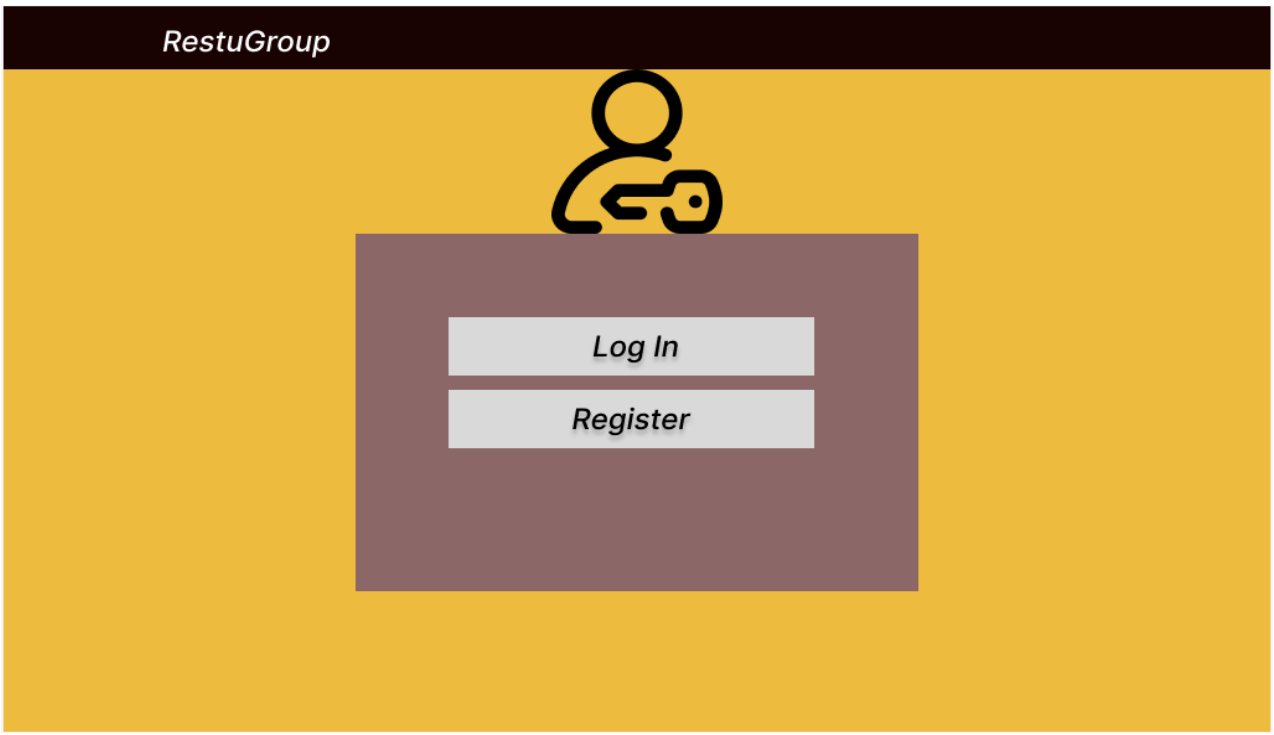

Рисунок 3.4 – Авторизація адміністратора ЦС

Рисунок 3.5 – Меню адміністратора

На рисунку 3.3 зображено меню, після того як адміністатор увійшов в систему, та має декілька сторінок, таких як:

– Заклади, зазначено всі заклади які є в мережі;
– Постачальники, це фірми які постачають на центральний склад продукти

- Історія накладних, зберігаются документи переміщення продуктів по мережі закладів, та закупки з інших постачальників
- Продукти, відображає дату, час та кількість продуктів які були доставлені на ЦС

Папка components включає в себе підпапки з виконавчими файлами компонентів системи, при цьому кожен з компонентів буде відображений на рисунку.

Як раніше вказувалось, для захисту даних ми вбрали JWT Bear token. [8]

При заході адміністратор якщо не вводить зареєстровані дані, відкивається вікно та вказує на те що адміністратор не авторизувався, та пропонує перейти назад для заходу в систему, за допомогою кнопки Button.

Повний лістинг коду представлений у Додатку А.

ВИСНОВКИ

Представлена кваліфікаційна робота призначена вирішити задачу полегшення роботи співробітників закладів харчування, об’єднаних у мережі, а саме обліку продуктів харчування.

У першому розділі було описано структуру закладів харчування, їх принцип роботи, порівняння існуючих систем, виявлення їх переваг та недоліків, огляд проблем та способи їх вирішення.

В розділі два описано систему обліку для закладів харчування, методи створення бази даних, порівняння архітектурних моделей, розробка бази даних за допомогою ER – діаграми, та її реалізація в MySQL.

В розділі три представлено проєктування архіітектури ІС для мережі закладів харчування, оглянуто стейхолдінг та опис атрибутів якості ПЗ, показана розробка захисту даних від сторонніх осіб.

Розроблена інформаційна система може використовуватися не тільки для закладів харчування, а при бажанні для рітейлу, логістики тощо.

ПЕРЕЛІК ПОСИЛАНЬ

1. Завадинська О. Ю., Литвиненко Т.Є. Організація ресторанного господарства за кордоном: Навч. посіб [ел. ресурс] Режим доступу https://tourism-book.com/pbooks/book-33/ua/chapter-1621/

2. Dilovod Dilovod - український онлайн-сервіс для ведення управлінського, бухгалтерського обліку та складання звітності. [ел.ресурс] - https://dilovod.ua/

3. POS Sektor Система обліку в кафе та обладнання для автоматизації процесів [ел.ресурс] - https://www.livebusiness.com.ua/tool/424/

4. В.В. Архіпов Організація ресторанного господарства навч. Посібник [ел. ресурс] http://nkkep.com/wp-content/uploads/2015/10/ORG-Arhy-pov.pdf

5. Щепакіна Т.Є Роль змісту навчальних задач до теми «Бази даних. СУБД» у формуванні інформаційної культури учня // Комп'ютерно-орієнтовані системи навчання [ел. ресурс]

6. К. Дмитро Опис роботи API [ел.ресурс] https://brainlab.com.ua/uk/blog-uk/shho-take-api#title_13

7. Bearer Authentication - Аутентифікація носія схема аутентифікації [ел. ресурс] - https://swagger.io/docs/specification/authentication/bearer-authentication/+

8. React JS React -JavaScript-бібліотека з відкритим вихідним кодом для розробки користувальницьких інтерфейсів. [ел ресурс] - https://react.dev/

9. Node JS Середовище виконання сценаріїв Java, побудоване на механізмі сценаріїв Java Chrome V8. [ел. ресурс] - https://nodejs.org/en

10. JWT відкритий стандарт для створення токенів доступу, заснований на форматі JSON. [ел. ресурс] - https://jwt.io/

11. Бази даних, Система керування базами даних [ел.ресурс] - https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85
12. API набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення [ел. ресурс] - https://aws.amazon.com/what-is/api/

Братищенко Т.С., Сайківська Л.Ф. Вибір технологій для реалізації програмної частини системи відстеження переміщення автомобілів /Innovative trends of science and practice, tasks and ways to solve them. Proceedings of the XXV International Scientific and Practical Conference Athens, Greece June 28 – July 01, 2022 – p.p. 493-495.