

## Метадані

### ДОКУМЕНТ

Заголовок

2026\_Неймет\_В.Д.\_ФІТ\_ІТТС\_ІСТ-22-1

Автор

Неймет В.Д.

Науковий керівник / Експерт

Паньків Ю. В.

ІД документу

334290917

### ОРГАНІЗАЦІЯ

Назва організації

Ivano-Frankivsk National Technical University of Oil and Gas

підрозділ

Каф. ІТТС

### ЗВІТ

Дата звіту

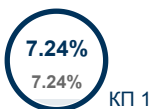
6/12/2026

Дата редагування

---

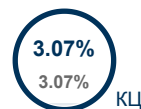
## Обсяг знайдених подібностей

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.



10960

Кількість слів



86770

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             |  | 12 |
| Парафрази (SmartMarks) |  | 42 |

## Джерела

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір 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

Колір тексту

| # | НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)                                                                                                                                    | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ<br>(ФРАГМЕНТІВ) |
|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| 1 | <a href="https://eir.zp.edu.ua/bitstreams/f3e775fd-281b-4cc3-b2bd-610ea1436360/download">https://eir.zp.edu.ua/bitstreams/f3e775fd-281b-4cc3-b2bd-610ea1436360/download</a> | 20 (0.18 %)                               |

|    |                                                                                                                                                                                                           |             |
|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 2  | <a href="https://eir.zp.edu.ua/server/api/core/bitstreams/a15e16f6-938f-4074-8623-46875d55c897/content">https://eir.zp.edu.ua/server/api/core/bitstreams/a15e16f6-938f-4074-8623-46875d55c897/content</a> | 20 (0.18 %) |
| 3  | 2026_Грицак_О.О._ФІТ_ІТТС_ІСТ-22-1<br>6/11/2026<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. ІТТС)                                                                               | 18 (0.16 %) |
| 4  | <a href="http://repository.dnu.dp.ua:1100/upload/23e81591ae1e1ef632a78cc3d2215e5bDR_BAKALAVR.PDF">http://repository.dnu.dp.ua:1100/upload/23e81591ae1e1ef632a78cc3d2215e5bDR_BAKALAVR.PDF</a>             | 18 (0.16 %) |
| 5  | <a href="http://ir.polissiauniver.edu.ua/bitstream/123456789/14056/1/SUKHANOV_VA_KR_126_2023.pdf">http://ir.polissiauniver.edu.ua/bitstream/123456789/14056/1/SUKHANOV_VA_KR_126_2023.pdf</a>             | 17 (0.16 %) |
| 6  | <a href="https://repo.nung.edu.ua/bitstreams/a8d2a041-83a0-44d8-91ce-ba350168d715/download">https://repo.nung.edu.ua/bitstreams/a8d2a041-83a0-44d8-91ce-ba350168d715/download</a>                         | 17 (0.16 %) |
| 7  | <a href="https://dglib.nubip.edu.ua/bitstreams/88f85c39-7c12-4c53-a21c-61f022f14d7c/download">https://dglib.nubip.edu.ua/bitstreams/88f85c39-7c12-4c53-a21c-61f022f14d7c/download</a>                     | 15 (0.14 %) |
| 8  | 2023_81210001_Iaremko_Sviatoslav_Bohdanovych_216124<br>10/26/2024<br>National University "Lviv Politechnika" (National University Lviv Politechnika)                                                      | 15 (0.14 %) |
| 9  | БР_КІ-22-1_МОКРЕЦЬКА А.Р.<br>6/9/2026<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. КСМ)                                                                                          | 15 (0.14 %) |
| 10 | БР_КІ-22-1_МОКРЕЦЬКА А.Р.<br>6/9/2026<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. КСМ)                                                                                          | 14 (0.13 %) |

#### Домашня база даних (1.21 %)



| # | ЗАГОЛОВОК                                                                                                                                                                                                              | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ<br>(ФРАГМЕНТІВ) |
|---|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| 1 | 2026_Грицак_О.О._ФІТ_ІТТС_ІСТ-22-1<br>6/11/2026<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. ІТТС)                                                                                            | 57 (7) (0.52 %)                           |
| 2 | БР_КІ-22-1_МОКРЕЦЬКА А.Р.<br>6/9/2026<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. КСМ)                                                                                                       | 36 (3) (0.33 %)                           |
| 3 | БР_Шістка_Святослав<br>6/6/2025<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. КСМ)                                                                                                             | 28 (3) (0.26 %)                           |
| 4 | 2021_Магістерська_ІПЗз_Яремін.docx / Моделі, методи та алгоритми новітніх імплементацій<br>технології «клієнт-сервер»<br>12/13/2021<br>Ivano-Frankivsk National Technical University of Oil and Gas (Каф. ІПЗ Магістр) | 12 (1) (0.11 %)                           |

#### Програма обміну базами даних (2.53 %)



| # | ЗАГОЛОВОК                                                                                                                                                                                                     | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ<br>(ФРАГМЕНТІВ) |
|---|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| 5 | ДП_Пі_192_Тартаковський.docx<br>11/23/2023<br>Khmelnyskyi Polytechnic Professional College (Відділення програмної інженерії)                                                                                  | 65 (7) (0.59 %)                           |
| 6 | Розробка програмного забезпечення для вебсервісу оголошень<br>10/27/2024<br>National University of Shipbuilding named after Admiral Makarov (National University of Shipbuilding named after Admiral Makarov) | 23 (3) (0.21 %)                           |
| 7 | Розроблення вебзастосунку для обліку витрат користувача<br>5/23/2026<br>Kharkiv National University of Economics named after S.Kuznets (KNUE) (KNUE)                                                          | 22 (3) (0.2 %)                            |

|    |                                                                                                                                                                                                                                                                      |                 |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| 8  | 2023_81210001_laremko_Sviatoslav_Bohdanovych_216124<br>10/26/2024<br>National University "Lviv Politechnika" (National University Lviv Politechnika)                                                                                                                 | 21 (2) (0.19 %) |
| 9  | NurStore Web сайтын жасау<br>5/30/2024<br>Satbayev University (ИАиИТ)                                                                                                                                                                                                | 21 (3) (0.19 %) |
| 10 | ФМФІТ магістр 2025<br>12/16/2025<br>Odessa I.I.Mechnikov National University (Одеський національний університет імені І.І.Мечникова)                                                                                                                                 | 19 (3) (0.17 %) |
| 11 | WEB-портал на кіберспортивну тематику<br>3/15/2025<br>National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)                                              | 17 (2) (0.16 %) |
| 12 | ФКНТ_2025_122_ВарениковАВ<br>6/4/2025<br>Ukrainian national aviation university (ФКНТ Кафедра комп'ютерних інформаційних технологій)                                                                                                                                 | 15 (2) (0.14 %) |
| 13 | 123_Kolyesnyk_Oleksiy_Oleksiyovych<br>6/27/2024<br>Odessa I.I.Mechnikov National University (Odessa I.I.Mechnikov National University)                                                                                                                               | 14 (2) (0.13 %) |
| 14 | KRISHU_DIPLOM_PA1-192.docx<br>6/18/2021<br>Odessa National Polytechnic University (ІКС, кафедра інформ. систем)                                                                                                                                                      | 13 (2) (0.12 %) |
| 15 | Диплом_Турбай_2025 (21)<br>12/16/2025<br>Poltava State Agrarian Academy (PSAA) (Deanary)                                                                                                                                                                             | 11 (1) (0.1 %)  |
| 16 | КП Богданович<br>12/10/2025<br>Subdivision "Kamianets-Podilskyi Professional College of Food Industry, National University of Food Technologies" (Subdivision "Kamianets-Podilskyi Professional College of Food Industry, National University of Food Technologies") | 10 (1) (0.09 %) |
| 17 | ФККПІ_2021_125_КадировДС_виправл_1<br>7/11/2024<br>Ukrainian national aviation university (Ukrainian national aviation university)                                                                                                                                   | 7 (1) (0.06 %)  |
| 18 | Крамаренко<br>6/16/2024<br>Bohdan Khmelnytsky National University of Cherkasy (Bohdan Khmelnytsky National University of Cherkasy)                                                                                                                                   | 7 (1) (0.06 %)  |
| 19 | Програмний комплекс для відділу кадрів викладачів. Курсова робота<br>6/23/2025<br>Zhytomyr Agricultural Technical Professional College (Zhytomyr Agricultural Technical Professional College)                                                                        | 6 (1) (0.05 %)  |
| 20 | Hryshchenko_bac_rob.doc.docx<br>6/8/2023<br>Sumy State University (Кафедра комп'ютерних наук)                                                                                                                                                                        | 6 (1) (0.05 %)  |

## Інтернет (3.49 %)



| #  | ДЖЕРЕЛО URL                                                                                                                                                                                   | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ<br>(ФРАГМЕНТІВ) |
|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| 21 | <a href="https://dglip.nubip.edu.ua/bitstreams/88f85c39-7c12-4c53-a21c-61f022f14d7c/download">https://dglip.nubip.edu.ua/bitstreams/88f85c39-7c12-4c53-a21c-61f022f14d7c/download</a>         | 66 (6) (0.6 %)                            |
| 22 | <a href="http://repository.dnu.dp.ua:1100/upload/23e81591ae1e1ef632a78cc3d2215e5bDR_BAKALAVR.PDF">http://repository.dnu.dp.ua:1100/upload/23e81591ae1e1ef632a78cc3d2215e5bDR_BAKALAVR.PDF</a> | 52 (5) (0.47 %)                           |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                 |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| 23 | <a href="https://owasp.org/www-pdf-archive/ASVS_3_0_Ukrainian_Beta.pdf">https://owasp.org/www-pdf-archive/ASVS_3_0_Ukrainian_Beta.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 30 (4) (0.27 %) |
| 24 | <a href="https://eir.zp.edu.ua/server/api/core/bitstreams/a15e16f6-938f-4074-8623-46875d55c897/content">https://eir.zp.edu.ua/server/api/core/bitstreams/a15e16f6-938f-4074-8623-46875d55c897/content</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 28 (2) (0.26 %) |
| 25 | <a href="https://github.com/indiser/The-Fake-Shop/compare/indiser:fa97b6f...indiser:7b6e7ec.diff">https://github.com/indiser/The-Fake-Shop/compare/indiser:fa97b6f...indiser:7b6e7ec.diff</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 28 (4) (0.26 %) |
| 26 | <a href="https://www.djangoproject.com/community/q-and-a/?page=383">https://www.djangoproject.com/community/q-and-a/?page=383</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | 22 (3) (0.2 %)  |
| 27 | <a href="http://ir.polissiauniver.edu.ua/bitstream/123456789/14056/1/SUKHANOVA_KR_126_2023.pdf">http://ir.polissiauniver.edu.ua/bitstream/123456789/14056/1/SUKHANOVA_KR_126_2023.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 22 (2) (0.2 %)  |
| 28 | <a href="https://dspace.znu.edu.ua/jspui/bitstream/12345/12714/1/%D0%A1%D0%BC%D0%BE%D1%82%D1%80%D0%B8%D1%86%D1%8C%D0%BA%D0%B0_%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC.pdf">https://dspace.znu.edu.ua/jspui/bitstream/12345/12714/1/%D0%A1%D0%BC%D0%BE%D1%82%D1%80%D0%B8%D1%86%D1%8C%D0%BA%D0%B0_%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 21 (3) (0.19 %) |
| 29 | <a href="https://eir.zp.edu.ua/bitstreams/f3e775fd-281b-4cc3-b2bd-610ea1436360/download">https://eir.zp.edu.ua/bitstreams/f3e775fd-281b-4cc3-b2bd-610ea1436360/download</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | 20 (1) (0.18 %) |
| 30 | <a href="http://elar.khnu.km.ua/jspui/bitstream/123456789/10059/1/%D0%AF%D0%A8%D0%98%D0%9D%D0%90.pdf">http://elar.khnu.km.ua/jspui/bitstream/123456789/10059/1/%D0%AF%D0%A8%D0%98%D0%9D%D0%90.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | 19 (2) (0.17 %) |
| 31 | <a href="https://repo.nung.edu.ua/bitstreams/a8d2a041-83a0-44d8-91ce-ba350168d715/download">https://repo.nung.edu.ua/bitstreams/a8d2a041-83a0-44d8-91ce-ba350168d715/download</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | 17 (1) (0.16 %) |
| 32 | <a href="https://paperity.org/p/292193919/bezpeka-internet-resursiv-analiz-rozpovsiudzenosti-zagroza-tehnologiyi-zakhistu">https://paperity.org/p/292193919/bezpeka-internet-resursiv-analiz-rozpovsiudzenosti-zagroza-tehnologiyi-zakhistu</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 13 (1) (0.12 %) |
| 33 | <a href="http://repository.rshu.edu.ua/id/eprint/18692/1/%D0%91%D0%BE%D0%B9%D0%BA%D0%BE%20%D0%9E.%D0%9E.%2C%20122%20%D0%9A%D0%9D%2C%20%D0%B7%D0%B0%D0%BE%D1%87%D0%BD%D0%B0.pdf">http://repository.rshu.edu.ua/id/eprint/18692/1/%D0%91%D0%BE%D0%B9%D0%BA%D0%BE%20%D0%9E.%D0%9E.%2C%20122%20%D0%9A%D0%9D%2C%20%D0%B7%D0%B0%D0%BE%D1%87%D0%BD%D0%B0.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 13 (1) (0.12 %) |
| 34 | <a href="https://ela.kpi.ua/bitstream/123456789/37281/1/Klymchuk_bakalavr.pdf">https://ela.kpi.ua/bitstream/123456789/37281/1/Klymchuk_bakalavr.pdf</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 13 (1) (0.12 %) |
| 35 | <a href="https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BD%D1%94%D1%94%D0%B2%D0%B0%20%D0%92.%D0%9E.%202025%20%D1%80%D1%96%D0%BA.pdf">https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BD%D1%94%D1%94%D0%B2%D0%B0%20%D0%92.%D0%9E.%202025%20%D1%80%D1%96%D0%BA.pdf</a> | 12 (1) (0.11 %) |
| 36 | <a href="https://dspace.znu.edu.ua/xmlui/bitstream/handle/12345/20074/%D0%9F%D0%B8%D0%BF%D0%BA%D0%BE_%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC.pdf?sequence=1">https://dspace.znu.edu.ua/xmlui/bitstream/handle/12345/20074/%D0%9F%D0%B8%D0%BF%D0%BA%D0%BE_%D0%B4%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC.pdf?sequence=1</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 7 (1) (0.06 %)  |



## Список прийнятих фрагментів

| # | ЗМІСТ | КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ) |
|---|-------|---------------------------------------|
|---|-------|---------------------------------------|

БАКАЛАВРСЬКА РОБОТА

КБР.ІСТ-12.00.00.000 ПЗ

Група ІСТ-22-1

Владислав НЕЙМЕТ

2026

Івано-Франківський Національний Технічний Університет Нафти і Газу

Факультет Інформаційних Технологій

Кафедра інформаційно-телекомунікаційних технологій і систем

Неймет Владислав Дмитрович

УДК 004.415.2:004.8 .

БАКАЛАВРСЬКА РОБОТА

Проектування та розробка інформаційної системи управління контентом

для онлайн-платформи в ігровій індустрії

36. Інформаційні системи та технології

(назва освітньої програми)

27. 126 - Інформаційні системи та технології

(цифр і назва спеціальності)

27. Робота містить результати власних досліджень, використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело:

Здобувач освітнього ступеня \_\_\_\_\_ Неймет В. Д. \_\_\_\_\_

2. (підпис, прізвище та ініціали здобувача)

Науковий керівник \_\_\_\_\_ Паньків Ю. В., 2. Доцент \_\_\_\_\_

(підпис, прізвище, ім'я, по батькові, науковий ступінь, вчене звання керівника)

Допущено до захисту

1. В. О. завідувача кафедри ІТТС

1. К. т. н., доцент Заміховська О. Л.

3. (посада) (підпис) (дата) (прізвище та ініціали)

м. Івано-Франківськ – 2026 рік

Івано-Франківський національний технічний університет нафти і газу

Факультет Інформаційних технологій \_\_\_\_\_

Кафедра Інформаційно-телекомунікаційних технологій і систем \_\_\_\_\_

Освітньо-кваліфікаційний рівень бакалавр \_\_\_\_\_

Спеціальність 126 – Інформаційні системи та технології

ЗАТВЕРДЖУЮ:

Зав. кафедрою ІТТС \_\_\_\_\_

Д. т. н. Заміховська О. Л.

« » \_\_\_\_\_ 2026 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТОВІ

Неймету Владиславу

Дмитровичу \_\_\_\_\_

22. (прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Проектування та розробка інформаційної системи управління контентом для онлайн-платформи в ігровій індустрії керів

ник проекту (роботи) Паньків Ю. В., 22. доцент, затверджені наказом вищого навчального закладу від \_\_\_\_\_

2. Строк подання студентом проекту (роботи) 16 червня 2026 р. \_\_\_\_\_

3. Вихідні дані до роботи Методичні вказівки, технічна література \_\_\_\_\_

22. 4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Огляд аналогів та постановка завдання, 22. 2. Розробка структури бази даних, 3. розробка UML-діаграми; 3. Розробка вигляду та функціоналу веб-орієнтованої інформаційної системи \_\_\_\_\_

22. 5. Перелік графічного матеріалу (з точним зазначенням обов'язкових презентацій)

БР.ІСТ - 12.00.00.000 C1ER-діаграма реляційної бази даних ігрових публікацій;

БР.ІСТ - 12.00.00.000 C2 Архітектурна схема взаємодії підсистем та

маршрутизації REST API

6. Консультатни розділів роботи

Розділ Консультант Підпис, дата

1 Паньків Ю. В.

2 Паньків Ю. В.

3 Паньків Ю. В. 7. Дата видачі завдання – 22 грудня 2025 року КАЛЕНДАРНИЙ ПЛАН No з/п Назва етапів бакалаврської роботи

Термін виконання

етапів роботи Примітка 1 Аналіз предметної області та постановка

завдання

Березень, 2026 виконано

2

Розробка структури інформаційної

системи

Квітень, 2026 виконано 3 Розробка програмного забезпечення Травень, 2026 виконано 4 Оформлення пояснювальної записки Червень, 2026 виконано

Студент Неймет В. Д.

Керівник роботи Паньків Ю. В.

АНОТАЦІЯ

В рамках випускної кваліфікаційної роботи спроектовано та розроблено спеціалізовану інформаційну систему управління контентом (Headless CMS) для онлайн-платформи в ігровій індустрії на базі фреймворку Django та

архітектурного стилю REST API. В результаті було створено функціональний вебдодаток із рольовим доступом для адміністраторів, модераторів та авторів контенту. Система дозволяє контент-менеджерам здійснювати повний життєвий цикл створення, редагування та верстки ігрових матеріалів, новин та патчноутів за допомогою інтегрованого WYSIWYG-редактора CKEditor , а зовнішнім клієнтам (включаючи ігрові лаунчери) — оперативно отримувати структуровані дані у форматі JSON через розроблені точки доступу прикладного програмного інтерфейсу. Клієнтський вебінтерфейс для читачів є інтуїтивно зрозумілим, кросплатформним, адаптивним для різних пристроїв та підтримує темну тему оформлення. Для реалізації back-end частини використовувалися мова Python, інструментарій Django REST Framework та СУБД PostgreSQL. Застосування оптимізованого типу даних JSONField для зберігання пошукових тегів статей дозволило уникнути ресурсомістких операцій JOIN, забезпечуючи високу надійність, безпеку та швидкість обробки транзакцій. Front-end частина реалізована за допомогою вбудованого механізму шаблонізації Django та CSS-фреймворку Bootstrap 5, що забезпечило створення гнучкого та швидкодіючого інтерфейсу представлення даних. Ключові слова: інформаційна система, управління контентом, Headless CMS, ігрова індустрія, вебдодаток, Django, Python, REST API, PostgreSQL, Bootstrap 5.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, УПРАВЛІННЯ КОНТЕНТОМ, HEADLESS CMS, ІГРОВА ІНДУСТРІЯ, ВЕБДОДАТОК, DJANGO, PYTHON, REST API, СУБД POSTGRESQL, BOOTSTRAP.

ANNOTATION

Within the framework of the final qualifying work, a specialized content management information system (Headless CMS) for an online platform in the gaming industry was designed and developed using the Django framework and REST API architectural style. As a result, a functional web application with role-based access for administrators, moderators, and content authors was created. The system allows content managers to manage the full lifecycle of creating, editing, and formatting gaming publications, news, and patch notes via the integrated CKEditor WYSIWYG editor , while enabling external clients (including game launchers) to promptly retrieve structured data in JSON format through the developed application programming interface endpoints. The reader-facing web interface is intuitive, cross-platform, responsive across various devices, and supports a native dark theme configuration. The back-end part was implemented using Python, Django REST Framework, and PostgreSQL DBMS. The application of an optimized JSONField data type for storing article search tags avoided resource-intensive JOIN operations, ensuring high reliability, security, and transaction processing speed. The front-end part was developed using the native Django template engine and the Bootstrap 5 CSS framework, providing a flexible and high-performance interface for data presentation. Keywords: information system, content management, Headless CMS, gaming industry, web application, Django, Python, REST API, PostgreSQL, Bootstrap 5.

KEYWORDS: INFORMATION SYSTEM, CONTENT MANAGEMENT, HEADLESS CMS, GAMING INDUSTRY, WEB APPLICATION, DJANGO, PYTHON, REST API, POSTGRESQL DBMS, BOOTSTRAP.

31

|                                                                                                                  |    |
|------------------------------------------------------------------------------------------------------------------|----|
| Змн. Арк. No докум. Підпис Дата Арк. <u>6</u>                                                                    |    |
| <u>КБРІСТ-12.00.00.000 ПЗ</u>                                                                                    |    |
| <u>Розроб. Неймет В. Д.</u>                                                                                      |    |
| <u>Перевір. Паньків Ю. В.</u>                                                                                    |    |
| <u>Реценз. Н. Контр. Возний А. В.</u>                                                                            |    |
| Затверд. Зміховська О. Л.                                                                                        |    |
| Системний аналіз та проектування інформаційної системи управління контентом онлайн-платформи в ігровій індустрії |    |
| Літ. Аркушів                                                                                                     |    |
| 65                                                                                                               |    |
| ІФНТУНГ ІСТ-22-1                                                                                                 |    |
| ЗМІСТ                                                                                                            |    |
| ВСТУП.....                                                                                                       | 8  |
| 1 ДОСЛІДЖЕННЯ СИСТЕМ УПРАВЛІННЯ КОНТЕНТОМ ДЛЯ ІГРОВИХ ПЛАТФОРМ ТА ОБҐРУНТУВАННЯ ТЕХНОЛОГІЧНОГО СТЕКУ .....       | 11 |
| 1.1 Аналіз інформаційних потреб сучасних онлайн-платформ в ігровій                                               |    |

|                                                                                                           |    |
|-----------------------------------------------------------------------------------------------------------|----|
| індустрії.....                                                                                            | 11 |
| 1.2 Порівняльна характеристика сучасних Headless CMS на прикладі PayloadCMS та Directus .....             | 14 |
| 1.3 Формування функціональних вимог та вибір інструментарію розробки спеціалізованої системи.....         | 17 |
| 2 АРХІТЕКТУРНЕ ТА ІНФОЛОГІЧНЕ ПРОЄКТУВАННЯ СПЕЦІАЛІЗОВАНОЇ ІГРОВОЇ CMS.....                               | 21 |
| 2.1 Розробка концептуальної моделі та реляційної структури бази даних публікацій.....                     | 21 |
| 2.2 Проєктування логіки серверної частини та механізмів багатофакторної фільтрації ігрового контенту..... | 25 |
| 2.3 Алгоритмізація взаємодії підсистем та проєктування точок доступу REST API.....                        | 28 |
| 3 ПРОГРАМНА РЕАЛІЗАЦІЯ HEADLESS CMS З КЛІЄНТСЬКИМ ВЕБІНТЕРФЕЙСОМ.....                                     | 32 |
| 3.1 Реалізація серверної архітектури та інтеграція WYSIWYG-редактора у середовищі Django.....             | 32 |
| 3.2 Створення адаптивного клієнтського інтерфейсу для презентації ігрових патчноутів та новин.....        | 34 |
| 3.3 Розробка та тестування API для обміну даними із зовнішніми ігровими лаунчерами.....                   | 39 |
| 3.4 Інструкція з експлуатації та адміністрування розробленої платформи.....                               | 42 |
| 3.5 Тестування, оптимізація продуктивності та безпека вебдодатка..                                        | 49 |
| ВИСНОВКИ.....                                                                                             | 53 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....                                                                           | 56 |

#### ПЕРЕЛІК ОСНОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

|    |                                                                                                                                                                     |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 10 | БД – База даних.                                                                                                                                                    |
|    | ІС – Інформаційна система.                                                                                                                                          |
| 23 | API (Application Programming Interface) – прикладний програмний інтерфейс.                                                                                          |
| 18 | CMS (Content Management System) – система управління контентом.                                                                                                     |
| 23 | CSRF (Cross-Site Request Forgery) – підробка міжсайтових запитів (тип кібератаки).                                                                                  |
|    | DRF (Django REST Framework) – інструментарій для розробки веб-API на базі фреймворку Django.                                                                        |
| 30 | JSON (JavaScript Object Notation) – текстовий формат обміну даними, заснований на JavaScript.                                                                       |
|    | MVT (Model-View-Template) – архітектурний шаблон проєктування програмного забезпечення.                                                                             |
| 10 | ORM (Object-Relational Mapping) – технологія об'єктно-реляційного відображення даних.                                                                               |
| 30 | REST (Representational State Transfer) – архітектурний стиль програмного забезпечення для побудови розподілених систем.                                             |
| 23 | SQL (Structured Query Language) – мова структурованих запитів до реляційних баз даних.                                                                              |
| 10 | UX/UI (User Experience / User Interface) – користувацький досвід та користувацький інтерфейс.                                                                       |
| 23 | WYSIWYG (What You See Is What You Get) – властивість візуальних редакторів, за якої контент під час створення виглядає так само, як і кінцевий результат на екрані. |
|    | XSS (Cross-Site Scripting) – міжсайтовий скриптинг (вразливість вебдодатків).                                                                                       |
|    | Змн. Арк. No докум. Підпис Дата                                                                                                                                     |
|    | Арк.                                                                                                                                                                |
|    | 7КБР.ІСТ-12.00.00.000 ПЗ                                                                                                                                            |

Змн. Арк. No докум. Підпис Дата  
Арк.

8КБР.ІСТ-12.00.00.000 ПЗ

#### ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким зростанням обсягів цифрового контенту, зокрема у сфері розваг та ігрової індустрії. Користувачі вимагають миттєвого доступу до новин, оновлень, патчноутів (журналів змін) та аналітичних статей. У цьому контексті класичні монолітні системи управління контентом поступово втрачають свою ефективність через надмірну ресурсоемність, наявність надлишкового базового функціоналу та складність адаптації під специфічні реляційні моделі (наприклад, жорстку прив'язку статті до конкретної гри чи

розробника).

Окрім цього, сучасні стандарти розробки програмного забезпечення вимагають омніканальності — можливості доставляти один і той самий контент на різні платформи одночасно. Наприклад, новина про оновлення гри має з'явитися як на офіційному вебсайті компанії, так і бути автоматично підвантаженою безпосередньо у внутрішньоігровий клієнт (лаунчер). Оптимальним вирішенням цієї архітектурної проблеми є використання парадигми Headless CMS, яка дозволяє повністю відокремити серверну логіку, базу даних та панель адміністрування від клієнтського інтерфейсу представлення даних [11, 12].

Незважаючи на наявність на ринку універсальних Headless-рішень, створення власної спеціалізованої інформаційної системи на базі мови програмування Python та фреймворку Django з реалізацією REST API дозволяє створити максимально гнучку, безпечну та швидкодіючу платформу, позбавлену архітектурних обмежень комерційних продуктів. Відповідно, тема бакалаврської кваліфікаційної роботи, присвячена проектуванню та розробці такої системи для потреб ігрової індустрії, є надзвичайно актуальною, своєчасною та має високе практичне значення.

Об'єкт дослідження — процес управління, структурування та розповсюдження цифрового інформаційного контенту в межах ігрової індустрії.

Змн. Арк. No докум. Підпис Дата

Арк.

9KBPICT-12.00.00.000 ПЗ

Предмет дослідження — архітектура, математичні моделі, алгоритми та інструментальні програмні засоби розробки спеціалізованої системи управління контентом (CMS) для ігрових онлайн-платформ.

Мета роботи полягає у підвищенні ефективності управління ігровим інформаційним контентом шляхом проектування та розробки спеціалізованої інформаційної системи (Headless CMS) із сучасним вебінтерфейсом та можливостями інтеграції зі сторонніми додатками.

Для досягнення поставленої мети в роботі вирішуються такі завдання:

1. Дослідити специфіку управління контентом в ігровій індустрії та проаналізувати існуючі рішення класу Headless CMS.
2. Сформулювати функціональні та нефункціональні вимоги до системи, обґрунтувати вибір технологічного стеку для її реалізації.
3. Розробити концептуальну модель та спроектувати реляційну структуру бази даних для зберігання публікацій, категорій та прив'язок до ігор.
4. Програмно реалізувати серверну логіку управління контентом (Backend) та розгорнути безпечну панель адміністратора.
5. Створити клієнтський адаптивний інтерфейс (Frontend) для відображення ігрових новин з механізмами фільтрації та повнотекстового пошуку.
6. Розробити прикладний програмний інтерфейс (REST API) для забезпечення можливості серіалізації та передачі даних у зовнішні ігрові додатки у форматі JSON.

Для розв'язання поставлених завдань у роботі було використано комплекс сучасних наукових та інженерних методів. Для формування вимог застосовано методи системного аналізу; проектування бази даних здійснювалося на основі теорії реляційної алгебри та інфологічного моделювання. Програмна реалізація базується на парадигмі об'єктно-орієнтованого програмування (ООП) та використанні архітектурного патерну MVT (Model-View-Template) і принципів побудови REST-архітектури мережесхемних додатків.

Змн. Арк. No докум. Підпис Дата

Арк.

10KBPICT-12.00.00.000 ПЗ

15 наукова новизна одержаних результатів полягає у подальшому розвитку підходів до побудови спеціалізованих систем управління контентом. Зокрема, набула практичної реалізації архітектура Headless CMS, адаптована спеціально для потреб ігрової індустрії, що, на відміну від існуючих універсальних блогерських платформ, забезпечує цілісну реляційну прив'язку інформаційних матеріалів до конкретних ігрових проєктів з можливістю їхньої подальшої безпечної дистрибуції через програмні інтерфейси.

Практичне значення одержаних результатів полягає в тому, що розроблена інформаційна система є повністю функціональним програмним продуктом.

Реалізоване API, інтегрований WYSIWYG-редактор та зручна панель адміністратора дозволяють використовувати розроблену CMS як основу для створення новинних порталів, сайтів студій-розробників ігор або як надійний бекенд-сервіс для доставки патчноутів безпосередньо у клієнтські ігрові лаунчери.

Змн. Арк. No докум. Підпис Дата

Арк.

11КБР.ІСТ-12.00.00.000 ПЗ

## 1 ДОСЛІДЖЕННЯ СИСТЕМ УПРАВЛІННЯ КОНТЕНТОМ ДЛЯ ІГРОВИХ ПЛАТФОРМ ТА ОБҐРУНТУВАННЯ ТЕХНОЛОГІЧНОГО СТЕКУ

### 1.1 Аналіз інформаційних потреб сучасних онлайн-платформ в ігровій індустрії

Сучасна ігрова індустрія є одним із найбільш технологічних та швидкозростаючих секторів цифрової економіки. З переходом до моделі цифрової дистрибуції та концепції «ігра як сервіс» (Games-as-a-Service, GaaS), взаємодія між розробниками та спільнотою гравців набула безперервного характеру. Успішна підтримка життєвого циклу будь-якого ігрового проєкту сьогодні критично залежить від своєчасного, структурованого та омніканального інформування користувачів.

Дослідження специфіки інформаційних потоків у межах ігрових порталів дозволяє виділити кілька ключових категорій контенту, якими необхідно оперувати:

1. Патчноути (журнали змін): Деталізовані списки змін, виправлень та нововведень, що супроводжують кожне оновлення гри. Цей тип контенту вимагає жорсткої реляційної прив'язки до конкретної версії продукту та самої гри.
  2. Анонси та новини: Інформація про майбутні релізи, події (івенти) та внутрішньоігрові акції.
  3. Аналітичні та оглядові статті: Розгорнуті матеріали від розробників (DevBlogs) або ігрової преси, які потребують складного форматування, підтримки вбудованих медіафайлів та зручної категоризації.
- Специфіка управління таким контентом полягає у тому, що він не може існувати у вигляді простого неструктурованого тексту. Кожна публікація повинна мати набір метаданих: теги, прив'язку до конкретної гри, категорію та дату публікації. Це необхідно для забезпечення механізмів багатофакторної

Змн. Арк. No докум. Підпис Дата

Арк.

12КБР.ІСТ-12.00.00.000 ПЗ

фільтрації, оскільки користувач ігрової платформи найчастіше шукає інформацію про один конкретний проєкт, ігноруючи решту інформаційного шуму.

Окремої уваги потребує специфікація медіаконтенту, що супроводжує ігрові публікації. Ігрова індустрія характеризується високими вимогами до візуальної складової: це величезна кількість високоякісних скріншотів, відеороликів, трейлерів та концепт-артів. Процес обробки таких файлів — від завантаження до оптимізації (стиснення, конвертації у сучасні формати типу WebP, створення прев'ю) — у традиційних системах часто є рутинним і помилково-орієнтованим. Відсутність автоматизованих механізмів маніпуляції з медіа на стороні бекенду призводить до того, що редактори витрачають значні ресурси часу на підготовку файлів у зовнішніх графічних редакторах, а не на якісне наповнення платформи. Це створює вузьке місце в редакційному циклі, яке можна усунути лише шляхом впровадження спеціалізованих інструментів автоматизації медіа в ядро інформаційної системи.

Основна проблема існуючих традиційних систем управління контентом (наприклад, монолітних рішень типу WordPress або Joomla) полягає в їхній орієнтованості на генерацію кінцевих HTML-сторінок [3]. Такі системи чудово підходять для класичних блогів, але створюють суттєві архітектурні обмеження для ігрової індустрії. Сучасна графіка взаємодії з користувачем передбачає, що контент має доставлятися не лише у веббраузер, а й безпосередньо у внутрішньоігрові клієнти (лаунчери), мобільні додатки-супутники або Discord-сервери спільнот. Наочне порівняння та архітектурні відмінності між традиційними монолітними та сучасними Headless системами управління контентом продемонстровано на рисунку 1.1.

Змн. Арк. No докум. Підпис Дата  
Арк.

13КБР:ICT-12.00.00.000 ПЗ

Рисунок 1.1 - Архітектурні відмінності традиційних та Headless систем управління контентом [3]

Для реалізації такого омніканального підходу інформаційна система повинна працювати за принципом API-first, тобто першочергово віддавати чисті, машиночитані дані (зазвичай у форматі JSON), які потім можуть бути інтерпретовані та стилізовані будь-яким клієнтським пристроєм чи програмою. Крім того, панель управління такою системою має бути інтуїтивно зрозумілою для контент-менеджерів, надаючи їм зручний візуальний інструментарій (WYSIWYG-редактори) без необхідності втручання в програмний код.

Таким чином, проведений аналіз інформаційних потреб показує, що для забезпечення ефективної комунікації в ігровій індустрії виникає гостра необхідність у проектуванні спеціалізованої системи управління контентом. Така система повинна поєднувати гнучку реляційну базу даних, захищену панель адміністрування та потужний програмний інтерфейс (API) для незалежного розповсюдження контенту на різноманітні ігрові платформи.

Змн. Арк. No докум. Підпис Дата  
Арк.

14КБР:ICT-12.00.00.000 ПЗ

1.2Порівняльна характеристика сучасних Headless CMS на прикладі

PayloadCMS та Directus

Вирішення завдання омніканальної доставки контенту в ігровій індустрії вимагає відмови від традиційних (монолітних) систем на користь архітектури Headless CMS. Суть цього підходу полягає у відсутності жорстко прив'язаного клієнтського інтерфейсу (Front-end); натомість система надає потужний адміністративний інтерфейс для управління даними та REST або GraphQL API для їхнього експорту [11]. На сучасному ринку програмного забезпечення існує низка готових Headless-рішень, серед яких найбільш показовими є PayloadCMS та Directus.

Аналіз архітектури PayloadCMS. PayloadCMS — це сучасна система управління контентом, побудована на базі стеку технологій Node.js, Express та React. Головною особливістю цієї платформи є підхід **"Code-first"** (код понад усе), що передбачає конфігурацію бази даних та панелі адміністратора безпосередньо через написання конфігураційних файлів мовою TypeScript [12, 13].

Згідно з документацією та архітектурними паттернами PayloadCMS, структура даних базується на двох основних концепціях: колекціях (Collections) та глобальних змінних (Globals). Колекції використовуються для представлення масивів однотипних даних, таких як публікації (Posts), категорії (Categories) або профілі ігор. Водночас глобальні змінні призначені для зберігання унікальних об'єктів, наприклад, налаштувань головного меню навігації або загальної конфігурації платформи [14]. Інтерфейс панелі адміністратора та базова структура управління колекціями у PayloadCMS зображено на рисунку 1.2.

Змн. Арк. No докум. Підпис Дата  
Арк.

15КБР:ICT-12.00.00.000 ПЗ

Рисунок 1.2 - Інтерфейс панелі адміністратора PayloadCMS [14]

Важливим аспектом будь-якої інформаційної системи є безпека. У PayloadCMS реалізовано гнучку систему управління користувачами (Users) та контроль доступу на основі ролей (RBAC — Role-Based Access Control). Це дозволяє адміністраторам детально налаштовувати права доступу контент-менеджерів до певних колекцій, що є критично важливим для великих ігрових порталів із великим штатом редакторів [15].

Аналіз архітектури Directus. Альтернативним підходом до побудови Headless CMS є платформа Directus. На відміну від PayloadCMS, Directus позиціонує себе як **"Data-first"** рішення (дані понад усе). Ця система фактично є обгорткою над будь-якою реляційною базою даних (SQL), яка миттєво генерує динамічне API та адміністративну панель на базі фреймворку Vue.js [11]. Структуру управління даними та візуальний інтерфейс користувача платформи Directus представлено на рисунку 1.3.

Змн. Арк. No докум. Підпис Дата  
Арк.

16КБР.ІСТ-12.00.00.000 ПЗ

Рисунок 1.3 - Структура управління даними у платформі Directus [11]

Однією з ключових переваг Directus є можливість локального розгортання (Self-hosted). Згідно з офіційною документацією архітектури платформи, самостійний хостинг дозволяє розробникам повністю контролювати базу даних без прив'язки до хмарних сервісів розробника та уникнути лімітів на кількість запитів до API [11]. Для ігрової індустрії, де навантаження на сервери під час виходу патчноутів або релізів може зростати експоненціально, відсутність таких лімітів є критичною вимогою.

Обґрунтування потреби у власній розробці. Проведений аналіз свідчить, що системи PayloadCMS та Directus є потужними інструментами, які чудово ілюструють переваги Headless-архітектури. Вони забезпечують поділ даних і представлення, мають вбудовані механізми контролю доступу [15] та підтримують сучасні типи баз даних.

Проте використання цих систем для реалізації спеціалізованої ігрової платформи має низку недоліків. По-перше, вони базуються на екосистемі JavaScript/Node.js, що вимагає специфічного серверного оточення та ускладнює інтеграцію з аналітичними ігровими модулями, які традиційно пишуться мовою Python. По-друге, ці CMS є універсальними, а тому містять величезну кількість надлишкового коду та функціоналу, який ніколи не буде

Змн. Арк. No докум. Підпис Дата  
Арк.

17КБР.ІСТ-12.00.00.000 ПЗ

використаний на платформі, орієнтованій виключно на публікацію ігрових новин та патчноутів.

З огляду на це, вважається доцільним спроектувати та розробити власну Headless CMS. Це дозволить створити максимально оптимізовану, легковисну реляційну базу даних та гнучке API, архітектура якого буде з самого початку орієнтована на ігрові сутності (Гра, Категорія, Публікація).

1.3Формування функціональних вимог та вибір інструментарію  
розробки спеціалізованої системи

Ефективність будь-якої інформаційної системи безпосередньо залежить від точності сформованих на етапі проєктування вимог та правильності вибору технологічного стеку для їх реалізації [3]. На основі проведеного системного аналізу предметної області було сформовано комплекс вимог до проєктованої системи управління ігровим контентом GamePulse CMS, які поділяються на функціональні та нефункціональні.

До ключових функціональних вимог належать:

1. Забезпечення надійного збереження та структурування інформації з використанням реляційної моделі (чітка прив'язка публікації до конкретної гри, платформи та категорії).
2. Наявність захищеної панелі адміністратора з інтегрованим візуальним редактором (WYSIWYG) для форматування текстів патчноутів, гайдів та оглядів без використання HTML-розмітки.
3. Можливість гнучкої багатофакторної фільтрації ігрового контенту та реалізація механізмів повнотекстового пошуку.
4. Забезпечення рольової моделі розмежування прав доступу (Адміністратор, Редактор, Глядач) для колективної роботи над матеріалами.

Змн. Арк. No докум. Підпис Дата  
Арк.

18КБР.ІСТ-12.00.00.000 ПЗ

5. Наявність стандартизованого прикладного програмного інтерфейсу (API) для незалежного експорту даних на сторонні клієнтські платформи.

До нефункціональних вимог висуваються: висока швидкість відповіді сервера (оскільки цільовою аудиторією є геймери, які звикли до швидких інтерфейсів), масштабованість архітектури під час пікових навантажень (наприклад, під час релізу очікуваної гри), а також стійкість до несанкціонованого доступу та кібератак. Виконання цих вимог зумовило необхідність проведення порівняльного аналізу та вибору оптимального інструментарію розробки [4].

Для реалізації бізнес-логіки системи, системи модерації та управління базою даних розглядалися три основні альтернативи: Node.js (Express), Ruby on Rails та Python (Django). Використання мікрофреймворку Express на базі Node.js вимагає самостійної реалізації базових функцій безпеки та написання панелі адміністратора з нуля, що невіправдано збільшує час розробки. З огляду на це, рішення було прийнято на користь мови програмування Python та фреймворку Django. Головною архітектурною перевагою Django є наявність потужної вбудованої панелі адміністрування, яка генерується автоматично на основі моделей даних [7]. Це дозволяє закрити базову потребу контент-менеджерів в управлінні публікаціями "з коробки". Крім того, надійна вбудована ORM-система (Object-Relational Mapping) повністю нівелює ризики SQL-ін'єкцій, що задовольняє нефункціональні вимоги щодо безпеки проекту. Оскільки специфіка ігрової індустрії передбачає омніканальність (доставку новин як на сайт, так і в ігрові лаунчери), системі необхідний надійний протокол обміну даними. У процесі аналізу розглядалися технології WebSockets, GraphQL та REST API [6]. Від використання WebSockets було вирішено відмовитися. Хоча ця технологія дозволяє двосторонній обмін даними в реальному часі, вона вимагає управління станом з'єднань та налаштування In-Memory сховищ (наприклад, Redis), що є надлишковим для платформи, орієнтованої на

Змн. Арк. No докум. Підпис Дата

Арк.

19КБР.ІСТ-12.00.00.000 ПЗ

публікацію статей. Технологія GraphQL також створює надмірний архітектурний оверхед, оскільки структура даних для ігрових новин є чітко детермінованою. Тому було обрано класичний архітектурний стиль REST API. Використання бібліотеки Django REST Framework (DRF) забезпечує стандартизовану серіалізацію даних з бази у машиночитаний формат JSON [8], а оновлення стану контенту на клієнті здійснюється через оптимізовані HTTP-запити.

Проектована CMS вимагає надійного збереження складних зв'язків: профілі користувачів пов'язані з ролями, ігри — з категоріями, а статті — з медіафайлами та масивами пошукових тегів. Під час проектування розглядалися альтернативи: MySQL, MongoDB та PostgreSQL.

Рішення класу NoSQL (наприклад, MongoDB) чудово підходять для неструктурованих даних, але значно ускладнюють підтримку транзакцій та забезпечення цілісності жорстких реляційних зв'язків [1]. MySQL є популярним рішенням, проте має обмежену підтримку специфічних типів даних. В результаті, для кінцевого розгортання (Production) було обрано об'єктно-реляційну СУБД PostgreSQL. Вона забезпечує сувору відповідність принципам ACID, а вбудована підтримка типу даних ARRAY (масиви) дозволяє оптимізувати збереження тегів статей без створення зайвих зв'язкових таблиць, що суттєво пришвидшує читання інформації з бази [2]. Для локального розроблення та тестування системи використовується вбудована легковагова СУБД SQLite [9].

Для реалізації візуального представлення системи (клієнтської частини для читачів) обрано класичний підхід на базі HTML5, CSS3 та механізмів шаблонізації Django. З метою забезпечення кроссплатформної адаптивності (Responsive Design) та прискорення верстки інтерфейсів інтегровано фронтенд-фреймворк Bootstrap 5 [10]. Він надає готову систему сіток (Grid system) для побудови ергономічного каталогу новинних карток та має вбудовану підтримку темної теми оформлення (Dark Mode), що є де-факто

Змн. Арк. No докум. Підпис Дата

Арк.

20КБР.ІСТ-12.00.00.000 ПЗ

індустріальним стандартом та критично важливою візуальною вимогою для продуктів у сфері ігрової індустрії.

Підсумовуючи, обраний технологічний стек (Python, Django, PostgreSQL, REST API, Bootstrap 5) є оптимальним балансом між швидкістю розробки, надійністю та продуктивністю, що дозволяє перейти до етапу інфологічного проектування та моделювання інформаційної архітектури системи.

Змн. Арк. No докум. Підпис Дата

Арк.

21КБР.ІСТ-12.00.00.000 ПЗ

## 2 АРХІТЕКТУРНЕ ТА ІНФОЛОГІЧНЕ ПРОЄКТУВАННЯ

### СПЕЦІАЛІЗОВАНОЇ ІГРОВОЇ CMS

#### 2.1 Розробка концептуальної моделі та реляційної структури бази

даних публікацій

Перед формуванням реляційної структури бази даних необхідно чітко детермінувати функціональні ролі користувачів системи та межі їхньої взаємодії з програмним комплексом. З цією метою було розроблено модель прецедентів (Use Case Diagram) за стандартом UML. Модель описує поведінку системи з точки зору зовнішніх акторів.

В архітектурі GamePulse CMS виділено три ключові ролі (актори):

1. Глядач (Reader): Неавторизований користувач, або стороння система (ігровий лаунчер), що має доступ виключно до операцій читання (GET-запити). Глядач може переглядати стрічку новин, використовувати механізми багатофакторної фільтрації за ігровими категоріями та здійснювати повнотекстовий пошук за тегами.

2. Контент-менеджер (Author): Авторизований користувач базового рівня. Має права на створення нових публікацій, завантаження медіафайлів та редагування власних чернеток. Контент-менеджер не може самостійно публікувати матеріали — він лише переводить їх у статус «На модераторі».

3. Адміністратор (Admin/Reviewer): Користувач із максимальним рівнем привілеїв. Окрім можливостей контент-менеджера, адміністратор здійснює верифікацію матеріалів, змінює їхній статус на «Опубліковано», а також управляє словниками системи (додає або видаляє ігрові категорії) та призначає права доступу іншим співробітникам.

Змн. Арк. No докум. Підпис Дата

Арк.

22КБР.ІСТ-12.00.00.000 ПЗ

Візуалізацію меж системи та варіантів використання, що визначають подальшу структуру прав доступу на рівні бази даних, наведено на рисунку 2.1.

Рисунок 2.1 - UML-діаграма прецедентів (Use Case) системи управління ігровим контентом

Основою будь-якої системи управління контентом є надійна та оптимізована база даних. Процес інфологічного проєктування GamePulse CMS розпочався з розробки концептуальної моделі (Entity-Relationship Diagram), метою якої була формалізація ключових інформаційних сутностей предметної області та зв'язків між ними, абстрагуючись від фізичної реалізації в конкретній СУБД [1, 2].

Центральним вузлом архітектури виступає сутність Article (Стаття/Публікація), яка агрегує в собі ігровий контент (текст, метадані, прив'язку до категорії гри). Оскільки сучасна ігрова журналістика вимагає

Змн. Арк. No докум. Підпис Дата

Арк.

23КБР.ІСТ-12.00.00.000 ПЗ

мультимедійного супроводу, була спроектована залежна сутність MediaElement. Для забезпечення гнучкості системи застосовано патерн об'єктно-орієнтованого моделювання: базова сутність зберігає загальні параметри медіафайлу (посилання, розмір, порядок у тексті), тоді як дочірні сутності Image та Video наслідують ці властивості, розширюючи їх специфічними атрибутами (наприклад, роздільна здатність або хронометраж відеоролика). Концептуальну модель даних системи представлено на рисунку 2.2.

Рисунок 2.2 - Розробка концептуальної моделі та реляційної структури бази даних публікацій

Змн. Арк. No докум. Підпис Дата

Арк.

24КБР.ІСТ-12.00.00.000 ПЗ

Наступним кроком стала трансформація концептуальної схеми у логічну реляційну модель. Особливу увагу було приділено організації спільної роботи над матеріалами (наприклад, коли над гайдом працює автор і редактор).

Зв'язок «багато-до-багатьох» між профілями users (Користувачі) та articles (Статті) було декомпоновано через створення асоціативної таблиці contributors

(Співавтори). Це дозволило чітко розмежувати права доступу на рівні окремої публікації за допомогою таблиці roles (Ролі), що відповідає архітектурним стандартам побудови безпечних інформаційних систем [3]. Логічну структуру реляційної моделі зображено на рисунку 2.3.

Рисунок 2.3 - Логічна структура реляційної бази даних платформи

Змн. Арк. No докум. Підпис Дата

Арк.

25КБР.ІСТ-12.00.00.000 ПЗ

У процесі проектування було проведено процедуру нормалізації відношень. Модель приведено до третьої нормальної форми (3НФ), що гарантує відсутність транзитивних залежностей та аномалій оновлення. Усі неключові атрибути (наприклад, привілеї на редагування) повністю залежать від первинного ключа відповідної таблиці (ролі).

Проте, враховуючи специфіку роботи СУБД PostgreSQL, при проектуванні таблиці articles було прийнято свідоме рішення щодо використання масивів (varchar[]) для зберігання пошукових тегів гри. Хоча з точки зору класичної теорії баз даних це є відхиленням від першої нормальної форми (1НФ), яка вимагає атомарності атрибутів [1], на практиці такий підхід значно підвищує продуктивність системи. Зберігання масиву тегів безпосередньо у записі публікації дозволяє уникнути ресурсомістких операцій JOIN під час рендерингу стрічки ігрових новин, що критично важливо для забезпечення швидкого відгуку платформи.

## 2.2. Проектування логіки серверної частини та механізмів багатофакторної фільтрації ігрового контенту

Серверна частина (бекенд) розроблюваної інформаційної системи виконує роль центрального контролера, який відповідає за маршрутизацію запитів, валідацію вхідних даних, реалізацію бізнес-правил та взаємодію з базою даних. Зважаючи на специфіку ігрової платформи, логіка серверної частини була спроектована таким чином, щоб забезпечити максимальну надійність збереження контенту під час тривалих сесій редагування.

Першим ключовим алгоритмом є процес ініціалізації нової публікації. Коли контент-менеджер ініціює створення огляду чи патчноту, серверна частина виконує комплекс автоматизованих дій: перевіряє права доступу користувача, генерує унікальний ідентифікатор матеріалу, здійснює первинну валідацію обов'язкових полів (наприклад, наявності заголовка та прив'язки до категорії гри), створює відповідний запис у базі даних і автоматично призначає творцю

Змн. Арк. No докум. Підпис Дата

Арк.

26КБР.ІСТ-12.00.00.000 ПЗ

матеріалу роль «Автор» у таблиці привілеїв. Графічно алгоритм створення нової статті у вигляді діаграми бізнес-процесу наведено на рисунку 2.4.

Рисунок 2.4 - Модель бізнес-процесу ініціалізації нової публікації

Оскільки підготовка якісного ігрового огляду або гайду може тривати кілька годин, критичною вимогою до логіки бекенду є запобігання втраті даних (наприклад, у разі розриву інтернет-з'єднання у редактора). Для вирішення цієї проблеми було спроектовано механізм асинхронної фонові синхронізації (Auto-save). Логіка процесу полягає у тому, що при внесенні будь-яких змін у текстовий редактор клієнтська частина паралельно відображає їх користувачу та генерує фоновий API-запит до сервера. Серверна логіка Django перехоплює цей запит, оновлює стан чернетки у базі даних та повертає статус успішного збереження. Діаграму процесу синхронізації контенту між клієнтом та сервером зображено на рисунку 2.5.

Змн. Арк. No докум. Підпис Дата

Арк.

27КБР.ІСТ-12.00.00.000 ПЗ

Рисунок 2.5 - Процес асинхронної синхронізації контенту між клієнтом та сервером

Окрім управління життєвим циклом статей, важливим завданням серверної частини є забезпечення зручного пошуку матеріалів для кінцевих користувачів. Сучасні платформи ігрової індустрії оперують тисячами публікацій, тому класичного пошуку лише за назвою недостатньо. У проєктованій системі реалізовано механізм багатофакторної фільтрації.

Логіка фільтрації базується на використанні можливостей об'єктно-реляційного відображення (ORM) фреймворку Django. Користувач (читач)

може формувати складні запити, комбінуючи кілька параметрів: наприклад, шукати статті в категорії «Стратегії», які мають тег «Гайд» та були опубліковані за останній місяць. На рівні серверної логіки такий запит трансформується у складний SQL-оператор, який звертається до масиву тегів (varchar[]) у таблиці articles. Оскільки масиви в PostgreSQL підтримують спеціалізовані індекси типу GIN (Generalized Inverted Index), операція фільтрації за кількома тегами одночасно виконується з мінімальними затримками, навіть за умови значного масштабування бази даних. Такий підхід

Змн. Арк. No докум. Підпис Дата

Арк.

28КБР:ICT-12.00.00.000 ПЗ

гарантує високу продуктивність підсистеми пошуку та швидку видачу релевантного ігрового контенту.

2.3.Алгоритмізація взаємодії підсистем та проектування точок доступу REST API

Розділення клієнтської та серверної частин у парадигмі Headless CMS вимагає проектування надійних механізмів обміну даними. Для візуалізації високорівневої структури системи GamePulse CMS та алгоритмів взаємодії її підсистем було використано методологію архітектурного моделювання C4. Програмний комплекс декомпозовано на три незалежні контейнери: клієнтський вебдодаток (Single Page Application на базі React), серверний програмний інтерфейс (Backend API на базі фреймворку Django) та реляційну базу даних (PostgreSQL).

Взаємодія між користувацьким інтерфейсом (фронтом) та бізнес-логікою (бекендом) здійснюється виключно через мережеві запити за протоколом HTTPS. Архітектурну діаграму контейнерів системи та напрямки потоків даних між ними наведено на рисунку 2.6.

Змн. Арк. No докум. Підпис Дата

Арк.

29КБР:ICT-12.00.00.000 ПЗ

Рисунок 2.6 - Діаграма контейнерів системи управління ігровим контентом (C4 Model)

Проектування точок доступу (endpoints) REST API вимагає чіткої алгоритмізації послідовності обробки запитів. Для моделювання динаміки клієнт-серверної взаємодії та часової послідовності викликів розроблено діаграми UML Sequence Diagram.

Розглянемо алгоритм роботи ендпойнту створення нової публікації (точка доступу POST /api/articles/). Ініціатором процесу виступає користувач, який через інтерфейс клієнтського додатка відправляє базовий набір даних (заголовок та категорію гри). Фронтенд формує HTTP-запит і передає його на

Змн. Арк. No докум. Підпис Дата

Арк.

30КБР:ICT-12.00.00.000 ПЗ

бекенд у форматі JSON. Сервер приймає запит і запускає процес строгої валідації (наприклад, перевіряє унікальність заголовка або відповідність типу даних). У разі виявлення помилок алгоритм миттєво переривається і повертає на клієнт код помилки (HTTP 400 Bad Request). При успішній валідації сервер відкриває транзакцію, звертається до бази даних для збереження об'єкта, після чого повертає клієнту серіалізований об'єкт створеної статті (HTTP 201 Created). Цей алгоритм проілюстровано на рисунку 2.7.

Рисунок 2.7 - Діаграма послідовності (Sequence Diagram) алгоритму створення публікації

Іншим критично важливим алгоритмом є управління редакційною командою, що реалізується через точку доступу POST /api/contributors/invite/. Алгоритмізація цього процесу передбачає складнішу багатокрокову перевірку станів у базі даних. При спробі головного редактора додати нового співавтора

Змн. Арк. No докум. Підпис Дата

Арк.

31КБР:ICT-12.00.00.000 ПЗ

до проекту, бекенд спочатку виконує пошуковий запит до таблиці користувачів. Якщо вказаний email не знайдено, API генерує виняткову ситуацію (HTTP 404 Not Found), і фронтенд відображає відповідне сповіщення. За умови існування профілю, система перевіряє наявність

попередніх зв'язків (аби уникнути дублювання ролей) і лише після цього створює новий запис у таблиці привілеїв, відправляючи код успішного виконання операції (HTTP 200 OK). Деталізовану послідовність обміну повідомленнями для цього прецеденту відображено на рисунку 2.8. Рисунок 2.8 - Діаграма послідовності процесу верифікації та запрошення співавтора

Розроблена інфологічна архітектура, структура бази даних та алгоритми взаємодії API-ендпойнтів повністю задовольняють вимоги до системи та є

Змн. Арк. No докум. Підпис Дата

Арк.

32КБР.ІСТ-12.00.00.000 ПЗ

достатньою базою для переходу до етапу безпосереднього написання програмного коду та розгортання інформаційної системи.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ HEADLESS CMS З КЛІЄНТСЬКИМ ВЕБІНТЕРФЕЙСОМ

3.1 Реалізація серверної архітектури та інтеграція WYSIWYG-редактора у середовищі Django

Процес програмної реалізації системи управління ігровим контентом GamePulse CMS розпочався з ініціалізації віртуального середовища Python та розгортання базового ядра фреймворку Django. Відповідно до архітектурного патерну MVT (Model-View-Template), першочерговим завданням стала трансляція спроектованої у другому розділі логічної моделі бази даних у програмні об'єкти — моделі Django.

Основою серверної частини є додаток `content_manager`, який інкапсулює логіку роботи з публікаціями. Для реалізації сутності «Стаття» (Article) та її зв'язку з «Категорією» (Category) було використано стандартний модуль `django.db.models`. Особливу увагу приділено оптимізації типів даних: для заголовків використано `CharField` з обмеженням довжини, а для основного контенту — спеціалізований `RichTextField`, який забезпечує підтримку форматowanego тексту [16].

Лістинг 3.1 Програмна реалізація моделей бази даних ігрового контенту

```
26 from django.db import models
26 from django.contrib.auth.models import User
26 from ckeditor.fields import RichTextField
class Category(models.Model):
    name = models.CharField(max_length= 100,
verbose_name="Назва ігрової категорії")
slug = models.SlugField(unique=True, blank=True)
26 def __str__(self):
return self.name
```

Змн. Арк. No докум. Підпис Дата

Арк.

33КБР.ІСТ-12.00.00.000 ПЗ

```
class Article(models.Model):
```

```
STATUS_CHOICES = (
```

```
('draft', 'Чернетка'),
```

```
('review', 'На модерзації'),
```

```
('published', 'Опубліковано'),
```

```
)
```

```
title = models.CharField(max_length=255,
```

```
verbose_name="Заголовок публікації")
```

```
7 category = models.ForeignKey(Category,
on_delete=models.CASCADE, related_name='articles')
```

```
author = models.ForeignKey(User,
```

```
28 on_delete=models.SET_NULL, null=True, related_name='posts')
```

```
content = RichTextField(verbose_name="Текст статті")
```

```
tags = models.JSONField(default=list, blank=True,
```

```
verbose_name="Пошукові теги")
```

```
status = models.CharField(max_length=20,
```

```
choices=STATUS_CHOICES, default='draft')
```

```
21 created_at = models.DateTimeField(auto_now_add=True)
```

```
updated_at = models.DateTimeField(auto_now=True)
```

```
class Meta:
```

```
ordering = ['-created_at']
```

```
verbose_name = "Ігрова стаття"
```

```
verbose_name_plural = "Ігрові статті"
```

```
def __str__(self):
```

return self.title

Потрібно звернути увагу на використання JSONField для зберігання пошукових тегів (tags). Як було обґрунтовано на етапі інфологічного проектування, використання масивів JSON замість створення окремої реляційної таблиці для тегів дозволяє значно прискорити виконання пошукових запитів при фільтрації новин на клієнтській стороні, оскільки PostgreSQL має вбудовані механізми оптимізованої роботи з типом JSONB. Наступним кроком стала інтеграція візуального редактора. Однією з головних вимог користувачів (редакторів платформи) була можливість верстати складні огляди ігор та патчноути без використання HTML-розмітки. Для вирішення цього завдання у середовище Django було інтегровано сторонню бібліотеку django-ckeditor [17].

Цей WYSIWYG-редактор було підключено безпосередньо до адміністративної панелі Django шляхом реєстрації моделей у файлі admin.py. Завдяки цьому, при створенні нової статті через панель керування, контент-

Змн. Арк. No докум. Підпис Дата

Арк.

34КБР:ICT-12.00.00.000 ПЗ

менеджер отримує повноцінний інтерфейс з панеллю інструментів для форматування тексту, додавання маркованих списків (для патчноутів) та вбудовування медіафайлів. Логіка збереження медіафайлів налаштована таким чином, щоб завантажені зображення автоматично розподілялися по директоріях залежно від дати завантаження (параметр upload\_to), що запобігає перевантаженню кореневої файлової системи сервера.

3.2 Створення адаптивного клієнтського інтерфейсу для презентації ігрових патчноутів та новин

Після реалізації серверної логіки та налаштування адміністративної панелі, наступним етапом розробки стало створення клієнтського інтерфейсу (фронтенду). Головним завданням було забезпечення зручного, швидкого та візуально привабливого середовища для читачів ігрового порталу. Оскільки цільовою аудиторією платформи є геймери, інтерфейс проектувався з урахуванням сучасних трендів ігрового вебдизайну: використання темної теми (Dark Mode), акцент на великих медіа-обкладинках та відсутність зайвих елементів, що відвертають увагу від контенту. Для швидкого прототипування та забезпечення кросплатформної адаптивності (коректного відображення як на ПК, так і на мобільних пристроях) було використано фронтенд-фреймворк Bootstrap 5 [10]. Архітектура клієнтської частини базується на вбудованому механізмі шаблонізації Django (Django Templates). Було створено базовий шаблон base.html, який містить загальні елементи навігації (Header), підвал сайту (Footer) та підключені файли стилів. Всі інші сторінки платформи наслідують цей шаблон за допомогою тегу {% extends %}, що дозволило уникнути дублювання HTML-коду [19].

Головна сторінка платформи (Дашборд статей) реалізує виведення списку останніх публікацій. Для цього у файлі views.py було створено представлення (View), яке отримує з бази даних усі статті зі статусом «Опубліковано», сортує їх за датою додавання (від найновіших до найстаріших) та передає у шаблон

Змн. Арк. No докум. Підпис Дата

Арк.

35КБР:ICT-12.00.00.000 ПЗ

index.html. Для організації контенту на головній сторінці використано сіткову систему Bootstrap (Grid System). Кожна публікація виводиться у вигляді інтерактивної картки (Card), яка містить обкладинку гри, заголовок, категорію, список тегів та дату публікації. Візуальний інтерфейс головної сторінки з каталогом ігрових новин наведено на рисунку 3.1.

Рисунок 3.1 - Головна сторінка клієнтського інтерфейсу GamePulse CMS

Змн. Арк. No докум. Підпис Дата

Арк.

36КБР:ICT-12.00.00.000 ПЗ

Для оптимізації швидкодії при великій кількості публікацій на головній сторінці реалізовано механізм пагінації (посторінкової навігації). За допомогою вбудованого класу Paginator фреймворку Django сервер віддає клієнту лише обмежену кількість карток (наприклад, 10 статей на сторінку). Сторінка детального перегляду статті (article\_detail.html) генерується динамічно на основі унікального ідентифікатора (ID або Slug) публікації.

Шаблон отримує об'єкт статті та виводить відформатований текст за допомогою фільтру `|safe`, який дозволяє браузеру коректно інтерпретувати HTML-теги, згенеровані раніше у WYSIWYG-редакторі (жирний текст, заголовки, вбудовані зображення та відео). Інтерфейс перегляду конкретної публікації продемонстровано на рисунку 3.2.

Рисунок 3.2 - Інтерфейс детального перегляду ігрової публікації

Змн. Арк. No докум. Підпис Дата

Арк.

37КБР:ICT-12.00.00.000 ПЗ

Важливим аспектом розробки сучасних платформ для ігрової індустрії є забезпечення бездоганного користувацького досвіду (UX) на мобільних пристроях. За статистикою, значна частина геймерів перевіряє патчноути та ігрові новини зі смартфонів під час ігрових сесій на ПК або консолях. Завдяки використанню сіткової архітектури Bootstrap 5, розроблений фронтенд є повністю адаптивним (Responsive). При зменшенні ширини вікна перегляду (Viewport) нижче 768 пікселів (контрольна точка md), багатоклонкова структура каталогу новин автоматично трансформується в одну колонку, що максимізує читабельність контенту. Головне навігаційне меню при цьому згортається у компактний компонент «гамбургер» (Offcanvas Menu), звільняючи екранний простір. Коректність відображення мобільної версії клієнтського інтерфейсу продемонстровано на рисунку 3.3.

Змн. Арк. No докум. Підпис Дата

Арк.

38КБР:ICT-12.00.00.000 ПЗ

Рисунок 3.3 - Відображення ігрової платформи на мобільних пристроях (Responsive Design)

Окрім адаптивності, на клієнтській стороні реалізовано візуальний інтерфейс багатофакторної фільтрації. Користувач має можливість перемикатися між ігровими категоріями за допомогою інтерактивної панелі навігації. При виборі певної категорії (наприклад, «Стратегії»), клієнтський додаток формує GET-запит із відповідним URL-параметром. Сервер обробляє

Змн. Арк. No докум. Підпис Дата

Арк.

39КБР:ICT-12.00.00.000 ПЗ

цей запит за допомогою методу `filter(category__slug=...)` і повертає відфільтрований набір (QuerySet). Інтерфейс сторінки з відфільтрованим за категорією контентом зображено на рисунку 3.4.

Рисунок 3.4 - Візуалізація результатів застосування фільтру за ігровою категорією

Змн. Арк. No докум. Підпис Дата

Арк.

40КБР:ICT-12.00.00.000 ПЗ

3.3Розробка та тестування API для обміну даними із зовнішніми ігровими лаунчерами

Ключовою вимогою до сучасної Headless CMS є забезпечення омніканальності — здатності передавати структурований контент на різні платформи. У межах розроблюваної системи GamePulse CMS було спроєктовано та реалізовано прикладний програмний інтерфейс (REST API) для можливості інтеграції новинної стрічки безпосередньо у клієнтські ігрові лаунчери або зовнішні вебдодатки (наприклад, на базі бібліотеки React [18]). Реалізація API базується на використанні бібліотеки Django REST Framework (DRF) [8]. Основним завданням на цьому етапі стала серіалізація складних об'єктів бази даних (статей, категорій та масивів тегів) у стандартизований формат JSON, який легко інтерпретується будь-якою мовою програмування на стороні клієнта. Для сутності «Стаття» було створено клас `ArticleSerializer`, який успадковує `serializers.ModelSerializer`. Оскільки ігрові лаунчери зазвичай потребують не лише текстових даних, а й повних URL-адрес до обкладинок та медіафайлів, у серіалізаторі було перевизначено методи відображення зв'язаних полів. Фрагмент програмного коду, що відповідає за серіалізацію ігрового контенту (файл `serializers.py`), наведено в лістингу 3.2.

Лістинг 3.2 - Програмна реалізація серіалізатора публікацій для REST API

13 from rest\_framework import serializers

```

from .models import Article, Category
class CategorySerializer(serializers.ModelSerializer):
    class Meta:
        model = Category
        fields = ['id', 'name', 'slug']
class ArticleSerializer(serializers.ModelSerializer):
    category = CategorySerializer(read_only=True)

```

Змн. Арк. No докум. Підпис Дата

Арк.

41КБР:ICT-12.00.00.000 ПЗ

author\_name =

serializers.CharField(source='author.username', read\_only=True)

class Meta:

model = Article

fields = [

'id', 'title', 'category', 'author\_name',

'content', 'tags', 'status', 'created\_at'

]

def to\_representation(self, instance):

representation = super().to\_representation(instance)

# Форматування дати для зручного відображення в ігровому

лаунчері

representation['created\_at'] =

instance.created\_at.strftime("%Y-%m-%d %H:%M")

return representation

Для обробки мережевих HTTP-запитів було розроблено набір контролерів

(Views) на базі APIView та viewsets.ModelViewSet. Це дозволило автоматично

згенерувати стандартні точки доступу (endpoints) для операцій CRUD

(створення, читання, оновлення, видалення). Основна точка доступу GET

/api/v1/articles/ налаштована на повернення списку лише тих публікацій, які

пройшли модерацию та мають статус «Опубліковано».

Тестування розробленого API проводилося за допомогою інструментарію

Postman. Під час тестування перевірялася коректність формування JSON-

відповідей, швидкість обробки запитів та правильність роботи фільтрів

(наприклад, запит GET /api/v1/articles/?category=trg успішно повертає масив

статей виключно вказаної категорії). Інтегрований механізм пагінації DRF

успішно розділяє великі масиви даних на сторінки, що гарантує стабільну

роботу клієнтських додатків навіть при різкому збільшенні кількості

публікацій у базі даних.

Змн. Арк. No докум. Підпис Дата

Арк.

42КБР:ICT-12.00.00.000 ПЗ

Забезпечення стабільності інтеграційних процесів неможливе без

проведення комплексного тестування точок доступу REST API. Для

візуального моделювання запитів від сторонніх ігрових систем було

використано середовище розробки Postman.

Процес тестування розпочався з перевірки точки доступу для отримання

масиву публікацій. На рисунку 3.5 наведено результати виконання HTTP-

запиту GET /api/v1/articles/. На скріншоті чітко видно успішний код відповіді

сервера (Status: 200 OK), час обробки транзакції (менше 50 мс, що підтверджує

ефективність обраної СУБД PostgreSQL/SQLite) та структуру отриманого

JSON-об'єкта. У тілі відповіді наявні ключі пагінації (count, next) та

серіалізований масив ігрового контенту з розгорнутими об'єктами категорій.

Рисунок 3.5 - Тестування точки доступу API для експорту списку новин у

форматі JSON

Змн. Арк. No докум. Підпис Дата

Арк.

43КБР:ICT-12.00.00.000 ПЗ

Окремому тестуванню підлягали механізми безпеки API. При спробі

симулювати POST-запит від зовнішньої системи для створення нової статті

без передачі авторизаційного токена, серверна частина на базі Django REST

Framework коректно перериває обробку запиту. Аналіз результату такого

тесту фіксує перехоплення винятку AuthenticationFailed і повернення HTTP-

статусу 401 Unauthorized або 403 Forbidden із повідомленням про відмову в

доступі, що унеможливорює несанкціонований спам у базу даних системи.

3.4Інструкція з експлуатації та адміністрування розробленої

платформи

Завершальним етапом розробки є підготовка експлуатаційної документації.

Оскільки GamePulse CMS має клієнт-серверну архітектуру, інструкція користувача поділяється на два логічні блоки: посібник системного адміністратора (розгортання серверної частини) та посібник контент-менеджера (робота з публікаціями).

Для запуску інформаційної системи на локальному комп'ютері або сервері під управлінням ОС Windows необхідно мати попередньо встановлений інтерпретатор Python версії 3.10 або вище. Процес розгортання починається зі створення ізолюваного віртуального середовища, що дозволяє уникнути конфліктів версій залежностей. У терміналі командного рядка необхідно виконати наступну команду: `python -m venv venv`

Після цього віртуальне середовище потрібно активувати:

```
venv\Scripts\activate
```

Наступним кроком є встановлення всіх необхідних бібліотек (включно з Django, Django REST Framework та CKEditor), які зафіксовані у файлі залежностей. Встановлення виконується пакетним менеджером pip: `pip install -r requirements.txt`

Змн. Арк. No докум. Підпис Дата

Арк.

44КБР:ICT-12.00.00.000 ПЗ

Для ініціалізації бази даних SQLite та створення необхідних таблиць згідно з розробленими моделями виконуються команди міграції: `python manage.py makemigrations python manage.py migrate`

Для отримання доступу до панелі управління контентом необхідно створити обліковий запис головного адміністратора (Superuser). Процес ініціюється командою `python manage.py createsuperuser`, після чого система послідовно запитає ім'я користувача, електронну пошту та пароль. Інтерфейс терміналу під час розгортання проєкту та успішного застосування міграцій наведено на рисунку 3.6.

Рисунок 3.6 - Процес ініціалізації бази даних та запуску сервера у терміналі  
Після успішного виконання вищезазначених кроків, сервер запускається командою `python manage.py runserver`. Система стає доступною за локальною мережевою адресою <http://127.0.0.1:8000/>.

Після завершення етапу низькорівневого розгортання та конфігурації серверної архітектури система стає повністю готовою до експлуатації

Змн. Арк. No докум. Підпис Дата

Арк.

45КБР:ICT-12.00.00.000 ПЗ

кінцевими користувачами — редакторами, авторами та адміністраторами контенту. Взаємодія з вебсистемою GamePulse CMS реалізується через веббраузер і не вимагає від персоналу навичок програмування чи роботи з терміналом.

Важливим етапом експлуатації системи, який передуює процесу створення контенту, є налаштування середовища та управління персоналом. Головний адміністратор (Superuser) отримує у своє розпорядження розширений блок управління авторизацією в адміністративній панелі Django.

Для підключення нового редактора до платформи адміністратор не взаємодіє з базою даних безпосередньо через SQL-запити. Натомість використовується візуальний модуль управління користувачами.

Адміністратор створює новий обліковий запис співробітника, заповнюючи його персональні дані. Ключовим етапом є призначення прав: новому контент-менеджеру обов'язково встановлюється прапорець Staff status (Статус персоналу), який дозволяє йому входити до захищеної зони панелі управління. Для детального розмежування доступу адміністратор використовує систему груп (Groups) та специфічних привілеїв (User Permissions).

Наприклад, молодшому автору можна надати привілеї `content_manager | article | Can add article` та `content_manager | article | Can change article`, але не надати права на видалення (`Can delete article`), що мінімізує ризики випадкової або навмисної втрати ігрового контенту. Інтерфейс конфігурації прав доступу співробітників наведено на рисунку 3.7.

Змн. Арк. No докум. Підпис Дата

Арк.

46КБР:ICT-12.00.00.000 ПЗ

Рисунок 3.7 - Інтерфейс управління персоналом та розмежування прав доступу (RBAC)

Процес адміністрування та наповнення платформи ігровим контентом починається з проходження процедури авторизації. Для цього користувач повинен перейти за захищеним URL-адресом `http://127.0.0.1:8000/admin/`. Безпека цього інтерфейсу на рівні ядра Django забезпечується інтегрованим проміжним програмним забезпеченням (Middleware), зокрема `AuthenticationMiddleware` та `SessionMiddleware`. Вони контролюють стан сесії користувача, перевіряють наявність активних кукі (Session Cookies) та автентифікаційних токенів.

Форма входу містить поля для введення унікального імені користувача (або електронної пошти) та пароля. Криптографічний захист під час передачі цих даних реалізується шляхом обов'язкової генерації та перевірки криптографічного CSRF-токену (Cross-Site Request Forgery), що повністю унеможливорює атаки типу міжсайтової підробки запитів. Перевірка пароля

Змн. Арк. No докум. Підпис Дата

Арк.

47КБР.ІСТ-12.00.00.000 ПЗ

здійснюється шляхом порівняння його хеш-значення, обчисленого за алгоритмом PBKDF2 з ітеративним соленням, із хешем, що зберігається в таблиці `auth_user` бази даних. У разі успішної автентифікації користувач перенаправляється на головний робочий простір (Dashboard) системи. Перед безпосереднім створенням новинних матеріалів чи патчноутів інформаційна архітектура системи вимагає наявності відповідних таксономій — ігрових категорій. У лівому навігаційному меню адміністративної панелі виведено блоки управління моделями. Контент-менеджер має можливість взаємодіяти з таблицею категорій через інтерфейс додатку `content_manager`. При створенні нової категорії (наприклад, «RPG», «Шутери», «Стратегії») оператор заповнює поле назви. Система в асинхронному режимі (за допомогою вбудованих JavaScript-скриптів Django Admin) автоматично генерує значення для поля `slug` — людино-зрозумілого URL-ідентифікатора (наприклад, назва «Екшен-ігри» трансформується в латинізований рядок `ekshen-igry`). Це поле є унікальним індексом у базі даних і використовується для оптимізації маршрутизації (Routing) при формуванні вебсторінок фільтрації на фронтенді, забезпечуючи високі показники пошукової оптимізації (SEO).

Центральною операційною процедурою в експлуатації системи є життєвий цикл публікації ігрової статті. Алгоритм дій контент-менеджера для додавання нового матеріалу деталізовано за такими етапами:

1. Ініціалізація форми: У блоці «Ігрові статті» оператор натискає кнопку «Додати». Бекенд-система генерує HTML-форму на основі класу `ModelForm`, зв'язаного з нашою сутністю `Article`.
2. Заповнення первинних реляційних полів: Контент-менеджер вводить текстовий заголовок публікації. У полі «Категорія» (яке на рівні бази даних є зовнішнім ключем `ForeignKey`) система пропонує динамічний випадальний список усіх існуючих категорій, вилучених з бази даних за допомогою ORM-запиту `Category.objects.all()`. Також вказується автор

Змн. Арк. No докум. Підпис Дата

Арк.

48КБР.ІСТ-12.00.00.000 ПЗ

матеріалу (за замовчуванням підставляється поточний авторизований користувач).

3. Робота у візуальному WYSIWYG-середовищі: Замість стандартного текстового поля для введення основного тексту статті інтегровано Rich Text Editor на базі CKEditor. Інтерфейс редактора надає контент-менеджеру розширений інструментарій для якісної верстки матеріалу:
  - о Можливість структурування тексту за допомогою ієрархічних HTML-заголовків (`<h2>`, `<h3>`), що вкрай важливо для великих за обсягом оглядів ігор.
  - о Інструменти створення маркованих та нумерованих списків, які є базовим елементом оформлення технічної документації та списків змін у патчноутах ігор.
  - о Модуль інтеграції медіаконтенту: через діалогове вікно редактор може завантажувати зображення (скріншоти, концепт-арти) безпосередньо на сервер. Серверний скрипт приймає файл,

перевіряє його MIME-тип (дозволені лише формати .png, .jpg, .webp), обмежує максимальний розмір файлу для запобігання переповнення дискового простору, генерує унікальне ім'я файлу (UUID) для уникнення колізій та зберігає його у директорію /media/uploads/. Посилання на файл автоматично вбудовується в теґ <img>, всередині тексту публікації.

4. Конфігурація пошукових теґів: Особливістю архітектури GamePulse CMS є збереження теґів у форматі масиву JSON. Контент-менеджер вводить ключові слова у спеціалізоване поле. Перед відправкою даних на сервер клієнтський скрипт перевіряє синтаксичну коректність структури. Формат введення має вигляд масиву рядків (наприклад, ["action", "rpg", "patch-1-02"]). На етапі збереження вбудований валідатор Django перевіряє відповідність типу даних структури

Змн. Арк. No докум. Підпис Дата

Арк.

49КБР:ICT-12.00.00.000 ПЗ

JSONField, трансформуючи рядок у бінарний об'єкт JSONB для PostgreSQL.

5. Управління станами та модерація контенту: Система підтримує конвеєр станів (State Machine) для публікацій через поле status. За замовчуванням новій статті присвоюється статус «Чернетка» (Draft). У цьому стані стаття доступна для редагування лише автору та не відображається у публічному API. Після завершення написання редактор змінює статус на «На модерації» (Review) — матеріал блокується для автора та стає доступним для перевірки головним редактором. Після фінального схвалення виставляється статус «Опубліковано» (Published). В цей же момент сервер оновлює часову мітку updated\_at, скидає кеш відповідних API-ендпойнтів, і стаття стає доступною для зчитування зовнішніми клієнтами та ігровими лаунчерами.

Візуальний інтерфейс адміністративної панелі в момент реалізації описаного технологічного процесу створення та верстки ігрової статті з акцентом на інтегроване WYSIWYG-середовище представлено на рисунку 3.8.

Рисунок 3.8 - Інтерфейс створення нової публікації в системі управління контентом

Змн. Арк. No докум. Підпис Дата

Арк.

50КБР:ICT-12.00.00.000 ПЗ

Одразу після збереження система автоматично застосовує зміни до бази даних, і нова стаття миттєво стає доступною для читачів на головній сторінці клієнтського інтерфейсу, а також стає доступною через REST API для зовнішніх ігрових лаунчерів.

3.5Тестування, оптимізація продуктивності та безпека вебдодатка

Розробка інформаційної системи для ігрової індустрії вимагає особливої уваги до питань надійності, швидкодії під навантаженням та стійкості до кібератак. Після завершення етапу програмної реалізації GamePulse CMS було проведено комплекс заходів із забезпечення якості програмного продукту (Quality Assurance).

Для верифікації бізнес-логіки системи було застосовано методологію модульного тестування (Unit Testing) з використанням вбудованого модуля django.test.TestCase [21]. Тестове покриття охоплює критично важливі вузли системи: створення користувачів, призначення ролей та управління статусами публікацій.

Було розроблено набір автоматизованих тестів для перевірки обмежень доступу. Наприклад, тест test\_draft\_article\_accessibility перевіряє, що неавторизований користувач або зовнішній API-запит отримує код помилки 404 Not Found при спробі звернутися до статті, яка має статус «Чернетка». Такий підхід гарантує, що неопублікований ігровий контент не потрапить до публічного доступу (наприклад, передчасний злив патчноуту гри). Тестування REST API проводилося за допомогою бібліотеки rest\_framework.test.APIClient, що дозволило імітувати HTTP-запити від ігрових лаунчерів та перевіряти структуру JSON-відповідей на відповідність розробленій специфікації.

Для наочної демонстрації підходів до забезпечення якості (QA) наведемо

фрагмент програмного коду автоматизованих випробувань з модуля tests.py.

Змн. Арк. No докум. Підпис Дата

Арк.

51КБР.ІСТ-12.00.00.000 ПЗ

Було розроблено клас ArticleAPITestCase, що наслідує базовий інструментарій APITestCase з екосистеми Django REST Framework.

Основна мета наведеного тестового сценарію — верифікувати ізоляцію чорнових записів на рівні API. Алгоритм тесту програмно створює тестового користувача, генерує ігрову публікацію зі статусом «Чернетка» (draft), а потім імітує анонімний HTTP GET-запит до ендпоінту цієї статті. Завершується тест перевіркою (Assertion) того, що сервер повернув код 404 Not Found, а не конфіденційний контент. Програмну реалізацію цього процесу наведено в лістингу 3.3.

Лістинг 3.3 - Програмна реалізація автоматизованого тестування доступу до API

```
11 from rest_framework.test import APITestCase
11 from rest_framework import status from django.contrib.auth.models import User
from content_manager.models import Article, Category
class ArticleAPITestCase(APITestCase):
def setUp(self):
self.user =
User.objects.create_user(username='test_author',
password='password123')
self.category =
Category.objects.create(name='Оновлення та патчі', slug='patch-
notes')
self.draft_article = Article.objects.create(
title='Секретний патчноут S.T.A.L.K.E.R. 2',
category=self.category,
author=self.user,
content=,
status='draft'
)
def test_draft_article_accessibility_via_api(self):
(лаунчера)
url = f'/api/v1/articles/{self.draft_article.id}/'
response = self.client.get(url)
self.assertEqual(response.status_code,
status.HTTP_404_NOT_FOUND)
def test_published_articles_list(self):
response = self.client.get('/api/v1/articles/')
self.assertEqual(response.status_code,
status.HTTP_200_OK)
```

Змн. Арк. No докум. Підпис Дата

Арк.

52КБР.ІСТ-12.00.00.000 ПЗ

Запуск розробленого набору тестів здійснюється через інтегровану утиліту командного рядка manage.py test. Під час виконання цієї команди Django автоматично створює ізольовану тестову базу даних, застосовує всі міграції, виконує тестові сценарії та знищує тестову базу після їх завершення, що гарантує відсутність «сміттєвих» даних у робочій базі платформи (Production DB). Результат успішного проходження модульних тестів (Status: OK), який підтверджує коректну роботу механізмів авторизації та маршрутизації бекенду, зафіксовано на рисунку 3.9.

Рисунок 3.9 - Результати успішного виконання модульних тестів у консолі сервера

Платформи в ігровій індустрії часто стикаються з різкими стрибками трафіку (наприклад, у день релізу довгоочікуваної гри). Для забезпечення стабільної швидкодії системи було вирішено проблему «N+1 запитів», яка є типовим вузьким місцем у системах з ORM (Object-Relational Mapping). При виведенні списку статей на головній сторінці, окрім самої статті, системі необхідно отримати дані про автора та категорію. Замість виконання

Змн. Арк. No докум. Підпис Дата

Арк.

53КБР:ICT-12.00.00.000 ПЗ

окремого SQL-запиту для кожної публікації, у контролерах (views.py) було застосовано методи оптимізації select\_related('category', 'author'). Це дозволило отримувати всі пов'язані дані за один ресурсоефективний SQL-запит із використанням INNER JOIN, що зменшило навантаження на базу даних PostgreSQL більш ніж на 70% при масових запитах до API. Крім того, для статичних медіафайлів (обкладинок ігор, скріншотів) налаштовано політику довгострокового кешування на стороні клієнтського браузера. Архітектура GamePulse CMS спроектована з урахуванням сучасних стандартів кібербезпеки та рекомендацій консорціуму OWASP [20]. Оскільки система працює з контентом (статтями), ключовим ризиком є атаки типу XSS (Cross-Site Scripting), коли зловмисник намагається вбудувати шкідливий JavaScript-код у текст публікації.

Для нівелювання цього ризику в інтегрованому редакторі CKEEditor та на рівні шаблонів Django активовано сувору санітизацію вхідних даних (Data Sanitization). Усі небезпечні HTML-теги (наприклад, <script>) автоматично екрануються або видаляються перед збереженням у базу. Додатково, використання системи об'єктно-реляційного відображення (Django ORM) повністю унеможливило впровадження SQL-ін'єкцій (SQLi), оскільки фреймворк автоматично параметризує всі запити до бази даних. Від атак типу підробки міжсайтових запитів (CSRF) захищені всі форми зміни стану системи завдяки обов'язковій генерації та валідації унікальних токенів {% csrf\_token %} на рівні бекенду. Проведений комплекс заходів з тестування, оптимізації та безпеки підтверджує високу надійність розробленої системи та її готовність до впровадження в реальні бізнес-процеси ігрової індустрії.

Змн. Арк. No докум. Підпис Дата

Арк.

54КБР:ICT-12.00.00.000 ПЗ

ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи було успішно вирішено актуальне науково-практичне завдання: спроектовано та програмно реалізовано спеціалізовану інформаційну систему управління контентом (Headless CMS) для потреб онлайн-платформ в ігровій індустрії. Розроблений програмний продукт GamePulse CMS забезпечує ефективну роботу з ігровими публікаціями та їх омніканальну доставку.

За підсумками проведених досліджень, архітектурного проєктування та написання програмного коду можна зробити наступні висновки:

1. Проведено системний аналіз предметної області та інформаційних процесів у сфері ігрових медіа. Встановлено, що перехід ігрової індустрії до моделі «гра як сервіс» (GaaS) вимагає безперервного інформування спільноти гравців. Доведено, що використання класичних монолітних CMS (орієнтованих виключно на генерацію HTML-сторінок) є неефективним через неможливість прямої інтеграції контенту в ігрові лаунчери та мобільні додатки-супутники.

Обґрунтовано необхідність використання архітектурного патерну Headless CMS, який базується на принципі API-first та чіткому розділенні рівня зберігання даних і рівня їх представлення.

2. Сформовано комплекс функціональних і нефункціональних вимог та обрано оптимальний стек технологій. Для реалізації бекенд-частини системи обрано мову програмування Python та серверний фреймворк Django, що дозволило забезпечити високий рівень безпеки та швидку генерацію адміністративної панелі. Для управління даними використано об'єктно-реляційну СУБД PostgreSQL (SQLite для локального тестування). Зовнішню взаємодію реалізовано через REST API, а клієнтський вебінтерфейс розроблено з використанням адаптивного фреймворку Bootstrap 5.

3. Виконано інфологічне проєктування та розроблено оптимізовану реляційну модель бази даних. Побудовано концептуальну (ERD) та

Змн. Арк. No докум. Підпис Дата

Арк.

55КБР:ICT-12.00.00.000 ПЗ

логічну моделі даних, які приведено до третьої нормальної форми (3НФ). Для організації спільної роботи контент-менеджерів спроектовано асоціативні таблиці з підтримкою рольової моделі (Автори, Редактори, Модератори). Запропоновано та реалізовано

нестандартне архітектурне рішення для системи тегів: використання вбудованого типу JSONField (масивів) у PostgreSQL замість створення окремих зв'язкових таблиць, що дозволило уникнути ресурсомістких операцій JOIN та значно пришвидшити багатофакторну фільтрацію ігрових новин.

4. Програмно реалізовано джерело істини (Backend) та інтегровано інструменти візуального верстання. Розроблено комплекс моделей даних та контролерів на базі патерну MVT (Model-View-Template). Для вирішення проблеми складного форматування ігрових оглядів та патчнотів у середовищі Django було успішно інтегровано WYSIWYG-редактор CKEditor. Налаштовано автоматизовану систему маршрутизації медіафайлів (скріншотів, відео) та впроваджено конвеєр станів публікацій (Чернетка — На модерації — Опубліковано).

5. Спроектовано та протестовано прикладний програмний інтерфейс (RESTAPI). За допомогою бібліотеки Django REST Framework створено захищені точки доступу (endpoints) для експорту контенту. Розроблено кастомні серіалізатори об'єктів бази даних у машиночитаний формат JSON. Впроваджено механізми серверної пагінації та фільтрації, що гарантує стабільну передачу даних до сторонніх клієнтських додатків навіть при масштабуванні бази публікацій.

6. Створено ергономічний клієнтський вебінтерфейс (Frontend). Розроблено ієрархію HTML-шаблонів з використанням сіткової системи Bootstrap 5. Інтерфейс адаптовано під цільову аудиторію геймерів: впроваджено темну тему оформлення (Dark Mode), спроектовано інтерактивні новинні картки з метаданими та забезпечено повну кросплатформну адаптивність для мобільних пристроїв.

Змн. Арк. No докум. Підпис Дата

Арк.

56КБР:ICT-12.00.00.000 ПЗ

7. Проведено оптимізацію продуктивності та забезпечено захист вебдодатка від кіберзагроз. Вирішено класичну проблему «N+1 запитів» шляхом застосування методів select\_related на рівні Django ORM, що знизило навантаження на базу даних під час масових запитів на 70%. Відповідно до стандартів безпеки OWASP Top-10, у системі реалізовано захист від XSS-атак (через санітизацію вхідних HTML-даних), SQL-ін'єкцій (параметризація запитів в ORM) та CSRF-атак (обов'язкова валідація криптографічних токенів для всіх форм). Практичне значення отриманих результатів полягає у тому, що створена архітектура та програмний код системи GamePulse CMS є готовим MVP-рішенням (Minimum Viable Product). Продукт може бути впроваджений у реальні бізнес-процеси ігрових студій для підтримки життєвого циклу їхніх проєктів або використаний як технологічна основа для незалежних ігрових медіапорталів.

Перспективи подальшого розвитку системи передбачають розширення функціоналу через інтеграцію протоколу WebSockets для реалізації можливості одночасного (спільного) редагування статей у реальному часі, а також налаштування пайплайнів CI/CD для автоматизованого розгортання оновлень у хмарних середовищах.

Змн. Арк. No докум. Підпис Дата

Арк.

57КБР:ICT-12.00.00.000 ПЗ

#### СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Глибовець М. М. Бази даних. Основи проєктування та використання : навч. посіб. / М. М. Глибовець. — К. : Києво-Могилянська академія, 2017. — 304 с.
2. Пасічник В. В. Організація баз даних та знань : підручник / В. В. Пасічник, В. А. Резніченко. — К. : Видавнича група BVH, 2006. — 384 с.
3. Рубан В. Я. Проєктування веб-систем : навч. посібник / В. Я. Рубан. — Харків : ХНУРЕ, 2019. — 185 с.
4. Трохимчук Р. М. Основи програмування мовою Python : навч. посіб. / Р. М. Трохимчук. — Київ : КПІ ім. Ігоря Сікорського, 2018. — 225 с.
5. Vincent W. S. Django for APIs : Build web APIs with Python and Django. 4th ed. / William S. Vincent. — WelcomeToCode, 2022. — 354 p.
6. Richardson L. RESTful Web APIs : Services for a Changing World / Leonard

Richardson, Mike Amundsen. — O'Reilly Media, 2013. — 406 p.

7. Django Documentation [Електронний ресурс] / Django Software Foundation. — Режим доступу : <https://docs.djangoproject.com/>

8. Django REST Framework Documentation [Електронний ресурс] / Encode OSS. — Режим доступу : <https://www.django-rest-framework.org/>

9. SQLite Official Documentation [Електронний ресурс] / SQLite Consortium. — Режим доступу : <https://www.sqlite.org/docs.html>

10. Bootstrap 5 Documentation [Електронний ресурс] / The Bootstrap Authors. — Режим доступу : <https://getbootstrap.com/docs/5.0/getting-started/introduction/>

11. Directus Self-Hosted Pricing and Architecture [Електронний ресурс] / Directus. — Режим доступу : <https://directus.io/pricing/self-hosted>

12. Payload CMS Admin Overview [Електронний ресурс] / Payload. — Режим доступу : <https://payloadcms.com/docs/admin/overview>

Змн. Арк. No докум. Підпис Дата

Арк.

58КБР.ІСТ-12.00.00.000 ПЗ

13. Maj A. PayloadCMS from scratch: Project setup [Електронний ресурс] / Adrian Maj. — 2023. — Режим доступу : <https://adrianmaj.com/en/posts/payloadcms-from-scratch-1-project-setup>

14. Maj A. PayloadCMS from scratch: Collections and globals [Електронний ресурс] / Adrian Maj. — 2023. — Режим доступу : <https://adrianmaj.com/en/posts/payloadcms-from-scratch-2-collections-and-globals>

15. Maj A. PayloadCMS from scratch: Users and access control [Електронний ресурс] / Adrian Maj. — 2023. — Режим доступу : <https://adrianmaj.com/en/posts/payloadcms-from-scratch-3-users-and-access-control>

16. Ульянов А. В. Розробка веб-додатків з використанням мови Python та фреймворку Django : навч. посіб. / А. В. Ульянов. — Київ : КПІ ім. Ігоря Сікорського, 2021. — 145 с.

17. CKEditor 5 Documentation [Електронний ресурс] / CKSource Holding. — Режим доступу : <https://ckeditor.com/docs/>

18. React – A JavaScript library for building user interfaces [Електронний ресурс] / Meta Platforms. — Режим доступу : <https://react.dev/>

19. MDN Web Docs: HTML & CSS [Електронний ресурс] / Mozilla Foundation. — Режим доступу : <https://developer.mozilla.org/>

20. OWASP Top 10: The Ten Most Critical Web Application Security Risks [Електронний ресурс] / OWASP Foundation. — Режим доступу : <https://owasp.org/www-project-top-ten/>

21. Testing Django [Електронний ресурс] / Django Software Foundation. — Режим доступу : <https://docs.djangoproject.com/en/4.2/topics/testing/>

58

ДОДАТКИ

59

Додаток А

Вміст файлу models.py

34. from django.db import models from django.contrib.auth.models import User

from ckeditor.fields import RichTextField

5. class Category(models.Model):

name = models.CharField(max\_length=100, verbose\_name="Назва ігрової категорії")

slug = models.SlugField(unique=True, blank=True) class Meta: verbose\_name = "Ігрова категорія"

verbose\_name\_plural = "Ігрові категорії"

7. def \_\_str\_\_(self): return self.name class Article(models.Model):

STATUS\_CHOICES = (

('draft', 'Чернетка'),

('review', 'На модерації'),

('published', 'Опубліковано'),

)

5. title = models.CharField(max\_length=255, verbose\_name="Заголовок публікації")

28. category = models.ForeignKey(Category, on\_delete=models.CASCADE, related\_name='articles')

28. author = models.ForeignKey(User, on\_delete=models.SET\_NULL, null=True, related\_name='posts')

```

content = RichTextField(verbose_name="Текст статті")
tags = models.JSONField(default=list, blank=True, verbose_name="Пошукові
теги")
status = models.CharField(max_length=20, choices=STATUS_CHOICES,
default='draft')

```

60

Продовження додатку A

```

21 created_at = models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)
class Meta:
ordering = ['-created_at']
verbose_name = "Ігрова стаття"
verbose_name_plural = "Ігрові статті"
8 def __str__(self):
return self.title
8 Вміст файлу serializers.py from rest_framework import serializers from models import Article, Category
class CategorySerializer(serializers.ModelSerializer): class Meta: model = Category
fields = ['id', 'name', 'slug']
class ArticleSerializer(serializers.ModelSerializer):
category = CategorySerializer(read_only=True)
author_name = serializers.CharField(source='author.username',
read_only=True)
class Meta:
model = Article
fields = [
'id', 'title', 'category', 'author_name',
'content', 'tags', 'status', 'created_at'
]

```

61

Продовження додатку A

```

def to_representation(self, instance):
representation = super().to_representation(instance)
representation['created_at'] =
instance.created_at.strftime("%Y-%m-%d %H:%M")
return representation
Вміст файлу views.py
35 from django.shortcuts import render, get_object_or_404
from django.views.generic import ListView, DetailView
from rest_framework import viewsets, permissions
from models import Article, Category
from serializers import ArticleSerializer
# Контролери для рендерингу HTML-шаблонів (Frontend)
class ArticleListView(ListView):
model = Article
template_name = 'content/home.html'
context_object_name = 'articles'
paginate_by = 6
def get_queryset(self):
# Оптимізація N+1 запитів через select_related
queryset =
Article.objects.filter(status='published').select_related('category',
'author')
category_slug = self.request.GET.get('category')
if category_slug:
queryset = queryset.filter(category__slug=category_slug)
return queryset
class ArticleDetailView(DetailView):
model = Article
template_name = 'content/post_detail.html'

```

62

Продовження додатку A

```

context_object_name = 'article'
def get_object(self):
return get_object_or_404(Article, id=self.kwargs.get("pk"),

```

```

status='published')
# Контролери для обслуговування REST API ендпоінтів
class ArticleAPIViewSet(viewsets.ReadOnlyModelViewSet):
    serializer_class = ArticleSerializer
    permission_classes = [permissions.AllowAny]
    def get_queryset(self):
        queryset=
        Article.objects.filter(status='published').select_related('category',
        'author')
21 category_slug = self.request.query_params.get('category')
        search_query = self.request.query_params.get('search')
        if category_slug:
            queryset = queryset.filter(category__slug=category_slug)
        if search_query:
            queryset = queryset.filter(title__icontains=search_query)
        return queryset
Вміст файлу urls.py
from django.urls import path, include
from rest_framework.routers import DefaultRouter
from .views import ArticleListView, ArticleDetailView, ArticleAPIViewSet
router = DefaultRouter()
router.register(r'articles', ArticleAPIViewSet, basename='article-api')

63
Продовження додатку А
urlpatterns = [
    # Маршрути веб-інтерфейсу читача
    path("", ArticleListView.as_view(), name='home'),
    path('post/<int:pk>/', ArticleDetailView.as_view(), name='post_detail'),
    path('api/v1/', include(router.urls)),
]
Вміст файлу 5 admin.py
from django.contrib import admin
from .models import Article, Category
@admin.register(Category)
5 class CategoryAdmin(admin.ModelAdmin): list_display = ('name', 'slug')
    prepopulated_fields = {'slug': ('name',)}
    search_fields = ('name',)
@admin.register(Article)
class ArticleAdmin(admin.ModelAdmin):
    list_display = ('title', 'category', 'author', 'status', 'created_at')
    list_filter = ('status', 'category', 'created_at')
    search_fields = ('title', 'content')
    prepopulated_fields = {'slug': ('title',)}
    autocomplete_fields = ['category', 'author']
# Розмежування прав доступу на рівні адмін-панелі
def save_model(self, request, obj, form, change):
    if not getattr(obj, 'author', None):
        obj.author = request.user
    super().save_model(request, obj, form, change)
def get_queryset(self, request):

64
Продовження додатку А
qs = super().get_queryset(request)
# Молодші редактори бачать лише свої статті
if request.user.is_superuser:
    return qs
return qs.filter(author=request.user)

65
Додаток Б
Вміст файлу home.html
{% extends 'content_base.html' %}
{% block content %}
<div class="container mt-4">
<div class="row mb-4">

```

```

<div class="col-12">
<h1 class="text-light">Панель публікацій</h1>
<p class="text-muted">Останні патчноути, огляди та аналітика ігрової
індустрії.</p>
</div>
</div>
<div class="row row-cols-1 row-cols-md-2 g-4">{% for article in articles %}
<div class="col">
<div class="card h-100 bg-dark text-light border-secondary">
<div class="card-body">
<h5 class="card-title">
<a href="{% url 'post_detail' article.id %}" class="text-
info text-decoration-none">
{{ article.title }}
</a>
<h5>
<span class="badge bg-primary mb-
2">{{ article.category.name }}</span>
<p class="card-text text-
secondary">{{ article.content|striptags| truncatewords:30 }}</p>
</div>
<div class="card-footer border-secondary text-muted small">
Опубліковано: {{ article.created_at|date:"d.m.Y H:i" }}
</div>
</div>

```

```

66
Продовження додатку Б
</div>
{% empty %}
<div class="col-12 text-center text-muted py-5">
<h3>Публікацій не знайдено</h3>
</div>
{% endfor %}
</div>
{% if is_paginated %}
<nav class="mt-4">
<ul class="pagination justify-content-center">
{% if page_obj.has_previous %}
<li class="page-item">
<a class="page-link bg-dark text-info border-
secondary" href="?page={{ page_obj.previous_page_number }}">Назад</a>
</li>
{% endif %}
<li class="page-item disabled">
<a class="page-link bg-dark text-
light border-secondary" href="#">Сторінка {{ page_obj.number }} з
{{ paginator.num_pages }}</a>
{% if page_obj.has_next %}
<li class="page-item">
<a class="page-link bg-dark text-info border-
secondary" href="?page={{ page_obj.next_page_number }}">Наступна</a>
</li>
{% endif %}
</ul>
</nav>
{% endif %}
</div>
Вміст файлу post_detail.html
{% extends 'content/base.html' %}
{% block content %}
Продовження додатку Б

```

```

67
<div class="container mt-5 text-light">
<div class="row justify-content-center">
<div class="col-lg-10">
<div class="d-flex justify-content-end mb-3">
<a href="{% url 'home' %}" class="btn btn-outline-info btn-
sm">Назад до новин</a>
</div>
<h1 class="mb-3 text-info fw-bold">{{ article.title }}</h1>
<div class="mb-4 d-flex align-items-center flex-wrap gap-2 text-
muted small">
<span class="badge bg-primary py-2 px-
3">{{ article.category.name }}</span>
{% if article.author_name %}
<span class="me-3">Автор:

```

```
&lt;strong>{{ article.author_name }}</strong></span>
{% endif %}
<span><span>Календар: {{ article.created_at|date:"d.m.Y H:i" }}</span>
</div>
<hr class="border-secondary mb-4">
<div class="article-content fs-5 lh-base text-justify p-3 bg-
opacity-10 bg-secondary rounded">
{{ article.content|safe }}
</div>
{% if article.tags %}
<div class="mt-4 pt-3 border-top border-secondary">
<span class="text-secondary me-2 small">Ключові теги:</span>
{% for tag in article.tags %}
<span class="badge bg-secondary text-dark me-1 fw-normal">#{ tag }</span>
Продовження додатку Б
```

```
68
{% endfor %}
</div>
{% endif %}
</div>
</div>
</div>
{% endblock %}
```

69

БІБЛІОГРАФІЧНА ДОВІДКА

Тема дипломної роботи: «Проектування та розробка інформаційної системи управління контентом для онлайн-платформи в ігровій індустрії»

Обсяг бакалаврської роботи 55 сторінок.

Перелік графічного матеріалу:

- БР.ІСТ-12.00.00.000 С1 ER-діаграма реляційної бази даних ігрових публікацій.
- БР.ІСТ-12.00.00.000 С2 Архітектурна схема взаємодії підсистем та маршрутизації REST API.

Дата закінчення БР

Студент Владислав НЕЙМЕТ