

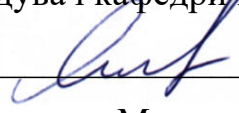
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " ____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

на тему: "Дослідження систем використання вебзастосунків управління командною взаємодією в організаціях"

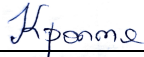
Виконав: студент групи 6144 м

 Козлов О.Ю.
(підпис)

Керівник роботи:

доцент кафедри ІУСТ, к.т.н.,

(посада, науковий ступінь вчене звання)

 Кроптя А.В.
(підпис)

м. Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 126 "Інформаційні системи та технології"
Освітня програма "Інформаційні системи та технології"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми


к.т.н, доц. Чернова Лб.С.

" ____ " _____ 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "магістр"

Студенту Козлову Олександрю Юрійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідження систем використання вебзастосунків управління командною взаємодією в організаціях

Керівник роботи Кроптя Арсеній Володимирович

Затверджені наказом ректора № 1355-уч. від "14" листопада 2025 року.

2. Термін подання роботи: 15 грудня 2025 р.

3. Вихідні дані до проєкту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування

4. Перелік питань, що належать до розробки (найменування розділів)

Дослідження інформаційних систем управління командною взаємодією, розробка проєкту та концепції інформаційної системи управління командною взаємодією, розробка проєктних рішень щодо веб-додатку для управління командною взаємодією, розрахунок економічної ефективності впровадження інформаційної системи для управління командною взаємодією в ресторані, охорона праці, охорона навколишнього середовища.

5. Перелік презентаційних матеріалів вступ, мета та актуальність роботи, схема запропонованої концепції, діаграма варіантів використання, концептуальна

модель, фізична модель, діаграма компонентів, діаграма переходу станів, діаграма розгортання системи, результати розробки, висновок.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Кроптя А.В.	05.09.2024	22.12.2024
2	Кроптя А.В.	23.12.2024	15.06.2025
3	Кроптя А.В.	15.06.2025	25.09.2025
4	Кроптя А.В.	25.09.2025	14.11.2025
5	Кроптя А.В.	25.09.2025	14.11.2025
6	Кроптя А.В.	25.09.2025	14.11.2025

7. Дата видачі завдання 05 вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

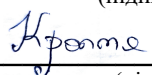
№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів роботи	Примітка
1	Дослідження та аналіз предметної галузі	22.12.2024	
2	Розробка концепції	15.03.2025	
3	Розділ з розробки проєктних рішень по системі (та її частинах)	15.06.2025	
4	Реалізація проєкту та розробка робочої документації	25.09.2025	
5	Обчислення економічної ефективності	14.11.2025	
6	Вирішення питань охорони праці та навколишнього середовища	14.11.2025	
7	Оформлення пояснювальної записки	14.11.2025	
8	Подання магістерської роботи на перевірку на плагіат	16.11.2025	
9	Подання магістерської роботи рецензенту	20.11.2025	
10	Подання магістерської роботи на державний захист	15.12.2025	

Студент


(підпис)

Козлов Олександр Юрійович
(ПІБ)

Керівник роботи


(підпис)

Кроптя Арсеній Володимирович
(ПІБ)

АНОТАЦІЯ

Робота складається з 101 сторінки, 9 ілюстрацій, 25 таблиць, 4 додатків та 20 джерела у списку літератури.

У роботі розглянуто розробку інформаційної системи для управління командною взаємодією в ресторані. Актуальність дослідження зумовлена стрімким ускладненням внутрішніх процесів у закладах громадського харчування, де одночасно працюють офіціанти, кухарі, адміністратори, бармени та менеджери. За таких умов координація дій, швидка передача інформації та узгодження завдань є критично важливими для якості обслуговування, проте традиційні методи комунікації (усні вказівки, паперові записи, телефонні дзвінки) призводять до затримок, помилок, втрати даних і зниження продуктивності. У процесі дослідження проаналізовано існуючі рішення, сформовано вимоги користувачів, розроблено архітектуру системи, моделі бази даних, діаграми використання та компонентів. Реалізовано веб-додаток із модулями управління замовленнями, меню, графіками роботи та push-сповіщеннями, а також проведено тестування та економічне обґрунтування впровадження.

Практичним результатом стала готова до використання платформа, що забезпечує швидку координацію роботи офіціантів, кухарів та адміністраторів, підвищує продуктивність персоналу, прискорює обробку замовлень та зменшує операційні втрати ресторану.

Ключові слова: командна взаємодія, ресторан, веб-система, цифрова координація ресторану, push-сповіщення, розробка, база даних.

ANNOTATION

The work consists of 101 pages, 9 illustrations, 25 tables, 4 appendices and 20 sources in the list of literature.

The paper considers the development of an information system for managing team interaction in a restaurant. The relevance of the study is due to the rapid complication of internal processes in catering establishments, where waiters, cooks, administrators, bartenders and managers work simultaneously. Under such conditions, coordination of actions, rapid transfer of information and agreement of tasks are critically important for the quality of service, however, traditional methods of communication (oral instructions, paper records, telephone calls) lead to delays, errors, data loss and reduced productivity. In the process of the study, existing solutions were analyzed, user requirements were formed, system architecture, database models, usage diagrams and components were developed. A web application with order management modules, menus, work schedules and push notifications was implemented, and testing and economic justification of the implementation were carried out.

The practical result was a ready-to-use platform that ensures rapid coordination of the work of waiters, cooks and administrators, increases staff productivity, speeds up order processing and reduces restaurant operational losses. The practical result was a ready-to-use platform that ensures rapid coordination of the work of waiters, cooks and administrators, increases staff productivity, speeds up order processing and reduces restaurant operational losses.

Keywords: team interaction, restaurant, web system, digital restaurant coordination, push notifications, development, database.

ЗМІСТ

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ	2
1.1. Поняття та класифікація інформаційних систем	2
1.2. Теоретичні підходи до автоматизації комунікаційних процесів	4
1.3. Огляд сучасних інструментів і платформ для управління.....	6
1.4. Роль інформаційних систем у ресторанному бізнесі та аналіз наявних пропозицій на ринку.....	7
Висновок до розділу 1	10
РОЗДІЛ 2. РОЗРОБКА ПРОЄКТУ ТА КОНЦЕПЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ.....	11
2.1. Аналіз предметної області та потреб організацій	11
2.2 Вимоги до інформаційної системи	12
2.2.1 Функціональні вимоги.....	12
2.2.2 Нефункціональні вимоги	13
2.3 Розробка концепції інформаційної системи.....	15
2.4 Побудова моделі варіантів використання.....	17
2.5 Рішення з інформаційного забезпечення.....	19
2.6. Розробка діаграми переходу станів.....	28
Висновок до розділу 2	32
Розділ 3. Розробка проєктних рішень щодо веб-додатку для управління командною взаємодією	33
3.1. Розробка схеми структури інформаційної системи.....	33
3.2 Рішення з програмного забезпечення	37
3.3. Розробка діаграми компонентів	39

3.4. Розробка діаграми розгортання підсистеми.....	40
Висновок до розділу 3	41
РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ В РЕСТОРАНІ.....	42
4.1 Вступ.....	42
4.2 Розрахунок витрат на створення та експлуатацію системи управління командною взаємодією в ресторані	43
4.3 Економічна ефективність та впровадження інформаційної системи управління командною взаємодією в ресторані	45
Висновок до розділу 4.....	48
РОЗДІЛ 5. Охорона праці	49
5.1 Вступ.....	49
5.2 Аналіз шкідливих та небезпечних факторів	49
5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій.....	51
5.4 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій.....	52
5.4.1 Заходи щодо зменшення впливу	52
5.4.2 Техніка безпеки при роботі з комп'ютером	54
5.5. Розрахунок освітленості робочого місця	56
Висновок до розділу 5	58
РОЗДІЛ 6. Охорона навколишнього середовища.....	59
6.1 Вступ.....	59
6.2 Забруднення навколишнього середовища.....	61

6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища	62
Висновок до розділу 6	64
Висновки.....	65
Список використаних джерел.....	67

ВСТУП

У сучасних умовах цифрової трансформації бізнесу питання ефективної командної взаємодії набуває особливої актуальності, зокрема в організаціях сфери обслуговування, де узгодженість дій персоналу безпосередньо впливає на якість сервісу та швидкість виконання операційних процесів.

Актуальність дослідження зумовлена тим, що наявні платформи для командної взаємодії часто не враховують специфіку окремих організацій та не забезпечують достатньої адаптивності під реальні бізнес-процеси.

Об'єктом дослідження є процеси командної взаємодії персоналу в організаціях сфери обслуговування, зокрема в ресторанах. Предметом дослідження виступають методи, засоби та інформаційні технології, що забезпечують цифрову підтримку комунікації, узгодження завдань та координації роботи команди.

Метою роботи є створення ефективної інформаційної системи, яка підвищує якість та швидкість командної взаємодії за рахунок автоматизації ключових бізнес-процесів, оптимізації комунікаційних потоків та забезпечення прозорості управління.

Наукова новизна роботи полягає у комплексному підході до моделювання командної взаємодії в ресторанній індустрії, де запропоновано інтегровану архітектуру, що поєднує управління замовленнями, складом, персоналом та графіками роботи в єдиному веб-додатку.

У межах дослідження розроблено програмну архітектуру системи, визначено принципи її функціонування, реалізовано модулі обробки замовлень, комунікації між працівниками, управління змінами та аналітики. Створений веб-додаток поєднує сучасні технології та підходи, забезпечуючи зручність використання, безпеку та можливість масштабування відповідно до потреб ресторанного бізнесу.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ

1.1. Поняття та класифікація інформаційних систем

Інформаційні системи (ІС) є фундаментом сучасного управління бізнес-процесами. Їхня роль полягає не лише в автоматизації рутинних операцій, а й у забезпеченні ефективної комунікації, аналізу даних та стратегічному плануванні. Розуміння сутності та різновидів ІС дозволяє вибрати оптимальне рішення для конкретних потреб організації.

Інформаційна система - це комплекс взаємопов'язаних компонентів: програмного забезпечення, апаратних засобів, баз даних, мереж, персоналу та процедур, спрямованих на збір, обробку, зберігання та поширення інформації. Головна мета ІС - забезпечити своєчасний доступ до точних даних для підтримки операційної діяльності, прийняття управлінських рішень і координації між підрозділами [1].

До ключових характеристик сучасних ІС належать інтеграція, масштабованість, безпека, інтуїтивний інтерфейс та адаптивність до змін.

Інтеграція означає, що система об'єднує дані з різних джерел, наприклад, онлайн-замовлення через додаток, транзакції через POS-термінали та відгуки клієнтів у соцмережах, у єдиний інформаційний простір.

Масштабованість - це здатність адаптуватися до зростання навантаження, дозволяючи додавати нових користувачів або підключати додаткові модулі, такі як платіжні шлюзи.

Безпека забезпечує захист конфіденційних даних, зокрема персональних даних клієнтів та фінансової звітності, за допомогою шифрування, двофакторної аутентифікації та резервного копіювання.

Інтуїтивний інтерфейс робить систему зручною у використанні для персоналу з різним рівнем технічної підготовки, наприклад, за допомогою drag-and-drop інструментів для створення меню.

Нарешті, адаптивність до змін дозволяє оновлювати функції системи, наприклад, додавати нові методи оплати, без зупинки її роботи.

Історично ІС пройшли шлях від простих систем обліку до складних інтелектуальних платформ. У 1980-х роках переважали локальні бази даних, які вимагали ручного введення інформації, наприклад, облік запасів у Excel. Сьогодні домінують хмарні рішення зі штучним інтелектом, такі як Oracle MICROS, яка аналізує продажі в реальному часі і пропонує оптимізацію меню на основі попиту. Ця еволюція відображає зростання потреби в швидкості, точності та прогнозуванні.

ІС класифікують за рівнем управління, специфікою завдань та галузевими особливостями.

Основні класифікації ІС наведені у табл.1.1.

Таблиця 1.1 – Основні класифікації ІС

Тип ІС	Основні задачі
TPS (системи обробки транзакцій)	Обробка щоденних операцій: прийом замовлень, фіксація оплат, бронювання столиків.
MIS (інформаційні системи управління)	Аналіз даних для формування звітів.
DSS (системи підтримки рішень)	Прогнозування попиту, моделювання сценаріїв (наприклад, оцінка ефективності знижок).
KMS (системи управління знаннями)	Зберігання корпоративної інформації.
ERP (планування ресурсів підприємства)	Інтеграція всіх бізнес-процесів: логістика, HR, фінанси, запаси.

TPS є фундаментом для багатьох організацій, оскільки без них неможлива обробка щоденних взаємодій. Ці системи автоматизують ключові етапи, такі як створення документів при замовленні, відправка запитів до відповідних відділів та формування звітів про діяльність. MIS трансформує сирі дані в корисну інформацію. Наприклад, аналізуючи час виконання певних операцій, система може виявити, що певний процес займає значно більше часу, ніж інші, що дає підстави для перегляду та оптимізації внутрішніх процедур [2].

ERP-системи особливо важливі для організацій з розгалуженою структурою. Вони дозволяють синхронізувати запаси між різними підрозділами, ефективно керувати графіками співробітників та контролювати витрати на ресурси [3].

Сучасним трендом є гібридні системи, які поєднують функції різних типів ІС. Наприклад, деякі платформи інтегрують TPS для обробки транзакцій з DSS для прогнозування попиту на основі історичних даних. Це дозволяє не лише фіксувати операції, але й оптимізувати закупівлю та планування ресурсів.

Поява хмарних обчислень, штучного інтелекту (AI) та інтернету речей (IoT) кардинально змінила ландшафт інформаційних систем. Наприклад, AI-алгоритми можуть аналізувати відгуки користувачів у соціальних мережах, щоб визначити загальні тренди, такі як зростання популярності певних продуктів чи послуг. IoT-датчики можуть автоматично відстежувати стан матеріалів та формувати заявки на їх поповнення. Технології, такі як блокчейн, забезпечують прозорість ланцюгів поставок, що є критично важливим для багатьох галузей. Ці технології роблять ІС не просто інструментами, а стратегічними партнерами бізнесу, що дозволяють приймати більш обґрунтовані рішення та підвищувати загальну ефективність.

1.2. Теоретичні підходи до автоматизації комунікаційних процесів

Автоматизація комунікаційних процесів є критичним елементом оптимізації роботи команд, забезпечуючи точність, швидкість передачі інформації та зменшення операційних помилок [4]. Цей розділ охоплює ключові методи, принципи побудови систем та сучасні інструменти, які можуть бути адаптовані до різних бізнес-середовищ.

Сучасні методи автоматизації спираються на систематизацію рутинних операцій та інтелектуальний аналіз даних. Найефективніші підходи включають методологію BPM, програмних "ботів" в рамках RPA, використання Agile-інструментів та систем зі штучним інтелектом (AI).

Методологія BPM охоплює моделювання, виконання та оптимізацію послідовних дій, спрямованих на покращення ефективності бізнес-процесів. Наприклад, автоматизація ланцюжка клієнтських запитів передбачає їх автоматичне розподілення між відповідними відділами з подальшим сповіщенням

про статус виконання [5]. Для реалізації таких процесів використовуються інструменти типу Camunda, які дозволяють візуалізувати алгоритми дій, налаштовувати автоматичні тригери для ескалації задач та аналізувати результативність кожного етапу [6]. Це забезпечує прозорість і контроль над складними операціями, зменшуючи ризики людських помилок.

Програмні "боти" в рамках RPA виконують рутинні завдання, такі як агрегація даних із транзакційних систем, онлайн-платформ або зовнішніх джерел. Вони автоматизують формування звітів, надсилання нагадувань або оновлення інформації в базах даних. Рішення на кшталт UiPath інтегруються з популярними додатками, включаючи електронну пошту та таблиці, що дозволяє скоротити витрати часу на ручну обробку інформації [7]. Наприклад, бот може щодня збирати дані про продажі з різних джерел, об'єднувати їх у єдиний звіт і розсилати його керівництву, що значно підвищує оперативність прийняття рішень.

Використання Agile-інструментів, таких як Trello або Jira, дозволяє команді візуалізувати завдання на цифрових дошках, розподіляти обов'язки між учасниками та відстежувати прогрес у реальному часі. Наприклад, дошка Trello може містити колонки "Заплановано", "В роботі" та "Виконано", де кожна картка відображає конкретну задачу з дедлайнами та відповідальними особами. Це спрощує комунікацію, забезпечує синхронізацію дій усіх членів команди і дозволяє швидко адаптуватися до змін у пріоритетах [8].

Системи зі штучним інтелектом (AI) аналізують комунікаційні дані, такі як час відгуку на запити, частота помилок або інші метрики, що впливають на ефективність командної роботи. Вони також автоматизують взаємодію з клієнтами через чат-боти, наприклад Dialogflow, які можуть обробляти стандартні запити, перенаправляти складні питання співробітникам або аналізувати зворотний зв'язок із соціальних мереж [9]. Наприклад, система на основі AI може виявляти шаблони у відгуках клієнтів, автоматично класифікуючи їх на позитивні, нейтральні та негативні, що допомагає швидше реагувати на проблеми.

1.3. Огляд сучасних інструментів і платформ для управління

Сучасний ринок пропонує широкий спектр інструментів для автоматизації комунікаційних процесів і управління командною роботою. Ці рішення відрізняються функціональністю, масштабованістю та спеціалізацією, що дозволяє підібрати оптимальний варіант для різних бізнес-потреб. Ключові категорії та приклади платформ, що домінують у сфері управління, включають комунікаційні платформи, інструменти управління проєктами, платформи для автоматизації бізнес-процесів, системи управління знаннями та аналітикою, а також спеціалізовані рішення для вертикальних ринків.

Комунікаційні платформи, такі як Slack або Microsoft Teams, стали стандартом для організації внутрішньої комунікації. Вони надають можливість створювати тематичні канали для окремих проєктів, відділів або задач, що структурує обмін інформацією. Наприклад, у Slack користувачі можуть інтегрувати ботів для автоматизації нагадувань, проводити відеоконференції та обмінюватися файлами без переключення між додатками. Microsoft Teams, інтегрований з пакетом Office 365, дозволяє працювати над документами в режимі реального часу, що особливо корисно для команд, які займаються спільним редагуванням звітів або презентацій [10].

Інструменти управління проєктами, такі як Trello, Asana та Jira, є лідерами у сфері планування та відстеження задач. Trello, заснований на принципах візуалізації, використовує цифрові дошки з картками для кожного завдання, що дозволяє команді бачити статус робіт на одному екрані. Asana пропонує більш деталізовані інструменти для розподілу обов'язків, встановлення дедлайнів і пріоритетів, що ідеально підходить для складних проєктів з багатьма етапами. Jira, популярна серед IT-команд, орієнтована на управління розробкою програмного забезпечення, включаючи створення спринтів, відстеження помилок та інтеграцію з системами контролю версій [8].

Платформи для автоматизації бізнес-процесів, такі як Camunda, UiPath та Zapier, допомагають автоматизувати рутинні операції. Camunda дозволяє моделювати

складні бізнес-процеси, наприклад, схвалення заявок або обробку клієнтських запитів, з подальшим моніторингом ефективності кожного кроку [6]. UiPath спеціалізується на роботизації процесів (RPA), де програмні "боти" імітують дії користувача: заповнюють форми, переносять дані між системами або генерують звіти [7].

Системи управління знаннями та аналітикою, такі як Notion і Airtable, поєднують функції баз даних, управління проєктами та документації. Notion дозволяє створювати інтерактивні бази знань, де команди можуть зберігати стандарти роботи, інструкції або шаблони. Airtable, заснований на гібриді електронних таблиць і баз даних, надає інструменти для планування ресурсів, управління запасами або навіть організації маркетингових кампаній [11]. Для аналітики популярні рішення типу Tableau або Power BI, які візуалізують дані з різних джерел, допомагаючи виявляти тренди, аналізувати ефективність процесів або прогнозувати завантаження [12].

1.4. Роль інформаційних систем у ресторанному бізнесі та аналіз наявних пропозицій на ринку

Інформаційні системи в ресторанній індустрії охоплюють широкий спектр функцій, спрямованих на оптимізацію роботи та підвищення якості обслуговування. Це включає автоматизацію замовлень і платежів, де ІС інтегрують POS-термінали, онлайн-замовлення через додатки та платіжні шлюзи, як, наприклад, система iiko, що дозволяє офіціантам приймати замовлення через планшети та автоматично передавати їх на кухню, а клієнтам - оплачувати рахунок через QR-код, зменшуючи час обслуговування та кількість помилок [13].

Крім того, важливими функціями є управління запасами та логістика, коли системи відстежують залишки інгредієнтів, автоматично формують заявки постачальникам та прогнозують попит на основі історичних даних, як це робить Lavi, надаючи інструменти для контролю термінів придатності продуктів. Також

значну роль відіграє аналітика та звітність. ІС збирають дані про продажі, популярність страв, середній чек і завантаженість залу, а такі інструменти, як Microsoft Power BI, візуалізують ці дані, допомагаючи керівництву приймати рішення щодо змін у меню або маркетингових стратегіях [12].

Керування взаємодією з клієнтами (CRM) також є ключовим аспектом, оскільки системи зберігають історію замовлень, вподобання клієнтів та відгуки, а Yumplingo, наприклад, аналізує відгуки в реальному часі, пропонуючи персоналізовані пропозиції для відвідувачів. Нарешті, оптимізація роботи персоналу передбачає, що інструменти типу 7shifts автоматизують складання графіків, відстежують відпрацьований час та навчають нових співробітників через онлайн-курси.

Сучасний ринок інформаційних систем для ресторанного бізнесу пропонує рішення різного рівня складності, від простих POS-систем до повноцінних ERP-платформ. Однією з найпопулярніших є iiko - комплексна система, яка охоплює управління замовленнями, запасами, лояльністю клієнтів та аналітику [13]. Її перевагою є глибока інтеграція з сервісами доставки, такими як Glovo або Bolt Food, що дозволяє автоматизувати передачу замовлень з онлайн-платформ прямо на кухню. Однак iiko вимагає значних витрат часу на налаштування, що робить її менш доступною для малих закладів.

Для невеликих ресторанів або стартапів більш підходящим варіантом стає Poster - легка POS-система з інтуїтивним інтерфейсом. Вона дозволяє керувати онлайн-замовленнями, проводити маркетингові акції, наприклад, знижки на "страву дня", та контролювати графіки персоналу [13]. Недоліком Poster є обмежений функціонал аналітики, який не задовольняє потреби великих мереж. У сегменті середнього бізнесу домінує Lavi, яка відрізняється інтеграцією з "розумним" обладнанням. Наприклад, кухонні принтери автоматично отримують замовлення з залу, а датчики в холодильниках попереджають про закінчення терміну придатності продуктів. Проте Lavi має обмежену підтримку українського ринку, що ускладнює її використання в локальних умовах. Для глобальних мереж, таких як McDonald's, оптимальним рішенням залишається Oracle MICROS. Ця платформа автоматизує всі аспекти роботи - від закупівлі інгредієнтів до аналізу

ефективності кожного ресторану в мережі. Вона дозволяє централізовано керувати меню, цінами та маркетинговими кампаніями, але її висока вартість і складність інтеграції роблять її недоступною для малого бізнесу.

Вибір інформаційної системи для ресторану залежить від масштабів бізнесу, бюджету та специфіки операцій. Для малих закладів, таких як кав'ярні або фудтраки, оптимальними є прості рішення на кшталт Poster або Square for Restaurants. Вони мають низьку абонплату, зрозумілий інтерфейс і базовий функціонал для обліку замовлень та платежів. В Україні також набирає популярність локальна система Plazius, яка адаптована під місцеві податкові вимоги та підтримує інтеграцію з українськими платіжними сервісами.

Середні та великі мережі звертаються до більш потужних рішень, таких як iiko або Lavu. Ці системи поєднують управління запасами, аналітику продажів і інструменти для лояльності клієнтів. Наприклад, iiko дозволяє створювати динамічні меню, які автоматично змінюються в залежності від наявності інгредієнтів, а Lavu пропонує детальні звіти про завантаженість залу в різний час доби. Для глобальних мереж, де критичною є централізація управління, лідером залишається Oracle MICROS. Вона забезпечує єдиний простір для контролю фінансів, логістики та персоналу в усіх філіях, але вимагає значних інвестицій і кваліфікованих фахівців для підтримки.

Тенденцією останніх років є активне впровадження AI та IoT. Наприклад, система Zenput використовує штучний інтелект для аналізу даних з IoT-датчиків на кухні, прогнозуючи попит на страви або виявляючи відхилення від стандартів якості. Іншим трендом є розвиток мобільних додатків для клієнтів, які поєднують функції онлайн-замовлення, програм лояльності та збору відгуків. Головними викликами залишаються висока вартість впровадження складних систем для малого бізнесу, необхідність перепідготовки персоналу та обмежена доступність локалізованих рішень. Проте прогресивні технології, такі як хмарні обчислення та низькокодовий функціонал, поступово зменшують ці бар'єри, роблячи автоматизацію доступною для більшої кількості закладів.

Висновок до розділу 1

Узагальнюючи результати першого розділу, можна зробити висновок, що інформаційні системи управління командною взаємодією є ключовим інструментом підвищення ефективності роботи персоналу та якості управлінських рішень. Розглянуті поняття, класифікації та теоретичні підходи до автоматизації комунікаційних процесів формують методологічне підґрунтя для подальшого проєктування спеціалізованої системи. Аналіз сучасних платформ показав, що існуючі рішення охоплюють широкий спектр функцій, однак здебільшого орієнтовані на загальні бізнес-процеси й не завжди повною мірою враховують специфіку командної роботи в закладах ресторанного господарства. Огляд інструментів, що застосовуються в ресторанному бізнесі (POS-системи, ERP-рішення, CRM-платформи), виявив наявність часткових рішень, які часто не інтегрують управління комунікацією персоналу в єдиний інформаційний простір. Це створює підґрунтя для розробки концепції спеціалізованої інформаційної системи, спрямованої на комплексне управління командною взаємодією в ресторані та підвищення узгодженості роботи всіх учасників процесу.

РОЗДІЛ 2. РОЗРОБКА ПРОЄКТУ ТА КОНЦЕПЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ

2.1. Аналіз предметної області та потреб організацій

Ефективне функціонування ресторанного закладу значною мірою залежить від раціональної організації управління персоналом та чіткого розподілу обов'язків між працівниками. Формування структурованої системи ролей і зон відповідальності є основою узгодженої роботи підрозділів, сприяє зменшенню дублювання функцій і забезпечує злагоджену командну взаємодію.

Наявність чітко визначених функціональних меж підвищує рівень персональної відповідальності, оптимізує робочі процеси та дозволяє уникнути зайвих перетинів у завданнях. Коли кожен співробітник володіє конкретним колом обов'язків, це створює умови для підвищення продуктивності та якості обслуговування. Наприклад, адміністратор зосереджується на координації персоналу й контролі бронювань, офіціанти - на безпосередньому обслуговуванні гостей, а технічний персонал - на забезпеченні справності обладнання. Такий підхід забезпечує раціональний розподіл ресурсів і сприяє ефективному досягненню спільної мети - задоволенню потреб клієнтів та стабільному розвитку закладу.

Крім того, структурована організація персоналу дозволяє мінімізувати конфліктні ситуації. Чітке визначення відповідальності та підпорядкованості формує прозорість управлінських процесів, підвищує мотивацію працівників і забезпечує стабільність у командній роботі. Це, у свою чергу, створює основу для впровадження інформаційних систем управління, оскільки чітка організаційна структура є передумовою ефективної цифровізації бізнес-процесів.

Основні задачі кожного робітника у табл.2.1.

Таблиця 2.1 – Основні задачі для співробітників

Посада	Основні щоденні задачі
Генеральний менеджер	Планування роботи ресторану, контроль виконання стратегічних завдань, забезпечення дотримання стандартів обслуговування, координація роботи персоналу.
Адміністратор	Організація роботи зміни, управління бронюваннями, контроль за роботою залу, вирішення поточних питань у взаємодії з клієнтами та працівниками.
Шеф-кухар	Розроблення меню, контроль якості продуктів і готових страв, управління роботою кухарів і кухонного персоналу.
Бармен	Приготування та подача напоїв, підтримання чистоти барної зони, контроль запасів продукції.
Офіціанти	Приймання та передача замовлень, обслуговування гостей, підтримка порядку і готовності залу до роботи.
Кухарі та кухонний персонал	Приготування страв відповідно до рецептур і стандартів, дотримання санітарних вимог, взаємодія з офіціантами.
Прибиральники	Підтримання чистоти у залах і службових приміщеннях, прибирання після робочих змін.
Технічний персонал	Обслуговування технічного обладнання, усунення дрібних несправностей, підтримання належного стану інфраструктури.
Бухгалтер	Облік фінансових операцій, формування звітності, контроль витрат і доходів закладу.
Логіст / складські працівники	Приймання і зберігання товарів, контроль запасів, організація постачання та інвентаризації.

2.2 Вимоги до інформаційної системи

2.2.1 Функціональні вимоги

На основі аналізу предметної області та існуючих аналогів сформовано наступні функціональні вимоги до системи, орієнтовані на різні категорії користувачів:

1. Адміністратор

- 1.1. Перегляд, створення, редагування та видалення меню.
- 1.2. Управління категоріями страв (створення, редагування, видалення).
- 1.3. Робота зі столиками: створення, редагування, перегляд та видалення.
- 1.4. Керування замовленнями: перегляд, створення, редагування та видалення.

1.5.Ресстрація нових користувачів та редагування їхніх профілів.

1.6.Перегляд та налаштування заробітної плати працівників.

1.7.Перегляд статистики та аналітичних даних.

1.8.Авторизація та аутентифікація користувачів.

1.9.Перегляд і модерація відгуків клієнтів.

1.10. Формування та редагування робочих графіків персоналу.

1.11. Підтвердження або відхилення запитів працівників на зміну графіку.

2. Кухар

2.1.Авторизація та аутентифікація.

2.2.Перегляд, редагування та оновлення статусів замовлень.

2.3.Перегляд меню та його структури.

2.4.Робота з категоріями страв (перегляд).

2.5.Отримання push-сповіщень про нові або змінені замовлення.

3. Робітник залу

3.1.Перегляд, редагування та оновлення статусів замовлень.

3.2.Перегляд меню та його структури

3.3.Робота з категоріями страв (перегляд).

3.4.Отримання push-сповіщень про нові або змінені замовлення.

Перелік вимог охоплює широкий спектр сценаріїв взаємодії між персоналом ресторану та системою. Він забезпечує повну підтримку внутрішніх операцій: від управління меню та замовленнями до контролю робочих графіків, інвентаризації та обробки клієнтських відгуків. Такий розподіл функціоналу дозволяє створити цілісну, логічно структуровану та ефективну систему управління командою в ресторані.

2.2.2 Нефункціональні вимоги

Для забезпечення ефективності, стабільності та практичної придатності розроблювана система повинна відповідати ряду критично важливих вимог. Одним із ключових аспектів є безпека даних, адже система оперує фінансовою, персональною та операційною інформацією. Передача даних між клієнтською та серверною частинами має здійснюватися через захищений протокол HTTPS із

використанням сучасних методів шифрування. Доступ до функціональних модулів повинен регулюватися ролями користувачів - офіціанти, кухарі, адміністратори - що унеможлиблює несанкціоновані зміни, наприклад, у фінансових звітах чи даних про зарплати.

Другим важливим чинником є зручність і доступність інтерфейсу. Оскільки система призначена для використання широким колом працівників із різним рівнем технічної підготовки, інтерфейс має бути інтуїтивно зрозумілим, з логічною структурою меню, чіткою візуальною ієрархією та мінімальною кількістю кроків для виконання основних дій. Додаток має бути адаптивним - коректно відображатися на комп'ютерах, планшетах і смартфонах - що дозволить персоналу працювати з системою з будь-якого робочого місця. Це зменшить час навчання користувачів і підвищить ефективність роботи персоналу.

Не менш суттєвою характеристикою є інтеграційна здатність системи, яка забезпечує взаємодію з іншими інформаційними підсистемами підприємства. Система має автоматично синхронізуватися з бухгалтерськими інструментами, складським обліком та CRM-модулями, що дозволить оновлювати запаси інгредієнтів після кожного замовлення, контролювати витрати та формувати аналітичні звіти без втручання користувача. Такий підхід забезпечує цілісність даних і прозорість управлінських процесів.

Важливою характеристикою системи є продуктивність, адже оперативність обробки запитів безпосередньо впливає на якість обслуговування клієнтів. Час оновлення даних, наприклад статусу замовлення, не повинен перевищувати 2–3 секунд. Завдяки цьому забезпечується синхронність між кухнею та залом, а також мінімізується ризик затримок у подачі страв.

Крім того, система повинна бути масштабованою - тобто здатною до розширення функціональності без порушення поточної роботи. Така архітектура дає можливість у майбутньому інтегрувати нові модулі, наприклад онлайн-бронювання, аналітичні панелі або систему лояльності, без зупинки основних процесів. Це дозволить адаптувати систему до змін у потребах бізнесу.

Останнім, але не менш важливим параметром є надійність, яка гарантує безперервність роботи закладу. Вона досягається завдяки реалізації механізмів автоматичного резервного копіювання даних, можливості швидкого відновлення після збоїв та контролю стану серверного середовища. Такі заходи забезпечують стабільність роботи навіть у випадку непередбачених технічних проблем.

У сукупності зазначені вимоги створюють фундамент для розроблення ефективної, безпечної та гнучкої інформаційної системи. Вона здатна підтримувати оперативну командну взаємодію між підрозділами ресторану, зменшувати ризики помилок персоналу та підвищувати загальну ефективність управління бізнесом.

2.3 Розробка концепції інформаційної системи

На основі поставлених задач розроблено концепцію інформаційної системи яка представлена на рис 2.1.

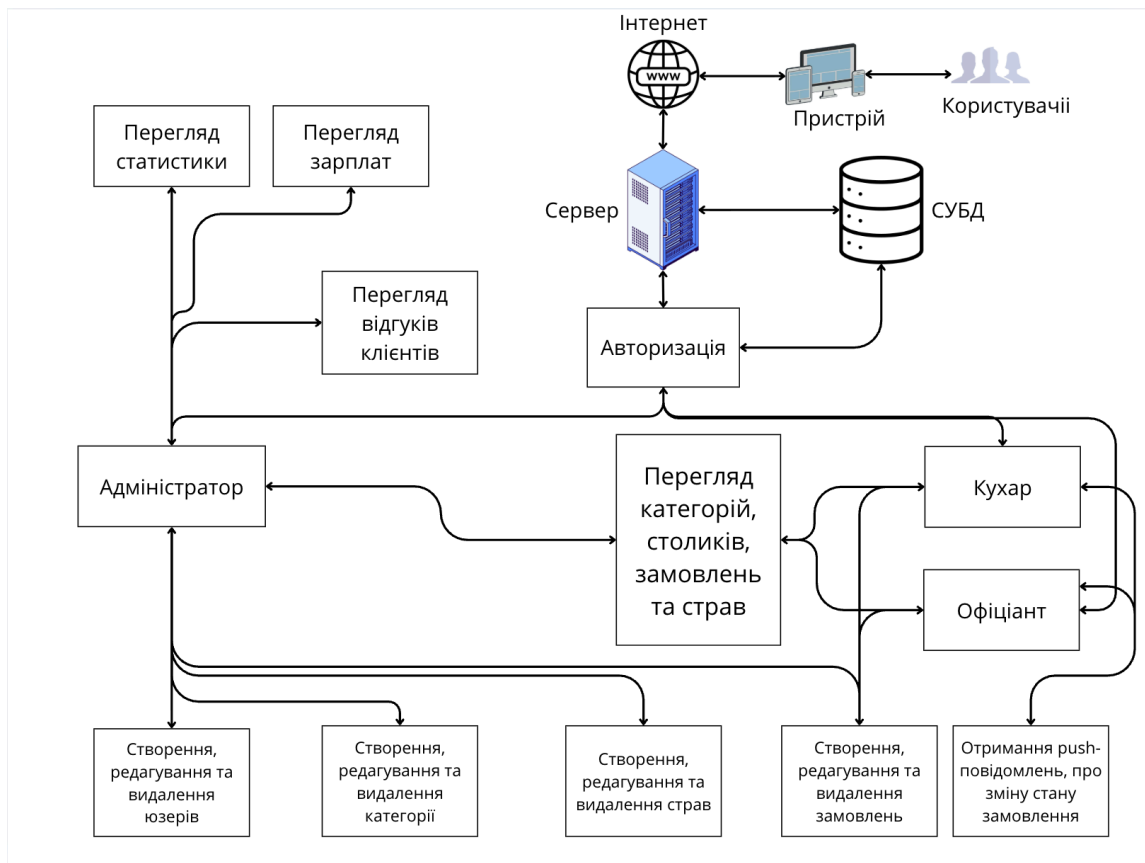


Рисунок 2.1 – Концепція інформаційної системи

Запропонована концепція інформаційної системи управління командною взаємодією в ресторанному бізнесі спрямована на повну автоматизацію ключових процесів обслуговування, адміністрування та аналітики. Вона поєднує централізоване керування даними з гнучким доступом користувачів до необхідного функціоналу через веб інтерфейс [14].

На початковому етапі користувачі проходять авторизацію через сервер, який здійснює взаємодію із системою управління базами даних (СУБД). Авторизація забезпечує доступ до відповідного набору функцій залежно від ролі працівника - адміністратора, офіціанта або кухаря. Такий підхід гарантує безпеку, розмежування повноважень і контроль над діями користувачів.

Адміністратор виступає ключовою ланкою системи. Він має можливість створювати, редагувати та видаляти користувачів, категорії та страви, а також керувати замовленнями. Крім того, адміністратор отримує доступ до перегляду статистики, заробітних плат і відгуків клієнтів, що дозволяє комплексно оцінювати роботу персоналу й рівень задоволеності відвідувачів. Система автоматично формує статистичні звіти та графіки на основі даних із СУБД, забезпечуючи аналітичну підтримку управлінських рішень.

Офіціанти працюють безпосередньо з клієнтами, використовуючи вебінтерфейс для перегляду меню, оформлення замовлень та їх редагування. Усі створені замовлення миттєво передаються на кухню, де кухарі отримують їх у режимі реального часу. Після приготування страв кухар змінює статус замовлення на “готово”, а офіціант отримує push-повідомлення про зміну статусу, що усуває потребу у фізичному обміні інформацією та пришвидшує процес обслуговування.

Система також підтримує механізм збору відгуків клієнтів - після завершення замовлення клієнт може залишити оцінку або короткий коментар, який зберігається в базі даних і доступний адміністратору для подальшого аналізу. Це дозволяє керівництву відстежувати рівень задоволеності гостей, своєчасно реагувати на скарги та вдосконалювати якість сервісу.

Таким чином, запропонована інформаційна система не лише автоматизує процеси прийому, обробки та виконання замовлень, а й створює єдиний

інформаційний простір для всіх учасників робочого процесу. Її використання сприяє підвищенню ефективності роботи персоналу, скороченню часу обслуговування клієнтів, мінімізації помилок та формуванню бази даних для управлінських рішень.

2.4 Побудова моделі варіантів використання

Діаграма варіантів використання (Use Case Diagram) є одним із базових засобів моделювання поведінки інформаційної системи у нотації UML [15]. Вона дозволяє відобразити основні функціональні можливості системи через взаємодію користувачів (акторів) із конкретними сценаріями роботи - варіантами використання. Така діаграма допомагає узагальнено представити, які дії виконує користувач, які функції надає система та як між ними відбувається взаємодія.

Основною метою побудови діаграми варіантів використання є визначення функціональних вимог до системи на етапі проектування. Вона описує систему з точки зору зовнішнього користувача, фокусуючись не на технічних деталях, а на тому, які саме завдання користувач може виконати за допомогою системи. Це дозволяє виявити ключові бізнес-процеси, логічно згрупувати функції та забезпечити чітке розмежування ролей.

Кожен варіант використання (use case) відображає окрему функціональну можливість системи - наприклад, створення замовлення, редагування меню, перегляд звітів або отримання push-повідомлення. Варіанти використання позначаються овальною фігурою з короткою назвою дії, яку ініціює користувач. Вони демонструють, що саме може виконувати система, не деталізуючи спосіб реалізації.

Актори - це зовнішні користувачі або інші системи, які взаємодіють із програмою. У контексті ресторанної системи актором може бути адміністратор, офіціант, кухар, бухгалтер або клієнт, який залишає відгук. Кожен актор ініціює

один або кілька варіантів використання, отримуючи доступ до певних функцій відповідно до своєї ролі.

Основними акторами у системі будуть: адміністратор, шеф-кухар, офіціант, бармен, кухар, бухгалтер. Кожен із них має власний набір прав доступу, який визначає перелік доступних дій у системі.

Короткий опис акторів представлений в табл. 2.2.

Таблиця 2.2. Короткий опис акторів

Роль	Актор	Опис функціоналу
Адміністратор	Адміністратор	Виконує повний контроль над системою: створює, редагує та видаляє користувачів, категорії та страви; адмініструє замовлення, столики та бронювання. Має доступ до статистики, аналітики, зарплат, відгуків клієнтів і складських запасів. Створює графік роботи персоналу, затверджує запити на зміну графіку та може експортувати дані у формат Excel.
	Бухгалтер	Переглядає фінансові показники та статистику продажів, формує та експортує звіти (включно з Excel). Обліковує витрати та прибуток закладу, розраховує заробітну плату та керує її налаштуваннями.
	Шеф-кухар	Отримує інформацію про нові замовлення, змінює їх статус ("у процесі", "готово"), фіксує використання інгредієнтів. Має можливість переглядати меню, склад страв і список активних замовлень.
Офіціант	Кухар	Приймає та редагує замовлення від клієнтів, переглядає меню, категорії та статуси столиків. Отримує push-повідомлення про готовність страв і може фіксувати завершення обслуговування клієнта.
	Офіціант	Створює, редагує та завершує замовлення; переглядає меню, категорії, статуси столиків. Отримує push-повідомлення про готовність страв, подає запити на зміну графіку та може переглядати свій робочий розклад. Фіксує завершення обслуговування клієнтів.
Бармен	Бармен	Приймає та виконує замовлення на напої; переглядає меню, категорії та статуси столиків. Позначає готовність напоїв та передає інформацію офіціантам. Може отримувати push-сповіщення про нові замовлення.

У таблиці 2.2 представлено групування акторів відповідно до їхніх ролей у системі. Для кожної ролі передбачено окремий рівень доступу до функціональних можливостей додатка, що визначається наданими правами користувача.

На основі проведеного аналізу функцій та обов'язків кожного актора було спроектовано діаграму варіантів використання, яка відображає їхню взаємодію із системою. Схематичне зображення цієї діаграми наведено на рисунку 2.2.

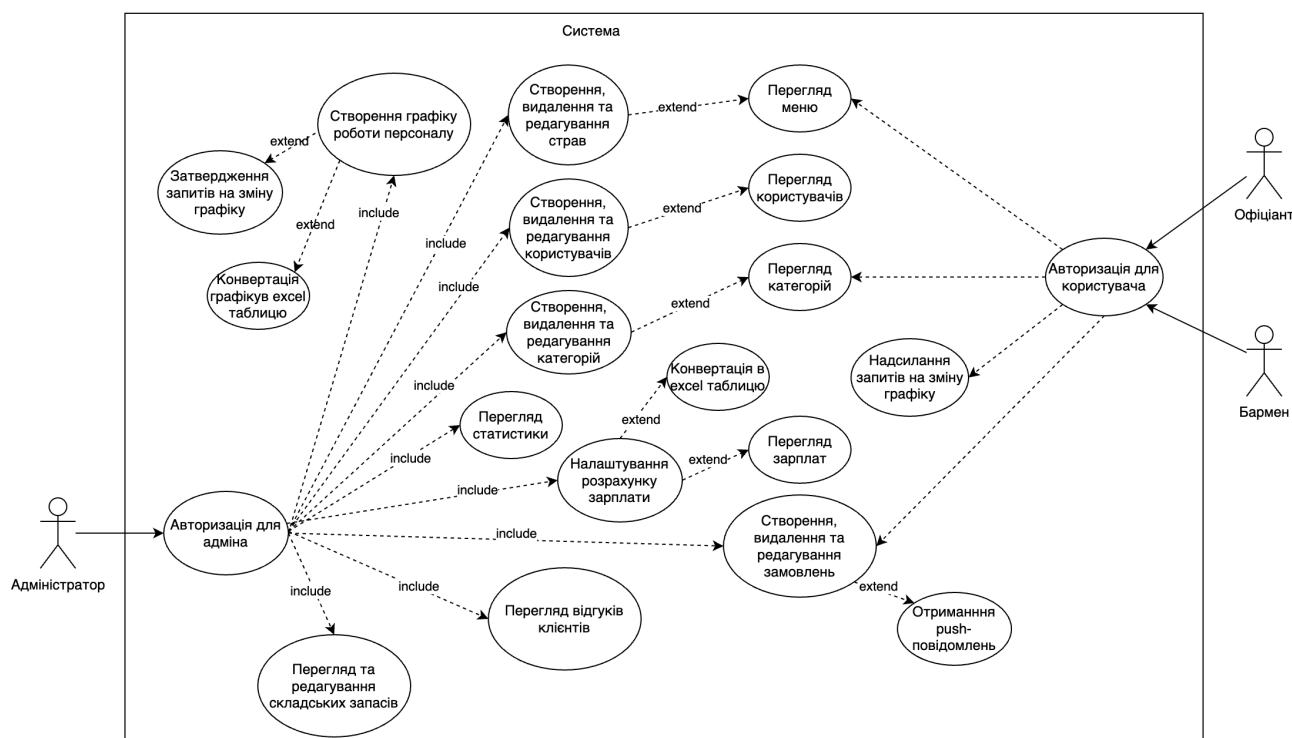


Рисунок 2.2 – Діаграма варіантів використання

2.5 Рішення з інформаційного забезпечення

У цьому підрозділі розглянуто структуру та організацію бази даних, необхідної для коректної та ефективної роботи розроблюваної інформаційної системи. Представлено концептуальну, логічну та фізичну моделі даних, а також виконано нормалізацію бази даних відповідно до вимог нормальних форм.

Концептуальна модель бази даних - це узагальнене, високорівневе подання інформації, яке не залежить від специфіки конкретної СУБД чи технічної реалізації. Така модель відображає основні сутності предметної області, їх ключові атрибути та взаємозв'язки. Її призначення - сформулювати цілісне уявлення про

логічну структуру даних, що використовуються в системі, та забезпечити спільне розуміння між розробниками, аналітиками та замовниками [16].

Концептуальна модель виконує роль базового опису даних, який слугує фундаментом для подальшої побудови логічної та фізичної моделей і є ключовим етапом під час проєктування бази даних.

На рисунку 2.3 представлено концептуальну модель даних. Виділено наступні сутності:

- Ролі користувачів (Roles): Ця сутність описує типи користувачів, які взаємодіють із системою. Роль визначає рівень доступу до функцій та підсистем ресторану. Таким чином, система може чітко регулювати, які можливості доступні адміністратору, кухарю, офіціанту, бухгалтеру та іншим учасникам робочого процесу.
- Працівники (Employees): Сутність відображає інформацію про всіх співробітників ресторану. Вона містить персональні дані, облікові відомості для входу в систему, а також зв'язок із відповідною роллю. Завдяки цьому забезпечуються процедури автентифікації, авторизації та персоніфікованого контролю дій у системі.
- Налаштування оплати праці (Salary Settings): Ця сутність містить правила формування заробітної плати для різних категорій працівників. Вона описує базові ставки, коефіцієнти, бонуси та штрафи, що використовуються для розрахунків. Завдяки цьому система забезпечує автоматизований і прозорий механізм формування зарплат.
- Розрахування заробітної плати (Salary Calculation): Сутність фіксує результати проведених розрахунків оплати праці за певні періоди. Вона зберігає дані про нарахування, фінальні суми та період, за який здійснено виплату. Це дозволяє формувати звіти та забезпечує фінансову прозорість.
- Категорії страв (Categories): Представляє класифікацію страв у меню. Вона використовується для структуризації меню ресторану, спрощення навігації та пришвидшення роботи персоналу при формуванні замовлень.

- Страви меню (Menu Items): Сутність описує окремі позиції меню. Вона містить основну інформацію про страви: назву, опис, вартість та належність до певної категорії. В межах концептуальної моделі страва виступає основним продуктом, який може бути замовлений клієнтом.
- Інгредієнти (Ingredients): Відповідає за опис продуктів, що використовуються під час приготування страв. Містить базову інформацію, необхідну для контролю наявності та планування закупівель. Сутність забезпечує основу для обліку запасів на складі.
- Склад страв (Ingredient Items): Визначає, які інгредієнти та в якій кількості входять до складу конкретної страви. Ця сутність моделює рецептури та забезпечує контроль списання інгредієнтів відповідно до замовлень.
- Столи (Tables): Містить інформацію про dining-зони ресторану. Сутність дозволяє відстежувати статус кожного столу (вільний, зайнятий, заброньований) і використовується під час бронювання та оформлення замовлень.
- Статуси столів (Table Statuses): Описує можливі значення станів столів у залі. Сутність використовується для систематизації логіки роботи інтерфейсу офіціанта та адміністратора.
- Замовлення (Orders): Представляє основний процес взаємодії клієнта з рестораном. Містить інформацію про прийняті замовлення, столик, працівника, який його обробляє, та дату створення. Сутність є центральною частиною операційної діяльності.
- Позиції замовлення (Order items): Сутність деталізує склад замовлення. Вона описує, які страви були замовлені та в якій кількості. Завдяки цьому система може формувати чек, готувати інформацію для кухні та здійснювати списання інгредієнтів.
- Статуси замовлень (Statuses): Визначає етапи проходження замовлення, такі як «очікує», «в роботі», «готове», «оплачене». Сутність допомагає координувати роботу кухні, офіціантів та адміністрації.

- Робочі зміни (Shifts): Визначає типи змін, у яких працює персонал (ранкова, денна, вечірня тощо). Сутність використовується для планування графіків.
- Графік роботи персоналу (Work Schedules): Відображає розклад роботи співробітників. Містить інформацію про те, хто і коли працює, а також дозволяє контролювати вихідні дні.
- Налаштування змін (Schedule settings): Сутність визначає параметри, за якими формується розклад: кількість офіціантів, кухарів або інших працівників у певну зміну.
- Запит на зміну графіка (Shift Change Request): Дозволяє працівникам надсилати запити на зміну або перенесення робочих змін. Сутність забезпечує прозорий механізм погодження та фіксацію рішень адміністратора.
- Відгуки клієнтів (Reviews): Представляє коментарі й оцінки від клієнтів щодо роботи закладу чи якості страв. Сутність застосовується для формування статистики, аналізу рівня сервісу та прийняття управлінських рішень.

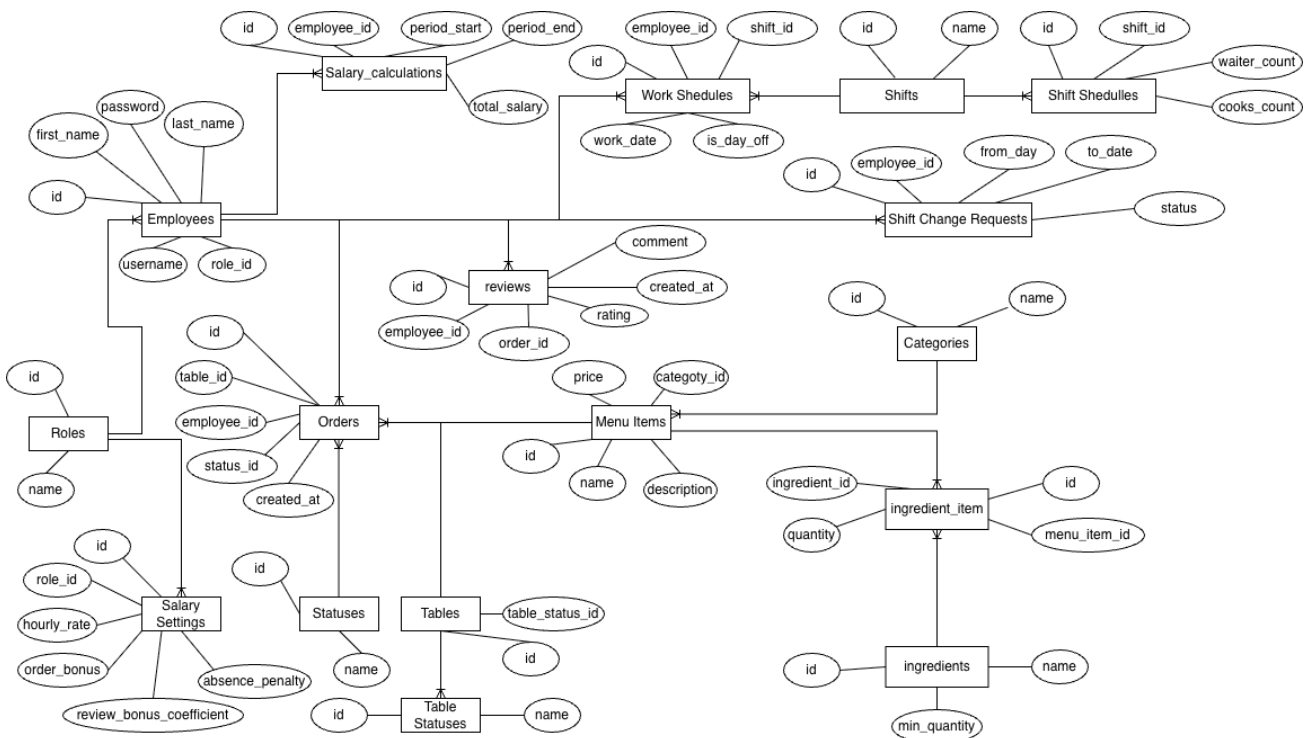


Рисунок 2.3 – Концептуальна модель бази даних

Логічна модель бази даних - це подальший етап після концептуальної моделі, де концепції та зв'язки концептуальної моделі перетворюються на конкретні структури даних, такі як таблиці, колонки та зв'язки між ними [16]. Логічна модель визначає структуру даних в рамках обраної системи управління базами даних (СУБД) і надає детальніше уявлення про те, як будуть зберігатися та оброблятися дані в межах бази даних. Вона описує таблиці, їх атрибути, типи даних та зв'язки між таблицями у вигляді схеми бази даних, що є основою для розробки фізичної реалізації бази даних. Логічна модель дозволяє уточнити дизайн бази даних та підготувати її для подальшої імплементації у вибраній СУБД.

На рисунку 2.4 представлена логічна модель нашої бази даних, де кожна сутність представлена в формі таблиці. Також представленні зв'язки між таблицями за допомогою зовнішніх ключів.

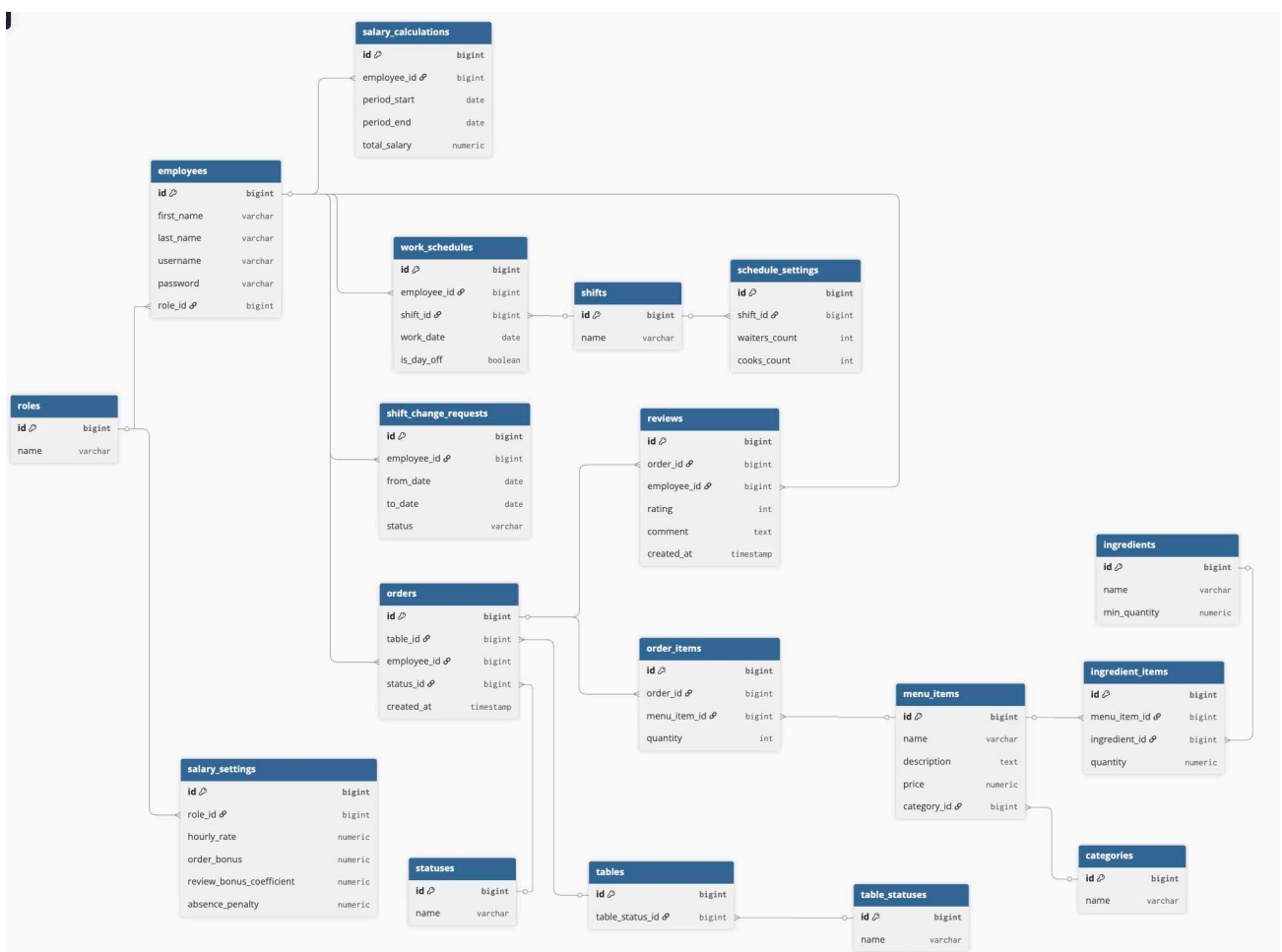


Рисунок 2.4 – Логічна модель бази даних

Для кожної таблиці чітко визначається первинний ключ, який є унікальним ідентифікатором, що однозначно розпізнає кожен рядок у таблиці. Також можуть присутні зовнішні ключі, які встановлюють зв'язки між таблицями. Зовнішній ключ - це стовпець у одній таблиці, що посиляється на первинний ключ у іншій таблиці. В таблицях 2.3 – 2.21 наведено характеристики атрибутів таблиць БД.

Таблиця 2.3. Схема відношення «Roles»

Ім'я поля	Тип	Обмеження цілісності	Ключі
id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.4. Схема відношення «Statuses»

Ім'я поля	Тип	Обмеження цілісності	Ключі
id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.5. Схема відношення «Employees»

Ім'я поля	Тип	Обмеження цілісності	Ключі
employee_id	Long	Not Null	PK
first_name	String	Not Null	
last_name	String	Null	
role_id	Long	Not Null	FK (до Role)
username	String	Not Null	
password	String	Not Null	

Таблиця 2.6. Схема відношення «Salary_Settings»

Ім'я поля	Тип	Обмеження цілісності	Ключі
salary_setting_id	Long	Not Null	PK
role_id	Long	Not Null	FK (до Roles)
hourly_rate	Long	Not Null	
order_bonus	Long	Not Null	
review_bonus_coefficient	Long	Not Null	
absence_penalty	Long	Not Null	

Таблиця 2.7. Схема відношення «Salary_Calculation»

Ім'я поля	Тип	Обмеження цілісності	Ключі
salary_calc_id	Long	Not Null	PK
employee_id	Long	Not Null	FK (до Employees)
period_start	Date	Not Null	
period_end	Date	Not Null	
total_salary	Long	Not Null	

Таблиця 2.8. Схема відношення «Salary_Calculation»

Ім'я поля	Тип	Обмеження цілісності	Ключі
shift_id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.9. Схема відношення «Schedule_Settings»

Ім'я поля	Тип	Обмеження цілісності	Ключі
schedule_id	Long	Not Null	PK
shift_id	Long	Not Null	FK (до Shifts)
waiters_count	Long	Not Null	
cooks_count	Long	Not Null	

Таблиця 2.10. Схема відношення «Shifts»

Ім'я поля	Тип	Обмеження цілісності	Ключі
shift_id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.11. Схема відношення «Work_Schedules»

Ім'я поля	Тип	Обмеження цілісності	Ключі
work_id	Long	Not Null	PK
employee_id	Long	Not Null	FK (до Employees)
shift_id	Long	Not Null	FK (до Shifts)
work_date	Date	Not Null	
is_day_off	Boolean	Not Null	

Таблиця 2.12. Схема відношення «Shift_Change_Requests»

Ім'я поля	Тип	Обмеження цілісності	Ключі
request_id	Long	Not Null	PK
employee_id	Long	Not Null	FK (до Employees)
from_date	Date	Not Null	
to_date	Date	Not Null	
status	String	Not Null	

Таблиця 2.13. Схема відношення «Reviews»

Ім'я поля	Тип	Обмеження цілісності	Ключі
review_id	Long	Not Null	PK
order_id	Long	Not Null	FK (до Orders)
employee_id	Long	Not Null	FK (до Employees)
rating	Integer	Not Null	
comment	String	Null	
created_at	Timestamp	Not Null	

Таблиця 2.14. Схема відношення «Tables»

Ім'я поля	Тип	Обмеження цілісності	Ключі
table_id	Long	Not Null	PK
table_status_id	Long	Not Null	FK (до Table_Statuses)

Таблиця 2.15. Схема відношення «Table_Statuses»

Ім'я поля	Тип	Обмеження цілісності	Ключі
table_status_id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.16. Схема відношення «Orders»

Ім'я поля	Тип	Обмеження цілісності	Ключі
order_id	Long	Not Null	PK
table_id	Long	Not Null	FK (до Tables)
employee_id	Long	Not Null	FK (до Employees)
status_id	Long	Not Null	FK (до Statuses)
created_at	Timestamp	Not Null	

Таблиця 2.17. Схема відношення «Order_Items»

Ім'я поля	Тип	Обмеження цілісності	Ключі
order_item_id	Long	Not Null	PK
order_id	Long	Not Null	FK (до Orders)
menu_item_id	Long	Not Null	FK (до Menu_Items)
quantity	Integer	Not Null	

Таблиця 2.18. Схема відношення «Categories»

Ім'я поля	Тип	Обмеження цілісності	Ключі
category_id	Long	Not Null	PK
name	String	Not Null	

Таблиця 2.19. Схема відношення «Menu_Items»

Ім'я поля	Тип	Обмеження цілісності	Ключі
menu_item_id	Long	Not Null	PK
name	String	Not Null	
description	String	Null	
price	Long	Not Null	
category_id	Long	Not Null	FK (до Categories)

Таблиця 2.20. Схема відношення «Ingredients»

Ім'я поля	Тип	Обмеження цілісності	Ключі
ingredient_id	Long	Not Null	PK
name	String	Not Null	
min_quantity	Decimal	Not Null	

Таблиця 2.21. Схема відношення «Ingredients_Item»

Ім'я поля	Тип	Обмеження цілісності	Ключі
ingredient_item_id	Long	Not Null	PK
menu_item_id	Long	Not Null	FK (до Menu_Items)
ingredient_id	Long	Not Null	FK (до Ingredients)
quantity	Decimal	Not Null	

2.6. Розробка діаграми переходу станів

Діаграма станів – це графічне уявлення процесу чи системи, що дозволяє уявити всі можливі стану та переходи з-поміж них. Вона є одним із ключових інструментів розробки та аналізу програмного забезпечення [15].

Основною метою створення діаграми станів є візуалізація логіки поведінки системи та визначення різних сценаріїв, які можуть виникнути у процесі її роботи. Це допомагає краще зрозуміти і пояснити, як система повинна поводитися в різних ситуаціях, а також виявити потенційні проблеми та покращити проєкт ще на ранній стадії розробки.

На діаграмі переходів станів адміністратора представлено послідовність можливих дій користувача з адміністративною роллю після входу в систему. Початковим станом є процес авторизації, під час якого система очікує введення облікових даних адміністратора. Якщо авторизація проходить успішно, користувач переходить до стану вибору дії, де він може обрати один із доступних функціональних модулів. У випадку помилки авторизації перехід не відбувається, і система повертає адміністратора до початкового стану введення даних.

Після успішного входу адміністратор має можливість переходити між різними станами, що відображають функціональні можливості системи. Одним із таких станів є налаштування умов для робочих змін, яке дозволяє керувати параметрами графіка роботи персоналу. З цього стану адміністратор може або повернутися до головного меню, або перейти до створення графіка змін для співробітників.

Стан створення графіку роботи персоналу забезпечує можливість формування розкладу змін, після чого адміністратор може продовжити роботу в іншому розділі або повернутися до попереднього стану.

Ще одним важливим станом є перегляд складських запасів, який дозволяє адміністратору перевіряти залишки інгредієнтів. З нього можливий перехід до стану редагування запасів, де адміністратор може змінювати кількість інгредієнтів або оновлювати інформацію про склад.

Функціонал управління персоналом реалізований у стані перегляду співробітників, звідки адміністратор може переходити до операцій редагування, додавання або видалення інформації про працівників.

Аналогічно, у стані перегляду меню адміністратор має змогу перейти до редагування, додавання чи видалення страв або категорій. Така сама логіка зберігається у станах перегляду категорій та перегляду замовлень - кожен із них має підстави, що дозволяють виконувати операції створення, редагування або видалення відповідних елементів.

Адміністратор також може перейти до стану перегляду статистики, де доступні аналітичні показники та інші зведені дані. Окремим станом є перегляд відгуків клієнтів, де адміністратор має можливість аналізувати отримані оцінки роботи закладу.

Нарешті, у стані перегляду зарплат адміністратор може переглядати нарахування та переходити до стану налаштування розрахунку заробітної плати, де здійснюється коригування параметрів оплати праці.

Завершальним станом є вихід із облікового запису, після якого діаграма повертається до початкового стану авторизації, що завершує поточну сесію користувача.

Діаграму переходів станів для адміна зображено на рисунку 2.5

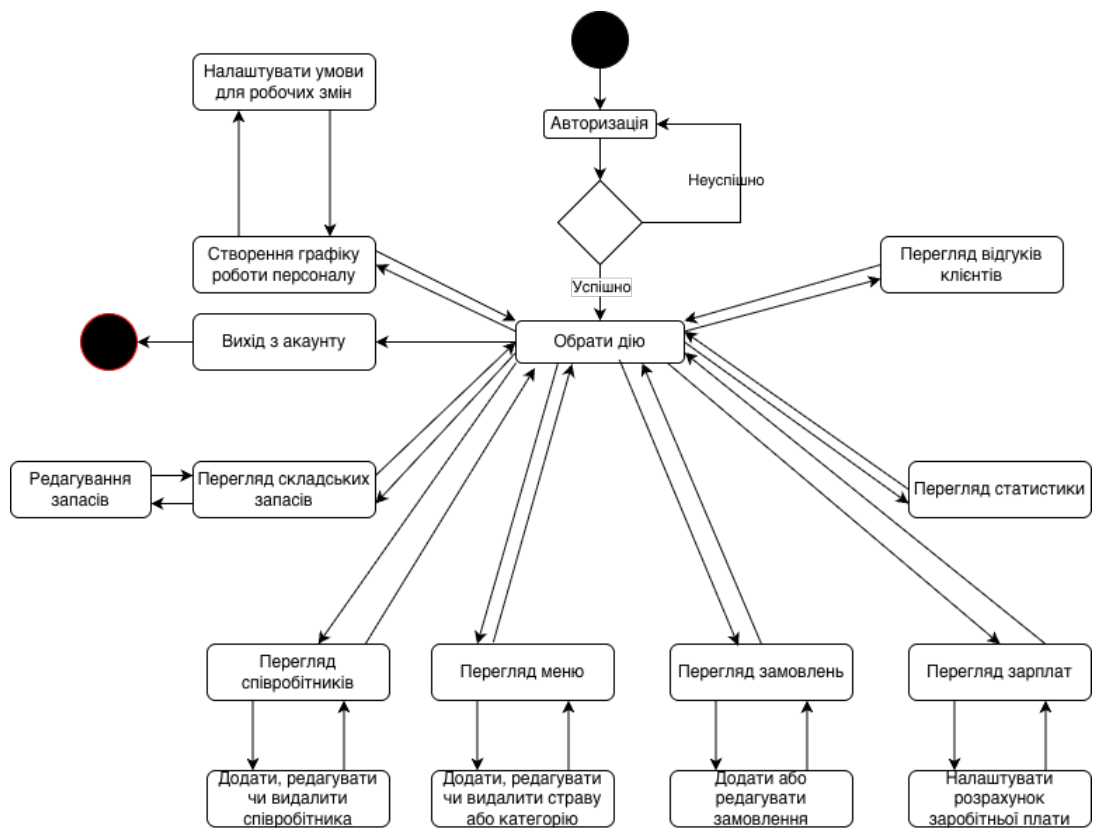


Рисунок 2.5 – Діаграма переходу станів для адміна

На рисунку 2.6 зображена діаграма переходу станів для повара та офіціанта.

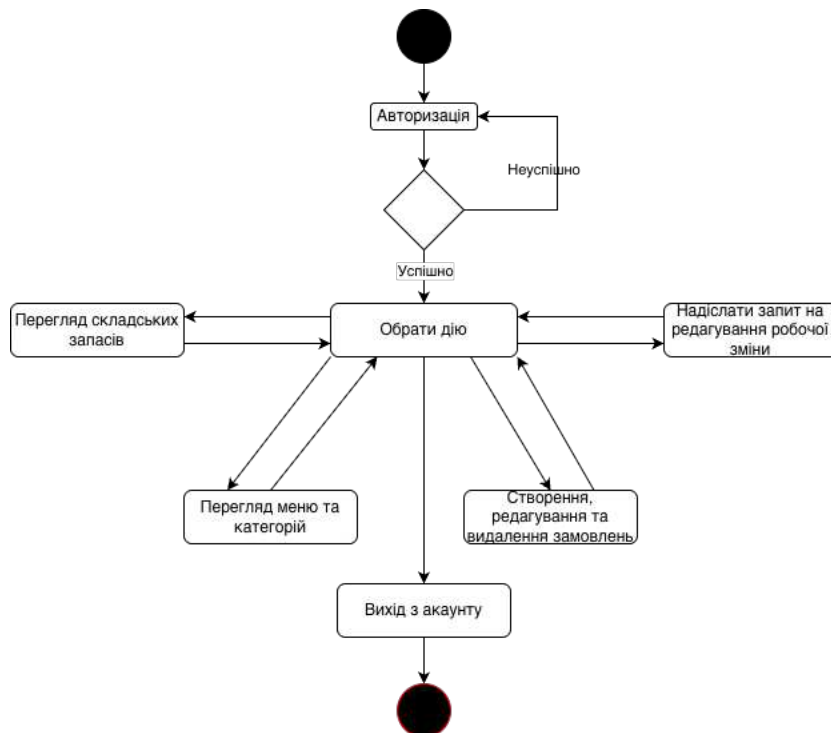


Рисунок 2.6 – Діаграма переходу станів для офіціанта та повара

Діаграма переходів станів для користувачів із неадміністративними ролями відображає послідовність дій, які можуть виконувати працівники ресторану під час роботи з системою. Початковим станом для будь-якого користувача є авторизація, де працівник вводить дані для входу до системи. Якщо авторизація завершується успішно, користувач переходить до стану вибору подальшої дії. У разі некоректних даних або помилки входу система повертає користувача до початкового стану авторизації.

Після входу до системи користувачеві доступні декілька функціональних станів, залежно від його ролі. Одним із них є перегляд складських запасів, що дозволяє ознайомитися з доступною кількістю інгредієнтів або продукції. Цей стан має лише інформаційний характер, і після перегляду працівник може повернутися до головного меню для вибору іншої дії.

Також користувач може перейти до стану перегляду меню та категорій, де йому доступний перегляд списку страв, їх категорій, описів і вартості. З цього стану не передбачено розширених дій, оскільки редагування меню належить до функціоналу адміністратора.

Окремо передбачено стан створення, редагування та видалення замовлень, який є ключовим для офіціантів та барменів. У цьому стані користувач може оформити нове замовлення, змінювати його склад або видаляти некоректно створені записи. Після завершення дій з замовленням можливий перехід назад до вибору дії.

Ще один важливий стан - надсилання запиту на зміну робочої зміни, який використовується працівниками для формування заявки на зміну графіка. Ця дія не змінює дані миттєво, а лише формує запит, що буде опрацьовано адміністратором. Після відправлення запиту користувач повертається у головне меню.

Завершальним станом є вихід з облікового запису, після чого робота користувача із системою припиняється, і діаграма переходить до фінального стану. У подальшому для продовження роботи необхідне повторне проходження авторизації.

Висновок до розділу 2

У межах другого розділу було проаналізовано наявні програмні рішення для управління командною взаємодією в ресторанній сфері, визначено їхні переваги, недоліки та функціональні обмеження. Це дало змогу окреслити ключові вимоги до майбутньої системи та сформуванати власну концепцію її побудови. На основі проведеного аналізу було розроблено початкову архітектурну модель веб-додатку та визначено основні сценарії взаємодії користувачів, що стало фундаментом для побудови діаграми варіантів використання.

У розділі створено концептуальну та логічну схеми бази даних, які відображають структуру майбутньої інформаційної системи та взаємозв'язки між даними, необхідними для підтримки командної роботи в ресторані. Побудовані моделі забезпечили узгодженість між функціональними вимогами та даними, що оброблятимуться системою.

Також визначено принципи інформаційного та технічного забезпечення, сформовано вимоги до програмної й апаратної частини, що дозволяють забезпечити стабільність роботи та масштабованість платформи. У результаті сформована цілісна концепція системи управління командною взаємодією в ресторані, яка поєднує структуровану архітектуру, продуману модель даних та чітко описані сценарії роботи користувачів.

РОЗДІЛ 3. РОЗРОБКА ПРОЄКТНИХ РІШЕНЬ ЩОДО ВЕБ-ДОДАТКУ ДЛЯ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ

3.1. Розробка схеми структури інформаційної системи

Після аналізу вимог до майбутньої системи та оцінки сучасних технологічних рішень було прийнято рішення використовувати фреймворк Spring MVC як основу серверної частини інформаційної системи управління рестораном. Дане рішення зумовлене тим, що Spring MVC реалізує концепцію архітектурного шаблону Model–View–Controller, яка забезпечує логічне розділення програмних компонентів. Такий підхід дозволяє ізолювати бізнес-логіку в моделі, обробку запитів - у контролері, а представлення інформації - у шарі уявлення. Завдяки цьому система стає значно гнучкішою, простішою в супроводі та придатною до масштабування [17].

Використання Spring MVC в поєднанні з багаторівневою клієнт-серверною архітектурою дає змогу формалізувати структуру взаємодії між клієнтською частиною та сервером. Це забезпечує оптимальний розподіл навантаження, оскільки обробка бізнес-логіки, авторизації, перевірки прав доступу та взаємодії з базою даних виконується саме на серверному рівні. Такий підхід підвищує продуктивність усієї системи, а також скорочує обсяг даних, що передаються мережею, що є особливо важливим у випадках активної роботи персоналу ресторану у години пікового навантаження.

Крім того, Spring MVC забезпечує високу безпеку за рахунок тісної інтеграції зі Spring Security, що дозволяє централізовано керувати доступом до окремих модулів системи відповідно до ролей користувачів [18]. Такий підхід унеможливорює доступ неавторизованих осіб до чутливої інформації - наприклад, фінансових даних, записів про зарплати або системних налаштувань. Подібна гнучкість політик доступу робить Spring MVC одним із найбільш придатних фреймворків для корпоративних застосунків та комплексних адміністративних систем.

Загалом, сукупність механізмів, які надає Spring MVC, дозволяє сформувати надійну та масштабовану архітектуру системи, здатну підтримувати складні бізнес-процеси ресторанного менеджменту та забезпечувати стабільну роботу всіх модулів. Описані переваги підтверджуються офіційною документацією Spring Framework, де підкреслюється універсальність і продуктивність MVC-підходу в серверних веб-додатках.

Основні компоненти інформаційної підсистеми:

1. Веб-додаток (Spring MVC):

1.1.Інтерфейс для користувачів (адміністратори, офіціанти, кухарі, бармени, шеф-кухарі, бухгалтера) перегляд складських запасів, перегляд меню та категорій, створення редагування та видалення замовлень, створення запиту на редагування робочої зміни.

1.2.Інтерфейс для адміністраторів для перегляду і редагування складських запасів, управління персоналом, меню та категоріями страв, опрацювання замовлень, перегляд статистики та налаштування параметрів розрахунку заробітної плати. Також адміністратор може формувати графік роботи, змінювати умови робочих змін та виконувати вихід із облікового запису.

1.3.JavaScript-функції застосовуються для покращення інтерактивності користувацького інтерфейсу

1.4.Thymeleaf використовується для відображення даних.

2. Система авторизації:

2.1.Розподіл доступу до функціональних модулів здійснюється відповідно до ролей користувачів (адміністратор, офіціант, кухар, бухгалтер та інші), що забезпечує контрольоване використання можливостей системи.

2.2.Механізм автентифікації та реєстрації користувачів реалізовано на основі Spring Security, який гарантує безпечний вхід до системи та перевірку правомірності дій користувача.

3. База даних:

3.1.База даних виступає єдиним сховищем, що накопичує відомості про працівників, графіки роботи, складські запаси, інгредієнти, позиції меню,

замовлення, відгуки клієнтів, та інші критично важливі дані, необхідні для стабільного функціонування системи.

3.2. Використання ORM Hibernate для взаємодії з базою даних.

4. Функціональні модулі:

4.1. Модуль інвентаризації забезпечує ведення обліку інгредієнтів, контроль залишків на складі та автоматичне оновлення інформації після виконання замовлень.

4.2. Модуль управління меню дозволяє формувати перелік категорій і страв, редагувати їхні характеристики та видаляти застарілі позиції.

4.3. Модуль опрацювання замовлень відповідає за створення, редагування та відстеження статусів замовлень, а також взаємодію між офіціантами та кухарями.

4.4. Модуль аналітики та звітності забезпечує побудову фінансових, статистичних та операційних звітів, необхідних для управління рестораном.

4.5. Модуль управління персоналом охоплює створення та редагування профілів працівників, призначення ролей та прав доступу, а також контроль активності користувачів.

4.6. Модуль формування робочих графіків дозволяє адміністраторам створювати та редагувати розклади змін, узгоджувати запити працівників на зміну графіка та експортувати готові таблиці у формат Excel.

4.7. Модуль розрахунку заробітної плати забезпечує налаштування параметрів нарахування, перегляд історії виплат та автоматизацію обчислень на основі виконаних змін або відпрацьованого часу.

4.8. Модуль обробки клієнтських відгуків збирає оцінки, коментарі та зауваження клієнтів, дозволяючи переглядати їх та використовувати для аналізу якості обслуговування.

4.9. Модуль push-сповіщень забезпечує передачу миттєвих повідомлень персоналу щодо нових замовлень, змін статусів або важливих внутрішніх подій.

На рисунку 3.1 представлено схему взаємодії програмних модулів.

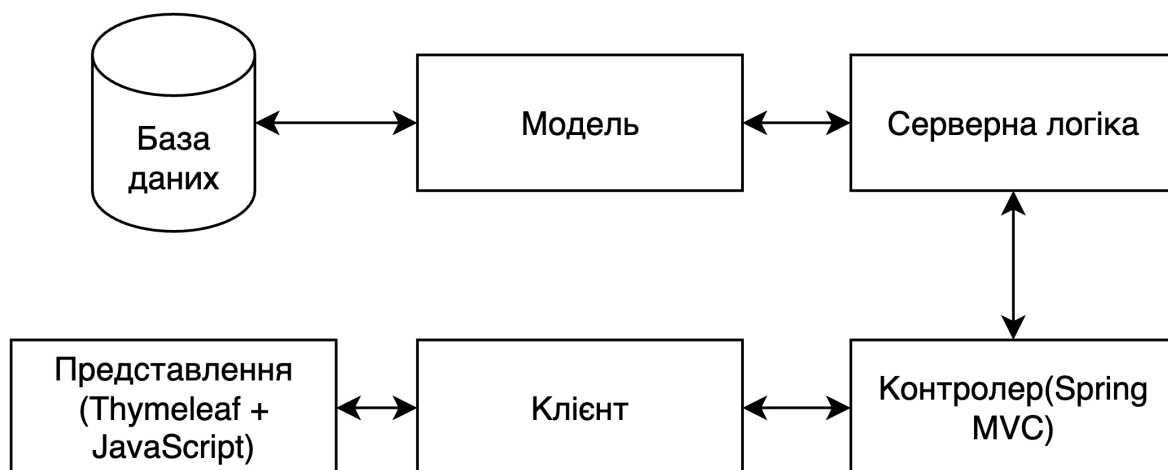


Рисунок 3.1 – Схема взаємодії програмних модулів

Опис взаємодії модулів:

Користувач взаємодіє з інформаційною системою через веб-інтерфейс, який працює у браузері або на мобільному пристрої. Інтерфейс забезпечує доступ до різних можливостей залежно від ролі: перегляд складських запасів, обробка замовлень, робота з меню, категоріями та змінами графіка. Для адміністраторів передбачено розширений функціонал, що охоплює керування персоналом, перегляд статистики, налаштування зарплат та модифікацію робочих змін. Інтерактивність інтерфейсу забезпечується використанням JavaScript-функцій, які дозволяють оновлювати дані без повного перезавантаження сторінки та спрощують роботу користувача. Відображення інформації здійснюється за допомогою шаблонізатора Thymeleaf, що формує сторінки на основі серверних даних та забезпечує коректний їх рендеринг.

Доступ до функціональних можливостей здійснюється на основі моделі ролей, яка визначає, які дії доступні працівнику: адміністратору, офіціанту, кухарю, бармену, бухгалтеру чи шеф-кухарю. Кожна роль має унікальний набір дозволів, що гарантує безпечний і контрольований доступ до ресурсів. Процес автентифікації реалізовано за допомогою Spring Security - він забезпечує захищений вхід користувачів, перевірку їхніх повноважень та підтвердження права виконання певних операцій у системі.

Уся важлива інформація системи зберігається у єдиному централізованому сховищі. База даних містить відомості про співробітників, графіки змін, інгредієнти, складські залишки, меню, категорії страв, замовлення, відгуки клієнтів, нарахування заробітної плати та інші сутності, що формують ядро системи. Для роботи з БД застосовується ORM-технологія Hibernate, завдяки якій взаємодія з даними виконується через об'єкти, без потреби у прямому написанні SQL-запитів. Це підвищує безпеку, прискорює розробку та спрощує підтримку системи.

Роботу системи забезпечує комплекс незалежних, але пов'язаних між собою модулів. Модуль інвентаризації контролює складські залишки та автоматично оновлює кількість інгредієнтів після формування замовлень. Модуль управління меню дозволяє редагувати страви та категорії, підтримуючи актуальність меню ресторану. Модуль обробки замовлень відповідає за створення, зміну статусів та передачу інформації між офіціантом і кухнею. Модуль звітності формує фінансові, статистичні та аналітичні документи. Модуль керування персоналом дозволяє адміністратору створювати профілі працівників і керувати їхніми правами. Модуль робочих графіків надає можливість створювати розклади та опрацьовувати запити на зміну зміни. Модуль заробітної плати автоматизує розрахунок виплат на основі відпрацьованого часу та змін. Модуль відгуків збирає оцінки клієнтів, а модуль push-сповіщень забезпечує миттєве інформування працівників про нові або змінені події в системі.

3.2 Рішення з програмного забезпечення

У процесі розробки інформаційної системи управління рестораном було сформовано стек технологій, який забезпечує стабільність, масштабованість і простоту подальшого розширення функціоналу. Основу серверної частини становить фреймворк Spring MVC, що реалізує архітектурну модель Model–View–Controller та дозволяє чітко розділити бізнес-логіку, контролери та рівень

представлення. Це суттєво полегшує підтримку коду й забезпечує можливість поступового розвитку системи відповідно до змін у предметній області.

Для реалізації механізмів авторизації та захисту доступу застосовано Spring Security, який відповідає за автентифікацію, перевірку прав користувачів і безпечну взаємодію з ресурсами системи.

Робота з базою даних здійснюється за допомогою ORM-фреймворку Hibernate, що мінімізує необхідність прямого написання SQL-запитів і сприяє зручному відображенню об'єктної моделі на реляційну структуру даних. Як основну систему управління базами даних обрано PostgreSQL - високопродуктивну та надійну СКБД із потужною підтримкою транзакцій, складних зв'язків і індексації. PostgreSQL повністю відповідає вимогам ресторанної системи, яка містить сутності зі значною кількістю залежностей: замовлення, інгредієнти, облік складу, графіки роботи, зарплати та інші пов'язані дані.

Для забезпечення інтерактивної взаємодії між працівниками ресторану, зокрема миттєвих push-сповіщень, у системі використано модуль spring-boot-starter-websocket. Він дозволяє встановлювати постійне двостороннє з'єднання між клієнтом і сервером, завдяки чому кухарі, адміністратори та офіціанти оперативно отримують повідомлення про нові замовлення, зміну статусів або важливі внутрішні події.

Формування динамічних Excel-файлів, необхідних для експорту робочих графіків, звітів або статистики, реалізовано за допомогою бібліотеки Apache POI (poi-ooxml). Цей інструмент забезпечує створення, редагування та стилізацію документів формату XLSX без використання сторонніх сервісів, що робить систему автономною та гнучкою у налаштуванні.

Клієнтська частина побудована з використанням HTML, CSS і JavaScript, що дає змогу сформувати зручний та адаптивний інтерфейс. Для серверного рендерингу сторінок застосовується Thymeleaf, який інтегрується зі Spring-компонентами та забезпечує коректне виведення даних користувачам.

У сукупності цей набір технологій формує надійне середовище розробки, у якому легко реалізовувати складні бізнес-процеси ресторану, масштабувати функціонал та забезпечувати стабільну взаємодію всіх учасників робочого процесу.

3.3. Розробка діаграми компонентів

Діаграми компонентів UML надають концептуальну картину взаємодії між різними системами. Можуть бути присутні як аспекти логічного, так і фізичного моделювання. Крім того, компоненти автономні. Це модульний системний елемент в UML, який можна замінити на альтернативні. Вони містять конструкції будь-якої складності та є самодостатніми. Лише через інтерфейси вкладені частини взаємодіють з іншими компонентами. Крім того, компоненти мають свої інтерфейси, але вони також можуть отримати доступ до операцій і служб інших компонентів за допомогою їхніх інтерфейсів. На діаграмі компонентів інтерфейси також показують зв'язки та залежності в архітектурі програмного забезпечення[19].

На рисунку 3.2 зображено діаграму компонентів системи.

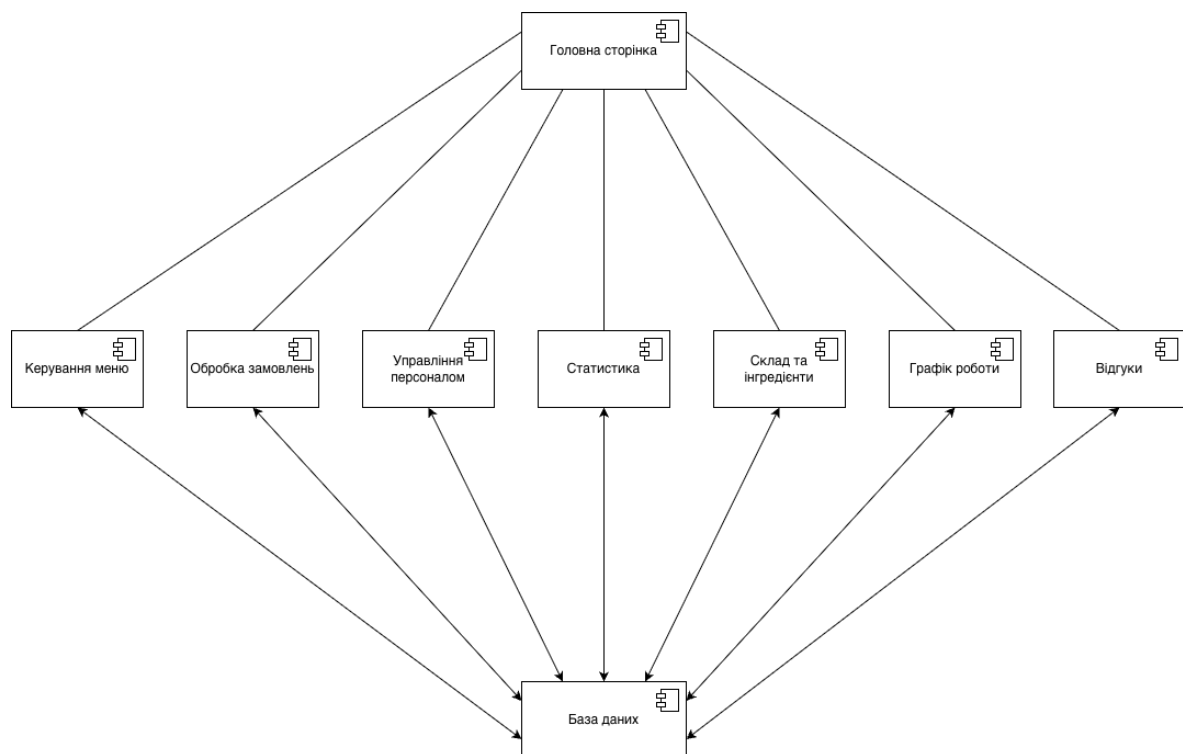


Рисунок 3.2 – Діаграма компонентів

3.4. Розробка діаграми розгортання підсистеми

Діаграма розгортання відображає графічне представлення інфраструктури, на яку буде розгорнуто додаток: топологію системи і розподіл компонентів по її вузлах, а також з'єднання - маршрути передачі даних між вузлами. Діаграма допомагає раціональніше організувати компоненти, від чого залежить в числі іншого і продуктивність системи, а також вирішити допоміжні завдання, наприклад, пов'язані з безпекою [20].

Діаграма розгортання системи зображено на рисунку 3.3.

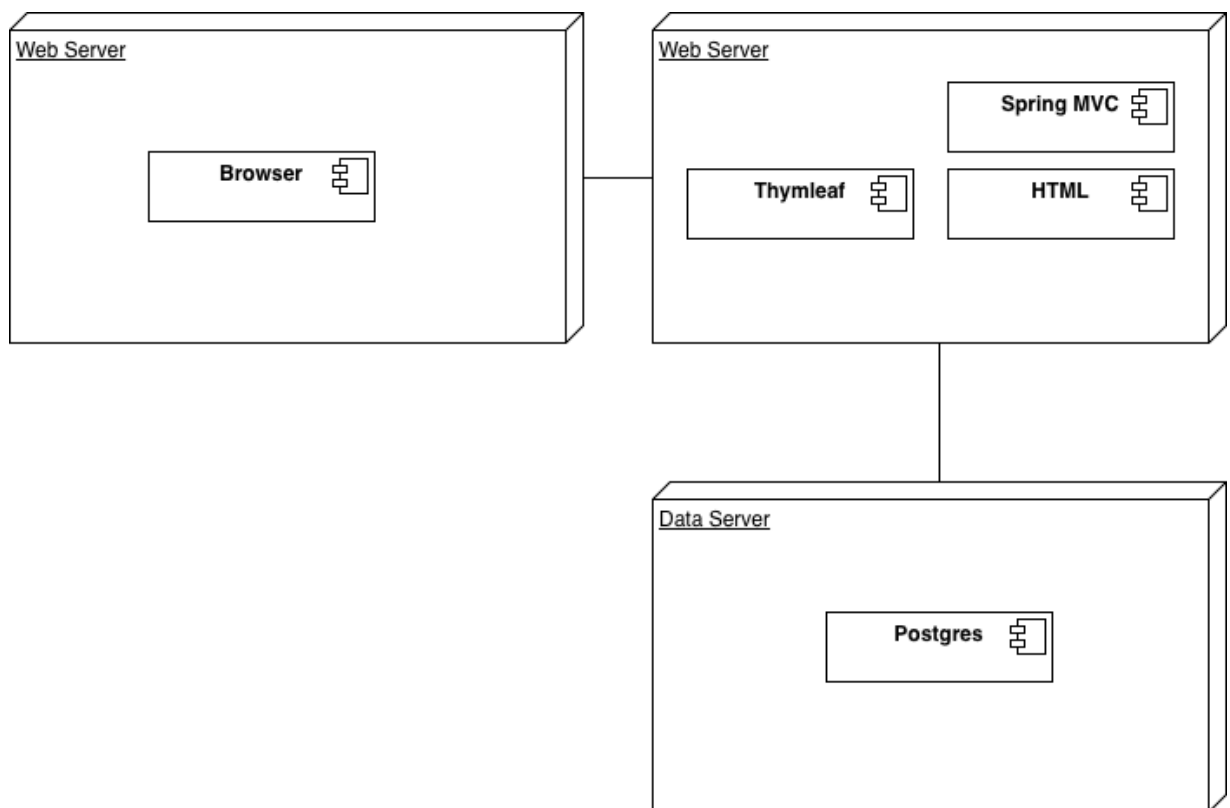


Рисунок 3.3 – Діаграма розгортання системи

Висновок до розділу 3

У третьому розділі було сформовано технічні засади для реалізації веб-додатку, призначеного для управління командною взаємодією в ресторанному середовищі. Проаналізовано можливі технологічні стеки та обґрунтовано вибір мов програмування, бібліотек і фреймворків, які забезпечують оптимальне поєднання продуктивності, безпеки та зручності подальшої підтримки системи.

На основі визначених вимог було розроблено діаграму компонентів, що відобразила структуру основних модулів додатку та характер їх взаємодії. Це дозволило чітко окреслити відповідальність кожної частини системи та забезпечити логічний поділ функціональності. Діаграма розгортання, своєю чергою, показала архітектуру фізичного розміщення компонентів, включаючи серверну інфраструктуру, мережеві взаємозв'язки та середовище виконання додатку.

Важливим результатом розділу стало формування узгодженої архітектурної моделі веб-додатку, яка враховує вимоги надійності, масштабованості та безперервної роботи в реальних умовах ресторанної діяльності. Таким чином, у розділі було визначено технічний фундамент, необхідний для подальшої реалізації системи та впровадження її функціональних можливостей.

РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ В РЕСТОРАНІ

4.1 Вступ

Сучасний ресторанний бізнес функціонує в умовах високої конкуренції та постійної потреби у швидкому обробленні інформації. Щодня ресторан оперує значним обсягом даних: замовленнями, інгредієнтами, графіками персоналу, фінансовими показниками та клієнтськими відгуками. Обсяг цих процесів зростає разом зі збільшенням кількості гостей та розширенням меню, що ускладнює контроль і координацію роботи закладу без відповідного програмного забезпечення.

До основних чинників, що впливають на ефективність роботи ресторану, належать:

1. Швидкість оброблення замовлень;
2. Точність обліку інгредієнтів та складських запасів;
3. Злагоджена робота кухні та персоналу залу;
4. Якість обслуговування клієнтів та зворотний зв'язок;
5. Здатність керівництва оперативно отримувати статистичні звіти;
6. Оптимізація графіків роботи та розрахунку заробітної плати.

Наявність цих чинників безпосередньо визначає прибутковість закладу: чим швидше та якісніше працює персонал - тим вище оборот та задоволеність клієнтів. У той же час будь-які затримки, неузгоджені дії або помилки приводять до втрат - як фінансових, так і репутаційних.

Після запуску ресторану власник переслідує основну мету - отримати максимальну вигоду при мінімальних витратах на операційні процеси. Проте витрати можуть включати:

1. Підвищене споживання інгредієнтів через неточний облік;
2. Неефективний розподіл робочих змін;

3. Помилки офіціантів при обробленні замовлень;
4. Затримки між кухнею та залом;
5. Час, витрачений на ручне формування фінансових звітів.

Часто такі непомітні на перший погляд втрати накопичуються і суттєво знижують прибутковість закладу.

Для подолання цих проблем доцільним є впровадження інформаційної системи, яка автоматизує ключові бізнес-процеси ресторану. Використання такого програмного забезпечення дозволяє:

1. Зменшити ризик людських помилок;
2. Оптимізувати використання запасів і зменшити списання;
3. Прискорити комунікацію між кухнею та офіціантами;
4. Підвищити продуктивність персоналу;
5. Забезпечити керівництво аналітичними даними в режимі реального часу;
6. Покращити якість обслуговування клієнтів.

Таким чином, впровадження розробленої інформаційної системи є інвестицією, спрямованою на підвищення операційної ефективності та фінансової стабільності ресторану.

4.2 Розрахунок витрат на створення та експлуатацію системи управління командною взаємодією в ресторані

Розрахунок витрат на розробку та впровадження системи управління командною взаємодією в ресторані залежить від багатьох факторів: складності функціоналу, кількості залучених фахівців, технічної інфраструктури, потреб у підтримці, а також інтеграцій із зовнішніми сервісами. В таблиці 4.1 відображено основні витрати для розробки, розгортання та підтримку системи.

Таблиця 4.1 – Основі витрати для розробки, розгортання та підтримку системи

Категорія витрат	Детальний опис	Орієнтовна сума, грн
Розробники (Backend + Frontend)	2 спеціалісти × 2 місяці	120 000
Тестувальник (QA)	Функціональне + навантажувальне тестування	25 000
UI/UX дизайнер	Макети інтерфейсу, адаптивність, дизайн системи	15 000
Аналітик	Вимоги, моделювання процесів, супровід ТЗ	18 000
DevOps / Адміністратор БД	Налаштування БД, CI/CD, моніторинг	20 000
IDE та розробницьке ПЗ	IntelliJ IDEA Ultimate, WebStorm, інструменти аналізу	4 000 / рік
Сервіси тестування	емулятори, хмарні тест-середовища	3 000
Дизайнерські інструменти	Figma / Adobe XD	1 000
Інфраструктурні утиліти	Моніторинг, логування, DevOps-утиліти	2 000
Сервер під backend	4 vCPU, 8 GB RAM, хмарний VPS	12 000 / рік
База даних PostgreSQL	окремий сервер або managed DB	6 000 / рік
Хостинг статичного фронтенду	CDN або cloud storage	2 000 / рік
Домен + SSL	домен .com/.ua + сертифікат	1 200 / рік
Резервні сервери / Backups	автоматизоване зберігання копій	4 000 / рік

Узагальнюючи наведені витрати, можна зробити висновок, що створення та впровадження інформаційної системи для управління командною взаємодією в ресторані потребує комплексного підходу та участі фахівців різного профілю. Основна частина бюджету припадає на оплату праці команди розробки - бекенд- і фронтенд-розробників, тестувальника, дизайнера та аналітика, які забезпечують повний цикл створення продукту: від формування вимог і макетів до програмування та тестування функціоналу. Додаткові витрати пов'язані з технічною інфраструктурою: налаштуванням серверів, баз даних, CI/CD, забезпеченням домену, SSL, резервного копіювання, а також придбанням професійного програмного забезпечення та інструментів для розробки, дизайну й тестування. Сукупно ці статті витрат формують реальну собівартість розробки системи та відображають масштаб і складність проєкту, демонструючи, що ефективна й надійна цифрова платформа потребує як людських ресурсів, так і стабільної технічної інфраструктури. Сукупно всі ці статті формують реальну собівартість створення системи: орієнтовна загальна сума інвестицій становить

близько 233 200 грн (включно з одноразовими витратами на розробку та річними витратами на інструменти й інфраструктуру).

4.3 Економічна ефективність та впровадження інформаційної системи управління командною взаємодією в ресторані

Економічна результативність запровадження інформаційної системи у ресторанному бізнесі залежить від широкого спектра чинників, більшість з яких проявляються вже після початку використання системи. Витрати на розробку та впровадження можуть істотно відрізнятися залежно від масштабу ресторану, складності бізнес-процесів, кількості працівників та способу організації комунікації між ними. Нижче представлено таблицю 4.2 де описано основні групи факторів, що впливають на економічну доцільність інформаційної системи.

Таблиця 4.2 - Основні групи факторів, що впливають на економічну доцільність інформаційної системи

Функціональний ефект	Оцінка економії часу, люд.-год/міс	Орієнтовна ставка, грн/год	Економічний ефект, грн/міс
Скорочення часу комунікації між залом і кухнею завдяки push-сповіщенням	58	125 (офіціанти/кухарі)	≈ 7 250
Зменшення часу на оформлення і редагування замовлень через єдину систему	80	120 (офіціанти)	≈ 9 600
Автоматизація формування управлінських звітів (продажі, завантаженість, відгуки)	5	180 (адміністратор/менеджер)	≈ 900
Автоматизація складання та коригування робочих графіків персоналу	8	180 (адміністратор/менеджер)	≈ 1 440
Зменшення часу на ручний контроль складських залишків та інвентаризацію	6	150 (відповідальний за склад/менеджер)	≈ 900

Узагальнюючи розрахунки, впровадження системи дає сумарну економію близько 157 людино-годин на місяць, що за прийнятими ставками відповідає

орієнтовному фінансовому ефекту на рівні $\approx 20\,000$ грн/міс за рахунок скорочення витрат робочого часу офіціантів, кухарів та адміністративного персоналу.

Модель ціноутворення, побудована на основі ролей користувачів, дозволяє максимально точно відобразити реальне навантаження на систему, оскільки офіціанти, кухарі та адміністратори виконують принципово різні функції. Такий підхід робить оплату гнучкою та справедливою: заклад не переплачує за тих співробітників, які використовують обмежений функціонал, і водночас отримує повний спектр можливостей для ключових ролей. Для розробника така модель є більш стабільною, оскільки залежить не лише від кількості персоналу, а й від типу їх взаємодії з системою. Це дає можливість формувати гнучкі пакети для різних типів закладів - від маленької кав'ярні до великого ресторанного комплексу. Ціноутворення відображено в таблиці 4.3.

Таблиця 4.3 - Ціноутворення

Роль користувача	Опис функціоналу	Місячна вартість, грн	Коментар
Офіціант	Створення замовлень, передача на кухню, статуси, перегляд меню, робота зі столами	150 грн/міс	Найбільша кількість користувачів, базове навантаження
Кухар	Приймання замовлень, оновлення статусів, push-сповіщення	120 грн/міс	Менше функцій, але постійне використання
Адміністратор / Менеджер	Графіки, управління персоналом, інвентар, аналітика, перегляд звітів	250 грн/міс	Найширший функціонал, доступ до всіх модулів
Власник / Директор	Аналітика, фінанси, доступ до dashboard, перегляд KPI	200 грн/міс	Роль з високим рівнем доступу, але рідкісне використання

Розрахуємо період, за який дохід покриє витрати на покупку застосунка для закладу, якому працює:

- 10 офіціантів
- 6 кухарів
- 3 адміністратори
- 1 власник

Сума за місяць для такого закладу:

- Офіціанти: $10 \times 150 = 1\,500$ грн

- Кухарі: $6 \times 120 = 720$ грн
- Адміністратори: $3 \times 250 = 750$ грн
- Власник: $1 \times 200 = 200$ грн

Разом: 3 170 грн / місяць

$$\text{Період окупності} = \frac{\text{Витрати на придбання}}{\text{Очікуєий фінансовий ефект} - \text{Витрати на придбання}}$$

$$\text{Період окупності} = \frac{3170}{20000 - 3170} = 0.188 \text{ місяця} \approx 6 \text{ днів}$$

Таким чином, впровадження інформаційної системи для управління командною взаємодією є економічно високоефективним для закладу ресторанного типу. Враховуючи орієнтовну щомісячну економію робочого часу персоналу на рівні $\approx 20\,000$ грн та вартість підписки 3 170 грн на місяць, чистий економічний ефект становить 16 830 грн щомісяця. Це означає, що система повністю окуповується менш ніж за шість днів роботи, а надалі генерує стабільний позитивний фінансовий результат.

Для оцінки економічної доцільності проєкту необхідно визначити, скільки закладів повинні підключитися до системи, щоб повністю покрити сумарні витрати на розробку та інфраструктуру, що становлять 233 200 грн. Оскільки один ресторан із середньою структурою персоналу (10 офіціантів, 6 кухарів, 3 адміністратори та 1 власник) приносить близько 38 040 грн на рік, саме ця величина є базою для розрахунку окупності. Порівняння загальних інвестицій із річним доходом від одного клієнта дозволяє визначити мінімальну кількість закладів, необхідних для покриття витрат і виходу проєкту на беззбитковий рівень.

$$\text{Кількість закладів} = \frac{\text{Витрати на розробку і підтримку}}{\text{Прибуток від закладу за рік}}$$

$$\text{Кількість закладів} = \frac{233\,200}{38\,040} = 6.13 \text{ закладів}$$

Отже, з урахуванням округлення до цілого: потрібно мінімум 7 закладів такого масштабу, щоб повністю окупити розробку системи та витрати на її підтримку протягом першого року.

Висновок до розділу 4

У четвертому розділі було здійснено поглиблений економічний аналіз доцільності впровадження інформаційної системи управління командною взаємодією в ресторанному бізнесі. Проведені розрахунки витрат на розробку, розгортання та подальшу підтримку дали можливість сформувати цілісну картину інвестиційної складової проєкту та визначити її вплив на фінансову модель закладу. Загальні витрати на створення системи становлять 233 200 грн, до яких також додаються щорічні витрати на підтримку інфраструктури.

Оцінювання економічних вигід підтвердило, що автоматизація ключових бізнес-процесів - комунікації між персоналом, оформлення замовлень, управління графіками, інвентаризації та контролю виконання завдань - забезпечує суттєве скорочення трудозатрат. Сумарний економічний ефект для одного закладу з 20 працівниками становить приблизно 20 000 грн на місяць, що означає, що ресторан повністю окупує витрати на придбання доступу до системи менш ніж за 6 днів роботи.

Розрахунок доходів від реалізації програмного продукту показав, що з одного закладу система приносить близько 38 040 грн на рік. За таких умов мінімальна кількість клієнтів, необхідна для покриття витрат на розробку, становить щонайменше 7 закладів на рік. Таким чином, за умови залучення щонайменше 7 клієнтів система виходить на точку беззбитковості, а подальше масштабування забезпечує стабільний прибуток розробнику.

Отже, розроблена інформаційна система є не лише технологічним інструментом оптимізації операцій, але й економічно обґрунтованою інвестицією. Вона забезпечує швидку окупність для ресторану, зменшує його операційні витрати, підвищує продуктивність персоналу та створює реальні довгострокові переваги як для бізнесу-клієнта, так і для розробника.

РОЗДІЛ 5. ОХОРОНА ПРАЦІ

5.1 Вступ

Під час виконання дипломної роботи більша частина завдань студента пов'язана з роботою за комп'ютером. Таке навантаження є характерним для спеціалістів у галузі інформаційних технологій, але навіть у рамках навчального проєкту воно має свої особливості та ризики. Тривале сидіння, статичне напруження тіла, висока концентрація уваги та постійна робота з монітором можуть спричиняти втому, зниження працездатності та неприємні фізіологічні наслідки. Частину з них можна попередити завдяки організації правильного режиму роботи і відпочинку, інші потребують дотримання рекомендацій фахівців.

Метою цього розділу є аналіз умов, у яких студент виконує дослідження та програмну розробку, а також виявлення можливих ризиків, пов'язаних з тривалою роботою за комп'ютерною технікою. На основі чинних норм охорони праці оцінюються фактори впливу, визначаються потенційно небезпечні ситуації, порівнюються умови роботи з нормативними вимогами та пропонуються заходи для мінімізації негативного впливу на здоров'я під час створення та тестування програмної системи.

5.2 Аналіз шкідливих та небезпечних факторів

Робоче місце програміста-початківця або студента, який тривалий час працює за комп'ютером, може піддаватися впливу різних шкідливих та небезпечних факторів. Вони впливають як на фізичне здоров'я, так і на психоемоційний стан, тому важливо враховувати їх ще під час організації робочого простору. Нижче наведено основні групи ризиків, що найчастіше виникають при неправильному використанні техніки чи нерациональній роботі за комп'ютером.

1. Ергономічні особливості робочого місця

1.1.Невідповідні меблі та обладнання. Неправильно підібраний стіл або крісло можуть спричинити викривлення постави, перенапруження спини та ший.

1.2.Розташування техніки. Монітор, який стоїть занадто високо чи низько, або клавіатура, що вимагає постійного напруження кистей, негативно впливають на руки, плечі та зір.

2. Зорове навантаження

2.1.Блиск та яскравість екрана. Надто яскравий монітор, відблиски від ламп чи вікна можуть викликати головний біль і швидко втомлюваність очей.

2.2.Тривала концентрація на екрані. Довге фіксування погляду без перерви сприяє синдрому «сухого ока» та відчуттю «піску» в очах.

3. Фізичні та емоційні навантаження

3.1.Повторювані дії. Одноманітне натискання клавіш протягом багатьох годин виснажує м'язи передпліччя та плечового пояса.

3.2.Психологічний стрес. Виконання складних завдань під обмежені дедлайни може спричиняти нервові напруження та емоційне вигорання.

4. Малорухливий спосіб життя

4.1.Тривале сидіння. Фіксована поза протягом кількох годин поспіль негативно впливає на кровообіг, сприяє появі болю в попереку та знижує рівень фізичної активності.

5. Технічні ризики

5.1.Стара чи несправна техніка. Використання застарілих пристроїв може призвести до збоїв, перегрівання, непередбачуваних ситуацій або навіть травм при експлуатації.

6. Рутинність і перевтома

6.1.Одноманітність завдань. Довготривала робота над однотипними задачами знижує мотивацію, спричиняє втому та зменшує продуктивність.

Щоб зменшити вплив небезпечних факторів, слід ретельно підійти до облаштування робочого місця. Важливо користуватися ергономічними меблями, правильно виставити монітор та периферійні пристрої, періодично змінювати

положення тіла. Регулярні перерви сприяють розвантаженню очей і м'язів, а короткі вправи допомагають підтримувати тонус протягом дня.

Особливу увагу слід звернути на освітлення - поєднання природного та штучного світла мінімізує втому очей і створює комфортні умови роботи. Не менш важливим є оновлення техніки та програмного забезпечення, що гарантує стабільність роботи та зменшує кількість технічних ризиків.

Для збереження психоемоційного здоров'я студенту або молодому фахівцю корисно практикувати методи релаксації, такі як дихальні техніки чи короткі паузи з мінімальною активністю. Планування робочих завдань і чергування діяльності допомагають уникати стресу, а підтримка колег і близьких - легше долати напружені періоди.

Усе це разом створює безпечне та комфортне середовище, що позитивно впливає на здоров'я та підвищує ефективність роботи програміста чи студента.

5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій

Тривала робота за комп'ютером неминуче пов'язана зі значним зоровим навантаженням, що може негативно впливати на стан очей та викликати швидку втому. Одним із ключових факторів, що дозволяє мінімізувати такі ризики, є правильно організоване освітлення робочого місця. Якісне світлове середовище не лише покращує психофізіологічний стан працівника, але й підвищує продуктивність та зменшує рівень стомлюваності протягом робочого дня.

Оптимальні умови праці передбачають поєднання природного освітлення у світлий час доби та ефективного штучного освітлення у вечірній чи темний період. Згідно з вимогами ДБН В.2.5-28-2006 «Природне і штучне освітлення», робочі приміщення повинні відповідати визначеним нормам освітленості, які регламентуються Міністерством будівництва, архітектури та житлово-

комунального господарства України. Крім того, у зонах, де можлива небезпека відключення електроживлення, має бути передбачене аварійне освітлення.

Система освітлення в робочому приміщенні повинна забезпечувати рівномірний розподіл світлового потоку, комфортну яскравість, відсутність різких тіней, блисків та пульсацій. Важливим є правильний підбір колірної температури та контрастності, оскільки ці параметри впливають на адаптацію зору та загальне самопочуття користувача.

Робоче місце, на якому виконується дипломний проєкт, освітлюється природним світлом протягом дня та штучними світильниками у вечірній час. При цьому враховується можливий вплив різних факторів, таких як розташування монітора відносно вікна, інтенсивність світлового потоку, відблиски від поверхонь. Правильно організоване освітлення дозволяє зменшити навантаження на очі, уникнути перенапруги зорового апарату та створити сприятливі умови для ефективної роботи.

Таким чином, відповідно спроектована система освітлення є важливим елементом забезпечення безпечного та комфортного робочого середовища, що позитивно впливає як на якість виконання завдань, так і на загальний стан здоров'я працівника.

5.4 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій

5.4.1 Заходи щодо зменшення впливу

Для зниження негативного впливу шкідливих та небезпечних факторів, що можуть виникати під час роботи з комп'ютером, доцільно впроваджувати комплекс рекомендацій, спрямованих на покращення умов праці та збереження здоров'я користувача.

1. Фізична активність:

- 1.1.Короткі розминки: Під час тривалої роботи обов'язково виконувати легкі фізичні вправи, розтяжки або просто піднятися та пройтись для стимулювання кровообігу.
- 1.2.Зміна позиції: Періодично змінювати положення тіла, щоб уникати затерпання м'язів та перевантаження спини.
2. Очисний комфорт та профілактика втоми очей:
 - 2.1.Перерви для очей: Рекомендовано застосовувати правило 20–20–20 - кожні 20 хвилин робити паузу та дивитися на об'єкт, розташований не менше ніж за 20 метрів протягом 20 секунд.
 - 2.2.Антивідблискові засоби: Використання захисних фільтрів або правильне налаштування яскравості та контрасту монітора для зменшення перенапруження очей.
3. Організація робочого місця:
 - 3.1.Ергономічне розташування обладнання: Монітор має бути встановлений на рівні очей, а клавіатура - розміщена так, щоб мінімізувати напруження в зап'ястях.
 - 3.2.Меблі: Використання зручного ергономічного крісла зі спинкою, що підтримує поперек, а також робочого столу відповідної висоти.
4. Різноманітність завдань:
 - 4.1.Чередування типів роботи: Щоб уникнути монотонності, варто змінювати види завдань, поєднуючи інтелектуальну діяльність із технічною або більш простою роботою.
 - 4.2.Короткі перерви: Регулярне відволікання від монотонних задач допомагає зберігати концентрацію та продуктивність.
5. Оптимізація освітлення:
 - 5.1.Комбіноване освітлення: Бажано використовувати природне світло у денний час, а в затемнений період - рівномірне штучне освітлення, яке не створює тіней і не подразнює очі.
 - 5.2.Уникнення блисків: Освітлювальні прилади мають бути розташовані так, щоб світло не відбивалось у моніторі.

6. Правильне використання техніки:

6.1.Оновлення ПЗ: Регулярне оновлення операційної системи та програм є необхідним для стабільної роботи й зниження ризиків збоїв.

6.2.Справність обладнання: Перевірка стану комплектуючих допомагає уникнути небажаних технічних ситуацій під час роботи.

7. Безпека та захист даних:

7.1.Використання антивірусних систем: Своєчасні оновлення та перевірка комп'ютера на наявність загроз забезпечують стабільну роботу та захищають інформацію.

7.2.Обережність у мережі: Дотримання базових правил кібербезпеки під час роботи з вебресурсами.

8. Психоемоційне здоров'я:

8.1.Психологічна підтримка: У випадку високих навантажень важливо мати можливість звернутися по допомогу до спеціаліста або знайти підтримку серед колег чи близьких.

8.2.Релаксація: Методи відновлення, такі як глибоке дихання, музика для концентрації чи коротка медитація, сприяють зменшенню стресу.

Виконання запропонованих рекомендацій дозволить значною мірою зменшити вплив шкідливих факторів, а дотримання їх на регулярній основі сприятиме покращенню загального самопочуття, працездатності та продуктивності. Робота за комп'ютером неминуче пов'язана з навантаженням, однак грамотна організація робочого процесу допомагає запобігти вигоранню та формує здорові професійні звички.

5.4.2 Техніка безпеки при роботі з комп'ютером

Робота за комп'ютером може бути безпечною та комфортною лише за умови дотримання рекомендацій щодо організації робочого місця та здорового режиму праці. Нижче наведено основні поради, які допомагають зменшити вплив негативних факторів під час роботи за ПК:

1. Організація ергономічного простору:

1.1.Стіл і стілець повинні бути налаштовані так, щоб спина залишалася прямою, а ноги - зігнутими під кутом 90°. Ергономічні меблі значно знижують втому та навантаження.

1.2.Монітор встановлюється на рівні очей, на відстані 50–70 см, що дозволяє уникнути зайвого напруження зору та шиї.

2. Освітлення робочої зони:

2.1.Використовуйте поєднання природного освітлення з якісним штучним світлом. Правильне освітлення допомагає усунути блиски та покращити комфорт для очей.

2.2.Антиблікові фільтри або штори можуть зменшити віддзеркалення від вікон та ламп.

3. Періодичні перерви та активність:

3.1.Кожні 20–30 хвилин робіть коротку паузу, щоб зняти напругу з очей та спини - достатньо відірвати погляд від монітора чи пройтися.

3.2.Виконуйте легкі розтяжки для плечей, шиї та спини, щоб підтримувати тонус та попереджати перенапруження м'язів.

3.3.Правильне користування комп'ютерною технікою:

3.4.Розміщуйте монітор так, щоб світло від ламп і вікон не потрапляло безпосередньо на екран.

3.5.Використовуйте клавіатуру і мишу так, щоб руки лишалися у природному положенні, а зап'ястя не зазнавали надмірного тиску.

3.6.Надійно фіксуйте кабелі, щоб уникнути небезпеки спотикання та зайвого безладу.

4. Комфортні умови для продуктивної роботи:

4.1.Забезпечте достатній простір для рухів та зручне розташування робочих матеріалів.

4.2.Використовуйте меблі та аксесуари, які сприяють правильній поставі та зменшують навантаження на суглоби та м'язи.

Усі наведені рекомендації допомагають зберігати здоров'я під час роботи за комп'ютером та створюють комфортне й безпечне робоче середовище.

Дотримуючись цих принципів, можна суттєво зменшити ризики перевтоми, погіршення зору та інших негативних наслідків тривалої роботи за ПК.

Остаточно, підтримання техніки безпеки не лише покращує самопочуття працівника, але й сприяє підвищенню продуктивності та якості роботи, що особливо важливо в умовах сучасного інформаційного середовища.

5.5. Розрахунок освітленості робочого місця

Освітлення робочого простору має суттєвий вплив на комфорт та ефективність працівника, особливо у сфері роботи за комп'ютером. Неналежне світлове середовище може спричиняти перенапруження зору, швидку втому та зниження продуктивності. Тому розрахунок освітленості виконується з використанням методу коефіцієнта використання світлового потоку, що дозволяє підібрати оптимальну кількість світильників для конкретного приміщення.

Загальна формула визначення освітленості має вигляд:

$$E = \frac{F * N * K}{S}$$

де:

- E - освітленість робочої площини, лк;
- F - світловий потік одного світильника, лм;
- n - необхідна кількість світильників;
- k - коефіцієнт запасу (зазвичай коливається в межах 1.2–1.4, залежно від старіння ламп і забруднення обладнання);
- S - площа приміщення, м².

Приклад розрахунку освітлення для приміщення 10 м × 15 м і з висотою стелі 3.5 м:

$$S = 10 \times 15 = 150 \text{ м}^2$$

Нормативна освітленість для роботи з ПК - 300–500 лк (ДБН В.2.5-28:2006).

Візьмемо середнє значення E = 420 лк.

Тип світильників: LED-модулі потужністю 25 Вт, світловий потік ≈ 2600 лм.

Коефіцієнт запасу $K=1.3$:

$$F_{\text{потрібний}} = E \cdot S = 420 \times 150 = 63000 \text{ лм}$$

Необхідна кількість світильників:

$$N = \frac{F_{\text{потрібний}}}{F \cdot K} = \frac{63000}{2600 \cdot 1.3} \approx 18.6$$

Отже, округлюємо до 19 світильників.

Для забезпечення нормативної освітленості на рівні приблизно 400–450 лк у приміщенні площею 150 м² доцільно встановити 19 світлодіодних світильників потужністю близько 25 Вт. Така кількість забезпечує рівномірний розподіл світла, відповідає вимогам ДБН та створює комфортні умови для тривалої роботи за комп'ютером.

Висновок до розділу 5

У розділі було проаналізовано умови праці студента під час виконання дипломної роботи та визначено основні фактори ризику, що виникають при тривалій роботі за комп'ютером. Досліджено ергономічні, зорові, фізичні та психологічні навантаження, які можуть негативно впливати на працездатність і здоров'я.

Показано, що правильна організація робочого місця, раціональне освітлення, використання якісного обладнання та дотримання режиму праці й відпочинку суттєво зменшують ризики втоми, перенапруження та професійних порушень. Запропоновані рекомендації включають ергономічне облаштування робочої зони, регулярні перерви, вправи для очей і тіла, а також технічні заходи з безпеки.

З'ясовано, що дотримання норм охорони праці є необхідною умовою для комфортної та ефективної роботи над програмним проєктом. Виконання запропонованих заходів сприяє збереженню здоров'я, підвищенню продуктивності та забезпечує безпечне середовище при роботі з комп'ютерною технікою.

РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

Збереження навколишнього середовища в умовах стрімкого розвитку цифрових технологій набуває дедалі більшого значення. Робота програміста, попри те, що не пов'язана з прямим промисловим виробництвом, також має свій екологічний вплив. Він проявляється як у споживанні енергоресурсів, так і у використанні техніки, створенні електронних відходів та формуванні цифрового сліду. Тому важливо оцінювати як позитивні, так і проблемні аспекти взаємодії програміста з довкіллям.

Проблемні аспекти впливу програмістської діяльності на довкілля включають:

1. Споживання електроенергії
2. Тривала робота за комп'ютером збільшує загальне енергоспоживання. Потужні робочі станції, монітори високої роздільності та периферійні пристрої формують значне навантаження на електричні мережі. У масштабах компаній це призводить до зростання вуглецевого сліду.
3. Електронні відходи (e-waste)
4. Швидке моральне старіння техніки та регулярне оновлення обладнання створюють додаткові об'єми відходів. Неправильна утилізація комп'ютерних комплектуючих призводить до забруднення ґрунтів і вод токсичними речовинами.
5. Цифровий слід і навантаження на дата-центри
6. Кожен запит, збережений файл чи онлайн-процес збільшує навантаження на сервери, які споживають колосальні об'єми енергії. Дата-центри, що забезпечують роботу цифрових сервісів, є одним з найбільших промислових споживачів електрики.
7. Виробництво та транспортування техніки
8. Створення комп'ютерів, мікросхем і периферії супроводжується високим рівнем викидів CO₂, використанням водних ресурсів і рідкоземельних матеріалів. Також значний екологічний слід формує логістика.

9. Неекономне використання ресурсів

Залишення техніки увімкненою у фоновому режимі, неконтрольоване зберігання даних у хмарі та недбале ставлення до енергоощадних режимів збільшують викиди та витрати електроенергії.

Позитивні аспекти та можливості зменшення екологічного впливу включають:

1. Енергоефективні рішення
2. Сучасні технології дозволяють значно знизити споживання електроенергії: енергоощадні процесори, OLED-монітори, режими Sleep/Standby. Використання таких рішень сприяє зменшенню вуглецевого сліду.
3. Дистанційна робота
4. Віддалена робота зменшує потребу у транспорті, що знижує кількість викидів CO₂. Зменшується також навантаження на інфраструктуру та офісні приміщення.
5. Можливість повторного використання або модернізації техніки
6. Замість купівлі нового обладнання можна проводити його апгрейд: збільшення оперативної пам'яті, заміна диска на SSD, підсилення охолодження. Це продовжує строк служби техніки та зменшує кількість відходів.
7. Цифровізація процесів
8. Програмісти підтримують перехід до безпаперового документообігу, автоматизації й оптимізації процесів. Це скорочує використання паперу, витратників і зменшує обсяг фізичних ресурсів.
9. Участь у створенні екологічних цифрових продуктів
10. Розробники мають можливість створювати сервіси, що сприяють екології: системи моніторингу викидів, оптимізації транспорту, енергообліку, аналітики споживання ресурсів тощо.

Підтримання балансу між продуктивністю та відповідальним використанням ресурсів є важливим завданням кожного програміста. Використання енергоощадних технологій, правильна утилізація техніки, контроль власного цифрового сліду та впровадження екологічних підходів у роботі допомагають створити більш стале та безпечне середовище для майбутніх поколінь.

6.2 Забруднення навколишнього середовища

Хоча діяльність програміста не пов'язана з прямими викидами шкідливих речовин чи виробництвом небезпечних матеріалів, ІТ-галузь усе ж має низку прихованих екологічних наслідків, які накопичуються у великих масштабах. Робочі станції, технічне обладнання та інфраструктура, якою користуються фахівці, споживають значні обсяги електроенергії, що впливає на збільшення викидів CO₂, якщо електрика надходить із традиційних джерел, таких як вугільні чи газові електростанції.

Важливою екологічною проблемою є також електронні відходи. Часті оновлення ноутбуків, моніторів, телефонів та іншого обладнання призводять до накопичення токсичних речовин, які за неправильного поводження можуть потрапляти у ґрунт і воду. Ситуацію ускладнює те, що виробництво нової техніки вимагає значних природних ресурсів: рідкісноземельних металів, великих обсягів води та енергії. Видобуток цих матеріалів часто шкодить екосистемам і призводить до виснаження природних резервів.

Не менш вагомим фактором є робота дата-центрів, які забезпечують зберігання та обробку даних. Вони споживають величезну кількість електроенергії та виділяють тепло, через що потребують постійного охолодження. Це створює додаткове навантаження на енергосистему, а у світовому масштабі сприяє зростанню вуглецевого сліду цифрової індустрії.

Окремо варто звернути увагу на транспортний аспект. Хоча багато програмістів працюють віддалено, частина спеціалістів регулярно пересувається на роботу або бере участь у зустрічах та конференціях. Використання автомобілів чи авіатранспорту супроводжується викидами шкідливих газів, що також впливає на стан довкілля. До цього додається виробництво пластикових комплектуючих та аксесуарів, які надзвичайно повільно розкладаються й збільшують обсяг полімерних відходів.

Отже, можна сказати, що програміст не є прямим джерелом забруднення навколишнього середовища, однак сама індустрія цифрових технологій створює

комплексний вплив на природу. Саме тому важливо звертати увагу на енергоефективність обладнання, відповідальне споживання ресурсів та належну утилізацію техніки, щоб мінімізувати екологічний слід сучасних інформаційних технологій.

6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища

Зменшення негативного впливу ІТ-сфери на довкілля дедалі частіше розглядається як важливий елемент екологічної відповідальності компаній та окремих фахівців. Попри те, що робота програміста зазвичай не пов'язана з фізичними викидами або прямим забрудненням, існує низка непрямих чинників, які можуть формувати екологічний слід діяльності в цифровому середовищі. Тому важливо впроваджувати комплекс заходів, спрямованих на раціональне використання ресурсів та мінімізацію впливу на природу.

Одним із найдієвіших напрямів є оптимізація енергоспоживання. Використання комп'ютерів та серверного обладнання, що працюють з високим рівнем енергоефективності, дозволяє суттєво зменшити загальне навантаження на енергосистему. Застосування сучасних процесорів, енергоощадних блоків живлення та правильне налаштування режимів роботи техніки дає можливість скорочувати витрати електроенергії без втрати продуктивності. Це особливо важливо у великих ІТ-компаніях, де кількість робочих місць може бути значною.

Не менш важливою складовою є відповідальна утилізація електронних пристроїв. Застарілі ноутбуки, периферійне обладнання, монітори або комплектуючі містять матеріали, які при неналежному поводженні можуть завдати шкоди довкіллю. Саме тому важливо передавати таку техніку на переробку спеціалізованим організаціям, які здійснюють безпечне розбирання та вилучення матеріалів, придатних для повторного використання. Цей підхід дає змогу зменшити кількість електронних відходів, що потрапляють на сміттєзвалища.

Ще одним напрямом екологічної відповідальності є перехід на віддалений формат роботи у тих випадках, коли це можливо. Зменшення щоденних поїздок

працівників істотно скорочує кількість викидів, пов'язаних із транспортом, а також знижує навантаження на міську інфраструктуру. Досвід пандемічних років показав, що дистанційна діяльність для програмістів є цілком ефективною, а її екологічний ефект - значним.

Важливу роль відіграє також вибір офісного простору. Компанії, що прагнуть підвищити власну екологічність, дедалі частіше віддають перевагу будівлям із хорошою енергоізоляцією, сучасними вентиляційними системами та можливістю використання природного освітлення. Застосування екологічних матеріалів у меблях та ремонті, а також раціональне планування робочих зон дозволяють створити здоровіші умови для працівників і одночасно зменшити вплив на довкілля.

Додатково вагомим напрямом є свідоме скорочення споживання ресурсів у повсякденній роботі. Це може включати мінімізацію використання паперу, перехід на цифрові документи, свідомий вибір товарів із меншим вмістом пластику або багаторазових матеріалів. Формування культури екологічного мислення сприяє тому, що відповідальність за природу стає не лише обов'язком, а й нормою корпоративної поведінки.

Корисним доповненням до вищезазначених заходів є залучення працівників до різноманітних екоініціатив. Це можуть бути тематичні вебінари, участь у зелених проєктах, підтримка організацій, що займаються збереженням природи, або проведення внутрішніх акцій із сортування та збору відходів. Такі активності допомагають не лише зменшити екологічне навантаження, а й формують свідоме ставлення до екології всередині команди.

У комплексі ці кроки дозволяють програмістам і IT-підрозділам реально впливати на зменшення свого екологічного сліду. Раціональне використання ресурсів, підтримка відповідального споживання та перехід на стійкі технологічні рішення формують основу для екологічно безпечного робочого середовища та сприяють гармонійному поєднанню професійної діяльності з турботою про природу.

Висновок до розділу 6

У розділі було проаналізовано екологічні аспекти діяльності в ІТ-сфері та визначено, що робота програміста, хоча й не пов'язана з прямими виробничими процесами, має відчутний непрямий вплив на навколишнє середовище. До ключових проблем віднесено значне споживання електроенергії комп'ютерною технікою, утворення електронних відходів, навантаження на глобальні дата-центри, екологічні витрати виробництва обладнання та нераціональне використання ресурсів.

Разом із тим сучасні цифрові технології створюють і можливості для зниження негативного впливу: впровадження енергоощадних пристроїв, перехід на дистанційну роботу, продовження строку служби техніки через модернізацію, цифровізація процесів та участь фахівців у розробці екологічно орієнтованих цифрових сервісів.

У роботі також визначено джерела забруднення, пов'язані з ІТ-інфраструктурою: електроспоживанням, виробництвом техніки, утилізацією електронних відходів та транспортними переміщеннями працівників. Акцентовано увагу на тому, що масштаб цифрових процесів вимагає свідомого підходу до використання технологій.

Запропоновані заходи щодо зменшення екологічного навантаження включають оптимізацію енергоспоживання, правильну утилізацію техніки, перехід на віддалений режим роботи, екологічні офісні рішення, зменшення використання матеріальних ресурсів та участь у корпоративних екоініціативах.

Загалом розділ доводить, що програміст або ІТ-підрозділ можуть зробити вагомий внесок у збереження довкілля шляхом відповідального ставлення до техніки, ресурсів і цифрової інфраструктури. Впровадження екологічно орієнтованих практик сприяє зменшенню екологічного сліду цифрової діяльності та формує культуру сталого розвитку в ІТ-середовищі.

ВИСНОВКИ

У рамках виконання магістерської роботи було здійснено дослідження процесів командної взаємодії в ресторані та проведено аналіз сучасних інформаційних систем, що застосовуються для командної взаємодії. Дослідження дозволило визначити основні проблеми, пов'язані з комунікацією, управлінням персоналом, координацією замовлень і контролем ресурсів, а також виявити обмеження наявних рішень. На основі узагальнення цих даних сформовано концепцію власної інформаційної системи, орієнтованої на підвищення ефективності внутрішніх процесів.

Після проведеного аналізу було розроблено веб-застосунок для управління командною взаємодією в ресторані, який спрямований на підвищення ефективності внутрішніх процесів та оптимізацію роботи персоналу. На основі аналізу сучасних інформаційних систем ресторанного бізнесу було визначено їх переваги та недоліки, що дозволило сформувати власну концепцію системи та обґрунтувати необхідність її впровадження.

Розробка здійснювалася на основі стеку технологій Spring Boot, Spring MVC, Spring Security, Hibernate, PostgreSQL, а також використано WebSocket для push-сповіщень та Apache POI (poi-ooxml) для формування Excel-звітів.

У проєкті реалізовано ключові функціональні модулі: управління замовленнями, персоналом, робочими графіками, меню, інгредієнтами, звітністю, відгуками клієнтів і push-сповіщеннями. Така структура забезпечує прозору взаємодію між офіціантами, кухарями, шеф-кухарем, барменом, бухгалтером та адміністратором, формуючи єдиний інформаційний простір.

Проведено економічну оцінку ефективності впровадження системи, яка підтвердила доцільність її використання завдяки скороченню помилок персоналу, оптимізації запасів, прискоренню комунікацій та зменшенню операційних витрат. Окремо проаналізовано питання охорони праці та екологічної безпеки, що дозволило сформувати рекомендації щодо раціонального використання обладнання та зменшення негативного впливу ІТ-діяльності на довкілля.

Загалом розроблений веб-застосунок демонструє практичну цінність у сфері ресторанного бізнесу, забезпечуючи цифровізацію командної взаємодії, підвищення продуктивності персоналу та створення сучасної інфраструктури управління процесами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Журавель, О. В., Халахур, Ю. Л., Кремена, Р. М. (2025). Роль управлінських інформаційних систем у підтримці прийняття стратегічних рішень. Актуальні питання економічних наук. Економічні науки, (8). <https://doi.org/10.5281/zenodo.14955655>.
2. Типи інформаційних систем: MIS, TPS, DSS, Pyramid Diagram [Електронний ресурс] – Режим доступу до ресурсу: <https://www.guru99.com/uk/mis-types-information-system.html>.
3. Долгова Л., Ямненко Г. Використання інструментарію інформаційних систем для автоматизації бізнес-процесів підприємства. Економічний аналіз, 31, № 2. 2021р. 90-96 с.
4. Сакун, Л., Ведініна, Ю., Шишлова, Ю. (2023). Вплив сучасних інформаційних технологій та командної роботи на систему менеджменту організацій в умовах глобалізаційних викликів. Modeling the Development of Economic Systems, 85–92. <https://doi.org/10.31891/mdes/2023-7-12>.
5. Business Process Management, BPM [Електронний ресурс] / Махым Zosym – Режим доступу до ресурсу: <https://www.it.ua/knowledge-base/technology-innovation/business-process-management-bpm>.
6. Camunda BPM: основні можливості платформи та її переваги для розробників і бізнесу [Електронний ресурс] – Режим доступу до ресурсу: <https://scriptum.ua/camunda-bpm-osnovni-mozhlyvosti-platformy-ta-yiyi-perevagy-dlya-rozrobnykiv-i-biznesu/>.
7. Автоматизація бізнесу з інструментом UiPath [Електронний ресурс] – Режим доступу до ресурсу: <https://evergreens.com.ua/ua/articles/uipath.html>.
8. Кращі інструменти для Agile управління проєктами у 2022 році [Електронний ресурс] – Режим доступу до ресурсу: <https://worksection.com/ua/blog/best-tools-for-agile-project-management.html>.
9. Dialogflow – Платформа Conversational AI для віртуальних агентів і чат-ботів [Електронний ресурс] – Режим доступу до ресурсу: <https://hellip.com/ua/product/cloud.html>.

10. Slack проти Microsoft Teams: який інструмент є найкращим? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.chanty.com/blog/uk/slack-vs-microsoft-teams-uk/>.

11. Найкращі аналоги для заміни Notion у 2025 році [Електронний ресурс] – Режим доступу до ресурсу: <https://worksection.com/ua/blog/notion-alternatives.html>.

12. Tableau vs Power BI: Який інструмент бізнес-аналітики кращий у 2025 році? [Електронний ресурс] – Режим доступу до ресурсу: <https://data.softico.ua/tableau-vs-power-bi-povne-porivniannia-u-2025-rotsi/>.

13. Порівняння, переваги та недоліки систем автоматизації бізнесу iiko і poster [Електронний ресурс] – Режим доступу до ресурсу: <https://konotop.city/articles/213176/porivnyannya-perevagi-ta-nedoliki-sistem-avtomatizacii-biznesu-iiko-i-poster>.

14. Павленко А.Ю., Козлов О.Ю. Дослідження систем використання веб-додатків управління командною взаємодією в організаціях. Миколаїв, Україна. 2025.

15. Як будувати UML-діаграми. Розбираємо три найпопулярніші варіанти [Електронний ресурс] – Режим доступу до ресурсу: <https://dou.ua/forums/topic/40575/>.

16. Моделювання даних (Data Modelling) [Електронний ресурс] – Режим доступу до ресурсу: <https://www.maxzosim.com/data-modelling/>.

17. Кривенко, І. П. (2021). Використання Java Spring Framework для створення корпоративних веб-застосунків. *Системи управління, навігації та зв'язку*, (4) 64, 125-130. <https://doi.org/10.26906/SUNZ.2021.4.125>.

18. Spring Security [Електронний ресурс] – Режим доступу до ресурсу: <https://spring.io/projects/spring-security>.

19. Повне розуміння діаграми компонентів UML за допомогою легкого методу [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mindonmap.com/uk/blog/uml-component-diagram/>.

20. Діаграма розгортання - Deployment Diagram [Електронний ресурс] – Режим доступу до ресурсу: <https://evergreens.com.ua/ua/articles/uml-diagrams.html> -

:~:text=Діаграма розгортання - Deployment Diagram,маршрути передачі даних між вузлами.

ДОДАТОК А – Технічне завдання

1. Загальні відомості

1.1. Найменування системи

Інформаційна система "Управління командною взаємодією в ресторані"

1.2. Номер договору

Договір №123456 від 21 квітня 2025 року на розробку веб-додатку для управління командною взаємодією в ресторані.

1.3. Найменування Розробника та Замовника

Розробник: Олександр Козлов

Замовник: Національний університет кораблебудування імені адмірала Макарова

1.4. Підстава для проведення робіт

Виконання кваліфікаційної магістерської роботи за спеціальністю 126 “Інформаційні системи і технології”.

1.5. Строки початку та закінчення робіт

- Початок робіт: 01.09.2025

- Закінчення робіт: 31.12.2025

1.6. Джерела та порядок фінансування робіт

Державне замовлення. Проєкт реалізується як навчальний.

1.7. Порядок оформлення та пред'явлення результатів робіт

Оформлення відповідно до вимог кваліфікаційної роботи. Фінальний результат включає функціонуючий веб-додаток, документацію та демонстрацію можливостей.

2. Призначення та цілі створення системи

2.1. Призначення системи

Розроблена система призначена для автоматизації внутрішніх процесів ресторану та забезпечення ефективної командної взаємодії між персоналом. Вона підтримує управління замовленнями, комунікацію між кухнею та залом, ведення меню, контроль робочих змін, облік виконаних завдань, формування аналітики та централізовану роботу адміністратора, кухарів, офіціантів і бухгалтера.

Система також дозволяє швидко обмінюватися повідомленнями завдяки push-сповіщенням, узгоджувати дії персоналу й оптимізувати ключові бізнес-процеси закладу.

2.2. Цілі створення системи

Основною метою створення системи є формування сучасного, зручного та надійного інструменту для управління щоденною роботою ресторану, який забезпечує:

- підвищення швидкості обробки замовлень;
- оптимізацію взаємодії між кухнею та залом;
- зменшення кількості помилок персоналу;
- прозорий облік процесів, аналітику та контроль;
- ефективний розподіл робочих змін і ресурсів;
- зниження операційних витрат і підвищення продуктивності команди.

3. Характеристика об'єкта автоматизації

3.1. Об'єкт автоматизації

Автоматизації підлягає комплекс процесів роботи ресторану, що включає взаємодію адміністратора, кухарів, офіціантів, барменів і бухгалтера. До об'єкта автоматизації входять:

- створення та управління замовленнями;
- комунікація між залом і кухнею;
- управління меню та категоріями страв;
- формування графіків і контроль робочих змін;
- облік виконаних завдань персоналу;
- перегляд статистики та фінансової інформації;
- push-повідомлення та система швидких сповіщень.

4. Вимоги до системи

4.1. Загальні вимоги

Система повинна коректно працювати на ПК, планшетах і мобільних пристроях персоналу. Інтерфейс має бути інтуїтивним, швидким, оптимізованим під навантаження та підтримувати різні ролі доступу.

Особливу увагу приділено безпеці, стабільності роботи та можливості масштабування.

4.1.1 Вимоги до структури системи

Бекенд(Java Spring Boot (Spring MVC, Spring Security, WebSocket, Hibernate), фронтенд (HTML, CSS, JavaScript), база даних(PostgreSQL), розгортання(локально або DigitalOcean).

4.1.2 Вимоги до режимів функціонування системи

- створення та редагування облікових записів персоналу;
- авторизація різних ролей (адміністратор, офіціант, кухар, бухгалтер);
- формування, зміна та виконання замовлень;
- push-повідомлення між ролями (готовність страви, зміна статусу замовлення тощо);
- ведення меню та категорій страв;
- розподіл робочих змін;
- облік виконаних робіт персоналу;
- управління столиками;
- формування статистики та Excel-звіту;
- система внутрішніх сповіщень і журнал дій.

4.1.3. Вимоги до сумісності

Система повинна бути сумісною з основними браузером та мобільними платформами.

4.1.4 Перспективи розвитку

Можливість масштабування, додавання нових функцій (наприклад, інтеграція з платіжними системами, інтеграція з інформаційними системами для бухгалтерського обліку).

4.1.5. Вимоги щодо безпеки

Аутентифікація з токенами, базова авторизація, Spring Security з розмежуванням доступу

4.1.6 Вимоги до надійності

Система повинна мати механізми автоматичного резервного копіювання бази даних, стабільну роботу WebSocket-каналу.

4.1.7 Вимоги щодо збереження інформації

Розмежування доступу до даних між ролями за допомогою Spring Security. Підтримка автоматичного резервного копіювання щодня, зберігання копій протягом щонайменше 30 днів. Резервні копії зберігаються в окремому географічному регіоні.

4.2. Вимоги до видів забезпечення

4.2.1. Загальні відомості

Система повинна мати такі види забезпечення: програмне, технічне, інформаційне, організаційне.

4.2.2. Вимоги до лінгвістичного забезпечення

Основна мова інтерфейсу - українська. Усі елементи інтерфейсу мають бути перекладені відповідно до обраної мови.

4.2.3. Вимоги до технічного забезпечення

Для стабільної роботи системи необхідно передбачити серверне обладнання, здатне підтримувати одночасну роботу персоналу кухні, залу та адміністрації. Система повинна функціонувати на хмарній або локальній інфраструктурі з такими мінімальними характеристиками:

- процесор із не менше ніж 6–8 віртуальними ядрами;
- оперативна пам'ять від 12–16 GB для забезпечення безперервної роботи WebSocket та обробки замовлень;
- SSD-накопичувач обсягом 150–250 GB для зберігання історії замовлень, звітів та службових файлів;
- резервний вузол або репліка БД для швидкого відновлення у разі збою;
- підтримка контейнеризації (Docker) або хмарного масштабування (AWS, GCP, Azure).

4.2.4. Вимоги до програмного забезпечення

Серверна частина - Java (Spring Boot), клієнтська - HTML. База даних - PostgreSQL. Сервери розміщуються на DigitalOcean, файли зберігаються на DigitalOcean S3.

4.2.5. Вимоги до інформаційного забезпечення

Інформаційне забезпечення має включати структуровану систему даних, що охоплює всі бізнес-процеси ресторану.

У базі даних повинні зберігатися:

- довідник страв та категорій меню;
- історія замовлень і статуси їх виконання;
- дані про персонал, ролі та права доступу;
- робочі зміни та планування графіків;
- службові події (push-повідомлення, логи дій користувачів);
- фінансові показники, статистика завантаженості залу та кухні;
- архів згенерованих Excel-файлів.

Також необхідно забезпечити регулярне оновлення довідників, коректну індексацію та можливість швидкого отримання аналітичної інформації.

4.2.6. Вимоги до організаційного забезпечення

Організаційне забезпечення визначає правила та процедури роботи з системою:

- призначення відповідальних осіб за адміністрування системи та управління користувачами;
- створення інструкцій для персоналу щодо використання модулів замовлень, меню та звітності;
- забезпечення наявності служби підтримки або відповідального працівника для вирішення технічних питань;
- регламентація прав доступу для кожної ролі (офіціант, кухар, адміністратор, бухгалтер);
- проведення періодичного навчання персоналу;
- контроль за виконанням правил резервного копіювання та оновлення

ПЗ.

1. Склад та зміст робіт зі створення системи

Таблиця В.1 – Зміст роботи зі створення системи

Стадія роботи	Виконання роботи	Термін	Підсумки виконання роботи
Формування вимог	Аналіз існуючих систем, визначення проблем взаємодії персоналу, збір вимог, формування ТЗ	1.5 місяці	Звіт з аналізу, вимоги до функціональності Затверджене технічне завдання
Проектування архітектури	Розробка архітектури, ER-діаграми БД, прототипів інтерфейсу, діаграм компонентів і розгортання .Розробка прототипу інтерфейсу користувача	1 місяць з початку робіт	Технічне рішення, опис архітектури Схеми БД, документація UX/UI-макети, опис користувацьких сценаріїв
Реалізація	Створення бекенду (Spring Boot), фронтенду (React), реалізація модулів: замовлення, зміни персоналу, сповіщення, аналітика, складський модуль	2-4 місяці	Програмний модуль бекенду, документація API Готовий веб-інтерфейс, підключений до серверної частини
Тестування	Проведення модульного та системного тестування, виправлення помилок	1 місяць	Звіт з тестування, акти приймання-передачі
Впровадження	Розгортання системи на сервері (DigitalOcean), Підготовка документації (інструкція користувача, супровідна документація, опис API)	1 місяць	Готовий до використання веб-додаток Повний комплект технічної документації

6. Порядок контролю та приймання системи

Система вважається прийнятою після демонстрації роботи всіх базових функцій згідно з ТЗ.

7. Вимоги до документування

Передбачено:

- Технічна документація
- Інструкції користувача
- База знань (FAQ)
- API-документація

ДОДАТОК Б – Інструкція користувача

На головному екрані (див. рисунок А.1) користувач із роллю адміністратора може контролювати всі активні замовлення ресторана. Робоча область поділена на два статуси: «Готується» - для замовлень, що перебувають у процесі приготування, та «Готово до видачі» - для тих, які вже очікують передачі клієнту. Якщо замовлень немає, система відображає відповідні повідомлення.

У верхній частині розташовано головну навігацію адміністратора: замовлення, меню, столики, управління персоналом, зарплата, відгуки клієнтів, графік роботи, дашборд, інвентар, аналітика. У правому нижньому куті доступна кнопка «Нове замовлення», що дозволяє адміністратору вручну створити нове замовлення та додати його в робочий процес.

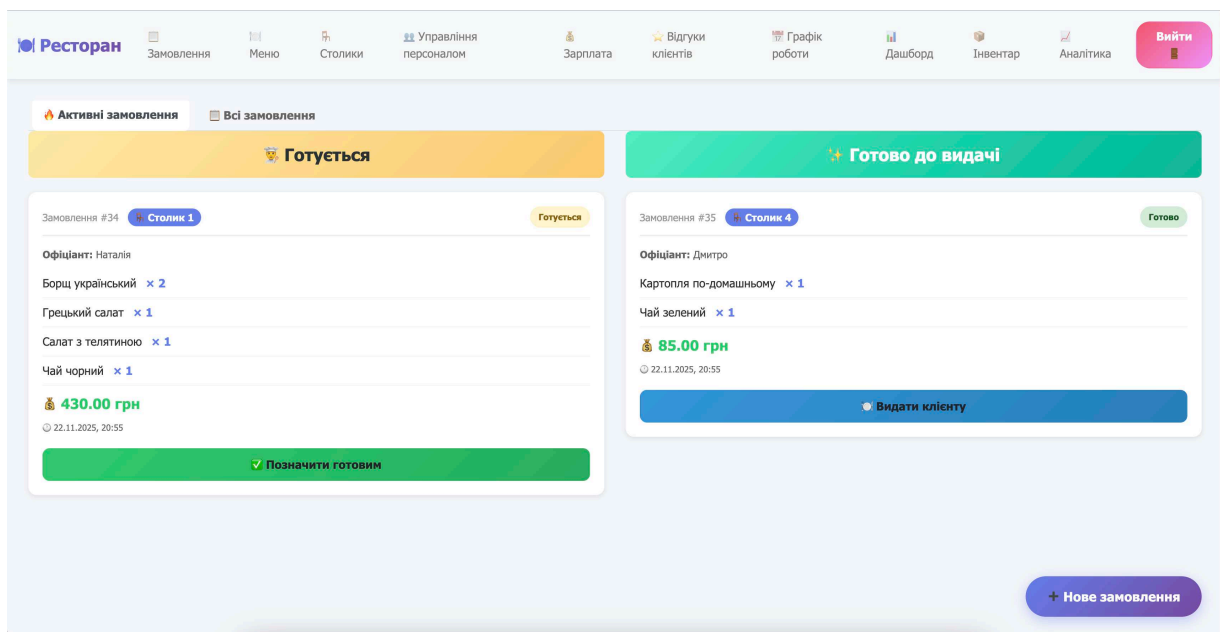


Рисунок А.1 – Головний екран користувача з роллю адміністратора

На головному екрані (див. рисунок А.2) користувач без ролі адміністратора може переглядати активні замовлення у двох статусах - «Готується» та «Готово до видачі». У разі відсутності замовлень у відповідній категорії система показує повідомлення. Також доступна кнопка «Нове замовлення», що дозволяє швидко створити новий запит без доступу до інших адміністративних функцій.

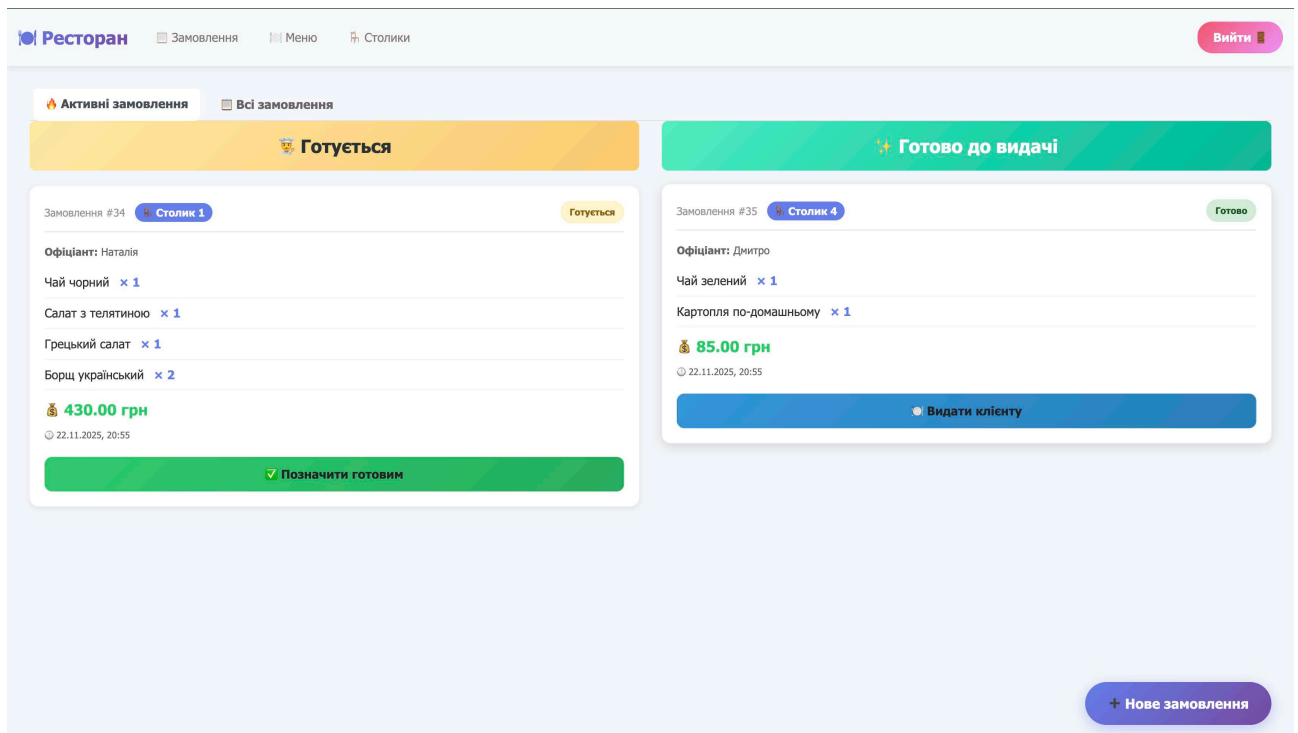


Рисунок А.2 – Головний екран користувача без ролі адміністратора

На сторінці «Графік роботи» (див. рисунок А.3) користувач може переглядати сформований розклад змін персоналу за вибраний тиждень. Інтерфейс дозволяє обрати діапазон дат, згенерувати оновлений графік, переглянути розподіл працівників по змінах, а також експортувати розклад у формат Excel. За потреби доступні дії для переходу на попередній або наступний тиждень. Сторінка відображає списки співробітників у кожній зміні та надає можливість адмініструвати робочий графік у зручному табличному форматі.

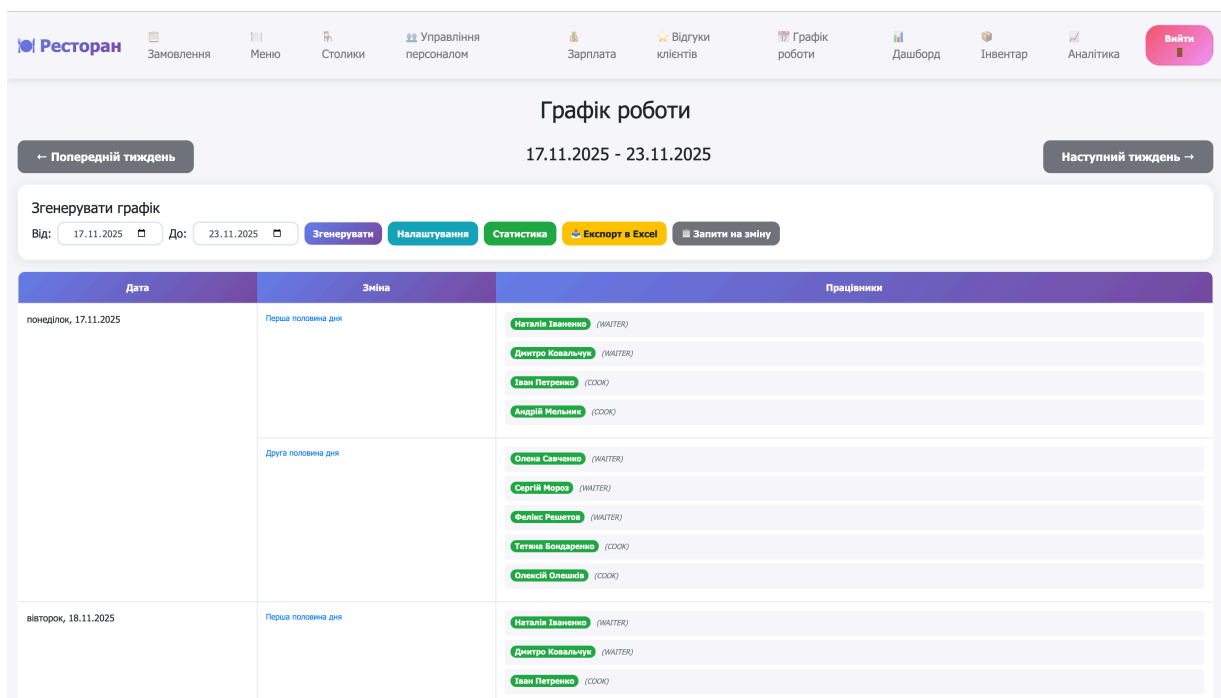


Рисунок А.3 – Сторінка «Графік роботи»

Сторінка «Дашборд менеджера» (див. рисунок А.4) надає керівнику закладу швидкий доступ до ключових показників роботи ресторану за поточний день і тиждень. У верхній частині відображаються основні метрики: сума виручки, кількість замовлень, рейтинг закладу та кількість працівників, що перебувають на зміні. Нижче розміщено графіки виручки за тиждень і кількості замовлень за годинами, що дозволяє аналізувати пікові навантаження та планувати роботу персоналу. Також доступні блоки «Хто зараз працює» та «Працівник тижня», які допомагають менеджеру контролювати активність персоналу та оцінювати ефективність роботи співробітників.

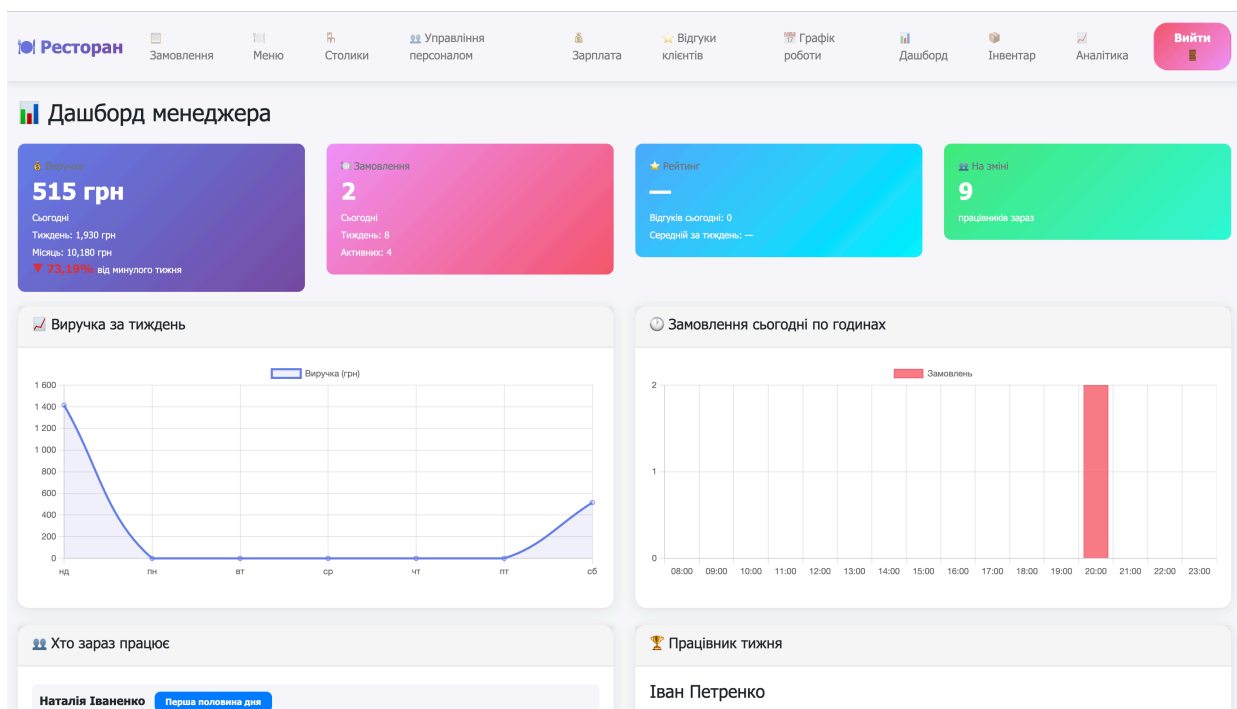


Рисунок А.4 – Сторінка «Дашборд менеджера»

Кухарі автоматично отримують таке push-сповіщення (див. рисунок А.5) одразу після того, як офіціант створює нове замовлення. Повідомлення містить його номер і позначку, що воно призначене для кухні, що дає змогу швидко розпочати приготування.

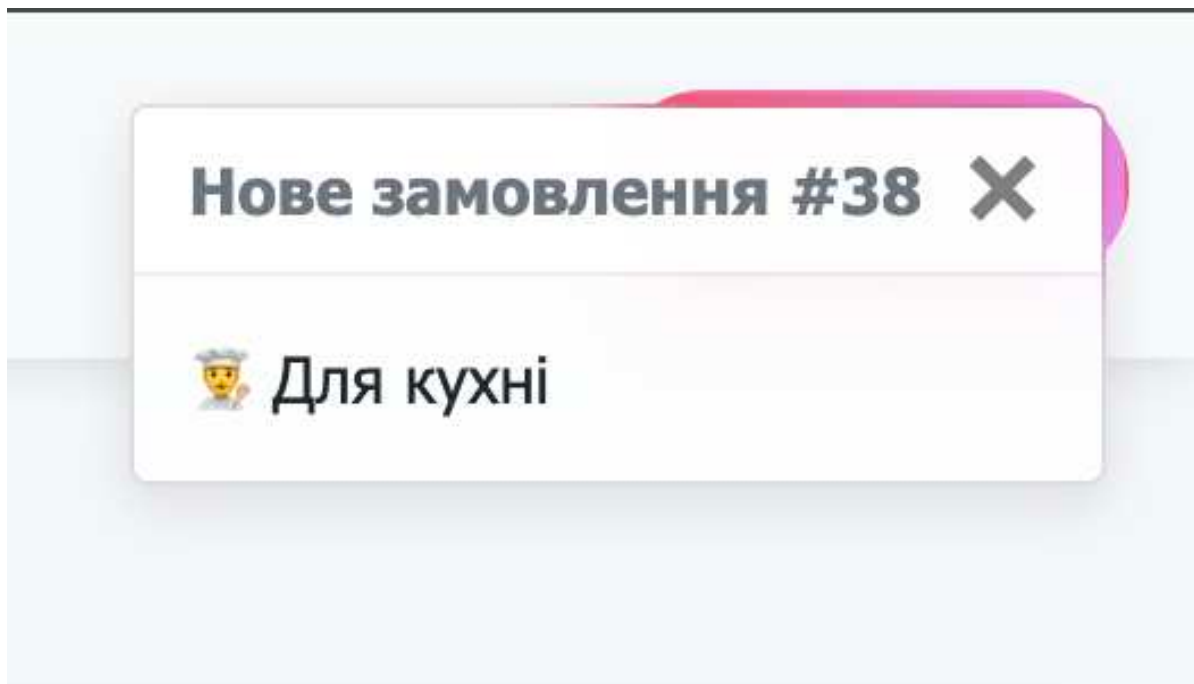


Рисунок А.5 – Push-сповіщення, для кухарів про нове замовлення

Офіціанти отримують push-сповіщення про (див. рисунок А.6) готовність замовлення.

Коли кухар позначає страву як готову, офіціантам автоматично надходить сповіщення з номером замовлення та вказівкою «Віднесіть клієнту». Це дозволяє швидко реагувати на готові страви та забезпечує оперативне обслуговування гостей.

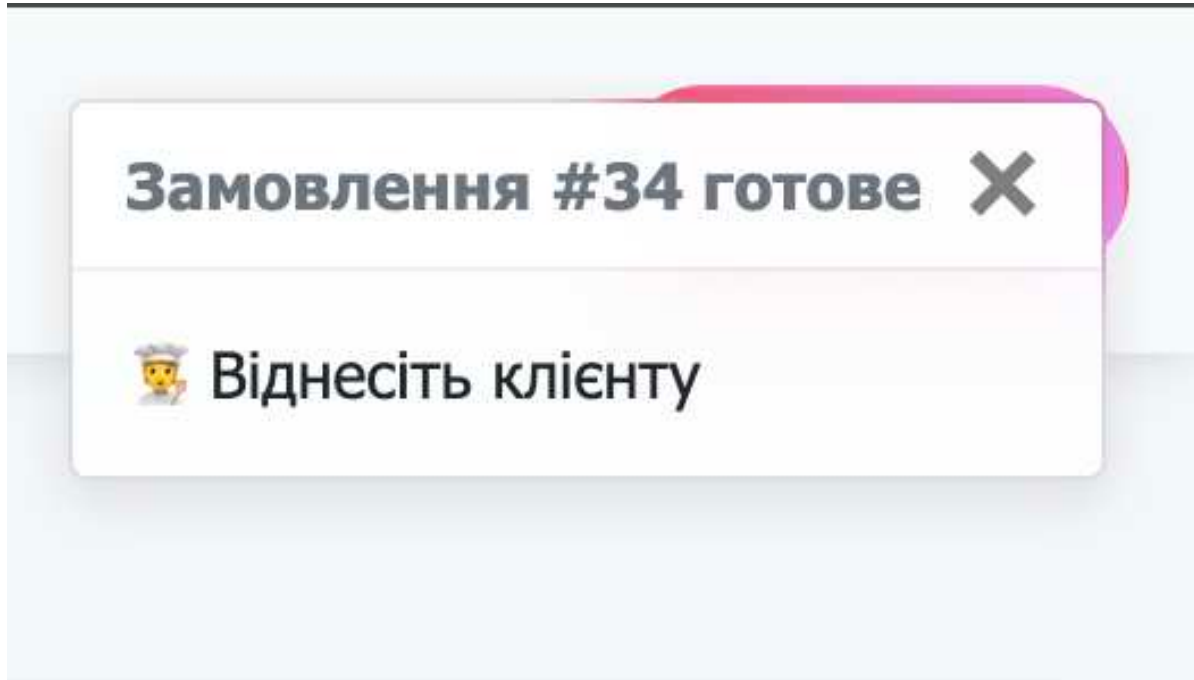


Рисунок А.6 – Push-сповіщення, для офіціантів про нове замовлення

Сторінка створення замовлення (див. рисунок А.7) дозволяє офіціанту швидко сформувати нове замовлення, обравши потрібний столик, відповідального офіціанта та початковий статус. У нижній частині відображається меню страв, де можна додавати позиції, змінювати їх кількість або видаляти. Кожна обрана страва автоматично відображається у правому блоці «Підсумок замовлення», де формується загальна сума.

Після того як усі позиції додано, офіціант може переглянути фінальний перелік страв, перевірити підрахунок вартості та натиснути кнопку «Підтвердити замовлення» для його оформлення. Це забезпечує швидкий, інтуїтивний і зручний процес створення замовлень у ресторані.

Ресторан

Замовлення

Меню

Столики

Управління персоналом

Зарплата

Відгуки клієнтів

Графік роботи

Дашборд

Інвентар

Аналітика

Вийти

Створити нове замовлення

Оберіть столики, страви та підтвердіть замовлення

Основна інформація

Номер столика

Оберіть столики...

Офіціант

Оберіть офіціанта...

Статус

В обробці

Пошук страв...

Меню страв

Борщ український

Традиційний червоний борщ з м'ясом та сметаною

85.0 грн

- 1 +

Солянка

Насичений м'ясний суп з оливками та лимоним

95.0 грн

- 2 +

Курячий бульйон

Легкий бульйон з домашньою лошатиною

70.0 грн

Додати

Цезар з куркою

Класичний салат з куркою, сиром пармезан

120.0 грн

Додати

Грецький салат

Свіжі овочі з сиром фета та оливками

95.0 грн

Додати

Салат з телятиною

Теплий салат з телятиною та овочами гриль

135.0 грн

Додати

Підсумок замовлення

Борщ український

1 x 85 грн

85.00 грн

Солянка

2 x 95 грн

190.00 грн

Всього:

275.00 грн

Підтвердити замовлення

Рисунок А.7 – Сторінка створення замовлення

На сторінці запитів на зміну графіку (див. рисунок А.8) адміністратор або менеджер бачить перелік усіх звернень працівників щодо коригування їх робочих змін. Для кожного запиту відображається дата, обрана зміна, причина та час подання, що дозволяє швидко оцінити ситуацію. Керівник може одразу схвалити або відхилити запит, використовуючи відповідні кнопки, забезпечуючи зручне та оперативне управління графіком персоналу.

Ресторан

Замовлення

Меню

Столики

Управління персоналом

Зарплата

Відгуки клієнтів

Графік роботи

Дашборд

Інвентар

Аналітика

Вийти

Запити на зміну графіку

Назад до графіку

Наталія Іваненко

Дата: 17.11.2025 (понеділок)

Зміна: Перша половина дня

Причина: Сімейні обставини

Подано: 22.11.2025 21:20

✓ Схвалити

✗ Відхилити

Сергій Мороз

Дата: 21.11.2025 (п'ятниця)

Зміна: Друга половина дня

Причина: Хворію

Подано: 22.11.2025 21:20

✓ Схвалити

✗ Відхилити

Рисунок А.8 – Сторінка запитів на зміну графіку

На сторінці «Налаштування розрахунку зарплати» (див. рисунок А.9) адміністратор може керувати всіма параметрами формули оплати праці для різних категорій персоналу. У верхній частині подано пояснення, як саме система обчислює зарплату, включаючи базову ставку, бонуси за замовлення, премії за відгуки та штрафи за прогули. Нижче розміщені окремі блоки налаштувань для офіціантів і кухарів, де можна змінити погодинну ставку, бонуси, коефіцієнти премій та розмір штрафів. Після внесення змін адміністратор зберігає параметри, і система автоматично використовує оновлені значення під час розрахунку зарплати.

Рисунок А.9 – Сторінка «Налаштування розрахунку зарплати»

На сторінці «Управління інвентарем» (див. рисунок А.10) менеджер ресторану може контролювати залишки всіх інгредієнтів, відстежувати їх поточну кількість та порівнювати її з установленим мінімумом. Система автоматично підсвічує позиції з низьким запасом та виводить попередження у верхній частині сторінки. Для кожного інгредієнта доступна дія «Редагувати», що дозволяє змінити залишок або коригувати мінімальні значення. Це забезпечує своєчасне поповнення продуктів та ефективний контроль інвентарю ресторану.

Ресторан

Замовлення

Меню

Столики

Управління персоналом

Зарплата

Відгуки клієнтів

Графік роботи

Дашборд

Інвентар

Аналітика

Вийти

Управління інвентарем

⚠️ Увага! 2 продуктів з низьким запасом!

Інгредієнт	Поточна кількість	Одиниця	Мінімум	Статус	Дії
Рис	18.0	кг	8.0	ОК	Редагувати
Сік	8.0	л	10.0	Низький запас!	Редагувати
Чай	3.0	кг	0.5	ОК	Редагувати
Сир твердий	2.5	кг	4.0	Низький запас!	Редагувати
Кава	5.0	кг	1.0	ОК	Редагувати
Сіль	10.0	кг	2.0	ОК	Редагувати
Хліб	100.0	шт	20.0	ОК	Редагувати
Яйця	497.0	шт	100.0	ОК	Редагувати
Олія соняшникова	20.0	л	5.0	ОК	Редагувати
Буряк	20.0	кг	5.0	ОК	Редагувати

Рисунок А.10 – Сторінка «Управління інвентарем»

ДОДАТОК В – Фрагмент програмного коду

Повний код: https://github.com/kozlov2777/Diplom_Spring

SecurityConfig.java

```
package com.example.demo;

import com.example.demo.services.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configurers.AbstractHttpConfigurer;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;

import static org.springframework.security.config.Customizer.withDefaults;

@Configuration
public class SecurityConfig {
    @Autowired
    private EmployeeService employeeService;
    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration authenticationConfiguration) throws Exception {
        return authenticationConfiguration.getAuthenticationManager();
    }

    @Bean
    public BCryptPasswordEncoder bCryptPasswordEncoder() {
        return new BCryptPasswordEncoder();
    }

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        AuthenticationManagerBuilder authenticationManagerBuilder =
            http.getSharedObject(AuthenticationManagerBuilder.class);

        authenticationManagerBuilder.userDetailsService(employeeService).passwordEncoder(bCryptPasswordEncoder());
        AuthenticationManager build = authenticationManagerBuilder.build();

        SecurityFilterChain securityFilterChain = http
            .authorizeHttpRequests(authorize -> authorize
                .requestMatchers("/login").permitAll()
                .requestMatchers("/feedback/**").permitAll()
                .requestMatchers("/register").hasAuthority("ADMIN")
                .requestMatchers("/salary").hasAuthority("ADMIN")
                .anyRequest().authenticated()
            ).csrf(AbstractHttpConfigurer::disable)
            .logout(logout -> logout
                .logoutUrl("/logout")
            );
    }
}
```

```

        .logoutSuccessUrl("/login")
        .invalidateHttpSession(true)
        .deleteCookies("JSESSIONID")
    )
    .formLogin(form -> form
        .loginPage("/login")
        .defaultSuccessUrl("/", true)
        .failureUrl("/login?error=true")
        .permitAll()
    )
    .authenticationManager(build)
    .exceptionHandling(exceptionHandling -> exceptionHandling
        .accessDeniedPage("/access-denied")
    )
    .build();

    return securityFilterChain;
}
}

```

WebSocketConfig.java

```

package com.example.demo.config;

import org.springframework.context.annotation.Configuration;
import org.springframework.messaging.simp.config.MessageBrokerRegistry;
import org.springframework.web.socket.config.annotation.EnableWebSocketMessageBroker;
import org.springframework.web.socket.config.annotation.StompEndpointRegistry;
import org.springframework.web.socket.config.annotation.WebSocketMessageBrokerConfigurer;

@Configuration
@EnableWebSocketMessageBroker
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    @Override
    public void configureMessageBroker(MessageBrokerRegistry config) {
        config.enableSimpleBroker("/topic");
        config.setApplicationDestinationPrefixes("/app");
    }

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws-orders").withSockJS();
    }
}

```

ExcelExportService.java

```

package com.example.demo.services;

import com.example.demo.dto.ScheduleDto;
import com.example.demo.dto.SalaryDetailDto;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.springframework.stereotype.Service;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.List;

@Service
public class ExcelExportService {

```

```

    public byte[] exportScheduleToExcel(List<ScheduleDto> scheduleList, LocalDate
startDate, LocalDate endDate) throws IOException {
        Workbook workbook = new XSSFWorkbook();
        Sheet sheet = workbook.createSheet("Графік роботи");

        // Стили
        CellStyle headerStyle = workbook.createCellStyle();
        Font headerFont = workbook.createFont();
        headerFont.setBold(true);
        headerFont.setFontHeightInPoints((short) 12);
        headerStyle.setFont(headerFont);

        headerStyle.setFillForegroundColor(IndexedColors.GREY_25_PERCENT.getIndex());
        headerStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
        headerStyle.setBorderBottom(BorderStyle.THIN);
        headerStyle.setBorderTop(BorderStyle.THIN);
        headerStyle.setBorderLeft(BorderStyle.THIN);
        headerStyle.setBorderRight(BorderStyle.THIN);

        CellStyle dataStyle = workbook.createCellStyle();
        dataStyle.setBorderBottom(BorderStyle.THIN);
        dataStyle.setBorderTop(BorderStyle.THIN);
        dataStyle.setBorderLeft(BorderStyle.THIN);
        dataStyle.setBorderRight(BorderStyle.THIN);

        // Заголовок
        Row titleRow = sheet.createRow(0);
        Cell titleCell = titleRow.createCell(0);
        titleCell.setCellValue("Графік роботи з " +
startDate.format(DateTimeFormatter.ofPattern("dd.MM.yyyy")) +
            " по " +
endDate.format(DateTimeFormatter.ofPattern("dd.MM.yyyy")));

        // Заголовки колонок
        Row headerRow = sheet.createRow(2);
        String[] headers = {"Дата", "Зміна", "Прізвище", "Ім'я", "Статус"};
        for (int i = 0; i < headers.length; i++) {
            Cell cell = headerRow.createCell(i);
            cell.setCellValue(headers[i]);
            cell.setCellStyle(headerStyle);
        }

        // Дані
        int rowNum = 3;
        DateTimeFormatter dateFormatter = DateTimeFormatter.ofPattern("dd.MM.yyyy
(EEEE)", new java.util.Locale("uk"));

        for (ScheduleDto schedule : scheduleList) {
            Row row = sheet.createRow(rowNum++);

            Cell cell10 = row.createCell(0);
            cell10.setCellValue(schedule.getWorkDate().format(dateFormatter));
            cell10.setCellStyle(dataStyle);

            Cell cell11 = row.createCell(1);
            cell11.setCellValue(schedule.getShiftName());
            cell11.setCellStyle(dataStyle);

            Cell cell12 = row.createCell(2);
            cell12.setCellValue(schedule.getLastName());
            cell12.setCellStyle(dataStyle);

            Cell cell13 = row.createCell(3);

```

```

        cell3.setCellValue(schedule.getFirstName());
        cell3.setCellStyle(dataStyle);

        Cell cell4 = row.createCell(4);
        cell4.setCellValue(schedule.getIsDayOff() ? "Вихідний" : "Робочий
день");
        cell4.setCellStyle(dataStyle);
    }

    // Автоматична ширина колонок
    for (int i = 0; i < headers.length; i++) {
        sheet.autoSizeColumn(i);
    }

    // Конвертуємо в байти
    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    workbook.write(outputStream);
    workbook.close();

    return outputStream.toByteArray();
}

public byte[] exportSalaryToExcel(List<SalaryDetailDto> salaries, LocalDate
startDate, LocalDate endDate) throws IOException {
    Workbook workbook = new XSSFWorkbook();
    Sheet sheet = workbook.createSheet("Зарплата");

    // Стили
    CellStyle headerStyle = workbook.createCellStyle();
    Font headerFont = workbook.createFont();
    headerFont.setBold(true);
    headerFont.setFontHeightInPoints((short) 12);
    headerStyle.setFont(headerFont);

    headerStyle.setFillForegroundColor(IndexedColors.GREY_25_PERCENT.getIndex());
    headerStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
    headerStyle.setBorderBottom(BorderStyle.THIN);
    headerStyle.setBorderTop(BorderStyle.THIN);
    headerStyle.setBorderLeft(BorderStyle.THIN);
    headerStyle.setBorderRight(BorderStyle.THIN);

    CellStyle dataStyle = workbook.createCellStyle();
    dataStyle.setBorderBottom(BorderStyle.THIN);
    dataStyle.setBorderTop(BorderStyle.THIN);
    dataStyle.setBorderLeft(BorderStyle.THIN);
    dataStyle.setBorderRight(BorderStyle.THIN);

    // Заголовок
    Row titleRow = sheet.createRow(0);
    Cell titleCell = titleRow.createCell(0);
    titleCell.setCellValue("Зарплата з " +
startDate.format(DateTimeFormatter.ofPattern("dd.MM.yyyy")) +
        " по " +
endDate.format(DateTimeFormatter.ofPattern("dd.MM.yyyy")));

    // Заголовки колонок
    Row headerRow = sheet.createRow(2);
    String[] headers = {
        "Прізвище", "Ім'я", "Роль",
        "Годин", "Ставка, грн/год", "Базова зарплата, грн",
        "Замовлень", "Бонус за замовлення, грн",
        "Відгуків", "Середній рейтинг", "Премія за відгуки, грн",
        "Прогули", "Штрафи, грн",
        "Всього, грн"
    }

```

```

};
for (int i = 0; i < headers.length; i++) {
    Cell cell = headerRow.createCell(i);
    cell.setCellValue(headers[i]);
    cell.setCellStyle(headerStyle);
}

// Дани
int rowNum = 3;
for (SalaryDetailDto salary : salaries) {
    Row row = sheet.createRow(rowNum++);

    int col = 0;

    Cell c0 = row.createCell(col++);
    c0.setCellValue(salary.getLastName());
    c0.setCellStyle(dataStyle);

    Cell c1 = row.createCell(col++);
    c1.setCellValue(salary.getFirstName());
    c1.setCellStyle(dataStyle);

    Cell c2 = row.createCell(col++);
    c2.setCellValue(salary.getRoleName());
    c2.setCellStyle(dataStyle);

    Cell c3 = row.createCell(col++);
    c3.setCellValue(salary.getHoursWorked() != null ?
salary.getHoursWorked() : 0);
    c3.setCellStyle(dataStyle);

    Cell c4 = row.createCell(col++);
    c4.setCellValue(salary.getHourlyRate() != null ?
salary.getHourlyRate() : 0);
    c4.setCellStyle(dataStyle);

    Cell c5 = row.createCell(col++);
    c5.setCellValue(salary.getBaseSalary() != null ?
salary.getBaseSalary() : 0);
    c5.setCellStyle(dataStyle);

    Cell c6 = row.createCell(col++);
    c6.setCellValue(salary.getOrdersCompleted() != null ?
salary.getOrdersCompleted() : 0);
    c6.setCellStyle(dataStyle);

    Cell c7 = row.createCell(col++);
    c7.setCellValue(salary.getOrderBonusTotal() != null ?
salary.getOrderBonusTotal() : 0);
    c7.setCellStyle(dataStyle);

    Cell c8 = row.createCell(col++);
    c8.setCellValue(salary.getReviewsCount() != null ?
salary.getReviewsCount() : 0);
    c8.setCellStyle(dataStyle);

    Cell c9 = row.createCell(col++);
    c9.setCellValue(salary.getAverageRating() != null ?
salary.getAverageRating() : 0);
    c9.setCellStyle(dataStyle);

    Cell c10 = row.createCell(col++);
    c10.setCellValue(salary.getReviewBonus() != null ?
salary.getReviewBonus() : 0);

```

```

        c10.setCellStyle(dataStyle);

        Cell c11 = row.createCell(col++);
        c11.setCellValue(salary.getAbsences() != null ? salary.getAbsences() :
0);
        c11.setCellStyle(dataStyle);

        Cell c12 = row.createCell(col++);
        c12.setCellValue(salary.getPenaltiesTotal() != null ?
salary.getPenaltiesTotal() : 0);
        c12.setCellStyle(dataStyle);

        Cell c13 = row.createCell(col++);
        c13.setCellValue(salary.getTotalSalary() != null ?
salary.getTotalSalary() : 0);
        c13.setCellStyle(dataStyle);
    }

    // Автоматична ширина колонок
    for (int i = 0; i < headers.length; i++) {
        sheet.autoSizeColumn(i);
    }

    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    workbook.write(outputStream);
    workbook.close();

    return outputStream.toByteArray();
}
}

```


ДОДАТОК Г – Тези

УДК 004.75

ДОСЛІДЖЕННЯ СИСТЕМ ВИКОРИСТАННЯ ВЕБ-ДОДАТКІВ УПРАВЛІННЯ КОМАНДНОЮ ВЗАЄМОДІЄЮ В ОРГАНІЗАЦІЯХ

Павленко А.Ю.¹; Козлов О.Ю.²

^{1,2}Національний університет кораблебудування імені адмірала Макарова

^{1,2}Україна, Миколаїв

¹pavlenko.a.yu@gmail.com; ²201133@nuos.edu.ua

Анотація. Розглянуто процес розробки інформаційної системи управління командною взаємодією в організаціях з урахуванням сучасних інформаційних технологій, що сприяють ефективній комунікації, координації та співпраці працівників. Проаналізовано особливості роботи в умовах гнучких форм зайнятості, віддаленої роботи та складної організаційної структури.

Ключові слова: інформаційна система; управління командною взаємодією; комунікація в організації; цифрова трансформація; адаптивність системи; безпека даних; інтеграція проектних інструментів.

Вступна частина. У сучасних організаціях, особливо тих, що працюють у сфері обслуговування, виникає низка викликів, пов'язаних із забезпеченням ефективної командної взаємодії. Серед основних проблем — відсутність єдиної платформи для координації дій між підрозділами, затримки в обміні інформацією, обмежена адаптивність систем до специфіки внутрішніх процесів, недостатній рівень безпеки даних та складність інтеграції з іншими цифровими інструментами. Ці чинники знижують продуктивність, ускладнюють управління персоналом і перешкоджають оперативному прийняттю рішень [5].

Актуальність впровадження веб-додатків для управління командною взаємодією зростає на тлі цифрової трансформації бізнесу, поширення гнучких форм зайнятості та потреби в синхронізації дій між віддаленими працівниками та фізичними підрозділами [1]. Важливо забезпечити прозорість управлінських процесів, гнучке налаштування системи під потреби конкретної організації, а також можливість масштабування та інтеграції з іншими сервісами без зупинки роботи [2].

Метою роботи є підвищення ефективності процесів управління командною взаємодією в організаціях. За рахунок аналізу існуючих систем, виявлення їхніх переваг та недоліків, дослідження потреб користувачів і розробку архітектури, яка покращить адаптивність, масштабованість та безпеку [4].

Основна частина. Сучасні інформаційні технології відіграють ключову роль у забезпеченні ефективної комунікації, координації та співпраці між працівниками, особливо в умовах гнучких форм зайнятості, віддаленої роботи та складної організаційної структури [4]. Інформаційні системи управління взаємодією стали необхідним інструментом у діяльності багатьох компаній, забезпечуючи прозорість, контроль та оперативність в управлінських процесах [5].

Такі системи дозволяють ефективно організувати робочі процеси, делегування завдань, моніторинг прогресу та обмін інформацією в рамках однієї платформи [2]. В Україні та за кордоном існує низка рішень для командної взаємодії, однак не всі вони враховують специфіку певних організацій або потреби у гнучкому налаштуванні під внутрішні процеси [4]. Зростаючі вимоги до ефективності командної роботи та цифрової трансформації бізнесу зумовлюють необхідність розробки нових або вдосконалення існуючих інформаційних систем [1].

Розроблена інформаційна система управління рестораном ґрунтується на поєднанні Java (Spring MVC) у серверній частині та HTML у клієнтському інтерфейсі [3]. Для зберігання та обробки даних застосовано MariaDB у зв'язці з ORM Hibernate, що забезпечує стабільність, масштабованість і зручність у подальшому розширенні функціоналу [3]. Архітектурна модель MVC (Model–View–Controller) дозволила чітко відокремити логіку, представлення та керування, а інтеграція Spring Security реалізувала багаторівневу систему доступу для різних ролей користувачів [1]. Система орієнтована на комплексне вирішення завдань ресторанного бізнесу, де ключовим є швидкий та безпомилковий обмін інформацією між залом і кухнею [3].

Система підтримує роботу кількох категорій користувачів: адміністратора, шеф-кухаря, офіціанта, бармена, кухаря та бухгалтера. Для кожної ролі визначено набір прав і функцій. Наприклад, адміністратор має повний контроль над меню, замовленнями, категоріями страв, зарплатами персоналу та статистикою; офіціанти працюють із замовленнями та переглядають статуси столиків; кухарі й бармени отримують замовлення у реальному часі та позначають етапи їх виконання; бухгалтер має доступ до фінансових даних і звітності.

Система складається з кількох ключових модулів:

- модуль управління замовленнями, який забезпечує створення, редагування та відстеження статусів замовлень від моменту прийому до завершення;
- модуль управління меню дозволяє адмініструвати позиції страв, категорії та ціни;
- модуль інвентаризації, що автоматично списує інгредієнти зі складу під час оформлення замовлення, контролює залишки та формує потреби у поповненні;
- модуль звітності генерує аналітичні та фінансові звіти, визначає популярні страви, години пікового навантаження та продуктивність персоналу.

Взаємодія між залом і кухнею відбувається у режимі реального часу: замовлення, створене офіціантом, миттєво відображається у кухарів і барменів, а оновлення статусів синхронізується для всіх зацікавлених користувачів. Це усуває затримки, знижує ризик помилок і сприяє більш рівномірному розподілу навантаження.

З технічної точки зору, система відповідає ряду нефункціональних вимог:

- продуктивність: оновлення даних (наприклад, статусу замовлення) відбувається протягом 2-3 секунд;
- безпека: дані шифруються, доступ контролюється за ролями;
- масштабованість: можливість підключення нових модулів, наприклад, онлайн-бронювання, без зупинки роботи;
- надійність: резервне копіювання та механізми відновлення після збоїв.

В результаті, система не лише автоматизує рутинні завдання, але й створює цілісну інформаційну екосистему ресторану, у якій усі підрозділи працюють узгоджено, а керівництво отримує актуальні дані для прийняття рішень у режимі реального часу.

Висновок. Розроблена система демонструє, як інтеграція сучасних технологій здатна оптимізувати складні бізнес-процеси. Вона поєднує модульність, гнучкість та безпеку, створюючи основу для подальшого розвитку та масштабування. Проєкт підтверджує, що цифрові рішення можуть забезпечити узгоджену роботу різних підрозділів і підвищити загальну ефективність організації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Котенко, Н. В., Жирова, Т. О., Чибасевський, В. М., Десятко, А. В. (2019). Дослідження основних тенденцій сучасної розробки веб-сайтів. *Кибербезпека: освіта, наука, техніка*, (1) 5, 6-15.

[2] Романюк, Т. О., Кононенко, О. В. (2022). *Сучасні технології веб-програмування: архітектури, фреймворки та практичні аспекти розробки (монографія)*. Київ: КПІ ім. Ігоря Сікорського.

[3] Кривенко, І. П. (2021). Використання Java Spring Framework для створення корпоративних веб-застосунків. *Системи управління, навігації та зв'язку*, (4) 64, 125-130. <https://doi.org/10.26906/SUNZ.2021.4.125>.

[4] Сакун, Л., Ведініна, Ю., Шишлова, Ю. (2023). Вплив сучасних інформаційних технологій та командної роботи на систему менеджменту організацій в умовах глобалізаційних викликів. *Modeling the Development of Economic Systems*, 85–92. <https://doi.org/10.31891/mdes/2023-7-12>.

[5] Журавель, О. В., Халахур, Ю. Л., Кремена, Р. М. (2025). Роль управлінських інформаційних систем у підтримці прийняття стратегічних рішень. *Актуальні питання економічних наук. Економічні науки*, (8). <https://doi.org/10.5281/zenodo.14955655>.

Pavlenko A.¹, Kozlov O.²

Research of Systems for Using Web Applications to Manage Team Interaction in Organisations

Abstract. *This article discusses the development of an information system for managing team interaction within organisations. It considers modern information technologies that promote effective communication, coordination and cooperation among employees. It analyses the specifics of working in conditions involving flexible forms of employment, remote working, and complex organisational structures.*

Keywords: *information system; team interaction management; communication in the organisation; digital transformation; system adaptability; data security; project tool integration.*