МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра економічної кібернетики та інформатики

ДЕНИСЮК Олексій Петрович

Інформаційна система оцінки надійності бронювань готельних номерів. Information system for assessing the reliability of hotel room reservations.

спеціальність: 124 - Системний аналіз
освітньо-професійна програма - Системний аналіз

Кваліфікаційна робота

Виконав студент групи САм-21
О. П. Денисюк

Науковий керівник:
Пасічник Р.М.

Кваліфікаційну роботу
допущено до захисту:

"22" 11 2022 р.

Завідувач кафедри
Л. М. Буяк

ТЕРНОПІЛЬ - 2022

ВСТУП

Актуальність. Пандемія COVID-19 та війна в Україні відчутно вплинули на сферу готельного бізнесу. Цей вид бізнесу постраждав від пандемії чи не найбільше серед усіх сегментів бізнес-активності. Протягом першої половини року кількість туристів у Києві зменшилася на 90% порівняно з таким самим періодом 2019 року [22].

На жаль, прогнози щодо відновлення основних показників в залежності від сегмента ринку не обіцяють миттєвих відновлень. Цей процес може тривати 3-4 роки. Однак за оцінками міжнародних експертів відновлення після пандемії, а також зняття всіх обмежень буде швидшим в порівнянні із відновленням після глибокої економічної кризи 2008-2009 рр. Не дивлячись на руйнівний вплив пандемії бізнес повинен удосконалювати свої операційні можливості та наполегливо готуватися до жорсткої конкурентної боротьби за клієнтів у період відновлення активності після завершення пандемії [3].

Ще одним вагомим чинником, який вплинув на розвиток готельної справи, є війна Росії проти України. Тільки за перший місяць війни готельний бізнес втратив більше, ніж за попередні два роки коронавірусу. При цьому важко оцінити обсяг завданої шкоди, адже бойові дії тривають.

Однак всупереч всім складнощам, бізнес повинен продовжувати удосконалювати свої операційні можливості та наполегливо готуватися до жорсткої конкурентної боротьби за клієнтів у період відновлення активності після завершення пандемії та війни.

Однією із ключових проблем готельного бізнесу є значна незаповненість номерів, в тому числі відмова від попередньо заброньованих місць клієнтами, плани яких змінилися. Одним із виходів із цієї ситуації є овербукінг, тобто перебронювання готельних місць з урахуванням можливих відмов клієнтів. Понадлімітне бронювання відрізняється від традиційного управління завантаженості готелів, адже завдяки цьому вдається використовувати якомога

більшу місткість готелю у випадку, якщо спостерігається значна кількість відмов від бронювання. Однак понаднормові продажі несуть у собі ризик появи “зайвих” клієнтів, тобто тих, кому було продано номер, але немає місця у готелі.

Крім цього, значне перебронювання призводить до необхідності додаткових переміщень клієнтів при вичерпанні готельних ресурсів, що викликає додаткові витрати, а також може призвести до втрати лояльності постійних клієнтів. Тому необхідно враховувати реальні можливості напливу клієнтів, а також формувати продуману стратегію їх переміщення із мінімізацією поточних та майбутніх втрат.

Ризики овербукінгу можна оцінити за допомогою математичних моделей, яким присвячена певна література, зокрема роботи [54, 57, 59, 60, 65, 67, 70]. В них розглядаються імітаційні, оптимізаційні, детерміновані та стохастичні математичні моделі над певними масивами даних узагальнених для класів об’єктів. Водночас сучасний рівень інформаційних технологій дозволяє фіксувати значні масиви серверних даних для окремих економічних об’єктів, таких як готелі. Тому відповідні математичні моделі можуть прив’язуватися до окремих об’єктів, враховуючи їх унікальність та унікальність їхнього бізнес-оточення.

Тому унікальною є задача побудови інформаційної системи прогнозування реального попиту на готельні місця на основі статистичних масивів активності окремих агентів туристичного бізнесу. Оброблення значних масивів інформації, які при цьому виникають, доцільно здійснювати методами машинного навчання. Розв’язанню поставленої задачі присвячена дана робота.

Метою дипломної роботи є розроблення інформаційної системи оцінки надійності бронювань готельних номерів та її ідентифікація за допомогою методів машинного навчання.

Завдання дипломної роботи:

- ✓ виявити джерела перевіреної інформації про процеси овербукінгу на підприємствах готельного бізнесу;

- ✓ розробити інформаційну систему надійності бронювань готельних номерів;
- ✓ розробити методи попередньої обробки інформації із об'єктом дослідження для спрощення реалізації моделі;
- ✓ вибрати методи ідентифікації запропонованої моделі та здійснити їх програмну реалізацію;
- ✓ здійснити експерименти із відбору вагомих пояснювальних змінних на основі реальних даних та порівняти ефективності запропонованих підходів.

Об'єктом дослідження є система надійності бронювань готельних номерів.

Предметом дослідження є методи та підходи, які дають змогу реалізувати інформаційну систему надійності бронювань готельних номерів.

Результатом роботи є розроблена інформаційна система та її ідентифікація за допомогою методів машинного навчання.

Використана методика дослідження: основні наукові результати і висновки, одержані на основі методів ідентифікації, машинного навчання, статистичних методів, методів теорії систем.

Практична цінність полягає у можливості прогнозувати реальний попит на бронювання номерів і здійснювати ефективне перебронювання для поселення більшої кількості людей і отримання більшого прибутку.

Дипломна робота складається зі вступу, трьох розділів, висновків і списку літератури та додатків. Основний зміст викладений на 91 сторінці друкованого тексту, містить 11 рисунків та 5 таблиць. Список літератури містить 70 найменувань.

Ключові слова: інформаційна система, математична модель, надійність бронювань, кореляційний аналіз, факторний аналіз, лінійна регресія, метод дерева рішень.

РОЗДІЛ 1

МОДЕЛЮВАННЯ НАДІЙНОСТІ ГОТЕЛЬНИХ БРОНЮВАНЬ ТА МЕТОДИ ЇЇ ІДЕНТИФІКАЦІЇ Й РОЗВ'ЯЗУВАННЯ

1.1 Аналіз чинних моделей надійності готельних бронювань та методів машинного навчання

Математичному моделюванню особливостей організації готельного бізнесу присвячено ряд публікацій. Значна кількість із них присвячена проблемі організації овербукінгу. Однак робота [57] присвячена прогнозуванню попиту на готельні послуги з метою уникнення ситуацій незадоволеного попиту на готельні послуги п'ятизіркового класу. У прогнозуванні використано поєднання експертних оцінок згідно з методом аналізу ієрархій Сааті, а також низку методів прогнозування часових рядів: ковзаючого середнього MA, метод Холта-Вінтера, експоненціального згладжування та ARIMA. Однак у цьому дослідженні враховувались припущення про відсутність практики попереднього бронювання, договорів купівлі-продажу та сегментації ринку управління доходами. Неврахування ефекту овербукінгу значно знижує цінність дослідження, оскільки для готельного бізнесу типовою є ситуація не перевантаженості, а недозавантаженості замовленнями.

У результаті практично в той же період низка дослідників присвятили увагу дослідження проблеми організації овербукінгу. Зокрема в роботі [70] розглядаються моделі лінійного та стохастичного програмування. Робота має певну цінність завдяки формалізації задачі управління овербукінгом, однак згадані моделі включають ряд параметрів, які важко оцінити для конкретного готелю, що знижує практичне значення роботи. Також у роботі зазначається важливість грамотної політики управління овербукінгом, щоб не викликати значного невдоволення у клієнтів.

Варіанти такої виваженої політики наведені у роботі [54]. На основі детермінованої моделі оцінюються втрати від незаповненого номера, а також втрати у випадку необхідності переміщення клієнта, який забронював місце. На основі рівності граничних збитків будуються рекомендації щодо оптимального обсягу перебронювань. Тобто будуються усереднені рекомендації на основі допущень про вхідний потік клієнтів у готель. Також в роботі детально проаналізовано варіанти політики компанії щодо дій у ситуації, коли внаслідок перебронювання, окремих клієнтів доводиться переміщати.

У роботі [67] досліджуються наслідки застосування різних рівнів знижок при передоплаті за бронювання і як це впливає на відмову клієнта від вже забронюваного номера. Дослідження проводилися на базі допущень щодо практично важливих рівнів знижок та реакції на них споживачів у масштабах двох досліджуваних готелів.

У роботі [65] методами імітаційного моделювання досліджується ефективність політики компанії у випадку перевищення вільних зарезервованих номерів у випадку пропонування клієнтам вільних дорожчих номерів за ціною замовлених дешевших. Показано рівень додаткових прибутків, який може бути отриманий для окремого готелю в австрійському відпочинковому регіоні.

У роботі [60] на основі вибірки із доступних вебресурсів американських готелів, а також телефонного опитування управлінців інших готелів сформовано статистичний масив для дослідження підходів до овербукінгу в галузі. Дослідження здійснюється шляхом дисперсійного, регресійного та кореляційного аналізу, а також шляхом перевірки статистичних гіпотез методами Краскела-Валліса та Манна-Уітні. Будуються рекомендації щодо особливостей політики овербукінгу на основі узагальнення досвіду галузі готельного бізнесу США.

Останнім часом поряд із методами статистичного аналізу даних все частіше застосовуються методи машинного навчання (machine learning), тобто узагальнення статистичних методів у бік самонавчання (здатності поступово

покращувати продуктивність завдяки даним, без того, щоби бути перепрограмованими).

Узагальнення власного досвіду може, зокрема, досягатися шляхом варіативного розбиття наявних даних на навчальну та контрольну вибірки. Параметри методів підбираються на навчальних вибірках, а якість побудованих моделей перевіряється на контрольних. При достатньо складних методах при цьому можуть мінятися не лише параметри моделей, а і структура самих методів. Такі методи зазвичай називаються ансамблевими, вони поєднують різнотипові алгоритми, які навчаються одночасно і виправляють помилки один одного. Для реалізації ансамблів використовують підходи стекінгу, беггіну та бустингу.

При стекінгу вибирають алгоритм-супервайзер, який за аналізом успішності підпорядкованих йому методів формалізовано приймає рішення про вибір ведучого алгоритму, який формуватиме остаточні результати. При реалізації беггінгу результати роботи різних алгоритмів усереднюються, що забезпечує відбір найпопулярнішого результату, а також формування багатоваріантного результату, тобто формування не тільки основного, а й альтернативних підходів. При бустінгу використовується послідовне навчання алгоритмів. Кожен наступний алгоритм налаштовується на корекцію максимальних похибок попереднього методу аж до забезпечення необхідного результату.

Методи машинного навчання покликані обробляти величезні масиви інформації, які нагромаджені на вебсерверах відповідних суб’єктів інформаційного простору.

Як показав аналіз попередніх робіт, присвячених та близьких до тематики овербукінгу, загальною тенденцією досліджень є нарощення тісноти прив’язки аналізу до емпіричних даних галузі та поглиблення диференціації аналізу. Водночас відчувається відсутність робіт даної тематики, які б ґрунтувалися на методах машинного навчання. Використання таких методів дозволило б якнайповніше врахувати специфіку досліджуваного готельного підприємства,

організовуючи та обробляючи величезні масиви конкретної підприємницької інформації, а також підвищувати точність здійснюваного аналізу.

Наведені міркування обумовлюють наступну постановку задачі дослідження. Вона полягає у дослідженні даних окремого об’єкта готельного бізнесу, поданих у вигляді великого масиву інформації із численними атрибутами з метою побудови моделі овербукінгу для окремого підприємства із використанням базових методів машинного навчання Для реалізації запропонованої мети доцільно реалізувати наступні етапи дослідження:

- виявити джерела правдивої інформації про процеси овербукінгу;
- розробити математичну модель надійності бронювань готельних номерів;
- розробити методи попередньої обробки інформації з об’єкту дослідження для спрощення реалізації моделі;
- вибрати методи ідентифікації запропонованої моделі та здійснити їх програмну реалізацію;
- здійснити експерименти із відбору вагомих пояснювальних змінних на основі реальних даних та порівняти ефективність підходів.

1.2 Математична модель надійності бронювань готельних номерів

З метою отримання реальних нетривіальних даних можна звернутися до платформи Kaggle [49], де здійснюються змагання з аналітики та передбачувального моделювання з метою створення найефективніших моделей для прогнозування та опису даних. Дані для цих змагань пропонуються компаніями або користувачами. Такий підхід до побудови інтелектуального артефакту отримав назву краудсорсингу, тобто – передачу тих чи інших виробничих функцій невизначеній групі осіб без укладання трудового договору.

Змагання на Kaggle готує організатор, який подає дані та описує проблему. Пропоновані підходи завантажуються на сервер організаторів і шляхом

відкритих тестувань визначають переможця. Переможцю конкурсу виплачуються певні кошти за можливість використання запропонованого ним підходу.

Для дослідження використано дані з матеріалів конкурсу Hotel Booking Demand [65], керованого Jesse Mostipak (Texas, USA). Згідно з умовами конкурсу подається на розгляд значні обсяги даних двох готелів: міського та курортного. Не ставлячи собі за мету брати участь у конкурсі вибір такого виду даних для досліджень має ряд переваг: це дані із реальних об'єктів, вони значні за обсягом, а також достатньо неоднорідні, щоб забезпечити виклики конкурсу.

Відмінність пропонованої моделі від наведених в літературному огляді полягає перш за все у виборі результативної змінної. Пропонується моделювати схильності індивідуального споживача до використання забронюваного готельного номера на основі параметрів його замовлення, а також передісторії його відвідувань, якщо вона існує. Тобто моделюються особливості поведінки індивідуальних споживачів. Базою такого моделювання може служити класифікаційна функція K_b виду:

$$K_{b,j}(\vec{x}_C) = \begin{cases} 1, & \text{якщо бронювання реалізується,} \\ 0, & \text{якщо бронювання анулюється} \end{cases} . \qquad (1.1)$$

Дана функція прогнозує заповненість місць готелю в умовах його політики обслуговування та зони проживання, а також значення атрибутів бронювання $\vec{x}_C$ клієнта С та передісторії його обслуговування готелем для номерів цінового класу j.

На основі відомих наборів $B_j = \{b_j(\vec{x}_C)\}$ кількостей бронювань кімнат класу j неважко оцінити очікуваний обсяг $C_{R,j}$ зайнятих номерів даного класу:

$$C_{R,j} = \sum_{i=1}^{N_j} b_j(\vec{x}_{C_i}) K_{b,j}(\vec{x}_{C_i}) \qquad (1.2)$$

де i – номер клієнта, що бронює номер,

N_j- кількість клієнтів, що забронювали номери класу j.

Знаючи обсяги вільних номерів по класах на відповідний період та керуючись політикою готелю щодо овербукінгу менеджмент приймає обґрунтовані рішення щодо продовження або припинення бронювання.

Оскільки класифікаційна функція може приймати лише два можливих значення, її апроксимація може містити значні рівні похибок. Однак згортку виду (2) можна розглядати як певне усереднення при оцінці обсягу зайнятих номерів. Тому похибка обсягів $C_{R,j}$ може зменшуватися відносно похибки побудови значень $K_{b,j}(\vec{x}_{C_i})$ у величину порядку $\sqrt{N_j}$ разів.

1.3 Попередній аналіз даних та методи відбору головних змінних

При отриманні таблиці $B_C(C_1, \dots, C_{N_C})$ спостережень за процесом заселення заброньованих номерів клієнтами готельного підприємства від замовника дослідження насамперед необхідно виявити семантику стовпчиків вхідної таблиці. Серед згаданих стовпчиків ідентифікуємо результівний C_yта кандидатів пояснювальні атрибути $C_{x_1}, \dots, C_{x_M}$. При цьому в таблиці можуть зустрічатися атрибути із нечисловими значеннями. Значення таких атрибутів після консультацій із замовником можна оцифрувати або виключити атрибути із подальшого аналізу.

На основі відношення $B_X^p(C_{x_1}, \dots, C_{x_M})$ попередньо відібраних пояснювальних атрибутів формуємо попередню матрицю пояснювальних змінних X^p. Оскільки таблиця отримана емпірично, не виключена наявність сильно корельованих пояснювальних змінних, що спотворить весь подальший аналіз. З метою їх усунення будуємо кореляційну матрицю попередньо відібраних пояснювальних змінних $R_X^p(X_1, \dots, X_M)$. Якщо абсолютна величина елемента цієї матриці перевищує певний поріг

$$\left|r_{ij}^p\right| > r_T, \tag{1.3}$$

то один з атрибутів C_{x_i} або C_{x_j} виключається із відношення B_X^p, що призводить до відповідного зменшення матриці X^p. В результаті приходимо до фільтрованого відношення пояснювальних змінних $B_X^f\left(C_{x_1},\dots,C_{x_{M_f}}\right)$, де $M_f \leq M$.

На наступному етапі будуємо кореляції фільтрованих пояснювальних змінних із результивною

$$r_i^y = corr(X_i, \vec{y}), \tag{1.4}$$

отримуючи кореляційний вектор $\vec{R}^y = \left\{r_1^y,\dots,r_{M_f}^y\right\}$. Значення даного вектора впорядковуються за спаданням

$$\vec{R}_D^y = descending(\vec{R}^y) \tag{1.5}$$

Будуємо графік виду

$$(B_X^{f,D}, \vec{R}_D^y), \tag{1.6}$$

де $B_X^{f,D}$- вектор відношення B_X^f, компоненти якого посортовані відповідно до компонент вектора $\vec{R}_D^y$.

З аналізу діаграми визначаємо номер компоненти k_X^r вектора $B_X^{f,D}$ після якої спостерігається різке зменшення градієнта графіка $(B_X^{f,D}, \vec{R}_D^y)$. До основних компонент моделі $B_X^{P,r}$ віднесемо перші k_X^r компонент вектора $B_X^{f,D}$:

$$B_X^r = \left\{C_{x_i} \in \ B_X^{f,D}\right\}_{i=1}^{k_X^r} \quad . \tag{1.7}$$

відбір головних пояснювальних змінних за допомогою факторного аналізу.

Альтернативний метод вибору головних пояснювальних змінних реалізується на основі методу факторного аналізу. Подамо формалізований опис цього підходу. Факторний аналіз розпочинаємо на основі фільтрованого відношення пояснювальних змінних $B_X^f\left(C_{x_1},\dots,C_{x_{M_f}}\right)$ та відповідної кореляційної матриці

$$R_X^f\left(X_1,\dots,X_{M_f}\right). \tag{1.8}$$

Для матриці (8) будуємо вектор власних значень

$$L_X^f = \left\{\lambda_1, \dots, \lambda_{M_f}\right\} \tag{1.9}$$

та матрицю власних векторів

$$U_X^f = \left\{\vec{u}_1, \dots, \vec{u}_{M_f}\right\}. \tag{1.10}$$

Значення вектора власних значень впорядковуються за спаданням

$$L_{X,D}^f = descending(L_X^f) \tag{1.11}$$

Позначимо вектор перестановок компонент вектора L_X^f, що використовується для отримання вектора $L_{X,D}^f$, через $I_{L,D}^f$.

Будуємо графік виду

$$(I_{L,D}^f., L_{X,D}^f), \tag{1.12}$$

З аналізу діаграми визначаємо номер компоненти k_L^r вектора $L_{X,D}^f$ після якої спостерігається різке зменшення градієнта графіка $(I_{L,D}^f., L_{X,D}^f)$. До основних власних векторів віднесемо ті із векторів матриці U_X^f, індекси яких визначаються перші k_L^r компонент вектора $I_{L,D}^f$. Позначимо матрицю відібраних головних власних векторів через $U_X^{f,2}$.

Ця матриця визначає певний лінійний простір для побудови ефективної моделі. Далі важливо визначити такий його базис, який можна було б найлегше інтерпретувати в термінах початкових змінних. Такі перетворення можна здійснювати методами обертань. Із множини методів обертань найчастіше використовується варімакс, бо він найбільше сприяє полегшенню інтерпретації векторів базису простору пошуку параметрів математичних моделей.

Після обертань та інтерпретації факторів здійснюється зіставлення переліку отриманих найголовніших B_X^f пояснювальних змінних з аналогічним переліком, отриманим методом кореляційного аналізу B_X^r. Переліки пояснювальних змінних використовуються в подальшому моделюванні.

Після виділення головних пояснювальних компонентів моделі можна приступати до її ідентифікації. Як методи ідентифікації пропонуються

альтернатива із двох класичних підходів, які надалі можна розвинути в ансамбль методів.

Найчастіше дерево рішень служить узагальненням досвіду експертів, засобом передачі знань майбутнім співробітникам або моделлю бізнес-процесу компанії. На кожному кроці добудовується розгалуження, яке максимально зменшує невизначеність ситуації.

Описані методи реалізовані мовою програмування Python 3 із використанням основних бібліотек Pandas, Numpy та низки додаткових.

1.4 Експериментальні дослідження методів ідентифікації моделі надійності бронювань

На першому етапі із датафрейму hotel_bookings.csv було відібрано 15 оцифрованих стовпчиків із 32 наявних. В першу чергу було виділено результативний атрибут “is_canceled”, який приймає значення із множини {0, 1}. Значенню 1 відповідає скасування бронювання, а значенню 0 – використання забронюваного номера. Пояснення інших попередньо відібраних атрибутів наведено в наступній таблиці.

Таблиця 1.1

Зміст попередньо відібраних пояснювальних змінних файлу hotel_bookings.csv

Номер	Назва атрибута	Зміст атрибута
1	lead_time	Кількість днів, що минули між введенням дати бронювання в систему управління майном та датою прибуття
2	arrival_date_week_number	Номер тижня року для дати прибуття (1-53)
3	adults	Кількість дорослих

Продовження таблиці 1.1

Номер	Назва атрибута	Зміст атрибута
4	children	Кількість дітей
5	babies	Кількість немовлят
6	meal	Харчування - множина значень: Undefined/SC (без харчування) BB – Bed & Breakfast (ліжко і сніданок) HB – Half board (сніданок, lunch or dinner) FB – Full board (breakfast, lunch, dinner)
7	is_repeated_guest	Значення вказує, чи ім'я бронювання було від повторного гостя) 1: Так 0: Ні
8	previous_cancellations	Попереднє скасовування бронювання
9	previous_bookings_not_canceled	Кількість попередніх бронювань, які клієнт не анулював до поточного бронювання
10	booking_changes	Кількість змін / змін, внесених до бронювання з моменту введення бронювання на ПМС до моменту реєстрації або скасування)

Продовження таблиці 1.1

Номер	Назва атрибута	Зміст атрибута
11	deposit_type	Вказівка на те, чи зробив клієнт депозит для гарантування бронювання. Ця змінна може приймати три категорії: 1) No Deposit (депозит не вносився) 2) 1 sometimes 0 Non Refund (було внесено заставу у розмірі суми загальної вартості перебування) 3) 0 or 1 Refundable (було внесено заставу на меншу суму)
12	day_in_waiting_list	Кількість днів, коли бронювання було в списку очікування, перш ніж воно було підтверджено замовником
13	required_car_parking_ spaces	Кількість паркувальних місць, необхідних замовнику
14	total_of_special_requ ests	Кількість спеціальних запитів, зроблених замовником (наприклад, двоспальне ліжко або високий поверх)

Аналіз здійснювався над масивом даних, що містив понад 40000 записів. Для попередньо відібраних пояснювальних змінних була побудована кореляційна матриця, коефіцієнти якої наведені в наступній таблиці.

Таблиця 1.2

Коефіцієнти стрічок кореляційної матриці пояснювальних змінних

Номер стрічки	Ідентифікатор змінної	Значення кореляційних коефіцієнтів
1	lead	1.000 0.121 0.137 0.001 0.001 0.150 -0.150 0.094 -0.108 0.075 0.202 0.089 -0.151 -0.008
2	arri	0.121 1.000 0.053 0.016 0.017 0.049 -0.078 0.043 -0.052 0.011 -0.014 0.062 0.008 0.045
3	adul	0.137 0.053 1.000 0.073 0.023 0.062 -0.125 0.006 -0.133 -0.011 0.001 -0.012 0.013 0.079
4	chil	0.001 0.016 0.073 1.000 0.020 0.029 -0.043 -0.020 -0.036 0.044 -0.061 -0.021 0.047 0.028
5	babi	0.001 0.017 0.023 0.020 1.000 0.024 -0.021 -0.009 -0.017 0.098 -0.026 -0.008 0.035 0.131
6	meal	0.150 0.049 0.062 0.029 0.024 1.000 -0.047 0.086 -0.032 0.019 0.095 -0.061 -0.011 0.020
7	is_rep	-0.150 -0.078 -0.125 -0.043 -0.021 -0.047 1.000 0.005 0.428 0.001 -0.045 -0.011 0.055 0.001
8	prev_can	0.094 0.043 0.006 -0.020 -0.009 0.086 0.005 1.000 0.022 -0.026 0.232 -0.005 -0.027 -0.045

Продовження таблиці 1.2

Номер стрічки	Ідентифікатор змінної	Значення кореляційних коефіцієнтів
9	prev_book	-0.108 -0.052 -0.133 -0.036 -0.017 -0.032 0.428 0.022 1.000 -0.002 -0.032 -0.008 0.044 0.015
10	book	0.075 0.011 -0.011 0.044 0.098 0.019 0.001 -0.026 -0.002 1.000 -0.067 0.015 0.062 0.030
11	depo	0.202 -0.014 0.001 -0.061 -0.026 0.095 -0.045 0.232 -0.032 -0.067 1.000 -0.001 -0.082 -0.165
12	days	0.089 0.062 -0.012 -0.021 -0.008 -0.061 -0.011 -0.005 -0.008 0.015 -0.001 1.000 -0.013 -0.049
13	requ	-0.151 0.008 0.013 0.047 0.035 -0.011 0.055 -0.027 0.044 0.062 -0.082 -0.013 1.000 0.075
14	total	-0.008 0.045 0.079 0.028 0.131 0.020 0.001 -0.045 0.015 0.030 -0.165 -0.049 0.075 1.000

Як бачимо, максимальний коефіцієнт кореляції значень пояснюючих змінних складає 0.232, що не дає жодних підстав для виключення будь-якої із них із розгляду. Отримано наступні значення компонент кореляційного вектора

$$\vec{R}^{y} = \{0.229 \quad 0.022 \quad 0.081 \quad 0.081 \quad -0.023 \quad 0.081 \quad -0.104\}$$

$$\vec{R}^{y} = \{0.114 \quad -0.077 \quad -0.115 \quad 0.317 \quad -0.036 \quad -0.244 \quad -0.101\} \tag{1.13}$$

Як бачимо, найвищу кореляцію на результівну змінну має одинадцята пояснювальна. Компоненти впорядкованого за спаданням кореляційного вектора $\vec{R}_D^y$ наведені у наступній таблиці.

Таблиця 1.3

Координати впорядкованого кореляційного вектора $\vec{R}_D^y$

Номер компонента	Назва атрибута	Значення компонента
1	*deposit_type*	0.317
2	*required_car_parking_spaces*	0.244
3	*lead_time*	0.229
4	*booking_changes*	0.115
5	*previous_cancellations*	0.114
6	*is_repeated_guest*	0.104
7	*total_of_special_requests*	0.101
8	*meal*	0.081
9	*children*	0.081
10	*adults*	0.081
11	*previous_bookings_not_canceled*	0.081
11	*days_in_waiting_list*	0.036
12	*babies*	0.023
13	*arrival_date_week_number*	0.022

На основі аналізу діаграми впорядкованих коефіцієнтів кореляції результівної та пояснювальної змінних, наведеної на наступному рисунку, виділено чотири перших компоненти вектора $\vec{R}_D^y$:

$$B_X^r = \{B_{C,10},\ B_{C,12}, B_{C,1}, B_{C,9}\} =$$
$$= \{deposit\_type\ \ , require\_car\_parking\_spaces\ ,$$
$$lead\_time, booking\_changes\ \} \quad (1.14)$$

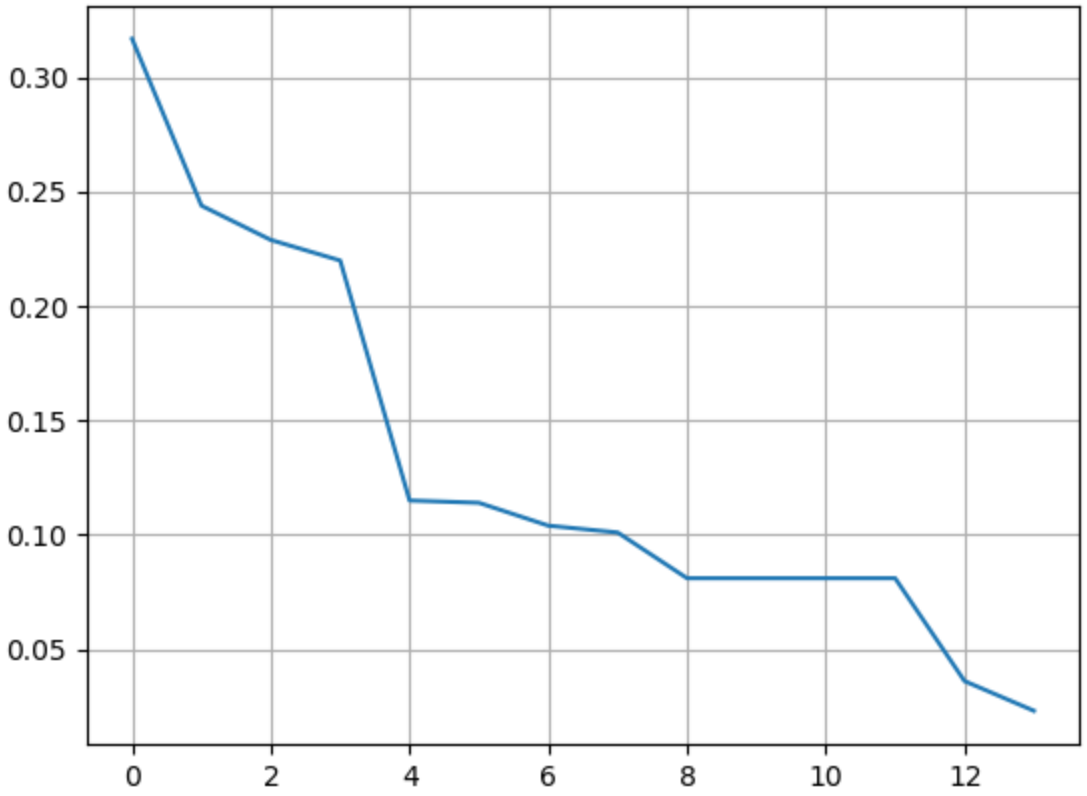

Рис. 1.1. Діаграма впорядкованих коефіцієнтів кореляції результівної та пояснювальної змінних

Надалі спробуємо встановити базові пояснювальні змінні на основі факторного аналізу умови задачі. Будуємо власні значення та власні вектори кореляційної матриці пояснювальних змінних.

$$L_X^f = [1.72956295, 1.4532328, 0.56786566, 0.64191685, 1.22104972, 0.76009482, 1.10278745, 0.82141734, 0.86156586, 0.90409295, 0.92765071, 0.97695338, 1.00891922, 1.0228903].$$

(1.15)

Значення компонентів власних векторів наводяться в наступній таблиці.

Таблиця 1.4

Компоненти власних векторів кореляційної матриці пояснювальних змінних

Номер	Значення компонент власного вектора
1	[-0.45954115 -0.11751983 0.11474242 0.59060216 -0.23132488 0.42239521 0.25015893 0.03538018 0.07842739 0.14645348 0.02966238 -0.29537074 0.03903252 -0.005233]
2	[-0.20328913 0.108688 0.01847447 -0.1575506 -0.19754699 -0.00268382 0.35046543 -0.32635939 0.21421008 -0.35725709 -0.44421942 0.22245724 0.03795172 -0.48119792]
3	[-0.28173454 0.24402367 -0.05418573 -0.14179024 -0.10106789 -0.29986827 -0.1618554 -0.34024459 0.2204815 0.47135454 0.42841723 -0.15155521 -0.01582751 -0.34407887]
4	[-0.04757039 0.25238408 0.01736972 -0.03314575 -0.1014525 0.13727187 -0.16486794 0.07596584 -0.07614902 -0.51778433 0.27451514 -0.27461256 -0.65618585 -0.12369634]
5	[-0.01452054 0.26248372 0.00907958 0.12539204 -0.35730397 -0.06878446 0.06574734 -0.42031609 -0.24574247 -0.30814841 0.34029676 0.20952429 0.31654214 0.43180325]

Продовження таблиці 1.4

Номер	Значення компонент власного вектора
6	[-0.24227045 -0.06058174 -0.00937275 -0.10011842 -0.4061815 -0.2521217 -0.26636588 0.05155718 -0.6218577 0.13453712 -0.41814869 -0.20900339 -0.03383831 -0.02970901]
7	[0.48115141 -0.26195136 0.70756537 -0.0654215 -0.31810333 -0.02156232 0.07532487 -0.13905574 0.03152276 0.06558524 0.09355407 -0.18425349 -0.0164461 -0.14147008]
8	[-0.18920467 -0.3658158 0.02201252 0.21617265 -0.3200214 -0.45122184 -0.14376712 0.38598919 0.26982914 -0.17726185 0.18192069 0.40303502 -0.07765652 -0.04124707]
9	[0.45110449 -0.27558043 -0.69173431 0.0904975 -0.36567275 0.06686598 0.1005598 -0.10253129 0.03524444 0.01356575 0.07347705 -0.19821806 0.00277539 -0.17011253]
10	[-0.00575927 0.20810803 -0.03755381 -0.24884718 -0.30893229 -0.14342924 0.31316248 0.10039873 0.38985261 0.20645319 -0.22553854 -0.13907122 -0.30658346 0.55709164]
11	[-0.29656378 -0.48101901 -0.03296513 -0.56388838 -0.11963107 0.47183332 -0.12341819 -0.13732277 0.00330941 0.03046211 0.17815889 0.18363419 -0.05756002 0.14447084]
12	[-0.06554806 -0.04995497 -0.01944855 -0.15518953 0.08812456 -0.14840405 0.72009908 0.22662562 -0.44561879 0.1221669 0.32016436 0.11808966 -0.14041781 -0.13923349]
13	[0.19947056 0.24327874 0.01642626 0.22353933 -0.18341684 0.29804316 -0.11539109 -0.0586025 -0.13066 0.39141047 -0.08792733 0.61205223 -0.38889022 -0.0878495]
14	[0.06390394 0.40473487 0.01760843 -0.25742551 -0.32531324 0.28622478 -0.06577784 0.57341883 0.06138638 -0.00153993 0.1102185 -0.00355443 0.43622671 -0.19599404]

Сортовані власні значення кореляційної матриці:

$$L^{f}_{X,D} = $$
$$= [1.72956295, 1.4532328, 1.22104972, 1.10278745, 1.0228903,$$
$$1.00891922, 0.97695338, 0.92765071, 0.90409295,$$
$$0.86156586, 0.82141734, 0.76009482, 0.64191685\ 0.56786566].$$

(1.16)

При цьому індекси сортованих власних значень складають:

$$I^{f}_{L,D} = [0\ \ 1\ \ 4\ \ 6\ 13\ 12\ 11\ 10\ \ 9\ \ 8\ \ 7\ \ 5\ \ 3\ \ 2] \quad (1.17)$$

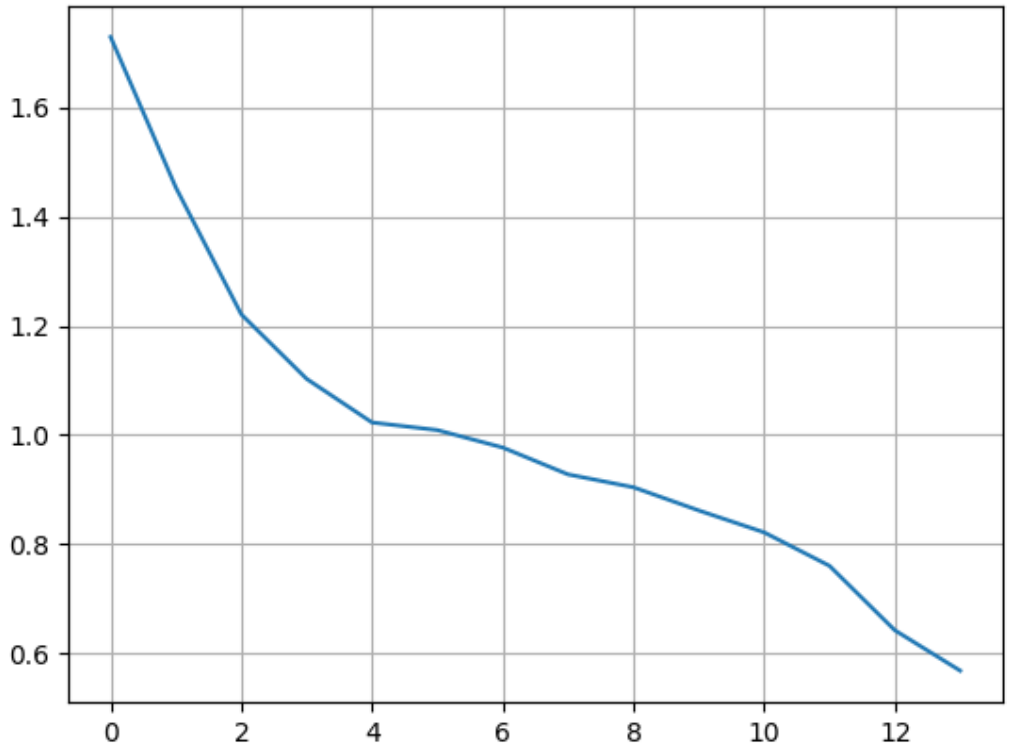

Рис. 1.2. Діаграма впорядкованих власних значень кореляційної матриці пояснювальних змінних

В результаті аналізу для подальшого дослідження було відібрано 5 найбільших власних значень, значення яких перевищують одиницю. В результаті обертання відповідних власних векторів методом “варімакс” отримуємо вектори максимально прив’язані до певних пояснювальних змінних.

Для спрощення виявлення такої прив’язки вибираємо поріг значимості компонентів власних векторів на рівні 0.33. Всі компоненти власних векторів, значення яких менші порогового рівня прирівнюємо до нуля. В результаті отримуємо подання власних векторів, зручні побудови інтерпретацій з точки зору пояснювальних змінних.

```
[[ 0.      0.     -0.3316  0.      0.    ]
 [ 0.      0.     -0.4083  0.4862  0.    ]
 [ 0.      0.     -0.4369  0.      0.    ]
 [ 0.      0.      0.      0.      0.    ]
 [ 0.      0.      0.      0.      0.5989]
 [ 0.      0.     -0.4539  0.      0.    ]
 [ 0.6522  0.      0.      0.      0.    ]
 [ 0.     -0.4106  0.      0.      0.    ]
 [0.6702   0.      0.      0.      0.    ]
 [ 0.      0.      0.      0.      0.7328]
 [ 0.     -0.5941  0.      0.      0.    ]
 [ 0.      0.      0.      0.7339  0.    ]
 [ 0.      0.      0.      0.      0.    ]
 [ 0.      0.3989 -0.3713  0.      0.    ]]
```

Рис. 1.3. Приклад подання власних векторів

Для виявлення семантики компонент власних векторів виводимо номери пояснювальних змінних, які відповідають значущим значенням:

```
[[ 0  0  1  0  0]
 [ 0  0  2  2  0]
 [ 0  0  3  0  0]
 [ 0  0  0  0  0]
 [ 0  0  0  0  5]
 [ 0  0  6  0  0]
 [ 7  0  0  0  0]
 [ 0  8  0  0  0]
 [ 9  0  0  0  0]
```

[0 0 0 0 10]

[0 11 0 0 0]

[0 0 0 12 0]

[0 0 0 0 0]

[0 14 14 0 0]]

Із аналізу перших двох власних векторів вибираємо компоненти

$$B_X^f = [7, 9, 10, 13] \quad (1.17)$$

Третій власний вектор вже важко інтерпретується, оскільки йому відповідає аж чотири значимих пояснювальних змінних.

Для моделювання надійності бронювань були вибрані моделі множинної лінійної регресії, а також модель дерева рішень. Як змінні моделі вибиралися пояснювальні змінні із номерами, що визначаються співвідношеннями (14) та (18). Результати ідентифікації наведені в наступній таблиці.

Таблиця 1.5

Результати ідентифікації моделей надійності бронювань

Вид моделі	Спосіб відбору змінних моделі	Похибка ідентифікації (%)
Множинна лінійна регресія	Кореляційний аналіз, (14); коефіцієнти: 5.43596652e-04 -5.79119088e-02 7.57438669e-01 -2.77633703e-01	34.87
Множинна лінійна регресія	Факторний аналіз (18) -0.15377676 -0.05765907 0.80458172 - 0.06787049	34.92
Дерево рішень	Кореляційний аналіз, (14)	24.41
Дерево рішень	Факторний аналіз (18)	24.41

Скласти уявлення про масштаби побудованих дерев рішень можна за допомогою наступних ілюстрацій.

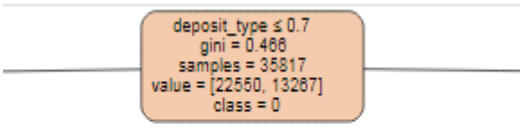

Рис. 1.4 Кореневий вузол дерева прийняття рішень,
побудованого на основі набору даних (1.1)

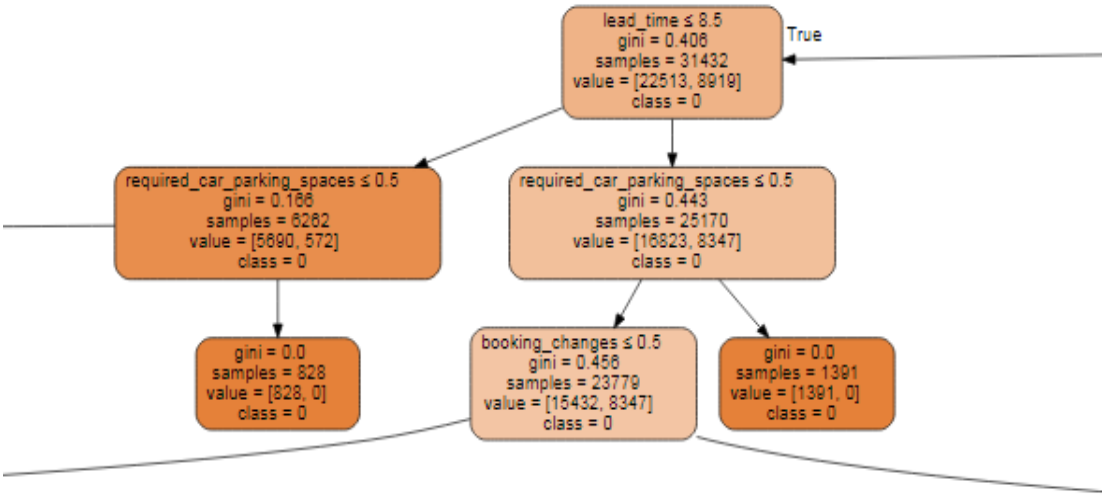

Рис. 1.5. Фрагмент лівого піддерева прийняття рішень,
побудованого на основі набору даних (1.1)

Аналіз результатів таблиці 5 дозволяє зробити висновок про рівноцінність застосування методів кореляційного та факторного аналізу при підготовленні даних для моделювання. Тому враховуючи простоту кореляційного аналізу, йому належить віддати цілковиту перевагу.

Порівняно велика максимальна похибка оцінки лояльності клієнта щодо здійсненого бронювання (24%) при врахуванні завантаженості великої кількості

номерів може бути значно зменшена шляхом взаємної компенсації окремих похибок.

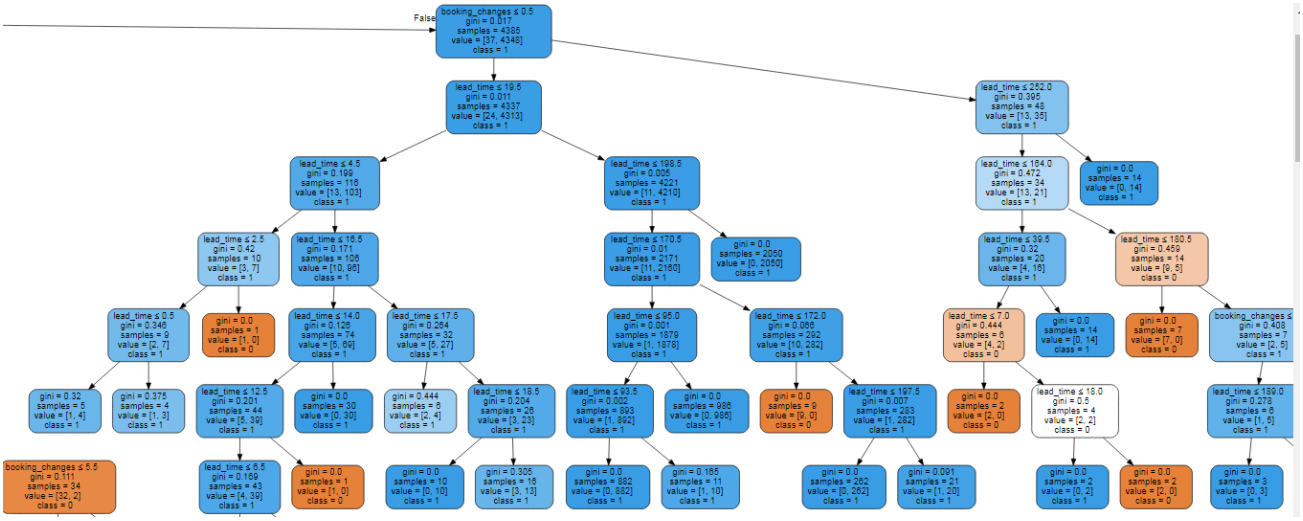

Рис. 1.6. Фрагмент правого піддерева прийняття рішень, побудованого на основі набору даних (14)

ВИСНОВКИ ДО РОЗДІЛУ 1

1. На основі аналізу літературних джерел у роботі запропоновано модель надійності бронювань номерів у готельному бізнесі, яка дозволяє прогнозувати очікувану завантаженість готельних номерів по цінових класах для підтримки особливостей політики овербукінгу готельного підприємства.
2. Запропоновано та реалізовано методики попередньої обробки даних підприємства щодо наслідків здійснюваної політики овербукінгу на основі методів кореляційного та факторного аналізу.
3. Проаналізовано ефективність реалізації моделі надійності бронювань за допомогою методів множинної лінійної регресії та дерева рішень.
4. Експерименти на реальних даних засвідчили вищу ефективність поєднання методів попередньої обробки інформації на основі кореляційного аналізу та реалізації моделі надійності бронювань за допомогою дерев рішень. Значна максимальна похибка оцінки лояльності клієнта щодо здійсненого бронювання (24%) при врахуванні завантаженості великої кількості номерів може бути значно зменшена.
5. Надалі точність моделювання можна буде підвищити шляхом побудови ансамблевих методів ідентифікації.

РОЗДІЛ 2. ЗАСОБИ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ВЕБ-ІНФОРМАЦІЙНОЇ СИСТЕМИ ГОТЕЛЬНОГО БІЗНЕСУ

2.1. Засоби реалізації диспетчера веб-інформаційної системи

Розробка Веб-інформаційної системи – це доволі складний процес, для створення якої необхідно залучити різні програмні засоби та технології. Для організації правильної та коректної роботи Веб-інформаційної системи, необхідно визначити оптимальні засоби реалізації.

Система автоматизованої обробки інформації має важливу роль для підвищення ефективності діяльності готельного бізнесу. Мають значення способи реєстрації та обробки інформації про гостей, а також збереження і передачу інформації у потрібній формі. Крім цього, на основі базової інформації виводиться нова числова чи графічна, яка використовується для подальшої оптимізації діяльності готелю.

Для розробки Веб-інформаційної системи доцільно використовувати операційну систему Linux Ubuntu. Вона базується на Debian GNU/Linux. Основна особливість цієї ОС – зручність та комфортне користування звичайного користувача. Також вона повністю безкоштовна, а тому розв’язує актуальну проблему інтернет-піратства.

«Ubuntu містить базовий набір програм загального призначення: багатовіконне стільничне середовище, браузер, текстовий редактор, засіб організації електронної пошти, офісні програми тощо. Серед базових програм міститься також термінал – вікно для написання текстових команд. Завдяки йому користувач може виконувати переважну більшість дій та налаштувань в самій ОС» [20].

Нині використовуються різні об'єктно орієнтовані засоби проєктування. Популярною є UML (від англ. The Unified Modeling Language). Ця стандартизована мова використовує графічні позначення (діаграми) для

створення абстрактної моделі (візуалізації) системи. Вона була створена для проєктування, створення та документування програмних додатків і систем. Основними перевагами використання UML є:

1) можливість розглянути систему різнобічно з подальшим просуванням;
2) підтримка всіх етапів життєвого циклу інформаційної системи та можливість обрати відповідний набір діаграм [10].

UML представляє збірку найкращих інженерних практик, які виявилися успішними в моделюванні великих і складних систем. «UML є дуже важливою частиною розробки об'єктноорієнтованого програмного забезпечення та процесу розробки програмного забезпечення. UML використовує переважно графічні нотації для вираження дизайну проєктів програмного забезпечення. Використання UML допомагає командам проєктів спілкуватися, досліджувати потенційні завдання та перевіряти архітектурний дизайн програмного забезпечення» [19].

Використання UML актуальне, адже стратегічна цінність програмного забезпечення зростає для багатьох компаній, галузь шукає методи автоматизації виробництва програмного забезпечення та покращення якості та скорочення вартості та часу виходу на ринок. Ці методи включають технологію компонентів, візуальне програмування, шаблони та фреймворки.

«Підприємства також шукають методи управління складністю систем у міру того, як вони збільшуються в масштабі. Зокрема, вони визнають необхідність розв’язання повторюваних архітектурних проблем, таких як фізичний розподіл, паралелізм, реплікація, безпека, балансування навантаження та відмовостійкість» [21].

Враховуючи поставлені раніше завдання, оптимальними засобами для розробки такої інформаційної системи є застосування мови програмування Python, фреймворку Angular**,** кросплатформного фреймворку ASP.NET Core, а також бази даних SQL.

2.2. Засоби розроблення інформаційної системи з використанням ASP.NET Core Web API (Python), розроблення авторизації (JWT Token)

ASP.NET Core – це міжплатформна структура з відкритим вихідним кодом, яка використовується для створення сучасних хмарних вебпрограм для Windows, macOS або Linux. Вона створена на основі дизайну ASP.NET 4.x, однак містить зміни в архітектурі, що призвело до створення компактнішої та більш модульної структури.

ASP.NET Core має такі переваги:

1) уніфікована історія створення вебінтерфейсу користувача та веб-API;
2) можливість розробки та запуску на Windows, MacOS та Linux;
3) відкритий вихідний код і орієнтований на спільноту;
4) інтеграція сучасних клієнтських фреймворків і робочих процесів розробки;
5) система налаштування на основі середовища, готова до роботи в хмарному середовищі;
6) вбудоване впровадження залежностей;
7) легкий, високопродуктивний і модульний конвеєр запитів HTTP;
8) можливість розміщення на IIS, Nginx, Apache, Docker або самостійного розміщення у власному процесі;
9) паралельне керування версіями додатків при націленні на .NET Core [45].

Варто зазначити, що існує два види ASP.NET – MVC і Web API. Контролер представлення моделі (MVC) ділить програму на три частини: модель, представлення та контролер. ASP.NET має багато варіантів для створення вебдодатків за допомогою вебформ ASP.NET. MVC framework Поєднує такі

функції ASP.NET, як головні сторінки, автентифікація на основі членства. MVC існує в збірці "System.Web.MVC".

Водночас вебінтерфейс ASP.NET дозволяє зображати дані в різних форматах, таких як XML і JSON. Це структура, яка використовує служби HTTP і дозволяє легко надавати відповідь на запит клієнта. Відповідь залежить від запиту клієнтів. Веб-API створює служби HTTP та керує запитом за допомогою протоколів HTTP. Веб-API є відкритим кодом, і його можна розмістити в програмі або в IIS. Запит може бути GET, POST, DELETE або PUT. Як результат, Web API – це:

1) служба HTTP;
2) розрахований на охоплення широкого кола клієнтів;
3) використовує програму HTTP [34].

Між MVC і Web API існує істотна різниця, зокрема:

1) MVC можна використовувати для розробки вебпрограми, яка відповідає і як дані, і як представлення даних, але веб-API використовується для створення HTTP-сервісів, які відповідають лише як дані;
2) у веб-API запит виконує трасування з діями залежно від служб HTTP, але запит MVC виконує трасування з назвою дії;
3) вебінтерфейс API повертає дані в різних форматах, таких як JSON, XML та інших форматах на основі заголовка accept запиту. Але MVC повертає дані у форматі JSON за допомогою JSONResult;
4) веб-API підтримує узгодження вмісту, самостійне розміщення. Усе це не підтримується MVC;
5) веб-API містить різні функції MVC, такі як маршрутизація, прив’язка моделі, але ці функції відрізняються та визначені в збірці «System.Web.Http». А функції MVC визначені в збірці "System.Web.Mvc";
6) веб-API допомагає створювати служби RESTful через .Net Framework, але MVC не підтримує цю функцію [48].

Поєднання контролера MVC і API в одному проєкті дозволяє керувати запитами AJAX і повертає відповідь у XML, JSON та інших форматах. Крім цього, поєднання можливе для ввімкнення авторизації для програми. У ньому потрібно створити два фільтри, один для Web API, а інший для MVC.

ASP.NET Core забезпечує більшу кількість покращень у порівнянні з ASP.NET MVC/Web API. По-перше, це один фреймворк, а не два. Це зручно і не викликає зайвої плутанини. По-друге, з'являються контейнери журналювання та DI без додаткових бібліотек, що економить час і дозволяє зосередитися на написанні кращого коду замість того, щоб вибирати та аналізувати найкращі бібліотеки.

Створити проєкт ASP.NET Core Web API можна за допомогою Visual Studio. Це можна зробити двома способами:

1) Web-API з проєктом MVC;
2) Автономний проєкт Web API.

Далі доцільніше зосередитися на створенні Web API у Visual Studio 2019, зважаючи на особливості двох підходів.

1. Відкрити Visual Studio 2019 і натиснути «Створити новий проєкт».
2. Обрати шаблон ASP.NET Core Web Application і натиснути «Далі».
3. Вказати відповідне ім'я, розташування та ім'я рішення для програми ASP.NET Core. У цьому прикладі можна назвати «MyFirstCoreWebApp» і натиснути кнопку «Створити».
4. Обрати відповідний шаблон вебпрограми ASP.NET Core, наприклад Empty, API, Web Application, MVC тощо. Тут потрібно створити вебпрограму, тому варто обрати шаблон вебпрограми. На цьому етапі не потрібен HTTPS, тому варто зняти прапорець «Налаштувати для HTTPS». Також варто переконатися, що обрано відповідні версії .NET Core і ASP.NET Core. Натиснути кнопку «Створити», щоб створити проєкт.
5. Щоб запустити цю вебпрограму, потрібно натиснути IIS Express або Ctrl + F5. Відкриється браузер і зобразиться пуста початкова сторінка. Цей

результат надходить з Index.cshtml сторінки в папці «Сторінки». Піктограму IIS Express також можна побачити на панелі запущених завдань [50].

Створення інформаційного проєкту за допомогою мови Python можливе з використанням IronPython. Це – реалізація мови програмування Python з відкритим кодом, яка тісно інтегрована з .NET. IronPython може використовувати бібліотеки .NET і Python, водночас інші мови .NET теж можуть використовувати код Python.

IronPython – це вдале доповнення до .NET, завдяки якому розробники Python використовують потужність .NET як швидку та експресивну мову сценаріїв для вбудовування, тестування або написання нової програми з нуля [53].

Створення інформаційної системи потребує також розробку авторизації. Вона є незалежною від автентифікації. Однак для авторизації потрібен механізм автентифікації. «Автентифікація – це підтвердження того, ким є користувач на вході. Це проходження перевірки автентичності» [2]. При використанні аутентифікації можливе створення одного або кількох посвідчень для користувача.

ASP.NET Core авторизація надає просту декларативну роль та повнофункціональні моделі на основі політик. Авторизація виявляється у вимогах, і обробники оцінюють твердження користувача за вимогами. Імперативні перевірки можуть бути засновані на простих політиках або політиках, які оцінюють як посвідчення користувача, і властивості ресурсу, до якого намагається отримати доступ користувач. Простір імен Компоненти авторизації, включаючи атрибути AuthorizeAttribute, AllowAnonymousAttribute знаходяться в Microsoft.AspNetCore.Authorization просторі імен.

JSON Web Token (JWT) – це відкрита галузева стандартна техніка для безпечної передачі претензій між двома сторонами. «Веб-токен JSON (JWT) — компактний та безпечний спосіб представлення «заявок» (claims) для передачі

між двома сторонами. Заявки у JWT являють собою JSON-об'єкт, підписаний за допомогою JSON Web Signature (JWS)» [46].

Це означає, що дані, які пересилаються між двома сторонами за допомогою JWT, мають цифровий підпис, їх можна легко підтвердити та довіряти. За допомогою JWT можна реалізувати автентифікацію та авторизацію в будь-якій вебпрограмі з будь-яким внутрішнім стеком, тобто Asp.net Core, Java, NodeJs, Python тощо.

2.3. Засоби реалізації інформаційної системи за допомогою фреймворків (Angular)

Фреймворки — це програмне забезпечення, яке розробляється та використовується розробниками для створення нових програм та вебпродуктів. За їх допомогою можна створювати складніші та масштабніші проєкти, а також додавати до чинних нові нестандартні рішення. Фреймворки є універсальними, надійними та ефективними, адже часто створюються, тестуються та оптимізуються багатьма досвідченими інженерами програмного забезпечення та програмістами. Програмні інфраструктури полегшують життя розробників, дозволяючи їм контролювати весь процес розробки програмного забезпечення або більшу його частину з однієї платформи [58].

Як правило, фреймворк складається з трьох частин:

1) модель – відповідає за саму структуру, каркас, формування бізнес-логіки;
2) подання–відображення масивів інформації;
3) контролер – реалізація зв’язку з користувачами, перетворення його інформації в команди для роботи попередніх двох частин.

Використання програмного середовища для розробки додатків дозволяє зосередитися на функціональності додатка високого рівня. Це тому, що будь-яка низькорівнева функціональність опікується самою структурою.

Переваги використання програмного фреймворку:

1) сприяє встановленню кращих практик програмування та адаптованому використанні шаблонів проєктування;
2) код більш безпечний, а дані зберігаються з максимальною безпекою;
3) ймовірність уникнення повторюваного та зайвого коду;
4) сприяє послідовній розробці коду без помилок;
5) оптимізація роботи над складними технологіями;
6) сприяє розвитку продукту, додавання нових рішень;
7) можна створити свою програмну структуру або зробити внесок у фреймворки з відкритим кодом. Як результат, функціональність постійно вдосконалюється;
8) кілька сегментів коду та функціональних можливостей попередньо створено та перевірено. Це робить програми більш надійними;
9) тестування та налагодження коду набагато простіше, і це можуть зробити навіть розробники, які не володіють кодом;
10) час, необхідний для розробки програми чи сайту, скорочується [1].

Фреймворки створюються відповідно до потреб. Незалежно від того, чи це робота над вебсайтом, наука про дані, керування базами даних або мобільні додатки, програмні інфраструктури існують для всіх видів програмування.

Варто також розрізняти терміни «бібліотеки» та «фреймворки», які інколи використовують як синоніми, однак вони відрізняються. Зокрема, «бібліотека – це набір раніше написаного коду, який можна використовувати для створення власного коду. Фреймворк – це опорна конструкція, яка вимагає конкретності. Необхідно дотримуватися шаблону фреймворку» [38].

Фреймворк, по суті, є скелетом, який використовує написаний програмістом код. Цей код звертається до бібліотеки. Фреймворки можуть включати й часто містять бібліотеки. Бібліотеки використовуються для заповнення функцій.

Існує багато типів програмних інфраструктур, які полегшують розробку додатків для широкого діапазону доменів розробки додатків. «Фреймворки

зазвичай пов’язані з певною мовою програмування та підходять для різних типів завдань. Наприклад: популярні фреймворки для Python – Flask, Django, Tornado, Web2py; поширені рішення для JavaScript – React, VueJS, Angular; кращі фреймворки для вебдодатків – Angular, Rails, Express» [12].

Angular – це зовнішній фреймворк, створений на основі JavaScript. Це один із найпопулярніших вебфреймворків, який був створений Google, а нині це фреймворк із відкритим кодом. Він містить такі функції, як двостороннє зв’язування даних, що скорочує час розробки, і впровадження залежностей, що полегшує взаємодію різних фрагментів коду один з одним. Крім цього, він містить вбудовані атрибути, за допомогою яких можна підвищити продуктивність [47].

«Фреймворк Angular можна розділити на три основні частини:

1) ng–app – ця директива визначає та пов’язує програму Angular з HTML;
2) ng-model – прив’язує значення даних програми Angular до елементів керування введенням HTML;
3) ng-bind – прив’язує дані програми Angular до тегів HTML» [26].

Angular змінює статичний HTML на динамічний HTML. Він розширює можливості HTML, додаючи вбудовані атрибути та компоненти, а також надає можливість створювати власні атрибути за допомогою простого JavaScript.

Основні функції Angular такі:

1) прив’язка даних – це автоматична синхронізація даних між компонентами моделі та представлення;
2) область – це об’єкти, які посилаються на модель. Вони діють як сполучення між контролером і представленням;
3) контролер – це функції JavaScript, прив’язані до певної області;
4) служби – Angular поставляється з кількома вбудованими службами, такими як $http, для створення

XMLHttpRequests. Це єдині об'єкти, екземпляри яких створюються лише один раз у програмі;

5) фільтри – вони вибирають підмножину елементів із масиву та повертають новий масив;
6) директиви – це маркери елементів DOM, таких як елементи, атрибути, css тощо. Їх можна використовувати для створення власних тегів HTML, які слугуватимуть новими власними віджетами. Angular має вбудовані директиви, такі як ngBind, ngModel тощо;
7) шаблони – це подання інформації від контролера та моделі. Це може бути один файл (наприклад, index.html) або кілька переглядів на одній сторінці з використанням часткових елементів;
8) маршрутизація – це концепція перемикання переглядів;
9) Model View Whatever – MVW – це шаблон проєктування для поділу програми на різні частини, які називаються Model, View і Controller, кожна з яких має окремі обов'язки. Angular не реалізує MVC у традиційному розумінні, а радше щось ближче до MVVM (Model-View-ViewModel).
10) глибинне посилання – воно дозволяє закодувати стан програми в URL-адресі, щоб її можна було додати до закладок. Потім програму можна відновити з URL-адреси до того самого стану;
11) ін'єкція залежностей – Angular має вбудовану підсистему ін'єкції залежностей, яка допомагає розробнику легко створювати, розуміти та тестувати програми [28].

На відміну від React і VueJS, фреймворк Angular складніший. Він використовується для створення вебдодатків, вебсайтів і односторінкових програм. Він розширює HTML і робить його динамічним. Angular повністю базується на HTML і JavaScript, тому це відкидає потребу у вивченні сторонньої мови програмування або синтаксису.

Основні переваги Angular:

1) надає можливість писати клієнтські програми за допомогою JavaScript у чистий спосіб Model View Controller (MVC);
2) програми, написані на Angular, є кросбраузерними. Angular автоматично обробляє код JavaScript, який підходить для кожного браузера;
3) Angular є відкритим кодом, повністю безкоштовним має ліцензією Apache версії 2.0.
4) немає потреби вивчати іншу мову сценаріїв, адже це чистий JavaScript і HTML;
5) має вбудовані атрибути (директиви), які роблять HTML динамічним;
6) фреймворк легко розширювати та налаштовувати;
7) підтримка можливості створення односторінкової програми чи додатку;
8) фреймворк використовує ін'єкцію залежностей і розділення проблем;
9) має простий модульний тест.

Для налаштування середовища розробки для Angular потрібно використати такі інструменти:

1) бібліотека Angular;
2) редактор/IDE;
3) браузер;
4) вебсервер.

Завантажити бібліотеку Angular можна з сайту Angular.org. Потрібно натиснути кнопку завантаження, обрати потрібну версію зі спливного вікна та натиснути кнопку завантаження у спливному вікні.

Angular є кодом HTML і JavaScript. Тому можна встановити будь-який хороший редактор/IDE, наприклад, Visual Studio. Також можна використовувати будь-який вебсервер, наприклад, IIS, Apache тощо.

Крім цього, можна використовувати будь-який браузер, адже Angular підтримує кросбраузерну сумісність. Однак під час розробки програми рекомендується використовувати Google Chrome.

Налаштування проєкту Angular у Visual Studio:

1. Створити новий проєкт, натиснувши посилання «Новий проєкт» на початковій сторінці. Відкриється діалогове вікно «Новий проєкт».
2. Вибрати «Веб» на лівій панелі та «Вебпрограма ASP.NET» на середній панелі, а потім натиснути «ОК».
3. У діалоговому вікні натиснути «Новий проєкт ASP.NET» і обрати «Порожній шаблон», натиснути «ОК». Це створить порожній проєкт вебсайту у Visual Studio.
4. Встановити бібліотеку Angular з менеджера пакетів NuGet. Натиснути правою кнопкою миші на проєкті в «Solution Explorer» і вибрати «Manage NuGet Packages».
5. Знайти «angular» у діалоговому вікні «Керування пакетами NuGet» і встановити Angular Core. Це додасть файли Angular до папки Scripts, наприклад angular.js, angular.min.js і angular-mocks.js. З цього моменту можна почати писати вебдодаток Angular [64].

Angular – хороший фреймворк для розробки високоякісних динамічних вебдодатків. Це пов’язано з тим, що Angular має багато функцій, тому не потрібно покладатися на будь-яке стороннє програмне забезпечення для підтримки програм. Також можна заощадити багато часу та ресурсів під час роботи над проєктами за допомогою Angular.

2.4. Засоби розроблення інформаційного забезпечення Веб-інформаційної системи з використанням SQL

База даних (БД) – це організований набір даних або інформації, які спеціально зібрані для швидкого пошуку за допомогою комп’ютера. Вона полегшує зберігання, пошук, модифікацію та видалення даних у поєднанні з різними операціями обробки даних [35].

Різні типи баз даних відрізняються за схемою, структурою даних і типами даних, які їм найбільше підходять. Однак усі вони складаються з однакових п’яти основних компонентів:

1) обладнання – фізичний пристрій, на якому працює програмне забезпечення БД. Апаратне забезпечення включає комп’ютери, сервери та жорсткі диски;
2) програмне забезпечення або додаток – надає користувачам можливість керувати базою даних. ПЗ системи управління базами даних (СУБД) використовується для керування та контролю баз даних;
3) дані – необроблена інформація, яку зберігає база даних;
4) мова доступу – мова програмування, яка керує базою даних. Однією з найпоширеніших мов баз даних є SQL;
5) процедури – правила, які визначають, як працює база даних і як вона обробляє дані [14].

База даних зазвичай контролюється системою керування базами даних (СУБД). Разом дані та СУБД із додатками, які з ними пов’язані, називають системою баз даних, яку часто скорочують до просто бази даних. ПЗ для баз даних полегшує управління інформацією, дозволяючи користувачам зберігати дані в чіткій та послідовній формі, а у відповідь мати до них доступ.

Нині існує велика кількість типів баз даних. Під час вибору найкращої з них варто орієнтуватися на те, як її збираються використовувати. Поширені типи баз даних:

1) реляційні – елементи організовані як набір таблиць зі стовпцями та рядками;
2) об'єктноорієнтовані – інформація зображається у вигляді об'єктів, як в об'єктноорієнтованому програмуванні;
3) розподілені – має два або більше файли, що містяться на різних сайтах; може бути збережений на декількох комп’ютерах;
4) сховища даних – типи БД, які виконують оперативний запит і аналіз;

5) NoSQL (нереляційна) – дає змогу зберігати та маніпулювати неструктурованими та напівструктурованими даними (на відміну від реляційної БД, яка вказує, як всі дані складатимуться);
6) графові БД – зберігає дані в термінах сутностей і їхніх зв'язків;
7) БД із відкритим кодом – система, вихідний код якої є відкритим; такі БД можуть бути базами даних SQL або NoSQL;
8) хмарні БД – перелік даних, які знаходяться на платформі хмарних обчислень;
9) мультимодельна БД – поєднує різні типи моделей БД в одну інтегровану серверну частину;
10) БД документів / JSON – розроблені для зберігання, отримання та керування документно-орієнтованою інформацією;
11) самокеровані – базуються на хмарі та використовують машинне навчання для автоматизації налаштування БД, безпеки, резервного копіювання, оновлення та інших рутинних завдань керування [5].

Дані в найпоширеніших типах БД, що працюють нині, як правило, моделюються в рядках і стовпцях у серії таблиць, щоб зробити обробку та запити даних ефективними. Це дає змогу мати доступ до даних, керувати ними, зберігати, змінювати, оновлювати, контролювати та знищувати. Переважна більшість БД базуються мовою структурованих запитів (SQL), щоб записувати та запитувати дані [39].

SQL – це мова програмування, яка реляційні БД використовують для запитів, визначення та оброблення даних, а також для забезпечення контролю доступу. SQL вперше було розроблено в IBM у 1970-х роках за участю Oracle як основного учасника, що призвело до впровадження стандарту SQL ANSI. Нині SQL досить широко використовується, проте вже починають з'являтися нові мови програмування [14].

Під час виконання команди SQL у будь-якій системі керування реляційною базою даних, система автоматично знаходить найкращу процедуру для

виконання запиту, а механізм SQL визначає, як інтерпретувати цю конкретну команду.

Мова структурованих запитів містить у своєму процесі такі чотири компоненти: диспетчер запитів, двигуни оптимізації, класична система запитів, механізм запитів SQL.

Класичний механізм запитів дозволяє фахівцям з обробки даних і користувачам обслуговувати запити, відмінні від SQL.

SQL має ряд переваг, які роблять його більш популярним у сфері обробки даних. Це ідеальна мова запитів, яка дозволяє спеціалістам з обробки даних і користувачам спілкуватися з базою даних. Основні переваги SQL:

1) SQL не вимагає вмінь програмування для керування СУБД;
2) високошвидкісна обробка запитів;
3) стандартизована мова;
4) портативність, можливість використання з іншими програмами та пристроями;
5) інтерактивна та проста мова;
6) можливість створення декілька переглядів структури БД для різних користувачів [18].

Механізми запитів SQL-on-Hadoop — це новітнє відгалуження SQL, яке дає змогу організаціям з архітектурою великих даних, побудованою на сховищах даних Hadoop, використовувати SQL як мову запитів, а фахівцям із баз даних використовувати звичну мову запитів замість використання складніших і менш знайомих мов – зокрема, середовище програмування MapReduce для розробки програм пакетної обробки [24].

ВИСНОВКИ ДО РОЗДІЛУ 2

1. Проаналізовано проблеми підтримки прийняття рішень в готельному бізнесі та виявлено необхідність розроблення Веб-інформаційної системи, яка містить засоби моделювання обсягів замовлень на готельні послуги.
2. Проаналізовано засоби розроблення Веб-інформаційних систем та їх інформаційного забезпечення. Встановлено, що фреймворк Angular дозволяє розробити високопродуктивні інформаційні системи, а СУБД SQL надає можливість розробити гнучке інформаційне забезпечення.
3. У даному розділі досліджена реалізація веборієнтованої інформаційної системи. Розглянуті сучасні системи та технології мають широкий функціонал, що дозволяє швидко та ефективно враховувати всі нюанси створення Веб-інформаційної системи готельного бізнесу.
4. Визначено основні засоби реалізації інформаційного забезпечення Веб-інформаційної системи з використанням міжплатформної структури ASP.NET Core, авторизації на основі техніки JSON Web Token (JWT), фреймворку Angular, системи управління базами даних, а також мовою структурованих запитів SQL.

РОЗДІЛ 3. СТВОРЕННЯ ВЕБ-ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1. Розробка макета інтерфейсу системи

Створення Веб-інформаційної системи розпочинається з етапу дослідження. Він передбачає розробку концепції, яка містить дані про те, хто буде використовувати систему, які можливості матиме кожен із користувачів, а також те, на кого буде орієнтована система.

Розробка макета потребує детальний збір інформації про компанію та сайт замовника, його конкурентів, статистику використання поточного інтерфейсу, цільову аудиторію та її поведінку тощо. Цей етап дає змогу зрозуміти, хто використовуватиме інтерфейс, які обмеження або переваги перед конкурентами можуть виникнути у процесі розроблення системи.

Наступний етап – визначення варіантів використання інтерфейсу. Система відрізняється для різних користувачів у залежності від їхньої Ролі. Користувачем системи може бути як Клієнт, який вирішив забронювати готельний номер, так і Реєстратор, який займається внутрішнім наповненням і організацією роботи цієї системи.

Ролі для користувачів:

1. Клієнт – користувач, що може бачити тільки “зовнішню” частину системи, має можливість бронювання та реєстрації в системі, внаслідок цього отримує роль Користувача;
2. Реєстратор – користувач, що може бачити панель адміністратора (він бачить її у ролі звичайного Реєстратора), реєструвати клієнта в системі, переглядати інформацію про нього, змінювати умови проживання тощо;
3. Власник – це власник бізнесу, який має повний доступ до системи та може змінювати перелік послуг, вартість та умови проживання.

Представлення інформаційної системи починають з побудови UML (Unified Modeling Language) діаграм [40]. На основі опису роботи інтерфейсу створюється перелік завдань (use cases), які користувач може виконувати в межах інтерфейсу системи. Use case – це діаграма, що зображає відношення між акторами які мають певну роль в системі та прецедентами в системі наведено на рис. 3.1 [66].

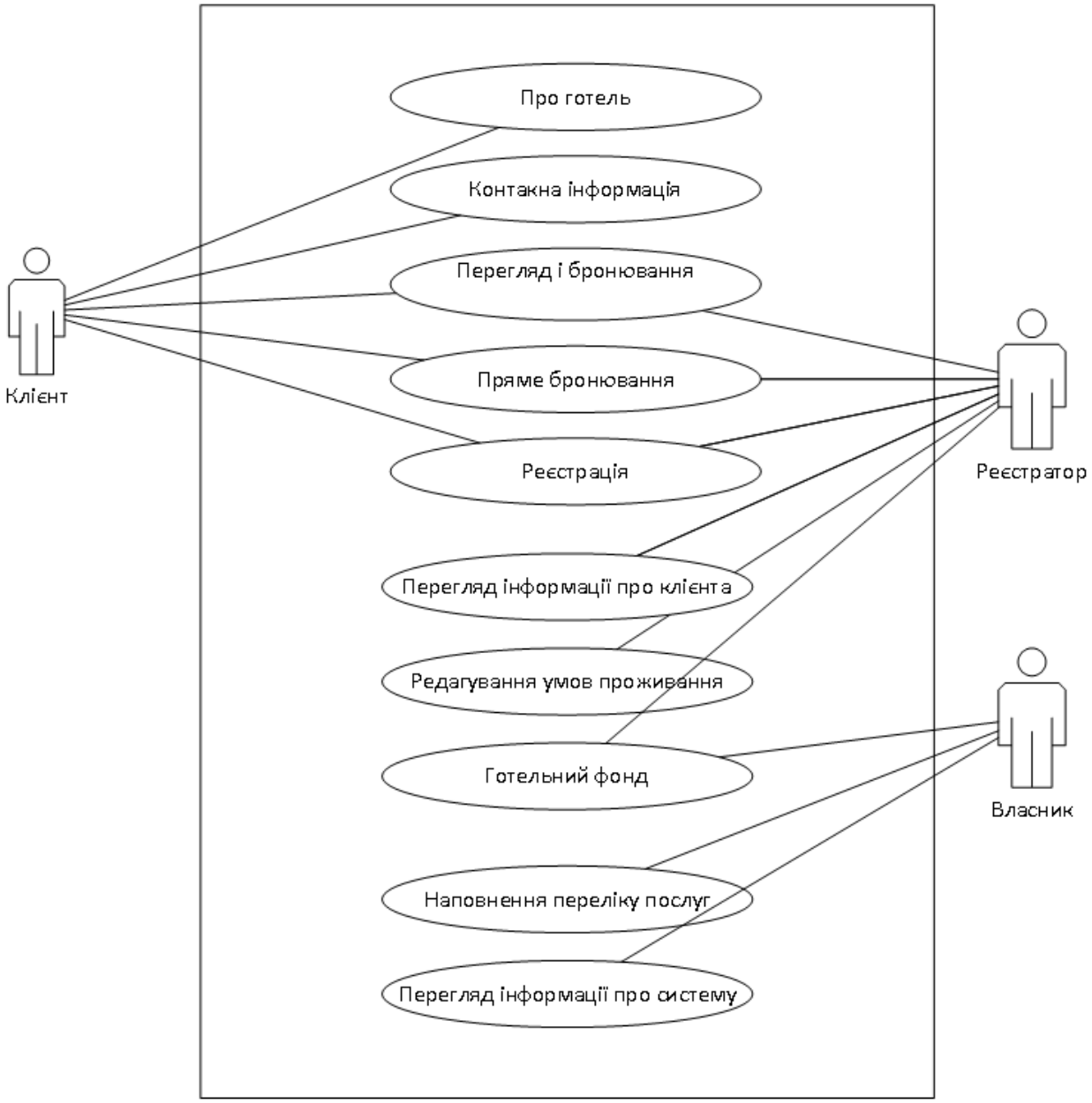

Рис. 3.1. Діаграма використання системи підтримки готельного бізнесу

Користувач цієї системи може виконувати такі дії:

1) зайти на сайт;

2) авторизуватися;
3) перейти у власний профіль;
4) вносити, редагувати власні дані у профілі;
5) обирати, закладати варіанти для бронювання;
6) власне бронювати готельний номер, оплачувати його онлайн;
7) обирати додаткові послуги готелю;
8) вносити корективи, скасовувати бронювання.

Процес реєстрації користувача у системі відбувається за стандартною схемою. Клієнт отримує власний кабінет з інформацією про доступні варіанти бронювання та інші функції, що згадані вище. Якщо клієнт вже раніше реєструвався і користувався послугами цієї системи, то йому доступне меню із переліком його попередніх бронювань.

Водночас ролі Реєстратора відрізняються від ролей Користувача. Він може бачити інформацію про надійність клієнта (на основі його попередніх дій, пов'язаних із бронюванням). Реєстратор також може здійснювати підтвердження бронювання як у ручному, так і в автоматичному режимі. Реєстратор може вибрати із кількох клієнтів того, кому підтвердити бронювання, а для інших написати, що бронювання скасоване і назвати причину, наприклад, “немає місць, але ми повідомимо, коли з'являться доступні” або пропозиція обрати інші дати.

Важливим елементом системи є можливість бронювання готельних номерів у реальному часі. Бронювання також залежить від дати, а інтерфейс максимально зручний для усіх користувачів системи.

Власник готельного бізнесу може переглядати повну статистику того, скільки зайнято номерів на певний період, а також мати повний доступ до системи, як в Реєстратора.

Наступні етапи розробки інтерфейсу системи полягають у створенні самої структури інтерфейсу, створення прототипів, у яких плануються особливості розташування елементів сторінок відносно один одного, але не зовнішній вигляд. Далі визначається стилістика майбутнього інтерфейсу, пропонуються різні

варіанти, один яз яких ляже в основу концептуального дизайну. Після цього створюється дизайн-концепція, яка зображає дизайн сайту та робить зрозумілим майбутнє бачення всього сайту та системи. DFD (Data Flow Diagram). Ця діаграма розширена use case діаграма в ній графічно представлені «потоки» даних в інформаційній системі, які наведено на рис. 3.2.

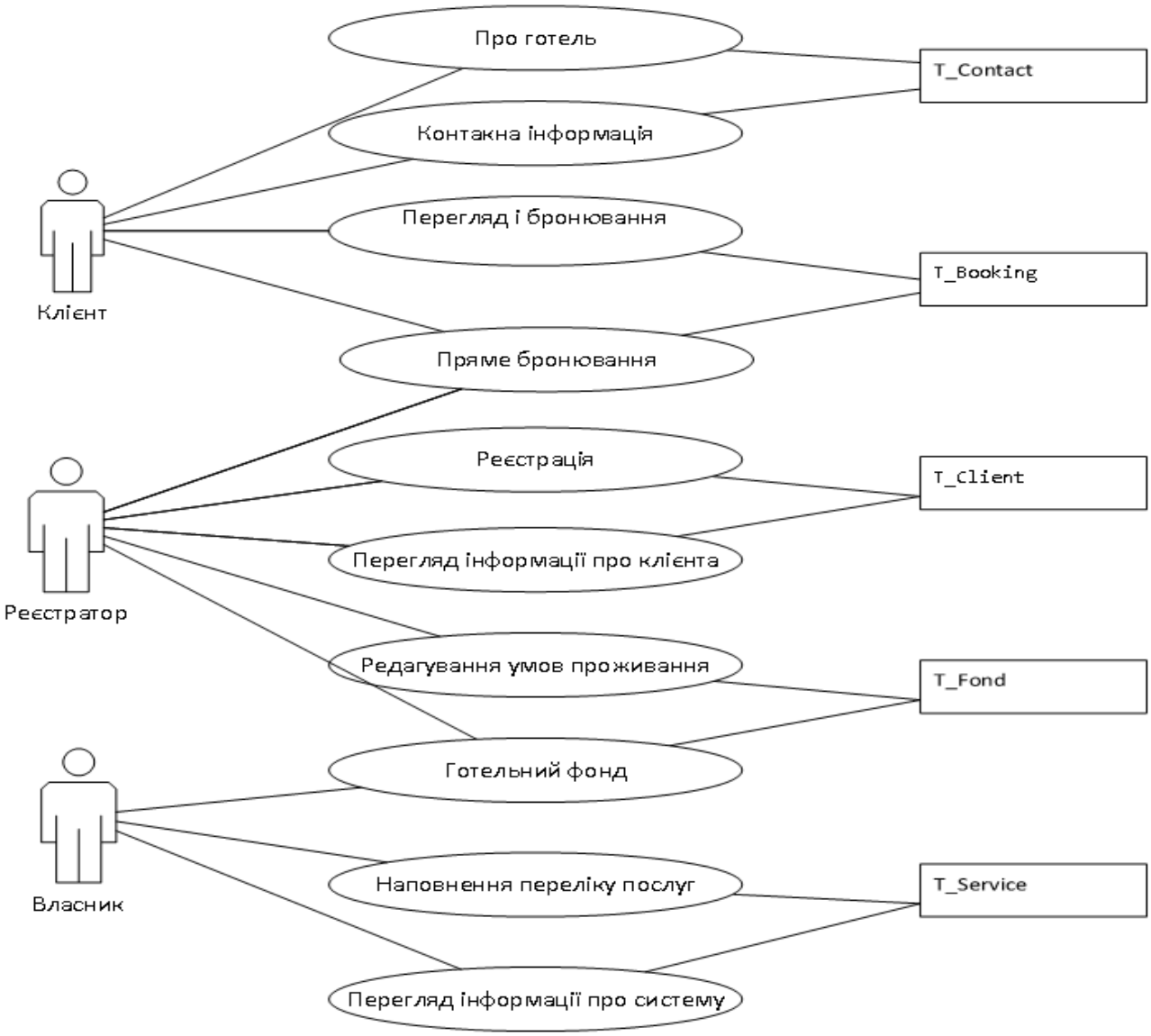

Рис. 3.2. Діаграма використання системи підтримки готельного бізнесу з графічним представленням «потоків» даних

Після DFD-діаграми будується ER-модель (Entity-relationship model), яка зазвичай реалізується у вигляді баз даних. У випадку застосування реляційної

бази даних, в якій зберігають дані в таблицях, кожен рядок цих таблиць є одним екземпляром сутності ER-діаграми як моделі даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків, які наведено на рис. 3.3.

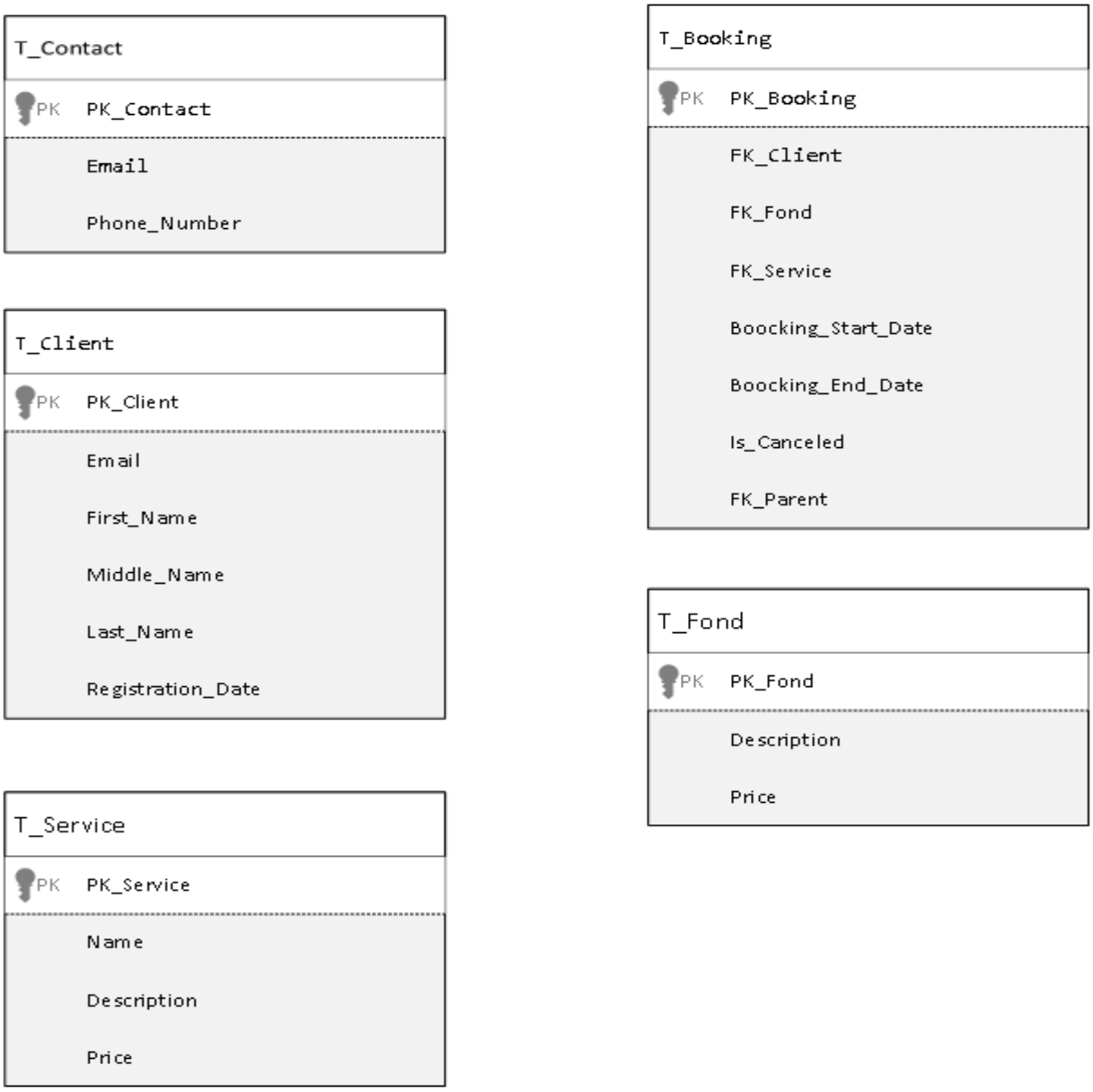

Рис. 3.3. Діаграма представлення «потоків» даних в системі підтримки готельного бізнесу

Після підтвердження замовником макета Веб-інформаційної системи можна розробляти архітектуру сайту.

«Відповідно до стандарту ANSI/IEEE 1471–2000, загальноприйнятим є таке визначення архітектури системи – це опис організації системи в термінах компонентів, їх взаємозв'язків і з довкіллям, принципи управління їх розробкою і розвитком. Архітектуру ІС можна описати як концепцію, яка визначає модель, структуру, виконувані функції та взаємозв'язок компонентів» [37, С. 85].

Для ефективного проєктування програмного забезпечення краще використовувати багаторівневу архітектуру. Трирівнева модель процесу допомагає працювати ефективно на початковому етапі. Крім цього, якщо в бізнес-логіці відбувається якась зміна, це можна легко зробити, внісши відповідну зміну на певному рівні. Наприклад, зміна на рівні бізнес-логіки не вплине на інші два рівні, а саме на рівні доступу до даних та інтерфейсу користувача. Схему трирівневої архітектури наведено на рис. 3.4.

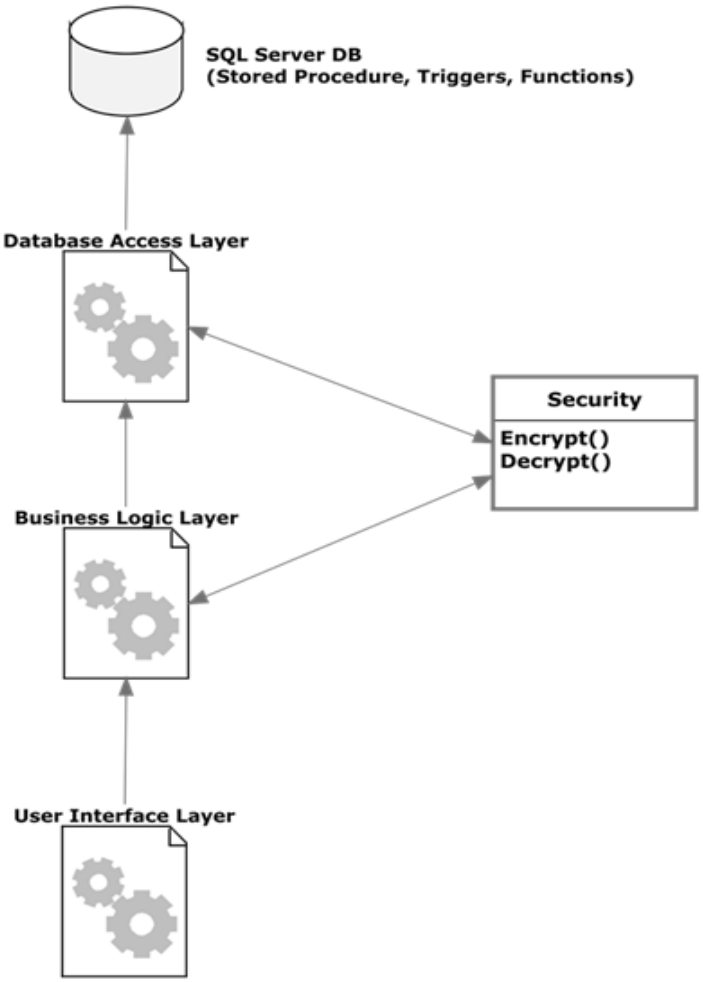

Рис. 3.4. Схема трирівневої архітектури

До трирівневої архітектури входять такі компоненти:

1) DAL;
2) BLL;
3) UI Layer.

DAL або рівень доступу до даних, безпосередньо взаємодіє з базою даних за допомогою SQL. Цей рівень є єдиним зв'язком між сайтом і цими даними. Тут використовуються рядки з'єднання і тут пишуться оператори для зв'язку з базою даних.

Водночас у BLL або рівні бізнес-логіки, можна написати логіку про те, що робити з даними і як взаємодіяти з рівнем DAL. Це розділення шарів зроблено тому, що якщо потрібно внести зміни в один шар, це можна зробити відносно легко, навіть не торкаючись інших шарів. Також це допомагає у багаторазовому використанні.

UI Layer – це рівень інтерфейсу користувача. Цей рівень містить вебформи та їх елементи управління для вебдодатків.

Переваги використання трирівневої архітектури у Веб-інформаційній системі підтримки готельного бізнесу:

1) масштабованість: Сервери можуть бути розгорнуті на багатьох машинах, а база даних більше не вимагає підключення від кожного клієнта;
2) багаторазовість використання: користувачі можуть повторно використовувати рівні BLL та DAL з іншим проєктом;
3) покращити цілісність даних: BLL, тобто середній рівень, може гарантувати, що в базу даних дозволяється вставляти, оновлювати або видаляти лише дійсні дані;
4) підвищення безпеки: Оскільки клієнт не має прямого доступу до бази даних, BLL є безпечнішим, адже розміщений на більш захищеному центральному сервері;

5) зменшення розповсюдження: Зміни в BLL / DLL повинні оновлюватися тільки на сервері сайту і не повинні турбувати всіх клієнтів;
6) фактична структура бази даних прихована від звичайного користувача.

Водночас трирівнева архітектура може трохи ускладнювати роботу системи, зокрема через збільшення складності та зусиль розробника, адже вона є складнішою для побудови в порівнянні з 2-рівневою архітектурою, проте її переваги очевидні для використання і якісної роботи системи бронювання номерів у готельному бізнесі.

В архітектурному компоненті було розділено Angular та Web API в різні проєкти, тому що надалі можна зробити мобільний додаток і застосувати у ньому цю Веб-інформаційну систему.

3.2. Диспетчер інформаційної системи. Імплементація Web API .NET Core 6

Для зв'язку проєктів використовуються контролери, які прив'язуються до певної кінцевої точки. Автентифікація та авторизація допомагає зв’язати два додатки Angular та Web API. Автентифікація підтверджує, що Користувач є тим, за кого він себе видає, на основі наданих облікових даних. Водночас авторизація дає дозвіл Користувачу на виконання дій у системі на основі наданих нами повноважень.

В ASP.NET Core процес автентифікації виконує служба IAuthenticationService, яка використовується проміжним програмним забезпеченням аутентифікації. Ця служба використовує зареєстровані обробники автентифікації та виконує дії, пов'язані з автентифікацією.

Схема автентифікації – це ім'я, яке відповідає обробці автентифікації і містить параметри для налаштування цього конкретного екземпляра обробника. Схема автентифікації може вибрати, який обробник автентифікації відповідає за генерацію правильного набору вимог. Під час налаштування автентифікації була

вказана схема за замовчуванням, адже система не запитувала певну схему. Однак надалі можна налаштувати різні схеми за замовчуванням, які будуть використовуватися для аутентифікації, оскарження і заборони дій, а також об'єднати кілька схем в одну за допомогою схем політики.

Убезпечити мінімальні кінцеві точки API в ASP.NET Core 6 за допомогою JSON-токенів для аутентифікації та авторизації нескладно. ASP.NET Core 6 представляє спрощену модель хостингу, яка дає змогу створювати полегшені API з мінімальними залежностями. Як правило, потрібно захищати кінцеві точки таких API у системі.

Для захисту необхідний мінімальний API з використанням аутентифікації JWT. В цій версії API потрібно самостійно реалізовувати методи для автентифікації користувача в системі.

Водночас процес авторизації є незалежним від аутентифікації, проте авторизація потребує наявності механізму автентифікації. Автентифікація встановлює особу користувача і може створювати одну або декілька ідентифікацій для поточного користувача.

Компоненти авторизації в ASP.NET Core, включаючи атрибути AuthorizeAttribute та AllowAnonymousAttribute, знаходяться в просторі імен Microsoft.AspNetCore.Authorization.

У Веб-інформаційній системі готельного бізнесу дані усіх користувачів захищені авторизацією. Система зображає список контактів, які створили авторизовані (зареєстровані) користувачі. На цьому етапі існує три групи безпеки:

1) зареєстровані користувачі можуть переглядати всі підтверджені власні дані, а також редагувати/видаляти їх;
2) персонал готелю (Реєстратор) може затверджувати або відхиляти дані контактів. Користувачі можуть бачити тільки затверджені контакти;
3) Реєстратор також може затверджувати/відхиляти та редагувати/видаляти будь-які дані.

Наприклад, Користувач увійшов в систему. Він може переглядати лише затверджені контакти та редагувати/видаляти/створювати нові посилання для своїх контактів. Тільки для останнього запису, створеного Користувачем, показуються посилання “Редагувати” та “Видалити”. Інші користувачі не мають можливості побачити останній запис, поки Реєстратор не змінить його статус на “Затверджено”.

Кнопки “Затвердити” і “Відхилити” відображаються тільки для Реєстраторів готелю. Фактично Реєстратор має всі привілеї, адже він може переглядати, редагувати та видаляти будь-який контакт і змінювати статус контактів у системі.

JSON Web Token (JWT) – це закодоване в JSON представлення вимоги (вимог), яке може передаватися між двома сторонами. Вимога підписується цифровим підписом емітента токена, і сторона, яка отримує цей токен, може пізніше використовувати цей цифровий підпис для підтвердження права власності на вимогу. Access-токен має певний expired-date, тобто час, до якого цей токен має право звертатися до ресурсів.

Реалізація та налаштування проєкту та генерації маркера JWT:

1. Створити додаток ASP.NET Core 6 Web API. У програмі вказати назву «JWTTokenPOC».
2. Встановити «Microsoft.AspNetCore.Authentication.JwtBearer» за допомогою менеджера пакетів NuGet.
3. Створити нову папку «Entities» усередині рішення та додати клас сутності «User».
4. Створити нову папку Models всередині рішення та додати дві моделі AuthenticateRequest і AuthenticateResponse.
5. Створити нову папку «Helper» усередині рішення та додати два допоміжні класи «AppSettings» і «AuthorizeAttribute» у цій папці.
6. Додати налаштування програми у файл appsettings.json.

7. Створити нову папку «Service» усередині рішення та додати два класи обслуговування «AuthenticationService» і «UserService» у цій папці.
8. У папці Helper створити клас JwtMiddleware. Цей клас використовуватиметься для перевірки маркера, і він буде зареєстрований як проміжне програмне забезпечення.
9. Відкрити файл statrtup.cs. У методі ConfigureServices додати політику CORS і додати необхідні служби.
10. У методі Configure встановити політику CORS і зареєструвати проміжне програмне забезпечення JWT.
11. Створити й запустити додаток.
12. Відкрити інструмент Postman і згенерувати маркер JWT, як показано нижче:
    1) натиснути піктограму «Новий» і створити новий запит Http. Вказати назву запиту Http як «AuthToken»;
    2) змінити метод http-запиту на "GET" за допомогою спадного селектора ліворуч від поля введення URL-адреси;
    3) у полі URL-адреси ввести адресу маршруту локального API *http://localhost:60119/Authenticate/authenticate*;
    4) вибрати вкладку «Тіло» під полем URL-адреси, змінити тип перемикача «Тіп тіла» на «необроблений» і змінити розкривний селектор формату на «JSON (програма/json)»;
    5) ввести об’єкт JSON, що містить тестове ім’я користувача та пароль у тексті «Тіло».
13. Натиснути кнопку «Надіслати». Після цього з’явиться відповідь «200 OK» із деталями користувача, включаючи маркер JWT у тілі відповіді. Необхідно зробити копію значення маркера, оскільки він буде використовуватися на наступному кроці, щоб створити автентифікований запит.

14. Дослідити розділ тіла, де є атрибут «токен». Це маркер JWT, який отримали, і цей маркер використовуватиметься для підтвердження користувача та отримання даних користувача з контролера користувача.
15. Щоб зробити автентифікований запит за допомогою маркера JWT з попереднього кроку, потрібно виконати такі дії:
    1) відкрити вкладку нового запиту, натиснувши кнопку плюс (+) у кінці вкладок;
    2) змінити метод http-запиту на "GET" за допомогою спадного селектора ліворуч від поля введення URL-адреси;
    3) у полі URL-адреси введіть адресу маршруту користувача локального API http://localhost:60119/Users/GetUser;
    4) вибрати вкладку «Авторизація» під полем URL-адреси, змінити тип на «Маркер носія» у спадному списку типу та вставити маркер JWT із попереднього кроку автентифікації в поле «Маркер»;
    5) натиснути кнопку «Надіслати». Після цього з’явиться відповідь «200 OK», що містить масив JSON з усіма записами користувача в системі.

Як результат, програма має два контролери «AthenticateController» і «UserController». «AthenticateController» має три кінцеві точки, яка автентифікує користувача на основі ідентифікатора користувача та пароля генерує маркер JWT, наступна генерує новий маркер JWT коли попередній вже не дійсний та контролер, що повідомляє системі, що користувач вийшов із системи і що маркери JWT вже не дійсні. «UserController» має дві кінцеві точку, яка на основі маркера JWT повертає інформацію про користувача.

Реалізований фільтр авторизації для захисту кінцевої точки, і ця кінцева точка приймає запити HTTP GET і повертає список усіх користувачів у програмі, якщо заголовок авторизації HTTP містить дійсний маркер JWT. Якщо маркер автентифікації відсутній або маркер недійсний, повертається відповідь 401 Unauthorized.

Подібним чином GetUserById повертає відомості про користувача за ідентифікатором, якщо заголовок авторизації HTTP містить дійсний маркер JWT [52].

Модель запиту автентифікації визначає параметри вхідних запитів до маршруту тобто кінцевої точки. Вона приєднується до маршруту як параметр методу дії контролера користувачів для автентифікації. Під час отримання маршрутом HTTP POST запиту дані з тіла прив'язуються до екземпляра класу Request, валідуються та передаються в метод.

Після успішної аутентифікації метод генерує токен, що підписується цифровим підписом за допомогою секретного ключа. Це допомагає захистити маршрути за допомогою політик авторизації, а також змушує надавати інформацію про автентифікацію при виклику цієї кінцевої точки. Проміжне програмне забезпечення авторизації буде використовувати цю інформацію для перевірки запиту для поточного контексту виконання. Текст програмного модуля TokenService.cs винесено в Додаток А.

Сервіс користувача містить методи для перевірки облікових даних користувача та повернення JWT-токену, отримання всіх користувачів у додатку та отримання одного користувача за ідентифікатором. Токен JWT повертається до клієнтського додатка, який додає його в HTTP-заголовок авторизації наступних запитів для захисту маршрутів.

3.3. Інтеграція модуля прогнозування в середовищі Python

Характеристика розробки програмного забезпечення системи неможлива без опису використаних технологій та інструментів.

Інформаційна система створювалася мовою програмою Python – об'єктноорієнтованою мовою високого рівня, що має динамічну семантику, а також велику кількість вбудованих структур, що робить цю мову досить привабливою для розробки системи. Крім цього, Python може підтримувати різні

модулі, пакети та бібліотеки, завдяки чому модульність Веб-інформаційної системи зазнає спрощення та оптимізації.

Python має простий синтаксис, що робить його зручним для перегляду, а також це знижує кількість витрат на обслуговування програми. Крім цього, Python популярний в аналізі даних, штучному інтелекті та загалом у машинному навчанні. Ця мова є логічним вибором, тому що Веб-інформаційна система розроблена на основі машинного навчання.

Часто машинне навчання виконує завдання контрольованого навчання, яке містить вхідні дані та відомі вихідні дані. Основною метою є використання набору даних для навчання моделі, що може прогнозувати правдиві результати на основі вхідних даних. У результаті поєднання алгоритму машинного навчання та певного набору навчальних даних створюється модель, за допомогою якої надалі можна робити прогнози для нових даних.

У машинному навчанні та статистичному моделюванні взаємозв'язок між змінними використовується для прогнозування результатів майбутніх подій. Лінійна регресія використовує взаємозв'язок між точками даних, щоб провести через них пряму лінію. Ця лінія може бути використана для прогнозування майбутніх значень.

У мові Python є методи для пошуку взаємозв'язку між точками даних і для побудови лінії лінійної регресії, яка є одним із базових методів статистики та машинного навчання.

«Лінійна регресія – це загальновідомий статистичний метод, який був прийнятий в машинному навчанні досить давно, доповнений багатьма вдосконаленнями для підгонки результатів і вимірювання помилок. Даний метод, як правило, добре працює на великих і розріджених наборах даних, що не мають складних трендів» [29].

Лінійна регресія використовує взаємозв'язок між точками даних, щоб провести через них пряму лінію. Надалі ця лінія використовується для прогнозування майбутніх значень. Водночас мова Python має методи, що

допомагають знаходити зв'язки між точками даних, а також створювати лінії лінійної регресії. Зазвичай під час регресійного прогнозування аналізують деяке явище з рядом спостережень. Кожне спостереження має дві або більше ознак.

Для Веб-інформаційної системи лінійна регресія є доволі корисною тому, що завдяки їй можна спрогнозувати відповідь, використовуючи новий набір предикторів. У нашому випадку – спрогнозувати можливу кількість відмов від бронювання, а також заповненість готельних номерів залежно від пори року, популярності місцевості, кількості номерів тощо.

Лінійна регресія, ймовірно, одна з найбільш важливих і широко використовуваних регресійних методів. Це один з найпростіших методів регресії. Регресія шукає взаємозв'язки між змінними. Однією з її головних переваг є простота інтерпретації результатів.

Як правило, входи позначаються через x, а результати через y. Якщо є дві або більше незалежних змінних, то їх можна представити у вигляді вектора $x = (x_1, \ldots, x_r)$, де r – кількість вхідних даних.

Під час реалізації лінійної регресії деякої залежної змінної на множині незалежних змінних $x = (x_1, \ldots, x_r)$, де r – число предикторів припускають лінійну залежність між y і x: $y = \beta_1 x_1 + \beta_1 x_1 + \beta_r x_r + \varepsilon$. Це – рівняння регресії. $\beta_0, \beta_1, \ldots, \beta_\text{r}$ - коефіцієнти регресії, а ε - випадкова помилка.

Лінійна регресія обчислює оцінки коефіцієнтів регресії або просто передбачені ваги, позначені $b_0, b_1, \ldots, b_\text{r}$. Вони визначають оцінну функцію регресії $x\,(x) = b_1 x_1 + b_1 x_1 + \cdots + b_\text{r} x_\text{r}$. Ця функція повинна достатньо добре зображати залежності між входами та виходами.

Оцінена або прогнозована відповідь $f\,(x_\text{i})$ для кожного спостереження $1 = 1, \ldots, n$ повинна бути якомога ближче до відповідної фактичної відповіді. Відмінності $f - f\,(x_\text{i})$ для всіх спостережень $1 = 1, \ldots, n$ називають залишками. Регресія стосується визначення найкращих прогнозованих ваг, тобто ваг, що відповідають найменшим залишкам.

Щоб отримати кращі ваги, потрібно мінімізувати суму квадратних залишків (SSR) для всіх спостережень $= 1, \dots, n: SSR = \Sigma_i (y_i - f(x_i))^2$. Такий підхід називають методом звичайних найменших квадратів.

Зміна фактичних відповідей $y_i, 1 = 1, \dots, n$ відбувається частково шляхом залежності від предикторів. Однак існує також додаткова дисперсія виходу.

Коефіцієнт детермінації, позначений як R^2 вказує, яку величину варіації в можна пояснити залежністю від x за допомогою конкретної регресійної моделі. Більший indicates² вказує на покращене пристосування і означає, що модель може краще пояснити зміну виходу з різними входами.

Значення $R^2 = 1$ відповідає $SSR = 0$, тобто ідеально підходить, оскільки значення передбачених і дійсних відповідей повністю вписуються один в одного.

Проста або одноваріантна лінійна регресія є найпростішим випадком лінійної регресії з однією незалежною змінною $x = x$.

Наступний малюнок ілюструє просту лінійну регресію:

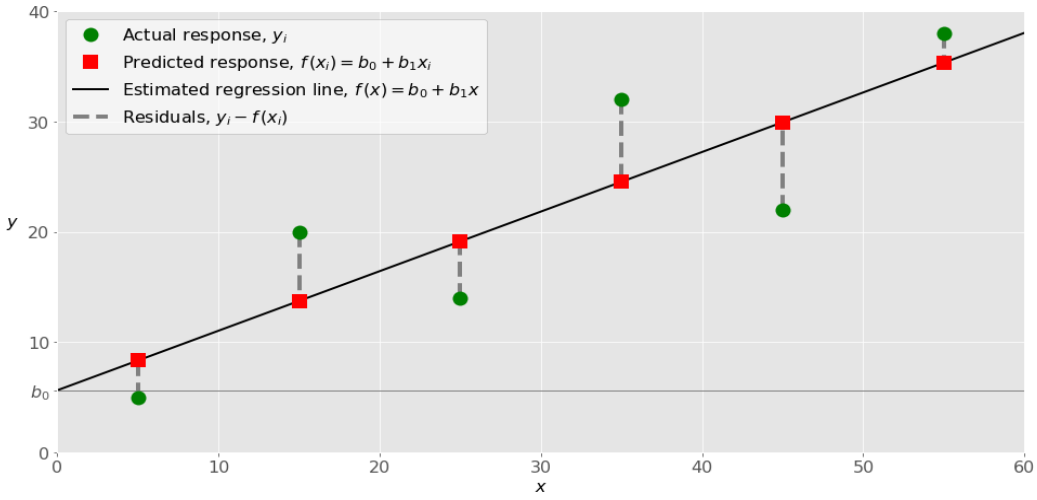

Рис. 3.5. Проста лінійна регресія

Реалізацію простої лінійної регресії починають із заданого набору парних входів-виходів $(y - y)$ (зелені кола). Ці пари є спостереженнями. Наприклад, ліве спостереження (зелене коло) має вхід $x = 5$ і фактичний вихід (відповідь) 5. Наступний має $x = 15$ і $20 = 20$, і так далі.

Оцінена функція регресії (чорна лінія) має рівняння $x\,(x) = b_0 + b_1 x$. Наша мета – розрахувати оптимальні значення прогнозованих ваг b_1 та b_1, які мінімізують РСБ та визначити оцінну функцію регресії. Значення b_0, яке називають перехопленням, показує точку, де розрахункова лінія регресії перетинає вісь y. Це значення оцінної відповіді $x\,(x)$ для $x = 0$. Значення b_1визначає нахил розрахункової лінії регресії.

Прогнозовані відповіді (червоні квадрати) - це точки на лінії регресії, які відповідають вхідним значенням. Наприклад, для входу 5 = 5 прогнозована відповідь $f\,(5) = 8{,}33$ (представлений з крайньою лівою червоною площею).

Залишки (вертикальні штрихові лінії) можна обчислити як $f - f\,(x_\text{i}) = b_0 - b_0 - b_1 x_\text{i}$ для $1 = 1, \ldots, n$. Це відстань між зеленими колами і червоними квадратами. Коли реалізують лінійну регресію, потрібно мінімізувати ці відстані і зробити червоні квадрати якомога ближче до зумовлених зелених кіл [68].

Реалізація лінійної регресії в середовищі Python розпочинається з отримання доступу до коду в іншому модулі шляхом його імпортування. Інструкція імпорту є найпоширенішим способом виклику механізму імпорту пакетів.

Оператор імпорту може виконувати дві операції: здійснювати пошук названого модуля, а після цього зв'язувати результати пошуку з іменем у локальній області видимості. Операція пошуку оператора імпорту визначається як виклик функції з відповідними аргументами. Внаслідок цього функцією повертаються значення, які надалі використовуються для того, щоб зв'язувати імена в операторі імпорту [69].

Python має один тип об'єкта модуля, і всі модулі мають цей тип, незалежно від того, чи модуль реалізовано мовою Python, чи чимось іншим. Для організації модулів та забезпечення ієрархії іменування у мові Python існує поняття пакетів.

Важливо, що всі пакети є модулями, проте не всі модулі можуть бути пакетами. Тобто пакети – це певний особливий вид модулів, тому будь-який модуль, що містить атрибут “path”, можна вважати пакетом. Як правило,

необхідно застосувати відповідні пакети та їхні функції та класи. Вміст файлу script.py. винесено в Додаток Б.

Для роботи програми потрібно встановити менеджер пакетів для Python, а також додаткові пакети, які використовує програма за допомогою менеджера пакетів. Рекомендується встановлювати пакунки Python інструментом PyPA – програмою, що використовується для інсталяції пакунків з індексу пакунків Python та інших індексів [63].

Пакет Pandas дозволяє читати дані з файлів, які мають розширення .csv. CSV – це файл який містить дані з таблиці через кому і кожен новий запис починається із нового рядка. Взаємодія Pandas із даними проявляється у вбудованих структурах даних для Python, текстових файлах або таблицях.

Pandas має дві головні структури даних:

1) Series – одновимірна структура, що може мати ряд значень, які впорядковані у певній послідовності;
2) DataFrame – двовимірна структура, яка подібна до матриць, але в її основі закладена велика кількість методів для дій із даними.

Ці структури даних мають здатність обробляти значну частину стандартних випадків використанні у різних сферах, зокрема науки, техніки, статистиці та фінансовій справі.

Користувачі DataFrame можуть використовувати всі ресурси data.frame, які має мова програмування R. Крім цього, Pandas має змогу забезпечити користувачам якісну інтеграцію в науково-обчислювальному середовищі, а також зв'язок зі сторонніми бібліотеками, адже він створений на основі NumPy.

Пакет NumPy – це базовий пакет Python, який дає змогу реалізовувати велику кількість високопродуктивних операцій на одно- і багатовимірних масивах. Крім цього, він забезпечує великою кількістю математичних процедур.

Пакет Scikit-learn – це доволі поширена бібліотека Python, що використовується для машинного навчання. Вона створена на основі NumPy та деяких інших пакетів. Scikit-learn дає можливість використовувати засоби для

попередньої обробки даних, зменшувати розмірність, реалізовувати регресію, виконувати процеси кластеризації та класифікації тощо.

Входи (регресори, x) і вихідні дані (предиктор, y) мають бути масивами (екземплярами класу numpy.ndarray) або подібними об'єктами.

Наступний крок – реалізація моделі лінійної регресії, а також підлаштовування відповідно до відомих даних. Отримання результатів відбувається після встановлення моделі. Таким чином можна перевірити, наскільки ця модель працює ефективно та коректно, а також чи можна інтерпретувати її.

Регресійні моделі часто використовуються для створення прогнозу. Для цього Це означає, що потрібно використовувати вбудовані моделі для обчислення результатів на основі деяких інших нових входів [68].

Для реалізації лінійної регресії був створений клас, що здійснює зчитування даних із файлу, а також перетворює та видаляє їх. Завантаження даних зручно здійснювати за допомогою пакета Pandas, що має функції зчитування даних із файлів різних форматів: data = pd.read_csv(file_path).

Наведений набір даних може здійснювати розкид значень параметрів, тому необхідно також використати min-max нормалізацію, щоб передбачити методи. Нормалізація min-max – це розповсюджений спосіб нормалізації даних. Кожна функція, що має мінімальне значення 0, має максимальне значення 1. Водночас кожне інше значення може мати десяткове число від 0 до 1. Нижче наводиться формула для нормалізації:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}},$$

(3.1)

де $xnorm$ – це нормалізоване значення, x – значення, яке потрібно нормалізувати, $xmin$ – мінімальне значення у наборі, $xmax$– максимальне значення у наборі. Методи для нормалізації даних наведено у Додатку В.

Зображений набір даних може містити строкові значення, однак лише параметр понаднормової роботи використовується для аналізу, адже він містить два унікальних строкових значення.

3.4. Модель інтерфейсу користувача

Інтерфейс користувача має дуже важливу роль у “життєвому циклі розробки програмного забезпечення”. Користувальницький інтерфейс створюється для того, щоб зробити взаємодію з користувачем максимально простою та ефективною, а також допомогти користувачу досягти своїх цілей, у нашому випадку – знайти та забронювати найкращий готельний номер, який відповідатиме усім вимогам користувача [15].

Користувачі завжди звертають увагу на зовнішню взаємодію і зазвичай не замислюються над тим, як ця система працює, яка її “внутрішня частина”. Однак вони повинні невимушено відчувати залученість, коли відвідують сайт готелю. Саме для цього важливий якісний користувацький інтерфейс.

Повний цикл розробки інтерфейсу системи містить такі етапи:

1) дослідження;
2) варіанти використання;
3) структура інтерфейсу;
4) прототипування інтерфейсу;
5) визначення стилістики;
6) концепція дизайну;
7) проєктування всіх екранів;
8) анімація інтерфейсу;
9) підготовка матеріалів для розробників.

Найважливішим є етап проєктування структури інтерфейсу, тому його буде описано більш детально. Створення інтерфейсу для Веб-інформаційної системи попередньо потребує створення вікна, що дасть можливість вводити

необхідні параметри для системи. Це вікно міститиме основні елементи, завдяки яким Користувач вводитиме свої дані. Важливим є ряд таких даних: період проживання, класифікація готельного номера (від стандарту до люксу), необхідна кількість місць у готельному номері, додаткові послуги, наявність дітей або тварин тощо.

Аналіз значень параметрів дає змогу виокремити групи та використати відповідні графічні віджети. Наприклад, кількість місць або період проживання містять виключно цифрові значення, тому для цього краще застосовувати віджет spinBox, що використовується саме для таких даних. Для іншої інформації, наприклад, додаткові послуги доцільніше використовувати менші групи перемикачів, а саме набори віджетів radioButton, які об'єднуються елементом groupBox. Параметри, де можна вибрати лише два значення – так або ні, – можуть бути використані для даних про наявність дітей або тварин. Для такого параметра доцільніше використовувати віджет checkBox.

Важливим елементом інтерфейсу користувача також є кнопка, що дає змогу генерувати рекомендації на основі попередньо введених параметрів. Це важлива частина, адже за її допомоги Користувач шукає та оцінює пропозиції, тому вони мають якомога точніше відповідати усім його запитам.

Крім цього, має значення розташування кнопок на вікні, адже якщо вони розміщені хаотично і нелогічно, це негативно впливає на досвід Користувача і відштовхує його від бронювання готельного номера. Ефективним розв'язанням цієї проблеми є розташування віджетів для введення параметрів симетрично у дві колонки. Це дозволяє оперативно знайти необхідний параметр і зручно використовувати його. Текст програмного модуля booking.component.ts представлено у Додатку Г.

Як результат, побудова зручного інтерфейсу користувача є необхідним етапом реалізації Веб-інформаційної системи. Розглянуто співвідношення важливих елементів вводу і параметрів для вводу, що призначені для них, а також графічні віджети, що доцільні для представлення різних типів даних.

ВИСНОВКИ ДО РОЗДІЛУ 3

1. За допомогою фреймворків Node та Express здійснено програмну реалізацію елементів інформаційної системи підтримки готельного бізнесу, що дозволило реалізувати підтримку бронювання номерів та реєстрації перебування відвідувачів.
2. На основі використання програмного середовища Python здійснено програмну реалізацію регресійного аналізу обсягів замовлень послуг готельного бізнесу, яка інтегрована у Веб-інформаційну систему.
3. У розділі спроєктований макет інтерфейсу системи, визначені основні ролі для користувачів, структура та прототип інформаційної системи, а також побудовано відповідні діаграми. Визначено, що для ефективного проєктування ПЗ краще використовувати багаторівневу архітектуру.
4. Окреслено особливості використання автентифікації та авторизації, їхню роль у захисті даних користувачів, а також налаштування у системі з використанням відповідних кнопок.
5. Інтеграція модуля прогнозування реалізована за допомогою мови програмування Python і лінійної регресії. Для Веб-інформаційної системи лінійна регресія є доволі корисною тому, що завдяки їй можна спрогнозувати відповідь, використовуючи новий набір предикторів.
6. Виокремлено важливі складові зручної моделі інтерфейсу користувача. Показано співвідношення важливих елементів вводу і параметрів для вводу, що призначені для них, а також графічні віджети, що доцільні для представлення різних типів даних.

ВИСНОВКИ

У процесі написання представленої роботи була розроблена Веб-інформаційна система надійності бронювань готельних номерів. Ця система дає змогу прогнозувати реальний попит на готельні місця на основі статистичних масивів активності окремих агентів туристичного бізнесу.

Під час роботи було проаналізовано та виокремлено основні джерела перевіреної інформації про процеси овербукінгу на підприємствах готельного бізнесу.

Наступний етап полягав у розробці методів попередньої обробки інформації з об'єктом дослідження для спрощення реалізації моделі. Далі було здійснено експерименти із відбору вагомих пояснювальних змінних на основі реальних даних та порівняно ефективність запропонованих підходів.

У роботі також проаналізовано ефективність реалізації моделі надійності бронювань за допомогою методів множинної лінійної регресії та дерева рішень.

Експерименти на реальних даних засвідчили вищу ефективність поєднання методів попередньої обробки інформації на основі кореляційного аналізу та реалізації моделі надійності бронювань за допомогою дерев рішень. Значна максимальна похибка оцінки лояльності клієнта щодо здійсненого бронювання (24%) при врахуванні завантаженості великої кількості номерів може бути значно зменшена.

Визначено засоби розроблення Веб-інформаційних систем та їх інформаційного забезпечення. Встановлено, що фреймворк Angular дозволяє розробити високопродуктивні інформаційні системи, а СУБД SQL надає можливість розробити гнучке інформаційне забезпечення.

Розглянуті сучасні системи та технології мають широкий функціонал, що дозволяє швидко та ефективно враховувати всі нюанси створення Веб-інформаційної системи готельного бізнесу.

У роботі також визначено основні засоби реалізації інформаційного забезпечення Веб-інформаційної системи з використанням міжплатформної

структури ASP.NET Core, авторизації на основі техніки JSON Web Token (JWT), фреймворку Angular, системи управління базами даних, а також мовою структурованих запитів SQL.

На основі використання програмного середовища Python було здійснено програмну реалізацію регресійного аналізу обсягів замовлень послуг готельного бізнесу, яка інтегрована у Веб-інформаційну систему.

Макет інтерфейсу системи було спроєктовано, а також визначені основні ролі для користувачів, структура та прототип інформаційної системи, побудовано відповідні діаграми. Визначено, що для ефективного проєктування ПЗ краще використовувати багаторівневу архітектуру.

Окреслено особливості використання автентифікації та авторизації, їхню роль у захисті даних користувачів, а також налаштування у системі з використанням відповідних кнопок.

Інтеграція модуля прогнозування реалізована за допомогою мови програмування Python і лінійної регресії. Для Веб-інформаційної системи лінійна регресія є доволі корисною тому, що завдяки їй можна спрогнозувати відповідь, використовуючи новий набір предикторів.

Під час дослідження було використано основні наукові результати і висновки, одержані на основі методів ідентифікації, машинного навчання, статистичних методів, методів теорії систем.

Насамкінець після збору та аналізу усіх даних щодо необхідних елементів Веб-інформаційної системи було обрано методи ідентифікації запропонованої моделі та здійснено їх програмну реалізацію.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Архітектура та проектування програмного забезпечення : лекції. – Державний університет "Житомирська політехніка" - Освітній портал. – URL: https://learn.ztu.edu.ua/mod/book/tool/print/index.php?id=278
2. Аутентифікація і авторизація: що це і в чому відмінність. – QualityAssuranceGroup. – URL: https://qagroup.com.ua/publications/autentyfikatciia-i-avtoryzatciia/
3. Берещак В. Огляд ринку готелів: що відбувається з готелями в Україні. *ThePage.ua-Exclusive.* – URL: https://thepage.ua/ua/exclusive/oglyad-rinku-goteliv-sho-vidbuvayetsya-z-gotelyami-v-ukrayini
4. Бойко М. Овербукінг як інструмент ревеню-менеджменту / М. Бойко, М. Босовська, М. Кулик // CIENTIA RUCTUOSA (ВІСНИК Київського національного торговельно-економічного університету), 128 (6), 2019. – С. 45–54. – URL: https://doi.org/10.31617/visnik.knute.2019(128)04
5. Демиденко М. А. Введення в сучасні бази даних : навч. посібник / М.А. Демиденко // НТУ «Дніпровська політехніка». – Д. : 2020. – 38 с. – URL: http://ir.nmu.org.ua/bitstream/handle/123456789/154887/MA%20Demidenko%20INTRODUCTION%20TO%20MODERN%20DATABASES.pdf?sequence=1
6. Денисенко А. Фреймворки у веб-розробці – що це, які існують і для чого потрібні. – Highload, 2022. – URL: https://highload.today/uk/frejmvorki-u-veb-rozrobtsi-shho-tse-yaki-isnuyut-i-dlya-chogo-potribni/
7. Живко З. Б. Інформаційне забезпечення діяльності підприємства / З. Б. Живко, І. П. Мігус, О. М. Мартин, Л. В. Кухарська // II International Scientific Symposium "Modeling perspective of socio-economic and cultural-humanitarian systems in terms of transformation the information society". – Athens-Kyiv: nternational Academy of Information Science, 2018. – С. 24-26. – URL:https://sci.ldubgd.edu.ua/bitstream/123456789/8715/1/1.pdf

8. Захарова О. Платформи великих даних. Основні задачі, властивості та переваги. – Проблеми програмування. Експертні та інтелектуальні інформаційні системи, 2019. – URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/161499/07-Zakhrova.pdf?sequence=1
9. Іванов В. Г. Сучасні інформаційні системи і технології: конспект лекцій / В. Г. Іванов, С. М. Іванов, В. В. Карасюк та ін.; за заг. ред. В. Г. Іванова, В. В. Карасюка. – Х.: Нац. юрид. ун-т ім. Ярослава Мудрого, 2014. – 347 с. – URL: https://dspace.nlu.edu.ua/bitstream/123456789/6421/1/Konspekt_leksiy_95.pdf
10. Карпенко М. Ю. Технології створення програмних продуктів та інформаційних систем : навч. посібник / М. Ю. Карпенко, Н. О. Манакова, І. О. Гавриленко // Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2017. – С. 51-52.
11. Каштан В. Ю. Конспект лекцій з дисципліни “Бази даних в інформаційних системах” / В. Ю. Каштан, Д. В. Іванов. – Дніпро.: НТУ «ДП», 2020. – 58 с. – URL: https://it.nmu.org.ua/ua/scientific_method_materials/lecture_notes/%D0%9A%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82_%D0%BB%D0%B5%D0%BA%D1%86%D1%96%D0%B9_%D0%91%D0%94_%D1%87%D0%B0%D1%81%D1%82%D0%B8%D0%BD%D0%B01_2020.pdf
12. Козаков Т. ТОП-7 фреймворків Python. – Dev.ua, 2022. – URL: https://dev.ua/news/top-7-freimvorkiv-python-dlia-rozrobky-veb-dodatkiv
13. Корж Н. В. Reveneu-менеджмент: особливості групових продажів та бронювань у готелі / Н. В. Корж, О. В. Стасюк // Економіка та суспільство. – Мукачево: Мукачівський державний університет, 2018. – С. 601-605. – URL: https://chmnu.edu.ua/wp-content/uploads/2019/06/Ekonomika-i-suspilstvo-14-2018.pdf#page=601

14. Костенко О. Б. Організація баз даних та знань : конспект лекцій / О. Б. Костенко, І. О. Гавриленко // Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2021. – 92 с. – URL: http://eprints.kname.edu.ua/60505/1/2020%20%D0%BF%D0%B5%D1%87.%20134_%D0%9B.pdf
15. Костюкова Н. С. Навчальний модуль «Особливості тестування ігрових додатків» дисципліни «Якість програмного забезпечення та тестування» / Укл.: Н. С. Костюкова, С. О. Цололо. – Покровськ: ДонНТУ, 2018. – 87 с. – URL: https://ec.europa.eu/programmes/erasmus-plus/project-result-content/a2f85e46-3c1f-4b63-916b-13c24ea59d66/09_Bachelor_Features%20of%20Game%20Applications%20Testing.pdf
16. Кучер Д. Б. Підвищення якості готельних послуг через внутрішню оптимізацію процесів надання послуг на вітчизняних підприємствах готельного господарства / Д.Б. Кучер // Вісник Чернівецького торговельно-економічного інституту. Економічні науки. – 2012. – Вип.1. – С. 230-237 URL: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?I21DBN=LINK&P21DBN=UJRN&Z21ID=&S21REF=10&S21CNR=20&S21STN=1&S21FMT=ASP_meta&C21COM=S&2_S21P03=FILA=&2_S21STR=Vchtei_2012_1_38
17. Левкович У. Історія розвитку готельного господарства, суть, види та становлення готельної індустрії / У.Левкович // Молодь і ринок, 2014. – N 5. – С. 156-161. – URL: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?I21DBN=LINK&P21DBN=UJRN&Z21ID=&S21REF=10&S21CNR=20&S21STN=1&S21FMT=ASP_meta&C21COM=S&2_S21P03=FILA=&2_S21STR=Mir_2014_5_36
18. Лосєв М. Ю. Бази даних : навчально-практичний посібник для самостійної роботи студентів / М. Ю. Лосєв, В. В. Федько. – Харків : ХНЕУ ім. С. Кузнеця, 2018. – 233 с. – URL:

http://repository.hneu.edu.ua/bitstream/123456789/21468/1/2018-%D0%9B%D0%BE%D1%81%D1%94%D0%B2%20%D0%9C%20%D0%AE%2C%20%D0%A4%D0%B5%D0%B4%D1%8C%D0%BA%D0%BE%20%D0%92%20%D0%92.pdf

19. Марченко А. В. Проектування інформаційних систем, 2015. – URL: https://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20151208095132/content-20151208095132.pdf
20. Мізюк О. Путівник по Linux. – Основи використання Linux у школі, 2021. – URL: https://linuxguide.rozh2sch.org.ua/
21. Навчальний посібник з дисципліни «Технології розробки програмного забезпечення» / Укл. Л. М. Дегтярьова. – Полтава: ПолтНТУ, 2017. – 218 с. – URL: http://reposit.nupp.edu.ua/bitstream/PoltNTU/4431/1/%D0%A3%D1%87%D0%B5%D0%B1%D0%BD%D0%B8%D0%BA_%D0%A2%D0%A0%D0%9F%D0%97_%D0%BF%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%B5%D0%BD%D0%BE-converted.pdf
22. Настич. І. Падіння завантаженості, зниження Rack rates та складне відновлення: готельний бізнес під час COVID 19. *PropertyTimes*, 2020. – URL: https://propertytimes.com.ua/gostinichnaya_nedvizhimost/padinnya_zavantazhenosti_znizhennya_rack_rates_ta_skladne_vidnovlennya_gotelniy_biznes_pid_chas_covid_19
23. Новітні інформаційно-комунікаційні технології в освіті : матеріали III Всеукраїнської науково-практичної Інтернет-конференції молодих учених та студентів (Полтава, 18-19 листопада 2015 р.). – Полтава: ФОБ Болотін А.В., 2015. – 224 с. – URL: https://dspace.udpu.edu.ua/bitstream/6789/4548/1/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA_IICTE_2015.pdf

24. Олещенко Л. М. Технології оброблення великих даних: конспект лекцій з дисципліни «Технології оброблення великих даних» : навч. посіб. / Л.М. Олещенко // КПІ ім. Ігоря Сікорського. – Київ: КПІ ім. Ігоря Сікорського, 2021. – 227 с. – URL: https://ela.kpi.ua/bitstream/123456789/42206/1/%D0%9AonspLekts_Tekhnolohii-obroblennia-velykykh-danykh_%D0%9Eleshchenko.pdf
25. Основні відомості про бази даних. – Microsoft.com, 2021. – URL: https://support.microsoft.com/uk-ua/office/%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%96-%D0%B2%D1%96%D0%B4%D0%BE%D0%BC%D0%BE%D1%81%D1%82%D1%96-%D0%BF%D1%80%D0%BE-%D0%B1%D0%B0%D0%B7%D0%B8-%D0%B4%D0%B0%D0%BD%D0%B8%D1%85-a849ac16-07c7-4a31-9948-3c8c94a7c204
26. Особливості фреймворку AngularJS. – dzudzylo.com, 2017. – URL: https://dzudzylo.com/javascript/osoblyvosti-frejmvorku-angularjs.html
27. Пінчук Н. С. Інформаційні системи і технології в маркетингу: навч.метод. посібник для самостійного вивчення дисципліни / Н. С. Пінчук, Г. П. Галузинський, Н. С. Орленко // Міністерство освіти і науки України, КНЕУ. – Київ: КНЕУ, 2010. – 352 с.
28. Попівщий В. І. Курс «Використання сучасних Web-фреймворків», тема «JavaScript фреймворк Angular». – ІННІ ЗНУ, каф. ПЗАС, 2021. – URL: https://moodle.znu.edu.ua/pluginfile.php?file=/652681/mod_resource/content/1/%d0%9b%d0%b5%d0%ba%d1%86%d1%96%d1%8f%201%20%d0%a4%d1%80%d0%b5%d0%b9%d0%bc%d0%b2%d0%be%d1%80%d0%ba%20Angular%20%d0%86%d0%9d%d0%9d%d0%86%20%d0%97%d0%9d%d0%a3%202021.pdf

29. Пулеко І. В. Особливості застосування алгоритмів лінійної регресії у службі машинного навчання Microsoft Azure / І. В. Пулеко, С. В. Обіход // Тези доповідей III Всеукраїнської науково-технічної конференції «Комп'ютерні технології: інновації, проблеми, рішення», 26-27 листопада 2020 р. – Житомир: Житомирська політехніка, 2020. – С. 79-80. – URL: https://conf.ztu.edu.ua/wp-content/uploads/2021/01/tezy-dopovidej-kt2020_os-2.pdf
30. Проектування інформаційних систем: Загальні питання теорії проектування ІС (конспект лекцій) : навч. посіб. / КПІ ім. Ігоря Сікорського; уклад.: О. С. Коваленко, Л. М. Добровська. – Київ : КПІ ім. Ігоря Сікорського, 2020. – 192 с. – URL: https://ela.kpi.ua/bitstream/123456789/33651/1/PIS_KL.pdf
31. Роглєв Х. Й. Основи готельної справи / Х. Й. Роглєв // Основи готельного менеджменту: навч.посіб. – Київ: Видавництво, 2004. – С. 6-41.
32. Роглєв Х. Й. Сучасні інформаційні технології в управлінні готелем / Х.Й. Роглєв // Основи готельного менеджменту: навч.посіб. – Київ: Видавництво, 2004. – С. 136-153.
33. Рудик О. Java: бібліотеки й модулі. – URL: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/66_Java/index.html
34. Серіалізація JSON і XML у веб-API ASP.NET. – Microsoft.com, 2022. – URL: https://learn.microsoft.com/ru-ru/aspnet/web-api/overview/formats-and-model-binding/json-and-xml-serialization
35. Сидоренко В. В. Організація баз даних: навч. посібник / В. В. Сидоренко, Л. В. Константинова, С. А. Смірнов // Видавець. Лисенко В. Ф, Кропивницький: ЦНТУ, 2018. – 274 с. – URL: http://dspace.kntu.kr.ua/jspui/bitstream/123456789/10527/1/NavPosOBD.pdf
36. Соловьев А. А. Интернет как средство развития туризма в Крыму / А. А. Соловьев // Культура народов Причерноморья: научн. журнал, 2008. – N 137. – С. 130-133.

37. Табунщик Г. В. Проєктування інформаційної інфраструктури медичних та телемедичних систем / Г. В. Табунщик, Т. І. Каплієнко, О. А. Петрова, О. В. Шитікова // Вид. ПП "Євро-Волинь", Житомир, 2021. – 198 с. – URL: http://eir.zp.edu.ua/bitstream/123456789/8099/1/NP_Tabunshchik.pdf
38. Тартачний О. Навіщо потрібні фреймворки та бібліотеки. – Robot_dreams Media. – URL: https://robotdreams.cc/uk/blog/251-zachem-nuzhny-freymvorki-i-biblioteki
39. Толстохатько В. А. Бази даних: проектування та використання для обліку нерухомого майна : навч. посібник / В. А. Толстохатько, О. Є. Поморцева, І. М. Патракеєв // Харк. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Х. : ХНУМГ, 2014. – 174 с. – URL: https://core.ac.uk/download/pdf/33757091.pdf
40. Фаулер М. UML. Основы. 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 192 с.
41. Фостолович В. А. Сучасні інструменти системи управління бізнесом у сфері готельно-ресторанної справи / В. А. Фостолович // Науково-практичний журнал "Агросвіт" – Київ: Видавництво ТОВ "ДКС-Центр", 2022. – Економічна наука; № 11. – URL: http://eprints.zu.edu.ua/34375/1/Fostolovych.pdf
42. Херрон Д. Node.js. Розробка серверних веб-додатків в JavaScript : / Пер. з англ. Слінкіна А.А. - Москва: ДМК Пресс, 2012. – 144 с.
43. Черномазюк А. Г. Інновації у сфері готельно-ресторанного бізнесу. – Вісник Хмельницького національного університету. Економічні науки. – No 5, Т. 2, 2014. – С. 269-272.
44. Юрченко М. Є. Прогнозування та аналіз часових рядів / Укл.: М. Є. Юрченко. – Чернігів: ЧНТУ, 2018. – 88 с.
45. Юрчишин В. Я. Хмарні та грід-технології конспект лекцій : навч. посіб. / В.Я.Юрчишин; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2019. – 264 с. – URL:

https://ela.kpi.ua/bitstream/123456789/29960/1/Khmarni_ta_grid-tekhnolohii_Konspekt_lektsii1.pdf

46. Як використовувати JSON Web Tokens (JWT) для автентифікації. – Codeguida.com, 2019. – URL: https://codeguida.com/post/1567
47. Angular. – Вікі Центральноукраїнського державного педагогічного університету, 2019. – URL: https://wiki.cuspu.edu.ua/index.php/Angular
48. Compare ASP.NET Web API 2 and ASP.NET Core. – Microsoft.com, 2022. – URL: https://learn.microsoft.com/uk-ua/dotnet/architecture/porting-existing-aspnet-apps/webapi-differences
49. Competitions. – Kaggle.com. – URL: https://www.kaggle.com/competitions
50. Create web APIs with ASP.NET Core. – Microsoft.com, 2022. – URL: https://learn.microsoft.com/uk-ua/aspnet/core/web-api/?view=aspnetcore-7.0
51. Hadjinicola G.C., Panayi C. The overbooking problem in hotels with multiple touroperators. – International Journal of Operations & Production Management, Vol 17, No 9. 1997. – pp 874-885. – URL: https://www.emerald.com/insight/content/doi/10.1108/01443579710171208/full/html
52. Introduction to JSON Web Tokens. – JWT.io. – URL:https://jwt.io/introduction
53. IronPython. – IronPython.net. – URL: https://ironpython.net/
54. Ivanov S. Management of Overbookings in the Hotel Industry – Basic Concepts and Practical Challenges. 2008. – URL: https://www.researchgate.net/publication/228254871_Management_of_Overbookings_in_the_Hotel_Industry_-_Basic_Concepts_and_Practical_Challenges
55. Ivanov, S. Optimal overbooking limits for a hotel with three room types and with upgrade and downgrade constraints Tourism Economics. 21 (1). 2015. – pp. 223-240. – URL: https://journals.sagepub.com/doi/abs/10.5367/te.2014.0444
56. Lane, D. Introduction to Statistics (англ.) / David Scott, Mikki Hebl, Rudy Guerra, Dan Osherson, Heidi Zimmer. – С. 462. – URL: http://onlinestatbook.com/Online_Statistics_Education

57. Liu S, Lai K. K, Wang S. Y. Booking models for hotel revenue management considering multiple-day stays. – International Journal of Revenue Management. 2(1). 2008. – C. 78-91. – URL: https://www.researchgate.net/publication/5172548_Booking_models_for_hotel_revenue_management_considering_multiple-day_stays
58. Lutkevich B. What is Framework. – Whalts.com, 2020. – URL: https://www.techtarget.com/whatis/definition/framework
59. Meuwissen K. Using mathematical models in the hotel industry: Maximizing revenues through discount strategies. – Rochester Institute of Technology. Department of Computational and Applied Mathematics, 2011. – URL: https://scholarworks.rit.edu/cgi/viewcontent.cgi?article=6002&context=theses
60. Mostipak J. Hotel Booking Demand. 2019. – URL: https://www.kaggle.com/jessemostipak/hotel-booking-demand
61. Munro, J. An Introduction to Mongoose for MongoDB and Node.js. – Envatotutsplus. 2017. – URL: https://code.tutsplus.com/uk/articles/an-introduction-to-mongoose-for-mongodb-and-nodejs--cms-29527
62. Noone B.M., Lee C.H. Hotel overbooking: The effect of overcompensation on customers' reaction to denied service. – Journal of Hospitality & Tourism Research, Vol 35, No 3. 2011. – pp 334-357. – URL: https://journals.sagepub.com/doi/10.1177/1096348010382238
63. PIP. – The Python Package Index. – URL: https://pypi.org/project/pip/
64. Property pages for React, Angular, and Vue projects in Visual Studio. – Microsoft.com, 2022. – URL: https://learn.microsoft.com/uk-ua/visualstudio/ide/reference/property-pages-javascript-esproj?view=vs-2022
65. Riasi A. Overbooking practices in the hotel industry and their impact on hotels' financial performance. – University of Delaware. 2018. – URL: https://udspace.udel.edu/handle/19716/24053
66. Rumbaugh, J. The unified modeling language reference manual (англ.) / Ivar Jacobson, Grady Booch. – Addison Wesley Longman Inc. 1999.

67. Stetsenko, D, Kapitanov, V. Optimizing hotel booking through simulation modeling. – Chapter 58 in DAAAM International Scientific Book. 2013. – C. 941-950. – URL: https://www.daaam.info/Downloads/Pdfs/science_books_pdfs/2013/Sc_Book_2013-058.pdf
68. Stojiljkovic, M. Linear Regression in Python. 2022. – URL: https://realpython.com/linear-regression-in-python/
69. The import system. – Python.org. – URL: https://docs.python.org/3/reference/import.html
70. Yuksel S. An integrated forecasting approach to hotel demand. – Department of Tourism Management. – Education. Mathematical and computer modelling. 46. 2007. – C. 1063-1067. – URL: ttps://www.sciencedirect.com/science/article/pii/S0895717707000866