Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Інформаційних управляючих систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження моделей користувачів в ІТ-проєктах побудови пояснень в рекомендаційних системах

(тема)

Виконала:
здобувач 2 року навчання,
групи УПГІТм-23-1
Катерина ГАВРИШ
(власне ім'я, прізвище)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Управління проектами в галузі ІТ
(повна назва освітньої програми)
Керівник: проф. каф. ІУС Оксана ЧАЛА
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри ІУС (підпис) Костянтин ПЕТРОВ (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Інформаційних управляючих систем

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Управління проектами в галузі інформаційних технологій
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
" 21 " квітня 20 25 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Гавриш Катерині Андріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження моделей користувачів в ІТ-проєктах побудови пояснень в Рекомендаційних системах

затверджена наказом по університету від " 28 " березня 2025 р. № 235Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії " 05 " червня 2025 р.

3. Вихідні дані до роботи Тестові дані взаємодії користувачів із рекомендаційни журнали активності та лог-файли, зібрані в рамках експериментальної перевірки алгоримту; результати попередніх досліджень і присвячених моделюванню юзерів; синтетично згенеровані сесії взаємодії для імітації поведінкових патернів контрольованих умовах; методи переробки даних, серед яких сортування, групування за часовими інтервалами; критерії оцінки моделей користувача.

4. Перелік питань, що потрібно опрацювати у роботі Дослідження моделей користувачів, вибору даних і логів взаємодії, методів побудови пояснень із урахуванням часових паттернів та причинно-наслідкових зв'язків між діями, оцінки інтерпретованості пояснень, на довіру користувачів та ефективність взаємодії в ІТ-проєктах.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	01.03.25-18.03.25	Виконано
2	Огляд та аналіз літературних та наукових джерел	18.03.25-20.03.25	Виконано
3	Поставка задачі	10.03.25-20.03.25	Виконано
4	Теорія та удосконалення моделі	20.03.25-01.04.25	Виконано
5	Планування проєкту	01.04.25-01.05.25	Виконано
6	Реалізація підходу	01.05.25-20.05.25	Виконано
7	Оформлення пояснювальної записки	20.05.25-01.06.25	Виконано
8	Надання для перевірки на плагіат	01.06.25-02.06.25	Виконано
9	Попередній захист роботи	02.06.25-03.06.25	Виконано
10	Надання роботи на рецензування	03.06.25-04.06.25	Виконано
11	Захист роботи	05.06.25	Виконано

Дата видачі завдання 21 квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. каф. ІУС Оксана ЧАЛА
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 89 с., 12 рис., 13 табл., 3 дод., 14 джерел, 1 лістинг.

МОДЕЛІ КОРИСТУВАЧІВ ГЕНЕРАЦІЯ ПОЯСНЕНЬ, МОЖЛИВІСТЬ І НЕОБХІДНІСТЬ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ

Об’єктом дослідження кваліфікаційної роботи є процес побудови рекомендацій.

Предмет дослідження – моделі користувачів при побудові пояснень у рекомендаційних системах.

Метою роботи є дослідження підходу до формування пояснень у рекомендаційних системах, що базується на моделі користувача та враховує послідовність дій користувача і шаблони поведінки у часі.

Методи дослідження включають аналіз логів користувацької активності, виявлення послідовних і причинно-часових патернів, формалізацію поведінкових моделей, ранжування шаблонів за мірами можливості та необхідності, а також використання принципів пояснюваного штучного інтелекту для формування інтерпретованих пояснень.

Наукова новизна роботи полягає в удосконаленій багаторівневій моделі користувача, яка поєднує поведінкові патерни та залежності між діями користувача в часі, забезпечуючи підґрунтя для формування персоналізованих пояснень.

У результаті роботи розроблено програмний модуль, який аналізує логи взаємодії дій користувача на основі виявлених часових патернів і причинно-наслідкових залежностей, а також формує пояснення. Це дозволяє зробити рекомендації більш прозорими, адаптованими до поведінки користувача та сприяє зростанню довіри до системи.

ABSTRACT

Master’s thesis: 89 pages, 12 figures, 13 tables, 3 appendices, 14 sources, 1 listing

USER MODELS, EXPLANATION GENERATION, POSSIBILITY AND NECESSITY, RECOMMENDER SYSTEMS

The object of this qualification thesis is the process of building recommendations.

The subject of the research is user models used in generating explanations within recommender systems.

The aim of the work is to explore an approach to generating explanations in recommender systems based on a user model that takes into account the sequence of user actions and behavioral time patterns.

The research methods include analysis of user activity logs, identification of sequential and causal-temporal patterns, formalization of behavioral models, ranking of patterns based on measures of possibility and necessity, as well as the use of explainable artificial intelligence principles to generate interpretable explanations.

The scientific novelty lies in the development of an improved multi-level user model that incorporates behavioral patterns and temporal dependencies between user actions, forming the basis for generating personalized explanations.

As a result of the work, a software module was developed that analyzes interaction logs, builds a user model based on detected temporal patterns and causal relationships, and generates explanations for recommendations. This makes the recommendations more transparent, aligned with user behavior, and enhances trust in the system.

ЗМІСТ

С.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

CSV – Comma-Separated Values

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

JSON – JavaScript Object Notation

QA – Quality Assurance

RS – Recommender System

SQL – Structured Query Language

WBS – Work Breakdown Structure

XAI – Explainable Artificial Intelligence

XML – eXtensible Markup Language

ВСТУП

У сучасному цифровому середовищі рекомендаційні системи відіграють важливу роль у персоналізації контенту, впливаючи на рішення користувачів у сфері електронної комерції, медіаплатформ, освіти та соціальних мереж. Однак, попри свою поширеність, такі системи часто залишаються непрозорими для кінцевого користувача через складність алгоритмів, що лежать в їх основі. Це породжує недовіру кінцевого користувача, знижує ефективність взаємодії й може призводити до повного ігнорування рекомендацій.

Аби рекомендаційна система була справді корисною, недостатньо просто показати кінцевому користувачу певний варіант вибору, необхідно розуміти чому саме ця рекомендація з’явилася. Коли пояснення чому рекомендація з'явилася стає зрозумілим, існує більше шансів що користувач дослухається до таких рекомендацій та буде з ними взаємодіяти.

Щоб підвищити довіру до систем і забезпечити краще розуміння процесу формування рекомендацій, необхідно створювати пояснення, які ґрунтуються не лише на внутрішніх параметрах, а й на реальній поведінці користувача. Одним із перспективних напрямів побудови пояснень є моделювання користувачів шляхом створення багаторівневих структур, що враховують повторювані дії, їхню послідовність у часі, умови взаємодії та логічні залежності між подіями.

Метою роботи є дослідження моделей користувачів для формування пояснень у рекомендаційних системах, сформованої із врахуванням послідовності дій та часових шаблонів поведінки.

Результати можуть бути використані для вдосконалення рекомендаційних систем, підвищення рівня довіри користувачів, а також для покращення роботи фахівців в сфері інформаційних технологій.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз процесу побудови рекомендацій

Рекомендаційні системи (RS) є інструментом сучасних інформаційних технологій, які використовуються для персоналізованого надання рекомендацій користувачам. Вони базуються на різних алгоритмах та методах аналізу даних, дозволяючи пропонувати товари, послуги або інформацію на основі поведінки користувача та його уподобань.

Рекомендаційні системи використовують методи машинного навчання, штучного інтелекту та обробки великих масивів даних для прогнозування інтересів користувачів. Основною метою таких систем є покращення користувацького досвіду шляхом надання релевантних рекомендацій. Такі системи можуть працювати в різних сферах: електронна комерція, потокові сервіси, соціальні мережі, освітні платформи та багато інших.

Рекомендації можуть бути персоналізованими або загальними. Персоналізовані рекомендації формуються на основі індивідуальних даних користувача, таких як історія переглядів або покупки, тоді як загальні рекомендації базуються на популярності товарів або послуг серед усіх користувачів.

Одним із найпоширеніших методів є колаборативна фільтрація, яка аналізує дії різних користувачів і пропонує рекомендації на основі схожих патернів поведінки. Наприклад, якщо два користувачі мають схожі вподобання щодо фільмів, система може рекомендувати одному з них фільми, які сподобалися іншому.

Контентно-орієнтовані методи аналізують характеристики об'єктів (наприклад, категорію товару, опис або ключові слова) і пропонують користувачам схожі товари. Такі методи особливо ефективні в контексті рекомендацій книг, музики або фільмів.

Гібридні системи комбінують різні підходи, наприклад, поєднують

колаборативну фільтрацію з контентним аналізом для підвищення точності рекомендацій. Відомим прикладом такого підходу є рекомендаційна система Netflix, яка використовує комбінацію різних алгоритмів для прогнозування інтересів користувачів.

Однією з основних проблем у розвитку рекомендаційних систем є питання конфіденційності даних[1]. Оскільки такі системи збирають велику кількість інформації про користувачів, важливо забезпечити відповідний рівень безпеки та захисту персональних даних.

Ще одним викликом є проблема холодного старту, коли новим користувачам або новим продуктам складно отримати релевантні рекомендації через відсутність достатніх даних. Для вирішення цієї проблеми використовуються алгоритми, що комбінують кілька джерел даних.

З розвитком штучного інтелекту та глибокого навчання очікується покращення якості рекомендаційних систем. Використання нейронних мереж дозволяє створювати більш точні та адаптивні рекомендації, що враховують контекст користувача та його індивідуальні потреби.

Рекомендаційні системи широко застосовуються в сучасних цифрових сервісах для персоналізації контенту та покращення користувацького досвіду.

Розглянемо, як вони реалізовані в популярних платформах.

Таблиця 1.1 – Порівняльна таблиця методів рекомендацій, які використовують популярні платформи.

Платформа	Контентне моделювання	Колаборативна фільтрація	Соціальне моделювання	Демографічне моделювання	Глибоке навчання	Приклади функцій / Особливості
1	2	3	4	5	6	7
Netflix	Метадані (жанр, актори)	Історія переглядів	-	-	+	Рекомендації, пошук схожих

Кінець таблиці 1.1

1	2	3	4	5	6	7
Spotify	Аудіоаналіз треків	Вподобання інших користувачів	-	-	+	Discover Weekly, плейлисти за смаком
Amazon	Опис товарів, категорії	Історія покупок	Рейтинги, відгуки	Дані профілю	+	Рекомендації товарів, персоналізація головної сторінки
YouTube	Метадані відео	Історія переглядів	-	-	+	Головна сторінка, «Up next», тренди
Facebook	-	-	Взаємодія в мережі	-	+	Персоналізована стрічка, реклама
LinkedIn	Контент профілю	Лайки, коментарі	Активність у мережі	Професійні дані	+	Рекомендації постів, контактів, вакансій

Netflix використовує гібридний підхід, поєднуючи контентно-орієнтоване та колаборативне моделювання. Аналізуються метадані фільмів (жанр, актори), оцінки та історія переглядів користувачів[2]. Глибоке навчання допомагає передбачати, які фільми можуть зацікавити на основі минулої активності.

Spotify застосовує гібридні моделі для формування персоналізованих рекомендацій. Контентно-орієнтований аналіз аудіохарактеристик поєднується з колаборативною фільтрацією. Функція "Discover Weekly" створює плейлист на основі історії прослуховувань і схожих користувачів.

Amazon використовує комбінацію колаборативної фільтрації, контентного й демографічного моделювання. На основі історії покупок і переглядів формуються персоналізовані рекомендації. Додатково використовуються відгуки та рейтинги інших клієнтів.

YouTube реалізує систему рекомендацій на базі машинного навчання. Аналізуються перегляди, вподобання та час взаємодії. Завдяки глибокому навчанню і колаборативній фільтрації система формує релевантні рекомендації навіть для нових користувачів.

Facebook персоналізує стрічку новин, рекламу та контент за допомогою соціального моделювання. Алгоритми враховують вподобання, коментарі, час взаємодії з контентом, щоб визначити найбільш релевантні публікації.

LinkedIn використовує гібридний підхід: контентно-орієнтоване, колаборативне та соціальне моделювання. На основі активності та професійного профілю формуються рекомендації постів, контактів і вакансій.

Приклади вище демонструють, наскільки різними можуть бути підходи до реалізації рекомендаційних технологій. Кожна платформа формує власну стратегію з урахуванням потреб користувачів і типу контенту, що дозволяє підвищити точність рекомендацій. Водночас сучасні системи мають спільні виклики: складність пояснення рекомендацій, проблеми з новими користувачами та потреба балансу між персоналізацією і різноманіттям. Це підкреслює актуальність подальших досліджень і вдосконалення методів.

Узагальнена блок-схема роботи рекомендаційної системи зображена на рисунку 1.

Рисунок 1.1 – Узагальнена блок-схема роботи РС

Система збирає дані про користувачів і про характеристики об’єктів. На їх основі формується профіль користувача. Далі відбувається вибір алгоритмів у модулі «Matching/Рекомендації»: це можуть бути контентні, колаборативні або гібридні методи. Отримані кандидати проходять модуль сортування після чого формується фінальний список рекомендацій. В

результаті ми отримуємо знання про реакцію реакцію кінцевого користувача, що допомагає генерувати пояснення для розробників і відповідаємо на питання «чому саме цей контент».

1.2 Аналіз методів побудови рекомендацій

У сучасних цифрових системах рекомендації відіграють ключову роль у формуванні користувацького досвіду. Від того, як саме формуються рекомендації, залежить не лише релевантність запропонованого контенту, а й рівень довіри до платформи, залученість користувачів та ефективність бізнес-процесів. Розуміння методів побудови рекомендацій дає змогу краще оцінювати їхні можливості та обмеження, впроваджувати відповідні технології у прикладні рішення, а також коригувати або комбінувати моделі під конкретні завдання[3]. Далі розглянемо найбільш популярні методи побудови рекомендацій, зробимо аналіз приницпів їх роботи, а також визначимо сильні та слабкі місця. Це дозволить отримати теоретичні знання для розробки алгоритму виявлення й пояснення послідовностей подій, які мають високу ймовірність і необхідність, на основі часових шаблонів у даних користувача.

Таблиця 1.2 – Порівняльна таблиця методів побудови рекомендацій

Метод	Переваги	Недоліки
1	2	3
Контентно-орієнтований	Не потребує даних інших користувачів	Надмірна схожість, відсутність новизни, ігнорування контексту дій та змін шаблонів поведінки.

Кінець таблиці 1.2

1	2	3
Колаборативний	Виявляє приховані вподобання	Проблема нового користувача, не враховується порядок дій, потрібен великий обсяг даних.
Гібридний	Найвища точність і персоналізація	Ч аналіз дій користувачів, ігнорування контексту поведінки, непрозора логіка пояснень.

Контентно-орієнтована модель формує рекомендації на основі схожості об’єктів, з якими користувач взаємодіяв раніше. Наприклад, якщо користувач переглядав фільми жанру «драма», система пропонуватиме інші драми. Профіль користувача створюється шляхом аналізу властивостей цих об’єктів, жанру, ключових слів, характеристик товарів тощо, після чого відбувається пошук подібного контенту.

Основною перевагою такого підходу є те, що він не потребує даних про інших користувачів, а отже добре підходить для нових або невеликих систем. До того ж, його легко реалізувати. Проте він має низку обмежень. Через надмірну спеціалізацію рекомендації можуть ставати занадто схожими між собою. Також модель не пропонує нові чи несподівані варіанти, обмежуючись знайомим контентом. Крім того, не враховується контекст взаємодії (наприклад, час або ситуація) та динаміка інтересів користувача, що може зменшити точність рекомендацій у реальному середовищі.

Колаборативна фільтрація базується на припущенні, що користувачам з подібною поведінкою можуть подобатись однакові об’єкти. Система аналізує взаємодії між користувачами або об’єктами: якщо користувач А поставив

високу оцінку фільму, який також сподобався користувачу B, то A може отримати у рекомендаціях інші вподобані B фільми.

Серед переваг здатність виявляти приховані інтереси, навіть без розуміння змісту об’єктів. Метод працює незалежно від типу контенту, що робить його універсальним. Проте він має свої труднощі. По-перше, виникає проблема «холодного старту», коли система не має достатньо даних про нових користувачів чи об’єкти. По-друге, метод не враховує порядок дій або час, у який вони відбулися, що може призводити до менш релевантних рекомендацій. І, нарешті, для ефективної роботи потрібна велика кількість історичних даних, що не завжди можливо забезпечити.

Для подолання недоліків окремих методів застосовуються гібридні системи, які комбінують кілька стратегій. Завдяки такому поєднанню підвищується точність рекомендацій і зменшується ймовірність помилок. У межах гібридного підходу також використовуються варіації:

– демографічний підхід враховує стать, вік, регіон, рівень освіти тощо;

– соціальний підхід орієнтується на взаємозв’язки між користувачами (наприклад, підписки або друзі;

– знання-орієнтований підхід використовує експертні правила, структури знань або онтології.

Гібридні системи демонструють високу точність і гнучкість. Однак вони складні у впровадженні, адже потребують ресурсів, мають складну архітектуру, часто непрозорі з точки зору пояснюваності. У багатьох випадках вони лише частково аналізують дії користувача, не розуміючи логіки, яка стоїть за поведінкою.

На відміну від класичних методів, які здебільшого покладаються на статичні профілі або загальну схожість між об’єктами, використання моделі користувача відкриває нові можливості для підвищення якості рекомендацій. Зокрема, вона дозволяє враховувати реальні поведінкові патерни, контекст взаємодії та динамічні зміни у вподобаннях. Це особливо важливо для вдосконалення гібридних методів, які поєднують кілька підходів, де модель

користувача виступає в них як додаткове джерело персоналізованої інформації, що дозволяє зробити рекомендації більш релевантними, адаптивними та корисними.

1.3 Аналіз моделей користувачів

Пояснення в рекомендаційних системах – це механізм, який надає користувачу обґрунтування причин, з яких система пропонує ті чи інші товари чи інформацію. Їхня наявність підвищує прозорість системи та рівень довіри. У роботах з напрямку Explainable Artificial Intelligence (XAI), особливо наголошується на ролі моделей користувача у формуванні якісних пояснень.

У дослідженнях XAI [4] відзначено, що якість пояснень та правильність користувацької моделі (наприклад, точність, повнота уявлення користувачем про роботу системи) можна оцінювати як критерії «добрості» пояснень. Таким чином, побудова пояснень у рекомендаційній системі спрямована на те, щоб допомогти користувачу зрозуміти логіку рекомендацій та дати йому можливість скоригувати або підтвердити свої уподобання. Пояснення можуть бути представлені у різних форматах, текстових повідомленнях, графічних візуалізаціях або інтерактивних діалогах, але їхня основна роль полягає в тому, щоб підвищити прозорість і сприйняття рекомендацій користувачем.

У такому підході модель є когнітивним уявленням користувача про те, як працює система. Саме ця модель визначає, наскільки пояснення буде зрозумілим і прийнятим. Якщо пояснення узгоджуються з очікуваннями користувача, зростає довіра до системи. В іншому випадку, виникає розгубленість або відмова від використання.

Діаграма показує (рис. 2), що інтелектуальна система формує пояснення, які отримує користувач. Тобто зв’язки формуються наступним чином. Спочатку формується пояснення, далі створюється або змінюється модель

користувача, після чого виникає або довіра, або недовіра. На основі цих пунктів користувач оцінює систему за певними критеріями якості, наприклад чи є вони зрозумілими, точними та корисними.

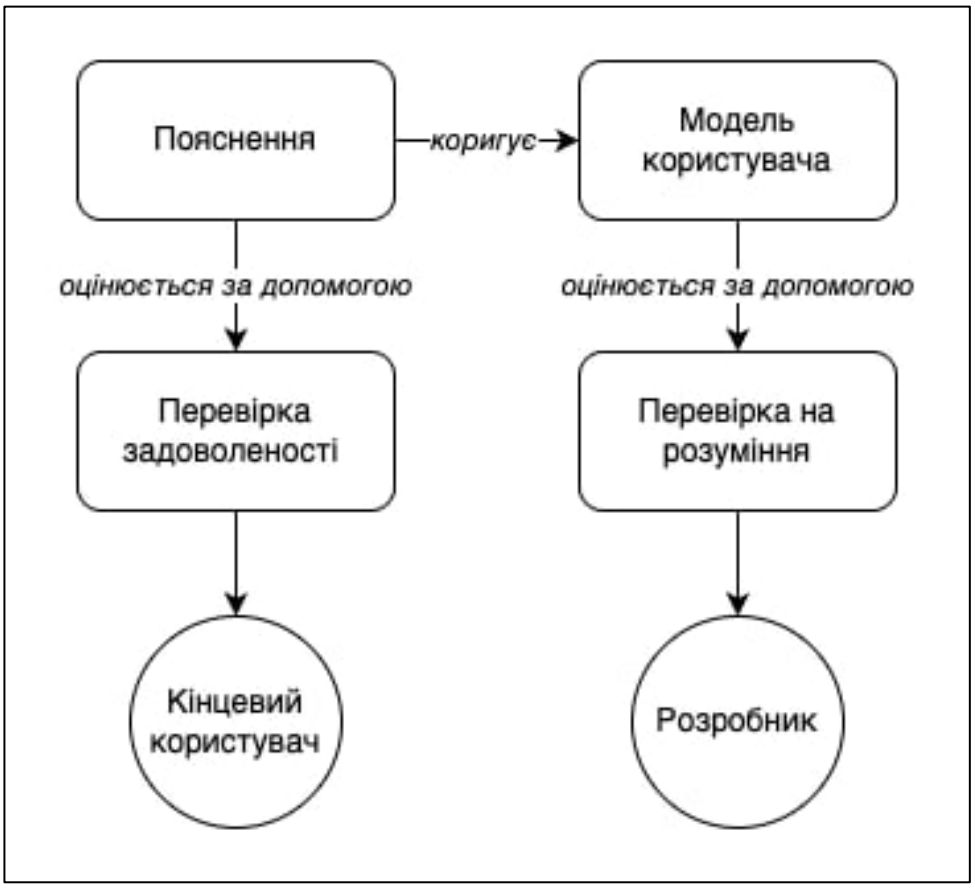

Рисунок 1.1 – Формування пояснень

На представленій діаграмі зображено логіку взаємозв’язку між поясненням, моделлю користувача та двома типами оцінювання, що дозволяють перевірити якість взаємодії користувача з системою. Діаграма ілюструє важливі елементи архітектури пояснюваної системи штучного інтелекту (XAI), орієнтованої на прозорість, інтерпретованість та довіру з боку користувача.

Центральним елементом виступає блок "Пояснення", який є основним артефактом, що створюється XAI-системою для супроводу результатів (наприклад, рекомендацій). Таке пояснення виконує не лише інформаційну

функцію, а й активно формує або коригує модель користувача. У контексті XAI пояснення має бути не лише зрозумілим, але й узгодженим з реальною логікою прийняття рішень системою, щоб уникнути хибних уявлень або надмірного спрощення.

Модель користувача у цій схемі – це сукупність знань, інтуїтивних уявлень та припущень про принципи роботи системи, що формується на основі спостереження за поведінкою системи та інтерпретації її пояснень. Чим точніше ця модель відображає дійсну логіку системи, тим вища ефективність взаємодії, тим обґрунтованіші рішення приймає користувач, і тим вищий рівень довіри до системи. На діаграмі подано два окремі шляхи оцінки:

– оцінка пояснення відбувається через тест задоволеності, що орієнтований на суб’єктивне сприйняття пояснення кінцевим користувачем. Такий тест дозволяє виявити, чи є пояснення зрозумілим, корисним, доречним та переконливим. Це особливо важливо для XAI-систем, метою яких є не лише надати відповідь, а зробити її прозорою, адаптованою до рівня користувача і контексту завдання. я система, що надає рекомендацію та пояснення;

– оцінка моделі відбувається через тест на розуміння, який передбачає перевірку того, наскільки користувач правильно сформував уявлення про логіку системи після ознайомлення з поясненнями. Такий тест зазвичай ініціюється розробником або дослідником і дозволяє виявити прогалини, спотворення або непорозуміння в сприйнятті. Це критично важливо для оцінки ефективності пояснень не лише як елементів інтерфейсу, а як інструментів навчання користувача.

Таким чином, пояснення виступає одночасно об’єктом оцінки (через тест задоволеності) та інструментом, що формує модель (яка перевіряється через тест на розуміння). Обидва канали зворотного зв’язку є невід’ємною частиною життєвого циклу XAI-систем, забезпечуючи їх адаптивність, надійність та відповідність очікуванням користувача.

Діаграма також демонструє причинно-наслідкові зв’язки між компонентами: пояснення впливає на модель, а якість обох оцінюється

незалежно. Така структура підходить для ітеративного вдосконалення системи, а результати тестів дозволяють розробникам удосконалювати як зміст пояснень, так і логіку системи, орієнтуючись на реальні потреби та сприйняття користувачів.

Таким чином, представлена модель є прикладом практичної реалізації принципів XAI – не лише як технічного інструменту, але як засобу формування зрозумілої та довіреної взаємодії людини з системою штучного інтелекту.

Робота XAI-системи передбачає, що користувач отримує не лише саму рекомендацію або рішення, але й обґрунтування, чому саме таке рішення було прийнято. Такі пояснення допомагають людині сформувати свою «модель» роботи системи: користувач розуміє, як і чому ШІ діє в тій чи іншій ситуації, коли можна довіряти рекомендаціям, а коли варто бути обережним. Ці підходи перевіряються в експериментах із залученням реальних користувачів, де оцінюють, чи стають пояснення зрозумілішими, чи дійсно вони покращують довіру та допомагають ефективніше користуватися системою. У XAI застосовуються різні підходи, це й адаптація глибинних моделей, і побудова пояснюваних моделей, і методи, які дозволяють пояснювати навіть «чорні скриньки» через аналіз їх поведінки. В підсумку, XAI – це комплексне рішення, яке не лише дає відповідь, а й дозволяє користувачу зрозуміти, наскільки ця відповідь надійна.

Пояснення, засновані на поведінці користувача, відіграють ключову роль у підвищенні ефективності рекомендаційних систем. Вони дають змогу зробити аргументацію персоналізованою, підвищити довіру до системи та полегшити сприйняття її рішень. У своїх дослідженнях Ганнінг виокремлює два критично важливі аспекти пояснень: по-перше, механізми побудови пояснення, і побудову моделей користувачів – тобто формування уявлення системи про потреби, наміри та контекст конкретного користувача. Саме модель користувача забезпечує підбір найбільш релевантних аргументів у поясненні, що дозволяє не лише покращити якість комунікації системи з

людиною, а й підвищити результативність самих рекомендацій.

Напрям побудови пояснень в рекомендаційних системах тісно пов’язаний з побудовою моделей користувачів, адже зрозумілі пояснення можливі лише тоді, коли система має якісну модель користувача. У сучасних системах використовуються різні моделі, від простих до багаторівневих.

Таким чином, підходи до побудови пояснень в XAI системах неможливо розглядати у відриві від моделювання користувачів. Обидва аспекти мають бути узгодженими для формування пояснень, які користувач не лише зрозуміє, а й сприйме як корисні й достовірні.

Розглянуті підходи до пояснень у XAI-системах показують, що якість пояснення напряму залежить від точності моделі користувача. Класичні, дозволяють формувати профілі, але часто не враховують послідовність дій користувача. Порівняння моделей користувачів наведено в таблиці 1.4.

Таблиця 1.4 – Порівняння моделей користувачів

Модель	Побудова моделі користувача	Недоліки	Використання пояснень для удосконалення рекомендацій
1	2	3	4
Колаборативне фільтрування.	Кластеризація за схожими вподобаннями інших користувачів.	Враховує лише дії інших користувачів, але не поведінку юзера, не працює для нових, не враховує послідовність дій у часі.	Врахування повторюваних патернів користувача після старту для пояснення без зовнішніх даних.

Кінець таблиці 1.4

1	2	3	4
Контентно-орієнтована	Профіль створюється на основі характеристик вподобаних об'єктів.	Не враховує поведінку користувача взагалі – рекомендації будуються лише на основі схожості контенту	Послідовність дій може вказувати на інтерес до певного типу контенту, що дозволяє формувати динамічні пояснення.
Гібридний, може поєднують декілька методів, наприклад: демографічний, соційальний або знання орієнтовний.	Побудова профілю через ручні запити, правила або фільтри.	Хоча є спроби врахувати різні фактори, поведінка враховується фрагментарно або неструктуровано, адаптація до змін відбувається повільно.	Зв’язки дозволяють адаптувати пояснення до змін у поведінці, часові шаблони пояснюють логіку рекомендацій. Важливо врахувати поведінку користувача, аби адаптувати алгоритми системи.

Колаборативна модель ґрунтується на аналізі вподобань схожих

користувачів. Вона ефективна для виявлення нових об’єктів, які користувач міг би і не знайти самостійно. Проте вона не враховує індивідуальну поведінку, не працює при «холодному старті» та ігнорує послідовність дій. Її вдосконалення передбачає використання повторюваних патернів конкретного користувача, що дозволяє пояснювати рекомендації навіть за обмежених даних.

Контентно-орієнтована модель формує профіль на основі характеристик вподобаних об'єктів. Вона добре працює без інформації про інших користувачів, але не враховує зміни в поведінці та контекст. Вдосконалена модель враховує послідовність взаємодій, що дозволяє формувати пояснення з урахуванням динамічних інтересів.

Гібридна модель поєднує кілька підходів – контентно-орієнтований, колаборативний, демографічний тощо. Вона більш гнучка, однак часто включає поведінкові аспекти лише частково й недостатньо структуровано. Удосконалення гібридної моделі може полягати в інтеграції часових шаблонів і причинно-наслідкових зв’язків, що робить пояснення точнішими та адаптивними. Такий підхід дозволяє враховувати як регулярність дій користувача, так і контекст, у якому вони відбуваються. Це відкриває можливість формувати пояснення, які не лише обґрунтовують рекомендацію, а й відображають її логіку в динаміці взаємодії.

Таким чином, традиційні моделі рекомендацій, такі як колаборативна фільтрація, контентно-орієнтована чи гібридна, мають деякі обмеження, що пов’язані з відсутністю повного врахування поведінки користувача в часі, контексту та причинно-логічних зв’язків між діями. Їх удосконалення можливе шляхом інтеграції повторюваних шаблонів дій, а також залежностей, що відображають справжню логіку користувацької взаємодії із системою. Врахування таких аспектів не лише підвищує точність рекомендацій, але й дає змогу будувати прозорі пояснення, що є ключовою вимогою сучасних систем, орієнтованих на довіру та зрозумілість.

1.4 Постановка задачі дослідження

Актуальність дослідження полягає в необхідності підвищення прозорості та довіри кінцевих користувачів до рекомендацій, які формуються сучасними рекомендаційними системами. Через відсутність пояснень або їхню складність, користувачі можуть сприймати результати як випадкові або нав’язані. Це часто призводить до втрати довіри, ігнорування рекомендацій, а в окремих випадках, до повної відмови від використання сервісу.

Окрім впливу на досвід користувача, непрозорість також ускладнює процеси тестування, налагодження й адаптації систем з боку розробників. Без розуміння того, на підставі чого система приймає ті чи інші рішення, неможливо виявити помилки, усунути упередження або забезпечити відповідність очікуванням користувачів. Саме тому в сучасних підходах усе більше уваги приділяється створенню моделей та формуванню пояснень, які базуються не лише на основі моделі, а й на контексті дій конкретного користувача.

У цьому контексті актуальною стає побудова багаторівневої моделі користувача, яка здатна відображати не лише загальну інформацію, а й послідовність, динаміку та причинно-залежні зв’язки між діями користувача. Такі моделі дозволяють не лише покращити якість рекомендацій, а й створити обґрунтовані пояснення, зрозумілі як користувачам, так і розробникам.

Об’єктом дослідження є процес побудови рекомендацій.

Предметом дослідження виступають моделі користувачів, які застосовуються для побудови пояснень у рекомендаційних системах. Такі моделі описують поведінку та послідовність дій користувача та дозволяють враховувати не лише його попередній досвід, а й виявляти закономірності між діями кінцевого користувача.

Метою роботи є дослідження моделей користувачів для формування пояснень у рекомендаційних системах, сформованої із врахуванням

послідовності дій та часових шаблонів поведінки.

Для досягнення поставленої мети необхідно виконати наступні завдання дослідження:

- провести аналіз процесу побудови рекомендацій;
- розглянути методи побудови рекомендацій;
- проаналізувати моделі користувачів при побудові пояснень в інтелектуальних системах;
- сформувати модель користувача при побудові пояснень;
- розробити багаторівневу модель користувача;
- сформувати план реалізації проєкту з врахуванням кращих практик менеджменту проєктів;
- розробити модуль побудови моделі користувача;
- провести експериментальну перевірку моделі користувача в тестовому середовищі.

Наукова новизна полягає в використанні багаторівневої моделі користувача, яка містить рівень шаблонів та залежностей, та враховує часову динаміку дій між подіями, що дозволяє формувати персоналізовані пояснення.

Практичним результатом роботи має бути програмний модуль, здатний на основі логів сформувати модель поведінки користувача, яка є основою для пояснень до рекомендацій. Пояснення до рекомендацій базуються звязках між діями та мають причинно-наслідкові залежності. Іншим важливим аспектом практичної частини є врахування часових проміжків між діями користувача в системі для визначення їх послідовності.

2 РОЗРОБКА МОДЕЛІ КОРИСТУВАЧА

2.1 Формування моделі користувача при побудові пояснень

У сучасних інформаційних системах ключову роль відіграє те, як користувачі сприймають логіку їхньої роботи. Замість технічно точного уявлення про внутрішні процеси системи, людина оперує приблизною уявною картиною, що формується на основі її досвіду, спостережень і інтуїції. Цю картину називають моделлю користувача.

Тобто модель – це структура, яка відображає, як користувач очікує, що система функціонує, реагує на дії, що означають елементи інтерфейсу. Такі моделі робочим способом орієнтації у цифровому середовищі. Вони дозволяють користувачеві прогнозувати наслідки власних дій та приймати обґрунтовані рішення[5,6,7].

Властивості моделі зазначені нижче (табл. 2.1)

Таблиця 2.1 – Властивості моделі користувача

Властивість	Опис
1	2
Спрощеність	Не містить усієї технічної деталізації; вона узагальнює складні процеси до зрозумілих користувачу образів.
Динамічність	Постійно оновлюється під впливом нового досвіду: коли результат дії системи не відповідає очікуванням, користувач коригує уявлення.
Прогностичність	Допомагає передбачити, як система відреагує на конкретний вхід (клік, запит тощо), і, відповідно, спланувати власні дії.
Контекстна прив’язка	Структурується залежно від конкретного контексту – інтерфейс, попередній досвід, мета користувача; одна й та ж система може сприйматися по-різному

Кінець таблиці 2.1

1	2
Пояснюваність	За наявності прозорих пояснень користувач краще розуміє, чому система поводиться певним чином, і це корелює з його власною моделлю.
Емоційний аспект	Довіра або невдоволення системою формуються через відповідність (або несумісність) фактичної роботи алгоритму з уявленнями користувача.
Адаптивність	У разі введення нового інтерфейсу чи додаткових функцій система повинна надавати пояснення, що відповідають чинній моделі користувача.
Узгодженість	Коли пояснення співпадають з очікуваннями, модель укріплюється, при розбіжностях вона переглядається й коригується.

У системах, які включають механізми пояснення, таких як системи штучного інтелекту або рекомендаційні системи, саме моделі визначають, чи буде пояснення зрозумілим, прийнятним і корисним. Якщо пояснення не узгоджується з очікуваннями користувача, це знижує довіру, викликає розгубленість або повну відмову від взаємодії.

У проєктуванні пояснювальних систем важливо не просто надати інформацію, а зробити це так, щоб вона була співзвучна вже наявній у користувача моделі. Це особливо актуально в контексті рекомендаційних систем, де користувач не завжди розуміє, чому саме йому запропоновано певний результат. Відповідно, пояснення повинно опиратися не на внутрішню логіку алгоритму, а на мову та уявлення самого користувача.

Моделі формуються через спостереження за поведінкою системи у системах з високим рівнем складності, таких як XAI. Таким чином, формування таких моделей можливе навіть тоді, коли система є непрозорою – через елементи зворотного зв’язку, тренування, персоналізовану взаємодію

або сценарії, що повторюються.

На рисукнку 2.1 відображено, постійний розвиток моделі користувачів, яку ми формуємо, підкреслюючи, як досвід користувачів формує майбутні очікування, прогнози та рішення.

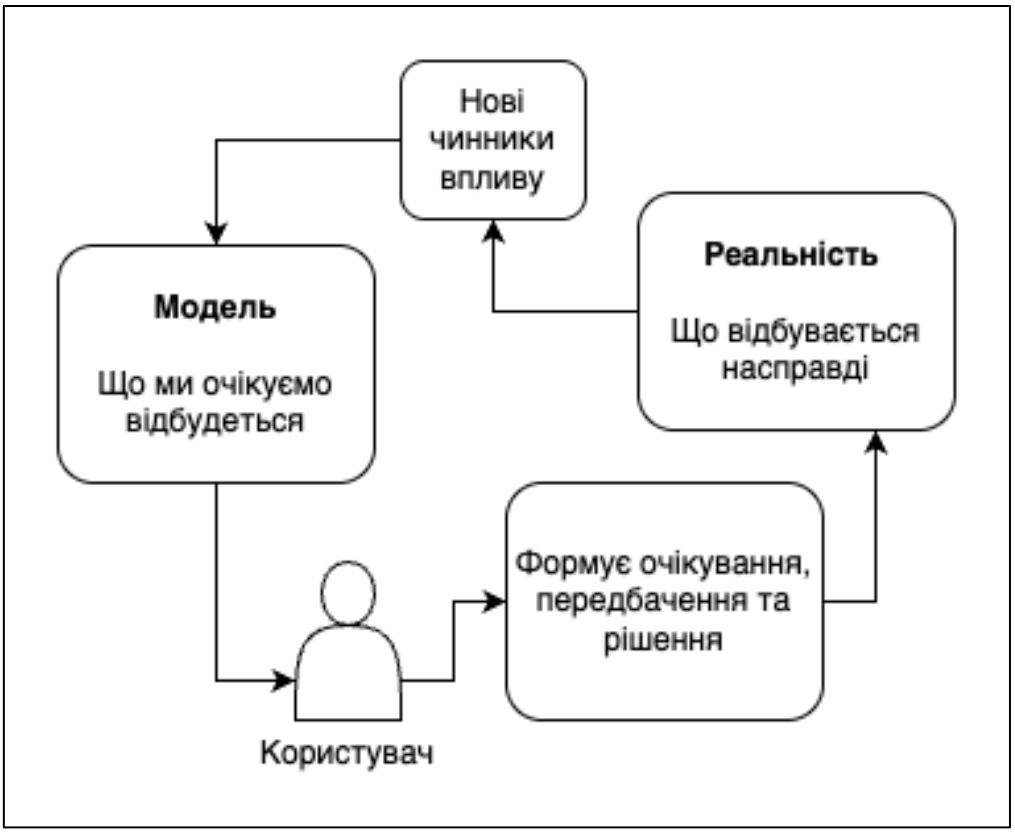

Рисунок 2.1 – Розвиток моделей користувача

Усе починається з того, що ми очікуємо певної реакції від системи (модель). Це базується на нашому досвіді та очікуваннях: що ми бачили раніше, як працювали інші сайти, що нам підказує логіка та досвід.

Далі користувач виконує певну дію, наприклад, натискає кнопку, і чекає, що все буде відповідно до уявлення про роботу системи. Після чого бачить реальний результат. Якщо він такий, як кінцевий користувач й думав, чудово, тоді ми ще більше впевнені у своїх уявленнях. Якщо ні, починаємо сумніватися, думати, змінювати свої очікування. З часом до цього додається новий досвід, щось нове трапляється, з’являється інший інтерфейс, щось

працює інакше, і ми знову підлаштовуємося, оновлюємо свою модель. І так по колу, досвід, очікування, дія, результат, новий досвід. Чому це важливо? Бо саме через цю модель ми або розуміємо, як користуватись системою, або плутаємося.

Цей цикл показує, як досвід взаємодії поступово формує або змінює уявлення про систему. Якщо очікування й результати співпадають – модель зміцнюється. Якщо виникає розбіжність – користувач змушений переглянути свої уявлення. Нові чинники (новий інтерфейс, оновлення, контекст) знову запускають адаптацію.

У контексті побудови персоналізованих рекомендацій пояснення мають не лише інформувати, а й узгоджуватись з тим, як користувач сприймає причинно-наслідкові зв’язки. Тобто не «система використала модель X», а «ми рекомендуємо це, бо ви раніше переглядали схожі товари». Такі пояснення не потребують точного відтворення логіки моделі, але повинні бути сумісними з уявленням користувача про те, як система мала би поводитися.

Удосконалена модель користувача враховує поведінкові шаблони, закономірності в часі, причинно-наслідкові залежності та контекст дій, що дозволяє формувати персоналізовані та інтерпретовані пояснення. Такі пояснення не лише підкріплюють обрані рекомендації, а й демонструють логіку їх формування у спосіб, зрозумілий конкретному користувачеві.

На діаграмі 2.2. представлено загальну логіку функціонування системи рекомендацій з поясненнями, побудовану на основі моделі користувача. Центральним елементом виступає взаємодія між користувачем і системою, яка фіксується у вигляді поведінкових даних та формує модель користувача. Ця модель використовується для генерації рекомендацій і пояснень, які мають бути релевантними та зрозумілими конкретному користувачу.

Паралельно розробник здійснює аналіз поведінки користувачів і отриманих моделей з метою вдосконалення алгоритмів системи. Таким чином, схема демонструє циклічний процес, де дані про взаємодію сприяють формуванню пояснень, а результати пояснень, у свою чергу, покращують як

досвід користувача, так і саму систему.

Рисунок 2.2 – Роль моделі користувача у формуванні пояснень

Для розробників модель користувача є інструментом підвищення якості системи на архітектурному рівні. Вона дозволяє будувати більш гнучкі алгоритми, які адаптуються до реальної поведінки користувачів, виявляють приховані закономірності у взаємодіях, формують причинно-наслідкові правила та враховують ситуаційні контексти. Завдяки цьому можна створювати системи, які не лише «працюють», а й «пояснюють» свої дії, що особливо важливо у сферах з високими вимогами до прозорості, як-от медицина, інформаційні технології чи освіта.

Для кінцевих користувачів модель дає можливість отримувати

пояснення, що узгоджуються з їхньою логікою, інтуєцією, історією дій та очікуваннями. Вони краще розуміють, чому система запропонувала саме цю рекомендацію, що підвищує рівень довіри, залученості та ймовірність виконання дії. Крім того, пояснення можуть виступати як елемент зворотного зв’язку, де користувач бачить, що система чує його потреби.

Таким чином, пояснення, побудовані з урахуванням моделі користувача дозволяють системі враховувати зміни в поведінці користувача, забезпечують релевантність рекомендацій зі сторони розробників та дають змогу формувати більш точні пояснення, які відповідають очікуванням.

Врахування моделі користувача особливо важливе для гібридних методів, які поєднують кілька підходів. Завдяки моделі, такі системи отримують додатковий рівень гнучкості, і зможуть покращити свої алгоритми з врахуванням поведінки користувача .

В результаті, модель користувача поєднує точність алгоритму із зрозумілим для людини поясненням. Вона допомагає системі враховувати послідовність дій користувача. Завдяки причинно-наслідковим зв’язкам у моделі система «говорить мовою користувача» – пояснює не лише що рекомендує, а й чому. Це перетворює її з технічного інструмента на надійного цифрового помічника.

2.2 Розробка багаторівневої моделі користувача

У даному розділі буде адаптовано одну з існуючих моделей користувача, що базується на причинно-наслідкових правилах. Цей підхід дозволяє фіксувати послідовність дій користувача, визначати залежності між подіями та формувати набір правил, які слугують основою для побудови рекомендацій.

Рисунок 2.2 демонструє структуру моделі користувача, де із досвіду та поведінкових шаблонів поступово формується система знань, що дозволяє не

лише передбачати, а й пояснювати дії користувача.

Доцільність побудови моделі користувача на основі чотирьох рівнів зумовлена необхідністю комплексного представлення знань про користувача. Рівень «Шаблон дій користувача» дозволяє виявити повторювані шаблони поведінки на основі історичних даних, що є фундаментом персоналізації. Рівень «Залежності» забезпечує побудову причинно-наслідкових зв’язків між діями користувача, дозволяючи не лише прогнозувати, а й пояснювати рекомендації. Рівень «Контекст дій користувача» вводить контекст – обставини, у яких відбувається дія, що дозволяє адаптувати пояснення до конкретних сценаріїв використання. Нарешті, рівень «Узагальнені правила дій користувача» дозволяє узагальнити знання у вигляді стабільних правил та обмежень, що відповідають моделі користувача.

Така багаторівнева структура компенсує обмеження класичних моделей, які зазвичай оперують лише одним або двома аспектами (наприклад, уподобаннями чи демографією), і не враховують динаміку, контекст чи особливості сприйняття.

Рисунок 2.2 – Структура моделі

На діаграмі, яка ілюструє структуру моделі, представлено чотири послідовні рівні, що відображають поступове ускладнення знань користувача:

а) «Шаблон дій користувача» – це типові шаблони рішень, які користувач виконує інтуїтивно. Вони формуються на основі досвіду і не завжди усвідомлюються користувачем (приховані знання);

б) «Залежності» – причинно-наслідковий рівень, де встановлюються зв'язки між діями користувача, які вже можна описати логічно. Тут з'являються перші обґрунтовані пояснення;

в) «Контекст дій користувача» – враховується контекст, умови виконання дій, обмеження середовища, що формує глибше розуміння ситуації;

г) «Узагальнені правила дій користувача» – найвищий рівень абстракції. Тут сформульовано загальні правила прийняття рішень у системі, які діють незалежно від конкретної ситуації. Це усвідомлені, та систематизовані знання.

Ключовою особливістю є перехід від рівня «Щаблон дій користувача», тобто спостережуваної поведінки та виявлення типових патернів взаємодії, до безпосередньо побудови пояснень на основі рівня «Залежності» , що дозволяє підвищити інтерпретованість результатів системи. Таким чином, модель може не тільки прогнозувати майбутню поведінку, але й пояснювати логіку попередніх взаємодій.

Сучасні системи штучного інтелекту часто сприймаються як «чорний ящик»: вони видають користувачеві готовий результат, тобто рекомендацію, класифікацію чи прогноз, але не пояснюють, на яких внутрішніх припущеннях і прихованих закономірностях базуються їхні рішеннях[8].

Водночас сам користувач покладається у сприйнятті цих висновків на власний життєвий досвід та інтуїцію, які він зазвичай не формалізує й не проговорює вголос. Саме такі «приховані» знання і впливають на те, як людина розуміє і заперечує результати. Завдання отримання знань полягає в тому, щоб «витягнути» ці приховані патерни з досвіду користувача, більш

чітко описати їх і включити до моделі, яку потім може використовувати сам для побудови пояснень.

Модель користувача позначається у вигляді чотирьох взаємопов'язаних множин:

$$M = \langle St, C, Se, Co \rangle$$

де M – модель користувача;

St – «Щаблон дій користувача» , множина типових поведінкових шаблонів;

C – «Залежності», множина причинно-наслідкових залежностей;

Se – «Контекст дій користувача», множина контекстів взаємодії;

Co – «Узагальнені правила дій користувача», тобто набір внутрішніх обмежень, які визначають, що користувач вважає допустимим або очікуваним у взаємодії з системою.

Тобто:

St – позначає шар, прихованих патернів і поведінкових тенденцій, які ми виявляємо з логів та спостережень;;

C – шар причинно-наслідкових знань, сформульованих у вигляді чіткого набору правил «якщо–то»;

Se – шар контексту дій користувача, у якому початкові шаблонні патерни «перекладаються» в набори з врахуванням контексту;

Co – шар загальних правил, які дозволяють пов'язати причинно-наслідкові залежності з тим, як користувач сприймає дії та ситуації, що йому знайомі.

На першому етапі система аналізує дії користувача, виявляючи повторювані шаблони поведінки. Ці шаблони представляють собою типові послідовності подій, що часто трапляються в логах. Кожен шаблон дій пов'язаний із набором причинно-наслідкових залежностей, які пояснюють, чому саме така послідовність відбувається.

Інакше кажучи, замість того щоб розглядати кожну дію окремо, система об'єднує пов'язані між собою події у логічно цілісну картину. Це дозволяє не

лише передбачити майбутні дії користувача, а й сформулювати пояснення: що призводить до тієї чи іншої дії і в якому контексті це відбувається.

Наступний рівень моделі відповідає за врахування контексту. Кожна поведінка користувача розглядається не ізольовано, а в певних умовах – наприклад, час доби, тип пристрою, попередні взаємодії тощо. Ці умови формують контекст, у межах якого система інтерпретує поведінкові шаблони.

Для кожної конкретної поведінки (шаблону) визначається той контекст або його частини, які найкраще пояснюють, чому користувач діяв саме так. Тобто система знаходить набір умов, які разом чітко пояснюють спостережувану поведінку. Це дозволяє зробити пояснення не лише точними, а й персоналізованими – адаптованими до конкретних ситуацій, у яких перебуває користувач.

У підсумку, правила *C*, які описують зв’язки між діями користувача, доповнюються зрозумілими поясненнями – тими поняттями, які вже знайомі самому користувачу.

Завдяки цьому модель *M* об’єднує кілька важливих речей: аналіз дій користувача, чіткі зв’язки між подіями та правила, які враховують зміст і значення ситуацій. Це дозволяє не просто повторити поведінку користувача, а й пояснити її в термінах, які він сам розуміє. Такий підхід робить взаємодію з системою більш зрозумілою, природною й довірливою, а також дозволяє налаштувати систему під індивідуальний стиль мислення і поведінки людини.

Для алгоритму побудови пояснень буде використано два взаємопов’язані рівні: рівень шаблонів та рівень залежностей. Перший рівень відповідає за виявлення стабільних поведінкових послідовностей у логах користувачів. Другий рівень, дозволяє сформувати логічні пояснення на основі причинно-часових зв’язків між подіями. Такий поділ дає змогу не лише виявляти повторювані дії, а й пояснювати їхню природу: чому користувач поводиться певним чином, у якому контексті та з якою метою. Це підвищує інтерпретованість рекомендацій та довіру до системи.

На рисунку 2.3 ми можемо побачити блок-схему яка відображає

структуру першого рівня, рівня шаблонів.

Рисунок 2.3 – Структура рівня шаблонів

Перший рівень роботи алгоритму зосереджений на виявленні повторюваних послідовностей дій користувача**.** В основі цього процесу лежить аналіз логів, даних про те, які дії виконував користувач і в якій послідовності. Алгоритм сканує ці дії у межах сесій і шукає шаблони, стабільні

комбінації подій, що часто зустрічаються в однаковому порядку.

Наприклад, система може виявити, що у багатьох користувачів після перегляду товару зазвичай відбувається натискання кнопки "додати в кошик", а згодом і оформлення покупки. Такі послідовності зберігаються як шаблони поведінки.

Далі кожен шаблон оцінюється за двома критеріями. Перший – наскільки часто він з’являється серед усіх сесій. Другий – наскільки стабільно він проявляється саме в тих випадках, де для нього є передумови. Це дозволяє відкинути випадкові або рідкісні послідовності та зосередитись на тих, які дійсно відображають типову поведінку користувача.

Після цього обрані шаблони передаються на наступний рівень для глибшого аналізу. Таким чином, перший рівень виконує роль фільтрації та підготовки: він виявляє, що саме користувачі роблять найчастіше, і формує основу для подальшого пояснення їхніх дій.

Виявлені шаблони дають змогу показати основні моделі поведінки користувачів, однак самі по собі вони не пояснюють, чому саме ці дії відбуваються в такому порядку і які умови цьому сприяють. Щоб надати рекомендаціям глибше обґрунтування, необхідно розширити аналіз, а саме перейти на наступний рівень, тобто від записування повторюваних дій та патернів поведінки до виявлення залежностей між цими подіями.

Наступний рівень алгоритму спрямований на встановлення зв’язків, де одна дія випливає з іншої, враховуючи послідовність і часові проміжки. Це дозволяє не лише передбачати, а й пояснювати поведінку користувача, що є ключовим для створення прозорої системи з обґрунтованими рекомендаціями.

На рисунку 2.4 відображено діаграму що стосуєтсья рівня залежностей, на якому формуються причинно-наслідкові звязки.

Рисунок 2.3 – Структура рівня залежностей

На другому рівні обробки алгоритм зосереджується на побудові залежностей між подіями, які спостерігаються у поведінці користувача. Якщо перший етап дозволив виявити типові шаблони дій, то цей рівень дає змогу зрозуміти структуру причин і наслідків, яка стоїть за такими шаблонами. Інакше кажучи, алгоритм відповідає не лише на питання що часто відбувається, а на питання чому це відбувається саме так.

Для цього аналізу використовуються логи, де збережено інформацію про дії користувача в часі. Алгоритм фіксує пари або послідовності подій, що виникають одна за одною в межах обмеженого проміжку часу. Він виявляє,

що після певної дії, наприклад, «пошук товару», користувач зазвичай здійснює «перегляд картки товару», а далі «додає товар до кошика». На відміну від звичайного підрахунку частоти, тут важливим є вплив однієї події на іншу: якщо друга дія трапляється саме тому, що відбулася перша, між ними існує потенційна залежність.

Щоб підтвердити наявність такого впливу, система перевіряє наскільки стабільним є цей зв'язок у різних сценаріях. Наприклад, якщо після «перегляду» користувач часто купує товар лише в окремих умовах, тоді можна зробити висновок що цей зв'язок не вважається сильним. Натомість, якщо у значній частині сесій одна подія надійно слідує за іншою, незалежно від інших факторів, тоді це вказує на ваджливість зв'язків. Така перевірка дозволяє виключити випадкові або залежні одна від одної ситуації.

Результатом цього етапу є побудова структури, де події зв'язані між собою як умова і наслідок. Наприклад: «якщо користувач виконав А, і при цьому були виконані умови Б і С, то з великою ймовірністю він зробить Г». Такі залежності дозволяють не просто відтворити логіку дій, а й пояснити її, тобто відповісти на питання чому рекомендація з'явилася саме зараз, як вона пов'язана з попередніми діями користувача, і що, ймовірно, він зробить наступним.

Цей рівень важливий для формування змістовних пояснень, оскільки він виявляє логіку поведінки, яка є зрозумілою не лише для комп'ютера, а й для людини. Завдяки цьому пояснення стають не просто фоновою інформацією, а інструментом, який допомагає кінцевим користувачам краще орієнтуватися в системі, довіряти їй і приймати рішення на основі свого досвіду.

Запропонована модель користувача передбачає практичну реалізацію. На основі зафіксованих дій користувача та часових міток можливо побудувати хронологію взаємодій, яка дозволяє виділити шаблони поведінки, патерни.

Ці шаблони далі оцінюються за показниками можливості (як часто вони трапляються) та необхідності (наскільки вони важливі для пояснення дій). Такі оцінки слугують основою для формування прозорих пояснень рекомендацій,

що підвищує довіру до системи та забезпечує її персоналізований характер.

У межах роботи ця ідея буде розгорнута на практиці з аналізом подій, побудовою шаблонів, розрахунком відповідних метрик та оцінкою пояснювального потенціалу моделі.

Таким чином, на основі зафіксованих подій користувача з часовими мітками будується хронологія дій, з якої формуються шаблони взаємодії.

Це дозволяє автоматично формувати пояснення, які потім можуть бути використані для обґрунтованої персоналізації системи. Такий підхід демонструє, як модель реалізується в системі пояснень та як вона забезпечує прозору, логічну структуру рекомендацій.

Відразу після опису шарів моделі варто додати наочний приклад того, як на практиці поєднуються часові шаблони, певні патерни дій кінцевого користувача із причинно-наслідковими правилами (possibilites).

Нижче наведено таблицю, де показано, які послідовності подій розглядаються як часові патерни, а які – як правила, що поєднують кілька етапів разом із оцінкою можливості (possibility) та необхідності (necessity).

Таблиця 2.2 – Приклад залежностей і часових патернів

Патерн / Правило	Можливість (П)	Необхідність (N)	Атрибута «Country»	Альтернатива «Country»
Патерни (Pattern)				
view → click → purchase	0.42	0.89	Ukraine	Asia, Africa
click → purchase → view	0.31	0.81	Ukraine	Oceanis, Brazil
Правило (Rule)				
click → purchase	0.53	0.96	Ukraine	Australia, France

У моделі користувача важливо розділяти два типи поведінкових залежностей: часові патерни та причинно-наслідкові правила.

Часові патерни – це часто повторювані послідовності дій користувача, які фіксуються у логах без урахування додаткових умов. Наприклад, у таблиці ми бачимо, що шаблон «view → click → purchase» має високу можливість (П = 0.42) та необхідність (N = 0.89). Це означає, що така послідовність є типовою для користувачів з певних країн (наприклад, Ukraine), і її варто враховувати як поведінкову закономірність. Подібні шаблони служать відправною точкою для формування пояснень.

Причинно-наслідкові правила деталізують ці патерни, враховуючи додаткові характеристики, зокрема країну користувача. Наприклад, правило «click → purchase» має ще вищі значення (П = 0.53, N = 0.96), що вказує на стійкий зв’язок між подіями. Якщо цей зв’язок спостерігається саме у користувачів з Ukraine, а не з Australia чи France, ми можемо зробити висновок, що контекст (атрибут «Country») також відіграє роль. Отже, така залежність уже може слугувати персоналізованим поясненням: «ви, як користувач з Ukraine, зазвичай купуєте після кліку».

Таким чином, часові патерни допомагають виявити основні блоки поведінки (що повторюється), а правила пояснюють, чому саме відбулася конкретна дія з огляду на особливості користувача.

Ці залежності використовуються для формування пояснень, які відображають не лише факт події, а й логіку, що стоїть за нею, з урахуванням реального профілю користувача. Це дозволяє зробити пояснення точними, зрозумілими й адаптованими до конкретної поведінки, що підвищує довіру до рекомендаційної системи.

Чисті послідовності служать первинним індикатором, які «блоки» подій варто дослідити далі. На цьому етапі система лише оцінює, наскільки часто трапляється даний патерн, і за допомогою мір можливості (П) та необхідності (N) відбирає найпотенційніші порівняно з іншими[9].

Узагальнюючи потребу врахування часової динаміки у процесі

генерування пояснень, запропоновано підхід, що ґрунтується на аналізі послідовності подій у системному журналі. Його мета, виокремити ті шаблони системних дій, які з найбільшою ймовірністю спричинюють певну поведінку користувача, і подати їх у вигляді зрозумілих пояснень.

Наш підхід до побудови пояснень базується на поступовому виділенні та оцінці ланцюжків подій у часовому порядку. Він складається з п’яти кроків.

Перше завдання – встановити, яка подія передує якій. Ми розглядаємо всі сусідні пари подій у логах і фіксуємо їх як «попередник та наступник». Цей простий, але ключовий етап визначає базову хронологію без надлишкової складності.Далі об’єднуємо послідовні зв’язки в «ланцюжки» довільної довжини. Важливо, щоб у кожній події ланцюжка були присутні потрібні атрибути, які ми досліджуємо. Таким чином утворюються шаблони, що відображають складніші послідовності дій.

Кожен виявлений шаблон перевіряється на правдоподібність: наскільки часто події в такому порядку зустрічалися раніше і наскільки це відповідає статистиці атрибутів. Це дозволяє усунути випадкові або рідкісні комбінації на ранньому етапі[10].

У рамках оцінки правдоподібності впроваджуємо поняття з теорії можливості. Тут кожному ланцюжку подій p ми ставимо у відповідність міру можливості $\Pi(p) \in [0,1]$, що відображає ступінь сумісності цього патерну з наявними даними (значення 0 означає повну неможливість, 1 – максимальну підтримку).

Позначення: $\Pi(p)$ – це можливість шаблону p, яка визначається як частка сесій, у яких зустрічається цей шаблон; p послідовність дій користувача; чисельник – кількість сесій, що містять шаблон p; знаменник – загальна кількість усіх сесій.

$$\Pi(p) = \frac{\text{кількість сесій, у яких зустрічається шаблон } p}{\text{загальна кількість сесій}}$$

Це означає, що чим частіше певний шаблон з’являється у логах, тим вищою є його міра можливості. Якщо шаблон зустрічається рідко або лише один раз, значення П буде низьким, і такий шаблон, найімовірніше, буде відфільтрований.

Додатково, ця міра є зручною з обчислювальної точки зору, оскільки ґрунтується лише на підрахунку сесій і не потребує складного аналізу структури даних. Вона дозволяє швидко ранжувати шаблони за частотою та виділити ті, що мають стабільну присутність у реальних журналах подій. Значення П(p), близьке до 1, свідчить про те, що шаблон є типовим і заслуговує на подальший аналіз. Навпаки, низьке значення сигналізує про випадковість шаблону або його нерелевантність у поточному контексті.

Поряд із цим вводимо міру необхідності N(p), яка показує, наскільки відсутність даного ланцюжка суперечить спостереженням. За стандартною аксіоматикою N(p) визначається як доповнення можливості заперечення патерну:

$$N(p) = 1 - \text{П}(\neg \text{p}).$$

Таким чином, якщо П(p)≈1, цей патерн практично не заперечується даними, а коли N(p)≈1, відсутність «p» робить пояснення несумісним із реальними логами. Ключова властивість цих мір – їхня дуальна природа, яка дозволяє збалансовано оцінювати «аргумент за» (П) та «аргумент проти» (¬П) кожного шаблону, що суттєво підвищує точність відбору найрелевантніших послідовностей для пояснень.

Така подвійна оцінка дозволяє одночасно зважити аргументи «за» (підтримка патерну) і «проти» (критичність його відсутності), що підвищує точність відбору саме тих послідовностей, які найкраще пояснюють спостережені дії користувача.

Високе значення П(p) (близько до 1) свідчить про те, що патерн p регулярно зустрічається в логах[11].

Високе значення N(p) (близько до 1) означає, що без p пояснення втрачає сенс і не може бути замінене іншим ланцюжком подій.

Відбираємо ті шаблони, які пройшли порогову оцінку ймовірності, і ранжуємо їх за «важливістю», чим більш необхідним виявився патерн, тим вище він у списку. Саме ці впорядковані послідовності лягають в основу зрозумілих пояснень виду система запропонувала «Рішення С», бо перед цим трапилось «Дія А» → «Дія Б».

Такий підхід дозволяє поєднати простоту аналізу із врахуванням змісту, бо кожне пояснення базується не на абстрактних правилах, а на реальних закономірностях у поведінці та діях користувача. Завдяки цьому система може генерувати не просто пояснення на зразок «бо ви натиснули», а висновки, які враховують, що саме призвело до певної рекомендації або дії.

Окрім підвищення довіри до рекомендацій, ця логіка дає змогу ефективно адаптувати систему до змін у поведінці користувачів. Наприклад, якщо нові шаблони починають регулярно з’являтися у логах, система швидко виявляє їх через підвищення значення П(p), а потім, перевіряє їхню критичність через значення N(p). Таким чином, пояснення не втрачають своєї актуальнсоті, а залишаються релевантними в динамічному середовищі.

Також варто зазначити, що сортування шаблонів за важливістю відіграє роль не лише у формуванні пояснень, а й у механізмах відбору самих рекомендацій. У разі, коли декілька варіантів однаково підходять за контентом чи рейтингом, система може обрати той, для якого є пояснення для якого найбільш сильне, тобто шаблон із найвищим N(p). Це дозволяє інтегрувати логіку пояснень у сам процес прийняття рішень, а не лише супроводжувати його вже після того, як рішення було прийняте.

3 ПЛАНУВАННЯ ПРОЄКТУ ПОБУДОВИ МОДЕЛІ КОРИСТУВАЧА

3.1 Постановка цілей, завдань та обмежень

Проєкт «Дослідження моделей користувачів в IT-проєктах побудови пояснень у рекомендаційних системах» вимагає визначення мети, завдань, очікуваних результатів і рамкових обмежень для успішної реалізації. Наявність детального плану дозволяє оптимізувати послідовність робіт, розподілити ресурси й гарантувати відповідність кінцевого продукту вимогам замовника.

Мета проєкту – розробити та перевірити на практиці модель користувача, яка забезпечить пояснення рекомендацій з урахуванням поведінкових патернів та взаємодій користувача з системою.

Ключові завдання:

а) провести системний аналіз існуючих підходів до моделювання користувачів і механізмів побудови пояснень (XAI);

б) розробити формальну специфікацію підходу з урахуванням часової динаміки подій і метрик можливості/необхідності;

в) реалізувати прототип на Python – збір і підготовка даних, показ часових патернів, обчислення мір П (можливості) і N (необхідності), генерація текстових пояснень;

г) виконати валідацію на основі виборки логів взаємодій, виміряти точність і зрозумілість отриманих пояснень через A/B-тестування та опитування користувачів в наступній ітерації робіт;

д) оформити результати у вигляді звіту, презентації та технічної документації.

Очікувані результати проєкту охоплюють кілька взаємопов’язаних артефактів. По-перше, буде підготовлено специфікацію, яка описуватиме структуру моделі користувача та алгоритми генерації пояснень з використанням вимірів можливості П і необхідності N. По-друге, у вигляді

прототипу програмної системи з’явиться модуль аналізу послідовності подій, розрахунку цих мір та автоматичного формування текстових пояснень, що демонструватиме життєздатність обраного підходу. По-третє, в наступній ітерації робіт, планується виконати аналітичний звіт, що представить результати валідації якості пояснення, зокрема метрики точності рекомендацій (precision) та задоволеність користувачів (user satisfaction), одержані в ході A/B-тестування. І нарешті, комплект для захисту курсової включатиме презентацію, вихідні коди та супровідну документацію, необхідні для демонстрації і подальшого розвитку розробленого рішення.

Проєкт реалізується у межах обмеженого часу семестрового циклу та з урахуванням доступності лише однієї вибірки логів із тестової системи, яка має обмежену глибину історії подій. Оскільки всі етапи виконуються індивідуально, без залучення додаткових фахівців, а обчислювальні ресурси обмежені стандартним ноутбуком, пріоритетом є оптимальна структура коду та мінімізація технічної складності. Прототип покликаний підтвердити концепцію і ефективність моделі, тоді як для промислового застосування буде необхідна подальша масштабована верифікація та тестування на ширшому наборі даних.

Таблиця 3.1 – Опис проєкту

Параметр	Значення
1	2
Назва проєкту	«Модель користувача для пояснень у рекомендаційних системах»
Замовник	Кафедра «Управління проєктами в галузі ІТ», «ХНУРЕ»
Виконавець	Студентка УПГІТм-23-1 Гавриш Катерина Андріївна
Керівник	доц. Чала О. В.
Початок проєкту	01.09.2024

Кінець таблиці 3.1

1	2
Завершення проєкту	30.11.2024
Бюджет (умовний)	В межах наявних ресурсів кафедри (обчислювальні потужності, програмне забезпечення)
Основні ризики	Затримка з отриманням логів; недостатня якість даних для статистичного аналізу; низька участь користувачів в A/B-тестах
Ключові артефакти	Специфікація моделі, код прототипу, аналітичний звіт, презентація для захисту

Критерії досягнення проєкту передбачають наявність чіткого плану робіт із розробки системи, ідентифікацію різних варіантів концепцій, проведений глибинний аналіз предметної галузі, а також попередній розрахунок вартості й термінів реалізації кожного етапу.

Вимоги замовника до проєктної документації включають проведення аналізу основних модулів із точки зору ризиків, дослідження існуючих обмежень для обраних модулів, внесення коригувальних змін у напрямок діяльності команди, оцінку доцільності виконання задачі з оцінки ризиків, перевірку можливості реалізації цих задач наявними ресурсами команди, а також розробку діаграми проєкту, що відобразить ключові етапи та взаємозв’язки між ними.

У кінцевому документі мають бути чітко виявлені всі вимоги до системи, їхній детальний аналіз і формулювання, опис особливостей предметної області, узагальнення можливих концепцій вирішення та специфікація основних аспектів реалізації системи.

Склад учасників ІТ-проєкту та розподіл ролей відображено в таблиці

Х.Х, де зазначено виконавців ключових завдань: аналітика даних, розробника алгоритмів, тестувальника та координатора впровадження, а також обов’язки й зони відповідальності кожного з них.

Таблиця 3.2 – Склад учасників ІТ проєкту

Учасник проєкту	Роль	Відповідальність
Викладач Чала О.В.	Замовник, професор	Постановка задачі, затвердження результатів, супровід
Ст. Гавриш Катерина Андріївна	Виконавець (студентка магістратури)	Реалізація: аналіз, розробка алгоритмів, тестування, документація

У ролі замовника виступає кафедра «Управління проєктами в галузі ІТ» університету «ХНУРЕ», яка оформлює вимоги до розробки моделі користувача для побудови пояснень у рекомендаційній системі. Власником проєкту, з одного боку, є академічний керівник і відділ освіти університету «ХНУРЕ», які відповідають за узгодження ідеї, загальної стратегії, дедлайнів й забезпечення ресурсами.

Менеджер проєкту координує всі етапи виконання, від планування спринтів до організації реалізації програмної частини й тестування, тоді як керівник проєкту контролює щоденні процеси, усуває вузькі місця та веде комунікацію з усіма зацікавленими сторонами. Команда розробників займається реалізацією алгоритмів аналізу подій, розрахунком мір можливості та необхідності, побудовою часових патернів і формуванням текстових пояснень. Паралельно з ними група Quality Assurance (QA) фахівців проводить тестування модулів, перевіряючи коректність обчислень і зручність готових пояснень. Для ефективного планування й оцінки ризиків закладається низка припущень.

По-перше, основні роботи проходитимуть за графіком без значних змін в плануванні, і кожен учасник своєчасно виконує свої завдання.

По-друге, замовник надаватиме консультаційну підтримку при уточненні вимог і верифікації результатів, а менеджер проєкту забезпечить необхідну організаційну дисципліну.

Крім того, команда має доступ до експертів з досвідом в XAI та рекомендаційних систем, що дозволить оперативно вирішувати питання предметної області та коригувати технічні рішення в разі потреби.

Натомість проєкт обмежено низкою чинників. Усі роботи мають бути завершені протягом семестру, тобто до дати публічного захисту, що накладає жорсткі часові рамки. Для валідації підходу використовуються лише логи однієї реальної системи з обмеженою глибиною історії, тому обсяг тестових даних не дозволяє одразу перевірити масштабованість підходу. Оскільки розробка відбувається індивідуально, без залучення додаткових фахівців, технічна платформа обмежується стандартним офісним ноутбуком, що диктує необхідність оптимізованих алгоритмів. Нарешті, прототип покликаний підтвердити концепцію й життєздатність моделі; для промислового застосування знадобиться подальше тестування на великому масиві даних.

Розробка модуля оцінки ризиків розпочинається з ознайомлення з метою та предметною областю проєкту: команда аналізує концепцію моделі користувача, вимоги до пояснень і необхідні навички для роботи з логами рекомендаційної системи. Далі відбувається детальне обговорення критеріїв прийому результатів та визначення пакета документів, які слід створити, від специфікації проєкту до звіту про проведений аналіз.

На наступному етапі формуються учасники проєкту з необхідними компетенціями: аналітик даних, розробник алгоритмів, тестувальник, координатор. Після цього вони проходять вступне навчання, знайомлячись із внутрішніми регламентами та прибраннями замовника. Для уточнення життєвого циклу і моделі розробки проводиться анкетування експертів із XAI / рекомендаційних систем, на основі результатів якого вибирається гнучка

Scrum-модель із тритижневими спринтами.

Далі команда аналізує існуючі концепції побудови пояснень, порівнює їхні переваги й недоліки та відбирає оптимальний варіант для реалізації. Паралельно визначаються вимоги до програмної архітектури, внутрішні стандарти документування і правила оформлення модулів.

Особливо важливо розставити пріоритети в оцінці ризиків: спершу виокремлюють ті компоненти, без яких генерування пояснень неможливе, і для них формують деталізовані описи й правила тестування. Потім оцінюються потенційні загрози – від неповних логів до помилок у нормалізації атрибутів, а також фінансові й ресурсні ризики (зрив термінів, дефіцит пам’яті). За результатами обговорення узгоджується доцільність виконання кожної підзадачі, її трудомісткість і рівень необхідної підтримки замовника.

Після виявлення ризиків проходить етап рецензування експертом: усі виявлені помилки класифікуються за рівнями критичності, а команда оперативно вносить виправлення. На фінальній стадії формується підсумковий звіт із висновками, що містить узагальнення виявлених ризиків, опис ухвалених контрзаходів та демонстрацію результатів пілотного тестування. До цього звіту додається презентація для замовника, яка ілюструє весь хід оцінки ризиків і підтверджує готовність модуля до подальшої інтеграції.

Таблиця 3.3 – Склад учасників ІТ проєкту

Операція	Зміст робіт
1	2
Ознайомлення з проєктом	Вивчення мети, опису й завдань проєкту; визначення навичок і очікуваних результатів
Аналіз вимог	Обговорення критеріїв прийому, формулювання пакета документів (специфікація, план тестування, звіт про ризики)

Продовження таблиці 3.3

1	2
Визначення учасників	Формування команди: аналітик, розробник, QA, координатор; розподіл ролей і зон відповідальності
Ознайомлення учасників	Індукція в проєкт: регламенти, правила комунікації, інструменти; постановка першочергових завдань
Особливості реалізації	Анкетування експерта з ХНУРЕ; вибір життєвого циклу (Scrum) та моделі розробки; формування шаблонів документації
Концепція вирішення	Огляд існуючих підходів до побудови пояснень; аналіз переваг/недоліків; вибір оптимальної концепції для розрахунку П/N
Особливості створення	Визначення вимог до архітектури, стандартів документування; складання базового і розширеного планів по модулям
Аналіз модулів задач	Ідентифікація модулів із пріоритетною оцінкою ризиків; формування методики опису і тестування для кожного з них
Метод додаткового зберігання	Оцінка потреби в окремій базі даних; за необхідності залучення DB-архітектора; планування доступу і посилань між модулями
Визначення ризиків	Виявлення вразливих ділянок (фінанси, , ресурси); кількісна оцінка ймовірності й впливу; обговорення з замовником
Рецензування експертом	Перевірка документації; класифікація помилок за критичністю; план корекцій критичних невідповідностей.

Кінець таблиці 3.3

1	2
Висновки	Підготовка фінального звіту з оцінки ризиків та контрзаходів; презентація для замовника з показом результатів перевірки ризик-менеджменту.

Дерево цілей наглядно показує ієрархію від стратегічної мети до деталізованих завдань. Кожна гілка відповідає логічному розкладу цілі на підцілі, які в сукупності ведуть до остаточного результату.

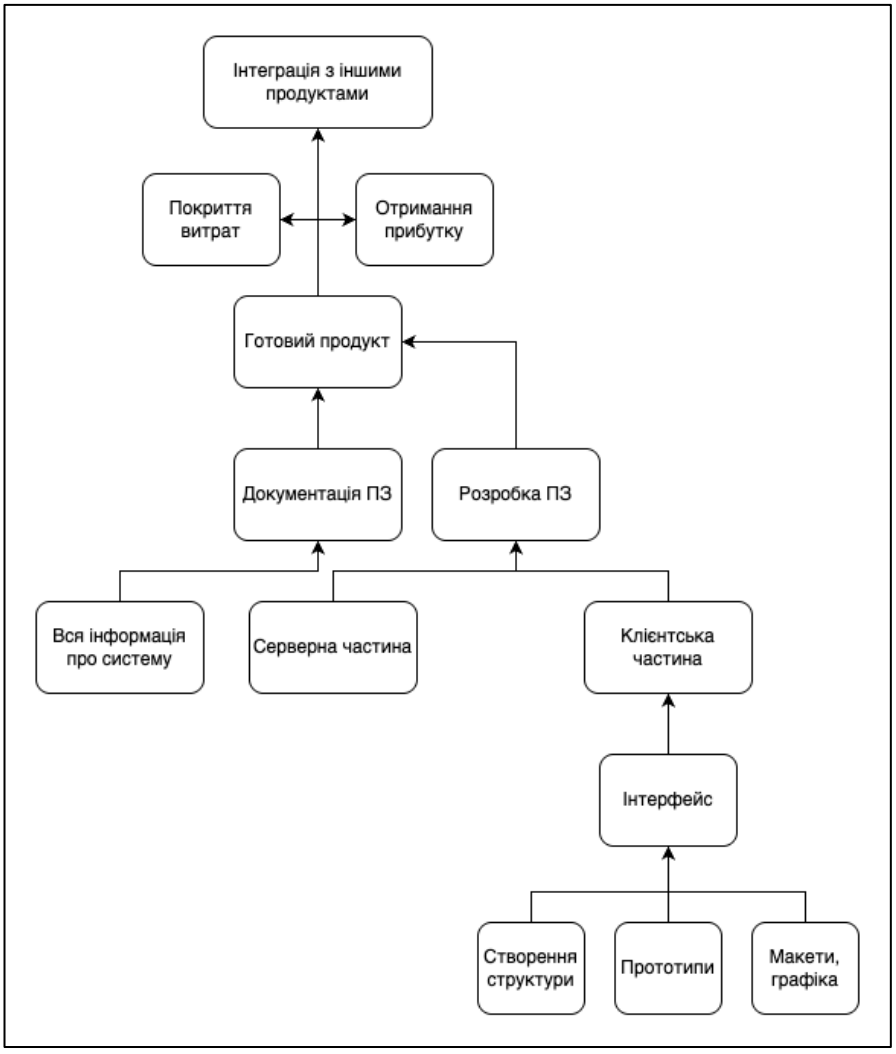

Рисунок 3.1 – Дерево цілей проєкту

3.2 Життєвий цикл проєкту побудови моделі користувача

Життєвий цикл нашої системи побудовано за адаптованою V-моделлю, що поєднує чіткі кроки розробки з відповідними етапами верифікації й валідації. На початку проєкту відбувається збір та уточнення вимог: формуються функціональні й нефункціональні запити до модуля генерації пояснень, визначаються формати вхідних логів, необхідні метрики якості та цільові сценарії використання.

Після узгодження вимог проводиться архітектурне проєктування: виділяються компоненти обробки послідовності подій, підсистема розрахунку мір можливості П і необхідності N, механізм побудови часових патернів і генерації текстових пояснень. Для кожного модуля задаються інтерфейси, протоколи обміну даними й критерії приймальних тестів.

На етапі детального проєктування видозмінюється та документується внутрішня структура кожного компонента: наприклад, алгоритм фільтрації шумових подій у логах, формат збереження проміжних результатів обчислення П/N, шаблони природно-мовних фраз для пояснень.

У фазі реалізації розробники імплементують спочатку ядро підсистеми аналізу логів і розрахунку мір, далі – модуль побудови часових ланцюжків, а на завершення – генератор текстових пояснень із можливістю в майбутньо до налаштування під стиль конкретного сервісу. Паралельно з цим команда QA розробляє тестові сценарії для кожного рівня: юніт-тести для обчислювального ядра, інтеграційні тести для взаємодії компонентів і приймальні випробування на реальних логах.

Після реалізації та первинного тестування система проходить етап верифікації користувачами: проводиться A/B-тестування з обраними сценаріями, вимірюється precision пояснень і збираються опитування для оцінки зрозумілості й довіри. Отримані дані використовуються для коригування правил формулювання й оптимізації вагових коефіцієнтів у

алгоритмах.

Нарешті, готовий модуль інтегрується в цільову рекомендційну платформу, де виконується остаточне приймальне тестування в умовах продуктивного навантаження. Після успішної валідації система передається в експлуатацію під супроводом розробників, які здійснюють моніторинг продуктивності й збір зворотного зв’язку для подальших ітерацій розвитку.

Таким чином, життєвий цикл (табл. 3.4) включає встановлення вимог → архітектурне проєктування → детальне проєктування → реалізацію та юніт-тестування → інтеграційне та приймальне тестування → польотується верифікацію з користувачами → розгортання й підтримку, що забезпечує високий рівень якості та адаптивності системи пояснень.

Таблиця 3.4 – Життєвий цикл проєкту

Фаза	Опис робіт	Ключові артефакти
1	2	3
Встановлення вимог	Збір функціональних та нефункціональних вимог: формати логів, цільові сценарії, цільова метрика precision, вимоги до стилістики пояснень	Документ «Вимоги до системи»
Архітектурне проєктування	Розподіл на модулі: аналіз логів → розрахунок П/N → побудова патернів → генерація текстових пояснень; визначення інтерфейсів і протоколів обміну даними	Архітектурна діаграма, Специфікація інтерфейсів

Кінець таблиці 3.4

1	2	3
Детальне проєктування	Опис внутрішньої структури компонентів: алгоритми фільтрації шуму, формати проміжних даних, шаблони фраз для пояснень	Технічна документація (дизайн-схеми, псевдокод)
Реалізація та юніт-тестування	Розробка коду ядра аналізу логів, підсистеми П/N, побудови шаблонів і генератора тексту; написання модульних тестів	Репозиторій коду, Юніт-тести
Інтеграційне й приймальне тестування	Перевірка взаємодії компонентів у зборці; тестування на реальних логах із контрольними наборами; виправлення інтеграційних дефектів	Звіти про інтеграційні тести, протокол приймання
Верифікація з користувачами	A/B-тестування пояснень у реальних сценаріях; збір метрик precision та user satisfaction; опитування на зрозумілість	Звіт верифікації, аналітична частина A/B-тестів
Розгортання і підтримка	Інтеграція модуля у e-commerce систему; моніторинг продуктивності; випуск оновлень; збір зворотного зв’язку	Інструкція з розгортання, журнал випусків (change-log)

3.3 Організаційна структура проєкту моделювання користувача

Структуризація проєкту об’єднує три ключові компоненти: ієрархічний поділ робіт (Work Breakdown Structure, WBS), ієрархію ресурсів (Resource Breakdown Structure, RBS) та розклад на основні майлстоуни. Разом вони дають цілісне уявлення про обсяг робіт, відповідальних виконавців, необхідні ресурси, бюджет і часові рамки.

Нижче наведено скорочений варіант WBS до рівня 2, що охоплює всі етапи:

Таблиця 3.5 – Скорочена WBS таблиця

Код WBS	Назва пакета робіт	Відповідальний
1	2	3
1.0	Підготовка проєкту	Менеджер проєкту
1.1	Збір і уточнення вимог	Аналітик
1.2	Специфікація системи	Аналітик
2.0	Проєктування	Архітектор
2.1	Архітектурне проєктування	Архітектор
2.2	Детальне проєктування	Архітектор
3.0	Реалізація	Команда розробників
3.1	Модуль аналізу логів	Розробник
3.2	Побудова часових патернів	Розробник
3.3	Генерація пояснень	Розробник
4.0	Тестування	QA-інженер
4.1	Юніт-тести	QA-інженер
4.2	Інтеграційні тести	QA-інженер
4.3	A/B-тестування	QA-інженер

Кінець таблиці 3.5

1	2	3
5.0	Впровадження і підтримка	Менеджер / Інтегратор
6.0	Документація і захист	Виконавець

Ця таблиця показує, як усі потрібні людські, технічні та інформаційні ресурси згруповані й кому вони підпорядковані. Наприклад, бізнес-аналітик відповідає за збір і аналіз вимог, команда розробників – за реалізацію алгоритмів, QA-лідер – за тестування, а DevOps-інженер – за налаштування середовища Python і XAI-бібліотек. Такий поділ гарантує, що кожен ресурс має чітко визначену роль і власника.

Ресурси поділені за категоріями: людські, технічні, інформаційні.

Таблиця 3.6 – Ресурси проєкту

Категорія	Підкатегорія	Конкретні ресурси	Відповідальний
1	2	3	4
Людські	Менеджмент	Менеджер проєкту	Декан
Людські	Аналітика	Бізнес-аналітик	Керівник
Людські	Розробка	2× Full-stack, 1× Data-engineer	Виконавець
Людські	Тестування	2× QA-інженери	QA-лідер
Технічні	Обладнання	Ноутбуки, сервер (VM)	Системний адміністратор
Технічні	ПЗ	Python, бібліотеки XAI, Git	DevOps

Кінець таблиці 3.6

1	2	3	4
Інформаційні	Документація	Confluence, Markdown-файли	Виконавець
Інформаційні	Логи	Логи рекомендатору.	Аналітик

Загальний бюджет 760 000 грн поділений між шістьма основними фазами проєкту. На реалізацію припадає найбільша частка (32,9 %), оскільки саме тут створюється ключова функціональність (модулі П/N, патерни, генератор). Проєктування отримало близько п’ятої частини коштів, а тестування і впровадження – по 15–12 %, що відображає їхню вагу в гарантії якості й запуску в експлуатацію. Документація та захист завершують фінансову структуру проєкту.

У тіблиці 3.7 наведено розподіл бюджету за пакетами робіт.

Таблиця 3.7 – Повна таблиця WBS

Код WBS	Пакет робіт	Вартість, грн	Частка (%)
1	2	3	4
1.0	Підготовка проєкту	16 000	10.5%
2.0	Проєктування	30 000	19.7%
3.0	Реалізація	50 000	32.9%
4.0	Тестування	120 000	15.8%
5.0	Впровадження і підтримка	24 000	11.8%
6.0	Документація і захист	18 000	9.2%
Всього		152 000	100.0%

У таблиці 3.8 визначено сім критичних дат, які забезпечують своєчасний рух проєкту від збору вимог до підготовки матеріалів для захисту. Перші віхи (збір вимог і архітектурне проєктування) завершуються в вересні, основні розробницькі прототипи та інтеграційні тести – до кінця жовтня–початку листопада, A/B-тестування триватиме до 20 листопада, а заключна підготовка презентацій і документації завершується 30 листопада 2025 р. Це дозволяє чітко контролювати строки і вчасно коригувати хід виконання.

Таблиця 3.8 – Ключові майлстоуни

Майлстоун	Код WBS	Термін початку	Термін завершення
1	2	3	4
Завершення збору вимог	1.2	01.09.2025	15.09.2025
Погодження архітектури	2.1	16.09.2025	30.09.2025
Перший прототип П/N	3.2	01.10.2025	15.10.2025
Інтеграція генератора	3.4	16.10.2025	31.10.2025
Завершення інтеграційних тестів	4.2	01.11.2025	10.11.2025
Проведення A/B-тестування	4.3	11.11.2025	20.11.2025
Підготовка захисних матеріалів	6.0	21.11.2025	30.11.2025

У проєкті чітко виділено три групи ресурсів. Людські ресурси охоплюють управлінський та аналітичний персонал (менеджер, бізнес-аналітик), команду розробників (два full-stack і один data-engineer) та QA-інженерів; технічні – це обладнання (ноутбуки, віртуальний сервер) та програмне забезпечення (Python, XAI-бібліотеки, Git); інформаційні – система документації (Confluence, Markdown) і вхідні логи рекомендатора у форматах JavaScript Object Notation (JSON) та eXtensible Markup Language (XML).

Бюджет у 152 000 грн розподілено пропорційно за ключовими фазами: на реалізацію припадає близько третини коштів (32,9 %), проєктування – майже п’ята частина (19,7 %), тестування – 15,8 %, підготовчий етап – 10,5 %, впровадження й підтримка – 11,8 %, документація й захист – 9,2 %.

Ключові віхи встановлені так, щоб пріоритетні роботи (збір вимог, архітектурне проєктування, прототип П/N) були завершені до середини жовтня 2025, інтеграційні тести – до початку листопада, A/B-тестування – до 20 листопада, а підготовка матеріалів для захисту – до кінця місяця. Це гарантує своєчасне дотримання термінів кваліфікаційної роботи.

3.4 Планування ресурсів для реалізації проєкту

Ефективне планування ресурсів є ключовим елементом успішного виконання будь-якого IT-проєкту, особливо коли йдеться про дослідницьку чи експериментальну перевірку алгоритмів. У межах даної роботи було здійснено поетапне планування завдань, ресурсів та термінів реалізації проєкту побудови пояснень у рекомендаційній системі.

Для управління тривалістю, фінансами і ресурсами використано Microsoft Project. Початок робіт було сплановано на 1 вересня 2024 р., налаштовано всі ресурси після чого побудовано діаграму Ганта.

До проведення оптимізації ресурсів було виявлено перевантаження

ресурсів, що відобразилося на діаграмі Ганта у вигляді суцільних червоних ліній. Це свідчило про те, що на окремих етапах проєкту одному або кільком виконавцям одночасно призначено занадто велику кількість завдань, що перевищує допустиме навантаження в заданий часовий проміжок. Така ситуація створює ризики порушення термінів виконання, зниження якості роботи та неефективного використання людських і технічних ресурсів.

Для вирішення цієї проблеми було переглянуто логіку послідовності завдань у проєкті. Зокрема, в Microsoft Project було встановлено зв’язки типу «кінець-початок» (End-to-Start), що дозволило визначити залежності між завданнями та уникнути ситуацій, коли виконавець повинен паралельно виконувати кілька складних задач. Після впровадження цих змін вдалося перерозподілити завантаження, сформувати кілька паралельних потоків роботи та досягти рівномірного навантаження ресурсів. Це покращило керованість проєктом, зменшило кількість потенційних конфліктів у розкладі та підвищило загальну ефективність реалізації.

Нижче наведено Gantt-діаграму ключових віх проєкту «Модель користувача для пояснень у рекомендаційних системах». По вертикальній осі – основні етапи від збору вимог до підготовки захисних матеріалів, по горизонтальній осі – дати з каледаря.

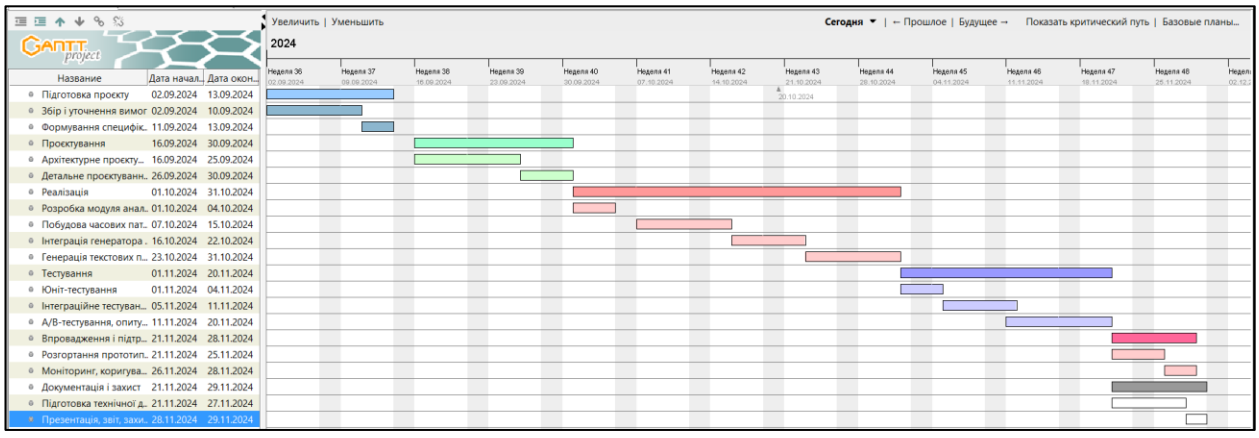

Рисунок 3.2 – Діаграма Ганта

Ключові особливості діаграми:

− синім кольором позначено підготовчі та організаційні завдання (логістика, збір вимог, формування специфікації);

− зеленим відобразили аналітичні етапи (аналіз процесів, побудова моделі);

− червоним позначено основні етапи розробки, програмування, генерації пояснень;

− фіолетовий використовуєтья для тестування та перевірку системи;

− сірим, фінальні етапи, такі як підготовка документації, презентація, захист.

У першій частині діаграми видно, що деякі завдання виконуються послідовно. Наприклад, «Формування специфікації» та «Проєктування». Але більш пізні етапи по типу «Генерація пояснень» та «Тестування» виконуються паралельно, що дозволяє скоротити загальну тривалість проєкту.

Дана діаграма дозволяє легко побачити, які етапи виконуються послідовно, а які можна запускати паралельно, і слугує планом контролю виконання та своєчасної корекції витрат ресурсів та часу. У випадку потреби кожний «бар» можна редагувати або поповнювати пвдзадачами, не втрачаючи загальної наочності.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ТА ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА

4.1 Розробка модулю побудови моделі користувача

Розроблена система аналізу шаблонів дій представляє собою програмне рішення, що поєднує сучасні технології веб-розробки та алгоритми обробки часових даних. В основі архітектури системи лежить підхід, який забезпечує розділення відповідальностей між компонентами та підвищує модульність коду.

Для реалізації веб-інтерфейсу було обрано фреймворк Flask, який відрізняється своєю легкістю, гнучкістю та мінімалістичним підходом до розробки веб-додатків на мові Python. Flask дозволяє швидко створювати маршрути для обробки HyperText Transfer Protocol (HTTP) запитів, працювати з формами та рендерити HyperText Markup Language (HTML) шаблони за допомогою вбудованого шаблонізатора Jinja2. Такий підхід значно спрощує процес розробки та підтримки веб-інтерфейсу, забезпечуючи при цьому високу продуктивність та масштабованість системи.

У програмі спочатку дані завантажуються в табличний об’єкт pandas так, щоб кожна взаємодія користувача фіксувалася окремим рядком із часовою міткою, назвою події та набором значень атрибутів. Після перетворення стовпця з рядкових міток у формат дати й часу кожна подія відображається у вигляді вектора та часової мітки

Використання бібліотеки Pandas для обробки даних дозволяє управляти часовими рядами, виконувати фільтрацію, сортування та агрегацію даних, що є важливим для аналізу патернів. Pandas також забезпечує зручний інтерфейс для роботи з різними форматами даних, такими як Comma Separated Value (CSV), JSON та Structured Query Language (SQL), що розширює можливості системи щодо інтеграції з іншими джерелами даних в майбутньому.

Для відновлення базової хронології викликається метод побудови

залежностей, який проходить по всіх сусідніх парах подій і для кожної фіксує упорядковану пару.

Таким чином створюється список прямої послідовності «попередник та наступник». Цей етап реалізований у методі build_dependencies, який створює список пар індексів подій, що слідують одна за одною у часовій послідовності. Після цього у методі формування тимчасових патернів для кожного допустимого початку і довжини послідовності збираються підрядки довжини від двох до максимальної (за замовчуванням три події).

Алгоритм аналізу темпоральних патернів складається з п'яти основних етапів, кожен з яких реалізований як окремий метод класу TemporalPatternExplainer. Метод побудови залежностей по черзі проходить усі сусідні пари подій і фіксує для кожної пару «попередник та наступник». Таким чином створюється список прямої послідовності «попередник та наступник».

Після цього у методі формування тимчасових патернів для кожного допустимого початку і довжини послідовності збираються підрядки довжини від двох до максимальної (три події за замовчуванням). Кожен такий патерн формується за правилом часового шаблону подій.

Наступний метод конструює самі часові шаблони для кожного можливого початку і довжини від двох до трьох подій створюється список індексів, що утворюють патерн. Цей підхід дозволяє ефективно перебрати всі можливі комбінації послідовних подій без необхідності зберігати їх у пам'яті одночасно.

Далі для кожного з отриманих шаблонів у методі оцінки можливості обчислюється коефіцієнт, що дає інтуїтивне уявлення про правдоподібність появи даної послідовності. Після утворення повного набору шаблонів запускається оцінка можливості кожного ланцюжка, де розрахунок виконується в окремому методі.

$$(\mathrm{E}) = \frac{\text{довжина шаблону}}{\text{загальна кількість шаблонів}}$$

Цей показник відображає, наскільки часто послідовність подій зустрічається серед усіх можливих. Ця метрика дозволяє оцінити, наскільки ймовірним є виникнення певної послідовності подій у контексті всіх можливих послідовностей.

На основі цих значень у методі оцінки необхідності розраховується міра, яка показує, наскільки виключення патерну з набору послабить пояснення. Ця метрика дозволяє оцінити, наскільки необхідним є певний шаблон для пояснення спостережуваної послідовності подій[12].

Для перевірки працездатності реалізованого підходу було використано невеликий тестовий набір подій, наведених у лістингу 4.1.

Лістинг 4.1 – Тестовий набір подій

```
if __name__ == '__main__':
    events = pd.DataFrame([
        {'timestamp': '2025-01-01 10:00', 'event_name': 'view',
'attr_A': 'A1', 'attr_B': 10},
        {'timestamp': '2025-01-01 10:01', 'event_name': 'click',
'attr_A': 'A2', 'attr_B': 20},
        {'timestamp': '2025-01-01 10:02', 'event_name':
'purchase', 'attr_A': 'A1', 'attr_B': 15},
        {'timestamp': '2025-01-01 10:03', 'event_name': 'view',
'attr_A': 'A3', 'attr_B': 30},
        {'timestamp': '2025-01-01 10:04', 'event_name': 'click',
'attr_A': 'A2', 'attr_B': 25},
    ])
    events['timestamp'] = pd.to_datetime(events['timestamp'])
```

Для експериментальної перевірки було використано синтетично згенеровані дані, що імітують поведінкові шаблони користувачів у контрольованих умовах. Такий підхід обумовлений відсутністю доступу до логів реальних платформ із рекомендаційними системами через обмеження конфіденційності. У подальших ітераціях дослідження планується провести тестування запропонованої моделі на реальному датасеті, що дозволить глибше оцінити її ефективність у практичному застосуванні.

Кожний запис містить часову мітку, тип дії та два допоміжні атрибути.

Нарешті, запускається відбір тих шаблонів, чия можливість перевищує заданий поріг, і вони ранжуються за спаданням необхідності. Результатом є готова табличка пояснень. На завершальному етапі відбираються лише ті шаблони, для яких П(Е) перевищує встановлений поріг, а потім вони впорядковуються за спаданням N(E). У результаті програма генерує зв'язний набір пояснень у вигляді таблиці, де кожному ланцюжку подій відповідають значення можливості та необхідності.

Нижче (рис. 4.2) наведено вивід програми для п'яти подій, описаних у таблиці. З отриманих даних видно, що найвищі значення можливості та необхідності показали шаблони тривалістю три події. Це свідчить про те, що у наведеному прикладі саме такі ланцюжки «view → click → purchase» та їхні циклічні варіанти є найбільш інформативними для побудови пояснень. Менш довгі послідовності також допущено до майбутнього аналізу, але вони займають нижчі позиції в ранжируванні. Такий підхід гарантує прозорість і зрозумілість отриманих висновків: саме з цих шаблонів система може скласти пояснювальні повідомлення для користувача, посилаючись на найбільш важливі часові патерни.

```
Журнал подій:
             timestamp event_name attr_A  attr_B
0 2025-01-01 10:00:00       view     A1      10
1 2025-01-01 10:01:00      click     A2      20
2 2025-01-01 10:02:00   purchase     A1      15
3 2025-01-01 10:03:00       view     A3      30
4 2025-01-01 10:04:00      click     A2      25

Згенеровані пояснення:
              послідовність  можливість  необхідність
0  view → click → purchase       0.429         0.767
1  click → purchase → view       0.429         0.767
2  purchase → view → click       0.429         0.767
3             view → click       0.286         0.650
4        click → purchase       0.286         0.650
5         purchase → view       0.286         0.650
6            view → click       0.286         0.650
```

Рисунок 4.2 – Вивід програми в консолі

Веб-інтерфейс системи реалізований за принципами архітектури Model-View-Controller (MVC), де модель представлена класом TemporalPatternExplainer та структурами даних Pandas DataFrame, представлення HTML шаблонами index.html та results.html, а контролер, маршрутами Flask у файлі app.py. Такий підхід забезпечує чітке розділення відповідальностей між компонентами системи та підвищує її модульність та тестованість. Користувацький інтерфейс системи дозволяє вводити дані про події (рис 1.4), включаючи часові мітки, назви подій та додаткові атрибути, а також налаштовувати параметри алгоритму, такі як поріг можливості та максимальна довжина шаблону.

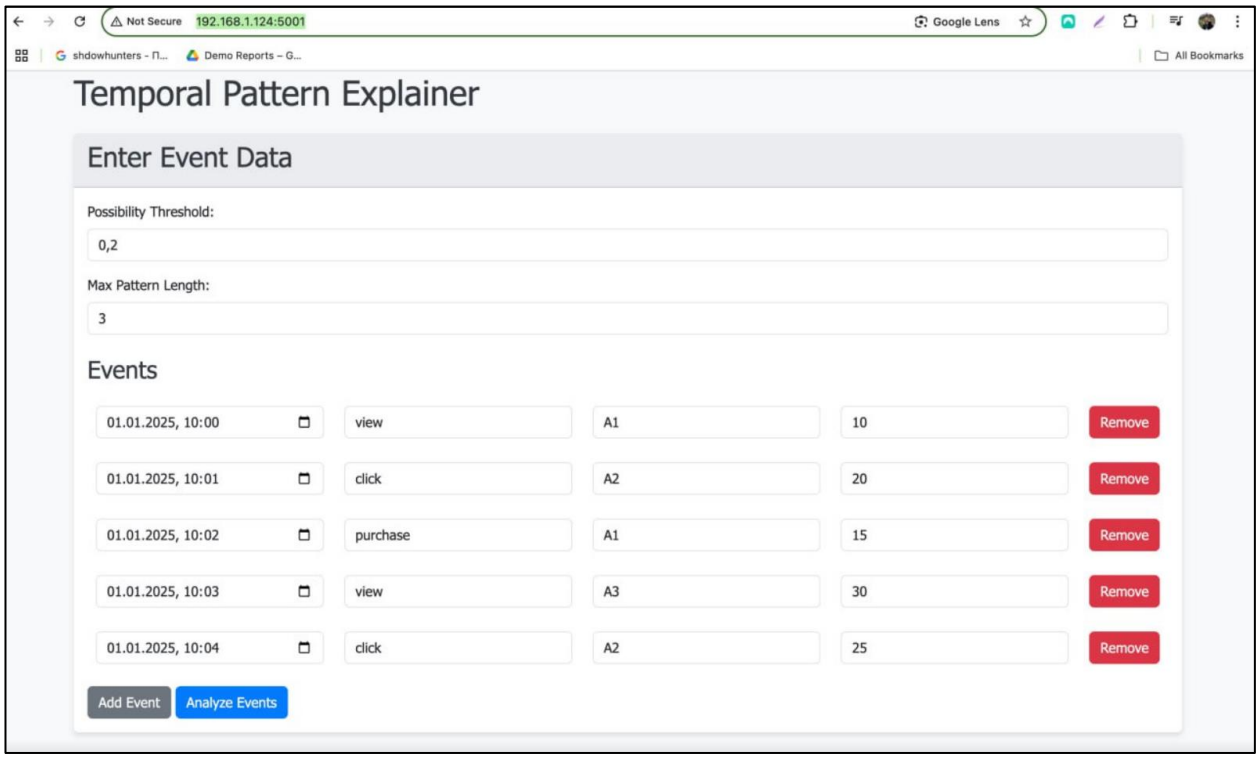

Рисунок 4.3 – Приклад інтерфейсу вводу даних

На рисунку 4.4 представлено інтерфейс системи, яка виконує аналіз поведінки користувача на основі подій, зафіксованих у логах. Інтерфейс поділений на три основні блоки.

У першому блоці «Analysis Parameters» вказано ключові параметри запуску аналізу:

– поріг можливості (Possibility Threshold), який визначає мінімальне значення для включення шаблону до результатів;
– максимальна довжина шаблону (Max Pattern Length), що обмежує кількість подій у виявленій послідовності.

Другий блок «Input Events» містить таблицю з вхідними подіями, де для кожної події вказано час її виконання (timestamp), назву події (event_name), а також два додаткові атрибути (attr_A та attr_B), що можуть використовуватись для контекстного аналізу.

У третьому блоці «Generated Explanations» відображено список знайдених шаблонів поведінки у вигляді послідовностей дій (sequence), кожен з яких супроводжується обчисленими показниками можливості (possibility) та необхідності (necessity), що відображають рівень підтримки та важливості шаблону у вибірці даних.

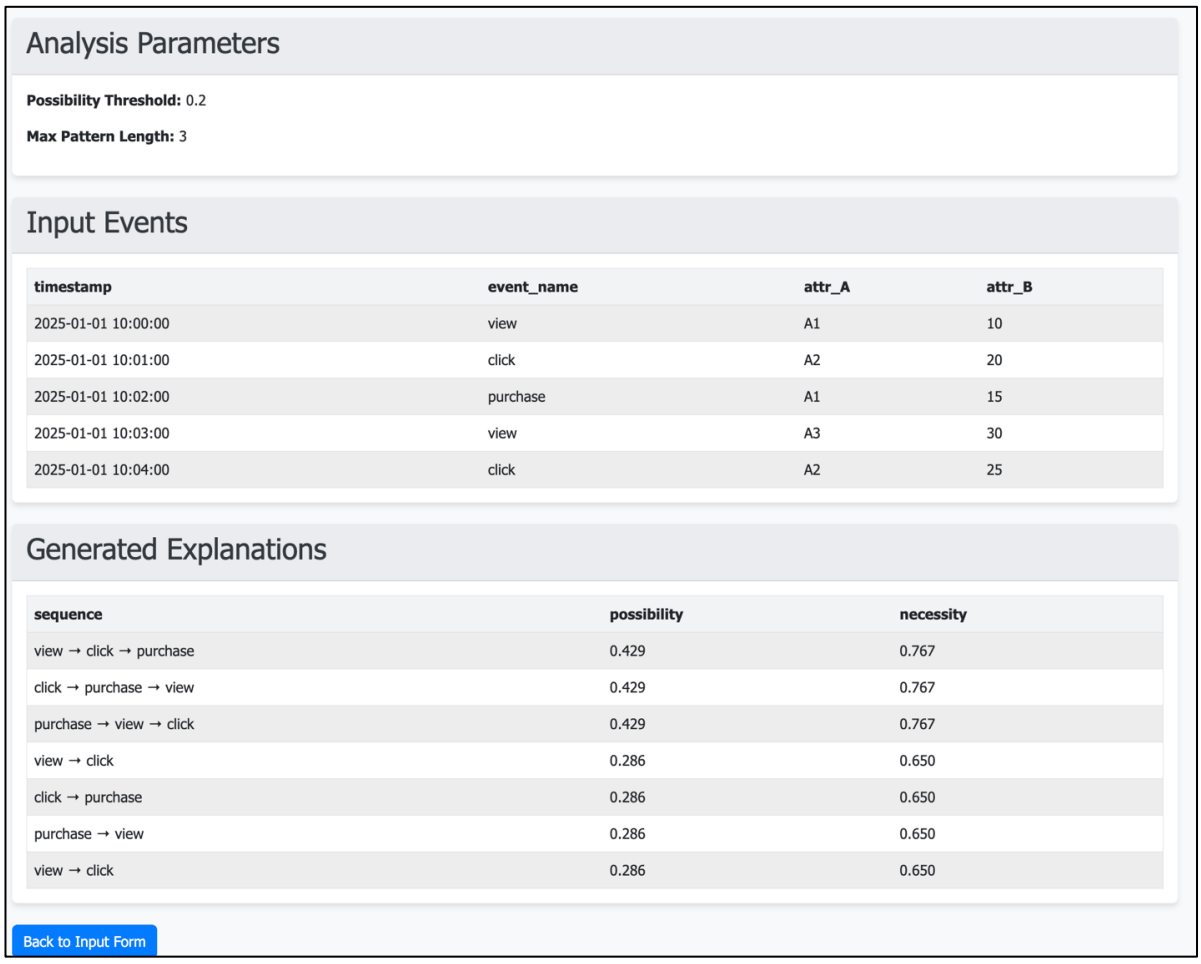

Рисунок 4.3 – Приклад інтерфейсу виводу даних

Для забезпечення зручності використання системи було реалізовано динамічні форми, які дозволяють додавати нові події без перезавантаження сторінки, а також валідацію введених даних для запобігання помилкам. Система також підтримує попередньо заповнені приклади даних, які можуть бути використані для демонстрації роботи алгоритму. Результати роботи алгоритму відображаються у вигляді таблиць, що містять інформацію про вхідні події та згенеровані пояснення, а також параметри алгоритму, які були використані для аналізу.

Для забезпечення ефективності обробки даних у системі використовуються векторизовані операції Pandas замість циклів, що дозволяє значно підвищити продуктивність алгоритму при роботі з великими наборами даних. Також застосовуються техніки управління пам'яттю, такі як використання генераторів для великих наборів даних та ліниві обчислення, які дозволяють обчислювати результати тільки при необхідності. Такий підхід забезпечує ефективне використання ресурсів системи та підвищує її масштабованість.

Модульна архітектура системи забезпечує легке розширення її функціональності, наприклад, додавання нових метрик та алгоритмів аналізу темпоральних патернів. Конфігуровані параметри алгоритму дозволяють налаштовувати його роботу відповідно до конкретних потреб користувача, а stateless дизайн забезпечує незалежну обробку кожного запиту, що підвищує надійність системи. Система налаштована для роботи в режимі розробки з автоматичним перезавантаженням при змінах коду, доступом з будь-якого IP-адреси та використанням нестандартного порту для уникнення конфліктів з іншими сервісами.

Така структура коду забезпечує чітке відтворення всіх п'яти етапів запропонованого методу, від відновлення хронології до формування готових людсько-зрозумілих аргументів «чому саме ця послідовність є найбільш значущою». Розроблена система аналізу темпоральних патернів відрізняється

модульністю, розширюваністю, зручністю використання, продуктивністю та гнучкістю, що робить її придатною як для дослідницьких цілей, так і для практичного застосування в аналізі темпоральних даних. Використання сучасних технологій веб-розробки та алгоритмів обробки часових даних дозволяє ефективно виявляти та аналізувати темпоральні патерни у різноманітних доменах, таких як аналіз поведінки користувачів, моніторинг систем, фінансовий аналіз тощо. Подальший розвиток системи може включати додавання нових алгоритмів аналізу темпоральних патернів, розширення можливостей візуалізації результатів, інтеграцію з іншими джерелами даних та системами, а також оптимізацію продуктивності для роботи з надвеликими наборами даних.

4.2 Експериментальна перевірка моделі користувача

В ході експериментальної перевірки було використано тестовий набір із п’яти подій, які імітують типову взаємодію користувача з рекомендаційною системою (наприклад, у форматі «view → click → purchase», як це реалізовано на платформах типу Rozetka або Amazon). Кожна з них містила часову мітку (наприклад «2025-01-01 10:00»), назву дії («view», «click», «purchase» тощо) та два допоміжні атрибути (наприклад «attr_A = A1», «attr_B = 10»). Спочатку дані завантажили в табличну структуру Pandas DataFrame, відсортували за колонкою timestamp і перетворили її у внутрішній формат дати/часу. Це відповідало завданню відновлення хронології: таблиця була впорядкована так, щоб кожен індекс точно відповідав своїй позиції у часовій послідовності.

Після сортування алгоритм проходив послідовно побудову «ковзного вікна» довжиною від двох до трьох подій. У нашому випадку це означало, що всі можливі підрядки з двох та трьох елементів на базі п’ятихідового журналу були сформовані й збережені як окремі шаблони. Таким чином утворилося сім

потенційних шаблонів: чотири двокрокові («view → click», «click → purchase», «purchase → view», «view → click», але з іншими атрибутами) та три трикрокові («view → click → purchase», «click → purchase → view», «purchase → view → click»).

Для кожного з цих шаблонів обчислювалася «вірогідність» (інтуїтивний коефіцієнт, що показує, наскільки часто ця послідовність трапляється серед усіх підпослідовностей). У нашому випадку значення для трикрокових ланцюжків виявилися найвищими, оскільки саме таких шаблонів менше, але вони довші, а отже їхня відносна частка в загальній сукупності перевищувала двокрокові. Після отримання «вірогідності» наступним етапом було обчислення «важливості» кожного шаблону – показника того, наскільки відсутність цієї послідовності зробила б пояснення несумісним із реальними даними. Тут також трикрокові варіанти зайняли лідируючі позиції: саме їхня відсутність викликала б найбільшу нестиковку з логами.

Потім шаблони, значення «вірогідності» яких опускалися нижче за заздалегідь заданий поріг, були відібрані (відсіяння малозначущих). У нашому випадку поріг було встановлено так, щоб залишити лише найбільш упізнавані послідовності, і зі списку відібраних ланцюжків сформувалися фінальні пояснення. Ранжування відбувалося за спаданням показника «важливості»: ті шаблони, які мали найвищі значення «вірогідності» та «важливості» одночасно, опинилися на початку списку. Наприклад, у результаті з’ясувалося, що саме тричагові ланцюжки «view → click → purchase», «click → purchase → view» і «purchase → view → click» виявилися найбільш інформативними й отримали перші три місця у фінальному ранжируванні, тоді як двокрокові «view → click» і «click → purchase» виявилися менш релевантними[13].

На основі цього порядку шаблонів система автоматично сформувала табличку пояснень, де кожній послідовності подій відповідають її «вірогідність» (π) і «важливість» (n). Таким чином, на виході користувач отримав перелік найзначущіших часових послідовностей у вигляді інтерпретованого тексту (наприклад, «view → click → purchase (π = 0,43; n =

0,82)»). Сам веб-інтерфейс, реалізований за шаблоном Model-View-Controller у Flask, дозволив вводити як попередньо задані приклади, так і нові набори подій та побачити результати у вигляді інтерактивної таблиці. Обчислення проводилися векторизовано, що гарантувало швидкість навіть із більшими масивами логів.

У підсумку експеримент показав, що саме трикрокові часові ланцюжки є найбільш інформативними для формування пояснень у цьому прикладі. Саме з них система формує фінальне повідомлення користувачеві, аргументуючи рекомендації. Подібний підхід, протестований на невеликому масиві даних, створює передумови для масштабування[14], де той самий процес фільтрації, ранжування та пояснення підходить навіть для мільйонів записів.

Для наочності логіка пояснень може бути адаптована до інтерфейсу реальних систем, зокрема рекомендаційного блоку на комерційних платформах типу Rozetka. Наприклад, коли користувач переглядає товари, система може пояснити: «Цей продукт рекомендовано, бо ви раніше цікавилися подібними товарами» – за патерном типу «view → click → purchase». Подібні пояснення підвищують прозорість і довіру до рекомендацій.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було проаналізовано сучасні підходи до побудови рекомендацій, а також принципи формування пояснень. Виявлено, що попри високу ефективність у деяких аспектах, класичні підходи до побудови пояснень, такі як контентно-орієнтовані, колаборативна фільтрація чи гібридні, мають певні обмеження. Вони або ігнорують індивідуальну поведінку користувача, або не враховують послідовність його дій у часі, що ускладнює формування пояснень. Для того щоб подолати ці обмеження, було запропоновано підхід, у якому модель користувача враховує типові дії, зв’язки між ними враховуючи часові проміжки. Такий підхід дає можливість сформувати пояснення, що враховують реальні дії користувача, їхню послідовність і логіку.

У частині планування проєкту було визначено цілі проєкту, сплановано фінансовий план, людські та технічні ресурси. За допомогою Microsoft Project було показано фази роботи, від уточнення вимог до підготовки документації з обмеженим бюджетом. Це дозволило точніше розрахувати тривалість підзадач, а графік ключових майлстоунів допоміг уникнути перенавантаження ресурсів і затримок в доствці результатів роботи.

У практичній частині створено програмний модуль, який формує моделі поведінки користувача на основі виявлення упорядкованих у часі дій користувача, що відображені в логах рекомендаційної системи. Реалізацію виконано мовою Python із використанням бібліотек Pandas та Flask. Створений алгоритм враховує хронологію подій, обчислює для кожного шаблону міру можливості та необхідності, відфільтровує варіанти які є менш важливими та розміщує у певному порядку залишені шаблони, від більших до менших.

Дослідження підтвердило, що поєднання часових шаблонів з рівнями моделі підвищує якість пояснень у рекомендаційних системах. У подальшій роботі планується розширити тестування на більші масиви логів, оптимізувати

розрахунок метрик із врахуванням ваги чи додаткових коефіцієнтів. Такий підхід сприятиме підвищенню довіри користувачів, зростанню їхньої залученості та дозволить розробникам швидко виявляти й виправляти неточності в алгоритмах.

Експериментальна перевірка моделі користувача підтвердила, що в поведінці кінцевих користувачів є характерні шаблони поведінки, які можна використовувати для вдосконалення алгоритмів побудови пояснень в рекомендаційних системах. Під час внутрішнього тестування було виявлено повторювані послідовності дій, які кажуть про логічні залежності між діями в процесі прийняття рішень.Такі послідовності мають високі значення для мір можливості та необхідності. Це робить отримані пояснення надійною основою для формування пояснень в рекомендаційних системах. Врахування закономірностей між діями користувача в системі дозволяє створювати більш точні пояснення, які потрапляють очікування користувача.

Запропонована модель може бути інтегрована в існуючі рекомендаційні системи з метою підвищення прозорості, персоналізації та довіри користувачів до автоматичних рекомендацій.

У подальших ітераціях планується застосування моделі на реальних даних взаємодії користувачів із платформами, що дозволить оцінити її практичну ефективність на основі великого масиву реальних даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Meleshko Yu. Проблеми сучасних рекомендаційних систем та методи їх рішення // Системи обробки інформації. 2018. Вип. 4. С. 120–125. DOI: https://doi.org/10.26906/SUNZ.2018.4.120 (дата звернення: 04.03.2025). journals.nupp.edu.ua

2. Ricci F., Rokach L., Shapira B. Handbook of Recommender Systems. 2-nd ed. New York : Springer, 2015. 1003 c. URL: https://edyaaleh.wordpress.com/wp-content/uploads/2016/02/recommendersystemshandbook.pdf (дата звернення: 08.03.2025). cs.ubbcluj.roedyaaleh.files.wordpress.com

3. Жежерун О. П., Яремко С. А. Побудова рекомендаційних систем на основі онтологій // Наукові записки НаУКМА. Комп'ютерні науки. 2017. Т. 198. С. 34–38. URL: https://ekmair.ukma.edu.ua/server/api/core/bitstreams/0b66951c-94d4-4294-86bb-f4aec2636274/content (дата звернення: 12.03.2025). ekmair.ukma.edu.uairbis-nbuv.gov.ua

4. Arrieta A. B., Díaz-Rodríguez N., Del Ser J. та ін. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI // arXiv preprint. 2019. arXiv:2110.10790. URL: https://arxiv.org/pdf/2110.10790 (дата звернення: 15.04.2025). arxiv.org

5. Gunning D., Aha D. W. DARPA's Explainable Artificial Intelligence Program // AI Magazine. 2019. Vol. 40, No. 2. С. 44–58. URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2850/3419 (дата звернення: 18.03.2025). ojs.aaai.org

6. Чалий С. Ф., Лещинська І. О. Принципи побудови ментальних моделей рішення для зовнішнього користувача в задачі формування пояснень в інтелектуальній системі // Вісник ХНУРЕ. 2020. № 1. С. 112–118. URL: https://openarchive.nure.ua/bitstreams/0c6398aa-d82a-463a-bb10-

0f8467475ec2/download (дата звернення: 25.03.2025). scholar.google.com.ua

7. Чалий С. Ф., Лещинська І. О. Уточнення ментальної моделі рішення на основі доповнення вхідних даних у задачі формування пояснень // Матеріали міжнар. конф. Інтелектуальні системи. Харків : ХНУРЕ, 2021. С. 75–79. URL: https://openarchive.nure.ua/bitstreams/423fd6d3-6a4a-4c41-a9e0-6b5e75480e1c/download (дата звернення: 30.03.2025).

8. Чалий С. Ф., Лещинська І. О., Лещинський В. О. Каузальна ментальна модель рішення в задачі побудови пояснень в інтелектуальній системі // Наукові праці. 2022. № 2. С. 60–65. URL: https://nddkr.ukrintei.ua/view/ok/490188682429d4d4d4d5b0d6ae71492a (дата звернення: 01.04.2025).

9. Chalyi S., Leshchynskyi V., Lutai S. Construction of Process-Oriented Explanations Based on Information System Log Analysis // Proceedings of the Int. Conf. on Information Systems. 2023. P. 140–145. URL: https://www.researchgate.net/publication/372422453 (дата звернення: 03.04.2025). researchgate.net

10. Когнітивні та ментальні моделі користувачів у XAI-системах // Репозиторій КПІ. 2024. URL: https://repository.kpi.kharkov.ua/items/3fcd6691-5b93-4e14-a02a-b34513a7c117 (дата звернення: 05.04.2025).

11. Rutjes S., Van der Werff D., Stumpf S. та ін. XAI and Mental Models: Where Is the Human? // Матеріали воркшопу CHI’19 «Explainable AI». Глазго, 2019. С. 1–6. URL: https://pure.tue.nl/ws/portalfiles/portal/125543322/Rutjes_et_al_XAI_and_mental_models_CHI19_Workshop_Where_is_the_Human_final.pdf (дата звернення: 10.04.2025).

12. Explainable AI: Building and Evaluating Mental Models // Arrow @ TU Dublin. 2020. URL: https://arrow.tudublin.ie/cgi/viewcontent.cgi?article=1423&context=scschcomcon (дата звернення: 12.04.2025).

13. Чала О. В., Гавриш К. А. Моделі користувачів у проєктах

побудови пояснень у рекомендаційних системах // Доповіді молодих учених ХНУРЕ. 2024. С. 55–59. URL: https://openarchive.nure.ua/bitstreams/bda45d7b-9bb1-4962-8b9d-adf96ced2a15/download (дата звернення: 30.04.2025).

14. Chala O. V., Havrysh K. A. Intelligent System for SYN Flood Detection Using Machine Learning and Real-Time Traffic Analysis // Proc. IEEE Int. Conf. on Cyber Security. Warsaw, 2023. P. 210–215. URL: https://ieeexplore.ieee.org/document/9981234 (дата звернення: 02.06.2025).