МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Андрій ФОРСЮК_	_____ Сергій ГРИБКОВ__
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
« 02 » _____червня_____2025р.	« 02 » _____червня_____2025р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки»
(код та назва спеціальності)
освітньо-професійної програми Комп'ютерні науки
на тему: «Розроблення інформаційної системи для управління замовленнями та поселенням клієнтів у готелі»

Виконав: здобувач _4_ курсу, групи __КН-4-3___

_____Чубенко Дмитро Олександрович_____ _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник____Мазуренко Ольга Олександрівна_____ _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач_____
(підпис)

Київ - 2025р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра інформаційних технологій, штучного інтелекту та кібербезпеки

Освітній ступінь бакалавр
Спеціальність 122 комп'ютерні науки
(код і назва)
Освітньо-професійна програма комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри інформаційних технологій, штучного інтелекту і кібербезпеки
Сергій ГРИБКОВ
«28» квітня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Чубенка Дмитра Олександровича
(прізвище, ім'я, по батькові)

1.Тема роботи «Розроблення інформаційної системи для управління замовленнями та поселенням клієнтів у готелі»
керівник роботи старший викладач, кандидат технічних наук Мазуренко Ольга Олександрівна
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «28» квітня 2025 р. № 254-кс
2.Строк подання здобувачем роботи: 30.05.2025 р.
3.Вихідні дані до роботи: Нормативно-правова база надання готельних послуг. Результати аналізу аналогічних рішень на ринку готельних інформаційних систем. Технічні та програмні ресурси підприємства, зокрема наявна інфраструктура, програмне забезпечення, обладнання.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
1) Загальна характеристика готелю, 2) Огляд сучасного стану готельного бізнесу та аналіз існуючих інформаційних систем, 3) Опис бізнес-процесів, 4) Виявлені проблеми, 5) Огляд існуючих рішень, 6). Економічний ефект від впровадження нової системи, 7)Формування технічного завдання на створення інформаційної системи, 8)Проєктування бази даних: розробка логічної та фізичної моделі для зберігання даних про клієнтів, бронювання, номери 9)Реалізація основних функціональних модулів системи, зокрема обробки бронювання, реєстрації поселення та виселення клієнтів.
5. Перелік графічного матеріалу: Організаційно-функціональна схема управління готелем. Схема архітектури. Скріншоти інтерфейсу користувача

реалізованої інформаційної системи. Діаграма бази даних.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	ктн, Мазуренко О.О.	28.04.2025	30.04.2025
2	ктн, Мазуренко О.О.	01.05.2025	03.05.2025
3	ктн, Мазуренко О.О.	04.05.2025	15.04.2025

7. Дата видачі завдання: 28 квітня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми, обґрунтування актуальності, аналіз предметної області, постановка завдань	28.04.2025-30.04.2025	
2	Розробка технічного завдання	01.05.2025-03.05.2025	
3	Проєктування структури бази даних та модулів системи, реалізація функціональних модулів, розробка системи	04.05.2025-15.04.2025	
4	Написання пояснювальної записки та оформлення графічного матеріалу	16.05.2025-30.05.2025	

Здобувач _____ Чубенко Д.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Мазуренко О.О.
(підпис) (прізвище та ініціали)

Анотація

Кваліфікаційна робота присвячена розробці інформаційної системи для автоматизації процесів управління замовленнями та поселенням клієнтів у готельному комплексі. Основною метою дослідження є створення ефективного, та зручного у використанні веб-додатку, який забезпечує виконання ключових функцій: облік клієнтів, здійснення бронювання, реєстрація заселень і виселень, а також адміністрування користувачів із різними рівнями доступу.

У процесі реалізації було застосовано сучасні інструменти програмної інженерії, зокрема технологічний стек, що включає мову програмування C#, фреймворк ASP.NET Core MVC, систему управління базами даних Microsoft SQL Server, а також ORM-технологію Entity Framework Core. Архітектурна модель побудована за принципами багаторівневого розділення відповідальностей, що підвищує модульність та спрощує супровід системи.

Інформаційна система реалізована у вигляді веб-додатку з адаптивним інтерфейсом, який забезпечує інтерактивну взаємодію з користувачем і підтримує виконання основних операцій у межах готельного сервісу. Розроблена система дозволяє оптимізувати внутрішні процеси, підвищити ефективність роботи персоналу та зменшити кількість помилок, пов’язаних з людським фактором.

Обсяг роботи становить 62 сторінок основного тексту, містить 4 таблиць, 31 ілюстрацій. Список використаних джерел налічує 30 найменувань.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, АВТОМАТИЗАЦІЯ ПРОЦЕСІВ, ГОТЕЛЬНИЙ СЕРВІС, БРОНЮВАННЯ, ASP.NET CORE, SQL SERVER, ENTITY FRAMEWORK, ВЕБ-ДОДАТОК, МОДУЛЬНА СТРУКТУРА.

Annotation

The qualification work is devoted to the development of an information system for automating the processes of managing orders and settling clients in a hotel complex. The main goal of the research is to create an effective and user-friendly web application that provides the implementation of key functions: customer registration, making reservations, registering settlements and evictions, as well as administering users with different access levels.

In the implementation process, modern software engineering tools were used, in particular, a technological stack that includes the C# programming language, the ASP.NET Core MVC framework, the Microsoft SQL Server database management system, as well as the Entity Framework Core ORM technology. The architectural model is built on the principles of multi-level separation of responsibilities, which increases modularity and simplifies system maintenance.

The information system is implemented in the form of a web application with an adaptive interface that provides interactive interaction with the user and supports the execution of basic operations within the hotel service. The developed system allows you to optimize internal processes, increase staff efficiency and reduce the number of errors related to the human factor.

The volume of the work is 62 pages of the main text, contains 4 tables, 31 illustrations. The list of used sources includes 30 items.

Keywords: INFORMATION SYSTEM, PROCESS AUTOMATION, HOTEL SERVICE, RESERVATION, ASP.NET CORE, SQL SERVER, ENTITY FRAMEWORK, WEB APPLICATION, MODULAR STRUCTURE.

ЗМІСТ

ВСТУП 8

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ 9

1.1 Загальна характеристика готелю «Гірське повітря 2» 9

1.2 Організаційно-функціональна модель готелю «Гірське повітря 2» і взаємодія підрозділів. 10

1.3 Аналіз існуючої комп'ютеризації на ринку інформаційних технологій .. 14

1.4 Функціональне моделювання та системний аналіз діяльності ІТ-відділу .. 16

1.5 Огляд існуючих рішень для розв'язання виявлених проблем 19

1.6 Обґрунтування доцільності проєктування й розроблення інформаційної системи для управління замовленнями та поселенням клієнтів у готелі 23

1.7 Концептуальна модель системи 25

1.8 Розрахунок економічного ефекту від впровадження системи 26

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЄКТУВАННЯ 29

2.1 Найменування розробки 29

2.2 Підстава для розробки 29

2.3 Мета створення системи 29

2.4 Цілі та завдання розробки 29

2.5 Вимоги до функціонування системи 30

2.6 Архітектура та компоненти 30

2.7 Вимоги до надійності, безпеки та захисту інформації 32

2.8 Стадії та етапи розробки 33

РОЗДІЛ 3. ПРОЄКТУВАННЯ, СТВОРЕННЯ ТА АПРОБАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 35

3.1 Опис та обґрунтування вибору програмно-технічних засобів розроблення програмного продукту. 35

3.2 Проєктування та створення моделі бази даних 36

3.3 Реалізація функцій системи 37
3.3 Реалізація функцій системи 41
3.4 Інструкція користувача 47
3.5 Тестування програмного продукту 55
ВИСНОВКИ 58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 60
ДОДАТКИ 63
Додаток А. Моделі та схеми бази даних 63
Додаток Б. Фрагменти коду програми 65

ВСТУП

У сучасних умовах розвитку інформаційних технологій особливого значення набуває автоматизація бізнес-процесів у сфері обслуговування. Готельна галузь, як одна з найдинамічніших складових туристичної індустрії, вимагає ефективних і гнучких інструментів для організації та контролю основних операцій — бронювання номерів, обліку клієнтів, управління поселенням і виселенням, адміністрування номерного фонду тощо.

Стандартні методи управління, що базуються на ручному введенні даних або використанні обмежених функціональних можливостей типових офісних програм, часто призводять до зниження продуктивності праці персоналу, збільшення кількості помилок і, як наслідок, зниження рівня обслуговування клієнтів. Саме тому створення спеціалізованих інформаційних систем, орієнтованих на потреби конкретного закладу, є доцільним і стратегічно важливим напрямом розвитку.

Метою даної кваліфікаційної роботи є розробка програмного забезпечення, яке дозволяє здійснювати управління клієнтами та замовленнями у межах готельного підприємства, включаючи авторизацію користувачів, створення та редагування бронювань, автоматичне відстеження статусів номерів, облік заселень та виселень. Для досягнення цієї мети застосовано сучасні засоби розробки програмного забезпечення, такі як мова програмування C#, фреймворк ASP.NET Core MVC, Entity Framework Core для роботи з базою даних, а також система управління базами даних SSMS 20.

Об’єктом дослідження є процес організації та автоматизації управлінської діяльності у готельному сервісі. Предметом дослідження є сукупність методів і засобів розробки програмних рішень для інформаційного забезпечення готельного бізнесу.

Практична цінність роботи полягає у створенні прототипу інформаційної системи, що забезпечує підвищення продуктивності праці персоналу, зниження витрат часу на обробку запитів клієнтів і підвищення точності даних у внутрішніх бізнес-процесах.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальна характеристика готелю «Гірське повітря 2»

Готель **«Гірське повітря 2»** знаходиться в місті Яремче, яке є популярним туристичним напрямом у Карпатах. Це сучасний готель, який пропонує зручні умови для відпочинку та перебування в одному з найкрасивіших куточків України.

Готель розташований приблизно за 600 метрів від центру Яремче, що забезпечує зручний доступ до основних туристичних місць, ресторанів та магазинів. Також неподалік знаходяться природні пам’ятки, такі як скеля «Слон» і водоспад «Пробій».

У готелі є різноманітні номери, оснащені всім необхідним для комфортного перебування:

- безкоштовний Wi-Fi для зручності гостей;
- система опалення для комфортної температури в будь-яку пору року;
- тераса або балкон з чудовим видом на Карпати;
- власна ванна кімната з гігієнічними засобами;
- телевізор з плоским екраном;
- зона для відпочинку в номерах.

Також на території готелю є спільна кухня, обладнана всім необхідним для приготування їжі, а також зона для барбекю та тераса для відпочинку на свіжому повітрі.

Наразі готель **«Гірське повітря 2»** доступний для бронювання виключно через платформу Booking.com [1]. Хоча ця платформа є зручним інструментом для резервації номерів, було б набагато зручніше для гостей, якщо б готель мав власний сайт. Це дозволило б швидше отримати всю необхідну інформацію про готель, наявність вільних номерів та умови бронювання, а також знизило б залежність від третіх осіб для обробки замовлень.

1.2 Організаційно-функціональна модель готелю «Гірське повітря 2» і взаємодія підрозділів.

1.2.1 Загальна схема організаційної структури

Організаційна структура готелю «Гірське повітря 2» має на меті ефективне управління та оптимізацію робочих процесів для забезпечення високого рівня обслуговування гостей. На рисунку 1.1 наведена структура готелю є лінійною та передбачає розподіл обов'язків між кількома основними відділами, кожен з яких виконує конкретні функції, пов'язані з обслуговуванням гостей та управлінням готелем [2].

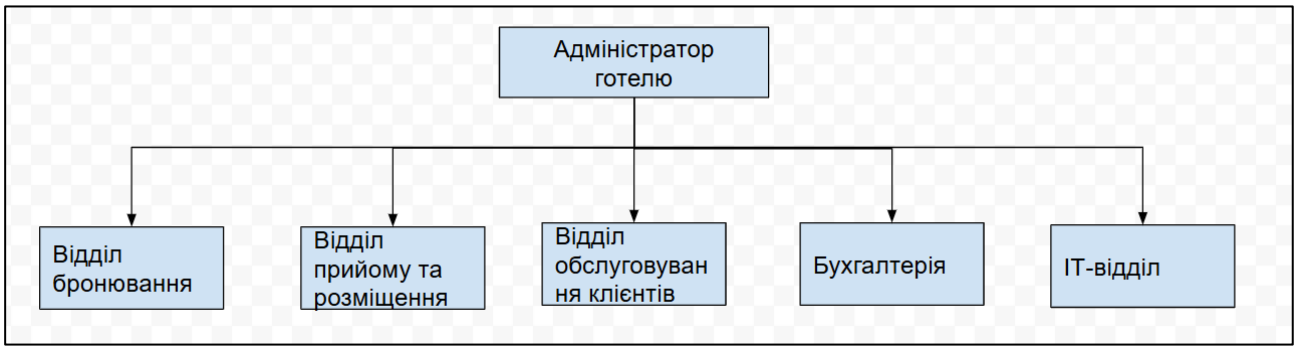


Рисунок 1.1 – Організаційно-функціональна схема готелю

1. Адміністратор готелю

- Головна посадова особа, яка відповідає за управління готелем та координацію роботи всіх відділів.
- Контролює виконання стандартів обслуговування, займається вирішенням організаційних питань та підтримує взаємодію між персоналом.

2. Відділ бронювання

- Обробляє запити на бронювання номерів через онлайн-платформи та інші канали.
- Відстежує зміни в наявності номерного фонду та веде відповідну базу даних.
- Забезпечує ефективне управління розміщенням гостей у відповідності до їхніх запитів.

3. Відділ прийому та розміщення

- Займається реєстрацією гостей при заселенні та виселенні.
- Надає клієнтам повну інформацію про готельні послуги.
- Координує взаємодію між гостями та іншими підрозділами готелю.

4. Відділ обслуговування клієнтів

- Відповідає за комфортне перебування гостей у готелі.
- Контролює роботу покоївок, обслуговуючого персоналу та технічного забезпечення номерів.
- Вирішує питання та скарги клієнтів, прагнучи підвищити рівень сервісу.

5. Бухгалтерія

- Веде фінансовий облік та контроль за грошовими операціями готелю.
- Здійснює нарахування заробітної плати персоналу та веде податкову звітність.
- Контролює оплату за проживання та інші послуги готелю.

6. IT-відділ

- Забезпечує безперебійну роботу комп’ютерних систем, інтернету та електронної бази даних.
- Обслуговує системи бронювання та реєстрації клієнтів.
- Відповідає за інформаційну безпеку та підтримку технічного обладнання.

1.2.2 Структура IT-відділу

Для роботи було обрано IT-відділ (Рис. 1.2), оскільки він відіграє ключову роль у забезпеченні безперебійного функціонування інформаційних систем готелю. Саме цей відділ відповідає за підтримку та розвиток програмного забезпечення, що використовується для управління бронюваннями, реєстрацією гостей, фінансовим обліком та іншими бізнес-процесами [3].

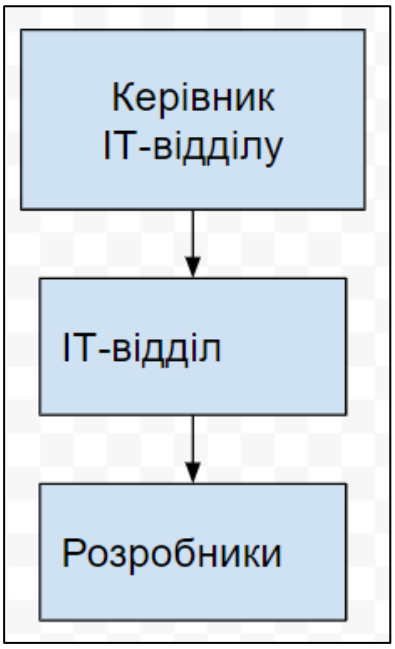


Рисунок 1.2 – Організаційно-функціональна ІТ-відділу

Загальні функції наведені у таблиці 1.1.

Таблиця 1.1. Завдання і функції ІТ-відділу

№	**Задачі**	**Функції**
1	Забезпечення роботи інформації систем	Підтримання та оновлення системи Забезпечення оперативного реагування на відмовлення інформаційної системи
2	Забезпечення безпеки інформації	захист даних клієнтів та співробітників, управління доступом до систем Забезпечення цілісності даних (наприклад резервне копіювання)

Керівник ІТ-відділу відповідає за:

- Організація та контроль роботи ІТ-відділу.
- Визначення стратегії розвитку інформаційних систем готелю.
- Контроль за безпекою даних та стабільністю роботи всіх ІТ-систем.
- Розподіл завдань між співробітниками ІТ-відділу.

- Координація взаємодії з іншими відділами готелю для забезпечення безперебійної роботи інформаційних систем (Табл. 1.2) .
- Прийняття рішень щодо оновлення та впровадження нового програмного забезпечення.
- ІТ-відділ відповідає за:
- Адміністрування серверів, баз даних та мережевого обладнання готелю [4].
- Підтримка роботи інформаційних систем та швидке реагування на технічні збої.
- Забезпечення інформаційної безпеки та захисту персональних даних клієнтів [5].
- Впровадження та обслуговування нового програмного забезпечення.
- Надання технічної підтримки працівникам готелю.
- Розробка та оптимізація ІТ-інфраструктури відповідно до потреб бізнесу.

Розробники відповідають за:

- Створення, тестування та впровадження нового програмного забезпечення для автоматизації процесів готелю.
- Оптимізація роботи існуючих інформаційних систем.
- Інтеграція програмного забезпечення з зовнішніми сервісами (наприклад, платіжні системи, сторонні платформи бронювання).
- Усунення багів та оновлення функціоналу системи.
- Розробка внутрішнього вебсайту готелю для бронювання номерів напряму без використання сторонніх платформ.
- Створення та підтримка мобільних додатків або веб-інтерфейсів для адміністраторів та клієнтів.

Таблиця 1.2 . Взаємодія IT-відділу з іншими підрозділами та відділами

№	Відділ	Одержання	Надання
1	Відділ бронювання	Запит на надання інформації про бронюванню номерів	Надання актуальної інформації про бронюванню номерів
2	Служба прийому та розміщення	Інформація про нового або оновлення старого клієнта	Збереження нового або оновлення старого клієнта
3	Відділ бухгалтерії	Надання інформації про оплату номера	Збереження інформації

1.3 Аналіз існуючої комп’ютеризації на ринку інформаційних технологій

На сьогоднішній день готельний бізнес активно використовує різноманітні інформаційні технології для автоматизації та оптимізації роботи персоналу. Комп’ютеризація дозволяє значно спростити процеси бронювання, реєстрації клієнтів, управління фінансами та технічної підтримки.

Технічне забезпечення

У роботі готелю використовується наступне обладнання:

- Персональні комп’ютери та ноутбуки – використовуються адміністраторами для реєстрації гостей, ведення бази клієнтів, роботи з фінансовими та звітними документами.
- Принтери та сканери – необхідні для друку договорів, чеків, квитанцій та інших документів.
- Сервери – для зберігання баз даних, систем бронювання та інших важливих даних [6].

- Мережеве обладнання (Wi-Fi маршрутизатори, комутатори, сервери доступу) – забезпечує стабільну роботу внутрішніх систем та доступ до Інтернету.
- Камери відеоспостереження – підключені до серверної системи для контролю безпеки в готелі.

Програмне забезпечення

Для ефективної роботи готелю використовується комплексне програмне забезпечення:

1. Система управління готелем (PMS – Property Management System) [7]:

- Використовується для управління номерами, бронюванням, поселенням та випискою клієнтів.
- Автоматизує процес бронювання та ведення обліку номерного фонду.
- Дозволяє вести історію клієнтів (запити, частота проживання, особливі побажання).
- Інтегрується з іншими сервісами, такими як платіжні системи, електронні ключі, служби прибирання.

2. База даних клієнтів

- Використовується для збереження інформації про постійних клієнтів, їхні контактні дані, історію проживання та побажання.
- Реалізована на MySQL або PostgreSQL з доступом через спеціалізовані CRM-системи [8, 9].

3. Системи онлайн-бронювання

- Основний інструмент для бронювання номерів – платформа Booking.com. Всі номери доступні для бронювання лише через цей сервіс.
- Через відсутність власного сайту готель не може напряму керувати бронюваннями, що створює певні незручності.

4. Програмне забезпечення для фінансового обліку

- Використовується програма 1С:Бухгалтерія для ведення фінансового обліку, формування звітності та контролю надходжень і витрат.

- Excel використовується для формування внутрішніх звітів, розрахунку заробітної плати, ведення обліку витрат та доходів.

5. Програми для комунікації та обслуговування клієнтів [10].

- Telegram, Viber, WhatsApp – використовуються адміністраторами та службою підтримки для комунікації з клієнтами.
- E-mail (Gmail, Outlook) – застосовується для прийому заявок, надсилання рахунків та листування з партнерами.

1.4 Функціональне моделювання та системний аналіз діяльності IT-відділу

1.4.1 Функціональна модель бронювання номерів та поселення клієнтів в готель.

Функціональне моделювання діяльності готелю дозволяє детально визначити всі виконувані процеси, їхню взаємодію та розподіл ресурсів та оцінити ефективність існуючих механізмів і знайти шляхи їх покращення за рахунок автоматизації.

Ціль моделювання: Визначення ключових процесів готелю, які можна оптимізувати за допомогою автоматизованих систем, зокрема управління бронюваннями, поселенням клієнтів та супутніми послугами.

Область моделювання: Включає всі етапи роботи із клієнтом – від моменту подання запиту на бронювання до завершення проживання, розрахунку та формування підсумкової звітності.

Для побудови функціональної моделі було використано середовище AllFusion Process Modeler 7, у якому розроблені процеси у нотаціях IDEF0 (Рис. 1.3) [11].

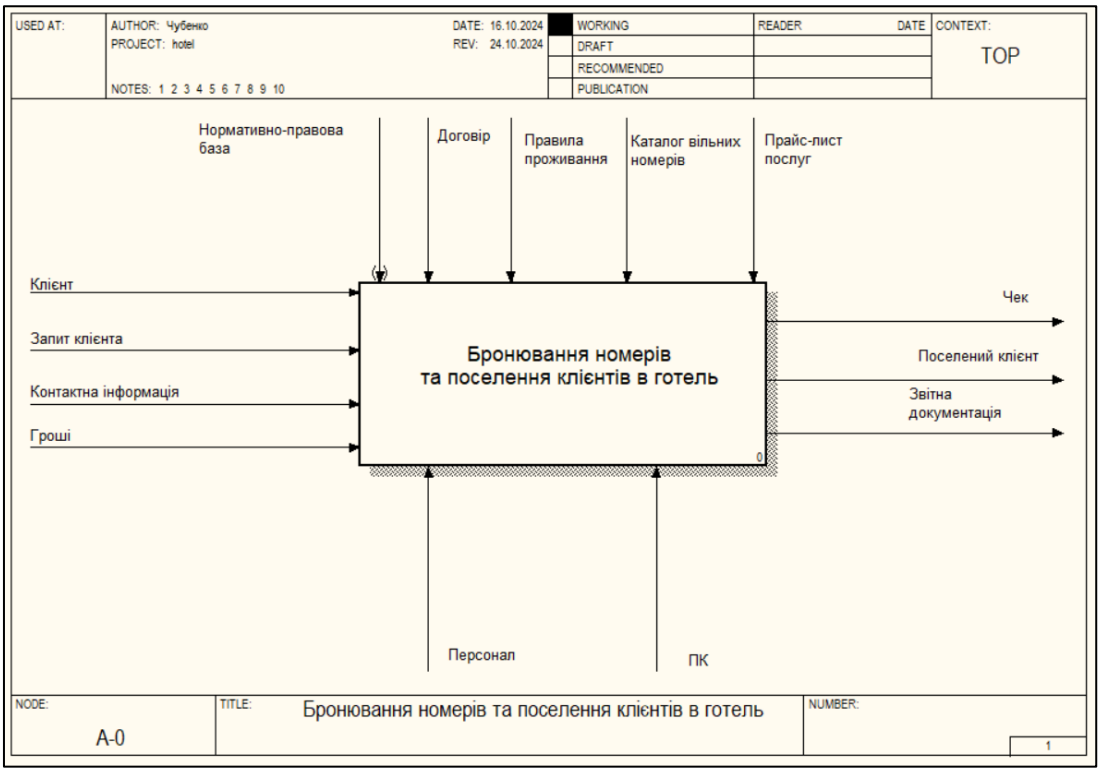


Рисунок 1.3 – Діаграма верхнього рівня

Вхідні дані:

- Запит клієнта.

Вихідні дані:

- Виконане замовлення.
- Заявка на матеріали.
- Чек.
- Звіт по роботі.

Контроль в системі:

- Прайс-лист.
- Журнал записів.
- Нормативні документи.

Механізми управління:

- Клієнт.
- Адміністратор.
- Персонал.

Далі дана модель розбивається на більше точніші етапи, які наведені на рисунку 1.4.

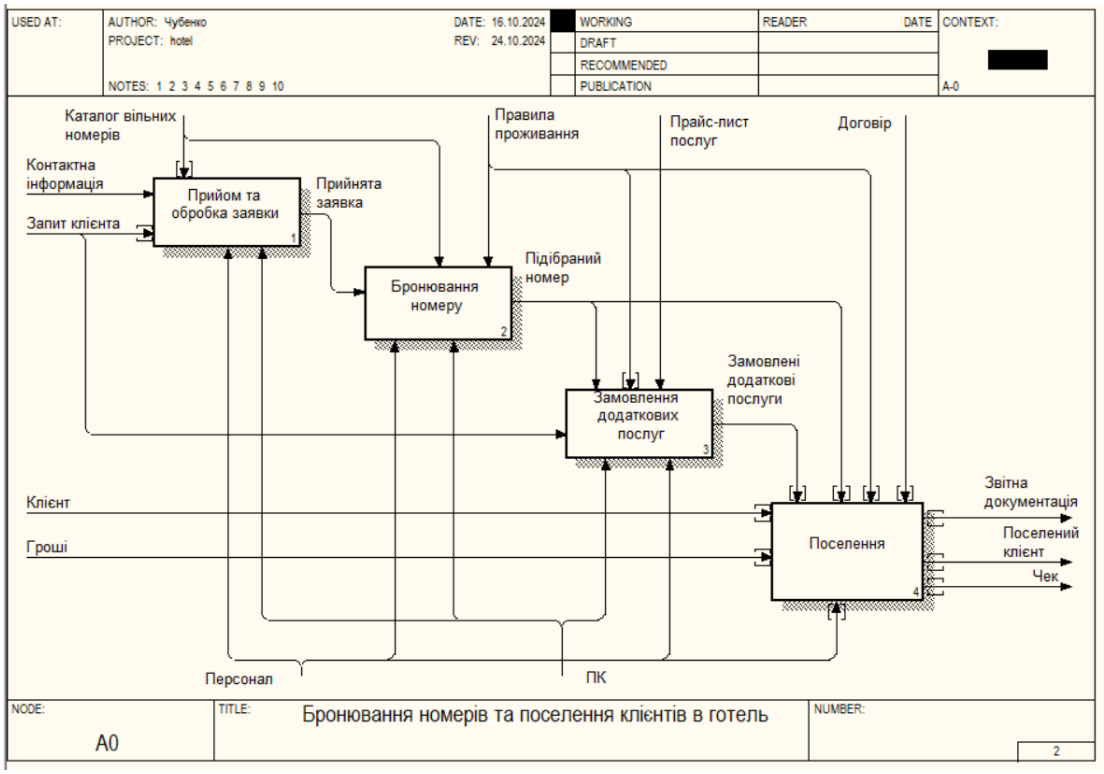


Рисунок 1.4 – Діаграма декомпозиції бронювання номерів та поселення клієнтів в готель.

1.4.2 **Задачі автоматизації**

У процесі розробки інформаційної системи для управління замовленнями та поселенням клієнтів у готелі будуть реалізовані наступні функціональні можливості:

1. **Автоматизація процесу бронювання**

- Реєстрація нових бронювань через інтерфейс адміністратора.
- Відображення статусу номерів у реальному часі.
- Можливість зміни або скасування бронювання.
- Інтеграція з базою даних клієнтів для швидкого пошуку постійних гостей.

2. **Управління поселенням та виселенням клієнтів**

- Автоматична реєстрація заїзду гостей.
- Відстеження строків перебування.
- Автоматизований процес виселення та оновлення статусу номерів.

3. **Ведення бази даних клієнтів**

Управління персональними даними клієнтів відповідно до вимог безпеки.

4. Управління номерним фондом

- Візуальне представлення всіх доступних номерів.
- Категоризація номерів за рівнем комфорту та ціною.
- Автоматизоване оновлення інформації про зайнятість.

5. Інтеграція з іншими сервісами

- Можливість підключення до сторонніх платформ для бронювання.
- API для обміну даними між системами [12].

Ці функціональні можливості дозволять зробити процес управління готелем більш ефективним, швидким і зручним як для адміністраторів, так і для гостей.

1.5 Огляд існуючих рішень для розв'язання виявлених проблем

1.5.1 Існуючі програмні продукти на ринку для розв'язання виявлених проблем

Для вирішення завдань автоматизації готельного обліку, таких як бронювання номерів, управління персоналом та обробка замовлень клієнтів, на ринку існує низка готових програмних рішень. До найпопулярніших належать RoomRaccoon і Cloudbeds — сучасні хмарні платформи, що поєднують функціонал систем управління готелем (PMS), CRM, а також канали онлайн-бронювання.

На рисунку 1.5 представлено інтерфейс програми RoomRaccoon, який дозволяє адміністратору зручно переглядати статус номерного фонду, здійснювати бронювання та працювати з клієнтськими запитами.

Рисунок 1.6 ілюструє систему Cloudbeds, що підтримує інтеграцію з платформами Booking, Airbnb, Expedia тощо, і забезпечує повноцінне керування готельним ресурсом у режимі реального часу.

RoomRaccoon надає незалежним власникам готелів інтегровану платформу для максимізації прибутку, контролю операцій і задоволення гостей завдяки автоматизації процесів. У 2020 та 2021 роках RoomRaccoon була визнана

найкращою системою управління готелем за версією Hotel Tech Report і пропонує інноваційні рішення власникам готелів у всьому світі [13].

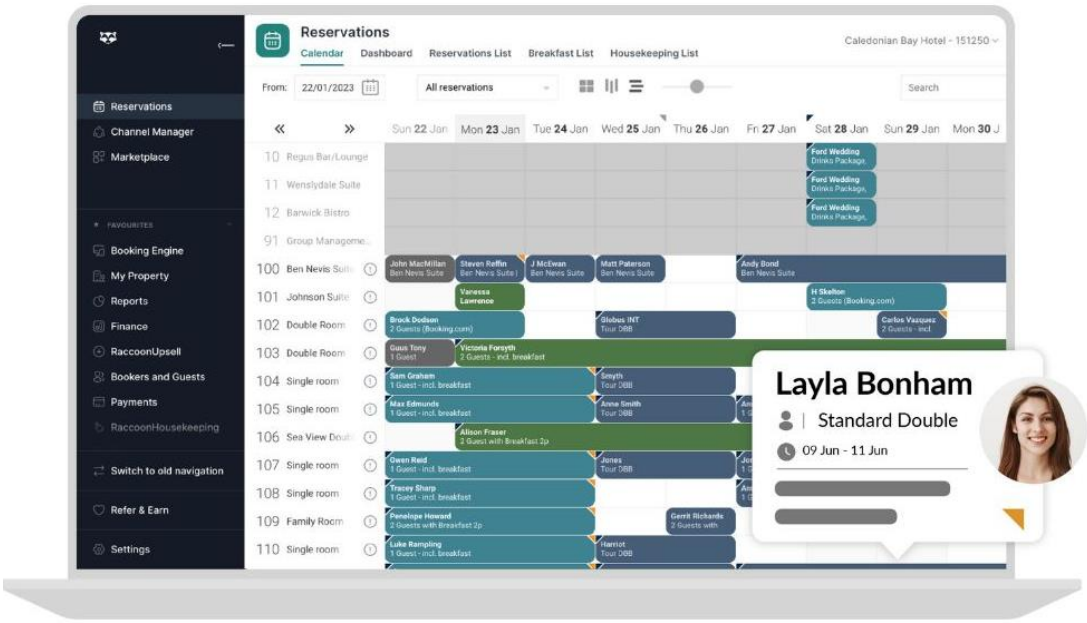


Рисунок 1.5 - Приклад інтерфейсу

Основний продукт поєднує першу в готелі PMS, менеджер каналів, систему бронювання, а також широкий спектр модулів і оновлень для платежів, прибирання, звітності та цифрових послуг для гостей. Усі ці функції повністю інтегровані, що дозволяє власникам готелів керувати всією роботою з єдиної платформи.

– Розробник RoomRaccoon B.V.

– Архітектура: SaaS (Software as a Service) платформа, доступна через веб-браузер.

Функції та характеристики:

- Календар бронювань з можливістю управління доступністю номерів.
- Інтеграція з каналами продажів (Booking.com, Airbnb, Expedia тощо).
- Система управління готельними послугами (SPA, ресторани).
- Фінансові звіти та аналітика продажів.

- Автоматизація комунікації з гостями (відправка підтверджень, нагадувань).

Переваги:

- Легкий у використанні інтерфейс.
- Широкі можливості інтеграції з іншими сервісами.
- Наявність мобільного додатку.

Недоліки:

- Досить висока вартість підписки.
- Обмежений функціонал у базових тарифних планах.

2. Cloudbeds

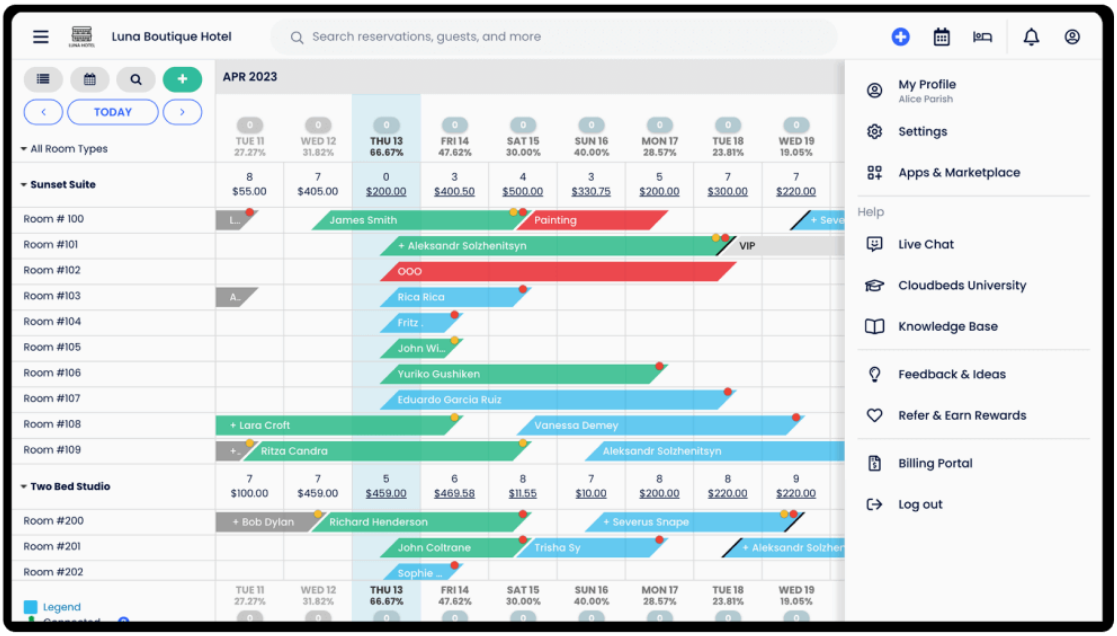


Рисунок 1.6 - Приклад інтерфейсу

Cloudbeds - це хмарна платформа для управління готелем, що підходить для малих та середніх готелів, будинків для відпустки, хостелів та готелів типу B&B [14].

Незалежно від того, є власником незалежного готелю або частиною готельної мережі, Cloudbeds пропонує єдину платформу для управління гостями, персоналом, інвентарем, цінами та даними в режимі реального часу.

Cloudbeds дозволяє отримувати бронювання по ряду каналів, таких як Booking.com, Expedia або Facebook, онлайн бронювання або від реєстрованих на Вашому сайті клієнтів. Платформа пропонує календар з перетягуванням усіх номерів та бронювань, і адміністратор може бачити, хто змінив, скасував, створив, перемістив чи скоригував бронювання.

Cloudbeds має інтуїтивно-зрозумілий інтерфейс і працювати на ньому дуже легко. У ньому є безліч інструментів, які допоможуть вам керувати не лише своєю власністю, а й вашим персоналом. платформа включає дозволи, що настроюються по ролях, грошову скриньку для управління грошима і службовий модуль.

Платформа підтримує кілька мов та валют. Адміністратор сам встановлює податки, збори та бонуси, які будуть застосовуватись одразу до всіх номерів, груп номерів або до кожного номера індивідуально.

– Розробник: Cloudbeds Inc.

– Архітектура: Хмарна платформа з доступом через браузер та мобільний додаток.

Функції та характеристики:

- Управління бронюваннями та доступністю номерів.
- Підключення до понад 300 каналів продажів.
- CRM (Customer Relationship Management) для відстеження взаємодії з клієнтами [15].
- Управління доходами з використанням динамічного ціноутворення.
- Інтеграція з платіжними системами.

Переваги:

- Високий рівень автоматизації процесів.
- Можливість використання на малих та великих підприємствах.
- Підтримка багатомовності та мультивалютності.

Недоліки:

- Відсутність офлайн-режиму роботи.
- Складний процес налаштування для початківців.

1.5.2 Порівняння програмних продуктів

Для більш повного розуміння можливостей існуючих систем управління готельним бізнесом доцільно провести порівняльний аналіз їх функціональності.

Таблиця 1.3 Порівняльна таблиця функцій

Параметри	**RoomRaccoon**	**Cloudbeds**
Календар бронювань	+	+
Інтеграція з каналами	+	+
Управління доходами	-	+
CRM	-	+
Мобільний додаток	+	+
Автоматизація комунікацій	+	+
Управління послугами	+	-
Динамічне ціноутворення	-	+

На основі порівняльної таблиці (Табл. 1.3), можна зробити висновок щодо доцільності впровадження кожної з представлених систем в готельний бізнес для автоматизації процесів бронювання номерів та управління. Cloudbeds - ця система надає широкий спектр функцій, вона є найбільш функціональною серед представлених систем і має всі необхідні інструменти для управління готельним бізнесом.

1.6 Обґрунтування доцільності проєктування й розроблення інформаційної системи для управління замовленнями та поселенням клієнтів у готелі

В умовах сучасного готельного бізнесу ефективне управління замовленнями та поселенням клієнтів відіграє ключову роль у забезпеченні

якісного сервісу та оптимізації внутрішніх процесів. Багато готелів, особливо невеликі та середні заклади, досі використовують застарілі методи обліку, такі як паперові журнали або нескладні електронні таблиці, що може призводити до помилок, дублювання даних та затримок у роботі персоналу [16].

Розробка **інформаційної системи для управління замовленнями та поселенням клієнтів** дозволить автоматизувати ключові бізнес-процеси, підвищити рівень обслуговування гостей та знизити витрати на адміністрування.

Основні аргументи на користь розробки системи:

1. Підвищення точності та швидкості обробки бронювань

- Автоматизація прийому замовлень дозволить знизити ймовірність людських помилок.

2. Зручність для персоналу

- Адміністратори отримають єдину платформу для управління всіма процесами, що скоротить час на виконання рутинних завдань.
- Спрощена система обліку клієнтів та платежів дозволить швидко знаходити потрібну інформацію.

3. Поліпшення якості обслуговування гостей

- Можливість швидкої реєстрації та виселення клієнтів без зайвих процедур.
- Формування персоналізованих пропозицій та системи лояльності для постійних клієнтів.

4. Оптимізація фінансового обліку та звітності

- Автоматичне формування рахунків та фінансових звітів.
- Контроль оплат та боргів у реальному часі.
- Аналітика доходів та витрат для прийняття управлінських рішень.

5. Інтеграція з іншими сервісами

- Можливість підключення до сторонніх платформ для онлайн-бронювання (Booking, Airbnb тощо).
- API для обміну даними між системами готелю.

6. Безпека даних

- Захист персональних даних клієнтів відповідно до сучасних стандартів.
- Контроль доступу користувачів до різних функцій системи.

Крім того, важливою перевагою є використання вебтехнологій для реалізації інформаційної системи. Web-застосунок дозволяє працювати з системою з будь-якого пристрою, що має доступ до Інтернету, без необхідності встановлення додаткового програмного забезпечення на кожен комп'ютер. Це особливо зручно для готельного персоналу, який може здійснювати керування бронюваннями, перевірку заселеності номерів чи обробку платежів із різних робочих місць або навіть віддалено. Вебінтерфейс забезпечує простоту обслуговування та кращу масштабованість.

Таким чином, розробка та впровадження даної інформаційної системи дозволить значно оптимізувати процеси управління готелем, підвищити ефективність роботи персоналу, покращити якість обслуговування клієнтів і, як наслідок, підвищити конкурентоспроможність готелю.

1.7 Концептуальна модель системи

Функціональна модель TO-BE, яка буде аналізувати роботу готелю з бронювання номерів та поселення клієнтів з використанням інформаційної WEB-системи. В функціональній моделі батьківською діаграмою є «Бронювання номерів та поселення клієнтів в готель» [17, 18].

Вхідними стрілками на діаграмі є:

– Запит клієнта.

Контроль в системі забезпечує:

– Нормативно-правова база;

– Договір;

– Правила проживання;

– Каталог вільних номерів;

– Прайс-лист послуг.

Механізмами управління є:

– Клієнт;

– Персонал;

– ПК.

Вихідними даними системи є:

– Чек;

– Поселений клієнт;

– Звітна документація.

Головна батьківська діаграма декомпозується на нижній рівень, який складається з діяльностей: обробка запиту клієнта; бронювання номера; поселення клієнта; формування документації та звітів.

1.8 Розрахунок економічного ефекту від впровадження системи

Розробка та впровадження інформаційної системи для управління замовленнями та поселенням клієнтів у готелі має на меті підвищення ефективності роботи персоналу, зменшення витрат часу на адміністративні процеси та покращення якості обслуговування гостей. Для оцінки економічного ефекту від впровадження системи розглянемо основні показники, які впливають на фінансові результати підприємства.

1. Оптимізація роботи персоналу

Завдяки автоматизації процесів, таких як бронювання номерів, реєстрація клієнтів та ведення фінансового обліку, скорочується час, який витрачає персонал на виконання рутинних завдань.

- Очікуване скорочення витрат робочого часу адміністратора: 30-40%
- Можливе зменшення потреби в додатковому персоналі, що скорочує витрати на заробітну плату

2. Зниження кількості помилок при бронюванні та розрахунках

Автоматизована система знижує ризик людських помилок при веденні документації, бронюванні номерів та фінансових операціях, що:

- Зменшує кількість дублюючих або помилкових бронювань (економія до **5-10%** прибутку)
- Скорочує витрати на виправлення помилок

3. Підвищення доходів від ефективного управління номерами

Система дозволяє аналізувати завантаженість номерного фонду, прогнозувати попит та встановлювати оптимальні ціни, що:

- Підвищує коефіцієнт заповнюваності готелю на 5-15%
- Дозволяє ефективніше керувати ціноутворенням (динамічне ціноутворення)

4. Скорочення витрат на паперовий документообіг

Перехід на цифрову форму обліку бронювань, оплат та звітності дозволяє зменшити витрати на:

- Друк квитанцій, договорів, рахунків
- Архівацію документів
- Канцелярські товари

Очікуване скорочення витрат: до 20-30% на рік.

5. Поліпшення взаємодії з клієнтами

Система дозволяє інтегрувати сервіси онлайн-бронювання (Booking.com, Airbnb), автоматизувати сповіщення клієнтів про підтвердження бронювання, спеціальні пропозиції, що сприяє:

- Підвищенню рівня задоволеності клієнтів
- Збільшенню повторних бронювань

6. Зменшення витрат на обслуговування IT-інфраструктури

Оскільки нова система буде мати централізоване управління, вона дозволить зменшити витрати на підтримку застарілого програмного забезпечення та обладнання.

Підсумковий розрахунок економічного ефекту

Річна економія витрат:

- Оптимізація витрат на персонал: ≈ 10-20%
- Скорочення втрат від помилок та дублювань: ≈ 5-10%

- Збільшення доходу від ефективного управління бронюванням: ≈ 5-15%
- Зниження витрат на документообіг: ≈ 20-30%
- Скорочення витрат на підтримку ІТ-інфраструктури: ≈ 10-15%

Очікувана загальна економія: 10-25% від загального обороту готелю на рік.

Таким чином, впровадження інформаційної системи дозволить значно підвищити продуктивність роботи, зменшити витрати та збільшити прибуток підприємства, що підтверджує доцільність її розробки та впровадження.

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЄКТУВАННЯ

2.1 Найменування розробки

Назва проєкту: "Інформаційна система управління замовленнями та поселенням клієнтів у готелі".

2.2 Підстава для розробки

Розробка здійснюється в межах виконання кваліфікаційної (дипломної) роботи здобувача вищої освіти за освітнім ступенем "бакалавр" відповідно до навчального плану підготовки фахівців за спеціальністю 122 "Комп’ютерні науки". Підставою є затверджене завдання на кваліфікаційну роботу.

2.3 Мета створення системи

Метою створення даної системи є підвищення ефективності обслуговування клієнтів готелю, спрощення операцій з обліку, мінімізація людських помилок, забезпечення швидкого доступу до інформації та формування звітів. Система має забезпечити зручний та інтуїтивно зрозумілий інтерфейс, який дозволяє персоналу готелю працювати з бронюваннями, клієнтами та звітністю без спеціальних технічних знань.

2.4 Цілі та завдання розробки

1. забезпечити автоматизовану обробку заявок на бронювання;
2. надати персоналу інструменти для швидкого поселення та виселення клієнтів;
3. реалізувати зберігання та доступ до історії клієнтів і фінансових операцій;
4. інтегрувати підтримку звітності щодо завантаженості номерів і доходів;
5. забезпечити налаштування прав доступу для різних категорій користувачів;
6. передбачити розширення функціоналу в межах модульної архітектури.

2.5 Вимоги до функціонування системи

2.5.1 Вимоги до функціонування системи

Система повинна реалізовувати такі функції:

1. авторизація та аутентифікація користувачів з ролями (адміністратор, оператор, менеджер);
2. створення, редагування, перегляд і видалення замовлень;
3. фільтрація даних за датою, типом послуг, статусом оплати тощо;
4. автоматичне оновлення статусу номерів;
5. інтеграція з зовнішніми сервісами (платіжні системи, онлайн-бронювання);
6. забезпечення інформаційної безпеки відповідно до чинного законодавства.

2.5.2 Вимоги до технічного забезпечення

- **Платформа:** Windows 10 і вище;
- **Мова програмування:** C# (ASP.NET Core MVC)[19];
- **Бібліотеки:** PDFSharp або iTextSharp для PDF-генерації;
- **Архітектура:** Локальна або клієнт-серверна;
- **Зберігання даних:** централізовано в БД з логуванням дій користувача(microsoft sql server management studio 2020) [20, 21];
- **Масштабованість:** можлива реалізація з урахуванням майбутнього переходу на мережевий режим.

2.6 Архітектура та компоненти

Архітектура системи базується на принципах модульності, розділення обов’язків та простоти обслуговування. Вона реалізується у вигляді настільного застосунку, що взаємодіє з локальною базою даних та використовую вхідні та вихідні данні (Табл. 2.1). Для забезпечення масштабованості передбачається можливість подальшого розширення системи до мережевої архітектури.

Складові компоненти:

- **Форма входу:** перевірка користувача, автентифікація, перевірка ролей (адміністратор, персонал).

- **Головне меню:** точка доступу до всіх функцій, адаптована до ролі користувача.
- **Модуль клієнтів:** введення ПІБ, контактних даних, паспорту, пошук клієнтів, редагування й збереження даних.
- **Модуль номерів:** відображення номерів у таблицях або візуально (плитка), з фільтрацією за статусами: вільний, заброньований, зайнятий.
- **Модуль бронювання:** вибір клієнта, підбір доступного номера, введення дати, автоматичний розрахунок тривалості проживання.
- **Модуль поселення/виселення:** підтвердження заселення, оновлення статусу номеру, завершення перебування.
- **Модуль адміністрування:** створення, редагування, блокування облікових записів.

Кожен з модулів буде реалізований у вигляді окремої форми у застосунку з логічно пов’язаними елементами, доступними з головного меню.

Таблиця 2.1 - Структура вхідних та вихідних даних системи

№ п/п	Найменування функції	Вхідна інформація	Вихідна інформація
1	Формування клієнтської бази	Інформація про клієнтів (ПІБ, телефон, email)	Таблиця з переліком клієнтів
2	Опис номерного фонду	Дані про номери (тип, ціна, кількість місць)	Таблиця номерів з усіма характеристиками
3	Реєстрація замовлень	Бронювання (дата заїзду, виїзду, клієнт, номер)	Список активних/архівних бронювань
4	Додавання додаткових сервісів	Інформація про послуги (назва, ціна, опис)	Список послуг для вибору при бронюванні

Продовження таблиці 2.1

№ п/п	Найменування функції	Вхідна інформація	Вихідна інформація
5	Автоматичне заповнення форм	Вибір із ComboBox (ПІБ, тип номеру тощо)	Відображення зв’язаних значень у формах
6	Пошук і аналітика	Параметри для фільтрації (дати, ПІБ, ціна)	Відфільтровані дані відповідно до умов запиту
7	Обробка змін у формі	Кнопки "додати", "зберегти", "видалити"	Оновлення записів у базі даних
8	Генерація звітів	Обрані параметри для формування звіту	Статистичні звіти по клієнтах, послугах, фінансах
9	Експорт інформації	Дані у ReportViewer	Звіт у форматі Excel або PDF
10	Доступ до бази	Параметри підключення до SQL Server	Завантаження актуальних даних у форми

2.7 Вимоги до надійності, безпеки та захисту інформації

Система повинна забезпечувати високий рівень надійності та безперервності функціонування. Надійність досягається завдяки використанню перевірених технологій, впровадженню механізмів автоматичного резервного копіювання бази даних, а також передбаченню процедур відновлення у випадку збоїв або втрати інформації.

З метою гарантування безпеки інформації система повинна реалізовувати багаторівневу модель доступу, що передбачає авторизацію користувачів із поділом за ролями. Доступ до критичних модулів має бути обмеженим відповідно до повноважень персоналу.

Дані передаються із застосуванням захищеного протоколу HTTPS [22]. Усі операції з клієнтською інформацією та фінансовими транзакціями мають фіксуватись у журналі дій для аудиту та внутрішнього контролю. Система повинна відповідати вимогам законодавства щодо обробки персональних даних (у тому числі вимогам GDPR, якщо передбачена взаємодія з іноземними клієнтами) [23].

2.8 Стадії та етапи розробки

У межах розроблення інформаційної системи управління замовленнями та поселенням клієнтів у готелі було визначено послідовність основних етапів виконання проєкту. Кожен етап передбачає виконання певного обсягу робіт, спрямованих на досягнення загальної мети розробки – створення ефективного програмного забезпечення для готельного бізнесу.

Календарний план розроблення проєкту наведено в таблиці 2.2. Він відображає розподіл робіт у часі та дозволяє здійснювати контроль за дотриманням строків виконання кожного етапу.

Таблиця 2.2 – Календарний план розроблення проєкту

№	**Етап**	**Термін виконання**
1	Збір вимог	28 квітня – 30 квітня 2025
2	Розробка ТЗ	1 травня – 3 травня 2025
3	Проєктування БД	4 травня – 6 травня 2025
4	Реалізація системи	7 травня – 12 травня 2025
5	Тестування	13 травня -15 травня 2025
6	Підготовка документації	16 травня – 20 травня 2025
7	Захист диплома	11 червня 2025

Для візуального представлення плану робіт використано діаграму Ганта (Рис. 2.1), яка демонструє послідовність і тривалість виконання кожного з етапів розроблення. Діаграма дозволяє наочно оцінити часові межі виконання проєкту та взаємозв'язок між окремими стадіями [24].

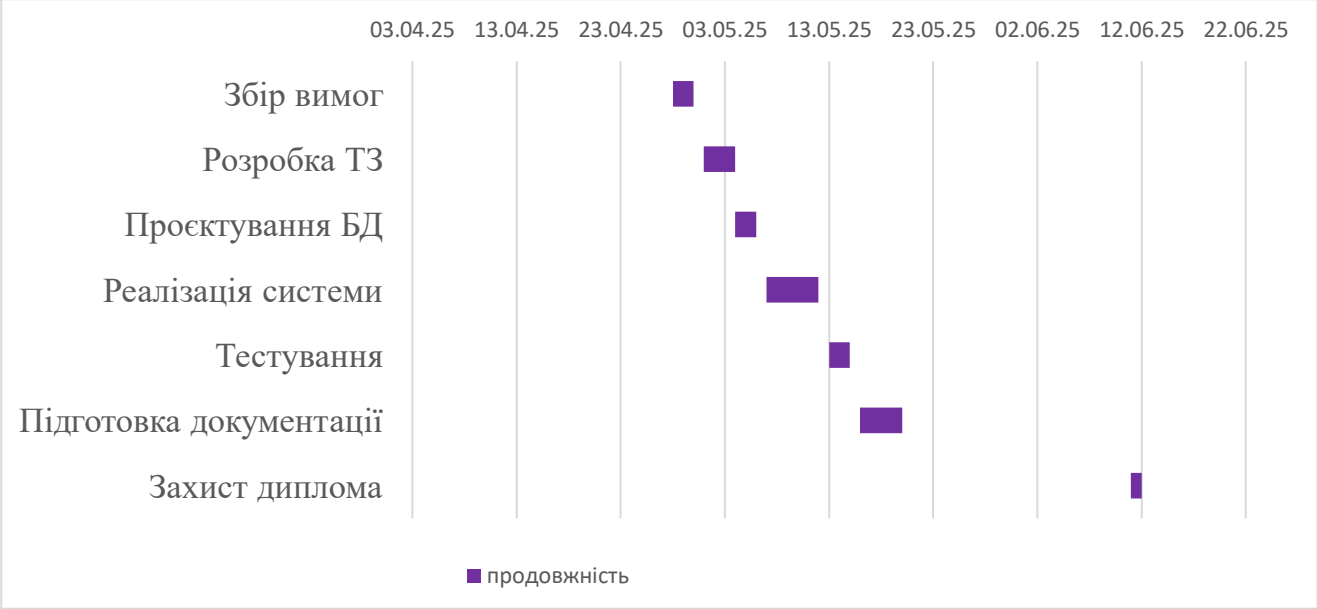


Рисунок 2.1 – Діаграма Ганта календарного плану розроблення системи

З аналізу діаграми Ганта видно, що проєкт складається з семи основних етапів, починаючи від збору вимог і закінчуючи захистом дипломного проєкту. Найбільше часу відведено на реалізацію системи, що є критично важливим етапом, оскільки передбачає розроблення серверної та клієнтської частин програмного продукту. Завершальні етапи включають тестування, підготовку документації та захист проєкту.

РОЗДІЛ 3. ПРОЄКТУВАННЯ, СТВОРЕННЯ ТА АПРОБАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис та обґрунтування вибору програмно-технічних засобів розроблення програмного продукту

У рамках створення інформаційної системи для автоматизації роботи готелю було поставлено завдання розробити веб-додаток, який дозволяє адміністратору та персоналу ефективно здійснювати бронювання номерів, заселення і виселення клієнтів, вести облік номерного фонду та управляти користувачами системи.

Вибір технологій та обґрунтування

Для розробки системи було обрано сучасний стек технологій, що забезпечує надійність, масштабованість та зручність підтримки:

- Мова програмування: C#
- Фреймворк для створення веб-застосунків: ASP.NET Core MVC
- ORM та робота з базою даних: Entity Framework Core + SSMS
- Інструмент розробки: Visual Studio 2022
- Фронтенд бібліотеки: Razor Pages + Bootstrap для адаптивного дизайну

Цей вибір дозволив реалізувати багаторівневу архітектуру, де чітко розділені такі частини:

- Дані (база даних, Entity Framework моделі);
- Бізнес-логіка (Controllers, Services)
- Інтерфейс користувача (Views, Razor Pages, Bootstrap компоненти).

Основні функціональні блоки системи

Для досягнення цілей проєкту, система побудована навколо таких ключових модулів:

- Управління клієнтами – зберігання даних про гостей, можливість додати, редагувати, шукати.

- Бронювання – створення бронювання з вибором клієнта, дат заїзду/виїзду та автоматичним підбором вільних номерів.
- Заселення – відображення актуальних бронювань на сьогодні та функція «Заселити».
- Виселення – облік виселень з розрахунком суми до сплати та зміною статусу номера на вільний.
- Адміністрування користувачів – функціонал лише для адміністраторів: додавання, редагування та видалення користувачів системи.
- Звіти – формування інформації щодо бронювань, клієнтів, завантаження номерів (цей блок в майбутньому планується для розширення).

Загальна схема архітектури наведена в **Додатку А**, рисунок **А.1**.

3.2 Проєктування та створення моделі бази даних

У рамках розробки системи управління готелем однією з ключових задач стало створення грамотно побудованої, зрозумілої та гнучкої бази даних. Це необхідно для ефективного зберігання інформації про клієнтів, бронювання, заселення та виселення, а також для організації взаємодії користувачів системи. Ретельно продумана структура бази даних — це запорука стабільної та надійної роботи всієї системи. Структура бази даних наведена на рисунку 3.1.

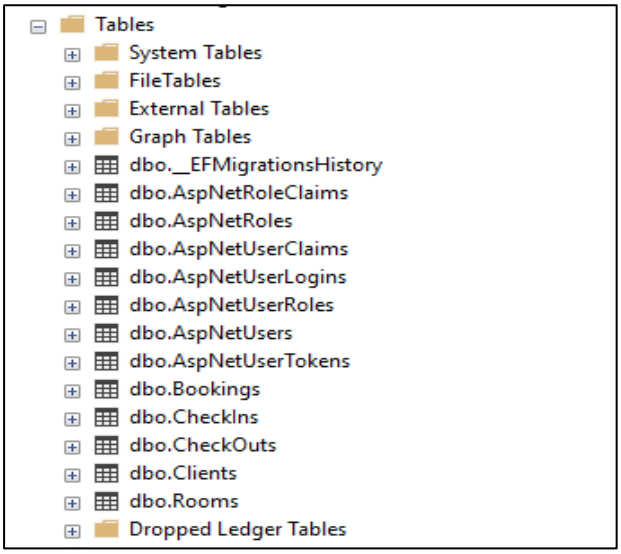


Рисунок 3.1 – структура бази даних

3.3 Реалізація функцій системи

База даних є реляційною та побудована за допомогою Entity Framework Core з використанням Code First міграцій [25, 26, 27, 28]. Код для відображення таблиць бази даних наведений у **Додатку Б**. Це дозволяє не тільки швидко розгортати структуру на будь-якому сервері, а й легко вносити зміни в майбутньому.

Основною точкою зв'язку між програмним кодом і базою даних слугує клас **ApplicationDbContext**, який оголошує всі таблиці у вигляді DbSet властивостей:

Таблиці користувачів і ролей

Усі функції авторизації та аутентифікації реалізовані за допомогою вбудованого механізму **ASP.NET Identity**. Тому в базі присутні наступні службові таблиці:

- **AspNetUsers** (Рис. 3.2) — зберігає інформацію про користувачів (email, пароль, повне ім'я та інше);

Results Messages

	Id	UserName	NormalizedUserName	Email	NormalizedEmail	EmailConfirmed	PasswordHash	SecurityStamp
1	40100709-5101-487d-bb20-f60486cd5153	admin@hotel.com	ADMIN@HOTEL.COM	admin@hotel.com	ADMIN@HOTEL.COM	1	AQAAAAIAAYagAAAAEA22bHMw/3Q7PfDV3N1lotE2xuIjtT83...	7TMXOGYKVRWPUNSDA4RY7HPJ6RA
2	ce8b5d25-32f1-486b-8e24-4ab3e2f16127	user12345@gmail.com	USER12345@GMAIL.COM	user12345@gmail.com	USER12345@GMAIL.COM	0	AQAAAAIAAYagAAAAEJaxqdlze498dQa7nrNXQMTsOqA4BWtY...	Q6VII7BZC2ZLWKKBCXTV4IEUNQKJOM

Рисунок 3.2 – AspNetUsers

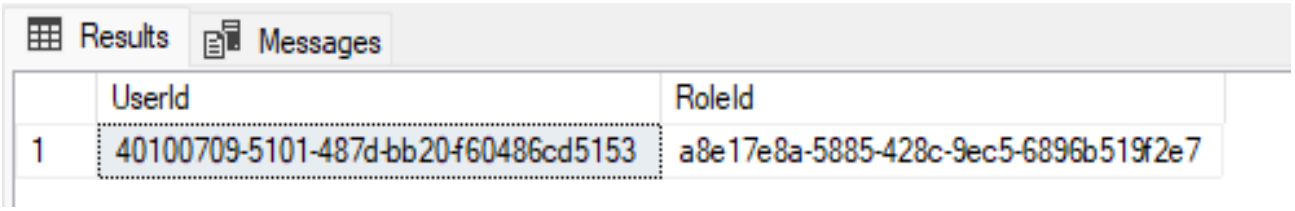
- **AspNetRoles** (Рис. 3.3) — містить ролі, які визначають права користувачів (наприклад, "Admin" або "Staff").

Results Messages

	Id	Name	NormalizedName	ConcurrencyStamp
1	a8e17e8a-5885-428c-9ec5-6896b519f2e7	Admin	ADMIN	NULL
2	b59f1347-e193-44e4-bb9c-62578242caf8	Staff	STAFF	NULL

Рисунок 3.3 - AspNetRoles

- **AspNetUserRoles** (Рис. 3.4) — зв'язує користувачів із ролями.



Results Messages

	UserId	RoleId
1	40100709-5101-487d-bb20-f60486cd5153	a8e17e8a-5885-428c-9ec5-6896b519f2e7

Рисунок 3.4 – AspNetUserRoles

- **AspNetUserClaims**, **AspNetUserLogins**, **AspNetUserTokens**, **AspNetRoleClaims** — таблиці, що забезпечують гнучке управління доступом та додатковою інформацією про користувачів.

Це стандартна інфраструктура Identity, яка дає змогу зручно керувати обліковими записами та правами.

Таблиці готельної системи

Окрім службових, у базі даних створено ряд таблиць, специфічних для доменної області – управління готелем.

Clients (Рис. 3.5):

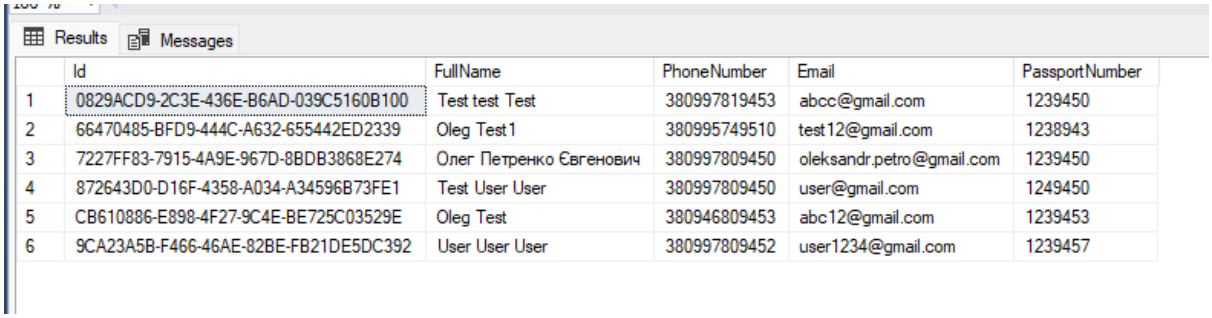
– Id (автоматично генерується)

– FullName — повне ім’я клієнта

– PhoneNumber — номер телефону

– Email — email

– PassportNumber — паспортні дані



Results Messages

	Id	FullName	PhoneNumber	Email	PassportNumber
1	0829ACD9-2C3E-436E-B6AD-039C5160B100	Test test Test	380997819453	abcc@gmail.com	1239450
2	66470485-BFD9-444C-A632-655442ED2339	Oleg Test1	380995749510	test12@gmail.com	1238943
3	7227FF83-7915-4A9E-967D-8BDB3868E274	Олег Петренко Євгенович	380997809450	oleksandr.petro@gmail.com	1239450
4	872643D0-D16F-4358-A034-A34596B73FE1	Test User User	380997809450	user@gmail.com	1249450
5	CB610886-E898-4F27-9C4E-BE725C03529E	Oleg Test	380946809453	abc12@gmail.com	1239453
6	9CA23A5B-F466-46AE-82BE-FB21DE5DC392	User User User	380997809452	user1234@gmail.com	1239457

Рисунок 3.5 – Таблиця Клієнти

Rooms (Рис. 3.6)

– Id — ідентифікатор номера

– Number — номер кімнати

– Type — тип (наприклад, одномісний, двомісний

– PricePerNight — ціна за ніч

– Status — статус (вільна, зайнята, заброньована)

Results Messages

	Id	Number	Type	PricePerNight	Status
1	336C7F6B-9714-4ED9-8EE5-2D4FD0ACD88A	110	Люкс на 2 персони	2500.00	2
2	28FF0B9D-0C08-4B9F-B87A-4E3BFFE96E40	104	Стандарт на 3 персону	350.00	2
3	B86FDF6C-080D-4795-80C8-58D65BE28360	106	Люкс на 1 персону	1000.00	2
4	A48E100D-096F-4B62-B58E-5F587FD58582	103	Стандарт на 2 персону	300.00	0
5	F08C920E-4E56-4C9A-8A71-740CE5F2A0A4	101	test	100.00	1
6	E384B7F8-8785-4BC9-AE93-804FD1CC0AAE	105	Стандарт на 4 персону	500.00	2
7	DDB5E47E-F77E-4688-AFE9-93A2A222D517	115	Люкс на 3 персони	3000.00	0
8	30A2F233-6636-49E7-B750-E4019D2F573B	102	Стандарт на 1 персону	200.00	1

Рисунок 3.6 – Таблиця Кімнати

Rooms тісно пов’язана з бронюванням і відображає наявність номерів.

Bookings (Рис. 3.7)

– Id

– ClientId — зв’язок з таблицею **Clients**

– RoomId — зв’язок з таблицею **Rooms**

– CheckInDate — дата заїзду

– CheckOutDate — дата виїзду

– CreatedAt — коли було зроблено бронювання

– Status — статус бронювання (активне, завершене)

Results Messages

	Id	ClientId	RoomId	CheckInDate	CheckOutDate	CreatedAt	Status
1	0D35E418-4287-46F6-97F3-061A110F7994	0829ACD9-2C3E-436E-B6AD-039C5160B100	28FF0B9D-0C08-4B9F-B87A-4E3BFFE96E40	2025-04-25 00:00:00.0000000	2025-05-01 00:00:00.0000000	2025-04-25 14:01:32.8816960	0
2	D07545C2-0258-4706-B75A-0B7A59A39422	9CA23A5B-F466-46AE-82BE-FB21DE5DC392	F08C920E-4E56-4C9A-8A71-740CE5F2A0A4	0001-01-01 00:00:00.0000000	0001-01-01 00:00:00.0000000	2025-04-25 10:49:44.9946124	0
3	E1703BA8-09AE-4683-8E1E-33D842E6995A	872643D0-D16F-4358-A034-A34596B73FE1	A48E100D-096F-4B62-B58E-5F587FD58582	2025-04-25 00:00:00.0000000	2025-04-28 00:00:00.0000000	2025-04-25 13:40:57.9164870	2
4	1F69BFEE-AAFF-4F62-9F0C-65D933456BD6	7227FF83-7915-4A9E-967D-8BDB3868E274	30A2F233-6636-49E7-B750-E4019D2F573B	0001-01-01 00:00:00.0000000	0001-01-01 00:00:00.0000000	2025-04-25 12:52:30.0870898	0
5	EB77FCA1-1E44-446D-B7CE-B2D3BE5F42D2	66470485-BFD9-444C-A632-655442ED2339	DDB5E47E-F77E-4688-AFE9-93A2A222D517	2025-04-26 00:00:00.0000000	2025-04-27 00:00:00.0000000	2025-04-26 22:47:48.0732452	2
6	A88E8148-BB00-4F0E-B50A-F4D03EE28B94	CB610886-E898-4F27-9C4E-BE725C03529E	336C7F6B-9714-4ED9-8EE5-2D4FD0ACD88A	2025-04-26 00:00:00.0000000	2025-05-05 00:00:00.0000000	2025-04-26 17:20:32.8332171	0
7	DD741C29-107A-42F9-8666-FF39C075237B	0829ACD9-2C3E-436E-B6AD-039C5160B100	B86FDF6C-080D-4795-80C8-58D65BE28360	2025-04-26 00:00:00.0000000	2025-04-30 00:00:00.0000000	2025-04-26 21:12:46.2565709	0

Рисунок 3.7 – Таблиця Бронювання

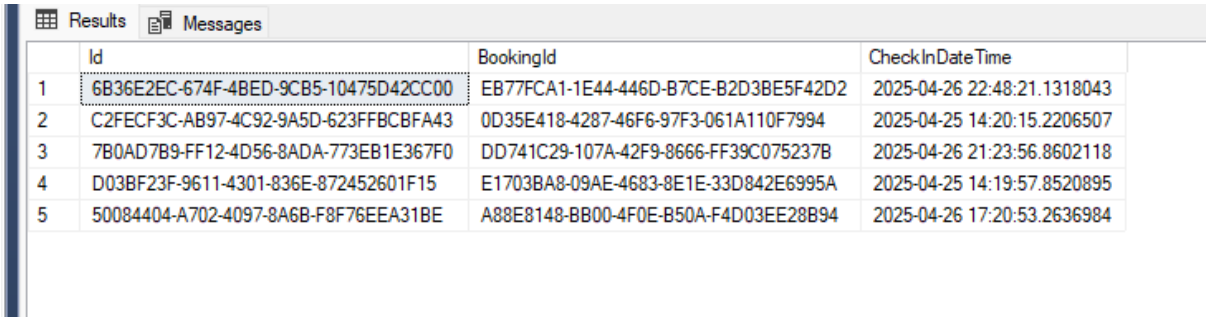
Ця таблиця є центральною у процесі бронювання. Вона об'єднує клієнта і номер, а також вказує період перебування.

CheckIns (Рис. 3.8)

– Id

– BookingId — посилання на бронювання

– CheckInDateTime — дата і час фактичного заселення.



Results Messages

	Id	BookingId	CheckInDateTime
1	6B36E2EC-674F-4BED-9CB5-10475D42CC00	EB77FCA1-1E44-446D-B7CE-B2D3BE5F42D2	2025-04-26 22:48:21.1318043
2	C2FECF3C-AB97-4C92-9A5D-623FFBCBFA43	0D35E418-4287-46F6-97F3-061A110F7994	2025-04-25 14:20:15.2206507
3	7B0AD7B9-FF12-4D56-8ADA-773EB1E367F0	DD741C29-107A-42F9-8666-FF39C075237B	2025-04-26 21:23:56.8602118
4	D03BF23F-9611-4301-836E-872452601F15	E1703BA8-09AE-4683-8E1E-33D842E6995A	2025-04-25 14:19:57.8520895
5	50084404-A702-4097-8A6B-F8F76EEA31BE	A88E8148-BB00-4F0E-B50A-F4D03EE28B94	2025-04-26 17:20:53.2636984

Рисунок 3.8 – Таблиця ChekIns

Фіксує момент, коли клієнт реально заселився у номер.

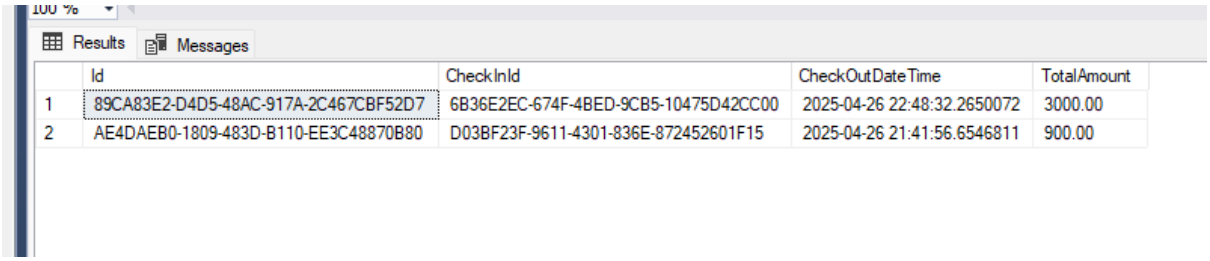
CheckOuts (Рис. 3.9)

– Id

– CheckInId — зв’язок із заселення

– CheckOutDateTime — фактична дата виїзду

– TotalAmount — сума до сплати.



100 %

Results Messages

	Id	CheckInId	CheckOutDateTime	TotalAmount
1	89CA83E2-D4D5-48AC-917A-2C467CBF52D7	6B36E2EC-674F-4BED-9CB5-10475D42CC00	2025-04-26 22:48:32.2650072	3000.00
2	AE4DAEB0-1809-483D-B110-EE3C48870B80	D03BF23F-9611-4301-836E-872452601F15	2025-04-26 21:41:56.6546811	900.00

Рисунок 3.8 – Таблиця ChekOuts

Ця таблиця завершує життєвий цикл бронювання, підраховуючи суму та фіксуючи факт виселення.

Діаграма бази даних

Для кращої візуалізації побудована **ER-діаграма** (Entity Relationship Diagram), наведена у **Додатку А, рисунок А.2**. , яка демонструє взаємозв’язки між таблицями, які наведені вище :

- Клієнт може мати багато бронювань.
- Кожне бронювання пов’язане з одним номером і одним клієнтом.
- Бронювання може мати лише одне заселення (CheckIn), яке в свою чергу може мати лише одне виселення (CheckOut).

Це дозволяє гарантувати цілісність даних і виключає можливість логічних помилок.

Особливості та гнучкість

Завдяки Entity Framework Code First підхід дозволяє:

- Легко оновлювати модель бази даних.
- Використовувати міграції для поступового розширення функціоналу.
- Зберігати чистоту та наочність структури.
- Гарантувати підтримку будь-якого сучасного SQL сервера.

3.3 Реалізація функцій системи

Під час розробки веб-додатку управління готелем особливу увагу було приділено серверній частині, адже саме вона є серцем системи. Серверна частина відповідає за:

- зберігання даних (клієнти, бронювання, номери);
- забезпечення зв'язків між сутностями;
- реалізацію бізнес-логіки;
- обробку запитів користувача.

Для створення серверної частини було обрано **ASP.NET Core MVC + Entity Framework Core + SQL Server**.

Це дозволило створити структуровану, масштабовану і надійну систему.

Архітектура

Архітектура серверної частини побудована за класичною схемою:

- Controllers (Обробка запитів)
- Models (Опис сутностей БД)

- Data (ApplicationDbContext) (Доступ до бази)
- Views

Моделі даних (Models)

ApplicationUser.cs (Користувачі)

Розширена модель користувача від Identity. Додано поле FullName:

```
namespace WebApplication2.Models
{
    Ссылок: 11
    public class ApplicationUser : IdentityUser
    {
        Ссылок: 3
        public string? FullName { get; set; }
    }
}
```

Це дозволяє зберігати ім'я користувача прямо в таблиці користувачів.

Client.cs (Клієнти)

```
public class Client
{
    Ссылок: 9
    public Guid Id { get; set; }
    Ссылок: 8
    public string FullName { get; set; } = string.Empty;
    Ссылок: 4
    public string PhoneNumber { get; set; } = string.Empty;
    Ссылок: 4
    public string Email { get; set; } = string.Empty;
    Ссылок: 4
    public string PassportNumber { get; set; } = string.Empty;

    Ссылок: 0
    public ICollection<Booking> Bookings { get; set; } = new List<Booking>();
}
```

Таблиця Client зберігає детальну інформацію про гостей готелю.

Room.cs (Номери)

```
public enum RoomStatus
{
    Free,
    Booked,
    Occupied
}

Ссылок: 10
public class Room
{
    Ссылок: 10
    public Guid Id { get; set; }
    Ссылок: 11
    public string Number { get; set; } = string.Empty;
    Ссылок: 7
    public string Type { get; set; } = string.Empty;
    Ссылок: 5
    public decimal PricePerNight { get; set; }
    Ссылок: 14
    public RoomStatus Status { get; set; } = RoomStatus.Free;

    Ссылок: 0
    public ICollection<Booking> Bookings { get; set; } = new List<Booking>();
}
```

Номери мають статус: вільний / зайнятий / заброньований.

Booking.cs (Бронювання)

```
public class Booking
{
    Ссылок: 4
    public Guid Id { get; set; }

    Ссылок: 1
    public Guid ClientId { get; set; }
    Ссылок: 7
    public Client? Client { get; set; }


    Ссылок: 1
    public Guid RoomId { get; set; }
    Ссылок: 21
    public Room? Room { get; set; }


    Ссылок: 8
    public DateTime CheckInDate { get; set; }
    Ссылок: 7
    public DateTime CheckOutDate { get; set; }
    Ссылок: 2
    public DateTime CreatedAt { get; set; } = DateTime.Now;
    Ссылок: 5
    public BookingStatus Status { get; set; } = BookingStatus.Active;

    Ссылок: 9
    public CheckIn? CheckIn { get; set; }
}
```

Тут важливим є зв'язок "один до одного" між бронюванням і CheckIn.

CheckIn.cs (Поселення)

```
public class CheckIn
{
    Ссылок: 9
    public Guid Id { get; set; }

    Ссылок: 2
    public Guid BookingId { get; set; }
    Ссылок: 25
    public Booking Booking { get; set; }

    Ссылок: 1
    public DateTime CheckInDateTime { get; set; }


    Ссылок: 1
    public CheckOut? CheckOut { get; set; }
}
```

CheckIn фіксує дату фактичного поселення.

CheckOut.cs (Виселення)

```
public class CheckOut
{
    Ссылок: 2
    public Guid Id { get; set; }

    Ссылок: 5
    public Guid CheckInId { get; set; }
    Ссылок: 7
    public CheckIn CheckIn { get; set; }

    Ссылок: 4
    public DateTime CheckOutDateTime { get; set; }
    Ссылок: 3
    public decimal TotalAmount { get; set; }
}
```

Запис виселення містить суму до сплати за всі дні.

ApplicationDbContext.cs (Контекст)

Контекст вказує таблиці:

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    Ссылок: 0
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
        : base(options) { }

    Ссылок: 10
    public DbSet<Client> Clients { get; set; }
    Ссылок: 11
    public DbSet<Room> Rooms { get; set; }
    Ссылок: 6
    public DbSet<Booking> Bookings { get; set; }
    Ссылок: 3
    public DbSet<CheckIn> CheckIns { get; set; }
    Ссылок: 5
    public DbSet<CheckOut> CheckOuts { get; set; }

    Ссылок: 0
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        base.OnModelCreating(modelBuilder);

        modelBuilder.Entity<Booking>()
            .HasOne(b => b.CheckIn)
            .WithOne(c => c.Booking)
            .HasForeignKey<CheckIn>(c => c.BookingId);

        modelBuilder.Entity<CheckIn>()
            .HasOne(ci => ci.CheckOut)
            .WithOne(co => co.CheckIn)
            .HasForeignKey<CheckOut>(co => co.CheckInId);
    }
}
```

Це дозволяє коректно будувати JOIN-запити в БД.

Контролери (Controllers)

BookingsController.cs

```
public async Task<IActionResult> List()
{
    var bookings = await _context.Bookings
        .Include(b => b.Client)
        .Include(b => b.Room)
        .Include(b => b.CheckIn)
        .OrderByDescending(b => b.CreatedAt)
        .ToListAsync();

    return View(bookings);
}
```

Для створення нового бронювання відповідає фрагмент коду, який наведений в **Додатку Б**.

CheckInsController.cs

У системі реалізовано контролер CheckInsController (реалізація коду наведена у **Додатку Б**), який забезпечує управління процесом поселення клієнтів у готель. Даний контролер є частиною архітектури ASP.NET Core MVC та взаємодіє з базою даних через контекст ApplicationDbContext, що реалізує шаблон доступу до даних Entity Framework Core.

Контролер містить два основні методи:

- Index() – призначений для отримання списку бронювань, запланованих на поточну дату, які ще не були оброблені (тобто не відбувся факт поселення). Метод здійснює фільтрацію активних бронювань, використовуючи LINQ-запити з підключенням супутніх даних про клієнтів та номери. Отриманий перелік відображається у View для подальших дій адміністратора.
- CheckIn(Guid id) – обробляє запит на підтвердження факту заселення клієнта за переданим ідентифікатором бронювання. У процесі виконання цього методу система перевіряє існування відповідного запису бронювання та пов’язаного номеру.

У разі успішного знаходження:

1. оновлюється статус номера на "Зайнятий";
2. створюється новий запис про поселення з фіксацією дати і часу;
3. зв’язок між бронюванням і фактом поселення зберігається в базі даних.

CheckOutsController.cs

Контролер CheckOutController дозволяє системі надійно обробляти процедуру виїзду клієнтів, забезпечуючи коректне збереження даних про проживання, розрахунок вартості послуг і звільнення номерного фонду для подальшого використання. Завдяки цьому модулю забезпечується повний цикл обслуговування гостей, що є невід'ємною частиною автоматизації діяльності готелю. Реалізація коду наведена у **Додатку Б**.

AdminController.cs

Контролер для Роботи з користувачами: додавання ролей, видалення та редагування. Реалізація коду наведена у **Додатку Б**.

Серверна частина проєкту реалізує повний цикл управління готелем:

- Додавання клієнтів.
- Бронювання номерів.
- Поселення клієнтів.
- Виселення та розрахунок суми.
- Адміністрування користувачів та ролей.

Усі процеси пов'язані між собою і дозволяють адміністратору працювати ефективно.

3.4 Інструкція користувача

В даному підрозділі наведено повну покрокову інструкцію щодо використання основного функціоналу інформаційної системи **HotelApp**. Кожен модуль супроводжується скриншотами, описом і поясненнями, щоб користувач міг легко зрозуміти призначення та можливості інтерфейсу.

1. **Вхід в систему (Авторизація)** наведено на рисунку 3.9**.**

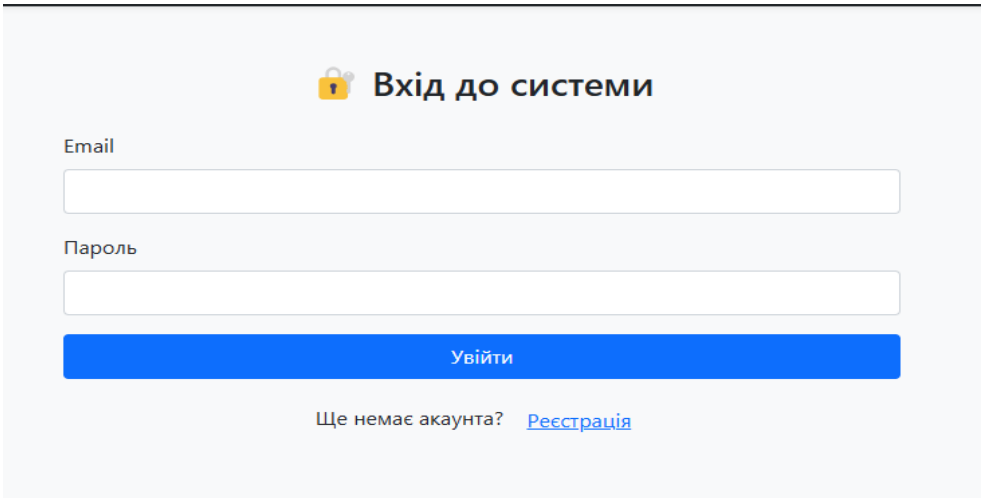


Рисунок 3.9 – вхід до системи

Що робимо:

1) Переходимо на сторінку входу.
2) Вказуємо Email та Пароль.

3) Натискаємо кнопку "Увійти".

Результат:

1) Якщо дані вірні — система перенаправить вас на головну сторінку.

2) Якщо облікового запису немає — можна перейти за посиланням "Реєстрація" для створення нового облікового запису(Рис. 3.10).

3) AccountController → Login (GET + POST) (див. **Додаток Б**).

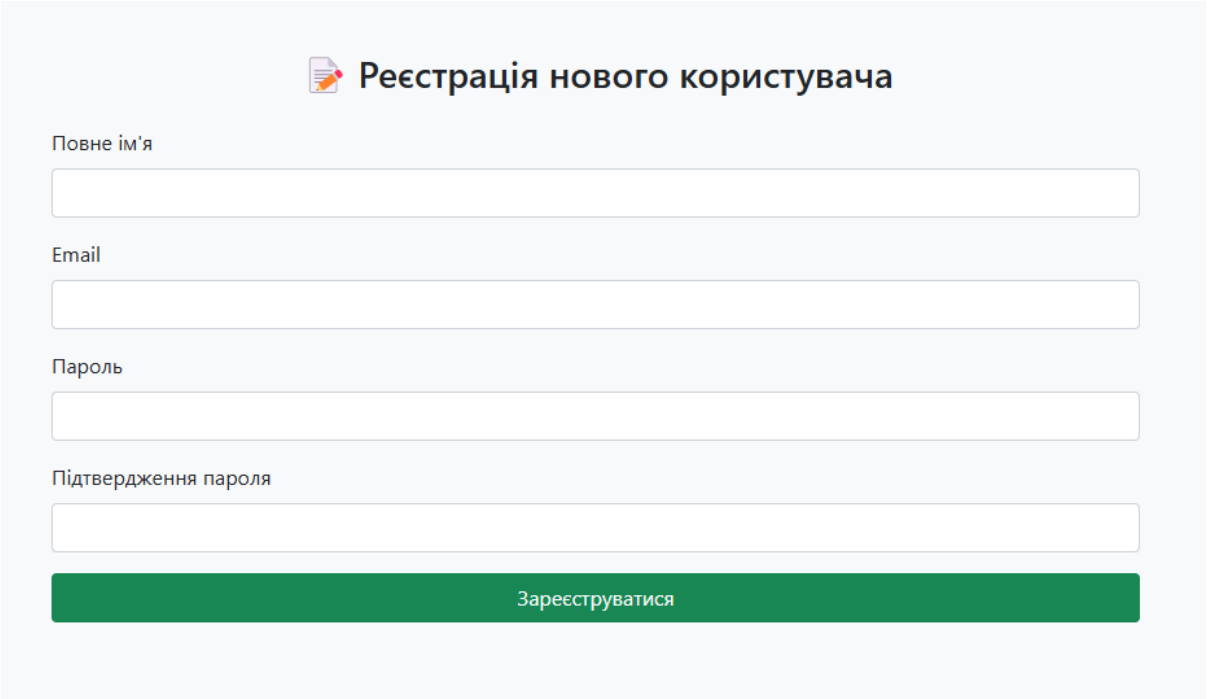


Рисунок 3.10 - Реєстрація нового користувача

Що робимо:

1) Натискаємо на посилання "Реєстрація" на сторінці входу.

2) Заповнюємо форму:

3) Повне ім'я

4) Email

5) Пароль

6) Підтвердження пароля

7) Натискаємо "Зареєструватися".

Результат:

1) Новий обліковий запис буде створено та збережено в системі.

2) Можна увійти до системи.

AccountController → Register (GET + POST) (див. **Додаток Б**).

2. **Управління клієнтами** наведено на рисунку 3.11.

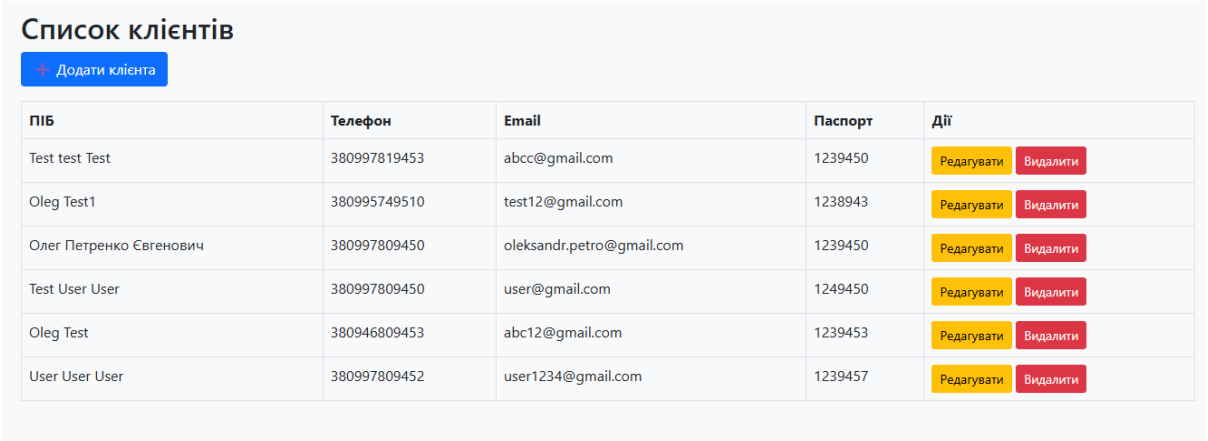


Рисунок 3.11 – управління клієнтами

Що робимо:

1) Вибираємо в меню пункт "Клієнти".
2) На сторінці бачимо список усіх клієнтів.
3) Натискаємо **"Додати клієнта"**, щоб створити новий запис (Рис. 3.12).

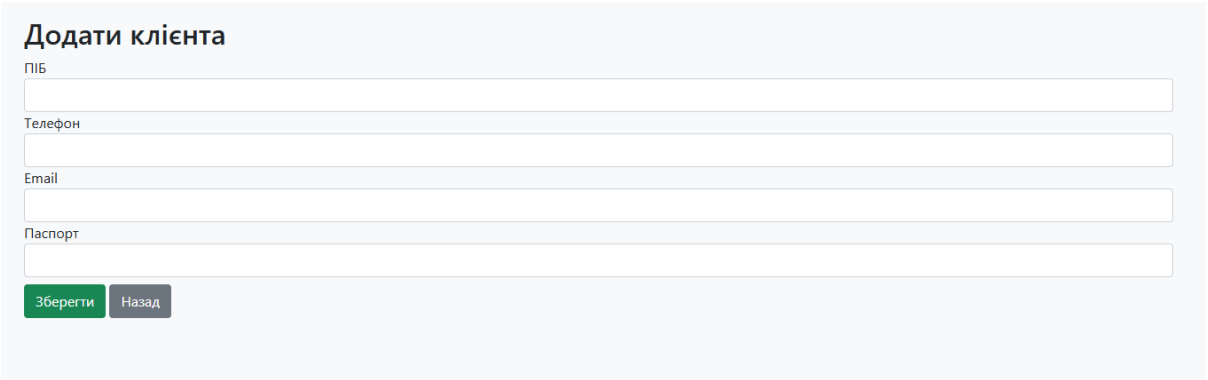


Рисунок 3.12 – додати клієнта

Для редагування натискаємо **"Редагувати"** (Рис. 3.13).

Рисунок 3.13 – редагування клієнта

Для видалення — **"Видалити"**.

Результат:

1) Усі зміни миттєво фіксуються в базі.
2) Клієнти стають доступними для бронювання.

3. Наступний етап це **- Створення бронювання,** яке наведенно на рисунку 3.14.

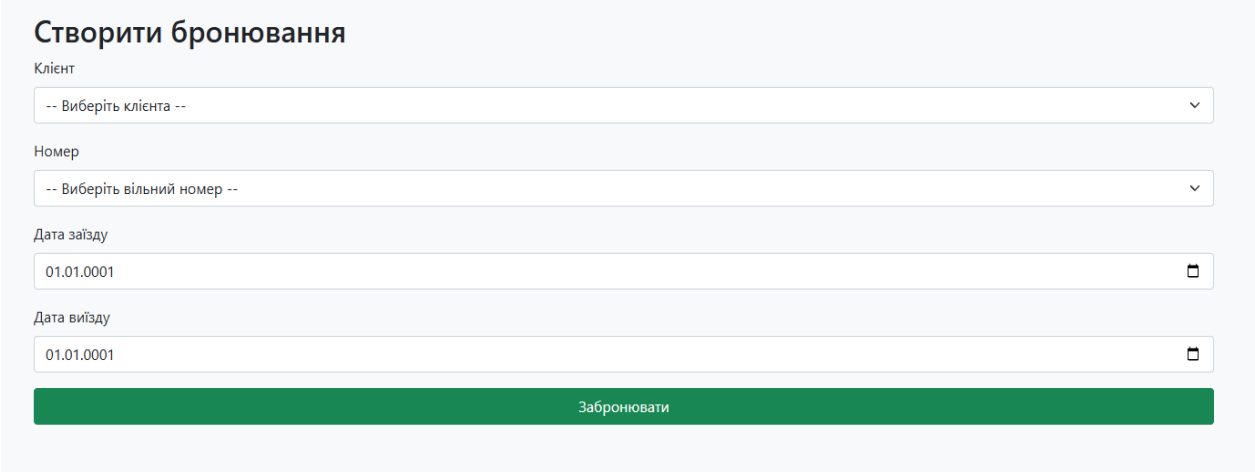


Рисунок 3.14 - Створення бронювання

Обираємо меню **"Бронювання"** → **"Створити бронювання"** (Рис. 3.15)

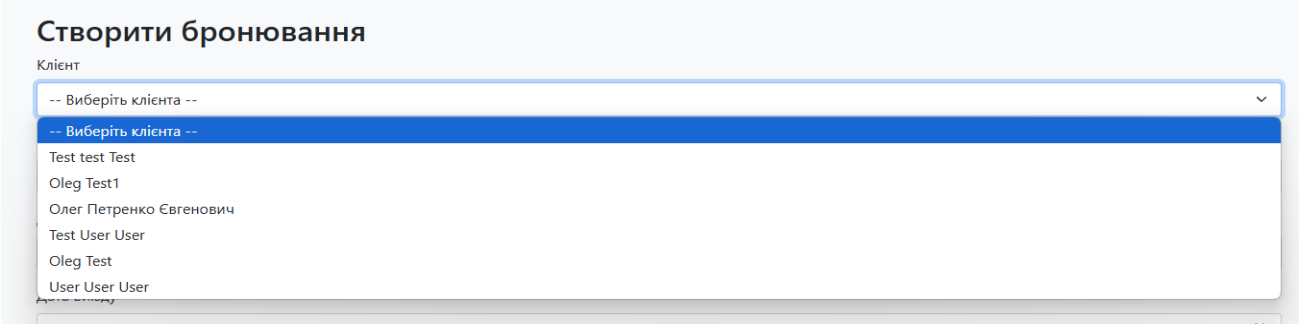


Рисунок 3.15 – Інструкція оформлення бронювання

1) Вибираємо клієнта.
2) Вибираємо вільний номер.
3) Вказуємо дату заїзду та виїзду.
4) Натискаємо **"Забронювати"**.

Результат:

1) Створюється запис в таблиці Bookings.
2) Статус номера автоматично зміниться на **"Забронювано"**.

4. **Заселення (CheckIn)** на рисунку 3.16.

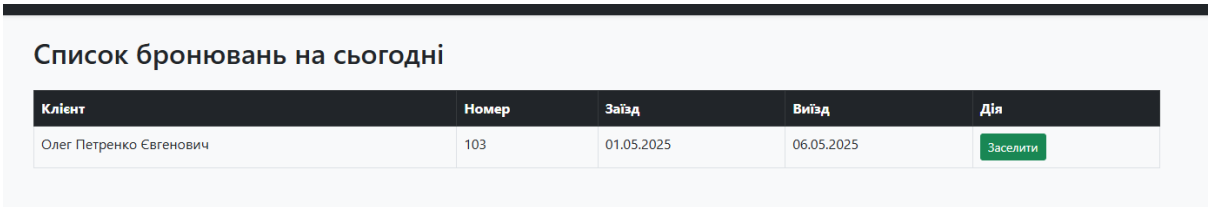


Рисунок 3.16 – Меню Заселення

Що робимо:

1) Переходимо в розділ **"Поселення"**.
2) Вибираємо бронювання для заселення.
3) Натискаємо **"Заселити"** – результат показано на рисунку 3.17.

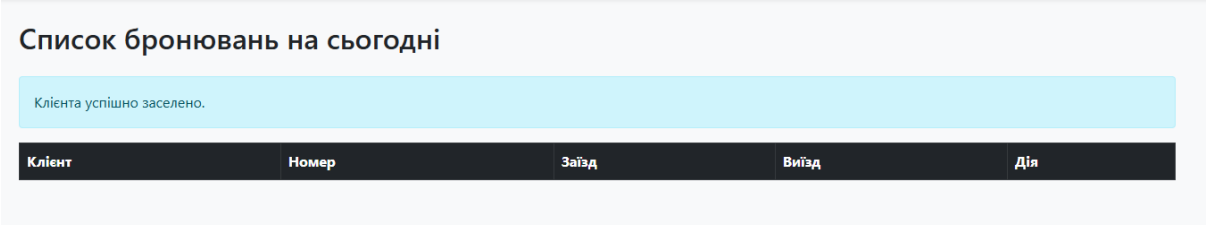


Рисунок 3.17 – Результат нажаття на кнопку Заселити

Результат:

1) Створюється запис в таблиці CheckIns.
2) Номер стає **"Заселеним"**.

5. **Виселення (CheckOut),** інструкція наведена на рисунку 3.18. Результат Виселення показаний на рисунку 3.19.

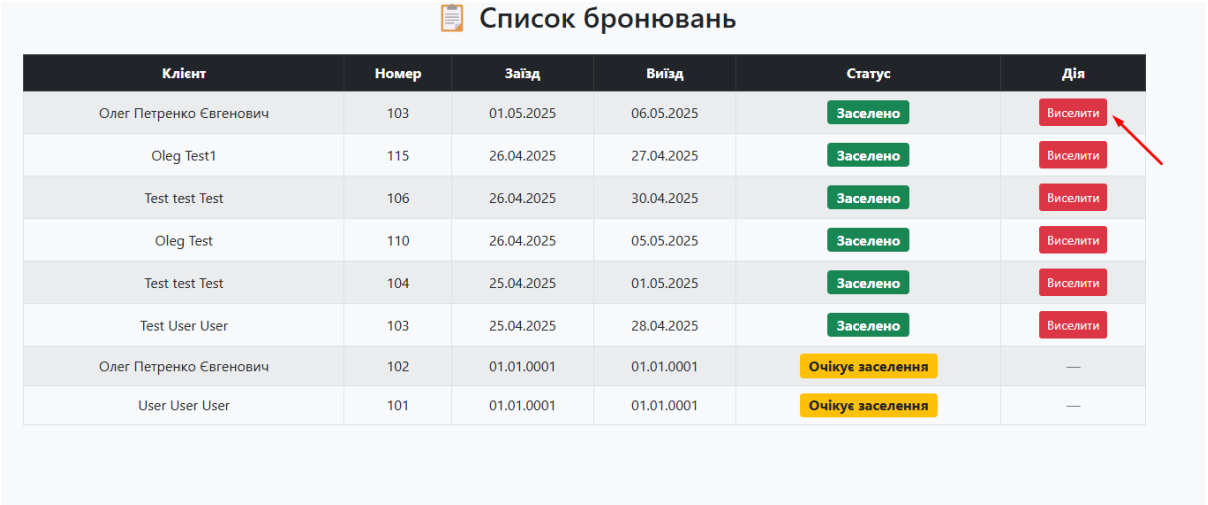


Рисунок 3.18 – Оформлення Виселення

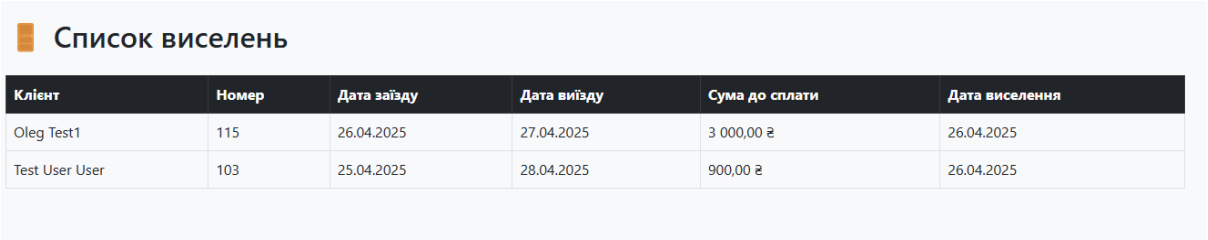


Рисунок 3.19 – Список виселень

Що робимо:

1) Вибираємо розділ **"Виселення"**.
2) Натискаємо **"Виселити"** біля клієнта, що проживає.
3) Система обчислить суму до сплати.

Результат:

1) Номер стане вільним.
2) Бронювання буде завершено.
3) З'явиться запис в таблиці CheckOuts.

6. **Робота з номерним фондом** показана на рисунку 3.20 та 3.21.

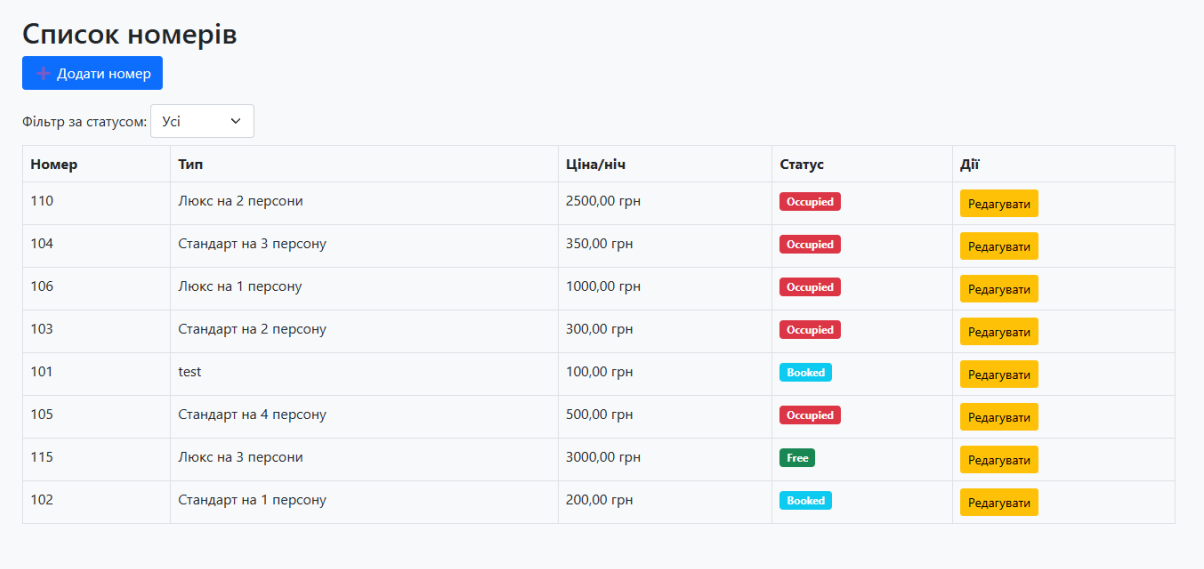


Рисунок 3.20 – Номерний форд

Що робимо:

1) Переходимо в меню **"Номери"**.
2) Додаємо нові або редагуємо існуючі.
3) Вказуємо номер, тип, ціну, статус.

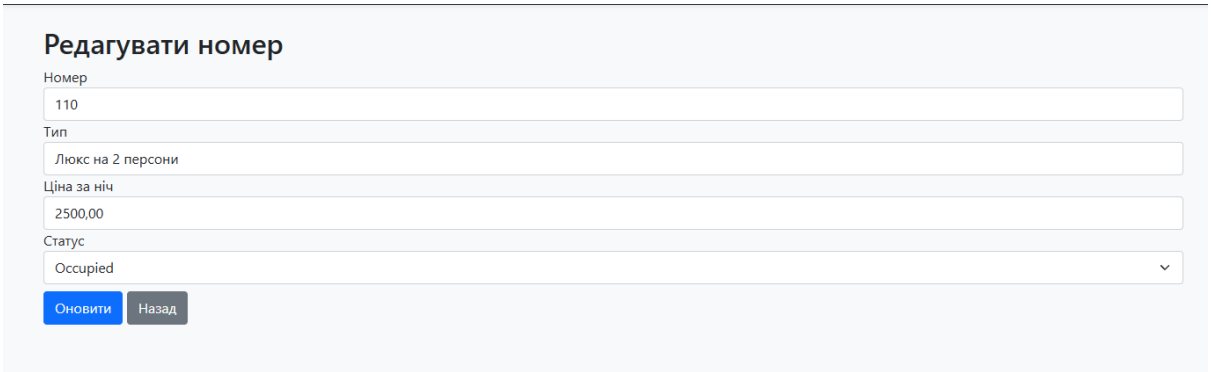


Рисунок 3.21 – Взаємодія з номерами

Результат:

Інформація про номери оновиться для бронювання та поселення.

7. **Панель адміністратора**

Що робимо:

Переходимо в розділ **"Адміністрування"** (Рис. 3.22).

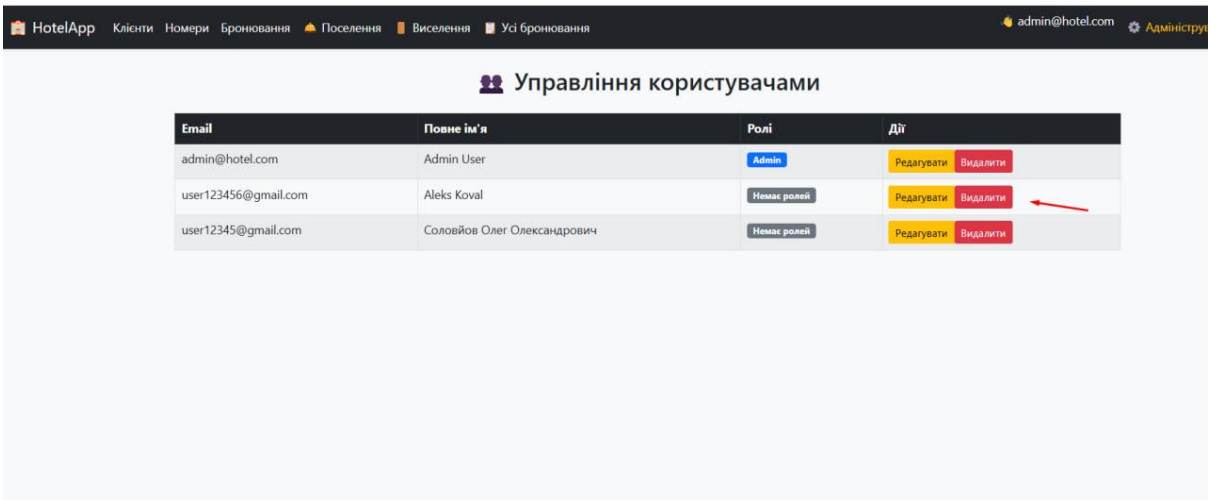


Рисунок 3.22 – Панель Адміністатора

Доступні можливості:

1) **Користувачі** — управління ролями, видалення/редагування. Видалення користувача показана на рисунку 3.23.

2) **Резервне копіювання** — створення резервної копії бази (поки опціонально).

3) **Налаштування** — поки заглушка.

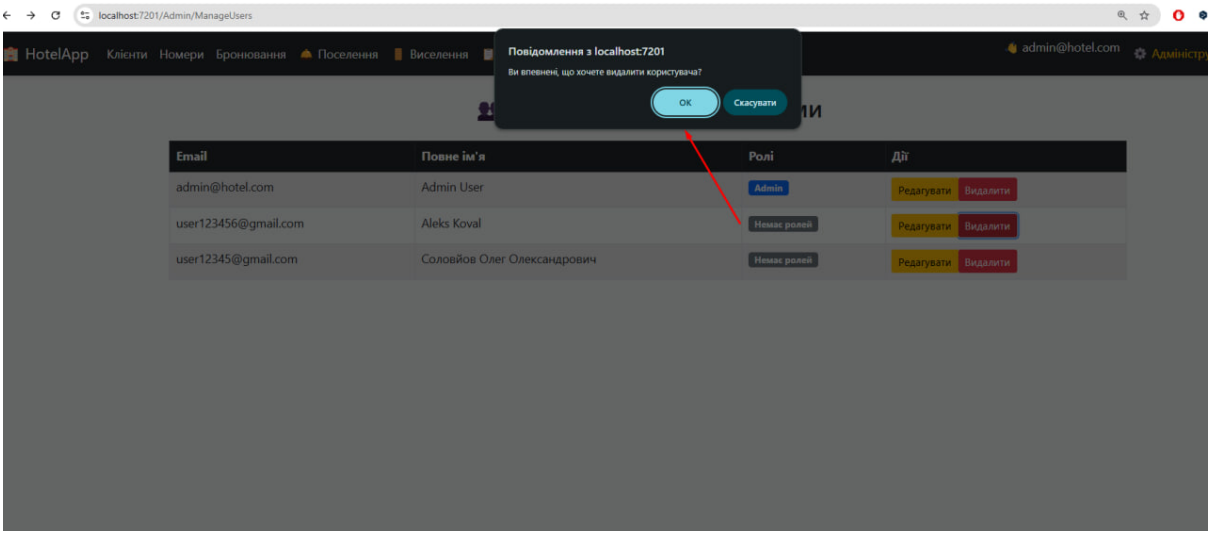


Рисунок 3.23 – Видалення користувача

Результат:

Адміністратор керує правами доступу, додає і видаляє користувачів. Демостація на рисунку 3.24.

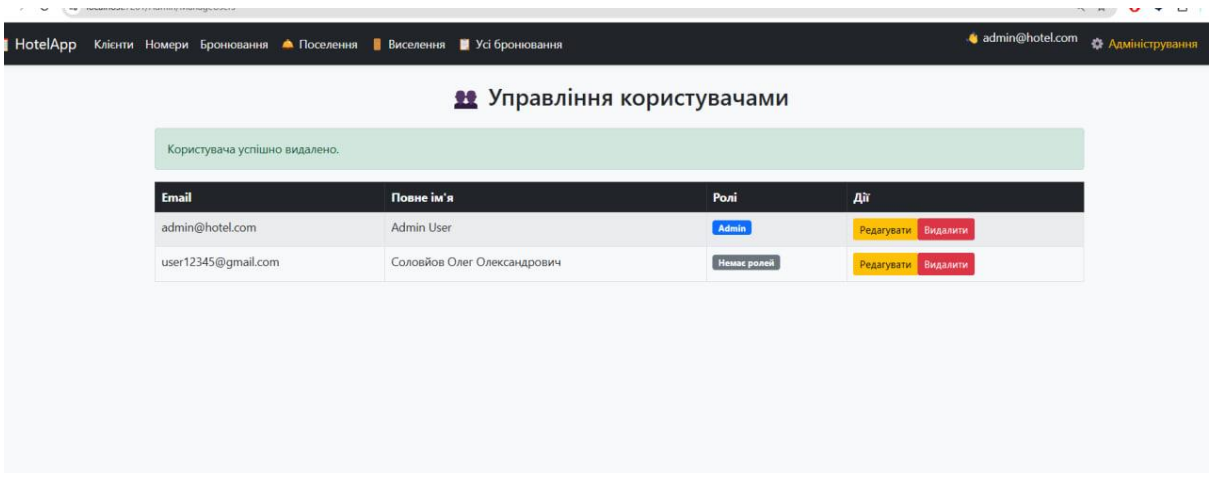


Рисунок 3.24 – Результат

Система має повністю інтуїтивний і зручний інтерфейс. Усі етапи — від створення користувачів і клієнтів до бронювання, поселення і виселення — реалізовано на основі сучасних веб-технологій ASP.NET MVC Razor Pages. Для кожного процесу створено відповідний контроллер та View. Користувачеві не потрібно вивчати інструкції — достатньо слідувати логічним підказкам на сторінках системи.

3.5 Тестування програмного продукту

3.5.2 Загальні відомості про тестування

Тестування програмного продукту є важливим етапом розробки, що дозволяє виявити помилки системи [29, 30]. У процесі тестування було перевірено працездатність функціональних характеристик розробленої інформаційної системи управління замовленнями та поселенням клієнтів у готелі.

Тестування виконувалося на таких рівнях:

- Модульне тестування – перевірка окремих компонентів і методів системи.
- Інтеграційне тестування – перевірка взаємодії між модулями.
- Системне тестування – оцінка роботи всієї системи загалом.

3.5.2 Тест-план тестування

Мета тестування: Перевірити коректність роботи всіх функціональних модулів системи, зокрема бронювання номерів, реєстрації клієнтів, обробки заселення та виселення, а також формування звітів.

Середовище тестування:

- Операційна система: Windows 10 Pro
- Браузер: Google Chrome 123.0
- Серверна платформа: .NET 6, SQL Server 2019
- Інтерфейс користувача: ASP.NET Core MVC (Razor Pages)

Об'єкти тестування:

- Модуль реєстрації бронювання
- Модуль реєстрації поселення
- Модуль реєстрації виселення
- Генерація звітної документації
- Авторизація та управління доступом

У таблиці 3.1 наведено результати проведеного тестування функціональних модулів системи, включаючи очікувану та фактичну поведінку кожного елемента.

Таблиця 3.1 – Проведення тестування

№	Назва тесту	Очікуваний результат	Фактичний результат	Статус
1	Додавання нового бронювання	Бронювання зберігається в базі, відображається у списку	Відповідає очікуванню	Успішно
2	Перегляд бронювань на поточну дату	Відображаються актуальні бронювання	Відповідає очікуванню	Успішно
3	Заселення клієнта	Створюється запис про заселення, змінюється статус номера	Відповідає очікуванню	Успішно
4	Повторне заселення вже заселеного	Відображається повідомлення про помилку	Відповідає очікуванню	Успішно
5	Виселення клієнта	Створюється запис про виселення, статус номера стає вільним	Відповідає очікуванню	Успішно
6	Повторне виселення клієнта	Відображається повідомлення про помилку	Відповідає очікуванню	Успішно
7	Генерація фінансового звіту	Формується звіт з даними про доходи	Відповідає очікуванню	Успішно
8	Некоректний логін або пароль	Відображається повідомлення про помилку доступу	Відповідає очікуванню	Успішно
9	Доступ до функцій без авторизації	Перенаправлення на сторінку входу	Відповідає очікуванню	Успішно

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було досягнуло поставленої мети – розробка інформаційної системи управління замовленнями та поселенням клієнтів в готель, яка забезпечує автоматизацію основних бізнес-процесів підприємства.

Вирішено всі поставлені завдання, зокрема:

- проведено аналіз сучасних інформаційних технологій і ринку програмного забезпечення для готелів;
- сформульовано технічне завдання, спроєктовано архітектуру та базу даних інформаційної системи;
- реалізовано основні функціональні модулі, включаючи обробку бронювань, реєстрацію поселення та виселення клієнтів, формування звітів;
- проведено тестування системи, яке підтвердило її працездатність і відповідність технічним вимогам.

Розроблена система дозволяє вирішувати такі практичні задачі:

- автоматизоване ведення бази клієнтів і бронювань;
- управління номерним фондом готелю в реальному часі;
- реєстрацію фактів поселення та виселення клієнтів;
- розрахунок вартості послуг і формування фінансових звітів;
- контроль доступу користувачів відповідно до їхніх ролей;
- інтеграцію з онлайн-сервісами бронювання та платіжними системами.

Очікуваний ефект від впровадження системи полягає у:

- підвищенні швидкості обробки замовлень і зниженні кількості помилок персоналу;
- оптимізації фінансового обліку та скороченні витрат на паперовий документообіг;
- покращенні якості обслуговування клієнтів завдяки зручності і швидкості процедур поселення;

- підвищенні конкурентоспроможності готелю за рахунок впровадження сучасних інформаційних технологій.

Система є масштабованою та може бути доповнена новими модулями, такими як управління додатковими послугами, система лояльності для постійних клієнтів або мобільний додаток для зручності взаємодії з користувачами.

Рекомендації щодо практичного використання:

- впровадити систему на готельних підприємствах для підвищення ефективності управління;
- використовувати розроблену архітектуру як базову для розширення функціоналу;
- забезпечити навчання персоналу для ефективної експлуатації програмного забезпечення.

За результатами тестування та апробації на тестових даних система продемонструвала стабільну роботу та готовність до впровадження в реальне середовище. Практичне впровадження може бути здійснено на основі подальшого укладання договору між розробником і замовником із зазначенням умов підтримки та супроводу системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Booking.com. Готель «Гірське повітря 2», Яремче [Електронний ресурс]. – Режим доступу: https://www.booking.com/ (дата звернення: 02.05.2025).
2. Організація готельного господарства: підручник / Байлік, І. М. Писаревський; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2015. – 329 с
3. Дубодєлова, А. В., Л. С. Лісовська, and А. О. Нищота. "Система управління якістю обслуговування клієнтів готельного підприємства." *Вісник Національного університету Львівська політехніка. Проблеми економіки та управління* 754 (2013): 159-165.
4. Катков, Ю. І., В. П. Лисак, and В. В. Вишнівський. "Розроблення класифікації інструментів системного адміністрування серверів." *Зв'язок* 2 (2022): 12-21.
5. України, Закон. "Про захист персональних даних." *Відомості Верховної Ради України* 34 (2010): 2297-17.
6. **Database server** [Електронний ресурс] // Wikipedia : the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Database_server – Назва з екрана. – Дата звернення: 09.05.2025.
7. **OtelMS**. OtelMS Front Desk – програмне забезпечення для управління готелем [Електронний ресурс]. – Режим доступу: https://otelms.com/uk/otelms-front-desk/ (дата звернення: 09.05.2025).
8. MySQL. Офіційний сайт. URL: https://www.mysql.com (дата звернення: 02.05.2025).
9. **PostgreSQL Global Development Group**. PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/ (дата звернення: 09.05.2025).
10. Microsoft Corporation. Excel – програмне забезпечення для роботи з електронними таблицями. URL: https://www.microsoft.com/uk-ua/microsoft-365/excel (дата звернення: 02.05.2025).

11. **[Без автора].** Course Project Documentation [Електронний ресурс] // CollegeSidekick. – Режим доступу: https://www.collegesidekick.com/study-docs/5503854 – Назва з екрана. – Дата звернення: 15.05.2025.
12. **Hire1**. API – що це таке? Просте пояснення терміну [Електронний ресурс]. – Режим доступу: https://hire1.com.ua/glossary/api (дата звернення: 09.05.2025).
13. RoomRaccoon. Офіційний сайт [Електронний ресурс]. – Режим доступу: https://roomraccoon.com (дата звернення: 02.05.2025).
14. Cloudbeds. Офіційний сайт [Електронний ресурс]. – Режим доступу: https://www.cloudbeds.com/ (дата звернення: 02.05.2025).
15. **Wikipedia**. Customer relationship management [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Customer_relationship_management (дата звернення: 09.05.2025).
16. Гаврилюк О. В., Гончаренко Л. М. *Системний аналіз та проєктування інформаційних систем*: навч. посібник. – Харків: ХНУРЕ, 2019. – 264 с.
17. Балабанов І. Т. *Аналіз і моделювання бізнес-процесів*: навч. посібник. – Київ: КНЕУ, 2020. – 240 с.
18. **Davies, G.** Customer Relationship Management. – University of Gloucestershire, 1998. – 56 p. – Режим доступу: https://core.ac.uk/download/pdf/11320265.pdf (дата звернення: 09.05.2025).
19. **Microsoft**.Overview of ASP.NET Core MVC [Електронний ресурс].–Режим доступу: https://learn.microsoft.com/en_us/aspnet/core/mvc/overview?view=aspnetcore-9.0 (дата звернення: 08.05.2025).
20. **Microsoft**. SQL Server Management Studio (SSMS) documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/ssms/sql-server-management-studio-ssms?view=sql-server-ver16 (дата звернення: 08.05.2025).
21. Microsoft. SQL Server Management Studio (SSMS) Documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/ssms/sql-server-management-studio-ssms (дата звернення: 09.05.2025).

22. **Wikipedia**. HTTPS [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/HTTPS (дата звернення: 09.05.2025).
23. Жанна М., Софія К. Міжнародні стандарти інформаційної безпеки. *Інформація, комунікація, суспільство 2024*, Львів, Україна, 23–25 трав. 2024 / ред.: М. О., Б. Ю. ; Національний університет «Львівська політехніка», Донецький національний університет імені Василя Стуса. Львів : Видавництво Львівської політехніки, 2024. С. 35–36.
24. Кунда, Н. Т., and Н. М. Крамарчук. "Сучасні засоби управління проектами." *Управління проектами, системний аналіз і логістика. Технічна серія* 9 (2012): 89-94.
25. Microsoft. Migrations (Code First) – Entity Framework 6 [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/ef/ef6/modeling/code-first/migrations/ (дата звернення: 09.05.2025).
26. Microsoft. Entity Framework Migrations (Code First) [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/ef/ef6/modeling/code-first/migrations/ (дата звернення: 09.05.2025).
27. Microsoft. Entity Framework Core Documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/ef/core/ (дата звернення: 09.05.2025).
28. Microsoft. ASP.NET Core Security Documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/aspnet/core/security/ (дата звернення: 09.05.2025).
29. Авраменко А.С., Авраменко В.С., Косенюк Г.В. Тестування програмного забезпечення. Навчальний посібник. – Черкаси: ЧНУ імені Богдана Хмельницького, 2017. – 284 с.
30. **Foxminded.** Тестування програмного забезпечення [Електронний ресурс] // Foxminded.ua. – Режим доступу: https://foxminded.ua/testuvannia-prohramnoho-zabezpechennia/ – Назва з екрана. – Дата звернення: 15.05.2025.

ДОДАТКИ

Додаток А. Загальна схема архітектури та діаграма бази даних

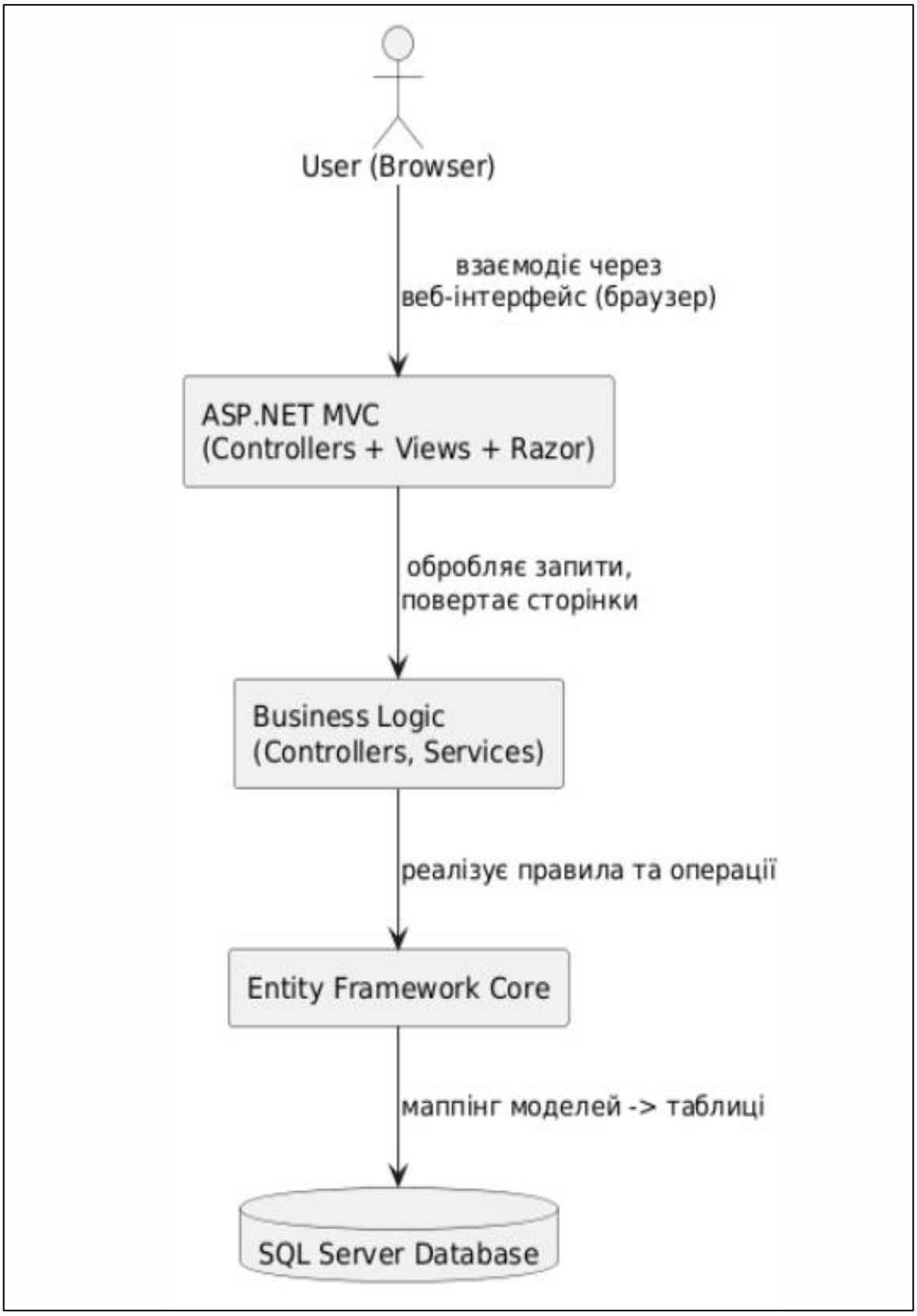


Рисунок А.1 – Загальна схема архітектури

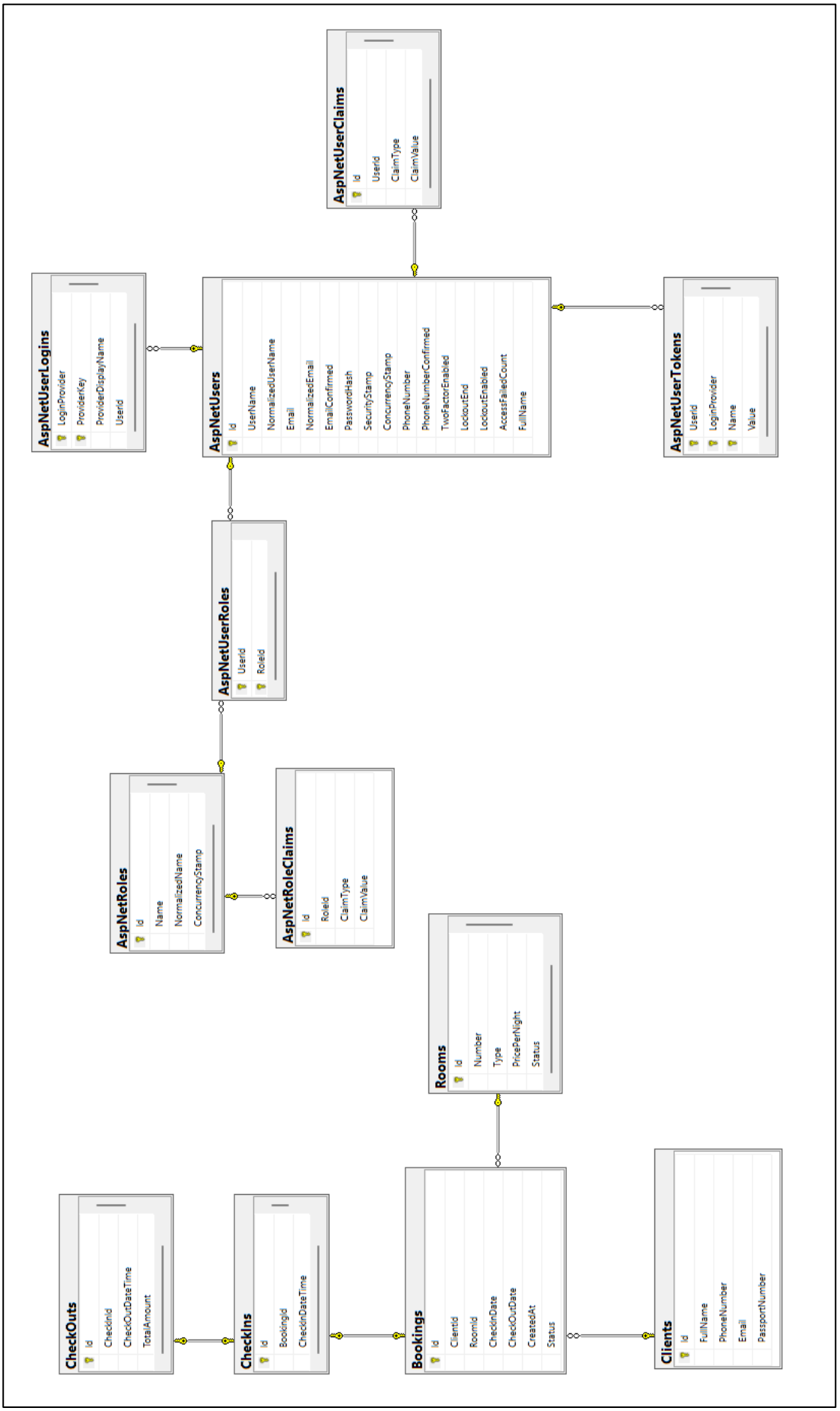


Рисунок А.2 – Діаграма бази даних

Додаток Б. Фрагменти коду програми

Фрагмент коду, який відображає відповідні таблиці в базі даних

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using System.Reflection.Emit;
using WebApplication2.Models;

namespace WebApplication2.Data
{
    Ссылок: 18
    public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
    {
        Ссылок: 0
        public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
            : base(options) { }

        Ссылок: 10
        public DbSet<Client> Clients { get; set; }
        Ссылок: 11
        public DbSet<Room> Rooms { get; set; }
        Ссылок: 6
        public DbSet<Booking> Bookings { get; set; }
        Ссылок: 3
        public DbSet<CheckIn> CheckIns { get; set; }
        Ссылок: 5
        public DbSet<CheckOut> CheckOuts { get; set; }

        Ссылок: 0
        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            base.OnModelCreating(modelBuilder);

            modelBuilder.Entity<Booking>()
                .HasOne(b => b.CheckIn)
                .WithOne(c => c.Booking)
                .HasForeignKey<CheckIn>(c => c.BookingId);

            modelBuilder.Entity<CheckIn>()
                .HasOne(ci => ci.CheckOut)
                .WithOne(co => co.CheckIn)
                .HasForeignKey<CheckOut>(co => co.CheckInId);
        }
    }
}
```

Фрагмент коду для створення нового бронювання

```
// створення нового бронювання
Ссылок: 0
public async Task<IActionResult> Create(BookingCreateViewModel model)
{
    ModelState.Remove(nameof(model.Clients));
    ModelState.Remove(nameof(model.Rooms));

    if (!ModelState.IsValid)
    {
        model.Clients = new SelectList(_context.Clients, "Id", "FullName", model.ClientId);
        model.Rooms = new SelectList(_context.Rooms
            .Where(r => r.Status == RoomStatus.Free)
            .Select(r => new { r.Id, RoomInfo = r.Number + " (" + r.Type + ")" }),
            "Id", "RoomInfo", model.RoomId);

        return View(model);
    }

    if (model.CheckInDate >= model.CheckOutDate)
    {
        ModelState.AddModelError("", "Дата виїзду повинна бути пізніше дати заїзду.");
        model.Clients = new SelectList(_context.Clients, "Id", "FullName", model.ClientId);
        model.Rooms = new SelectList(_context.Rooms
            .Where(r => r.Status == RoomStatus.Free)
            .Select(r => new { r.Id, RoomInfo = r.Number + " (" + r.Type + ")" }),
            "Id", "RoomInfo", model.RoomId);

        return View(model);
    }

    var room = await _context.Rooms.FindAsync(model.RoomId);
    if (room == null || room.Status != RoomStatus.Free)
    {
        ModelState.AddModelError("", "Оберіть доступний номер.");
        model.Clients = new SelectList(_context.Clients, "Id", "FullName", model.ClientId);
        model.Rooms = new SelectList(_context.Rooms
            .Where(r => r.Status == RoomStatus.Free)
            .Select(r => new { r.Id, RoomInfo = r.Number + " (" + r.Type + ")" }),
            "Id", "RoomInfo", model.RoomId);

        return View(model);
    }
```

Фрагмент коду, який демонструє контроллер заселення

CheckInsController.cs

```
// POST: CheckIns/CheckIn/{id}
[HttpPost]
[ValidateAntiForgeryToken]
Ссылок: 0
public async Task<IActionResult> CheckIn(Guid id)
{
    var booking = await _context.Bookings
        .Include(b => b.Room)
        .FirstOrDefaultAsync(b => b.Id == id);

    if (booking == null || booking.Room == null)
    {
        TempData["Message"] = "Бронювання не знайдено або номер відсутній.";
        return RedirectToAction(nameof(Index));
    }


    booking.Room.Status = RoomStatus.Occupied;
    _context.Entry(booking.Room).State = EntityState.Modified;


    var checkIn = new CheckIn
    {
        Id = Guid.NewGuid(),
        BookingId = booking.Id,
        CheckInDateTime = DateTime.Now
    };


    booking.CheckIn = checkIn;
    _context.CheckIns.Add(checkIn);

    try
    {
        await _context.SaveChangesAsync();
        TempData["Message"] = "Клієнта успішно заселено.";
    }
    catch (DbUpdateConcurrencyException)
    {
        TempData["Message"] = "❌ Помилка заселення. Дані були змінені іншою операцією.";
    }

    return RedirectToAction(nameof(Index));
}
```

Фрагмент коду, який демонструє контроллер виселення

CheckOutsController.cs

```
// POST: CheckOuts/CheckOut
[HttpPost]
[ValidateAntiForgeryToken]
Ссылок: 0
public async Task<IActionResult> CheckOut(Guid checkInId)
{
    var checkIn = await _context.CheckIns
        .Include(ci => ci.Booking)
            .ThenInclude(b => b.Room)
        .FirstOrDefaultAsync(ci => ci.Id == checkInId);

    if (checkIn == null)
    {
        TempData["ErrorMessage"] = "Поселення не знайдено.";
        return RedirectToAction("Index", "Bookings");
    }

    bool alreadyCheckedOut = await _context.CheckOuts.AnyAsync(co => co.CheckInId == checkIn.Id);
    if (alreadyCheckedOut)
    {
        TempData["ErrorMessage"] = "Клієнта вже виселено!";
        return RedirectToAction("Index", "Bookings");
    }

    int nights = (checkIn.Booking.CheckOutDate - checkIn.Booking.CheckInDate).Days;
    if (nights <= 0) nights = 1;

    decimal totalAmount = nights * checkIn.Booking.Room.PricePerNight;

    var checkOut = new CheckOut
    {
        Id = Guid.NewGuid(),
        CheckInId = checkIn.Id,
        CheckOutDateTime = DateTime.Now,
        TotalAmount = totalAmount
    };
    _context.CheckOuts.Add(checkOut);

    if (checkIn.Booking.Room != null)
    {
        checkIn.Booking.Room.Status = RoomStatus.Free;
        _context.Rooms.Update(checkIn.Booking.Room);
    }

    checkIn.Booking.Status = BookingStatus.Completed;
    _context.Bookings.Update(checkIn.Booking);

    await _context.SaveChangesAsync();

    TempData["SuccessMessage"] = "Клієнта успішно виселено!";
    return RedirectToAction("Index");
}
```

Фрагмент коду, який демонструє контроллер панель адміністратора AdminController.cs

```
[Authorize(Roles = "Admin")]
Ссылок: 1
public class AdminController : Controller
{
    private readonly UserManager<ApplicationUser> _userManager;

    Ссылок: 0
    public AdminController(UserManager<ApplicationUser> userManager)
    {
        _userManager = userManager;
    }

    Ссылок: 0
    public IActionResult Index()
    {
        return View();
    }

    Ссылок: 2
    public async Task<IActionResult> ManageUsers()
    {
        var users = await _userManager.Users.ToListAsync();
        var userViewModels = new List<ManageUserViewModel>();

        foreach (var user in users)
        {
            var model = new ManageUserViewModel
            {
                Id = user.Id,
                Email = user.Email,
                FullName = user.FullName,
                Roles = (await _userManager.GetRolesAsync(user)).ToList()
            };
            userViewModels.Add(model);
        }

        return View(userViewModels);
    }
```

Фрагмент коду, який відповідає за вхід користувача

```
public class AccountController : Controller
{
    private readonly UserManager<ApplicationUser> _userManager;
    private readonly SignInManager<ApplicationUser> _signInManager;

    Ссылок: 0
    public AccountController(UserManager<ApplicationUser> userManager, SignInManager<ApplicationUser> signInManager)
    {
        _userManager = userManager;
        _signInManager = signInManager;
    }

    [HttpGet]
    Ссылок: 0
    public IActionResult Login() => View();

    [HttpPost]
    Ссылок: 0
    public async Task<IActionResult> Login(LoginViewModel model)
    {
        if (!ModelState.IsValid) return View(model);

        var result = await _signInManager.PasswordSignInAsync(model.Email, model.Password, true, false);

        if (result.Succeeded)
            return RedirectToAction("Index", "Clients");

        ModelState.AddModelError("", "Невірні логін або пароль");
        return View(model);
    }

    [HttpGet]
    Ссылок: 0
    public IActionResult Register() => View();

    [HttpPost]
    Ссылок: 0
    public async Task<IActionResult> Register(RegisterViewModel model)
    {
        if (!ModelState.IsValid) return View(model);

        var user = new ApplicationUser
        {
            UserName = model.Email,
            Email = model.Email,
            FullName = model.FullName
        };
```

Фрагмент коду, який відповідає за реєстрація нового користувача

```
public async Task<IActionResult> Register(RegisterViewModel model)
{
    if (!ModelState.IsValid) return View(model);

    var user = new ApplicationUser
    {
        UserName = model.Email,
        Email = model.Email,
        FullName = model.FullName
    };

    var result = await _userManager.CreateAsync(user, model.Password);

    if (result.Succeeded)
    {
        await _signInManager.SignInAsync(user, isPersistent: false);
        return RedirectToAction("Index", "Clients");
    }

    foreach (var error in result.Errors)
        ModelState.AddModelError("", error.Description);

    return View(model);
}

[HttpPost]
Ссылок: 0
public async Task<IActionResult> Logout()
{
    await _signInManager.SignOutAsync();
    return RedirectToAction("Login");
}
```