

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___» _____ 2025 р.

Дипломний проєкт
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційні управляючі системи
та технології»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Інформаційна система прогнозування попиту на транспортні
послуги у великих містах на основі машинного навчання»

Виконав:

студент IV курсу, групи ІС-12

Примак Олександр Володимирович _____

Керівник:

доцент кафедри ІСТ, к.т.н., доцент,

Писаренко Андрій Володимирович _____

Рецензент:

доцент кафедри ІП, к.т.н., доцент,

Ліщук Катерина Ігорівна _____

Засвідчую, що у цьому дипломному
проєкті немає запозичень з праць інших
авторів без відповідних посилань.

Студент _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Олександр РОЛІК

«___» _____ 2025 р.

ЗАВДАННЯ
на дипломний проєкт студенту
Примаку Олександру Володимировичу

1. Тема проєкту «Інформаційна система прогнозування попиту на транспортні послуги у великих містах на основі машинного навчання», керівник проєкту Писаренко Андрій Володимирович, доцент кафедри ІСТ, к.т.н., затверджені наказом по університету від «23» травня 2025 р. № 1705-с
2. Термін подання студентом проєкту: 9 червня 2025 року
3. Вихідні дані до проєкту: середня абсолютна похибка моделі не перевищує 15%, коефіцієнт дотермінації вище 70%, швидкість відповіді при запиті на один або декілька маршрутів менше 5 секунд.
4. Зміст пояснювальної записки: опис предметної області, аналіз готових рішень, проєктування інформаційної системи, реалізація компонентів системи, реалізація інформаційної системи, математичне забезпечення, тестування та валідація системи.
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо): структурна схема, ER-діаграма бази даних, алгоритм підготовки ознак, діаграма використання системи.

6. Дата видачі завдання 8 березня 2025 року

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Ознайомлення з технічним завданням	14.04. – 20.04.2025	
2	Аналіз предметної області та огляд існуючих рішень	21.04. – 27.04.2025	
3	Розроблення функціональної моделі та проєктування бази даних	28.04. – 11.05.2025	
4	Дослідження та розроблення математичних моделей	12.05. – 18.05.2025	
5	Розроблення та реалізація інформаційної системи	19.05. – 01.06.2025	
6	Тестування системи	02.06. – 05.06.2025	
7	Оформлення текстової документації	06.06. – 09.06.2025	

Студент

Олександр ПРИМАК

Керівник

Андрій ПИСАРЕНКО

АНОТАЦІЯ

Інформаційна система прогнозування попиту на транспортні послуги у великих містах на основі машинного навчання.

Проект містить 62 с. тексту, 17 рисунків, 3 таблиці, посилання на 18 літературних джерел, додатки та 4 конструкторських документи.

ІНФОРМАЦІЙНА СИСТЕМА, ПРОГНОЗ ПОПИТУ, ГРОМАДСЬКИЙ ТРАНСПОРТ, МАШИННЕ НАВЧАННЯ, ТРАНСПОРТНА АНАЛІТИКА.

Об'єктом розроблення є інформаційна система прогнозування попиту на транспортні послуги у великих містах на основі машинного навчання.

Предметом – методи прогнозування попиту на транспортні послуги з використанням машинного навчання.

Мета розроблення – підвищення точності прогнозування попиту на транспортні послуги за рахунок застосування машинного навчання для оптимізації роботи міської транспортної системи.

У роботі використані методи регресійного прогнозування, аналізу часових рядів, обробки геоданих і просторового аналізу. Технічна реалізація базується на поєднанні Python, бібліотеки LightGBM для машинного навчання, Flask як серверного фреймворку, PostgreSQL з PostGIS для зберігання просторових даних і React для побудови вебінтерфейсу. Інтеграція з API Open-Meteo забезпечує отримання погодних умов, а API alerts.in.ua – даних про повітряні тривоги.

Результатом стала інформаційна система, яка дозволяє прогнозувати кількість пасажирів на обраних маршрутах у заданий час з урахуванням погодних та зовнішніх факторів. Інтерфейс підтримує побудову графіків, виведення таблиць, експорт у CSV/Excel, історію запитів і автоматичне оновлення даних.

Отримані результати можуть використовуватись міськими службами, перевізниками та аналітичними платформами, а сама система легко масштабується для інших міст і регіонів.

SUMMARY

Information system for predicting demand for transportation services in large cities based on machine learning.

The project contains 62 pages of text, 17 figures, 3 tables, references to 18 literary sources, annexes, and 4 design documents.

Keywords: information system, demand forecasting, public transport, machine learning, transport analytics.

The object of development is an information system for predicting demand for public transport services in large cities using machine learning.

The subject is the set of forecasting methods for transport demand based on machine learning algorithms.

The purpose of the development is to improve the accuracy of demand forecasting for public transport by applying machine learning techniques to optimize the performance of urban transportation networks.

The work employs methods of regression forecasting, time series analysis, geospatial data processing, and spatial analysis. The technical implementation is based on the integration of Python, the LightGBM library for machine learning, Flask as a server-side framework, PostgreSQL with PostGIS for storing spatial data, and React for developing the web interface. The system also integrates the Open-Meteo API for weather data and the alerts.in.ua API for air raid alerts.

The resulting information system enables passenger demand forecasting for selected routes at specific times, taking into account weather and external factors. The interface supports chart generation, table output, export to CSV/Excel, request history, and automatic data updates.

The results can be used by municipal authorities, transportation companies, and analytical platforms. The system is easily scalable for other cities and regions.

№ рядка	Формат	Позначення	Найменування	Кіл. аркушів	№ екз.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	A4	IC12.250БАК.005 ПЗ	Інформаційна система прогнозування	62		
6			попиту на транспортні послуги у			
7			великих містах на основі машинного			
8			навчання. Пояснювальна записка			
9	A3	IC12.250БАК.005 Д1	Інформаційна система прогнозування	1		
10			попиту на транспортні послуги у			
11			великих містах на основі машинного			
12			навчання. Структурна схема			
13	A3	IC12.250БАК.005 Д2	Інформаційна система прогнозування	1		
14			попиту на транспортні послуги у			
15			великих містах на основі машинного			
16			навчання. ER-діаграма бази даних			
17	A3	IC12.250БАК.005 Д3	Інформаційна система прогнозування	1		
18			попиту на транспортні послуги у			
19			великих містах на основі машинного			
20			навчання. Алгоритм підготовки ознак			
21	A3	IC12.250БАК.005 Д4	Інформаційна система прогнозування	1		
22			попиту на транспортні послуги у			
23			великих містах на основі машинного			
24			навчання. Діаграма використання			
25			системи			
26						
27						
28						
			IC12.250БАК.005 ТП			
Зм.	Лист	№ докум.	Підпис			
Розробив	Примак О. В.			Літ.	Арк.	Аркушів
Перевірів	Писаренко А.В.			Т	1	1
			Інформаційна система прогнозування попиту на транспортні послуги у великих містах на основі машинного навчання. Відомість дипломного проєкту	КПІ ім. Ігоря Сікорського		
				Група IC-12		
Затв.						

**Пояснювальна записка
до дипломного проєкту
на тему: «Інформаційна система прогнозування попиту
на транспортні послуги у великих містах на основі
машинного навчання»**

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	4
ВСТУП.....	5
1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ	7
1.1 Сучасні проблеми транспортних систем у великих містах.....	7
1.2 Методи прогнозування попиту в транспортній галузі	8
Висновки до розділу 1.....	10
2 АНАЛІЗ ГОТОВИХ РІШЕНЬ.....	11
2.1 Сервіс EasyWay	11
2.2 Мобільний застосунок Moovit	13
2.3 Мобільний застосунок Kyiv Digital	15
2.4 Комплексна платформа OpenPaths (Bentley)	17
Висновки до розділу 2.....	19
3 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ	20
3.1 Вимоги до системи	20
3.2 Архітектура системи	21
3.3 Вибір технологій та алгоритмів машинного навчання.....	23
3.4 Проєктування бази даних.....	26
Висновки до розділу 3.....	28
4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ.....	30
4.1 Попередня обробка даних	30
4.2 Навчання моделей та оцінка їх ефективності	32
4.3 Модуль збору та інтеграції даних.....	34
4.3.1 Робота з API	34
4.3.2 Обробка поточкових даних у реальному часі	35
4.3.3 Візуалізація даних	36

					ІС12.250БАК.005 ПЗ			
Зм.	Лист	№ докум.	Підпис		Інформаційна система прогнозування попиту на транспортні послуги у великих містах на основі машинного навчання. Пояснювальна записка	Літ.	Арк.	Аркушів
Розробив	Примак О.В.					Т		
Перевірив	Писаренко А.В.						2	63
						КПІ ім. Ігоря Сікорського		
Затв.						Група ІС-12		

4.4 Прогностичний модуль	37
Висновки до розділу 4.....	38
5 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ	40
5.1 Демонстраційний варіант системи (MVP)	40
5.2 Аналіз ефективності системи.....	44
5.3 Рекомендації щодо масштабування.....	45
Висновки до розділу 5.....	46
6 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ	47
6.1 Змістовна постановка задачі	47
6.2 Математична постановка задачі	48
6.3 Обґрунтування методу розв'язання.....	50
6.4 Опис методу розв'язання	51
Висновки до розділу 6.....	53
7 ТЕСТУВАННЯ ТА ВАЛІДАЦІЯ СИСТЕМИ	54
7.1 Стратегія тестування	54
7.2 Тестування прогнозів та граничних сценаріїв	55
7.3 Аналіз результатів тестування.....	56
Висновки до розділу 7.....	57
ВИСНОВКИ	59
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	61

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних.

ІС – інформаційна система.

API – Application Programming Interface.

CSV – Comma-Separated Values.

HTTP – HyperText Transfer Protocol.

JSON – JavaScript Object Notation.

LightGBM – Light Gradient Boosting Machine.

XGBoost – Extreme Gradient Boosting.

					ІС12.250БАК.005 ПЗ	Арк.
						4
Зм.	Лист	№ докум.	Підпис	Дата		

ВСТУП

Питання ефективного управління транспортними потоками постає вже протягом довгого періоду і стає все гострішим, особливо це видно на прикладі великих міст. Швидке зростання населення, збільшення кількості приватного транспорту, нерівномірне навантаження на громадський транспорт, а останнім часом сильно вплинув і воєнний стан в країні – усе це призводить до виникнення заторів, перевантаження окремих маршрутів та погіршення загального комфорту життя. Зі зростанням складності міської інфраструктури класичні методи планування та регулювання транспортного попиту перестають давати бажаний ефект. Стало очевидно, що для прийняття обґрунтованих рішень у транспортній сфері потрібні більш гнучкі та точні інструменти, які враховують множину факторів у режимі реального часу.

Останні роки в усьому світі спостерігається активне впровадження інтелектуальних транспортних систем, що поєднують великі обсяги даних з методами машинного навчання. Такі підходи дозволяють не лише фіксувати стан транспортної системи, а й передбачати її поведінку, наприклад, вплив окремих факторів на дорожню інфраструктуру. У результаті органи місцевого самоврядування, транспортні оператори та користувачі можуть краще планувати пересування, ефективніше розподіляти ресурси та уникати перевантажень. Попри це, в українських містах системи прогнозування попиту на транспорт залишаються на початковому етапі впровадження.

Більшість застосунків, які працюють із міським транспортом, обмежуються навігацією, відстеженням руху та розкладом. Функціональність повноцінного передбачення попиту зазвичай або взагалі відсутня, або реалізована лише для внутрішнього користування транспортними компаніями. З іншого боку, сучасні відкриті API надають доступ до погодних даних, інформації про повітряні тривоги, геопросторових даних про маршрути, що відкриває можливість будувати власні моделі прогнозу без необхідності доступу до закритих даних.

Тому актуальним є створення інформаційної системи, яка дозволяє зібрати всі релевантні дані в єдину модель, здатну будувати точкові прогнози попиту на громадський транспорт залежно від часу, району міста, погодних умов і надзвичайних подій. Така система може стати корисним інструментом для планування перевезень, адаптації маршрутів у кризових ситуаціях, оптимізації витрат і підвищення стійкості міської мобільності.

Метою цієї роботи є підвищення точності системи прогнозування попиту на транспортні послуги за рахунок застосування машинного навчання для оптимізації роботи міської транспортної системи.

Замість прямого розрахунку або використання статистичних шаблонів, така система повинна виявляти приховані закономірності в історичних даних, моделювати вплив зовнішніх факторів і забезпечувати адаптивність до змін.

Проектне рішення базується на поєднанні технологій обробки великих даних, просторового аналізу та машинного навчання. Основним напрямком дослідження є розроблення механізму прогнозування, що враховує погодні аномалії, сезонність, поведінкові патерни та неочікувані впливи (як-от повітряна тривога або зміни графіка). Важливо не лише побудувати математичну модель, а й реалізувати її в інтерфейсі, зрозумілому для кінцевого користувача – диспетчера, міського аналітика або пересічного пасажирка.

Очікувані результати можуть бути застосовані в системах управління транспортом, міських інформаційних платформах, інтегрованих панелях моніторингу та аналітики. Зокрема, такі модулі можуть стати частиною Smart City-інфраструктури, допомагаючи містам оперативно реагувати на зміни ситуації та забезпечувати безперервну мобільність для мешканців.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Сучасні проблеми транспортних систем у великих містах

Транспортна система великих міст України за останні роки переживає значне навантаження, спричинене як природним зростанням урбанізації, так і зовнішніми проблемами, серед яких значну роль відіграє воєнний стан. Міста на кшталт Києва, Львова чи Харкова зазнають тиску як з боку зростання кількості мешканців, так і через зміну поведінки пасажирів під впливом непередбачуваних факторів: сирен повітряної тривоги, нестабільного розкладу, погіршення інфраструктури.

Основна функція транспортної системи полягає у забезпеченні повсякденної мобільності населення. Однак ефективність цієї функції суттєво знижується, коли система не встигає адаптуватися до змін у структурі попиту. Більшість міст все ще використовують застарілу практику планування перевезень за фіксованими шаблонами: традиційні маршрути, статичні розклади, обмежений вплив контексту. У результаті виникає дисбаланс між реальними потребами користувачів та пропозицією транспортних ресурсів.

У дослідженні, проведеному McKinsey Global Institute, зазначається, що в умовах цифрової трансформації саме системи на основі даних та прогнозування стають головним інструментом для побудови "розумної мобільності" [1]. Ідеться не лише про навігацію або оптимізацію маршрутів, а про здатність аналізувати історичну поведінку, сезонні коливання, кліматичні умови та надзвичайні ситуації.

Зміни у структурі мобільності особливо помітні у мегаполісах. У Києві через регулярні повітряні тривоги робота транспорту часто порушується, метро переходить у режим укриття, наземні види транспорту зупиняються або змінюють маршрут. Це створює не тільки затримки, а й хаотичні зміщення попиту. Наприклад, одразу після закінчення тривоги спостерігається різкий стрибок навантаження на маршрути, які були раніше зупинені. У Харкові, який має схожу структуру з розвиненою мережею метро, спостерігається аналогічна ситуація. У Львові, де метро немає, головною перешкодою стала зміна профілю пасажирів,

адже місто прийняло сотні тисяч внутрішньо переміщених осіб, що змінило основні транспортні потоки.

Нові реалії вимагають переходу до адаптивних моделей управління транспортом. Потрібна система, яка враховує багатофакторний вплив: день тижня, годину, температуру повітря, видимість, сигнали тривоги – і на основі цього прогнозує попит наперед. Саме ця логіка стала основою реалізованої інформаційної системи, що побудована на принципах машинного навчання, збору потокових даних у реальному часі, інтеграції з відкритими та просторовою аналітики.

1.2 Методи прогнозування попиту в транспортній галузі

Методи прогнозування постійно змінюються, адаптуючись до нових обставин: зростання населення, нестабільна ситуація у містах, впровадження нових форматів мобільності (наприклад, мікромобільність або on-demand транспорт).

Загалом, методи прогнозування можна умовно поділити на класичні статистичні, машинного навчання, а також гібридні.

Класичні підходи включають регресійний аналіз, методи згладжування та авторегресивні моделі, наприклад ARIMA. Вони добре працюють у випадках, коли попит є регулярним і піддається сезонному або часовому аналізу. Такі моделі активно застосовувалися в плануванні громадського транспорту з 1970-х років і досі використовуються для стратегічного планування. Наприклад, ARIMA або SARIMA дозволяють моделювати пасажиропотік на основі часових рядів, враховуючи циклічні зміни (робочі дні/вихідні, пікові години, сезонність). Проте, у великих містах, де на попит впливають складні багатофакторні залежності, ефективність таких моделей часто обмежена.

З появою великих обсягів даних та обчислювальних потужностей дедалі більше застосовуються методи машинного навчання. Їх головна перевага полягає у здатності виявляти складні, нелінійні залежності між ознаками, що складно описуються вручну. Наприклад, моделі на основі дерев рішень, такі як Random

Forest або градієнтний бустинг, дозволяють враховувати десятки взаємопов'язаних параметрів: температура, тип транспорту, час доби, наявність повітряної тривоги тощо.

Найбільш поширеними алгоритмами для задач прогнозування попиту в міських умовах є XGBoost, LightGBM, нейронні мережі (особливо LSTM), а також гібридні підходи на їхній основі [2]. Останні дозволяють поєднувати переваги кількох методів, наприклад, використовувати ARIMA для загального тренду, а бустинг для врахування короткострокових флуктуацій. При цьому моделі, побудовані на деревоподібних структурах, часто виграють у стабільності, швидкості та простоті пояснення результатів, що робить їх популярними у сфері транспорту.

Інструменти прогнозування в транспортній галузі сьогодні все частіше інтегруються в загальні міські платформи (наприклад, інтегровані транспортні системи, цифрові двійники міст). Такі платформи збирають дані з численних джерел: GPS-трекерів, метеостанцій, камер відеонагляду, онлайн-запитів – і використовують їх для постійного оновлення моделей попиту в реальному часі. У містах на зразок Сінгапуру чи Хельсінкі реалізовані інфраструктури, які дозволяють прогнозувати транспортне навантаження на рівні кварталів чи навіть окремих вулиць.

У деяких країнах активно досліджується використання моделей глибокого навчання у поєднанні з графовими нейронними мережами для врахування топології транспортної мережі. Графові моделі виявили себе особливо ефективними при прогнозуванні трафіку на перехрестях і складних транспортних розв'язках [3].

Попри високий потенціал сучасних моделей, існують й відповідні перешкоди. Далеко не всі міста мають публічні дані достатньої якості, навіть для Києва публічну інформацію про транспортні дані почали оновлювати зовсім нещодавно, а інформації про поточний пасажиропотік міським транспортом не знайти, а це ускладнює навчання складних моделей. Крім того, навіть високоточна модель вимагає коректної інтеграції в робочий процес, а саме необхідне

забезпечення регулярного оновлення даних, можливість швидкого перенавчання, автоматизації запуску прогнозу, збереження результатів тощо.

Висновки до розділу 1

Розглянута загальна ситуація у сфері міських транспортних систем та основні підходи до прогнозування попиту на транспортні послуги. Окреслені основні проблеми, з якими стикаються великі міста – це перевантаження маршрутів у години пік, недостатня адаптивність до змін зовнішнього середовища, а також обмежена ефективність традиційного планування транспорту. Особливої актуальності ці проблеми набувають в умовах постійних змін інфраструктури, погоди, воєнного стану, зростання населення та появи нових форматів мобільності.

Розглянуті головні методи прогнозування, які використовуються в транспортній галузі. Показано, що класичні статистичні залишаються актуальними для базових сценаріїв, але не здатні повноцінно враховувати велику кількість контекстних змінних, таких як метеоумови чи надзвичайні події. З огляду на це, дедалі більшого поширення набувають моделі машинного навчання, зокрема градієнтний бустинг, дерева рішень, нейромережі, які демонструють вищу точність і краще пристосовуються до динамічних умов.

Окремо наголошено, що ефективне прогнозування можливе лише при наявності якісних даних і правильної архітектури обробки: від збору інформації з різних джерел, до інтеграції прогнозів у робочі процеси транспортного планування. Розвиток smart-city рішень та розумних транспортних систем суттєво стимулює впровадження таких підходів у провідних містах світу.

Підсумовуючи, можна зазначити, що сучасні міста потребують гнучких, адаптивних рішень для аналізу транспортного попиту. Використання моделей машинного навчання у поєднанні з актуальними даними створює підґрунтя для ефективнішого управління транспортними потоками, зменшення перевантажень і підвищення зручності для пасажирів.

					IC12.250БАК.005 ПЗ	Арк.
						10
Зм.	Лист	№ докум.	Підпис	Дата		

2 АНАЛІЗ ГОТОВИХ РІШЕНЬ

Станом на 2025 рік існують як споживчі застосунки для навігації громадським транспортом, так і професійні платформи для моделювання транспортних систем. Більшість рішень зосереджені або на зручності користувача, або на аналітиці для міст, але повноцінні прогностичні системи залишаються поодинокими. Далі проаналізовані найбільш релевантні приклади.

2.1 Сервіс EasyWay

EasyWay – один із перших українських онлайн-сервісів, орієнтованих на громадський транспорт. Його основне призначення – забезпечення користувачів актуальною інформацією про маршрути, графіки руху та розташування транспорту в реальному часі. Сервіс працює у понад 30 містах України, зокрема в Києві, Львові, Харкові, Дніпрі, Одесі, Запоріжжі та інших. EasyWay має вебверсію та мобільний застосунок для Android і iOS.

Інтерфейс сервісу досить простий: користувач може обрати своє місто, після чого відкривається інтерактивна мапа з позначеними маршрутами транспорту. Доступна фільтрація за типами: автобуси, трамваї, тролейбуси, маршрутки. Для кожного маршруту можна переглянути повну схему руху, перелік зупинок, час роботи та інтервали між рейсами. У деяких містах підтримується відображення місцеперебування транспорту онлайн, хоча ця функція залежить від підключення відповідної інфраструктури перевізників. Крім базових можливостей, EasyWay також надає дані про орієнтовний час прибуття транспорту на зупинки, що особливо зручно в години пік, також дозволяє здійснювати пошук оптимального маршруту між точками, що робить його корисним як для мешканців, так і для гостей міста. У деяких містах реалізована система сповіщень про зміни у маршрутах або затримки, а також можливість збереження обраних маршрутів для швидкого доступу. На рисунку 2.1 зображено інтерфейс сервісу EasyWay з прикладом візуалізації маршрутів громадського транспорту у місті Київ.

					ІС12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		11

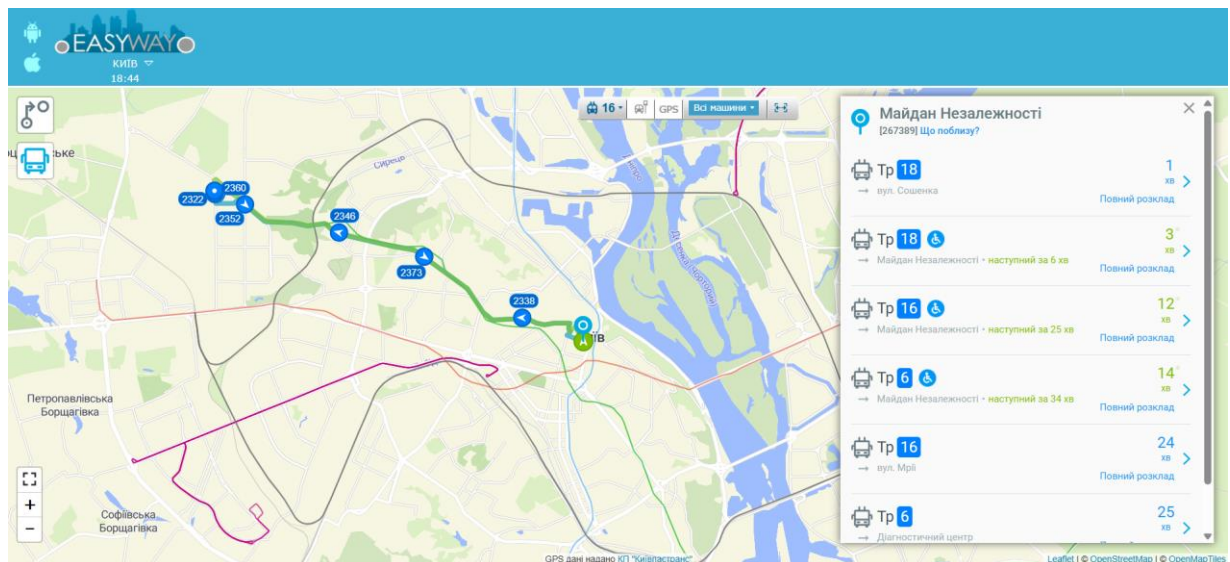


Рисунок 2.1 – Інтерфейс вебверсії сервісу EasyWay [4]

Серед головних функцій: пошук оптимального маршруту з точки А в точку Б. Алгоритм враховує пересадки, піші переходи, відстань та орієнтовний час у дорозі. Для цього EasyWay використовує дані у форматі GTFS або відкриті API перевізників. Проте у більшості випадків сервіс ґрунтується на наперед завантажених графіках руху, без врахування реальної динаміки дорожньої ситуації або попиту.

Варто зазначити, що EasyWay не використовує алгоритми прогнозування. Уся інформація, яка подається користувачу, є або статичною, або поточною. Це означає, що система працює виключно у реактивному режимі, тобто фіксує факт, але не передбачає майбутню ситуацію. В умовах повітряних тривог, погодних змін або нештатних ситуацій така модель взаємодії є дещо обмеженою.

Сервіс також не інтегрується з міськими інформаційними системами типу «розумного міста» або платформами обліку попиту. Вся аналітика, якщо і наявна, залишається внутрішньою і не надається користувачам. Доступ до API для сторонніх розробників отримати важко, що ускладнює інтеграцію з іншими платформами чи побудову на основі EasyWay аналітичних рішень для муніципалітетів.

Попри це, EasyWay лишається зручним інструментом для пасажирів, які хочуть спланувати поїздку в межах конкретного міста. Його перевага полягає в

охопленні багатьох міст і зручний доступ до маршрутів навіть у менш цифровізованих населених пунктах. Проте з огляду на відсутність можливості прогнозувати зміну попиту або вплив подій, цей інструмент не є придатним для транспортного планування в умовах нестабільності чи пікового навантаження.

2.2 Мобільний застосунок Moovit

Moovit – один із найпопулярніших застосунків для навігації у громадському транспорті, що працює у понад трьох тисячах міст по всьому світу, зокрема в Києві, Львові, Харкові та Одесі. Мета застосунку – надати користувачам зручний інструмент для планування поїздок. Після введення початкової та кінцевої точки система пропонує кілька варіантів маршруту з урахуванням пересадок, часу очікування, типу транспорту та навіть пішохідних сегментів. Доступна інтерактивна мапа з маршрутами, інформація про прибуття транспорту на зупинку в реальному часі, сповіщення про затримки, відображення наближення транспорту, можливість зберігати улюблені маршрути та користуватися додатком у режимі офлайн. На рисунку 2.2 видно інтерфейс Moovit із побудованим маршрутом.

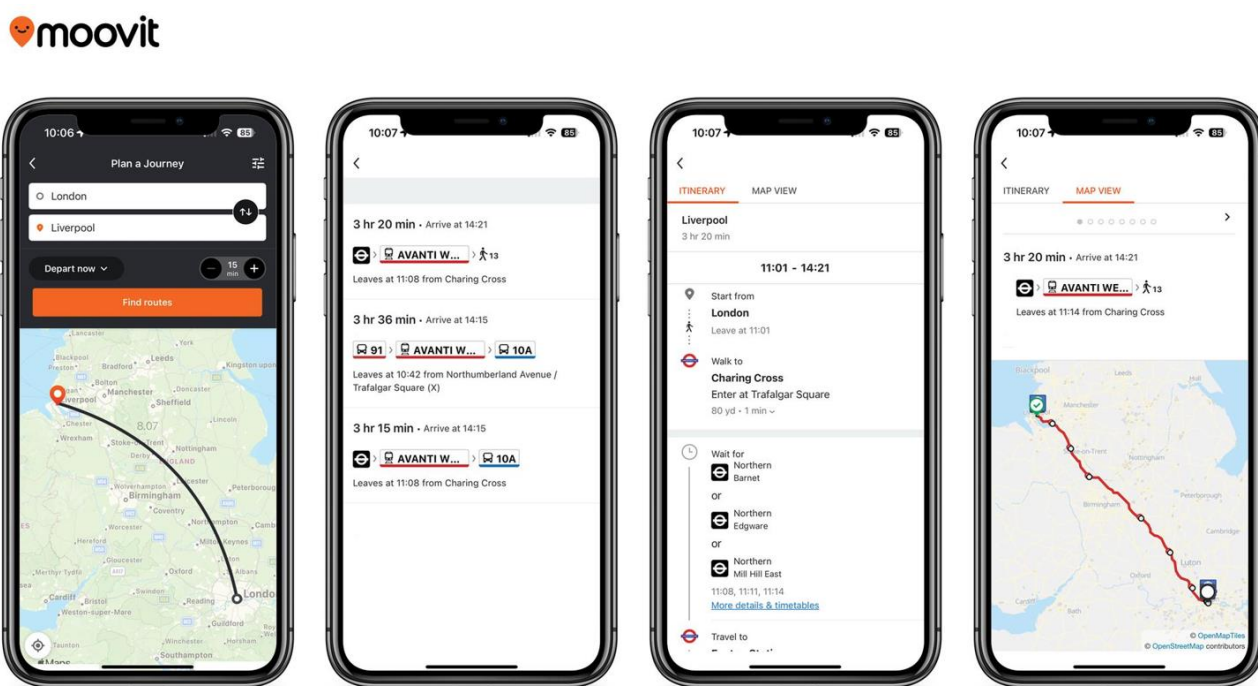


Рисунок 2.2 – Інтерфейс застосунку Moovit [5]

					IC12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		13

Окрім мобільного застосунку, орієнтованого на звичайного користувача, платформа Moovit має окремий продукт для муніципалітетів і операторів транспорту – аналітичний модуль Moovit Urban Mobility Analytics (MUMA). Цей інструмент використовує анонімізовані дані мільйонів користувачів, які взаємодіють із застосунком, щоб формувати повну картину транспортної динаміки в місті. Система надає доступ до глибокої статистики в розрізі часу та простору, що особливо цінно для міського планування та адаптації транспортної мережі під актуальні потреби. Серед функцій аналітики:

- аналіз пасажирських потоків у межах обраної зони, включно з часом відправлення, прибуття та використовуваними видами транспорту;
- визначення найбільш завантажених ділянок маршрутів;
- аналіз того, скільки пасажирів заходить/виходить на станціях, тривалість пересадок;
- ідентифікація центральних вузлів маршрутів й основних потоків трафіку.

Такі звіти дозволяють містам побачити реальну структуру пасажирських потоків, зокрема пікові години, нетипові завантаження або зони, де попит не відповідає доступності транспорту. На рисунку 2.3 представлений приклад демонстраційного інтерфейсу MUMA.

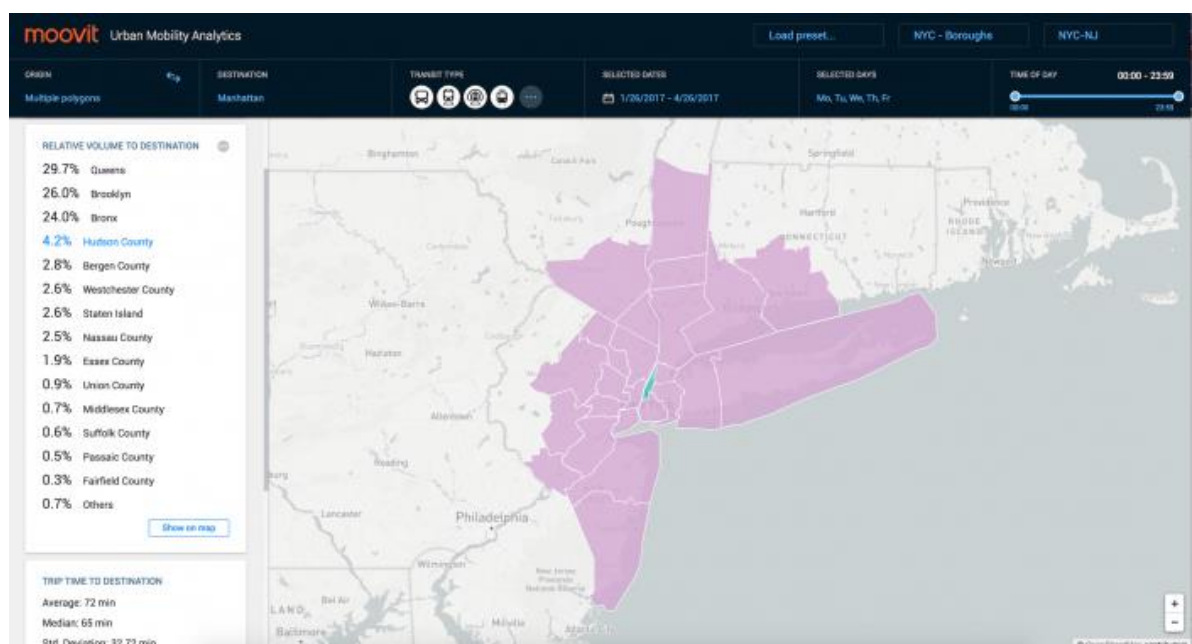


Рисунок 2.3 – Інтерфейс Moovit Urban Mobility Analytics [6]

Хоча для користувачів Moovit виглядає як звичайний навігаційний застосунок, на рівні міської аналітики він стає джерелом великих даних, які можуть використовуватись для оптимізації маршрутної мережі, розкладів і планування ресурсів. Важливо, що доступ до MUMA надається тільки офіційним партнерам компанії: муніципалітетам і транспортним операторам, які укладають відповідні угоди. Дані, які збираються, залишаються закритими й не публікуються у відкритому доступі. Крім того, ефективність аналітики залежить від активності користувачів Moovit у місті: чим більше людей користуються застосунком, тим точніші й повніші звіти. Водночас у містах із меншою цифровою проникністю аналітична потужність MUMA може бути обмеженою.

2.3 Мобільний застосунок Kyiv Digital

Kyiv Digital – офіційний мобільний додаток від Київської міської ради, створений 2021 року. Спочатку слугував для реалізації базових міських сервісів: оплати парковки, поповнення транспортної картки, оплати комунальних та штрафів. З другої половини 2022 року додаток трансформувався в найважливіший інформаційний інструмент для мешканців столиці під час війни [7].

Він інформує про повітряні тривоги з push-сповіщеннями, показує карту працюючих бомбосховищ та відкритих аптек, АЗС, їдалень, а також дає можливість повідомляти про пошкодження інфраструктури. Додано функції воєнного часу: сервіси допомоги армії та можливість повідомляти місце обстрілів.

Що стосується транспорту, Kyiv Digital надає інформацію про маршрути міського транспорту: час прибуття на зупинку, графіки руху, можливість поповнити або купити цифровий QR-квиток у додатку. Є також функція е-петицій і голосування за проекти міського бюджету, що показує тенденцію до застосування додатку як інструменту е-демократії. В користувацькому інтерфейсі можна побачити, наприклад, мапу зупинок і підказку про транспортування QR-квитка через Wallet.

					IC12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		15

Безпековий функціонал включає повідомлення про повітряні тривоги: червоне сповіщення та направлення до найближчого укриття, а пізніше – зелене сповіщення про безпеку. Також доступна інтерактивна мапа укриттів із Wi-Fi та предметами першої потреби. На рисунку 2.4 зображено інтерфейс застосунку.

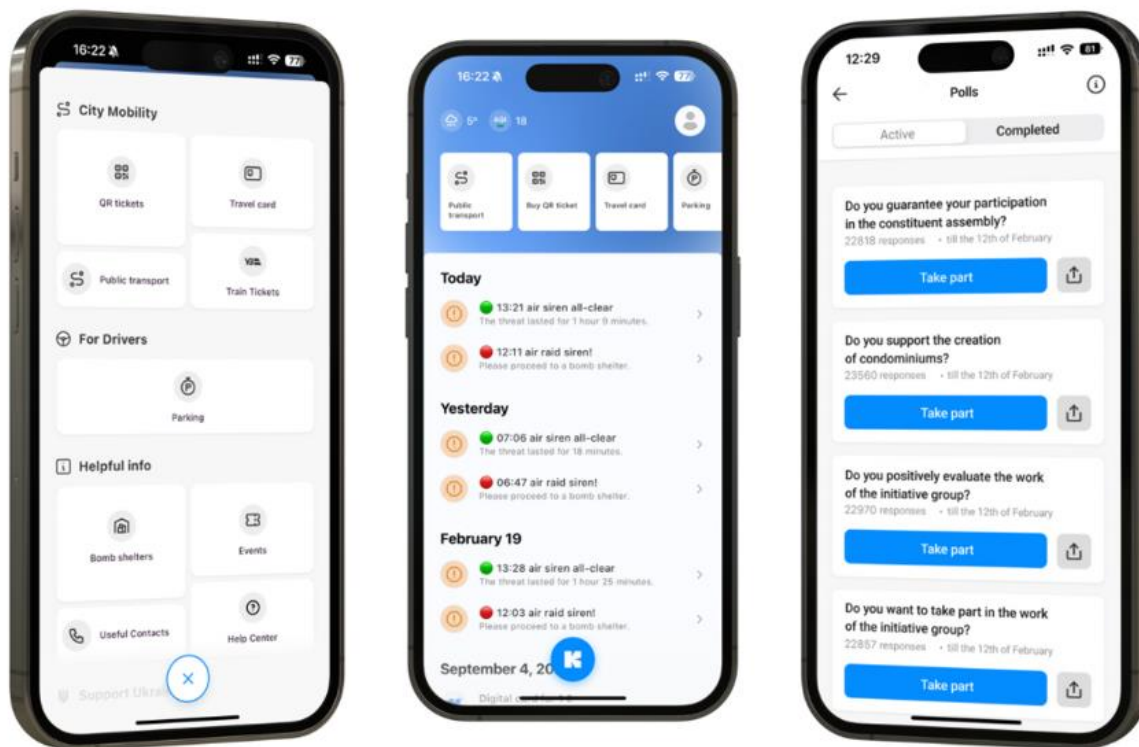


Рисунок 2.4 – Інтерфейс Kyiv Digital [8]

Незважаючи на багатофункціональність, є обмеження: дані про транспортування базуються на API і GPS повідомленнях, але поки що відсутня справжня аналітика чи прогнозування попиту. Інформація про трафік, затори або зміни руху є реактивною. Удосконалення включає інтеграцію student ID, е-петицій, parking-сервісів та документів в Diia. Однак аналітичного модуля попит-пропозиція в транспорті немає.

Kyiv Digital демонструє, як муніципальний застосунок може поєднувати транспорт, комунальні сервіси, безпеку та демократію, зосередившись на потребах користувачів у кризових умовах. Недоліком є відсутність просунутої аналізу

транспортних потоків або вбудованого планування маршрутів, а це залишає прогалини у глобальному управлінні транспортною системою.

2.4 Комплексна платформа OpenPaths (Bentley)

OpenPaths – це програмна платформа від компанії Bentley Systems, яка використовується для моделювання, прогнозування та аналізу транспортних потоків у містах. Її призначення відрізняється від споживчих мобільних застосунків: OpenPaths не орієнтована на звичайного користувача, а створена для інженерів, транспортних аналітиків і органів міського управління. Система дозволяє створювати транспортні моделі, аналізувати ефекти змін у схемах руху, оцінювати навантаження на мережу і моделювати попит у різних сценаріях. Додатково, платформа OpenPaths дозволяє створювати моделі мобільності населення на основі даних про щільність руху, що відкриває можливості для побудови прогнозних сценаріїв у випадку аварій, перекриттів доріг або надзвичайних ситуацій. Система також підтримує аналіз багаторівневої інфраструктури, дозволяючи враховувати залежності між транспортними вузлами, включаючи метрополітен, зупинки автобусів і пересадкові центри. Такі можливості є важливими для антикризового реагування в умовах міських транспортних систем.

Програмне середовище включає кілька модулів, які взаємодіють між собою. OpenPaths EMME та CUBE використовуються для побудови мультимодальних моделей руху на рівні міста або району з урахуванням маршрутів, зупинок, зонального попиту та інших параметрів. Додатковий модуль CityPhi дозволяє створювати інтерактивну тривимірну візуалізацію транспортної ситуації. Модуль AGENT відповідає за моделювання поведінки окремих учасників руху, базуючись на принципах activity-based моделювання. DYNAMIQ, у свою чергу, дозволяє створювати детальне моделювання руху транспорту у динаміці, враховуючи світлофорні цикли, затори, обмеження швидкості та інші фактори, що й зображено на рисунку 2.5.

					IC12.250БАК.005 ПЗ	Арк.
						17
Зм.	Лист	№ докум.	Підпис	Дата		

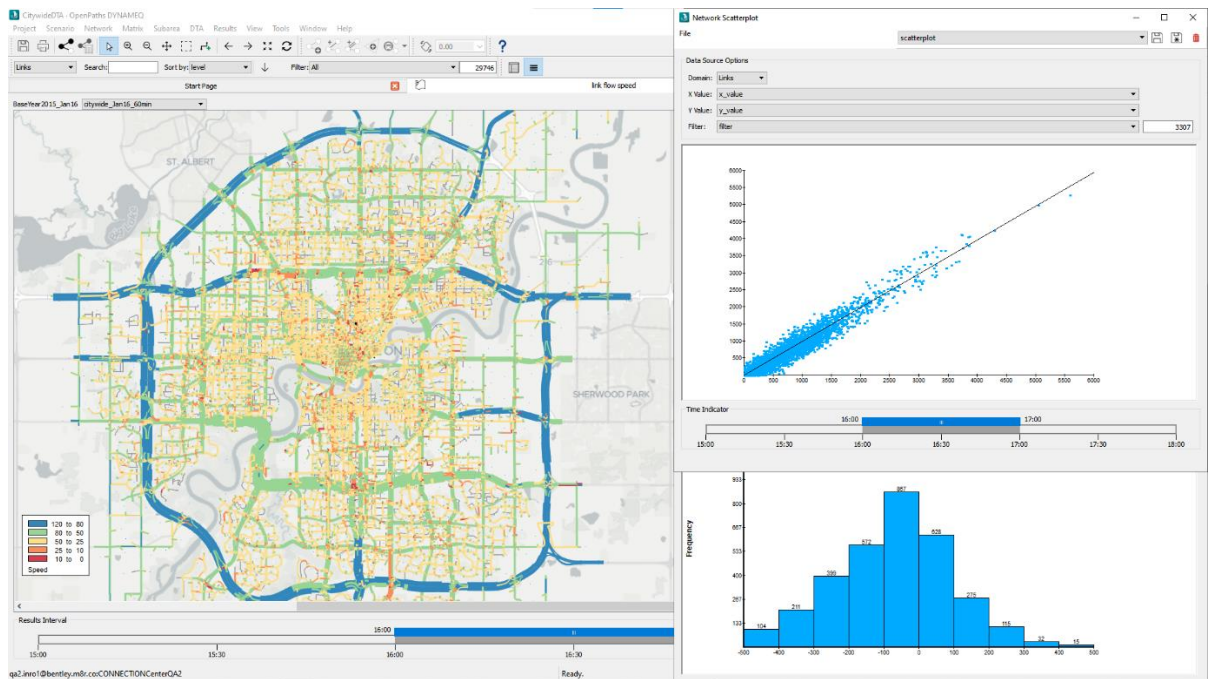


Рисунок 2.5 – Моделювання руху транспорту на платформі OpenPaths [9]

Платформа підтримує роботу з реальними просторовими даними, у тому числі з GPS-треками, GTFS-файлами, демографічною статистикою, схемами дорожньої мережі. Це дає змогу формувати високоточні моделі, які відображають реальні умови у місті. Візуалізація результатів виконується у вигляді графіків, мап, анімацій та просторових діаграм, що значно спрощує інтерпретацію даних для подальшого ухвалення управлінських рішень.

Серед сильних сторін OpenPaths – здатність моделювати сценарії на перспективу. Це можуть бути як невеликі зміни, наприклад перенесення зупинки, так і великі – відкриття нової лінії транспорту чи впровадження обмежень на рух приватного транспорту в центрі міста. Модель дозволяє побачити, як зміниться потік поїздки, навантаження на маршрути, швидкість пересування і загальна ефективність мережі.

Разом із цим використання OpenPaths потребує кваліфікованих спеціалістів і ресурсів. Система не є простою у впровадженні, вимагає значної підготовки, обчислювальних потужностей і придбання комерційної ліцензії. Це робить платформу малопридатною для невеликих міст або команд без досвіду роботи з

транспортними моделями. Також варто зауважити, що OpenPaths не має мобільного інтерфейсу і не використовується кінцевими користувачами безпосередньо.

Висновки до розділу 2

Проаналізовані сучасні програмні рішення, що використовуються у сфері міської мобільності. Вивчені приклади охоплюють як застосунки для кінцевих користувачів, так і професійні платформи для аналітики й моделювання. Застосунки на зразок Moovit і Kyiv Digital надають зручний доступ до розкладів, маршрутів та навігації, однак працюють переважно в реактивному режимі. Вони не формують прогнози попиту, не враховують зміну транспортної поведінки під впливом зовнішніх подій і не інтегровані з аналітичними модулями міського рівня.

Окремим випадком є аналітична платформа MUMA від Moovit, яка вже дозволяє оцінювати пасажиропотоки, завантаженість маршрутів і характер пересадок. Проте цей інструмент залишається закритим, доступним лише за ліцензією для партнерських організацій, і не має відкритого API чи публічного доступу до зібраних даних. Його використання потребує домовленостей із розробником, а точність залежить від активності користувачів у конкретному місті.

Ще один приклад, OpenPaths від Bentley, ілюструє підхід до стратегічного транспортного планування на основі складного моделювання. Цей інструмент дозволяє працювати з великими даними, візуалізувати потоки, будувати сценарії «що буде, якщо...» та оцінювати вплив інфраструктурних змін. Проте його складність і вартість обмежують використання лише професійними командами й великими міськими центрами.

У результаті можна зробити висновок, що на ринку є чіткий поділ між застосунками для повсякденного використання та інструментами для глибокого аналізу. Але жоден із розглянутих рішень не забезпечує відкритого, адаптивного прогнозування попиту на основі поєднання факторів.

3 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Вимоги до системи

У класичному підході до розроблення програмного забезпечення всі вимоги до системи поділяються на функціональні та нефункціональні. Функціональні вимоги описують поведінку системи з погляду її реакції на зовнішні події, тоді як нефункціональні визначають обмеження та очікувані характеристики роботи: продуктивність, безпеку, зручність [10].

До функціональних вимог належить можливість формування прогнозу попиту на громадський транспорт з урахуванням простору (район або маршрут) і часу (година або день). Користувач повинен мати змогу задати параметри запиту, а система – повернути числове значення очікуваного навантаження у визначеній геозоні. Прогнозування повинно враховувати історичні закономірності, погодні умови, день тижня, час доби, сигнали повітряної тривоги, тип транспорту та інші контекстні змінні.

Крім основного функціоналу, система має підтримувати супутні операції: обробку вхідних даних (агрегація, нормалізація, очищення), збереження результатів запитів до моделі, повторне відтворення попередніх прогнозів, перегляд історії запитів. Також важливо реалізувати можливість експорту результатів у поширених форматах – CSV, Excel, що дозволить інтегрувати дані з іншими системами або використовувати їх в аналітиці вручну.

Інтерфейс користувача повинен дозволяти легко формувати запит, переглядати прогноз у табличному та графічному вигляді, а також бачити відображення результатів на карті міста. При цьому система має логувати усі запити, зберігати час, параметри, результат і статус обробки кожного прогнозу. Це необхідно як для верифікації точності, так і для внутрішньої перевірки якості роботи.

Окремо слід виділити функцію періодичного оновлення даних. Система повинна автоматично отримувати нову інформацію з різних джерел: погодні API,

					IC12.250БАК.005 ПЗ	Арк.
						20
Зм.	Лист	№ докум.	Підпис	Дата		

сервіси сповіщення про тривоги, транспортні дані. Такий підхід дозволить забезпечити актуальність вхідних параметрів і підвищити точність прогнозу.

Нефункціональні вимоги охоплюють характеристики, які не стосуються безпосередньо дій системи, але мають велике значення для її роботи. Найважливішими є надійність, масштабованість, безпечність та швидкодія. Надійність означає, що система повинна стабільно працювати в умовах навантаження, мати механізми обробки помилок, автоматичні повторні запити до API в разі збоїв, а також створення резервних копій бази даних. У разі виникнення помилки система не повинна зупинятись повністю, а має відновлювати роботу із збереженням стабільного стану.

Швидкодія системи є важливою, особливо якщо мова йде про інтерактивне використання. Користувач не повинен чекати на відповідь моделі довше ніж кілька секунд. Це стосується як обробки одиночних запитів, так і пакетного прогнозування для кількох маршрутів або зон одночасно. Для цього необхідно враховувати обсяг даних, тип обчислень і архітектуру серверної частини.

Ще одним важливим аспектом є масштабованість. З часом кількість даних, що обробляються, зростатиме: нові райони, маршрути, історичні записи, прогнозні запити. Система повинна бути побудована так, щоб її можна було масштабувати без повної переробки. Це стосується і компонентів бази даних, і логіки обробки, і моделей машинного навчання.

У підсумку, система має відповідати низці вимог: приймати гнучкі вхідні параметри, швидко повертати прогноз, працювати з різнорідними даними, підтримувати автоматичне оновлення та архівацію, зберігати історію запитів, бути надійною, масштабованою і безпечною.

3.2 Архітектура системи

Архітектура системи розробляється з урахуванням необхідності обробки великої кількості вхідних даних, виконання машинного прогнозування та представлення результатів у зручній для користувача формі. У зв'язку з цим обрана

					ІС12.250БАК.005 ПЗ	Арк.
						21
Зм.	Лист	№ докум.	Підпис	Дата		

модель архітектури ґрунтується на трирівневій структурі: рівень представлення, рівень логіки обробки та рівень даних.

На рівні представлення взаємодія з користувачем забезпечується через вебінтерфейс, який дозволяє формувати запити, передавати параметри, а також переглядати результати прогнозування у різних форматах. Логіка обробки відповідає за отримання запиту, підготовку вхідних ознак та запуск моделі машинного навчання. Результат повертається користувачу у вигляді числових значень або візуалізації. Рівень даних забезпечує зберігання вхідної інформації, проміжних значень, результатів прогнозів, а також структурує історію запитів.

Загальна структура системи включає низку окремих компонентів, які виконують свої завдання, але взаємодіють між собою через чітко визначені інтерфейси. Це дозволяє забезпечити гнучкість, масштабованість і можливість подальшого розвитку кожного модуля без зміни всієї системи.

У таблиці 3.1 наведено загальний перелік основних компонентів та їх призначення в рамках архітектури.

Таблиця 3.1 – Компоненти та їх призначення в архітектурі

Компонент	Загальне призначення
Рівень представлення	Формування запитів користувача, виведення результатів прогнозу
Серверна логіка обробки	Прийом параметрів, обробка, підготовка ознак, взаємодія з моделлю
Модуль машинного навчання	Побудова прогнозу на основі заданих параметрів
Рівень даних	Зберігання вхідної, історичної та результатної інформації

У межах архітектури передбачається, що кожен рівень функціонує незалежно від інших, що забезпечує слабкий зв'язок між компонентами. Наприклад, оновлення моделі машинного навчання не вимагає змін на рівні представлення або

структури даних. Це дозволяє поступово оновлювати та вдосконалювати систему, зберігаючи її цілісність.

Крім основної взаємодії між рівнями, система підтримує обмін через внутрішні API або окремі точки доступу, що спрощує інтеграцію з іншими сервісами або модулями. Така архітектурна побудова дає змогу реалізувати як автономне рішення для одного міста, так і адаптовану версію для кількох регіонів.

Архітектура також враховує потенційне зростання кількості запитів, обсягів даних та складності прогнозів. Закладена структура дозволяє розширювати функціональність за рахунок додавання нових типів даних, джерел інформації або модулів виводу без значного втручання в основну логіку. Структурна схема наведена на кресленику IC12.250БАК.005 Д1.

3.3 Вибір технологій та алгоритмів машинного навчання

Під час розроблення системи важливим етапом стало обґрунтування вибору технологій і методів, які забезпечать необхідну функціональність, гнучкість та точність. Оскільки система працює з реальними просторово-часовими даними, включаючи погодні показники, сигнали повітряної тривоги та особливості маршрутної мережі, важливо було підібрати інструменти, здатні ефективно обробляти великі обсяги табличної інформації, взаємодіяти з геоданими, а також працювати у форматі REST-запитів.

Для розроблення серверної частини системи було обрано мову програмування Python, яка вважається стандартом у сфері аналітики даних, машинного навчання й побудови прикладних сервісів. Перевагою Python є велика кількість підтримуваних бібліотек і готових рішень, що дозволяють реалізовувати повноцінні обробки даних: від отримання та трансформації вхідних даних до збереження результатів і передавання прогнозів клієнтам у вигляді REST API. У межах проекту використовуються такі бібліотеки, як pandas для роботи з табличними даними, sqlalchemy для взаємодії з базою, joblib для серіалізації моделей, sklearn і lightgbm для машинного навчання. Завдяки динамічному

					IC12.250БАК.005 ПЗ	Арк.
						23
Зм.	Лист	№ докум.	Підпис	Дата		

синтаксису та високій читабельності Python дозволяє легко модифікувати код системи, масштабувати її функціонал і адаптувати до нових сценаріїв без значних змін у логіці.

Як фреймворк для побудови API було обрано Flask, легкий, мінімалістичний, але гнучкий інструмент для створення серверної логіки [11]. Flask дозволяє не лише формувати маршрути й обробляти HTTP-запити, а й просто інтегрується з машинним навчанням, бібліотеками візуалізації, зовнішніми API та фронтендом. Його модульна структура забезпечує чітке розділення відповідальностей: окремо реалізовані функції для підготовки ознак, обробки запитів користувача, обчислення прогнозу та взаємодії з базою. У поєднанні з Python це дало змогу швидко реалізувати повноцінний API з підтримкою багатьох маршрутів, пакетного прогнозування, роботи з базою та перевірки валідності вхідних параметрів.

Для зберігання й обробки даних у системі використовується реляційна СУБД PostgreSQL з просторовим розширенням PostGIS. PostgreSQL надає стабільну та перевірену платформу для зберігання структурованих даних, а PostGIS додає підтримку просторових об'єктів, що важливо у проєкті, де дані маршруту мають географічну прив'язку. У системі зберігаються геозони, траєкторії маршрутів, погодні умови з координатами, сигнали повітряних тривог із часовими мітками. Завдяки геофункціям є можливість фільтрувати маршрути за належністю до зон, будувати зв'язки між погодою і просторовими об'єктами, а також формувати прогноз із урахуванням місця розташування. Постійна підтримка транзакцій, цілісність даних, масштабованість і сумісність із Python через SQLAlchemy роблять PostgreSQL/PostGIS логічним вибором для системи, де працюють одночасно декілька джерел даних.

У ролі моделі машинного навчання застосовується LightGBM (LGBMRegressor) – високопродуктивна реалізація градієнтного бустингу від Microsoft [12]. Цей алгоритм обраний через поєднання трьох основних переваг: висока швидкість тренування, здатність працювати з великим числом ознак та висока точність прогнозів. LightGBM працює з числовими й категоріальними ознаками, дозволяє інтегрувати one-hot закодовані змінні, а також має автоматичну

обробку пропущених значень без необхідності попередньої фільтрації. Порівняно з XGBoost, LightGBM показує кращу продуктивність на стадії оптимізації гіперпараметрів, а CatBoost хоча і має подібну точність, потребує більше часу для налаштувань та обмежує гнучкість при роботі з нестандартними типами даних. Модель була серіалізована через joblib і завантажується один раз при запуску сервера, що мінімізує час відповіді на запити.

Фронтенд частина реалізована з використанням популярного JavaScript-фреймворку для створення інтерфейсів – React [13]. Його компонентна архітектура забезпечує високу гнучкість і можливість масштабування системи без втрати керованості. Завдяки React реалізований односторінковий вебзастосунок (SPA), який працює без повного перезавантаження сторінки при кожній дії користувача. Це важливо для проєктів, де передбачено часту взаємодію з API, відображення карт, графіків і великих таблиць. У реалізації інтерфейсу використані такі бібліотеки як:

- react-leaflet для інтерактивної карти, яка показує геометрію маршрутів у реальному масштабі;
- recharts – для побудови графіків попиту по годинах;
- react-select – для зручного вибору маршрутів і параметрів прогнозу.

Додаткові можливості інтерфейсу включають перегляд історії запитів, експорт результатів у CSV/Excel, перевірку введених параметрів та повідомлення про помилки.

Завдяки обраному стеку технологій система функціонує як цілісний програмний продукт, де всі компоненти: модель, база, API, інтерфейс – узгоджені між собою. А завдяки модульній структурі кожен компонент системи можна змінювати незалежно від інших. Така комбінація інструментів дозволяє ефективно масштабувати застосунок, швидко адаптувати його до нових вимог та зберігати стабільність у роботі як локально, так і на сервері. Це дозволяє розробникам швидко адаптувати систему під змінні вимоги або додавати нові функції без серйозних переробок.

					ІС12.250БАК.005 ПЗ	Арк. 25
Зм.	Лист	№ докум.	Підпис	Дата		

У таблиці 3.2 наведено узагальнений перелік основних технологій та аргументи на користь їхнього вибору.

Таблиця 3.2 – Обрані технології та обґрунтування їх використання

Компонент	Вибрана технологія	Обґрунтування використання
Мова програмування	Python 3.13.1	Широка підтримка ML, обробки даних, API, бібліотеки pandas, LightGBM, SQLAlchemy
Фреймворк для API	Flask	Простий у використанні, добре масштабується, легко поєднується з ML-модулями
База даних	PostgreSQL, PostGIS	Підтримка просторових даних, сумісність з GeoJSON, ефективна для зональних запитів
Модель машинного навчання	LightGBM (LGBMRegressor)	Висока швидкість і точність, підтримка категоріальних змінних, зручність у застосуванні
Інтерфейс користувача	React	Побудова SPA, інтеграція з мапою (Leaflet), компонентами фільтрації та візуалізації

3.4 Проектування бази даних

Для збереження прогнозів, параметрів запитів, погодинних характеристик, сигналів повітряної тривоги, а також просторових об'єктів побудована реляційна база даних. Основою обрана БД PostgreSQL із розширенням PostGIS, яке дозволяє працювати з геометрією маршрутів та зон, здійснювати просторові фільтрації й агрегації.

Схема побудована з урахуванням вимог до структури даних, які надходять у модель прогнозування. Зокрема, зберігаються погодинні значення температури, опадів, типу погоди, сигнали тривоги з часом початку і завершення, типи транспорту, маршрути з їхньою геометрією та середнім добовим попитом. Уся ця інформація у процесі запиту об'єднується за ознаками та передається у модель LightGBM. Тому структура БД повинна підтримувати гнучке злиття даних через поля, які є спільними для кількох таблиць, передусім `datetime`, `zone_id` і `route_id`.

В основі проєктування лежить принцип розділення логіки: довідники (`zones`, `routes`), динамічні дані (`weather`, `alerts`), зв'язки (`route_zones`) і лог користувацьких дій (`requests`, `forecasts`). Наприклад, зв'язок між маршрутами і геозонами винесений в окрему таблицю `route_zones`, що дозволяє маршруту належати до кількох зон. Це важливо, оскільки модель формує погодні ознаки не для маршруту напряду, а для зон, через які він проходить. Отже, просторовий зв'язок між маршрутом і погодними умовами реалізований логічно й масштабовано.

Таблиця `requests` зберігає кожен запит користувача, а також час, на який зроблений прогноз. Вона містить також `json`-поля з вхідними параметрами та результатами у збереженому вигляді. При цьому безпосередньо прогнозовані значення (кількість пасажирів) зберігаються в окремій таблиці `forecasts`, що дозволяє розділити логіку запиту та обробки. Кожен прогноз містить `zone_id`, `forecast_time`, `route_id` і `request_id` як зв'язкові поля, що дає змогу легко повертатися до будь-якого збереженого результату або проводити перевірку на актуальність.

Проєктування виконано з урахуванням нормалізації. У схемі немає надлишкового дублювання, зв'язки між таблицями логічно побудовані через ключі. Наприклад, погодні дані для кожної години не дублюються на рівні маршруту, достатньо раз записати показники для зони, а потім з'єднати їх через `route_zones`. Аналогічно, тривоги не дублюються на рівні маршрутів, а просто фільтруються за часом при запиті: маршрут вважатиметься «під впливом тривоги», якщо вона активна в момент прогнозу.

Геометричні поля в таблицях `routes` і `zones` зберігаються у форматі PostGIS `geometry`. Це дозволяє виконувати запити напряду в базі для визначення перетину,

					IC12.250БАК.005 ПЗ	Арк.
						27
Зм.	Лист	№ докум.	Підпис	Дата		

належності, близькості, тощо. У процесі запиту, коли користувач обирає маршрут, система через таблицю route_zones визначає відповідні зони, далі бере погодні показники по zone_id + datetime, додає ознаки з тривоги, фільтрує через alerts, і формує ознаки для моделі. ER-діаграма бази даних наведена на кресленику ІС12.250БАК.005 Д2.

Нижче подано таблицю 3.3 з коротким описом основних сутностей.

Таблиця 3.3 – Опис таблиць бази даних

Назва таблиці	Призначення в системі
zones	Зберігає райони Києва з геометрією (полігонами); використовується для просторової прив'язки погоди й тривоги
weather	Інформація про погоду в кожній зоні: температура, опади, видимість, тип погоди
alerts	Дані про сигнали повітряної тривоги (початок, кінець, тип); фільтруються по містах
routes	Опис усіх маршрутів громадського транспорту (назва, тип, геометрія, середній добовий попит)
route_zones	Прив'язка маршрутів до зон, які вони обслуговують
requests	Зберігає історію запитів користувача: параметри запиту, час, зона, маршрут, статус
forecasts	Містить результати прогнозу: попит, час, зона, маршрут, ідентифікатор запиту (як логічний зовнішній ключ)

Висновки до розділу 3

Сформульовані основні принципи побудови інформаційної системи. Визначені вимоги до функціоналу охоплюють можливість прогнозування з урахуванням просторових і часових параметрів, підтримку історії запитів, обробку погодних і тривожних факторів, а також подальший розвиток системи з точки зору

масштабованості та швидкодії. Також приділена увага нефункціональним аспектам, таким як надійність, безпека та можливість розширення.

Обрана архітектура має трирівневу структуру, що дозволяє розмежувати логіку, зберігання даних та інтерфейс взаємодії з користувачем. Такий підхід забезпечує гнучкість у розробленні та дає змогу надалі змінювати або розширювати окремі модулі без порушення цілісності всієї системи. Компоненти системи взаємодіють через стандартні інтерфейси, що також полегшує інтеграцію із зовнішніми джерелами даних.

У межах вибору технологій проаналізовані кілька варіантів алгоритмів машинного навчання та інструментів серверної й базової інфраструктури. Обґрунтоване використання Python і Flask як серверної основи, PostgreSQL з PostGIS для просторового зберігання даних, LightGBM як ефективного алгоритму для регресійного прогнозу попиту, а також React для побудови вебінтерфейсу. Обрані рішення відповідають вимогам точності, швидкодії та сумісності з іншими елементами системи.

Розроблена база даних забезпечує чітке розмежування інформації, необхідної для роботи моделі, зберігання історії прогнозів та підтримки просторових запитів. В основі лежить структура, що дозволяє гнучко оперувати транспортними маршрутами, погодою, тривогами й користувацькими запитами. Передбачена підтримка геометрії, основних зв'язків між сутностями та можливість розширення структури у майбутньому.

4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

4.1 Попередня обробка даних

Ефективність моделі машинного навчання значною мірою залежить від якості та узгодженості вхідних даних. Тому перед запуском процесу навчання виконана комплексна попередня обробка, яка охоплювала отримання даних з різних джерел, їхню очистку, трансформацію, об'єднання та формування повного набору ознак. Основною задачею є створення цілісного датафрейму, в якому кожен рядок представляє один маршрут у певний момент часу з усіма супутніми характеристиками: погодою, типом транспорту, тривогами, часом доби тощо.

Дані про попит на транспорт отримані безпосередньо від Київської міської державної адміністрації. У відповідь на офіційний запит надано таблиці у форматі Excel з кількістю зареєстрованих поїздок за травень 2025 року. Інформація містила розбивку за маршрутом, видом транспорту та кількістю пасажирів. У зв'язку з тим, що не було погодинної деталізації, для формування тренувального набору даних застосоване евристичне моделювання: кількість поїздок за місяць була поділена на 31, а далі розподілена по годинах дня з урахуванням пікових періодів, комендантської години та денних коливань. Також враховані дні тижня: для п'ятниці, вівторка–четверга, вихідних попит коригувався згідно з типовими поведінковими патернами. До кожного значення попиту доданий випадковий шум у межах 7% для уникнення надмірної одноманітності.

Погодні дані отримувалися через прямі запити до Open-Meteo API [14], яке дозволяє завантажити погодинний архів. Для навчання використовувались показники з 26 травня по 1 червня включно. Зібрана інформація включала температуру повітря, кількість опадів, тип погоди та індекс видимості. Кожен погодинний запис пов'язаний із конкретною зоною міста, до якої належить маршрут, через таблицю просторового зв'язку `route_zones`. Геометрія маршрутів отримана за допомогою запитів до Overpass API [15], який надає доступ до даних OpenStreetMap.

					IC12.250БАК.005 ПЗ	Арк.
						30
Зм.	Лист	№ докум.	Підпис	Дата		

4.2 Навчання моделей та оцінка їх ефективності

Після завершення попередньої обробки даних сформований повний датафрейм розміром понад 34 000 рядків, кожен з яких відповідав одній годині для певного маршруту в заданий день тижня. Усі записи містили набір ознак, що включало: погодні умови, тип транспорту, час доби, день тижня, активність повітряної тривоги, середній добовий попит на маршрут, а також категоріальні ознаки, закодовані за допомогою one-hot encoding. Підготовлений набір даних використовувався для тренування моделі машинного навчання, яка реалізовує прогнозування пасажиропотоку на рівні окремих маршрутів у конкретний час.

Вибір алгоритму прогнозування зумовлений як практичними, так і технічними критеріями. У результаті аналізу альтернативних підходів обрана модель градієнтного бустингу LightGBM, яка поєднує високу точність, швидке навчання та ефективність при роботі з великим числом вхідних ознак. Модель використовувалась у вигляді LGBMRegressor з наступними основними параметрами:

- objective="regression" – регресійна постановка задачі;
- n_estimators=100 – кількість ітерацій бустингу;
- learning_rate=0.1 – швидкість навчання;
- random_state=42 – для відтворюваності результатів.

Для збереження хронологічної структури даних та уникнення інформаційних витоків використовувався метод TimeSeriesSplit з п'ятьма розбиттями. Це дало змогу послідовно навчати модель на більш ранніх часових відрізках і тестувати її на пізніших, що відповідає реальним умовам прогнозування.

Навчання проводилося на 80% даних (27 417 записів), ще 20% залишалось для тестування. Для оцінки якості моделі застосовувались класичні метрики регресії: MAE (Mean Absolute Error) становила 36.61, RMSE – 68.12, а коефіцієнт детермінації R^2 досяг 0.9607, що свідчить про високу узгодженість прогнозу з фактичними значеннями.

					IC12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		32

Додатково проводилась перехресна перевірка з використанням TimeSeriesSplit. Модель поступово тренувалась на збільшуваних підмножинах даних, зберігаючи хронологічну послідовність. Результати крос-валідації показали стабільну точність у перших трьох фолдах (MAE ~ 10), однак у 4–5 фолдах, які охоплювали більш "свіжі" дані, спостерігалось збільшення похибки (MAE ~ 33), що можна пояснити впливом змінних зовнішніх факторів. У середньому по п'яти фолдах MAE становила 19.61, що вважається прийнятним значенням для задачі міського прогнозування з високою варіативністю поведінки пасажирів.

Результати продемонстрували стабільну точність при тестуванні: модель коректно відтворювала патерни попиту з урахуванням часу доби, дня тижня та погодних умов. Для аналізу вкладу окремих ознак у прогнозуванні побудована діаграма важливості ознак, наведена на рисунку 4.2.

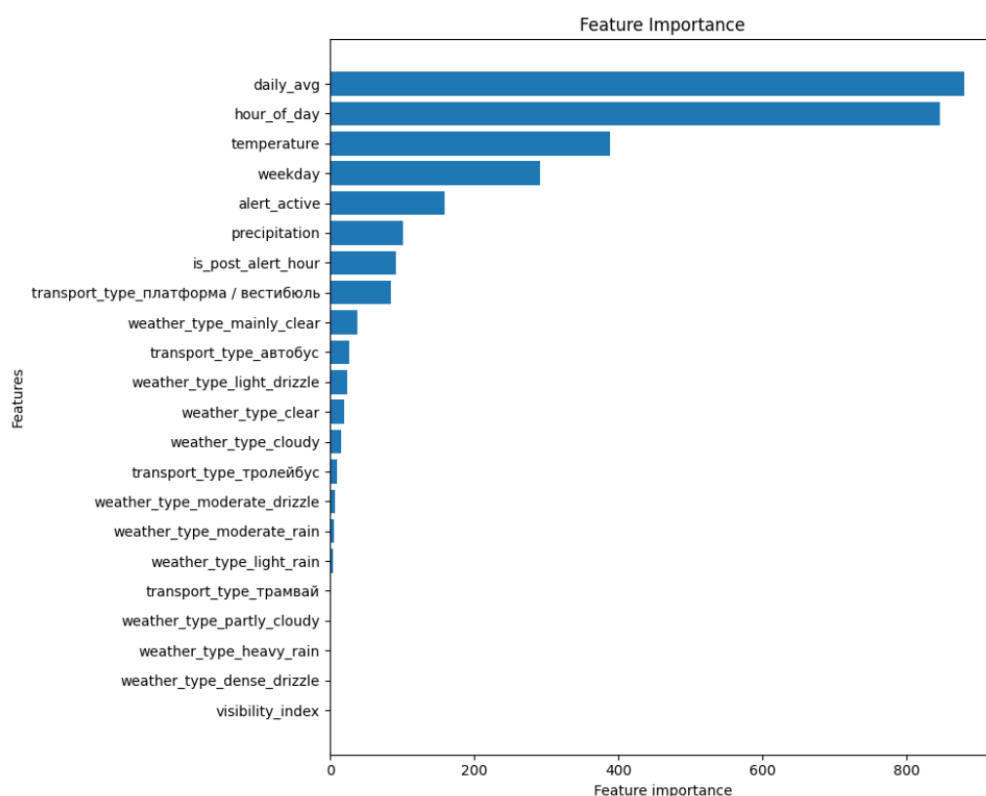


Рисунок 4.2 – Важливість ознак для моделі прогнозування

Перевірка моделі проводилась у кілька етапів. Спочатку – вручну в середовищі розроблення: оцінка метрик, аналіз залишків, порівняння передбачених

та фактичних значень. Після цього модель була інтегрована до серверної частини системи, з можливістю отримання прогнозів через REST API. Тестування продовжилось у режимі реального часу через запити /predict, /predict_batch_routes та /predict_range. Усі запити логуються до бази, що дозволяє зберігати історію прогнозів для подальшого аналізу. Додатково передбачено візуалізацію результатів у вебінтерфейсі з можливістю відображення попиту на графіку та експорту даних.

Підсумовуючи сказане, навчена модель продемонструвала належну якість та стабільність, а обраний підхід дозволяє інтегрувати прогнозування в систему, адаптовану до міських реалій.

4.3 Модуль збору та інтеграції даних

4.3.1 Робота з API

Щоб система прогнозування могла оперувати актуальними контекстними даними, до її складу включений модуль, що автоматично звертається до зовнішніх джерел через API. У продакшн-версії наразі інтегровані два основні сервіси: Open-Meteo API для отримання погодних умов по кожній зоні та alerts.in.ua API для моніторингу повітряних тривог у Києві. Обидва API використовуються регулярно, за заданим графіком, без участі користувача.

Погодні дані отримуються щогодини. Система надсилає запити до Open-Meteo із зазначенням координат зон і зберігає відповідь у таблиці weather. Задля уникнення дублювань, усі старі прогнози з datetime більше або рівним поточному часу видаляються перед записом нового прогнозу. Це дозволяє підтримувати в базі лише релевантні дані.

Інтеграція з alerts.in.ua побудована на двох типах запитів: active.json для отримання поточних тривог і month_ago.json для виявлення завершення раніше активних тривог. Оновлення відбувається кожні 15 хвилин. Перед записом у таблицю alerts система перевіряє, чи існує вже запис із таким самим start_time і zone_id. Якщо відповідний запис знайдено, а тривога ще не завершена, її end_time оновлюється.

Збір даних реалізований у вигляді окремих фонових потоків, що запускаються одночасно з Flask-сервером. Це дозволяє не створювати окрему чергу задач чи залежність від scheduler'a. У майбутньому цю логіку можна буде перенести на APScheduler або Celery з Redis для розподіленого виконання, однак наразі використовується простий механізм threading + time.sleep().

Усі зібрані дані зберігаються в базу PostgreSQL і формують контекст для подальшого прогнозування. Завдяки цьому прогноз моделі може адаптуватися до актуального стану погоди або наявності тривоги, не потребуючи ручного оновлення або втручання адміністратора.

4.3.2 Обробка поточкових даних у реальному часі

У структурі системи реалізована підтримка потокової обробки даних як для оновлення вхідних параметрів, так і для генерації прогнозів у відповідь на дії користувача. Це дозволяє моделі адаптуватися до змін середовища без потреби запуску вручну або попереднього обчислення значень.

Оновлення погодних умов та сигналів повітряної тривоги здійснюється автоматично. Щогодини система звертається до Open-Meteo API для отримання прогнозу погоди по кожній зоні міста. Нові записи замінюють попередні значення у таблиці weather, що гарантує актуальність даних без дублювання. Окремо, кожні 15 хвилин, запускається фоновий процес перевірки активних та завершених тривог через API alerts.in.ua. За потреби, система створює новий запис у таблиці alerts або оновлює end_time у вже існуючому, якщо завершено наявну сирену. Обидві процедури виконуються у фонових потоках, ініціалізованих при старті Flask-сервера.

Окрім цього, система активно реагує на дії користувача в режимі реального часу. У вебінтерфейсі користувач самостійно обирає маршрут, часовий проміжок і дату прогнозу. У відповідь на ці дії формується запит, який передається до серверної частини через один із ендпоінтів API. На основі параметрів запиту система формує відповідний набір ознак, з урахуванням останніх погодних даних,

активних сирен, типу транспорту та середнього добового попиту. Цей набір передається до модуля машинного навчання, який генерує прогноз у реальному часі.

Отриманий результат повертається користувачу в інтерфейсі та одночасно зберігається в базі: запит фіксується у таблиці requests, а відповідь – у forecasts. Це дозволяє як відображати результат негайно, так і зберігати повну історію запитів для подальшого аналізу або повторного використання. Така побудова забезпечує збереження хронології, відтворюваність результатів і дає змогу оцінювати навантаження на систему.

4.3.3 Візуалізація даних

Візуалізація результатів прогнозування є важливою складовою інформаційної системи, оскільки вона дозволяє користувачеві швидко оцінити стан попиту, порівняти маршрути, побачити тенденції протягом дня або тижня. У системі реалізований сучасний односторінковий вебінтерфейс на основі React, який взаємодіє з серверною частиною через REST API.

Основними елементами візуалізації є мапа міста, графіки, таблиця результатів та історія запитів. Мапа реалізована на базі Leaflet з підтримкою GeoJSON. Вона дозволяє відобразити траєкторії маршрутів та виділити обрані зони. Користувач може навести на виділену траєкторію на карті і побачити інформацію про номер маршруту.

Графіки реалізовані за допомогою Recharts. Вони відображають зміну попиту протягом доби: можна побачити пікові години, періоди зниження, а також вплив зовнішніх чинників, таких як тривога або опади. Для кожного прогнозу будується окремий часовий графік, де по осі X – години доби, по осі Y – прогнозоване значення попиту.

Таблична частина містить історію всіх запитів, збережених у таблиці requests. Користувач може переглянути, коли зроблений прогноз, для якого маршруту, який

					IC12.250БАК.005 ПЗ	Арк.
						36
Зм.	Лист	№ докум.	Підпис	Дата		

результат і які параметри запиту. При натисканні на запис відкривається розширене вікно з деталями.

Уся інформація у фронтенді отримується динамічно через запити до Flask API. Це дозволяє підтримувати актуальність без перезавантаження сторінки, а також розширювати функціональність без зміни структури фронтенду. Інтерфейс побудований так, щоб його можна було легко адаптувати під мобільні пристрої або змінити структуру фільтрів для нових міст.

4.4 Прогностичний модуль

Центральним елементом системи прогнозування є модуль, що безпосередньо відповідає за передбачення попиту на транспортні послуги. У його основі лежить модель машинного навчання, навчена на структурованих ознаках, що відображають погодинні характеристики маршрутів, погодні умови, сигнали тривоги, тип транспорту, добові патерни попиту тощо. Для реалізації модуля обраний алгоритм градієнтного бустингу LightGBM, який забезпечує високу точність і швидкодію при обробці табличних даних із великою кількістю змінних.

Модель збережена у форматі .pkl за допомогою `joblib.dump()` і завантажується один раз під час запуску Flask-сервера. Це дозволяє уникнути повторного тренування або читання файлу при кожному запиті, що істотно знижує час відповіді. Завантаження здійснюється у головному тілі програми `app.py`, і після цього модель готова до використання через набір API-ендпоінтів.

Підготовка вхідних ознак здійснюється в уніфікованій функції всередині `_predict_batch(...)`. Незалежно від типу запиту (один маршрут, кілька маршрутів або діапазон годин), система формує датафрейм, що включає всі необхідні ознаки: погодні показники (температура, тип погоди, опади, видимість), стан сирен, середній добовий попит, годину доби, день тижня та тип транспорту. Для категоріальних ознак застосовується one-hot encoding. Щоб гарантувати сумісність з навченою моделлю, перед формуванням прогнозу перевіряється наявність усіх колонок із `feature_list.json`, і у разі нестачі додаються порожні (з нулями).

					IC12.250БАК.005 ПЗ	Арк.
						37
Зм.	Лист	№ докум.	Підпис	Дата		

API підтримує кілька сценаріїв прогнозування:

- /predict – точковий прогноз для одного маршруту в один момент часу;
- /predict_batch_routes – прогноз одразу для кількох маршрутів;
- /predict_range – прогноз на кілька годин наперед (для одного або декількох маршрутів).

Усі вони використовують спільну логіку для підготовки даних і обробляються за одним викликом функції прогнозу. Користувач у запиті вказує лише route_id (наприклад, "Автобус 105") і datetime. Зона визначається автоматично через таблицю route_zones, що спрощує інтерфейс та виключає помилки з прив'язкою. У відповідь система повертає JSON-об'єкт із прогнозованим значенням попиту.

Жодні додаткові метрики, як-от довірчий інтервал чи пояснення прогнозу, не повертаються, однак повна інформація про параметри запиту та результат зберігається у таблиці requests (вхідні дані) та forecasts (прогноз). Це дозволяє переглядати історію, трасувати запити та візуалізувати результати без необхідності повторного обчислення.

Щодо оптимізації, то реалізовані кілька рішень. По-перше, lazy loading моделі дозволяє уникнути надмірного навантаження на сервер. По-друге, використовується батчевий режим прогнозування, де модель отримує одразу масив ознак і повертає масив результатів. Це особливо корисно при запитах /predict_batch_routes або прогнозах на тривалий період. У системі не використовуються multiprocessing або асинхронні підходи, однак структура модуля допускає масштабування в майбутньому.

Висновки до розділу 4

Розглянуті практичні аспекти реалізації основних компонентів інформаційної системи прогнозування попиту на транспортні послуги. Показано, як на основі оброблених вхідних даних, зібраних із зовнішніх API, формується

повноцінний набір ознак, який використовується для передбачення кількості пасажирів у заданий момент часу.

На етапі попередньої обробки об'єднані дані з кількох джерел: статистика пасажиропотоку, погодні умови, сигнали повітряної тривоги, тимчасові параметри. Для обліку реальних змін попиту використані евристичні правила, які дозволили отримати синтетичну, але наближену до реальності картину годинного попиту. Це стало основою для ефективного навчання моделі.

Модель машинного навчання реалізована на базі LightGBM. Вона навчена на понад 34 тисячах прикладів і демонструє високу точність та стабільну роботу при крос-валідації. Застосовані TimeSeriesSplit, one-hot encoding для категоріальних змінних, а також оптимізовану систему підготовки ознак. Уся логіка побудована так, щоб бути уніфікованою для різних сценаріїв прогнозування: точкових, батчевих або по діапазону часу.

Модулі збору й оновлення даних автоматизовані: погода оновлюється щогодини, сигнали тривоги – кожні 15 хвилин. Крім цього, реалізований механізм обробки запитів користувача в реальному часі: маршрути та час прогнозу задаються через інтерфейс, і відповідь формується миттєво. Система зберігає всі запити та результати у базі, що дозволяє будувати історію, візуалізувати попит і повторно використовувати дані.

Завдяки поєднанню автоматизації збору, адаптивної логіки обробки та швидкої моделі прогнозування вдалося створити модуль, що поєднує точність, гнучкість і практичну цінність для реального використання в умовах міського середовища.

					IC12.250БАК.005 ПЗ	Арк.
						39
Зм.	Лист	№ докум.	Підпис	Дата		

5 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

5.1 Демонстраційний варіант системи (MVP)

На основі створеної серверної архітектури реалізований повноцінний вебінтерфейс, який дозволяє взаємодіяти з прогножною системою у зручному форматі, він зображений на рисунку 5.1.

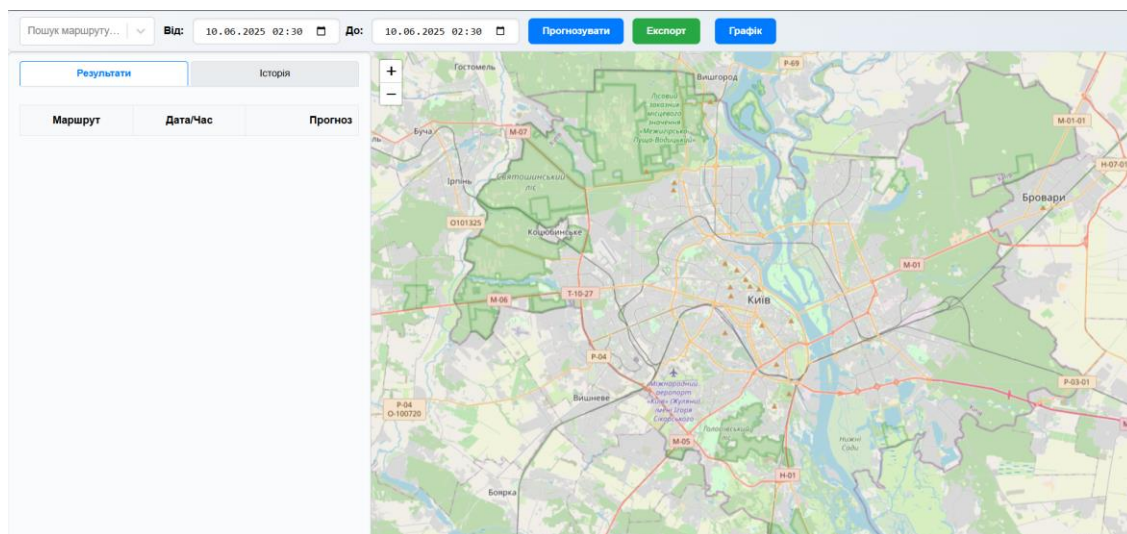


Рисунок 5.1 – Інтерфейс інформаційної системи

Інтерфейс реалізований як односторінковий застосунок, що працює через браузер і поєднує карту міста, форми запити, графіки, таблиці та історію прогнозів.

У верхній частині розташована панель керування. Користувач може обрати один або кілька маршрутів громадського транспорту за допомогою пошуку з автодоповненням. Далі задається часовий інтервал прогнозування: дата й години, які потрібно спрогнозувати. Після цього достатньо натиснути кнопку «Прогнозувати», і система миттєво повертає результат.

Результати відображаються одразу у двох форматах: у вигляді таблиці з прогнозами та на інтерактивній мапі. Мапа реалізована на базі Leaflet із використанням геометрій маршрутів. Кожен маршрут зображений кольоровою лінією, яка дозволяє швидко зорієнтуватись у просторі. При наведенні курсору можна побачити назву маршруту, це зображено на рисунку 5.2.

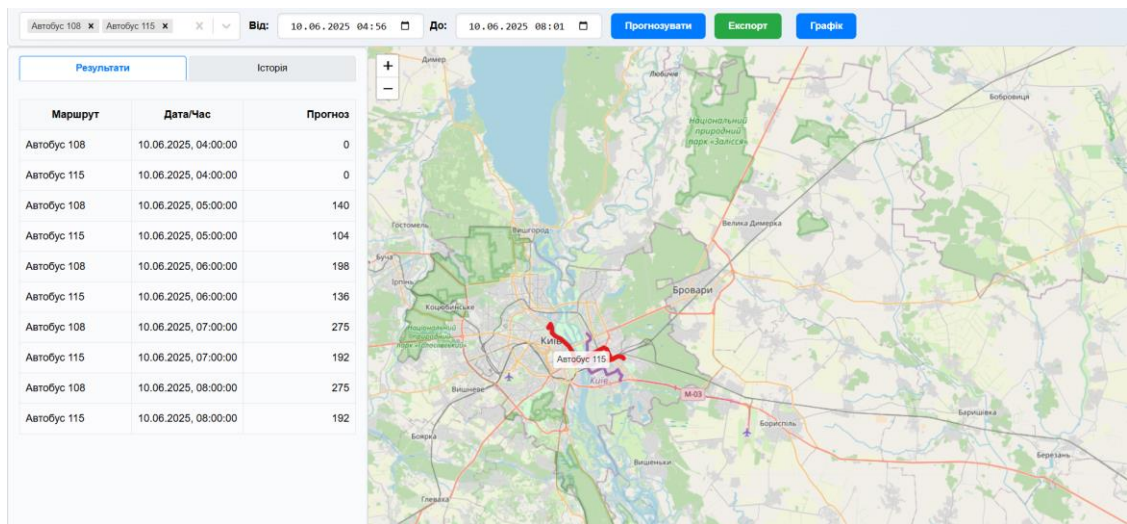


Рисунок 5.2 – Функціонал системи

Таблиця результатів відображає прогноз для кожного обраного маршруту в обрані години. Вона містить три колонки: назву маршруту, дату/час прогнозу та передбачену кількість пасажирів. Усі дані згруповані, і можна прокручувати великий список результатів без втрати навігації.

На рисунку 5.3 зображена окремо реалізована вкладка «Історія», яка містить список усіх попередніх запитів користувача.

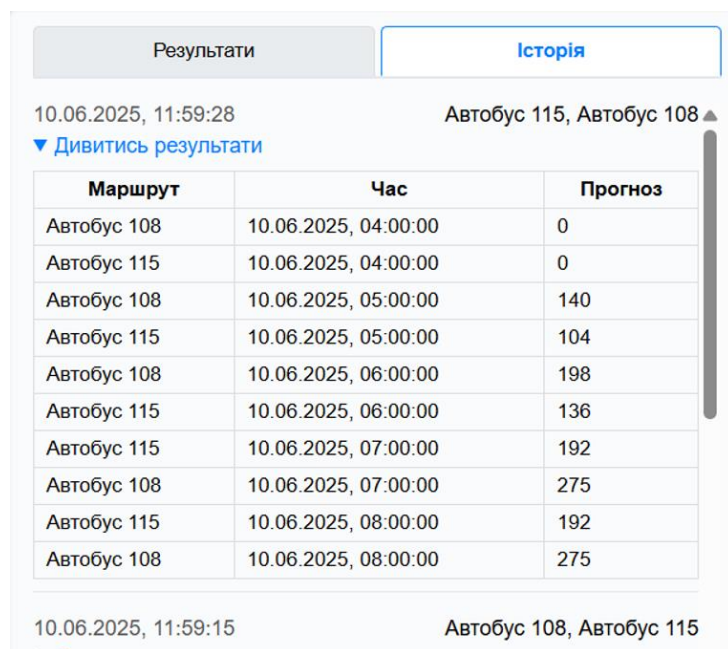


Рисунок 5.3 – Вкладка «Історія»

Для кожного з запитів них вказується час його формування, список маршрутів, а також можна переглянути збережений результат. Історія дозволяє повернутись до попередніх прогнозів без повторного обчислення, що знижує навантаження на модель та забезпечує простий аудит. Усі записи згруповані за часом запиту та можуть бути згортані або розгортані для перегляду деталей. Це зручно при роботі з великою кількістю маршрутів та частих прогнозах.

Додатково передбачений функціонал побудови графіка. Після натискання кнопки «Графік» відкривається модальне вікно з лінійним графіком попиту по годинах для обраних маршрутів. Це дозволяє побачити динаміку зміни навантаження у пікові та непікові години. Кожен маршрут має окрему лінію з кольоровим позначенням, а при наведенні доступні точні значення прогнозу по годинах, що представлено на рисунку 5.4.

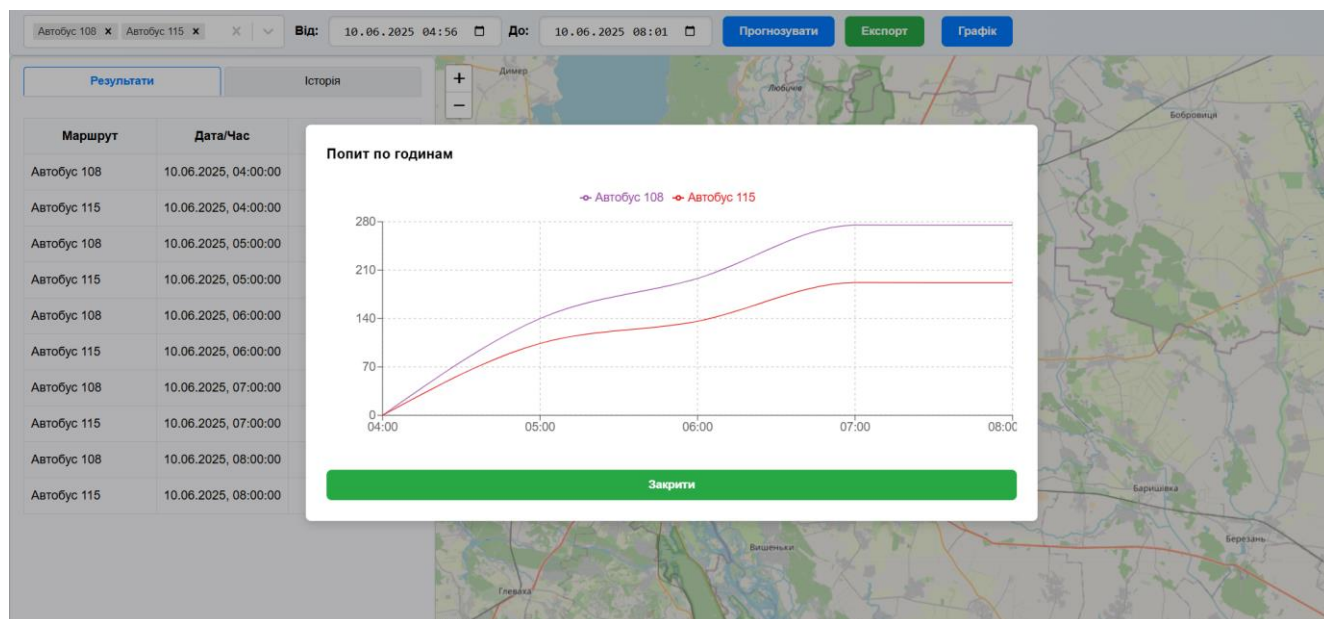


Рисунок 5.4 – Функціонал побудови графіків

Для зручності аналітики реалізована функція експорту. Кнопка «Експорт» дозволяє зберегти результати у форматі CSV або Excel, якщо експорт здійснюється з вкладки «Результати», зберігаються лише поточні прогнози. При використанні функції у вкладці «Історія» експортуються останні 200 запитів із параметрами та прогнозами, це показано на рисунку 5.5.

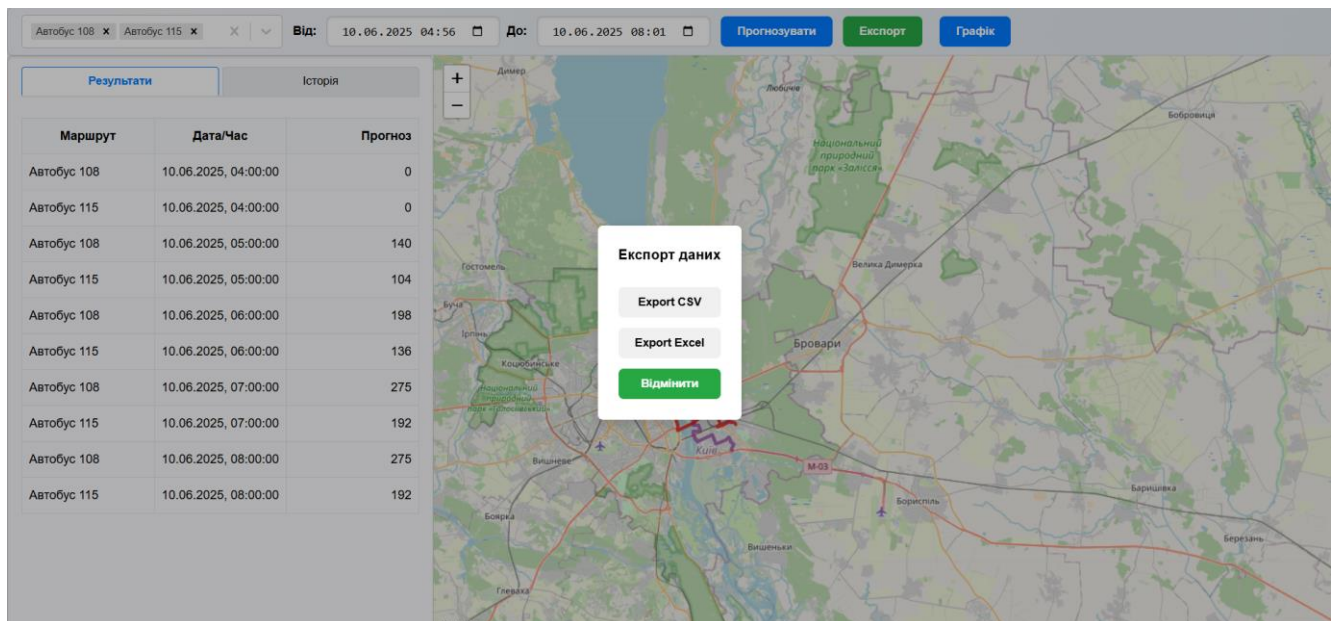


Рисунок 5.5 – Функціонал експорту результатів

Користувач отримує файл із маршрутами, датою, годиною та числом пасажирів – усе, що потрібно для подальшої роботи або презентації. На рисунку 5.6 показано приклад такого експорту у вигляді Excel-таблиці.

route_id	datetime	prediction
Автобус 108	2025-06-10T04:00:00	0
Автобус 115	2025-06-10T04:00:00	0
Автобус 108	2025-06-10T05:00:00	140
Автобус 115	2025-06-10T05:00:00	104
Автобус 108	2025-06-10T06:00:00	198
Автобус 115	2025-06-10T06:00:00	136
Автобус 108	2025-06-10T07:00:00	275
Автобус 115	2025-06-10T07:00:00	192
Автобус 108	2025-06-10T08:00:00	275
Автобус 115	2025-06-10T08:00:00	192

Рисунок 5.6 – Результат експорту в таблицю Excel

Діаграма використання системи наведена на кресленику IC12.250БАК.005Д4.

5.2 Аналіз ефективності системи

Після реалізації MVP-версії системи прогнозування попиту проведена її практична перевірка, зосереджена на оцінці швидкодії, зручності інтерфейсу, повноти функціоналу та наявних обмежень. Оскільки система повністю інтегрована, оцінка охоплює як роботу серверної частини, так і взаємодію через вебінтерфейс.

З точки зору швидкодії, система показала дуже хороший результат. Прогноз для одного маршруту на одну годину обробляється практично миттєво – відповідь від моделі надходить майже без затримки. Навіть при запиті прогнозу на 12 годин для кількох маршрутів одночасно відповідь надходить протягом кількох секунд. Така швидкість дозволяє використовувати систему в реальному часі без потреби в додатковому кешуванні або чергах запитів. Завдяки тому, що модель LightGBM завантажується лише один раз при запуску сервера, а підготовка ознак оптимізована, система ефективно працює навіть у батчевому режимі.

Інтерфейс виявився зручним і зрозумілим. Усі головні дії: вибір маршруту, часовий діапазон, перегляд результатів на мапі, графіку чи таблиці реалізовані інтуїтивно. У рамках тестування зручно працювати з історією запитів, а також з функцією експорту, яка дозволяє швидко отримати результати у вигляді таблиці. Навіть без авторизації або персоналізації інтерфейс виконує всі базові функції MVP і готовий до публічного використання або демонстрації.

Основним досягненням на цьому етапі стало налагодження повного циклу роботи: від підключення до зовнішніх API, формування ознак, прогнозування й обробки результату до візуалізації у фронтенді. Всі компоненти взаємодіють злагоджено, а сам застосунок демонструє стабільну роботу без збоїв чи втрат даних. Це підтверджує життєздатність архітектури та готовність системи до розширення.

Водночас були виявлені об'єктивні обмеження. Найпомітніше з них – обмеженість вихідних даних. Модель навчалась на штучно розподіленому попиті лише за травень, без розподілу по конкретних днях чи годинах. Це не дає змоги

					ІС12.250БАК.005 ПЗ	Арк.
						44
Зм.	Лист	№ докум.	Підпис	Дата		

зробити об'єктивну оцінку точності моделі в реальному місті, а також може вплинути на передбачуваність у нетипових сценаріях. Крім того, наразі система працює лише по Києву, оскільки джерела тривоги і зональні дані прив'язані до конкретного міста. Для масштабування на інші міста необхідно створити відповідну інфраструктуру зон, маршрутів і погодних запитів.

Ще одне обмеження стосується прогнозного горизонту, він обмежений 16 днями, оскільки саме на такий період доступний погодний прогноз із джерела Open-Meteo. Також варто зазначити, що система поки що не охоплює всі можливі маршрути Києва, частина з них не включена в базу через нестачу геоданих, дублікати у вихідних джерелах або нестачу даних про пасажиропотік.

5.3 Рекомендації щодо масштабування

На поточному етапі система функціонує в межах Києва і працює з обмеженим набором маршрутів та геозон. Однак архітектура побудована так, щоб без суттєвих змін масштабуватись на інші міста або більший обсяг даних. Для цього варто врахувати кілька аспектів.

Перший крок до масштабування – це розширення бази zones. Достатньо додати полігони інших міст (наприклад, Львова чи Харкова), після чого можна буде підключити ці зони до вже існуючої структури запитів. Аналогічно, у таблицю routes потрібно завантажити маршрути для кожного нового міста, з геометрією та базовими ознаками (тип транспорту, середній попит).

Оскільки модель прогнозування не містить жорсткої прив'язки до конкретних назв, а працює з route_id та zone_id, її можна перенавчити або скопіювати й повторно використати для нового регіону. Для підвищення точності бажано формувати окрему модель для кожного великого міста, або ж додати ознаку міста у ознаки й створити універсальну модель з багатовходовою логікою.

З точки зору збору даних, API Open-Meteo підтримує запити по будь-яких координатах, тож можна отримати погоду для нових зон без змін у коді. Щодо

тривог, то API alerts.in.ua також підтримує фільтрацію за location_uid, тож потрібно лише визначити відповідний ідентифікатор для нового міста.

На рівні вебінтерфейсу масштабування можливе шляхом додавання фільтра вибору міста. Це дозволить користувачу працювати з різними зонами, маршрутами та історією прогнозів у межах обраного населеного пункту.

Висновки до розділу 5

Описана реалізація інформаційної системи на практиці. Продемонстровано, як створена модель машинного навчання інтегрована у серверну частину, пов'язана з базою даних та винесена у зручний вебінтерфейс. Зібрані компоненти утворюють єдину систему, яка функціонує у реальному часі: приймає запити, формує ознаки, генерує прогноз, виводить результати та зберігає історію.

Графічний інтерфейс реалізований у вигляді сучасного вебзастосунку, який надає доступ до основних функцій: вибір маршрутів, прогнозування на заданий часовий інтервал, перегляд історії, виведення результатів на мапу та у вигляді графіків, а також експорт у форматі CSV/Excel. Усі ці дії виконуються без перезавантаження сторінки, що робить роботу з системою простою та зручною навіть для користувача без технічного досвіду.

Проведена перевірка ефективності підтвердила стабільність і швидкодію застосунку. Час відповіді моделі для одиничного запиту є практично миттєвим, а при великому діапазоні прогнозування затримка не перевищує кількох секунд. Інтерфейс не вимагає додаткового навчання, а всі функції організовані інтуїтивно.

Також виявлені деякі обмеження, які не знижують цінності реалізації, але потребують уваги в подальшій роботі. Зокрема, це обмеженість даних для навчання моделі, підтримка лише одного міста, і залежність прогнозного горизонту від можливостей зовнішнього API погоди.

6 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

6.1 Змістовна постановка задачі

У рамках реалізованої інформаційної системи ставиться задача прогнозування кількості пасажирів на конкретному маршруті громадського транспорту в обраний момент часу. Модель має на вхід отримувати набір параметрів (ознаковий вектор), що описує контекст прогнозу, і повертати передбачену кількість пасажирів, яка очікується на цьому маршруті у задану годину.

Вхідними даними для вирішення задачі є:

- маршрутні характеристики: назва маршруту, тип транспорту (автобус, тролейбус, трамвай, метро), середній добовий попит;
- погодні параметри: температура повітря, тип погоди, кількість опадів, індекс видимості;
- темпоральні фактори: день тижня, година доби;
- зовнішні події: активність повітряної тривоги на момент прогнозу;
- географічна зона: зона міста, до якої належить маршрут, визначається автоматично.

Потрібно побудувати залежність між вхідними змінними та числовим виходом і прогнозованою кількістю пасажирів у зазначений момент. Особливістю задачі є те, що вона має часову структуру, але не є класичною time-series моделлю: кожен прогноз оцінюється окремо, без залежності від попередніх значень. Крім того, дані для тренування моделі обмежені в часі маємо лише один тиждень, що змушує використовувати евристичні методи для генерації годинних патернів.

Постановка задачі вкладається у загальний клас задач регресійного прогнозування з числовим виходом, де необхідно передбачити значення цільової змінної на основі ознак середовища. Як зазначено, "у задачах прогнозу на табличних даних, де ознаки мають як числову, так і категоріальну природу, найбільш ефективними часто виявляються ансамблеві дерева прийняття рішень із градієнтним бустингом" [17].

					IC12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		47

На відміну від задач прогнозування завантаженості мережі або глобального моделювання пасажиропотоків, ця постановка зосереджена на локальному прогнозі – тобто на передбаченні попиту по маршруту в окрему годину, враховуючи локальні фактори. Це робить її релевантною до задач оперативної логістики міського транспорту.

6.2 Математична постановка задачі

Задачу прогнозування попиту на транспортні послуги можна формалізувати як задачу регресії з множиною незалежних змінних, де вихідне значення є кількістю пасажирів на маршруті в конкретну годину.

Нехай $X \in R^{n \times m}$ – матриця ознак, де n – кількість прикладів (рядків), а m – кількість ознак, що характеризують кожен приклад. Кожен рядок $x_i = \{x_{i1}, x_{i2}, \dots, x_{im}\}$ включає погодні, часові, географічні та маршрутні параметри. Цільова змінна $y \in R^n$ – це відповідне значення пасажиропотоку в годину на конкретному маршруті.

Тому ставиться задача знаходження такої функції $f : R^m \rightarrow R$, яка мінімізує помилку між передбаченими та фактичними значеннями y . Основною метою моделі є мінімізація похибки на тестовому наборі даних:

$$\hat{y} = f(x_i), i = 1, 2, \dots, n,$$

де $\hat{y}$ – передбачене значення;

x_i – вектор ознак для i -го прикладу.

Для оцінки точності передбачення використовуються класичні метрики регресії, серед яких: середня абсолютна похибка (MAE), корінь середньоквадратичної похибки (RMSE) та коефіцієнт детермінації R^2 .

Середня абсолютна похибка (MAE) – це середня абсолютна різниця між прогнозованими та фактичними значеннями (6.1).

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|, \quad (6.1)$$

де n – кількість прикладів у вибірці (кількість годин/рядків);

i – індекс конкретного прикладу;

y_i – фактичне значення кількості пасажирів у i -й годині;

$\hat{y}_i$ – передбачене моделлю значення.

Корінь середньоквадратичної похибки (RMSE) – це квадратний корінь із середньоквадратичної різниці між прогнозованими та спостережуваними результатами (6.2).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}, \quad (6.2)$$

де $(y_i - \hat{y}_i)^2$ – квадрат похибки моделі для i -го прикладу.

Коефіцієнт детермінації – це статистичний показник, який вимірює пропорцію варіації залежної змінної, що пояснюється моделлю прогнозу. Значення коефіцієнта детермінації коливається від 0 до 1. Чим ближче значення до 1, тим краще модель пояснює варіацію даних, тобто, тим сильніший зв'язок між змінними (6.3).

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad (6.3)$$

де $\bar{y}$ – середнє значення фактичних y_i .

Усі ці метрики дозволяють оцінити, наскільки точно модель передбачає фактичний попит на транспортні послуги. Наприклад, значення $MAE = 36.61$, отримане при тестуванні, означає, що в середньому прогноз моделі відрізняється

від реального значення на 36 осіб за годину, що вважається прийнятним рівнем похибки для задач такого типу.

Загальна задача, з урахуванням оптимізації моделі, зводиться до задачі мінімізації функції втрат L на тренувальному наборі (6.4).

$$\min_f \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i)), \quad (6.4)$$

де $\min_f$ – пошук такої функції f , яка мінімізує втрати;

$\sum_{i=1}^n L(y_i, f(x_i))$ – функція втрат (наприклад, MSE, MAE);

$f(x_i)$ – прогноз моделі для прикладу x_i .

6.3 Обґрунтування методу розв’язання

Задача прогнозування попиту на громадський транспорт за своїм типом є регресійною, але має ряд специфічних особливостей. По-перше, вона передбачає роботу з табличними даними з великою кількістю ознак, які можуть бути як числовими (температура, година), так і категоріальними (тип транспорту, тип погоди, день тижня). По-друге, дані включають випадкові збурення та вплив зовнішніх чинників, що ускладнює побудову класичних аналітичних моделей. Тому для розв’язання задачі доцільно використовувати підхід, що поєднує гнучкість, швидкодію та здатність працювати з різнорідними ознаками.

На практиці, для таких задач зазвичай розглядаються наступні класи моделей:

- лінійна регресія, яка добре працює на простих залежностях, але не здатна захопити складні нелінійності, що характерні для реальних транспортних потоків;
- random forest – ансамблевий метод, який часто показує гарну якість, але поступається в швидкодії та потребує більше ресурсів при розгортанні;
- метод опорних векторів – ефективний для невеликих наборів, але погано масштабується при великій кількості прикладів і ознак;

- нейронні мережі – потужний інструмент, особливо для time series, але вимагає великої кількості даних, складного налаштування та часу на навчання;
- градієнтний бустинг на деревах – метод, який поєднує високу точність, роботу з табличними даними та контрольовану складність.

Для задач прогнозування по маршрутних та погодних даних, де присутні пропуски, змішані типи ознак та обмежений обсяг навчальних прикладів, моделі на основі градієнтного бустингу демонструють стабільну перевагу. Серед популярних реалізацій таких алгоритмів (XGBoost, CatBoost, LightGBM), саме LightGBM виділяється найвищою швидкістю тренування та прогнозу, що важливо для інтеграції в онлайн-систему.

Вибір саме LightGBM як методу розв’язання задачі зумовлений такими факторами:

- якість – при тестуванні модель показала високе значення $R^2 = 0.96$, що свідчить про добру відповідність реальним даним;
- швидкодія – прогноз для одного або кількох маршрутів займає менше 5 секунд;
- масштабованість – модель не потребує GPU, не блокує основний потік серверу та може працювати в батчевому режимі;
- сумісність – можливість збереження у форматі .pkl і інтеграція у Flask-проект без додаткових залежностей.

Окрім технічних переваг, вибір LightGBM підкріплений тим, що саме цей метод рекомендовано до використання у схожих задачах транспортного моделювання, зокрема у роботах, де аналізується прогнозування пасажиропотоків у контексті змін погодних умов [18].

6.4 Опис методу розв’язання

Для розв’язання задачі прогнозування попиту на транспортні послуги в обрану годину використано модель градієнтного бустингу LightGBM, яка попередньо навчена на ознаках, сформованих на основі реальних і синтетично

адаптованих даних. Її інтеграція у систему відбувається через серверну частину, де модель використовується як окремий модуль для побудови прогнозу на запит користувача.

У момент запиту з вебінтерфейсу користувач передає маршрут, дату й годину. Система автоматично визначає відповідну геозону, до якої належить маршрут, а також витягує погодні умови й сигнали тривоги для вказаного моменту. Далі формується вектор ознак, що включає такі параметри, як температура, тип погоди, середній добовий попит, тип транспорту, день тижня, година доби, активність тривоги тощо. Для категоріальних змінних застосовується one-hot кодування, і формується таблиця з ознаками, яка відповідає структурі навчальної вибірки. У разі запиту на кілька маршрутів одночасно система формує окремі вектори ознак для кожного з них, що забезпечує паралельну обробку даних та скорочує час відповіді.

Після цього вектор ознак подається у LightGBM-модель, яка повертає числове значення – прогнозовану кількість пасажирів. Усі результати запиту зберігаються в базу даних, що дозволяє не лише повторно звертатися до історії, але й використовувати дані для подальшого аналізу. Результати моделі оцінювались за класичними регресійними метриками.

Після навчання отримані значення які вже зазначалися в попередніх пунктах та середнє MAE по крос-валідації (5 фолдів): 19.61

Ці показники свідчать про високу узгодженість передбачень моделі з реальною структурою вхідних даних, навіть незважаючи на обмежений набір навчальних прикладів.

Для перевірки роботи системи проведені приклади запитів на прогноз. Зокрема, для маршруту Автобус 104 на 03.06.2025 о 15:31 отримане значення попиту 176 осіб. При запиті одразу на два маршрути – Автобус 120 та Оболонсько-Теремківська лінія станція Либідська на 17:00 того ж дня – модель повернула відповідно 40 і 678 пасажирів відповідно. Ще один набір прогнозів наведений на рисунку 6.1, де відображений прогноз попиту для трьох маршрутів за два години поспіль.

					ІС12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		52

Маршрут	Дата/Час	Прогноз
Оболонсько-Теремківська лінія станція Тараса Шевченка	10.06.2025, 20:00:00	523
Платформа / вестибюль Райдужний	10.06.2025, 20:00:00	79
Тролейбус 14	10.06.2025, 20:00:00	295
Оболонсько-Теремківська лінія станція Тараса Шевченка	10.06.2025, 21:00:00	385
Платформа / вестибюль Райдужний	10.06.2025, 21:00:00	55
Тролейбус 14	10.06.2025, 21:00:00	218

Рисунок 6.1 – Приклади прогнозу попиту на маршрути в системі

Висновки до розділу 6

Змістовна постановка охоплює визначення основних впливових чинників: погодних умов, часових параметрів, типу маршруту та зовнішніх, що формують контекст для передбачення пасажиропотоку.

Математична модель подана у вигляді задачі регресії, де метою є мінімізація функції втрат на множині прикладів. Для оцінки точності прогнозу використані класичні метрики: MAE, RMSE та коефіцієнт детермінації R^2 , що дозволяють кількісно оцінити відхилення між передбаченням і фактичним значенням.

Розглянуті кілька поширених підходів до задач регресії: від лінійних моделей до ансамблевих методів. Основною причиною вибору саме LightGBM стала його здатність ефективно працювати з табличними даними, які містять як числові, так і категоріальні ознаки, а також висока швидкодія й простота інтеграції.

Реалізація методу в системі включає підготовку ознак із кількох джерел, формування вектору ознак, обчислення прогнозу через модель та збереження результатів. Запропонований підхід пройшов тестування на реальних прикладах і показав здатність точно відтворювати очікувані закономірності пасажиропотоку, навіть за умов обмеженого обсягу історичних даних.

7 ТЕСТУВАННЯ ТА ВАЛІДАЦІЯ СИСТЕМИ

7.1 Стратегія тестування

Тестування системи відбувалося поступово та поетапно, в міру розроблення окремих компонентів. Такий підхід дав змогу вчасно виявляти проблеми у функціонуванні окремих модулів ще до їх повної інтеграції.

Першочергово тестувалась взаємодія з базою даних – перевірялась правильність створення таблиць, цілісність зв'язків між ними, обробка вставки та читання даних. Особлива увага приділялась таблицям routes, zones, weather, alerts, requests, forecasts, адже саме вони забезпечують основні процеси: підготовку ознак, збереження запитів та прогнозів, і обробку змінних у реальному часі.

Другий етап тестування сфокусований на REST API. За допомогою Postman тестувались основні ендпоінти: /predict, /predict_batch_routes, /predict_range, /api/routes, /api/history. Для кожного з них перевірялась коректність формування запиту, структура відповіді, обробка помилок, поведінка при некоректних або порожніх параметрах. У випадках помилки очікувалась відповідна відповідь 400 або повідомлення з поясненням.

На останньому етапі приділена увага тестуванню графічного інтерфейсу. Тут перевірялась логіка роботи з формами, реакція на дії користувача (натискання кнопок без вибору параметрів), відображення результатів, побудова графіків та експорт. Окрім того, тестувались граничні ситуації: робота вночі, при тривозі, при відсутності погоди, для далеких дат.

Паралельно із цим проводилась перевірка поведінки моделі в окремих випадках, наприклад, у період комендантської години (з 23:00 до 5:00), де очікується нульовий попит. Такі правила були закладені ще при підготовці тренувальних даних, тому модель демонструє правильну реакцію на подібні вхідні умови.

Стратегія тестування не була суворо документованою, але охопила всі основні рівні: від роботи окремих функцій до наскрізної перевірки всієї системи. Це дозволило вчасно локалізувати проблеми, і до моменту підготовки MVP

					ІС12.250БАК.005 ПЗ	Арк.
						54
Зм.	Лист	№ докум.	Підпис	Дата		

забезпечити стабільну й передбачувану поведінку системи у штатних і нештатних ситуаціях.

7.2 Тестування прогнозів та граничних сценаріїв

Тестування охоплювало не лише типові запити, а й граничні ситуації, які користувач може створити навмисно або випадково. Перевірка таких сценаріїв має важливе значення, адже саме в них найчастіше проявляються помилки валідації, проблеми з обробкою виключень або неочікувані помилки системи.

У межах інтерфейсу тестувались різні типи запитів. Найбільш поширеним є прогноз для одного маршруту на одну годину, наприклад, для автобуса №104 на 03.06.2025 о 15:31. У результаті система правильно підтягує всі супутні ознаки і повертає прогноз – у цьому випадку 176 пасажирів. Наступний варіант – запит одразу на кілька маршрутів на ту саму годину, що дозволяє порівняти попит. Наприклад, для автобуса №120 і станції метро "Либідська" модель повертає 40 і 678 відповідно.

Окремо перевірялись батчеві запити на кілька маршрутів і багато годин. Такі запити використовують /predict_range і мають на меті побудову графіка попиту протягом доби або тижня. Система справляється з такими запитами, хоч і потребує більше часу для обробки, порівняно з одиничними.

Крім стандартних, тестувались граничні ситуації. Якщо користувач натискає кнопку «Прогноз» без вибраного маршруту – з'являється повідомлення про необхідність обрати хоча б один маршрут, зображено на рисунку 7.1.

Повідомлення з localhost:3000

Будь ласка, виберіть хоча б один маршрут



Рисунок 7.1 – Повідомлення про відсутній вибір маршруту

					ІС12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		55

При виборі дати прогнозу далі ніж на 16 днів – запит записується в БД, але прогноз не генерується, оскільки система використовує API погоди, обмежене в часі. Якщо викликати функцію побудови графіка без попереднього запиту – з’являється повідомлення про те, що немає даних для побудови, бачимо на рисунку 7.2.

Повідомлення з localhost:3000

Спочатку виконайте хоча б один запит для прогнозу.



Рисунок 7.2 – Повідомлення при відсутньому запиті на графік

При спробі експорту без наявних результатів – система повідомляє про відсутність даних як на рисунку 7.3.

Повідомлення з localhost:3000

Немає даних для експорту 🙄



Рисунок 7.3 – Повідомлення про відсутність даних для експорту

7.3 Аналіз результатів тестування

У процесі розроблення виявлені і виправлені низка технічних помилок, багато з яких виявлялись саме під час тестування граничних ситуацій або у фазі інтеграції.

Однією з проблем стало коректне визначення моменту завершення повітряної тривоги. API активних тривог (/active.json) не зберігає завершення подій

– тривога просто зникає. Для фіксації фактичного часу завершення система додатково звертається до архівного API (/alerts/month_ago.json), і якщо поточна тривога зберігається в БД як незавершена (end_time IS NULL), виконується оновлення. Така логіка забезпечила цілісність даних у таблиці alerts.

Інша проблема – відсутність архівної видимості в Open-Meteo. Через це ознака visibility_index не використовувалась ні при навчанні моделі, ні при побудові прогнозів. Модель перенавчена без цієї ознаки, і результати залишились прийнятними, проте надалі цю ознаку планується додати, якщо вдасться знайти альтернативне джерело.

Також були складнощі з геометрією маршрутів – розбіжності в назвах, використання префіксів (наприклад, A104 замість 104) або змішаних букв при використанні запитів через overpass API. Аналогічно зі станціями метро – деякі були записані з малої літери, або мали варіації в форматі. Після впровадження нормалізації назв ситуація покращилась.

Останньою технічно складною проблемою стало зависання при великих запитах, зокрема під час батчевого прогнозування. Причиною стало використання conn.begin() без належного управління транзакцією, що блокувало пул з'єднань. Помилка типу InvalidRequestError з'являлась при спробі повторно ініціалізувати транзакцію. Рішенням стало впровадження engine.begin() як контекстного менеджера, що самостійно керує commit/rollback, а також обмеження використання з'єднань на кожен запит.

На момент завершення розроблення критичних помилок у роботі системи не виявлено, усі відомі баги були локалізовані та виправлені.

Висновки до розділу 7

Здійснена перевірка працездатності розробленої інформаційної системи шляхом тестування окремих модулів та загальної інтеграції. Незважаючи на відсутність формальної стратегії тестування, підхід із поступовою перевіркою в процесі розроблення дозволив ефективно виявляти технічні помилки й логічні

					ІС12.250БАК.005 ПЗ	Арк.
						57
Зм.	Лист	№ докум.	Підпис	Дата		

помилки на ранніх етапах. Тестування охопило основні компоненти системи: базу даних, API-запити, інтерфейс користувача та поведінку моделі при крайових вхідних значеннях.

Перевірені типові сценарії прогнозування: як для одиночного маршруту, так і для кількох маршрутів на різні часові діапазони. Окрема увага приділена валідації граничних ситуацій: відсутність маршруту, вибір недоступного діапазону дат, запит на графік без прогнозу або спроба експорту без даних. Усі ці ситуації передбачені в логіці системи, що дозволяє уникати збоїв і забезпечує зручність взаємодії.

Аналіз виявлених проблем показав, що значна частина помилок була пов'язана з інтеграцією сторонніх API та роботою з нестандартними форматами даних. Успішно вирішене питання з визначенням моменту завершення тривоги, адаптована робота з назвами маршрутів, виправлені проблеми з блокуванням пулу з'єднань до бази даних. Також враховані обмеження зовнішніх джерел, зокрема неможливість отримати значення видимості з архівної погоди.

Проведене тестування підтвердило, що інформаційна система функціонує стабільно, логіка прогнозування відповідає заданим правилам, а інтерфейс забезпечує зрозумілу взаємодію з користувачем.

					ІС12.250БАК.005 ПЗ	Арк.
						58
Зм.	Лист	№ докум.	Підпис	Дата		

ВИСНОВКИ

В дипломному проєкті виконана повноцінна реалізація інформаційної системи, яка дозволяє прогнозувати попит на громадський транспорт у великому місті з урахуванням широкого спектра чинників. Створене рішення охоплює етапи від збору та підготовки даних до побудови моделі машинного навчання, її інтеграції в серверну частину та візуалізації результатів у вебінтерфейсі. Система враховує особливості міської інфраструктури, динаміку погодних умов, тимчасові обмеження, а також вплив надзвичайних подій, зокрема повітряних тривог.

У першій частині роботи проаналізовані типові проблеми міських транспортних систем, зокрема навантаження на маршрути, нерівномірність пасажиропотоків, а також відсутність гнучких інструментів для оперативного реагування. Огляд існуючих рішень показав, що більшість сучасних сервісів, як-от Moovit, EasyWay або Kyiv Digital, зосереджені переважно на навігації та відображенні графіків, але не на прогнозуванні. Зроблений висновок, що впровадження системи прогнозу попиту може суттєво покращити планування перевезень і підвищити стійкість транспортної системи.

У третьому розділі роботи сформульовані вимоги до майбутньої системи, зокрема щодо охоплення кількох джерел даних (погодні умови, маршрути, розклад, сигнали тривоги), гнучкості архітектури, можливості масштабування на інші міста та підтримки інтерактивної взаємодії з користувачем. Архітектура системи спроектована як трикомпонентна: клієнтська частина (React), сервер (Flask) і база даних (PostgreSQL з PostGIS). Такий підхід дозволив розмежувати логіку, спростити обслуговування та забезпечити масштабованість.

Під час етапу попередньої обробки даних враховані десятки ознак, включаючи день тижня, годину доби, тип транспорту, середній добовий попит, погодні умови, наявність тривоги. Дані об'єднувались у єдиний датафрейм, на якому була навчена модель LightGBM. Для її оцінки використані MAE, RMSE і R^2 , отримані результати показали високу точність прогнозу, навіть попри обмежений

обсяг історичних даних. Особливістю реалізації стало дотримання хронологічного порядку при кросвалідації, що забезпечило більш реалістичну оцінку якості.

Уся логіка обробки запитів на прогноз уніфікована. Користувач може обрати маршрут, вказати час або діапазон годин, і отримати прогнозований попит у числовому вигляді або у вигляді графіка. Дані автоматично зберігаються в базу, що дає змогу аналізувати історію звернень, повторно використовувати результати та будувати аналітику. У реалізованому інтерфейсі передбачений також експорт у CSV/Excel, відображення маршруту на мапі, побудову графіків попиту та реакцію на граничні випадки (помилки запитів, порожній вибір тощо).

Проведене тестування виявило як сильні сторони системи (швидкодія, гнучкість, точність), так і слабкі місця. Зокрема, основним обмеженням наразі залишається обсяг вхідних даних через те, що статистика пасажиропотоку отримана лише за один місяць, модель не має повного уявлення про сезонність, свята чи інші довготривалі ефекти. Водночас логіка збору даних, нормалізації назв маршрутів, обробки нестандартних повністю реалізована.

З технічного боку, система вже зараз працює як єдиний цілісний застосунок, який об'єднує серверну логіку, клієнтський інтерфейс і взаємодію з базою даних. Попри відсутність авторизації або багатокористувацького режиму, архітектура дає змогу легко додати ці функції в майбутньому. Так само й можливість розширення на інші міста – достатньо підключити нові зони та маршрути.

Отримані результати можна застосувати в міських департаментах транспорту, аналітичних панелях для управлінців, сервісах, що надають дані про міську мобільність, або навіть як частина системи реагування на надзвичайні події. Практична цінність полягає не лише у візуалізації, а у здатності прогнозувати зміну попиту з урахуванням часу доби, дня тижня, погодних умов і кризових ситуацій.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. McKinsey & Company. The future of mobility is at our doorstep, 2023. URL: <https://www.mckinsey.com/industries/public-and-social-sector/our-insights/the-future-of-mobility-is-at-our-doorstep> (дата звернення 10.06.2025).
2. S. Memon, M. A. Chaudhry, U. Hassan. Demand forecasting based machine learning algorithms on customer information: an applied approach, 2022. URL: https://www.researchgate.net/publication/358567204_Demand_forecasting_based_machine_learning_algorithms_on_customer_information_an_applied_approach (дата звернення 10.06.2025).
3. Bai, S., Kolter, J. Z., Koltun, V. An Empirical Evaluation of Generic Graph Attention Networks on Learn to Optimize Problems, 2022. URL: <https://openreview.net/forum?id=SJiHXGWAZ> (дата звернення 10.06.2025).
4. EasyWay. Інформаційний довідник міст (Kyiv). URL: <https://www.eway.in.ua/ua/cities/kyiv>
5. Офіційний сайт застосунку Moovit. URL: <https://moovitapp.com/>
6. Moovit Urban Mobility Analytics. URL: <https://moovit.com/maas-solutions/urban-mobility-analytics/>
7. World Cities Culture Forum. Kyiv Digital. URL: <https://worldcitiescultureforum.com/city-project/kyiv-digital/>
8. Kyiv Digital. Офіційний мобільний застосунок міста Київ. URL: <https://kyiv.digital/>
9. Bentley Systems. OpenPaths Help. URL: <https://docs.bentley.com/LiveContent/web/OpenPaths%20Help-v1/en/GUID-AD82CEF4-FE0B-4E45-BF4D-E91532EEE914.html> (дата звернення 10.06.2025).
10. Sommerville I. Software Engineering, видавництво Pearson Education, 2015. 960 с.
11. Flask documentation. URL: <https://flask.palletsprojects.com/en/stable/api/>
12. Microsoft Corporation. LightGBM Documentation. вебсайт. URL: <https://lightgbm.readthedocs.io/en/stable/>

					IC12.250БАК.005 ПЗ	Арк.
Зм.	Лист	№ докум.	Підпис	Дата		61

13. React documentation: вебсайт. URL: <https://react.dev/learn>
14. Open-Meteo documentation. URL: <https://open-meteo.com/en/docs>
15. Overpass API documentation. OpenStreetMap Wiki.
URL: https://wiki.openstreetmap.org/wiki/Overpass_API
16. Alerts.in.ua API documentation. URL: <https://devs.alerts.in.ua/>
17. Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., Liu T.-Y.
LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Advances in Neural
Information Processing Systems, 2017. 8 с.
18. Ding H., Guo Y., Li Y. Short Term Passenger Demand Forecasting under
Weather Impact Using Tree Based Ensemble Models. Journal of Advanced
Transportation, 2020. 12 с.