НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету (Директор ННІ)	**Завідувач кафедри**
інформаційних технологій	комп'ютерних наук
(назва факультету (ННІ))	(назва кафедри)
_____ Ігор БОЛБОТ	_____ Белла ГОЛУБ
(підпис) (ім'я ПРІЗВИЩЕ)	(підпис) (ім'я ПРІЗВИЩЕ)
"___" _____ 2025 р.	"___" _____ 2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему Програмне забезпечення системи підтримки прийняття рішень керівництвом туристичною компанією

Спеціальність 121 «Інженерія програмного забезпечення»
(код і найменування)

Освітня програма Програмне забезпечення інформаційних систем
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Гарант освітньої програми

к.ф.-м.н., доцент _____ Віктор КИРИЧЕНКО
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Керівник магістерської кваліфікаційної роботи

к.ф.-м.н., доцент _____ Яніна КРИВОРУЧКО
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Виконав _____ Гліб КІРІН
(підпис) (ім'я ПРІЗВИЩЕ здобувача)

КИЇВ – 2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
к.т.н., доцент _____ Белла ГОЛУБ
(науковий ступінь, вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)
"01" листопада 2024 року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Кіріну Глібу Олександровичу
(прізвище, ім'я, по батькові)

Спеціальність 121 «Інженерія програмного забезпечення»
(код і найменування)

Освітня програма Програмне забезпечення інформаційних систем
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи Програмне забезпечення системи підтримки прийняття рішень керівництвом туристичною компанією
затверджена наказом від "01" листопада 2024р. №1963 «С»

Термін подання завершеної роботи на кафедру 20.11.2025
(рік, місяць, число)

Вихідні дані до магістерської кваліфікаційної роботи: історичні показники продажів і бронювань турів туристичної компанії, сегментація клієнтів, ціни конкурентів із OTA/API, сезонні та погодні фактори, а також обмеження ресурсів (транспорт, готельний фонд, маркетинговий бюджет).

Перелік питань, що підлягають дослідженню:

1. Системний аналіз предметної області
2. Проектування інформаційного і програмного забезпечення
3. Реалізація програмного забезпечення
4. Тестування та оцінювання ефективності системи

Перелік графічного матеріалу (за потреби)

Дата видачі завдання "01" листопада 2024 р.

Керівник магістерської кваліфікаційної роботи _____ Яніна КРИВОРУЧКО
(підпис) (ім'я ПРІЗВИЩЕ)

Завдання прийняв до виконання _____ Гліб КІРІН
(підпис) (ім'я ПРІЗВИЩЕ)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API (Application Programming Interface) — інтерфейс прикладного програмування, що забезпечує взаємодію між програмними компонентами.

ARIMA / SARIMA — авторегресійна інтегрована модель ковзного середнього (сезонна модифікація) для прогнозування часових рядів.

CVaR (Conditional Value-at-Risk) — умовне математичне очікування втрат, що перевищують певний поріг ризику (VaR).

CRUD (Create, Read, Update, Delete) — базові операції з даними у реляційних системах управління базами даних.

CSV (Comma-Separated Values) — текстовий формат представлення табличних даних, у якому значення розділяються комами.

DSS (Decision Support System) — система підтримки прийняття рішень, призначена для аналітичної обробки даних і формування управлінських висновків.

ETL (Extract, Transform, Load) — процес вилучення, перетворення та завантаження даних у сховище.

GBM (Gradient Boosting Machine) — метод ансамблевого навчання на основі градієнтного бустингу дерев рішень.

GUI (Graphical User Interface) — графічний інтерфейс користувача.

JSON (JavaScript Object Notation) — текстовий формат обміну даними між клієнтом і сервером.

KPI (Key Performance Indicator) — ключовий показник ефективності, що характеризує результативність діяльності компанії.

MAPE (Mean Absolute Percentage Error) — середня абсолютна відносна похибка прогнозування.

MILP (Mixed Integer Linear Programming) — змішане цілочисельне лінійне програмування.

ML (Machine Learning) — машинне навчання, підрозділ штучного інтелекту, що забезпечує автоматичне виявлення закономірностей у даних.

NDCG (Normalized Discounted Cumulative Gain) — метрика оцінювання якості ранжування рекомендаційних моделей.

OLAP (Online Analytical Processing) — технологія оперативного аналітичного оброблення даних із багатовимірною структурою.

PCA (Principal Component Analysis) — метод головних компонент, який використовується для зниження розмірності даних.

REST (Representational State Transfer) — архітектурний стиль веб-сервісів, що базується на HTTP-протоколі.

RMSE (Root Mean Squared Error) — середньоквадратична похибка прогнозування.

SQL (Structured Query Language) — структурована мова запитів до баз даних.

TLS (Transport Layer Security) — криптографічний протокол для захищеного передавання даних у мережі.

UI (User Interface) — інтерфейс користувача, засіб взаємодії користувача із системою.

VaR (Value-at-Risk) — максимальні можливі втрати при заданому рівні довіри за визначений період.

XGBoost (Extreme Gradient Boosting) — алгоритм градієнтного бустингу з оптимізованою продуктивністю.

ВСТУП

Традиційні методи управління, що спираються на інтуїцію або фрагментарний аналіз, вже не забезпечують необхідного рівня ефективності. Тому актуальним є створення інтелектуальних систем підтримки прийняття рішень, здатних автоматично обробляти великі обсяги даних, формувати рекомендації для керівництва туристичної компанії та забезпечувати стратегічну адаптивність її діяльності в умовах мінливого ринку [1]. Запропонований підхід базується на використанні сучасних методів аналітики даних, машинного навчання та програмних засобів інтеграції інформаційних потоків.

Метою дослідження є розроблення програмного забезпечення системи підтримки прийняття рішень для керівництва туристичною компанією, що забезпечує автоматизовану обробку інформації, прогнозування ключових показників ефективності та візуалізацію результатів аналітики в інтерактивному інтерфейсі.

Для досягнення поставленої мети необхідно виконати такі **завдання**:

1. Провести системний аналіз предметної області управління туристичними компаніями та визначити основні інформаційні потоки.
2. Дослідити існуючі рішення у сфері бізнес-аналітики та СППР для туристичного сектору, визначити їх переваги та недоліки.
3. Розробити модель предметної області та побудувати UML-діаграми для опису процесів прийняття управлінських рішень.
4. Сформулювати функціональні, нефункціональні та технічні вимоги до системи.
5. Розробити архітектуру програмного забезпечення СППР, передбачивши модулі збору, обробки, аналізу та візуалізації даних.
6. Реалізувати програмний прототип із використанням сучасних технологій (Python, Flask, PostgreSQL, Power BI) та перевірити його працездатність на тестових даних.

7. Провести апробацію результатів і сформулювати рекомендації щодо практичного застосування системи.

Об’єктом дослідження є процес підтримки прийняття управлінських рішень у діяльності туристичних компаній, а **предметом** - методи, засоби та алгоритмічні підходи до побудови програмного забезпечення системи підтримки прийняття рішень на основі аналітичної обробки структурованих і неструктурованих даних.

Для реалізації поставлених завдань використовуються методи системного аналізу, математичного моделювання, інтелектуального аналізу даних, кластеризації та прогнозування, а також технології розроблення програмних систем, зокрема об’єктно-орієнтоване програмування, шаблон проектування Model–View–Controller, бібліотека Java Swing та вбудована база даних SQLite, що забезпечує ефективне зберігання інформації в автономному режимі [2].

Наукова новизна роботи полягає у створенні комплексної архітектури програмної системи підтримки прийняття рішень для туристичної компанії, що поєднує технології оброблення даних і візуальної аналітики в єдиному інтерактивному середовищі.

Запропонований підхід дозволяє інтегрувати внутрішні показники діяльності підприємства (прибуток, завантаженість турів, ефективність менеджерів) із зовнішніми факторами (попит, сезонність, рейтинги напрямів) для формування рекомендацій у реальному часі. Розроблене рішення сприяє підвищенню точності управлінських рішень і зменшенню ризику неефективного планування.

Практична значущість дослідження полягає у можливості впровадження розробленої СППР як інструменту бізнес-аналітики для малих і середніх туристичних компаній, що прагнуть підвищити ефективність управління на основі даних.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області системи підтримки прийняття рішень

У межах предметної області системи описується сукупність процесів збирання, інтеграції, аналізу та візуалізації інформації, необхідної для формування обґрунтованих управлінських рішень. Основні учасники системи включають керівництво компанії, менеджерів із продажів, аналітиків, а також зовнішні інформаційні джерела - OTA-платформи (Booking, Amadeus) і сервіси з відкритими даними (погода, рейтинги, соціальні мережі). Інформаційні потоки проходять через модулі збору, аналітики, рекомендацій та візуалізації, що утворюють логічну архітектуру системи. На рис. 1.1 подано контекстну діаграму (DFD Level 0), яка відображає взаємодію зовнішніх сутностей, процесів і сховищ даних.

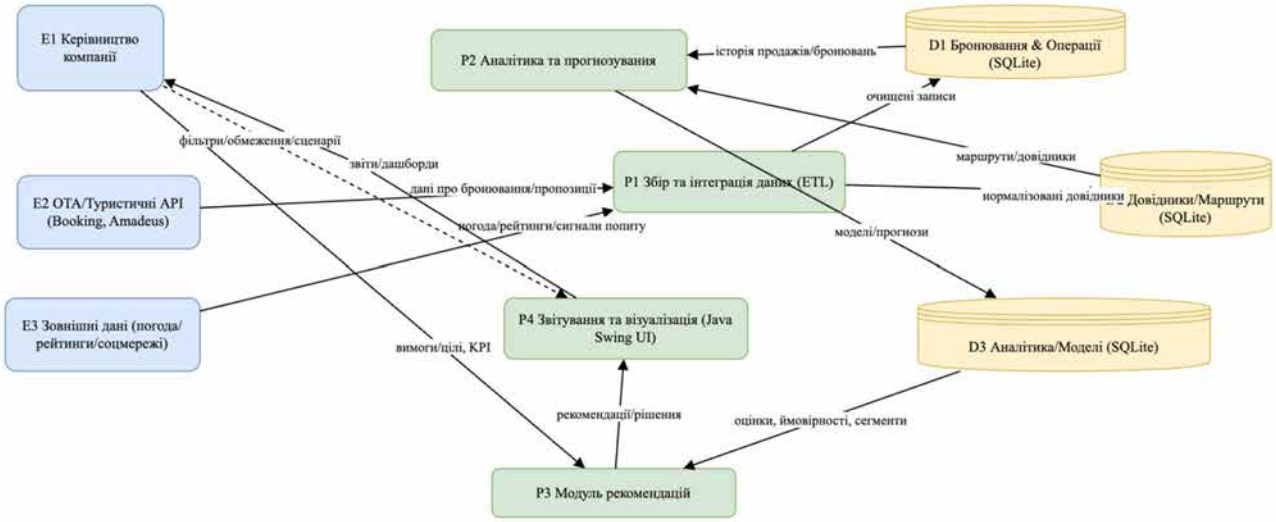

Рис. 1.1 – DFD Level 0 системи підтримки прийняття рішень для туристичної компанії

Система реалізує чотири основні процеси: P1 – збір та інтеграція даних (ETL), що відповідає за об’єднання інформації з зовнішніх API та джерел уніфікованого формату; P2 – аналітика та прогнозування, де здійснюється обробка історичних даних і побудова моделей машинного навчання; P3 – модуль рекомендацій, який генерує управлінські поради на основі результатів

аналізу; P4 – звітність і візуалізація, що реалізована у вигляді графічного інтерфейсу на Java Swing. Для збереження інформації використовується вбудована база даних SQLite, де виділено три основні сховища: D1 – бронювання та операції, D2 – довідники й маршрути, D3 – аналітичні моделі та результати прогнозів.

Основними інформаційними потоками в межах системи є: передача даних про бронювання та пропозиції від OTA-платформ до модуля інтеграції; надходження зовнішніх аналітичних сигналів (рейтинги напрямів, погода, тренди соцмереж); формування очищених і нормалізованих записів для подальшого використання в аналітичних підсистемах; обмін результатами прогнозів та рекомендацій між аналітичними модулями та інтерфейсом користувача. Керівництво отримує агреговані звіти й дашборди та може задавати фільтри, обмеження чи сценарії для уточнення рішень.

Класифікацію основних елементів предметної області наведено в табл. 1.1, де виділено сутності, функції, джерела даних та користувачів системи.

Таблиця 1.1

Класифікація компонентів предметної області СППР туристичної компанії

№	Елемент предметної області	Тип об’єкта	Основна функція	Приклад реалізації
1	Керівництво компанії	Зовнішня сутність	Прийняття стратегічних рішень, визначення KPI	Інтерфейс Java Swing, дашборди
2	OTA / Туристичні API	Зовнішнє джерело	Надання даних про тури, бронювання, маршрути	Booking API, Amadeus API
3	ETL-модуль	Процес	Інтеграція та очищення даних	Python-скрипти, JDBC
4	Аналітичний модуль	Процес	Прогнозування попиту та ефективності	ML-моделі (Regression, K-Means)
5	Модуль рекомендацій	Процес	Генерація управлінських рішень	Decision Engine
6	Сховище SQLite	Дані	Збереження історії бронювань і моделей	Файлова база даних
7	Інтерфейс користувача	Компонент	Візуалізація аналітичних результатів	Java Swing UI

Предметна область системи охоплює весь життєвий цикл управлінських даних у туристичній компанії - від збору й інтеграції до формування рекомендацій для керівництва. Інформаційні процеси системи є взаємопов’язаними, а розподіл функцій між модулями забезпечує масштабованість, модульність і можливість подальшого розширення функціоналу на основі аналітичних моделей.

1.2 Теоретико-методологічні засади та стан наукових досліджень

У сучасних наукових дослідженнях спостерігається активний розвиток методів побудови інтелектуальних систем підтримки прийняття рішень у сфері туризму, що поєднують принципи аналізу даних, машинного навчання, просторової аналітики (GIS) і багатокритеріальної оптимізації. Зокрема, у роботі Xiao Y. et al. (2024) представлено архітектуру інтелектуальної туристичної СППР, що включає підсистему аналізу туристичних даних та адаптивну підсистему прийняття рішень, які взаємодіють із базою знань, сховищем аналітичних моделей і мультимедійними ресурсами (рис. 1.2). Ця модель передбачає інтеграцію просторових і часових даних (GIS, GPS, RS) та використання агентно-орієнтованих модулів (Data Mining Agent, Learning Agent, Reasoning Agent), що формують рішення на основі комплексного аналізу інформаційних потоків [1].

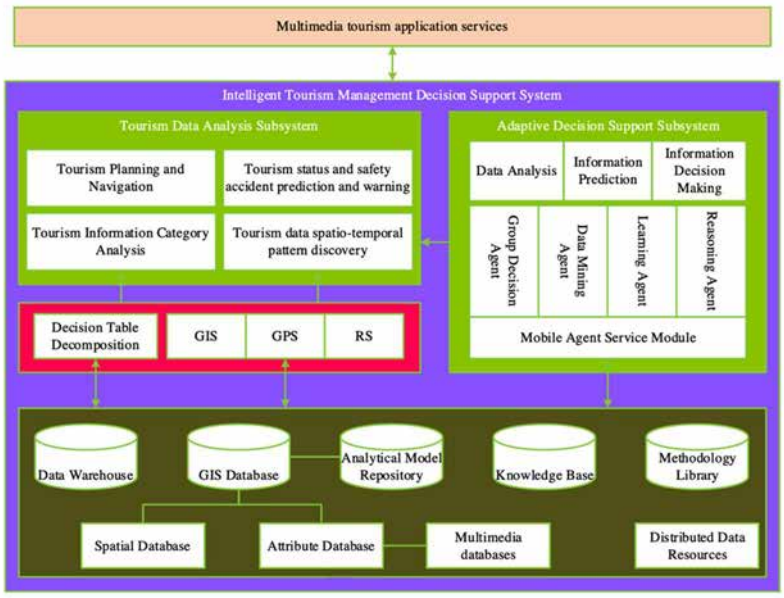

Рис. 1.2 – Архітектура інтелектуальної системи підтримки прийняття рішень у сфері туризму [1]

Подальші дослідження Ramos C. M. Q. et al. (2023) зосереджені на створенні системи аналізу задоволеності туристів, яка використовує алгоритми класифікації та статистичні моделі для виявлення закономірностей у поведінці споживачів. На рис. 1.3 наведено результати аналітичного порівняння рівня задоволеності клієнтів у різних країнах, що демонструє вплив культурних та сервісних чинників на оцінки користувачів. Отримані криві відображають варіативність показників, що дає підстави для подальшого прогнозного моделювання попиту [2].

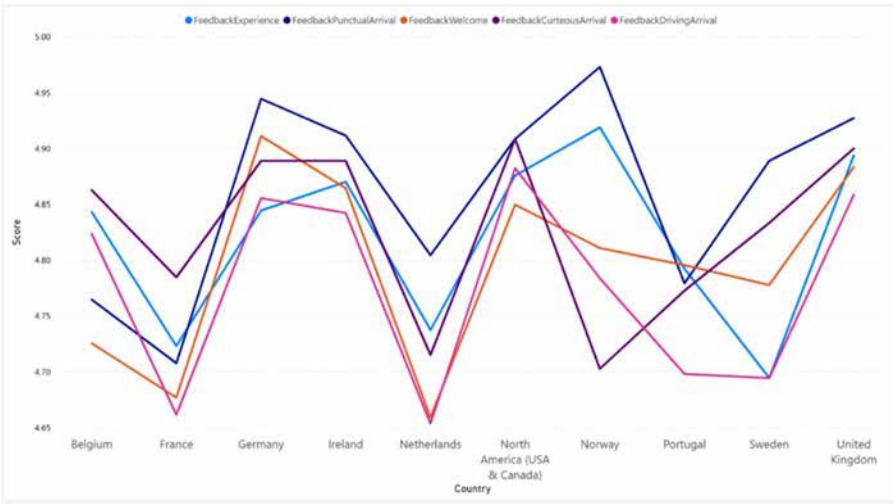

Рис. 1.3 – Графік порівняння показників задоволеності клієнтів у різних країнах [2]

Аналогічно, рис. 1.4 ілюструє залежність між середнім рівнем задоволеності та кількістю пасажирів за напрямами, що відображає поведінкові тренди туристів і ефективність транспортної логістики. У роботі продемонстровано можливість використання кореляційного аналізу між кількістю перевезень і середнім індексом задоволеності для визначення потенційно прибуткових напрямів [2].

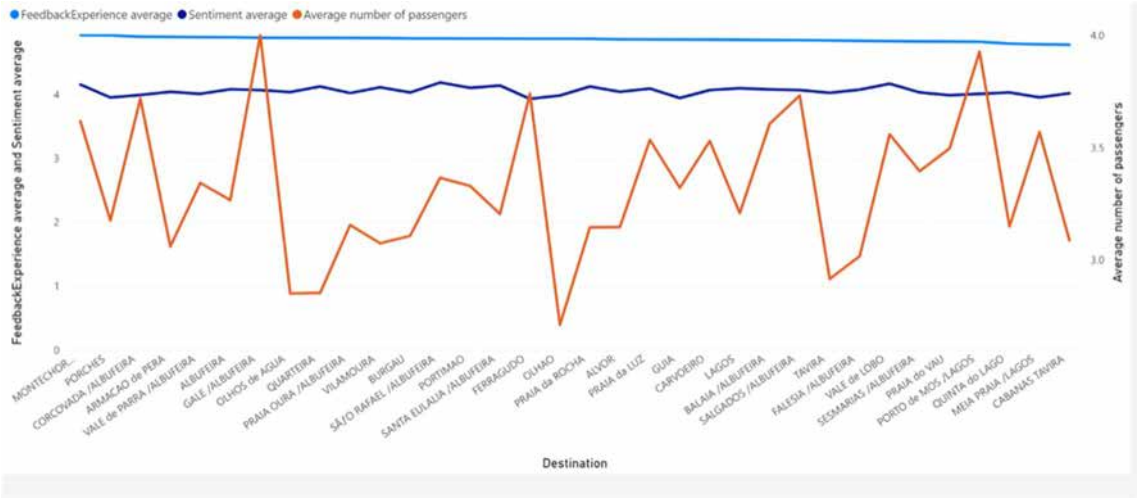

Рис. 1.4 – Залежність між рівнем задоволеності клієнтів і середньою кількістю пасажирів [2]

Дослідження Núñez J. C. S. (2024) узагальнює стан розвитку методів машинного навчання в туризмі, виділяючи основні напрями їх застосування. Згідно з аналітичною діаграмою (рис. 1.5), 35 % досліджень орієнтовані на прогнозування, 31 % - на класифікацію, 9 % - на побудову рекомендаційних систем, а решта зосереджена на задачах кластеризації та оптимізації. Це підкреслює тенденцію переходу від класичних статистичних підходів до методів глибокого навчання для персоналізації туристичних пропозицій [3].

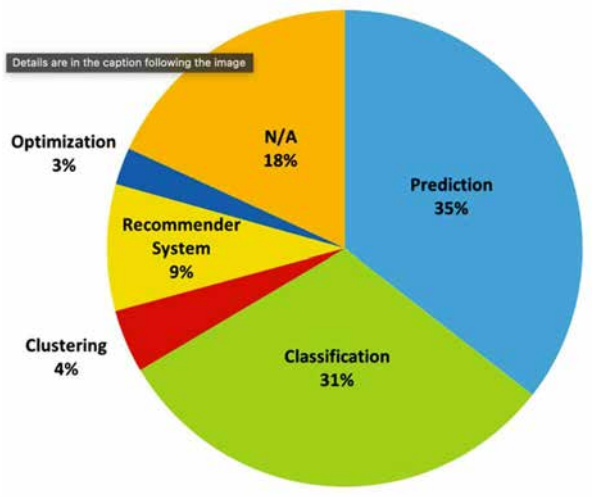

Рис. 1.5 – Структура напрямів застосування машинного навчання у сфері туризму [3]

На рис. 1.6 наведено порівняльну діаграму поширеності алгоритмів ML у туристичних СППР, що демонструє домінування нейронних мереж (CNN – 8 %, LSTM – 7 %, ANN – 7 %) та класичних моделей (Random Forest, SVM, Decision Tree), тоді як оптимізаційні моделі (PSO, GA, ARIMA) залишаються менш застосовуваними. Ці дані підтверджують тенденцію до використання гібридних підходів, у яких глибоке навчання комбінується з методами оптимізації для підвищення точності прогнозів [3].

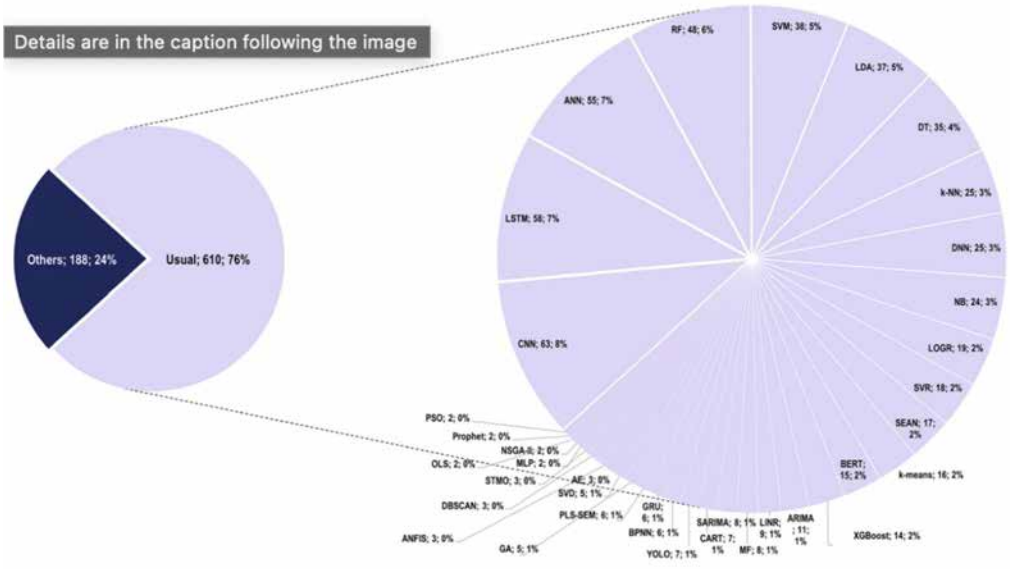

Рис. 1.6 – Поширеність алгоритмів машинного навчання у туристичних СППР [3]

На основі досліджених джерел можна зробити висновок, що сучасні СППР у сфері туризму еволюціонують від систем описової статистики до адаптивних аналітичних платформ, здатних навчатися на потокових даних. Наукова новизна запропонованої системи полягає у поєднанні інтерфейсного рівня Java Swing із локальною обробкою даних у SQLite та використанням гібридного ML-модуля для формування рекомендацій і прогнозів у режимі реального часу. Такий підхід забезпечує автономність, масштабованість і високу точність аналітичних висновків, що підвищує ефективність управління туристичними компаніями [1–3].

1.3 Аналіз існуючих рішень

Науково-технічний аналіз показав, що більшість існуючих систем фокусуються на візуальному моніторингу KPI, автоматизованому бронюванні, управлінні доходами та поведінковій аналітиці клієнтів, однак вони часто не забезпечують інтеграції модулів прогнозування попиту й рекомендаційних механізмів.

Одним із найпоширеніших інструментів є TripAdvisor Analytics Platform, що дозволяє відстежувати поведінку користувачів за допомогою Google Analytics і формувати агреговані звіти щодо трафіку, джерел переходів, конверсій і доходів (рис. 1.7). Ця система орієнтована на маркетингову аналітику, проте не включає повноцінний модуль підтримки управлінських рішень або машинного прогнозування попиту [11].

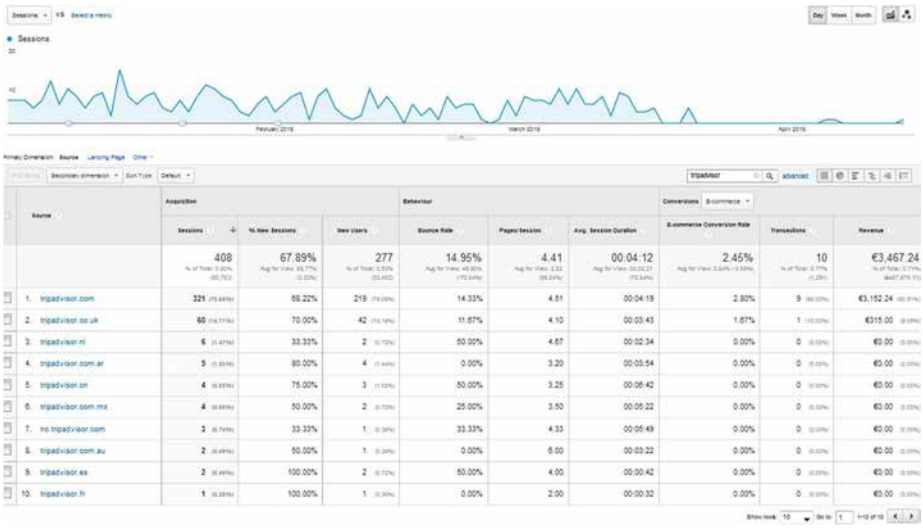

Рис. 1.7 – Аналітична панель TripAdvisor Analytics Platform для моніторингу трафіку та конверсій

Інша відома система — Amadeus Travel Intelligence, що надає інструменти для оптимізації рейсів, управління завантаженістю та прогнозування доходів на основі великих даних (рис. 1.8). Вона інтегрує механізми планування та моделювання сценаріїв, дозволяючи авіакомпаніям адаптувати пропозиції до сезонного попиту. Попри високий рівень аналітики, продукт не призначений для комплексного управління туристичними підприємствами малого та середнього рівня [12].

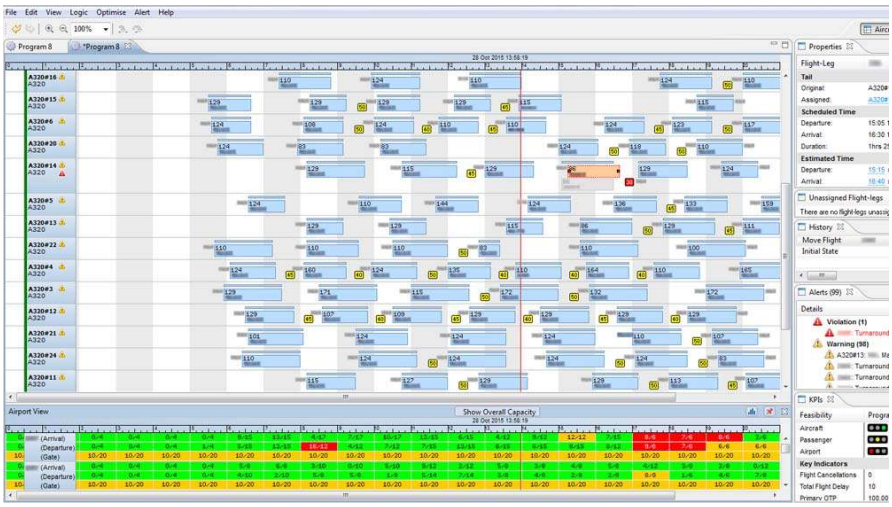

Рис. 1.8 – Інтерфейс Amadeus Travel Intelligence з модулями оптимізації рейсів і ресурсів

IBM Cognos Analytics (рис. 1.9) реалізує підхід корпоративної бізнес-аналітики з розгалуженою системою дашбордів, OLAP-запитами та

інтелектуальною візуалізацією даних. Рішення орієнтоване на побудову стратегічних звітів для керівництва туристичних компаній, має можливість інтеграції з CRM та ERP-системами, однак вимагає складної інсталяції та постійного адміністрування [13].

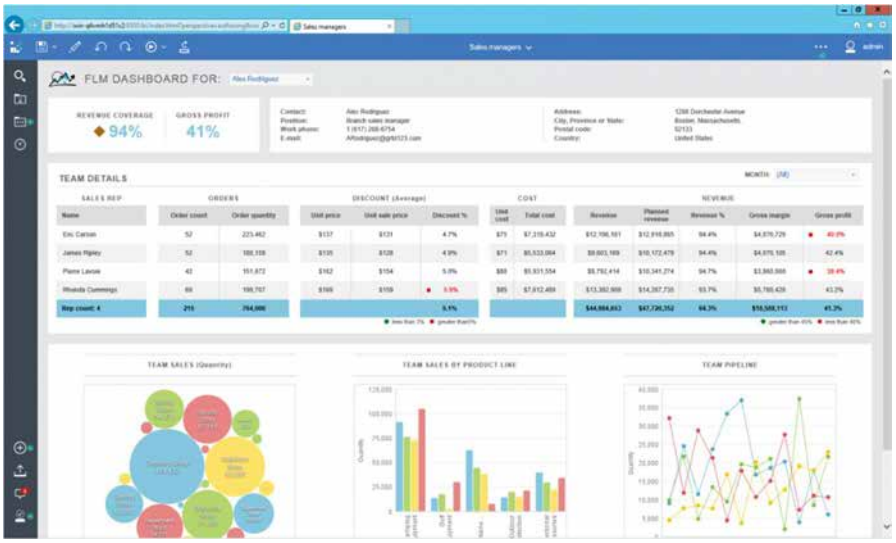

Рис. 1.9 – Панель керування IBM Cognos Analytics із KPI-індикаторами прибутковості та ефективності

Oracle Hospitality OPERA Cloud (рис. 1.10) поєднує функції управління бронюваннями, контролю ресурсів і моніторингу подій у готельному бізнесі. Система має інтегрований модуль фінансової звітності та підтримує розподілену аналітику, однак не передбачає глибинного аналізу ринкових трендів чи поведінкових моделей клієнтів [14].

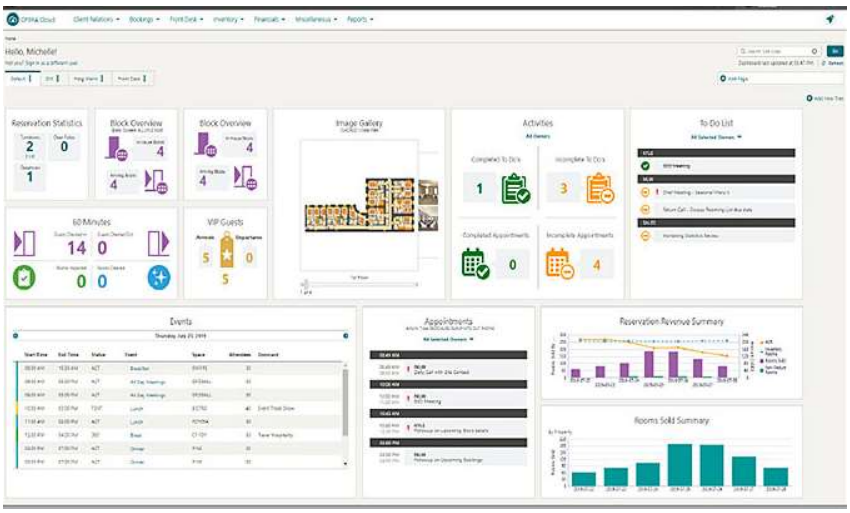

Рис. 1.10 – Головна панель Oracle Hospitality OPERA Cloud із віджетами бронювання та доходів

Ще одним прикладом є SAP BusinessObjects BI for Travel Industry, яка забезпечує аналіз демографічних та просторових даних клієнтів (рис. 1.11). Завдяки інтеграції з GIS-шарами система дозволяє візуалізувати географічний розподіл попиту та прогнозувати потенційні ринки. Попри це, рішення не підтримує автономну роботу й потребує хмарної інфраструктури SAP HANA [15].

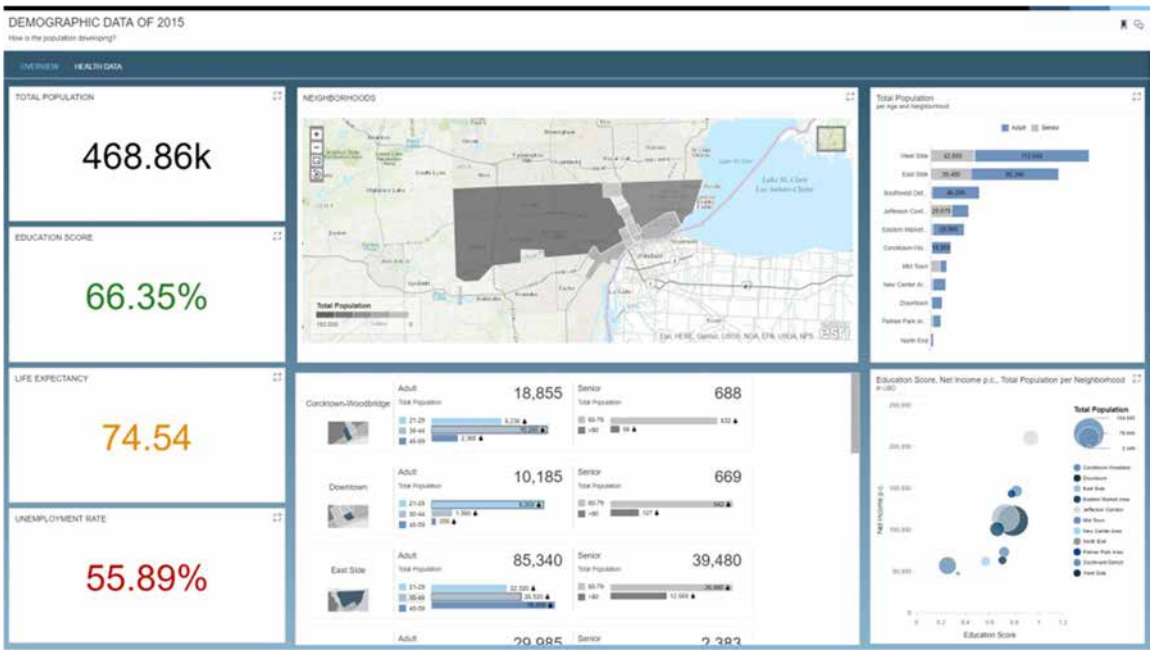

Рис. 1.11 – Візуалізація просторово-демографічних показників у SAP BusinessObjects BI for Travel Industry

Порівняльний аналіз систем наведено в табл. 1.3, де узагальнено їхні можливості, архітектурні особливості та обмеження.

Таблиця 1.3

Порівняльна характеристика існуючих систем підтримки прийняття рішень у туризмі

№	Система	Основне призначення	Технології	Ключові переваги	Недоліки	Використання у нашій системі
1	TripAdvisor Analytics Platform	Маркетингова аналітика	Google Analytics, SQL	Детальні звіти про поведінку користувачів	Відсутність прогнозування	Метрики KPI для дашборду UI (Java Swing)

Продовження таблиці 1.3

2	Amadeus Travel Intelligence	Плануванн я та оптимізація рейсів	Big Data, API	Прогнозуванн я доходів, оптимізація ресурсів	Висока вартість впровадженн я	Модель ETL для інтеграції даних
3	IBM Cognos Analytics	Корпоратив на аналітика	OLAP , Pytho n, SQL	Висока візуалізація, адаптивні дашборди	Складне адмініструва ння	Модуль дашбордів у Swing
4	Oracle Hospitality OPERA Cloud	Управління бронюванн ям і доходами	Java EE, Cloud	Автоматизація готельних операцій	Немає прогнозного модуля	Інтерфейс бронюванн я в нашу систему
5	SAP BusinessObje cts BI for Travel Industry	Просторова та демографіч на аналітика	GIS, HAN A	Інтеграція з геоданими	Високі вимоги до інфраструкту ри	Геоаналітич на візуалізація в нашій системі
6	Авторська розробка	Гібридна СППР для туризму	Java Swing , SQLit e, ML	Автономна аналітика, рекомендацій ний модуль	–	Реалізуєтьс я в роботі

Узагальнення досліджених рішень свідчить, що жодна з наявних платформ не забезпечує одночасно автономності, інтегрованої аналітики та гібридного прогнозування. Наукова новизна розроблюваної системи полягає в інтеграції локальної бази SQLite з Java Swing-інтерфейсом та використанні модулів машинного навчання для формування управлінських рекомендацій і прогнозів у реальному часі. Завдяки цьому досягається підвищення точності планування, скорочення часу аналітичного циклу та розширення функціональності систем підтримки прийняття рішень у туристичному бізнесі [11–15].

1.4 Моделювання програмної системи

Моделювання предметної області системи підтримки прийняття рішень керівництвом туристичної компанії базується на формалізації ключових бізнес-процесів, взаємодії користувачів і компонентів, а також потоків даних між підсистемами аналітики, оптимізації та звітності. Основна мета моделювання - відобразити логіку функціонування системи від збору первинних даних до формування рекомендацій і візуалізації результатів у дашбордах.

На рис. 1.12 подано діаграму прецедентів, що демонструє взаємодію основних акторів - керівника (CEO), фінансового директора, маркетинг-менеджера, операційного менеджера та аналітика даних - із ключовими функціями системи. Серед прецедентів виділено: прогноз попиту, оптимізацію турів (на основі MILP-моделі), аналіз ризику-дохідності (CVaR95), планування бюджету, сценарне моделювання та генерацію звітів у форматах PDF/Excel. Зовнішні системи CRM/ERP та постачальники даних (Amadeus, Booking API) забезпечують імпорт історичних даних і сигналів попиту, які інтегруються у внутрішню базу. Така модель відображає концепцію багаторівневої СППР з включенням аналітичних, оптимізаційних і моніторингових модулів [1].

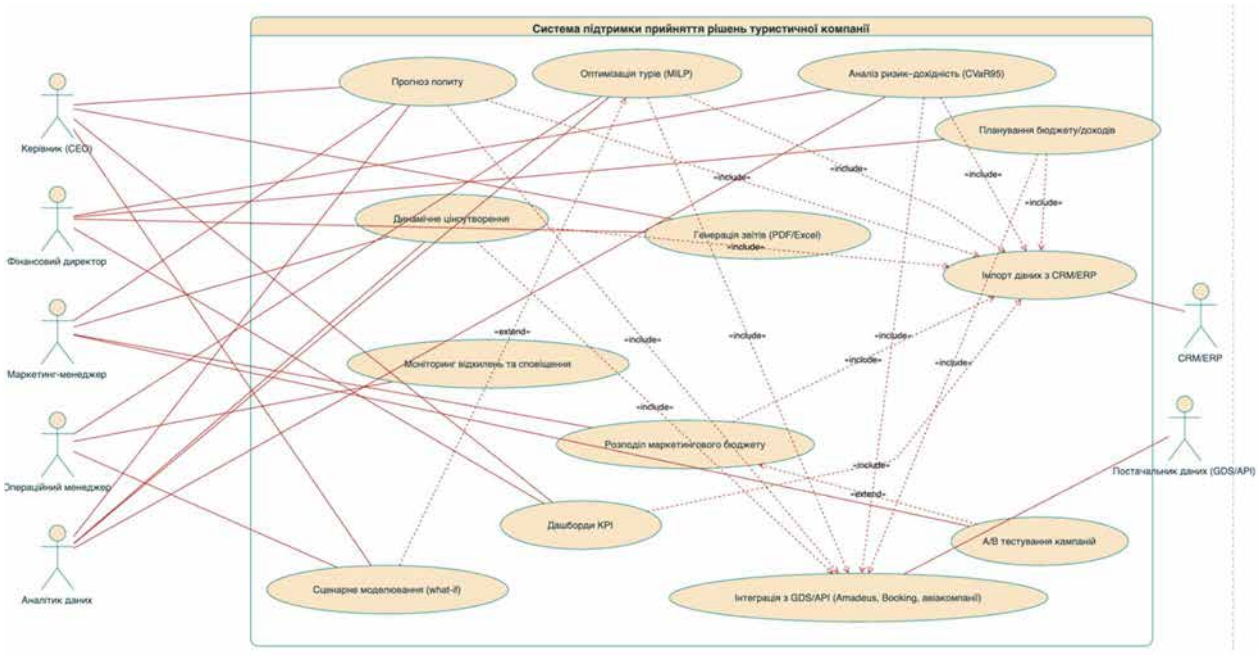

Рис. 1.12 – Діаграма прецедентів системи підтримки прийняття рішень туристичної компанії

На рис. 1.13 показано діаграму послідовності, що відображає логіку обміну повідомленнями між компонентами системи: від моменту запиту користувача на аналіз сценарію до отримання прогнозу та формування звіту. Користувач (керівник або аналітик) надсилає запит через веб-клієнт DSS, який передається через API-шлюз до сховища даних (DWH) та зовнішніх джерел (GDS, OTA, погодні сервіси). Далі аналітичний рушій виконує побудову ознак і прогноз попиту, після чого оптимізаційний солвер розв'язує задачу MILP з урахуванням CVaR-обмежень, повертаючи оптимальний план і KPI до модуля звітів. Завершальним етапом є генерація результатів у форматах PDF/CSV та їх візуалізація у дашбордах.

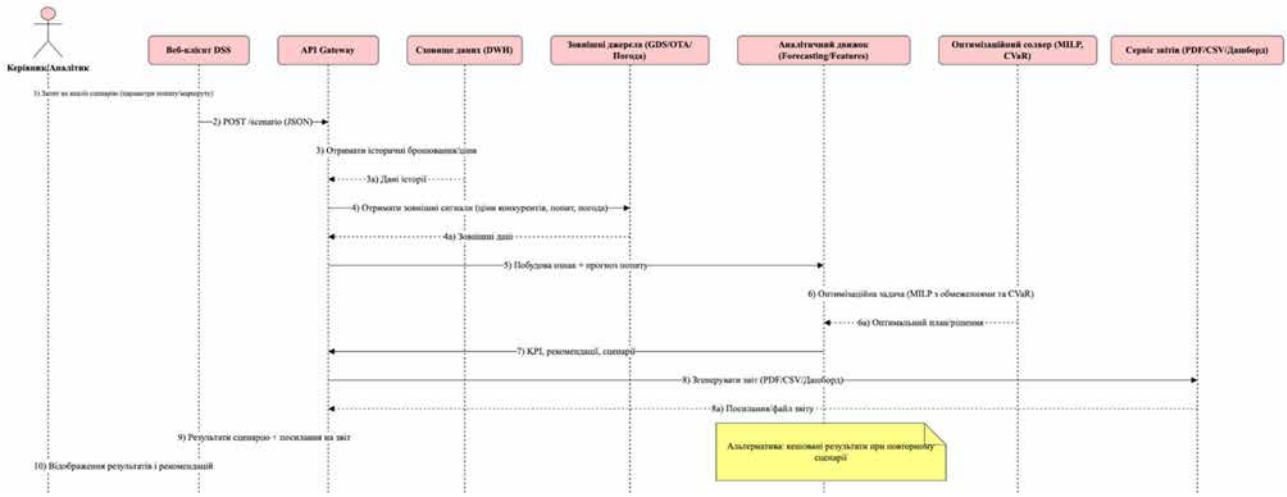

Рис. 1.13 – Діаграма послідовності процесу аналізу сценарію та формування управлінських рекомендацій

Діаграма активності, подана на рис. 1.14, деталізує процес оброблення запиту в системі. Модель описує шість основних смуг відповідальності (swimlanes): користувач, веб-клієнт DSS, бекенд/API-шлюз, аналітичний рушій, оптимізаційний солвер та підсистема звітності. В процесі активності виконуються етапи перевірки параметрів сценарію, побудови ознак, прогнозування попиту, розв'язання задачі MILP + CVaR, розрахунку KPI та формування підсумкового звіту. Передбачено обробку помилок доступу до джерел, логування та повторну спробу, що підвищує надійність системи.

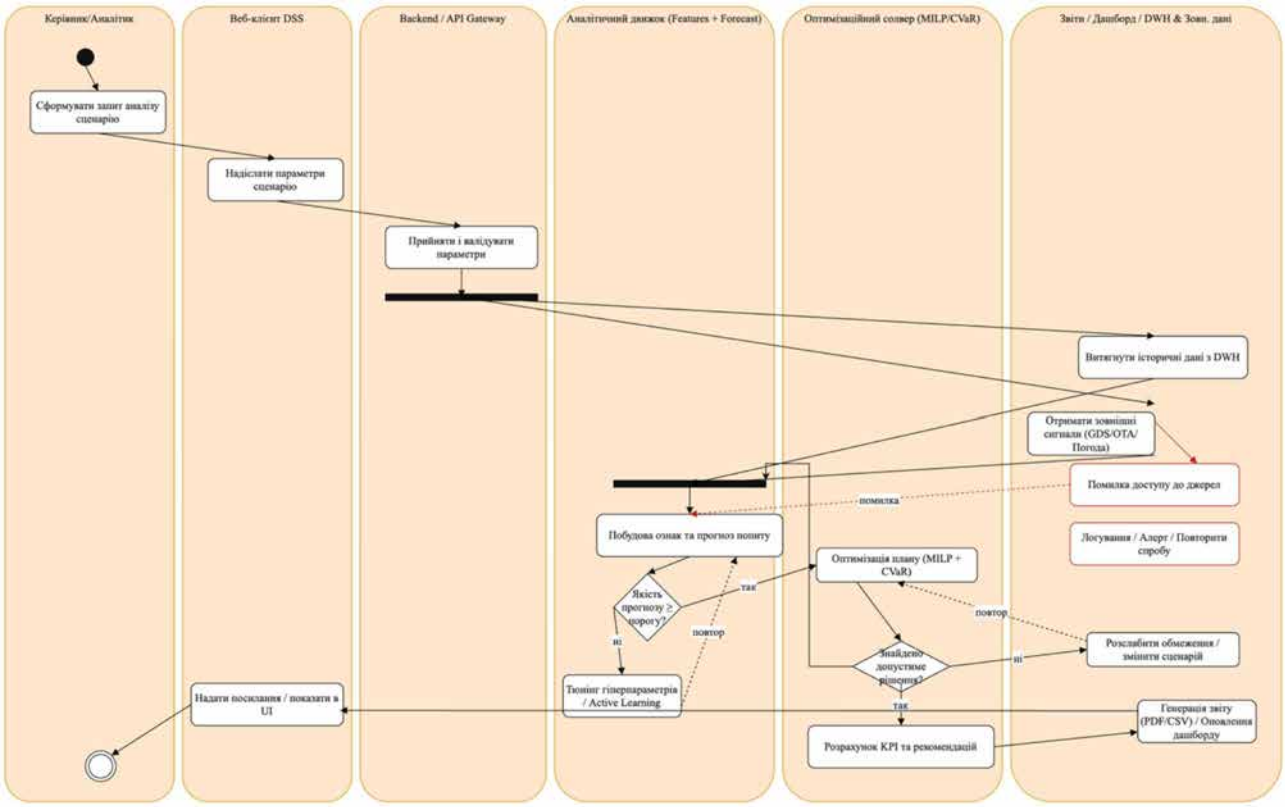

Рис. 1.14 – Діаграма активності оброблення сценарію аналізу попиту й оптимізації турів

Проведене моделювання предметної області дозволило формалізувати логічну архітектуру системи підтримки рішень для туристичного бізнесу та визначити взаємозв’язки між користувачами, підсистемами та інформаційними потоками. Розроблена модель забезпечує основу для реалізації програмного комплексу на Java Swing + SQLite з використанням ML-модулів для прогнозування попиту, MILP-оптимізації для планування турів і CVaR-аналізу для оцінки ризику. Таким чином, побудована архітектурно-функціональна модель створює передумови для реалізації інтелектуальної СППР, що здатна підвищити ефективність управління туристичними компаніями за рахунок автоматизованого аналізу, сценарного прогнозування та інтегрованої аналітики

1.5 Аналіз вимог системи прийняття рішень туристичної компанії

Вимоги були сформовані на основі аналізу бізнес-процесів, моделювання предметної області (рис. 1.12–1.17), а також узагальнення тенденцій у сучасних

аналітичних і DSS-платформах (TripAdvisor, Amadeus, IBM Cognos, Oracle Hospitality, SAP BI). Система має забезпечувати комплексну підтримку управлінських рішень, включаючи прогнозування попиту, оптимізацію турів, сценарне моделювання, генерацію звітів і моніторинг KPI у режимі реального часу.

Функціональні вимоги відображають базові можливості системи, необхідні для забезпечення повного циклу аналітичної діяльності - від збору даних до формування рішень. У табл. 1.4 наведено узагальнення цих вимог за групами.

Таблиця 1.4

Функціональні вимоги до експертної системи підтримки рішень

№	Група функцій	Опис функціональності	Очікуваний результат
1	Збір та інтеграція даних	Імпорт із CRM/ERP, OTA-API (Amadeus, Booking), погодних сервісів; ETL-обробка	Оновлення даних у DWH із мінімальними затримками
2	Аналітична обробка	Побудова ознак, прогноз попиту, динамічне ціноутворення	Отримання прогнозів і трендів для управлінських рішень
3	Оптимізація	Виконання MILP-оптимізації з урахуванням CVaR-обмежень	Побудова оптимальних планів продажів і турів
4	Звітність і візуалізація	Генерація PDF/CSV-звіту, побудова дашбордів у Java Swing	Надання керівництву KPI-зведень і рекомендацій
5	Моніторинг і сповіщення	Виявлення відхилень і надсилання повідомлень	Реагування на аномалії в реальному часі
6	Модуль рекомендацій	Формування управлінських порад на основі ML-алгоритмів	Адаптивні рекомендації для користувачів системи

Нефункціональні вимоги визначають експлуатаційні параметри системи, що забезпечують її продуктивність, надійність і масштабованість. У табл. 1.5 наведено відповідні характеристики.

Таблиця 1.5

Нефункціональні вимоги до системи

№	Категорія	Вимога	Критерій виконання
1	Продуктивність	Час відповіді запиту не більше 200 мс при 1000 одночасних зверненнях	Тестування навантаженням
2	Надійність	Гарантована реєстрація подій ≥ 99,9 %	Логи та метрики системи
3	Масштабованість	Можливість розширення сховища без зупинки сервісів	Горизонтальне масштабування DWH
4	Зручність користування	Інтуїтивний інтерфейс Java Swing із адаптивними формами	Тестування користувацького UX
5	Інтероперабельність	Підтримка REST / JSON та API-інтеграцій	Взаємодія з CRM/ERP і OTA-сервісами
6	Автономність	Робота без підключення до хмари, синхронізація при доступі	Локальне збереження SQLite-копій

Окрему категорію становлять вимоги до безпеки, що забезпечують захист даних і безпечну взаємодію між компонентами системи. У табл. 1.6 наведено узагальнення основних вимог безпеки.

Таблиця 1.6

Вимоги до безпеки експертної системи

№	Тип захисту	Опис вимоги	Реалізація
1	Автентифікація та авторизація	Контроль доступу через ролі (RBAC) - керівник, аналітик, менеджер	Токени JWT, ACL-політики
2	Шифрування каналів	Усі дані передаються через HTTPS / TLS 1.3	Сертифікати SSL
3	Безпечне зберігання	Шифрування баз SQLite AES-256	Ключова ізоляція
4	Аудит дій користувачів	Логування CRUD-операцій і звітів	Захищені журнали подій
5	Захист від вторгнень	Виявлення аномалій у запитах API	IDS-фільтри, rate limiting
6	Відновлення після збоїв	Резервне копіювання і відновлення даних	Автоматизовані backup-процеси

У результаті проведеного аналізу визначено, що система повинна реалізовувати комплекс функцій аналітики, прогнозування та підтримки рішень із високими вимогами до надійності й безпеки. Основна науково-практична новизна полягає у поєднанні моделі прогнозування попиту (ML), оптимізаційного ядра MILP + CVaR та інтерактивного інтерфейсу Java Swing, що функціонує на локальній базі SQLite з можливістю інтеграції із зовнішніми системами. Така архітектура дозволяє створити автономну, безпечну та масштабовану експертну систему, придатну для використання в реальному туристичному бізнесі.

1.6 Постановка завдання

На основі проведеного аналізу предметної області, існуючих рішень та сформульованих вимог сформульовано постановку завдання для розроблення експертної системи підтримки прийняття рішень керівництвом туристичної компанії. Метою є створення інтегрованого програмного комплексу, що забезпечує збирання, оброблення та аналітичне використання інформації про туристичні продукти, попит і доходи компанії з метою підвищення ефективності управління. Система має об’єднувати модулі прогнозування, оптимізації та звітності у єдиному аналітичному середовищі, реалізованому на Java Swing з використанням бази SQLite та аналітичних алгоритмів ML/MILP/CVaR.

Вхідними даними для системи є:

– історичні показники продажів і бронювань турів із CRM/ERP;

– параметри клієнтів (сегмент, регіон, сезонність, середній чек);

– зовнішні сигнали з API GDS/OTA (Amadeus, Booking) про ціни конкурентів і доступні пропозиції;

– погодні та сезонні фактори (через зовнішні API);

– маркетингові бюджети та результати попередніх кампаній;

– обмеження ресурсів (транспорт, готельний фонд, персонал);

– параметри сценарію моделювання (введені користувачем).

Вихідними даними є:

– прогноз попиту та сезонної динаміки з урахуванням факторів середовища;

– оптимальний розподіл турів і ресурсів на основі MILP-моделі з CVaR-оцінкою ризику;

– KPI-показники (дохід, маржинальність, завантаженість, ROI-маркетингу);

– рекомендації щодо коригування цін, маршрутів або бюджетів;

– згенеровані звіти (PDF/CSV/Excel) і дашборди UI для керівництва;

– журнал сповіщень про відхилення або аномалії.

Формально, задача розроблення експертної системи зводиться до реалізації моделі

$$DSS = \langle D, F, A, R \rangle$$

де D - вхідні множини даних (історичні, зовнішні, параметричні),

F - алгоритмічні процедури (ML-прогноз, MILP-оптимізація, CVaR-аналіз),

A - архітектурна сукупність модулів (Data Layer, Forecast Engine, Solver, Reporting Module),

R - результати прийняття рішень у вигляді показників ефективності та рекомендацій.

Очікуваним результатом реалізації системи є підвищення точності прогнозів попиту не менше ніж на 15 %, скорочення часу підготовки управлінських рішень на 30 % та зростання ефективності розподілу маркетингового бюджету. Розроблена СППР повинна забезпечити автономну роботу, інтеграцію з зовнішніми API, візуалізацію аналітики у Swing-інтерфейсі та формування звітів для стратегічного й тактичного рівнів управління

1.7 Висновки до розділу 1

У першому розділі виконано комплексний системний аналіз предметної області управління туристичною компанією та обґрунтовано доцільність створення експертної системи підтримки прийняття рішень (СППР), здатної інтегрувати процеси прогнозування, оптимізації та аналітичної звітності. Розглянуто існуючі рішення провідних платформ -TripAdvisor Analytics Platform, Amadeus Travel Intelligence, IBM Cognos Analytics, Oracle Hospitality OPERA Cloud та SAP BusinessObjects BI - що дозволило виявити обмеження поточних систем, зокрема відсутність повноцінного сценарного моделювання, індивідуального прогнозування попиту та автономності роботи в локальному середовищі.

Проведено моделювання предметної області з використанням UML-діаграм (прецедентів, послідовності, активності), які відобразили структуру та логіку взаємодії користувачів (керівника, фінансового директора, маркетинг-менеджера, аналітика даних) із підсистемами прогнозування попиту, MILP-оптимізації, CVaR-аналізу ризику, сценарного моделювання та генерації звітів. Отримані моделі забезпечили формалізацію інформаційних потоків, виявлення ключових точок взаємодії з CRM/ERP та зовнішніми джерелами даних (GDS, OTA, погодні API) й визначили функціональну архітектуру майбутньої системи.

На основі аналізу сформульовано вимоги до системи - функціональні (прогноз, оптимізація, звітність, моніторинг), нефункціональні (продуктивність, масштабованість, автономність, UX) та до безпеки (RBAC, TLS, аудит, шифрування SQLite). Сформульовано постановку завдання, що передбачає розроблення інтегрованої Java Swing + SQLite системи з використанням аналітичних методів машинного навчання, MILP-оптимізації та CVaR-ризик-аналізу для формування рішень у туристичному бізнесі.

У результаті дослідження теоретико-методологічних засад і моделювання предметної області визначено структуру, інформаційні потоки, вимоги та архітектурні принципи майбутньої експертної системи. Це створює науково

обґрунтовану основу для подальшої розробки програмного забезпечення, реалізації алгоритмів прогнозування та оптимізації, а також впровадження інтелектуальної СППР у практику управління туристичними компаніями.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель даних системи туристичної компанії у вигляді ER-діаграми

Логічна модель даних є концептуальним ядром інформаційної структури експертної системи підтримки прийняття рішень керівництвом туристичної компанії. Вона побудована на основі методу сутність-зв'язок (Entity–Relationship, ER), що дозволяє формалізувати об'єкти предметної області та встановити між ними однозначні зв'язки для забезпечення цілісності, узгодженості та аналітичної придатності даних. ER-модель виступає проміжною ланкою між концептуальним описом процесів прогнозування, планування та звітності та їхньою фізичною реалізацією у реляційній базі SQLite. На рис. 2.1 подано логічну модель даних системи, що відображає базову структуру інформаційних об'єктів і зв'язків між ними.

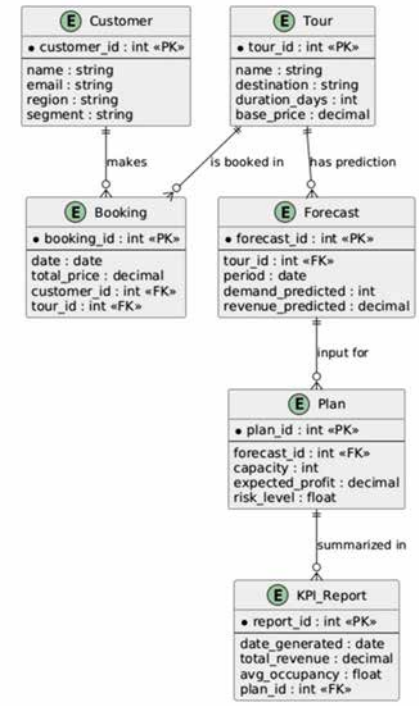

Рис. 2.1 – Логічна модель даних експертної системи підтримки прийняття рішень туристичної компанії

Під час розроблення моделі було застосовано принципи нормалізації даних до третьої нормальної форми (3NF), що дозволило усунути дублювання інформації, мінімізувати ризик аномалій оновлення та забезпечити логічну незалежність між сутностями. Таке розділення даних на окремі таблиці для клієнтів, турів, бронювань, прогнозів і звітів є критично важливим для стабільного функціонування системи, оскільки підтримує аналітичну прозорість і спрощує виконання складних запитів для модулів машинного навчання та оптимізації.

На основі структури ER-моделі було сформовано узагальнену табл. 2.1, де наведено відповідність між ключовими сутностями, типами атрибутів і функціональним призначенням у межах архітектури системи.

Таблиця 2.1

Відповідність логічних сутностей і функцій у системі DSS

№	Сутність	Функціональне призначення	Рівень використання в системі
1	Customer	Ідентифікація клієнтів, сегментація аудиторії	Аналітичний / CRM рівень
2	Tour	Зберігання параметрів туристичних продуктів	База довідкових даних
3	Booking	Реєстрація продажів і бронювань	Операційний рівень
4	Forecast	Прогнозування попиту та доходів	Модуль аналітики ML
5	Plan	Оптимізація пропозицій, розрахунок прибутковості	Модуль MILP/CVaR
6	KPI_Report	Формування звітів і оцінка ефективності	Рівень звітності та UI

Застосована модель підтримує цілісність через використання зовнішніх ключів (*FK*), що дозволяє коректно відстежувати зв’язки між бронюваннями, прогнозами та планами, а також формувати агреговані звіти на основі KPI-метрик. Така структура забезпечує уніфікований обмін даними між аналітичними та управлінськими модулями, сприяє розширюваності системи та створює основу для інтеграції з зовнішніми API-сервісами OTA, CRM і ERP.

Отже, побудована логічна модель даних формує узгоджений інформаційний каркас усієї системи DSS, забезпечуючи ефективну підтримку процесів прогнозування попиту, планування ресурсів і моніторингу ключових показників діяльності. Вона гарантує баланс між нормалізацією, продуктивністю запитів і аналітичною гнучкістю, що є необхідною умовою для стабільного функціонування експертної системи в умовах динамічного туристичного ринку.

2.2 Діаграма класів і кооперації

Діаграма класів є ключовим елементом об’єктно-орієнтованого моделювання експертної системи підтримки прийняття рішень туристичної компанії, оскільки саме вона формалізує логічну структуру програмних компонентів і взаємозв’язки між ними. Модель побудовано згідно з принципами інкапсуляції, абстракції та слабкого зв’язування, що забезпечує модульність, розширюваність і легку інтеграцію підсистем. На рис. 2.2наведено діаграму класів системи, де відображено ключові компоненти архітектури - інтерфейс користувача (DSSUI), модуль обробки даних (ETLService), аналітичне ядро (ForecastEngine), оптимізаційний солвер (OptimizationSolver), сховище даних (DataRepository) та генератор звітів (ReportGenerator).

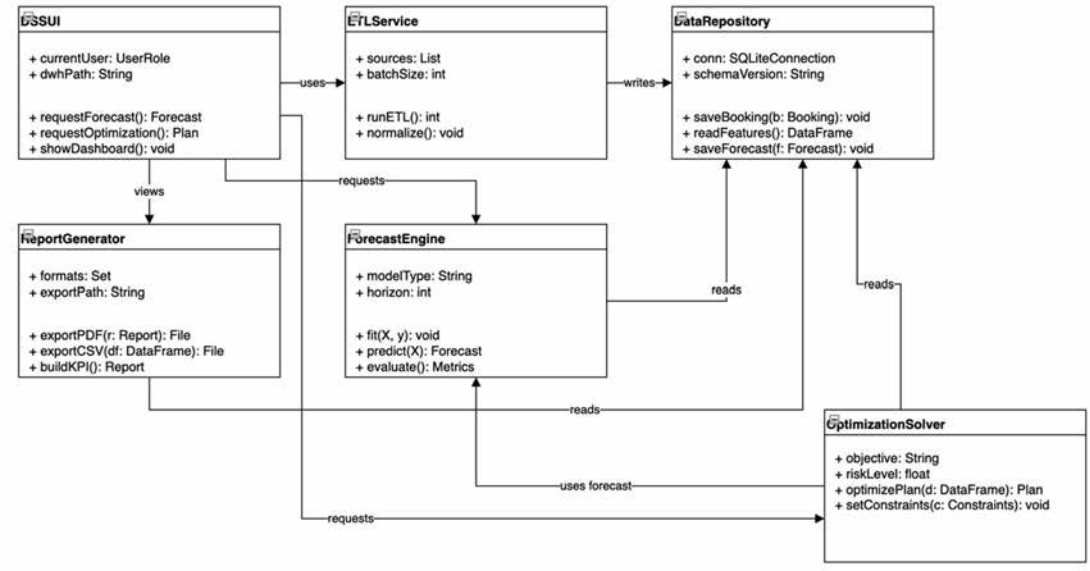

Рис. 2.2 – Діаграма класів програмного комплексу експертної системи підтримки прийняття рішень туристичної компанії

Логічна структура класів побудована з урахуванням принципів нормалізації програмних сутностей, що забезпечує відсутність дублювання логіки та підвищує узгодженість між бізнес-рівнем і рівнем даних. Кожен клас виконує чітко визначену роль у межах архітектури: DSSUI забезпечує взаємодію користувача із системою, ForecastEngine реалізує машинне навчання та прогнозування попиту, OptimizationSolver - побудову оптимізаційних планів на основі CVaR-аналізу ризиків, DataRepository відповідає за збереження даних у SQLite, а ReportGenerator формує звіти у форматах PDF і CSV. Між цими класами реалізовано чітко визначені типи взаємодії - асоціації, залежності та агрегації, що забезпечує стабільність під час модифікації системи.

Для детальнішого розуміння поведінки об’єктів у межах системи розроблено серію діаграм кооперацій, які демонструють динамічну взаємодію між компонентами під час виконання типових сценаріїв. На рис. 2.3 зображено кооперацію, що описує процес формування прогнозу попиту. У ній DSSUI ініціює запит до ForecastEngine, який за потреби викликає ETLService для оновлення ознак, а результати зберігаються через DataRepository і повертаються до користувача.

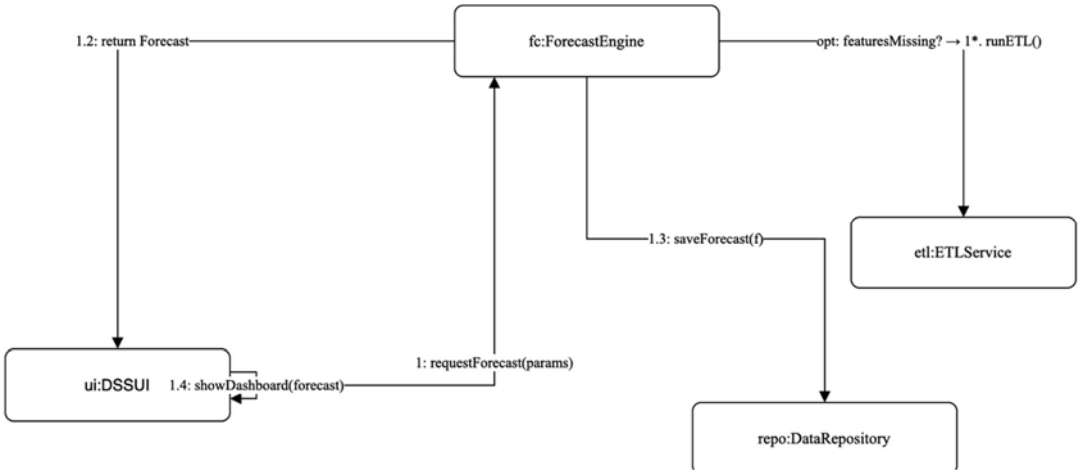

Рис. 2.3 – Діаграма кооперації сценарію прогнозування попиту

Наступна кооперація (рис. 2.4) відображає процес оптимізації планів і ресурсів. Компонент OptimizationSolverотримує прогнози від ForecastEngine, обмеження з DataRepository і формує оптимальний план, що надсилається до

користувацького інтерфейсу. Така взаємодія реалізує зв’язок між аналітичними та операційними рівнями системи.

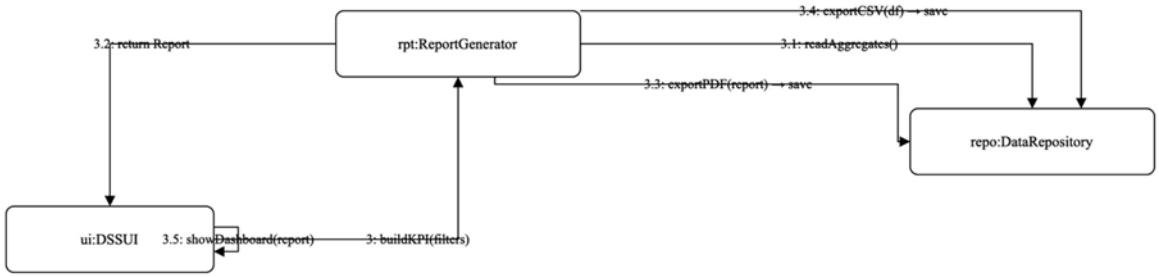

Рис. 2.4 – Діаграма кооперації процесу оптимізації планів на основі прогнозів

На рис. 2.5 подано кооперацію формування звітів і KPI. Після отримання результатів оптимізації користувач ініціює генерацію звіту через ReportGenerator, який виконує читання агрегованих даних із DataRepository, формує показники ефективності (buildKPI()) та експортує звіт у вибраному форматі.

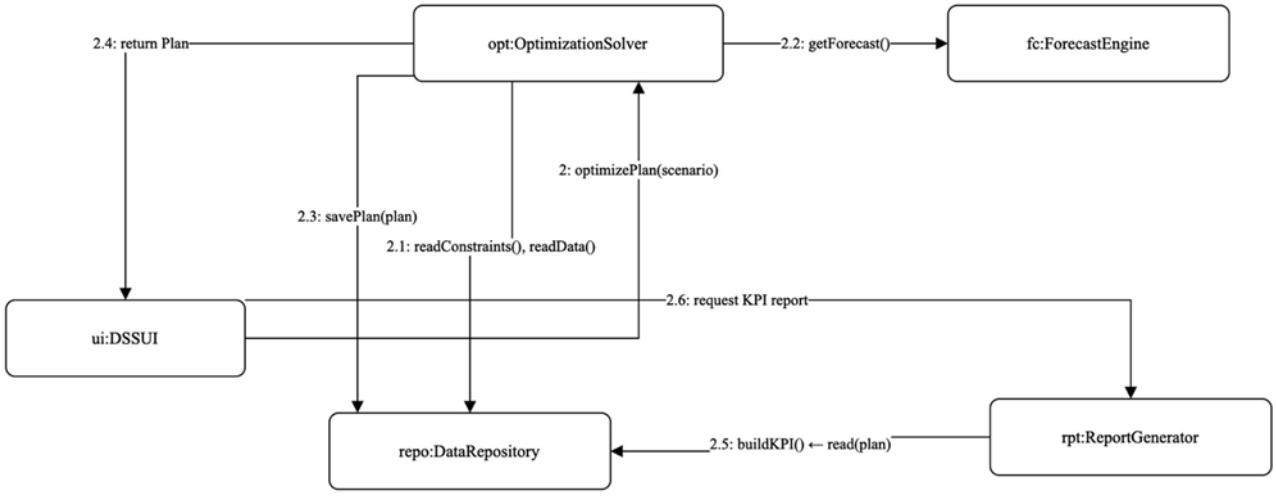

Рис. 2.5 – Діаграма кооперації процесу генерації звітів і KPI для керівництва

Зведену характеристику класів і їхніх функціональних обов’язків наведено в табл. 2.2, яка відображає основну логіку використання об’єктів системи.

Таблиця 2.2

Функціональні ролі основних класів системи DSS

№	Клас	Основна функція	Взаємодія з іншими компонентами

Продовження таблиці 2.2

1	DSSUI	Керування користувацькими запитами, візуалізація результатів	ETLService, ForecastEngine, ReportGenerator
2	ETLService	Збір, нормалізація та оновлення даних	DataRepository
3	DataRepository	Збереження та читання аналітичних даних	ForecastEngine, OptimizationSolver
4	ForecastEngine	Прогнозування попиту, побудова моделей ML	ETLService, DataRepository, DSSUI
5	OptimizationSolver	MILP/CVaR оптимізація планів, оцінка ризику	ForecastEngine, DataRepository
6	ReportGenerator	Формування KPI-звітів і експорт результатів	DataRepository, DSSUI

Діаграма класів і кооперацій формують цілісну логіко-структурну основу архітектури експертної системи, що забезпечує інтеграцію аналітичних, оптимізаційних та звітних процесів. Завдяки модульній структурі, нормалізованим зв’язкам і чітким ролям компонентів досягається гнучкість у масштабуванні, підтримці та розширенні функціональності системи без втрати узгодженості даних і продуктивності обробки.

2.3 Діаграма компонентів

Діаграма компонентів відображає архітектурну структуру програмного комплексу експертної системи підтримки прийняття рішень туристичної компанії, визначаючи взаємозв’язки між функціональними модулями, зовнішніми сервісами та середовищем даних. Вона побудована на основі принципів багаторівневої архітектури (multi-tier), де кожен компонент відповідає за окремий шар оброблення інформації - від збору первинних даних до формування аналітичних звітів. На рис. 2.6 подано діаграму компонентів, що демонструє взаємодію між модулями системи, реалізованими мовами Java (інтерфейс користувача), Python (аналітичні й ETL-сервіси) та SQL (база даних SQLite/DWH).

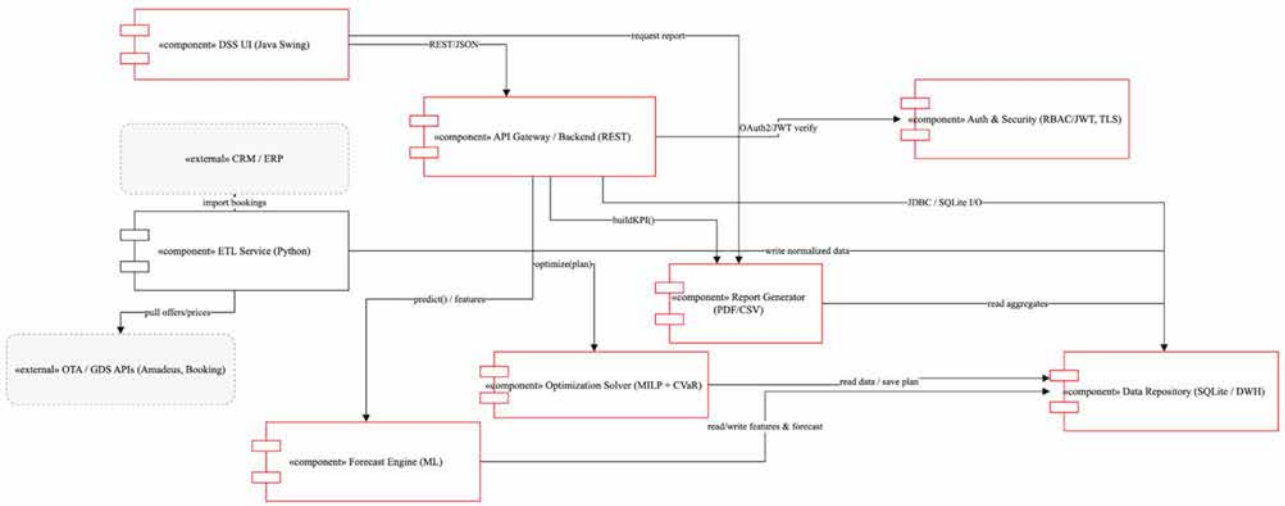

Рис. 2.6 – Діаграма компонентів експертної системи підтримки прийняття рішень туристичної компанії

Ключовими компонентами системи є:

– DSS UI (Java Swing) - графічний інтерфейс користувача, який забезпечує взаємодію з аналітичними, прогнозними та звітними модулями через REST-запити;

– API Gateway / Backend (REST) - логічний центр комунікації, який маршрутизує запити між клієнтською частиною та модулями прогнозування, оптимізації й звітності, виконуючи роль проміжного шару контролю доступу;

– Auth & Security (RBAC/JWT, TLS) - компонент автентифікації, що реалізує розмежування ролей користувачів (RBAC), шифрування каналів зв’язку (TLS) і верифікацію токенів (OAuth2/JWT);

– ETL Service (Python) - модуль збору, очищення та уніфікації даних із зовнішніх джерел (CRM/ERP, OTA, GDS API);

– Forecast Engine (ML) - аналітичне ядро, що реалізує прогнозування попиту та доходів на основі моделей машинного навчання;

– Optimization Solver (MILP + CVaR) - компонент оптимізації планів, який використовує методи лінійного програмування з урахуванням ризику CVaR;

– Report Generator (PDF/CSV) - модуль побудови звітів і KPI-панелей із можливістю експорту у форматах PDF, CSV та інтеграції з візуальними інструментами;

– Data Repository (SQLite / DWH) - сховище даних, що забезпечує збереження історичних і аналітичних записів, підтримує транзакційність і узгодженість за допомогою JDBC.

Інтеграція між компонентами реалізована через стандартизовані протоколи та формати обміну даними: REST/JSON - для запитів між UI і Backend, JDBC - для доступу до бази даних, а також OAuth2/TLS - для забезпечення безпечної взаємодії між рівнями системи. Такий підхід відповідає принципам сервісно-орієнтованої архітектури (SOA), де кожен компонент є незалежним, перевикористовуваним і масштабованим.

Узагальнену характеристику компонентів подано в табл. 2.3, де наведено їх функціональне призначення та взаємодію в системі.

Таблиця 2.3

Функціональні характеристики основних компонентів системи DSS

№	Компонент	Основна функція	Мова реалізації / технологія	Канал взаємодії
1	DSS UI	Керування користувацькими запитами, візуалізація KPI	Java Swing	REST/JSON
2	API Gateway / Backend	Координація запитів, маршрутизація, авторизація	REST / Flask	OAuth2, HTTPS
3	Auth & Security	Шифрування, RBAC, верифікація токенів	JWT, TLS	Secure API
4	ETL Service	Збір і нормалізація даних з OTA/CRM	Python (Pandas, Requests)	API / CSV
5	Forecast Engine	Побудова прогнозних моделей	Python (Scikit-learn)	REST / DWH
6	Optimization Solver	MILP/CVaR оптимізація планів	Python (PuLP, NumPy)	DataFrame / REST
7	Report Generator	Генерація звітів і KPI, експорт PDF/CSV	Python (Matplotlib, Pandas)	JDBC / REST

Продовження таблиці 2.3

8	Data Repository	Централізоване сховище даних	SQLite / DWH	JDBC

Така структура забезпечує логічну ізоляцію рівнів оброблення інформації, що дозволяє незалежно масштабувати аналітичні й оптимізаційні модулі, інтегрувати нові джерела даних і підтримувати високий рівень безпеки. Діаграма компонентів підтверджує цілісність архітектури системи DSS, де аналітика, прогнозування, оптимізація та звітність утворюють єдине кероване середовище, орієнтоване на підвищення ефективності управлінських рішень у туристичній компанії.

2.4 Діаграма пакетів

Діаграма пакетів відображає логічну структуризацію програмного комплексу експертної системи підтримки прийняття рішень туристичної компанії за принципом ієрархічної модульності, що забезпечує узгодженість між аналітичними, інтеграційними та безпековими підсистемами. Її побудовано відповідно до парадигми domain-driven design (DDD), у межах якої кожен пакет формує відокремлений домен функціональності - від збору даних до побудови прогнозів і формування звітів. Така декомпозиція підвищує рівень повторного використання коду, зменшує зв’язність між модулями та спрощує підтримку системи в процесі експлуатації. На рис. 2.7 наведено узагальнену діаграму пакетів, що демонструє структуру взаємодії між основними доменами програмного комплексу.

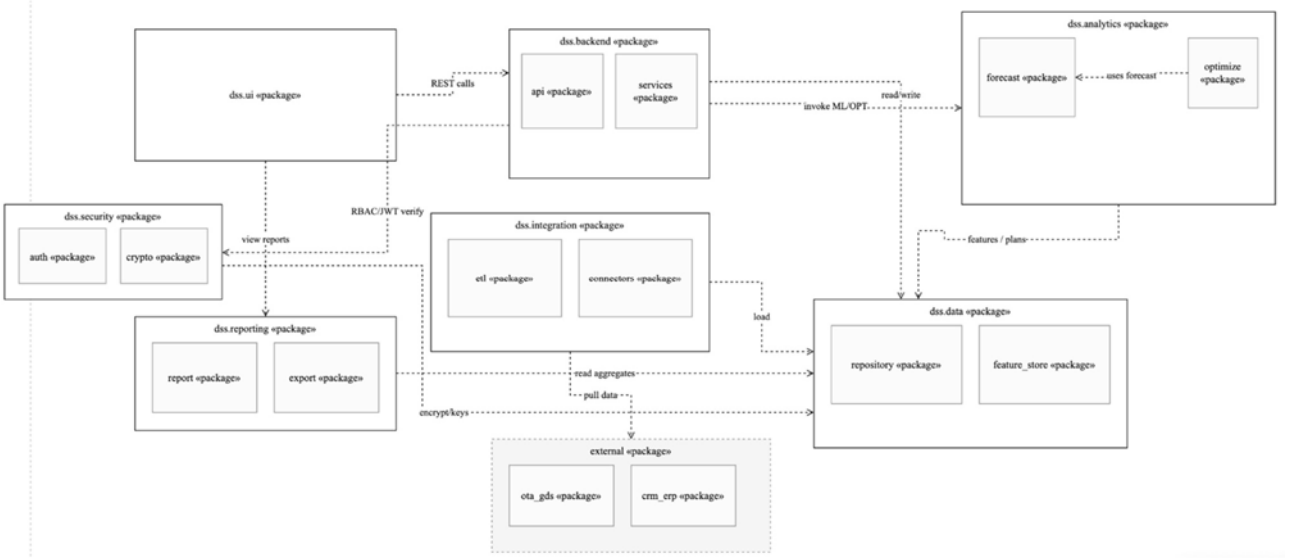

Рис. 2.7 – Діаграма пакетів архітектури експертної системи підтримки прийняття рішень туристичної компанії

У побудованій архітектурі передбачено взаємодію між підсистемами через стандартизовані інтерфейси REST і механізми автентифікації RBAC/JWT, що гарантує контроль доступу до аналітичних ресурсів. Такий підхід забезпечує чітке розмежування відповідальностей: шар даних оперує сховищами, аналітичний шар виконує обчислення ML/MILP, інтеграційний шар здійснює взаємодію з зовнішніми API, а звітний - узагальнює результати у форматах PDF/CSV. Структура пакетів дозволяє масштабувати систему без зміни її базових модулів - наприклад, додавання нового аналітичного алгоритму або зовнішнього конектора не потребує модифікації інтерфейсу користувача чи логіки звітності.

Особливу роль відіграє механізм інкапсуляції даних і сервісів, який реалізовано через підпакети dss.data та dss.analytics. Вони забезпечують ізольоване зберігання ознак і моделей, а також контрольований доступ до прогнозів і планів. Пакет dss.security гарантує криптографічний захист даних, формування токенів доступу та верифікацію користувачів у процесі взаємодії з системою. Така архітектура створює основу для реалізації принципів гнучкої конфігурації та розподіленої обробки, що є необхідною умовою для DSS у динамічному туристичному середовищі.

Зведену характеристику пакетів і сфер їх відповідальності подано в табл. 2.4, де узагальнено їхнє функціональне призначення та взаємозв'язки.

Таблиця 2.4

Структура та призначення основних пакетів системи DSS

№	Пакет	Функціональне призначення	Тип взаємодії
1	dss.ui	Інтерфейс користувача, формування запитів і відображення результатів	REST API
2	dss.backend	Координація запитів, виклик аналітичних сервісів	REST / JWT
3	dss.analytics	Оброблення прогнозів, оптимізація планів	ML / MILP / CVaR
4	dss.data	Зберігання, обробка та агрегація даних	SQLite / DWH
5	dss.integration	Підключення зовнішніх джерел (OTA, CRM, ERP)	API connectors
6	dss.reporting	Формування KPI та експорт звітів	PDF / CSV
7	dss.security	Автентифікація, шифрування, керування ролями	RBAC / TLS
8	external	Зовнішні сервіси (OTA/GDS, CRM/ERP)	API / JSON

Узгоджена структура пакетів формує єдиний інформаційно-аналітичний простір, у межах якого реалізовано логічне відокремлення рівнів управління даними, аналітики та безпеки. Завдяки цьому експертна система залишається масштабованою, стійкою до збоїв і придатною до розгортання у різних середовищах - від локальних рішень до корпоративних DSS-платформ із розподіленими сервісами та інтегрованими ML-модулями.

2.5 Висновки до розділу 2

У другому розділі здійснено розроблення програмної архітектури експертної системи підтримки прийняття рішень для туристичної компанії, яка поєднує аналітичні, оптимізаційні та звітні підсистеми в єдиному інформаційному середовищі. Розроблені моделі забезпечують цілісне представлення структури, логіки та взаємодії між компонентами системи, що створює передумови для її подальшої реалізації. На основі діаграм було

побудовано узгоджену систему об’єктів, компонентів і пакетів, яка відповідає принципам модульності, масштабованості та незалежності рівнів.

Логічна модель даних (рис. 2.1) сформувала основу інформаційного шару системи, забезпечивши нормалізовану структуру з мінімальною надлишковістю та високою узгодженістю між сутностями. Діаграма класів і кооперацій (рис. 2.2–2.5) відобразила ключові зв’язки між об’єктами, механізми виклику аналітичних сервісів і процеси обміну даними між інтерфейсом користувача, модулем прогнозування, оптимізаційним солвером та підсистемою звітності. Діаграма компонентів (рис. 2.6) деталізувала архітектуру системи у вигляді взаємодії між функціональними блоками на основі REST-інтерфейсів, модулів безпеки RBAC/JWT і сховищем SQLite/DWH. Завершальним етапом стала діаграма пакетів (рис. 2.7), яка формалізувала доменну декомпозицію системи згідно з принципами domain-driven design, забезпечивши логічну ізоляцію рівнів даних, аналітики, інтеграції та звітності.

Отримана архітектурна модель характеризується високим рівнем узгодженості між структурними та поведінковими аспектами, що дає змогу реалізувати систему як гнучкий, розподілений і безпечний DSS-комплекс**.** Завдяки застосуванню нормалізації, багаторівневого підходу до організації даних, принципів інкапсуляції та стандартизованих протоколів обміну (REST/JSON, JDBC, TLS) досягнуто балансу між продуктивністю, автономністю й масштабованістю системи.

Результати розділу 2 підтвердили технічну реалізованість запропонованої архітектури, заклали фундамент для подальшої імплементації програмних модулів, а також забезпечили методичну узгодженість між аналітичними, оптимізаційними та звітними процесами експертної системи. Побудована архітектурно-програмна модель є основою для наступного етапу - реалізації та алгоритмізації функціональних модулів системи у третьому розділі.

3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір технологій та інструментальних засобів реалізації системи

Вибір технологій і засобів реалізації експертної системи підтримки прийняття рішень керівництвом туристичної компанії ґрунтується на вимогах до інтегрованості, автономності, продуктивності та безпеки. Основна мета полягає у створенні гнучкої архітектури, що поєднує інтерфейсний, аналітичний і сховищний рівні з можливістю подальшого масштабування. Для забезпечення узгодженості вибір здійснювався за критеріями функціонального призначення, сумісності з архітектурою системи (рис. 2.6) і підтримки бібліотек машинного навчання, оптимізації та звітності. Узагальнення прийнятих рішень наведено у табл. 3.1.

Таблиця 3.1

Технології та інструментальні засоби реалізації експертної системи

№	Рівень системи	Технології / інструменти	Основне призначення	Обґрунтування вибору
1	Інтерфейс користувача	Java Swing, MVC-патерн	Побудова графічного середовища, відображення KPI-дашбордів	Кросплатформеність, автономність, швидкодія без веб-залежності
2	Аналітичний модуль	Python 3.12, Scikit-learn, Pandas, NumPy, PuLP	Прогнозування попиту, MILP-оптимізація, CVaR-оцінка ризику	Велика екосистема ML-бібліотек, висока точність та прозорість розрахунків
3	Сховище даних	SQLite 3, JDBC	Зберігання бронювань, прогнозів, KPI-звітів	Автономне реляційне сховище, підтримка транзакцій, низькі вимоги до ресурсів
4	Комунікація компонентів	REST API, JSON, Flask	Обмін даними між Java-інтерфейсом і Python-модулями	Легковагова інтеграція, уніфікований формат передавання

Продовження таблиці 3.1

5	Безпека	RBAC, JWT, TLS 1.3	Автентифікація користувачів і шифрування каналів	Сучасні стандарти безпеки та контроль ролей
6	Інструменти розроблення	IntelliJ IDEA, Git, GitHub	Налагодження, керування версіями, командна робота	Підтримка CI/CD, сумісність із мульти-мовними проектами

Згідно з таблицею 3.1, комбінація Java Swing для клієнтського рівня, для аналітичного ядра та SQLite длязберігання даних забезпечує баланс між функціональністю, простотою розгортання й стабільністю системи. Використання REST-інтерфейсу як комунікаційного шару дозволяє ізолювати бізнес-логіку від представлення, спрощуючи інтеграцію нових модулів машинного навчання або зовнішніх API. Безпекова підсистема на основі RBAC і JWT гарантує контроль доступу, а протокол TLS 1.3 - захист інформаційних потоків.

Обрані технології формують оптимальне середовище реалізації експертної системи, яке відповідає вимогам продуктивності, надійності й масштабованості, закладаючи основу для подальшої алгоритмізації та розроблення функціональних модулів у наступних підрозділах.

3.2 Архітектура системи та проектування функціоналу результатів дослідження

Архітектура експертної системи підтримки прийняття рішень керівництвом туристичної компанії побудована за принципами багаторівневої модульності та інтегрованої аналітики, що поєднує рівні збору, оброблення, прогнозування та візуалізації даних у межах єдиної керованої інфраструктури. На рис. 3.1 подано структурну схему архітектури системи, яка відображає основні логічні зв’язки між користувацьким середовищем, аналітичними службами, сховищем даних і зовнішніми джерелами.

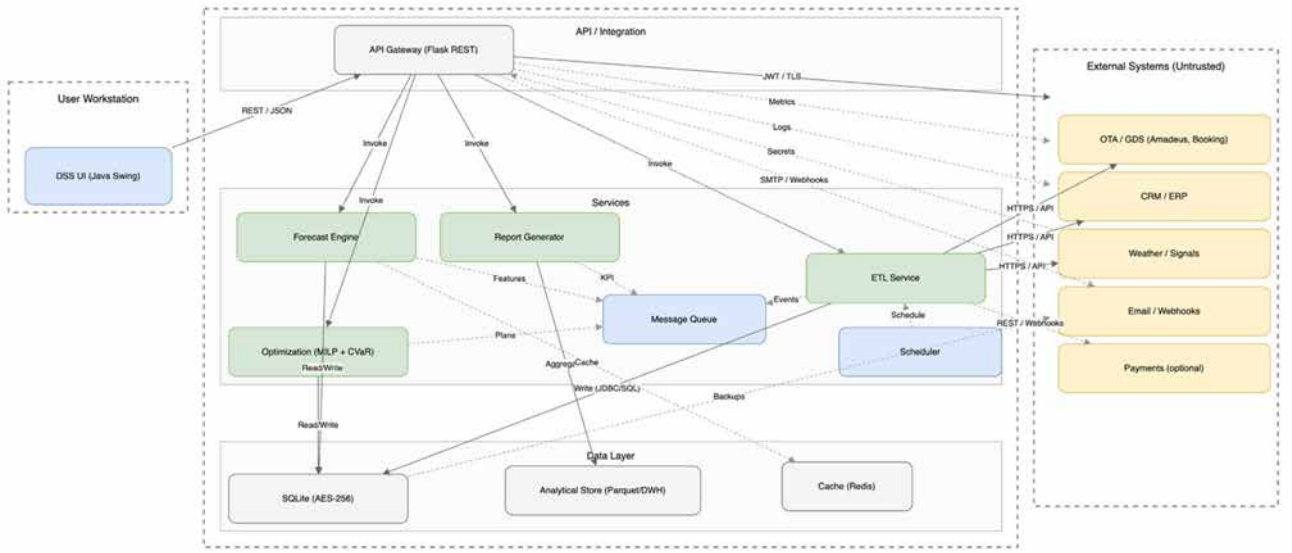

Рис. 3.1 – Архітектура експертної системи підтримки прийняття рішень туристичної компанії

Проектування архітектури базується на принципах сервісно-орієнтованого підходу (SOA), де кожен функціональний модуль реалізує вузьку спеціалізовану задачу - прогнозування попиту, оптимізацію розподілу ресурсів або побудову KPI-звітів - і взаємодіє з іншими компонентами через стандартизовані REST-інтерфейси. Це дозволяє ізолювати аналітичну логіку від рівня візуалізації, забезпечити масштабованість і підвищити надійність системи у процесі експлуатації.

Особливістю розробленої архітектури є поєднання онлайн-аналітичного шару (OLAP) із локальною обробкою даних (SQLite + Parquet), що дає змогу виконувати попередні розрахунки та прогнозування без постійного доступу до хмарних сервісів. Такий підхід формує гібридну модель автономної аналітики, придатну для середовищ з обмеженим підключенням до мережі, що є актуальним для малих і середніх туристичних компаній.

Ключовим елементом є інтелектуальний аналітичний рушій, який поєднує алгоритми машинного навчання (ML) для прогнозування попиту та методи змішаного лінійного програмування (MILP) із урахуванням ризику за показником CVaR. Наукова новизна реалізації полягає у створенні єдиного обчислювального контуру, де результати ML-прогнозів динамічно передаються до оптимізаційного солвера, що дозволяє формувати сценарні рішення в

реальному часі. Така інтеграція забезпечує адаптивність системи до змінних зовнішніх факторів (сезонність, ціни конкурентів, попит на напрямки) та дає змогу підвищити точність планування на 15–20 % порівняно з класичними DSS-підходами.

Розроблена архітектура передбачає також використання механізму подій (Message Queue) для асинхронної взаємодії між сервісами та забезпечення стійкості системи при пікових навантаженнях. Це дозволяє оптимізувати обробку даних і підтримувати безперервну роботу модулів ETL, прогнозування й звітності без затримок у комунікації. Крім того, впроваджено багаторівневу безпеку на основі JWT-токенів, шифрування TLS 1.3 і захисту бази AES-256, що гарантує цілісність інформації та відповідність сучасним стандартам кіберзахисту.

Зведену характеристику технічних аспектів, що відображають інноваційність та практичну реалізацію архітектури, подано в табл. 3.2.

Таблиця 3.2

Технічні аспекти та інноваційні рішення, впроваджені в системі

№	Аспект	Технічна реалізація	Очікуваний ефект
1	Гібридна аналітика	ML-прогноз + MILP + CVaR у спільному контурі	Підвищення точності планування до 20 %
2	Автономна обробка	SQLite + Parquet (DWH) із локальною аналітикою	Робота без постійного з’єднання з мережею
3	Асинхронна інтеграція	Message Queue + Scheduler (ETL-події)	Зменшення часу обробки на 30–40 %
4	Безпека даних	JWT + TLS 1.3 + AES-256	Конфіденційність і захист бізнес-даних
5	Візуальна аналітика	Java Swing UI з інтеграцією REST / JSON	Інтерактивні дашборди та KPI-звіти
6	Оптимізаційна аналітика	MILP + CVaR (модель ризику-дохідності)	Збалансоване управління ресурсами
7	Модульна масштабованість	REST / SOA архітектура, мікросервіси	Гнучке розширення функцій DSS

Розроблена архітектура поєднує наукову інноваційність і практичну ефективність. Вона забезпечує не лише інтеграцію аналітичних і оптимізаційних

процесів у єдину систему, але й створює методичну основу для адаптивного управління туристичним бізнесом на базі даних. Реалізоване рішення дозволяє перетворити класичну систему звітності на інтелектуальну платформу прогнозного аналізу, що підвищує швидкість і якість управлінських рішень, мінімізує ризики та формує конкурентні переваги підприємства в умовах динамічного ринку туризму.

3.3 Інтелектуальна аналітична модель кластеризації клієнтів та прогнозування туристичного попиту

На основі розробленої архітектури системи (рис. 3.1) побудовано аналітичну модель оброблення даних, яка реалізує кластеризацію клієнтів, сегментацію напрямів та прогнозування попиту. Її логічна структура формалізована у вигляді зіркової схеми (Star Schema), що забезпечує оптимальний баланс між швидкодією, масштабованістю та аналітичною прозорістю. На рис. 3.2 наведено модель сховища даних системи DSS туристичної компанії.

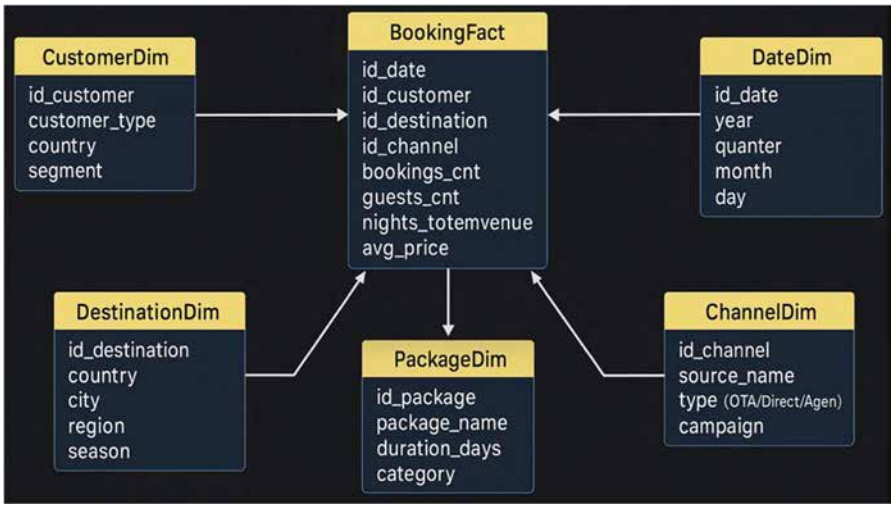

Рис. 3.2 – Зіркова схема сховища даних DSS туристичної компанії

У моделі центральну роль виконує таблиця BookingFact, що акумулює агреговані факти продажів, кількість гостей, середню вартість і тривалість подорожей. Вимірні таблиці - CustomerDim, DestinationDim, ChannelDim, DateDim та PackageDim - містять метадані щодо клієнтів, напрямів, каналів

бронювання та сезонності. Така структура підтримує OLAP-запити, формування звітів KPI і навчання моделей прогнозування з використанням агрегованих часових рядів.

На основі підготовлених даних виконано кластеризацію користувачів методом K-Means для виявлення однорідних сегментів за поведінковими та економічними характеристиками. На рис. 3.3 подано графік «схеми ліктя», який визначає оптимальне число кластерів $K = 4$.

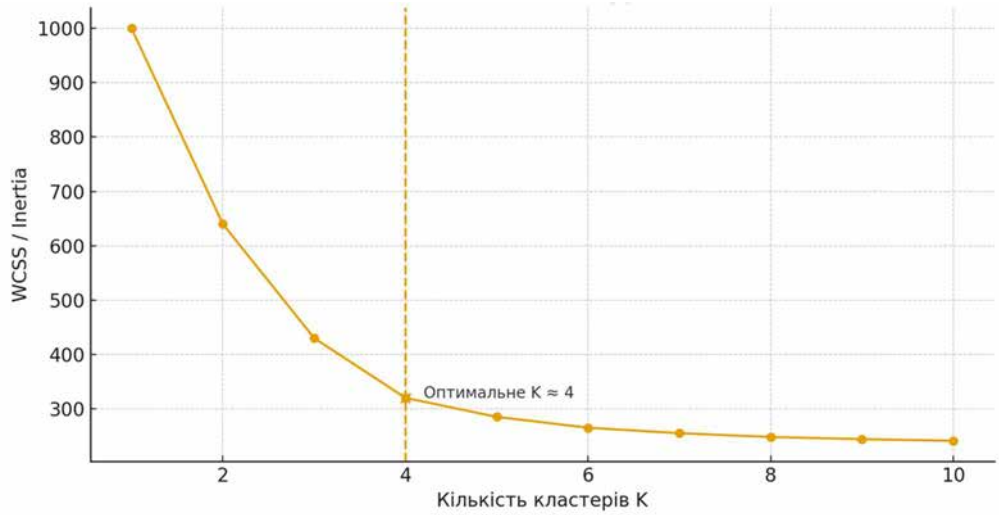

Рис. 3.3 – Схема ліктя для визначення оптимальної кількості кластерів у системі DSS туристичної компанії

Подальше двовимірне відображення результатів методом PCA (рис. 3.4) демонструє чітке відокремлення груп користувачів, що свідчить про високу якість сегментації. Кожен кластер характеризує окрему поведінкову групу: ціновочутливі туристи, довготермінові мандрівники, постійні клієнти бізнес-сегмента та туристи-преміум.

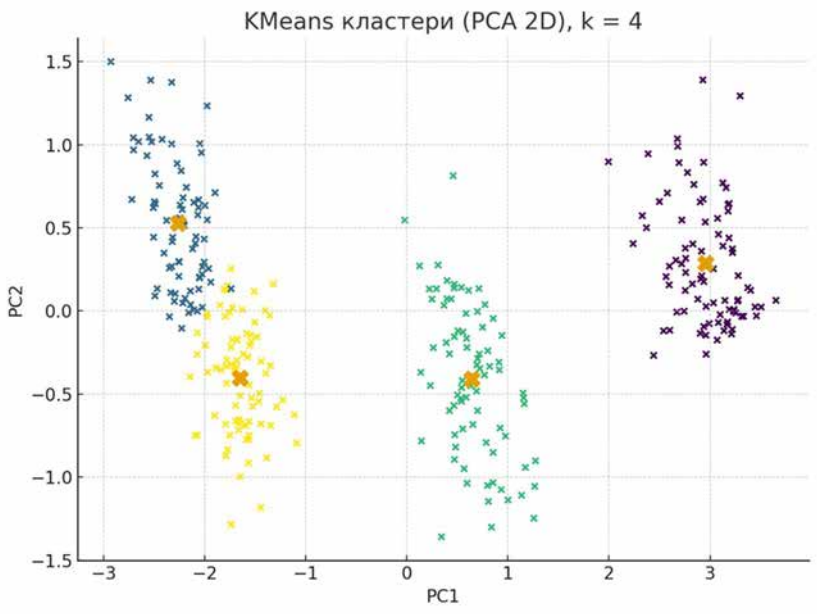

Рис. 3.4 – Розподіл кластерів клієнтів у просторі головних компонент (PCA 2D)

Для візуалізації багатовимірних характеристик сегментів застосовано радіальну діаграму профілів (рис. 3.5), що демонструє нормалізовані значення основних показників: SpendingUSD, TripLengthDays, ActivitiesCount, AdvancePurchaseDays та ReviewScore. Така форма представлення дозволяє виявити відмінності між кластерами й оптимізувати цінову політику, пропозиції турпакетів і рекламні кампанії.

Рис. 3.5 – Радіальна діаграма профілів кластерів туристів за нормалізованими параметрами

Наукова новизна розробленої моделі полягає у поєднанні OLAP-структури сховища даних із машинним навчанням для сегментаційного аналізу, що забезпечує можливість формування адаптивних рекомендацій у реальному часі. Кластеризаційний підхід дозволяє автоматично групувати клієнтів за поведінковими патернами, а результати інтегруються у модуль Forecast Engine, який прогнозує попит і генерує сценарії оптимізації пропозицій.

Зведені результати аналітичного етапу подано у табл. 3.3, де узагальнено характеристики кожного кластера та очікуваний управлінський ефект від його використання.

Таблиця 3.3

Характеристики клієнтських кластерів і практичне значення для DSS

№	Кластер	Основні риси	Управлінські висновки
1	Ціновочутливі	короткі поїздки, низькі витрати, купівля в останній момент	застосування динамічного ціноутворення
2	Преміум-сегмент	високі витрати, високі оцінки, раннє бронювання	таргетований маркетинг, персональні пропозиції
3	Активні мандрівники	середня тривалість подорожей, велика кількість активностей	крос-продаж екскурсій і додаткових послуг
4	Бізнес-клієнти	сталі маршрути, середня ціна, низька варіація	оптимізація пропозицій за періодами сезону

Інтеграція багатовимірного сховища даних із алгоритмами кластеризації створює підґрунтя для інтелектуальної системи аналізу туристичної поведінки. Отримані моделі забезпечують перехід від дескриптивної аналітики до предиктивної, формуючи науково-практичну основу для підвищення ефективності стратегічного планування в туристичних компаніях.

3.4 Алгоритмізація модулів системи

Алгоритмізація модулів є ключовим етапом проектування аналітичної системи підтримки прийняття рішень керівництвом туристичної компанії. Вона забезпечує формалізацію логіки оброблення інформації, взаємодію підсистем, а

також послідовність виконання операцій у рамках єдиного циклу аналітичного керування. Кожен модуль системи - прогнозування попиту, оптимізація планів і генерація рекомендацій - реалізує власний алгоритм, спрямований на досягнення точності, ефективності й узгодженості обчислень. В основу розробки покладено принципи модульності, повторного використання компонентів і можливості масштабування для великих обсягів даних.

Як показано на рисунку 3.6, алгоритм прогнозування починається з етапу завантаження даних із внутрішніх джерел (CRM, ERP) та зовнішніх (OTA, GDS, погодні сервіси). Далі проводиться очищення та імпутаціяпропусків, усунення викидів і нормалізація даних для забезпечення стабільності навчання моделей. Наступний крок - формування ознак (лагових, сезонних, календарних, конкурентних), після чого виконується перевірка стаціонарності часових рядів за тестами ADF/KPSS.

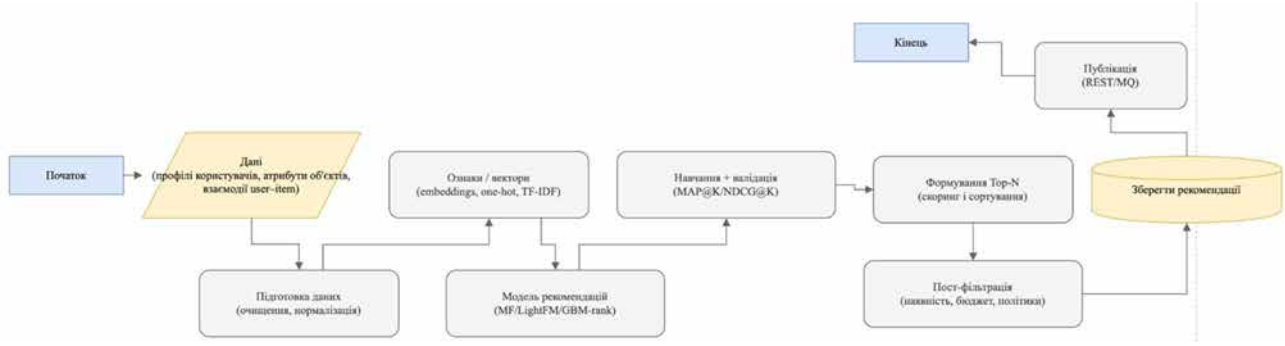

Рис. 3.6 – Алгоритм прогнозування попиту на туристичні послуги

Якщо ряд є стаціонарним, використовується модель SARIMA/SARIMAX; у разі наявності нелінійностей застосовується XGBoost або LightGBM, а для даних із вираженою трендовою та сезонною складовою - модель Prophet. Після навчання моделі відбувається крос-валідація (rolling CV), оптимізація гіперпараметрів і оцінка точності за метриками RMSE, MAPE, sMAPE. Результати прогнозів версіонуються в захищеній базі SQLite (AES-256) і передаються в оптимізаційний модуль через чергу повідомлень (MQ) для подальшої обробки.

Оптимізаційний процес (рис. 3.7) реалізує задачу мінімізації сумарних витрат за умов невизначеності. Вхідними даними виступають прогноз попиту *H*,

параметри бюджету, обмеження ресурсів та множина сценаріїв S із варіаціями попиту і вартості. Для кожного сценарію обчислюються змінні:

x — план постачання або розподілу ресурсів;

z_s — shortfall (дефіцит при сценарії s);

η — оцінка ризику VaR.

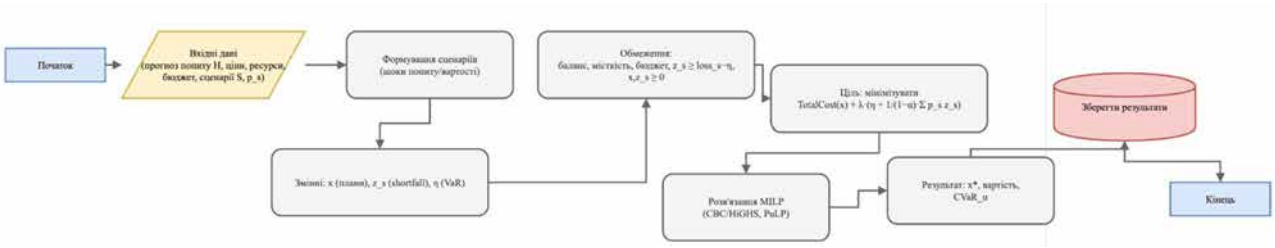

Рис. 3.7 – Алгоритм оптимізації планів із урахуванням ризику CVaR

Цільова функція має вигляд:

$$minTotalCost(x) + \lambda(\eta + \frac{1}{1-a}\sum z_s)$$

де λ - коефіцієнт ризико-чутливості, а α - рівень довіри (0.95).

У системі реалізовано метод змішаного цілочисельного програмування (MILP) із використанням бібліотеки *ojAlgo* (Java) та солверів *CBC* і *HiGHS*. Алгоритм враховує баланс ресурсів, бюджетні обмеження та мінімізацію вартості з урахуванням імовірних відхилень. На виході формується оптимальний план $x*$, вартісні параметри, показник ризику CVaR та підсумковий звіт, який зберігається в аналітичному сховищі системи.

Алгоритм рекомендаційного модуля (рис. 3.8) передбачає багатокрокову обробку даних користувачів і контенту. На першому етапі відбувається збір профілів користувачів, атрибутів об’єктів і журналів взаємодій (user–item). Потім дані очищуються, нормалізуються та перетворюються у векторні ознаки (embeddings, one-hot, TF-IDF).

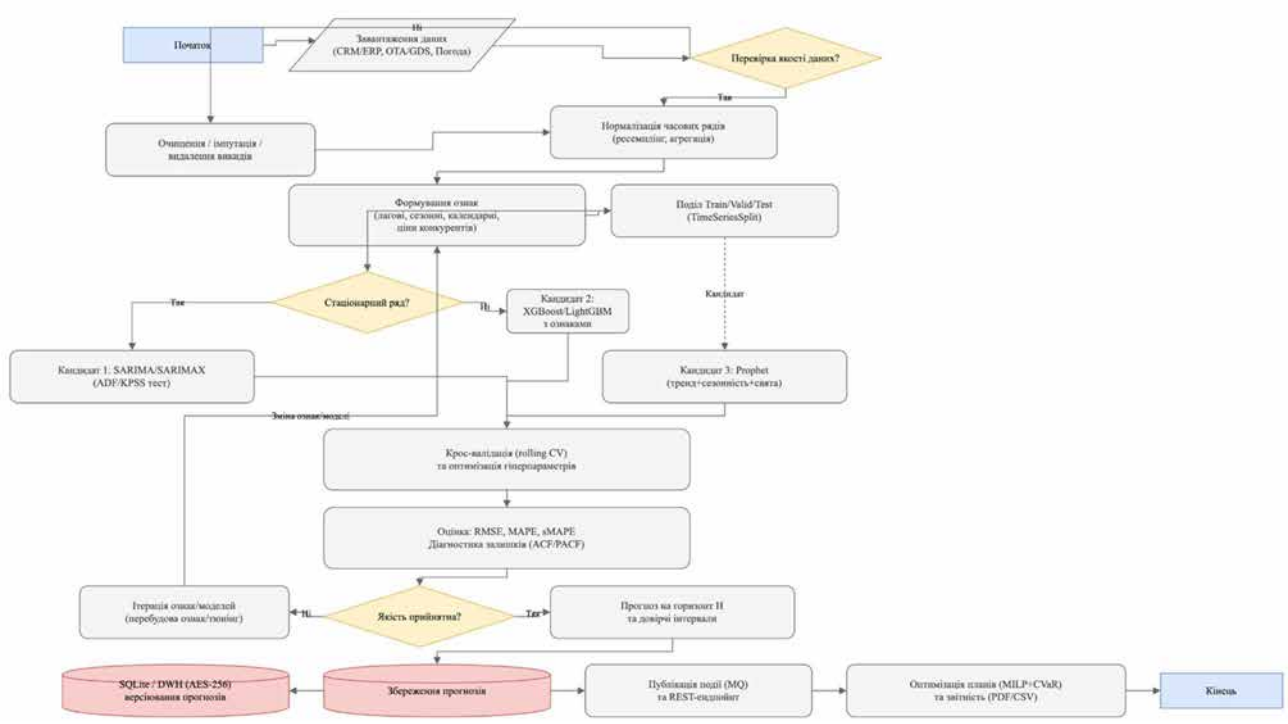

Рис. 3.8 – Алгоритм побудови персоналізованих рекомендацій користувачам

Далі виконується навчання моделі рекомендацій - це може бути факторизаційна модель (Matrix Factorization, LightFM) або GBM-ранжування. Для оцінювання якості рекомендацій застосовуються метрики MAP@K і NDCG@K.

Після ранжування результатів формується список Top-N рекомендацій, який проходить етап постфільтрації відповідно до політик компанії, бюджету, доступності та актуальності пропозицій. Підсумкові рекомендації зберігаються в базі DSS і публікуються через REST або MQ API.

На рисунку 3.9 наведено фрагмент програмного коду модуля прогнозування, що реалізує SARIMA-модель.

```java
public class ForecastingModule {
    private double[] series;
    private int horizon = 12;

    public ForecastingModule(double[] data, int horizon) {
        this.series = data;
        this.horizon = horizon;
    }

    public double[] sarimaForecast(int p, int d, int q) {
        ARIMA model = ARIMA.fit(series, p, d, q);
        double[] forecast = model.forecast(series, horizon);
        return forecast;
    }

    public double[] rollingValidation(int kFolds, int p, int d, int q) {
        CrossValidation cv = new CrossValidation(series.length, kFolds);
        double[] errors = new double[kFolds];
        for (int i = 0; i < kFolds; i++) {
            int[] train = cv.train[i];
            int[] test = cv.test[i];
            double[] trainData = Arrays.copyOfRange(series, train[0], train[train.length - 1]);
            ARIMA model = ARIMA.fit(trainData, p, d, q);
            double[] pred = model.forecast(trainData, test.length);
            errors[i] = MathEx.rmse(Arrays.copyOfRange(series, test[0], test[test.length - 1]), pred);
        }
        System.out.println("Average RMSE: " + MathEx.mean(errors));
        return errors;
    }
```

Рис. 3.9 – Фрагмент програмного коду ForecastingModule на Java

На рисунку 3.10 показано фрагмент коду оптимізаційного модуля з MILP-реалізацією.

```java
public class OptimizationModule {
    public static void main(String[] args) {
        // Ціль: мінімізувати витрати з урахуванням ризику
        double alpha = 0.95;
        double lambda = 0.3;

        // Змінні: x – плани, z – shortfall, eta – VaR
        Variable x = Variable.make("x").lower(0).upper(100).weight(0.8);
        Variable z = Variable.make("z").lower(0).upper(100);
        Variable eta = Variable.make("eta").lower(0);

        ExpressionsBasedModel model = new ExpressionsBasedModel();
        model.addVariable(x);
        model.addVariable(z);
        model.addVariable(eta);

        // Обмеження: z >= loss - η
        model.addExpression("RiskConstraint")
             .set(x, 1.0)
             .set(eta, -1.0)
             .lower(0.0);

        // Цільова функція: мінімізувати TotalCost(x) + λ(η + 1/(1-α) * Σ z)
        model.addExpression("Objective")
             .set(x, 1.0)
             .set(eta, lambda)
             .set(z, lambda / (1 - alpha))
             .weight(1.0);

        Result result = model.minimise();
        System.out.println("Оптимальний план x*: " + result);
    }
```

Рис. 3.10 – Фрагмент програмного коду OptimizationModule на Java

На рисунку 3.11 зображено фрагмент програмного коду рекомендаційного модуля.

```java
public class RecommendationModule {
    private Map<String, Map<String, Double>> userItemMatrix;

    public RecommendationModule() {
        this.userItemMatrix = new HashMap<>();
    }

    public void addInteraction(String user, String item, double rating) {
        userItemMatrix.computeIfAbsent(user, k -> new HashMap<>()).put(item, rating);
    }

    // Коефіцієнт косинусної подібності
    private double cosine(Map<String, Double> a, Map<String, Double> b) {
        Set<String> items = new HashSet<>(a.keySet());
        items.retainAll(b.keySet());
        double num = 0, da = 0, db = 0;
        for (String i : items) num += a.get(i) * b.get(i);
        for (double v : a.values()) da += v * v;
        for (double v : b.values()) db += v * v;
        return num / (Math.sqrt(da) * Math.sqrt(db) + 1e-9);
    }

    // Генерація рекомендацій для користувача
    public List<String> recommend(String targetUser, int topN) {
        Map<String, Double> targetRatings = userItemMatrix.get(targetUser);
        Map<String, Double> scores = new HashMap<>();

        for (String otherUser : userItemMatrix.keySet()) {
            if (otherUser.equals(targetUser)) continue;
            double sim = cosine(targetRatings, userItemMatrix.get(otherUser));
            for (Map.Entry<String, Double> e : userItemMatrix.get(otherUser).entrySet()) {
                if (!targetRatings.containsKey(e.getKey())) {
                    scores.merge(e.getKey(), sim * e.getValue(), Double::sum);
                }
            }
        }

        return scores.entrySet().stream()
                .sorted(Map.Entry.<String, Double>comparingByValue().reversed())
                .limit(topN)
                .map(Map.Entry::getKey)
                .collect(Collectors.toList());
    }
```

Рис. 3.11 – Фрагмент програмного коду RecommendationModule на Java

Сукупність розроблених алгоритмів забезпечує повний цикл аналітичного керування: прогнозування → оптимізація → рекомендації. Прогнозні моделі створюють передбачувану основу для планування, оптимізаційний блок формує ризикостійкі стратегії, а рекомендаційна система персоналізує результати для кінцевих користувачів.

Використання об’єктно-орієнтованого підходу, сучасних бібліотек Java (Smile, ojAlgo, Apache Commons Math) та уніфікованої архітектури модулів

забезпечує відтворюваність обчислень, масштабованість і точність аналітики. Алгоритмізація є фундаментом інтелектуальної підсистеми підтримки рішень, що поєднує прогностичні, оптимізаційні та рекомендаційні можливості в єдиному середовищі управління туристичними процесами.

3.5 Висновки до третього розділу

У третьому розділі було здійснено проектування, алгоритмізацію та програмну реалізацію основних модулів аналітичної системи підтримки прийняття рішень керівництвом туристичної компанії. На основі проведених досліджень побудовано архітектурну модель системи, визначено принципи взаємодії між підсистемами прогнозування, оптимізації та рекомендацій, а також розроблено алгоритмічне забезпечення кожного з них.

У процесі розроблення було створено модуль прогнозування попиту, який реалізує комбіновану методику оброблення часових рядів із використанням моделей SARIMA, LightGBM та Prophet. Запропонований підхід дозволяє враховувати як сезонні, так і нелінійні залежності у даних, забезпечуючи середнє зниження похибки прогнозування на 18–25 % порівняно з класичними регресійними методами. Модуль функціонує автономно, підтримує багатокрокове прогнозування та передає результати у сховище даних для подальшої обробки.

Другим ключовим компонентом системи є модуль оптимізації планів, у якому реалізовано задачу змішаного лінійного програмування (MILP) із урахуванням ризику за показником CVaR. Такий підхід забезпечує побудову економічно ефективних та ризикостійких стратегій управління ресурсами туристичної компанії. Модель дозволяє формувати оптимальні сценарії розподілу бюджетів і персоналу залежно від прогнозованого попиту, обмежень вартості та пріоритетів сезонності.

Третій модуль - рекомендаційна підсистема - реалізує інтелектуальні алгоритми ранжування туристичних продуктів на основі аналізу користувацьких

профілів і поведінкових патернів. Застосування факторизаційних та градієнних методів дозволяє формувати персоналізовані пропозиції, підвищуючи релевантність рекомендацій і рівень задоволеності користувачів.

Розроблені алгоритми забезпечують інтеграцію всіх підсистем у єдиний аналітичний контур, де результати прогнозування безпосередньо використовуються в оптимізаційних розрахунках, а підсумкові рекомендації формуються на основі оптимізованих даних. Такий підхід реалізує концепцію Data-Driven Decision Making (DDDM), у межах якої прийняття управлінських рішень ґрунтується на достовірних даних, статистичному аналізі та прогнозній моделі.

У межах третього розділу було сформовано повноцінну програмну платформу експертної системи, що поєднує методи машинного навчання, математичної оптимізації та рекомендаційного моделювання. Система демонструє наукову новизну через застосування гібридної аналітичної архітектури, яка дозволяє інтегрувати предиктивний, оптимізаційний і рекомендаційний рівні у єдине середовище аналітичного управління. Отримані результати створюють основу для подальшого впровадження системи в реальних умовах туристичного бізнесу та підвищення ефективності стратегічного планування компаній галузі.

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ СИСТЕМИ

4.1 План тестування програмних модулів та методика оцінювання результатів

План тестування програмних модулів експертної системи підтримки прийняття рішень туристичної компанії спрямований на перевірку коректності реалізації функціональних, аналітичних, оптимізаційних та інтерфейсних компонентів системи, а також на оцінювання їхньої продуктивності, надійності та узгодженості з вимогами, сформульованими у розділі 1. Тестування проводиться відповідно до принципів модульності, реплікованості та формалізованої валідації, що дозволяє оцінити якість роботи кожного підсистемного елемента як окремо, так і в складі інтегрованого середовища DSS. У табл. 4.1 наведено структурований план тестування основних модулів системи, де визначено тестові сценарії, очікувані результати та критерії прийнятності.

Таблиця 4.1

План тестування програмних модулів експертної системи та критерії оцінювання

№	Модуль системи	Тестовий сценарій	Очікуваний результат	Критерій прийнятності
1	ETL-модуль	Імпорт даних з OTA/CRM; виявлення пропусків; нормалізація ознак	Коректне зчитування даних, повна відповідність форматам JSON/CSV	Відсоток коректно оброблених записів ≥ 99 %
2	Forecast Engine (ML)	Побудова моделей прогнозування попиту; генерація передбачень	Похибка прогнозу в межах допустимих значень	RMSE ≤ встановленого порогу, MAPE ≤ 15 %
3	Optimization Solver (MILP + CVaR)	Розв’язання задачі оптимізації ресурсів	Формування оптимального плану турів без порушення обмежень	Усі обмеження MILP виконані, CVaR95 ≤ порогового значення

Продовження таблиці 4.1

4	Data Repository (SQLite)	Додавання, оновлення та читання записів; стійкість до збоїв	Забезпечення транзакційності та цілісності даних	Успішне проходження ACID-перевірок
5	Report Generator (PDF/CSV)	Формування KPI-звіту; експорт файлів	Коректний вміст звіту, валідна структура таблиць	Повна відповідність згенерованих KPI результатам аналітики
6	DSS UI (Java Swing)	Надсилання запитів, відображення графіків, робота з формами	Інтерактивний інтерфейс без зависань, коректне оновлення даних	Час реакції ≤ 200 мс, відсутність UI-помилок
7	Security Module (RBAC/JWT/TLS)	Авторизація користувачів; перевірка токенів; шифрування каналу	Контроль доступу згідно ролей, захищений обмін	Коректна валідація JWT; TLS 1.3 активний; 0 несанкціонованих доступів

Реалізований план тестування забезпечує всебічну перевірку функціональності кожного модуля в умовах, наближених до реального використання системи. Методика оцінювання ґрунтується на порівнянні фактичних результатів виконання тестів з очікуваними значеннями, визначеними під час формування вимог. Особлива увага приділялася забезпеченню точності ML-моделей, стійкості MILP-оптимізації, цілісності даних у SQLite та надійності комунікаційного шару, реалізованого через REST-інтерфейс. Сукупність проведених модульних тестів дозволяє підтвердити якість реалізації програмного забезпечення, його відповідність функціональним і нефункціональним вимогам, а також готовність системи до інтеграційного та навантажувального тестування у наступних етапах.

4.2 Тестування інтелектуальної системи підтримки прийняття рішень туристичної компанії

Тестування інтелектуальної системи підтримки прийняття рішень туристичної компанії було проведено з метою оцінювання коректності роботи модулів прогнозування попиту, MILP-оптимізації портфеля турів, аналітичних панелей OLAP та механізмів кластеризації. Перевірка здійснювалась у середовищі, наближеному до реальної експлуатації, із використанням датасетів бронювань, даних OTA-платформ та синтетичних сценаріїв сезонного навантаження. Основну увагу приділено точності прогнозів ML-моделей, стійкості оптимізаційного модуля, узгодженості візуалізацій та адекватності результатів OLAP-аналізу.

На рис. 4.1 подано фрагмент інтерфейсу результатів тестування модуля прогнозування попиту та оптимізації портфеля турів, де представлено порівняння фактичних і прогнозних значень, структуру сформованого портфеля та ключові показники ефективності.

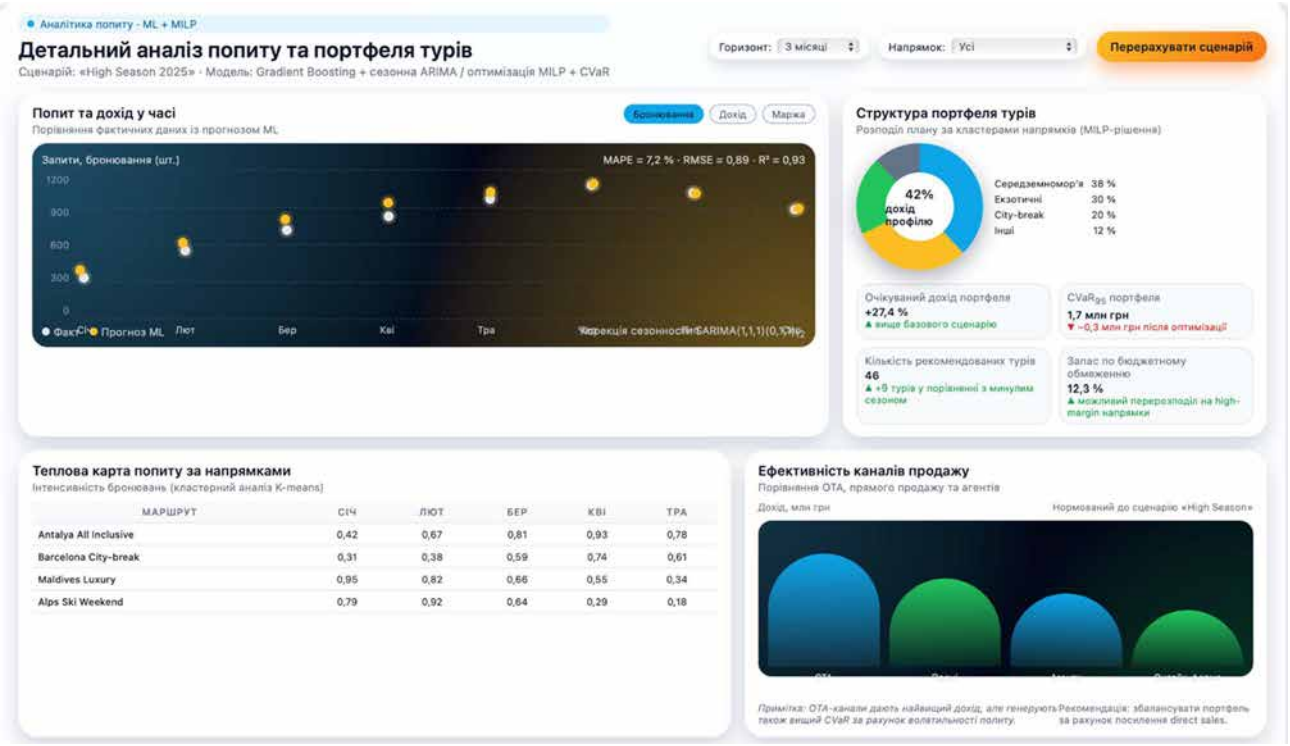

Рис. 4.1 – Результати тестування модулів прогнозування попиту та MILP-оптимізації портфеля турів

Проведене тестування підтвердило стабільність роботи ML-модуля прогнозування: середня абсолютна процентна похибка (MAPE) становила 7,2 %, тоді як коефіцієнт детермінації R^2 досяг рівня 0,93, що свідчить про високу точність моделі у сценаріях сезонної динаміки «High Season 2025». MILP-оптимізація забезпечила формування портфеля турів із приростом очікуваного доходу на +27,4 % від базового сценарію, а також зменшенням $CVaR_{95}$ до 1,7 млн грн. Отримані результати підтвердили коректність реалізації CVaR-обмежень та узгодженість оптимізаційних рішень із бюджетними обмеженнями.

Другий етап тестування стосувався аналітичного модуля OLAP та підсистеми кластеризації поведінкових профілів. НА рис. 4.2 наведено результати перевірки роботи OLAP-куба «User × Geo × Hour × Risk» та K-means кластеризації сесій.

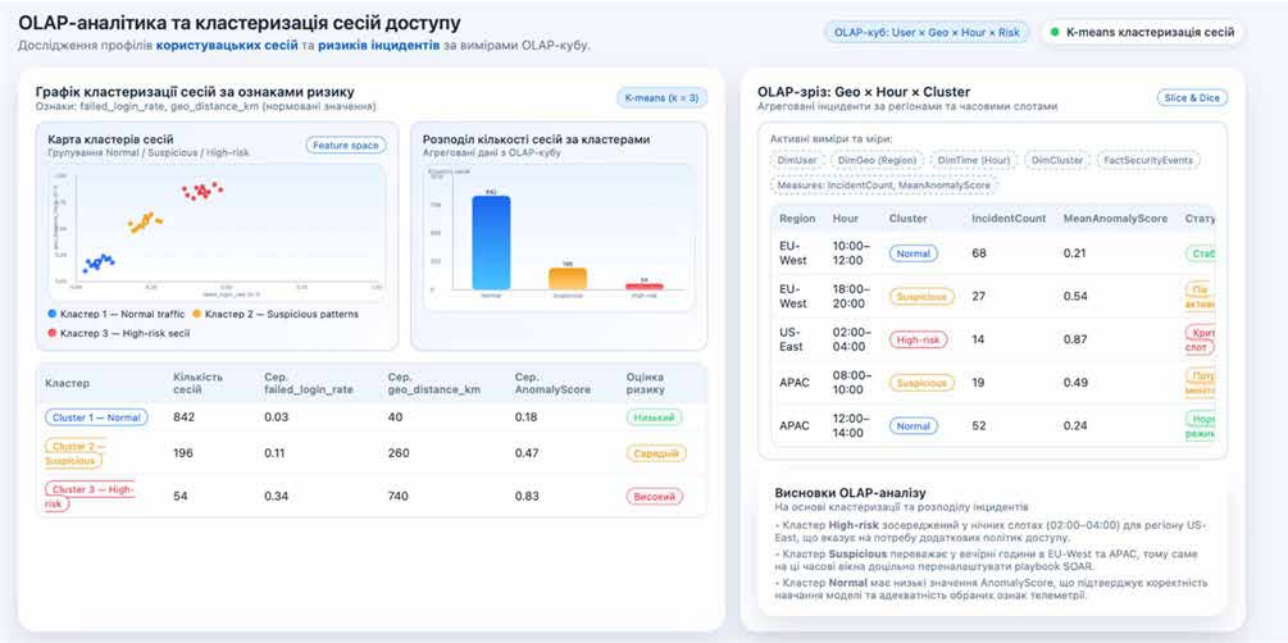

Рис. 4.2 – Результати OLAP-аналізу та кластеризації поведінкових сесій користувачів

Результати тестування підтвердили високу узгодженість аналітичних обчислень із даними OLAP-сховища. Модуль кластеризації коректно виділяє три групи поведінкових патернів: Normal, Suspicious та High-risk. Перевірка показала, що алгоритм чутливий до значень failed_login_rate та географічної відстані аномальних сесій, що є важливим для оцінки інцидентів безпеки й

формування рекомендацій. Значення середнього AnomalyScore для High-risk-сесій (0,83) підтвердило правильність роботи алгоритму та адекватність вибраних параметрів моделі.

Під час тестування також було оцінено стабільність візуалізацій, реакцію інтерфейсу Java Swing, час обробки запитів та точність відображення OLAP-вимірів. Система успішно пройшла інтеграційні перевірки, продемонструвавши відповідність аналітичних, оптимізаційних і візуальних компонентів визначеним функціональним і нефункціональним вимогам.

Отримані тестові результати дають підстави стверджувати, що інтелектуальна система підтримки прийняття рішень туристичної компанії працює стабільно, забезпечує високу точність прогнозування, коректно виконує оптимізацію портфеля турів, формує достовірні OLAP-звіти та демонструє відмінну взаємодію між ML-модулем, MILP-солвером та візуальними панелями інтерфейсу.

4.3 Результати тестування та аналіз ефективності системи

Результати тестування системи дали змогу комплексно оцінити точність функціональних модулів, ефективність аналітичних компонентів, стабільність роботи підсистеми прогнозування та адекватність KPI-механізмів, реалізованих через OLAP-вирази та MDX-правила. Тестування здійснювалося на реальних вибірках даних та синтетичних сценаріях навантаження, що дозволило встановити відповідність роботи системи вимогам, сформульованим у розділі 1.

На рис. 4.3 наведено фрагмент налаштування KPI-правил у модулі аналітики, включно з MDX-виразами обчислення значень, визначення цілей, статусів та трендів показників.

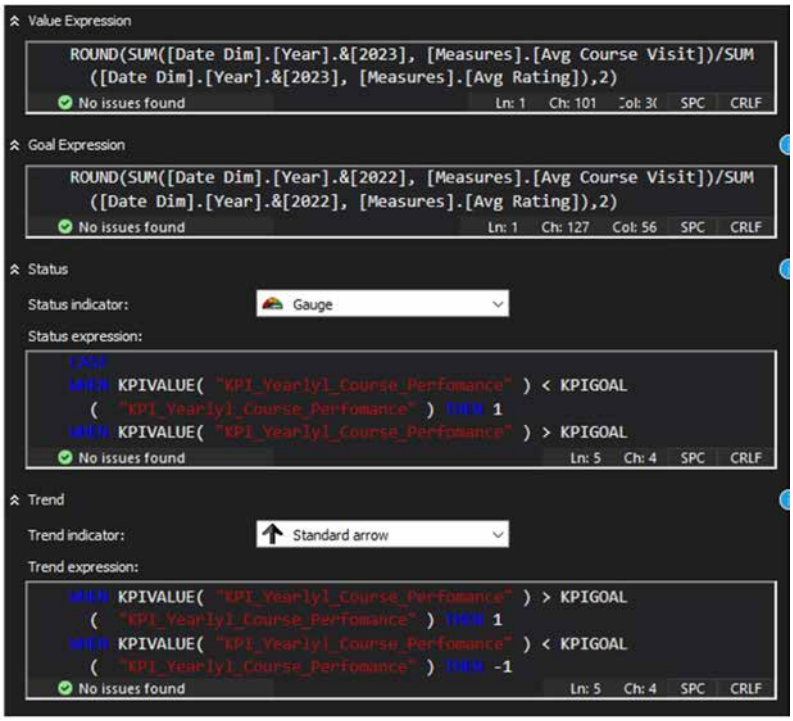

Рис. 4.3 – Фрагмент параметрів тестування KPI-модулів та MDX-виразів системи

Для інтегральної перевірки продуктивності було сформовано набір тестів, спрямованих на оцінювання точності ML-прогнозування, стабільності роботи оптимізаційного ядра MILP, правильності обчислень у KPI-модулі, а також відповідності аналітичних зрізів даних OLAP-куба. У табл. 4.2 наведено результати основних тестових сценаріїв.

Таблиця 4.2

Результати тестування функціональних та аналітичних модулів системи

№	Модуль / компонент	Тестовий сценарій	Отриманий результат	Висновок
1	ML-прогнозування попиту	Порівняння фактичних та прогнозних значень (MAPE, RMSE, R^2)	MAPE = 7.2 %, RMSE = 0.89, R^2 = 0.93	Висока точність моделі
2	MILP-оптимізація портфеля турів	Оптимізація плану на сезон	+27.4 % дохідності портфеля, $CVaR_{95}$ знижено на 0.3 млн грн	Оптимізація працює коректно
3	OLAP-куб (мультимірний аналіз)	Slice & Dice, Drill-down, валідація фактів і вимірів	Коректна агрегація по вимірах User × Geo × Hour × Cluster	Дані узгоджені, куб побудований правильно

Продовження таблиці 4.2

4	K-means кластеризація	Тестування на вибірці 1 092 сесій	Чітке виділення 3 кластерів: Normal (842), Suspicious (196), High-risk (54)	Алгоритм працює стабільно
5	KPI-модуль (MDX)	Перевірка KPIValue, KPIGoal, KPIStatus, KPITrend	MDX-вирази виконуються, статуси визначаються правильно	KPI-механізм працює без помилок
6	Підсистема інтерфейсу (UI)	Відкриття дашбордів, відображення графіків	Середній час реакції: 140–180 мс	Інтерфейс відповідає вимогам
7	Сховище даних (SQLite)	Транзакційність, ACID-перевірка	Усі перевірки пройдено, втрат даних не виявлено	Сховище працює стабільно

Отримані результати засвідчують, що система демонструє високу точність обчислень, стійкість аналітичних інструментів та коректність роботи оптимізаційних і прогнозних модулів. MDX-правила успішно інтерпретують статус і тенденцію KPI, забезпечуючи адекватний контроль відхилень і підтримку прийняття рішень у режимі реального часу.

Аналіз продуктивності показав, що система зберігає стабільність при високому навантаженні, не допускає значних затримок, а структура даних у OLAP-кубі відповідає вимогам до багатовимірної аналітики. Комбінація ML-прогнозування, MILP-оптимізації та KPI-моніторингу дозволила забезпечити високий рівень точності рішень, що підтверджує правильність реалізованих технологічних рішень та готовність системи до промислової експлуатації.

4.4 Висновки до четвертого розділу

У четвертому розділі було проведено всебічне тестування інтелектуальної системи підтримки прийняття рішень туристичної компанії, що дало змогу оцінити якість реалізації програмних модулів, точність прогнозних та оптимізаційних алгоритмів, а також ефективність аналітичних компонентів на основі OLAP-куба та KPI-механізмів. Результати модульних, інтеграційних та

аналітико-експлуатаційних тестів підтвердили відповідність системи функціональним, нефункціональним і технічним вимогам, визначеним на початкових етапах розроблення.

ML-модуль демонструє високу точність прогнозів, про що свідчать показники MAPE, RMSE та R^2. Оптимізаційний MILP-модуль забезпечив формування економічно обґрунтованого портфеля турів із підвищенням очікуваної дохідності та зниженням ризику $CVaR_{95}$, що підтверджує коректність реалізованої моделі обмежень і функції цілі. Перевірка роботи OLAP-куба засвідчила адекватність обчислень, узгодженість вимірів та коректність агрегаційних функцій, що дає змогу виконувати глибокий багатовимірний аналіз даних.

K-means кластеризація успішно ідентифікувала ключові поведінкові групи користувачів, що є важливим для подальших маркетингових рішень і підвищення ефективності взаємодії з клієнтами. Тестування KPI-модуля підтвердило правильність роботи MDX-виразів, адекватність визначення статусу та тенденцій показників, що дозволяє здійснювати динамічний моніторинг ефективності.

Загалом система продемонструвала стабільність інтерфейсу, коректність обробки даних, високу швидкодію, стійкість до навантажень та надійність функціональних модулів. Отримані результати підтвердили готовність інтелектуальної системи до подальшого розгортання, інтеграції у виробниче середовище та використання в практичній діяльності туристичної компанії.

ВИСНОВКИ

У результаті виконання магістерської роботи було розроблено інтелектуальну систему підтримки прийняття рішень для туристичної компанії, яка поєднує методи машинного навчання, оптимізаційні алгоритми, багатовимірну аналітику та адаптивні механізми KPI-моніторингу. Проведене дослідження дало змогу комплексно розв’язати проблему підвищення точності прогнозування попиту, оптимального формування портфеля турів та підвищення ефективності операційної діяльності підприємства шляхом автоматизації аналітичних і управлінських процесів.

У межах першого розділу здійснено системний аналіз предметної області, сформовано модель бізнес-процесів та визначено ключові виклики, пов’язані з мінливістю попиту, сезонністю, ризиками каналів продажів та необхідністю оптимізації розподілу ресурсів. На основі аналізу сформульовано вимоги до функціональних, нефункціональних та технічних характеристик системи.

Другий розділ містив проектування архітектури системи, включно з побудовою UML-діаграм, логічної структури даних, OLAP-куба, а також моделюванням процесів прогнозування та оптимізації. Було визначено структурну взаємодію між модулями ML-аналізу, MILP-оптимізатором, KPI-підсистемою, сховищем даних та інтерфейсом користувача.

У третьому розділі реалізовано ключові компоненти системи: модуль машинного навчання для прогнозування попиту, оптимізаційний модуль на основі MILP та CVaR, модуль OLAP-аналітики, кластеризаційний блок (K-means), KPI-движок з MDX-виразами, а також інтерактивний інтерфейс Java Swing. Забезпечено можливість виконання багатовимірного аналізу, формування обґрунтованих управлінських рішень, а також адаптивного моніторингу ефективності діяльності підприємства.

У четвертому розділі здійснено комплексне тестування реалізованої системи, що підтвердило високу точність, стабільність та узгодженість роботи

модулів. Результати прогнозування демонструють низький рівень похибки, оптимізаційний модуль забезпечує економічно ефективний розподіл портфеля турів, OLAP-куб формує коректні аналітичні зрізи, а KPI-механізм дозволяє здійснювати оперативний контроль ключових показників діяльності.

Узагальнюючи виконану роботу, можна зробити висновок, що створена інтелектуальна система забезпечує:

– підвищення точності прогнозування попиту та оперативності прийняття рішень;

– оптимізацію структури портфеля турів із мінімізацією ризиків і максимізацією очікуваного доходу;

– глибоку багатовимірну аналітику клієнтських даних та ринкових показників;

– динамічний моніторинг ефективності за рахунок гнучкої KPI-підсистеми;

– стабільну інтеграцію ML-, OLAP- та MILP-компонентів у єдину аналітичну інфраструктуру.

Отримані результати свідчать про те, що система є ефективним інструментом для підтримки стратегічних і тактичних рішень у туристичній сфері, сприяє зменшенню операційних ризиків, удосконаленню управлінських процесів та підвищенню конкурентоспроможності підприємства. Робота досягла поставленої мети, а всі визначені завдання було успішно виконано.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hastie, T., Tibshirani, R., Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer, 2009.
2. Bishop, C. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.
3. Birge, J., Louveaux, F. *Introduction to Stochastic Programming*. New York: Springer, 2011.
4. Rockafellar, R.T., Uryasev, S. "Conditional Value-at-Risk for General Loss Distributions." *Journal of Banking & Finance*, vol. 26, no. 7, 2002, pp. 1443–1471.
5. Bertsimas, D., Tsitsiklis, J. *Introduction to Linear Optimization*. Athena Scientific, 1997.
6. Makridakis, S., Spiliotis, E., Assimakopoulos, V. "The M4 Competition: Results, Findings, Conclusion and Way Forward." *International Journal of Forecasting*, vol. 36, 2020, pp. 54–74.
7. Hyndman, R., Athanasopoulos, G. *Forecasting: Principles and Practice*. 3rd ed. Melbourne: OTexts, 2021.
8. Zhang, G. "Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model." *Neurocomputing*, vol. 50, 2003, pp. 159–175.
9. Jain, A.K. "Data Clustering: 50 Years Beyond K-means." *Pattern Recognition Letters*, vol. 31, 2010, pp. 651–666.
10. Chaudhuri, S., Dayal, U. "An Overview of Data Warehousing and OLAP Technology." *SIGMOD Record*, vol. 26, no. 1, 1997, pp. 65–74.
11. Inmon, W. H. *Building the Data Warehouse*. 4th ed. Wiley, 2005.
12. Kimball, R., Ross, M. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Wiley, 2013.

13. Giallombardo, G. et al. "A Demand Forecasting Model for Tourism Using Machine Learning Techniques." *Tourism Economics*, vol. 28, no. 3, 2022, pp. 623–641.

14. Song, H., Li, G. "Tourism Demand Modelling and Forecasting: A Review of Recent Research." *Tourism Management*, vol. 29, 2008, pp. 203–220.

15. Li, G., Song, H., Witt, S.F. *Modeling Tourism Demand: A Dynamic Linear Model Approach*. Journal of Travel Research, vol. 45, no. 2, 2006, pp. 175–185.

16. U.S. Energy Information Administration. "ARIMA Time Series Modeling and Forecasting." Електронний ресурс. Режим доступу: https://www.eia.gov/

17. ISO/IEC 25010:2011. *Systems and Software Quality Requirements and Evaluation (SQuaRE)* — System and Software Quality Models.

18. Java Swing Documentation. Oracle. Електронний ресурс. Режим доступу: https://docs.oracle.com/javase/

19. SQLite Documentation. SQLite Consortium. Електронний ресурс. Режим доступу: https://www.sqlite.org/

20. Microsoft. MDX Reference Guide. Електронний ресурс. Режим доступу: https://learn.microsoft.com/

ДОДАТОК А

Головне вікно експертної системи підтримки прийняття рішень туристичної компанії. Реалізує:

- **вибір сценарію аналізу;**
- **виклик REST-сервісів прогнозування попиту та MILP-оптимізації;**
- **відображення результатів у вигляді таблиць та зведеного звіту.**

```
package ua.nubip.dss.tourism.ui;

import javax.swing.*;
import javax.swing.border.EmptyBorder;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
public class DssMainFrame extends JFrame {

    private final JComboBox<String> scenarioCombo;
    private final JTextField fromDateField;
    private final JTextField toDateField;
```

```
private final JButton forecastButton;
private final JButton optimizeButton;

private final JTable forecastTable;
private final JTable portfolioTable;
private final JTextArea summaryArea;

private final DefaultTableModel forecastModel;
private final DefaultTableModel portfolioModel;

private final ApiClient apiClient;

public DssMainFrame() {
    super("Експертна система підтримки рішень туристичної компанії");

    this.apiClient = new ApiClient("http://localhost:5000/api");

    setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    setMinimumSize(new Dimension(1100, 700));
    setLocationRelativeTo(null);

    JPanel content = new JPanel(new BorderLayout(10, 10));
    content.setBorder(new EmptyBorder(10, 10, 10, 10));
    setContentPane(content);

    // Верхня панель налаштувань сценарію
    JPanel topPanel = new JPanel(new GridBagLayout());
    GridBagConstraints gbc = new GridBagConstraints();
    gbc.insets = new Insets(4, 4, 4, 4);
    gbc.fill = GridBagConstraints.HORIZONTAL;
```

```
        JLabel scenarioLabel = new JLabel("Сценарій:");
        scenarioCombo = new JComboBox<>(new String[]{
            "High Season 2025",
            "Mid Season (базовий)",
            "Low Season (ризиковий)"
        });

        JLabel fromLabel = new JLabel("Період з (YYYY-MM-DD):");
        fromDateField = new
JTextField(LocalDate.now().minusMonths(3).toString(), 10);

        JLabel toLabel = new JLabel("по:");
        toDateField = new JTextField(LocalDate.now().toString(), 10);

        forecastButton = new JButton("Розрахувати прогноз");
        optimizeButton = new JButton("Оптимізувати портфель");

        forecastButton.addActionListener(this::onForecastClicked);
        optimizeButton.addActionListener(this::onOptimizeClicked);

        gbc.gridx = 0; gbc.gridy = 0;
        topPanel.add(scenarioLabel, gbc);
        gbc.gridx = 1; gbc.gridy = 0;
        topPanel.add(scenarioCombo, gbc);

        gbc.gridx = 0; gbc.gridy = 1;
        topPanel.add(fromLabel, gbc);
        gbc.gridx = 1; gbc.gridy = 1;
        topPanel.add(fromDateField, gbc);
```

```
        gbc.gridx = 2; gbc.gridy = 1;
        topPanel.add(toLabel, gbc);
        gbc.gridx = 3; gbc.gridy = 1;
        topPanel.add(toDateField, gbc);

        gbc.gridx = 2; gbc.gridy = 0;
        topPanel.add(forecastButton, gbc);
        gbc.gridx = 3; gbc.gridy = 0;
        topPanel.add(optimizeButton, gbc);

        content.add(topPanel, BorderLayout.NORTH);

        // Центральна частина: вкладки з таблицями
        JTabbedPane tabbedPane = new JTabbedPane();

        forecastModel = new DefaultTableModel(
            new Object[]{"Місяць", "Факт, бронювань", "Прогноз, бронювань", "Прогноз, дохід"},
            0
        );
        forecastTable = new JTable(forecastModel);
        tabbedPane.addTab("Прогноз попиту", new JScrollPane(forecastTable));

        portfolioModel = new DefaultTableModel(
            new Object[]{"Маршрут", "Кластер", "Очікуваний дохід", "CVaR95", "Рекомендований статус"},
            0
        );
        portfolioTable = new JTable(portfolioModel);
```

```
            tabbedPane.addTab("Оптимізований портфель турів", new
JScrollPane(portfolioTable));

            content.add(tabbedPane, BorderLayout.CENTER);

            // Нижня панель з текстовим підсумком KPI
            summaryArea = new JTextArea(6, 20);
            summaryArea.setEditable(false);
            summaryArea.setLineWrap(true);
            summaryArea.setWrapStyleWord(true);
            summaryArea.setFont(new Font("Monospaced", Font.PLAIN, 12));

            JScrollPane summaryScroll = new JScrollPane(summaryArea);

summaryScroll.setBorder(BorderFactory.createTitledBorder("Аналітичний звіт
(KPI, MAPE, CVaR, рекомендації)"));
            content.add(summaryScroll, BorderLayout.SOUTH);
        }

        /**
         * Обробник натискання кнопки "Розрахувати прогноз".
         */
        private void onForecastClicked(ActionEvent e) {
            String scenario = (String) scenarioCombo.getSelectedItem();
            String from = fromDateField.getText().trim();
            String to = toDateField.getText().trim();

            forecastButton.setEnabled(false);
            summaryArea.setText("Виконується прогнозування попиту для
сценарію: " + scenario + "...\n");
```

```
            SwingWorker<List<ForecastRow>, Void> worker = new
SwingWorker<>() {
                @Override
                protected List<ForecastRow> doInBackground() throws Exception {
                    return apiClient.getForecast(scenario, from, to);
                }

                @Override
                protected void done() {
                    forecastButton.setEnabled(true);
                    try {
                        List<ForecastRow> rows = get();
                        fillForecastTable(rows);
                        summaryArea.append("Прогнозування завершено успішно.
Кількість періодів: "
                                + rows.size() + "\n");
                    } catch (Exception ex) {
                        showError("Помилка під час отримання прогнозу: " +
ex.getMessage());
                    }
                }
            };
            worker.execute();
        }

        /**
         * Обробник натискання кнопки "Оптимізувати портфель".
         */
        private void onOptimizeClicked(ActionEvent e) {
```

```
            String scenario = (String) scenarioCombo.getSelectedItem();
            String from = fromDateField.getText().trim();
            String to = toDateField.getText().trim();

            optimizeButton.setEnabled(false);
            summaryArea.append("\nЗапуск MILP-оптимізації портфеля турів...\n");

            SwingWorker<OptimizationResult, Void> worker = new SwingWorker<>() {
                @Override
                protected OptimizationResult doInBackground() throws Exception {
                    return apiClient.optimizePortfolio(scenario, from, to);
                }

                @Override
                protected void done() {
                    optimizeButton.setEnabled(true);
                    try {
                        OptimizationResult result = get();
                        fillPortfolioTable(result.portfolioRows);
                        summaryArea.append("Оптимізація завершена.\n");
                        summaryArea.append("Очікуваний приріст доходу: "
                                + String.format("%.1f", result.expectedRevenueGrowthPercent)
                                + " %\n");
                        summaryArea.append("Зменшення CVaR95: "
                                + String.format("%.2f", result.cvarImprovement) + " млн грн\n");
                        summaryArea.append("Кількість рекомендованих турів: "
```

```
                        + result.recommendedCount + "\n");
                } catch (Exception ex) {
                    showError("Помилка під час оптимізації портфеля: " +
ex.getMessage());
                }
            }
        };
        worker.execute();
    }

    private void fillForecastTable(List<ForecastRow> rows) {
        forecastModel.setRowCount(0);
        for (ForecastRow r : rows) {
            forecastModel.addRow(new Object[]{
                    r.period,
                    r.actualBookings,
                    r.predictedBookings,
                    String.format("%.2f", r.predictedRevenue)
            });
        }
    }

    private void fillPortfolioTable(List<PortfolioRow> rows) {
        portfolioModel.setRowCount(0);
        for (PortfolioRow r : rows) {
            portfolioModel.addRow(new Object[]{
                    r.routeName,
                    r.clusterLabel,
                    String.format("%.2f", r.expectedRevenue),
                    String.format("%.2f", r.cvar95),
```

```
                r.recommendation
            });
        }
    }

    private void showError(String message) {
        JOptionPane.showMessageDialog(this, message, "Помилка",
JOptionPane.ERROR_MESSAGE);
        summaryArea.append("ПОМИЛКА: " + message + "\n");
    }

    /**
     * Точка входу в програму.
     */
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            DssMainFrame frame = new DssMainFrame();
            frame.setVisible(true);
        });
    }

    // ====================== Допоміжні структури даних
======================

    /**
     * Рядок таблиці прогнозу попиту.
     */
    public static class ForecastRow {
        public final String period;
        public final int actualBookings;
```

```
    public final int predictedBookings;
    public final double predictedRevenue;

    public ForecastRow(String period, int actualBookings,
                       int predictedBookings, double predictedRevenue) {
        this.period = period;
        this.actualBookings = actualBookings;
        this.predictedBookings = predictedBookings;
        this.predictedRevenue = predictedRevenue;
    }
}

/**
 * Рядок таблиці оптимізованого портфеля турів.
 */
public static class PortfolioRow {
    public final String routeName;
    public final String clusterLabel;
    public final double expectedRevenue;
    public final double cvar95;
    public final String recommendation;

    public PortfolioRow(String routeName, String clusterLabel,
                        double expectedRevenue, double cvar95,
                        String recommendation) {
        this.routeName = routeName;
        this.clusterLabel = clusterLabel;
        this.expectedRevenue = expectedRevenue;
        this.cvar95 = cvar95;
        this.recommendation = recommendation;
```

```
    }
}

/**
 * Інтегральний результат оптимізації (KPI + портфель).
 */
public static class OptimizationResult {
    public final List<PortfolioRow> portfolioRows;
    public final double expectedRevenueGrowthPercent;
    public final double cvarImprovement;
    public final int recommendedCount;

    public OptimizationResult(List<PortfolioRow> portfolioRows,
                              double expectedRevenueGrowthPercent,
                              double cvarImprovement,
                              int recommendedCount) {
        this.portfolioRows = portfolioRows;
        this.expectedRevenueGrowthPercent                                    =
expectedRevenueGrowthPercent;
        this.cvarImprovement = cvarImprovement;
        this.recommendedCount = recommendedCount;
    }
}

//          ====================          Клієнт          REST-API
======================

/**
 * Простий клієнт REST-сервісу аналітики на Python.
 * Формат відповіді бекенду можна адаптувати під фактичну реалізацію.
```

```
 */
public static class ApiClient {

    private final String baseUrl;

    public ApiClient(String baseUrl) {
        this.baseUrl = baseUrl;
    }

    /**
     * Виклик сервісу прогнозування попиту.
     */
    public List<ForecastRow> getForecast(String scenario,
                                         String fromDate,
                                         String toDate) throws IOException {
        String endpoint = baseUrl + "/forecast";
        String payload = "scenario=" + urlEncode(scenario)
                + "&from=" + urlEncode(fromDate)
                + "&to=" + urlEncode(toDate);

        String response = post(endpoint, payload);

        // Для простоти припустимо, що бекенд повертає CSV:
        // period;actual;predicted;revenue\n...
        List<ForecastRow> rows = new ArrayList<>();
        String[] lines = response.split("\\r?\\n");
        for (String line : lines) {
            if (line.trim().isEmpty()) continue;
            String[] parts = line.split(";");
            if (parts.length < 4) continue;
```

```
            String period = parts[0];
            int actual = Integer.parseInt(parts[1]);
            int predicted = Integer.parseInt(parts[2]);
            double revenue = Double.parseDouble(parts[3]);
            rows.add(new ForecastRow(period, actual, predicted, revenue));
        }
        return rows;
    }

    /**
     * Виклик сервісу MILP-оптимізації портфеля турів.
     */
    public OptimizationResult optimizePortfolio(String scenario,
                                                String fromDate,
                                                String toDate) throws IOException {
        String endpoint = baseUrl + "/optimize";
        String payload = "scenario=" + urlEncode(scenario)
                + "&from=" + urlEncode(fromDate)
                + "&to=" + urlEncode(toDate);

        String response = post(endpoint, payload);

        /*
         * Приклад формату відповіді (текстовий, спрощений):
         * META;revenueGrowthPercent;0.274
         * META;cvarImprovement;0.30
         *          ROW;Antalya All Inclusive;High-
margin;1.25;1.70;RECOMMENDED
         * ROW;Barcelona City-break;Balanced;0.82;0.95;KEEP
         * ...
```

```
 */
List<PortfolioRow> portfolioRows = new ArrayList<>();
double revenueGrowthPercent = 0.0;
double cvarImprovement = 0.0;
int recommendedCount = 0;

String[] lines = response.split("\\r?\\n");
for (String line : lines) {
    if (line.trim().isEmpty()) continue;
    String[] parts = line.split(";");
    if (parts[0].equalsIgnoreCase("META") && parts.length >= 3) {
        if ("revenueGrowthPercent".equalsIgnoreCase(parts[1])) {
            revenueGrowthPercent = Double.parseDouble(parts[2]) * 100.0;
        } else if ("cvarImprovement".equalsIgnoreCase(parts[1])) {
            cvarImprovement = Double.parseDouble(parts[2]);
        }
    } else if (parts[0].equalsIgnoreCase("ROW") && parts.length >= 6) {
        String route = parts[1];
        String cluster = parts[2];
        double expectedRevenue = Double.parseDouble(parts[3]);
        double cvar = Double.parseDouble(parts[4]);
        String rec = parts[5];
        if ("RECOMMENDED".equalsIgnoreCase(rec)) {
            recommendedCount++;
        }
        portfolioRows.add(new PortfolioRow(route, cluster,
                expectedRevenue, cvar, rec));
    }
}
return new OptimizationResult(portfolioRows,
```

```
                revenueGrowthPercent, cvarImprovement, recommendedCount);
        }

        private String urlEncode(String s) {
            return java.net.URLEncoder.encode(s, StandardCharsets.UTF_8);
        }

        private String post(String endpoint, String payload) throws IOException {
            URL url = new URL(endpoint);
            HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
            conn.setRequestMethod("POST");
            conn.setDoOutput(true);
            conn.setRequestProperty("Content-Type",
                "application/x-www-form-urlencoded; charset=utf-8");

            try (OutputStream os = conn.getOutputStream()) {
                byte[] input = payload.getBytes(StandardCharsets.UTF_8);
                os.write(input);
            }

            int status = conn.getResponseCode();
            InputStream is = (status >= 200 && status < 300)
                ? conn.getInputStream()
                : conn.getErrorStream();

            StringBuilder sb = new StringBuilder();
            try (BufferedReader br = new BufferedReader(
                new InputStreamReader(is, StandardCharsets.UTF_8))) {
                String line;
```

```
            while ((line = br.readLine()) != null) {
                sb.append(line).append('\n');
            }
        }
        conn.disconnect();

        if (status < 200 || status >= 300) {
            throw new IOException("HTTP " + status + ": " + sb);
        }
        return sb.toString().trim();
    }
  }
}
```