МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами

Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»

Завідувач кафедри

_____ проф. Приходько С.Б.

«___» _____ 2025 р.

КВАЛІФІКАЦІЙНАРОБОТА

на здобуття ступеня вищої освіти «бакалавр»

на тему: Розробка програмного комплексу для автоматизації управління готелем «Lake Park»

Виконала: студентка групи 4151


Руденко К. Р.

(підпис, ПІБ)

Керівник роботи:

доцент, к. ф-м.н.

(посада, науковий ступень вчене звання)


Латанська Л.О.

(підпис, ПІБ)

Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
(підпис)
«_08_» ___04___ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студентці Руденко Катерині Русланівні
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розробка програмного комплексу для автоматизації управління готелем «Lake Park»

Керівник роботи Латанська Людмила Олексіївна

Затверджені наказом ректора №153 від 08.04.2025 р.

2. Термін подання роботи: 02.06.2025 р.

3. Вихідні дані по роботі:

4. Перелік питань, що належать до розробки (найменування розділів)
Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Аналіз практичної задачі інженерії програмного забезпечення; Проєкт програмного забезпечення; Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів
1 Титульний слайд, 2 Мета і завдання кваліфікаційної роботи, 3 Аналоги ПЗ, 4 Аналіз вимог (діаграма варіантів використання), 5 Архітектура ПЗ (діаграма компонентів), 6 Архітектура ПЗ (діаграма розгортання), 7 Ескізний проєкт (діаграма діяльності «Створення бронювання на вебсайті»), 8 Ескізний проєкт (діаграма діяльності «Редагування гостьового рахунку»), 9 Ескізний проєкт (концептуальна модель даних), 10 Ескізний проєкт (прототипи розміщення елементів на формі), 11 Ескізний проєкт (остаточні прототипи сторінок (бронювання вебсайт, логін desktop-застосунок)), 12 Ескізний проєкт (остаточні прототипи сторінок (готель desktop-застосунок, меню telegram-бот)), 13 Технічний проєкт (діаграма класів), 14 Технічний проєкт (діаграма послідовності «Створення бронювання на вебсайті»), 15 Технічний проєкт (діаграма послідовності «редагування гостьового рахунку»), 16 Технічний проєкт (логічна модель даних), 17 Робочий проєкт (вибір засобів розробки), 18 Результати розробки (вебсайт), 19 Результати розробки (desktop-застосунок), 20 Результати розробки (telegram-бот), 21 Висновки, 22 Заключний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Гурець Н.В., ст. викладач		

7. Дата видачі завдання 08.04.2025 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	31.03.2025	ПП*
2.	Аналіз практичної задачі інженерії ПЗ	07.04.2025	ПП*
3.	Аналіз вимог до ПЗ	14.04.2025	ПП*
4.	Розробка архітектури ПЗ	21.04.2025	
5.	Розробка проекту ПЗ	05.05.2025	
6.	Реалізація та тестування ПЗ	12.05.2025	
7.	Підготовка розділу Результати розробки ПЗ	16.05.2025	
8.	Підготовка розділу з охорони праці	19.05.2025	ПП*
9.	Підготовка розділу ВИСНОВКИ	23.05.2025	
10.	Оформлення списку використаних джерел та додатків	26.05.2025	
11.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	02.06.2025	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку
12.	Підготовка презентації та доповіді	06.06.2025	
13.	Попередній захист роботи на засіданні кафедри ПЗАС	09.06.2025	
14.	Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді, а також Авторського договору з додатком	16.06.2025	

* - за результатами переддипломної практики (ПП), яка була з 03.02.2025 до 23.03.2025 р.

Студент _____ Руденко К.Р.
(підпис) (ПІБ)

Керівник роботи _____ Латанська Л.О.
(підпис) (ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота присвячена розв’язанню практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, а саме задачі розробки програмного комплексу для автоматизації управління готелем «Lake Park». До складу програмного комплексу входять: telegram-бот, desktop-застосунок, вебсайт.

У кваліфікаційній роботі представлено аналіз практичної задачі інженерії програмного забезпечення, розглянуто існуючі програмні продукти та розроблено постановку задачі на створення програмного комплексу. У першому розділі проведено аналіз практичної задачі, розглянуто існуючі програмні рішення та сформульовано вимоги до нового ПЗ. У другому розділі описано процес розробки програмного комплексу: проаналізовано вимоги, побудовано модель варіантів використання, виконано етапи проєктування (ескізний, технічний і робочий проєкти). У третьому розділі наведено результати реалізації програмного комплексу. Четвертий розділ присвячено питанням охорони праці: розглянуто нормативно-правову базу, систему управління охороною праці у готелі та запропоновано заходи зі зниження впливу шкідливих факторів.

Об’єктом даної роботи є процес розробки програмного комплексу для автоматизації управління готелем «Lake Park».

Кваліфікаційна робота викладена на 151 сторінці друкованого тексту та містить 31 рисунок, 21 таблицю, список використаних джерел з 19 найменувань та 5 додатків.

ABSTRACT

The qualification work is devoted to solving a practical problem of software engineering, characterized by complexity and uncertainty of conditions, namely the task of developing a software complex for automating the management of the Lake Park hotel. The software complex includes: a Telegram bot, a desktop application, and a website.

The thesis presents an analysis of a practical software engineering problem, reviews existing software products, and develops a problem statement for the creation of a software complex. The first chapter analyses the practical problem, reviews existing software solutions, and formulates requirements for the new software. The second chapter describes the process of developing the software complex: requirements are analyzed, a model of usage options is built, and the design stages (preliminary, technical, and working designs) are completed. The third chapter presents the results of the software complex implementation. The fourth chapter is devoted to occupational safety issues: the regulatory framework and the occupational safety management system in the hotel are considered, and measures to reduce the impact of harmful factors are proposed.

The object of this work is the process of developing a software complex for automating the management of the Lake Park hotel.

The thesis is presented on 151 pages of printed text and contains 31 figures, 21 tables, a list of references with 19 titles and 5 appendices.

ЗМІСТ

ВСТУП

У сучасних умовах стрімкого розвитку інформаційних технологій та цифровізації бізнес-процесів автоматизація управління підприємствами набуває особливої актуальності. Готельний бізнес, як один із секторів сфери послуг, стикається з численними викликами, серед яких фрагментація даних, низька швидкість обробки інформації, складність інтеграції різних способів управління та недостатній рівень захисту персональних даних клієнтів. Традиційні методи управління вже не відповідають вимогам сучасного ринку, що зумовлює потребу у впровадженні інноваційних рішень для підвищення ефективності роботи готелю.

На сучасному ринку представлені низка готових програмних рішень – такі як Booking, Expedia, Airbnb – які забезпечують базове онлайн-бронювання та взаємодію з клієнтами. Проте ці платформи, орієнтовані на загальний масовий ринок, не враховують унікальні бізнес-процеси окремих готелів і часто пропонують лише стандартний набір функцій без глибокої аналітики чи внутрішньої автоматизації. Це ставить під сумнів їхню ефективність у контексті індивідуальних потреб готельного комплексу «Lake Park» та підкреслює необхідність розробки спеціалізованого програмного комплексу.

Необхідність розробки спеціалізованого програмного комплексу продиктована потребою створення єдиного інформаційного простору, який дозволить інтегрувати всі ключові аспекти управління готелем. Програмний комплекс буде складатися з трьох взаємопов’язаних компонентів:

- вебсайту для клієнтів;
- desktop-застосунку для співробітників готелю;
- telegram-бота для швидкої взаємодії з гостями.

Такий підхід сприятиме оптимізації внутрішніх процесів, мінімізації ризику помилок та оперативному реагуванню на змінені умови ринку. Необхідність розробки підтверджується тим, що сучасні програмні забезпечення, орієнтовані на загальний ринок онлайн-бронювання, не завжди враховують специфіку окремих

підприємств, що знижує ефективність їх використання в умовах індивідуальних бізнес-процесів готелю.

Готель «Lake Park» стикається із зниженням кількості гостей у зимовий період через некупальний сезон, що негативно впливає на загальний дохід закладу. Адміністрація готелю планує провести реструктуризацію та впровадити невеликі зміни в організаційних процесах з метою залучення більшої кількості клієнтів у цей період. Впровадження надійної програмного комплексу дозволить не лише оптимізувати внутрішні процеси, але й створити умови для активного маркетингового впливу, що сприятиме збільшенню потоку гостей і, як наслідок, підвищенню доходу підприємства.

Метою кваліфікаційної роботи є створення готового до впровадження програмного продукту для інтегрованої автоматизації основних бізнес-процесів готелю «Lake Park», який забезпечить:

- бронювання номерів;
- керування гостьовими рахунками;
- перегляд інформації про готель;
- обробку й захист персональних даних гостей та персоналу;
- обробка даних про готель;
- генерацію аналітичної звітності.

Для досягнення поставленої мети передбачено виконання таких завдань:

1) аналіз практичної задачі інженерії програмного забезпечення для автоматизації роботи готелю «Lake park»;

2) дослідження існуючих програмних рішень: оцінка їхніх можливостей і обмежень;

3) формулювання функціональних та нефункціональних вимог до програмного комплексу;

4) вибір проєктування архітектури комплексу з урахуванням трьох компонентів;

5) реалізація прототипів кожного компоненту;

6) тестування розроблених модулів і оцінка їхньої працездатності.

7) розробка додатку «Технічного завдання»;

8) розробка додатку «Опис програми»;

9) створення інструкції користувача.

Об’єктом кваліфікаційної роботи є процес розробки програмного комплексу для автоматизації управління готелем «Lake Park».

З огляду на вищезазначене, розробка спеціалізованого програмного комплексу для автоматизації управління готелем «Lake Park» є надзвичайно актуальною. Запропоноване рішення передбачає створення єдиного інформаційного простору шляхом інтеграції вебсайту, desktop-застосунку та telegram-боту, що дозволить оптимізувати внутрішні процеси, зменшити ризик помилок, покращити якість обслуговування клієнтів та підвищити конкурентоспроможність готелю на ринку. Дана кваліфікаційна робота спрямована на вирішення зазначених проблем шляхом розробки комплексного програмного продукту, який відповідає сучасним вимогам та стандартам у сфері автоматизації управління підприємствами.

1 АНАЛІЗ ПРАКТИЧНОЇ ЗАДАЧІ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ПРОГРАМНОГО КОМПЛЕКСУ ДЛЯ АВТОМАТИЗАЦІЇ УПРАВЛІННЯ ГОТЕЛЕМ «LAKE PARK»

1.1 Аналіз практичної задачі інженерії програмного забезпечення для автоматизації управління готелем «Lake Park»

Дана робота присвячена розв’язанню практичної задачі інженерії програмного забезпечення, а саме розробці програмного комплексу для автоматизації управління готелем «Lake Park».

У сучасних умовах цифровізація управління готельним бізнесом набуває особливої актуальності завдяки необхідності забезпечення швидкого та безпечного доступу до інформації. Проблематика розробки програмного комплексу для автоматизації управління готелем «Lake Park» пов’язана із комплексністю завдань, що включають інтеграцію численних компонентів, забезпечення збереження та обробки персональних даних клієнтів, а також підтримку високої продуктивності при зростанні навантаження.

Аналізуючи сучасний ринок, можна відзначити, що існуючі програмні забезпечення для управління зазвичай обмежуються окремими аспектами роботи закладу і не враховують специфічні вимоги кожного підприємства, що спричиняє необхідність розробки індивідуального рішення.

Особливу увагу слід приділити питанням безпеки інформації, оскільки обробка персональних даних є критичним аспектом у роботі готелю. Розроблюваний комплекс повинен відповідати сучасним стандартам захисту даних, що включає використання надійних алгоритмів шифрування та автентифікації користувачів. Крім того, інтеграція з базою даних має забезпечувати не лише збереження, а й оперативний доступ до інформації, що дозволить своєчасно реагувати на змінені умови ринку та потреби клієнтів.

Комплексність задачі обумовлюється також необхідністю розробки інтуїтивно зрозумілого інтерфейсу, який полегшить роботу як співробітникам готелю, так і самим клієнтам. Сучасні тенденції в галузі програмної інженерії свідчать про те, що успішність впровадження програмних комплексів значною мірою залежить від здатності програмного забезпечення адаптуватися до змін у бізнес-процесах, що вимагає високої гнучкості при проєктуванні архітектури застосунку. Таким чином, розробка програмного забезпечення для готелю «Lake Park» є комплексною задачею, що передбачає ретельний аналіз вимог, проєктування архітектури та реалізацію сучасних методів забезпечення безпеки та продуктивності.

Заміський готельно-ресторанний комплекс «Lake Park» розташований у місті Самар, за 45 км від центру Дніпра, що забезпечує ідеальне поєднання спокою природи та зручної доступності до великого міста. Сам комплекс представляє собою сучасне місце відпочинку, яке орієнтоване на надання високоякісних послуг як для постійних гостей, так і для тих, хто відвідує його вперше. Готель пропонує широкий вибір номерів – від стандартних двомісних до просторих вілл із приватними терасами, що дозволяє задовольнити різноманітні потреби клієнтів.

Особливу увагу комплекс приділяє не лише розміщенню, але й створенню унікального відпочинкового простору. Ресторан готелю готує вишукані страви європейської кухні, а також домашні рецепти з ексклюзивною подачею, що створює атмосферу затишку та гостинності. Крім того, комплекс має низку додаткових послуг, серед яких – тенісні корти та інші можливості для активного відпочинку, що дозволяє гостям повноцінно насолодитися своїм перебуванням.

З точки зору розробки програмного забезпечення, врахування специфіки роботи комплексу «Lake Park» є надзвичайно важливим. Програмний комплекс для автоматизації повинен не лише забезпечувати оперативне бронювання номерів і управління послугами, але й адаптуватися до різноманітних сценаріїв відпочинку: організації спеціальних заходів, індивідуальних запитів щодо харчування та дозвілля, а також враховувати сезонні коливання попиту. Всі ці аспекти вимагають

інтеграції кількох програмних компонентів, що забезпечують як зручність для користувачів, так і високий рівень безпеки обробки персональних даних гостей.

Програмні компоненти для управління повинні враховувати різноманітність номерів та послуг, що надаються, забезпечуючи прозорість інформації про ціни, наявність вільних номерів і можливість гнучкого резервування. Це дозволить комплексно задовольнити потреби готелю, який працює у конкурентному середовищі сучасного ринку готельного бізнесу.

1.2 Аналіз існуючих програмних рішень

На сучасному ринку існує певна кількість програмних рішень, орієнтованих на автоматизацію роботи готельних комплексів, проте більшість з них зосереджені на загальному процесі бронювання та не враховують специфіку окремих закладів. Нижче наведено аналіз трьох основних програм-аналогів, що використовуються у готельному бізнесі. Проаналізуємо аналоги для автоматизації процесу управління готелем:

1) Booking.com[1]

Назва: Booking.com

Розробник: Booking Holdings

Основні функціональні можливості:

– онлайн бронювання готелів та іншого житла;

– розширений пошук за різними критеріями (ціна, рейтинг, розташування);

– інтеграція з різноманітними платіжними системами;

– можливість перегляду відгуків і рейтингів.

Переваги:

– широкий вибір готелів та житлових опцій;

– зручний інтерфейс для користувачів;

– надійна система бронювання із інтегрованими платіжними рішеннями.

Недоліки:

– перевантаженість інформацією через велику кількість варіантів, що може ускладнювати вибір;

– можливі неточності в системі відгуків та рейтингів, що впливають на прийняття рішення.

Головна сторінка сайту зображена на рисунку 1.1.

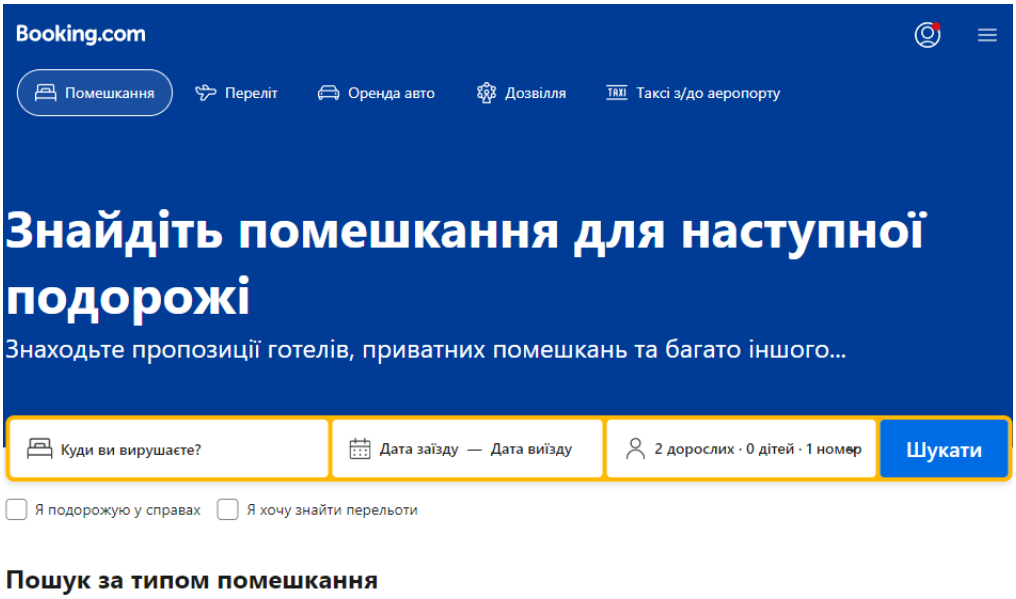

Рисунок 1.1 – Головна сторінка сайту Booking

2) Expedia[2]

Назва: Expedia

Розробник: Expedia Group

Основні функціональні можливості:

– бронювання готелів, авіаквитків, прокату автомобілів та інших транспортних засобів;

– можливість бронювання пакетних послуг (комплексні пропозиції: готель і авіаквиток тощо);

– персональні спеціальні пропозиції для зареєстрованих користувачів.

Переваги:

– універсальний підхід до бронювання, що дозволяє користувачам організувати повноцінну подорож;

– інтеграція численних послуг в одному сервісі;

– наявність спеціальних пропозицій для постійних клієнтів.

Недоліки:

– складність у розмежуванні основної вартості та додаткових витрат (комісії, податки);

– інколи незрозумілі умови бронювання через високу конкуренцію на ринку онлайн-бронювання.

Головна сторінка сайту зображена на рисунку 1.2.

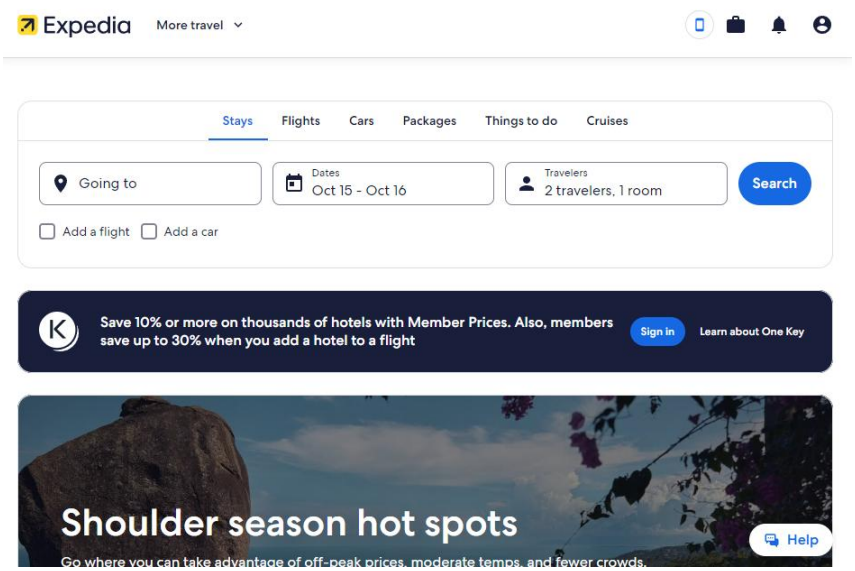

Рисунок 1.2 – Головна сторінка сайту Expedia

3) Airbnb[3]

Назва: Airbnb

Розробник: Airbnb, Inc.

Основні функціональні можливості:

– оренда житла, включаючи приватні квартири, будинки, окремі кімнати;

– розширений пошук за ціною, розташуванням та іншими критеріями;

– система взаємних відгуків між гостями та господарями, що створює довіру.

Переваги:

– унікальна пропозиція приватного житла та незвичайних варіантів розміщення;

– можливість персоналізованого підбору житла за індивідуальними вимогами;

– гнучкість у виборі як окремих кімнат, так і повних приміщень.

Недоліки:

– сезонність попиту, що може впливати на рівень завантаженості об’єктів;

– іноді виникають проблеми взаєморозуміння між господарями та гостями, що потребує втручання платформи.

Головна сторінка сайту зображена на рисунку 1.3.

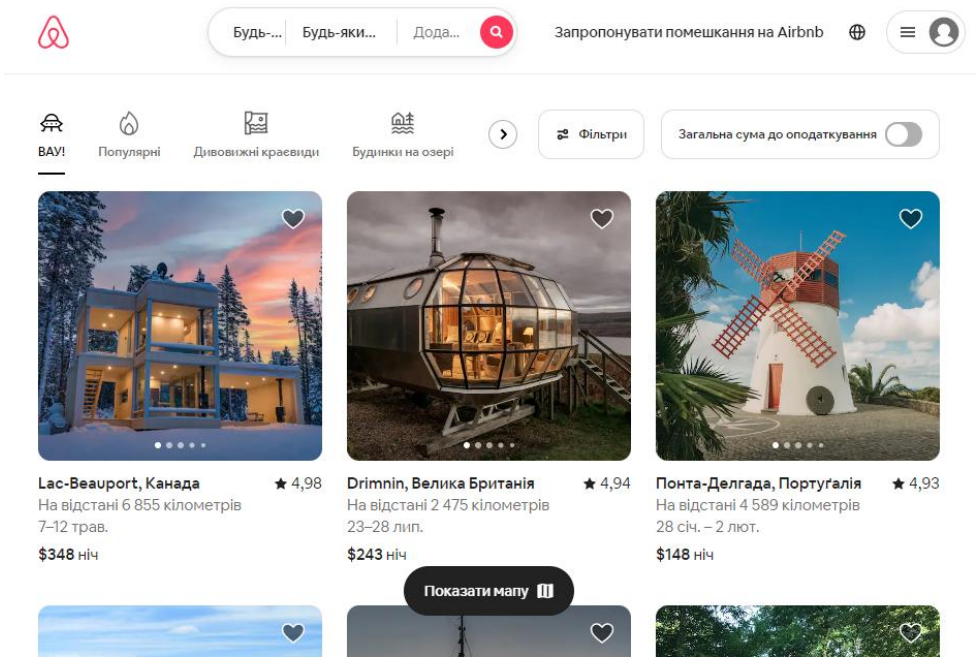

Рисунок 1.3 – Головна сторінка сайту Airbnb

Наведений аналіз показує, що існуючі програмні забезпечення, такі як Booking, Expedia та Airbnb, орієнтовані на загальний ринок онлайн-бронювання та

не враховують специфіку окремих готельних комплексів. З одного боку, вони пропонують зручні інструменти для пошуку і бронювання житла, але з іншого – їх функціональність обмежується стандартними можливостями, які не завжди відповідають потребам конкретного закладу.

Готель «Lake Park» має власний вебсайт, проте він не виконує ключових функцій, необхідних для сучасного управління готелем, таких як автоматизоване бронювання, детальна аналітика та управління внутрішніми процесами. Розробка спеціалізованого програмного комплексу дозволить:

– автоматизувати внутрішні процеси, зменшити ризики людських помилок і прискорити обробку даних;

– контролювати власну базу даних, що буде гарантувати високий рівень захисту персональної інформації гостей;

– адаптувати програмний комплекс до специфічних вимог готелю, що забезпечить індивідуальний підхід до клієнтів;

– оперативно реагувати на зміни ринку та оптимізувати бізнес-процеси за допомогою.

1.3 Постановка задачі на розробку програмного комплексу для автоматизації управління готелем «Lake Park»

Виходячи з усього вище викладеного сформулюємо постановку задачі. У даній кваліфікаційній роботі необхідно розробити програмний комплекс для автоматизації управління готелем «Lake Park», що дозволить забезпечити комплексний підхід до управління всіма аспектами роботи закладу.

У рамках роботи будуть розроблені:

– вебсайт;

– desktop-застосунок;

– telegram-бот.

Функції вебсайту:

– відображення інформації про наявні послуги готелю;

– відображення опису готелю;

– відображення контактних даних;

– авторизація;

– обробка особистих даних (додавання, редагування, видалення);

– обробка та перегляд бронювання (редагування, створення, видалення, перегляд бронювань клієнта).

Функції desktop-застосунку:

– авторизація;

– обробка гостьових рахунків (редагування та створення);

– обробка та перегляд бронювання (редагування, створення, видалення, перегляд усіх бронювань);

– обробка інформації про послуги (створення, редагування, видалення);

– обробка інформації про кімнати (створення, редагування, видалення);

– обробка інформації про персонал (створення, редагування, видалення);

– генерація звітів та статистик.

Функції telegram-боту:

– перегляд інформації про готель;

– перегляд інформації про контакти;

– перегляд інформації про послуги;

– створення бронювання;

– додавання особистих даних.

Вимоги до складу виконуваних функцій

1) Створення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат бронювання.

2) Редагування бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат редагування.

3) Видалення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат видалення.

4) Перегляд бронювань клієнта

Вхідні дані: номер телефону.

Вихідні дані: список з бронюванням клієнта.

5) Перегляд усіх бронювань

Вхідні дані: натиснення кнопки «Бронювання».

Вихідні дані: список усіх бронювань (ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги).

6) Перегляд наявних послуг готелю

Вхідні дані: натиснення кнопки «Послуги».

Вихідні дані: найменування, ціна.

7) Перегляд опису готелю

Вхідні дані: Натиснення кнопки «Про нас».

Вихідні дані: Опис готелю.

8) Перегляд інформації про контакти готелю

Вхідні дані: натиснення кнопки «Наші контакти».

Вихідні дані: контактні дані.

9) Додавання особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат додавання.

10) Редагування особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат редагування.

11) Видалення особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат видалення.

12) Створення гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат створення.

13) Редагування гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат редагування.

14) Створення статистики щодо бронювання

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика щодо бронювань за визначений період (загальна кількість бронювань, загальна кількість суми гостьових рахунків, середній відсоток зайнятих кімнат за період).

15) Створення статистики про популярні додаткові послуги

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика про популярні додаткові послуги (Послуга, кількість використання).

16) Створення звіту «Сума рахунків за період»

Вхідні дані: дата початку, дата кінця.

Вихідні дані: звіт про всю суму гостьових рахунків за визначеним періодом (загальна сума рахунків, кількість гостей, середня сума рахунків).

17) Авторизація

Вхідні дані: номер телефону, пароль.

Вихідні дані: повідомлення про результат авторизації.

18) Обробка інформації про персонал

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження, посада.

Вихідні дані: повідомлення про результат обробки.

19) Обробка інформації про кімнати

Вхідні дані: номер, кількість односпальних ліжок, кількість двоспальних ліжок, максимальна кількість гостей, статус.

Вихідні дані: повідомлення про результат обробки.

20) Обробка інформації про послуги

Вхідні дані: найменування, ціна, статус.

Вихідні дані: повідомлення про результат обробки.

Вимоги до організації вхідних та вихідних даних

Вхідною інформацією є дані, які вводяться вручну (в залежності від типу поля введення, дані можуть бути: числові цілі, числові десяткові, текстові, формату дати), або обираються із представлених списків (для уникнення можливих помилок, у випадках списків представлені готові варіанти вибору), або отримуються з бази даних PostgreSQL.

Вихідною інформацією є виведення результатів операцій у екранні форми, повідомлення та сформовані звіти.

Вимоги до інформаційної та програмної сумісності

desktop-застосунок:

Програма повинна працювати під ОС Windows версії не нижче 10. Необхідна наявність серверу або встановленої програми Open Server Panel або XAMPP чи інших. Для формування звітів необхідне встановлення тестових редакторів Microsoft Office або схожих.

вебсайт:

Необхідна наявність сучасного інтернет браузера Google Chrome, Opera GX чи інших, та стабільне підключення до інтернет з’єднання.

telegram-бот:

Необхідний встановлений застосунок Telegram на пристрої, який використовується, та зареєстрований аккаунт у месенджері.

Вимоги до складу параметрів технічних засобів

Сервер, на якому розміщуються база даних і вебсайт, повинен мати такі мінімальні параметри:

- CPU: 2x Xeon 64bit або еквівалент;
- RAM: 4 GB;
- HDD: 150 GB;
- постійний доступ до Інтернет мережі.

Desktop-застосунок:

– CPU: двоядерний із тактовою частотою не менше 2,0 GHz;
– RAM: 2 GB;
– HDD/SSD: 100 GB;
– підключення до Інтернет мережі.

Технічні вимоги для telegram-бота, що взаємодіє з користувачами:

– CPU: одне ядро з частотою 1,5 GHz;
– RAM: 1 GB;
– HDD: 50 GB;
– постійний доступ до Інтернет мережі.

Повністю постановка задачі сформульована у технічному завданні, яке наведено у Додатку А.

2 РОЗРОБКА ПРОГРАМНОГО КОМПЛЕКСУ ДЛЯ АВТОМАТИЗАЦІЇ УПРАВЛІННЯ ГОТЕЛЕМ «LAKE PARK»

2.1 Аналіз вимог до програмного комплексу

2.1.1 Побудова моделі варіантів використання

Для побудови моделі варіантів використання програмного забезпечення перш за все необхідно визначити акторів, які взаємодіятимуть із системою. Актори представляють зовнішні по відношенню до системи ролі, що ініціюють ті чи інші процеси. У таблиці 2.1 наведено короткий опис основних акторів, які братимуть участь у взаємодії з програмним комплексом.

Таблиця 2.1 – Актори програмного комплексу

Актор	Короткий опис
Клієнт	Може бронювати номера, керувати особистими даними та переглядати інформацію про готель.
Адміністратор	Керує бронюванням, особистим рахунком гостя, обробляє особисті дані гостя
Менеджер	Керує бронюванням, особистим рахунком гостя, обробляє інформацію про готель, послуги, персонал.
Власник	Може створювати звіти та статистики

Після визначення акторів було визначено варіанти використання, що характеризують основні сценарії роботи системи. У таблиці 2.2 наведено загальний перелік визначених варіантів за користувачами та їхнім коротким описом.

Таблиця 2.2 – Варіанти використання програмного комплексу

Актор	Назва	Короткий опис
Клієнт, Адміністратор, Менеджер	Створення бронювання	Клієнти можуть робити бронювання через вебсайт, адміністратори та менеджери можуть додавати бронювання вебсайт або за допомогою мобільного додатку.
Клієнт, Адміністратор, Менеджер	Видалення бронювання	Усі користувачі можуть видаляти бронювання, якщо вони більше не потрібні.
Клієнт, Адміністратор, Менеджер	Редагування бронювання	Усі користувачі можуть редагувати інформацію про бронювання.
Клієнт	Видалення особистих даних	Клієнти можуть видаляти свої особисті дані з системи за власним бажанням.
Клієнт	Редагування особистих даних	Клієнти можуть редагувати свої особисті дані, такі як контактна інформація.
Клієнт	Додавання особистих даних	Клієнти можуть створювати профіль, додаючи особисті дані.
Адміністратор, Менеджер	Редагування гостьового рахунку	Адміністратори та менеджери можуть редагувати інформацію на гостьовому рахунку, додавати оплати та інші дії.
Адміністратор, Менеджер	Створення гостьового рахунку	Адміністратори та менеджери можуть створювати нові гостьові рахунки для клієнтів.
Власник	Створення звіту про всю суму гостьових рахунків за визначеним періодом	Адміністратори можуть створювати звіти, які показують загальну суму гостьових рахунків за певний період.
Власник	Створення статистики про популярні додаткові послуги	Адміністратори можуть створювати статистику, що показує популярність різних додаткових послуг готелю.
Власник	Створення статистики щодо бронювання	Адміністратори можуть створювати статистику, що показує тенденції в резерваціях та завантаженість готелю.
Менеджер	Обробка інформації п ро послуги	Менеджери можуть додавати, редагувати та видаляти інформацію про послуги, які готель пропонує.
Менеджер	Обробка інформації про кімнати	Менеджери можуть додавати, редагувати та видаляти інформацію про різні типи кімнат та їх доступність.
Менеджер	Обробка інформації п ро персонал	Менеджери можуть додавати та управляти інформацією про персонал готелю.

На основі отриманої інформації було проведено розподіл ролей за варіантами використання та створено діаграму варіантів використання, що представлена на рисунку 2.1.

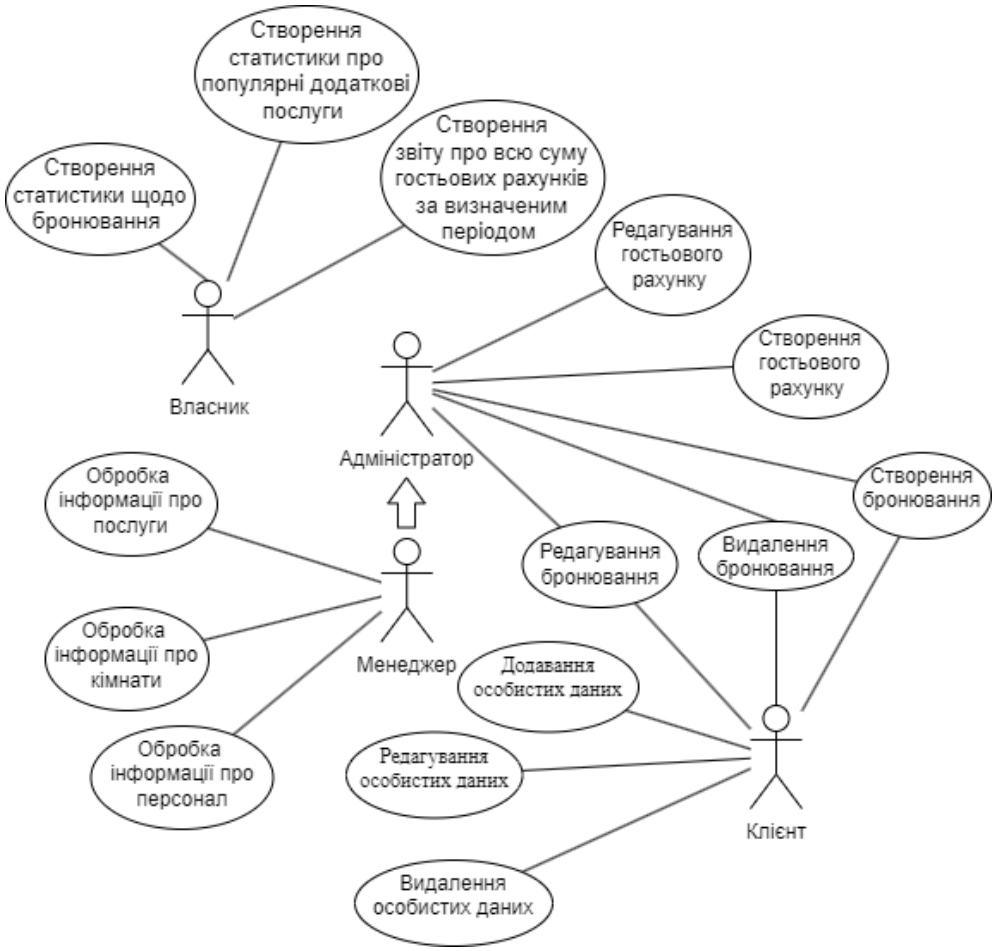

Рисунок 2.1 – Діаграма варіантів використання програмного комплексу для автоматизації роботи готелю «Lake Park»

Далі розглянемо специфікації варіантів використання.

2.1.2 Специфікації варіантів використання

У межах моделювання варіантів використання програмного комплексу для автоматизації роботи готелю «Lake Park» були визначені основні бізнес-процеси та відповідні сценарії взаємодії користувачів із системою. Специфікації варіантів використання згруповані відповідно до ключових потоків обробки інформації. Вони подані у вигляді таблиць, що деталізують цілі, учасників, передумови, основні та альтернативні сценарії кожного випадку використання. У таблицях 2.3 - 2.9 наведено варіанти використання для потоку «Обробка бронювання», таблиці 2.10 і 2.11 – «Обробка особистих даних», «Обробка гостьового рахунку» – таблиці 2.12 і

2.13, «Створення звіту або статистики» – таблиці 2.14 - 2.16, у таблицях 2.17 - 2.20 – для потоку «Обробка інформації про готель, персонал».

Таблиця 2.3 – Специфікація варіанту використання «Створення бронювання»

Складова	Опис
Опис прецеденту	Створення бронювання (desktop-застосунок)
Дійові особи прецеденту	Адміністратор, менеджер
Передумови	Гість звернувся для бронювання номера, і адміністрації необхідно зарезервувати номер у системі.
Потік подій	Адміністратор або менеджер відкриває розділ "Бронювання" у системі.
Базовий потік	1) Користувач заходить у розділ "Бронювання". 2) Система виводить екран зі списком усіх бронювань. 3) Користувач натискає кнопку "Створити". 4) Система виводить екран з полями для введення даних. 5) Користувач вводить необхідні дані (клієнт, кімната тощо). 6) Натискає "Створити". 7) Система додає нове бронювання у базу даних.
Альтернативні потоки	Можливість забронювати кілька номерів одночасно для групи гостей
Постумови	Бронювання відображається у системі і є готовим для подальшої обробки (наприклад, для підготовки рахунку або зміни умов).

Таблиця 2.4 – Специфікація варіанту використання «Створення бронювання»

Складова	Опис
Опис прецеденту	Створення бронювання (вебсайт)
Дійові особи прецеденту	Клієнт
Передумови	Клієнт зайшов на сайт.
Потік подій	Клієнт обирає доступні дати та номер для бронювання на сторінці створення бронювання.
Базовий потік	1) Клієнт переходить на сторінку створення бронювання. 2) Система виводить поля для введення необхідних даних. 3) Користувач заповнює необхідні дані. 4) Натискає кнопку "Створити" 5) Система перевіряє коректність введених даних. 6) Система надає підтвердження про створення бронювання.
Альтернативні потоки	Якщо номер недоступний, клієнту пропонується інша дата або альтернативний номер.
Постумови	Клієнт може переглядати своє бронювання в особистому кабінеті.

Таблиця 2.5 – Специфікація варіанту використання «Створення бронювання»

Складова	Опис
Опис прецеденту	Створення бронювання (telegram-бот)
Дійові особи прецеденту	Клієнт
Передумови	Клієнт бажає створити бронювання в готелі за допомогою Telegram-бота.
Потік подій	Клієнт відкриває Telegram-бот і обирає опцію створення бронювання/
Базовий потік	1) Бот запитує у клієнта необхідні дані. 2) Користувач відправляє текст з необхідними даними. 3) Після кожного введеного поля система перевіряє дані на коректність. 4) Якщо все правильно система запитує наступні необхідні дані. Процедура повторюється доки необхідні дані не будуть заповнені 5) Система додає бронювання у систему і виводить користувачу повідомлення про успішне створення.
Альтернативні потоки	немає
Постумови	Клієнт отримує підтвердження про бронювання в чаті Telegram

Таблиця 2.6 – Специфікація варіанту використання «Видалення бронювання»

Складова	Опис
Опис прецеденту	Видалення бронювання (desktop)
Дійові особи прецеденту	Менеджер, Адміністратор
Передумови	Користувач вже авторизовані в системі, гість скасував бронювання або бронювання створене з помилкою.
Потік подій	Адміністратор або менеджер відкриває розділ "Бронювання" у системі.
Базовий потік	1) Користувач заходить у розділ "Бронювання". 2) Система виводить екран зі списком усіх бронювань. 3) Користувач натискає кнопку "Видалити" біля необхідного бронювання. 4) Система виводить екран з підтвердженням видалення. 5) Користувач натискає "Підтвердити". 6) Система видаляє бронювання з бази даних.
Альтернативні потоки	немає
Постумови	Видалене бронювання більше не відображається в системі.

Таблиця 2.7 – Специфікація варіанту використання «Видалення бронювання»

Складова	Опис
1	2
Опис прецеденту	Видалення бронювання (вебсайт)
Дійові особи прецеденту	Клієнт
Передумови	Клієнт має активне бронювання та зареєстрований на сайті.

Кінець таблиці 2.7.

1	2
Потік подій	Клієнт обирає своє бронювання у персональному кабінеті.
Базовий потік	1) Клієнт переходить на сторінку Мої бронювання. 2) Система виводить усі бронювання клієнта. 3) Клієнт обирає необхідне бронювання. 4) Система виводить інформацію про конкретне бронювання. 5) Користувач натискає кнопку "Видалити" 6) Система виводить повідомлення з підтвердженням видалення. 7) Користувач підтверджує видалення 8) Система видаляє бронювання і виводить повідомлення про успішне видалення.
Альтернативні потоки	Якщо термін скасування минув, клієнту пропонується зв’язатися з адміністрацією готелю.
Постумови	Клієнт отримує підтвердження скасування.

Таблиця 2.8 – Специфікація варіанту використання «Редагування бронювання»

Складова	Опис
Опис прецеденту	Редагування бронювання (desktop)
Дійові особи прецеденту	Адміністратор, менеджер
Передумови	Користувач вже авторизований в системі. Гість звернувся до адміністрації з проханням внести зміни в існуюче бронювання (зміна дат, номеру, послуг).
Потік подій	Адміністратор або менеджер відкриває розділ "Бронювання" у системі та знаходить необхідне бронювання
Базовий потік	1) Користувач заходить у розділ "Бронювання". 2) Система виводить екран зі списком усіх бронювань. 3) Користувач натискає кнопку "Редагувати" біля необхідного бронювання. 4) Система виводить екран з полями для введення даних, де дані вже заповнені. 5) Користувач редагує необхідні дані. 6) Користувач натискає "Зберегти". 7) Система перевіряє коректність введених даних і якщо все правильно зберігає зміни у базі даних.
Альтернативні потоки	немає
Постумови	Редаговане бронювання залишається доступним для подальшої обробки або перегляду.

Таблиця 2.9 – Специфікація варіанту використання «Редагування бронювання»

Складова	Опис
Опис прецеденту	Редагування бронювання (вебсайт)
Дійові особи прецеденту	Клієнт
Передумови	Клієнт має активне бронювання, яке можна редагувати і авторизований у системі.
Потік подій	Клієнт обирає своє бронювання в особистому кабінеті
Базовий потік	1) Клієнт переходить на сторінку Мої бронювання. 2) Система виводить усі бронювання клієнта. 3) Клієнт обирає необхідне бронювання. 4) Система виводить інформацію про конкретне бронювання. 5) Користувач редагує необхідні дані і натискає кнопку "Зберегти". 6) Система перевіряє коректність введених даних і якщо все правильно зберігає зміни у базі даних.
Альтернативні потоки	Якщо бронювання неможливо змінити онлайн, клієнт може звернутися до адміністрації.
Постумови	Оновлена інформація відображається в особистому кабінеті клієнта

Таблиця 2.10 – Специфікація варіанту використання «Додавання особистих даних»

Складова	Опис
Опис прецеденту	Додавання особистих даних
Дійові особи прецеденту	Клієнт, Адміністратор
Передумови	немає
Потік подій	Прецедент починається коли користувач натискає на кнопку «Створити»
Базовий потік	1) Система відображає клієнту форму для заповнення особистих даних. 2) Клієнт вводить свої особисті дані в відповідні поля форми. 3) Клієнт підтверджує введені дані або натискає на кнопку "Зберегти". 4) Система зберігає введені дані у базу даних, пов'язану з обліковим записом клієнта.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Клієнт успішно додав свої особисті дані, створивши свій обліковий запис.

Таблиця 2.11 – Специфікація варіанту використання «Редагування особистих даних»

Складова	Опис
Опис прецеденту	Редагування особистих даних
Дійові особи прецеденту	Клієнт, Адміністратор
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач авторизується і натискає на кнопку «Редагувати»
Базовий потік	1) Система відображає особисті дані користувача для перегляду та редагування. 2) Клієнт може редагувати будь-яку інформацію, яка включає в себе, наприклад, ім'я, прізвище, адресу, контактні дані та інші особисті дані. 3) Клієнт зберігає зміни, натискаючи на кнопку "Зберегти".
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Після успішного виконання прецеденту, система зберігає оновлену інформацію про особисті дані клієнта, і він може продовжити користуватися системою з оновленими даними.

Таблиця 2.12 – Специфікація варіанту використання «Видалення особистих даних»

Складова	Опис
Опис прецеденту	Видалення особистих даних
Дійові особи прецеденту	Клієнт, Адміністратор
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач авторизується і натискає на кнопку «Видалити»
Базовий потік	1) Система показує підтверджуюче попередження користувачу, запитуючи чи він впевнений у бажанні видалити свої особисті дані. 2) Користувач підтверджує свою дію. 3) Система видаляє всі особисті дані користувача з бази даних. 4) Після видалення даних, система автоматично виходить з облікового запису користувача та перенаправляє його на головну сторінку вебсайту.
Альтернативні потоки	Альтернативний потік 1: Підтвердження видалення – Після кроку 1, система може запитати користувача підтвердження видалення особистих даних, щоб уникнути випадкового видалення. – Користувач підтверджує видалення. – Особисті дані видаляються з системи. Альтернативний потік 2: Відміна видалення – Після кроку 1, якщо користувач передумав видаляти особисті дані, він може вибрати опцію "Відмінити." – Видалення скасовується, і бронювання залишається в системі без змін.
Постумови	Клієнт успішно видалив свої особисті дані та вийшов із системи. Особисті дані більше не доступні користувачу і були повністю видалені із системи.

Таблиця 2.13 – Специфікація варіанту використання «Редагування гостьового рахунку»

Складова	Опис
Опис прецеденту	Редагування гостьового рахунку
Дійові особи прецеденту	Менеджер, Адміністратор
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у відповідний гостьовий рахунок і натискає кнопку «Редагувати»
Базовий потік	1) Система відкриває форму для редагування інформації гостьового рахунку. 2) Користувач вносить необхідні зміни до інформації гостьового рахунку. 3) Користувач підтверджує внесені зміни. 4) Система зберігає оновлену інформацію гостьового рахунку в базі даних.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Гостьовий рахунок був успішно відредагований, і нова інформація збережена в системі.

Таблиця 2.14 – Специфікація варіанту використання «Створення гостьового рахунку»

Складова	Опис
Опис прецеденту	Створення гостьового рахунку
Дійові особи прецеденту	Менеджер, Адміністратор
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ Гостьові рахунки і натискає кнопку «Створити»
Базовий потік	1) Система відображає форму для створення нового гостьового рахунку. 2) Користувач вводить необхідні дані у відповідні поля форми. 3) Користувач підтверджує створення гостьового рахунку. 4) Система перевіряє введені дані на відповідність та валідність. 5) Після успішної перевірки, система створює новий гостьовий рахунок та зберігає введені дані в базі даних. 6) Система відображає повідомлення про успішне створення гостьового рахунку і надає користувачу можливість продовжити роботу з системою.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	У системі створено новий гостьовий рахунок з введеними даними, який може бути використаний для подальшого обліку та обслуговування гостей готелю.

Таблиця 2.15 – Специфікація варіанту використання «Створення звіту про всю суму гостьових рахунків за визначеним періодом»

Складова	Опис
Опис прецеденту	Створення звіту про всю суму гостьових рахунків за визначеним періодом
Дійові особи прецеденту	Власник
Передумови	Користувач вже авторизований в системі.
Потік подій	1) Прецедент починається, коли користувач, який вже авторизований у систему, переходить до розділу "Створення звіту". 2) Користувач вводить дату початку та дату кінця визначеного періоду. 3) Користувач натискає кнопку "Сума гостьових рахунків" для створення звіту.
Базовий потік	1) Система перевіряє правильність та валідність введених дат. 2) Після успішної перевірки, система аналізує базу даних гостьових рахунків та обчислює загальну суму рахунків за вказаний період. 3) Система створює звіт, який містить інформацію про загальну суму гостьових рахунків та інші відомості за визначений період. 4) Система надає користувачу можливість переглянути та завантажити створений звіт.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Система створила звіт, який містить інформацію про загальну суму гостьових рахунків за визначений період. Звіт доступний для перегляду та завантаження користувачем.

Таблиця 2.16 – Специфікація варіанту використання «Створення статистики про популярні додаткові послуги»

Складова	Опис
Опис прецеденту	Створення статистики про популярні додаткові послуги
Дійові особи прецеденту	Власник
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ «Створення звіту» і натискає кнопку «Популярні послуги».
Базовий потік	1) Система відображає інтерфейс для створення статистики популярних додаткових послуг готелю. 2) Користувач підтверджує створення статистики. 3) Система генерує статистичний звіт, який містить інформацію про популярність різних додаткових послуг готелю. 4) Звіт відображається на екрані користувача або доступний для завантаження.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Користувач отримує статистичний звіт про популярність додаткових послуг готелю

Таблиця 2.17 – Специфікація варіанту використання «Створення статистики щодо бронювання»

Складова	Опис
Опис прецеденту	Створення статистики щодо бронювання
Дійові особи прецеденту	Власник
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ «Створення звіту» і натискає кнопку «Бронювання».
Базовий потік	1) Система відображає форму для створення статистики щодо бронювання, яка містить можливість вибору періоду. 2) Користувач вибирає бажаний період для аналізу. 3) Користувач підтверджує вибір періоду. 4) Система обчислює статистичні дані щодо бронювань. 5) Система відображає отримані статистичні дані. 6) Користувач має можливість зберегти згенеровану статистику.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Користувач отримує статистичні дані щодо бронювань за обраний період і має можливість зберегти цю інформацію.

Таблиця 2.18 – Специфікація варіанту використання «Обробка інформації про послуги»

Складова	Опис
Опис прецеденту	Обробка інформації про послуги
Дійові особи прецеденту	Менеджер
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ «Послуги» і натискає кнопку «Створити/Редагувати/Видалити».
Базовий потік	1) Система відображає список доступних готельних послуг. 2) Користувач має можливість обрати одну з наступних опцій: – "Створити нову послугу": Менеджер може створити нову готельну послугу, вказавши інформацію про її назву, опис, ціну тощо. – "Редагувати існуючу послугу": Менеджер може редагувати існуючу інформацію про готельну послугу, змінюючи назву, опис, ціну тощо. – "Видалити послугу": Менеджер може видалити інформацію про готельну послугу з системи. 3) Після вибору опції, система виконує відповідну дію, і користувач може внести необхідні зміни в інформацію про готельні послуги.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Користувач має можливість створювати, редагувати та видаляти інформацію про готельні послуги, і система зберігає оновлену інформацію для подальшого використання.

Таблиця 2.19 – Специфікація варіанту використання «Обробка інформації про кімнати»

Складова	Опис
Дійові особи прецеденту	Менеджер
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ «Послуги» і натискає кнопку «Створити/Редагувати/Видалити».
Базовий потік	1) Система відображає список доступних кімнат. 2) Користувач має можливість обрати одну з наступних опцій: – "Додати нову кімнату": – "Редагувати існуючу кімнату": Менеджер може редагувати існуючу інформацію про кімнату. – "Видалити кімнату": Менеджер може видалити інформацію про кімнату з системи. 3) Після вибору опції, система виконує відповідну дію, і користувач може внести необхідні зміни в інформацію про кімнати.
Альтернативні потоки	Немає альтернативних потоків.
Постумови	Користувач має можливість створювати, редагувати та видаляти інформацію про кімнати, і система зберігає оновлену інформацію для подальшого використання.

Таблиця 2.20 – Специфікація варіанту використання «Обробка інформації про персонал»

Складова	Опис
1	2
Опис прецеденту	Обробка інформації про персонал
Дійові особи прецеденту	Менеджер
Передумови	Користувач вже авторизований в системі.
Потік подій	Прецедент починається коли користувач, вже авторизований у систему, заходить у розділ «Послуги» і натискає кнопку «Створити/Редагувати/Видалити».
Базовий потік	1) Система відображає список персоналу, який має доступ до мобільного застосунку. 2) Користувач має можливість обрати одну з наступних опцій: – "Додати нового працівника ": – "Редагувати існуючого працівника": Менеджер може редагувати існуючу інформацію про працівника. – "Видалити працівника": Менеджер може видалити інформацію про працівника. 3) Після вибору опції, система виконує відповідну дію, і користувач може внести необхідні зміни в інформацію про персонал.
Альтернативні потоки	Немає альтернативних потоків.

Кінець таблиці 2.20.

1	2
Постумови	Користувач має можливість створювати, редагувати та видаляти інформацію про персонал, і система зберігає оновлену інформацію для подальшого використання.

Розглянемо архітектуру програмного комплексу.

2.2 Архітектура програмного комплексу

Одним із ключових етапів проєктування програмного комплексу є вибір архітектури, яка забезпечить необхідні атрибути якості – масштабованість, безпечність, зручність використання, супроводжуваність тощо. Було проаналізовано кілька архітектурних стилів і обґрунтовано вибір найбільш придатного варіанту.

Для прийняття обґрунтованого рішення щодо архітектурного стилю було проведено порівняльний аналіз п’яти популярних підходів. В таблиці 2.21 подано оцінку їх відповідності ключовим атрибутам якості системи.

Таблиця 2.21 – Порівняльний аналіз архітектурних стилів за атрибутами якості

Атрибут якості	Архітектура шарів абстракцій	Архітектура потоків даних	Класна дошка	Модель-вид-контролер (MVC)	Мікросервісна архітектура
Зручність використання	+2	-1	0	+2	+2
Безпечність	+2	-1	-1	+1	+2
Супроводжуваність	+1	+1	-1	+1	+2
Ефективність	0	0	-1	0	+1
Надійність	+1	-2	0	+1	+2
Масштабованість	0	-1	-2	0	+2
Переносимість	+1	+2	0	+1	+1

Мікросервісна архітектура була обрана як оптимальна завдяки наступним перевагам:

Архітектура шарів абстракцій:

– Сильна підтримка безпечності (+2) та супроводжуваності (+1), оскільки шари дозволяють чітко розділити функції.

– Обмежена масштабованість (0) через складність горизонтального масштабування шарів.

Архітектура потоків даних:

– Недостатня безпечність (-1) через складність контролю за передачею даних.

– Висока переносимість (+2), адже архітектура орієнтована на модульність даних.

Класна дошка:

– Недостатня підтримка супроводжуваності (-1) через централізований механізм обміну даними.

– Слабка масштабованість (-2), адже центр управління є вузьким місцем.

Модель-вид-контролер (MVC):

– Чітке розділення ролей (+2 для зручності).

– Вища супроводжуваність (+1) завдяки ізоляції бізнес-логіки.

Мікросервісна архітектура:

– Максимальна підтримка супроводжуваності (+2) і масштабованості (+2) завдяки незалежності компонентів.

– Висока безпечність (+2), оскільки кожен сервіс може бути ізольованим.

Отже для проєкту найкраще підходить архітектура мікросервісів.

Обґрунтування:

1. Модульність. Система має чітко окреслені функції (вебсайт для клієнтів, мобільний додаток для персоналу, чат-боти), які можуть працювати як окремі сервіси.

2. Масштабованість. У майбутньому можна додати нові сервіси, не змінюючи основний код.

3. Надійність. Збій одного мікросервісу (наприклад, чат-бота) не вплине на інші сервіси.

4. Різноманіття технологій. Різні сервіси можуть використовувати різні мови програмування, бази даних чи фреймворки.

5. Інтеграція. Простота об'єднання з іншими системами (наприклад, платіжними шлюзами чи сторонніми API).

На рисунку 2.2 наведено діаграму компонентів системи, яка ілюструє основні складові архітектури, а також взаємозв'язки між ними.

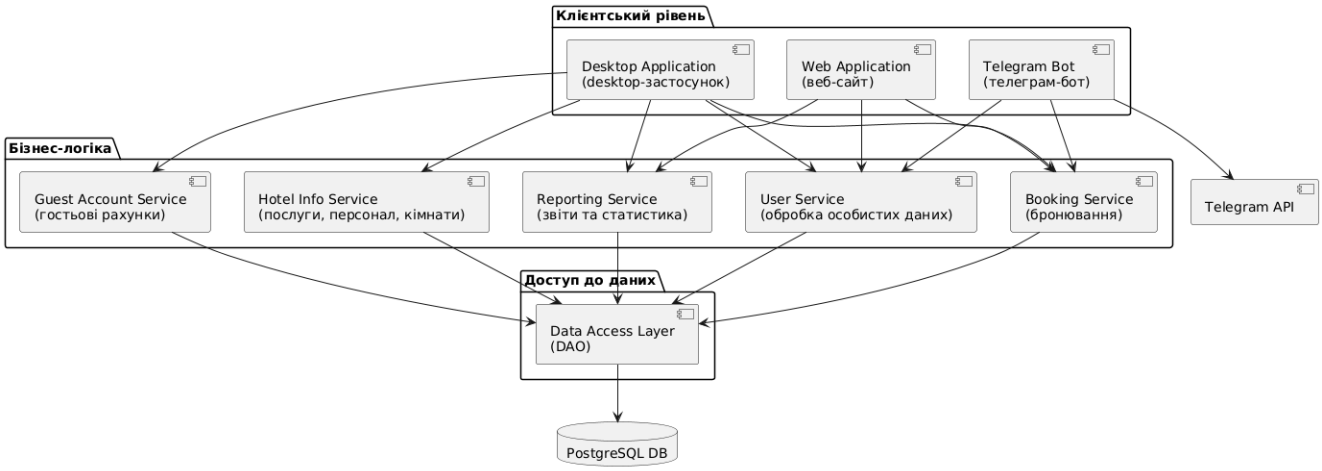

Рисунок 2.2 – Діаграма компонентів програмного комплексу для автоматизації роботи готелю «Lake Park»

Ґрунтуючись на діаграмі, система складається з окремих модулів, зокрема: вебсайту для клієнтів, desktop-застосунку для адміністративного персоналу, чатботу для комунікації, шлюзу API, що координує взаємодію, та централізованої бази даних.

Кожен компонент виконує чітко визначену роль, що відповідає принципам мікросервісної архітектури. Це дозволяє досягти високої гнучкості, розділення

відповідальності, а також забезпечує ізоляцію збоїв між окремими частинами системи.

На рисунку 2.3 зображено діаграму розгортання, яка демонструє, як саме розміщуються компоненти системи на фізичних або віртуальних вузлах інфраструктури.

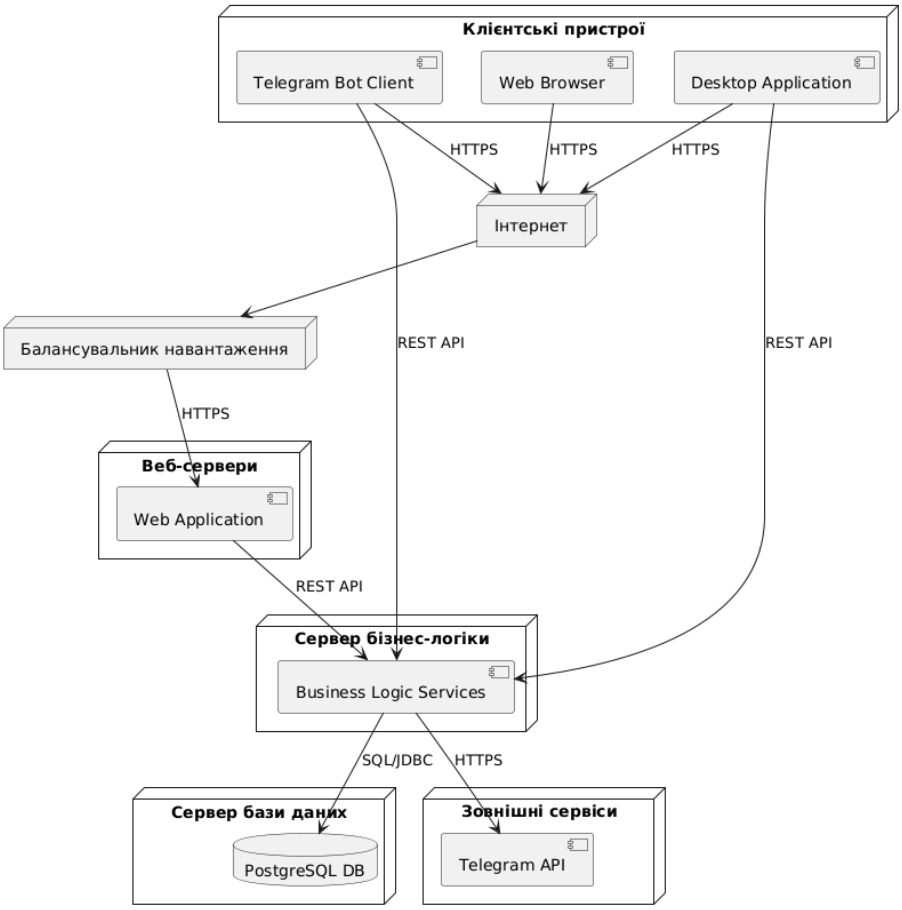

Рисунок 2.3 – Діаграма розгортання програмного комплексу для автоматизації роботи готелю «Lake Park»

Діаграма показує поділ на клієнтську, серверну та інфраструктурну частини. Це дозволяє масштабувати систему в майбутньому без порушення її цілісності.

Розглянемо специфікації компонентів діаграм

1) Вебсайт

Функціональність: онлайн-бронювання, перегляд номерів, інформаційний портал.

Вхідні дані: запити користувачів на пошук номерів, бронювання.

Вихідні дані: інформація про номери, підтвердження бронювань.

Технології: ASP.NET.

Протоколи: HTTPS.

2) Desktop-застосунок

Функціональність: управління бронюваннями, звітність, фінансовий облік.

Вхідні дані: облікові дані персоналу, інформація про номери та бронювання.

Вихідні дані: аналітика, фінансові звіти, оновлені статуси бронювань.

Технології: ASP.NET.

Протоколи: HTTPS, REST API.

3) Чатбот

Функціональність: підтримка клієнтів, бронювання через месенджер.

Вхідні дані: повідомлення клієнтів.

Вихідні дані: відповіді, підтвердження бронювань.

Технології: Python, Telegram Bot API.

Протоколи: HTTPS.

4) Шлюз API

Функціональність: координація між компонентами, обробка запитів.

Вхідні дані: API-запити від вебсайту, desktop-застосунку та чатботу.

Вихідні дані: результати взаємодії з базою даних.

Технології: ASP.NET Web API.

Протоколи: REST API, HTTPS.

5) База даних

Функціональність: зберігання інформації про клієнтів, бронювання, послуги.

Тип даних: реляційна база даних.

Технології: PostgreSQL.

Протоколи: SQL, TLS.

2.3 Проєкт програмного комплексу для автоматизації роботи готелю «Lake Park»

2.3.1 Ескізний проєкт програмного комплексу

У рамках ескізного проєкту були обрані два ключові варіанти використання: «Створення бронювання (вебсайт)» та «Редагування гостьового рахунку». Мета – показати основні етапи користувацького сценарію, а також виявити можливі альтернативні шляхи виконання дій.

Діаграма діяльності для варіанту «Створення бронювання (вебсайт)» зображена на рисунку 2.4.

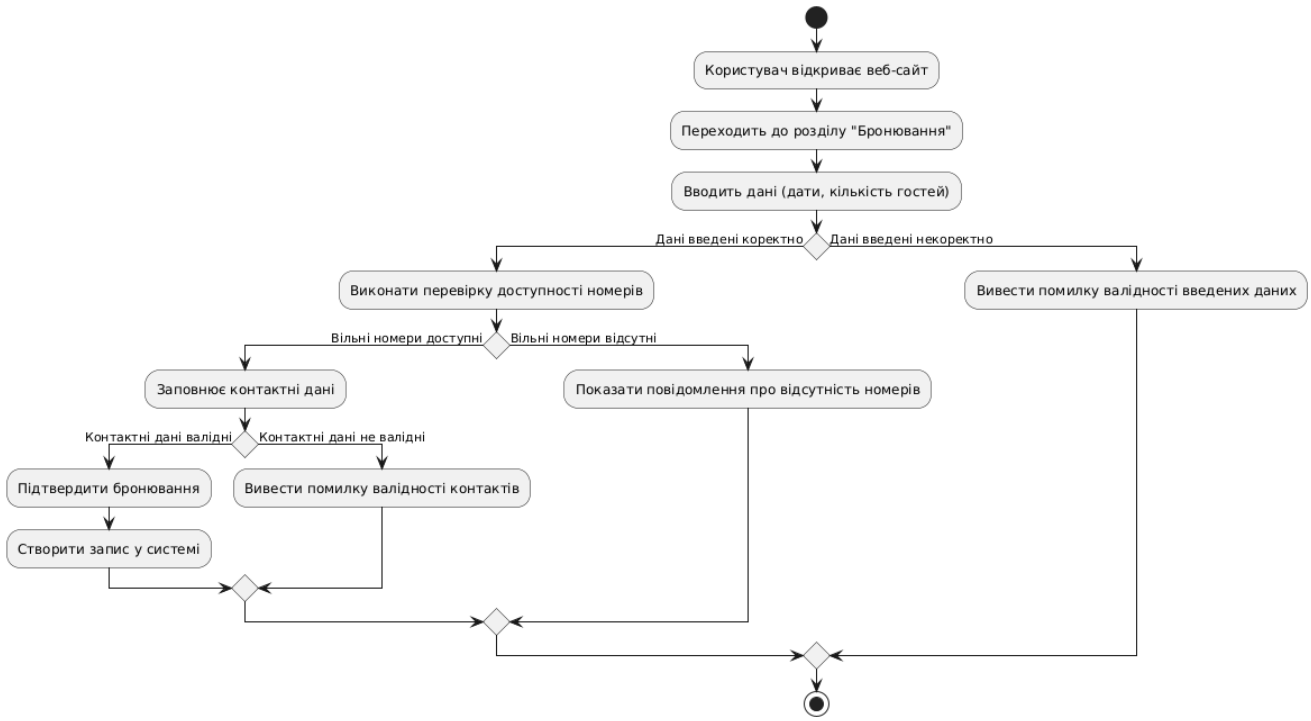

Рисунок 2.4 – Діаграма діяльності для «Створення бронювання (вебсайт)»

Діаграма діяльності для варіанту «Редагування гостьового рахунку» описує послідовність дій, що виконуються адміністратором для внесення змін у інформацію про поточний рахунок гостя. На рисунку 2.5 представлено всі можливі переходи між станами.

Рисунок 2.5 – Діаграма діяльності для «Редагування гостьового рахунку»

Розробка концептуальної моделі даних програмного комплексу для автоматизації роботи готелю «Lake Park»

У межах ескізного проєктування було сформульовано ключові сутності та зв’язки концептуальної моделі даних програмного комплексу для автоматизації роботи готелю «Lake Park». Концептуальна модель даних забезпечує загальне уявлення про структуру інформації, необхідної для функціонування системи, та задає основу для подальшої розробки фізичної схеми бази даних.

Сутності та їхні атрибути:

1) Клієнт

Означення: клієнт – це особа (гість готелю), яка реєструється в системі, виконує бронювання номерів і може користуватися послугами готелю.

Атрибути:

- id;
- ПІБ;

- email;
- пароль;
- стать;
- номер телефону;
- дата народження.

2) Персонал

Означення: персонал – це співробітники готелю.

Атрибути:

- id;
- ПІБ;
- email;
- пароль;
- стать;
- номер телефону;
- дата народження;
- посада.

3) Кімната

Означення: кімната – це окремий номер у готелі, який клієнти можуть забронювати на визначені дати.

Атрибути:

- id;
- номер;
- статус;
- максимальна кількість гостей;
- кількість односпальних ліжок;
- кількість двоспальних ліжок.

4) Бронювання

Означення: бронювання – запис у системі, який фіксує намір клієнта зайняти конкретну кімнату на певний інтервал дат і з певними умовами.

Атрибути:

- id;

- id клієнта;
- id кімнати;
- дата заселення;
- дата виселення;
- кількість гостей;
- статус.

5) Послуги

Означення: послуги – перелік додаткових сервісів, які надає готель клієнтам (харчування, пральня, трансфер тощо). Кожна послуга може бути замовлена клієнтом окремо або у складі бронювання.

Атрибути:

- id;
- найменування;
- ціна;
- статус.

6) Особистий рахунок

Означення: особистий рахунок – фінансовий запис або фактура, пов’язана з конкретним бронюванням. Містить суму за всі надані послуги та проживання, яку клієнт зобов’язаний оплатити.

Атрибути:

- id;
- id бронювання;
- сума.

Коротка схема зв’язків:

1) Клієнт → Бронювання

«Клієнт створює Бронювання»: один клієнт може мати багато бронювань, а в кожному бронюванні є один клієнт (зовнішній ключ id клієнта).

2) Бронювання → Кімната

«Бронювання має Кімнату»: кожне бронювання пов’язане з однією конкретною кімнатою, а одна кімната може фігурувати у багатьох бронюваннях у різні періоди (один-до-багатьох).

3) Бронювання → Особистий рахунок

«Бронювання має Особистий рахунок»: для кожного бронювання формується свій особистий рахунок, і навпаки – кожен особистий рахунок належить саме одному бронюванню (один-до-одного).

4) Клієнт ↔ Послуги

«Клієнт має Послуги» (багато-до-багатьох): один клієнт може скористатися багатьма послугами під час бронювання, а кожна окрема послуга може бути надана багатьом клієнтам.

5) Бронювання ↔ Послуги

«Бронювання має Послуги» (багато-до-багатьох): окреме бронювання може включати декілька додаткових послуг, а кожна послуга може бути включена до багатьох бронювань різних клієнтів.

На рисунку 2.6 зображено концептуальну модель даних програмного комплексу для автоматизації роботи готелю «Lake Park»

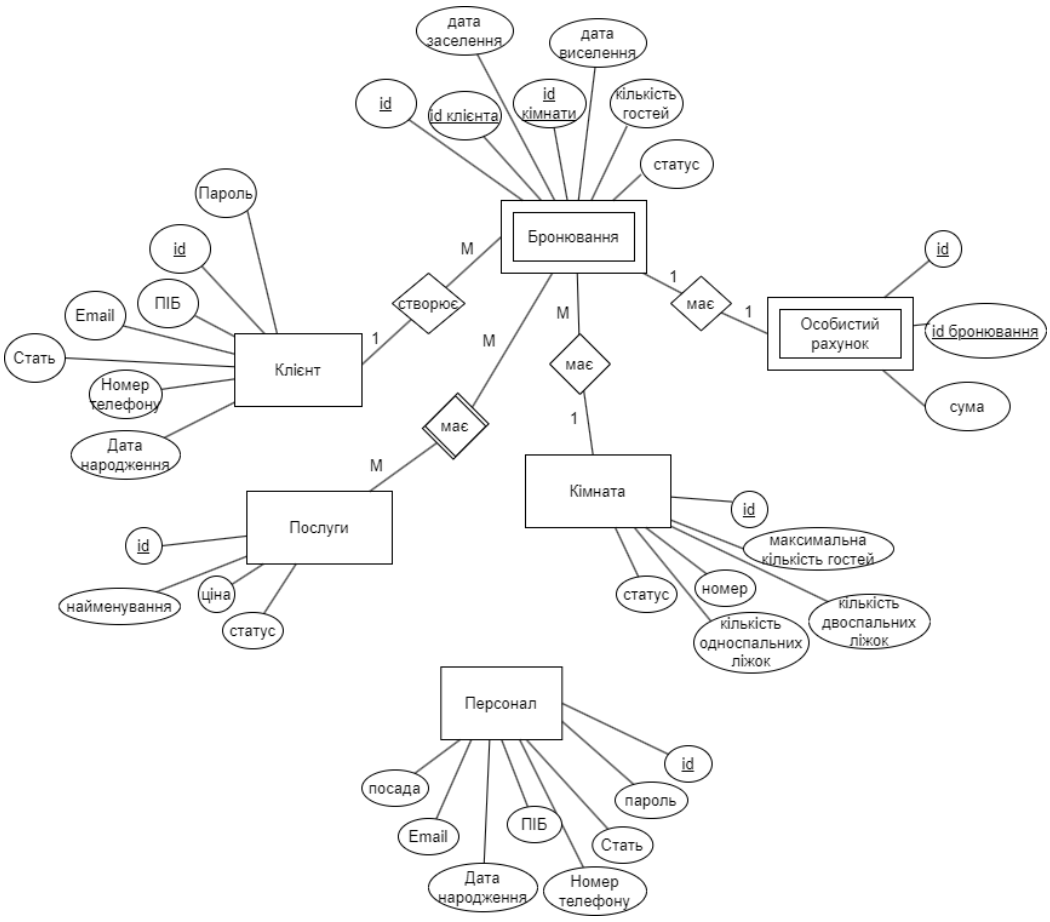

Рисунок 2.6 – Концептуальна модель даних програмного комплексу для автоматизації роботи готелю «Lake Park»

Розробка прототипу інтерфейсу.

У рамках підготовки ескізного проєкту було здійснено декілька варіантів візуального оформлення сторінки створення бронювання. Мета – визначити оптимальне розташування елементів керування, читабельність інформації та ергономіку інтерфейсу. Нижче наведено три початкових прототипи, кожен із яких має власні особливості компонування та стилістики.

Рисунок 2.7 – Перший прототип сторінки «Створити бронювання»

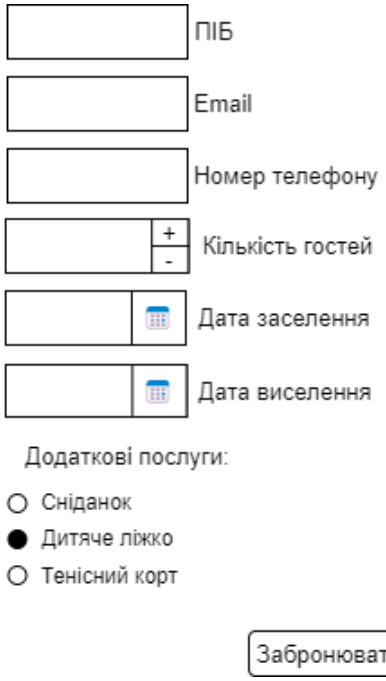

Рисунок 2.8 – Другий прототип сторінки «Створити бронювання»

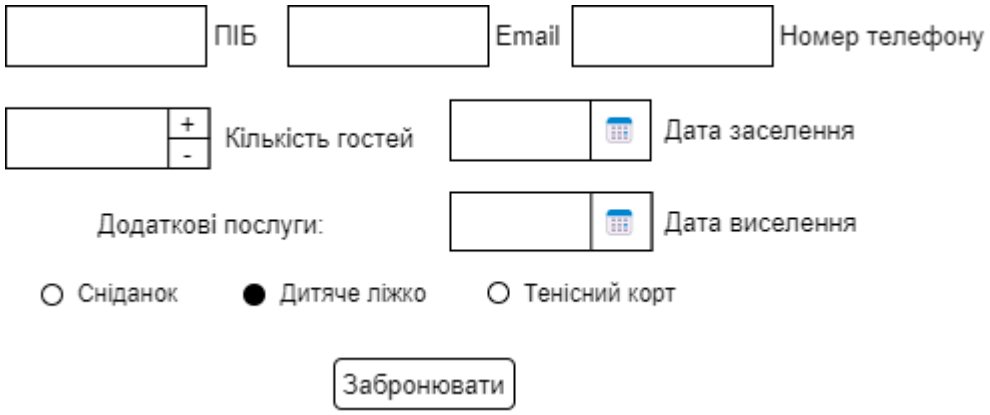

Рисунок 2.9 – Третій прототип сторінки «Створити бронювання»

У результаті порівняння трьох макетів та аналізу переваг другого прототипу було зроблено висновок, що саме цей варіант найкраще відповідає поточним вимогам. По-перше, достатні відступи між елементами забезпечують належну читабельність, особливо для користувачів із різною роздільною здатністю екранів. По-друге, інтеграція блоку «Додаткові послуги» безпосередньо під основною формою створює логічний лінійний потік дій: спочатку обирається основне (дати й тип кімнати), а потім додаються опції. Оскільки в майбутньому планується розширення переліку сервісів, компактне відображення цього блоку в рамках одного вікна виглядає оптимальним рішенням. Саме тому другий прототип було обрано як базовий для подальшої деталізації.

Після визначення оптимальної розкладки основних елементів було проведено підбір кольорових схем та стилістики тексту, що відповідають загальному стилю готелю «Lake Park» та сприяють формуванню відповідного настрою користувача.

На рисунках 2.10 - 2.12 наведено три експериментальні схеми палітр, кожна з яких поєднує колір фону, акцентні елементи (наприклад, кнопки, заголовки) та кольори тексту. Палітри було обрано на вебсайті Color Hunt[4]

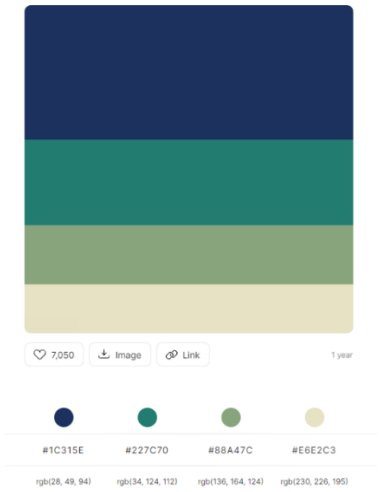

Рис 2.10 – Перший варіант кольору для оформлення сайту

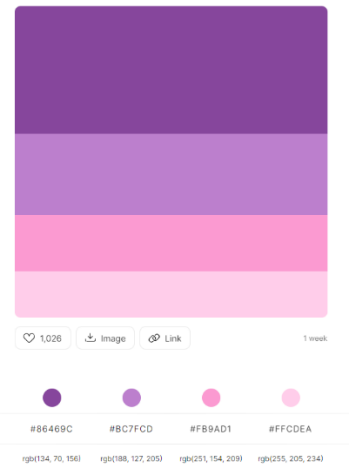

Рис 2.11 – Другий варіант кольору для оформлення сайту

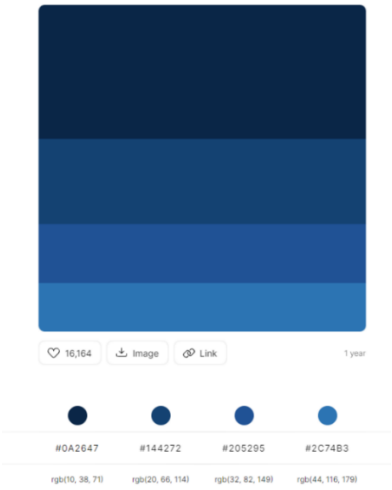

Рис 2.12 – Третій варіант кольору для оформлення сайту

З огляду на загальну стилістику інтер’єру готелю та очікуваний психологічний ефект, найбільш вдалим визнано перший варіант. Поєднання блакитних та коричневих тонів нагадує природні елементи парку біля озера, що створює відчуття затишку й довіри для користувача.

На рисунках 2.13 - 2.15 представлені три підходи до типографіки, у яких розглядаються різні поєднання шрифтів.

Рис 2.13 – Перший варіант шрифту для оформлення тексту

Рис 2.14 – Другий варіант шрифту для оформлення тексту

Рис 2.15 – Третій варіант шрифту для оформлення тексту

З урахуванням загальної концепції скорішого сприйняття та збереження професійного вигляду інтерфейсу, найкращим обрано перший варіант шрифту. Він забезпечує достатню контрастність і не перевантажує візуальне сприйняття при великій кількості інформації на сторінці.

Остаточні прототипи сторінок зображено на рисунках 2.16 - 2.19.

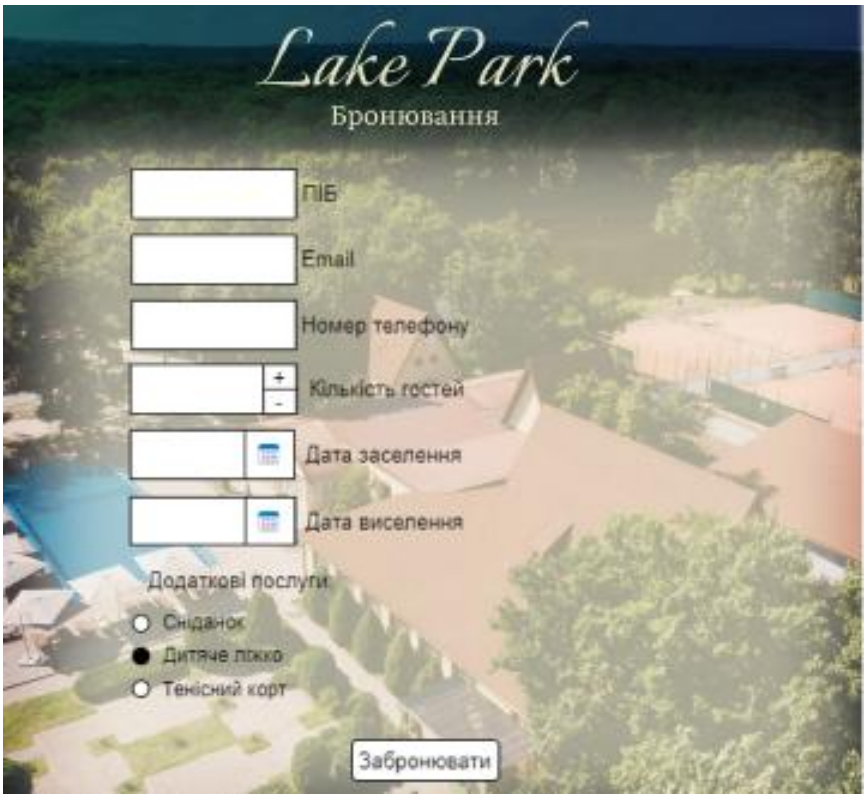

Рисунок 2.16 – Прототип сторінки «Створити бронювання» на вебсайті

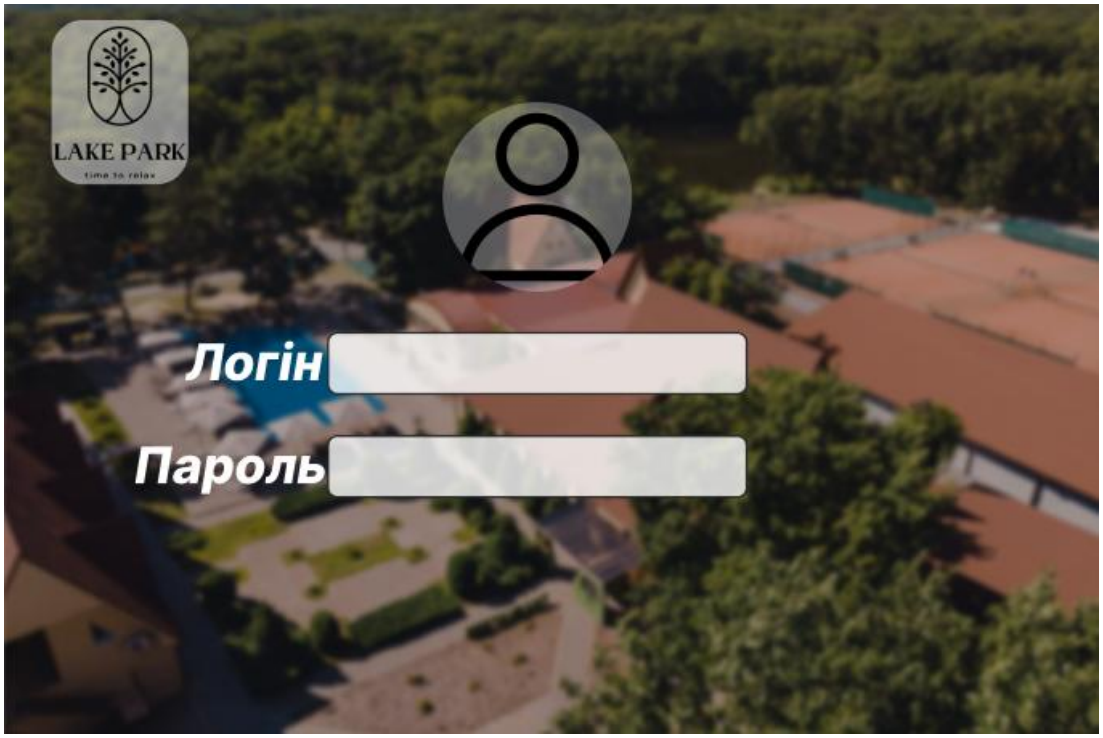

Рисунок 2.17 – Прототип сторінки «Авторизація» у desktop-застосунку

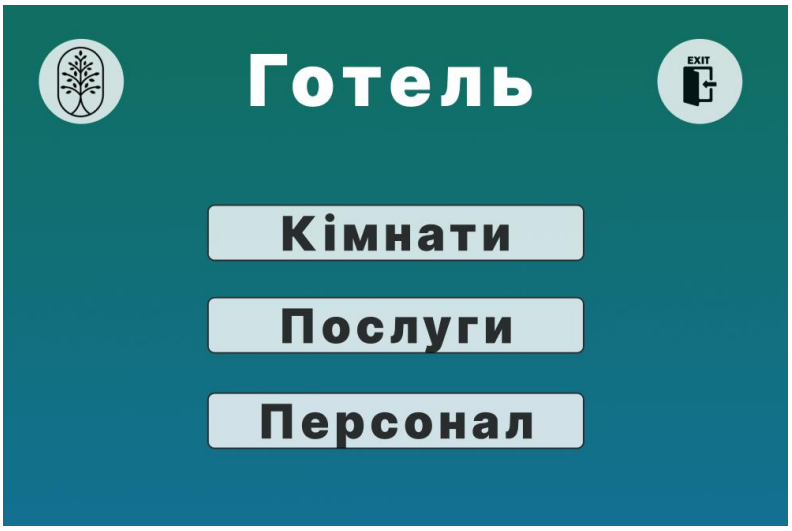

Рисунок 2.18 – Прототип сторінки Готель у desktop-застосунку

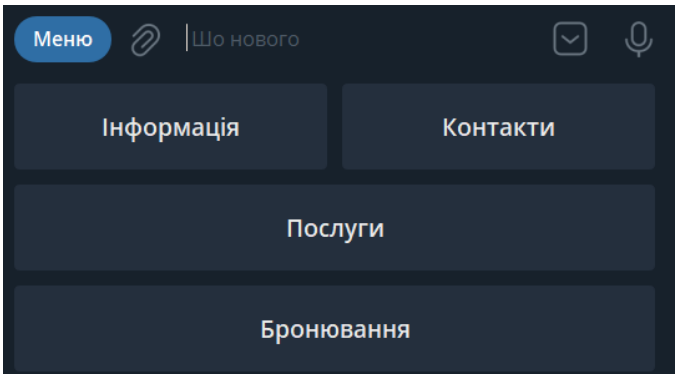

Рисунок 2.19 – Прототип форми меню Telegram боту

Далі розглянемо технічний проєкт програмного комплексу.

2.3.2 Технічний проєкт програмного комплексу

Діаграма класів

На цьому етапі проєктування було побудовано діаграму класів, що відображає основні сутності системи, їх атрибути та взаємозв’язки. Вона служить логічною моделлю структури об’єктів програмного забезпечення та є важливим інструментом для реалізації об’єктно-орієнтованого підходу. Діаграма класів програмного комплексу для автоматизації роботи готелю «Lake Park» представлена на рисунку 2.20.

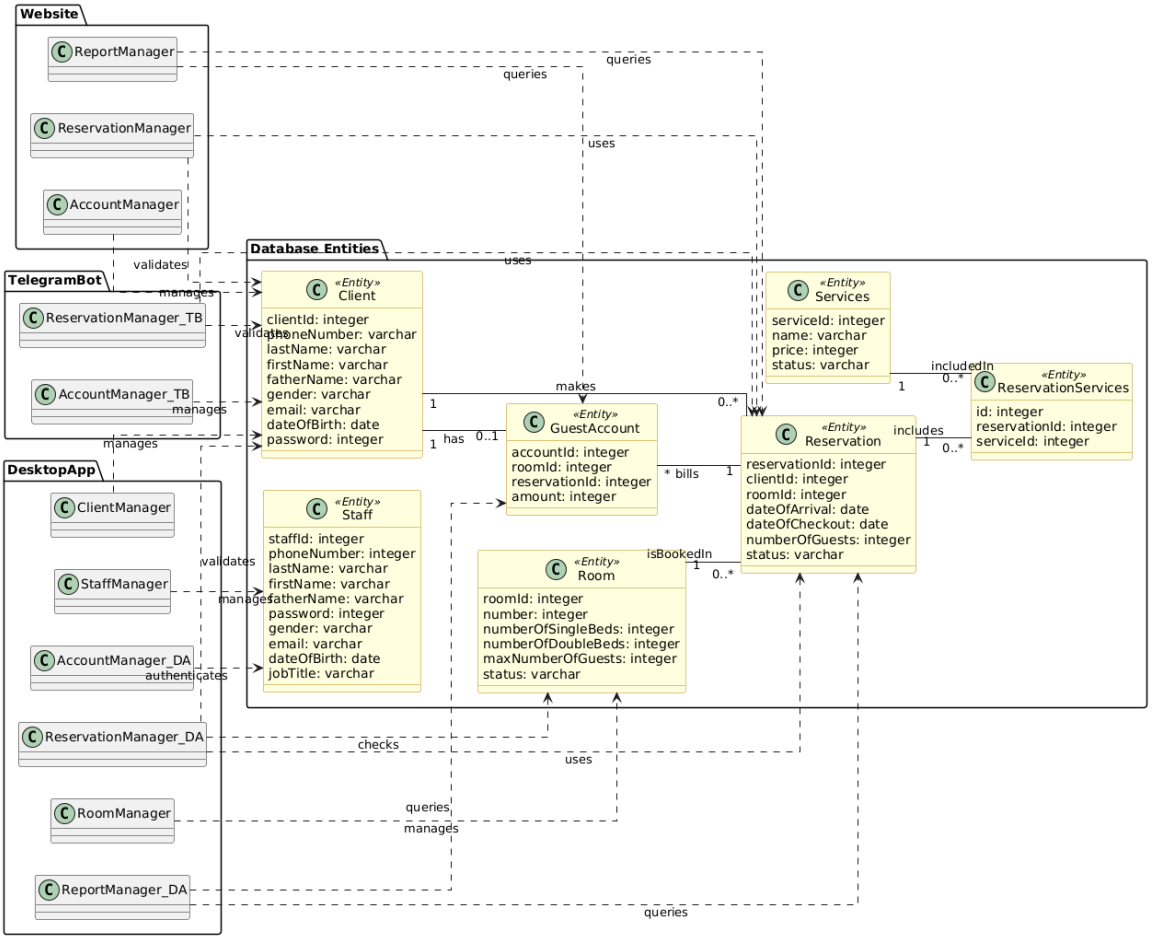

Рисунок 2.20 – Діаграма класів програмного комплексу для автоматизації роботи готелю «Lake Park»

На рисунках 2.21 - 2.22 зображено повну структуру компонентів.

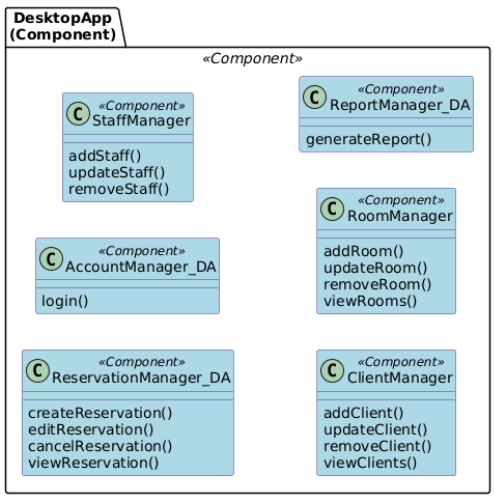

Рисунок 2.21 – Структура компоненту desktop-застосунок

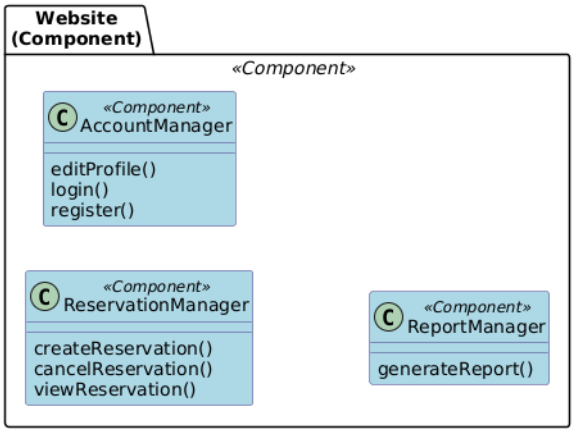

Рисунок 2.22 – Структура компоненту вебсайт

Для забезпечення чіткого розуміння логіки роботи програмного комплексу розроблено специфікації основних класів.

1) Entity-класи (Database Entities):

Client.

Призначення: зберігає інформацію про клієнта готелю.

Атрибути:

- clientId: integer – унікальний ідентифікатор;
- phoneNumber: varchar – телефон;
- lastName: varchar – прізвище;
- firstName: varchar – ім’я;
- fatherName: varchar – по батькові;
- gender: varchar – стать;
- email: varchar – електронна пошта;
- dateOfBirth: date – дата народження;
- password: integer – пароль (хеш або код).

Staff.

Призначення: зберігає дані працівника готелю.

Атрибути:

- staffId: integer;
- phoneNumber: integer;

- lastName: varchar;
- firstName: varchar;
- fatherName: varchar;
- password: integer;
- gender: varchar;
- email: varchar;
- dateOfBirth: date;
- jobTitle: varchar – посада.

Room.

Призначення: описує номер готелю.

Атрибути:

- roomId: integer;
- number: integer – номер кімнати;
- numberOfSingleBeds: integer;
- numberOfDoubleBeds: integer;
- maxNumberOfGuests: integer;
- status: varchar – доступність/зайнятість.

Reservation.

Призначення: зберігає інформацію про бронювання.

Атрибути:

- reservationId: integer;
- clientId: integer – FK на Client;
- roomId: integer – FK на Room (може бути null);
- dateOfArrival: date;
- dateOfCheckout: date;
- numberOfGuests: integer;
- status: varchar – стан бронювання (наприклад, “Confirmed”, “Cancelled”).

GuestAccount.

Призначення: обліковий запис гостя (рахунок на оплату).

Атрибути:

- accountId: integer;
- roomId: integer – FK на Room;
- reservationId: integer – FK на Reservation;
- amount: integer – сума рахунку.

Services.

Призначення: довідник додаткових послуг.

Атрибути:

- serviceId: integer;
- name: varchar – назва послуги;
- price: integer – ціна;
- status: varchar – активна/неактивна.

ReservationServices.

Призначення: проміжна таблиця для зв’язку бронювання й послуг.

Атрибути:

- id: integer;
- reservationId: integer – FK на Reservation;
- serviceId: integer – FK на Services.

2) Component-класи.

Website (пакет Website).

ReservationManager.

Призначення: логіка бронювання через вебінтерфейс.

Методи:

- createReservation(clientId: int, roomId: int, arrival: Date, checkout: Date, guests: int, services: List<int>): boolean;
- cancelReservation(reservationId: int): boolean;
- viewReservation(reservationId: int): Reservation.

AccountManager.

Призначення: управління обліковим записом клієнта на сайті.

Методи:

- register(clientData: Client): Client;
- login(phone: String, password: String): boolean;
- editProfile(clientId: int, updatedData: Client): boolean.

ReportManager.

Призначення: формування звітів для власника.

Методи:

- generateBookingStats(startDate: Date, endDate: Date): BookingStats;
- generateServicePopularity(startDate: Date, endDate: Date): ServiceStats;
- generateRevenueReport(startDate: Date, endDate: Date): RevenueReport.

DesktopApp (пакет DesktopApp).

ReservationManager_DA.

Призначення: управління бронюваннями для співробітників.

Методи:

- createReservation(...) (параметри як у Website);
- editReservation(reservationId: int, updatedData: Reservation): boolean;
- cancelReservation(reservationId: int): boolean;
- viewReservation(reservationId: int): Reservation.

AccountManager_DA.

Призначення: аутентифікація співробітників.

Методи:

- login(staffPhone: String, password: String): Staff.

StaffManager.

Призначення: CRUD-операції з даними персоналу.

Методи:

- addStaff(staffData: Staff): Staff;
- updateStaff(staffId: int, updatedData: Staff): boolean;
- removeStaff(staffId: int): boolean.

ClientManager.

Призначення: CRUD-операції з клієнтами у десктоп-застосунку.

Методи:

- addClient(clientData: Client): Client;
- updateClient(clientId: int, updatedData: Client): boolean;
- removeClient(clientId: int): boolean;
- viewClients(): List<Client>.

RoomManager.

Призначення: CRUD-операції з номерами.

Методи:

- addRoom(roomData: Room): Room;
- updateRoom(roomId: int, updatedData: Room): boolean;
- removeRoom(roomId: int): boolean;
- viewRooms(): List<Room>.

ReportManager_DA.

Призначення: формування звітів у десктоп-застосунку.

Методи:

- generateReport(type: ReportType, startDate: Date, endDate: Date): Report.

Динамічна модель описує поведінку системи в часі та ілюструє порядок взаємодії між об’єктами під час виконання ключових сценаріїв. Для цього використовуються діаграми послідовностей, що детально відображають обмін повідомленнями між компонентами програмного комплексу.

На рисунках 2.23 та 2.24 зображено діаграми послідовності для варіантів використання «Створення бронювання (вебсайт)», «Редагування гостьового рахунку» відповідно.

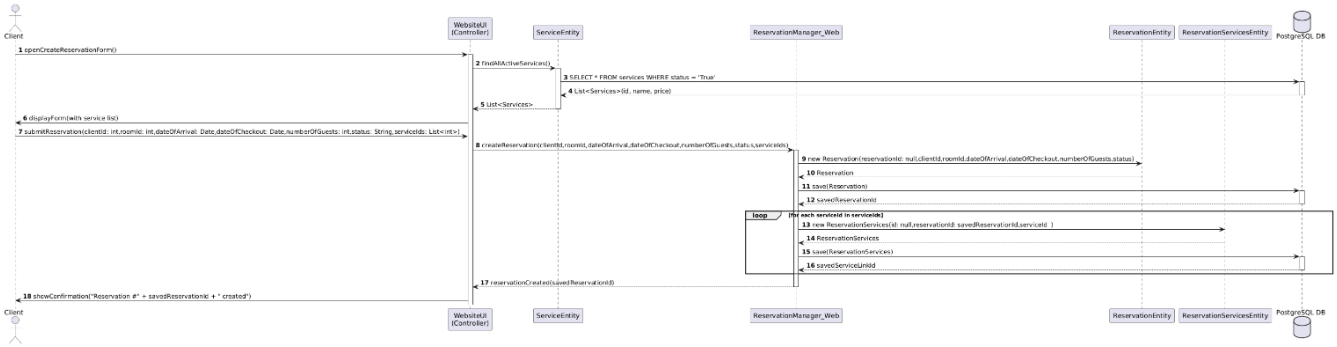

Рисунок 2.23 – Діаграма послідовності для варіанту використання «Створення бронювання (вебсайт)»

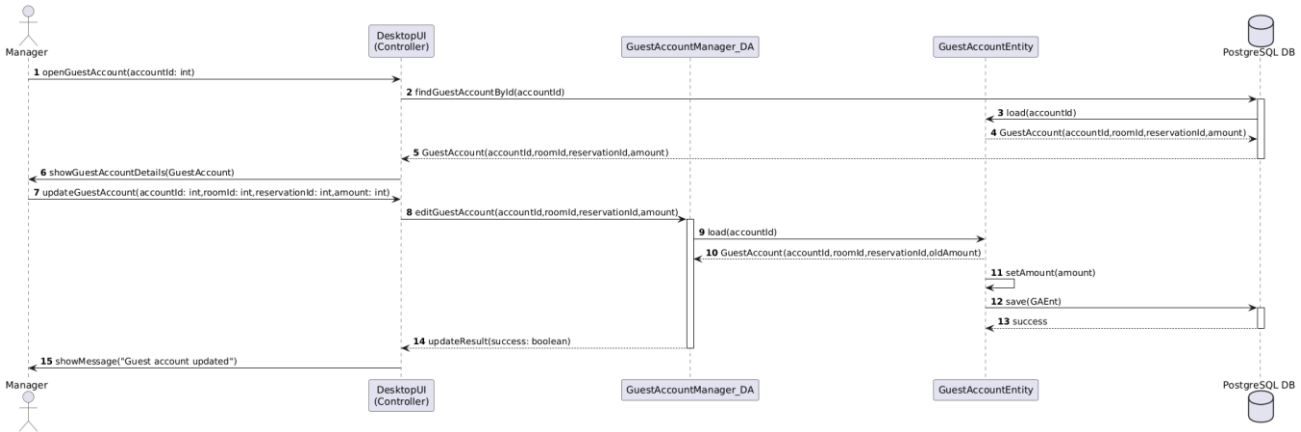

Рисунок 2.24 – Діаграма послідовності для варіанту використання «Редагування гостьового рахунку»

Логічна модель бази даних виконує роль містка між концептуальною моделлю та фізичною реалізацією схеми. На цьому етапі уточняються типи полів, формуються таблиці, встановлюються зовнішні ключі та обмеження для забезпечення цілісності даних.

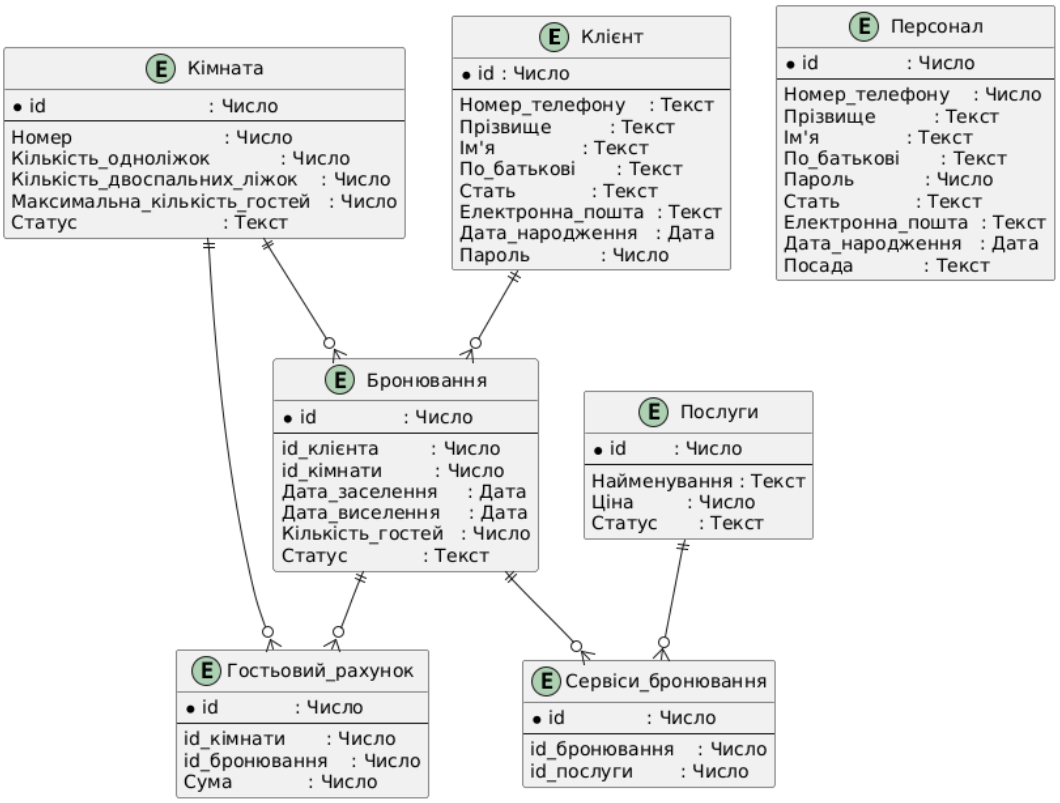

Рисунок 2.25 – Логічна модель даних програмного комплексу для автоматизації роботи готелю «Lake Park»

Таким чином було розроблена логічна модель даних.

2.3.3 Робочий проєкт програмного комплексу

Вибір інструментарію розробки

У даній кваліфікаційній роботі буде розроблено комплексну систему, що складається з трьох основних компонентів: telegram-бота, desktop-застосунку та вебсайту. Для кожного з них обрано сучасні технології, що дозволяють забезпечити високий рівень функціональності, безпеки та зручності у використанні, а також сприяють ефективній розробці та підтримці продукту.

Telegram-бот:

– розробка логіки бота здійснюватиметься мовою Python, яка відома своєю простотою та високою читабельністю. Простота та зручність використання цієї мови сприяють швидкому та ефективному розробленню складних систем [5].

Завдяки широкій спільноті розробників, Python дозволяє швидко знаходити рішення для виникаючих проблем, що є суттєвим плюсом при розробці інтерактивних ботів.

– для інтеграції з платформою Telegram буде використано Telegram Bot API, який дозволяє реалізувати повноцінну взаємодію з користувачами. API надає можливість безпосередньо відправляти та отримувати повідомлення, що відкриває широкі можливості для реалізації різних функціональних модулів [6]. Такий підхід забезпечує точну та оперативну обробку запитів, що дозволить створити зручний та інтуїтивно зрозумілий інтерфейс для користувачів.

– розробка коду буде здійснюватися в інтегрованому середовищі PyCharm, яке відоме своїми потужними інструментами для відлагодження та тестування. Це середовище забезпечує потужні можливості для аналізу коду та інтеграцію з системами контролю версій, що сприяє підтримці високої якості розробки [7]. Використання PyCharm дозволить ефективно керувати проектом та оперативно виявляти і виправляти помилки, що є важливим аспектом у створенні стабільного Telegram-бота.

Desktop застосунок:

– розробка desktop-застосунку базуватиметься на технології .NET MAUI, яка дозволяє створювати крос-платформні рішення з єдиною кодовою базою. Фреймворк забезпечує можливість створювати застосунки для різних операційних систем, що суттєво спрощує процес розробки та подальшу підтримку продукту [8]. Завдяки цій технології буде забезпечено високий рівень сумісності та зручність використання застосунку на різних пристроях.

– основною мовою програмування для desktop застосунку обрано C#, яка забезпечує потужний об’єктно-орієнтований підхід до розробки. За даними C# Guide, мова характеризується як сучасна мова програмування для створення надійних застосунків, що відповідає вимогам проєкту [9]. Використання C# дозволяє ефективно використовувати сучасні парадигми розробки, забезпечуючи стабільність та масштабованість системи.

– для розробки застосунку буде застосовано середовище Rider від JetBrains, яке підтримує інтеграцію з .NET[8]. Rider забезпечує інтегровані можливості для тестування та налагодження, що сприяє ефективному процесу розробки [10]. Завдяки цьому інструменту розробники зможуть оптимізувати робочий процес, оперативно виявляти помилки та підвищувати якість кінцевого продукту.

Вебсайт:

– вебсайт розроблятиметься з використанням Blazor, що дозволяє писати як клієнтський, так і серверний код мовою C#. Blazor забезпечує можливість створювати інтерактивні вебдодатки з єдиною екосистемою, що сприяє більш узгодженій розробці та легкому впровадженню нових функціональних можливостей [11]. Це дозволяє забезпечити високий рівень інтерактивності та динамічності користувацького інтерфейсу.

– для реалізації механізмів автентифікації та авторизації користувачів буде інтегровано .NET Identity Framework. Цей інструмент надає вбудовану підтримку безпеки та авторизації, що є критично важливим для захисту даних користувачів [12]. Використання Identity Framework дозволить створити надійну систему управління користувачами з гнучкими налаштуваннями рівнів доступу.

– зовнішній вигляд вебсайту буде оформлено за допомогою Bootstrap, який є найпопулярнішою HTML, CSS та JS платформою для розробки адаптивного дизайну [13]. Bootstrap дозволяє швидко створити сучасний та адаптивний інтерфейс, що автоматично підлаштовується під різні розміри екранів і пристроїв. Завдяки широкому набору готових компонентів, розробники зможуть зосередитись на логікі застосунку, не витрачаючи час на розробку базового дизайну.

– розробка вебсайту також здійснюватиметься в середовищі Rider, що забезпечує сучасні інструменти для налагодження та тестування коду. Використання мови C# у контексті вебсайту дозволить інтегрувати серверну та клієнтську логіку в єдину систему, що спрощує підтримку та розширення функціональності. Завдяки єдиній екосистемі розробки забезпечується високий рівень узгодженості між різними компонентами проекту.

База даних та розгортання:

– для зберігання та обробки даних буде використовуватись PostgreSQL, яка відома своєю надійністю та високою продуктивністю. PostgreSQL описується як одна з найбільш передових систем управління базами даних з відкритим вихідним кодом, що забезпечує стабільну роботу з великими обсягами інформації [14]. Завдяки своїй масштабованості, ця СУБД дозволить ефективно управляти даними навіть при збільшенні навантаження на систему.

– для локального розгортання вебсайту та тестування інтеграції компонентів буде використовуватись Open Server Panel. Це середовище забезпечує зручне середовище для розгортання вебпроєктів, що спрощує налаштування серверних служб та автоматизує процес тестування [15]. Використання Open Server Panel дозволить швидко впроваджувати зміни в конфігурації та оперативно перевіряти їхню ефективність, що є важливим для підтримки високої якості роботи системи.

Реалізація та тестування основних класів еквівалентності програмного забезпечення

Під час розробки програмного комплексу для автоматизації управління готелем “Lake Park” було використано середовища розробки, такі як PyCharm та Rider, СУБД PostgreSQL та мови програмування Python і C#. Для фронтенду вебсайту застосовувався Blazor. А також використано фреймворк .NET.

Перед безпосереднім описом фрагменту коду розглянемо сценарії тестування за методом чорної скриньки, розбивши усі можливі шляхи виконання на два класи еквівалентності: безпомилковий та із помилками.

Розглянемо успішний сценарій.

Ініціалізація:

– Система завантажує список доступних сервісів із БД (LakeParkService.GetServicesAsync → availableServices).

– Користувач вводить необхідні дані (прізвище, ім’я, по-батькові, коректний телефон і Email, дату заїзду < дати виїзду, кількість гостей у межах 1-5).

– Вибір додаткових послуг (за потреби ставить відмітки біля одного або кількох записів у списку послуг).

– Натискання “Забронювати”.

– Система перевіряє валідацію всіх полів → всі пройшли успішно.

– Запис у базу даних (додається Client → DbContext.SaveChangesAsync(), створюється Reservation зі статусом “Pending” → SaveChangesAsync(), для кожного вибраного ServiceViewModel створюється ReservationService → SaveChangesAsync()).

– Підтвердження (у resultMessage з’являється “Бронювання успішно створено!”, відображається в alert).

Розглянемо сценарії із помилками.

1) Пропущені обов’язкові поля:

– Система виявляє відсутній LastName/FirstName/…/NumberOfGuests.

– Підсвічує відповідні поля, показує повідомлення “Required”.

– Форма не відправляється.

2) Невірний формат Email:

– Email без “@” або “.” → валідатор EmailAddress спрацьовує.

– Відображається “The Email field is not a valid e-mail address.”.

– Запити до сервера не йдуть.

3) Невірний інтервал дат:

– DateOfCheckout ≤ DateOfArrival.

– Користувач бачить повідомлення про некоректний діапазон.

– БД без змін.

4) Кількість гостей поза діапазоном:

– Значення <1 або >5 → валідатор Range.

– Відображається “The field NumberOfGuests must be between 1 and 5.”.

– Без записів у таблицю Reservations.

5) Системні помилки (мережа/БД):

– Падіння SaveChangesAsync().

– Блок “try/catch” показує: “Не вдалося створити бронювання, спробуйте пізніше”.

– Ніяких часткових записів у таблицях.

Нижче представлено фрагмент коду з вебсайту, а саме форма, яка відповідає за бронювання.

Цей компонент Blazor реалізує форму бронювання, яка дозволяє користувачу вводити необхідні дані для створення бронювання, а також вибирати послуги зі списку доступних (список надається з бази даних). Після відправки форми створюються відповідні записи у базі даних за допомогою сервісу LakeParkService та контексту LakeParkContext. Для реалізації було використано .NET Core, Blazor Server та Entity Framework Core.

```
@page "/booking"
@using System.ComponentModel.DataAnnotations
@using Microsoft.AspNetCore.Components.Forms
@using Microsoft.AspNetCore.Antiforgery
@inject website.Components.LakeParkContext DbContext
@inject website.Components.LakeParkService LakeParkService

@rendermode @(new InteractiveServerRenderMode(prerender: true))

<EditForm Model="@bookingModel"
          FormName="BookingForm"
          OnValidSubmit="@HandleBooking">
    <AntiforgeryToken />
    <DataAnnotationsValidator />
    <ValidationSummary />

    <div class="mb-3">
        <label for="lastName">Прізвище</label>
        <input id="lastName"
               type="text"
               class="form-control"
               @bind="bookingModel.LastName" />
    </div>
```

```
<div class="mb-3">
    <label for="firstName">Ім’я</label>
    <input id="firstName"
           type="text"
           class="form-control"
           @bind="bookingModel.FirstName" />
</div>
<div class="mb-3">
    <label for="fatherName">По-батькові</label>
    <input id="fatherName"
           type="text"
           class="form-control"
           @bind="bookingModel.FatherName" />
</div>
<div class="mb-3">
    <label for="phone">Номер телефону</label>
    <input id="phone"
           type="tel"
           class="form-control"
           @bind="bookingModel.PhoneNumber" />
</div>
<div class="mb-3">
    <label for="email">Email</label>
    <input id="email"
           type="email"
           class="form-control"
           @bind="bookingModel.Email" />
</div>
<div class="row mb-3">
    <div class="col">
        <label for="arrival">Дата заселення</label>
        <input id="arrival"
               type="date"
               class="form-control"
               @bind="bookingModel.DateOfArrival" />
    </div>
```

```
    <div class="col">
        <label for="checkout">Дата виселення</label>
        <input id="checkout"
               type="date"
               class="form-control"
               @bind="bookingModel.DateOfCheckout" />
    </div>
</div>
<div class="mb-3">
    <label for="guests">Кількість гостей</label>
    <input id="guests"
           type="number"
           class="form-control"
           @bind="bookingModel.NumberOfGuests" />
</div>
<div class="mb-3">
    <label>Послуги</label>
    @if (availableServices == null)
    {
        <p>Завантаження послуг...</p>
    }
    else
    {
        @foreach (var svc in availableServices)
        {
            <div class="form-check">
                <input type="checkbox"
                       class="form-check-input"
                       @bind="svc.IsSelected" />
                <label class="form-check-label">
                    @svc.Name (@(svc.Price ?? 0)) грн
                </label>
            </div>
        }
    }
</div>
```

```
    <button type="submit" class="btn btn-success">Забронювати</button>
</EditForm>

@if (!string.IsNullOrEmpty(resultMessage))
{
    <div class="alert alert-info mt-3">@resultMessage</div>
}

@code {
    private BookingModel bookingModel = new();
    private List<ServiceViewModel> availableServices;
    private string resultMessage;

    protected override async Task OnInitializedAsync()
    {
        var services = await LakeParkService.GetServicesAsync();
        availableServices = services.Select(s => new ServiceViewModel {
            Id         = s.Id,
            Name       = s.Name,
            Price      = s.Price,
            IsSelected = false
        }).ToList();
    }

    private async Task HandleBooking()
    {
        var client = new website.Components.Client
        {
            LastName    = bookingModel.LastName,
            FirstName   = bookingModel.FirstName,
            FatherName  = bookingModel.FatherName,
            Phonenumber = bookingModel.PhoneNumber,
            Email       = bookingModel.Email,
            DateOfBirth = new DateTime(1900, 1, 1)
        };
```

```
            DbContext.Clients.Add(client);
            await DbContext.SaveChangesAsync();

            var reservation = new website.Components.Reservation
            {
                IdClient       = client.Id,
                DateOfArrival  = bookingModel.DateOfArrival,
                DateOfCheckout = bookingModel.DateOfCheckout,
                NumberOfGuests = bookingModel.NumberOfGuests,
                Status         = "Pending"
            };
            DbContext.Reservations.Add(reservation);
            await DbContext.SaveChangesAsync();

            foreach (var svc in availableServices.Where(x => x.IsSelected))
            {
                DbContext.ReservationServices.Add(new
website.Components.ReservationService
                {
                    IdReservation = reservation.Id,
                    IdServices    = svc.Id
                });
            }
            await DbContext.SaveChangesAsync();

            resultMessage = "Бронювання успішно створено!";
        }

        public class BookingModel
        {
            [Required] public string LastName       { get; set; } = "";
            [Required] public string FirstName      { get; set; } = "";
            [Required] public string FatherName     { get; set; } = "";
            [Required] public string PhoneNumber    { get; set; } = "";
            [Required, EmailAddress]
            public string Email                      { get; set; } = "";
```

```
            [Required] public DateTime DateOfArrival     { get; set; } =
DateTime.Today;
            [Required] public DateTime DateOfCheckout    { get; set; } =
DateTime.Today.AddDays(1);
            [Required, Range(1,5)]
            public int NumberOfGuests               { get; set; } = 1;
        }

        public class ServiceViewModel
        {
            public int    Id         { get; set; }
            public string Name       { get; set; }
            public int?   Price      { get; set; }
            public bool   IsSelected { get; set; }
        }
    }
```

Цей фрагмент коду – Blazor Server-сторінка /booking, яка реалізує форму бронювання номера в готелі, це повноцінна форма бронювання з підтримкою валідації, збереженням у БД та вибором додаткових послуг.

Опис функціональності:

– форма вводу даних збирає інформацію про клієнта (ПІБ, телефон, email), дати проживання, кількість гостей та обрані додаткові послуги;

– використовуються атрибути DataAnnotations для перевірки правильності введених даних;

– список доступних послуг завантажується асинхронно при ініціалізації компонента;

– після надсилання форми:

1) створюється запис про клієнта;

2) створюється нове бронювання;

3) додаються обрані послуги до цього бронювання.

– використовуються Blazor-ін’єкції – DbContext для роботи з базою даних і сервіс LakeParkService для отримання послуг;

– включено AntiforgeryToken для захисту від CSRF-атак;

– після успішного бронювання показується повідомлення користувачу.

Технології, що були використані:

1) Blazor Server (@page, EditForm, @bind):

– @page – визначає маршрут сторінки (/booking), щоб до неї можна було звернутися з браузера;

– EditForm – компонент для створення інтерактивної форми з підтримкою валідації;

– @bind – двостороннє прив'язування даних між полями форми та моделлю (BookingModel), що дозволяє автоматично оновлювати значення при введенні.

2) EF Core (DbContext):

– використовується для взаємодії з базою даних: збереження клієнтів, бронювань та обраних послуг;

– DbContext надає доступ до таблиць через властивості (Clients, Reservations, ReservationServices).

3) ASP.NET Antiforgery:

– AntiforgeryToken додає токен до форми, щоб захистити її від атак типу CSRF (міжсайтової підробки запитів);

– гарантує, що форму надсилає саме дозволений користувач із цього ж сайту.

Код програмного забезпечення представлено в додатку Б.

3 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО КОМПЛЕКСУ ДЛЯ АВТОМАТИЗАЦІЇ РОБОТИ ГОТЕЛЮ «LAKE PARK»

У цьому розділі наведено опис та ілюстрації реалізованих складових програмного комплексу: вебсайту, desktop-застосунку та telegram-боту. Кожен із компонентів був розроблений відповідно до сформульованих у другому розділі функціональних та нефункціональних вимог.

Вебсайт є першим інтерфейсом взаємодії клієнта з готелем «Lake Park». Головна мета цього модуля – забезпечити зручний доступ до інформації про готель, перегляд наявних послуг та бронювання номерів онлайн. Головна сторінка вебсайту зображена на рисунку 3.1.

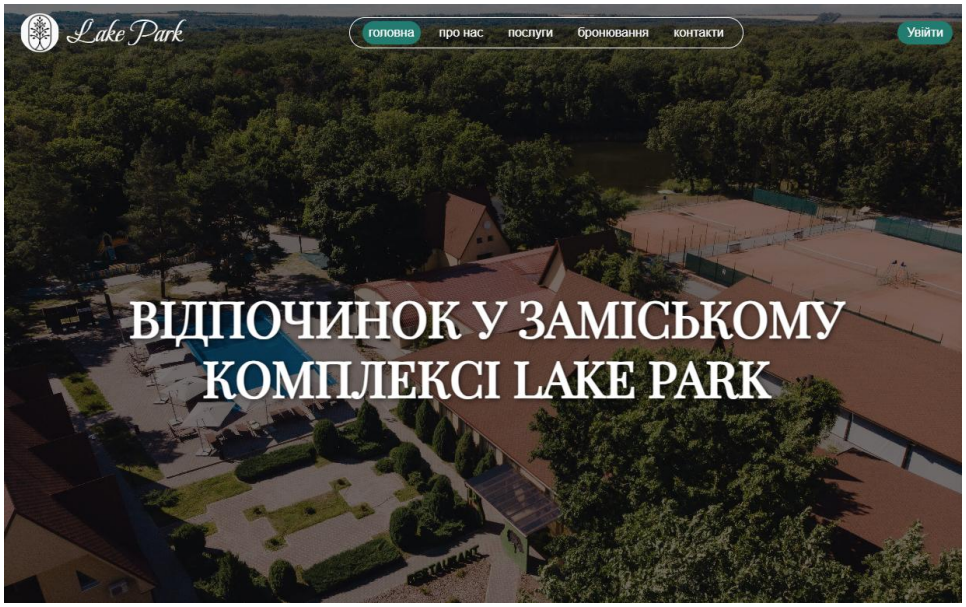

Рисунок 3.1 – Головна сторінка вебсайту

Desktop-застосунок призначений для співробітників готелю (адміністраторів, менеджерів) та власника, щоб вони могли оперативно працювати з бронюваннями, рахунками, даними про номери та персонал, створювати звіти. Зображення головного меню для менеджера представлено на рисунку 3.2.

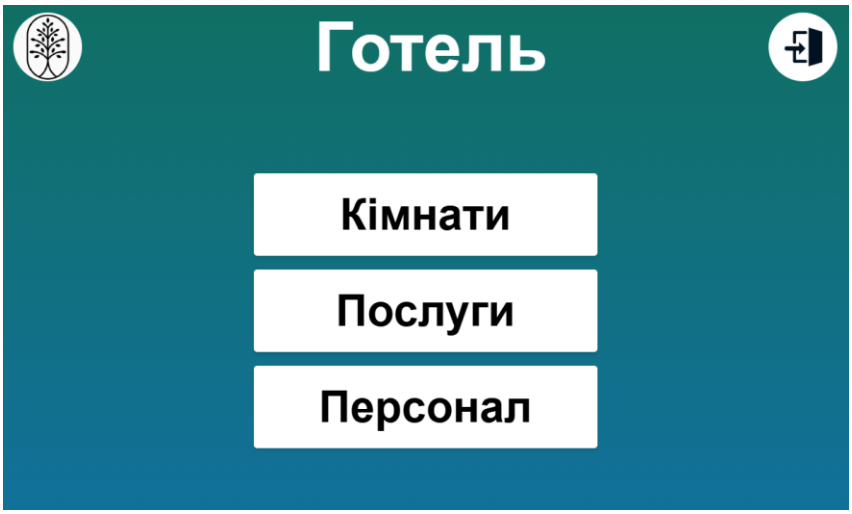

Рисунок 3.2 – Головна сторінка desktop-застосунку

Telegram-бот забезпечує швидку комунікацію між гостем і готелем, дозволяючи клієнту у чаті переглянути основну інформацію, забронювати номер або дізнатися контакти готелю. Результат розробки telegram-боту зображено на рисунку 3.3.

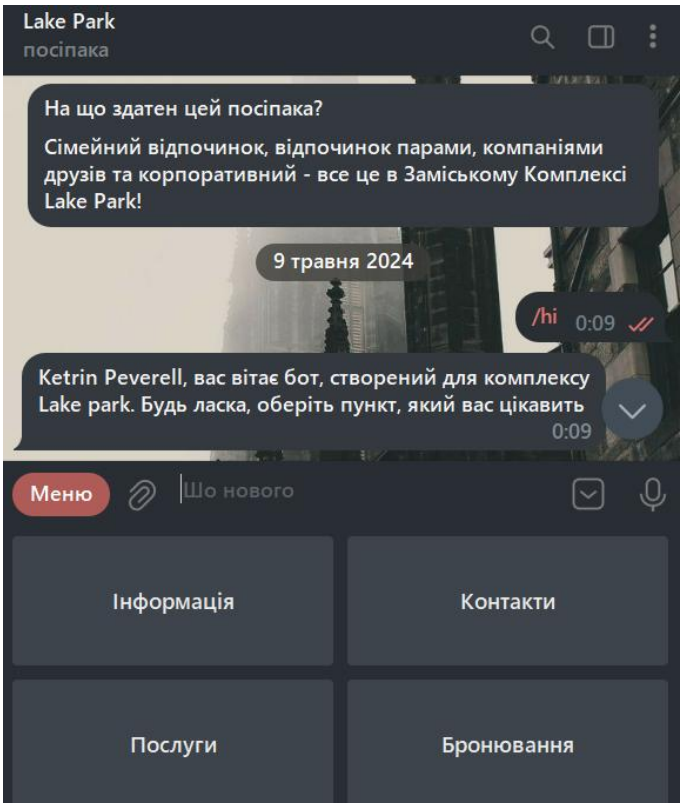

Рисунок 3.3 – Результат розробки Telegram-боту.

Під час виконання роботи було виконано:

1) Проаналізовано вимоги: узагальнено виклики, з якими стикається готельний бізнес, досліджено існуючі рішення (Booking, Expedia, Airbnb), сформульовано функціональні та нефункціональні вимоги до складових програмного комплексу.

2) Спроєктовано архітектуру: розроблено модель у вигляді трирівневої архітектури (клієнтський інтерфейс, серверна частина з REST-API, база даних PostgreSQL). Визначено механізми автентифікації, розмежування прав доступу, способи обробки даних (резервування, облік платежів, формування звітів).

3) Розроблено та реалізовано вебсайт: створено інтерфейс для клієнтів, який дозволяє переглядати інформацію про готель і послуги, реєструватися, авторизуватися, робити та редагувати бронювання. Задіяно перевірку даних на стороні клієнта і на стороні сервера.

4) Розроблено desktop-застосунок для співробітників готелю: реалізовано функціонал створення, редагування та видалення бронювань; облік гостьових рахунків; управління інформацією про номери, послуги та персонал; формування та експорт звітів (у форматі PDF/Excel).

5) Розроблено telegram-бот: забезпечено можливість швидкого отримання інформації про готель і послуги, а також оформлення бронювання через месенджер. Валідація введених даних виконується у режимі реального часу.

6) Забезпечено сумісність реалізованих компонентів: вебсайт коректно відображається у сучасних браузерах (Chrome, Opera GX), desktop-застосунок працює під Windows 10+, telegram-бот доступний на будь-якому пристрої, де встановлено Telegram.

7) Протестовано систему: проведено модульне тестування серверних ендпоінтів, інтеграційне тестування взаємодії компонентів, здійснено приймальні випробування із залученням тестових користувачів (персоналу готелю та декількох бета-клієнтів).

8) Підготовлено документацію: складено технічне завдання (додаток А), опис програми (додаток В), інструкцію користувача (додаток Г), а також програму

та методику випробувань (додаток Д).

У результаті виконаної роботи створено цілісний програмний комплекс, який дозволяє автоматизувати ключові бізнес-процеси готелю «Lake Park».

Розроблений комплекс забезпечує:

- оптимізацію внутрішніх процесів (автоматизація бронювання, облік рахунків, управління персоналом);
- покращення взаємодії з клієнтом (зручний онлайн-інтерфейс, миттєве бронювання через бот, швидкий доступ до інформації про послуги);
- забезпечення безпеки (надійний захист персональних даних, контроль доступу);
- можливість аналізу та прогнозування (формування статистичних звітів щодо завантаженості, популярності послуг, фінансових показників).

Доопрацювання та впровадження такого рішення дозволить готелю «Lake Park» підвищити рівень обслуговування, скоротити час на рутинні операції та збільшити доходи за рахунок ефективного управління ресурсами.

4 ОХОРОНА ПРАЦІ

4.1 Основні законодавчі і нормативні документи, які регламентують охорону праці в Україні

Охорона праці є однією з ключових складових організації діяльності будь-якого підприємства, зокрема й об’єкта готельного бізнесу, де великою мірою враховується не лише якість обслуговування клієнтів, але й безпека та здоров’я персоналу. Сучасний розвиток технологій спричиняє необхідність інтегрування електронних засобів управління з традиційними заходами охорони праці, що створює можливості для оптимізації управлінських процесів, швидкого реагування на надзвичайні ситуації та ефективного контролю за виконанням стандартів безпеки.

Система правового регулювання охорони праці в Україні формується на багаторівневій структурі нормативно-правових актів, де основоположним документом є Конституція України. Розділ II Конституції містить норми, що безпосередньо гарантують право кожного громадянина на працю та встановлюють обов’язок держави забезпечувати умови для безпечної та гідної роботи. Серед них важливе місце посідають положення, які містяться, зокрема, у статті 43, що гарантує право на працю, та статті 44, яка накладає на державу відповідальність за створення сприятливих умов для її реалізації. Ці статті визначають правові засади щодо підвищення стандартів охорони праці, формування умов для запобігання професійним захворюванням і нещасним випадкам на виробництві, що набуває особливої актуальності в умовах сучасної організації робочих процесів [16].

Правова база охорони праці містить не лише конституційні гарантії, але й спеціалізовані законодавчі акти, нормативні документи та інструкції, розроблені всіма компетентними державними структурами [17]. Серед них слід виділити Закон України «Про охорону праці», який встановлює основні принципи організації заходів з безпеки життєдіяльності, а також регулює відповідальність підприємств

за дотримання вимог безпеки. Особливу роль у системі регулювання охорони праці відіграють нормативні акти, розроблені Державною службою України з питань праці, що являє основний інструмент контролю за дотриманням нормативної бази у сфері безпеки праці. Ретельна адаптація нормативних документів до особливостей галузі гостинності дозволяє врахувати специфіку робочого середовища готелів, де підвищена увага приділяється як фізичним, так і психоемоційним аспектам безпеки співробітників.

Крім загальнодержавних законодавчих актів, система норм охорони праці доповнюється галузевими стандартами та рекомендаціями, розробленими спеціалістами з питань техніки безпеки. Вони спрямовані на вирішення специфічних завдань у сфері готельного бізнесу, де неповторні умови роботи визначають необхідність застосування комплексних заходів попередження негативних наслідків як для здоров'я співробітників, так і для комфорту відпочивальників. Так, наприклад, рекомендації щодо створення безпечного простору в готелі, що містять практичні кроки по мінімізації ризиків, являють собою важливе доповнення до нормативно-правової бази охорони праці. Ці рекомендації охоплюють як організаційні, так і технічні заходи, до яких відносяться стандарти облаштування робочих місць, системи вентиляції, протипожежного захисту та інформування персоналу про правила поведінки в екстрених ситуаціях.

4.2 Структура управління питаннями охорони праці у готелі

Ефективне управління заходами охорони праці в готелі «Lake Park» є невід'ємною складовою загальної стратегії менеджменту готельного бізнесу. Організаційна структура з питань охорони праці має забезпечити комплексний підхід до запобігання надзвичайним ситуаціям, оптимізації робочих процесів та мінімізації ризиків як для співробітників, так і для гостей готелю. Основною передумовою такої системи є чіткий розподіл відповідальності між керівництвом закладу, спеціалізованими службами та безпосередніми виконавцями завдань, пов'язаних із забезпеченням безпеки робочого середовища.

На найвищому рівні управління охороною праці у готелі відповідальність лягає на керівництво, яке повинне визначати стратегічні завдання та політику підприємства у сфері безпеки. Рішення з питань охорони праці включають розробку комплексної системи заходів, що охоплює як організаційні, так і технічні аспекти. Керівництво готелю несе відповідальність за забезпечення необхідних ресурсів для впровадження сучасних технологій, зокрема програмного забезпечення, яке автоматизує моніторинг і аналіз ризиків, що виникають у процесі експлуатації об'єкта. Стратегія керівництва повинна передбачати регулярний аналіз ефективності впроваджених заходів, а також коригування прийнятих рішень на основі зворотного зв'язку, що надходить від безпосередніх співробітників.

Важливим елементом організаційної структури є призначення відповідальної особи або команди з питань охорони праці. Створення внутрішньої служби контролю дозволяє не лише оперативно вирішувати поточні завдання, але й проводити системні перевірки, оцінки ризиків, а також аналіз інцидентів. Фахівці охорони праці здійснюють моніторинг стану робочих місць, оцінку впливу потенційно шкідливих факторів на здоров'я співробітників і впроваджують коригувальні заходи в разі виявлення невідповідностей. Організаційна модель управління має базуватися на принципах прозорості, інтегрованості інформаційних потоків та регулярного внесення змін у відповідності до вимог нормативно-правової бази, встановленої державними органами [16,17].

4.3 Розробка заходів по зменшенню дії небезпечних і шкідливих факторів у готелі «Lake Park»

Сучасна система управління умовами праці передбачає всебічний підхід до зменшення впливу небезпечних і шкідливих факторів на здоров'я співробітників та гостей готелю. Важливим завданням є своєчасна ідентифікація потенційних ризиків, що виникають як у виробничій діяльності, так і в сфері обслуговування. Основою розробки профілактичних заходів стає комплексна оцінка можливих

загроз, що дозволяє сформувати ефективний план дій, спрямований на мінімізацію негативного впливу несприятливих умов робочого середовища.

Впровадження заходів з охорони праці в готелі «Lake Park» ґрунтується на ретельному аналізі технічного стану обладнання, організації робочих процесів та умов перебування людей у приміщеннях закладу. Директор готелю наказом призначає відповідального за безпеку праці, формує склад служби охорони праці, призначає відповідальних за стан охорони праці в кожному підрозділі і відповідальних за навчання персоналу готельного бізнесу і перевірку знань з охорони праць [16]. Такий підхід сприяє своєчасному виявленню відхилень і оперативному реагуванню на виникаючі ситуації, що може запобігти нещасним випадкам та знизити ризики виникнення аварійних ситуацій.

До важливих заходів відноситься організація системи профілактичного технічного обслуговування обладнання, що використовується в готелі, а також забезпечення ефективної системи контролю за станом приміщень і інженерних мереж. Своєчасна заміна застарілих або несправних елементів сприяє підвищенню рівня безпеки й запобігає потенційним аварійним ситуаціям, що може мати негативний вплив на здоров’я персоналу та гостей.

Роботодавець зобов’язаний створити на робочому місці в кожному структурному підрозділі умови праці відповідно до нормативно-правових актів, а також забезпечити додержання вимог законодавства щодо прав працівників у галузі охорони праці (ст. 13 Закону № 2694) [16,18].

Особлива увага приділяється організації робочого простору з урахуванням сучасних вимог ергономіки. Раціональне планування робочих зон, дотримання правил організації робочого місця та розміщення аварійних виходів сприяють зниженню ризиків при надзвичайних ситуаціях. У приміщеннях повинно бути природне (не менше одного вікна) і штучне освітлення, що забезпечує освітленість цілодобово при лампах накалювання – 100 лк (у люменах), при люмінесцентних лампах – 200 лк, у коридорах – природне або штучне освітлення [19]. За недостатнього освітлення можуть виникати зорове перевтомлення, загальна втома, погіршення зору та зниження працездатності. Відповідна організаційна структура

дозволяє оперативно реагувати на зміни умов праці, забезпечуючи своєчасне впровадження заходів, спрямованих на мінімізацію негативного впливу шкідливих факторів.

Комплексний підхід до забезпечення безпеки в готелі «Lake Park» сприяє формуванню стабільного рівня охорони праці та створенню здорового робочого середовища. Впровадження зазначених заходів дозволяє не лише запобігти можливим небезпекам, але й підвищити загальний рівень організаційної культури, що є важливим чинником у збереженні здоров’я та безпеки співробітників і гостей закладу.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи мета, яка ставилася на початку, була досягнута, а саме розроблено програмний комплекс для автоматизації управління готелем «Lake Park», який складається з трьох компонентів: desktop-застосунок, вебсайт, telegram-бот.

Для досягнення поставленої мети було виконано такі кроки:

– проведено вивчення специфіки функціонування підприємств індустрії гостинності, зокрема, особливостей внутрішньої організації роботи готельних закладів;

– здійснено аналітичний огляд наявних комерційних та відкритих програмних рішень, що використовуються в готельному бізнесі (Booking.com, Expedia, Airbnb), з метою виявлення їхніх обмежень та невідповідностей вимогам індивідуального готелю;

– обґрунтовано доцільність створення спеціалізованого програмного комплексу, орієнтованого на потреби та масштаби готелю «Lake Park»;

– сформульовано постановку задачі щодо розробки автоматизованої системи керування операційною діяльністю готелю;

– визначено функціональні, інформаційні та технічні вимоги до розроблюваного комплексу, включаючи вимоги до підтримки обробки замовлень, управління особистими даними клієнтів, створення звітної документації та дотримання вимог інформаційної безпеки;

– проведено декомпозицію проєкту на окремі функціональні підсистеми: вебінтерфейс для клієнтів, клієнтську програму для співробітників і telegram-бот для мобільної взаємодії;

– розроблено архітектуру програмного комплексу, у тому числі визначено внутрішні зв’язки між компонентами, канали взаємодії із центральною базою даних, механізми авторизації та захисту;

– реалізовано компоненти системи засобами сучасних інструментів розробки;

– проведено тестування функціональності компонентів, а також приймальні випробування із залученням представників кінцевого користувача;

– підготовлено повний комплект технічної та супровідної документації: технічне завдання, опис програми, інструкцію користувача та методику випробувань;

– надано практичні рекомендації щодо впровадження системи в реальних умовах експлуатації, з урахуванням вимог охорони праці та безпеки інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Booking.com | Офіційний сайт |. URL: https://www.booking.com (дата звернення: 20.02.25).

2. Expedia Travel: Vacation Homes, Hotels, Car Rentals, Flights & More. URL: https://www.expedia.com (дата звернення: 20.02.25).

3. Airbnb | Помешкання для відпустки, зруби, будинки на пляжі тощо. URL: https://www.airbnb.com.ua/ (дата звернення: 20.02.25).

4. Color Hunt - Color Palettes for Designers and Artists. URL: https://colorhunt.co/ (дата звернення: 05.04.25).

5. Welcome to Python.org. https://www.python.org/ (дата звернення 29.03.2025).

6. Telegram Bot API. https://core.telegram.org/bots/api (дата звернення 29.03.2025).

7. PyCharm: the Python IDE for data science and web. https://www.jetbrains.com/pycharm/ (дата звернення 29.03.2025).

8. NET | Build. Test. Deploy – Microsoft. https://dotnet.microsoft.com/en-us/ (дата звернення 29.03.2025).

9. C# Guide - .NET managed language. https://learn.microsoft.com/en-us/dotnet/csharp/ (дата звернення 29.03.2025).

10. Rider: The Cross-Platform .NET IDE from JetBrains. https://www.jetbrains.com/rider/ (дата звернення 29.03.2025).

11. Blazor | Build client web apps with C# | .NET https://dotnet.microsoft.com/en-us/apps/aspnet/web-apps/blazor (дата звернення 29.03.2025).

12. Introduction to Identity on ASP.NET Core https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-9.0 (дата звернення 29.03.2025).

13. Bootstrap · The most popular HTML, CSS, and JS library in the world. https://getbootstrap.com/ (дата звернення 29.03.2025).

14. PostgreSQL: The world's most advanced open source database. https://www.postgresql.org/ (дата звернення 29.03.2025).

15. OSPanel/OpenServerPanel: Software environment for web. https://github.com/OSPanel/OpenServerPanel (дата звернення 29.03.2025).

16. Конституція України – Розділ II. URL: https://www.president.gov.ua/ua/documents/constitution/konstituciya-ukrayini-rozdil-ii (дата звернення: 12.03.25).

17. Нормативна база. URL: https://dsp.gov.ua/category/diyalnist/normatyvna-baza/ (дата звернення: 12.03.25).

18. 7 кроків до безпечного простору в готелі. URL: https://oppb.com.ua/articles/7-krokiv-do-bezpechnogo-prostoru-v-goteli (дата звернення: 12.03.25).

19. Охорона праці і техніка безпеки в готелях. URL: https://pidru4niki.com/1435012060420/turizm/ohorona_pratsi_tehnika_bezpeki_gotelyah (дата звернення: 12.03.25).

ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ ПРОГРАМНОГО КОМПЛЕКСУ ДЛЯ АВТОМАТИЗАЦІЇ УПРАВЛІННЯ ГОТЕЛЕМ «LAKE PARK»

Вступ

Повна назва проєкту: “Розробка програмного комплексу для автоматизації робочих процесів готелю «Lake Park»”.

Сферою застосування програмного комплексу є готель «Lake Park».

1 Підстави для розробки:

Підставою для розробки даного програмного комплексу є наказ на затвердження тем кваліфікаційних робіт бакалаврів зі спеціальності 121 Інженерія програмного забезпечення в Національному університеті кораблебудування ім. адмірала Макарова.

2 Призначення розробки

2.1 Експлуатаційне призначення

Експлуатаційним призначенням програмного комплексу є спрощення бронювання клієнтами номерів через вебсайт; спрощення управління менеджером і адміністратором бронюванням і іншою інформацією через desktop застосунок; спрощення створення звітності; покращення комунікації з клієнтами через telegram-бот.

2.2 Функціональне призначення

Програмний комплекс забезпечує можливість керувати даними про бронювання, клієнтів, кімнати, особисті рахунки, персонал, послуги у готелі Lake Park.

Програмний комплекс буде корисним для різних користувачів: клієнтів, адміністратора, менеджерів, власника.

Користувачі будуть мати можливість:

– адміністратор: обробляти бронювання, гостьові рахунки та особисті дані гостей;

– клієнт: переглядати інформацію про готель, контакти, наявні послуги готелю, обробляти особисті дані та бронювання;

– менеджер: обробляти бронювання, гостьові рахунки та особисті дані гостей, інформацію про послуги, кімнати, персонал;

– власник: створювати статистику щодо бронювань, про популярні додаткові послуги, звіт про всю суму гостьових рахунків за визначеним періодом.

3 Вимоги до програмного забезпечення

У рамках роботи будуть розроблені:

– вебсайт;

– desktop-застосунок;

– telegram-бот.

Функції вебсайту:

– відображення інформації про наявні послуги готелю;

– відображення опису готелю;

– відображення контактних даних;

– авторизація;

– обробка особистих даних (додавання, редагування, видалення);

– обробка та перегляд бронювання (редагування, створення, видалення, перегляд бронювань клієнта).

Функції desktop-застосунку:

– авторизація;

– обробка гостьових рахунків (редагування та створення);

– обробка та перегляд бронювання (редагування, створення, видалення, перегляд усіх бронювань);

– обробка інформації про послуги (створення, редагування, видалення);

– обробка інформації про кімнати (створення, редагування, видалення);

– обробка інформації про персонал (створення, редагування, видалення);

– генерація звітів та статистик.

Функції чат-боту:

– перегляд інформації про готель;

– перегляд інформації про контакти;

– перегляд інформації про послуги;

– створення бронювання;

– додавання особистих даних.

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

Розроблювальний програмний комплекс повинен надавати можливість виконання наступних функцій:

1) Створення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат бронювання.

2) Редагування бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат редагування.

3) Видалення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат видалення.

4) Перегляд бронювань клієнта

Вхідні дані: номер телефону.

Вихідні дані: список з бронюванням клієнта.

5) Перегляд усіх бронювань

Вхідні дані: натиснення кнопки «Бронювання».

Вихідні дані: список усіх бронювань (ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги).

6) Перегляд наявних послуг готелю

Вхідні дані: натиснення кнопки «Послуги».

Вихідні дані: найменування, ціна.

7) Перегляд опису готелю

Вхідні дані: натиснення кнопки «Про нас».

Вихідні дані: опис готелю.

8) Перегляд інформації про контакти готелю

Вхідні дані: натиснення кнопки «Наші контакти».

Вихідні дані: контактні дані.

9) Додавання особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат додавання.

10) Редагування особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат редагування.

11) Видалення особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат видалення.

12) Створення гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат створення.

13) Редагування гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат редагування.

14) Створення статистики щодо бронювання

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика щодо бронювань за визначений період (загальна кількість бронювань, загальна кількість суми гостьових рахунків, середній відсоток зайнятих кімнат за період).

15) Створення статистики про популярні додаткові послуги

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика про популярні додаткові послуги (Послуга, кількість використання).

16) Створення звіту «Сума рахунків за період»

Вхідні дані: дата початку, дата кінця.

Вихідні дані: звіт про всю суму гостьових рахунків за визначеним періодом (загальна сума рахунків, кількість гостей, середня сума рахунків).

17) Авторизація

Вхідні дані: номер телефону, пароль.

Вихідні дані: повідомлення про результат авторизації.

18) Обробка інформації про персонал

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження, посада.

Вихідні дані: повідомлення про результат обробки.

19) Обробка інформації про кімнати

Вхідні дані: номер, кількість односпальних ліжок, кількість двоспальних ліжок, максимальна кількість гостей, статус.

Вихідні дані: повідомлення про результат обробки.

20) Обробка інформації про послуги

Вхідні дані: найменування, ціна, статус.

Вихідні дані: повідомлення про результат обробки.

3.1.2 Вимоги до організації вхідних та вихідних даних

Вхідною інформацією є дані, які вводяться вручну (в залежності від типу поля введення, дані можуть бути: числові цілі, числові десяткові, текстові, формату дати), або обираються із представлених списків (для уникнення можливих помилок,

у випадках списків, представлені готові варіанти вибору), або отримуються з бази даних PostgreSQL.

Вихідною інформацією є виведення результатів операцій у екранні форми, повідомлення та сформовані звіти.

3.2 Вимоги до надійності

Програмне забезпечення має функціонувати неперервно, а також має бути забезпечений захист від збоїв. Для забезпечення надійності збереження інформації має бути регулярне резервне копіювання даних і можливість відновлення даних.

3.3 Умови експлуатації

3.3.1 Кліматичні умови експлуатації

Кліматичні умови експлуатації, при яких повинні забезпечуватися задані характеристики, повинні відповідати вимогам, що пред’являються до технічних засобів в частині умов їх експлуатації.

3.3.2 Вимоги до чисельності та кваліфікації користувачів

Користувач повинен мати базові навички роботи з веббраузером, комп’ютером та застосунком Telegram.

3.4 Вимоги до складу параметрів технічних засобів

Сервер, на якому розміщуються база даних і вебсайт, повинен мати такі мінімальні параметри:

- CPU: 2x Xeon 64bit або еквівалент;
- RAM: 4 GB;
- HDD: 150 GB;
- постійний доступ до Інтернет мережі.

Desktop-застосунок:

- CPU: двоядерний із тактовою частотою не менше 2,0 GHz;

– RAM: 2 GB;
– HDD/SSD: 100 GB;
– підключення до Інтернет мережі.

Технічні вимоги для телеграм-бота, що взаємодіє з користувачами:

– CPU: одне ядро з частотою 1,5 GHz;
– RAM: 1 GB;
– HDD: 50 GB;
– постійний доступ до Інтернет мережі.

3.5 Вимоги до інформаційної та програмної сумісності

Desktop-застосунок:

Програма повинна працювати під ОС Windows версії не нижче 10. Необхідна наявність серверу або встановленої програми Open Server Panel чи інших. Для формування звітів необхідне встановлення тестових редакторів Microsoft Office або схожих.

Вебсайт:

Необхідна наявність сучасного інтернет браузера Google Chrome, Opera GX чи інших, та стабільне підключення до інтернет з’єднання.

Телеграм-бот:

Необхідний встановлений застосунок Telegram на пристрої, який використовується, та зареєстрований аккаунт у месенджері.

3.6 Вимоги до маркування та пакування

Програмний продукт має бути упакований в електронний архів і мати найменування lake_park.rar та мати наступні файли:

– папка website;
– папка desktop_application;
– папка telegram_bot;
– Інструкція користувача.pdf;

– lake_park.sql.

4 Вимоги до програмної документації

Склад програмної документації повинен включати:

– технічне завдання;
– опис програми;
– програму та методику випробувань програмного забезпечення;
– інструкцію користувача;
– код програми.

5 Техніко-економічні показники

Не розраховуються.

6 Стадії та етапи роботи

Стадії та етапи роботи представлені у таблиці А.1.

Таблиця А.1 – Етапи роботи

№ п/п	Назва етапів роботи	Термін виконання
1.	Аналіз практичної задачі інженерії ПЗ	20.02.2025
2.	Аналіз вимог до ПЗ	01.03.2025
3.	Розробка архітектури ПЗ	19.03.2025
4.	Розробка проєкту ПЗ	16.04.2025
5.	Реалізація та тестування ПЗ	07.05.2025

7 Порядок контролю та приймання

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення на кожному етапі з урахуванням вимог, визначених у технічному завдані. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Приймання проводиться відповідно до програми і методики випробувань і документується за допомогою протоколу проведення випробувань. У разі знаходження помилок під час приймання програмного виробу складається акт про знайдені помилки, який підписується представниками замовника і розробника і затверджується керівниками організації-замовника та організації-розробника. Розробник повинен протягом не більше ніж 2 тижнів виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

ДОДАТОК Б ТЕКСТ ПРОГРАМИ

Лістинг 1 – telebot.py

```
import telebot
import configparser
import re
from loguru import logger
import psycopg2
from telebot import types
import datetime

# Підключення до бази даних PostgreSQL
db = psycopg2.connect(
    host="localhost",
    user="postgres",
    password="329600",
    dbname="lake_park"
)

class ClientInfo:
    def __init__(self):
        self.state = 0
        self.surname = ''
        self.name = ''
        self.father_name = ''
        self.phone_number = ''
        self.gender = ''
        self.email = ''
        self.date_of_birth = ''
        self.idClient = ''

client_info = ClientInfo()

class ReservationInfo:
```

```
        def __init__(self):
            self.state = 0
            self.date_of_arrival = ''
            self.date_of_check_out = ''
            self.number_of_guests = ''

    reservation_info = ReservationInfo()

    # Отримання списку послуг
    def get_services():
        cursor = db.cursor()
        cursor.execute("SELECT * FROM services")
        services = cursor.fetchall()
        cursor.close()
        return services

    # Ініціалізація телеграм-бота
    config = configparser.ConfigParser()
    config.read('config.ini', encoding='utf-8')

    bot = telebot.TeleBot(config['Telegram']['token'])
    logger.add('error.log', rotation='10 MB', level='ERROR', format='{time} -
{level} - {message}')

    def validate_date(year, month, day):
        try:
            datetime.datetime(int(year), int(month), int(day))
            return True
        except ValueError:
            return False

    def send_services(chat_id):
        services = get_services()
        if services:
            message = "Список доступних послуг:\n"
            for service in services:
```

```
                if service[3] != "False":
                    message += f"{service[0]}. {service[1]} - {service[2]} грн\n"
            if message == "Список доступних послуг:\n":
                message = "На даний момент немає доступних послуг."
            bot.send_message(chat_id, message)
        else:
            bot.send_message(chat_id, "На даний момент немає доступних послуг.")

    def menu():
        markup = types.ReplyKeyboardMarkup()
        markup.add(types.KeyboardButton('Інформація'),
types.KeyboardButton('Контакти'))
        markup.add(types.KeyboardButton('Послуги'),
types.KeyboardButton('Бронювання'))
        return markup

    @bot.message_handler(commands=['start', 'hi'])
    def start(message):
        try:
            markup = menu()
            bot.send_message(message.chat.id,    f'{message.from_user.first_name}
{message.from_user.last_name}, вас вітає бот комплексу <b>Lake park</b>. Оберіть
<b>пункт</b> меню.', parse_mode='html', reply_markup=markup)
        except Exception as e:
            logger.error(f"Start error for {message.chat.id}: {e}")

    @bot.message_handler()
    def info(message):
        try:
            markup = menu()
            markup2 = types.InlineKeyboardMarkup()
            match message.text.lower():
                case 'інформація':
                    bot.send_message(message.chat.id, 'Lake Park – заміський клуб
з готелем, рестораном, СПА та тенісними кортами.')
                case 'послуги':
```

```
                    send_services(message.chat.id)
                case 'контакти':
                    bot.send_message(message.chat.id,       'Контакти:\nVodafone:
+380504169476\nKyivstar: +380674969999\nLifecell: +380632245946')
                    markup2.add(types.InlineKeyboardButton('Instagram',
url="https://instagram.com/lakeparkdnipro"))
                    markup2.add(types.InlineKeyboardButton('Вебсайт',
url="https://lake-park.com.ua/"))
                    markup2.add(types.InlineKeyboardButton('Facebook',
url="https://www.facebook.com/lakeparkdnipro/"))
                    bot.send_message(message.chat.id,               'Соцмережі:',
reply_markup=markup2)
                case 'бронювання':
                    add_client(message)
                case _:
                    bot.send_message(message.chat.id, 'Такої команди не існує.',
reply_markup=markup)
        except Exception as e:
            logger.error(f"Message handler error: {e}")

    def add_client(message):
        try:
            if client_info.state == 0:
                bot.send_message(message.chat.id, "Введіть прізвище")
                client_info.state = 1
                bot.register_next_step_handler(message, add_client)
            elif client_info.state == 1:
                client_info.surname = message.text
                bot.send_message(message.chat.id, "Введіть ім'я")
                client_info.state = 2
                bot.register_next_step_handler(message, add_client)
            elif client_info.state == 2:
                client_info.name = message.text
                bot.send_message(message.chat.id, "Введіть по-батькові")
                client_info.state = 3
                bot.register_next_step_handler(message, add_client)
```

```
            elif client_info.state == 3:
                client_info.father_name = message.text
                bot.send_message(message.chat.id, "Введіть номер телефону (у
форматі 380...)")
                client_info.state = 4
                bot.register_next_step_handler(message, add_client)
            elif client_info.state == 4:
                if re.match(r'^\d{12}$', message.text):
                    client_info.phone_number = message.text
                    bot.send_message(message.chat.id, "Введіть гендер")
                    client_info.state = 5
                    bot.register_next_step_handler(message, add_client)
                else:
                    bot.send_message(message.chat.id, "Невірний номер телефону")
                    bot.register_next_step_handler(message, add_client)
            elif client_info.state == 5:
                client_info.gender = message.text
                bot.send_message(message.chat.id, "Введіть електронну пошту")
                client_info.state = 6
                bot.register_next_step_handler(message, add_client)
            elif client_info.state == 6:
                if re.match(r'^[^@\s]+@[^@\s]+\.[^@\s]+$', message.text):
                    client_info.email = message.text
                    bot.send_message(message.chat.id, "Введіть дату народження
(YYYY-MM-DD)")
                    client_info.state = 7
                    bot.register_next_step_handler(message, add_client)
                else:
                    bot.send_message(message.chat.id, "Невірна пошта")
                    bot.register_next_step_handler(message, add_client)
            elif client_info.state == 7:
                if re.match(r'^\d{4}-\d{2}-\d{2}$', message.text):
                    year, month, day = message.text.split('-')
                    if validate_date(year, month, day):
                        client_info.date_of_birth = message.text
                        client_info.state = 0
```

```
                        add_reservation(message)
                    else:
                        bot.send_message(message.chat.id, "Невірний формат дати")
                        bot.register_next_step_handler(message, add_client)
        except Exception as e:
            logger.error(f"Client input error: {e}")
            db.rollback()
            bot.send_message(message.chat.id,
                             'Сталася помилка при бронюванні. Спробуйте ще раз або
звернiться до адміністрації.')

    def add_reservation(message):
        try:
            if reservation_info.state == 0:
                bot.send_message(message.chat.id, "Дата заселення (YYYY-MM-DD)")
                reservation_info.state = 1
                bot.register_next_step_handler(message, add_reservation)
            elif reservation_info.state == 1:
                reservation_info.date_of_arrival = message.text
                bot.send_message(message.chat.id, "Дата виселення (YYYY-MM-DD)")
                reservation_info.state = 2
                bot.register_next_step_handler(message, add_reservation)
            elif reservation_info.state == 2:
                reservation_info.date_of_check_out = message.text
                bot.send_message(message.chat.id, "Кількість гостей")
                reservation_info.state = 3
                bot.register_next_step_handler(message, add_reservation)
            elif reservation_info.state == 3:
                reservation_info.number_of_guests = message.text
                reservation_info.state = 0
                insert_client_reservation(message)
        except Exception as e:
            logger.error(f"Reservation input error: {e}")

    def insert_client_reservation(message):
        try:
```

```
            cursor = db.cursor()
            client_query = """
                INSERT INTO client (last_name, first_name, father_name,
phonenumber, gender, email, dateofbirth)
                VALUES (%s, %s, %s, %s, %s, %s, %s)
                RETURNING id
            """
            client_values = (
                client_info.surname,
                client_info.name,
                client_info.father_name,
                client_info.phone_number,
                client_info.gender,
                client_info.email,
                client_info.date_of_birth
            )
            cursor.execute(client_query, client_values)
            client_info.idClient = cursor.fetchone()[0]

            reservation_query = """
                INSERT INTO reservation (idclient, dateofarrival, dateofcheckout,
numberofguests)
                VALUES (%s, %s, %s, %s)
            """
            reservation_values = (
                client_info.idClient,
                reservation_info.date_of_arrival,
                reservation_info.date_of_check_out,
                reservation_info.number_of_guests
            )
            cursor.execute(reservation_query, reservation_values)
            db.commit()
            cursor.close()

            bot.send_message(message.chat.id, "Бронювання успішно додано! Наш
менеджер зв'яжеться з вами.")
```

```
    except Exception as e:
        logger.error(f"DB insert error: {e}")

bot.polling(none_stop=True)
```

Лістинг 2 – Booking.razor

```
@page "/booking"
@using System.ComponentModel.DataAnnotations
@using Microsoft.AspNetCore.Components.Forms
@using Microsoft.AspNetCore.Antiforgery
@inject website.Components.LakeParkContext DbContext
@inject website.Components.LakeParkService LakeParkService

@rendermode @(new InteractiveServerRenderMode(prerender: true))

<EditForm Model="@bookingModel"
          FormName="BookingForm"
          OnValidSubmit="@HandleBooking">
    <AntiforgeryToken />
    <DataAnnotationsValidator />
    <ValidationSummary />

    <div class="mb-3">
        <label for="lastName">Прізвище</label>
        <input id="lastName"
               type="text"
               class="form-control"
               @bind="bookingModel.LastName" />
    </div>
    <div class="mb-3">
        <label for="firstName">Ім’я</label>
        <input id="firstName"
               type="text"
               class="form-control"
               @bind="bookingModel.FirstName" />
    </div>
```

```
<div class="mb-3">
    <label for="fatherName">По-батькові</label>
    <input id="fatherName"
           type="text"
           class="form-control"
           @bind="bookingModel.FatherName" />
</div>
<div class="mb-3">
    <label for="phone">Номер телефону</label>
    <input id="phone"
           type="tel"
           class="form-control"
           @bind="bookingModel.PhoneNumber" />
</div>
<div class="mb-3">
    <label for="email">Email</label>
    <input id="email"
           type="email"
           class="form-control"
           @bind="bookingModel.Email" />
</div>
<div class="row mb-3">
    <div class="col">
        <label for="arrival">Дата заселення</label>
        <input id="arrival"
               type="date"
               class="form-control"
               @bind="bookingModel.DateOfArrival" />
    </div>
    <div class="col">
        <label for="checkout">Дата виселення</label>
        <input id="checkout"
               type="date"
               class="form-control"
               @bind="bookingModel.DateOfCheckout" />
    </div>
```

```
    </div>
    <div class="mb-3">
        <label for="guests">Кількість гостей</label>
        <input id="guests"
               type="number"
               class="form-control"
               @bind="bookingModel.NumberOfGuests" />
    </div>
    <div class="mb-3">
        <label>Послуги</label>
        @if (availableServices == null)
        {
            <p>Завантаження послуг...</p>
        }
        else
        {
            @foreach (var svc in availableServices)
            {
                <div class="form-check">
                    <input type="checkbox"
                           class="form-check-input"
                           @bind="svc.IsSelected" />
                    <label class="form-check-label">
                        @svc.Name (@(svc.Price ?? 0)) грн
                    </label>
                </div>
            }
        }
    </div>

    <button type="submit" class="btn btn-success">Забронювати</button>
</EditForm>

@if (!string.IsNullOrEmpty(resultMessage))
{
    <div class="alert alert-info mt-3">@resultMessage</div>
```

```
}

@code {
    private BookingModel bookingModel = new();
    private List<ServiceViewModel> availableServices;
    private string resultMessage;

    protected override async Task OnInitializedAsync()
    {
        var services = await LakeParkService.GetServicesAsync();
        availableServices = services.Select(s => new ServiceViewModel {
            Id         = s.Id,
            Name       = s.Name,
            Price      = s.Price,
            IsSelected = false
        }).ToList();
    }

    private async Task HandleBooking()
    {
        var client = new website.Components.Client
        {
            LastName    = bookingModel.LastName,
            FirstName   = bookingModel.FirstName,
            FatherName  = bookingModel.FatherName,
            Phonenumber = bookingModel.PhoneNumber,
            Email       = bookingModel.Email,
            DateOfBirth = new DateTime(1900, 1, 1)
        };
        DbContext.Clients.Add(client);
        await DbContext.SaveChangesAsync();

        var reservation = new website.Components.Reservation
        {
            IdClient       = client.Id,
            DateOfArrival  = bookingModel.DateOfArrival,
```

```
                DateOfCheckout = bookingModel.DateOfCheckout,
                NumberOfGuests = bookingModel.NumberOfGuests,
                Status         = "Pending"
            };
            DbContext.Reservations.Add(reservation);
            await DbContext.SaveChangesAsync();

            foreach (var svc in availableServices.Where(x => x.IsSelected))
            {
                DbContext.ReservationServices.Add(new
website.Components.ReservationService
                {
                    IdReservation = reservation.Id,
                    IdServices    = svc.Id
                });
            }
            await DbContext.SaveChangesAsync();

            resultMessage = "Бронювання успішно створено!";
        }

        public class BookingModel
        {
            [Required] public string LastName       { get; set; } = "";
            [Required] public string FirstName      { get; set; } = "";
            [Required] public string FatherName     { get; set; } = "";
            [Required] public string PhoneNumber    { get; set; } = "";
            [Required, EmailAddress]
            public string Email                      { get; set; } = "";
            [Required]  public  DateTime  DateOfArrival      {  get;  set;  }  =
DateTime.Today;
            [Required]  public  DateTime  DateOfCheckout    {  get;  set;  }  =
DateTime.Today.AddDays(1);
            [Required, Range(1,100)]
            public int NumberOfGuests               { get; set; } = 1;
        }
```

```
    public class ServiceViewModel
    {
        public int    Id         { get; set; }
        public string Name       { get; set; }
        public int?   Price      { get; set; }
        public bool   IsSelected { get; set; }
    }
}
```

Лістинг 3 – LakeParkContext.cs

```
using Microsoft.EntityFrameworkCore;
using website.Components;

namespace website.Components
{
    public class LakeParkContext : DbContext
    {
        public LakeParkContext(DbContextOptions<LakeParkContext> options)
            : base(options)
        {
        }
        public DbSet<Client> Clients { get; set; }
        public DbSet<Room> Rooms { get; set; }
        public DbSet<Reservation> Reservations { get; set; }
        public DbSet<GuestAccount> GuestAccounts { get; set; }
        public DbSet<Service> Services { get; set; }
        public DbSet<ReservationService> ReservationServices { get; set; }

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

            // ----------------------------
            // 1) Клієнт → таблиця "client"
            // ----------------------------
```

```
modelBuilder.Entity<Client>(entity =>
{
    entity.ToTable("client");

    // Колонки
    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.Phonenumber)
          .HasColumnName("phonenumber")
          .HasMaxLength(200)
          .IsRequired();
    entity.Property(e => e.LastName)
          .HasColumnName("last_name")
          .HasMaxLength(200)
          .IsRequired();
    entity.Property(e => e.FirstName)
          .HasColumnName("first_name")
          .HasMaxLength(200)
          .IsRequired();
    entity.Property(e => e.FatherName)
          .HasColumnName("father_name")
          .HasMaxLength(200)
          .IsRequired();
    entity.Property(e => e.Gender)
          .HasColumnName("gender")
          .HasMaxLength(150)
          .IsRequired();
    entity.Property(e => e.Email)
          .HasColumnName("email")
          .HasMaxLength(300)
          .IsRequired();
    entity.Property(e => e.DateOfBirth)
          .HasColumnName("dateofbirth")
          .HasColumnType("date")
          .IsRequired();
    entity.Property(e => e.PasswordHash)
```

```
            .HasColumnName("passwordhash")
            .HasMaxLength(255);

    // Індекси
    entity.HasKey(e => e.Id);
    entity.HasIndex(e => e.Phonenumber).IsUnique();
    entity.HasIndex(e => e.Email).IsUnique();
});

// ----------------------------
// 2) Room → таблиця "room"
// ----------------------------
modelBuilder.Entity<Room>(entity =>
{
    entity.ToTable("room");

    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.Number)
          .HasColumnName("number")
          .IsRequired();
    entity.Property(e => e.NumberOfSingleBeds)
          .HasColumnName("numberofsinglebeds")
          .IsRequired();
    entity.Property(e => e.NumberOfDoubleBeds)
          .HasColumnName("numberofdoublebeds")
          .IsRequired();
    entity.Property(e => e.MaxNumberOfGuests)
          .HasColumnName("maxnumberofguests")
          .IsRequired();
    entity.Property(e => e.Status)
          .HasColumnName("status")
          .HasMaxLength(200)
          .IsRequired();
});
```

```
// ---------------------------------
// 3) Reservation → таблиця "reservation"
// ---------------------------------
modelBuilder.Entity<Reservation>(entity =>
{
    entity.ToTable("reservation");

    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.IdClient)
          .HasColumnName("idclient")
          .IsRequired();
    entity.Property(e => e.IdRoom)
          .HasColumnName("idroom");
    entity.Property(e => e.DateOfArrival)
          .HasColumnName("dateofarrival")
          .HasColumnType("date")
          .IsRequired();
    entity.Property(e => e.DateOfCheckout)
          .HasColumnName("dateofcheckout")
          .HasColumnType("date")
          .IsRequired();
    entity.Property(e => e.NumberOfGuests)
          .HasColumnName("numberofguests")
          .IsRequired();
    entity.Property(e => e.Status)
          .HasColumnName("status")
          .HasMaxLength(150);

    entity.HasOne(r => r.Client)
          .WithMany(c => c.Reservations)
          .HasForeignKey(r => r.IdClient)
          .OnDelete(DeleteBehavior.Cascade);

    entity.HasOne(r => r.Room)
          .WithMany(rm => rm.Reservations)
```

```
            .HasForeignKey(r => r.IdRoom)
            .OnDelete(DeleteBehavior.SetNull);
});

// ------------------------------------------------
// 4) GuestAccount → таблиця "guestaccount"
// ------------------------------------------------
modelBuilder.Entity<GuestAccount>(entity =>
{
    entity.ToTable("guestaccount");

    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.IdRoom)
          .HasColumnName("idroom")
          .IsRequired();
    entity.Property(e => e.IdReservation)
          .HasColumnName("idreservation")
          .IsRequired();
    entity.Property(e => e.Amount)
          .HasColumnName("amount")
          .IsRequired();

    entity.HasOne(ga => ga.Room)
          .WithMany(r => r.GuestAccounts)
          .HasForeignKey(ga => ga.IdRoom)
          .OnDelete(DeleteBehavior.Cascade);

    entity.HasOne(ga => ga.Reservation)
          .WithMany(r => r.GuestAccounts)
          .HasForeignKey(ga => ga.IdReservation)
          .OnDelete(DeleteBehavior.Cascade);
});

// ----------------------------
// 5) Service → таблиця "services"
```

```
// ----------------------------
modelBuilder.Entity<Service>(entity =>
{
    entity.ToTable("services");

    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.Name)
          .HasColumnName("name")
          .HasMaxLength(350)
          .IsRequired();
    entity.Property(e => e.Price)
          .HasColumnName("price");
    entity.Property(e => e.Status)
          .HasColumnName("status")
          .HasMaxLength(200)
          .IsRequired();
});

// ------------------------------------------------
// 6) ReservationService → таблиця "reservationservice"
// ------------------------------------------------
modelBuilder.Entity<ReservationService>(entity =>
{
    entity.ToTable("reservationservice");

    entity.Property(e => e.Id)
          .HasColumnName("id");
    entity.Property(e => e.IdReservation)
          .HasColumnName("idreservation")
          .IsRequired();
    entity.Property(e => e.IdServices)
          .HasColumnName("idservices")
          .IsRequired();

    entity.HasOne(rs => rs.Reservation)
```

```
                        .WithMany(r => r.ReservationServices)
                        .HasForeignKey(rs => rs.IdReservation)
                        .OnDelete(DeleteBehavior.Cascade);

                    entity.HasOne(rs => rs.Service)
                        .WithMany(s => s.ReservationServices)
                        .HasForeignKey(rs => rs.IdServices)
                        .OnDelete(DeleteBehavior.Cascade);
                });
            }
        }
    }
```

Лістинг 4 – App.razor

```
@using Microsoft.AspNetCore.Components.Authorization
@using website.Components.Layout
<head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <base href="/"/>
    <link   rel="stylesheet"   href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css" />
    <link                                                    rel="stylesheet"
href="@Assets["lib/bootstrap/dist/css/bootstrap.min.css"]"/>
    <link rel="stylesheet" href="@Assets["app.css"]"/>
    <link rel="stylesheet" href="@Assets["website.styles.css"]"/>
    <ImportMap/>
    <link rel="icon" type="image/png" href="favicon.png"/>
    <HeadOutlet/>
</head>

<body>
<CascadingAuthenticationState>
    <Router AppAssembly="@typeof(Program).Assembly">
        <Found Context="routeData">
```

```
                <AuthorizeRouteView                          RouteData="@routeData"
DefaultLayout="typeof(MainLayout)" />
            </Found>
            <NotFound>
                <LayoutView Layout="typeof(MainLayout)">
                    <p>Сторінку не знайдено.</p>
                </LayoutView>
            </NotFound>
        </Router>
    </CascadingAuthenticationState>

    <script src="_framework/blazor.web.js"></script>
    </body>
    </html>
```

Лістинг 5 – IAuthService.cs

```
using System.Security.Claims;
using System.Text;
using System.Security.Cryptography;
using Microsoft.AspNetCore.Components.Authorization;
using Microsoft.EntityFrameworkCore;
using website.Components.Models;

namespace website.Components.Services
{
    public interface IAuthService
    {
        Task<bool> LoginAsync(string phone, string password);
        Task<bool> RegisterAsync(RegisterModel model);
        Task LogoutAsync();
        Task<Client?> GetCurrentUserAsync();
    }

    public class AuthService : IAuthService
    {
        private readonly LakeParkContext _db;
```

```
        private readonly CustomAuthStateProvider _authStateProvider;

        public  AuthService(LakeParkContext  db,  AuthenticationStateProvider
authProvider)
        {
            _db = db;
            _authStateProvider = (CustomAuthStateProvider)authProvider;
        }

        public async Task<bool> LoginAsync(string phone, string password)
        {
            var hash = ComputeHash(password);

            var user = await _db.Clients
                .SingleOrDefaultAsync(c =>
                    c.Phonenumber == phone
                    && c.PasswordHash == hash);

            if (user == null)
                return false;
            await _authStateProvider.MarkUserAsAuthenticated(user);
            return true;
        }

        public async Task<bool> RegisterAsync(RegisterModel model)
        {
            var exists = await _db.Clients
                .AnyAsync(c => c.Phonenumber == model.PhoneNumber);
            if (exists)
                return false;

            var user = new Client
            {
                Phonenumber  = model.PhoneNumber,
                FirstName    = model.FirstName,
                LastName     = model.LastName,
```

```
                    FatherName   = model.FatherName,
                    Email        = model.Email,
                    DateOfBirth  = model.DateOfBirth,
                    Gender       = model.Gender,
                    PasswordHash = ComputeHash(model.Password)
                };

                _db.Clients.Add(user);
                await _db.SaveChangesAsync();
                await _authStateProvider.MarkUserAsAuthenticated(user);
                return true;
            }

            public async Task LogoutAsync()
            {
                await _authStateProvider.MarkUserAsLoggedOut();
            }

            public async Task<Client?> GetCurrentUserAsync()
            {
                var                authState                =                await
_authStateProvider.GetAuthenticationStateAsync();
                var name      = authState.User.Identity?.Name;
                if (string.IsNullOrEmpty(name))
                    return null;
                return await _db.Clients
                    .SingleOrDefaultAsync(c =>
                        c.Email == name || c.Phonenumber == name);
            }
            private static string ComputeHash(string input)
            {
                using var sha = SHA256.Create();
                var bytes = sha.ComputeHash(Encoding.UTF8.GetBytes(input));
                return Convert.ToBase64String(bytes);
            }
        }}
```

ДОДАТОК В ОПИС ПРОГРАМИ

1 Загальні відомості

Найменування: Програмний комплекс для автоматизації роботи готелю «Lake Park».

Призначення: інтегрована система, що складається з трьох взаємопов’язаних компонентів: вебсайту, desktop-застосунку та telegram-бота – для повноцінної автоматизації основних бізнес-процесів готелю «Lake Park».

Компоненти комплексу:

– вебсайт: забезпечує клієнтам можливість перегляду інформації про готель, наявні послуги, контактів та самостійну онлайн-реєстрацію або бронювання номерів;

– desktop-застосунок: призначений для адміністратора, менеджерів та власника готелю – використовується для обробки та перегляду бронювань, керування гостьовими рахунками, уявлення аналітики, обробки інформації про персонал, послуги і кімнати;

– telegram-бот: спрощує оперативний обмін даними з гостями (швидке створення бронювання, перегляд інформації про готель та послуги, додавання особистих даних).

Програмний комплекс автоматизує керування готелем «Lake Park», а саме – реєстрація і зміна бронювань, облік гостьових рахунків, ведення бази клієнтів, облік даних про готель, аналітику і звітність.

2 Функціональне призначення

Програмний комплекс забезпечує автоматизацію основних процесів роботи готелю через три взаємопов’язані інтерфейси. Основні функції:

1) Бронювання номерів:

– створення, редагування, скасування бронювань;

– перегляд власних бронювань клієнтом та всіх бронювань адміністратором.

2) Управління гостьовими рахунками:

– формування і зміна рахунків залежно від бронювання;

– перегляд інформації про рахунок.

3) Обробка особистих даних клієнтів:

– реєстрація, редагування та видалення інформації про клієнта;

– аутентифікація клієнтів і співробітників.

4) Управління інформацією про номери, послуги та персонал:

– Додавання, оновлення, видалення записів про кімнати, додаткові послуги та працівників;

– модульне зберігання і відображення поточного стану номеру, переліку послуг і списку персоналу.

5) Формування аналітичних звітів.

6) Інформаційна підтримка клієнтів: відображення статичної інформації (опис готелю, перелік послуг, контакти) через вебінтерфейс і telegram-бот;

3 Опис логічної структури

Програмний комплекс складається з трьох взаємопов’язаних компонентів, кожен із яких реалізовано з використанням сучасних технологій та взаємодіє через єдину базу даних. Основна ідея архітектури – модульність: клієнтські інтерфейси (вебсайт і Telegram-бот) та desktop-застосунок обмінюються даними через єдиний RESTful API, а весь масив даних зберігається в СУБД PostgreSQL. Для локального тестування вебсайту за необхідності використовується Open Server Panel (для налаштування серверного оточення)

3.1 Клієнтська частина (Frontend)

3.1.1 Вебсайт

Виконаний на фреймворку Blazor (.NET 9.0), що дозволяє використовувати C# як на клієнті, так і на сервері. Для оформлення інтерфейсу застосовано Bootstrap,

що гарантує адаптивність верстки та багатий набір готових компонентів. Розробка ведеться в IDE Rider від JetBrains, яке інтегрується з .NET і надає потужні інструменти для налагодження й тестування.

3.1.2 Telegram-бот

Реалізація логіки боту здійснюється мовою Python 3.8+ (IDE PyCharm). Для зв'язку з платформою Telegram використовується офіційний Telegram Bot API, який дозволяє відправляти й отримувати повідомлення, формувати клавіатури та обробляти команди користувача. На сервері бот працює як довготривалий процес у режимі 24/7: він слухає оновлення, робить запити до REST API і віддає результати у вигляді повідомлень або інтерактивних кнопок.

3.2 Серверна частина (Backend)

3.2.1 RESTful API (ASP .NET Core Web API)

Використовується ASP .NET Core 9.0: C# як основна мова, .NET Identity Framework для аутентифікації та авторизації користувачів (рівні доступу «клієнт», «адміністратор», «менеджер», «власник»).Контролери REST API обробляють усі клієнтські запити:

- керування бронюваннями (створення, редагування, видалення, пошук);
- керування особистими даними (реєстрація, оновлення профілю, видалення);
- операції з гостьовими рахунками;
- формування аналітичних звітів (SQL-запити до бази + обчислення статистичних показників);
- отримання статичної інформації (опис готелю, перелік послуг, контакти).

Серверна логіка розгортається як окремий вебсервіс і взаємодіє з базою через Entity Framework Core (ORM для PostgreSQL). Для кешування часто вживаних запитів (наприклад, перелік доступних кімнат або послуг) передбачено використання Redis.

3.2.2 Desktop-застосунок

Реалізовано за допомогою .NET MAUI (мультиплатформовий фреймворк для C#), що дозволяє мати єдиний код для Windows, macOS і мобільних ОС (за потреби масштабування). Основна мова – C#, розробка в IDE Rider. Desktop-застосунок безпосередньо звертається до REST API для всіх операцій:

- авторизація адміністраторів, менеджерів та власника;
- обробка бронювань (створення/редагування/видалення);
- керування гостьовими рахунками (створення та змінення суми);
- управління інформацією про послуги, номери та персонал;
- запит аналітичних звітів і відображення їх у вигляді таблиць або графіків (генерація PDF і/або Excel-звітів).

Використання .NET MAUI гарантує, що інтерфейс матиме нативний вигляд на різних ОС, а код легко підтримувати й розширювати.

3.2.3 СУБД

У якості основної СУБД обрана PostgreSQL (версія 14+), яка відома своєю надійністю, високою продуктивністю та багатими можливостями SQL. Для зв’язку з .NET API використано Entity Framework Core, що спрощує створення/редагування таблиць та реалізацію складних запитів через LINQ.

3.3 Аутентифікація та авторизація

Використовується ASP .NET Identity Framework: паролі зберігаються у базі у вигляді хешів (PBKDF2).

Ролі користувачів:

- client – може створювати/редагувати/видаляти власні бронювання й особисті дані;
- manager – обробляє бронювання та гостьові рахунки, керує даними про послуги, номери та персонал;

– admin – також має доступ до всіх функцій менеджера, плюс може редагувати інформацію про інших співробітників;

– owner – має доступ до формування аналітичних звітів (загальні суми, статистика послуг тощо).

Кожен запит, що вимагає прав, перевіряється через політики Authorization Policy в ASP .NET Core.

4 Необхідні технічні засоби

Сервер, на якому розміщуються база даних і вебсайт, повинен мати такі мінімальні параметри:

– CPU: 2x Xeon 64bit або еквівалент;

– RAM: 4 GB;

– HDD: 150 GB;

– постійний доступ до Інтернет мережі.

– необхідна наявність сучасного інтернет браузера Google Chrome, Opera GX чи інших.

Desktop-застосунок:

– CPU: двоядерний із тактовою частотою не менше 2,0 GHz;

– RAM: 2 GB;

– HDD/SSD: 100 GB;

– підключення до Інтернет мережі.

– програма повинна працювати під ОС Windows версії не нижче 10.

– необхідна наявність серверу або встановленої програми Open Server Panel чи інших.

– для формування звітів необхідне встановлення тестових редакторів Microsoft Office або схожих.

Telegram-бот:

– CPU: одне ядро з частотою 1,5 GHz;

– RAM: 1 GB;

– HDD: 50 GB;

– постійний доступ до Інтернет мережі;

– необхідний встановлений застосунок Telegram на пристрої, який використовується, та зареєстрований аккаунт у месенджері.

5 Виклик і завантаження

Механізми розгортання та середовище виконання:

1) Локальне розгортання

Для локального розгортання вебсайту використовується Open Server Panel (версія 5.x). Desktop-застосунок запускається локально на машині розробника або тестера: потрібен лише .NET Runtime 9.0. Бот піднімається як окремий процес на локальному комп’ютері.

2) Розгортання на сервері

У випадку розгортання на сервері вебAPI розгортається як сервіс systemd на Linux (наприклад, Ubuntu 22.04), під керуванням Kestrel + Nginx (як зворотний проксі). PostgreSQL встановлено на окремому сервері (версія 14+), з налаштованими бекапами. Telegram-бот працює в Docker-контейнері. Desktop-застосунок інсталюється на робочих станціях працівників готелю: мінімальні вимоги – Windows 10, .NET Runtime 9.0 і доступ в інтернет до API.

6 Вхідні дані та вихідні дані

Дані вводяться користувачем із застосуванням стандартних периферійних пристроїв: клавіатури та миші, підключених до персонального комп’ютера або ноутбука з монітором, а також з використанням сенсорного екрана або клавіатури мобільного пристрою. Зокрема, у веб-інтерфейсі оператор або клієнт заповнює поля форми у веббраузері, натисканням клавіш вводить текстові дані, а вибір із випадаючих списків виконується за допомогою маніпулятора чи сенсорного інтерфейсу. У випадку роботи через telegram-бота введення здійснюється шляхом надсилання текстових повідомлень із необхідними параметрами за допомогою клавіатури мобільного або ПК пристроїв.

У випадку desktop-застосунку, користувач вводить інформацію у відповідні поля у вікнах програми, вводячи текст із клавіатури та здійснюючи вибір зі списків за допомогою миші чи сенсорного пристрою. Після завершення заповнення полів оператор натискає кнопку підтвердження, використовуючи вказівний пристрій, і система приймає дані на сервер для подальшої обробки.

Виведення даних реалізується на моніторі в таких форматах: табличні представлення, списки чи картки з інформаційними полями у веб-інтерфейсі та desktop-застосунку. Відповіді telegram-бота надходять у вигляді текстових повідомлень із розбиттям на рядки та можуть супроводжуватися інтерактивними кнопками, які відображаються на екрані мобільного пристрою чи комп’ютера. Підсумкові результати обчислень, наприклад формування звітів за визначений період, можуть бути представлені у табличному форматі й у разі необхідності надаватися користувачу у вигляді файлів формату PDF або Excel, що завантажуються на локальний диск через відповідну дію інтерфейсу.

ДОДАТОК Г ІНСТРУКЦІЯ КОРИСТУВАЧА

1 Призначення програми

1.1 Функціональне призначення програми

Програмний комплекс призначений для автоматизації ключових бізнес-процесів готелю:

– бронювання номерів;

– управління гостьовими рахунками;

– обробка персональних даних гостей і персоналу;

– управління інформацією про послуги та кімнати, готель;

– генерація аналітичних звітів та статистики.

Цей комплекс поєднує три взаємопов’язані компоненти:

1) Вебсайт. Призначений для гостей готелю. Зручний та гарний інтерфейс клієнтоорієнтований з зрозумілим дизайном. Вебсайт дозволяє виконувати наступні дії: переглянути інформацію про готель, увійти в або зареєструвати профіль гостя. Користувач вебсайту має можливість створювати або редагувати та видаляти наявні бронювання, також редагувати інформацію про себе.

2) Desktop-застосунок. Призначений для менеджера адміністратора та власника готелю. Застосунок пропонує зручний та простий інтерфейс і зрозумілий інструментарій для спрощення процесів готелю, таких як бронювання та обробки, перегляду гостьових рахунків кімнат персоналу послуг гостей. Застосунок дозволяє також створювати звіти власнику, що допоможе ефективно керувати готелем.

3) Telegram-бот. Призначений для використання гостями готелю. Зручний інтерфейс легкий в розумінні, полегшує навігаційну систему. Бот дозволяє переглядати інформацію про готель, послуги, контакти та створювати бронювання.

1.2 Експлуатаційне призначення програми

Програмний комплекс експлуатується у різних сценаріях.

Для гостей готелю (через вебінтерфейс або telegram-бот):

– самостійно переглядати інформацію про готель (опис, послуги, контакти).

– реєструватися та входити в особистий кабінет, де можна створювати, редагувати або скасовувати власне бронювання.

– швидко взаємодіяти за допомогою telegram-бота для запиту інформації та оформлення бронювання у зручному чат-форматі.

Для адміністратора, менеджерів (через desktop-застосунок):

– обробляти бронювання (створювати, редагувати та видаляти);

– формувати та коригувати гостьові рахунки;

– редагувати інформацію про готель (персонал, номери, послуги);

– обробляти персональні дані клієнтів (додавати нові профілі гостей, оновлювати існуючі, видаляти).

Для власника (через desktop-застосунок):

– генерувати звіти з можливістю завантажити їх у зручному форматі.

1.3 Склад функцій

Нижче наведено перелік функцій програмного комплексу, розбитий за компонентами.

Вебсайт (для гостей):

1) Створення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат бронювання.

2) Редагування бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат редагування.

3) Видалення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат видалення.

4) Перегляд бронювань клієнта

Вхідні дані: номер телефону.

Вихідні дані: список з бронюванням клієнта.

6) Перегляд наявних послуг готелю

Вхідні дані: натиснення кнопки «Послуги».

Вихідні дані: найменування, ціна.

7) Перегляд опису готелю

Вхідні дані: натиснення кнопки «Про нас».

Вихідні дані: опис готелю.

8) Перегляд інформації про контакти готелю

Вхідні дані: натиснення кнопки «Наші контакти».

Вихідні дані: контактні дані.

9) Додавання особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат додавання.

10) Редагування особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат редагування.

11) Видалення особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат видалення.

12) Авторизація

Вхідні дані: номер телефону, пароль.

Вихідні дані: повідомлення про результат авторизації.

Telegram-бот (для гостей):

1) Створення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат бронювання.

2) Перегляд опису готелю

Вхідні дані: натиснення кнопки «Про нас».

Вихідні дані: опис готелю.

3) Перегляд інформації про контакти готелю

Вхідні дані: натиснення кнопки «Наші контакти».

Вихідні дані: контактні дані.

4) Додавання особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат додавання.

Desktop-додаток (для адміністратора, менеджера, власника):

3.1.1 Вимоги до складу виконуваних функцій

Розроблювальний програмний комплекс повинен надавати можливість виконання наступних функцій:

1) Створення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат бронювання.

2) Редагування бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат редагування.

3) Видалення бронювання

Вхідні дані: ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги.

Вихідні дані: повідомлення про результат видалення.

4) Перегляд усіх бронювань

Вхідні дані: натиснення кнопки «Бронювання».

Вихідні дані: список усіх бронювань (ПІБ, номер телефону, email, дата заселення, дата виселення, кількість гостей, послуги).

5) Додавання особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат додавання.

6) Редагування особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат редагування.

7) Видалення особистих даних

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження.

Вихідні дані: повідомлення про результат видалення.

8) Створення гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат створення.

9) Редагування гостьового рахунку

Вхідні дані: бронювання, кімната, сума.

Вихідні дані: повідомлення про результат редагування.

10) Створення статистики щодо бронювання

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика щодо бронювань за визначений період (загальна кількість бронювань, загальна кількість суми гостьових рахунків, середній відсоток зайнятих кімнат за період).

11) Створення статистики про популярні додаткові послуги

Вхідні дані: дата початку, дата кінця.

Вихідні дані: статистика про популярні додаткові послуги (Послуга, кількість використання).

12) Створення звіту «Сума рахунків за період»

Вхідні дані: дата початку, дата кінця.

Вихідні дані: звіт про всю суму гостьових рахунків за визначеним періодом (загальна сума рахунків, кількість гостей, середня сума рахунків).

13) Авторизація

Вхідні дані: номер телефону, пароль.

Вихідні дані: повідомлення про результат авторизації.

14) Обробка інформації про персонал

Вхідні дані: номер телефону, ПІБ, пароль, стать, email, дата народження, посада.

Вихідні дані: повідомлення про результат обробки.

15) Обробка інформації про кімнати

Вхідні дані: номер, кількість односпальних ліжок, кількість двоспальних ліжок, максимальна кількість гостей, статус.

Вихідні дані: повідомлення про результат обробки.

16) Обробка інформації про послуги

Вхідні дані: найменування, ціна, статус.

Вихідні дані: повідомлення про результат обробки.

2 Умови виконання програми

2.1 Мінімальний склад апаратних засобів

Сервер (для розміщення бази даних, вебсайту та telegram-боту):

– CPU: двоядерний процесор типу Xeon 64 bit або еквівалент (частота ≥ 2,0 GHz);

– RAM: ≥ 4 GB оперативної пам’яті;

– HDD: ≥ 150 GB вільного місця на диску;

– постійний доступ до Інтернет (канал зі швидкістю не менше 20 Мбіт/с для забезпечення коректної роботи вебінтерфейсу та чатботу).

Робочі пристрої (для адміністратора, менеджерів, власника):

– CPU: двоядерний процесор з тактовою частотою ≥ 2,0 GHz (Intel Core i3 / AMD Ryzen 3 чи еквівалент);

– RAM: ≥ 2 GB оперативної пам’яті;

– HDD/SSD: ≥ 100 GB вільного дискового простору;

– стабільне підключення до Інтернет (доступ до серверу бази даних).

2.2 Мінімальний склад програмних засобів

На сервері (розгортання вебсайту та бази даних):

– операційна система: Linux (Ubuntu 18.04+/CentOS 7+);

– СУБД: PostgreSQL 12+;

– вебсервер: Apache 2.4+ або Nginx 1.18+ для обробки HTTP/HTTPS запитів;

– наявність SSL-сертифікату (Let's Encrypt або інший) для захищеного з'єднання по HTTPS (обов’язкова вимога для обробки персональних даних і оплат).

На робочих станціях співробітників (desktop-застосунок):

– ОС: Windows 10 (x64) або вище;

– .NET Framework: 4.7.2+ (або інша версія, необхідна для запуску desktop-застосунку);

– тестові редактори Microsoft Office (Word, Excel) або суміжне програмне забезпечення (WPS Office, LibreOffice) для перегляду та друку звітів у форматах .docx / .xlsx;

Клієнтські пристрої гостей (вебсайт, telegram-бот):

– будь-який сучасний браузер (Google Chrome $v \geq 80$, Mozilla Firefox $v \geq 75$, Opera GX, Microsoft Edge $v \geq 80$);

– будь-який застосунок або вебдодаток Telegram:

  – браузер – доступ до вебінтерфейсу (будь який сучасний браузер);

  – desktop-застосунок Telegram (Windows, macOS, Linux, Chrome OS);

  – через мобільний застосунок Telegram (iOS 10.0+, Android 5.0+).

2.3 Вимоги до персоналу (користувачів)

Гість (клієнт готелю):

– вміння користуватися інтернет-браузером (введення URL, робота з формами, заповнення полів, натискання кнопок).

– наявність базових навичок користування Telegram (навчитися вводити команди, натискати на кнопки меню).

– розуміння термінів «бронювання», «сервіс», «дата заселення/виселення» тощо.

Стаціонарний оператор (адміністратор/менеджер, що працює з Desktop-додатком):

– досвід роботи з ОС Windows 10 (вміння запускати програми, розуміння розташування ярликів, робота з файлами та папками).

– навички базового користування мережними програмами (налаштування підключення до серверу, робота з брандмауером у разі потреби).

– розуміння структури готельного бізнесу та ролей (адміністратор, менеджер, власник).

– уміння вводити та редагувати табличні дані (перевірка коректності введених значень – дати, суми).

Власник (керівник, що передусім потребує звітності):

– базові навички роботи з офісними програмами для перегляду звітів (Word/Excel) – для друку/експорту статистичних форм.

– розуміння основних фінансових показників.

– уміння інтерпретувати статистику (популярність послуг, тенденції бронювань).

3 Виконання програми

3.1 Основні терміни

Головне меню – це основний навігаційний інтерфейс, який надає користувачеві доступ до основних розділів. Містить посилання на ключові сторінки, які використовуються найчастіше.

Посилання на сторінки меню – це текстові (графічні) елементи в головному меню, які призначені для переходу на конкретні розділи програми. Вони допомагають користувачеві швидко знаходити та переходити до необхідних розділів.

Кнопка виходу з облікового запису – це елемент інтерфейсу, який користувач може натиснути для виходу зі свого облікового запису. Після натискання на цю кнопку користувач виходить зі свого облікового запису та переходить до сторінки авторизації.

Логін – це унікальне ідентифікуюче ім'я, яке користувач використовує для входу до свого облікового запису. Він може бути відомим також як ідентифікатор користувача.

Пароль – це конфіденційний рядок символів, який користувач використовує разом з логіном для перевірки своєї особистості при вході до свого облікового запису.

Роль – при додавання персоналу, кожному надається роль, яка визначає його права та доступ до функціоналу desktop-застосунку. Роль може бути "власник", "адміністратор" чи "менеджер", і вона визначає обсяг доступних користувачеві функцій та ресурсів.

Доступ до елементів ПЗ – це можливість користувача мати доступ до різних функцій та ресурсів програмного забезпечення в залежності від його ролі. Доступ до елементів ПЗ може бути обмеженим або повним залежно від прав користувача.

Попередження про видалення – це повідомлення, яке відображається користувачеві перед видаленням об'єкту або даних програми. Воно містить інформацію про можливі наслідки видалення та запит на підтвердження видалення від користувача.

Інформаційне повідомлення – це повідомлення, яке відображається користувачеві для передачі певної інформації. Воно може використовуватися для повідомлення про помилки, неправильні введені дані або інші важливі повідомлення для користувача.

Кнопки меню telegram-боту – елементи інтерфейсу, призначеними для швидкого виконання певних команд або дій в боті.

Команди боту – слова-словосполучення, які вказують боту на конкретні дії, які потрібно виконати після їх введення користувачем.

3.2 Вебсайт:

Як зареєструватися на вебсайті?

Відкрийте веббраузер та перейдіть за адресою localhost:8080. Ви потрапите на головну сторінку вебсайту (рисунок Г.1).

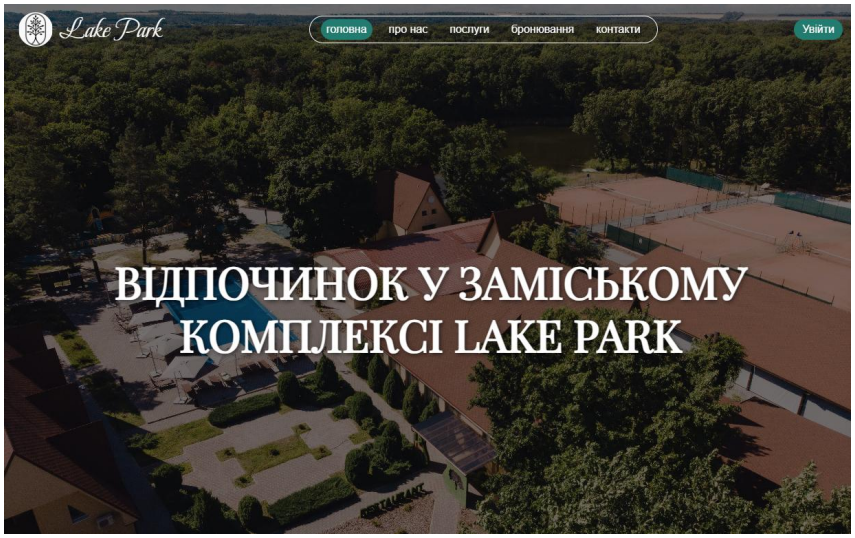

Рисунок Г.1 – Головна сторінка вебсайту

У правому верхньому кутку ви побачите кнопку «Увійти», натиснувши на неї, ви потрапите на сторінку авторизації (рисунок Г.2).

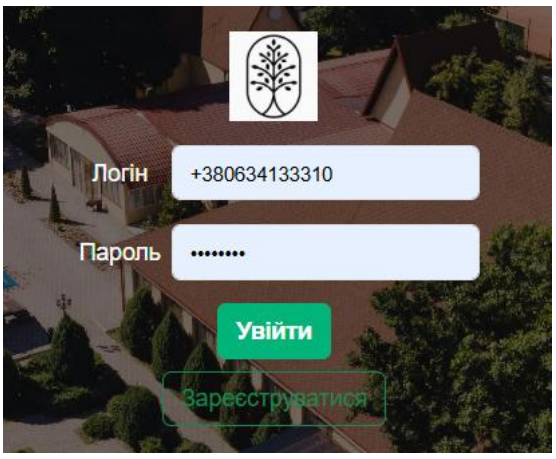

Рисунок Г.2 – Сторінка «Увійти»

Під кнопкою увійти ви побачите кнопку «Зареєструватися», натиснувши на неї, ви потрапите на сторінку реєстрації (рисунок Г.3).

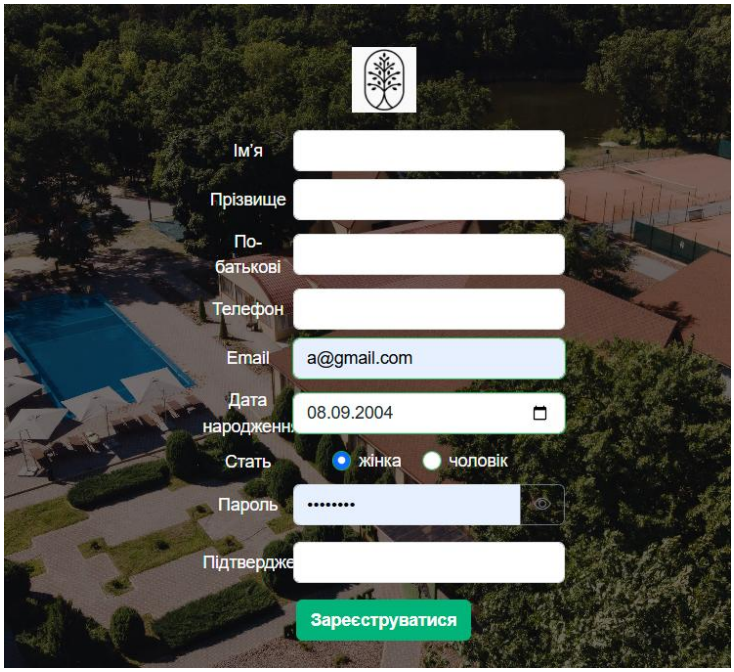

Рисунок Г.3 – Сторінка «Зареєструватися»

У відповідні поля потрібно ввести необхідну інформацію, після чого, натискаєте на кнопку «Зареєструватися». Система перевіряє коректність введення інформації та перенаправить вас на головну сторінку (рисунок Г.4).

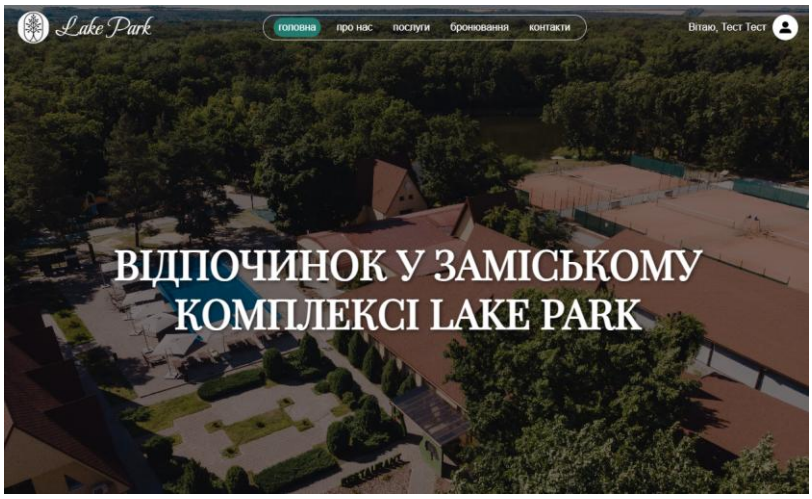

Рисунок Г.4 – Головна сторінка вебсайту для зареєстрованих користувачів

Як оформити бронювання на вебсайті?

Відкрийте веббраузер та перейдіть за адресою localhost:8080. Ви потрапите на головну сторінку вебсайту (рисунок Г.1). Натисніть на кнопку «Бронювання», вас перенаправить на сторінку «Бронювання» (рисунок Г.5).

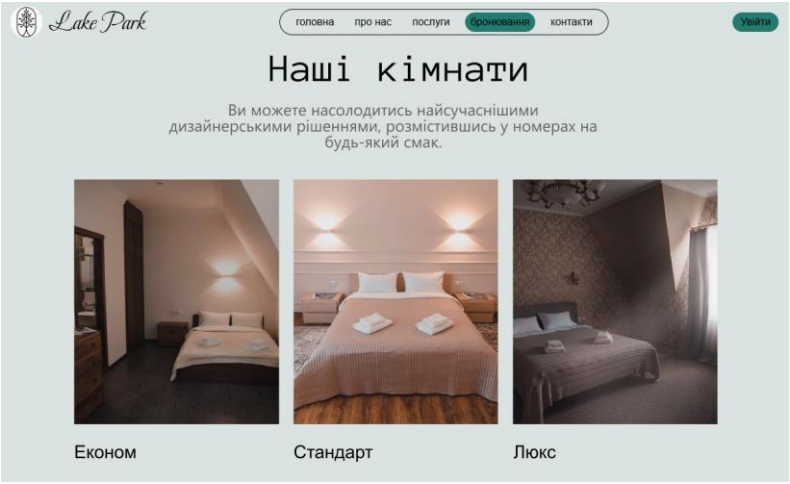

Рисунок Г.5 – Сторінка «Бронювання»

Прокрутивши сторінку нижче, ви побачите форму бронювання (рисунок Г.6).

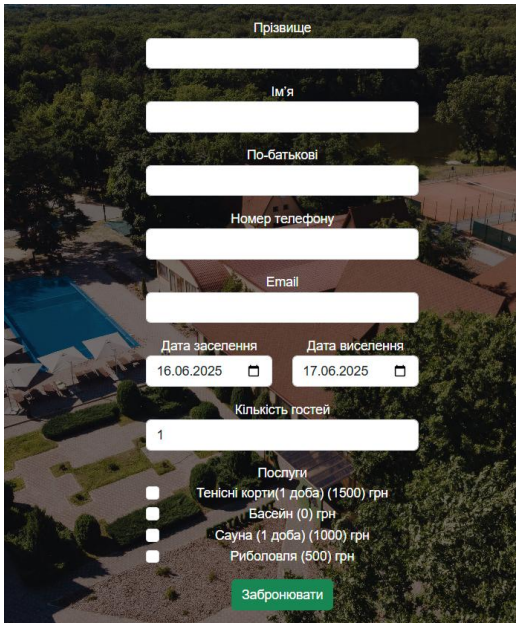

Рисунок Г.6 – Форма бронювання

Після введення необхідної інформації у відповідні поля форми, натисніть кнопку «Забронювати». Система автоматично перевірить коректність заповнення полів. Після успішної перевірки, сторінка оновиться.

Як працювати з меню на вебсайті?

На головній та інших сторінках вебсайту, зверху кожної сторінки, ви можете побачити меню(рисунок Г.7).

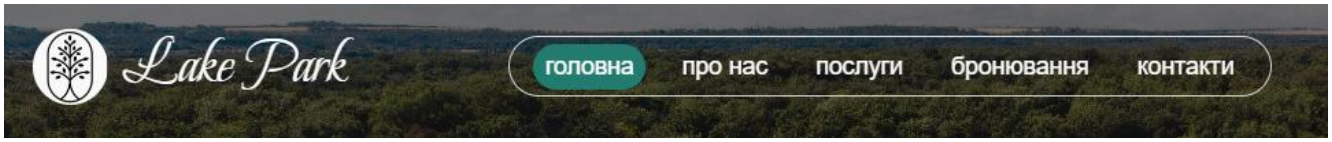

Рисунок Г.7 – Меню вебсайту

З їх допомогою ви маєте можливість переходити на різні сторінки сайту, такі як «Головна», «Про нас», «Послуги», «Бронювання», «Контакти». Натисніть на кнопку потрібної сторінки і вас перенаправить на відповідну сторінку.

Як редагувати інформацію про бронювання або особисті дані, видалити бронювання?

Відкрийте веббраузер та перейдіть за адресою localhost:8080. Ви потрапите на головну сторінку вебсайту (рисунок Г.1). У правому верхньому кутку ви побачите кнопку «Увійти», натиснувши на неї, ви потравите на сторінку авторизації (рисунок Г.2). Після заповнення полів, натисніть кнопку «Увійти», вас перенаправить на головну сторінку вебсайту (рисунок Г.3). Натисніть на кнопку з зображенням іконки користувача у правому верхньому кутку, після ви потрапите на сторінку «Профіль». Ви побачите форму з кнопками (рисунок Г.8).

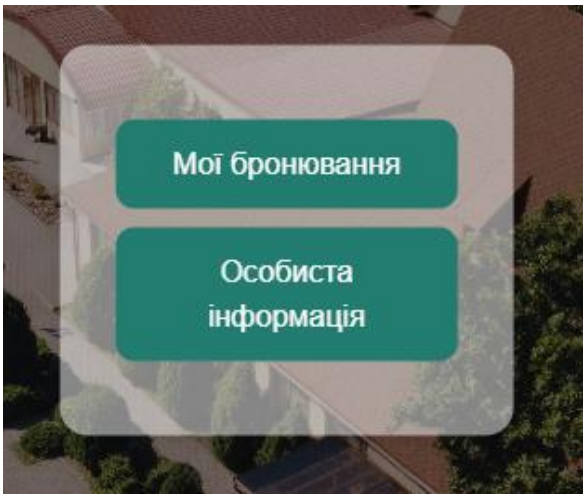

Рисунок Г.8 – Форма з кнопками на сторінці «Профіль»

Натисніть на кнопку «Мої бронювання». Ви потрапите на сторінку зі своїми бронюванням. Натисніть на кнопку «Редагувати» поруч з відповідним бронюванням, яке хочете редагувати. Відкриється форма бронювання, де відповідні поля вже будуть заповнені інформацією (рисунок Г.9).

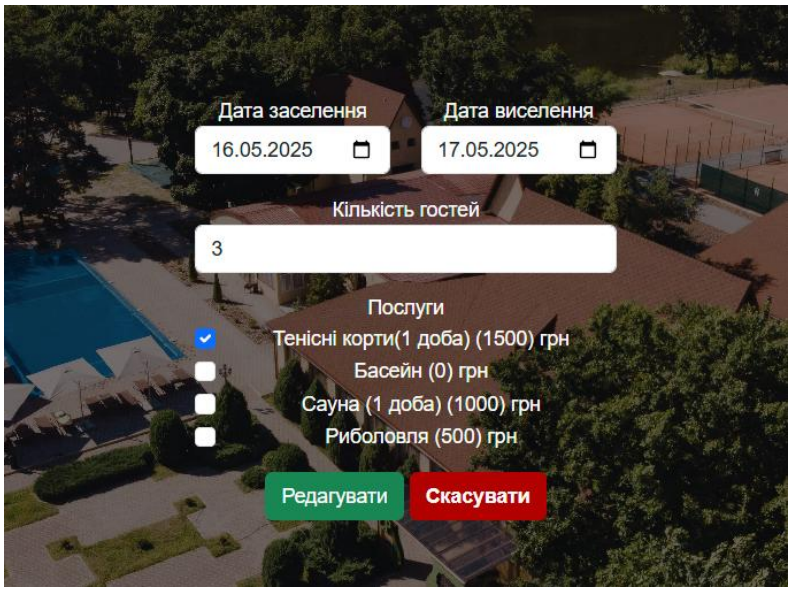

Рисунок Г.9 – Форма редагування інформації про бронювання

Після виправлення інформації у потрібних вам полях, натисніть кнопку «Редагувати». Після ви потрапите на сторінку з вашими бронюваннями.

Якщо вам потрібно скасувати бронювання – на сторінці з редагуванням бронювання, натисніть кнопку «Скасувати». (рисунок Г.10).

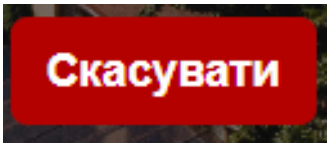

Рисунок Г.10 – Кнопка «Скасувати» для відміни бронювання

Після з’явиться підтверджувальна форма, натисніть кнопку підтвердити, щоб скасувати бронювання.

Якщо вам потрібно редагувати вашу особисту інформацію – на сторінці «Профіль», натисніть кнопку «Особиста інформація» (рисунок Г.8). Після відкриється форма редагування особистої інформації (рисунок Г.11) з уже заповненими полями. Виправте необхідну інформацію і натисніть кнопку «Редагувати».

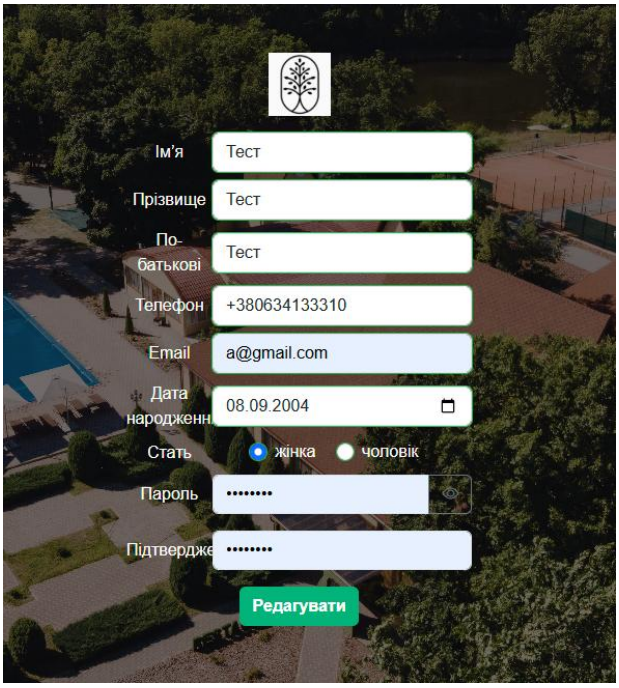

Рисунок Г.11 – Форма редагування особистої інформації

3.3 Desktop-застосунок:

Як авторизуватися в застосунку?

На робочому столі ви знайдете іконку застосунку (рисунок Г.12).

Рисунок Г.12 – Іконка застосунку «Системи управління готелем Lake Park»

Натисніть два рази і застосунок запуститься. Далі ви побачите сторінку авторизації (рисунок Г.13).

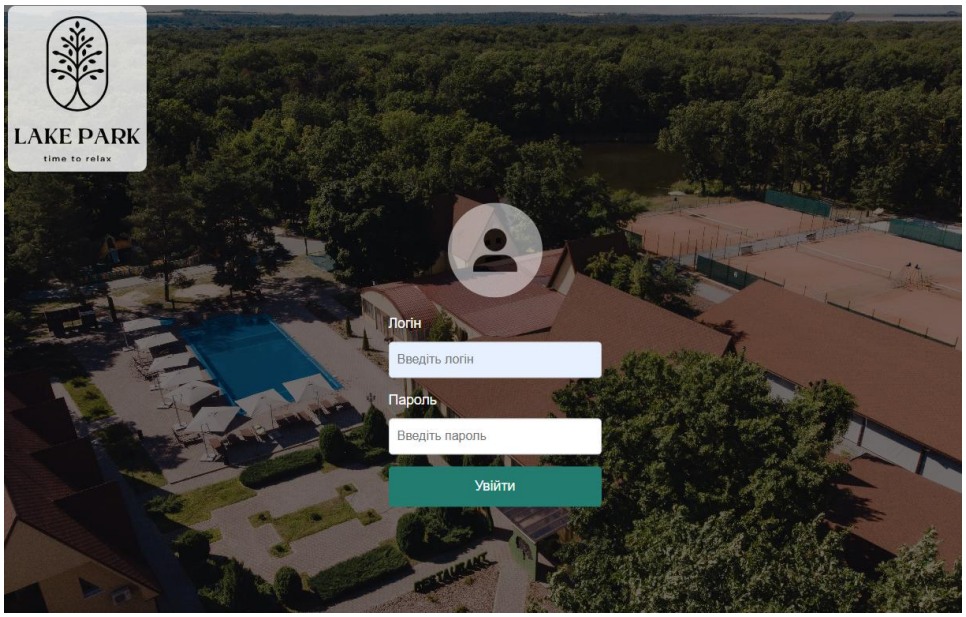

Рисунок Г.13 – Сторінка «Авторизація» у desktop-застосунку

Після введення необхідної інформації, натисніть кнопку «Увійти». Ви потрапите на головну сторінку застосунку, де, в залежності від вашої ролі, покажуться різні кнопки.

Як працювати з головним меню?

Після успішної авторизації, ви потрапите на головну сторінку застосунку, де, в залежності від вашої ролі, покажуться різні кнопки. У верхній частині екрану ви побачите іконку готелю (перенаправляє на головну сторінку застосунку), назву сторінки та вихід з акаунту (рисунок Г.14).

Рисунок Г.14 – Меню застосунку

Якщо вам потрібно повернутися до головної сторінки, натисніть кнопку з іконкою готелю. Якщо ви хочете вийти з акаунту, натисніть кнопку з іконкою виходу у верхньому правому кутку.

Як додати інформацію (на прикладі послуг)?

Знаходячись на головному меню, натисніть кнопку «Готель». Ви потрапите на сторінку «Готель» (рисунок Г.15).

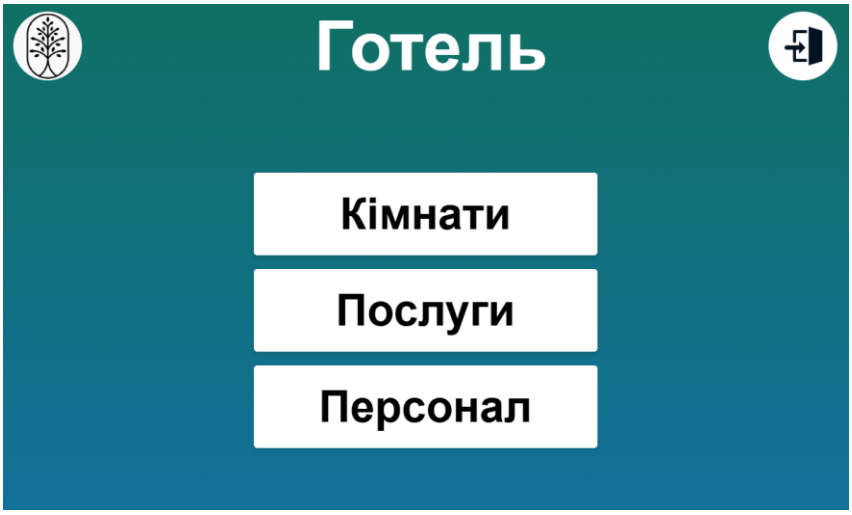

Рисунок Г.15 – Сторінка «Готель»

Натисніть на кнопку «Послуги». Після вам відкриється сторінка з таблицею у якій знаходиться інформація про усі послуги (рисунок Г.16). Зліва буде форма з додаванням інформації про послуги. У відповідних полях введіть необхідну інформацію, а після натисніть кнопку «Додати». Сторінка оновиться і нову послугу буде додано.

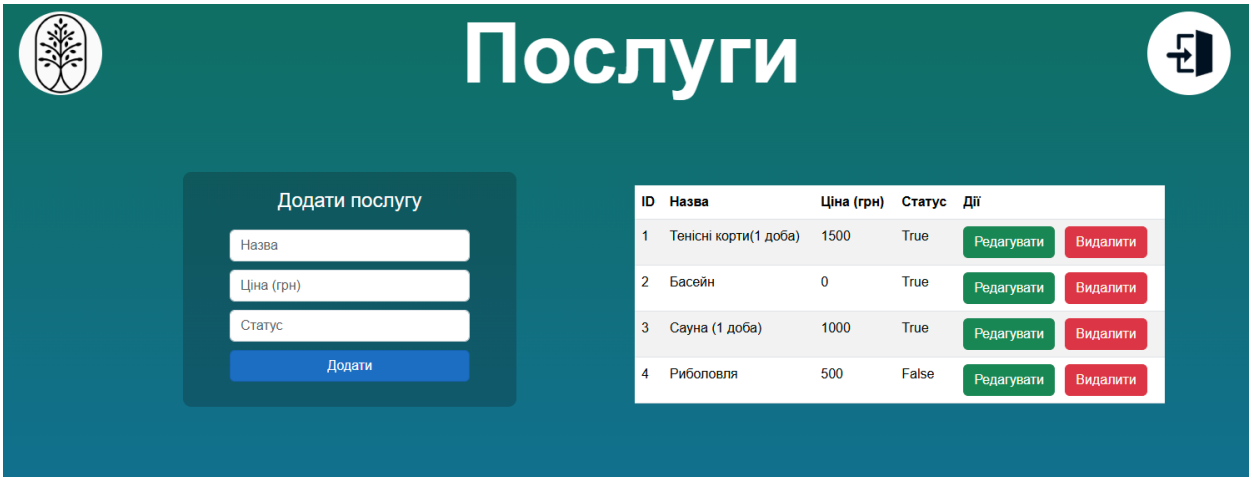

Рисунок Г.16 – Сторінка з інформацією про послуги

Як редагувати, видаляти інформацію?

Щоб редагувати або видалити інформацію зайдіть у необхідну сторінку застосунку. Наприклад, після натиснення кнопки «Послуги» на сторінці «Готель», ви потрапите на сторінку з інформацією про послуги (рисунок Г.16). Поряд з полем відповідної послуги, справа, ви побачите кнопки «Редагувати», «Видалити». Після натиснення кнопки «Редагувати», ви потрапите на сторінку з редагуванням послуги (рисунку Г.17). Виправте необхідну інформацію і натисніть кнопку «Зберегти».

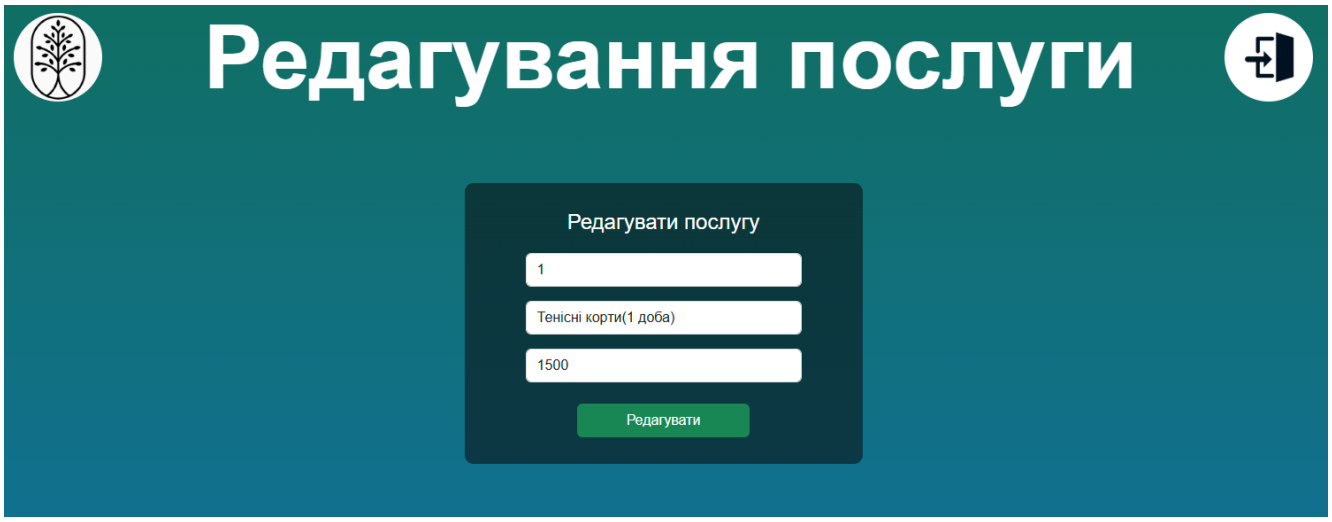

Рисунок Г.17 – Сторінка редагування інформації

Для видалення послуги, натисніть кнопку «Видалити» у відповідному полі послуги. Відкриється форма підтвердження видалення (рисунок Г.18). Натисніть кнопку підтвердити.

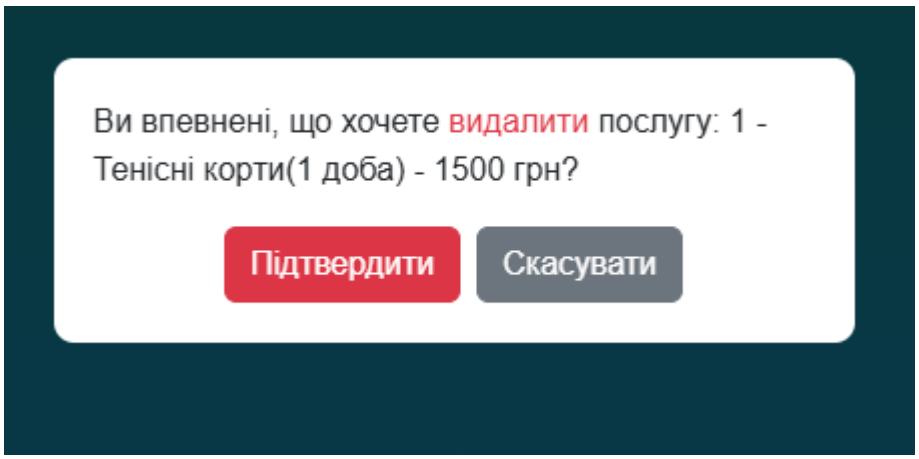

Рисунок Г.18 – Форма видалення послуги.

3.4 Telegram бот:

Як розпочати роботу з ботом?

Зайдіть у застосунок Telegram на вашому пристрої. У полі пошук (рисунок Г.19) напишіть lake_park_bot. Ви побачите бота з назвою Lake Park (рисунок Г.20). Натисніть на нього.

Рисунок Г.19 – Поле пошуку у Telegram

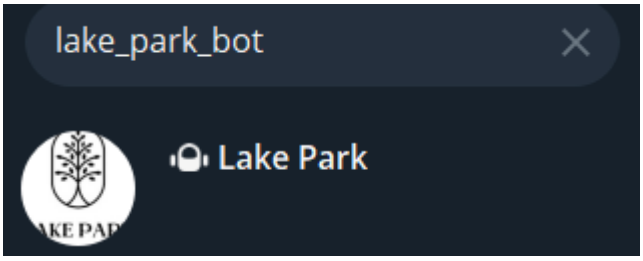

Рисунок Г.20 – Бот готельного комплексу Lake Park

Відкриється головна сторінка з ботом, натисніть кнопку «Розпочати» внизу екрану (рисунок Г.21).

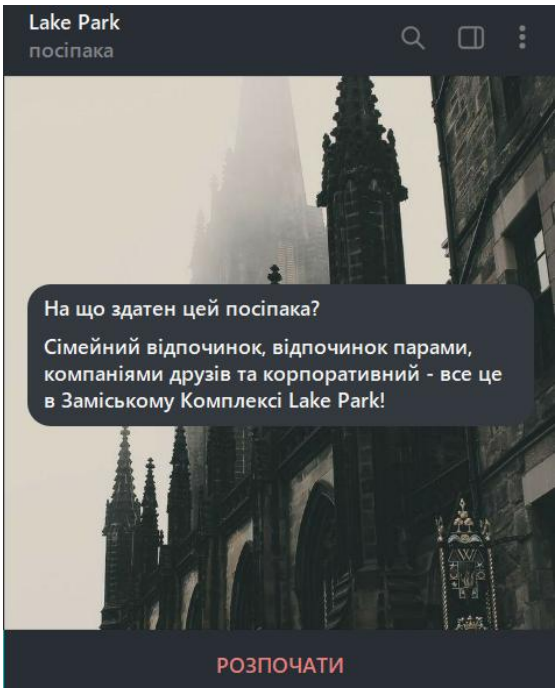

Рисунок Г.21 – Головна сторінка бота

Після натиснення кнопки, ви отримаєте привітальне повідомлення (рисунок Г.22).

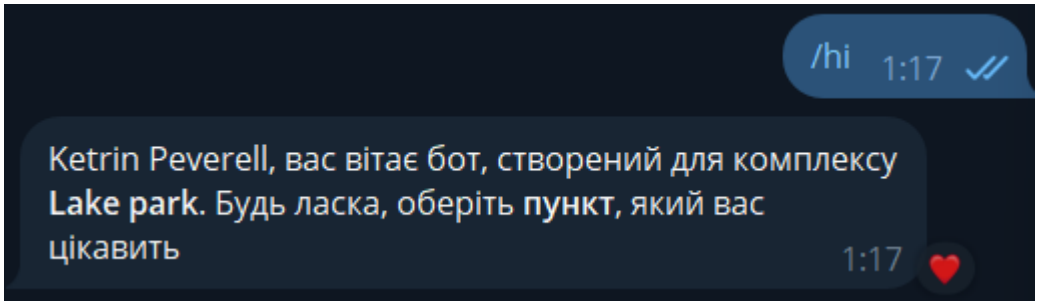

Рисунок Г.22 – Привітальне повідомлення

Як працювати з меню?

Після того як ви розпочали роботу з ботом, внизу з'явиться меню (рисунок Г.23).

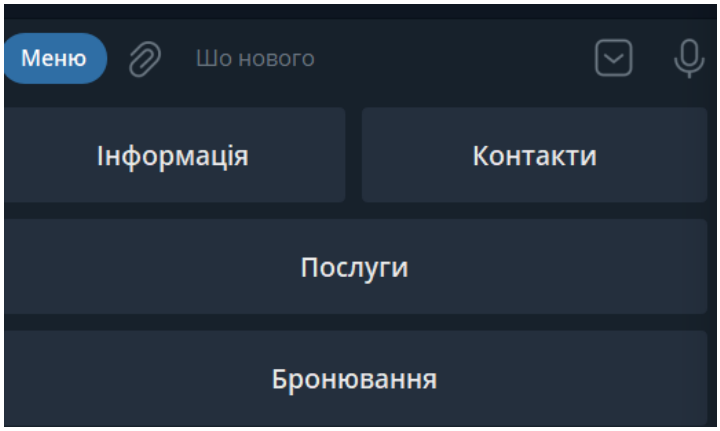

Рисунок Г.23 – Меню telegram боту

Натискаючи на відповідні кнопки меню, ви надсилаєте відповідні команди боту. Якщо вам потрібна інформація про готель, натисніть кнопку «Інформація», якщо контакти готелю, натисніть кнопку «Контакти». Також ви можете переглянути доступні послуги готелю, натиснувши кнопку «Послуги», після чого ви отримаєте список послуг з їх цінами.

Як створити бронювання?

Після відкриття меню (рисунок Г.23) натисніть кнопку «Бронювання», якщо бажаєте створити бронювання. Далі бот почергово буде запитувати у вас

необхідну, для створення бронювання, інформацію (рисунок Г.24). Якщо ви правильно введете усю необхідну інформацію, бот створить бронювання і повідомить вас про це (рисунок Г.25). У випадку некоректно введеної інформації, бот запитає у вас цю інформацію ще раз і вкаже, що інформація була введено неправильно (рисунок Г.26).

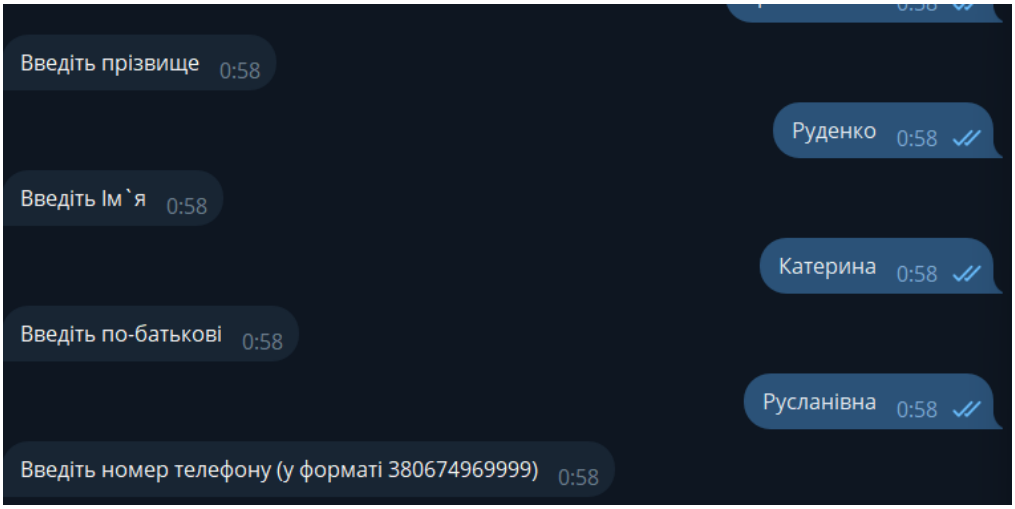

Рисунок Г.24 – Запит на необхідну інформацію для бронювання

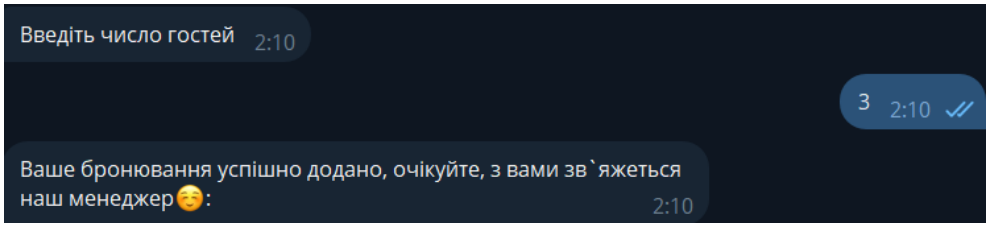

Рисунок Г.25 – Повідомлення про успішне бронювання

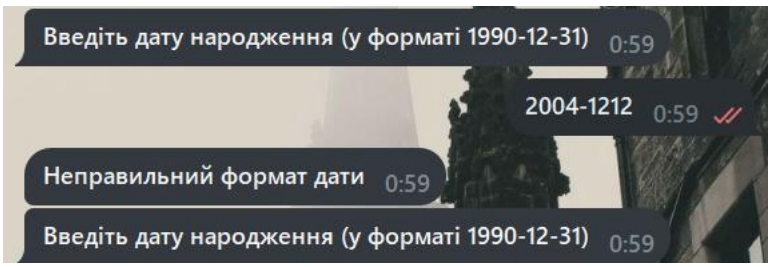

Рисунок Г.26 – Повідомлення про неправильно введену інформацію

ДОДАТОК Д ПРОГРАМА ТА МЕТОДИКА ВИПРОБОВУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1 Об’єкт випробувань

1.1 Найменування випробуваної програми

Програмний комплекс для автоматизації роботи готелю «Lake Park», що складається з вебсайту, desktop-застосунку, telegram-бота.

1.2 Область застосування випробуваної програми

Програмний комплекс призначений для автоматизації процесів бронювання номерів, обліку гостей, управління інформацією про ресурси та формування звітності, з метою підвищення якості обслуговування у готелі.

1.3 Позначення випробуваної програми

«Програмний комплекс для автоматизації роботи готелю “Lake Park”», розроблений у рамках кваліфікаційної роботи «Розробка програмного комплексу для автоматизації роботи готелю “Lake Park”».

2 Мета випробувань

Метою є перевірка відповідності реалізованого комплексу вимогам сформульованих у технічному завданні. У залежності від характеру тестування передбачаються такі різновиди випробувань:

– забезпечення коректної взаємодії між вебсайтом, desktop-застосунком та telegram-ботом;

– оцінка продуктивності системи під навантаженням (до 20 одночасних користувачів);

– перевірка захищеності передавання даних (TLS 1.2+, автентифікація).

3 Вимоги до програми

Під час проведення випробувань перевіряються функціональні характеристики програмного комплексу на відповідність вимогам технічного завдання, а саме:

– реалізація усіх Use Case, передбачених специфікацією варіантів використання;

– забезпечення інтеграції між вебінтерфейсом, API, desktop-застосунком і базою даних;

– підтримка сумісності з операційною системою Windows 10 і браузерами Google Chrome (остання версія), Opera (остання версія) та Edge (остання версія);

– здатність обробляти до 100 запитів за секунду протягом 1 години без втрати стабільності;

– підтримка безпечних механізмів автентифікації та шифрування (не нижче TLS 1.2).

4 Вимоги до програмної документації

4.1 Склад програмної документації, пропонованої на випробування

До складу документації, що подається на випробування, будуть входити наступні документи:

– технічне завдання на розробку програмного комплексу;

– специфікація варіантів використання (Use Case);

– проєкт програмного забезпечення (архітектурні діаграми, опис модулів);

– опис програмного комплексу (загальна характеристика та структура компонентів);

– програми та методика випробувань (згідно вимог ДСТУ 2853-94 і IEEE 829);

– інструкція користувача (керівництво з встановлення, налаштування та експлуатації вебсайту, desktop-застосунку та Telegram-бота).

4.2 Спеціальні вимоги

Документація та процес випробувань мають відповідати таким вимогам:

– документи мають бути оформлені згідно з нормами ДСТУ (шрифт Times New Roman 14 pt, міжрядковий інтервал 1,5, поля 20 мм);

– методика випробувань повинна відповідати структурі IEEE 829: Test Plan Outline;

– результати тестування документуються у формі IEEE 829 Test Log та дефект-репортів у системі Jira;

– кожен тест-кейс повинен містити запис зі статусом виконання, часовою міткою та скріншотами у разі невдалого проходження.

5 Засоби і порядок випробувань

5.1 Технічні засоби, що використовуються під час випробувань

Під час випробувань застосовуються наступні технічні засоби:

– серверна платформа (віртуальна машина Windows Server) із встановленим програмним забезпеченням для запуску вебсайту та бази даних;

– клієнтські робочі станції (Windows 10) із процесором двоядерним не менше 2,0 GHz, оперативною пам’яттю не менше 2 GB, жорстким диском (HDD/SSD) об’ємом не менше 100 GB;

– локальна мережа з підключенням до Інтернету для доступу до вебсайту та тестування механізмів аутентифікації;

– мережеве обладнання (комутатор, маршрутизатор) для емуляції різних умов мережевої затримки.

5.2 Програмні засоби, що використовуються під час випробувань

Під час випробувань використовуються такі програмні засоби:

– операційна система Windows 10 (клієнт) та Windows Server (сервер);

– СУБД PostgreSQL (остання стабільна версія) для бази даних;

– вебсервер Apache/Nginx для розгортання сайту (за необхідності);

– Selenium WebDriver для автоматизованого тестування вебінтерфейсу;

– OWASP ZAP для сканування вразливостей та тестування безпеки;

– JMeter для проведення навантажувального тестування (до 100 запитів/с);

– Jira для реєстрації дефектів і ведення логів тестування;

– інструмент Postman для ручного тестування API.

5.3 Порядок проведення випробувань

Випробування проводяться у наступній послідовності.

1) Підготовка тестового середовища:

– розгортання серверної віртуальної машини з ОС Windows Server та встановлення бази даних PostgreSQL;

– налаштування середовища для вебсервера (Apache/Nginx) та розгортання вебзастосунку;

– підключення клієнтських робочих станцій (Windows 10) до тестової мережі.

2) Виконання функціональних тестів:

– ручне виконання тестових сценаріїв згідно зі специфікацією Use Case;

– автоматизоване тестування вебінтерфейсу за допомогою Selenium (формування тестових скриптів, запуск перевірок).

3) Виконання інтеграційних тестів:

– перевірка коректної взаємодії веб → API → БД за допомогою Postman та інструментів моніторингу запитів;

– аналіз логів бази даних та вебсерверу для виявлення можливих помилок під час взаємодії.

4) Навантажувальне тестування:

– запуск JMeter з конфігурацією до 100 запитів/с протягом 1 години;

– моніторинг завантаження CPU, пам’яті та I/O під час тесту;

– оцінка продуктивності під навантаженням 20 одночасних користувачів.

5) Тестування безпеки:

– сканування вебінтерфейсу та API за допомогою OWASP ZAP;

– тестування механізмів автентифікації та авторизації (перевірка TLS 1.2+, обробка сесій).

6) Документування результатів:

– заповнення форм IEEE 829 Test Log для кожного кейсу (статус, час виконання, коментарі);

– реєстрація дефектів у Jira із зазначенням кроків відтворення, скриншотів та пріоритету;

– підготовка підсумкового звіту про результат випробувань із рекомендаціями щодо усунення виявлених дефектів.

6 Методи випробувань

Випробування буде проводитись за стратегією «чорної скриньки».

6.1 Функція: Створення бронювання (вебсайт)

Тестовий випадок 1.1 Створення бронювання з усіма коректними даними

Ідентифікатор: TC_CreateBooking_01.

Предмети тестування: Функція створення бронювання.

Специфікація вхідних даних:

– ПІБ: «Іваненко Іван Іванович».
– Номер телефону: «380501234567».
– Email: «ivan@example.com».
– Дата заселення: «2025-04-10».
– Дата виселення: «2025-04-15».
– Кількість гостей: 2.
– Обрані послуги: «Сніданок».

Специфікація результатів:

– Бронювання створюється у системі.
– Відображається повідомлення про успішне бронювання.

– Запис з’являється в базі даних.

Тестове оточення: Вебсайт системи управління готелем; актуальна база даних.

Спеціальні процедурні вимоги: Перед тестуванням (якщо потрібно) – авторизація користувача; перевірка бази даних на наявність нового запису.

Залежність з іншими тестами: Може залежати від попередньої перевірки коректності роботи форми введення.

Тестовий випадок 1.2 Перевірка валідації поля Email.

Ідентифікатор: TC_CreateBooking_02.

Предмети тестування: Перевірка коректності валідації даних у формі бронювання.

Специфікація вхідних даних:

– Email: «ivanexample.com» (без символа «@»).

– Інші поля заповнені коректно.

Специфікація результатів:

– Відображається повідомлення про помилку валідації формату email.

– Бронювання не створюється.

Тестове оточення: Вебсайт, модуль валідації даних форми.

Спеціальні процедурні вимоги: –.

Залежність з іншими тестами: –.

Метод: еквівалентне розбиття (недопустима форма email).

Тестовий випадок 1.3 Відсутність обов’язкового поля (ПІБ).

Ідентифікатор: TC_CreateBooking_03.

Предмети тестування: Перевірка наявності обов’язкових полів.

Специфікація вхідних даних:

– ПІБ: не заповнено.

– Інші поля – коректні дані.

Специфікація результатів:

– Система відображає повідомлення про необхідність заповнення поля ПІБ.

– Бронювання не створюється.

Тестове оточення: вебсайт.

Спеціальні процедурні вимоги: –.

Залежність з іншими тестами: –.

Метод: перевірка негативного сценарію за допомогою error guessing.

Тестовий випадок 1.4 Перевірка конфлікту при одночасному бронюванні.

Ідентифікатор: TC_CreateBooking_04.

Предмети тестування: Функція створення бронювання з перевіркою конфліктів.

Специфікація вхідних даних:

– Дані бронювання, які збігаються з уже існуючим бронюванням (один і той самий номер і період).

Специфікація результатів:

– Система виявляє конфлікт бронювання.

– Відображається повідомлення про неможливість бронювання через наявність конфлікту.

Тестове оточення: Система, в якій вже існує бронювання для заданого номера та періоду.

Спеціальні процедурні вимоги: Попередньо створити бронювання для перевірки конфлікту.

Залежність з іншими тестами: Тестування можливості створення бронювання без конфлікту.

Метод: використання сценарію з негативним тестуванням.

6.2 Функція: Редагування гостьового рахунку (desktop-застосунок)

Тестовий випадок 2.1 Редагування рахунку з коректними даними.

Ідентифікатор: TC_EditBill_01.

Предмети тестування: Функція редагування гостьового рахунку.

Специфікація вхідних даних:

– Ідентифікатор бронювання: 12345.

– Ідентифікатор кімнати: 101.

– Сума до сплати: 1500.00.

Специфікація результатів:

– Рахунок оновлюється у системі.

– Відображається повідомлення про успішне редагування.

– Оновлена інформація зберігається в базі даних.

Тестове оточення: Desktop-застосунок; тестова база даних.

Спеціальні процедурні вимоги: Виконання операції авторизованим користувачем.

Залежність з іншими тестами: Наявність попереднього рахунку, який підлягає редагуванню.

Тестовий випадок 2.2 Редагування рахунку з негативною сумою.

Ідентифікатор: TC_EditBill_02.

Предмети тестування: Перевірка коректності введення суми до сплати.

Специфікація вхідних даних:

– Сума до сплати: -100.00 (негативне значення).

– Інші поля – коректні.

Специфікація результатів:

– Система відображає повідомлення про некоректне значення суми.

– Рахунок не оновлюється.

Тестове оточення: Desktop-застосунок.

Спеціальні процедурні вимоги: Операцію проводить авторизований користувач.

Залежність з іншими тестами: немає.

Метод: тестування граничних значень (boundary value analysis) та негативний сценарій

Тестовий випадок 2.3 Редагування рахунку з некоректними символами у полі суми.

Ідентифікатор: TC_EditBill_03.

Предмети тестування: Перевірка формату даних для суми до сплати.

Специфікація вхідних даних:

– Сума до сплати: «ABC» (літерали замість чисел).

– Інші поля – коректні.

Специфікація результатів:

– Відображається повідомлення про некоректний формат суми.

– Рахунок не оновлюється.

Тестове оточення: Desktop-застосунок.

Спеціальні процедурні вимоги: Операцію проводить авторизований користувач.

Залежність з іншими тестами: –.

Метод: еквівалентне розбиття (недопустимий формат даних).

Тестовий випадок 2.4 Спроба редагування рахунку без авторизації.

Ідентифікатор: TC_EditBill_04.

Предмети тестування: Перевірка доступу до функції редагування.

Специфікація вхідних даних:

– Спроба виконання операції редагування рахунку без авторизації.

Специфікація результатів:

– Система блокує доступ до операції редагування.

– Відображається повідомлення про необхідність авторизації.

– Рахунок залишається незмінним.

Тестове оточення: Desktop-застосунок у режимі неавторизованого користувача.

Спеціальні процедурні вимоги: немає.

Залежність з іншими тестами: немає.

Метод: негативне тестування (error guessing).