

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ  
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут  
«Інститут інформаційних технологій в економіці»**

**Кафедра інформаційних систем в економіці**

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА  
«Інформаційні управляючі системи і технології»**

|               |                            |
|---------------|----------------------------|
| галузь знань  | 12 Інформаційні технології |
| спеціальність | 122 Комп'ютерні науки      |

Форма навчання: заочна

**КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА**

на тему: «Розроблення інформаційної управляючої системи для автоматизації  
рекомендаційної стрічки новин»

здобувача Базюка Івана Вікторовича

Науковий керівник: д.е.н., проф. Мозгаллі Ольга Петрівна

---

(підпис)

**Робота допущена до захисту перед екзаменаційною  
комісією з атестації здобувачів вищої освіти (ЕК)**

Завідувач кафедри: канд. екон. наук, доцент, Тішков Богдан  
Олександрович

---

(підпис)

**Київ 2025**

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  
**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ**  
**ІМЕНІ ВАДИМА ГЕТЬМАНА**  
**Навчально-науковий інститут «Інститут інформаційних технологій в економіці»**  
**Кафедра інформаційних систем в економіці**

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА**  
**«Інформаційні управляючі системи і технології»**

|               |                            |
|---------------|----------------------------|
| галузь знань  | 12 Інформаційні технології |
| спеціальність | 122 Комп'ютерні науки      |

ПОГОДЖЕНО:  
Керівник проєктної групи (гарант)  
освітньо-професійної програми

\_\_\_\_\_  
Устенко С.В.  
«07» березня 2025 р.

ЗАТВЕРДЖУЮ:  
Завідувач кафедри інформаційних  
систем в економіці

\_\_\_\_\_  
Тішков Б.О.  
«07» березня 2025 р.

**ІНДИВІДУАЛЬНЕ ЗАВДАННЯ**

здобувача вищої освіти **Базюка Івана Вікторовича**

*(прізвище, ім'я, по батькові)*

заочної форми навчання

*очної (денної), заочної, дистанційної*

на підготовку кваліфікаційної магістерської роботи

**на тему:** «Розроблення інформаційної управляючої системи для автоматизації  
рекомендаційної стрічки новин»

**Тему затверджено наказом ректора Університету від «07» 03 2025 р. №\_465-ст**

**Кваліфікаційна магістерська робота виконується на матеріалах**

---

**План кваліфікаційної магістерської роботи**

**Розділ I** «Дослідження та аналіз підходів до створення інформаційних управляючих систем предметної області»

**Розділ II** «Характеристика інформаційних управляючих систем та методи і моделі»

**Розділ III** «Розроблення проєктних рішень та їх реалізація»

**Об'єкт дослідження** кваліфікаційної магістерської роботи є процес розробки інформаційної  
управляючої системи для автоматизації рекомендаційної стрічки новин.

Предмет дослідження є сукупність теоретичних, методичних і практичних підходів до технології проєктування управляючої інформаційної системи

**Мета кваліфікаційної магістерської роботи:** розробка інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин.

**Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:**

У розділі I Необхідно здійснити всебічне дослідження відповідної предметної області, в межах якого визначаються її ключові переваги та наявні проблеми, з виявленням причин їх виникнення та можливих наслідків. Доцільно провести порівняльний аналіз існуючих аналогів інформаційно-управляючих систем з автоматизації рекомендаційної стрічки новин, встановити їх сильні та слабкі сторони, а також сформулювати обґрунтовані висновки за результатами аналізу. На основі отриманих даних слід обґрунтувати вибір підходів і методів проєктування інформаційно-управляючої системи з автоматизації рекомендаційної стрічки новин, із детальним описом кожного підходу, аналізом його переваг і недоліків та формуванням відповідних висновків.

У розділі II необхідно описати інформаційну управляючу систему автоматизації рекомендаційної стрічки новин, визначити її мету та основні функції. Слід охарактеризувати архітектуру інформаційної управляючої системи з автоматизації рекомендаційної стрічки новин. Визначити структуру та представити характеристику інформаційної управляючої системи. Описати методи та моделі інформаційної управляючої системи з автоматизації рекомендаційної стрічки новин.

У розділі III У межах розробки інформаційної управляючої системи з автоматизації рекомендаційної стрічки новин необхідно сформувати проєктні рішення та реалізувати їх. Слід спроектувати базу даних для зберігання інформації про повідомлення, користувачів і події, описати сутності, їх атрибути, створити інфологічну та фізичну модель БД із характеристиками кожного атрибута. Варто виконати тестування бази даних з урахуванням вимог до тестових даних. Потрібно обґрунтувати вибір архітектури, технологій і середовища реалізації системи, визначити необхідне програмне та технічне забезпечення. Також слід протестувати готову систему, результати подати у вигляді скріншотів у додатках. Провести аналіз процесу тестування системи. На захисті КМР продемонструвати роботу спроектованої ІУС. Окрему увагу приділити техніко-економічному обґрунтуванню, оцінці ризиків, а також надати рекомендації та висновки на основі проведених досліджень і реалізації підсистеми. Надати висновки, в яких описати основні результати дослідження ІУС, прототипу, подальші дослідження з даною темою.

**Завдання підготував**

**науковий керівник**

\_\_\_\_\_

(підпис)

\_\_\_\_\_

(ініціали, прізвище)

«\_\_\_» \_\_\_\_\_ 202\_ р.

**Завдання одержав**

**здобувач**

\_\_\_\_\_

(підпис)

\_\_\_\_\_

(ініціали, прізвище)

«\_\_\_» \_\_\_\_\_ 202\_ р.

## Реферат

Кваліфікаційна магістерська робота містить 106 сторінок, 4 таблиці, 20 рисунків, список використаних джерел з 51 найменувань, 1 додаток.

### **«Розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин»**

*Об'єктом дослідження* кваліфікаційної магістерської роботи є процес розроблення інформаційної управляючої системи для автоматизації формування та персоналізації рекомендаційної стрічки новин.

*Предметом дослідження* є сукупність теоретичних, методичних і практичних підходів до проєктування та реалізації інформаційної управляючої системи, призначеної для автоматизованого аналізу, обробки та відображення новинного контенту відповідно до інтересів користувача.

*Мета і завдання дослідження.* Метою роботи є розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин.

Відповідно до поставленої мети визначено такі основні *завдання*:

- дослідити предметну область, проаналізувати існуючі рішення та підходи до реалізації рекомендаційних систем у сфері новин;
- обґрунтувати вибір технологій, архітектурних рішень і методів побудови інформаційної управляючої системи для автоматизації формування стрічки новин;
- визначити структуру та функціональні характеристики інформаційної управляючої системи;
- описати алгоритми фільтрації новинного контенту та методи персоналізації (зокрема, на основі контенту або поведінкових даних користувача);
- спроектувати базу даних для ефективного зберігання, обробки та швидкого доступу до новинного контенту та інформації про вподобання користувачів;
- розробити програмне забезпечення, яке забезпечує збір, обробку, фільтрацію та ранжування новин;
- описати технічне забезпечення та середовище розгортання системи;
- продемонструвати результати впровадження інформаційної управляючої системи у вигляді прототипу або дослідного зразка.

*Теоретична, методична та практична значущість отриманих результатів.* У процесі дослідження було вивчено предметну область рекомендаційних інформаційних систем, проаналізовано алгоритми фільтрації інформації (зокрема, content-based та collaborative filtering), способи побудови користувацьких профілів, джерела збору новинного контенту та підходи до оцінки релевантності результатів. Визначено потенціал підвищення релевантності новинної інформації, що надходить до користувача, за рахунок персоналізації та адаптивного оновлення стрічки. Практична цінність полягає в можливості впровадження розробленої системи у медіаплатформи, корпоративні портали або внутрішні інфосистеми з метою покращення інформування, зниження перевантаження користувача нерелевантною інформацією та підвищення рівня залучення до контенту.

Впровадження рекомендаційної стрічки новин є логічною складовою сучасної інформаційної управляючої системи, яка має на меті не лише автоматизацію процесів збору та доставки новин, а й підвищення якості сприйняття інформації. З технічної точки зору, така система забезпечує динамічне оновлення стрічки, пріоритезацію новин відповідно до користувацьких інтересів та швидке реагування на нові події. З управлінської точки зору, це створює ефективний канал комунікації між системою та користувачем, сприяючи прийняттю інформованих рішень на основі актуальної інформації.

Рік виконання кваліфікаційної магістерської роботи – 2025.

Рік захисту роботи – 2025.

*Ключові слова:* новини, ACID, база даних, аналіз новин, моделі даних.

**Відгук**  
про кваліфікаційну магістерську роботу  
здобувача навчально-наукового інституту «Інститут інформаційних технологій в економіці»  
освітньо-професійної програми «Інформаційні управляючі системи і технології»  
Базюка Івана Вікторовича  
на тему «Розроблення інформаційної управляючої системи для автоматизації  
рекомендаційної стрічки новин»

1. Актуальність теми: обумовлена стрімким зростанням обсягів цифрового інформаційного контенту та необхідністю персоналізованого й оперативного донесення новин до користувачів. Розробка інформаційної управляючої системи, що автоматизує формування та подання рекомендаційної стрічки новин, відповідає сучасним викликам у сфері медіа, інформаційних технологій і цифрового маркетингу. Така система сприяє підвищенню ефективності сприйняття інформації, задоволенню потреб користувачів і оптимізації редакційних та аналітичних процесів.
2. Позитивні риси кваліфікаційної магістерської роботи: У роботі проведено детальний аналіз існуючих інформаційних систем, що реалізують рекомендаційні механізми, а також обґрунтовано вибір архітектури, технологій та алгоритмів інтелектуальної обробки даних. Особливістю є інтеграція знань із суміжних галузей – машинного навчання, баз даних, інтерфейсного дизайну, що забезпечує комплексний і міждисциплінарний підхід до побудови ІУС.
3. Наявність самостійних розробок автора: Автором запропоновано та реалізовано власні рішення щодо структури бази знань, логіки формування стрічки новин, вибору методу персоналізації та механізмів аутентифікації користувачів. Розробка виконана з урахуванням особливостей предметної області та відповідає потребам користувачів новинних платформ.
4. Цінність теоретичних висновків та практичних рекомендацій: Отримані теоретичні результати дозволяють глибше зрозуміти принципи побудови ІУС для автоматизації новинних потоків, а практичні рекомендації можуть бути використані при впровадженні подібних систем у мас-медіа, аналітичних службах або в корпоративному середовищі.
5. Наявність недоліків: недоліків не виявлено.
6. Загальна оцінка кваліфікаційної магістерської роботи та її допущення до захисту перед ЕК:

Науковий керівник: доктор економічних наук, професор Мозгаллі Ольга Петрівна  
(посада, учене звання, науковий ступінь )

\_\_\_\_\_  
(підпис)

Мозгаллі О. П.  
(прізвище, ініціали)

« \_\_\_\_ » \_\_\_\_\_ 2025 р.

## **Рецензія**

на кваліфікаційну магістерську роботу

здобувача вищої освіти

Базюка Івана Вікторовича

|      |                                                                        |                |
|------|------------------------------------------------------------------------|----------------|
| Тема | <u>Розроблення інформаційної управляючої системи для автоматизації</u> |                |
|      | <u>рекомендаційної</u>                                                 | <u>стрічки</u> |
|      |                                                                        | <u>новин</u>   |

**Актуальність теми кваліфікаційної магістерської роботи і доцільність її розроблення:** Актуальність теми обумовлена зростаючим обсягом цифрового контенту та необхідністю оперативного й персоналізованого подання новин користувачам. Розробка інформаційної управляючої системи для автоматизації рекомендаційної стрічки є доцільною, оскільки сприяє покращенню взаємодії з користувачем, підвищенню ефективності інформування та адаптації контенту до індивідуальних інтересів.

**Якість проведеного дослідження:** Дослідження виконано на високому науково-практичному рівні із застосуванням сучасних підходів до побудови інформаційних систем та технологій обробки великих даних. У роботі ефективно поєднані теоретичні засади з інструментальними рішеннями, що забезпечує її практичну цінність і можливість впровадження.

**Позитивні риси кваліфікаційної магістерської роботи:** Робота має чітко визначену структуру, логічну послідовність викладення, вдале формулювання цілей і завдань. Вона містить авторські технічні рішення щодо формування новинної стрічки, інтеграції бази знань і використання алгоритмів рекомендацій. Це свідчить про самостійність підходу автора та орієнтацію на практичне використання.

**Зауваження.** Робота виконана якісно, зауваження відсутні.

**Практична значимість висновків і рекомендацій:** Практична значимість роботи полягає в можливості підвищення ефективності цифрових новинних сервісів за рахунок автоматизації подання персоналізованих новин. Реалізація запропонованої системи дозволить скоротити час на пошук актуальної інформації, підвищити рівень користувацького досвіду та залучення аудиторії до новинного контенту.

Місце роботи та посада рецензента Appflame, Software Engineer

Науковий ступінь, учене звання (за наявності) ФОП «Степаненко Миколай Ігорович»

## ЗМІСТ

|                                                                                                                    |    |
|--------------------------------------------------------------------------------------------------------------------|----|
| Вступ.....                                                                                                         | 3  |
| РОЗДІЛ 1 Дослідження та аналіз підходів до створення інформаційних<br>управляючих систем предметної області.....   | 10 |
| 1.1 АНАЛІЗ ІСНУЮЧИХ ІНФОРМАЦІЙНИХ СИСТЕМ .....                                                                     | 12 |
| 1.1.1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ .....                                                                         | 13 |
| 1.2 ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ .....                                                             | 14 |
| 1.3 ОБҐРУНТУВАННЯ ВИБОРУ ПІДХОДІВ І ТЕХНОЛОГІЙ ДЛЯ<br>СТВОРЕННЯ ІНФОРМАЦІЙНОЇ УПРАВЛЯЮЧОЇ СИСТЕМИ .....            | 21 |
| РОЗДІЛ 2 ХАРАКТЕРИСТИКА ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ<br>ТА МЕТОДИ І МОДЕЛІ.....                                | 23 |
| 2.1 Структура і характеристика системи .....                                                                       | 23 |
| 2.2 Методи та моделі в інформаційних управляючих системах і технологіях..                                          | 30 |
| Розділ 3 Розроблення проєктних рішень та їх реалізація .....                                                       | 39 |
| 3.1. Проєктування бази даних та/або сховища даних для інформаційної<br>управляючої системи .....                   | 39 |
| 3.1.1 Проєктування інфологічної (логічної) моделі БД.....                                                          | 40 |
| 3.1.2 Обґрунтування вибору СКБД та проєктування фізичної моделі БД в<br>середовищі вибраної СКБД.....              | 44 |
| 3.1.3 Контрольний приклад та обрахунок прогнозних обсягів БД .....                                                 | 53 |
| 3.2 Проєктування бази знань інформаційної управляючої системи та/або засоби<br>інтелектуального аналізу даних..... | 55 |
| 3.2.1 Реалізація інтелектуального аналізу великих даних з метою наповнення<br>бази знань .....                     | 56 |
| 3.3 Програмне забезпечення.....                                                                                    | 58 |
| 3.4 Технічне забезпечення.....                                                                                     | 80 |
| 3.4.1 Обґрунтування вибору технічного забезпечення .....                                                           | 80 |
| 3.4.2 Ресурсні вимоги до технічного забезпечення .....                                                             | 82 |
| 3.5 Реалізація інформаційної системи.....                                                                          | 82 |
| ВИСНОВКИ.....                                                                                                      | 87 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....                                                                                    | 89 |
| ДОДАТКИ.....                                                                                                       | 93 |

## ВСТУП

У сучасному інформаційному суспільстві новини стали не лише головним джерелом знань про навколишній світ, а й важливим інструментом формування громадської думки, соціальних орієнтирів і навіть політичних рішень. Зі стрімким зростанням обсягів цифрового контенту перед користувачем постає проблема надмірної інформації (інформаційного перевантаження), що ускладнює швидкий доступ до релевантних новин. У зв'язку з цим, автоматизація відбору та подання новинної інформації набуває особливої актуальності. Застосування інформаційно-управляючих систем, побудованих на базі алгоритмів штучного інтелекту, зокрема методів машинного навчання та обробки природної мови, дозволяє створити адаптивні рекомендаційні стрічки, що враховують інтереси, поведінку та динаміку споживання контенту кожного окремого користувача.

Актуальність теми обумовлена необхідністю підвищення ефективності взаємодії людини з інформаційними ресурсами та оптимізації процесу ознайомлення з актуальними новинами. У той час як глобальні платформи вже активно використовують подібні рішення, вітчизняні сервіси мають обмежену кількість ефективних рекомендаційних механізмів, що створює попит на нові, більш гнучкі програмні системи, адаптовані до локальних реалій. Розробка такої інформаційної управляючої системи є науково-технічним завданням, що поєднує сучасні IT-рішення, знання про моделювання користувацької поведінки, аналіз текстового контенту та принципи управління потоками інформації.

Метою дипломної роботи є розроблення інформаційної управляючої системи для автоматизованого формування персоналізованої стрічки новин, що динамічно адаптується до потреб користувача.

Об'єктом дослідження виступає процес інформаційної взаємодії користувача з новинним контентом у цифрових середовищах.

Предметом дослідження є програмні та алгоритмічні засоби побудови інформаційно управляючої системи для формування рекомендаційної стрічки новин.

Для досягнення мети в роботі поставлені наступні завдання:

- проаналізувати науково-технічний стан розробок у сфері інформаційних та рекомендаційних систем;
- дослідити структуру, функції та моделі роботи рекомендаційних механізмів;
- спроектувати архітектуру інформаційної управляючої системи;
- реалізувати базову модель формування стрічки новин на основі зібраних даних;
- провести оцінку ефективності системи та визначити перспективи її вдосконалення.

У роботі використано такі методи дослідження: метод аналізу й синтезу наукової літератури, системного підходу, моделювання користувацької поведінки, кластеризації даних, алгоритмічного аналізу, а також засоби програмної реалізації рекомендаційних моделей з використанням РНР та суміжних технологій.

Апробація результатів роботи відбулася в рамках внутрішнього тестування інформаційної системи на базі ІТ-компанії (умовного замовника), де було перевірено релевантність результатів та швидкодію прототипу. За результатами апробації сформовано рекомендації щодо масштабування функціоналу системи.

Наукова проблема, що лежить в основі даного дослідження, полягає у підвищенні ефективності автоматизованої фільтрації та персоналізації новинного контенту в умовах стрімкого зростання інформаційних потоків та мінливих інформаційних запитів користувачів. Незважаючи на наявність окремих прикладів рекомендаційних систем у великих інформаційних сервісах, універсальні адаптивні рішення, здатні якісно й швидко формувати персоналізовану новинну стрічку, залишаються обмеженими, що створює потребу в подальших дослідженнях та розробках у цій сфері.

Результати дослідження були апробовані у тестовому середовищі внутрішнього проєкту ІТ-компанії, яка спеціалізується на інформаційній підтримці веб-ресурсів. В рамках апробації здійснено практичну перевірку працездатності системи, визначено її переваги щодо підвищення релевантності подачі новин, а також зібрано зворотний зв'язок для подальшого вдосконалення. Отримані результати можуть бути застосовані при розробці систем динамічного контентного супроводу новинних сайтів, блогів, навчальних платформ і спеціалізованих порталів.

Попри стрімкий розвиток цифрових технологій та зростаючий інтерес до систем рекомендацій, тема розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин ще не отримала повного й всебічного висвітлення у спеціальній вітчизняній літературі. Значна частина існуючих досліджень зосереджена або на загальних принципах побудови інформаційних систем, або на специфіці рекомендаційних алгоритмів, тоді як цілісне поєднання цих напрямів із практичною реалізацією саме у форматі автоматизованої новинної стрічки в межах єдиної керуючої системи залишається малодослідженим. Це й обумовлює наукову новизну обраної теми та необхідність її.

Для досягнення зазначеної мети в дипломній роботі необхідно вирішити такі задачі:

- вивчити сучасний стан розробок у галузі інформаційно-рекомендаційних систем;
- описати архітектурні підходи до побудови інформаційних управляючих систем;
- встановити вимоги до системи автоматизованого формування стрічки новин;
- виявити ефективні методи збору, обробки та фільтрації новинного контенту;
- змодельовати логіку взаємодії користувача з системою;
- розробити структурну модель та функціональні модулі системи;

- реалізувати програмний прототип системи в середовищі Open Server;
- провести тестування, оцінити ефективність роботи та сформулювати рекомендації щодо подальшого застосування.

Особистий внесок автора полягає у самостійному аналізі теоретичних основ проблеми, побудові логіки функціонування інформаційної управляючої системи, проектуванні її архітектури, реалізації прототипу та проведенні тестування системи. Автором також сформульовано рекомендації щодо подальшого використання та масштабування запропонованої системи.

У процесі досягнення поставленої мети було використано комплекс методів дослідження, що поєднує теоретичний аналіз, моделювання, системне проектування та практичну реалізацію.

На теоретичному рівні застосовано методи аналізу і синтезу наукової літератури, контент-аналізу та порівняльного огляду існуючих рішень у сфері рекомендаційних систем і інформаційного управління. Для формалізації структурних елементів системи та її логіки використано методи об'єктно-орієнтованого моделювання, зокрема UML-діаграми, що дали змогу спроектувати архітектуру системи відповідно до сучасних стандартів розробки програмного забезпечення.

На прикладному рівні реалізації застосовано методи компонентного проектування, програмної верифікації, тестування та оптимізації алгоритмів. Також використано елементи машинного навчання при обробці новинного контенту й моделюванні уподобань користувача.

Усе це дозволило забезпечити повноцінне проходження всіх етапів життєвого циклу системи – від аналізу вимог до створення та апробації прототипу.

Висновки кваліфікаційної магістерської роботи підтверджують її високу практичну та теоретико-методичну значущість. Обрана тема є актуальною в умовах зростаючої потреби в персоналізованому доступі до інформації, особливо для підприємств, які займаються медіа, контент-маркетингом або автоматизованим супроводом новинних платформ. Розроблена інформаційна управляюча система

пропонує сучасний підхід до автоматизації рекомендаційної стрічки новин із використанням алгоритмів персоналізації, що враховують інтереси користувача. Це дозволяє суттєво підвищити ефективність подання новинного контенту, зменшити навантаження на редакторські команди та покращити користувацький досвід.

Принципові рішення, реалізовані в роботі, мають прикладну цінність і можуть бути впроваджені в реальні проєкти або адаптовані під потреби конкретних установ. Створений прототип системи продемонстрував працездатність і потенціал масштабування. У роботі наявні елементи наукової новизни зокрема, поєднання логіки інформаційного управління з рекомендаційними механізмами. Також було застосовано сучасні методи об'єктно-орієнтованого проєктування й технології моделювання, що дозволяє використовувати напрацювання не лише на практиці, але й як методичну базу в навчальному процесі з курсів програмування, системного аналізу та розробки ПЗ. Робота рекомендована до впровадження й подальшого вдосконалення в умовах реальних інформаційних систем.

Інформаційною базою дослідження стали наукові праці вітчизняних і зарубіжних дослідників у сфері інформаційних систем, автоматизації, рекомендаційних алгоритмів та обробки даних, зокрема публікації науковців, присвячені розробці новинних стрічок із застосуванням технологій штучного інтелекту та машинного навчання.

Вагому роль у формуванні теоретичного підґрунтя відіграли огляди сучасних підходів до побудови систем персоналізованих рекомендацій, опубліковані в електронній бібліотеці arXiv, у виданнях *Information Processing & Management*, *ACM Transactions on Information Systems*, а також у спеціалізованих конференційних збірниках з комп'ютерних наук.

Також було використано статистичні матеріали з офіційних джерел, таких як звіти компаній Google, Meta та українських ІТ-платформ, що стосуються тенденцій споживання цифрового контенту.

Окреме місце в інформаційній базі посіли методичні рекомендації з проєктування інформаційних управляючих систем, надані у фаховій навчальній та

технічній літературі. Прикладні аспекти реалізації та тестування прототипу ґрунтувалися на офіційній документації до інструментів Open Server Panel, MySQL, PHP та мов програмування, які використовувались у роботі.

Таким чином, інформаційна база забезпечила комплексне охоплення як теоретичних основ, так і практичних інструментів дослідження.

Структура комплексної магістерської роботи побудована відповідно до логіки дослідницько-прикладного підходу та охоплює всі етапи проектування, реалізації й аналізу функціонування інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин. Робота складається зі вступу, трьох основних розділів, висновків, списку використаних джерел та додатків. У вступі обґрунтовано актуальність теми, визначено мету, завдання, об'єкт і предмет дослідження, наведено характеристику використаних методів, окреслено наукову новизну, практичну значимість та інформаційну базу дослідження.

У першому розділі здійснено теоретичний аналіз проблематики автоматизації інформаційних систем, висвітлено сучасний стан розвитку рекомендаційних сервісів, класифікацію відповідних алгоритмів і підходів, а також проаналізовано особливості архітектури новинних стрічок. Другий розділ присвячено моделюванню й проектуванню системи: визначено функціональні вимоги, розроблено логічну та фізичну архітектуру, створено UML-діаграми, а також описано процес побудови бази даних. Третій розділ має прикладний характер і містить опис реалізації програмного прототипу, середовищ розробки, обраних мов і технологій, результати тестування, оцінку ефективності та аналіз доцільності впровадження в реальних умовах. У висновках підбито підсумки виконаної роботи, сформульовано основні результати та вказано напрями подальших досліджень. Така послідовність розміщення розділів забезпечує логічність, послідовність викладення матеріалу та відображає цілісний підхід до вирішення поставленої задачі.

Таким чином, дана робота має як теоретичну, так і практичну цінність, оскільки поєднує науковий підхід до організації системи комунікації з реальними задачами розробки програмного забезпечення, а також є не лише спробою

вирішити актуальну прикладну проблему, але й робить внесок у наукову дискусію щодо оптимізації комунікацій у складних інформаційних системах.

## **РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ПРЕДМЕТНОЇ ОБЛАСТІ**

Основна частина комплексної магістерської роботи присвячена ґрунтовному аналізу теоретичних основ, що лежать в основі розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин. У першому розділі викладено огляд сучасного стану досліджень у цій галузі, зокрема проаналізовано підходи до побудови інформаційних систем, роль управляючих компонентів у їх структурі, а також особливості функціонування новинних платформ в умовах цифрової трансформації. Значну увагу приділено вивченню принципів побудови систем персоналізованих рекомендацій, зокрема контентно-орієнтованих, колаборативних та гібридних методів, які широко застосовуються для підвищення релевантності інформації, що пропонується користувачеві.

Огляд літератури охоплює праці українських і зарубіжних науковців, які досліджують ефективність та адаптивність рекомендаційних механізмів, зокрема в умовах швидкоплинного новинного середовища. Вивчено різні підходи до побудови новинних стрічок із використанням машинного навчання, великих мовних моделей (LLM), обробки природної мови [1] (NLP), а також інструментів моніторингу інформаційних потоків. Окремо проаналізовано діючі системи, що використовуються на медіа-платформах, таких як Google News [2], Facebook News Feed [3], Upday [4], тощо, та визначено їх ключові особливості, переваги і недоліки, що стало підґрунтям для формулювання вимог до проєктованої системи.

Розглянуто моделі представлення інтересів користувача, підходи до адаптації системи до змін у поведінці аудиторії, механізми обліку часових і тематичних характеристик новинного контенту. Такий теоретичний аналіз і огляд сучасних рішень забезпечили надійну основу для подальшого проєктування і розробки інформаційної управляючої системи, яка б відповідала актуальним технологічним вимогам і запитам цифрового інформаційного простору.

Значення теми дипломної роботи полягає у її відповідності сучасним технологічним викликам та інформаційним потребам суспільства. В умовах постійного зростання обсягів цифрового контенту, особливо новинного характеру, виникає гостра потреба в ефективних механізмах фільтрації, структуризації та персоналізації інформації для кінцевого користувача. Рекомендаційні стрічки новин, які функціонують на базі інформаційних управляючих систем, стають ключовим інструментом у забезпеченні релевантного інформаційного доступу, підвищенні залученості аудиторії та якості комунікації між джерелами інформації і споживачами.

Розроблення такої системи не лише вирішує задачу адаптації стрічки новин до індивідуальних інтересів користувача, а й має вагоме значення для цифрових медіа, державного сектору, освітніх платформ та ІТ-компаній, що працюють з великими масивами даних. Вона дозволяє підвищити ефективність інформаційного обслуговування, зменшити когнітивне навантаження, покращити навігацію в інформаційному середовищі та оптимізувати взаємодію між системою і користувачем. Таким чином, тема дослідження має міждисциплінарну значущість і є надзвичайно актуальною як у науково-дослідному, так і у прикладному вимірі.

На підставі проведеного теоретичного аналізу існуючих моделей та методів побудови рекомендаційних систем, було здійснено синтез власної інформаційної управляючої системи для автоматизації новинної стрічки. Основою синтезу стало узагальнення ефективних рішень, адаптованих до специфіки новинного контенту, динамічної зміни користувацьких інтересів і вимог до швидкої обробки даних. Було враховано особливості поєднання контентно-орієнтованих і колаборативних підходів, що дозволило побудувати гібридну архітектуру, здатну до гнучкої адаптації під індивідуальні потреби кожного користувача.

У процесі синтезу системи сформовано структурну схему, яка логічно відображає всі функціональні компоненти. Центральною частиною системи виступає модуль обробки даних, що отримує інформацію з вхідного потоку новин, проводить її попередню обробку, класифікацію та подальше формування користувацьких профілів. Управляючий модуль відповідає за аналіз поведінки

користувача, обчислення релевантності контенту на основі історії взаємодій та формування персоналізованої стрічки. Інтерфейсний модуль забезпечує зручне представлення результатів у вигляді стрічки новин, а також збирання зворотного зв'язку для подальшого навчання системи.

Розроблена структурна схема є логічною основою для реалізації програмного продукту та демонструє послідовність і взаємозв'язок між усіма етапами автоматизації рекомендаційного процесу в новинному середовищі.

## **1.1 АНАЛІЗ ІСНУЮЧИХ ІНФОРМАЦІЙНИХ СИСТЕМ**

У розділі "Дослідження предметної області" доцільно зосередитися на описі сучасного стану, ключових понять та особливостей функціонування інформаційних управляючих систем в ІТ-галузі України. Інформаційні технології виступають фундаментальною складовою цифрової трансформації економіки[5], а сама ІТ-галузь протягом останніх десяти років демонструє стабільну позитивну динаміку. Зокрема, фіксується значне зростання обсягів експорту програмних продуктів та послуг, що свідчить про високий рівень довіри до українських розробників на міжнародному ринку. Разом із тим, системною проблемою галузі залишається дефіцит кваліфікованих кадрів, що обумовлює необхідність автоматизації ряду процесів, у тому числі й управління інформаційними потоками, взаємодії з клієнтами, обробки запитів тощо.

### 1.1.1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

У межах дослідження предметної області інформаційних управляючих систем, що забезпечують автоматизацію рекомендаційних стрічок новин, слід зазначити, що сучасні цифрові платформи та інформаційні ресурси дедалі частіше потребують інструментів, здатних забезпечити користувача релевантним, своєчасним і персоналізованим контентом. Рекомендаційна стрічка новин — це механізм, який формує індивідуалізовану послідовність інформаційних повідомлень, новин або статей на основі аналізу інтересів, поведінки та уподобань конкретного користувача.

Ключовими об'єктами в даній предметній області є користувачі, інформаційні джерела, алгоритми фільтрації та ранжування, контентні та поведінкові профілі, а також канали подання інформації. Важливу роль відіграє чітке структурування процесу збору даних, аналізу активності користувача, формування рекомендацій та доставки контенту через зручні для сприйняття інтерфейси. Це зумовлює необхідність розроблення інтегрованого модуля рекомендацій, який би забезпечував масштабованість, персоналізацію та високу релевантність відображуваної інформації.

Процеси в системах рекомендацій зазвичай включають фіксацію дій користувача (перегляди, кліки, оцінки), обробку зібраних даних, запуск алгоритмів рекомендації (наприклад, контентно-орієнтованих чи колаборативних фільтрів)[6], формування списку новин та його виведення у зручному форматі. Між об'єктами системи встановлюються взаємозв'язки, що базуються на моделях «поведінка — інтерес — контент», де дії користувача активують механізми персоналізації згідно з його попереднім досвідом або подібністю до інших користувачів.

Інформаційні управляючі системи з функцією рекомендаційної стрічки новин значною мірою покращують користувацький досвід, підвищують зацікавленість у споживанні контенту, оптимізують навантаження на інформаційні платформи та сприяють прийняттю рішень на основі актуальної інформації. Це

робить дослідження цієї предметної області актуальним як у сфері мас-медіа, так і в освітніх, комерційних та урядових інформаційних системах.

## 1.2 ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ

У процесі дослідження предметної області автоматизації рекомендаційної стрічки новин важливо проаналізувати наявні реалізації подібних інформаційних управляючих систем, що вже використовуються на практиці. Ознайомлення з аналогами дозволяє не лише оцінити рівень розвитку галузі, а й виокремити найбільш ефективні архітектурні та функціональні рішення, які можуть бути адаптовані або вдосконалені при створенні власної системи. Зокрема, актуальним є аналіз платформ, які реалізують персоналізовану подачу інформації та механізми автоматизованого інформування користувачів, що прямо корелює з цілями проекту.

У контексті автоматизації рекомендаційної стрічки новин та реалізації нотифікаційних механізмів можна виокремити низку актуальних аналогів готових інформаційних управляючих систем, які застосовуються у сучасних цифрових продуктах та платформах такі наприклад як:

- **google news;**
- **flipboard;**
- **microsoft start;**
- **facebook news feed;**
- **smartnews.**

**Google News** — це один із провідних світових сервісів агрегування новин, розроблений корпорацією Google, який автоматично формує персоналізовану стрічку новин для кожного користувача. Система використовує складні інформаційні управляючі механізми, що поєднують елементи штучного інтелекту, машинного навчання та обробки природної мови (NLP), що дозволяє забезпечувати

високий рівень адаптації інформаційного контенту до індивідуальних інтересів користувача.

З технічного погляду, Google News реалізує модель багаторівневої фільтрації та класифікації новинного контенту з тисяч джерел у реальному часі. Ключовим елементом є механізм кластеризації новин за тематиками та важливістю, а також побудова користувацького профілю на основі попередніх взаємодій, місцеперебування, запитів у пошуковій системі та поведінки в суміжних сервісах Google. Повідомлення та рекомендації можуть формуватися як у пасивному вигляді (за замовчуванням у стрічці), так і у формі push-нотифікацій, що активізуються на основі алгоритмічного передбачення зацікавленості користувача в конкретній події чи темі.

У контексті інформаційних управляючих систем Google News слугує зразком високорівневої автоматизації процесів обробки великих обсягів неструктурованих даних, реалізуючи принципи багатокритеріальної [7] оцінки інформації, адаптивного управління потоками даних та персоналізованої доставки повідомлень у масштабах глобального інтернету. Інформаційна система Google News наведена на рис. 1.1.

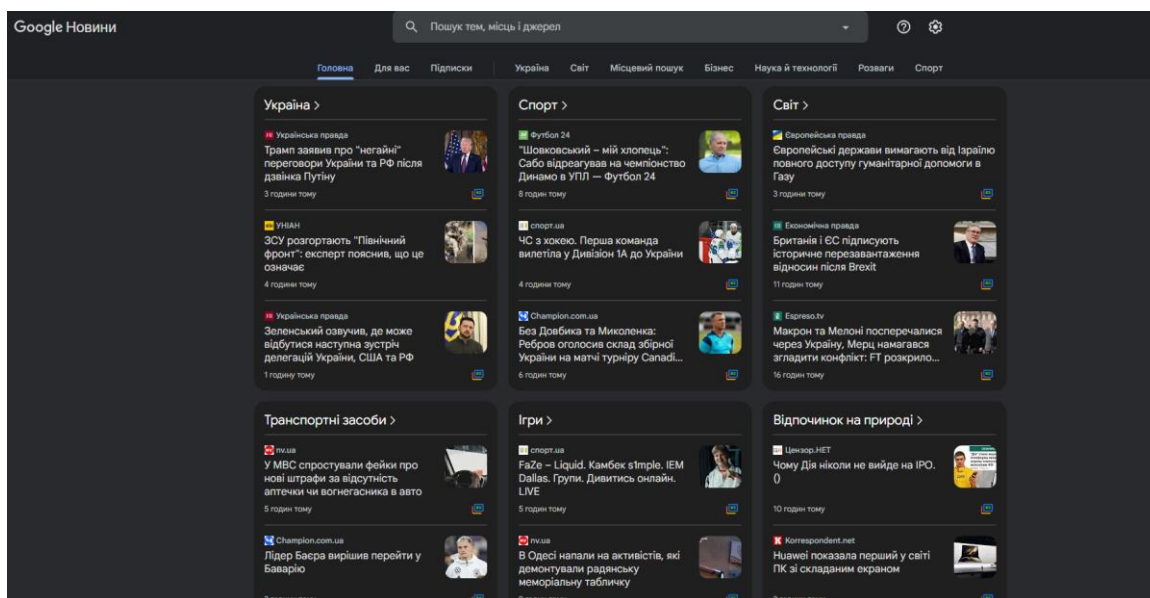


Рисунок 1.1 - Інформаційна система Google News

Джерело:[2]

**Flipboard** — це [8] інтерактивна платформа агрегування новин та контенту, яка функціонує як персоналізований цифровий журнал. Вона дозволяє користувачам отримувати актуальні новини, аналітичні матеріали, публікації з блогів та соціальних мереж у формі візуально привабливих добірок, що адаптуються відповідно до інтересів кожного користувача. З точки зору інформаційної управляючої системи, Flipboard реалізує інтелектуальні механізми контентної фільтрації, агрегування та рекомендації, поєднуючи алгоритмічну обробку даних з елементами ручного налаштування.

Платформа використовує складний аналітичний апарат, що включає машинне навчання, тематичну класифікацію та семантичний аналіз контенту. Вона здійснює моніторинг джерел з усього світу, а також дозволяє користувачам створювати власні «журнали», в які можна додавати вподобані статті. У системі функціонує алгоритм рекомендацій, який на основі дій користувача (вподобання, читання, підписки) формує динамічний профіль і персоналізує подальшу подачу інформації. Також реалізовані можливості push-сповіщень, що спрямовані на оперативне інформування про важливі новини або публікації за заданими темами.

Таким чином, Flipboard є прикладом інформаційної управляючої системи з розвиненою модульною архітектурою, яка забезпечує гнучке управління потоками інформації, адаптацію під інтереси користувача та підтримку інтерактивної взаємодії, що робить її ефективним інструментом автоматизації персонального інформування. Інформаційна система Flipboard наведена на рис. 1.2.

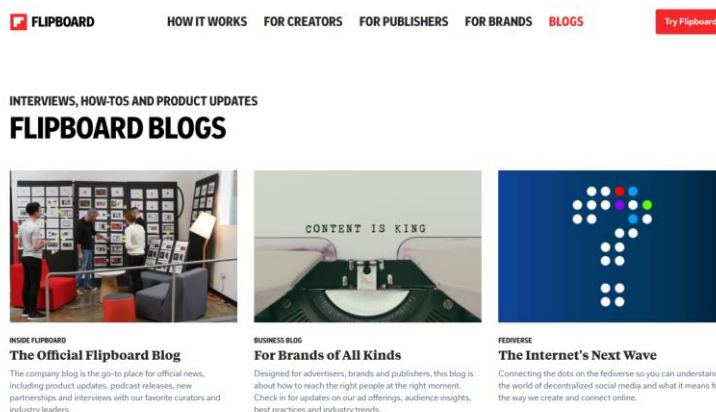


Рисунок 1.2 - Інформаційна система Flipboard

*Джерело: [8]*

Microsoft Start — це [9] персоналізована інформаційна платформа, розроблена компанією Microsoft для забезпечення користувачів актуальними новинами, прогнозами погоди, фінансовими зведеннями, порадами щодо способу життя та іншими даними, адаптованими до індивідуальних інтересів. Система є інтегрованою частиною екосистеми Microsoft і доступна на різних пристроях, зокрема у вигляді мобільного застосунку, веб-порталу, а також у вигляді віджета в Windows 10/11.

З позиції інформаційної управляючої системи, Microsoft Start реалізує складні алгоритми персоналізації, які ґрунтуються на аналізі дій користувача, його геолокації, історії переглядів і пріоритетів, заданих вручну. Механізм обробки контенту базується на технологіях машинного навчання та обробки природної мови (NLP), що дозволяє системі ефективно відбирати, категоризувати й рекомендувати новинні матеріали, підвищуючи релевантність інформаційного потоку.

Система також підтримує адаптивні push-нотифікації, спрямовані на оперативне сповіщення користувача про важливі події або зміни у визначених темах. Інтеграція з Microsoft Edge, Windows та іншими службами корпорації дозволяє формувати єдине середовище споживання та керування інформацією. Крім того, Microsoft Start використовує моделі модерації контенту, зокрема автоматичну та ручну фільтрацію, що забезпечує якість і безпечність інформації.

Таким чином, Microsoft Start є сучасною інформаційною управляючою системою, що поєднує інструменти агрегування, персоналізації та автоматизації нотифікацій, забезпечуючи високий рівень інформованості користувача в рамках цифрової екосистеми Microsoft. Інформаційна система Microsoft Start наведена на рис. 1.3.

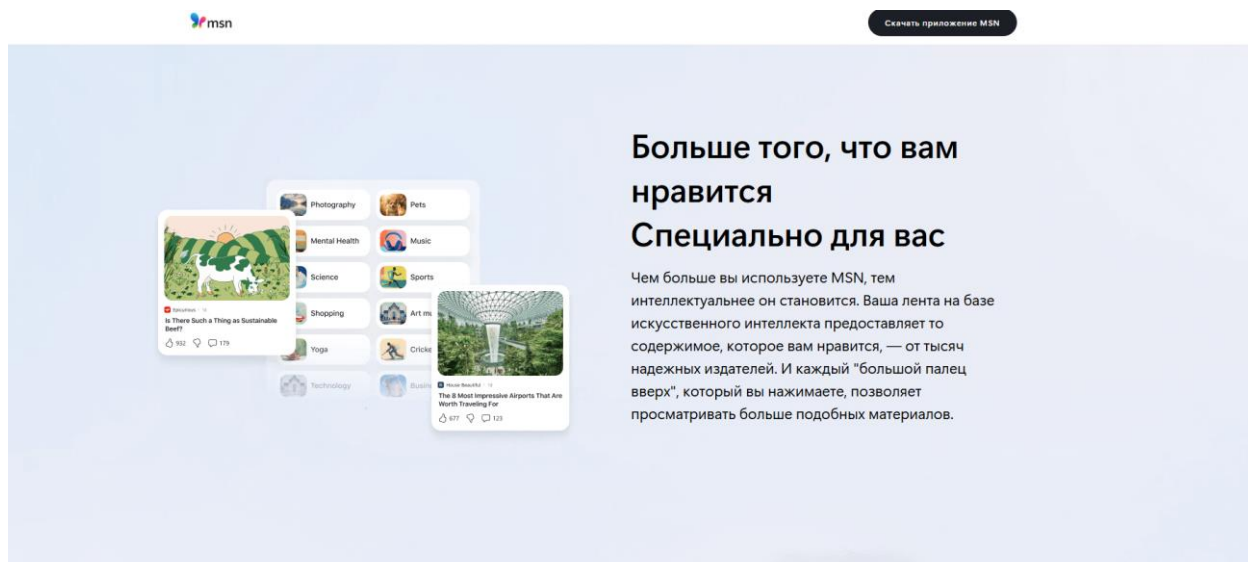


Рисунок 1.3 - Інформаційна система Microsoft Start

Джерело: [ 9]

Facebook News Feed — це центральний компонент соціальної мережі Facebook, який функціонує як динамічна інформаційна стрічка, призначена для демонстрації персоналізованого контенту кожному користувачеві. З наукової точки зору, Facebook News Feed можна розглядати як складну інформаційну управляючу систему, яка виконує функції збору, фільтрації, класифікації та дистрибуції даних в реальному часі.

Ця система базується на алгоритмах ранжування, що реалізують підходи машинного навчання, обробки великих даних і штучного інтелекту. Ключовим завданням є визначення пріоритетності контенту — новини, пости друзів, оновлення сторінок, рекламні повідомлення тощо — відповідно до передбачуваної зацікавленості конкретного користувача. Алгоритм News Feed враховує численні параметри: історію взаємодій користувача з контентом, тип контенту (відео, текст, зображення), час публікації, залученість інших користувачів, та навіть поведінкові маркери, які не є явно вираженими.

У контексті автоматизації нотифікацій, Facebook News Feed реалізує складну систему сповіщень, що включає push-нотифікації, оновлення в інтерфейсі, пріоритетні повідомлення та нагадування. Нотифікації створюються з урахуванням контекстної релевантності, часу доби, активності користувача та історії його

взаємодії з соціальною мережею. Уся ця система працює в режимі реального часу та масштабується на мільярди користувачів щодня.

Таким чином, Facebook News Feed є прикладом високоадаптивної інформаційної управляючої системи, здатної не лише агрегувати та персоналізувати контент, а й активно керувати увагою користувача через інтелектуальну систему рекомендацій і нотифікацій. Інформаційна система Facebook News Feed наведена на рис. 1.4.

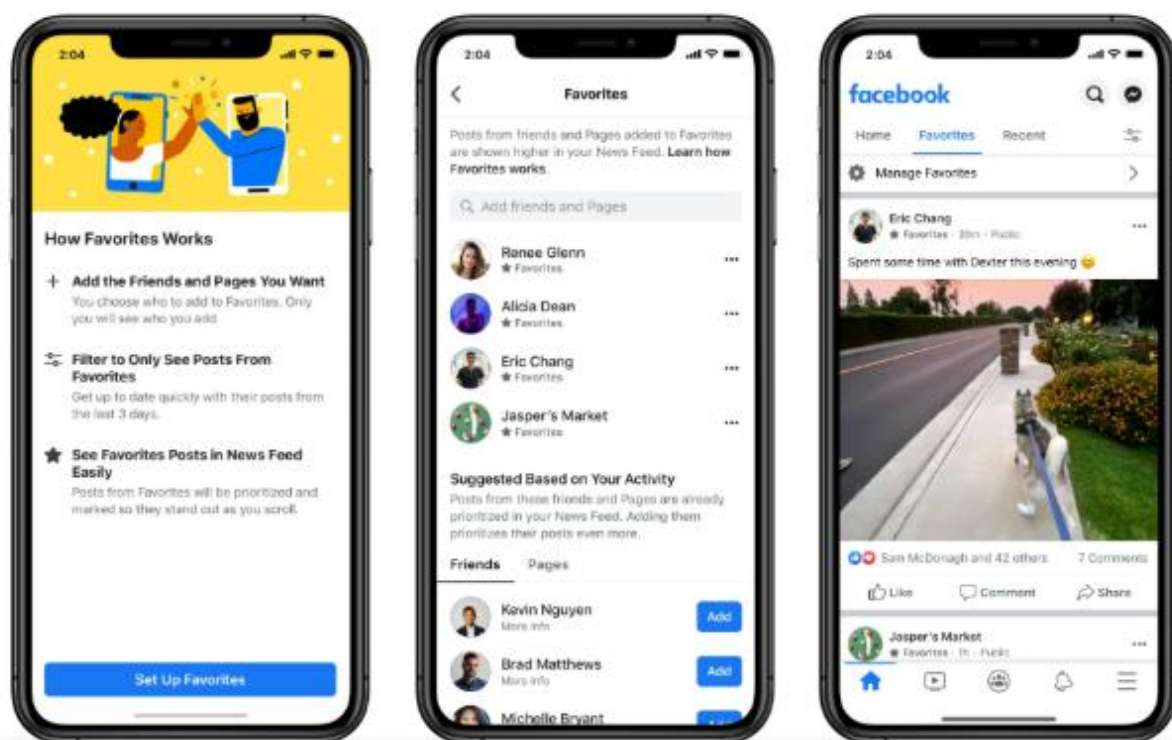


Рисунок 1.4 - Інформаційна система Facebook News Feed

*Джерело: [3]*

SmartNews [10] — це мобільна платформа агрегування новин, яка функціонує як інтелектуальна інформаційна управляюча система з автоматизованою подачею та фільтрацією актуального контенту. Розроблена в Японії, система орієнтована на користувачів, які бажають отримувати перевірені новини з різних авторитетних джерел без необхідності самостійного пошуку. SmartNews поєднує в собі принципи обробки великих даних, алгоритмічної класифікації та адаптивної персоналізації.

Ключовою особливістю цієї системи є те, що вона функціонує навіть за умов нестабільного або відсутнього інтернет-з'єднання, завдяки технології офлайн-кешування контенту [11]. Алгоритми SmartNews автоматично аналізують тисячі новинних статей щодня, класифікуючи їх за тематикою, джерелами, регіональними уподобаннями користувача, популярністю та емоційним забарвленням.

З погляду автоматизації процесу нотифікацій, платформа реалізує гнучку систему push-повідомлень, які враховують не лише інтереси користувача, а й ступінь актуальності інформації, події надзвичайного характеру або ті, що викликають суспільний резонанс. Це дозволяє ефективно інформувати користувача без перевантаження зайвими сповіщеннями, забезпечуючи високий рівень персоналізації.

Таким чином, SmartNews є прикладом сучасної інформаційної управляючої системи, яка поєднує автоматизоване агрегування, тематичну фільтрацію та нотифікацію новин, оптимізовану для зручності мобільного користувача, та може слугувати ефективною моделлю для розробки систем із подібною функціональністю. Інформаційна система SmartNews наведена на рис. 1.5.

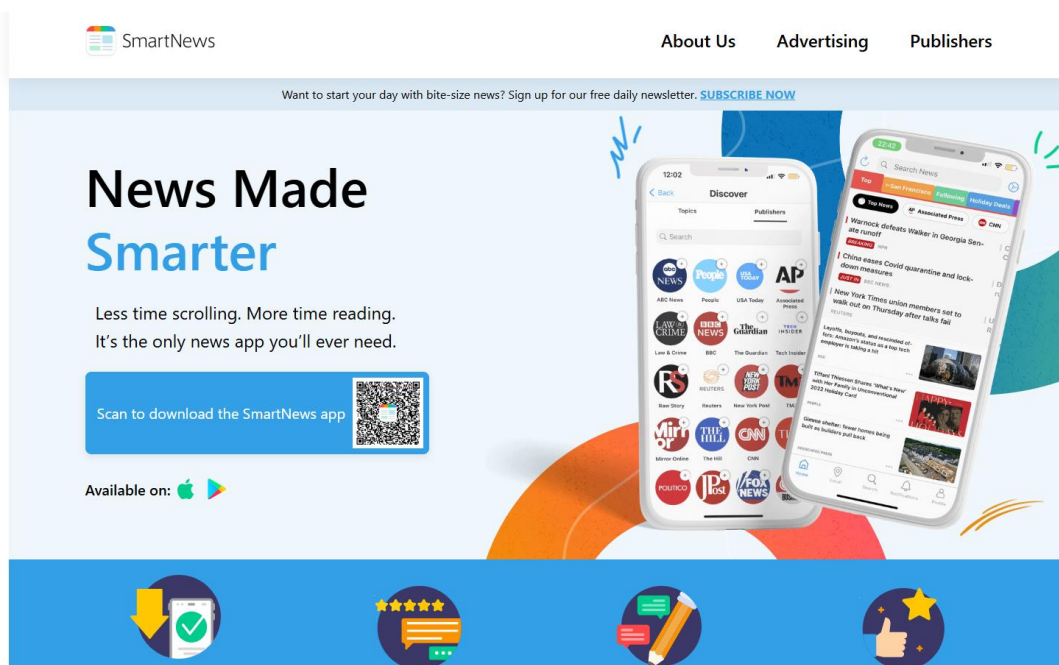


Рисунок 1.5 - Інформаційна система SmartNews

Джерело: [10]

### **1.3 ОБҐРУНТУВАННЯ ВИБОРУ ПІДХОДІВ І ТЕХНОЛОГІЙ ДЛЯ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ УПРАВЛЯЮЧОЇ СИСТЕМИ**

У цьому підрозділі буде представлено концептуальні рішення щодо розроблення інформаційної управляючої системи (ІУС) для автоматизації рекомендаційної стрічки новин з персоналізованими нотифікаціями. Основною метою є побудова ефективної, масштабованої, зручної для користувача системи, яка забезпечуватиме релевантну та своєчасну доставку інформації за інтересами користувача. Для цього були проаналізовані методологічні, технологічні та організаційні аспекти, що дозволили сформуванню цілісної концептуальної моделі запропонованого рішення.

Методологічно система проєктується відповідно до принципів мікросервісної архітектури з урахуванням концепцій Onion Architecture [12], що забезпечує модульність, ізолюваність компонентів та можливість гнучкого масштабування. Основна бізнес-логіка зосереджена в центральному ядрі системи, а зовнішні сервіси, такі як модулі сповіщень, аналітики, користувацького інтерфейсу тощо, взаємодіють через адаптери або API. Така побудова дозволяє чітко розділяти відповідальність між компонентами, зменшуючи зв'язність та підвищуючи підтримуваність.

У ролі технологічної платформи обрано мову програмування PHP [13] та фреймворк Laravel [14], що реалізує патерн MVC [15] і підтримує сучасні стандарти розробки, такі як SOLID-принципи [16], чистий код та вбудовані інструменти для реалізації безпечної автентифікації, управління сесіями, маршрутизації та тестування. Для організації даних використовується реляційна СКБД MySQL [17], яка добре підходить для обробки структурованої інформації, забезпечує підтримку транзакцій ACID та має широку підтримку в екосистемі Laravel. Архітектура бази даних базується на логічному моделюванні предметної області, з подальшою реалізацією фізичної моделі в рамках обраної СКБД.

Система передбачає наявність механізмів збору, фільтрації, зберігання та доставки новинного контенту з урахуванням персональних інтересів користувача. З цією метою використовуються алгоритми рекомендацій на основі ключових слів, категорій та історії взаємодій. В якості методу аутентифікації обрано сесійний підхід із використанням cookie, що забезпечує стабільне та безпечне управління станом користувача в системі.

Інтерфейс користувача реалізується у вигляді веб-застосунку з адаптивним дизайном, що дозволяє ефективно використовувати систему як на ПК, так і на мобільних пристроях. Основна увага при проєктуванні UI [18] приділяється зручності навігації, швидкому доступу до налаштувань та належному виведенню персоналізованих повідомлень.

Оцінювання альтернативних підходів до реалізації системи здійснювалося за низкою критеріїв: швидкодія, безпека, масштабованість, простота підтримки, вартість розгортання, доступність інструментів. За результатами аналізу, запропоноване рішення показало найкращий баланс між функціональністю, продуктивністю та гнучкістю адаптації до змінних вимог користувачів.

Таким чином, розроблена концептуальна модель інформаційної управляючої системи є обґрунтованою як з теоретичної, так і з практичної точки зору, забезпечує високий рівень відповідності вимогам предметної області, а також дає змогу уніфіковано та ефективно реалізувати автоматизований механізм рекомендаційних нотифікацій.

## **РОЗДІЛ 2 ХАРАКТЕРИСТИКА ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ТА МЕТОДИ І МОДЕЛІ**

В даному розділі йдеться мова про характеристику об'єкта дослідження, визначається структура ІУС, а також притаманні їй особливості, формується постановка задачі та обґрунтовується вибір методів і моделей ІУС.

### **2.1 Структура і характеристика системи**

Концептуальні рішення щодо створення інформаційної управляючої системи (ІУС) для автоматизації рекомендаційної стрічки новин сформовані на основі комплексного аналізу сучасних методів персоналізованої фільтрації інформації, моделей представлення користувача та контенту, а також практик впровадження подібних систем у цифровому середовищі. Визначальним принципом побудови системи є інтеграція гібридного підходу до формування рекомендацій, що передбачає поєднання контентно-орієнтованих алгоритмів з колаборативними фільтрами. Така концепція дозволяє не лише забезпечити релевантність новинного контенту відповідно до поточних інтересів користувача, а й оперативно адаптуватися до змін у його поведінкових характеристиках.

Однією з ключових концептуальних засад є використання структурованих користувацьких профілів, які формуються на основі історії взаємодії, контентних вподобань, часових преференцій та семантичного аналізу переглянутого матеріалу. Для ефективного збору і обробки даних застосовуються інструменти обробки природної мови (NLP), зокрема моделі тематичного моделювання та класифікації тексту. Особливу увагу приділено розробці моделі оновлення рекомендацій у реальному часі, що базується на механізмах онлайн-навчання і дозволяє системі залишатися актуальною в умовах швидкоплинного новинного потоку.

З архітектурної точки зору, ІУС реалізується як модульна система з чітко визначеними функціональними блоками: модуль збору даних, модуль обробки інформації, аналітичний модуль формування рекомендацій, керуючий модуль та модуль представлення результатів. Внутрішня взаємодія між модулями здійснюється за принципами сервісно-орієнтованої архітектури (SOA) [19], що забезпечує масштабованість, адаптивність та можливість інтеграції з іншими інформаційними платформами.

Таким чином, запропоновані концептуальні рішення враховують як теоретичні напрацювання, так і вимоги сучасної практики розробки інформаційних систем, що дозволяє створити ефективну ІУС для персоналізованого інформаційного супроводу користувача в цифровому новинному середовищі.

У межах розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин особливу увагу приділено побудові моделей структури та поведінки системи, які є базовими для забезпечення її функціональної повноти, логічної цілісності та адаптивності. Структурна модель системи описує її внутрішню архітектуру, розподіл функціональних компонентів та взаємозв'язки між ними. Вона включає такі ключові підсистеми: модуль збору та попередньої обробки даних, аналітичний модуль для формування профілю користувача та метаданих новин, модуль генерації рекомендацій на основі визначених релевантних характеристик, а також керуючий модуль, який здійснює контроль за логікою обміну інформацією між складовими системи.

Модель поведінки системи базується на принципах динамічної реакції на зовнішні події, такі як оновлення новинного контенту, зміни у поведінці користувача, зворотний зв'язок тощо. Поведінкову логіку описано за допомогою сценаріїв взаємодії користувача із системою, діаграм послідовності та схем переходів між станами. Наприклад, у процесі першого використання система здійснює ініціалізацію користувацького профілю, а надалі — автоматично оновлює його залежно від кліків, переглядів і часової активності. Система також передбачає елементи самонавчання, що реалізуються через збирання зворотного зв'язку та коригування вагових коефіцієнтів у алгоритмах оцінювання релевантності.

Завдяки поєднанню моделей структури та поведінки, забезпечується цілісне функціонування інформаційної управляючої системи, її гнучка реакція на зміни в середовищі та здатність до самостійного вдосконалення на основі накопичених даних. Це дозволяє системі не лише виконувати свої функції у статичному режимі, а й розвиватися відповідно до нових інформаційних запитів користувачів і динаміки новинного простору.

Поведінкова модель інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин може бути формалізована за допомогою нотації UML [20] (Unified Modeling Language), яка є загальновизнаним стандартом для візуального моделювання програмних систем. Серед діаграм, що найбільш ефективно відображають динаміку та логіку роботи системи, ключову роль відіграють діаграма діяльності (Activity Diagram) [21] та діаграма станів (State Diagram) [22].

Діаграма діяльності використовується для опису послідовності дій, які виконує система або користувач у певному процесі. У контексті рекомендаційної системи така діаграма дозволяє відобразити типовий сценарій взаємодії — наприклад, від моменту отримання новинного потоку до генерації та виведення персоналізованої стрічки. Тут можна чітко побачити етапи: ініціалізація користувача, обробка новинного контенту, аналіз профілю, застосування алгоритмів рекомендацій, виведення результатів і збір зворотного зв'язку. Діаграма діяльності надає змогу виділити паралельні та умовні гілки виконання процесу, що є критичним для складних інформаційних систем із високим рівнем інтерактивності.

Діаграма станів, у свою чергу, фокусується на внутрішніх змінах стану системи або її компонентів у відповідь на зовнішні події. Вона дозволяє моделювати поведінку об'єкта (наприклад, користувача або профілю користувача) з урахуванням різних сценаріїв розвитку подій. Наприклад, профіль користувача може перебувати у станах "новий", "частково навчений", "адаптований", і залежно від вхідних дій — переходити між цими станами. Така діаграма є надзвичайно корисною для моделювання реактивних процесів — тобто тих, що залежать від

зовнішнього впливу, що часто трапляється в інформаційних системах реального часу.

Формалізація поведінки системи за допомогою UML-діаграм не лише покращує розуміння логіки функціонування системи для розробників, але й сприяє виявленню неузгодженостей, дублювання функцій або пропущених переходів. Крім того, така документація є важливою складовою технічного супроводу та подальшої модернізації інформаційної системи.

Побудова моделей структури та поведінки системи є критично важливою складовою етапу наукового обґрунтування і проектування інформаційної управлінської системи автоматизації нотифікацій.

Для детального опису роботи окремих підсистем інформаційної управляючої системи, що автоматизує рекомендаційну стрічку новин, доцільно застосовувати UML-діаграми прецедентів, послідовності та діяльності, які дозволяють наочно представити функціональні можливості, взаємодію між компонентами та логіку виконання задач на рівні підсистеми.

Діаграма прецедентів (Use Case Diagram) [23] фіксує основні сценарії використання підсистеми з точки зору зовнішнього користувача або іншої системи, що взаємодіє з нею. Наприклад, у межах підсистеми формування рекомендацій можуть бути виділені прецеденти: "Отримати дані про користувача", "Аналізувати новинний потік", "Генерувати персоналізовану стрічку", "Зберегти зворотний зв'язок". Актором тут виступає користувач або адміністратор системи. Така діаграма дозволяє визначити, які саме функції повинна реалізовувати підсистема з погляду кінцевої взаємодії.

Діаграма послідовності (Sequence Diagram) [24] деталізує процеси, що відбуваються під час виконання певного прецеденту. Вона відображає обмін повідомленнями між об'єктами системи у хронологічному порядку. Наприклад, для задачі "Генерація рекомендацій" діаграма покаже, як ініціатор (користувач) надсилає запит до модуля аналізу профілю, далі — до модуля оцінки релевантності новин, після чого формується відповідь через модуль виводу стрічки. Така модель

ілюструє чітку послідовність викликів функцій і передачі даних, що критично важливо для налаштування ефективної взаємодії підсистем.

Діаграма діяльності (Activity Diagram) [25] дає змогу змодельовати алгоритм виконання задачі у вигляді потоку дій, який може включати паралельні, умовні або циклічні гілки. Наприклад, під час обробки даних користувача діаграма показує: початок сеансу, перевірку наявного профілю, вилучення поведінкових характеристик, запуск механізмів аналізу, оновлення профілю — до завершення обробки. Така модель дозволяє легко ідентифікувати етапи, що можуть бути оптимізовані або автоматизовані.

У межах розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин особливу роль відіграє формалізація процесів шляхом побудови UML-діаграм, які дозволяють структурувати логіку функціонування підсистем, визначити сценарії взаємодії з користувачем, а також відобразити динаміку поведінки системи. Нижче наведено короткі науково обґрунтовані описи основних типів діаграм, які використовуються для моделювання підсистеми генерації персоналізованих новинних рекомендацій.

Діаграма прецедентів відображає зовнішні функціональні сценарії використання системи з позиції основних акторів. У контексті підсистеми генерації рекомендацій визначено наступні прецеденти: ініціалізація запиту на формування стрічки новин, аналіз профілю користувача, отримання актуальних новинних матеріалів, побудова персоналізованої стрічки та реєстрація зворотного зв'язку. Головним актором є користувач системи, а також передбачено можливість взаємодії із зовнішніми службами аналітики, що можуть автоматично ініціювати оновлення рекомендаційного потоку.

Діаграма послідовності описує часову послідовність обміну повідомленнями між основними об'єктами системи при реалізації функціонального сценарію. Зокрема, сценарій "Формування персоналізованої стрічки новин" включає взаємодію користувача з інтерфейсом системи, який, у свою чергу, надсилає запити до модуля аналізу профілю, служби новин та модуля генерації рекомендацій. Після завершення обробки результати передаються назад через

інтерфейс, а дані про взаємодію з користувачем зберігаються у базі для подальшого аналізу.

Діаграма діяльності [26] формалізує логіку реалізації окремого бізнес-процесу в межах системи у вигляді послідовності дій. Так, процес побудови рекомендаційної стрічки включає такі кроки: початкова ініціалізація процесу, отримання даних профілю користувача, аналіз активності, відбір релевантних новин, обчислення індивідуальних оцінок важливості матеріалів, побудова структури стрічки, виведення результату та збереження зворотного зв'язку. Кожен з етапів може бути реалізований як окрема підпрограма або підсистема в архітектурі ІУС.

Застосування зазначених типів UML-діаграм забезпечує цілісне бачення внутрішньої логіки функціонування системи, дозволяє формалізувати вимоги до реалізації функцій, мінімізувати ризики логічних помилок під час розробки та забезпечити ефективне документування проектного рішення.

У контексті побудови структурної моделі підсистеми рекомендаційної стрічки новин інформаційної управляючої системи, діаграми класів UML відіграють концептуально важливу роль для опису статичних аспектів системи. Вони забезпечують формалізоване подання взаємопов'язаних елементів об'єктно-орієнтованої архітектури, де кожен клас виступає абстракцією сутностей предметної області, а зв'язки між класами відображають семантику їх взаємодії у межах функціональної логіки системи.

У типових структурних моделях для підсистеми персоналізованих рекомендацій доцільно виокремити кілька пакетів: пакет управління користувачем (UserManagement), пакет новинного контенту (NewsRepository), рекомендаційне ядро (RecommendationCore) та модуль зворотного зв'язку (FeedbackProcessor). Кожен з цих пакетів об'єднує підмножини класів, що реалізують функціональні компоненти відповідної підсистеми. Наприклад, у пакеті RecommendationCore можуть бути представлені класи RecommenderEngine, NewsRanker, FilteringStrategy, які взаємодіють відповідно до обраної моделі рекомендації (контентно-орієнтованої, колаборативної чи гібридної).

Подальше уточнення класів відбувається шляхом моделювання атрибутів та методів, що дозволяє встановити логіку доступу до даних, алгоритмічну поведінку об'єктів та чітко визначити інтерфейси між різними модулями системи. Така деталізація є критично важливою для подальшого процесу імплементації, тестування та супроводження програмного забезпечення, що створюється. Дотримання принципів інкапсуляції, узагальнення та повторного використання коду, які безпосередньо реалізуються у структурі класів, є запорукою ефективного функціонування складної інформаційної системи у реальних умовах.

У процесі побудови структурної моделі підсистеми автоматизації нотифікацій важливим є виділення основних стереотипів класів, які відіграють специфічні ролі в архітектурі інформаційної системи. До таких стереотипів належать:

У рамках побудови діаграм класів UML для інформаційної управляючої системи автоматизації рекомендаційної стрічки новин доцільно використовувати основні стереотипи класів, що забезпечують структуроване й узагальнене подання ключових компонентів системи з урахуванням архітектурної ролі кожного елемента. До таких стереотипів належать класи-сутності, граничні класи та класи керування (менеджери), які разом формують архетипову модель об'єктно-орієнтованого проектування згідно з підходом RUP (Rational Unified Process) [27] або аналогічними методологіями.

Класи-сутності у контексті цієї системи відображають об'єкти предметної області, які зберігають інформацію протягом тривалого часу і відтворюють основні структури даних. Наприклад, такими класами є `UserProfile`, `NewsItem`, `Category`, `FeedbackRecord`, які репрезентують зареєстрованих користувачів, новинні публікації, тематичні категорії та історію взаємодії користувачів відповідно. Вони мають сталі атрибути (ID, назви, часові мітки) і методи доступу до цих даних.

Граничні класи (`<<boundary>>`) призначені для моделювання точок взаємодії користувача з системою, тобто реалізації інтерфейсів введення та виведення. У межах рекомендаційної системи такими класами можуть виступати `NewsFeedUI`, `SearchInterface`, `UserAuthPanel`, які відповідають за візуалізацію

персоналізованої стрічки, пошуковий механізм та процес автентифікації користувача. Ці класи, як правило, взаємодіють із класами-менеджерами, передаючи їм вхідні дані та приймаючи оброблену інформацію для відображення.

Класи керування (<<control>>) виконують роль посередника, що координує дії між класами-сутностями та граничними класами. Вони втілюють логіку одного або кількох прецедентів використання і відповідають за послідовність виконання подій у межах цих сценаріїв. До прикладу, клас RecommendationManager обробляє запити інтерфейсу, ініціює отримання релевантних новин через RecommenderEngine, фільтрує результати згідно з профілем користувача та передає їх до інтерфейсу. Інші керувальні класи, як-от UserSessionController або FeedbackAnalyzer, можуть регулювати динаміку персоналізації або забезпечення адаптивності стрічки до змін у поведінці користувача.

Отже, узагальнюючи результати проєктування структурної моделі підсистеми інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин, можна зробити висновок, що застосування діаграм класів UML із чітко визначеними стереотипами дозволяє ефективно відобразити логіку побудови системи, її основні компоненти та взаємозв'язки між ними. Виокремлення класів-сутностей, граничних класів та класів керування сприяє чіткому розмежуванню функціональних обов'язків об'єктів системи, підвищує прозорість архітектури та забезпечує передумови для масштабованості і супроводжуваності розробленого рішення. Такий підхід є обґрунтованим з позицій сучасного об'єктно-орієнтованого аналізу та дозволяє створити надійну та адаптивну інформаційну систему, здатну до ефективної роботи в умовах динамічного змінного новинного середовища.

## **2.2 Методи та моделі в інформаційних управляючих системах і технологіях**

У підрозділі, присвяченому методам та моделям в інформаційних управляючих системах і технологіях, розглядаються концептуальні та прикладні

підходи, які забезпечують функціональність, адаптивність та ефективність систем управління інформаційними потоками, зокрема у контексті автоматизації рекомендаційної стрічки новин. Методи в таких системах поділяються на загальносистемні, аналітичні, алгоритмічні та інтерактивні, кожен з яких відіграє окрему роль у побудові й реалізації логіки системи.

З методологічної точки зору, найбільш поширеними є методи штучного інтелекту, машинного навчання та обробки природної мови, які дозволяють автоматично аналізувати великі обсяги текстової інформації, класифікувати новинні повідомлення, моделювати поведінку користувача й формувати персоналізовані рекомендації. Такі моделі включають байєсівські мережі [28], дерева рішень, кластеризацію та нейронні мережі (зокрема трансформери та рекурентні архітектури). Моделі колаборативної фільтрації дозволяють визначити подібність між користувачами або новинами на основі попередніх взаємодій, у той час як контентно-орієнтовані моделі опираються на семантичний аналіз самого новинного матеріалу.

Інформаційні управляючі системи також використовують формальні моделі, зокрема моделі автоматів, моделі потоків даних (DFD) [29] та ієрархічні моделі процесів (IDEF0) [30], які дозволяють формалізувати поведінку, зв'язки між компонентами системи та трансформацію вхідних даних у результати. Для забезпечення адаптивності й самонавчання системи в умовах динамічного новинного середовища, доцільно також використовувати онтологічні моделі знань, які підтримують семантичну узгодженість інформації.

Таким чином, сучасні інформаційні управляючі системи базуються на поєднанні формальних, статистичних та інтелектуальних методів і моделей, що забезпечує можливість реалізації гнучких і масштабованих технологічних рішень. Вибір відповідних моделей обумовлений вимогами до продуктивності, точності рекомендацій, швидкості адаптації системи до змін і можливості масштабування під час розширення користувацької бази або обсягу оброблюваних новин.

Складність завдань у технічній, фінансово-кредитній та соціально-економічній сферах діяльності безперервно зростає внаслідок динамічного розвитку інформаційних технологій, зростання обсягів даних, необхідності прийняття рішень у режимі реального часу та підвищених вимог до точності, адаптивності та індивідуалізації обслуговування. У таких умовах традиційні підходи до обробки інформації та управління процесами стають недостатньо ефективними, що зумовлює потребу у впровадженні нових методів, заснованих на інтеграції інтелектуальних інформаційних систем.

Особливо актуальними є вимоги до засобів вирішення прикладних завдань, які передбачають гнучке налаштування систем під конкретні умови функціонування, здатність до самоорганізації, адаптації до змін середовища, автоматичної обробки неструктурованих даних та високої стійкості до інформаційного шуму. Це вимагає використання складних алгоритмічних конструкцій, машинного навчання, систем підтримки прийняття рішень, гібридних моделей аналізу, що поєднують статистичні й евристичні методи.

В контексті розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин особливої ваги набувають вимоги до швидкодії, масштабованості, персоналізації контенту, здатності системи адаптуватися до поведінки користувачів та змін у зовнішньому інформаційному середовищі. Сучасні ІУС повинні не лише забезпечувати оперативний облік та аналіз даних, а й здійснювати прогностичне моделювання, контекстуальну обробку інформації та багатокритеріальну оптимізацію рекомендацій, що уможливорює прийняття обґрунтованих рішень у складних умовах невизначеності.

Таким чином, загальний вектор розвитку ІУС відповідає актуальним потребам цифрової трансформації суспільства й економіки та орієнтується на побудову складних програмно-апаратних рішень, здатних виконувати високорівневі функції аналізу, прогнозування та автоматизованого управління інформаційними потоками.

У сучасних умовах переважна більшість завдань, які розв'язуються в рамках інформаційних управляючих систем, належать до класу багатокритеріальних та багатоцільових задач прийняття управлінських рішень. Такий тип задач характеризується необхідністю одночасного врахування і узгодження низки цілей, які можуть бути як взаємодоповнюючими, так і взаємовиключними. Це створює додаткову складність у процесі прийняття рішень, оскільки досягнення однієї мети часто передбачає компроміс щодо іншої, а обґрунтований вибір оптимального варіанта потребує комплексного аналізу та пріоритетизації численних критеріїв.

Особливістю таких задач є не лише велика кількість параметрів, що впливають на процес прийняття рішень, а й необхідність урахування взаємозв'язків між ними, нечіткості або неповноти початкової інформації, високого рівня динамічності середовища та швидкоплинності змін. Це зумовлює потребу у використанні відповідних моделей і методів, здатних працювати в умовах багатокритеріальності — зокрема методів нечіткої логіки, теорії корисності, багатокритеріального аналізу альтернатив, еволюційного моделювання, штучних нейронних мереж тощо.

У випадку автоматизації рекомендаційної стрічки новин в інформаційній управляючій системі багатокритеріальність виявляється у необхідності одночасного забезпечення релевантності новин, персоналізації рекомендацій, дотримання етичних стандартів, врахування контексту користувача, забезпечення різноманітності контенту та мінімізації інформаційного шуму. Це вимагає інтегрованого підходу до моделювання системи, в якому комбінуються аналітичні, евристичні та навчальні компоненти для забезпечення якісного й ефективного управління інформаційним потоком.

У контексті розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин багатокритеріальність та багатоцільовість завдань проявляється на всіх етапах функціонування системи — від збору та класифікації інформації до формування остаточного набору рекомендацій для користувача. Основна складність полягає у тому, що система повинна не лише забезпечувати оперативне постачання контенту, а й динамічно

адаптувати його відповідно до множини параметрів: інформаційних уподобань користувача, темпоральної актуальності новин, рівня достовірності джерел, контексту взаємодії (мобільний пристрій, час доби, геолокація), а також системних обмежень, пов'язаних із продуктивністю, захистом персональних даних та уникненням інформаційного перенавантаження.

У реалізованій системі ці аспекти враховані через впровадження модульної багаторівневої архітектури [31], де кожен підмодуль відповідає за певний критерій або підзадачу. Наприклад, модуль обробки поведінкових патернів користувача реалізує механізми машинного навчання на основі історичних даних, що дозволяє формувати прогностичні моделі уподобань. Для цього застосовується, зокрема, модель градієнтного бустингу або рекурентні нейронні мережі (наприклад, LSTM), які дозволяють враховувати часову динаміку споживання новин.

Ще одним прикладом є інтеграція методу аналітичної ієрархічної обробки (АНР) [32] для вагового оцінювання важливості критеріїв при формуванні списку рекомендацій. Система за допомогою експертно-обґрунтованої моделі АНР здійснює пріоритетизацію факторів: наприклад, достовірність джерела може отримати вищу вагу у новинній стрічці, орієнтованій на професійну аудиторію, тоді як для соціального сегменту — більшу роль може відігравати візуальний стиль або емоційна тональність повідомлення.

Крім того, для забезпечення балансу між релевантністю й різноманітністю контенту в системі реалізовано механізм диверсифікації на основі кластеризації новинних тем за допомогою методу *k*-середніх або Latent Dirichlet Allocation (LDA). Це дозволяє уникнути ефекту «інформаційної бульбашки», забезпечуючи користувача матеріалами, які виходять за межі його звичних інтересів, але потенційно можуть бути цінними.

Таким чином, багатокритеріальний підхід втілюється у системі не лише на рівні алгоритмів відбору інформації, а й у структурних компонентах, що забезпечують її адаптивність, гнучкість і масштабованість. Завдяки цьому досягається комплексна реалізація цілей, що часто є конфліктними: швидкість — проти глибини аналізу, персоналізація — проти узагальнення, актуальність —

проти довготривалої цінності контенту. Застосування зазначених моделей дозволяє вирішити зазначені суперечності в оптимальний спосіб.

У процесі функціонування інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин особливу складність становить ухвалення рішень в умовах як інформаційного дефіциту, так і надлишковості даних. У першому випадку система може стикатися з обмеженістю вхідних параметрів щодо нових користувачів (проблема "холодного старту") або щойно опублікованих новин, які ще не мають взаємодій чи відгуків. У другому випадку виникає інша крайність — перенасичення потоку інформації, серед якої складно ідентифікувати достовірну, релевантну та корисну для конкретного користувача інформацію.

Для подолання дефіциту даних застосовуються гібридні підходи, що поєднують колаборативну фільтрацію з контент-орієнтованими методами, а також використовуються демографічні та поведінкові евристичні методи, які дозволяють побудувати базову модель рекомендацій навіть за відсутності історії активності. Також, із метою оперативного реагування на появу нових джерел або публікацій, застосовується аналіз метаданих і тематична класифікація текстів із використанням моделей трансформерів (наприклад, BERT) [33].

Щодо надлишковості даних, то тут ключовим є впровадження механізмів семантичного фільтрування, автоматичного виявлення фейкових повідомлень (fake news detection) та алгоритмів очищення інформаційного потоку. Це реалізується за допомогою машинного навчання, систем розпізнавання надійності джерел і міждисциплінарного аналізу текстового контенту (з урахуванням стилістичних, емоційних, статистичних та логічних характеристик). Застосування таких підходів забезпечує відбір даних, які є релевантними, коректними та мають високу ймовірність достовірності.

Таким чином, здатність інформаційної системи ефективно функціонувати в умовах як браку, так і надлишку даних визначає її адаптивність, гнучкість і антикризовий потенціал, що критично важливо для динамічного середовища новинного контенту.

У сучасних інформаційно-аналітичних системах, зокрема в розробленій інформаційній управляючій системі для автоматизації рекомендаційної стрічки новин, значна частина задач прийняття рішень має слабоструктурований або навіть неструктурований характер. Це означає, що для таких задач важко або неможливо сформулювати чіткі алгоритми, визначити всі вхідні параметри та побудувати однозначну логічну модель прийняття рішень. Як правило, ці задачі відзначаються високим рівнем невизначеності, варіативності та залежності від зовнішнього контексту, зокрема соціальних, політичних або часових чинників.

Слабоформалізовані задачі характерні тим, що в них відсутній повний опис залежностей між вхідними даними і результатами, що унеможливорює використання традиційних детермінованих моделей. Наприклад, визначення того, які новини будуть найбільш релевантними для користувача з урахуванням його поточного емоційного стану, світоглядних уподобань або неявних поведінкових патернів, вимагає аналізу нефіксованих і складноформалізованих ознак.

Для розв'язання подібних задач у межах розробленої ІУС застосовуються сучасні інструменти штучного інтелекту: нейронні мережі, нечіткі логічні моделі (fuzzy logic), байєсівські мережі, алгоритми підкріплювального навчання. Їх використання дозволяє здійснювати наближене моделювання прийняття рішень, враховуючи численні нечітко визначені фактори та взаємозв'язки. Крім того, ітеративний характер функціонування таких систем дозволяє адаптувати моделі на основі нових даних, що постійно надходять у систему, і поступово вдосконалювати точність прогнозування.

Отже, врахування слабоструктурованої природи задач у розробці ІУС забезпечує її релевантність до реального інформаційного середовища, підвищує адаптивність до складних управлінських ситуацій та розширює можливості системи у вирішенні інтелектуальних завдань.

Подолання викликів, пов'язаних із слабоформалізованими, багатокритеріальними та невизначеними задачами, стає можливим завдяки впровадженню в інформаційні управляючі системи адекватного математичного

апарату, зокрема економіко-математичних методів і моделей. Ці інструменти дозволяють формалізувати складні управлінські ситуації, здійснювати багатфакторний аналіз, моделювати сценарії розвитку подій та знаходити оптимальні рішення серед великої кількості альтернатив.

У контексті розробленої системи автоматизації рекомендаційної стрічки новин, математичні моделі, що реалізовані в окремих модулях ІУС, забезпечують підтримку прийняття рішень у реальному часі, обробку великих обсягів вхідної інформації, а також аналіз динаміки інформаційного середовища. Зокрема, доцільним є використання методів багатокритеріальної оптимізації, кластерного аналізу, стохастичних моделей, байєсівських мереж і елементів теорії прийняття рішень.

Удосконалення аналітичної складової системи можливе шляхом інтеграції компонентів штучного інтелекту — нейромереж, моделей машинного навчання, алгоритмів глибинного аналізу даних, які здатні виявляти приховані закономірності, адаптуватися до змін у поведінці користувачів і навчатися на основі історичних даних. Такий підхід дозволяє не лише покращити якість рекомендацій, але й забезпечити гнучкість та інтелектуальність системи в умовах інформаційної невизначеності.

Таким чином, впровадження математичних методів і штучного інтелекту в структуру ІУС є ключовим напрямом підвищення ефективності управлінських рішень, що формуються в умовах складного, динамічного інформаційного середовища.

Узагальнюючи розглянуті аспекти, можна зробити висновок, що сучасні інформаційні управляючі системи повинні володіти здатністю до гнучкої обробки даних, адаптації до зміни умов і підтримки прийняття управлінських рішень у складних, невизначених і багатокритеріальних середовищах. Зростаюча складність завдань у технічних, економічних і соціальних сферах зумовлює необхідність використання потужного математичного апарату, що поєднує економіко-математичне моделювання, штучний інтелект та інструменти аналізу даних. Особливої актуальності набуває впровадження інтелектуальних модулів у

структуру ІУС, які дозволяють ефективно вирішувати слабоструктуровані задачі, працювати з неповною або надлишковою інформацією та підвищувати якість управлінських рішень. Використання таких підходів у розробленій системі автоматизації рекомендаційної стрічки новин забезпечує її релевантність, адаптивність і практичну значущість у сучасному інформаційному просторі.

## **РОЗДІЛ 3 РОЗРОБЛЕННЯ ПРОЄКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ**

Даний розділ присвячений детальному опису практичного етапу створення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин. У цьому розділі представлено концепцію побудови архітектури системи, обґрунтовано вибір технологій і програмних засобів, описано процеси розробки окремих функціональних компонентів, їх взаємодію та інтеграцію в єдине програмне середовище. Особливу увагу приділено реалізації ключових алгоритмів, що забезпечують інтелектуальний аналіз даних та формування персоналізованих рекомендацій. Окремо розглянуто питання програмної реалізації системи, тестування функціоналу, а також умови впровадження та практичного використання проєкту. Такий підхід дозволяє не лише забезпечити відповідність поставленим вимогам, але й створити ефективний інструмент для роботи з великими обсягами інформації в умовах сучасного медіапростору.

### **3.1. Проєктування бази даних та/або сховища даних для інформаційної управляючої системи**

Даний підрозділ присвячено проєктуванню бази даних як фундаментального елементу інформаційної управляючої системи, що забезпечує збереження, структуризацію та ефективний доступ до інформації, необхідної для функціонування автоматизованої рекомендаційної стрічки новин. У рамках цього підрозділу розглядається логічне і фізичне моделювання бази даних, зокрема побудова ER-діаграм, визначення основних сутностей, атрибутів і зв'язків між ними. Обґрунтовується вибір моделі даних, типу СУБД та особливості реалізації схеми бази відповідно до вимог продуктивності, масштабованості, безпеки й підтримки цілісності інформації. Описано структуру таблиць, механізми індексації,

нормалізацію даних, засоби забезпечення резервного копіювання та відновлення. Особливу увагу приділено інтеграції бази даних з модулями обробки користувацьких даних і генерації рекомендацій, а також реалізації запитів, що забезпечують персоналізацію новинного контенту в режимі реального часу. Розроблена структура бази даних відповідає специфіці предметної області, дозволяє ефективно підтримувати функціонування системи в умовах великої кількості користувачів і високої динаміки інформаційного потоку.

### **3.1.1 Проєктування інфологічної (логічної) моделі БД**

У межах підпункту, присвяченого проєктуванню інфологічної (логічної) моделі бази даних, здійснено побудову концептуального представлення інформаційної структури системи, яке відображає основні об'єкти предметної області, їхні атрибути та взаємозв'язки. Інфологічна модель побудована у вигляді ER-діаграми (діаграми “сутність–зв'язок”), яка є узагальненим логічним описом інформації, що має зберігатися у базі даних інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин.

У моделі виокремлено ключові сутності, зокрема: Користувач, Новина, Категорія, Рекомендація, Перегляд, Оцінка, Теги. Сутність Користувач описується атрибутами: ідентифікатор, ім'я, електронна пошта, дата реєстрації, тип користувача тощо.

Сутність Новина включає атрибути: ідентифікатор, заголовок, зміст, дата публікації, джерело. Категорія містить ієрархічну структуру для класифікації новин (наприклад, політика, технології, спорт). Рекомендація описує взаємозв'язок між користувачем і новинним контентом, що був йому запропонований, із фіксацією часу, джерела рекомендації та показника релевантності. Зв'язки між сутностями реалізуються через відношення “один-до-багатьох” (наприклад, один користувач має багато переглядів новин) або “багато-до-багатьох” (наприклад,

новина може мати багато тегів, а тег застосовуватися до багатьох новин), які формалізуються через допоміжні сутності.

Побудована логічна модель відповідає вимогам нормалізації (до третьої нормальної форми включно), що забезпечує уникнення надмірності, підтримку цілісності та ефективність виконання запитів. Інфологічна модель стала основою для наступного етапу – фізичного проєктування бази даних і створення відповідних таблиць у середовищі реалізації.

Сутність інфологічного проєктування [34] полягає у формалізованому відображенні структури предметної області шляхом ідентифікації ключових інформаційних об'єктів (сутностей), їхніх атрибутів та взаємозв'язків між ними, що мають бути відображені у базі даних. Цей етап проєктування спрямований на формування абстрактної, незалежної від конкретної СКБД моделі, яка максимально точно відображає логіку функціонування інформаційної системи. У контексті створення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин визначаються сутності на кшталт Користувач, Новина, Категорія, Рекомендація, Перегляд, які репрезентують основні об'єкти системної взаємодії.

Результатом інфологічного моделювання є інформаційно-логічна модель предметної області, що представлена у вигляді ER-діаграми, а також текстовий опис кожної з її складових: визначення сутностей, їхніх атрибутів, первинних та зовнішніх ключів, типів зв'язків (один-до-багатьох, багато-до-багатьох), обмежень цілісності. Така модель є основою для переходу до даталогічного рівня проєктування, де визначається структура таблиць бази даних у межах обраної системи керування базами даних. Важливо, що інфологічна модель повинна бути узгодженою з вимогами кінцевих користувачів та функціональними потребами прикладних програм, що забезпечує її адекватність і функціональну завершеність у рамках системи.

ACID — це аббревіатура, що позначає чотири основні вимоги до транзакцій у системах керування базами даних [35] (СКБД), які гарантують надійність, узгодженість і коректність виконання операцій над даними. Вимоги ACID є фундаментальними для забезпечення цілісності даних у будь-якій інформаційній

управляючій системі, зокрема й у системі автоматизації рекомендаційної стрічки новин.

Під атомарністю (Atomicity) розуміють принцип «усе або нічого»: транзакція виконується повністю або не виконується зовсім. Якщо під час виконання будь-якого з етапів транзакції виникає збій, то система повертає базу даних до початкового стану до початку транзакції. Це забезпечує уникнення часткового оновлення даних.

Узгодженість (Consistency) означає, що після завершення транзакції база даних переходить з одного допустимого (узгодженого) стану до іншого. Тобто всі правила цілісності (ключі, обмеження, бізнес-логіка) повинні залишатися дійсними. Наприклад, не може бути створено рекомендацію новини для неіснуючого користувача.

Ізольованість (Isolation) гарантує, що одночасне виконання кількох транзакцій не призведе до порушення логіки виконання кожної з них. Кожна транзакція сприймається як така, що виконується незалежно, навіть якщо технічно відбувається паралельна обробка. Це особливо важливо в умовах високої навантаженості системи рекомендацій, де багато користувачів можуть переглядати та взаємодіяти з новинами одночасно.

Довговічність (Durability) забезпечує збереження результатів завершеної транзакції навіть у разі системного збою чи відключення. Тобто, якщо користувач переглянув новину або залишив вподобання, ця інформація не буде втрачена після збереження в БД.

Забезпечення вимог ACID дозволяє побудувати надійне, відмовостійке та узгоджене середовище збереження та обробки даних, що є критично важливим для інформаційно управляючих систем будь-якого рівня складності.

ER-діаграму таблиць бази даних наведено на рис. 3.1.

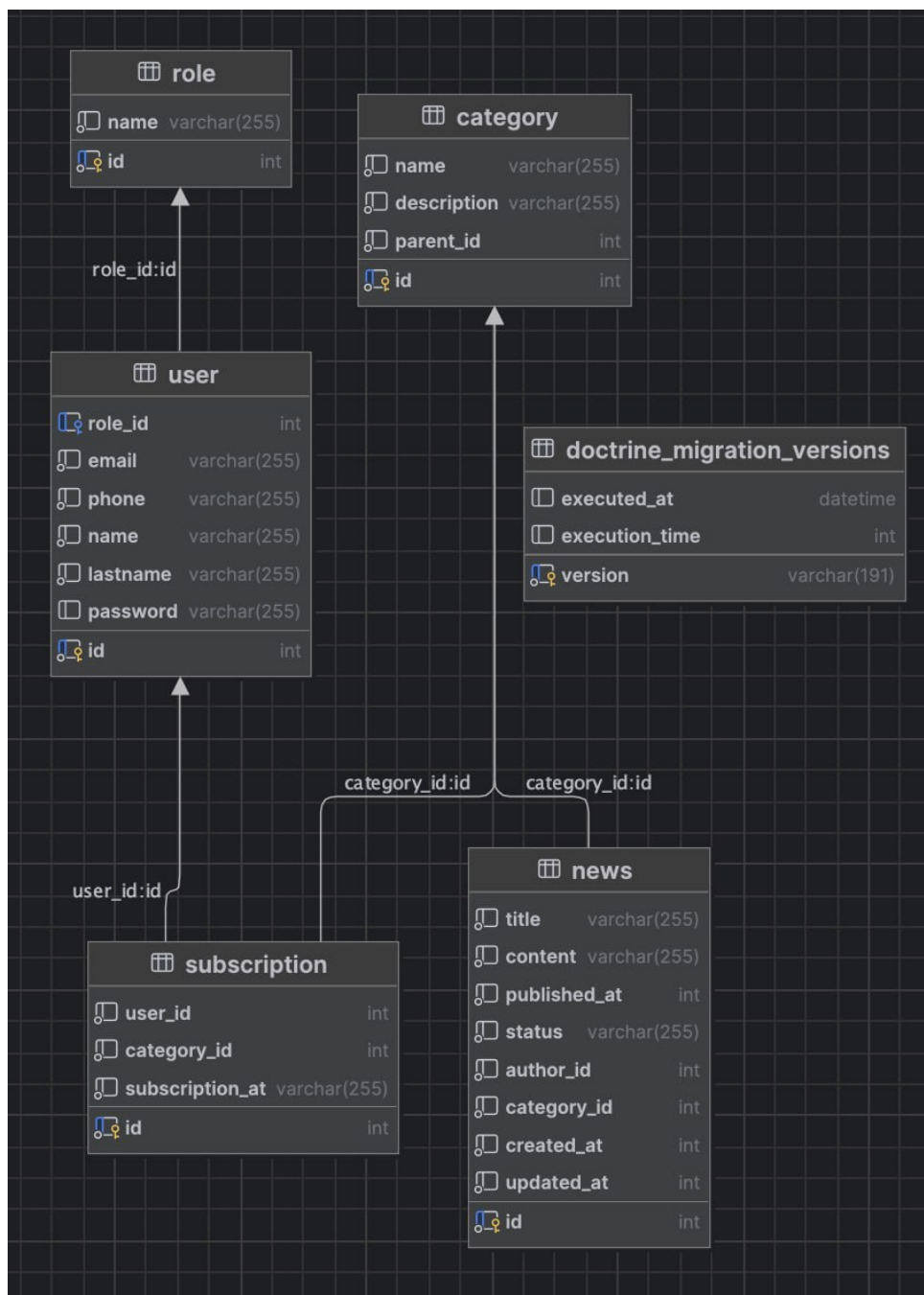


Рисунок 3.1 - ER-діаграма

*Джерело: розроблено автором*

### 3.1.2 Обґрунтування вибору СКБД та проектування фізичної моделі БД в середовищі вибраної СКБД

(СКБД) ґрунтується на низці критеріїв, зокрема: підтримка сучасних стандартів зберігання та обробки даних, масштабованість, продуктивність, безпека, відповідність ACID-вимогам, наявність зручного інструментарію для адміністрування та інтеграції з іншими компонентами інформаційно управляючої системи. У контексті розроблення системи автоматизації рекомендаційної стрічки новин оптимальним вибором є реляційна СКБД PostgreSQL. Вона забезпечує повну підтримку транзакцій, гнучке моделювання даних, широкі можливості з розширення функціональності, має відкритий вихідний код і потужну спільноту розробників.

PostgreSQL [36] дозволяє ефективно працювати з великими обсягами структурованої інформації, що характерно для новинних систем, де зберігається значна кількість користувацьких даних, текстів новин, історій взаємодій, метаданих тощо. Крім того, вона забезпечує підтримку складних запитів, індексації (включно з повнотекстовим пошуком), а також підтримує JSON/JSONB, [37] що дозволяє зберігати напівструктуровані дані у межах тієї ж системи.

Проектування фізичної моделі бази даних відбувається після створення інфологічної (логічної) моделі. На фізичному рівні визначаються типи даних для кожного атрибута сутності, створюються первинні та зовнішні ключі, індекси, обмеження цілісності, тригери та інші об'єкти БД. У середовищі PostgreSQL для реалізації фізичної моделі можуть бути використані засоби графічного інтерфейсу, як-от pgAdmin, або текстові SQL-скрипти. Наприклад, таблиця users може мати такі поля: id (PRIMARY KEY, тип SERIAL), email (UNIQUE, тип VARCHAR), created\_at (тип TIMESTAMP), а таблиця news\_interactions — зовнішні ключі на users та news, що забезпечує зв'язки між сутностями.

Таким чином, обрана СКБД дозволяє забезпечити необхідну функціональність, масштабованість та підтримку вимог до надійності, що робить її

ефективною основою для реалізації системи рекомендацій новин в умовах реального використання.

Реляційні бази даних мають низку ключових переваг, що зумовлює їхнє широке використання в інформаційних управляючих системах, зокрема при проектуванні систем автоматизації новинної стрічки. Насамперед, однією з найважливіших переваг є структурованість і чіткість організації даних. Дані зберігаються в таблицях, які мають заздалегідь визначену схему з чіткими типами даних, що забезпечує високий рівень логічної узгодженості та мінімізує ймовірність помилок при обробці.

Іншою суттєвою перевагою є забезпечення цілісності та несуперечності даних. Завдяки механізмам первинних і зовнішніх ключів, обмежень (constraints), тригерів та правил, реляційні СКБД ефективно підтримують логічні зв'язки між таблицями, автоматично запобігаючи порушенням цілісності.

Також слід відзначити потужний запитний апарат, що реалізується через мову SQL, яка є стандартом для маніпуляцій із даними. SQL дозволяє виконувати складні запити з об'єднаннями, вкладеними структурами, агрегацією та фільтрацією, що особливо важливо при аналізі великих масивів користувацької поведінки чи взаємодій з новинним контентом.

Крім того, реляційні СКБД відповідають вимогам ACID (атомарність, узгодженість, ізолюваність, довговічність), що гарантує надійність обробки транзакцій і збереження даних навіть у разі збоїв системи або апаратного забезпечення. Це критично важливо для систем, які працюють в режимі 24/7.

Варто також підкреслити масштабованість, безпеку та розвинуті механізми контролю доступу, а також можливість інтеграції з іншими системами і платформами через стандартизовані інтерфейси.

Завдяки цим перевагам реляційні бази даних залишаються домінуючим вибором для побудови інформаційно-насичених, надійних та гнучких систем, що потребують високої якості обробки даних (табл. 3.1).

Таблиця 3.1

**ПЕРЕВАГИ ТА НЕДОЛІКИ РЕЛЯЦІЙНОЇ БАЗИ ДАНИХ MYSQL**

| Переваги                                                                 | Недоліки                                                                             |
|--------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| Висока швидкодія та ефективність при роботі з великими обсягами даних    | Обмежена підтримка деяких складних транзакцій та тригерів у порівнянні з PostgreSQL  |
| Відкрите програмне забезпечення (open-source), безкоштовна базова версія | Недостатня підтримка для деяких аналітичних запитів та OLAP-операцій                 |
| Широка підтримка мов програмування та фреймворків                        | Порівняно простіший механізм збережених процедур та відсутність розвиненої типізації |
| Простота встановлення та налаштування                                    | Частина розширених функцій доступна лише у комерційній версії (MySQL Enterprise)     |
| Добре документована, має активну спільноту розробників                   | Менш ефективна масштабованість у порівнянні з деякими NoSQL рішеннями                |
| Підтримка кластеризації, реплікації, резервного копіювання               | Відсутність вбудованих засобів повноцінного версіонування даних                      |
| Сумісність з ACID-вимогами (за використання InnoDB)                      | Вимагає ретельного налаштування для забезпечення оптимальної продуктивності          |

*Джерело: розроблено автором*

У сучасних інформаційних управляючих системах важливу роль відіграють дві ключові парадигми обробки даних — OLTP (Online Transaction Processing) та OLAP (Online Analytical Processing). [38] Кожна з них виконує специфічні функції у процесах зберігання, обробки та аналізу інформації. OLTP-системи орієнтовані

на оперативну обробку транзакцій, забезпечуючи цілісність даних та високу швидкість роботи з великою кількістю коротких запитів. Натомість OLAP-системи призначені для проведення аналітичних розрахунків, підтримки прийняття рішень та побудови складних запитів на основі агрегованих даних. Порівняння цих двох підходів дозволяє глибше зрозуміти їхнє призначення, переваги та обмеження у контексті побудови ефективної інформаційної системи. (табл. 3.2).

Таблиця 3.2

### ПОРІВНЯЛЬНІ ХАРАКТЕРИСТИКИ OLTP ТА OLAP

| Критерій                | OLTP                                                   | OLAP                                                                   |
|-------------------------|--------------------------------------------------------|------------------------------------------------------------------------|
| Основне призначення     | Щоденні транзакції, операційна діяльність              | Аналітика, прогнозування, стратегічне планування                       |
| Тип операцій            | Вставка, оновлення, видалення, вибірка окремих записів | Складні запити, агрегація, багатовимірний аналіз                       |
| Структура бази даних    | Нормалізована (таблиці з чіткими зв'язками)            | Денормалізована (зіркоподібна або сніжинкоподібна схема)               |
| Обсяг даних             | Відносно невеликий обсяг, часто оновлюється            | Великий обсяг історичних даних                                         |
| Користувачі             | Працівники операційних підрозділів                     | Аналітики, менеджери, керівництво                                      |
| Швидкодія               | Оптимізована для великої кількості коротких транзакцій | Оптимізована для складних запитів з великим обсягом даних              |
| Час виконання запитів   | Мілісекунди – секунди                                  | Секунди – хвилини                                                      |
| Цілісність і транзакції | Високий пріоритет на цілісність та ACID-властивості    | Менший акцент на транзакційну цілісність, акцент на точність аналітики |

*Джерело: розроблено автором*

Отже, СУБД MySQL є багатофункціональним, перевіреним та надійним засобом для зберігання структурованих даних, який вдало поєднує формальну суворість організації з високою адаптивністю до використання в прикладних інформаційних системах різної складності.

NoSQL-сховища даних — це нереляційні системи управління базами даних, які виникли як відповідь на обмеження традиційних реляційних СУБД у роботі з великими обсягами даних, високою динамікою змін і різноманітністю структур. Основна відмінність між різними типами NoSQL [39] полягає в моделі представлення та організації даних, що впливає на сценарії їх застосування, ефективність та складність інтеграції в інформаційні управляючі системи.

Документоорієнтовані NoSQL-бази даних (наприклад, MongoDB) оперують даними у вигляді документів у форматах JSON чи BSON. [40] Вони добре підходять для зберігання структурованих та напівструктурованих даних, як-от новинні статті, користувацькі профілі, метадані. Головна їх перевага — це гнучка схема, яка дозволяє легко змінювати структуру без необхідності повної реконструкції бази.

Колонкові бази даних (наприклад, Apache Cassandra) зберігають дані у вигляді стовпців, а не рядків, що робить їх ефективними при аналітичній обробці великих масивів інформації, особливо в OLAP-сценаріях. Вони використовуються там, де потрібне швидке читання конкретних полів великої кількості записів, наприклад, у телеметрії чи логах користувачів.

Ключ-значення сховища (наприклад, Redis) забезпечують найвищу швидкість доступу до даних, але працюють з найпростішою структурою — значенням, прив'язаним до ключа. Вони зручні для кешування результатів запитів, зберігання сесій або рейтингових даних.

Графові бази даних [41] (наприклад, Neo4j) використовуються для моделювання складних зв'язків між об'єктами. У контексті новинних рекомендацій вони особливо корисні для аналізу соціальних графів, зв'язків між темами, тегами, вподобаннями користувачів. Такі бази дають змогу ефективно реалізовувати механізми персоналізації та рекомендації на основі відносин.

У загальному підсумку, вибір конкретного типу NoSQL-сховища залежить від вимог до структури даних, типів запитів, обсягів навантаження, гнучкості і масштабованості. Для комплексної системи рекомендується комбінувати кілька типів сховищ (наприклад, MongoDB + Redis або Neo4j + Cassandra), щоб досягти найкращого балансу між швидкодією, надійністю та зручністю моделювання,

що повністю відповідає потребам обраної предметної області. Інфологічну модель системи наведено на рис.3.2.

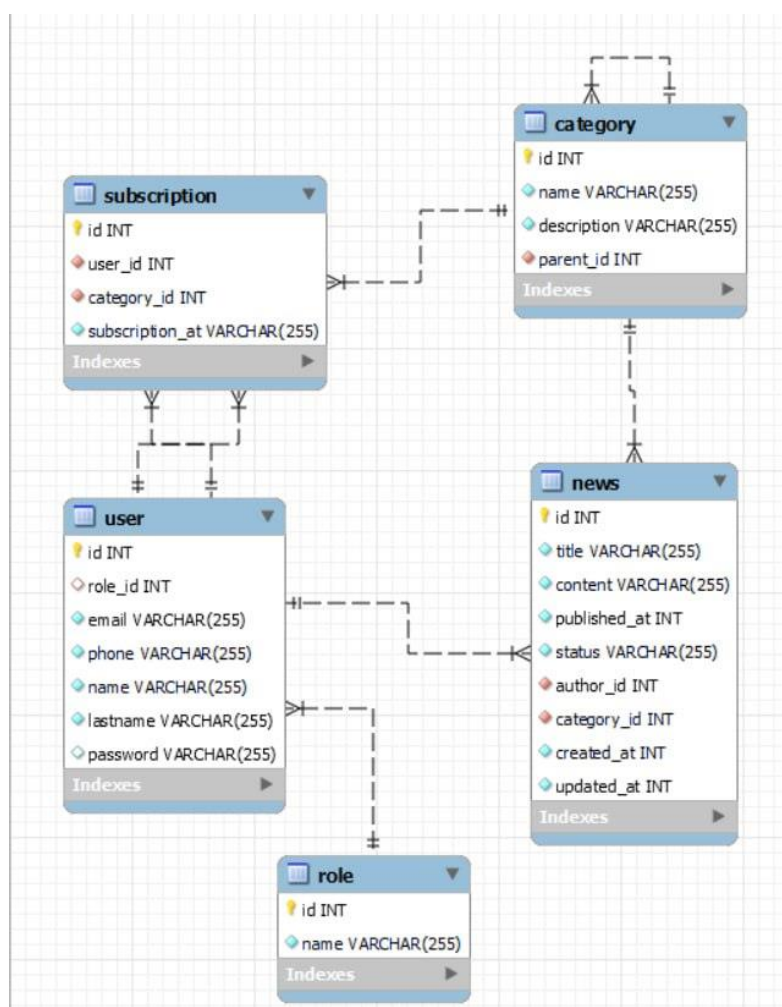


Рисунок 3.2 - Інфологічна модель системи

*Джерело: розроблено автором*

Фізична модель даних, яка демонструє, наведена на рис. 3.3.

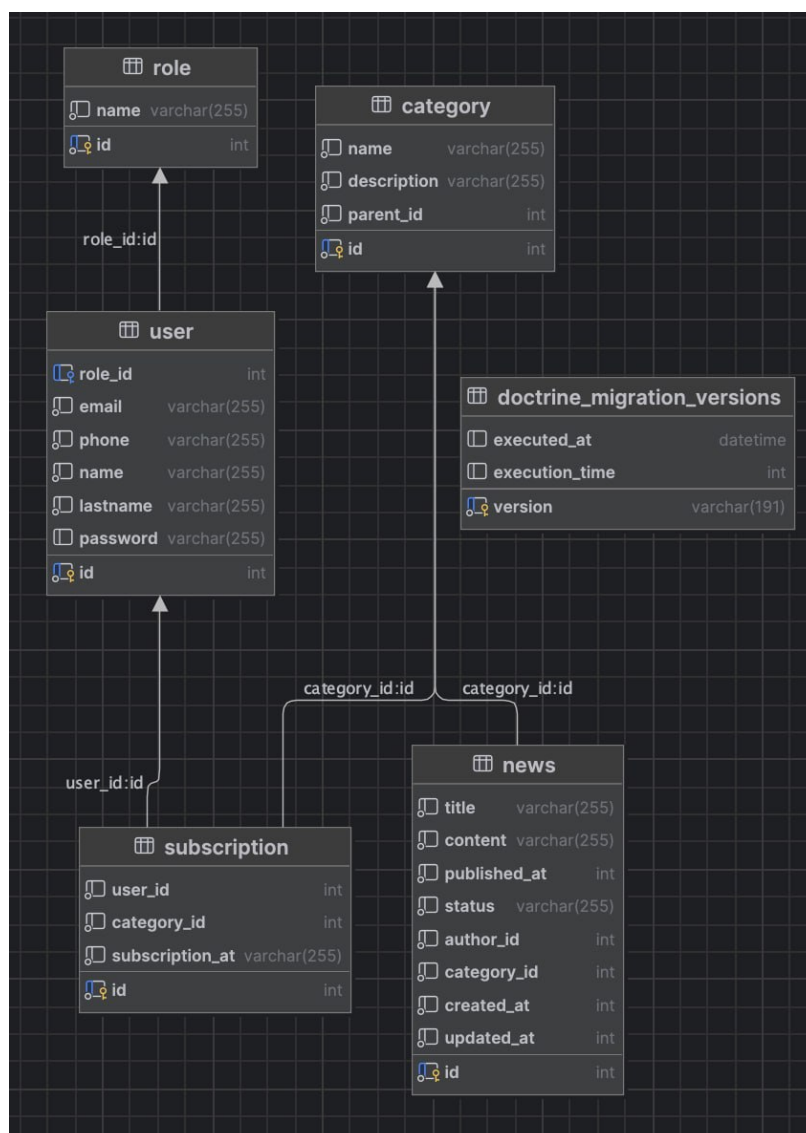


Рисунок 3.3 – Фізична модель даних

*Джерело: розроблено автором*

У контексті інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин, нижче подано структуру таблиць бази знань, необхідних для забезпечення функціонування системи. Ці таблиці відображають основні інформаційні об'єкти, їхні зв'язки та атрибути, які використовуються для зберігання, обробки та персоналізованого відображення новинного контенту. Візуалізація структури наведена на рисунках 3.4 – 3.8.

```
-- auto-generated definition
create table role
(
    id      int auto_increment
    |      primary key,
    name    varchar(255) not null
)

collate = utf8mb4_unicode_ci;
```

Рисунок 3.4 - структура таблиці role

*Джерело: розроблено автором*

```
-- auto-generated definition
create table category
(
    id          int auto_increment
    |          primary key,
    name        varchar(255) not null,
    description varchar(255) not null,
    parent_id   int          not null
)

collate = utf8mb4_unicode_ci;
```

Рисунок 3.5 - структура таблиці category

*Джерело: розроблено автором*

```
-- auto-generated definition
create table news
(
    id            int auto_increment
    primary key,
    title         varchar(255) not null,
    content       varchar(255) not null,
    published_at  int          not null,
    status        varchar(255) not null,
    author_id     int          not null,
    category_id   int          not null,
    created_at    int          not null,
    updated_at    int          not null
)

collate = utf8mb4_unicode_ci;
```

Рисунок 3.6 - News

*Джерело: розроблено автором*

```
-- auto-generated definition
create table user
(
    id            int auto_increment
    primary key,
    role_id       int          null,
    email         varchar(255) not null,
    phone         varchar(255) not null,
    name          varchar(255) not null,
    lastname      varchar(255) not null,
    password      varchar(255) null,
    constraint FK_8D93D649D60322AC
    foreign key (role_id) references role (id)
)

collate = utf8mb4_unicode_ci;

create index IDX_8D93D649D60322AC
on user (role_id);
```

Рисунок 3.7 - User

*Джерело: розроблено автором*

```
-- auto-generated definition
create table subscription
(
    id            int auto_increment
    primary key,
    user_id       int          not null,
    category_id   int          not null,
    subscription_at varchar(255) not null
)

collate = utf8mb4_unicode_ci;
```

Рисунок 3.8- Subscription

*Джерело: розроблено автором*

### 3.1.3 Контрольний приклад та обрахунок прогнозних обсягів БД

Для обґрунтування вибору підходів до реалізації бази даних та оцінки ефективності функціонування інформаційної управляючої системи доцільно провести розрахунок прогнозних обсягів бази даних. Такий аналіз дозволяє не лише передбачити необхідні обсяги пам'яті, але й забезпечити оптимальне планування ресурсів системи на етапі проєктування.

Оцінювання обсягу бази даних ґрунтується на аналітичному підході, що враховує кількість об'єктів, які підлягають зберіганню, середній розмір запису для кожної таблиці та очікувану кількість записів протягом визначеного періоду часу. Основними складовими бази даних у межах запропонованої системи є облікові записи користувачів, новинні повідомлення, категорії новин, ключові слова (теги), параметри персоналізації, історія взаємодії користувачів з контентом (вибірка,

перегляди, позначки вподобання), а також службова інформація, пов'язана з нотифікаціями.

Припустимо, що система орієнтована на одночасне обслуговування близько 10 000 активних користувачів. Для кожного користувача зберігаються основні ідентифікаційні дані, історія активності, персональні налаштування та зв'язки з переглянутими новинами. У середньому на одного користувача припадає близько 100 взаємодій з новинами на місяць, що генерує не менше ніж 1 000 000 записів взаємодії щомісячно. Кожен запис передбачає зберігання ID користувача, ID новини, часу взаємодії та додаткових параметрів (наприклад, оцінки релевантності чи реакції), загальний розмір одного такого запису оцінюється на рівні 200 байт.

Для самого новинного контенту передбачається щоденне надходження щонайменше 10 000 новин від агрегованих джерел, кожна з яких містить заголовок, короткий опис, повний текст, метадані (дата, джерело, категорія, теги) і медіа-посилання. Середній розмір одного повного запису новини становить приблизно 2-4 кілобайти залежно від насиченості контенту. За один місяць система накопичуватиме приблизно 300 000 новин, що формує близько 900 мегабайт до 1,2 гігабайт текстових даних.

З урахуванням індексації, структурних накладних витрат та резервування (наприклад, реплікація, резервне копіювання), загальний прогнозований розмір бази даних за перший рік функціонування системи може становити близько 20–30 гігабайт. Така оцінка дозволяє коректно визначити необхідні характеристики СКБД, параметри фізичного середовища для розміщення системи, а також політику масштабування і резервного зберігання інформації.

Таким чином, попередній обрахунок обсягів бази даних є критично важливим етапом підготовки до розгортання інформаційної управляючої системи, оскільки визначає вимоги до інфраструктури, забезпечує стабільність роботи системи та дозволяє ефективно управляти інформаційними потоками в умовах динамічного зростання обсягів даних.

### **3.2 Проєктування бази знань інформаційної управляючої системи та/або засоби інтелектуального аналізу даних**

Проєктування бази знань та засобів інтелектуального аналізу даних в інформаційній управляючій системі є ключовим етапом, що забезпечує якісну обробку великого масиву новинної інформації, її структурування, зберігання та подальше використання для персоналізованого інформування користувачів. База знань у контексті рекомендаційної стрічки новин виступає основою для акумуляції тематичних категорій, тегів, уподобань користувачів та історії їхньої взаємодії з контентом. Вона дозволяє не лише впорядковувати новинні матеріали, а й ефективно відслідковувати зв'язки між інтересами користувача та релевантністю новинних повідомлень.

Інструменти інтелектуального аналізу даних, інтегровані в систему, забезпечують виявлення прихованих залежностей у поведінці користувачів, оцінювання новинної активності, ранжування джерел, а також формування рекомендаційних моделей. Вони використовуються для побудови та навчання алгоритмів, що прогнозують найбільш релевантний контент для кожного окремого користувача.

У базі знань фіксуються всі ключові атрибути новин: дата публікації, джерело, тематика, кількість переглядів, оцінки користувачів, а також метаінформація — теги, ключові слова, час взаємодії. Це дозволяє системі точно враховувати як зміст новин, так і реакцію аудиторії на них.

### **3.2.1 Реалізація інтелектуального аналізу великих даних з метою наповнення бази знань**

У процесі створення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин одним із ключових етапів є реалізація інтелектуального аналізу великих даних. Це дозволяє системі не лише обробляти значні обсяги новинного контенту, але й формувати структуровану базу знань, що слугує основою для генерації персоналізованих рекомендацій.

Першочерговим завданням є визначення предметної області даних, тобто встановлення меж знань, що охоплюють ключові інформаційні джерела, типи новин, тематичні категорії, інтереси користувачів та пов'язані мета-дані. Далі відбувається аналіз і організація знань: ідентифікуються типи знань (факти, логічні залежності, інтереси, поведінкові патерни), а також розробляються онтології, які відображають зв'язки між об'єктами знань — новинами, користувачами, категоріями та джерелами.

На основі зібраних даних проєктується структура бази знань, яка включає схеми таблиць для збереження інформації про зміст новин, рівень зацікавленості користувачів, взаємодію з контентом (перегляди, вподобання, збереження), хронологію подій та семантичні теги. Після побудови схеми здійснюється безпосереднє наповнення бази знань новинною інформацією, яку отримано через інтегровані API, вебскрейпінг або імпорт даних з відкритих джерел. Важливою складовою є підтримка та розвиток бази знань, що передбачає постійне оновлення даних відповідно до поточних подій та змін у поведінці користувачів.

Засоби інтелектуального аналізу новинних даних реалізуються у декілька етапів. Перший — визначення цілей аналізу, наприклад, формування персоналізованої стрічки новин, виявлення фейкової інформації або прогнозування інтересів на основі історичних взаємодій. Другий етап полягає у зборі та підготовці даних, включаючи очищення текстів новин, нормалізацію назв джерел, обробку тегів та категорій, а також формування профілів користувачів.

На третьому етапі застосовуються алгоритми машинного навчання, методи обробки природної мови (NLP), тематичне моделювання, класифікація та кластеризація. За допомогою цих методів система може виявляти схожість новин, будувати моделі інтересів користувачів та формувати релевантні рекомендації. Результати аналізу проходять візуалізацію — у вигляді графів тематичних зв'язків, динаміки читання новин, карт популярності категорій тощо.

Завершальний етап полягає у впровадженні отриманих результатів у реальні сценарії використання системи. Рекомендаційні алгоритми інтегруються у користувацький інтерфейс, а ефективність рішень постійно моніториться за допомогою метрик, таких як CTR (Click-Through Rate), час перебування у стрічці, кількість взаємодій тощо. На основі зібраних показників здійснюється ітеративне вдосконалення моделей.

Для виконання інтелектуального аналізу даних у контексті новинної системи було обрано інструмент **WizWhy** — програмне забезпечення, яке спеціалізується на автоматизованому пошуку закономірностей у великих масивах інформації. WizWhy [42] демонструє високу ефективність у виявленні складних логічних зв'язків між змінними, що є актуальним у випадку новинного контенту з багаторівневою структурою.

Серед переваг WizWhy у контексті аналізу новин варто виокремити:

**Автоматизацію процесів аналізу**, що дозволяє швидко виявляти закономірності між категоріями новин та поведінкою користувачів без необхідності глибоких технічних налаштувань.

**Здатність до виявлення прихованих залежностей**, що дозволяє будувати ефективні гіпотези щодо інтересів користувачів на основі змісту новин.

**Зручну візуалізацію результатів**, яка дає змогу аналітикам та розробникам чітко інтерпретувати знайдені зв'язки.

**Простоту впровадження**, оскільки інтерфейс WizWhy орієнтований на швидку інтеграцію в середовище прийняття рішень без потреби у складному програмуванні.

Таким чином, реалізація інтелектуального аналізу великих даних із використанням WizWhy забезпечує ефективне наповнення бази знань інформаційної системи, підвищує релевантність новинних рекомендацій та дозволяє адаптувати контент до індивідуальних інформаційних потреб кожного користувача.

### 3.3 Програмне забезпечення

У цьому розділі проаналізовано актуальні архітектурні підходи до проєктування інформаційних управляючих систем, які сприяють досягненню високої модульності, гнучкої масштабованості та ефективної адаптації до змін. Цибулева (Onion), чиста (Clean) та гексагональна (Hexagonal або Ports and Adapters) архітектури — це підходи до побудови програмних систем, які мають спільну мету: забезпечити слабке зв'язування, чітке розмежування відповідальностей та високу модульність. Їх застосування особливо актуальне у складних інформаційно управляючих системах, зокрема таких, як системи автоматизації рекомендаційної стрічки новин. Нижче надається порівняльний аналіз кожної з архітектурних моделей у контексті її концептуальних засад, структури та доцільності використання.

Цибулева архітектура побудована навколо центральної бізнес-логіки (ядра), яка не залежить від зовнішніх шарів. Всі залежності спрямовані до центру. Зовнішні шари (інтерфейси, інфраструктура, сховища даних) взаємодіють із внутрішніми через чітко визначені інтерфейси. Головна ідея — розмістити бізнес-правила в центрі, захищаючи їх від змін у зовнішніх компонентах, таких як база даних або користувацький інтерфейс. Така модель особливо ефективна в системах, де збереження стабільної бізнес-логіки є критичним. наведена на рис. 3.9.

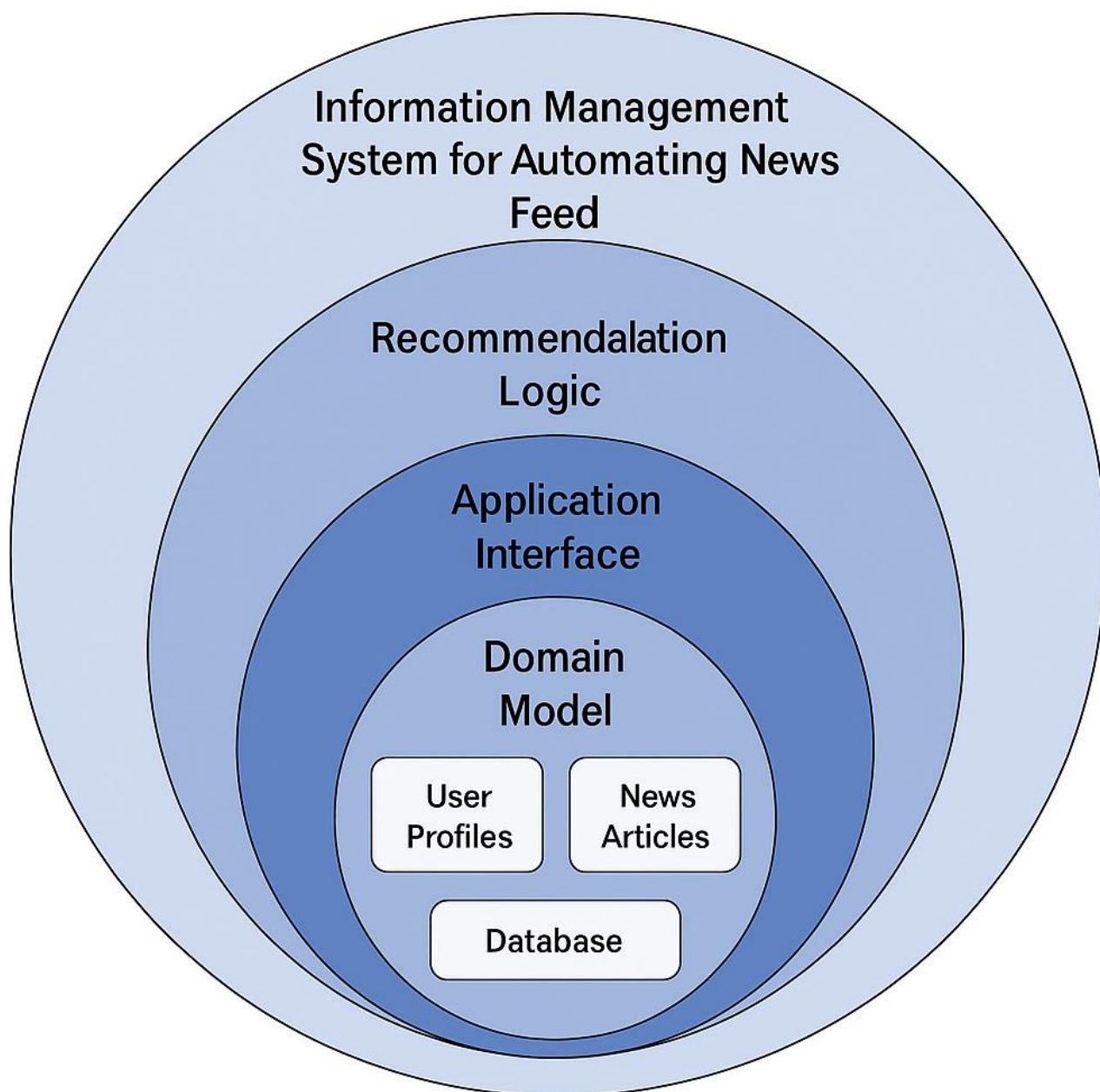


Рисунок 2.9 - діаграма цибулевої архітектури

*Джерело: розроблено автором*

У контексті розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин **цибулева архітектура (Onion Architecture)** забезпечує чіткий поділ відповідальностей між компонентами системи та сприяє високій модульності, тестованості й масштабованості програмного забезпечення.

У цій архітектурі ядро системи становить **доменна модель** — набір сутностей, що відображають предметну область новинного середовища

(наприклад, Користувач, Новина, Категорія, Інтереси, ІсторіяПереглядів). Ці сутності не мають залежностей від зовнішніх технологій, що гарантує незалежність бізнес-логіки від інфраструктури.

На наступному рівні розташовується **шар застосування (Application Services)**, який містить бізнес-логіку, зокрема правила формування стрічки, рекомендаційні алгоритми (наприклад, фільтрація за інтересами або кластеризація новин), а також сервіси для взаємодії з даними. Цей шар оперує інтерфейсами, які реалізуються у зовнішніх шарах.

Ще далі розміщено **шар інфраструктури (Infrastructure Layer)**, який реалізує специфічні технології: доступ до реляційної або NoSQL БД, підключення до зовнішніх API (наприклад, агрегаторів новин), логування, кешування тощо. Усі залежності мають спрямування лише до центру архітектури, що відповідає принципу **інверсії залежностей**.

Найзовнішній шар становить **шар представлення (Presentation Layer)**, наприклад, вебінтерфейс або мобільний застосунок. Тут користувач взаємодіє зі стрічкою новин, виставляє вподобання, читає публікації — усі ці дії передаються до шару застосування через контролери..

Сформулювати узагальнення наведеного опису цибулевої архітектури, а також визначити її основні переваги та недоліки доцільно у табличній формі, що сприятиме наочному представленню ключових характеристик архітектурного підходу.(табл. 3.1).

Таблиця 3.1

## ПЕРЕВАГИ ТА НЕДОЛІКИ ЦИБУЛЕВОЇ АРХІТЕКТУРИ

| Переваги                                                      | Недоліки                                                                                     |
|---------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| Чітке розділення відповідальностей між рівнями системи.       | Висока складність реалізації для невеликих проєктів                                          |
| Зменшення залежності бізнес-логіки від зовнішніх компонентів. | Потреба в глибокому розумінні принципів архітектури                                          |
| Полегшення тестування завдяки ізоляції шарів                  | Можливі накладні витрати на інтеграцію між шарами                                            |
| Підвищення гнучкості та масштабованості системи               | Ускладнення налагодження при поганій документації                                            |
| Полегшення супроводу та модернізації                          | Надлишкова структурованість у простих випадках може призводити до перевантаження архітектури |

*Джерело: розроблено автором*

Ця архітектура є доцільною для складних та динамічних систем, зокрема для рекомендаційних стрічок новин, де важливо відокремити бізнес-логіку від механізмів доставки, зберігання та обробки даних. Хоча вона потребує додаткових ресурсів і навичок для впровадження, її переваги стають очевидними у процесі масштабування та підтримки системи.

**Чиста архітектура**, [43] запропонована Робертом Мартіном є розвитком і узагальненням цибулевої. Вона також використовує концентричні кола, де внутрішнє коло — це сутності (entities), далі — випадки використання (use cases), потім — інтерфейсні адаптери, і нарешті — фреймворки та драйвери (інфраструктура). Основний принцип — залежності повинні йти тільки всередину, і кожен рівень може посылатися лише на більш внутрішній. Ця архітектура дозволяє створювати надзвичайно гнучкі та незалежні від технологій додатки, що спрощує тестування, підтримку й повторне використання коду. Діаграму чистої архітектури наведено на рис. 3.10.

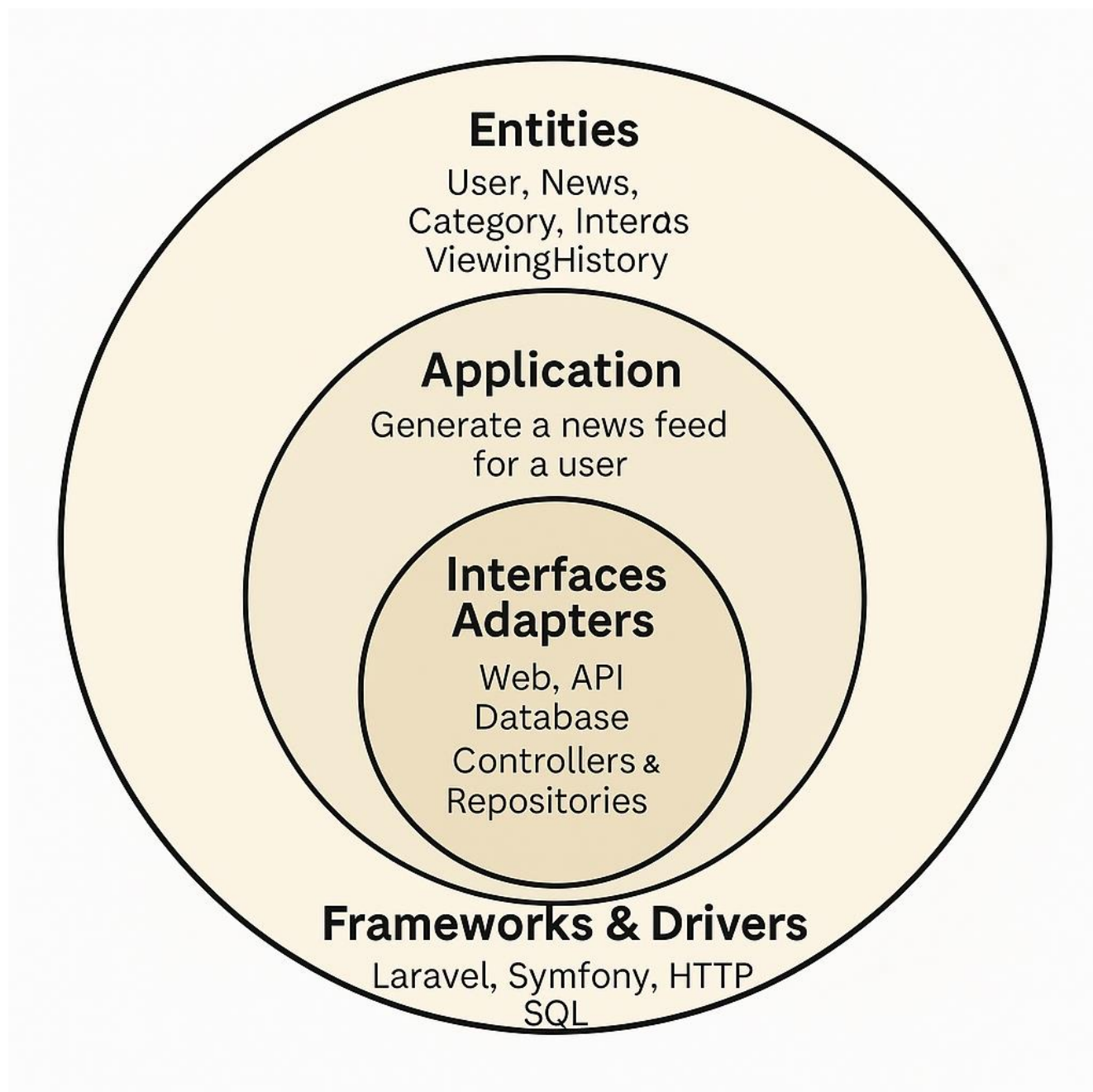


Рисунок 3.10 - Діаграма чистої архітектури

*Джерело: розроблено автором*

Основна мета чистої архітектури — забезпечити незалежність бізнес-логіки від зовнішніх елементів, таких як бази даних, фреймворки чи інтерфейси користувача. Це дає змогу легко змінювати або замінювати зовнішні компоненти без впливу на внутрішню логіку системи, що підвищує масштабованість, гнучкість і тестованість.

Центральним принципом архітектури є зосередження бізнес-логіки у ядрі системи, де розміщуються сутності (Entities), що представляють об'єкти

предметної області й реалізують ключові бізнес-правила. Ці сутності повністю ізольовані від зовнішніх систем, технологій та бібліотек.

Концепція розподілу на шари ґрунтується на побудові системи у вигляді концентричних кіл, де кожен внутрішній шар характеризується вищим рівнем абстракції, ніж зовнішній. При цьому зовнішні шари можуть звертатися до внутрішніх, але не навпаки. До зовнішніх рівнів належать інтерфейси користувача, бази даних, контролери та інтеграція зі сторонніми сервісами.

- **Entities (Сутності)** — найвнутрішнє коло, що містить бізнес-логіку високого рівня. У контексті системи новин сюди входять такі об'єкти, як Користувач, Новина, Категорія, Уподобання, які зберігають довготривалі дані та інваріанти системи.

- **Use Cases (Використання)** — наступне коло, де зосереджено сценарії взаємодії користувачів з системою. Наприклад, формування персоналізованої стрічки новин, фільтрація за інтересами, збереження історії перегляду.

- **Interface Adapters (Адаптери інтерфейсів)** — містять контролери, презентери, DTO (об'єкти передачі даних), які перетворюють вхідні дані зовнішнього світу у формат, що зручний для внутрішньої обробки, і навпаки. Тут реалізується обробка HTTP-запитів, серіалізація/десеріалізація новин тощо.

- **Frameworks and Drivers (Фреймворки і драйвери)** — найзовнішнє коло, що включає базу даних (наприклад, MySQL), веб-фреймворк, API, зовнішні сервіси аналітики або NLP. Це все те, що система використовує, але що можна легко замінити без зміни внутрішньої логіки.

Узагальнити вище описану інформацію щодо чистої архітектури та виокремити переваги та недоліки даної архітектури, можна зобразити у вигляді таблиці (табл. 3.1).

Таблиця 3.1

## ПЕРЕВАГИ ТА НЕДОЛІКИ ЧИСТОЇ АРХІТЕКТУРИ

| Переваги                                                                                                    | Недоліки                                                                                       |
|-------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|
| <b>Незалежність від технологій:</b> ядро системи не залежить від бази даних, UI або сторонніх бібліотек.    | <b>Складність проєктування:</b> потребує високої кваліфікації архітектора на етапі планування. |
| <b>Висока модульність:</b> кожен компонент має чітку відповідальність, що спрощує підтримку та тестування.  | <b>Збільшення обсягу коду:</b> додаткові рівні абстракції можуть призводити до роздутого коду. |
| <b>Полегшення тестування:</b> ядро системи можна тестувати незалежно від інфраструктурних компонентів.      | <b>Надмірна абстракція:</b> для невеликих проєктів може бути надмірно складною і неефективною. |
| <b>Покращена масштабованість і підтримка змін:</b> спрощує адаптацію до нових вимог.                        | <b>Витрати часу та ресурсів:</b> реалізація архітектури вимагає більше часу на старті проєкту. |
| <b>Забезпечення довготривалої підтримки:</b> дозволяє легко модернізувати або замінювати окремі компоненти. | <b>Початкове зниження продуктивності:</b> через додаткові рівні взаємодії.                     |

*Джерело: розроблено автором*

Підсумовуючи описані факти, Чиста архітектура є потужним підходом до проєктування складних інформаційних систем, який забезпечує чітке розмежування відповідальностей, модульність та незалежність бізнес-логіки від технічних деталей. Незважаючи на певні недоліки, зокрема складність реалізації та підвищені вимоги до проєктування, її переваги роблять її доцільною для створення гнучких, масштабованих і довготривало підтримуваних систем, таких як інформаційна управляюча система для рекомендаційної стрічки новин.

Гексагональна архітектура [44] (або архітектура портів і адаптерів) фокусується на ізоляції ядра додатку від зовнішнього середовища. Усі зовнішні взаємодії (користувачі, бази даних, API, тощо) реалізуються через порти

(інтерфейси), до яких підключаються адаптери (реалізації). Це дозволяє легко змінювати або тестувати зовнішні компоненти без впливу на логіку програми. Особливо зручна ця модель у мікросервісній архітектурі або у випадках, коли система повинна підтримувати декілька типів взаємодії з користувачем або іншими системами.

Гексагональна архітектура, або архітектура портів і адаптерів (Ports and Adapters), представляє сучасний підхід до проєктування інформаційних управляючих систем, у якому ключовою ідеєю є ізоляція бізнес-логіки від зовнішніх впливів. Основне призначення цієї архітектури — забезпечити максимальну незалежність внутрішнього ядра застосунку від зовнішніх пристроїв, інтерфейсів, баз даних, вебсервісів тощо. Завдяки використанню чітко визначених портів (інтерфейсів) і адаптерів, які реалізують ці порти, гексагональна архітектура сприяє високій тестованості, зручній інтеграції з різними середовищами, а також покращеній підтримці змін і розширення функціональності. У межах даної інформаційної системи, що автоматизує рекомендаційну стрічку новин, така архітектура дозволяє забезпечити масштабованість, гнучкість та стійкість системи до змін у зовнішньому програмному оточенні. Діаграма гексагональної архітектури наведена на рис. 3.11.

# Hexagonal Architecture

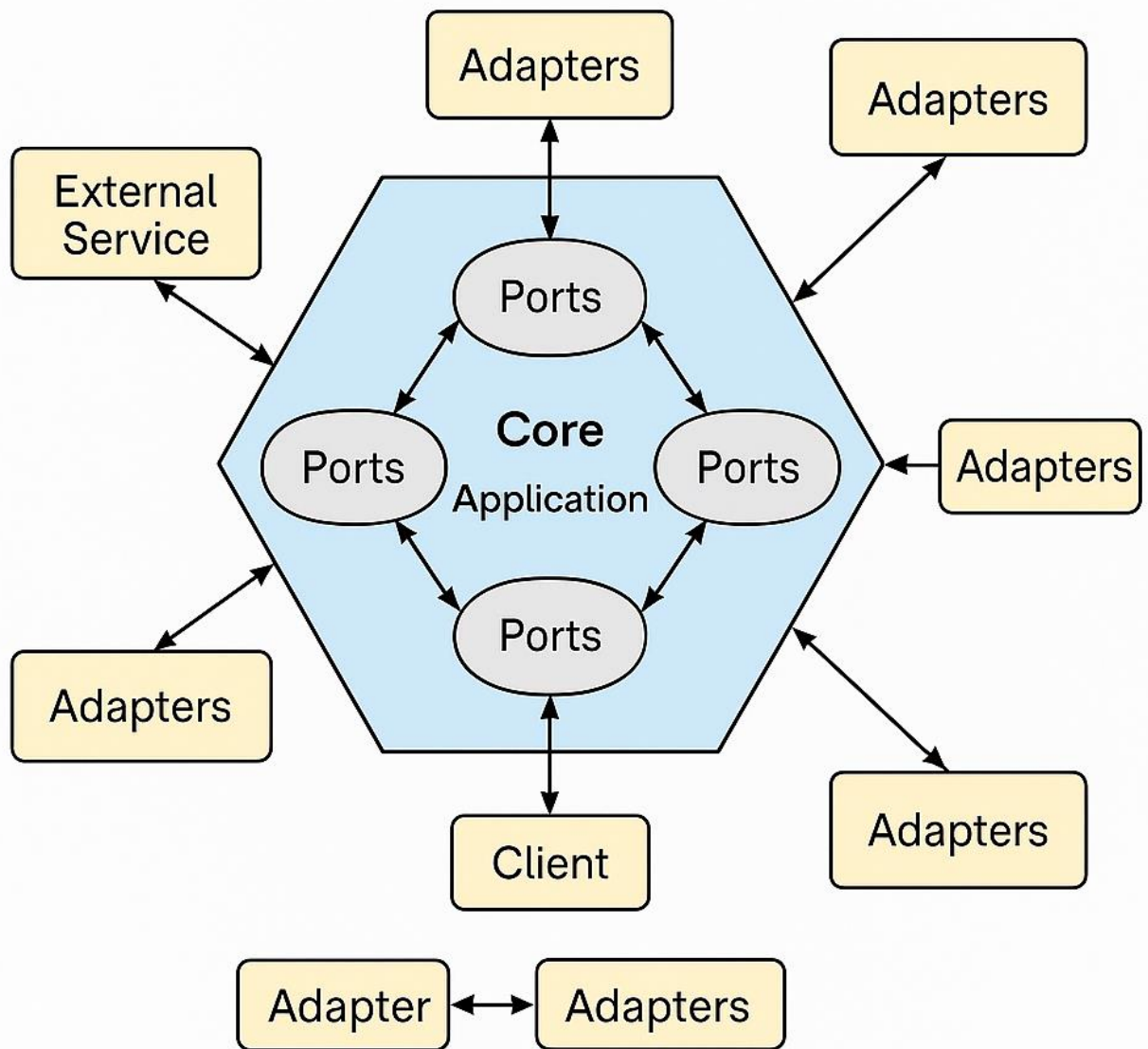


Рисунок 3.11 –

*Джерело: розроблено автором*

Систематизовану інформацію про гексагональну архітектуру разом із визначеними її перевагами та недоліками доцільно представити у табличній формі для наочності та кращого сприйняття.(табл. 3.1).

Таблиця 3.1

## ПЕРЕВАГИ ТА НЕДОЛІКИ ГЕКСАГОНАЛЬНОЇ АРХІТЕКТУРИ

| Переваги                                                             | Недоліки                                                      |
|----------------------------------------------------------------------|---------------------------------------------------------------|
| Забезпечує чітке розділення бізнес-логіки та зовнішніх інтерфейсів   | Ускладнене початкове проєктування та розуміння для новачків   |
| Підвищує гнучкість і можливість масштабування                        | Може бути надмірною для невеликих або простих проєктів        |
| Полегшує тестування бізнес-логіки в ізоляції від інфраструктури      | Необхідність ретельного управління залежностями та адаптерами |
| Полегшує заміну або інтеграцію зовнішніх компонентів без змін у ядрі | Потребує додаткових витрат часу на розробку адаптерів         |
| Підвищує модульність та повторне використання коду                   | Зростає кількість модулів і складність координації між ними   |

*Джерело: розроблено автором*

У результаті аналізу сучасних архітектурних підходів — цибулевої, чистої та гексагональної — встановлено, що всі три моделі забезпечують високий рівень модульності, масштабованості та гнучкості інформаційних управляючих систем. Проте їх відмінності в організації взаємодії компонентів системи, принципах залежностей та реалізації бізнес-логіки визначають різні сфери ефективного застосування.

Гексагональна архітектура вирізняється акцентом на взаємодії із зовнішнім світом через порти та адаптери, що робить її придатною для систем з великою кількістю зовнішніх інтерфейсів. Чиста архітектура пропонує чітке розділення рівнів абстракції, що є цінним для великих корпоративних систем із тривалим життєвим циклом. Водночас, складність реалізації та потреба у суворому дотриманні принципів інверсії залежностей можуть ускладнювати її практичне впровадження на ранніх етапах.

Цибулева архітектура, у свою чергу, поєднує переваги чіткої структуризації з практичністю реалізації. Центрування навколо бізнес-логіки, відсутність залежності від інфраструктурних компонентів, а також підтримка принципів

SOLID дозволяють створити систему, стійку до змін, легко тестовану й адаптивну до розширення. Саме ці властивості є критично важливими для реалізації інформаційної управляючої системи рекомендаційної стрічки новин, де актуальність, масштабованість і підтримка змінних алгоритмів обробки даних мають першочергове значення.

З огляду на наведене, доцільним і обґрунтованим вибором архітектури для проєкту є **цибулева архітектура**, як така, що оптимально поєднує функціональність, ефективність і зручність підтримки.

У процесі розробки сучасних веб-орієнтованих інформаційних управляючих систем особливе значення має обґрунтований вибір мови програмування, оскільки саме від неї залежить ефективність реалізації функціоналу, масштабованість, продуктивність, інтеграційні можливості та зручність подальшого супроводу системи. Аналіз мов програмування дозволяє визначити ті технологічні засоби, що найбільш повно відповідають вимогам до створення веб-систем — зокрема, підтримку серверного та клієнтського коду, взаємодії з базами даних, реалізації API та забезпечення безперебійної роботи в мережевому середовищі. У цьому підрозділі буде проведено порівняльний аналіз поширених мов програмування з урахуванням особливостей проєктованої системи, що дозволить обґрунтувати вибір оптимального інструментарію для її реалізації.

Мови програмування, які використовуються для розробки веб-систем, поділяються на дві основні категорії: мови для фронтенду (клієнтської частини) та бекенду (серверної частини). Кожна з них має свої переваги, обмеження та специфіку застосування.

Для розробки клієнтської частини веб-систем стандартом де-факто є мова JavaScript [45], яка підтримується всіма сучасними браузерами. У поєднанні з фреймворками, такими як React, Angular чи Vue.js, JavaScript дозволяє створювати інтерактивні, динамічні інтерфейси користувача. Розширенням JavaScript є мова TypeScript [46], яка додає статичну типізацію, підвищуючи надійність коду та полегшуючи його супровід.

На серверному боці вибір мови програмування значною мірою залежить від складності системи, вимог до продуктивності та наявного технологічного стеку. Однією з найпопулярніших є PHP — мова, яка має широку підтримку, численні фреймворки (Laravel, Symfony) та добре підходить для побудови систем середньої складності. Python (зокрема фреймворки Django, Flask) також є популярним вибором завдяки своїй простоті, широкій екосистемі та активному використанню у сфері аналітики та штучного інтелекту. Для складних і високонавантажених систем часто обирають Java (Spring) або C# (ASP.NET), які забезпечують високий рівень безпеки, масштабованості та інтеграції з корпоративним середовищем.

Окремо варто згадати Node.js, який дозволяє використовувати JavaScript на сервері. Це дає змогу використовувати одну й ту саму мову як для фронтенду, так і для бекенду, спрощуючи розробку та обслуговування веб-системи. Node.js відзначається високою швидкістю роботи та великою кількістю модулів для реалізації будь-яких задач.

З огляду на розробку інформаційної управляючої системи, яка має обробляти велику кількість даних, взаємодіяти з базою даних та бути гнучкою до змін, доцільним є вибір PHP у поєднанні з фреймворком Laravel. Це дозволяє досягти швидкої розробки, структурованості коду, високої гнучкості та доступності великої кількості бібліотек і модулів.

З огляду на проведений аналіз мов програмування для розробки веб-систем, можна дійти висновку, що мова PHP, у поєднанні з сучасним фреймворком Laravel, є оптимальним вибором для реалізації інформаційної управляючої системи. PHP має багаторічний досвід використання у веб-розробці, підтримує великий набір інструментів для роботи з базами даних, автентифікацією, маршрутизацією, а також забезпечує високу швидкість розгортання та обслуговування систем. Laravel, як один з найпопулярніших фреймворків PHP, забезпечує чітку архітектурну структуру, підтримку шаблонів проєктування, вбудовані механізми захисту та простоту масштабування. Крім того, PHP добре інтегрується з MySQL — обраною системою керування базами даних — що дозволяє забезпечити стабільну та ефективну роботу системи. Таким чином, PHP є доцільним вибором для реалізації

проєкту з урахуванням його функціональних вимог, обмежень та подальшого супроводу.

Для розробки на мові PHP існує широкий спектр інтегрованих середовищ розробки (IDE), кожне з яких має свої сильні сторони, залежно від вимог до продуктивності, функціональності та зручності використання. Серед найпопулярніших IDE можна виділити такі:

Visual Studio Code [47] (VS Code) — легка, розширювана та безкоштовна IDE з широкою підтримкою плагінів, включаючи розширення для PHP (наприклад, PHP Intelephense, PHP Debug). Підходить для початківців і досвідчених розробників, однак вимагає додаткової конфігурації для отримання повноцінного функціоналу.

NetBeans — [48] безкоштовне середовище з підтримкою PHP, має зручну навігацію по проєкту, підсвічування синтаксису, дебагер та інші базові інструменти. Проте воно поступається іншим IDE у плані швидкодії та глибини інтеграції з фреймворками, такими як Laravel.

Eclipse PDT (PHP Development Tools) — [49] розширення для Eclipse, яке надає інструменти для роботи з PHP. Підтримує автодоповнення, налагодження коду, інтеграцію з системами контролю версій. Але загалом Eclipse має складний інтерфейс і не є інтуїтивно зрозумілим для швидкої розробки.

PHPStorm — професійне середовище розробки від компанії JetBrains, спеціалізоване на PHP. Воно підтримує всі основні фреймворки (Laravel, Symfony, Yii тощо), має потужні засоби для рефакторингу, статичного аналізу коду, інтеграцію з базами даних, системами керування версіями (Git), та налагодженням (debug). PHPStorm також надає зручні інструменти для роботи з фронтендом, що важливо у розробці повноцінних веб-систем.

З урахуванням складності та функціональної насиченості інформаційної управляючої системи, що розробляється в межах цього проєкту, найбільш доцільним середовищем розробки є PHPStorm. Воно забезпечує високу продуктивність, глибоку інтеграцію з технологічним стеком проєкту та зручність у

підтримці коду, що істотно підвищує ефективність розробки і якість програмного продукту.

## ПОКАЗАТИ АРХІТЕКТУРУ АППКИ В ДІАГРАМІ

У процесі розробки інформаційної управляючої системи надзвичайно важливо дотримуватись чітко визначених принципів архітектури програмного забезпечення, що забезпечують масштабованість, супровідність та зрозумілість коду. Одним із таких фундаментальних підходів є застосування принципів

**SOLID** - сукупності п'яти базових правил об'єктно-орієнтованого програмування, спрямованих на створення програмних систем з високим рівнем гнучкості та розширюваності.

SOLID — це аббревіатура, що включає такі принципи:

- **S** - Принцип єдиної відповідальності (Single Responsibility Principle): кожен клас повинен мати лише одну причину для змін, тобто відповідати лише за одну функціональність. У контексті ІУС це дозволяє розділити бізнес-логіку, доступ до даних та представлення користувача на окремі компоненти.

- **O** - Принцип відкритості/закритості (Open/Closed Principle): програмні модулі мають бути відкритими для розширення, але закритими для модифікації. Це забезпечує можливість додавання нової поведінки без змін існуючого коду, що критично важливо для довгострокової підтримки та адаптації системи.

- **L** - Принцип підстановки Лісков (Liskov Substitution Principle): підкласи повинні повноцінно замінювати базові класи без порушення логіки програми. Це гарантує коректну роботу системи при використанні поліморфізму, що часто застосовується у реалізації інтерфейсів компонентів ІУС.

- **I** - Принцип розділення інтерфейсу (Interface Segregation Principle): краще створювати багато вузькоспеціалізованих інтерфейсів, ніж один універсальний, який реалізує зайву для багатьох компонентів функціональність. Це дозволяє зменшити зв'язність між модулями та спростує тестування.

- **D** - Принцип інверсії залежностей (Dependency Inversion Principle): високорівневі модулі не повинні залежати від низькорівневих; обидва типи мають залежати від абстракцій. Абстракції не повинні залежати від деталей — деталі

мають залежати від абстракцій. Це особливо важливо для забезпечення незалежності бізнес-логіки від конкретних реалізацій баз даних чи зовнішніх API.

Таким чином, дотримання принципів SOLID на етапі проєктування дозволяє створити гнучку, легко розширювану та адаптивну архітектуру програмного забезпечення, що відповідає сучасним вимогам до розробки складних інформаційних систем. У межах реалізації даної системи принципи SOLID є основою при моделюванні об'єктів, визначенні зв'язків між компонентами та формуванні контрактів взаємодії між підсистемами.

**Принципи чистого коду** (Clean Code) — [49] це набір практичних рекомендацій і правил, що спрямовані на підвищення читабельності, зрозумілості та підтримуваності програмного коду. Вони базуються на багаторічному досвіді розробників і сформульовані, зокрема, в роботах Роберта К. Мартіна. Чистий код — це код, який легко читати, розуміти, змінювати та розширювати іншими членами команди, навіть через тривалий час після написання. У контексті проєктування інформаційної управляючої системи ці принципи відіграють критичну роль у забезпеченні надійності та масштабованості розробки.

У процесі розробки складних інформаційних управляючих систем особливе значення набуває якість програмного коду, від якої залежить не лише працездатність програмного забезпечення, а й зручність його супроводу, масштабування та адаптації до змін. Одним із ключових підходів до досягнення високої якості коду є дотримання принципів **чистого коду** (Clean Code), які забезпечують зрозумілу, структуровану та легку для підтримки архітектуру програмного продукту. Застосування цих принципів дозволяє мінімізувати технічний борг, полегшити командну роботу та створити стабільне середовище для подальшого розвитку системи.

Основні принципи чистого коду включають:

– Зрозумілі імена змінних, функцій і класів.

Назви мають точно описувати призначення об'єкта. Уникають скорочень і абстрактних позначень — замість `x` краще написати `userEmail`.

– Малий розмір функцій.  
Функції повинні бути короткими й виконувати лише одну логічну дію. Це підвищує читабельність і полегшує тестування.

– Один рівень абстракції в межах одного блоку коду.  
Не слід поєднувати високорівневу логіку з низькорівневими деталями — наприклад, не варто в одному методі описувати і бізнес-логіку, і логіку читання з бази даних.

– Уникнення дублювання коду.  
Повторення логіки призводить до труднощів у підтримці. Спільні частини необхідно виділяти в окремі функції або класи.

– Коментарі — лише там, де це справді потрібно.  
Код повинен бути самодокументованим — добре написаний код пояснює себе сам. Коментарі використовуються тільки для пояснення складних ділянок або бізнес-правил.

– Обробка помилок — без порушення основного потоку логіки.  
Код повинен акуратно обробляти винятки, не знижуючи при цьому загальної читабельності.

– Незалежність і модульність.  
Кожен компонент або модуль повинен мати чітко визначене призначення і не залежати від внутрішніх деталей інших компонентів.

– Автоматичне тестування.  
Чистий код передбачає високу покритість тестами. Це забезпечує стабільність при внесенні змін і спрощує рефакторинг.

– Підтримка стилістичних стандартів.  
Дотримання конвенцій щодо форматування, відступів, назв тощо забезпечує єдину структуру в усьому проєкті.

– Безпечне видалення коду.  
Код, який не використовується, не має права на існування в репозиторії. Його потрібно своєчасно видаляти, щоб не збільшувати складність системи.

Загалом, чистий код — це не просто технічна естетика, а дисципліна, що прямо впливає на якість програмного продукту, знижує кількість помилок, полегшує командну роботу та прискорює реалізацію нових функціональностей. У рамках створення інформаційної управляючої системи дотримання цих принципів забезпечує ефективний розвиток і супровід проєкту.

У процесі створення інформаційних управляючих систем особливу увагу приділяють структуризації програмного коду з метою досягнення високого рівня модульності, повторного використання компонентів та забезпечення зручного супроводу. Одним із найпоширеніших архітектурних рішень у розробці веб-додатків є патерн MVC (Model-View-Controller), який передбачає розділення логіки додатка на три окремі компоненти: модель, яка відповідає за дані та бізнес-логіку; представлення (view), що відображає інформацію користувачу; а також контролер, який керує взаємодією між користувачем, моделлю і відображенням. Таке розділення дозволяє чітко організувати код, що спрощує масштабування додатку та дає можливість ефективно розподіляти обов'язки між учасниками команди розробників.

До переваг MVC можна віднести логічну ізоляцію частин системи, що підвищує гнучкість і полегшує тестування окремих компонентів. Зміни у візуальному представленні не впливають на бізнес-логіку, так само як і зміни в логіці не вимагають редагування інтерфейсу. Крім того, завдяки поділу обов'язків MVC дозволяє ефективно реалізовувати багатоцільові інтерфейси, зокрема для різних платформ. Проте цей підхід має і певні недоліки. Основним є відносна складність реалізації на початкових етапах, оскільки вимагається чітке структурування проєкту та додаткова увага до зв'язків між компонентами. У дрібних проєктах це може призвести до надлишкової абстракції.

Попри зазначені недоліки, обрання патерну MVC для розробки інформаційної управляючої системи є доцільним рішенням. Це пояснюється тим, що він забезпечує гнучку архітектуру, сприяє супроводжуваності коду, полегшує реалізацію нових функціональностей і відповідає принципам чистого коду, що є надзвичайно важливим у розробці сучасних масштабованих програмних рішень.

Laravel — це популярний open-source фреймворк для розробки веб-додатків мовою програмування PHP, який орієнтований на зручність розробника, високу продуктивність і відповідність сучасним стандартам розробки. Його архітектура побудована навколо патерну MVC (Model-View-Controller), що забезпечує чітке розділення відповідальностей у структурі коду. Основною метою Laravel є забезпечення гнучкого, масштабованого і водночас зручного середовища для створення веб-додатків будь-якого рівня складності — від невеликих сайтів до великих інформаційних систем.

Фреймворк активно застосовується у сферах, пов'язаних із розробкою корпоративних систем управління, електронної комерції, CRM- та ERP-рішень, онлайн-сервісів, RESTful API та інших типів веб-застосунків. Laravel містить розвинену систему маршрутизації, вбудовані засоби аутентифікації, авторизації, роботи з базами даних через ORM Eloquent, підтримку шаблонізації через Blade, а також має потужні механізми кешування, обробки черг, тестування та інтеграції з зовнішніми сервісами.

У своїй основі Laravel використовує ряд патернів проектування, серед яких окремо варто відзначити MVC, IoC (Inversion of Control) та DI (Dependency Injection), що сприяє низькому рівню зв'язності між компонентами і високій гнучкості архітектури. Крім того, фреймворк підтримує використання шаблону репозиторію, що забезпечує додаткову абстракцію доступу до даних, та сервіс-орієнтовану архітектуру, яка дозволяє чітко структурувати бізнес-логіку.

Серед ключових переваг Laravel слід відзначити зручність використання, зрозумілу та добре документовану структуру, велику екосистему (включаючи Laravel Breeze, Jetstream, Horizon, Nova), активну спільноту, а також високу інтегрованість із сучасними технологіями та інструментами CI/CD. Разом із цим, фреймворк має певні недоліки, зокрема високу залежність від самої екосистеми Laravel, що може ускладнити інтеграцію з деякими сторонніми рішеннями. Крім того, через велику кількість абстракцій і служб фреймворк іноді вимагає більших ресурсів і не завжди підходить для простих, ресурсно обмежених проєктів.

Узагальнюючі Laravel є потужним інструментом для створення сучасних веб-додатків, який поєднує елегантність коду, гнучкість архітектури та ефективність розробки, що робить його оптимальним вибором у багатьох проєктних контекстах.

Symfony — це високопродуктивний фреймворк для розробки веб-додатків на мові програмування PHP, який орієнтований на великі, масштабовані та розширювані проєкти. Створений із акцентом на гнучкість, модульність і відповідність принципам об'єктно-орієнтованого програмування, Symfony є одним із найбільш потужних і професійно орієнтованих фреймворків у екосистемі PHP. Його використання є доцільним у випадках, коли потрібне створення складних інформаційних систем, корпоративних платформ, державних сервісів, сервісів з мікросервісною архітектурою або проєктів із високими вимогами до надійності та підтримованості коду.

Symfony [50] реалізує низку ключових шаблонів проєктування, серед яких найважливішими є MVC (Model-View-Controller), Dependency Injection, Event Dispatcher, а також шаблон сервісів, що забезпечує ефективне управління залежностями в рамках контейнера інверсії керування. Завдяки цим патернам фреймворк дозволяє досягти високого рівня абстракції, гнучкості у налаштуванні компонентів та чіткого розмежування функціональних частин системи. В основі Symfony лежить набір компонентів, які можна використовувати окремо у будь-якому PHP-проєкті, що робить його особливо привабливим для розробників, які прагнуть до максимальної кастомізації та повторного використання коду.

До переваг Symfony можна віднести його високу модульність, зрілу архітектуру, підтримку стандартизованих підходів (наприклад, PSR), активну спільноту та довготривалу підтримку (LTS-версії), що важливо для критичних і довготривалих проєктів. Symfony має розвинені механізми кешування, безпеки, локалізації, валідації, а також зручну інтеграцію з ORM Doctrine, що забезпечує ефективну роботу з базами даних.

Втім, недоліки також присутні. Symfony має досить високу криву навчання, особливо для початківців, а також вимагає більше часу на початкове налаштування

у порівнянні з іншими фреймворками, зокрема Laravel. Через свою масштабність він може бути надмірним для невеликих або простих проєктів, де потрібна швидка розробка без глибокого налаштування.

Загалом Symfony є потужним вибором для побудови надійних, підтримуваних та розширюваних веб-систем, коли пріоритет надається контролю над архітектурою, строгому дотриманню стандартів та гнучкому управлінню залежностями.

У сучасній розробці програмного забезпечення вибір архітектурного підходу є критично важливим рішенням, яке визначає масштабованість, гнучкість та підтримуваність системи у довгостроковій перспективі. Серед найпоширеніших підходів до архітектурного проєктування веб-систем домінують монолітна та мікросервісна архітектури, кожна з яких має власну логіку побудови, переваги та обмеження. Вибір між ними зазвичай залежить від специфіки завдань, ресурсів команди та очікувань щодо майбутнього розвитку продукту.

Монолітна архітектура передбачає реалізацію всієї функціональності програмного продукту в єдиній структурі — одній великій програмі, яка компілюється та розгортається як єдине ціле. У цьому підході всі модулі тісно пов'язані між собою, працюють в одному процесі та мають спільну базу коду. Цей підхід є простішим у реалізації на початкових етапах розробки, забезпечує легшу інтеграцію між компонентами та менш витратний розгортальний процес. Однак, із зростанням проєкту виникають проблеми з масштабуванням, ускладнюється відлагодження та тестування окремих частин системи, а також підвищується ризик порушення стабільності всієї програми при внесенні змін до окремих модулів. Крім того, моноліт ускладнює роботу над системою кільком командам одночасно та гальмує впровадження нових технологій для окремих компонентів.

Натомість мікросервісна архітектура заснована на принципі поділу програми на набір автономних сервісів, кожен із яких реалізує окрему бізнес-функцію та взаємодіє з іншими сервісами через чітко визначені інтерфейси, переважно за допомогою мережевих протоколів. Кожен мікросервіс може мати власну базу даних, власний цикл життєдіяльності, і бути реалізованим навіть на

іншій мові програмування. Такий підхід забезпечує високу масштабованість, дозволяє швидше впроваджувати зміни, розгортати сервіси незалежно один від одного та використовувати окремі технології там, де вони найбільш доцільні. Водночас, мікросервіси потребують складної інфраструктури для керування сервісами, нагляду, логування, автентифікації та комунікації між компонентами. Ускладнюється також процес тестування та налагодження розподіленої системи.

Таким чином, монолітна архітектура є доцільною для невеликих або середніх проєктів із відносно стабільною функціональністю, де простота реалізації важливіша за гнучкість. У той час як мікросервісна архітектура є обґрунтованим вибором для складних, динамічно змінних систем із високими вимогами до масштабованості, незалежності команд і технологічної гнучкості.

У процесі розробки інформаційної управляючої системи особливу увагу необхідно приділяти механізмам автентифікації користувачів, оскільки саме вони забезпечують контроль доступу до функціональних компонентів системи, гарантують захист персональних даних і підтримують цілісність сеансу взаємодії. Для реалізації цих вимог у проєкті було обрано класичний підхід до автентифікації на основі cookie та сесій, що зарекомендував себе як надійний і ефективний у веб-розробці. Такий підхід дозволяє забезпечити сталу ідентифікацію користувача під час його взаємодії із системою, зберігаючи дані автентифікації на стороні клієнта у вигляді cookie та підтримуючи відповідні сесійні дані на сервері.

Використання cookie та сесій для автентифікації у веб-додатках має як свої переваги, так і певні недоліки, які важливо враховувати при проєктуванні безпечної та ефективною системи.

Серед основних переваг можна відзначити зручність зберігання та передачі ідентифікаційної інформації між клієнтом і сервером. Cookie дозволяють автоматично ідентифікувати користувача під час наступних запитів без необхідності повторного входу, що забезпечує безперервність сеансу та покращує користувацький досвід. Сесії, у свою чергу, дозволяють зберігати дані на сервері, що знижує ризик підробки або викрадення важливої інформації з боку клієнта. Вони також підтримують збереження складних об'єктів, таких як налаштування

користувача або тимчасові дані транзакцій, що не повинні зберігатися в cookie з міркувань безпеки.

Однак ці механізми мають і свої недоліки. Cookie є вразливими до атак типу XSS (міжсайтове скриптування), якщо веб-застосунок не реалізує відповідні механізми захисту, такі як HTTPOnly та Secure-атрибути. Крім того, обмеження на розмір cookie та їх кількість можуть створити труднощі у зберіганні великого обсягу даних. Що стосується сесій, вони вимагають серверних ресурсів для зберігання інформації про кожного активного користувача, що може бути обтяжливим при великій кількості одночасних сеансів. Також сесії можуть втрачатися при завершенні роботи сервера або бути видалені через тайм-аут, що вимагає ретельного адміністрування та контролю з боку розробників.

З огляду на ці фактори, поєднання cookie та сесій у проєкті дозволяє досягти балансу між зручністю, продуктивністю та безпекою, за умови дотримання сучасних рекомендацій щодо їхнього захищеного використання.

Аутентифікація на основі токенів є сучасним і широко використовуваним підходом до забезпечення доступу користувачів до веб-сервісів, особливо в розподілених і клієнт-серверних архітектурах. В основі цієї моделі лежить принцип передачі спеціального унікального маркера — токена — який видається користувачеві після успішної перевірки облікових даних і надалі використовується як доказ автентичності при кожному запиті до системи.

Найчастіше токени реалізуються у вигляді JSON Web Token (JWT), що є зашифрованим або підписаним об'єктом, який містить у собі основну інформацію про користувача та час дії маркера. Після входу в систему сервер генерує токен і передає його клієнтові, який зберігає його локально — у localStorage або sessionStorage — і додає до заголовка авторизації кожного запиту. Це дозволяє уникнути потреби зберігати стан сеансу на сервері, що є суттєвою перевагою при побудові масштабованих, безстанних (stateless) сервісів, зокрема REST API.

Перевагою токенової аутентифікації є її масштабованість, гнучкість та сумісність з багатьма платформами і протоколами. Вона чудово підходить для мобільних застосунків, односторінкових інтерфейсів (SPA) та мікросервісних

архітектур. Крім того, токени легко інтегруються з системами авторизації (наприклад, OAuth2), що дозволяє реалізовувати складні сценарії делегованого доступу та інтеграції зі сторонніми сервісами.

Водночас токени мають і певні виклики щодо безпеки. Зокрема, у разі викрадення токена (наприклад, через XSS [51] або уразливості клієнтського зберігання) зломисник отримає повний доступ до облікового запису, якщо не реалізовані додаткові захисні механізми — наприклад, перевірка IP-адреси, пристрою або обмеження часу дії. Також неможливість відкликати вже виданий токен (без додаткової реалізації на стороні сервера) створює потенційні ризики.

Після аналізу різних підходів до аутентифікації, зокрема заснованих на cookie, сесіях і токенах, доцільним є вибір аутентифікації на основі сесій для реалізації в межах даного проєкту. Цей спосіб є перевіреним, простим у впровадженні та добре підтримується більшістю фреймворків і серверних технологій, зокрема в екосистемі PHP. На відміну від токенів, сесії дозволяють зберігати стан на сервері, що підвищує рівень контролю за автентифікаційним процесом і спрощує реалізацію механізмів відкликання доступу, обмеження часу дії сеансу та додаткового захисту від атак типу replay або використання викрадених облікових даних. У контексті веб-додатку зі звичною архітектурою клієнт-сервер і обмеженою кількістю одночасних користувачів, сесійна модель забезпечує баланс між безпекою, зручністю й ефективністю, що робить її оптимальним вибором для реалізації механізму аутентифікації.

### **3.4 Технічне забезпечення**

#### **3.4.1 Обґрунтування вибору технічного забезпечення**

Розроблення інформаційної управляючої системи здійснювалося на персональному комп'ютері у вигляді ноутбука Lenovo IdeaPad Gaming, який забезпечує достатній рівень продуктивності для виконання завдань, пов'язаних із

проектуванням, програмуванням та тестуванням веб-систем. Обчислювальна платформа працює під керуванням операційної системи Windows 11, яка є сучасною, стабільною та зручною для розробника середовищем, що підтримує широкий спектр необхідного програмного забезпечення та інструментів. Обране технічне забезпечення дозволяло безперешкодно запускати локальні сервери, працювати з середовищем розробки, обробляти бази даних, проводити відлагодження та забезпечувати ефективну розробку всіх компонентів системи у відповідності до поставлених вимог. Зображення технічного забезпечення Lenovo IdeaPad Gaming наведено на рис. 3.1.



Рисунок 3.12 - Технічне забезпечення Lenovo IdeaPad Gaming

*Джерело [6]*

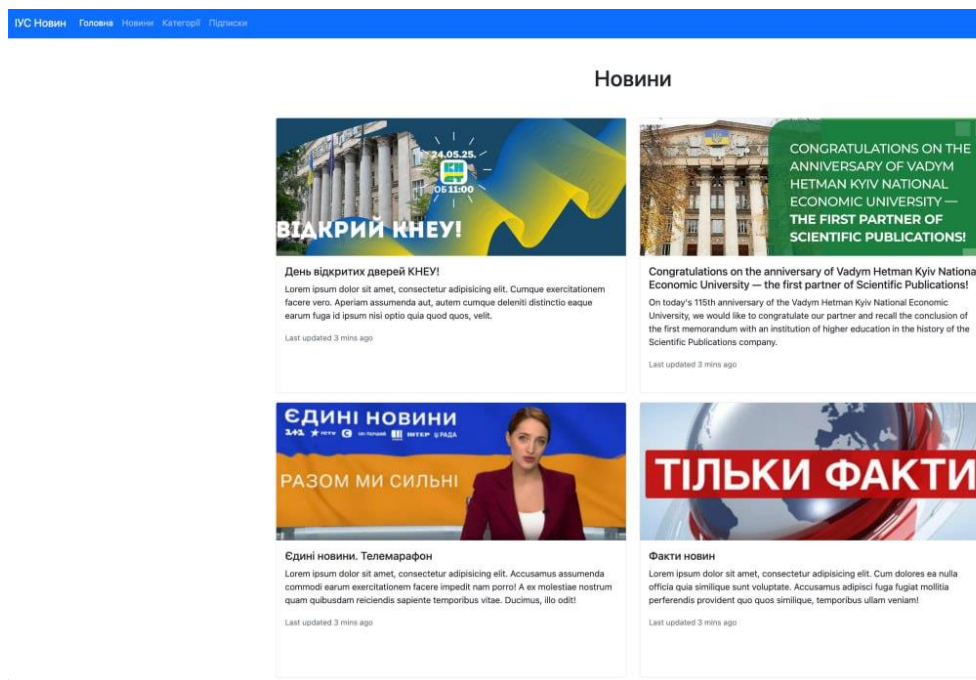
### **3.4.2 Ресурсні вимоги до технічного забезпечення**

Для забезпечення повноцінного функціонування розробленої інформаційної управляючої системи доцільно визначити мінімальні ресурсні вимоги до технічного забезпечення, які гарантують стабільну та ефективну роботу всіх її компонентів. З урахуванням архітектури системи, обраних технологій (зокрема, PHP, Laravel, MySQL, HTML/CSS, JavaScript), а також необхідності роботи веб-сервера та бази даних у фоновому режимі, мінімально рекомендована конфігурація комп'ютера має включати центральний процесор з не менш ніж чотирма фізичними ядрами (наприклад, Intel Core i5 або AMD Ryzen 5), оперативну пам'ять обсягом від 8 ГБ, твердотільний накопичувач (SSD) об'ємом не менше 256 ГБ для швидкого доступу до даних, а також стабільне підключення до мережі для обміну даними між клієнтською частиною та сервером. Операційна система має бути сучасною версією Windows, Linux або macOS з підтримкою віртуалізації, веб-серверів (Apache або Nginx), інтерпретатора PHP та СУБД MySQL або MariaDB. Така конфігурація забезпечить комфортне середовище як для розробки, так і для локального тестування та розгортання системи.

### **3.5 Реалізація інформаційної системи**

Результати функціонування створеної інформаційної управляючої системи демонструють її здатність ефективно автоматизувати процес формування та доставки персоналізованої стрічки новин з урахуванням заданих користувачем параметрів, актуальності повідомлень та вподобань аудиторії. Основними досягненнями реалізованого проєкту є динамічне оновлення контенту, можливість фільтрації інформації за різними критеріями, високий рівень інтерактивності користувацького інтерфейсу, а також адаптивність системи до розширення

функціональності завдяки використанню сучасних архітектурних рішень. Система забезпечує швидке реагування на зовнішні події та надає користувачу тільки релевантну інформацію, тим самим підвищуючи ефективність інформаційного обслуговування, Демонстрація головної сторінки ІУС новиноного сайту, наведено на рис 3.13



На рис 3.13 наведено Демонстрація головної сторінки

*Джерело: розроблено автором*

Розроблена інформаційна управляюча система може бути ефективно застосована в різних сферах, зокрема в медіа-агентствах, онлайн-ресурсах новин, корпоративних порталах, системах внутрішніх комунікацій підприємств, а також у мобільних додатках, що орієнтовані на персоналізовану подачу контенту. Завдяки модульності та масштабованості, система також може бути адаптована під потреби електронного урядування, освітніх платформ чи маркетингових аналітичних сервісів, де актуальність, точність і своєчасність інформаційних повідомлень є критично важливими. Наведено демонстрацію підписки та категорії новин на окремі новини , на рис 3.14-15

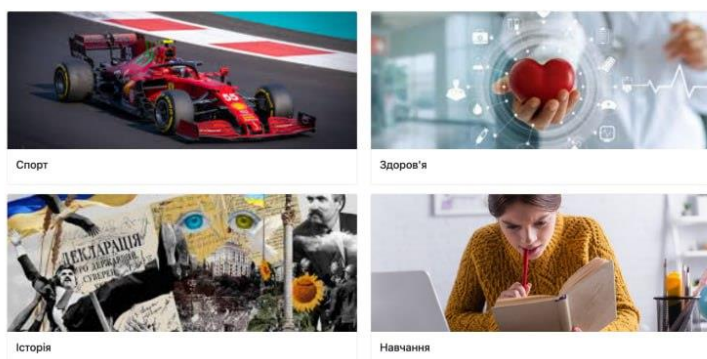
## Підписки



На рис 3.14 наведено демонстрацію підписки

*Джерело: розроблено автором*

## Категорії



На рис 3.15 наведено категорії новин на окремі новини

*Джерело: розроблено автором*

Можливості для подальшого покращення інформаційної управляючої системи відкривають перспективи її еволюції в напрямі розширення функціональності, підвищення точності персоналізації та вдосконалення технічних характеристик. Одним із пріоритетних напрямів розвитку є впровадження

елементів машинного навчання для формування індивідуальних рекомендацій на основі аналізу поведінкових шаблонів користувачів, що дозволить підвищити релевантність новинного контенту. Також доцільним є інтегрування модуля аналізу тональності текстів, який зможе класифікувати новини за емоційним забарвленням, надаючи користувачам змогу формувати більш збалансовану інформаційну картину.

З технічного боку система може бути масштабована для підтримки великої кількості одночасних користувачів шляхом переходу до хмарної інфраструктури або впровадження мікросервісної архітектури, що забезпечить підвищену гнучкість, відмовостійкість та продуктивність. Удосконалення інтерфейсу користувача за рахунок застосування сучасних фреймворків для фронтенд-розробки, таких як React або Vue.js, дозволить зробити взаємодію з системою більш інтуїтивною, зручною та привабливою з естетичної точки зору.

Крім того, перспективним є створення мобільної версії системи або кросплатформеного додатку, що розширить доступність сервісу та підвищить його конкурентоспроможність. З метою підвищення безпеки можуть бути реалізовані додаткові механізми захисту даних, включаючи двофакторну аутентифікацію та шифрування конфіденційної інформації. Таким чином, покращення системи має комплексний характер і може охоплювати як інтелектуальні, так і інфраструктурні аспекти її подальшого розвитку.

Наукова новизна розробленої інформаційної управляючої системи полягає у комплексному підході до автоматизації процесу формування персоналізованої стрічки новин з використанням сучасних архітектурних і технологічних рішень. Основним новаторським аспектом є інтеграція механізмів інтелектуального аналізу даних та фільтрації контенту на основі інтересів користувачів у контексті інформаційних управляючих систем, що дозволяє досягти високого рівня релевантності поданої інформації.

На відміну від традиційних підходів, система орієнтована не лише на пасивне відображення інформації, а й на активне управління інформаційними потоками на основі індивідуальних профілів користувачів, що формуються з

урахуванням їхньої поведінки, історії взаємодії та змістової оцінки новин. У системі реалізовано принципову підтримку багаторівневої взаємодії між модулями нотифікацій, зберіганням даних та механізмами персоналізації, що забезпечує як гнучкість, так і масштабованість.

Новизна також полягає у використанні сучасної архітектури програмного забезпечення, зокрема цибулевої архітектури, яка сприяє чіткому розділенню відповідальностей між логікою бізнес-процесів, обробкою даних та зовнішніми інтерфейсами. Такий підхід дозволяє ефективно адаптувати систему до змін у функціональних вимогах без значного рефакторингу коду.

Таким чином, наукова новизна системи виявляється у поєднанні архітектурних інновацій, технологій інтелектуального аналізу та орієнтації на автоматизоване інформаційне управління в умовах великого обсягу динамічних даних.

## ВИСНОВКИ

Узагальнюючи проведене дослідження, розробку та впровадження інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин, можна зробити висновок про її актуальність, ефективність та практичну значущість в умовах сучасного інформаційного середовища. Стрімке зростання обсягів цифрових новинних потоків зумовлює необхідність впровадження інтелектуальних систем, які здатні персоналізовано обробляти, аналізувати та доставляти користувачам релевантний контент у зручному форматі та в найкоротші терміни.

Розроблена система базується на сучасних програмних архітектурних підходах, таких як мікросервісна структура, використання фреймворку Laravel, принципів MVC, SOLID та чистого коду, що забезпечує гнучкість, масштабованість і високу якість програмного продукту. В процесі реалізації було здійснено аналіз і обґрунтування вибору технологій, мов програмування, підходів до зберігання даних, побудови бази знань та методів інтелектуального аналізу, зокрема через застосування таких інструментів, як WizWhy. Увага до питань аутентифікації користувачів за допомогою сесійного механізму, а також проектування ефективної системи нотифікацій дозволили підвищити безпеку, зручність та функціональність продукту.

Результатом виконаної роботи стало створення комплексної системи, що може бути адаптована до різних інформаційних платформ, зокрема новинних порталів, соціальних мереж або внутрішніх корпоративних інформаційних систем. Впровадження такої ІУС дозволяє не лише автоматизувати розповсюдження новин, але й забезпечує глибокий аналіз користувацької активності, що відкриває широкі можливості для подальшої персоналізації сервісу та стратегічного розвитку бізнесу.

Таким чином, розроблена система має високий потенціал для реального практичного застосування, демонструє наукову новизну, технічну досконалість і

відповідає сучасним вимогам до інформаційних технологій у сфері новинного контенту.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Metinvest Digital. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://metinvest.digital/ua/page/1052>.
2. Google News. Головна сторінка новин [Електронний ресурс]. – Режим доступу: <https://news.google.com/home?hl=uk&gl=UA&ceid=UA%3Auk>.
3. Facebook Newsroom. A new destination for news in the UK [Електронний ресурс]. – Режим доступу: <https://about.fb.com/news/2021/01/new-destination-for-news-in-the-uk/>.
4. Upday. Новини дня [Електронний ресурс]. – Режим доступу: <https://www.upday.com/uk>.
5. REICST. Модель рекомендаційної системи новин [Електронний ресурс]. – Режим доступу: <https://reicst.com.ua/pmt/article/view/2022-3-03-05>.
6. IBM Think. Content-Based Filtering [Електронний ресурс]. – Режим доступу: <https://www.ibm.com/think/topics/content-based-filtering>.
7. ResearchGate. Методи розв'язання задач багатокритеріальної оптимізації [Електронний ресурс]. – Режим доступу: <https://www.researchgate.net/publication/335701453>.
8. Flipboard. Про платформу [Електронний ресурс]. – Режим доступу: <https://about.flipboard.com/>.
9. MSN. Головна сторінка додатку [Електронний ресурс]. – Режим доступу: <https://msnapp.microsoft.com/msn>.
10. SmartNews. Інформаційна платформа новин [Електронний ресурс]. – Режим доступу: <https://www.smartnews.com/en>.
11. AboutMarketing. Чим корисне кешування сайту [Електронний ресурс]. – Режим доступу: <https://aboutmarketing.info/internet-marketynh/>.
12. Medium. Onion Architecture [Електронний ресурс]. – Режим доступу: <https://medium.com/expedia-group-tech/onion-architecture-deed8a554423>.

13. PHP.net. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://www.php.net/>.
14. Laravel. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://laravel.com/>.
15. JavaRush. Ознайомлення з патерном MVC [Електронний ресурс]. – Режим доступу: <https://javarush.com/ua/groups/posts/uk.2536>.
16. EPAM Campus. Блог [Електронний ресурс]. – Режим доступу: <https://campus.epam.ua/ua/blog/602>.
17. MySQL. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://www.mysql.com/>.
18. Redstone Agency. Що таке UX/UI дизайн? [Електронний ресурс]. – Режим доступу: <https://redstone.agency/blog/shcho-take-ux-ui-dysayn/>.
19. Art of BA. Основні типи архітектури ПЗ [Електронний ресурс]. – Режим доступу: <https://www.artofba.com/uk/post/main-types-of-software-architecture>.
20. KDE Umbrello. Основи UML [Електронний ресурс]. – Режим доступу: <https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html>.
21. SmartDraw. UML Activity Diagram – приклади [Електронний ресурс]. – Режим доступу: <https://www.smartdraw.com/uml-diagram/examples/uml-activity-diagram/>.
22. Max Zosim. State Modelling [Електронний ресурс]. – Режим доступу: <https://www.maxzosim.com/state-modelling/>.
23. QATestLab. Що таке use case і для чого він потрібен [Електронний ресурс]. – Режим доступу: <https://training.qatestlab.com/blog/technical-articles/what-is-a-use-case-and-what-are-they-for/>.
24. Max Zosim. Sequence Diagrams [Електронний ресурс]. – Режим доступу: <https://www.maxzosim.com/sequence-diagrams/>.
25. Visual Paradigm. What is Activity Diagram [Електронний ресурс]. – Режим доступу: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>.

26. Guru99. UML Activity Diagram [Електронний ресурс]. – Режим доступу: <https://www.guru99.com/uk/uml-activity-diagram.html>.
27. QA Evolution. Методологія RUP [Електронний ресурс]. – Режим доступу: <https://qaevolution.ru/metodologiya-menedzhment/rup/>.
28. Academia.edu. Байєсівські мережі в СППР [Електронний ресурс]. – Режим доступу: <https://www.academia.edu/107298205/>.
29. Max Zosim. Data Flow Diagrams [Електронний ресурс]. – Режим доступу: <https://www.maxzosim.com/data-flow-diagrams/>.
30. ЕБ ЛНТУ. Кондіус, стор. 9 [Електронний ресурс]. – Режим доступу: [https://elib.lntu.edu.ua/sites/default/files/elib\\_upload/](https://elib.lntu.edu.ua/sites/default/files/elib_upload/).
31. Studfile.net. Конспект лекцій (стор. 31) [Електронний ресурс]. – Режим доступу: <https://studfile.net/preview/5648144/>.
32. Mercy CX. Нечітка АНР [Електронний ресурс]. – Режим доступу: <https://grisha.mercy.cx.ua/shho-take-nechitka-ahp-z-vikoristannyam-metodu-serednogo-geometrichnogo/>.
33. Wedex. Алгоритм BERT [Електронний ресурс]. – Режим доступу: <https://wedex.com.ua/blog/algorithm-bert-shho-vmiye-ta-ne-vmiye-robiti/>.
34. Antibotan. Курсова робота (ID: 500762) [Електронний ресурс]. – Режим доступу: [https://antibotan.com/file.html?work\\_id=500762](https://antibotan.com/file.html?work_id=500762).
35. Solix. ACID-транзакції [Електронний ресурс]. – Режим доступу: <https://www.solix.com/uk/kb/acid-transactions/>.
36. PostgreSQL. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://www.postgresql.org/>.
37. Apix-Drive. Що таке JSON? [Електронний ресурс]. – Режим доступу: <https://apix-drive.com/ua/blog/useful/scho-take-json>.
38. Guru99. OLTP vs OLAP [Електронний ресурс]. – Режим доступу: <https://www.guru99.com/uk/oltp-vs-olap.html>.
39. QA Group. NoSQL: переваги та недоліки [Електронний ресурс]. – Режим доступу: <https://qagroup.com.ua/publications/nosql-perevagy-ta-nedoliky-nereliatcijnykh-baz-danykh/>.

40. Guru99. Що таке MongoDB? [Електронний ресурс]. – Режим доступу: <https://www.guru99.com/uk/what-is-mongodb.html>.
41. EPAM Careers. Графові бази даних [Електронний ресурс]. – Режим доступу: <https://careers.epam.ua/blog/graph-databases-revolutionizing-the-management-of-complex-relationships>.
42. Studfile.net. Навчальні матеріали (стор. 12) [Електронний ресурс]. – Режим доступу: <https://studfile.net/preview/2398157/page:12/>.
43. Google. Пошук: Чиста архітектура oJ wt [Електронний ресурс]. – Режим доступу: <https://www.google.com/search?q=Чиста+архітектура+oJ+wt>.
44. DOU.ua. Hexagonal Architecture [Електронний ресурс]. – Режим доступу: <https://dou.ua/lenta/articles/hexagonal-architecture-for-nodejs/>.
45. Mozilla Developer Network. JavaScript документація [Електронний ресурс]. – Режим доступу: <https://developer.mozilla.org/ru/docs/Web/JavaScript>.
46. TypeScript. Офіційний сайт [Електронний ресурс]. – Режим доступу: <https://www.typescriptlang.org/>.
47. Visual Studio Code. Редактор коду [Електронний ресурс]. – Режим доступу: <https://code.visualstudio.com/>.
48. NetBeans. IDE NetBeans [Електронний ресурс]. – Режим доступу: <https://netbeans.apache.org/front/main/index.html>.
49. Medium. Чистий код: 3 основні принципи [Електронний ресурс]. – Режим доступу: <https://medium.com/@greeflas/чистий-код-3-основні-принципи-3acf5a0470c4>.
50. Symfony.com.ua. Документація Doctrine [Електронний ресурс]. – Режим доступу: <https://symfony.com.ua/doc/current/doctrine.html>.
51. VPN Unlimited. Що таке XSS [Електронний ресурс]. – Режим доступу: <https://www.vpnunlimited.com/ua/help/cybersecurity/xss>.

## ДОДАТКИ

### Додаток А

Джерело: [ <https://news.google.com/home?hl=uk&gl=UA&ceid=UA%3Auk>]

Google News

Джерело: [<https://about.flipboard.com/blog/page/1>] Flipboard

Джерело: [ [https://msnapp.microsoft.com/msn?adjust=10b9f62c\\_10rdcyvv](https://msnapp.microsoft.com/msn?adjust=10b9f62c_10rdcyvv)]

Microsoft Start

Джерело: [ <https://about.fb.com/news/2021/03/more-control-and-context-in-news-feed/>] Facebook News Feed

Джерело: [<https://www.smartnews.com/en>] SmartNews

Джерело: [<https://m6hyxhcitg.execute-api.eu-central-1.amazonaws.com/prod/storage/app/uploads/public/65a/903/e25/65a903e25977e110344929.jpeg?format=jpeg>] amazonaws

<?php

**namespace App\Controller;**

**use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;**

**use Symfony\Component\HttpFoundation\Response;**

**use Symfony\Component\Routing\Attribute\Route;**

**class HomeController extends AbstractController**

**{**

**#[Route('/', name: 'app\_home')]**

**public function index(): Response**

**{**

**return \$this->render('home.html.twig', [  
    'controller\_name' => 'HomeController',**

**]);**

**}**

```
#[Route('/news', name: 'app_news')]  
public function news(): Response
```

```

{
    return $this->render('home.html.twig', [
        'controller_name' => 'HomeController',
    ]);
}

#[Route('/category', name: 'app_category')]
public function category(): Response
{
    return $this->render('category.html.twig', [
        'controller_name' => 'HomeController',
    ]);
}

#[Route('/subscription', name: 'app_subscription')]
public function subscription(): Response
{
    return $this->render('subscription.html.twig', [
        'controller_name' => 'HomeController',
    ]);
}
}
<?php

```

```

namespace App\Entity;

```

```

use App\Repository\CategoryRepository;
use Doctrine\ORM\Mapping as ORM;

```

```

#[ORM\Entity(repositoryClass: CategoryRepository::class)]
class Category
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column]
    private string $name;
    #[ORM\Column]
    private string $description;
    #[ORM\Column]
    private int $parentId;
}

```

```

public function getId(): ?int
{
    return $this->id;
}

public function getName(): string
{
    return $this->name;
}

public function setName(string $name): void
{
    $this->name = $name;
}

public function getDescription(): string
{
    return $this->description;
}

public function setDescription(string $description): void
{
    $this->description = $description;
}

public function getParentId(): int
{
    return $this->parentId;
}

public function setParentId(int $parentId): void
{
    $this->parentId = $parentId;
}
}
<?php

namespace App\Entity;

use App\Repository\NewsRepository;
use Doctrine\ORM\Mapping as ORM;

#[ORM\Entity(repositoryClass: NewsRepository::class)]
class News

```

```

{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column]
    private string $title;
    #[ORM\Column]
    private string $content;
    #[ORM\Column]
    private int $publishedAt;
    #[ORM\Column]
    private string $status;
    #[ORM\Column]
    private int $authorId;
    #[ORM\Column]
    private int $categoryId;
    #[ORM\Column]
    private int $createdAt;
    #[ORM\Column]
    private int $updatedAt;

    public function getId(): ?int
    {
        return $this->id;
    }

    public function getTitle(): string
    {
        return $this->title;
    }

    public function setTitle(string $title): void
    {
        $this->title = $title;
    }

    public function getContent(): string
    {
        return $this->content;
    }

    public function setContent(string $content): void

```

```
{
    $this->content = $content;
}

public function getPublishedAt(): int
{
    return $this->publishedAt;
}

public function setPublishedAt(int $publishedAt): void
{
    $this->publishedAt = $publishedAt;
}

public function getStatus(): string
{
    return $this->status;
}

public function setStatus(string $status): void
{
    $this->status = $status;
}

public function getAuthorId(): int
{
    return $this->authorId;
}

public function setAuthorId(int $authorId): void
{
    $this->authorId = $authorId;
}

public function getCategoryId(): int
{
    return $this->categoryId;
}

public function setCategoryId(int $categoryId): void
{
    $this->categoryId = $categoryId;
}
```

```

public function getCreatedAt(): int
{
    return $this->createdAt;
}

public function setCreatedAt(int $createdAt): void
{
    $this->createdAt = $createdAt;
}

public function getUpdatedAt(): int
{
    return $this->updatedAt;
}

public function setUpdatedAt(int $updatedAt): void
{
    $this->updatedAt = $updatedAt;
}
}
<?php

```

```

namespace App\Entity;

```

```

use App\Repository\RoleRepository;
use Doctrine\ORM\Mapping as ORM;

```

```

#[ORM\Entity(repositoryClass: RoleRepository::class)]
class Role
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    public function getId(): ?int
    {
        return $this->id;
    }

    #[ORM\Column]
    public string $name;

    public function getName(): string

```

```

    {
        return $this->name;
    }

    public function setName(string $name): self
    {
        $this->name = $name;
        return $this;
    }
}
<?php

```

```

namespace App\Entity;

```

```

use App\Repository\SubscriptionRepository;
use Doctrine\ORM\Mapping as ORM;

```

```

#[ORM\Entity(repositoryClass: SubscriptionRepository::class)]
class Subscription

```

```

{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column]
    private int $userId;
    #[ORM\Column]
    private int $categoryId;
    #[ORM\Column]
    private string $subscriptionAt;

```

```

    public function getId(): ?int
    {
        return $this->id;
    }

```

```

    public function getUserId(): int
    {
        return $this->userId;
    }

```

```

    public function setUserId(int $userId): void
    {

```

```

        $this->userId = $userId;
    }

    public function getCategoryId(): int
    {
        return $this->categoryId;
    }

    public function setCategoryId(int $categoryId): void
    {
        $this->categoryId = $categoryId;
    }

    public function getSubscriptionAt(): string
    {
        return $this->subscriptionAt;
    }

    public function setSubscriptionAt(string $subscriptionAt): void
    {
        $this->subscriptionAt = $subscriptionAt;
    }
}
<?php

```

```

namespace App\Entity;

```

```

use App\Enum\RoleEnum;
use App\Repository\UserRepository;
use Doctrine\ORM\Mapping as ORM;
use
Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
use Symfony\Component\Security\Core\User\UserInterface;
use Symfony\Component\Serializer\Annotation\SerializedName;
use Symfony\Component\Validator\Constraints as Assert;

```

```

#[ORM\Entity(repositoryClass: UserRepository::class)]
class User implements UserInterface, PasswordAuthenticatedUserInterface
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

```

```
#[ORM\Column]  
private string $email;
```

```
#[ORM\Column]  
private string $phone;
```

```
#[ORM\ManyToOne(targetEntity: Role::class, cascade: ['persist'])]  
private Role $role;
```

```
#[ORM\Column]  
private string $name;
```

```
#[ORM\Column]  
private string $lastname;
```

```
#[ORM\Column(nullable: true)]  
private string $password;
```

```
#[SerializedName('password')]  
#[Assert\NotBlank]  
private string $plainPassword;
```

```
public function getId(): ?int  
{  
    return $this->id;  
}
```

```
public function getPassword(): ?string  
{  
    return $this->password;  
}
```

```
public function getRoles(): array  
{  
    $roles[] = $this->getRole()?->getName() ?? RoleEnum::USER->value;  
  
    return array_unique($roles);  
}
```

```
public function getRole(): ?Role  
{  
    return $this->role;  
}
```

```

public function setRole(Role $role): self
{
    $this->role = $role;

    return $this;
}

/**
 * @see UserInterface
 */
public function eraseCredentials(): void
{
    $this->plainPassword = null;
}

/**
 * @see UserInterface
 */
public function getUserIdentifier(): string
{
    return (string)$this->email;
}

public function getEmail(): string
{
    return $this->email;
}

public function setEmail(string $email): self
{
    $this->email = $email;
    return $this;
}

public function getPhone(): string
{
    return $this->phone;
}

public function setPhone(string $phone): self
{
    $this->phone = $phone;
    return $this;
}

```

```
public function getName(): string
{
    return $this->name;
}

public function setName(string $name): self
{
    $this->name = $name;
    return $this;
}

public function getLastname(): string
{
    return $this->lastname;
}

public function setLastname(string $lastname): self
{
    $this->lastname = $lastname;
    return $this;
}

public function getPlainPassword(): string
{
    return $this->plainPassword;
}

public function setPlainPassword(string $plainPassword): self
{
    $this->plainPassword = $plainPassword;
    return $this;
}
}
```



## Звіт подібності

### метадані

Назва організації

Kyiv National Economic University named after Vadym Hetman KNEU

Заголовок

Розроблення інформаційної управляючої системи для автоматизації рекомендаційної стрічки новин

Автор

Науковий керівник / Експерт

Базюк Іван ВікторовичМозгаллі О.П.

підрозділ

кафедра інформаційних систем в економіці

### Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

25

Довжина фрази для коефіцієнта подібності 2.



КП 2

16902

Кількість слів



КЦ

139308

Кількість символів

### Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

|                        |  |   |
|------------------------|--|---|
| Заміна букв            |  | 0 |
| Інтервали              |  | 0 |
| Мікропробіли           |  | 0 |
| Білі знаки             |  | 0 |
| Парафрази (SmartMarks) |  | 8 |

### Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Копір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

| 10 найдовших фраз   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Копір тексту                              |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| ПОРЯДКОВИЙ<br>НОМЕР | НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)                                                                                                                                                                                                                                                                                                                                                                                                                                          | КІЛЬКІСТЬ ІДЕНТИЧНИХ<br>СЛІВ (ФРАГМЕНТІВ) |
| 1                   | <a href="https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download">https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download</a>                                                                                                                                                                                                                                                                                                     | 43 0.25 %                                 |
| 2                   | <a href="https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download">https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download</a>                                                                                                                                                                                                                                                                                                     | 25 0.15 %                                 |
| 3                   | <a href="https://www.studocu.com/uk/document/kiivs%D1%8Ckiy-natsional%D1%8Cniy-ekonomichnij-universitet-imeni-vadima-get%D1%8Cman/finansi-bankivs%D1%8Cka-sprava-ta-strakhuvannya/individual%D1%8Cni-investori-na-fondovomu-rinku/78336831">https://www.studocu.com/uk/document/kiivs%D1%8Ckiy-natsional%D1%8Cniy-ekonomichnij-universitet-imeni-vadima-get%D1%8Cman/finansi-bankivs%D1%8Cka-sprava-ta-strakhuvannya/individual%D1%8Cni-investori-na-fondovomu-rinku/78336831</a> | 21 0.12 %                                 |
| 4                   | <a href="https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download">https://ir.kneu.edu.ua/bitstreams/725b64da-359e-479b-b2e4-50ed2d83d86c/download</a>                                                                                                                                                                                                                                                                                                     | 20 0.12 %                                 |

## **КАНАЛИ РЕКОМЕНДАЦІЙНОЇ СТРІЧКИ ІНФОРМАЦІЙНОЇ УПРАВЛЯЮЧОЇ СИСТЕМИ ДЛЯ АВТОМАТИЗАЦІЇ СТРІЧКИ НОВИН**

**Базюк Іван Вікторович**

Здобувач вищої освіти інституту  
інформаційних технологій в економіці  
*Київський національний економічний  
університет імені Вадима Гетьмана,  
Україна*

**Науковий керівник: Мозгалі О.П.**

**д.е.н., професор**  
*Київський національний економічний  
університет імені Вадима Гетьмана,  
Україна*

У сучасних умовах стрімкого зростання обсягів інформації особливої актуальності набуває проблема ефективної організації доступу користувачів до релевантного контенту. Інформаційно управляючі системи для автоматизації стрічки новин покликані забезпечити персоналізовану доставку інформації шляхом використання рекомендаційних технологій. Центральне місце в архітектурі таких систем займають канали рекомендаційної стрічки — механізми, що визначають принципи збору, обробки та подачі новинного контенту відповідно до інтересів користувача. Розробка ефективних каналів рекомендаційної стрічки дозволяє не лише оптимізувати інформаційні потоки, а й сприяти підвищенню точності, швидкості та гнучкості взаємодії системи з користувачем. Дослідження підходів до побудови та функціонування таких каналів є необхідним етапом для створення сучасних медіа-платформ, здатних оперативно реагувати на зміну інформаційних потреб аудиторії [1].

Інформаційно управляюча система для автоматизації стрічки новин — це комплексне програмно-апаратне рішення, яке виконує збір, обробку, аналіз та інтелектуальну подачу новинного контенту на основі

індивідуальних характеристик користувача або заданих параметрів. Такі системи працюють за принципом безперервного управління інформаційним потоком: вони самостійно отримують дані з різних джерел (новинних агентств, сайтів, соціальних мереж), фільтрують їх за заданими критеріями, ранжують за ступенем актуальності чи важливості та формують персоналізовану стрічку новин [2].

Ключовою особливістю інформаційно управляючої системи є здатність не лише зберігати й передавати інформацію, а й приймати рішення про те, які новини і в якій послідовності має побачити користувач. Це рішення базується на аналізі великих обсягів даних: історії переглядів, уподобань, кліків, соціально-демографічних характеристик, а також трендових подій. Використовуючи методи машинного навчання, системи вдосконалюють свої рекомендаційні механізми, що дозволяє їм з часом краще розуміти інтереси користувача і пропонувати йому все більш релевантний контент [3].

Важливою складовою такої системи є рекомендаційні алгоритми, які можуть бути контентно-орієнтованими, колаборативно-фільтраційними або гібридними. Контентно-орієнтовані системи аналізують властивості новин, щоб знайти схожі матеріали, що вже зацікавили користувача. Колаборативна фільтрація базується на пошуку користувачів зі схожими інтересами й рекомендації контенту, який вони оцінили високо. Гібридні підходи поєднують обидва методи для досягнення кращої точності [4].

У процесі розробки та впровадження інформаційно управляючих систем для автоматизації стрічки новин важливо враховувати як їхні сильні сторони, так і можливі обмеження (табл. 1).

Таблиця 1

**Переваги і недоліки інформаційно управляючої системи для  
автоматизації стрічки новин**

| Переваги                                                 | Недоліки                                                           |
|----------------------------------------------------------|--------------------------------------------------------------------|
| Персоналізована подача новинного контенту                | Вірогідність формування інформаційних бульбашок                    |
| Фільтрація та ранжування новин за інтересами користувача | Потенційна упередженість алгоритмів                                |
| Оперативна обробка та аналіз великого обсягу даних       | Залежність від якості використаних даних                           |
| Адаптація рекомендацій до змін вподобань користувача     | Проблеми із захистом персональних даних користувачів               |
| Зниження інформаційного перевантаження                   | Складність забезпечення прозорості та пояснюваності роботи системи |

На основі проведеного аналізу можна зробити висновок, що інформаційно управляюча система для автоматизації стрічки новин є потужним інструментом, здатним значно покращити якість і швидкість взаємодії користувача з інформаційним середовищем. Завдяки персоналізації, інтелектуальній фільтрації та здатності адаптуватися до змін інтересів аудиторії, такі системи підвищують релевантність контенту та знижують рівень інформаційного перевантаження. Утім, поряд із перевагами існують і серйозні виклики — це ризик створення інформаційних бульбашок, загрози конфіденційності, а також складність забезпечення прозорості роботи алгоритмів. Отже, успішне впровадження подібних систем вимагає комплексного підходу, що поєднує технічну досконалість із етичними принципами, забезпеченням безпеки даних і збереженням інформаційної відкритості для користувачів.

### **Список використаних джерел**

1. Бутенко Т. А., Сирий В. М. Інформаційні системи та технології.. Харків : ХНАУ ім. В. В. Докучаєва, 2020.
2. Горбатенко Д. В. Рекомендаційні системи, що базуються на обмеженнях : дис. ... канд. техн. наук : 122 – Комп'ютерні науки / Нац. ун-т «Одеська політехніка». Одеса, 2024.
3. Бузінська М. Р. Рекомендаційна система з підбору фільмів за допомогою методів машинного навчання : курс. роб. / ЧНУ ім. Петра Могили. Миколаїв, 2022.
4. Wang R., Liesaputra V., Huang Z. A Survey on LLM-based News Recommender Systems : / R. Wang, V. Liesaputra, Z. Huang /. 2025.

## **ІНФОРМАЦІЙНО УПРАВЛЯЮЧА СИСТЕМА ДЛЯ АВТОМАТИЗАЦІЇ РЕКОМЕНДАЦІЙНОЇ СТРІЧКИ НОВИН**

**Базюк Іван Вікторович**

Здобувач вищої освіти інституту  
інформаційних технологій в економіці  
*Київський національний економічний  
університет імені Вадима Гетьмана,  
Україна*

**Науковий керівник: Мозгалі О.П.**

**д.е.н., професор**  
*Київський національний економічний  
університет імені Вадима Гетьмана,  
Україна*

Системи рекомендацій стрічки новин — це спеціалізовані інформаційні технології, що автоматично добирають та подають новинний контент користувачам відповідно до їхніх інтересів, поведінки й уподобань. Такі системи стали надзвичайно важливими в епоху інформаційного перенасичення, коли кількість новин перевищує можливості людини самотійно їх відфільтрувати й опрацювати. Основне завдання рекомендаційних систем — підвищити релевантність контенту для кожного конкретного користувача, забезпечити його постійною актуальною інформацією, мінімізувати непотрібний інформаційний шум і покращити загальний досвід споживання новин [1].

У своїй роботі системи рекомендацій зазвичай використовують кілька підходів. Контентно-орієнтовані методи аналізують характеристики самих новинних матеріалів і намагаються знайти статті, схожі на ті, які користувач читав або оцінював раніше. Колаборативна фільтрація базується на аналізі поведінки інших користувачів: система пропонує новини, які були популярні серед людей зі схожими інтересами. Гібридні

підходи поєднують обидва ці методи, прагнучи підвищити точність прогнозування і подолати слабкі місця кожного окремого методу [2].

Технічно рекомендаційні системи працюють за допомогою обробки великих обсягів даних: вони збирають інформацію про перегляди, кліки, лайки, час читання статей, демографічні особливості користувачів, аналізують текст новин за допомогою методів обробки природної мови, виявляють тематику й настрій повідомлень. Моделі машинного навчання, такі як нейронні мережі, деревні ансамблі або методи факторизації матриць, дозволяють будувати індивідуальні профілі користувачів та постійно оновлювати їх відповідно до нової активності.

Однак розробка таких систем супроводжується серйозними викликами. Одним з найважливіших є ризик створення так званих "інформаційних бульбашок", коли користувач отримує лише ті новини, що підтверджують його погляди, що обмежує доступ до альтернативних точок зору. Іншим викликом є упередженість алгоритмів, що може призвести до некоректної фільтрації інформації. Також важливо враховувати етичні питання прозорості роботи системи і захисту особистих даних користувачів [3].

Рекомендаційні системи новин не лише змінюють спосіб, у який ми споживаємо інформацію, але й мають значний вплив на формування громадської думки та соціальну динаміку. Саме тому створення ефективних, етичних та збалансованих систем є сьогодні одним із пріоритетних завдань для розробників у галузі штучного інтелекту та інформаційних технологій [4].

Для кращого розуміння ролі автоматизації у побудові рекомендаційної стрічки новин доцільно проаналізувати її основні переваги та недоліки (табл. 1).

Таблиця 1

**Переваги та недоліки автоматизації рекомендаційної стрічки новин**

| Переваги                                                              | Недоліки                                                                                               |
|-----------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Підвищення швидкості та точності добору новинного контенту            | Ризик створення інформаційних бульбашок, обмеження доступу до альтернативних точок зору                |
| Персоналізовані рекомендації без додаткових зусиль з боку користувача | Складність пояснення логіки роботи алгоритмів користувачеві                                            |
| Економія часу користувача, зменшення інформаційного перевантаження    | Зниження рівня довіри до системи через непрозорість роботи алгоритмів                                  |
| Залучення та утримання аудиторії на медіаплатформах                   | Проблеми з захистом персональних даних через необхідність збору великої кількості особистої інформації |

Отже у висновку, автоматизація рекомендаційної стрічки новин є важливим кроком у розвитку сучасних інформаційних технологій, що дозволяє значно підвищити ефективність обробки великих обсягів даних і забезпечити персоналізований доступ до контенту. Водночас вона породжує серйозні виклики, пов'язані із ризиком формування інформаційних бульбашок, алгоритмічною упередженістю та питаннями захисту персональних даних користувачів. Тому подальший розвиток таких систем має базуватися на балансі між технічними можливостями, етичними принципами та потребами суспільства у відкритому й різноманітному інформаційному середовищі.

**Список використаних джерел**

1. Бойко О. Д. Інформаційні системи та технології в управлінні. Київ : КНЕУ, 2020.
2. Клименко О. В. Інформаційні технології в медіасфері: персоналізація контенту та рекомендаційні системи. Київ : КНУКіМ, 2019.
3. Козловський С. В., Дерев'янка В. М. Аналіз алгоритмів рекомендаційних систем у середовищі новинних порталів. Східноєвропейський журнал передових технологій, 2021.
4. Грабовський Ю. Рекомендаційні системи в інформаційних потоках: загрози інформаційної ізоляції. Наукові записки Інституту журналістики, 2022.