НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки Кафедра обчислювальної техніки

«На правах рукопису»

УДК 0 0 4 . 5 8

До захисту допущено:
В.о. Завідючого кафердри
_____Михайло НОВОТАРСЬКИЙ
«___»_____2025 р.

Магістерська дисертація

за освітньо-професійною програмою «Інженерія програмного забезпечення комп'ютерних систем» зі спеціальності 121 “Інженерія програмного забезпечення” на тему: “ Система управління ресурсами та персоналом в готельному бізнесі”

Виконав:
студент VI курсу, групи ІМ-42мп
Варган Євген Олегович

_____

Науковий керівник:
ас., *Ph.D.*,
Коронко Дмитро Володимирович

_____

Консультант з нормоконтролю:
проф., д.т.н.,
Жабін Валерій Іванович

_____

Рецензент:
Ст.викладач кафедри СПіСКС,
Рядченко Костянтин Олександрович

_____

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

Київ – 2025 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
(повне найменування інституту, факультету)

Кафедра обчислювальної техніки
(повна назва кафедри)

Рівень вищої – освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма – «Інженерія програмного забезпечення комп'ютерних систем»

До захисту допущено:

В.о. завудіючого кафедри

_____ Михайло НОВОТАРСЬКИЙ
(підпис) (ім'я, прізвище)

«___»_____2025 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Варгану Євгену Олеговичу

1. Тема проекту *"Система управління ресурсами та персоналом в готельному бізнесі"*, керівник дисертації *Коренко Дмитро Володимирович, Ph.D., ас.* , затверджені наказом по університету від «*06*» *11* 2025р. № 4841-с

2.Термін подання студентом проекту *«15» 12 2025.*

3.Вихідні дані до проекту: *технічне завдання, теоретичні дані, наукові публікації.*

4.Зміст пояснювальної записки: *огляд існуючих веб-сервісів, розгляд схем їх реалізації, а також принципів роботи, пошук та аналіз їх основних переваг та недоліків, проектування та розробка програмного забезпечення, тестування програмного модулю та аналіз отриманих результатів.*

5.Перелік завдань, які потрібно розробити: *огляд існуючих веб-сервісів, розгляд схем їх реалізації, а також принципів роботи, проектування та розробка програмного забезпечення, тестування програмного модулю та аналіз отриманих результатів*

6.Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1.			
2.			
3.			
4.			

2.Дата видачі завдання 01.09.2025 року

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту(роботи)	Примітки
1	*Обговорення теми дисертації. Визначення предмету дослідження*	*1.09.25-7.09.25*	
2	*Дослідження існуючих рішень та проблем у досліджуваній області*	*22.09.25-28.09.25*	
3	*Побудова архітектурного рішення*	*29.09.25-05.10.25*	
4	*Програмна реалізація створеного алгоритму*	*06.10.25-10.10.25*	
5	*Програмування та загальне налагодження системи.*	*10.10.25-19.10.25*	
6	*Розробка стартам-проекту*	*14.11.25-18.11.25*	
7	*Оформлення магістерської дисертації.*	*19.11.25-24.11.25*	

Студент _____ Євген ВАРГАН
(підпис)

Керівник _____ Дмитро КОРЕНКО
(підпис)

РЕФЕРАТ
на магістерську дисертацію

виконану на тему: система управління ресурсом та персоналом у готельному бізнесі
студентом: Варганом Євгеном Олеговичом

Робота складається із вступу та чотирьох розділів. Загальний обсяг роботи: 112 аркушів основного тексту, 22 ілюстрації, 22 таблиці. При підготовці використовувалася література з 25 різних джерел.

Актуальність теми. Актуальність розробки інтегрованої веб-системи для управління ресурсами та персоналом у готельному бізнесі визначається кількома ключовими аспектами. В умовах зростаючої конкуренції та високих вимог клієнтів до якості обслуговування, готелі потребують ефективних інструментів для оптимізації операційних процесів (управління номерним фондом, прибиранням, складським обліком).

По-перше, традиційні методи планування персоналу часто є неефективними, що призводить до нерівномірного навантаження, помилок у нарахуванні заробітної плати та зниження рівня задоволеності як клієнтів, так і співробітників. Розробка спеціалізованої системи дозволить автоматизувати планування змін (робочого часу), облік завдань та комунікацію між адміністрацією та лінійним персоналом.

По-друге, необхідність у прозорості та контролі над використанням матеріальних ресурсів (засоби гігієни, білизна, витратні матеріали) вимагає впровадження сучасних цифрових рішень. Створення веб-платформи, що інтегрує функції Resource Management System (RMS) та Human Resource Management (HRM), забезпечить точний облік, мінімізує втрати та підвищить загальну фінансову ефективність готелю.

Мета роботи – розробка та впровадження інтегрованої веб-системи для ефективного управління ресурсами та персоналом у готельному бізнесі, що забезпечить оптимізацію операційних процесів, підвищення прозорості та якості обслуговування.

Для досягнення цієї мети було визначено та вирішено наступні завдання:

- Провести комплексний аналіз потреб готельного бізнесу в автоматизації управління персоналом (HRM) та ресурсами (RMS).
- Оглянути та проаналізувати існуючі галузеві системи для виявлення їхніх переваг і недоліків.
- Розробити архітектуру та функціональні модулі системи, використовуючи сучасні підходи до проектування програмного забезпечення (наприклад, мікросервісний підхід).
- Реалізувати ключовий функціонал, включаючи автоматизоване планування робочих змін, управління завданнями, облік номерного фонду та моніторинг використання ресурсів.
- Забезпечити високий рівень безпеки, авторизації та розмежування доступу до конфіденційної інформації.
- Розробити стартап-проєкт на базі платформи та визначити стратегію її виходу на ринок.

Об'єкт дослідження – процес роботи інтегрованої веб-системи для управління ресурсами та персоналом готельного бізнесу.

Предмет дослідження – методи та засоби забезпечення ефективного функціонування веб-системи для автоматизації процесів HRM та RMS.

Наукова новизна полягає у створенні інтегрованої веб-системи, яка поєднує функції управління персоналом (планування змін, облік робочого часу) та управління фізичними ресурсами (стан номерів, складський облік) в єдиній, гнучкій, масштабованій архітектурі, адаптованій до специфічних потреб готельної індустрії.

Особистий внесок здобувача. Магістерське дослідження являє собою авторський науково-практичний проєкт, в якому здобувач самостійно розробив та впровадив інтегровану веб-систему для управління операційними ресурсами та персоналом готельного бізнесу. Робота відображає особисто отримані прикладні результати з оптимізації процесів HRM (управління персоналом) та RMS (управління ресурсами), що забезпечує підвищення

ефективності обслуговування. Формулювання фінальної мети та ключових завдань дослідження проводилося спільно з науковим керівником.

Практична цінність. Створена система надає можливість готельним підприємствам:

- Зменшити операційні витрати та мінімізувати помилки в плануванні завдяки автоматизації.
- Підвищити продуктивність персоналу шляхом чіткого розподілу завдань та контролю їх виконання.
- Забезпечити прозорість у використанні ресурсів та прискорити реагування на потреби гостей.

Ключові слова: HRM, RMS, готельний бізнес, веб-система, управління персоналом, управління ресурсами, планування змін, Angular, .NET, SQL.

ЗМІСТ

ВСТУП

Сучасний готельний бізнес функціонує в умовах високої конкуренції, динамічних змін у потребах клієнтів та необхідності безперервної оптимізації операційних процесів. Ефективність роботи готелю, якість обслуговування та, зрештою, фінансова успішність, безпосередньо залежать від злагодженого управління ключовими ресурсами – матеріально-технічною базою та, що особливо важливо, персоналом. В умовах глобалізації, економічних викликів та зростаючих вимог до персоналізації послуг, традиційні методи управління стають недостатньо гнучкими та прозорими.

Важливим аспектом забезпечення стабільності та розвитку готельного бізнесу є впровадження інноваційних цифрових рішень, які дозволяють автоматизувати рутинні процеси, підвищити продуктивність праці та забезпечити ефективний контроль за використанням ресурсів. Зокрема, комплексна система управління ресурсами та персоналом (HRM & RMS) стає ключовим інструментом для координації роботи різних відділів, планування робочого часу, управління запасами та моніторингу завантаженості співробітників.

Наявні на ринку програмні продукти часто є або надмірно універсальними, що вимагає значної адаптації під специфіку готельної галузі, або мають обмежений функціонал, не здатний забезпечити повну інтеграцію процесів управління персоналом та матеріальними ресурсами одночасно. Відсутність єдиного, прозорого та гнучкого інструменту для оперативного контролю за графіками роботи, розподілом завдань та станом інвентарю може призводити до надмірних витрат, помилок у бронюванні та, як наслідок, до зниження рівня задоволеності клієнтів. Таким чином, виникає гостра потреба у розробці спеціалізованого веб-сервісу, який би враховував унікальні потреби готельного бізнесу та забезпечував централізоване, ефективне та прозоре управління.

Метою даної магістерської роботи є розробка та впровадження системи управління ресурсами та персоналом, яка забезпечить підвищення операційної ефективності готельного комплексу, оптимізацію використання матеріальних ресурсів та поліпшення якості управління трудовими процесами. Реалізація такого проєкту має сприяти не лише скороченню витрат, але й підвищенню загальної конкурентоспроможності готелю на ринку.

Практична цінність роботи полягає у створенні готового до використання, адаптивного інструменту, який забезпечує синхронізацію даних про персонал та матеріально-технічну базу, що є основою для прийняття обґрунтованих управлінських рішень. Для досягнення поставленої мети робота послідовно висвітлює теоретичні основи управління ресурсами та персоналом, проводить аналіз існуючих рішень, деталізує архітектуру та етапи розробки веб-системи, а також демонструє результати її тестування та впровадження.

РОЗДІЛ 1
ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ДЛЯ ВЕБ-СЕРВІСІВ ДЛЯ ЗАКУПІВЕЛЬ У РЕСТОРАННОМУ БІЗНЕСІ

1.1 Аналіз потреб готельного бізнесу в автоматизації управління ресурсами та персоналом

Сучасний готельний бізнес функціонує в умовах високої динамічності процесів, яка вимагає миттєвої реакції на зміни попиту та постійного підтримання бездоганного рівня сервісу. Ефективність роботи готелю безпосередньо залежить від злагодженого управління ключовими активами: персоналом та матеріально-технічною базою. В умовах посилення конкуренції та зростання вимог клієнтів до швидкості та якості обслуговування, традиційні, ручні методи управління стають неефективними, витратними та ризикованими. Необхідність впровадження автоматизованих систем продиктована не лише бажанням підвищити продуктивність, але й забезпеченням прозорості та контрольованості всіх операційних процесів.[1] Автоматизація дозволяє трансформувати хаотичні та розрізнені дані у єдину інформаційну базу, яка є основою для прийняття обґрунтованих управлінських рішень, прогнозування завантаження та ефективного розподілу ресурсів.

Дослідження показують, що впровадження автоматизації та технологій, зокрема IoT та II (штучного інтелекту), у готельному секторі значно підвищує ефективність використання ресурсів, дозволяє оптимізувати витрати енергії та покращити рівень обслуговування гостей. [2]

Автоматизація має охоплювати найбільш критичні та трудомісткі функціональні області готелю, оскільки саме в них накопичуються найбільші операційні втрати.

Бронювання, Заїзд/Виїзд (Check-in/Check-out): Цей процес включає реєстрацію гостей, розподіл номерного фонду та фінальне виставлення рахунків. Основною проблемою тут є несинхронізовані дані. Ручне внесення інформації, що надходить з різних каналів (онлайн-агрегатори, телефонні дзвінки, корпоративні запити), часто призводить до овербукінгу та помилок у тарифікації. Крім того, ручне заповнення великої кількості документів на рецепції значно збільшує час очікування гостей, що є прямим чинником зниження їхньої задоволеності.[3] Згідно з дослідженнями, автоматизація check-in/out процедур допомагає значно скоротити час обслуговування та знизити кількість помилок.

Обслуговування Номерів (Housekeeping): Процес стосується планування прибирання та відстеження поточного стану номера (чистий, брудний, у ремонті). Головна проблема - логістична неефективність і низька швидкість. Неефективна комунікація між службою клінінгу та рецепцією, відсутність автоматичного розподілу завдань за пріоритетом, призводить до затримок із заселенням (“waiting for cleaning”) і, відповідно, до простоїв номерного фонду. [4] Дослідження показують, що системи автоматизації housekeeping та інтеграція з певними інструментами планування змін покращують продуктивність працівників й скорочують час простою номерів.

Облік Робочого Часу та Планування Змін: Цей аспект включає складання гнучких графіків для всіх відділів та відстеження фактично відпрацьованих годин. Основна проблема полягає у відсутності гнучкості та обліку навантаження. Ручне планування часто не враховує пікове завантаження готелю, що призводить або до понаднормових годин і, як наслідок, до додаткових витрат, або до простоїв персоналу. Це також викликає складнощі з точним табелюванням та об’єктивним розрахунком заробітної плати. Сучасні автоматизовані рішення дозволяють прогнозувати навантаження на персонал, автоматично формувати графіки та знижувати витрати на понаднормову роботу.[2]

Управління Матеріальними Ресурсами (Склад): Процес охоплює облік усіх запасів (рушники, міні-косметика, миючі засоби), контроль термінів придатності та формування замовлень на закупівлю. Головною проблемою є відсутність точного контролю витрат. Неточний облік може призводити до надмірних запасів (що заморожує оборотні кошти) або, навпаки, до дефіциту ключових позицій, що безпосередньо знижує якість клієнтського сервісу. Інтеграція IoT сенсорів та аналітики дозволяє готелям більш точно контролювати запаси, зменшувати витрати та підвищувати рівень обслуговування.[1]

Таким чином, головна потреба готельного бізнесу полягає у створенні цілісної, інтегрованої системи, здатної забезпечити синергію між управлінням людським капіталом та матеріально-технічною базою. Розроблювана система управління ресурсами та персоналом (об'єднання HRM та елементів RMS) має вирішити виявлені проблеми за рахунок: централізації даних (створення єдиного робочого середовища); оптимізації трудових процесів (автоматичне формування графіків змін на основі прогнозованої завантаженості); підвищення операційної швидкості (миттєве оновлення статусу номера); та зниження операційних витрат (точний облік витратних матеріалів). Це дозволить досягти максимальної операційної ефективності та підвищити конкурентоспроможність готельного комплексу.

Переваги автоматизації в готелях включають підвищення персоналізації сервісу, покращення екологічної та економічної стійкості, а також зниження трудомісткості та витрат.

1.2 Огляд та порівняльний аналіз існуючих HRM та RMS систем для готельного сектору

У цьому підпункті слід здійснити системний аналіз ринку програмного забезпечення, яке вже використовується в готельному бізнесі для

автоматизації управління ресурсами (RMS) та персоналом (HRM). Зокрема, розглянемо популярні системи управління готелем (PMS) з функціоналом, що має пряме відношення до HRM/RMS, та оцінити їхні можливості, переваги й обмеження в контексті ваших вимог.

1.2.1 Вибрані системи для аналізу

Для цілей аналізу обрано такі рішення:

Oracle Hospitality OPERA PMS (раніше “OPERA”) - одне з найбільш розповсюджених PMS-рішень у світі, яке забезпечує комплексне управління готелем: бронювання, check-in/check-out, управління номерним фондом, housekeeping, CRM та продажі. [5]

MICROS-Fidelio Suite8 PMS (або “Fidelio”) - традиційна система, поширена в європейських готелях, що дозволяє інтегруватися з зовнішніми каналами, керувати запасами та оптимізувати розподіл номерів. [6]

1С: Готель (1С: Hotel) - локальне рішення з модульною архітектурою, орієнтоване на автоматизацію різноманітних процесів готельного бізнесу, включаючи функції управління персоналом (HRM) та ресурсами (RMS). [7]

Mews PMS - хмарна платформа нового покоління, яка об’єднує управління бронюваннями, номерним фондом, фінансами та персоналом, з інтуїтивним інтерфейсом і широкими можливостями інтеграцій із зовнішніми сервісами. [8]

Cloudbeds PMS - універсальна хмарна система для готелів, хостелів та орендних об’єктів, що поєднує PMS, канал-менеджер, систему бронювання та аналітичні інструменти, забезпечуючи автоматизацію ключових процесів і синхронізацію з OTA. [9]

1.2.2. Порівняльний аналіз функціоналу існуючих систем

Oracle Hospitality OPERA PMS

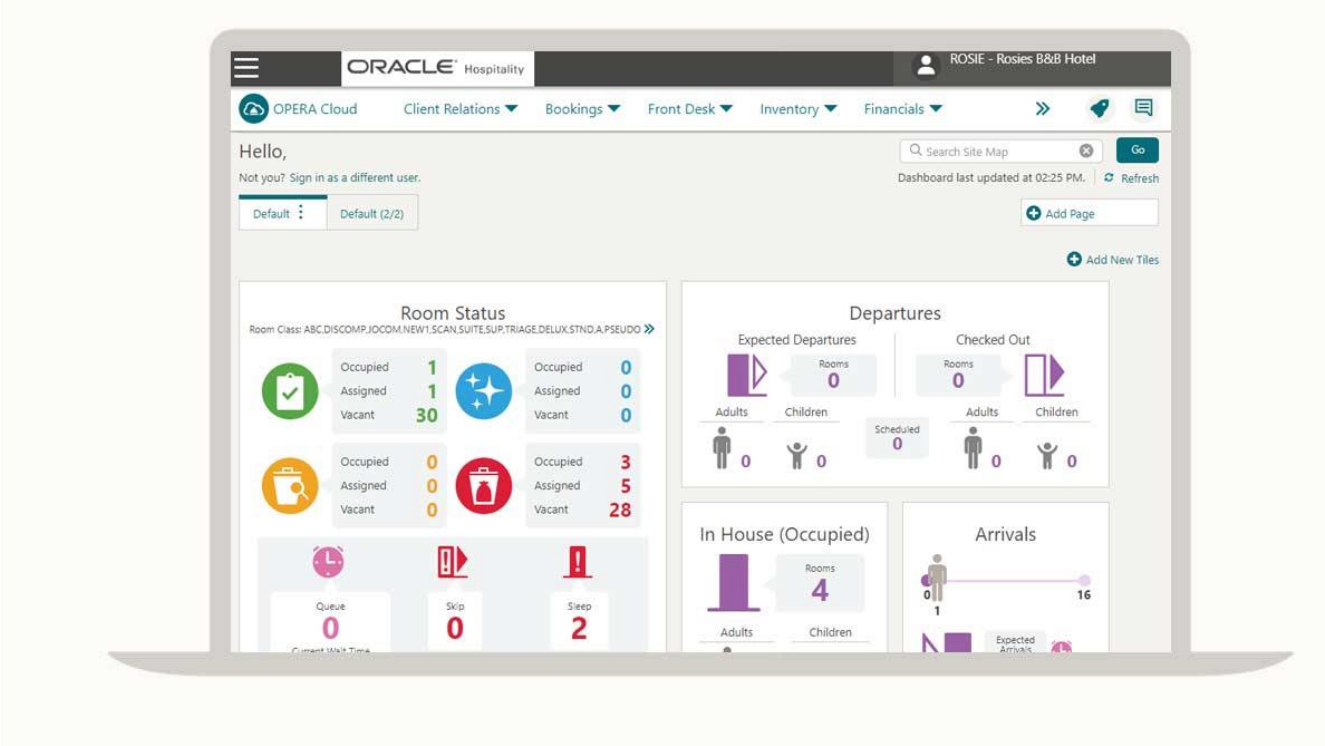

Рис 1.1. Графічний інтерфейс застосунку OPERA PMS

Система побудована на глибоко інтегрованих модулях, які автоматизують повний цикл взаємодії з гостем та внутрішні операції.

Фронт-офіс і бронювання є серцем OPERA, забезпечуючи ефективну взаємодію з клієнтами. Модуль керує процесами реєстрації (Check-in) та виїзду (Check-out), а також дозволяє гнучко управляти тарифами, депозитами та бронюваннями. Важливою особливістю є автоматична синхронізація з онлайн-турагентствами (OTA) та глобальними дистрибутивними системами (GDS), що гарантує миттєве оновлення даних про наявність номерів та цін у всіх каналах.

Контроль номерного фонду сфокусований на оптимізації роботи хаускіпінгу та технічного обслуговування. Система надає можливість відстежувати стан номерів у реальному часі (чистий, брудний, у ремонті), планувати графіки прибирання та технічного обслуговування. Застосування

мобільних додатків для персоналу дозволяє оперативно оновлювати статуси, суттєво прискорюючи готовність номерів до заселення.

Фінансовий та аналітичний блок охоплює фінансові потоки та управлінський облік. Він відповідає за формування рахунків, керування платежами та комісіями, а також підтримує інтеграцію з корпоративними бухгалтерськими програмами. Ключова цінність цього модуля полягає в інструментах аналітики, які дозволяють керівництву оцінювати доходи та витрати, оптимізуючи цінову політику та стратегічні рішення.

Для забезпечення злагодженої внутрішньої роботи передбачено управління персоналом, що включає планування змін та відпусток, а також моніторинг продуктивності співробітників. Цей модуль також підтримує інтеграцію з зовнішніми HRM-системами.

CRM та програми лояльності є інструментами для підвищення лояльності та персоналізації обслуговування. Система веде детальні профілі гостей з історією їхніх перебувань, що дозволяє ефективно керувати програмами лояльності, створювати персоналізовані пропозиції та проводити точну сегментацію клієнтів для цільових маркетингових кампаній.

Гнучкість OPERA підкреслюється її інтеграційними можливостями та API. Підтримка відкритих API дозволяє легко підключати сторонні додатки, включаючи системи продажу (POS), системи планування ресурсів підприємства (ERP) та інші CRM, забезпечуючи повну екосистему готельного менеджменту.

З технічної точки зору, OPERA є універсальною: вона доступна в обох архітектурах - хмарній (SaaS) або локальній (on-premise). Система демонструє високий рівень глобальної адаптивності, підтримуючи понад 20 мов, включно з українською, та понад 200 валют, що робить її ідеальною для міжнародних операцій. Крім того, Oracle Hospitality OPERA PMS відповідає суворим міжнародним стандартам безпеки, зокрема GDPR та PCI DSS, а доступ через мобільні додатки для персоналу та гостей забезпечує високу оперативність та якість обслуговування.

Переваги:

- Масштабованість під будь-який розмір готелю.
- Широкі можливості інтеграції з різними системами.
- Потужна аналітика для прийняття управлінських рішень.
- Гнучке налаштування під потреби конкретного закладу.

Обмеження:

- Високі витрати на ліцензію та впровадження для малих готелів.
- Складність налаштування вимагає кваліфікованого персоналу.
- Частина функцій може потребувати локалізації для українського ринку.

Аналіз для України:

- Переваги: інтеграція з міжнародними OTA/GDS, підтримка української мови та гривні, доступ до аналітичних інструментів.
- Обмеження: висока ціна для малих готелів, адаптація до локального законодавства, обмежена кількість сертифікованих партнерів в Україні.

MICROS-Fidelio Suite8 PMS

MICROS Fidelio Suite8 PMS - комплексне рішення для управління готелем, орієнтоване на координацію операційних процесів: бронювання, фронт-офіс, фінанси, обслуговування гостей та аналітика. Призначене для середніх і великих готельних комплексів, з можливістю локальної або хмарної інсталяції(Рис 1.2).

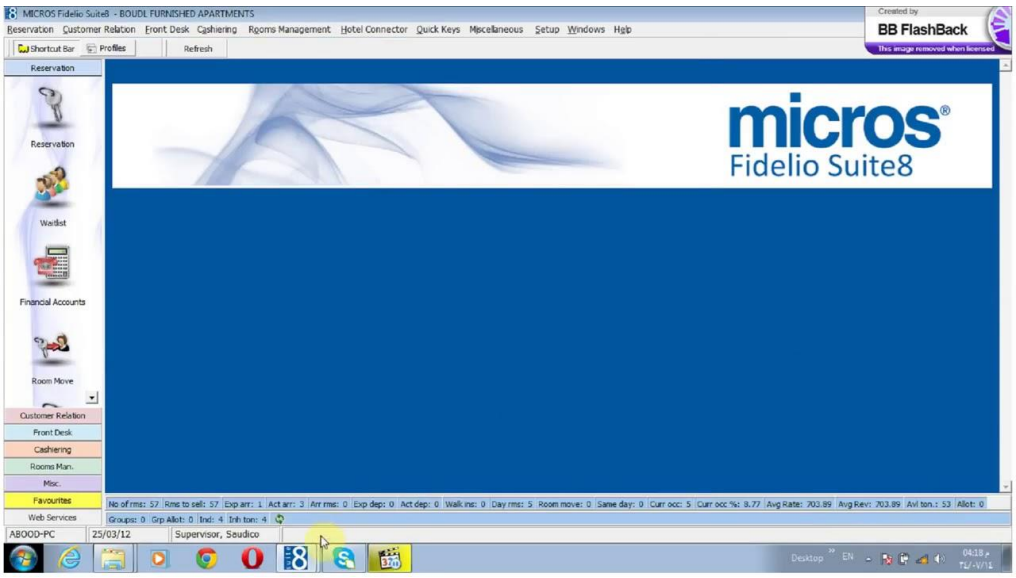

Рис 1.2. графічний інтерфейс застосунку MICROS-FIDELIO

Система підтримує як локальне розгортання, так і хмарну модель (SaaS), що дає змогу готелям обрати найбільш зручний формат залежно від технічних можливостей та рівня безпеки даних. Її інтерфейс багатомовний, включаючи українську, а підтримка понад 200 валют дозволяє ефективно працювати з міжнародними гостями. Завдяки сучасним механізмам шифрування, багаторівневій авторизації та системам резервного копіювання, MICROS Fidelio Suite8 забезпечує високий рівень захисту даних та стабільність роботи.

Одним із головних переваг рішення є повна автоматизація фронт-офісних процесів: від реєстрації та виїзду гостей до управління бронюваннями, тарифами та синхронізації з глобальними системами бронювання (OTA, GDS). Це значно підвищує швидкість обслуговування та знижує кількість помилок, пов’язаних із ручним введенням даних.

Модуль управління номерним фондом дозволяє у реальному часі відстежувати стан номерів, планувати прибирання і технічне обслуговування, а також забезпечує мобільні інструменти для персоналу Housekeeping. У фінансовому блоці реалізовані функції контролю платежів, формування рахунків, управління комісіями та інтеграції з бухгалтерськими системами, що спрощує фінансову звітність і підвищує прозорість операцій.

Окрему увагу приділено модулю HRM - плануванню змін, обліку відпусток і моніторингу ефективності працівників. Це допомагає адміністрації оптимізувати робочі графіки та забезпечити стабільну якість сервісу. Завдяки вбудованому CRM-гарантовано високий рівень персоналізації обслуговування: система зберігає історію перебувань гостей, підтримує бонусні програми та автоматизує розсилку персоналізованих пропозицій.

Важливою перевагою MICROS Fidelio Suite8 є її відкритість до інтеграцій. Вона підтримує API-з'єднання з POS, ERP, CRM та іншими бізнес-системами, що забезпечує єдиний інформаційний простір і масштабування під конкретні потреби готелю. Мобільні додатки та веб-доступ до CRM дозволяють менеджменту та персоналу оперативно реагувати на запити гостей і контролювати основні показники діяльності навіть поза офісом.

Переваги:

- Підходить для середніх і великих мереж.
- Широкий функціонал управління операціями та аналітикою.
- Можливість інтеграції з іншими системами.

Обмеження:

- Висока ціна ліцензії та впровадження.
- Потребує навчання персоналу.
- Локалізація під українські стандарти потребує додаткових налаштувань.
- HRM обмежений, потребує зовнішніх рішень.

Застосування в Україні:

- Переваги: підходить для міжнародних мереж, підтримує багатовалютність і мультимовність, дозволяє централізовано керувати кількома об'єктами.
- Обмеження: високі витрати для малих готелів, локалізація під законодавство та бухгалтерські стандарти, обмежена кількість сертифікованих партнерів.

Mews

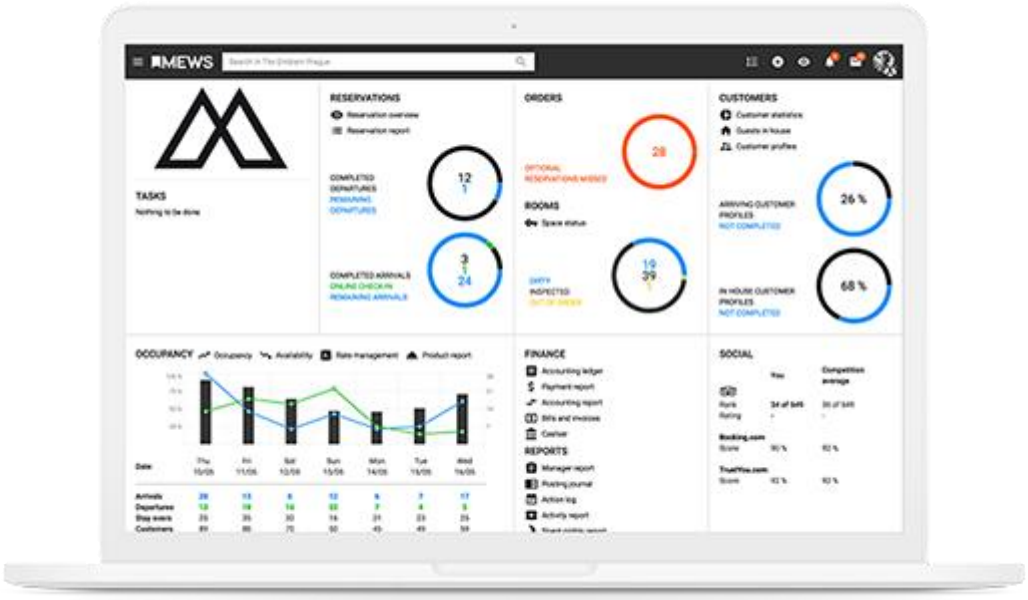

Рис 1.3. Графічний інтерфейс застосунку MEWS

Система базується на хмарній моделі (SaaS), що дозволяє керувати всіма операціями з будь-якого пристрою - комп’ютера, планшета чи смартфона. Такий підхід не лише спрощує доступ до даних, а й забезпечує стабільну роботу системи без потреби у складній локальній інфраструктурі. Mews підтримує понад 20 мов, серед яких і українська, а також понад 200 валют, що робить платформу зручною для міжнародних гостей.

У модулі бронювання реалізовано повний цикл управління: від онлайн-бронювання через сайт чи мобільний додаток до автоматичного оновлення тарифів і синхронізації з глобальними системами OTA та GDS. Це дає змогу мінімізувати ручну роботу персоналу та знизити ризик подвійних бронювань.

Фронт-офіс Mews забезпечує швидке оформлення заїзду та виїзду гостей (Check-in/Check-out), включаючи можливість використання цифрових ключів, що особливо актуально в умовах безконтактного сервісу. Система також зберігає профілі гостей з їхніми уподобаннями, дозволяючи персоналізувати обслуговування та підвищити рівень задоволеності клієнтів.

Модуль управління номерним фондом забезпечує відображення стану номерів у режимі реального часу, що допомагає ефективно планувати прибирання, технічне обслуговування та інші операції. Мобільний доступ для співробітників покоївки дозволяє оперативно оновлювати інформацію, що зменшує затримки й підвищує загальну продуктивність.

У фінансовому блоці система автоматизує процеси виставлення рахунків, обробки платежів і формування фінансової звітності. Інтеграція з бухгалтерськими системами забезпечує точність розрахунків і прозорість фінансового контролю. Модуль управління персоналом дає змогу планувати робочі зміни, вести облік відпусток і контролювати продуктивність працівників, що особливо корисно для готелів із великою кількістю співробітників.

Окремою перевагою Mews є її відкритість до інтеграцій. Платформа має API та доступ до Mews Marketplace - екосистеми додатків і сервісів, сумісних із PMS. Завдяки цьому готелі можуть легко підключати додаткові інструменти - від POS і CRM до ERP-систем - і розширювати функціональність платформи відповідно до власних потреб.

Високий рівень безпеки даних забезпечується багаторівневою авторизацією, шифруванням і відповідністю міжнародним стандартам PCI DSS. Це гарантує захист фінансової інформації та конфіденційних даних гостей.

Переваги:

- Автоматизація операцій зменшує ручну працю.
- Гнучке налаштування під специфіку готелю.
- Інтуїтивний інтерфейс з мінімальним часом навчання.
- Широкі можливості інтеграцій та масштабування.

Обмеження:

- Висока вартість для малих закладів.
- Залежність від стабільного інтернет-з’єднання.
- Деякі функції потребують адаптації під українські стандарти.

Використання в Україні:

- Переваги: підходить для міжнародної аудиторії, інтерфейс українською, централізоване управління кількома об’єктами, аналітика доходів та завантаження номерів.
- Обмеження: висока ціна та складність впровадження, потреба в локалізації.

1С: Готель

1С: Готель - це спеціалізоване програмне рішення, створене для комплексної автоматизації управлінських і операційних процесів у сфері готельного бізнесу. Його гнучка модульна структура дає змогу адаптувати систему під потреби різних типів закладів - від невеликих сімейних готелів до великих мережевих комплексів. Головна мета системи - забезпечити ефективне управління всіма аспектами діяльності готелю, зменшити навантаження на персонал та підвищити якість обслуговування гостей.

Ключовим елементом рішення є модуль управління номерним фондом і бронюваннями, який надає простий та інтуїтивний інтерфейс для реєстрації гостей, оформлення заїздів і виїздів, обробки бронювань та керування тарифними планами. Система підтримує синхронізацію з онлайн-каналами бронювання, що забезпечує актуальність даних та зменшує ризик подвійних бронювань.

Фінансовий блок системи 1С: Готель автоматизує основні операції з обліку - формування рахунків, роботу з різними валютами, облік платежів і комісій. Інтеграція з платіжними системами та можливість створення детальних звітів дозволяють керівництву оперативно аналізувати фінансові результати, контролювати витрати та планувати прибуток.

Важливу роль у системі відіграє модуль управління персоналом, який забезпечує повний цикл роботи з кадрами: від обліку робочого часу і складання графіків змін до розрахунку заробітної плати та контролю виконання завдань. Такий підхід сприяє ефективному плануванню роботи команди та зменшенню адміністративних витрат.

Модуль CRM та маркетинг допомагає створювати довгострокові відносини з клієнтами. Система збирає і аналізує інформацію про гостей, підтримує програми лояльності та автоматизовані розсилки. Це дозволяє готелю формувати персоналізовані пропозиції, підвищуючи рівень задоволеності клієнтів і повторні продажі.

Завдяки розвиненій системі інтеграцій, 1С: Готель легко взаємодіє з іншими бізнес-додатками: бухгалтерськими системами, POS-терміналами, системами електронних замків тощо. Відкриті API забезпечують можливість розширення функціональності, що робить рішення придатним для масштабування та налаштування під індивідуальні потреби конкретного закладу.

З технічного боку система побудована на платформі 1С:Підприємство 8, яка відома своєю стабільністю, надійністю та широкими можливостями кастомізації. Архітектура може бути як локальною, так і хмарною, що дає готелям свободу вибору способу розгортання. Програма підтримує українську, російську та англійську мови інтерфейсу, а також сумісна з операційними системами Windows і Linux (через емулятори). Безпека даних забезпечується за рахунок багаторівневої авторизації та сучасних засобів шифрування.

Переваги:

- Локалізація під українське законодавство та бухгалтерські стандарти.
- Широкі можливості інтеграції з різними пристроями та сервісами.
- Зменшення ручної праці та підвищення ефективності роботи персоналу.
- Інструменти для збору та аналізу даних для прийняття управлінських рішень.

Обмеження:

- Початкові витрати на впровадження можуть бути високими.
- Потреба у навчанні персоналу для повноцінного використання системи.
- Вимоги до технічної інфраструктури для стабільної роботи.

Застосування в Україні:

- Переваги: добре локалізована, наявність українських партнерів і сервісної підтримки; популярна серед місцевих мереж готелів і санаторіїв.
- Обмеження: потребує інвестицій у впровадження та навчання персоналу також є російським продуктом, що унеможливлює використання.

Cloudbeds

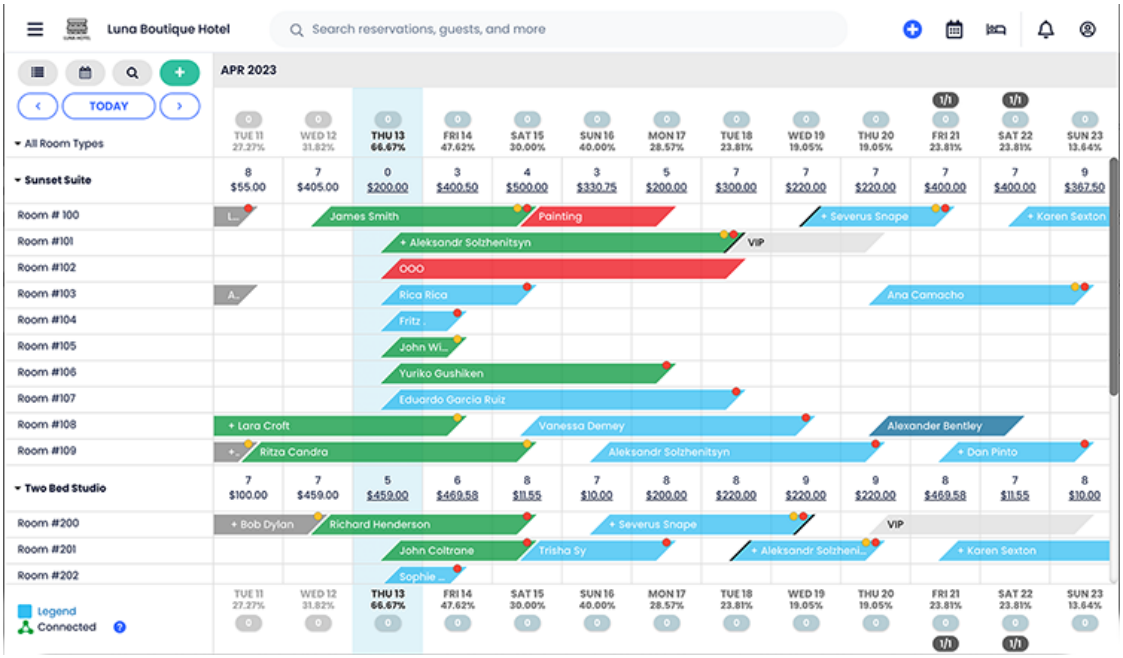

Рис 1.4. Графічний інтерфейс застосунку Cloudbeds

Завдяки інтуїтивному інтерфейсу модуль бронювання та управління номерним фондом дає змогу легко створювати, редагувати або скасовувати бронювання, контролювати доступність номерів у режимі реального часу й навіть обробляти групові заявки. Усі зміни автоматично синхронізуються між каналами, що мінімізує ризики подвійних бронювань і помилок у роботі.

Невід'ємною частиною Cloudbeds є канальний менеджер, який об'єднує управління популярними онлайн-платформами бронювання -

Booking.com, Expedia, Airbnb та іншими. Усі тарифи та дані про наявність номерів оновлюються миттєво, що дозволяє підтримувати актуальність інформації без ручного втручання та забезпечувати рівні умови для всіх каналів продажу.

Вбудована система бронювання (Booking Engine) інтегрується безпосередньо з вебсайтом готелю, що дає можливість приймати бронювання напряму, без посередників. Вона підтримує акції, спеціальні пропозиції та персоналізовані тарифи, стимулюючи прямі продажі й підвищуючи прибутковість бізнесу.

Модуль управління доходами надає готельєрам глибоку аналітику щодо прибутків, сезонних тенденцій і ефективності тарифної політики. Завдяки інтеграції з системами Revenue Management користувачі можуть впроваджувати динамічне ціноутворення, оперативно реагуючи на зміни попиту й ринкових умов.

Важливим елементом платформи є інструменти комунікації з гостями, які автоматизують взаємодію протягом усього циклу перебування. Система може надсилати автоматичні повідомлення про підтвердження бронювання, нагадування про заїзд, інформацію про послуги чи опитування після виїзду. Крім того, Cloudbeds підтримує інтеграцію з мобільними додатками для self-check-in, що відповідає сучасним тенденціям безконтактного сервісу.

Аналітичний блок платформи забезпечує детальну звітність та статистику - від фінансових результатів до рівня завантаження номерів і ефективності каналів продажу. Усі дані можна експортувати для подальшого аналізу або інтеграції з іншими управлінськими системами.

З технічного боку Cloudbeds - це повністю хмарна система, яка працює через веб-браузер або мобільні додатки. Вона сумісна з усіма основними операційними системами - Windows, macOS, Android та iOS. Платформа підтримує українську мову інтерфейсу, а також десятки інших, що робить її універсальною для міжнародного використання.

Безпека даних у Cloudbeds забезпечується багаторівневою аутентифікацією, шифруванням і сучасними засобами захисту відповідно до міжнародних стандартів. Широкі можливості інтеграції через відкриті API та готові модулі дозволяють з'єднати платформу з будь-якими зовнішніми сервісами - від бухгалтерських систем до маркетингових інструментів.

Переваги:

- Універсальність: підходить для різних типів об'єктів.
- Зручність використання: інтерфейс інтуїтивно зрозумілий, швидке освоєння персоналом.
- Автоматизація процесів: мінімізує ручну працю та підвищує ефективність.
- Інтеграції з OTA та іншими сервісами: розширює канали продажу.
- Мобільність: доступ з будь-якого пристрою та місця.

Обмеження:

- Вартість може бути високою для малих закладів.
- Потребує стабільного інтернет-з'єднання.
- Для повної відповідності українським стандартам деякі функції потребують адаптації.

Аналіз для українського ринку:

- Переваги: локалізація українською мовою, підтримка гривні, інтеграція з локальними OTA, гнучка тарифікація.
- Обмеження: високі початкові витрати, необхідність налаштування під специфіку українського ринку, конкуренція з локальними продуктами.

1.3. Аналіз існуючих рішень

У цьому розділі проводиться порівняльний аналіз основних систем управління готельним бізнесом, які дозволяють автоматизувати ключові процеси: бронювання номерів, обслуговування гостей, управління персоналом, фінансовий облік та аналітику. Цей аналіз є важливим для визначення головних проблем індустрії та оцінки потенціалу розробки нової системи автоматизації, адаптованої до потреб сучасних готелів.

Метою дослідження є виявлення сильних і слабких сторін кожного з розглянутих рішень та оцінка їх ефективності у підвищенні продуктивності готельних операцій. Основна увага приділяється наступним аспектам:

• обсяг та глибина функціоналу (бронювання, check-in/check-out, управління номерним фондом, облік робочого часу, фінансовий контроль, CRM, аналітика);

• простота та зручність користування;

• можливості інтеграції з зовнішніми платформами (OTA, GDS, POS, ERP, HRM);

• вартість ліцензій та впровадження та сегмент ринку, на який орієнтована система;

• адаптація до локальних умов та вимог українського ринку.

Для аналізу були обрані п’ять популярних рішень:

• Oracle Hospitality OPERA PMS - комплексне рішення для управління бронюваннями, номерним фондом, фінансами, персоналом та CRM із інтеграцією до глобальних OTA і GDS. Серед сильних сторін - висока масштабованість, потужні аналітичні інструменти та багатомовність; серед обмежень - висока вартість, складність налаштувань для невеликих готелів та необхідність локалізації для України.

• MICROS-Fidelio Suite8 PMS - традиційна система для середніх та великих готелів, що забезпечує комплексне управління Front-office,

бронюванням, фінансами та аналітикою. Переваги - широкий функціонал, інтеграція з ERP та HRM-системами, можливість централізованого управління кількома об’єктами. Недоліки - висока ціна, потреба у навчанні персоналу, додаткове налаштування під українські стандарти.

• Mews - хмарна платформа, що автоматизує бронювання, Front-office, Housekeeping і цифрові ключі, з мобільним доступом для персоналу. Основні переваги - інтуїтивний інтерфейс, гнучкі налаштування та масштабованість; слабкі сторони - залежність від стабільного інтернет-з’єднання та висока вартість для малих готелів.

• 1С: Готель - локальне рішення з підтримкою українського законодавства та стандартів бухгалтерії, що включає RMS та HRM-функції. Переваги - локальна адаптація, наявність технічної підтримки, популярність серед малих і середніх готелів; обмеження - стартові витрати на впровадження, потреба навчання персоналу та вимоги до інфраструктури.

• Cloudbeds - хмарна платформа з PMS, канал-менеджером, системою бронювання, управління доходами та аналітикою. Сильні сторони - легкість використання, інтеграція з OTA, мобільність та універсальність для малих і незалежних готелів. Обмеження - залежність від інтернету, потреба адаптації під локальні умови та висока вартість для невеликих об’єктів.

Проведений аналіз дозволяє виділити головні переваги та недоліки кожного з рішень, оцінити їх придатність для різних типів готелів і визначити напрямки для розробки нової інтегрованої системи управління, яка поєднувала б ефективність, зручність та адаптацію під український ринок.

Висновки до розділу 1

Сучасний готельний бізнес в Україні стикається з рядом викликів, серед яких високий рівень конкуренції, необхідність контролю фінансів, ефективного управління персоналом та ресурсами. Це робить впровадження автоматизованих систем управління ключовим чинником підвищення ефективності та якості обслуговування гостей.

Аналіз таких систем, як Oracle Hospitality OPERA PMS, MICROS-Fidelio Suite8 PMS, Mews, 1С: Готель та Cloudbeds, показав, що вони дозволяють комплексно оптимізувати процеси бронювання, check-in/check-out, управління номерним фондом, фінансами та персоналом. Платформи підтримують інтеграцію з міжнародними каналами бронювання, надають аналітичні інструменти та забезпечують багатомовність і багатовалютність, що актуально для роботи з іноземними гостями.

Водночас більшість рішень орієнтовані на великі або міжнародні мережі готелів, мають значну вартість впровадження і ліцензій, а також потребують адаптації до українських законодавчих норм і бухгалтерських стандартів. Локальні системи, такі як 1С: Готель, забезпечують підтримку української мови і часткову адаптацію до національного ринку, але поступаються за рівнем інтеграції та аналітичного функціоналу міжнародним платформам.

Таким чином, існує потреба у створенні інтегрованого рішення для українських готелів, яке поєднувало б функціональність та масштабованість міжнародних PMS з локальною адаптацією, доступністю для малого та середнього бізнесу та ефективним управлінням персоналом, ресурсами й фінансами. Результати проведеного аналізу стануть основою для визначення вимог та побудови архітектури розроблюваної системи автоматизації готельного управління.

РОЗДІЛ 2
ПРОЄКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ РЕСУРСАМИ ТА ПЕРСОНАЛОМ

2.1. Back-end частина системи

Під час розробки системи управління ресурсами та персоналом у готельному бізнесі важливим етапом є створення серверної (back-end) частини, яка відповідає за обробку бізнес-логіки, керування даними, взаємодію з базою даних та забезпечення безпечного обміну інформацією з клієнтською частиною. Back-end виступає ядром системи, що забезпечує стабільну роботу всіх функціональних модулів, таких як облік персоналу, управління матеріальними ресурсами, планування змін, фінансові розрахунки та інші.

Для реалізації серверної частини необхідно обирати сучасні технології, що гарантують масштабованість, надійність і простоту інтеграції з іншими компонентами. Важливою вимогою є підтримка REST API для ефективного обміну даними з клієнтською частиною та сторонніми сервісами (наприклад, системами бронювання або бухгалтерськими модулями). [10]

2.1.1. Вибір архітектури

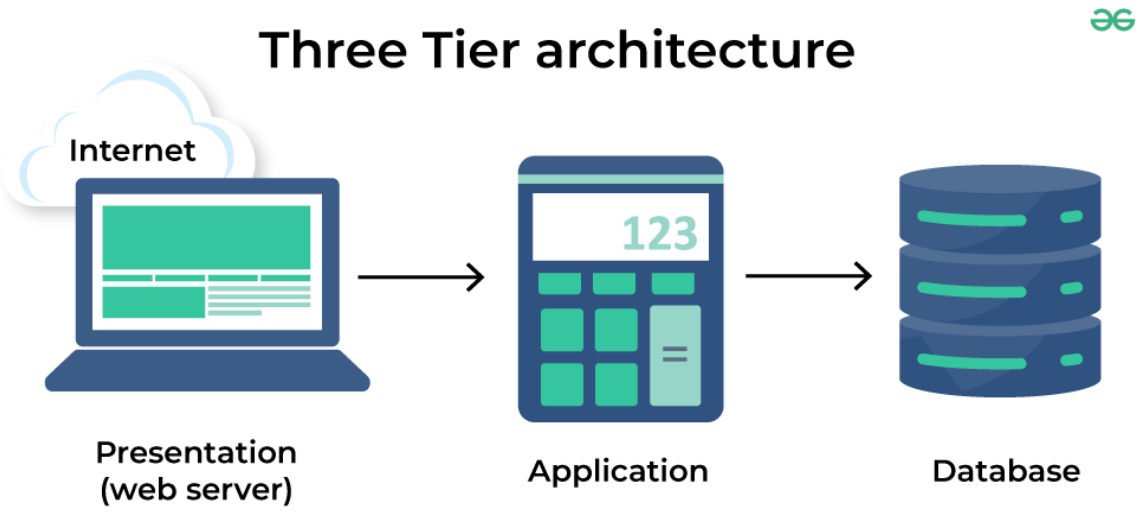

Рис 2.1. представлення три-шарової архітектури

Першим кроком при розробці back-end є визначення архітектурного підходу, який формує структуру взаємодії між модулями та рівнями програми. Для даної системи доцільним є застосування тришарової архітектури (three-tier architecture), що включає:

- рівень представлення (presentation layer) - взаємодія з користувачем через фронтенд;
- рівень логіки (application layer) - обробка бізнес-процесів, перевірок та запитів до бази даних;
- рівень даних (data layer) - зберігання, оновлення та пошук інформації у базі даних.

Така структура дозволяє чітко розділяти відповідальності, підвищує гнучкість, полегшує розширення та тестування системи. Для реалізації back-end можна обирати різні технологічні стекі та фреймворки, які підтримують модульну архітектуру, ін’єкцію залежностей, роботу з базами даних та забезпечують можливості реалізації REST API. Це можуть бути рішення на основі Node.js, Python, Java чи інших платформ, залежно від вимог проєкту. [11]

Архітектурна схема системи передбачає, що кожен функціональний модуль - управління персоналом, планування змін, облік номерного фонду, контроль матеріальних запасів - реалізується як окремий сервіс із взаємодією через загальний API. Такий підхід спрощує масштабування, дозволяє додавати нові функції без змін у базовому коді та забезпечує надійність і зручність супроводу програмного забезпечення, що особливо важливо для обробки великих обсягів даних у режимі реального часу.

2.1.2. Реалізація бізнес-логіки управління ресурсами та персоналом

Бізнес-логіка є центральним елементом серверної частини будь-якої інформаційної системи, що визначає послідовність виконання операцій, правила обробки даних та взаємодію між функціональними модулями. У системі управління ресурсами та персоналом у готельному бізнесі бізнес-логіка описує всі основні процеси, пов’язані з діяльністю готелю, координацією персоналу, плануванням ресурсів і забезпеченням ефективності управлінських рішень.

Основна мета побудови бізнес-логіки полягає у відокремленні прикладної частини системи від рівнів представлення та зберігання даних. Такий підхід забезпечує модульність, зручність тестування, розширюваність і повторне використання компонентів. [12]

Основні завдання бізнес-логіки:

- Обробка користувацьких запитів. Бізнес-логіка отримує запити від користувача через інтерфейс або API, перевіряє їхню коректність та передає дані на обробку або збереження.
- Реалізація бізнес-процесів. На цьому рівні визначаються алгоритми виконання дій - наприклад, порядок бронювання номерів, розподіл робочих змін, облік ресурсів або управління фінансовими потоками.

- Валідація даних. Усі вхідні дані перевіряються на відповідність встановленим правилам, щоб уникнути логічних помилок або некоректних операцій.
- Забезпечення цілісності інформації. Під час виконання складних транзакцій система має гарантувати узгодженість даних у всіх пов’язаних об’єктах.
- Керування ролями та доступом. Бізнес-логіка визначає, які користувачі можуть виконувати певні дії, залежно від їхніх посадових обов’язків або прав доступу.

При проєктуванні бізнес-логіки систем управління доцільно дотримуватись таких принципів:

- Інкапсуляція - ізоляція внутрішніх механізмів роботи модулів від зовнішніх компонентів;
- Модульність - поділ системи на окремі блоки, кожен з яких реалізує певну функцію;
- Повторне використання коду - уніфікація алгоритмів для подібних операцій;
- Гнучкість - можливість адаптації логіки під зміну бізнес-процесів без повної перебудови системи;
- Прозорість і контрольованість - забезпечення можливості відстеження стану виконання операцій і швидкого виявлення помилок.

Бізнес-логіка системи управління ресурсами та персоналом виступає основою її функціонування. Вона поєднує всі процеси діяльності готелю в єдину керовану структуру, забезпечуючи узгодженість операцій, ефективність управління та точність обліку.

2.1.3. Механізми авторизації, автентифікації та безпеки

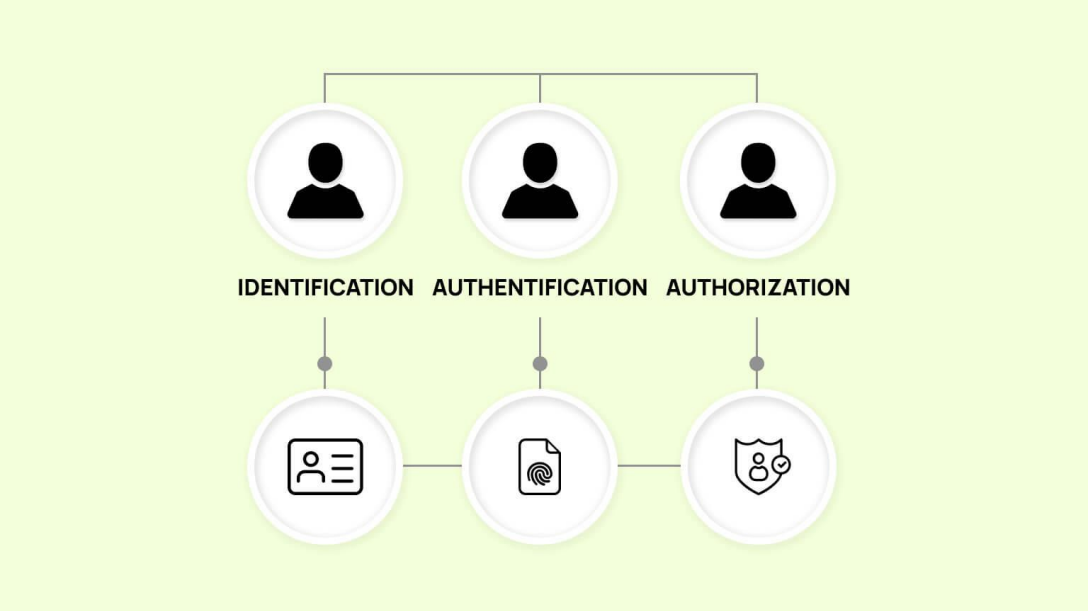

Рис 2.2. Три етапи валідації даних користувача

Одним із найважливіших аспектів побудови серверної частини інформаційної системи є забезпечення безпеки даних та контролю доступу користувачів. У системі управління ресурсами та персоналом у готельному бізнесі передбачається обробка конфіденційної інформації - персональних даних співробітників, фінансових звітів, внутрішньої документації тощо.

Тому механізми автентифікації, авторизації та захисту інформації мають бути інтегрованими складовими бізнес-логіки. [13]

Автентифікація - це процес перевірки достовірності особи користувача, який намагається отримати доступ до системи. Вона слугує першим рівнем безпеки та гарантує, що до системи можуть увійти лише зареєстровані користувачі.

У теоретичному аспекті автентифікація може бути реалізована різними способами:

- Однофакторна автентифікація - найпростіший підхід, який базується на перевірці пари «логін–пароль».
- Двофакторна автентифікація (2FA) - передбачає додатковий рівень перевірки, наприклад, через SMS-код, електронну пошту або мобільний застосунок.

- Багатофакторна автентифікація (MFA) - комбінує декілька незалежних методів і підвищує рівень захисту від несанкціонованого доступу.

Загальноприйнятим є використання системи шифрування паролів, що запобігає зберіганню відкритих даних у базі. Для цього застосовуються криптографічні хеш-функції, які перетворюють введені користувачем дані у зашифрований вигляд.

Авторизація - це процес надання користувачеві прав доступу до певних ресурсів або функцій після успішної автентифікації.

У системах управління персоналом і ресурсами авторизація є основним механізмом реалізації ролевої моделі доступу (Role-Based Access Control, RBAC).

Така модель передбачає визначення ролей (наприклад, адміністратор, менеджер, працівник, бухгалтер), кожна з яких має свій набір дозволених дій:

- адміністратор може створювати, редагувати та видаляти об’єкти системи;
- менеджер має доступ до перегляду інформації про персонал і ресурси;
- звичайний працівник може переглядати лише власні графіки чи робочі завдання.

Завдяки цьому забезпечується мінімізація ризиків несанкціонованих змін даних і контроль дій користувачів у межах їх компетенції.

Безпека даних у серверній частині системи визначається комплексом організаційних і технічних заходів, спрямованих на захист інформації від несанкціонованого доступу, спотворення або втрати.

У межах серверної частини систем безпека досягається за допомогою низки методів і технологій:

- Шифрування даних - застосовується під час передачі інформації між клієнтською і серверною частинами для запобігання перехопленню (наприклад, через протоколи HTTPS, SSL/TLS).
- Контроль сесій - відстеження активних користувацьких сесій і встановлення часу їхньої дії для уникнення несанкціонованого доступу.

- Журналювання дій (логування) - фіксація подій і запитів для подальшого аудиту та аналізу безпеки.
- Валідація вхідних даних - перевірка всіх даних, що надходять від користувачів, з метою запобігання SQL-ін’єкціям, міжсайтовим атакам (XSS) та іншим спробам зловмисного втручання.
- Резервне копіювання - регулярне створення копій баз даних і системних файлів для відновлення інформації у разі збоїв або атак.

Крім технічних методів, важливу роль відіграють і організаційні заходи:

- створення політики безпеки підприємства;
- регламентація доступу персоналу до різних рівнів системи;
- періодичне оновлення паролів і ключів шифрування;
- навчання персоналу правилам кібергігієни.

Отже, механізми автентифікації, авторизації та безпеки становлять невід’ємну частину серверної архітектури системи управління ресурсами та персоналом. Їхнє правильне поєднання дозволяє забезпечити надійний захист даних, підвищити рівень довіри до системи та запобігти несанкціонованим діям користувачів.

2.2. Front-end частина системи

Front-end частина є важливою складовою інформаційної системи, оскільки забезпечує взаємодію користувача з функціональністю програмного продукту. Саме через інтерфейс відбувається відображення інформації, введення даних, формування запитів до серверної частини та отримання результатів її обробки.

Рис 2.3. Основні представлення фронтенду

У системі управління ресурсами та персоналом у готельному бізнесі фронтенд виконує роль зв'язувальної ланки між користувачем і сервером, забезпечуючи інтуїтивно зрозумілий доступ до даних про персонал, ресурси, розклади, фінанси та аналітичні звіти.[15]

Основна мета розробки фронтенду полягає у створенні зручного, логічно структурованого й адаптивного користувацького середовища, яке відповідає особливостям діяльності готельного підприємства та сприяє підвищенню ефективності управління.

2.2.1. Концепція користувацького інтерфейсу

Користувацький інтерфейс (UI, від англ. User Interface) - це сукупність візуальних і логічних елементів, за допомогою яких користувач взаємодіє з інформаційною системою. Його структура, зручність і зрозумілість безпосередньо впливають на швидкість виконання операцій, кількість помилок користувачів і загальне сприйняття системи.

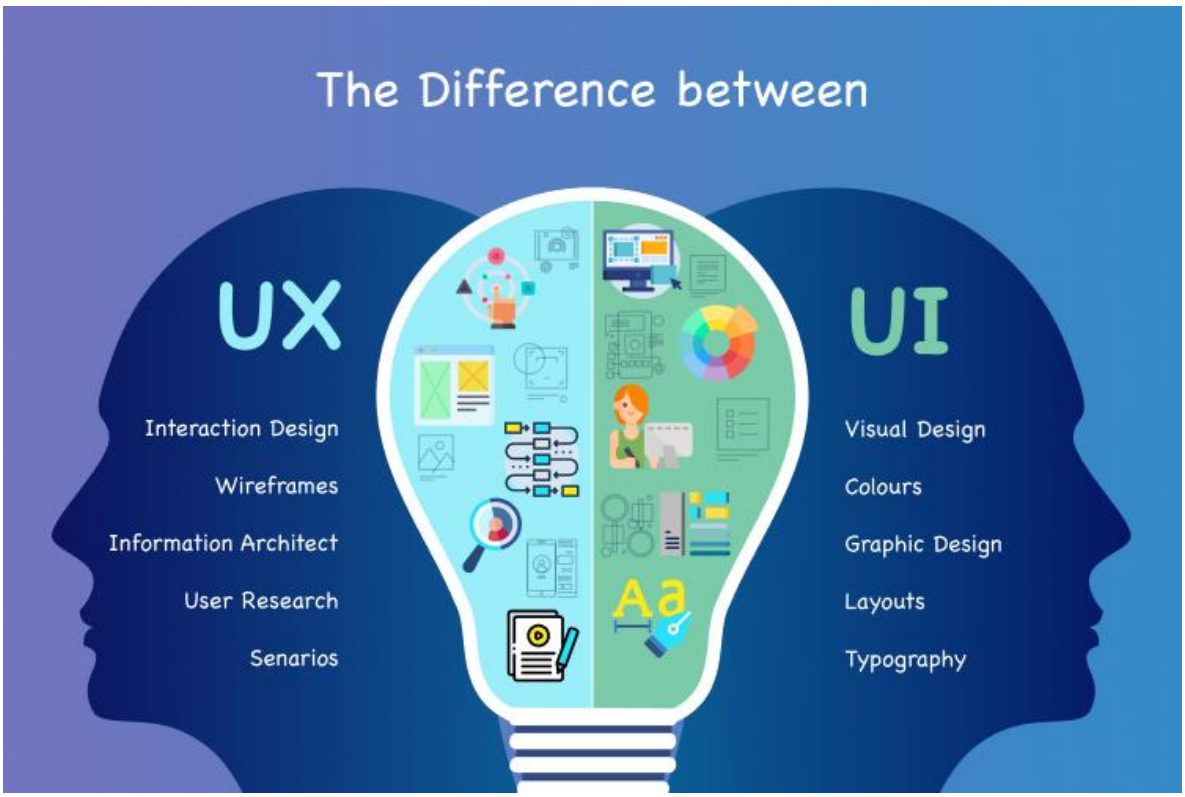

Рис 2.4. Відображення різниці між UX і UI

Під час проєктування інтерфейсу для систем управління персоналом і ресурсами важливо враховувати специфіку готельного бізнесу, де користувачами можуть бути представники різних відділів - адміністратори, менеджери, технічний персонал, бухгалтерія тощо. Це потребує адаптивного підходу до побудови структури вікон, меню, форм і панелей керування. [16]

Основні принципи побудови користувацького інтерфейсу

- Простота та зрозумілість. Елементи інтерфейсу мають бути інтуїтивно зрозумілими навіть для користувачів без спеціальної технічної підготовки.
- Єдність стилю. Використання єдиної кольорової гами, шрифтів і візуальних компонентів сприяє сприйняттю системи як цілісного продукту.
- Ієрархічність і логічність. Структура меню, вкладок і форм повинна відповідати логіці бізнес-процесів готелю: від загального огляду до конкретних операцій.
- Адаптивність. Інтерфейс має коректно відображатися на різних пристроях - комп’ютерах, планшетах, мобільних телефонах. [17]

- Зворотний зв’язок. Система повинна інформувати користувача про результати його дій (повідомлення про успіх, помилки, попередження тощо).
- Безпека та захист даних. В інтерфейсі не повинно відображатися надлишкової службової інформації, а всі дії мають бути підконтрольними відповідно до рівня доступу користувача.

Ефективність користувацького інтерфейсу визначається не лише функціональністю, а й ергономічними характеристиками.[16,18]

Основні вимоги до дизайну включають:

- мінімізацію кількості дій, необхідних для виконання завдання;
- оптимальне розміщення елементів управління;
- використання візуальних підказок (іконок, кольорових маркерів, підсвічування);
- забезпечення контрастності та читабельності тексту;
- можливість швидкого пошуку потрібної інформації.

Залежно від складності системи й кількості користувачів, може застосовуватися:

- Класичний багатосторінковий інтерфейс - підходить для систем із великою кількістю окремих функціональних екранів;
- Односторінковий інтерфейс (SPA, Single Page Application) - дозволяє забезпечити динамічну взаємодію користувача з системою без перезавантаження сторінок;
- Комбінований підхід, коли основні модулі реалізовані як окремі сторінки, а всередині них використовуються динамічні елементи.

Концепція користувацького інтерфейсу у системі управління ресурсами та персоналом у готельному бізнесі ґрунтується на принципах простоти, структурованості, адаптивності та безпеки. Якісно спроєктований інтерфейс сприяє підвищенню ефективності роботи користувачів, зменшенню кількості помилок і покращенню загального рівня автоматизації процесів у готельній сфері.

2.2.2. Структура та логіка взаємодії інтерфейсу з користувачем

Ефективність роботи користувача з інформаційною системою значною мірою визначається логічною структурою інтерфейсу та зручністю навігації між його елементами. У системах управління ресурсами та персоналом, які охоплюють широкий спектр функцій - від обліку кадрів до контролю матеріальних запасів, - важливо забезпечити чітку логічну організацію взаємодії, що мінімізує когнітивне навантаження на користувача. [19,20]

Логіка взаємодії інтерфейсу передбачає визначення послідовності дій, які виконує користувач для досягнення певної мети, а також способів реагування системи на його запити. Цей процес має бути максимально інтуїтивним, передбачуваним і стандартизованим.

Користувацький інтерфейс системи управління ресурсами та персоналом зазвичай має ієрархічну структуру, що складається з кількох рівнів:

- Головний рівень (навігаційний). Містить основні розділи системи, такі як “Персонал”, “Ресурси”, “Графіки”, “Звіти”, “Налаштування”. Цей рівень забезпечує швидкий доступ до ключових модулів.
- Функціональний рівень. Відображає основні дії в межах вибраного модуля - наприклад, перегляд списку працівників, редагування запису або формування звіту.
- Операційний рівень. Включає детальні дії користувача, що супроводжуються формами введення, таблицями або діалоговими вікнами.

Такий підхід дозволяє забезпечити чітку логіку переміщення користувача в системі - від загального до конкретного, а також уникнути перевантаження інтерфейсу надлишковими елементами.

Кожна взаємодія має бути побудована за принципом логічної завершеності, тобто користувач завжди повинен розуміти, на якому етапі процесу він перебуває та які подальші кроки можливі.

Ергономічна організація логіки інтерфейсу передбачає:

- передбачувану реакцію системи на кожну дію користувача;

- наявність підтверджень при виконанні важливих операцій (наприклад, видалення або зміни даних);
- уникнення дублювання функцій;
- чітке візуальне розділення основних і допоміжних елементів;
- використання підказок, спливаючих повідомлень та індикаторів завантаження.

Модель взаємодії " клієнт-сервер "(Рис 2.5)

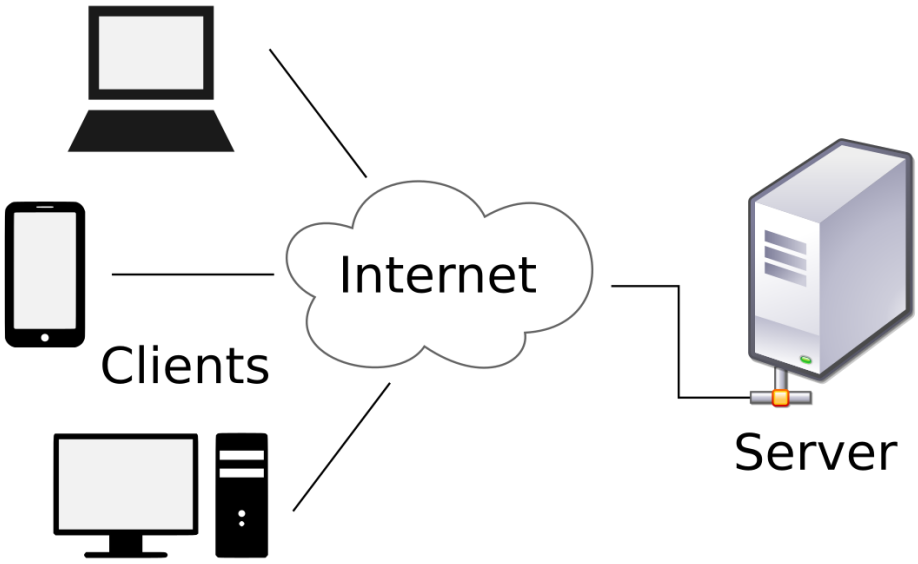

Рис 2.5. модель взаємодії клієнт-сервер

У теоретичному аспекті логіку взаємодії можна описати за допомогою моделі “ клієнт-сервер ”, що включає такі етапи:

- Введення даних користувачем (формування запиту).
- Обробка запиту системою (виконання алгоритму або звернення до серверної частини).
- Отримання та відображення результату у зручному форматі.
- Зворотний зв’язок у вигляді повідомлень про статус виконання дії.

Ця модель підкреслює циклічність процесу взаємодії: користувач постійно отримує підтвердження своїх дій і має можливість продовжити роботу без переривань чи повторних запитів.

Логіка взаємодії інтерфейсу з користувачем є основою ефективності системи управління ресурсами та персоналом. Вона визначає зручність навігації, швидкість виконання операцій і загальний рівень користувацького досвіду. Раціональна структура інтерфейсу сприяє підвищенню продуктивності працівників готельного комплексу та зменшенню кількості помилок під час роботи з інформаційною системою.

2.2.3. Принципи адаптивності та доступності інтерфейсу

У сучасних інформаційних системах одним із ключових аспектів побудови користувацького інтерфейсу є забезпечення його адаптивності та доступності. Ці характеристики визначають, наскільки система може ефективно використовуватися різними категоріями користувачів і на різних типах пристроїв, що особливо важливо для підприємств готельного бізнесу, де персонал працює у динамічних умовах і може користуватися як стаціонарними комп'ютерами, так і мобільними пристроями.[18]

Адаптивність користувацького інтерфейсу - це здатність системи автоматично змінювати свій вигляд, структуру та спосіб подання інформації залежно від характеристик пристрою користувача, роздільної здатності екрана або умов використання.

Основними критеріями адаптивності є:

- гнучке масштабування елементів інтерфейсу (зміна розмірів таблиць, панелей, текстів);
- перебудова макета сторінки залежно від ширини екрана (для мобільних, планшетних і десктопних пристроїв);
- динамічна оптимізація зображень і графіки для швидкого завантаження;
- збереження функціональності незалежно від типу пристрою.

Завдяки адаптивному підходу система управління ресурсами та персоналом може ефективно використовуватися адміністраторами готелю, менеджерами змін і

технічним персоналом як у робочому приміщенні, так і поза ним - наприклад, для оперативного внесення даних про наявність ресурсів або оновлення графіків роботи.

Принципи побудови адаптивного інтерфейсу

- Мобільна сумісність. Усі елементи мають бути доступними на сенсорних екранах, без потреби в точному наведенні курсора.
- Відповідна типографіка. Розмір і контрастність тексту повинні змінюватися автоматично залежно від пристрою.
- Пріоритезація контенту. На менших екранах мають відображатися лише найважливіші елементи, тоді як другорядні можна приховувати або згортати.
- Гнучка сітка (layout). Розташування блоків має змінюватися відповідно до роздільної здатності, зберігаючи логічну послідовність подання інформації.
- Підтримка жестів і швидких дій. Інтерфейс має реагувати на жести (прокручування, натискання, масштабування), що полегшує використання на мобільних пристроях.

Доступність (Accessibility) - це властивість інтерфейсу, яка забезпечує можливість його використання всіма категоріями користувачів, незалежно від їхніх фізичних можливостей, технічних навичок або умов роботи.

Основними принципами побудови доступного інтерфейсу є:

- Сприйнятність (Perceivable). Вся інформація має бути представлена у формах, що можуть бути сприйняті користувачем - візуально, аудіально або текстово.
- Керованість (Operable). Користувач повинен мати змогу виконувати всі дії за допомогою клавіатури, миші або сенсорного екрана.
- Зрозумілість (Understandable). Елементи управління мають бути передбачуваними, а повідомлення - зрозумілими для користувача.
- Надійність (Robust). Інтерфейс має коректно функціонувати з різними браузерами, пристроями й допоміжними технологіями (екранні диктори, збільшувачі тощо).

Розроблення інтерфейсу повинно базуватися на поєднанні візуальної простоти, логічної структури та універсального доступу.

Адаптивність і доступність інтерфейсу є необхідними характеристиками сучасних інформаційних систем, орієнтованих на реальних користувачів. Їх дотримання у системі управління ресурсами та персоналом у готельному бізнесі забезпечує ефективну взаємодію працівників із програмним середовищем, підвищує якість обслуговування гостей і сприяє оптимізації внутрішніх бізнес-процесів.

2.3. База даних

База даних є центральною складовою будь-якої інформаційної системи, що забезпечує збереження, структуроване подання та ефективний доступ до інформації. У системі управління ресурсами та персоналом у готельному бізнесі база даних виступає основним сховищем відомостей про співробітників, клієнтів, матеріальні ресурси, фінансові операції, графіки роботи та інші елементи діяльності готелю.

Раціональна побудова бази даних забезпечує цілісність, достовірність і доступність інформації, а також створює основу для аналітики, звітності та прийняття управлінських рішень.[21,22]

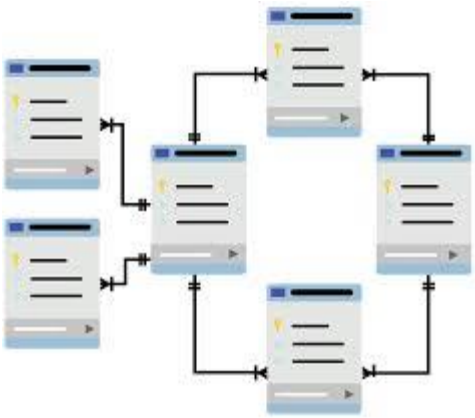

Рис 2.6. Залежностні між сутностями в базі даних

2.3.1. Принципи проєктування бази даних

Проєктування бази даних - це процес визначення структури, взаємозв'язків та обмежень між даними, який забезпечує ефективне зберігання й обробку інформації в межах системи. Головна мета цього процесу - створення моделі, що відображає реальні бізнес-процеси готелю та забезпечує можливість їх подальшої автоматизації. [23]

Основні етапи проєктування бази даних:

- Аналіз предметної області. На цьому етапі визначаються основні сутності (об'єкти) системи, їхні властивості та взаємозв'язки. У контексті готельного бізнесу до таких сутностей належать: персонал, посади, ресурси, постачальники, зміни, бронювання, фінансові операції тощо.
- Побудова концептуальної моделі. Концептуальна модель є абстрактним представленням інформаційної структури системи. Вона відображає логічні взаємозв'язки між сутностями без урахування конкретних технологічних особливостей реалізації. Зазвичай для цього застосовуються ER-діаграми (Entity–Relationship), що показують об'єкти, їх атрибути та типи зв'язків (один-до-одного, один-до-багатьох, багато-до-багатьох)
- Побудова логічної моделі. На цьому рівні відбувається деталізація структури даних з урахуванням обраної системи керування базами даних (СКБД). Уточнюються типи полів, ключі, обмеження цілісності, зовнішні ключі тощо.
- Нормалізація даних. Нормалізація - це процес упорядкування структури таблиць для усунення надлишковості даних і забезпечення цілісності інформації. Вона здійснюється шляхом приведення структури до однієї з нормальних форм (переважно до третьої нормальної форми - 3НФ). Це дозволяє уникнути дублювання даних і забезпечити узгодженість при оновленні записів.
- Фізичне проєктування. На цьому етапі визначаються способи фізичного зберігання даних, індексація, розподіл навантаження між таблицями та оптимізація запитів.

Рис 2.7. Візуалізація типів зв’язків у базі даних

Для систем управління персоналом і ресурсами типовими є такі зв’язки між сутностями:

- “Один-до-багатьох” - наприклад, один відділ може мати багато працівників;
- “Багато-до-багатьох” - один працівник може бути залучений до кількох змін, а одна зміна може включати кількох працівників;
- “Один-до-одного” - наприклад, кожному працівникові може відповідати один особистий профіль у системі.

Такі зв’язки відображають реальні взаємозалежності в структурі готелю та забезпечують точність представлення бізнес-процесів у базі даних.

Проєктування бази даних для системи управління ресурсами та персоналом у готельному бізнесі має ґрунтуватися на принципах цілісності, нормалізації, гнучкості та безпеки. Раціонально побудована структура даних забезпечує стабільність системи, її масштабованість і надійність у процесі експлуатації.

2.3.2. Структура бази даних і організація взаємозв’язків між сутностями

Структура бази даних є логічною моделлю, що відображає взаємопов’язану систему даних, необхідну для реалізації функціональних можливостей інформаційної системи. Вона визначає, яким чином зберігається, групується та обробляється інформація, а також як різні частини системи взаємодіють між собою.

У системі управління ресурсами та персоналом у готельному бізнесі структура бази даних має забезпечувати узгоджене представлення таких напрямів діяльності, як кадровий облік, планування робочих змін, контроль матеріальних ресурсів, фінансовий аналіз і звітність.

База даних будується навколо основних сутностей, що відповідають ключовим елементам бізнес-процесів готелю. До них можуть належати:

- кадрові сутності - описують працівників, їхні посади, навички, відділи;
- ресурсні сутності - містять інформацію про матеріальні запаси, постачальників, категорії ресурсів;
- операційні сутності - відображають процеси бронювання, формування графіків, облік змін і завдань;
- аналітичні сутності - використовуються для формування звітів, статистики та оцінювання ефективності роботи.

Кожна сутність містить набір атрибутів, які описують її властивості, та має унікальний ідентифікатор, що дозволяє зв’язувати дані між різними таблицями. Така організація створює основу для побудови узгодженої логічної моделі бази.

При розробленні структури бази даних важливо дотримуватись таких підходів[23,24]:

- модульність, тобто поділ таблиць за функціональним призначенням (кадрові, ресурсні, аналітичні)
- узгодженість між модулями, щоб дані могли обмінюватися без дублювання;
- розширюваність, яка дає змогу додавати нові таблиці або атрибути без зміни загальної структури;
- орієнтація на зв’язки між процесами, а не лише на зберігання окремих показників.

Добре спроєктована структура бази даних забезпечує:

- єдність інформаційного простору підприємства;
- зменшення обсягу дублювання даних;
- можливість побудови складних аналітичних звітів;

- підвищення швидкодії системи;
- легкість у підтримці й масштабуванні.

Структура бази даних і система взаємозв'язків між сутностями мають бути побудовані з урахуванням логіки бізнес-процесів готелю, забезпечуючи узгодженість, ефективність і гнучкість роботи системи. Вона виступає фундаментом, на якому базується вся функціональність системи управління ресурсами та персоналом.

2.3.3. Механізми збереження, пошуку та оновлення інформації

Ефективне управління даними є ключовим елементом будь-якої інформаційної системи управління ресурсами та персоналом у готельній сфері. Для забезпечення надійного збереження, швидкого пошуку та безпечного оновлення інформації в розроблюваній системі застосовуються наступні механізми:

Збереження даних:

- Використовується реляційна база даних, що забезпечує структуроване зберігання інформації про ресурси, співробітників, бронювання та інші ключові об'єкти системи.
- Для забезпечення цілісності даних застосовуються транзакції SQL, які гарантують виконання операцій «все або нічого».
- Дані резервуються автоматично через налаштовані механізми бекапу, що дозволяє відновити інформацію у разі збою.

Пошук інформації:

- Для оптимізації швидкості пошуку застосовуються індекси на ключові поля таблиць (наприклад, ID співробітника, номер кімнати, дата бронювання).
- Для складних запитів використовуються SQL-запити з фільтрацією, сортуванням та агрегуванням даних, що забезпечує гнучкий і точний пошук інформації.

- У системі передбачена можливість пошуку за декількома критеріями одночасно, що дозволяє менеджерам швидко отримувати необхідну інформацію для прийняття рішень.

Оновлення даних[25]:

- Оновлення даних здійснюється через захищені API-виклики, які контролюють права доступу користувачів та гарантують правильність внесених змін.
- Для запобігання конфліктам при одночасному редагуванні застосовується механізм блокування записів або оптимістичне блокування з контролем версій.
- Всі зміни логуються, що дозволяє відслідковувати історію внесених змін та забезпечує прозорість управління інформацією.

Запропоновані механізми збереження, пошуку та оновлення інформації забезпечують надійну роботу системи, високу швидкість обробки запитів та безпеку даних, що є критично важливим для готельного бізнесу.

2.3.4. Забезпечення цілісності й безпеки даних

Для надійної роботи системи та захисту інформації про ресурси і персонал передбачено комплекс заходів, що забезпечують цілісність і безпеку даних:

Цілісність даних:

- Використання реляційної моделі бази даних з обов’язковим визначенням зовнішніх ключів та обмежень цілісності забезпечує коректність зв’язків між таблицями.
- Транзакції бази даних гарантують виконання складних операцій без втрати даних та неконсистентності.
- Регулярна перевірка даних дозволяє виявляти та виправляти аномалії у базі.

Безпека даних:

- Для контролю доступу користувачів застосовується система ролей та прав, що обмежує доступ до конфіденційної інформації.

- Дані, що зберігаються в базі, шифруються за допомогою сучасних алгоритмів, а передача інформації між фронтендом і бекендом здійснюється через захищені протоколи HTTPS.
- Ведеться журнал подій (логування) для відстеження дій користувачів та виявлення можливих спроб несанкціонованого доступу.

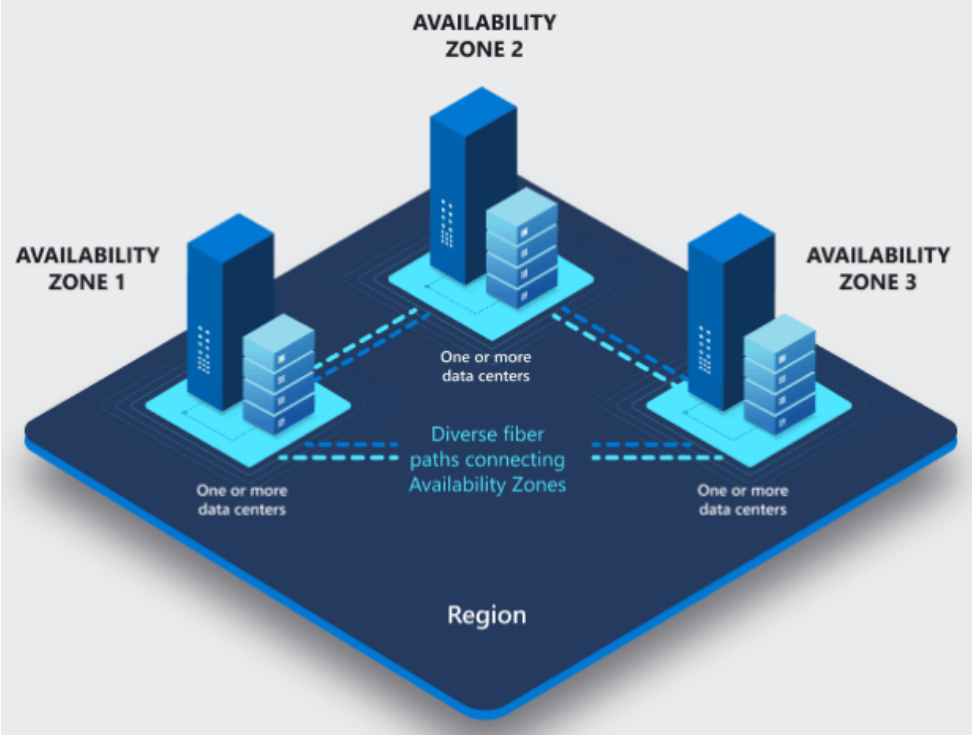

Рис 2.8. Принцип доступних зон для додатковий копій бази даних

Резервне копіювання та відновлення:

- Передбачено регулярне автоматичне резервне копіювання бази даних, що дозволяє швидко відновити систему у разі аварійних ситуацій.
- Розроблені процедури відновлення включають перевірку цілісності резервних копій і тестування відновлення на тестовому середовищі.

Завдяки застосуванню цих заходів забезпечується стабільна робота системи, захист від втрати даних і несанкціонованого доступу, що особливо важливо для готельного бізнесу, де обробляються персональні дані клієнтів і працівників[22,26].

Висновок до розділу 2

У розділі було детально розглянуто процес проєктування та розробки системи управління ресурсами та персоналом у готельному бізнесі, що включає серверну (back-end) та клієнтську (front-end) частини, а також базу даних.

Було обґрунтовано вибір тришарової архітектури для серверної частини, що забезпечує чітке розділення рівнів представлення, бізнес-логіки та зберігання даних. Встановлено, що використання сучасних ORM дозволяє реалізувати масштабовані REST API, організувати модульну структуру системи та забезпечити безпечний обмін інформацією між компонентами.

Особлива увага приділялася бізнес-логіці системи, яка реалізує управління персоналом, ресурсами, планування змін і аналітичні процеси. Визначено основні принципи побудови бізнес-логіки - інкапсуляція, модульність, повторне використання коду, гнучкість та прозорість.

Розглянуто механізми автентифікації, авторизації та безпеки, що гарантують захист конфіденційної інформації, контроль доступу користувачів та захист від несанкціонованих дій. Визначено принципи інтеграції системи з іншими сервісами, що дозволяє формувати єдиний інформаційний простір готелю та підвищує ефективність управлінських процесів.

Щодо фронтенду, акцент було зроблено на побудові зручного, інтуїтивного та адаптивного користувацького інтерфейсу, який забезпечує логічну взаємодію користувача з системою, доступність для різних категорій співробітників та ефективність виконання операцій. Визначено принципи адаптивності, доступності, ефективності та юзабельності інтерфейсу.

Також було обґрунтовано принципи проєктування бази даних, включаючи аналіз предметної області, побудову концептуальної та логічної моделей, нормалізацію даних і забезпечення цілісності, безпеки та масштабованості інформаційного сховища.

Отже, реалізована архітектура та структура системи повинні забезпечити надійну роботу всіх функціональних модулів, ефективну взаємодію користувачів із системою, безпечне зберігання даних і можливість інтеграції з іншими сервісами, що є необхідними умовами для автоматизації управління ресурсами та персоналом у готельному бізнесі.

РОЗДІЛ 3
РОЗРОБКА ВЕБ-ПЛАТФОРМИ ДЛЯ УПРАВЛІННЯ ПЕРСОНАЛОМ

Реалізація програмної системи управління готелем є ключовим етапом, що передбачає практичне втілення спроєктованої архітектури та функціональних вимог. Цей розділ присвячено детальному опису технічних аспектів створення системи, що включає розробку бази даних, серверної та клієнтської частин додатку.

У процесі реалізації використано сучасні технології та інструменти розробки, що забезпечують високу продуктивність, масштабованість та зручність підтримки системи. Розробка велася з використанням методології Code First, що дозволило спочатку визначити моделі даних у коді, а потім автоматично згенерувати структуру бази даних.

Система реалізована як тришарова веб-додаток, де кожен рівень відповідає за певний аспект функціональності: база даних забезпечує надійне зберігання інформації, серверна частина реалізує бізнес-логіку та API для взаємодії, а клієнтська частина надає зручний інтерфейс користувача.

У цьому розділі послідовно розглянуто всі етапи технічної реалізації системи, від створення структури бази даних до розробки користувацького інтерфейсу.

3.1. Реалізація бази даних

Основні таблиці бази даних представляють ключові сутності, що функціонують у межах системи управління готелем. До них відносяться:

Ролі (Role): Встановлюють рівні доступу користувачів до функціоналу системи.

Користувачі (User): Зберігають дані про зареєстрованих працівників готелю, включаючи персональну інформацію та контактні реквізити.

Номери (Room): Містять інформацію про готельні номери, їх характеристики та поточний стан.

Завдання з прибирання (CleaningAssignment): Реєструють призначені завдання для покоївок з деталями виконання.

Графік роботи (WorkSchedule): Зберігають інформацію про робочі графіки покоївок для планування завдань.

Для кожної сутності системи було створено структуру, що визначає будову відповідної таблиці бази даних.

3.1.1 Таблиця Role

Таблиця Role містить інформацію про ролі користувачів у системі. Основні поля включають:

- id (Primary Key): Унікальний ідентифікатор ролі, генерується автоматично.
- name: Назва ролі (ADMIN, MANAGER, HOUSEKEEPER), обмежена 50 символами, має унікальне значення.
- description: Текстовий опис призначення та повноважень ролі, обмежений 200 символами.

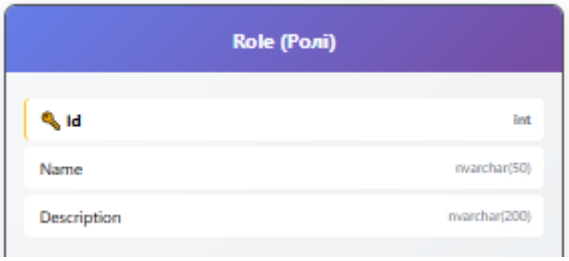

Рис 3.1 - Таблиця Role

Таблиця містить три основні ролі: адміністратор для управління системою та користувачами, менеджер для призначення завдань та контролю

виконання, покоївка для виконання завдань з прибирання номерів. Унікальний індекс на поле name запобігає дублюванню ролей у системі.

3.1.2 Таблиця User

Таблиця User зберігає персональні дані користувачів системи - працівників готелю. Основні поля включають:

- id (Primary Key): Унікальний ідентифікатор користувача, генерується автоматично.
- email: Електронна адреса для входу в систему, обмежена 100 символами, має унікальне значення. Використовується як логін при авторизації.
- password_hash: Хешований пароль для забезпечення безпеки авторизації. Зберігається у зашифрованому вигляді з використанням алгоритмів BCrypt або PBKDF2.
- first_name: Ім'я користувача, обмежене 50 символами, обов'язкове поле.
- last_name: Прізвище користувача, обмежене 50 символами, обов'язкове поле.
- phone: Контактний телефонний номер, обмежений 20 символами для підтримки міжнародного формату. Необов'язкове поле.
- is_active: Булеве поле, що визначає активність облікового запису (за замовчуванням true). Дозволяє тимчасово заблокувати доступ без видалення користувача з системи.
- created_at: Часова мітка створення облікового запису у форматі datetime2. Автоматично встановлюється поточна дата та час у форматі UTC.
- role_id (Foreign Key): Посилання на таблицю Role, визначає роль та права доступу користувача в системі.

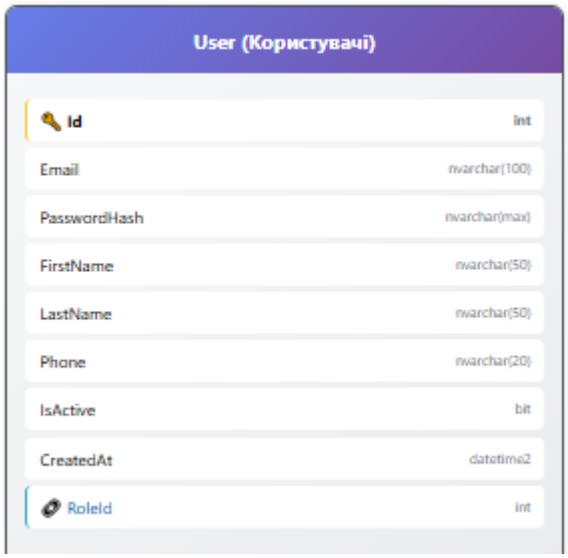

Рис 3.2 – Таблиця User

Для забезпечення цілісності даних встановлено зовнішній ключ на таблицю Role з обмеженням ON DELETE RESTRICT, що запобігає видаленню ролі, яка використовується активними користувачами. Унікальний індекс на поле email забезпечує швидкий пошук при авторизації та унікальність облікових записів.

3.1.3 Таблиця Room

- Таблиця Room містить відомості про готельні номери та їх поточний стан. Основні поля включають:
- id (Primary Key): Унікальний ідентифікатор номера в системі, генерується автоматично.
- room_number: Номер кімнати, обмежений 10 символами, має унікальне значення. Може містити як цифри, так і літери (наприклад, "101", "205A").
- floor: Числове значення поверху, на якому розташований номер. Використовується для оптимізації маршрутів покоївок та фільтрації номерів.
- room_type: Тип номера (перелічуваний тип). Значення: 1 = Standard (стандартний номер), 2 = Deluxe (покращений номер), 3 = Suite

(люкс). Визначає рівень комфорту та обслуговування.

- size: Розмір номера (перелічуваний тип). Значення: 1 = Small (малий, приблизно 20 хвилин прибирання), 2 = Medium (середній, приблизно 30 хвилин), 3 = Large (великий, приблизно 45 хвилин). Використовується для автоматичного розрахунку тривалості прибирання.
- status: Поточний статус номера (перелічуваний тип, за замовчуванням 1). Значення: 1 = Clean (прибраний, готовий до заселення), 2 = Dirty (потребує прибирання), 3 = InProgress (зараз прибирається), 4 = OutOfOrder (не працює, на ремонті). Критично важливе поле для управління операціями готелю.
- priority: Пріоритет прибирання (перелічуваний тип, за замовчуванням 1). Значення: 1 = Normal (звичайний пріоритет), 2 = High (високий пріоритет, скоро заселення гостя), 3 = Urgent (терміново, гість чекає). Впливає на черговість виконання завдань.
- last_cleaned_at: Дата та час останнього завершеного прибирання у форматі datetime2. Необов'язкове поле, автоматично оновлюється при завершенні завдання з прибирання.
- notes: Додаткові примітки про номер, обмежені 500 символами. Може містити інформацію про особливості, несправності або спеціальні вимоги.
- is_active: Булеве поле, що визначає активність номера (за замовчуванням true). Дозволяє тимчасово виключити номер з експлуатації без видалення з бази даних.

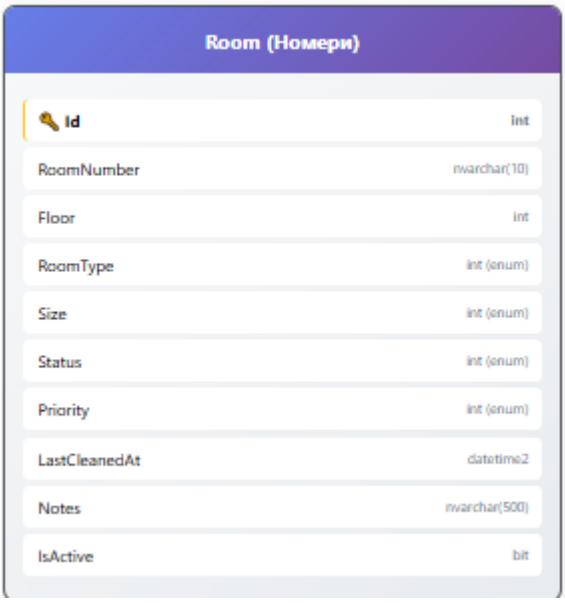

Рис 3.3 – Таблиця Room

.

3.1.4 Таблиця CleaningAssignment

Таблиця CleaningAssignment є центральною таблицею системи, що реєструє всі завдання з прибирання номерів готелю. Основні поля включають:

- id (Primary Key): Унікальний ідентифікатор завдання, генерується автоматично.
- room_id (Foreign Key): Посилання на таблицю Room, визначає номер, який потрібно прибрати.
- housekeeper_id (Foreign Key): Посилання на таблицю User, визначає покоївку, призначену на виконання завдання.
- assigned_by_user_id (Foreign Key): Посилання на таблицю User, вказує на менеджера або адміністратора, який призначив завдання. Необов'язкове поле, може бути NULL при автоматичному призначенні системою.
- assigned_date: Дата призначення завдання у форматі datetime2. Обов'язкове поле, використовується для планування робочого дня та формування денних звітів.
- scheduled_time: Запланований час початку прибирання у

форматі time. Необов'язкове поле, дозволяє точно планувати розклад роботи покоївок протягом дня.

- status: Поточний статус виконання завдання (перелічуваний тип, за замовчуванням 1). Значення: 1 = Pending (очікує виконання), 2 = InProgress (виконується), 3 = Completed (завершено), 4 = Cancelled (скасовано). Критично важливе поле для моніторингу поточного стану робіт.
- estimated_duration_minutes: Очікувана тривалість прибирання в хвилинах. Обов'язкове поле, розраховується автоматично на основі розміру номера з таблиці Room.
- actual_start_time: Фактичний час початку виконання завдання у форматі datetime2. Необов'язкове поле, заповнюється автоматично при зміні статусу на InProgress.
- actual_end_time: Фактичний час завершення завдання у форматі datetime2. Необов'язкове поле, заповнюється при зміні статусу на Completed.
- notes: Текстове поле для додаткових приміток та коментарів, обмежене 1000 символами. Може містити інформацію про виявлені проблеми, особливі вимоги або використані матеріали.
- is_auto_assigned: Булеве поле, що визначає спосіб призначення завдання (за замовчуванням false). Значення true вказує на автоматичне призначення системою, false - на ручне призначення менеджером.
- created_at: Часова мітка створення запису у форматі datetime2. Автоматично встановлюється поточна дата та час у форматі UTC при створенні завдання.

Зовнішні ключі room_id, housekeeper_id та assigned_by_user_id з обмеженням ON DELETE RESTRICT запобігають видаленню пов'язаних записів при наявності активних завдань. Композитний індекс на поля (assigned_date, status) оптимізує запити для отримання завдань за певну дату з фільтрацією за статусом.

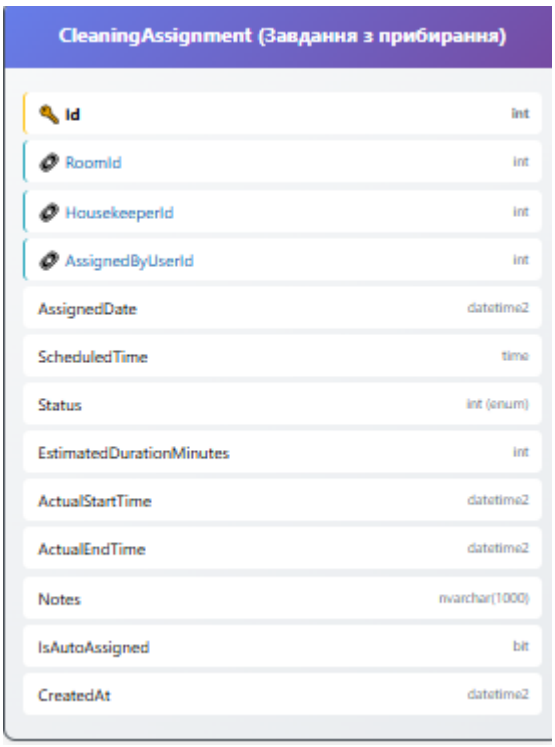

Рис 3.4 – Таблиця CleaningAssignment

3.1.5 Таблиця WorkSchedule

Таблиця WorkSchedule зберігає інформацію про робочі графіки покоївок, що дозволяє системі планувати завдання відповідно до доступності персоналу. Основні поля включають:

- id (Primary Key): Унікальний ідентифікатор запису графіку, генерується автоматично.
- housekeeper_id (Foreign Key): Посилання на таблицю User, визначає покоївку, для якої встановлюється графік роботи.
- day_of_week: Числове значення дня тижня від 0 до 6, де 0 відповідає неділі, 1 - понеділку і так далі до 6 - суботи. Використовується стандартна нумерація .NET Framework для узгодженості з бізнес-логікою.
- shift_start: Час початку робочої зміни у форматі time. Обов'язкове поле, визначає початок робочого дня працівника.
- shift_end: Час завершення робочої зміни у форматі time. Обов'язкове поле, визначає кінець робочого дня працівника.
- is_active: Булеве поле, що визначає активність графіку (за

замовчуванням true). Значення true вказує на поточний діючий графік, false - на архівний або тимчасово неактивний.

- effective_from: Дата початку дії графіку у форматі datetime2. Автоматично встановлюється поточна дата при створенні запису. Дозволяє планувати зміни графіків наперед.
- effective_to: Дата закінчення дії графіку у форматі datetime2. Необов'язкове поле, може бути NULL для постійних графіків. Використовується для тимчасових змін у розкладі, наприклад, під час відпусток чи лікарняних.
- notes: Текстове поле для додаткових приміток, обмежене 200 символами. Може містити інформацію про причини зміни графіку або особливі умови роботи.

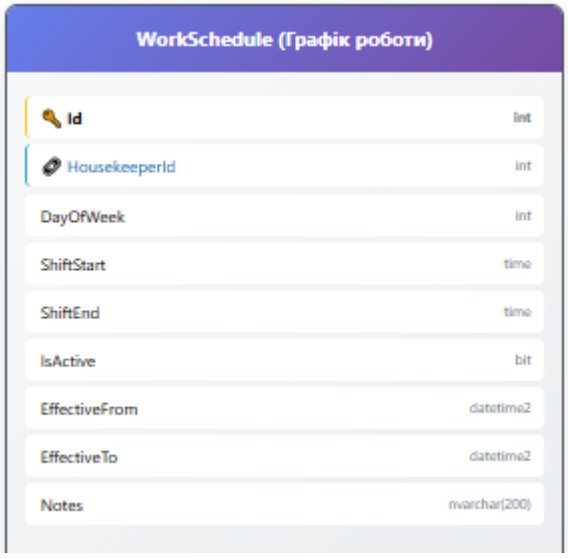

Рис 3.5 – Таблиця WorkSchedule

Зовнішній ключ housekeeper_id з налаштуванням ON DELETE CASCADE гарантує автоматичне видалення графіків при видаленні облікового запису працівника. Композитний індекс на поля (housekeeper_id, day_of_week, is_active) забезпечує швидкий пошук активних графіків роботи для конкретної покоївки у визначений день тижня.

3.1.6 Загальна інформація про базу даних

Для прискорення виконання запитів створено індекси на найбільш використовувані поля:

- Індекс на email користувачів для швидкого пошуку при авторизації
- Індекси на зовнішні ключі (role_id, room_id, housekeeper_id) для оптимізації JOIN операцій
- Індекси на поля is_active та status для фільтрації даних
- Композитний індекс на (assigned_date, status) у таблиці CleaningAssignment для швидкого отримання завдань за певну дату
- Композитний індекс на (housekeeper_id, day_of_week, is_active) у таблиці WorkSchedule для швидкого пошуку графіків роботи
- Унікальні індекси на room_number та email для забезпечення унікальності

Основні зв'язки між таблицями:

- Role → User (1:N): одна роль може бути призначена багатьом користувачам
- User → CleaningAssignment (1:N): один працівник (покоївка) може мати багато завдань
- User → CleaningAssignment (1:N): один менеджер може призначити багато завдань
- Room → CleaningAssignment (1:N): один номер може мати багато завдань з прибирання
- User → WorkSchedule (1:N): один працівник може мати багато записів графіку для різних днів тижня

3.2. Реалізація серверної(бекенд) частини

Серверна частина системи управління готелем відповідає за обробку бізнес-логіки, взаємодію з базою даних та надання RESTful API для клієнтської частини. Backend реалізовано з використанням сучасних технологій та архітектурних підходів, що забезпечують високу продуктивність, масштабованість та безпеку системи. Архітектура серверної частини побудована за принципом поділу відповідальності, де кожен компонент виконує чітко визначені функції. Це забезпечує легкість підтримки коду, можливість тестування окремих модулів та спрощує процес додавання нового функціоналу.

3.2.1. Вибір технологій та фреймворків

Для реалізації серверної частини обрано ASP.NET Core 8.0 як основний фреймворк для розробки веб-API. Це сучасна кросплатформенна технологія від Microsoft, що забезпечує високу продуктивність та оптимізоване використання ресурсів сервера. Фреймворк має вбудовану підтримку dependency injection для інверсії залежностей, що дозволяє створювати слабко зв'язані компоненти системи. ASP.NET Core підтримує розгортання на різних операційних системах, включаючи Windows, Linux та macOS, що забезпечує гнучкість при виборі інфраструктури.

Для роботи з базою даних використовується Entity Framework Core 8.0 як ORM (Object-Relational Mapping). Цей інструмент дозволяє працювати з базою даних через об'єкти C# без необхідності писати SQL запити вручну. Entity Framework Core підтримує підхід Code First, коли структура бази даних визначається через класи моделей у коді, а потім автоматично генерується в базі даних через механізм міграцій. Це спрощує версіонування схеми бази даних та забезпечує синхронізацію між кодом та структурою таблиць.

Основною мовою програмування обрано C# версії 12. Це строго типізована мова з потужними можливостями об'єктно-орієнтованого

програмування. C# забезпечує високу продуктивність, має сучасні можливості мови, такі як pattern matching, async/await для асинхронного програмування, та records для створення immutable типів даних. Відмінна інтеграція з .NET екосистемою та велика спільнота розробників забезпечують наявність готових рішень для більшості типових задач.

Для роботи з базою даних обрано Microsoft SQL Server як реляційну СУБД. Вибір обумовлений повною інтеграцією з Entity Framework Core, високою продуктивністю та надійністю при роботі з транзакціями, а також вбудованими засобами безпеки та можливостями масштабування. SQL Server Express версія є безкоштовною для розробки та невеликих проектів.

Для аутентифікації та авторизації використовується JWT (JSON Web Tokens). Це сучасний стандарт для безпечної передачі інформації між клієнтом та сервером. JWT токени містять зашифровану інформацію про користувача та його права доступу, що дозволяє серверу швидко перевіряти авторизацію без необхідності звертатися до бази даних при кожному запиті.

Для хешування паролів застосовується бібліотека BCrypt.NET. Це криптографічно стійкий алгоритм хешування, що автоматично генерує унікальну сіль (salt) для кожного пароля. BCrypt спеціально розроблений для хешування паролів та має вбудований захист від атак типу brute force завдяки налаштовуваній складності обчислень.

3.2.2. Архітектура серверної частини

Серверна частина організована за багатошаровою архітектурою з чітким поділом відповідальності між компонентами. Така організація коду відповідає принципам SOLID та забезпечує високу підтримуваність системи.

Шар моделей (Models) містить класи, що представляють сутності предметної області. До цього шару належать класи User, Role, Room, CleaningAssignment та WorkSchedule. Ці класи визначають структуру даних

та відображаються на таблиці бази даних через Entity Framework Core. Кожна модель містить властивості з відповідними атрибутами для налаштування типів даних, обмежень та зв'язків. Наприклад, клас User містить атрибути для визначення обов'язкових полів, максимальної довжини рядків та зовнішніх ключів до таблиці Role.

Шар контексту даних реалізовано через клас HotelAppContext, що успадковує DbContext з Entity Framework Core. Цей клас відповідає за визначення DbSet колекцій для кожної сутності, налаштування зв'язків між таблицями та конфігурацію індексів. У методі OnModelCreating визначаються всі зв'язки між таблицями, унікальні індекси та політики каскадного видалення. Наприклад, зв'язок між User та Role налаштовується з політикою ON DELETE RESTRICT, що запобігає видаленню ролі, яка використовується користувачами.

Шар інтерфейсів (Interfaces) визначає контракти для всіх сервісів системи. Кожен інтерфейс описує методи, які повинен реалізувати відповідний сервіс. Наприклад, інтерфейс IUserService визначає методи GetAllAsync, GetByIdAsync, CreateAsync, UpdateAsync та DeleteAsync для роботи з користувачами. Використання інтерфейсів дозволяє застосовувати dependency injection та створювати слабко зв'язані компоненти, які легко тестувати та замінювати.

Шар сервісів (Services) містить бізнес-логіку системи. Кожен сервіс реалізує відповідний інтерфейс та відповідає за обробку даних перед збереженням у базу даних або перед відправкою клієнту. Сервіси інкапсулюють складну логіку, таку як валідація даних, розрахунки, взаємодія з декількома таблицями одночасно. Наприклад, AuthService відповідає за логіку аутентифікації, включаючи перевірку паролів та генерацію JWT токенів, а CleaningDurationPredictionService реалізує алгоритми машинного навчання для прогнозування часу прибирання.

Шар контролерів (Controllers) містить API endpoints, що обробляють HTTP запити від клієнтів. Контролери успадковують клас ControllerBase та

містять методи, анотовані атрибутами HTTP методів (HttpGet, HttpPost, HttpPut, HttpDelete). Кожен контролер використовує відповідні сервіси через dependency injection для виконання операцій. Контролери відповідають тільки за отримання даних з запиту, виклик методів сервісу та формування відповіді з відповідним HTTP статус кодом.

Шар моделей передачі даних (DTOs) складається зі спеціальних класів для обміну даними між клієнтом та сервером. DTOs відрізняються від моделей Entity Framework тим, що містять тільки ті поля, які потрібні для конкретної операції. Використання DTOs дозволяє приховати внутрішню структуру моделей бази даних, контролювати які поля передаються клієнту та застосовувати різні правила валідації для різних операцій. Наприклад, CreateUserDto містить поле Password для створення користувача, а UserDto не містить PasswordHash при відправці даних клієнту.

3.2.3. Реалізація шару сервісів

AuthService реалізує функціональність аутентифікації. Метод LoginAsync перевіряє облікові дані користувача: шукає користувача за email, верифікує пароль через BCrypt.Verify, перевіряє статус IsActive та генерує JWT токен. Метод RegisterAsync створює новий обліковий запис з хешуванням пароля та автоматичним призначенням ролі Housekeeper.

JwtService відповідає за генерацію JWT токенів. Токен підписується алгоритмом HMAC-SHA256 та містить claims з інформацією про користувача: ідентифікатор, email, роль. Токен діє сім днів за замовчуванням.

UserService забезпечує CRUD операції для користувачів. Метод GetAllAsync завантажує всіх користувачів з ролями через Include. Метод CreateAsync валідує унікальність email та хешує пароль. Метод UpdateAsync виконує часткове оновлення, змінюючи тільки передані поля.

RoomService управляє готельними номерами. При створенні нового номера автоматично встановлюється статус Clean та пріоритет Normal.

CleaningAssignmentService керує завданнями з прибирання. Використовує множинні Include для завантаження пов'язаних даних про номер, покоївку та менеджера. При створенні автоматично встановлюється статус Pending.

WorkScheduleService управляє робочими графіками покоївок з автоматичним розрахунком тривалості зміни через ShiftDurationHours.

CleaningDurationPredictionService реалізує машинне навчання для прогнозування часу прибирання. Метод TrainModel аналізує завершені завдання та будує статистичні моделі: середній час по розміру номера, по типу номера та коефіцієнти ефективності покоївок. Метод PredictCleaningDuration враховує базовий прогноз, ефективність покоївки та додаткові фактори (давність прибирання, пріоритет). Результат округлюється до п'яти хвилин та обмежується діапазоном 15-90 хвилин.

3.2.4. Реалізація шару контролерів

AuthController забезпечує endpoints для аутентифікації. POST /api/auth/login приймає email та пароль, повертає JWT токен. POST /api/auth/register створює новий обліковий запис. GET /api/auth/me повертає інформацію про поточного користувача, витягуючи ідентифікатор з токену.

RoomsController керує номерами готелю. GET /api/rooms приймає query параметри для фільтрації за статусом, поверхом та типом. GET /api/rooms/statistics обчислює статистику по номерах з групуванням по поверхах та типах. PUT /api/rooms/{id}/status швидко змінює статус номера з автоматичним оновленням LastCleanedAt при встановленні Clean.

```
using System;
using System.Collections.Generic;
using System.Linq;
using Microsoft.AspNetCore.Mvc;
using HotelApp.DTOs.User;
using HotelApp.Interfaces;

namespace HotelApp.Controllers
{
    [Route("api/[controller]")]
    [ApiController]

    public class UsersController : ControllerBase
    {
        private readonly IUserService _userService;

        0 references
        public UsersController(IUserService userService)
        {
            _userService = userService;
        }

        // GET: api/Users
        [HttpGet]
        0 references
        public async Task<ActionResult<IEnumerable<UserDto>>> GetUsers()
        {
            var users = await _userService.GetAllAsync();
            return Ok(users);
        }
```

Рис 3.6– Приклад реалізації контролера

CleaningAssignmentsController має розумну логіку доступу: Admin бачить всі завдання, Housekeeper тільки свої. POST /api/cleaningassignments створює завдання з автоматичним розрахунком EstimatedDurationMinutes на основі розміру номера. PUT /api/cleaningassignments/{id}/start встановлює ActualStartTime та змінює статус на InProgress. PUT /api/cleaningassignments/{id}/complete завершує завдання, встановлює номер як Clean та оновлює LastCleanedAt.

WorkSchedulesController управляє графіками з фільтрацією по ролі. GET /api/workschedules/my/today повертає інформацію про поточний робочий день з завданнями та статистикою виконання.

MLPredictionController надає API для машинного навчання. POST /api/mlprediction/predict-duration прогнозує час прибирання. POST /api/mlprediction/retrain перетренує модель на актуальних даних.

3.2.5. Аутентифікація та авторизація

Система використовує JWT для аутентифікації. Після успішного логіну сервер генерує токен та відправляє клієнту. Клієнт додає токен в header кожного запиту через схему Bearer. Сервер перевіряє підпис токену без звернення до бази даних.

Токен містить claims: NameIdentifier для ідентифікатора користувача, Email та Role для авторизації. Токен діє сім днів, після чого стає невалідним.

Авторизація реалізована через атрибути. Атрибут Authorize вимагає валідного токену. Атрибут Authorize з параметром Roles обмежує доступ конкретними ролями. Атрибут AllowAnonymous дозволяє публічний доступ.

Паролі хешуються через BCrypt з автоматичною генерацією унікальності.

```
namespace HotelApp.Services
{
    4 references
    public interface IJwtService
    {
        3 references
        string GenerateToken(User user);
    }

    2 references
    public class JwtService : IJwtService
    {
        private readonly IConfiguration _configuration;

        0 references
        public JwtService(IConfiguration configuration)
        {
            _configuration = configuration;
        }
```

Рис 3.7 – Початок реалізації JWT

3.2.6. Реалізація системи машинного навчання для прогнозування

Система включає модуль машинного навчання для прогнозування тривалості прибирання номерів. Ця функціональність автоматично розраховує реалістичний час виконання завдань на основі історичних даних, що покращує точність планування та розподілу навантаження між покоївками.

Модуль реалізовано через сервіс CleaningDurationPredictionService, що використовує статистичні методи для аналізу історичних даних. Обрано підхід на основі статистичного аналізу, що забезпечує швидкість роботи та прозорість прогнозів.

Модель зберігає три типи статистичних даних у вигляді словників: avgDurationBySize містить середній час для кожного розміру номера, avgDurationByType для кожного типу номера, housekeeperEfficiency містить коефіцієнти ефективності покоївок.

Тренування відбувається через метод TrainModel, який викликається автоматично при ініціалізації сервісу. Спочатку завантажуються всі завершені завдання з фактичним часом виконання. Якщо в базі менше п'яти завдань, модель використовує базові значення: 20 хвилин для малих номерів, 30 для середніх, 45 для великих.

При достатній кількості даних завдання групуються по розміру та типу номера з обчисленням середнього часу виконання. Для покоївок розраховується коефіцієнт ефективності як відношення їх середнього часу до глобального середнього. Коефіцієнт менше одиниці означає швидшу роботу, більше одиниці - повільнішу.

Метод PredictCleaningDuration реалізує багатофакторний алгоритм. Базовий прогноз розраховується як середнє арифметичне оцінок за розміром та типом номера. Далі прогноз коригується на ефективність покоївки через множення на її коефіцієнт.

Враховуються додаткові фактори: якщо номер не прибирався більше трьох днів, час збільшується на 10%; для Urgent пріоритету додається 15%. Фінальний прогноз округлюється до п'яти хвилин та обмежується діапазоном

15-90 хвилин.

Прогнози використовуються при створенні завдань та автоматичному розподілі. Контролер MLPredictionController надає три endpoints: POST /api/mlprediction/predict-duration для прогнозування, POST /api/mlprediction/retrain для перетренування моделі адміністратором, GET /api/mlprediction/statistics для аналітики моделі.

```
4 references
public interface ICleaningDurationPredictionService
{
    2 references
    int PredictCleaningDuration(int roomId, int housekeeperId);
    4 references
    void TrainModel();
    2 references
    Dictionary<string, object> GetModelStatistics();
}
```

Рис 3.7 – Інтерфейс сервісу для прогнозування

Метод GetModelStatistics повертає кількість тренувальних даних, середні значення по категоріях та профілі ефективності покоївок з класифікацією на Fast, Average та Slow.

Обраний статистичний підхід не потребує великих обсягів даних, легко інтерпретується, працює швидко та автоматично адаптується до змін через перетренування на нових даних.

3.3. Реалізація клієнтської(фронтенд) частини

Клієнтська частина системи управління готелем забезпечує зручний інтерфейс для взаємодії користувачів з функціоналом системи. Frontend реалізовано з використанням сучасних веб-технологій, що забезпечують швидкість роботи, інтуїтивність інтерфейсу та адаптивність під різні типи

пристроїв.

3.3.1. Реалізація системи машинного навчання для прогнозування

Для реалізації клієнтської частини обрано Angular 21 як основний фреймворк для розробки single-page application. Angular забезпечує повноцінну екосистему для розробки складних веб-додатків з вбудованими інструментами для роутингу, управління станом, валідації форм та HTTP комунікації.

Для стилізації використовується SCSS (Sassy CSS) - розширення CSS з підтримкою змінних, вкладених селекторів та міксинів. Це дозволяє створювати модульні та підтримувані стилі з можливістю повторного використання коду.

TypeScript обрано як основну мову програмування для фронтенду. Строга типізація допомагає виявляти помилки на етапі розробки, покращує читабельність коду та забезпечує кращу підтримку в IDE через автодоповнення та перевірку типів.

Для HTTP комунікації з backend використовується вбудований HttpClient з Angular. Для роботи з реактивними даними застосовуються RxJS Observable, що дозволяє елегантно обробляти асинхронні операції та потоки даних.

3.3.2. Архітектура клієнтської частини

Клієнтська частина організована за модульною архітектурою з чітким поділом за функціональністю. Структура проекту розділена на кілька основних директорій.

Директорія core містить базові сервіси, моделі, guards та interceptors, що використовуються по всьому додатку. Це ядро додатку, яке завантажується один раз при старті.

Директорія features містить функціональні модулі, розділені за ролями користувачів: auth для аутентифікації, admin для адміністративних функцій, housekeeper для функцій покоївок, profile для управління профілем.

Директорія shared містить компоненти, які використовуються в різних частинах додатку, такі як навігаційна панель та бічне меню.

У директорії core/models визначено TypeScript інтерфейси для всіх сутностей системи. Файл auth.model.ts містить інтерфейси LoginRequest, RegisterRequest, AuthResponse з інформацією про токен та користувача. Файл user.model.ts визначає інтерфейс User з полями id, email, firstName, lastName, role. Файл room.model.ts описує структуру Room з параметрами номера. Файл assignment.model.ts містить інтерфейс CleaningAssignment для завдань з прибирання. Файл schedule.model.ts визначає WorkSchedule для графіків роботи.

Директорія core/services містить сервіси для взаємодії з backend API. Кожен сервіс інкапсулює логіку HTTP запитів для конкретної функціональної області.

Сервіс auth.ts відповідає за аутентифікацію користувачів. Метод login відправляє POST запит на /api/auth/login з credentials та зберігає отриманий JWT токен в localStorage. Метод register реєструє нового користувача. Метод logout очищує токен з localStorage та перенаправляє на сторінку логіну. Метод getCurrentUser повертає інформацію про поточного користувача з токену або через запит /api/auth/me.

Сервіс room.service.ts управляє номерами готелю. Метод getRooms отримує список номерів з можливістю фільтрації через query параметри. Метод getRoomById завантажує детальну інформацію про конкретний номер. Метод updateRoomStatus змінює статус номера через PUT /api/rooms/{id}/status. Метод createRoom додає новий номер в систему.

Сервіс assignment.service.ts керує завданнями з прибирання. Метод getAssignments отримує список завдань з фільтрацією. Метод getMyAssignments завантажує завдання поточної покоївки. Метод

startAssignment розпочинає виконання завдання через PUT /api/cleaningassignments/{id}/start. Метод completeAssignment завершує завдання з можливістю додати примітки.

Сервіс schedule.service.ts працює з графіками роботи. Метод getMySchedule отримує тижневий графік поточної покоївки. Метод getTodaySchedule завантажує інформацію про сьогоднішній робочий день з завданнями. Метод getAvailableHousekeepers повертає список доступних покоївок для призначення завдань.

Сервіс ml-prediction.service.ts взаємодіє з ML модулем. Метод predictDuration прогнозує час прибирання для комбінації номера та покоївки. Метод retrainModel запускає перетренування моделі. Метод getStatistics отримує аналітику моделі машинного навчання.

Директорія core/guards містить захисників маршрутів. Guard auth-guard.ts перевіряє чи користувач авторизований перед доступом до захищених сторінок. Якщо токен відсутній або невалідний, користувач перенаправляється на сторінку логіну. Guard role-guard.ts перевіряє чи користувач має необхідну роль для доступу до конкретного маршруту. Наприклад, адміністративні сторінки доступні тільки з роллю Admin.

Директорія core/interceptors містить HTTP перехоплювачі. Interceptor auth-interceptor.ts автоматично додає JWT токен в Authorization header кожного HTTP запиту через схему Bearer. Це звільняє від необхідності вручну додавати токен в кожному сервісі. Також interceptor перехоплює помилки 401 Unauthorized та автоматично перенаправляє користувача на сторінку логіну при закінченні терміну дії токену.

3.3.3. Реалізація системи машинного навчання для прогнозування

Модуль аутентифікації розташований в директорії features/auth та містить два компоненти: login для входу та register для реєстрації.

Компонент login реалізує форму входу в систему. Файл login.html містить template з полями для email та password. Використовується reactive forms з валідацією обов'язковості полів та формату email. При натисканні кнопки входу викликається метод onSubmit.

Файл login.ts містить логіку компонента. Створюється FormGroup з FormControl для email та password. Метод onSubmit перевіряє валідність форми, викликає метод login з AuthService та обробляє результат. При успішному логіні токен зберігається в localStorage та користувач перенаправляється на відповідну сторінку в залежності від ролі: Admin на /admin/dashboard, Housekeeper на /housekeeper/dashboard. При помилці відображається повідомлення про невірні облікові дані.

Файл login.scss містить стилі для форми входу з центруванням на сторінці, тінями для карточки форми та стилізацією полів вводу.

Компонент register реалізує форму реєстрації нового користувача. Template містить поля для email, password, firstName, lastName та phone. Додано валідацію мінімальної довжини пароля та підтвердження пароля через кастомний validator.

Логіка компонента створює розширену форму з додатковим полем confirmPassword. Кастомний validator passwordMatchValidator перевіряє чи співпадають password та confirmPassword. При успішній реєстрації користувач автоматично авторизується та перенаправляється на dashboard покоївки.

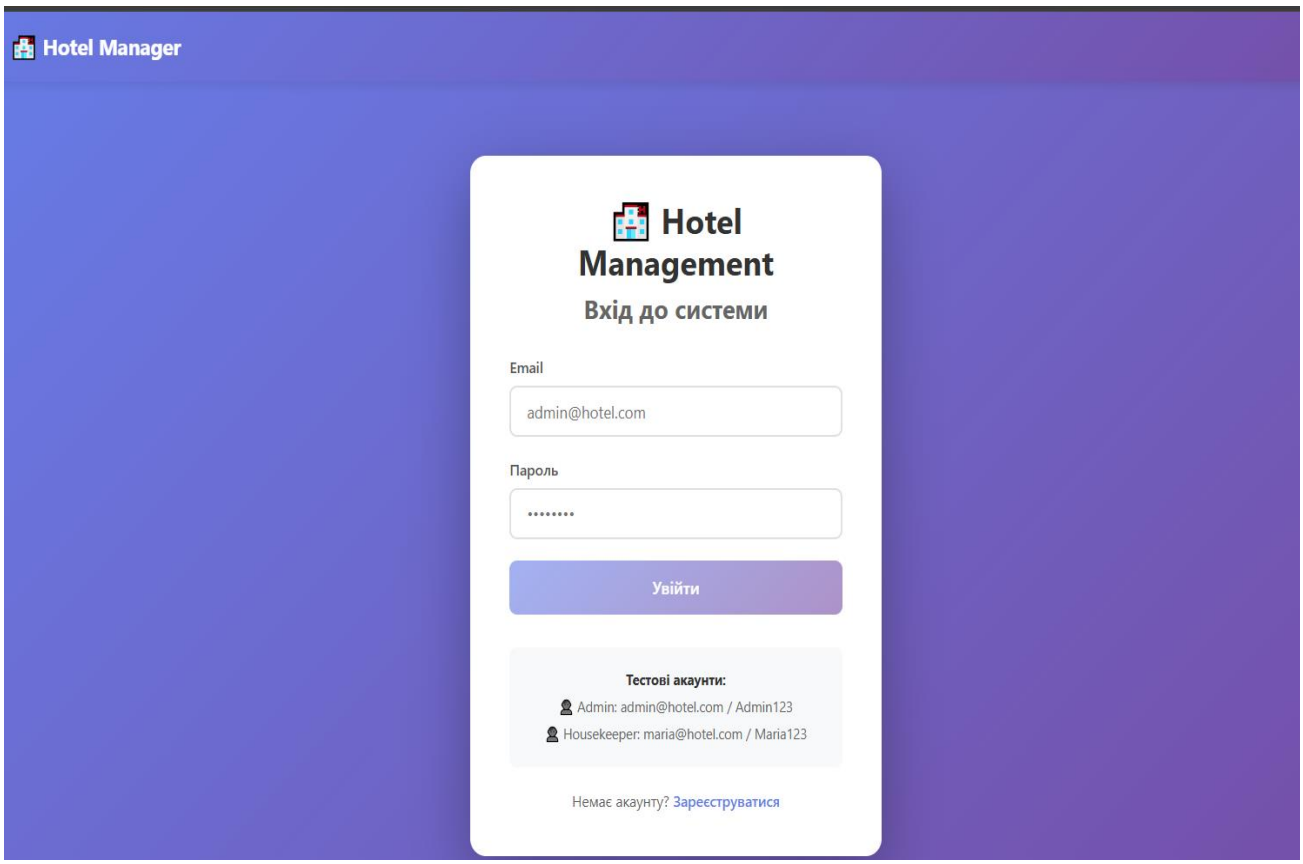

Рис 3.8 – Інтерфейс сторінки для логіну

3.3.4. Модуль адміністратора

Модуль адміністратора розташований в features/admin та містить шість компонентів для управління системою.

Компонент admin/dashboard відображає загальну статистику системи. Template розділений на секції з карточками статистики: загальна кількість номерів, чисті/брудні номери, активні покоївки, завдання на сьогодні.

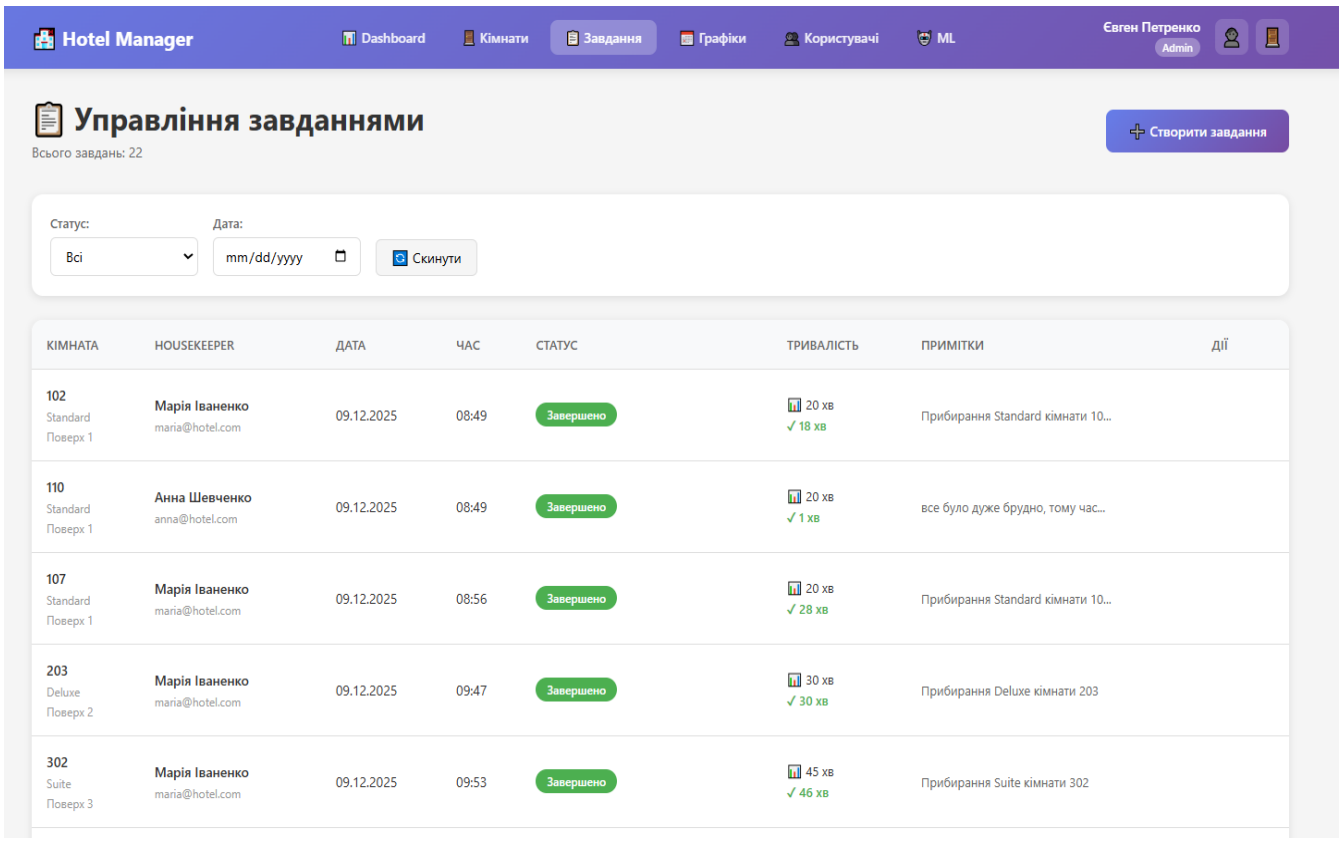

Рис 3.9 – Інтерфейс сторінки admin/dashboard

Використовуються графіки для візуалізації розподілу номерів по поверхах та статусах.

Логіка компонента завантажує статистику через RoomService.getStatistics при ініціалізації. Дані зберігаються в властивостях компонента та відображаються через data binding в template. Статистика автоматично оновлюється кожні тридцять секунд через setInterval для актуальності інформації.

Компонент rooms-management забезпечує повне управління номерами готелю. Template містить таблицю з списком номерів, фільтри по статусу, поверху та типу, кнопки для зміни статусу та редагування параметрів номера.

Логіка завантажує список номерів через RoomService.getRooms. Реалізовано методи для швидкої зміни статусу через updateRoomStatus, відкриття модального вікна для редагування номера, додавання нового номера. Таблиця підтримує сортування по колонках та пагінацію для великої кількості номерів.

Компонент assignments-management керує завданнями з прибирання. Template відображає список всіх завдань з інформацією про номер, покоївку, статус виконання. Реалізовано форму для створення нового завдання з вибором номера, покоївки, дати та часу.

Логіка компонента завантажує завдання через AssignmentService.getAssignments з можливістю фільтрації по статусу та даті. При створенні завдання викликається ML сервіс для прогнозування тривалості через MLPredictionService.predictDuration. Прогнозований час відображається в формі для інформування менеджера. Метод createAssignment відправляє дані на сервер та оновлює список після успішного створення.

Компонент users-list відображає список всіх користувачів системи. Template містить таблицю з інформацією про користувачів: email, ім'я, роль, статус активності. Реалізовано можливість деактивації користувачів та зміни їх ролей.

Компонент schedules-management управляє робочими графіками покоївок. Template відображає графіки у вигляді календарної сітки з днями тижня та часом зміни для кожної покоївки. Форма дозволяє створити новий графік з вибором покоївки, дня тижня, часу початку та закінчення зміни.

Логіка завантажує графіки через ScheduleService та відображає у зручному форматі. Метод createSchedule валідує що час початку менший за час закінчення та що графік на цей день ще не існує для даної покоївки.

Компонент ml-statistics відображає аналітику моделі машинного навчання. Template містить карточки з ключовими метриками: кількість тренувальних даних, середній час по категоріях, таблицю з профілями ефективності покоївок. Кнопка для перетренування моделі доступна адміністратору.

Логіка завантажує статистику через MLPredictionService.getStatistics. Метод retrainModel викликає перетренування та оновлює відображення після завершення. Профілі покоївок відображаються з кольоровим кодуванням: зелений для Fast, жовтий для Average, червоний для Slow.

3.3.5. Модуль покоївки

Модуль покоївки розташований в features/housekeeper та містить три компоненти для роботи персоналу.

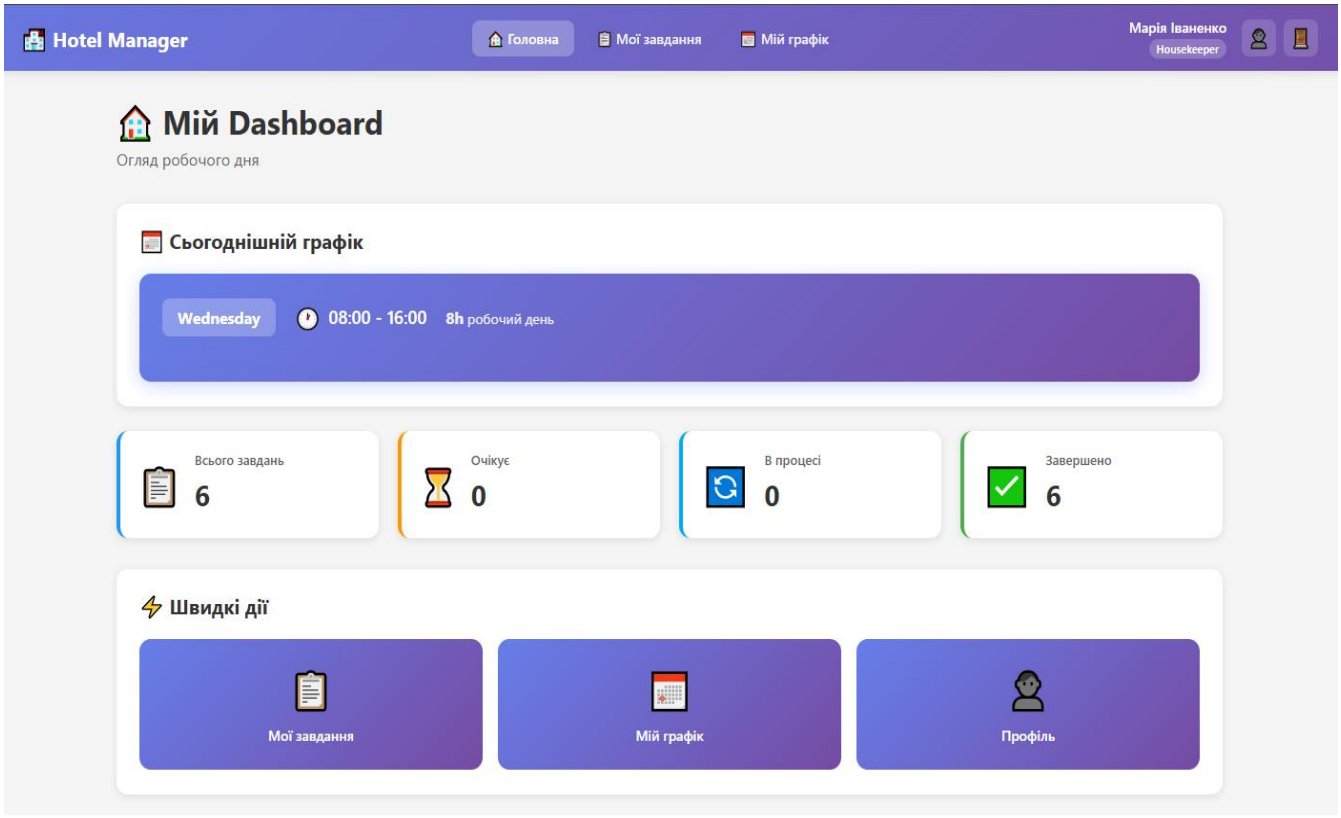

Рис 3.10 – Інтерфейс головної сторінки для покоївки

Компонент housekeeper/dashboard відображає персональну інформацію про робочий день. Template містит секції: сьогоднішній графік роботи, список завдань на сьогодні, статистику виконання. Завдання відображаються у вигляді карточок з кнопками для початку та завершення.

Логіка завантажує дані через ScheduleService.getTodaySchedule, що повертає інформацію про зміну та завдання одним запитом. При натисканні "Почати" викликається AssignmentService.startAssignment, що змінює статус на InProgress та встановлює час початку. Кнопка "Завершити" відкриває модальне вікно для додавання фінальних приміток та викликає completeAssignment.

Компонент my-assignments відображає повний список завдань покоївки. Template містить фільтри по даті та статусу, карточки завдань з детальною інформацією. Кожна карточка показує номер кімнати, запланований час, очікувану тривалість, поточний статус.

Логіка завантажує завдання через AssignmentService.getMyAssignments. Реалізовано методи для зміни статусу завдань аналогічно dashboard. Завдання автоматично групуються по датах для зручності перегляду.

Компонент my-schedule відображає тижневий графік роботи покоївки. Template показує календарну сітку з робочими днями, часом початку та закінчення зміни, тривалістю зміни. Виділяється поточний день тижня.

Логіка завантажує графік через ScheduleService.getMySchedule. Відображається загальна статистика: кількість робочих днів на тиждень, загальна кількість годин. Графік представлений у зручному візуальному форматі з кольоровим кодуванням робочих та вихідних днів.

Висновки до розділу 3

У третьому розділі детально описано процес практичної реалізації системи управління готелем, включаючи розробку бази даних, серверної та клієнтської частин додатку.

Реалізація бази даних виконана з використанням Microsoft SQL Server та Entity Framework Core з підходом Code First. Створено п'ять таблиць, що повністю покривають функціональні вимоги системи: Role для управління ролями користувачів, User для зберігання інформації про працівників, Room для готельних номерів, CleaningAssignment для завдань з прибирання та WorkSchedule для робочих графіків. Встановлено зв'язки типу один-до-багатьох між таблицями з відповідними політиками каскадного видалення. Реалізовано індексацію ключових полів для оптимізації продуктивності запитів та забезпечення цілісності даних через обмеження на рівні бази даних.

Серверна частина розроблена на базі ASP.NET Core 8.0 з використанням багатошарової архітектури. Реалізовано шар моделей для представлення сутностей, шар сервісів для бізнес-логіки, шар контролерів для API endpoints та шар DTOs для обміну даними. Впроваджено систему аутентифікації через JWT токени з хешуванням паролів за допомогою BCrypt. Розроблено RESTful API з повним набором endpoints для управління користувачами, номерами, завданнями та графіками роботи. Реалізовано розумну систему авторизації з розподілом прав доступу між адміністраторами та покоївками. Особливістю системи є інтегрований модуль машинного навчання для прогнозування тривалості прибирання на основі статистичного аналізу історичних даних, що враховує розмір та тип номера, ефективність конкретної покоївки та додаткові фактори.

Клієнтська частина реалізована з використанням Angular 21 з TypeScript та SCSS для стилізації. Застосовано модульну архітектуру з чітким поділом за функціональністю: модуль аутентифікації для входу та реєстрації, модуль адміністратора з шістьма компонентами для повного управління

системою, модуль покоївки з персональним dashboard та управлінням завданнями. Реалізовано систему guards для захисту маршрутів та interceptors для автоматичного додавання токену в HTTP запити. Інтерфейс побудований з урахуванням ролей користувачів, де кожна роль має доступ тільки до відповідних функцій. Впроваджено reactive forms з валідацією даних, автоматичне оновлення інформації та зручну візуалізацію статистики.

Розроблена система повністю реалізує всі функціональні вимоги, визначені в другому розділі. Забезпечено надійність збереження даних, безпеку доступу через токенову аутентифікацію, високу продуктивність завдяки оптимізації запитів та інтуїтивний користувацький інтерфейс. Використання сучасних технологій та архітектурних підходів гарантує масштабованість системи та легкість подальшої підтримки і розширення функціоналу.

РОЗДІЛ 4
РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

4.1 Опис ідеї проекту

Проєкт «Система управління ресурсами та персоналом у готельному бізнесі» розробляється для підвищення ефективності операційної діяльності готельних комплексів, оптимізації роботи персоналу та забезпечення прозорого контролю над використанням матеріальних ресурсів. Система покликана автоматизувати ключові процеси готелю, зменшити операційні витрати, підвищити швидкість обслуговування гостей та забезпечити якісну взаємодію між адміністрацією та співробітниками.

Відмінними якостями проєкту є:

- централізоване управління персоналом, включаючи планування змін, облік робочого часу та розподіл завдань;
- контроль номерного фонду та ресурсів, що забезпечує актуальну інформацію про стан номерів, запаси та їх використання;
- автоматизація комунікації між відділами, що зменшує кількість помилок та прискорює виконання операцій;
- можливість розширення функціоналу, інтеграції з PMS-системами, бухгалтерією та сторонніми сервісами;

Унікальність технології полягає в інтеграції HRM та RMS-функцій у єдину веб-платформу: система поєднує керування персоналом, станом номерного фонду та складським обліком, забезпечуючи комплексний підхід до управління готельною інфраструктурою. Опис ідеї стартап-проєкту наведено в таблиці 4.1.

Таблиця 4.1.

Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Система управління ресурсами та персоналом у готельному бізнесі	1. Управління персоналом	Забезпечує автоматизацію планування змін, розподілу завдань і контролю робочого часу, що підвищує ефективність працівників.
	2. Контроль ресурсів та номерного фонду	Дозволяє відстежувати стан номерів у реальному часі, керувати запасами та оптимізувати витрати матеріалів.
	3. Розподіл завдань	Дозволяє призначати завдання співробітникам, контролювати їх виконання та підвищувати оперативність роботи персоналу.

Основними аналогами системи є готельні PMS-рішення, що мають функціонал управління персоналом. Аналіз техніко-економічних переваг запропонованої ідеї порівняно з конкурентами наведено у таблиці 4.2.

Таблиця 4.2

Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	Cloudbed	Opera PMS	1С: Готель			
1	Вартість використання	Безкоштовний	Платний	Платний	Платний			+
2	Простота освоєння	Низька	Висока	Низька	Невисока			+
3	Орієнтація на український ринок	Так	Ні	Ні	Так		+	
4	Фокус на HRM-функціоналі без зайвих модулів	Так	Ні	Ні	Частково			+
5	Локалізація та україномовний інтерфейс	Так	Так	Так	Так		+	

4.2 Технологічний аудит ідеї проекту

Розробка веб-системи управління персоналом у готельному бізнесі потребує аналізу наявних технологічних рішень, оцінки готових інструментів та визначення потенційних проблем, що можуть виникати у разі відсутності необхідних засобів.

Розглянемо технологічний аудит у таблиці 4.3.

Таблиця 4.3

Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Реалізація модуля планування робочих змін	Відкриті бібліотеки для календарів, UI-компонентів, алгоритми формування графіків	Наявна	Засоби є загальнодоступними
2	Реалізація модуля обліку робочого часу	REST-API, бібліотеки для роботи з датою/часом, механізми логування подій	Наявна	Засоби є загальнодоступними
3	Реалізація модуля розподілу завдань між персоналом	Відкриті бібліотеки, приклади реалізації task-менеджменту	Необхідно розробити	Засоби є загальнодоступними

Таблиця 4.3(продовження)

Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
4	Реалізація ролей та прав доступу	Готові бібліотеки аутентифікації та авторизації (JWT, RBAC)	Наявна	Засоби є загальнодоступними
5	Реалізація модуля відстеження статусу виконання завдань	Компоненти UI, бібліотеки для побудови списків та станів	Необхідно розробити	Засоби є загальнодоступними
6	Програмний продукт, що включає всі модулі системи	Програмні засоби для веб-розробки, фреймворки та технології, описані вище	Необхідно розробити	Засоби є загальнодоступними
Обрані технології реалізації ідеї проекту: C#, Javascript, Angular , HTML				

Можна зробити висновок, що реалізація цієї ідеї можливо і включає в себе розробку нових модулів та використання відкритих бібліотек.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть пере- шкодити реалізації проекту, дозволяє спланувати напрями розвитку про- екту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

У таблиці 4.4 відображено попередню характеристику потенційного ринку стартап-проекту.

Таблиця 4.4

Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	5
2	Загальний обсяг продаж, грн/ум.од	-
3	Динаміка ринку (якісна оцінка)	Зростаюча
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	25-30%

Можна зробити висновок, що ринок є привабливим для входження за попереднім оцінюванням.

У таблиці 4.5 наводяться визначені потенційні клієнти, а також їх характеристика.

Таблиця 4.5

Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба в автоматизації управління персоналом готелю	Малі та середні готелі	Обмежений бюджет, потреба у простих та інтуїтивних рішеннях, базовий функціонал	Простота використання. Доступна вартість. Можливість швидкого впровадження. Базовий набір функцій (графіки роботи, облік персоналу).
2	Потреба в оптимізації операційних процесів	Адміністратори та менеджери готелів	Фокус на зручності планування, контролі виконання завдань та оперативному отриманні інформації	Швидкодія системи. Зручний інтерфейс для мобільних пристроїв. Можливість отримання оперативних сповіщень. Інструменти для планування та контролю.

Для виявлення потенційних загроз можна спиратися на наступний список питань: які нові компанії на ринку несуть потенційну загрозу, фактори політики і економіки, які можуть погіршити ситуацію, які найбільш привабливі умови і продукти пропонують клієнтам конкуренти.

У таблиці 4.6 наведені фактори загроз, що перешкоджають ринковому впровадженню.

Таблиця 4.6

Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Поява нових сильних конкурентів	Проблеми в просуванні системи на ринку через наявність відомих міжнародних рішень	Запуск системи в Україні та за її межами, вдала та продумана рекламна кампанія
2	Погіршення економічної ситуації в готельній галузі	Зменшення кількості потенційних клієнтів через скорочення витрат готелів на автоматизацію	Впровадження гнучкої цінової політики, пропозиція базових пакетів за доступною ціною, можливість поетапного впровадження системи
3	Низька готовність готелів до цифрової трансформації	Опір персоналу впровадженню нових технологій, звичка до традиційних методів управління	Організація навчальних програм та демонстраційних презентацій, надання безкоштовного пробного періоду, створення детальної документації та відеоінструкцій
4	Проблеми з інтеграцією з існуючими системами	Готелі можуть використовувати різні PMS та бухгалтерські системи, що ускладнює інтеграцію	Розробка API для інтеграції з популярними готельними системами, забезпечення можливості імпорту/експорту даних у стандартних форматах

У таблиці 4.7 наведені фактори можливостей, що сприяють ринковому впровадженню.

Таблиця 4.7

Фактори можливостей

	Фактор	Зміст можливості	Можлива реакція компанії
1	Великий потенціал розвитку ринку онлайн-послуг	Яскраво виражена тенденція росту інтересу до вибірки даних з медіа-ресурсів, використовуючи інтернет платформи.	Відповідність усім вимогам користувача, безвідмовна робота сервісу та відповідність оголошеній якості.
2	Вихід на світовий ринок	Відпрацьований та якісний продукт буде затребуваний і в інших країнах, що надає додаткові інвестиції	Адаптація системи під ринки з найбільшою кількістю користувачів.
3	Продаж бізнесу на початковій стадії росту	В теперішній час IT сервіси, орієнтовані на ринок надання послуг, у випадку розвитку та позитивній динаміці викликають інтерес у крупних інвесторів	Зібрання відгуків та побажань від користувачів та відповідно до отриманої інформації, покращення можливостей системи та її удосконалення
4	Орієнтованість сервісу на користувача	При наявності великої кількості споживачів (попит), бізнес буде зацікавлений в переході в систему	Збільшити попит користувачів на сервіс, шляхом впровадження рекламної кампанії та відгуків користувачів
5	Додаткові джерела монетизації	Отримання додаткових прибутків	Використання альтернативних методів стандартної підписки

У таблиці 4.8 наведено загальні риси конкуренції на ринку.

Таблиця 4.8

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції:- олігополістична	Конкуренція між обмеженою кількістю постачальників систем управління для готельного бізнесу	Диференціація продукту через адаптацію під специфіку українського ринку, конкурентне ціноутворення, якісна технічна підтримка
2. За рівнем конкурентної боротьби:- національний з елементами міжнародного	Основний фокус на український готельний ринок, але з можливістю конкуренції	Локалізація інтерфейсу, адаптація під законодавство України, врахування особливостей роботи вітчизняних готелів
3. За галузевою ознакою:- внутрішньогалузева	Всі продукти є програмними рішеннями для готельної індустрії	Унікальність продукту за рахунок інтеграції управління персоналом та ресурсами в єдиній системі
4. Конкуренція за видами товарів:- товарно-видова	Конкуренція між різними типами систем управління	Пропозиція інтегрованого рішення, що поєднує управління персоналом і ресурсами

Таблиця 4.8(продовження)

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
5. За характером конкурентних переваг:- нецінова	Полягає у задоволенні специфічних потреб готельного бізнесу, зручності використання, функціональності системи	Розширення функціональних можливостей (планування графіків, контроль доступу, звітність), вдосконалення існуючих модулів, оперативне реагування на запити клієнтів
6. За інтенсивністю:- немарочна	Вибір системи базується на функціональності та ціні, а не на впізнаваності бренду	Фокус на якість продукту, функціональність та обслуговування; формування репутації через відгуки клієнтів

У таблиці 4.9 наведено більш детальні риси конкуренції на ринку за М. Портером.

Таблиця 4.9

Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари замінники
	Oracle Hospitality OPERA PMS, MICROS-Fidelio Suite8 PMS, Mews PMS, 1С: Готель, Cloudbeds PMS	Інші стартапи, що розробляють системи управління для готельного бізнесу; розширення функціоналу існуючих PMS систем	Постачальники хмарної інфраструктури (AWS, Azure, Google Cloud),	Малі, середні та великі готелі, готельні мережі, санаторії, хостели	Окремі спеціалізовані системи (HRM-системи, складський облік), таблиці Excel, паперовий документообіг
Висновки:	На світовому ринку є достатня кількість конкурентів проте вони	Можливість виходу на ринок нових конкурентів є високою через розвиток хмарних технологій Потенційні конкуренти не	Постачальники диктують умови вартості хмарних сервісів та ліцензій на технології,	Клієнти диктують вимоги до функціональності, зручності, швидкості впровадження, вартості	Поки конкуренти не створили продукт, що поєднує управління персоналом та ресурсами в єдиній інтегрованій системі за

Таблиця 4.9(продовження)

Аналіз конкуренції в галузі за М. Портером

	Прямі конкурен ти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари замінники
	орієнтов анперева жно на великі готельні мережі та міжнародний ринок.	мають продукту з універсальним функціональним рішенням, що поєднує управління персоналом та ресурсами	але конкуренція між ними дозволяє обирати оптимальні за ціною та якістю рішення.	та якості технічної підтримки. Важлива локалізація під українське законодавство	доступною ціною. Excel та паперовий документообіг залишаються поширеними серед малих готелів через низьку вартість, але є неефективними

На основі вище наведених таблиць виведено фактори конкурентоспроможності, та наведено у таблиці 4.10.

Таблиця 4.10

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	ІІнтеграція HRM та RMS	Об'єднання управління персоналом та ресурсами в одній системі підвищує ефективність та зменшує витрати
2	Локалізація під український ринок	Підтримка української мови, адаптація під місцеве законодавство та бухгалтерські стандарти

Таблиця 4.10(продовження)

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
3	Доступна ціна	Конкурентна вартість для малих і середніх готелів порівняно з міжнародними аналогами
4	Простота використання	Інтуїтивний інтерфейс та швидке впровадження знижують витрати на навчання персоналу
5	API для інтеграцій	Можливість підключення до PMS, бухгалтерських систем та інших готельних сервісів
6	Масштабованість	Підходить для готелів різних розмірів - від малих до середніх мереж

Порівняльний аналіз сильних та слабких сторін системи збору та агрегації даних для аналізу вакансій відображено у таблиці 4.11.

Таблиця 4.11

Порівняльний аналіз сильних та слабких сторін модуля аналітичного опрацювання текстової інформації

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з системою збору та агрегації даних для аналізу вакансій						
			–3	–2	–1	0	+1	+2	+3
1	Інтеграція HRM та RMS	20						+	
2	Локалізація під український ринок	18							+
3	Доступна ціна	17						+	
4	Простота використання	15					+		
5	API для інтеграцій	16			+				
6	Масштабованість	14				+			

Враховуючи вище наведені таблиці зробимо висновок ринкового аналізу, та сформуємо його у вигляді SWOT- аналізу (таблиця 4.12).

Таблиця 4.12

SWOT- аналіз стартап-проекту

Сильні сторони:	Слабкі сторони:
Інтеграція HRM та RMS в єдиній системі. Локалізація під український ринок (мова, законодавство). Доступна цінова політика для малих і середніх готелів. Простота впровадження та використання. Можливість інтеграції через API. Адаптивність до різних типів готелів	Інтеграція HRM та RMS в єдиній системі. Локалізація під український ринок (мова, законодавство). Доступна цінова політика для малих і середніх готелів. Простота впровадження та використання. Можливість інтеграції через API. Адаптивність до різних типів готелів
Можливості:	Загрози:
Зростаючий попит на автоматизацію в готельному бізнесі України. Можливість виходу на ринки сусідніх країн (Польща, країни Балтії). Потенціал додаткової монетизації через преміум-функції та модулі. Partnerships з готельними асоціаціями та навчальними закладами. Орієнтованість на потреби малого та середнього бізнесу, який недостатньо охоплений конкурентами	Поява нових сильних конкурентів на українському ринку. Погіршення економічної ситуації може знизити платоспроможність готелів. Низька готовність малих готелів до впровадження цифрових рішень. Можлива недовіра до нового продукту без репутації на ринку. Технічні обмеження хмарної інфраструктури при масштабуванні

На основі SWOT-аналізу визначено альтернативи ринкового впровадження стартап-проекту (таблиця 4.13).

Таблиця 4.13

Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Розробка MVP системи з базовим функціоналомта тестування на пілотних готелях	Висока	6 місяців
2	Пошук перших клієнтів серед малих і середніх готелів через пряме звернення та демонстрацію системи	Середня	7 місяця
3	Розширення функціоналу на основі відгуків користувачів	Висока	8 місяців

Обрано першу альтернативу - розробку MVP з базовим функціоналом.

4.4 Розроблення ринкової стратегії проекту

Для визначення ринкової стратегії потрібно спочатку визначити стратегії охоплення ринку (таблиця 4.14).

Таблиця 4.14

Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Малі готелі	Середня	Високий	Низька на момент запуску	Висока складність залучення
2	Державні установи, що працюють з інформацією	Висока	Вище середнього	Середня на момент запуску	Середня складність залучення
3	Приватні установи, що працюють з інформацією	Середня	Середній	Висока на момент запуску	Низька складність залучення
Які цільові групи обрано: на початковому етапі обрано малі та середні готелі, оскільки вони мають високий попит на доступні рішення, нижчу конкуренцію та простіший вхід на ринок.					

При виборі одного сегменту використаємо стратегію диференційованого маркетингу та визначимо її базову стратегію розвитку (таблиця 4.15).

Таблиця 4.15

Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні і позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Розробка MVP системи з базовим функціоналом	Концентрований маркетинг	Інтеграція HRM та RMS в єдиній системі. Локалізація під український ринок. Доступна ціна. Простота впровадження	Стратегія диференціації

У таблиці 4.16 описано стратегію базової конкурентної поведінки.

Таблиця 4.16

Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопроходцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Частково	Змішаний підхід – залучення нових клієнтів та перехоплення існуючих клієнтів	Компанія буде аналізувати кращі практики конкурентів, але фокусуватиметься на унікальній інтеграції HRM+RMS та локалізації	Стратегія "челенджера" – заняття конкурентної ніші на українському ринку через диференціацію

Враховуючи описані вище стратегій визначимо стратегію позиціонування продукту (таблиця 4.17).

Таблиця 4.17

Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Інтеграція HRM та RMS в одній системі	Стратегія диференціації	Об'єднання управління персоналом та ресурсами	Комплексність, зручність
2	Адаптація під український ринок	Стратегія диференціації	Локалізація (мова, законодавство, стандарти)	"Своє" рішення для України
3	Доступна ціна та простота	Стратегія диференціації	Конкурентна вартість, швидке впровадження	Доступність, ціна/якість

4.5 Розроблення маркетингової програми стартап-проекту

Для формування маркетингової концепції продукту, спершу треба визначити результати аналізу конкурентоспроможності продукту (таблиця 4.18).

Таблиця 4.18

Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
1	Інтеграція управління персоналом та ресурсами	Єдина система для планування змін, обліку робочого часу та контролю ресурсів	Об'єднання HRM+RMS, на відміну від конкурентів, які пропонують окремі модулі
2	Локалізація під український ринок	Повна підтримка української мови, адаптація під місцеве законодавство та бухгалтерські стандарти	Повна локалізація, чого немає у міжнародних аналогів
3	Доступна вартість	Нижча ціна порівняно з міжнародними рішеннями при збереженні ключового функціоналу	Конкурентна ціна для малих і середніх готелів, недоступна у дорогих аналогів

Для просування продукту застосовується трирівнева модель маркетингу:

- Споживачі - малі та середні готелі України
- Інструменти маркетингу - пряме звернення до готелів, демонстраційні презентації, співпраця з готельними асоціаціями
- Політика комунікацій - акцент на локалізацію, доступність та комплексність рішення

Таблиця 4.19.

Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Веб-система для управління ресурсами та персоналом у готельному бізнесі		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	Швидкодія та надійність Ефективність Користувацький інтерфейс	Н м Н м Нм	Тх Тл Е
	Якість: функціональна веб-система з модулями управління персоналом		
	Пакування: власний сайт		
	Марка: HotelFlow		
III. Товар із підкріпленням	Програмний продукт		
	Програмний продукт		
За рахунок чого потенційний товар буде захищено від копіювання: захищений програмний код із захистом авторських прав, а також використання ліцензії			

Вартість на рекламу у товарі визначається типом реклами кількістю її відображення.. У таблиці 4.20 наведено ціни відповідно до кількості реклами.

Таблиця 4.20.

Визначення меж встановлення ціни

№ п/п	Рівень цін на товари замінники, тис. грн/програмний продукт	Рівень цін на товари аналоги, тис. грн/програмний продукт	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	Excel, паперовий документообіг (безкоштовно);	15 000 грн/рік	Малі готелі: низький дохід	Нижня межа: 3 000 грн/рік Верхня межа: 15 000 грн/рік

У таблиці 4.21 наведена оптимальна система збуту.

Таблиця 4.21.

Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Пошук рішення через інтернет, запит демо-версії	Інформування про продукт, демонстрація функціоналу,	Прямий канал (0 рівнів) - безпосередній продаж від розробника до готелю через веб-сайт або особисті зустрічі	Власний веб-сайт з можливістю реєстрації

У таблиці 4.22 наведено концепцію маркетингових комунікацій.

Таблиця 4.22.

Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціону-вання	Завдання рекламного повідомле - ння	Концепція рекламного звернення
1	Власники малих готелів шукають рішення через інтернет, звертаються за рекомендаціями	Професійні веб-сайти готельної тематики, галузеві форуми, готельні асоціації, тематичні конференції та виставки	Інтеграція HRM+RMS; локалізація під український ринок; доступна ціна; простота впровадження	Показати власникам готелів переваги комплексної автоматизації управління персоналом та ресурсами	"Керуйте готелем ефективно: персонал, ресурси та графіки в одній системі. Створено для українських готелів"

Висновки до розділу 4

У четвертому розділі дипломної роботи була визначена актуальність стартап-проекту "Система управління ресурсами та персоналом у готельному бізнесі", визначені основні конкуренти та доведена технологічна здійсненність проекту.

Був проведений аналіз ринкових можливостей запуску стартап-проекту та розроблена ринкова стратегія проекту.

На даний момент проект має конкурентів (Oracle OPERA, Mews, Cloudbeds, 1С: Готель), але вони не мають всі ті функції та можливості, що є у даній системі. Інтеграція управління персоналом та ресурсами в єдиній системі, локалізація під український ринок та доступна ціна є основними характеристиками конкурентоспроможності продукту.

Імплементація продукту є доцільною, адже доведена його технологічна здійсненність та потенційна рентабельність.

Отже, можна зробити висновок, що проект буде успішним у разі реалізації MVP з основними функціями (управління персоналом, планування змін, облік ресурсів) та запуску його на ринок з фокусом на малі та середні готелі України.

ВИСНОВКИ

У магістерській роботі проведено всебічне дослідження проблематики автоматизації операційних процесів у готельному бізнесі, зокрема управління матеріальними ресурсами та роботою персоналу, що є критичними складовими ефективної діяльності сучасних готельних комплексів. На основі проведеного аналізу, проєктування та програмної реалізації було досягнуто поставлену мету — розроблено інтегровану веб-систему управління ресурсами та персоналом, яка забезпечує підвищення продуктивності та оптимізацію ключових бізнес-процесів.

У першому розділі було здійснено глибокий огляд існуючих PMS, HRM та RMS рішень, таких як Oracle OPERA PMS, MICROS-Fidelio, Mews, Cloudbeds та 1С: Готель. Порівняльний аналіз показав, що хоча зазначені продукти забезпечують широкий набір функцій для управління готелем, вони залишаються або занадто дорогими, або недостатньо адаптованими до потреб українських малих та середніх готелів. Зокрема, існуючі системи часто мають обмежену гнучкість у налаштуванні HRM-процесів та недостатньо виражений модуль управління ресурсами. Це підтвердило актуальність розробки спеціалізованого рішення, орієнтованого на локальний ринок.

У другому розділі було спроєктовано архітектуру інтегрованої системи, засновану на тришаровій моделі, яка забезпечує масштабованість, надійність і легкість подальшого доопрацювання. Було визначено структуру бази даних, розроблено моделі сутностей, окреслено логіку обробки даних та принципи взаємодії модулів. Особлива увага приділена механізмам безпеки — автентифікації, авторизації, шифруванню даних і захисту від несанкціонованого доступу. Такий підхід гарантує як конфіденційність, так і цілісність даних системи.

У третьому розділі виконано програмну реалізацію системи. Розроблено повноцінний back-end на основі .NET Web Core API та реалізовано зв’язок із SQLServer-базою даних. Створено модулі для управління

персоналом, планування змін, обліку номерів, розподілу завдань та відстеження виконання робіт. Також реалізовано front-end частину з використанням Angular, що забезпечило інтуїтивний інтерфейс для адміністратора та працівників готелю. Крім того, інтегровано модуль машинного навчання, який виконує прогнозування завантаженості готелю та потреби у персоналі — це значно підвищує точність планування і дозволяє зменшити витрати.

У четвертому розділі сформовано стартап-проєкт для комерціалізації розробленої системи. Було проведено технологічний аудит, SWOT-аналіз, аналіз ринкових можливостей та конкурентного середовища. На основі цього розроблено стратегію виходу на ринок, визначено цільові сегменти, потенційні канали просування, моделі монетизації й методи масштабування. Сформовано маркетингову програму та окреслено перспективи розширення функціоналу системи.

Результатом виконання роботи є створення інтегрованої веб-системи, розробленої на основі технологій Angular та .NET, яка забезпечує повноцінну автоматизацію управління персоналом, включно з формуванням робочих графіків, розподілом завдань та обліком робочого часу. Система дає змогу оптимізувати процеси контролю номерного фонду та стану приміщень у режимі реального часу, що підвищує оперативність виконання внутрішніх операцій. Крім того, вона забезпечує точний облік матеріальних ресурсів, сприяючи зменшенню операційних витрат і підвищенню ефективності логістичних процесів. Інтеграція прогнозних моделей дозволяє використовувати аналітику для удосконалення управлінських рішень, а налагоджена взаємодія між адміністрацією та персоналом робить комунікацію прозорою та контрольованою. Усе це комплексно підвищує якість сервісу, скорочує кількість помилок і мінімізує затримки у роботі готельного комплексу.

Практичне значення роботи полягає у створенні масштабованого, адаптивного та економічно доступного рішення для українського готельного

бізнесу, яке може бути впроваджене як у невеликих готелях, так і у великих мережах.

Отже, проведена магістерська робота підтвердила актуальність теми та продемонструвала можливість створення інноваційної системи, здатної суттєво підвищити ефективність управління ресурсами та персоналом у готельній індустрії. Розроблений програмний продукт має потенціал подальшого розвитку та комерційного впровадження, а його функціональні можливості можуть бути розширені відповідно до потреб ринку та специфіки конкретних готельних комплексів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Albu C. E., Vintilă D. Digitalization and Automation in Hospitality Industry: Challenges and Opportunities. Sustainability. 2024. Vol. 16, No. 17. Article 7279. DOI: https://doi.org/10.3390/su16177279.

2. The Role of Automation in the Hospitality Industry. HospitalityNet. 2024. URL: https://www.hospitalitynet.org/news/4129121.html.

3. What Role Will Automation Play in Future Hotel Operations. International Hospitality Institute. 2023. URL: https://www.internationalhospitalityinstitute.com/articles/what-role-will-automation-play-in-future-hotel-operations.

4. Automation and Artificial Intelligence in the Hospitality Industry. International Journal of Tourism and Hotel Business Management. 2023. Vol. 7, No. 1. P. 37–45. URL: https://www.tourismjournal.net/article/view/140/7-1-37.

5. Oracle OPERA Property Services. Oracle. URL: https://www.oracle.com/hospitality/products/opera-property-services/.

6. Fidelio Suite 8. Micros. URL: https://www.micros.rs/en/fidelio-suite-8.html.

7. Integration: 1C Hotel. YieldPlanet. URL: https://www.yieldplanet.com/integration/1c-hotel/.

8. Mews Property Management System. Mews. URL: https://www.mews.com/.

9. Cloudbeds Property Management System. Cloudbeds. URL: https://www.cloudbeds.com/property-management-system/.

10. Fielding R. Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine, 2000.

11. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2003.

12. Sommerville I. Software Engineering. 10th ed. Pearson Education, 2015.

13. Stallings W. Cryptography and Network Security: Principles and Practice. Pearson, 2017.

15. MDN Web Docs. Guides on client-side web development, SPA principles and modern front-end best practices. URL: https://developer.mozilla.org/.

16. Nielsen Norman Group. Articles on UX research and interface design principles. URL: https://www.nngroup.com/.

17. Material Design / Bootstrap documentation. Practical recommendations on responsive design and component-based UI development. URLs: https://m3.material.io/

, https://getbootstrap.com/.

18. World Wide Web Consortium (W3C). Web Content Accessibility Guidelines (WCAG). URL: https://www.w3.org/WAI/standards-guidelines/wcag/.

19. ISO 9241-11:2018. Ergonomics of human-system interaction - Usability: Definitions and concepts. International Organization for Standardization, 2018.

20. Preece J., Rogers Y., Sharp H. Interaction Design: Beyond Human-Computer Interaction. Wiley, 2019.

21. Elmasri R., Navathe S. Fundamentals of Database Systems. 8th ed. Pearson, 2016.

22. Coronel C., Morris S. Database Systems: Design, Implementation, & Management. 13th ed. Cengage, 2020.

23. Date C. J. An Introduction to Database Systems. 8th ed. Pearson, 2003

24. Hoffer J. A., Ramesh V., Topi H. Modern Database Management. 13th ed. Pearson, 2020

25. Silberschatz A., Korth H., Sudarshan S. Database System Concepts. 7th ed. McGraw-Hill, 2020.

ДОДАТОК 1

Частина програмного коду

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Authorization;
using HotelApp.DTOs.User;
using HotelApp.Services;
using System.Security.Claims;

namespace HotelApp.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class AuthController : ControllerBase
    {
        private readonly IAuthService _authService;
        private readonly ILogger<AuthController> _logger;

        public AuthController(IAuthService authService, ILogger<AuthController> logger)
        {
            _authService = authService;
            _logger = logger;
        }

[HttpPost("login")]
        [AllowAnonymous]
        public async Task<ActionResult<AuthResponseDto>> Login([FromBody] LoginRequestDto loginDto)
        {
            try
            {
                var response = await _authService.LoginAsync(loginDto);
                return Ok(response);
            }
            catch (UnauthorizedAccessException ex)
            {
                return Unauthorized(new { message = ex.Message });
            }
```

```
            catch (Exception ex)
            {
                _logger.LogError(ex, "Login error");
                return StatusCode(500, new { message = "Помилка сервера" });
            }
        }

        [HttpPost("register")]
        [AllowAnonymous]
        public async Task<ActionResult<AuthResponseDto>> Register([FromBody]
RegisterRequestDto registerDto)
        {
            try
            {
                var response = await _authService.RegisterAsync(registerDto);
                return Ok(response);
            }
            catch (InvalidOperationException ex)
            {
                return BadRequest(new { message = ex.Message });
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "Registration error");
                return StatusCode(500, new { message = "Помилка сервера" });
            }
        }

        [HttpGet("me")]
        [Authorize] // Потрібна авторизація
        public async Task<ActionResult<UserDto>> GetMyProfile()
        {
            try
            {
                    var userIdClaim = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;
```

```
            if (string.IsNullOrEmpty(userIdClaim) || !int.TryParse(userIdClaim, out int
userId))
                return Unauthorized(new { message = "Невалідний токен" });
            var user = await _authService.GetUserByIdAsync(userId);
            return Ok(user);
        }
        catch (KeyNotFoundException ex)
        {
            return NotFound(new { message = ex.Message });
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "Get profile error");
            return StatusCode(500, new { message = "Помилка сервера" });
        }
    }

    [HttpPut("me")]
    [Authorize] // Потрібна авторизація
    public async Task<ActionResult<UserDto>> UpdateMyProfile([FromBody]
UpdateUserDto updateDto)
    {
        try
        {
            var userIdClaim = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;

            if (string.IsNullOrEmpty(userIdClaim) || !int.TryParse(userIdClaim, out int
userId))
                return Unauthorized(new { message = "Невалідний токен" });

            var user = await _authService.UpdateProfileAsync(userId, updateDto);
            return Ok(user);
        }
        catch (KeyNotFoundException ex)
```

```
            {
                return NotFound(new { message = ex.Message });
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "Update profile error");
                return StatusCode(500, new { message = "Помилка сервера" });
            }
        }

        [HttpGet("users/{id}")]
        [Authorize(Roles = "Admin")]
        public async Task<ActionResult<UserDto>> GetUserById(int id)
        {
            try
            {
                var user = await _authService.GetUserByIdAsync(id);
                return Ok(user);
            }
            catch (KeyNotFoundException ex)
            {
                return NotFound(new { message = ex.Message });
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "Get user error");
                return StatusCode(500, new { message = "Помилка сервера" });
            }
        }
    }
}

namespace HotelApp.Controllers
{
    [Route("api/[controller]")]
```

```
[ApiController]
[Authorize]
public class MLPredictionController : ControllerBase
{
    private readonly ICleaningDurationPredictionService _predictionService;

    public MLPredictionController(ICleaningDurationPredictionService predictionService)
    {
        _predictionService = predictionService;
    }

    // POST: api/MLPrediction/predict-duration
    [HttpPost("predict-duration")]
    public ActionResult<PredictionResponse> PredictDuration([FromBody] PredictionRequest request)
    {
        try
        {
            var predictedMinutes = _predictionService.PredictCleaningDuration(
                request.RoomId,
                request.HousekeeperId
            );

            return Ok(new PredictionResponse
            {
                PredictedDurationMinutes = predictedMinutes,
                ConfidenceLevel = "High", // Можна додати розрахунок довіри
                Message = $"Прогнозований час: {predictedMinutes} хв"
            });
```