

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

КУРСОВА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Бази даних»

на тему:

«База даних медіахостингу»

студента II курсу групи ПЗ-23-1
спеціальності 121 «Інженерія програмного
забезпечення»

Лебідя Едуарда Юрійовича
(прізвище, ім'я та по-батькові)

Керівник: доцент кафедри КН, к.пед.н., до-
цент Коротун Ольга Володимирівна

Дата захисту: " __ " _____ 20__ р.

Національна шкала _____

Кількість балів: _____

Оцінка: ECTS _____

Члени комісії

_____	<u>Ольга Коротун</u>
(підпис)	(прізвище та ініціали)
_____	<u>Олексій Чижмотря</u>
(підпис)	(прізвище та ініціали)
_____	<u>Світлана Кравченко</u>
(підпис)	(прізвище та ініціали)
_____	<u>Інна Сугоняк</u>
(підпис)	(прізвище та ініціали)

Житомир – 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
Факультет інформаційно-комп'ютерних технологій
Кафедра інженерії програмного забезпечення
Освітній рівень: бакалавр
Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Завідувач кафедри ІПЗ,
д.п.н., професор
_____ Ольга Коротун
«___» _____ 2024 р.

ЗАВДАННЯ
НА КУРСОВУ РОБОТУ СТУДЕНТУ
Лебідю Едуарду Юрійовичу

Тема роботи: База даних медіахостингу,
керівник роботи: доцент кафедри КН, к.пед.н., доцент Коротун Ольга Володимирівна
Строк подання студентом: «___» _____ 2024 р.

Вихідні дані до роботи: Організація бази даних медіахостингу

Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1. Постановка завдання
2. Аналіз аналогічних розробок
3. Алгоритми роботи програми
4. Опис роботи програми
5. Програмне дослідження

Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

1. Презентація до КР

2. Посилання на репозиторій: <https://git.ztu.edu.ua/ipz/2023-2027/ipz-23-1/lebid-eduard/boorudb>

Консультанти розділів роботи

Розділ	Прізвище, ініціали та посади консульта- нта	Підпис, дата	
		завдання видав	завдання прийняв

Дата видачі завдання “03” жовтня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів курсового проєктування	Строк виконання етапів роботи	Примітки
1	Початок	02.11.24	
2	Пошук та аналіз аналогів	03.11.24	
3	Формулювання технічного завдання	10.11.24	
4	Опрацювання джерел	15.11.24	
5	Проектування	25.11.24	
6	Написання програмного коду	10.12.24	
7	Тестування та доробки	18.12.24	
8	Написання пояснювальної записки	22.12.24	
9	Захист		

Студент

(підпис)

Едуард Лебідь

(прізвище та ініціали)

Керівник роботи

(підпис)

Ольга Коротун

(прізвище та ініціали)

РЕФЕРАТ

Завданням на курсовий проект (роботу) було створення бази даних для медіахостингу.

Пояснювальна записка до курсового проекту (роботи) на тему «База даних медіахостингу» складається з вступу, трьох розділів, висновків, списку використаної літератури та додатків.

Текстова частина викладена на 29 сторінках друкованого тексту.

Пояснювальна записка має 30 сторінок додатків. Список використаних джерел містить 5 найменувань і займає 1 сторінку. В роботі наведено 10 рисунків. Загальний обсяг роботи – 60 сторінок.

Ключові слова: БАЗА ДАНИХ, МЕДІАХОСТИНГ, SQL, POSTGRESQL, BOORU, МЕДІА, ЗАПИТ, ТЕГИ, СЕРВЕР.

					ДУ «Житомирська політехніка».23.121.12.000–ПЗ								
Змн.	Арк.	№ докум.	Підпис	Дата									
Розроб.		Лебідь Е.Ю.			База даних медіахостингу				Літ.	Арк.	Аркушів		
Перевір.											1	60	
Керівник		Коротун О.В.							ФІКТ Гр. ІПЗ-23-1[1]				
Н. контр.													
Зав. каф.													

Зміст

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		2

ВСТУП

Людство здавна використовувало різноманітне медіа, починаючи від наскальних малюнків та усних музичних партитур закінчуючи вже записаними фото, музикою та навіть відео, котрі можна відтворювати будь-коли. Раніше дані зберігались у матеріальному вигляді та оброблювались вручну, як-от книги у бібліотеках, чи фототека у альбомі. Але з часом даних стало настільки багато, що вручну опрацьовувати їх було надто складно, і тут людині допомогли комп'ютери. З їх приходом дані фактично поміщуються у комп'ютер, що значно зменшує займаний ними фізичний простір, а також завдяки обчислювальній потужності комп'ютерів працювати з даними стало значно легше. Згодом було розроблено бази даних та мову для роботи з ними – SQL. Зараз бази даних є невід'ємною частиною нашого життя, без них не було б можливо записатись на прийом до лікаря, подивитись відео у інтернеті чи банально зареєструватись у месенджері.

Метою цієї курсової роботи є вивчення та дослідження принципів роботи сайту-медіахостингу з базою даних, котра буде реалізовувати усі його можливості.

Об'єктом дослідження курсової роботи є узагальнення знань мови SQL та вивчення можливостей СУБД.

Предмет дослідження – база даних для сайту-медіахостингу та принципи її роботи.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ

1.1 Аналіз інформаційних потреб та визначення предметної області дослідження

Перед початком розробки слід детально дослідити те, що ви хочете реалізувати. В даному випадку це база даних для сайту-медіахостингу.

База даних для медіахостингу виконує важливу роль – вона зберігає у собі всі дані для сайту, такі як теги, дані про користувачів, коментарі, дані про зроблені пости та про файли, використані у пості.

Даною базою даних через веб-сайт мають користуватись звичайні користувачі з можливістю повної взаємодії з усією базою даних прямим (наприклад, написання коментаря) та непрямим шляхом (наприклад, коли користувач завантажує файл, дані про файл беруться автоматично з властивостей самого файлу).

Дана база даних потрібна для коректної роботи медіахостингу, тому вона містить у собі 7 таблиць: `uploader` – інформація про користувача, `content` – інформація про файл, `comments` – інформація про коментарі, `tags` – інформація про теги, `post` – інформація про пост, `post_tags` – зв'язуюча таблиця між `tags` та `posts`, щоб для кожного посту відображались лише певні теги, `post_comments` – аналогічно до `post_tags`, але у даному випадку про коментарі.

На створення даної бази даних мене надихнули платформи типу `booru`, котрі спеціалізуються на медіа з детальним його описом через теги. Найвідоміший з них – <https://danbooru.donmai.us>. Теги відображаються ліворуч, а решту місця займає пост, а під ним – коментарі (за наявності). На сайті є безліч різноманітних тегів, і кожен пост має свій набір тегів, котрі підходять для його опису (наприклад, тег «1girl» додається до постів, де наявна лише 1 дівчина). Сайт містить в собі контент для різного віку, тому до кожного посту застосовується рейтинг (наприклад, рейтинг `General` вказує на те, що пост є придатним для всіх, а тег `Sensitive` вказує на те, що деяким людям може бути некомфортно дивитись на даний пост)

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

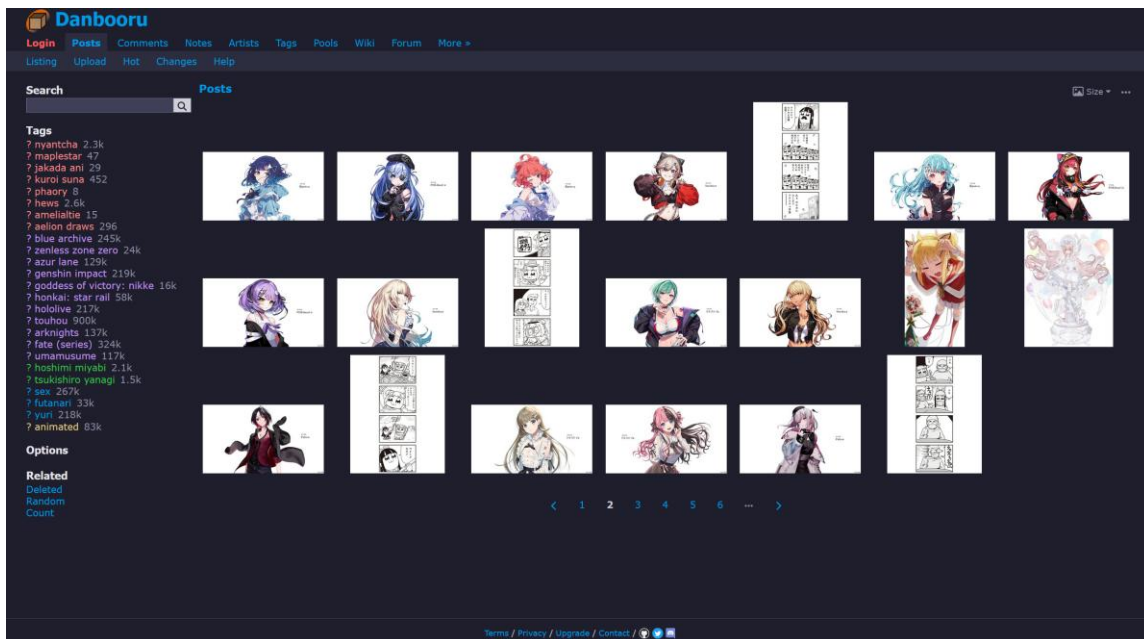


Рис. 1.1. Головна сторінка сайту

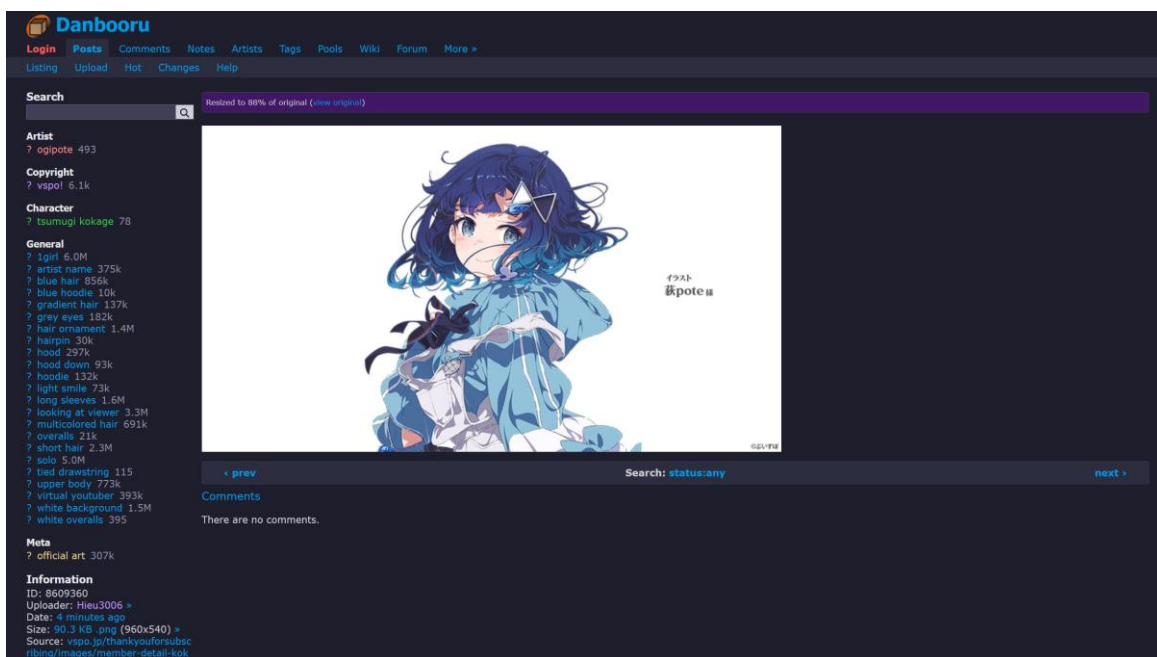


Рис. 1.2. Сторінка з постом

1.2 Обґрунтування вибору засобів реалізації

Для створення бази даних було обрано СУДБ PostgreSQL, для її реалізації – веб-сайт через їх простоту освоєння та потужний функціонал, що вони надають. Для розробки веб-сайту було використано середовище розробки WebStorm через свій широкий функціонал, можливість працювати з кількома файлами одночасно та зручними можливостями для тестування та налагодження.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до першого розділу

У першому розділі було розглянуто предмет дослідження та структуру бази даних для медіахостингу, також було обгрунтовано засоби, що будуть використуватись для роботи над проектом.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2 ПРОЕКТУВАННЯ БАЗИ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ РОБОТИ

2.1. Аналіз інформаційних процесів

Перед початком роботи слід розробити проект. Було визначено, який функціонал матиме медіахостинг:

- 1. Можливість додавання даних до бази даних через інтерфейс сайту.
- 2. Сайт бере всі дані з бази даних, окрім самих медіафайлів, на них він посилається за допомогою шляху до файлу, вказаного у базі даних.
- 3. Сайт має коректно відображати різні типи даних у своєму місці (наприклад, відображати різні категорії тегів у різних рядках).

База даних має задовольняти усім вищезазначеним вимогам.
Користувач має можливість працювати з базою даних через сайт:

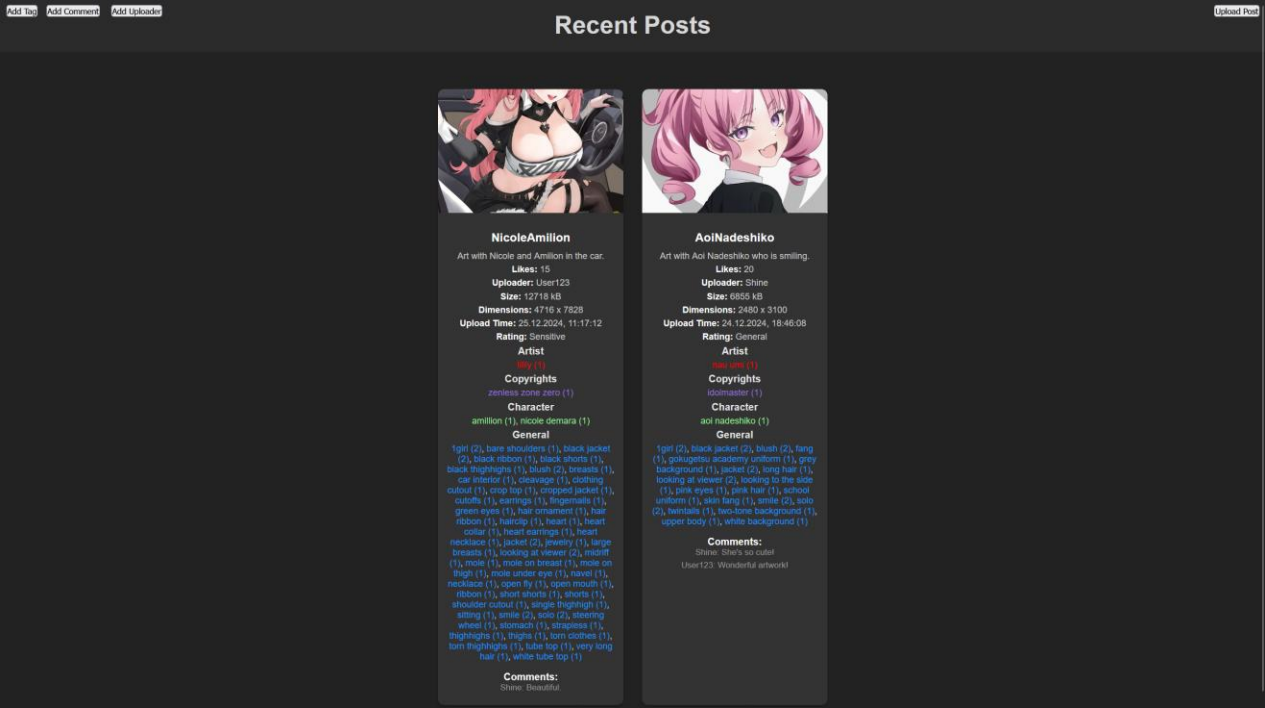


Рис. 2.1 інтерфейс сайту

На сайті відображаються картки з постами з зображеннями у верхній частині та інформацією про пост нижче (назва файлу, його опис, інформація про файл, хто завантажив та перелік тегів). Зверху наявні кнопки «Add Tag», «Add Comment», «Add Uploader» та «Upload Post», які дають можливість додати тег, коментар,

користувача та завантажити пост. Користувач вводить потрібні дані і потім натискає кнопку завантаження, що додає відповідні дані у базу даних.

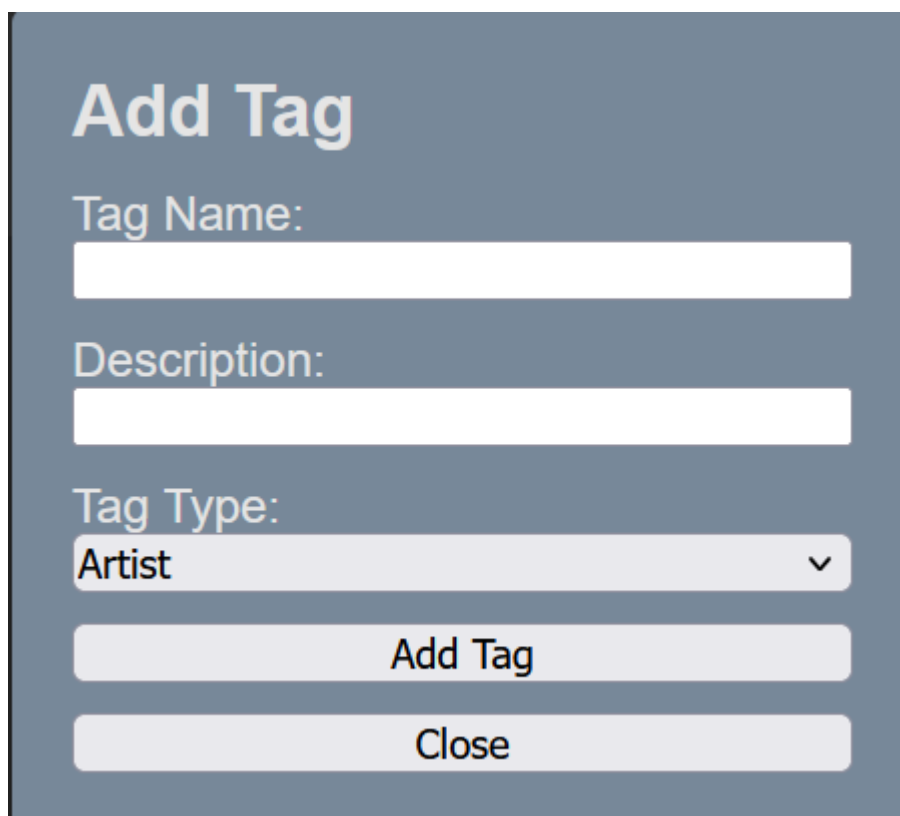


Рис. 2.2 Інтерфейс додавання тегів

Також при кліку на користувача або тег є можливість побачити відповідну інформацію:

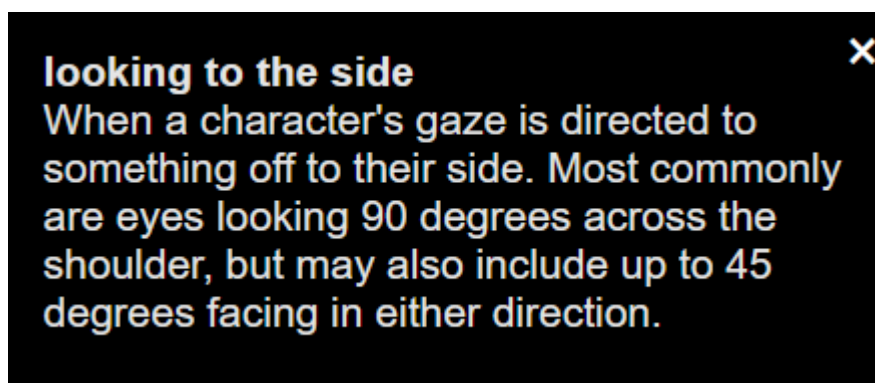


Рис. 2.3 Зображення опису тегу

2.2 Проектування структури бази даних

База даних містить у собі 7 таблиць, 5 з яких є основними:

1. content – в даній таблиці зберігається вся інформація про файл, а саме: content_id –лічильник, який надає кожному файлу унікальний ідентифікатор,

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

name – ім'я файлу, що завантажується, path_to_file – повний шлях до файлу включно з самим файлом (наприклад, "C:\Users\Shine\WebstormProjects\booruCourseProject\public\posts\AoiNadeshiko.jpg"), size – розмір файлу в кілобайтах, width та height – ширина та висота файлу відповідно у пікселях, description – опис файлу, що буде видний користувачу, type – тип файлу (false – фото, true – відео), extension – розширення файлу без крапки, length – довжина відео (null для фото), sound – наявність звуку для відео (false – без звуку (також для всіх фото), true – зі звуком), upload_time – час завантаження посту у форматі дд.мм.рррр гг:хх:сс, uploader_id – посилання на відповідне поле у таблиці uploader для присвоєння користувачу завантаженого файлу у форматі один-до-багатьох (один користувач може мати багато завантажених файлів, але кожен завантажений файл може мати лише одного користувача).

2. uploader – в даній таблиці зберігається вся інформація про користувача, а саме: uploader_id – лічильник, який надає кожному користувачу унікальний ідентифікатор, name – ім'я користувача, reg_date – час реєстрації користувача у форматі дд.мм.рррр гг:хх:сс, status – показник, чи користувач у мережі (false – офлайн, true – онлайн), is_admin – показник, чи є користувач адміністратором (false – звичайний користувач, true – адмін), description – опис профілю користувача, uploads – лічильник, що вказує на кількість завантажених користувачем постів, reputation – репутація користувача у числовому значенні (наприклад, 100 чи 50).
3. comments – в даній таблиці зберігаються написані користувачами коментарі, а саме: comment_id – лічильник, який надає кожному коментарю унікальний ідентифікатор, comment – поле, де зберігається сам коментар, uploader_id – посилання на відповідне поле у таблиці uploader для присвоєння користувачу коментаря у форматі один-до-багатьох.
4. tags – в даній таблиці зберігаються теги, а саме: tag_id – лічильник, який надає кожному тегу унікальний ідентифікатор, tag_type – тип тегу (0 – тег типу Artist (автор зображення), 1 – Copyright (серія, наприклад, комп'ютерна гра

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

«Zenless Zone Zero»), 2 – Character (персонаж, зображений на пості), 3 – General (загальний, дана категорія описує все інше, наприклад, одяг персонажів, об'єкти, як-от автомобілі і т.д.), tag – назва тегу, description – опис тегу (наприклад, тег називається «twintails», а його опис – «A hair style where the hair is tied into sections forming two ponytails.»).

5. post – в даній таблиці створюються самі пости шляхом комбінації даних з усіх вищезазначених таблиць. Дана таблиця містить наступні поля: post_id – лічильник, який надає кожному посту унікальний ідентифікатор, content_id – посилання на відповідне поле у таблиці content для присвоєння кожному посту свого файлу, likes – кількість вподобань до посту, rating – чи є пост придатним для всіх (false – рейтинг General (для всіх), true – рейтинг Sensitive (чутливий матеріал)).

Також дана база даних має 2 допоміжні таблиці для вставки тегів та коментарів до кожного посту:

1. post_comments – в даній таблиці кожному посту присвоюються коментарі, а саме: post_id – посилання на відповідне поле у таблиці post для присвоєння певному посту певного коментаря, comment_id – посилання на відповідне поле у таблиці comment для того, щоб присвоїти посту певний коментар.
2. post_tags – в даній таблиці кожному посту присвоюються теги, а саме: post_id –аналогічне використання з post_comments, tag_id – посилання на відповідне поле у таблиці tag для присвоєння посту певного тегу.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

Зв'язки між ними можна побачити на наступній діаграмі:

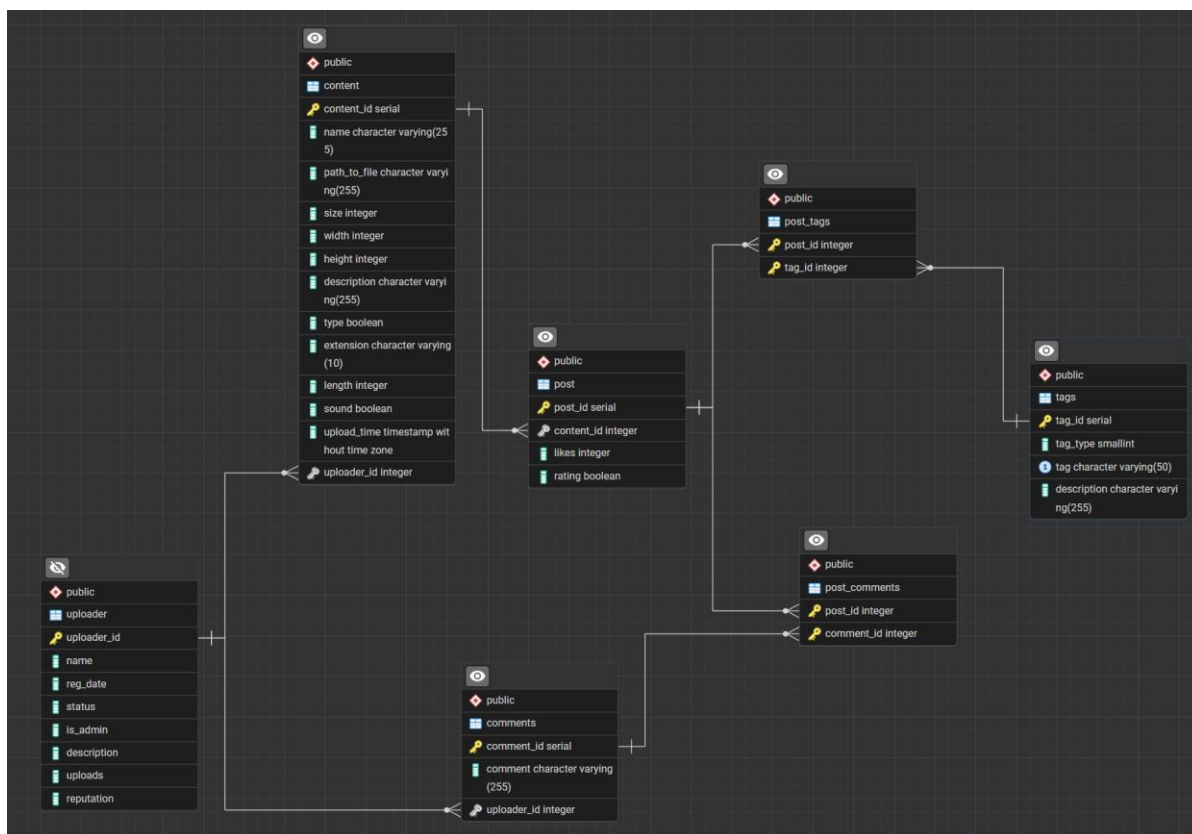


Рис. 2.4 Діаграма бази даних

Також у базі даних наявні функції `update_uploads_on_insert` та `update_uploads_on_delete` та їх відповідні тригери, які відповідають за поле `uploads` таблиці `uploader` (збільшують чи зменшують кількість завантажених користувачем постів)

2.3. Розробка математичної моделі та алгоритмів обробки даних в системі

Через великий обсяг коду, що взаємодіє з базою даних, його повна версія буде наведена у додатку А. Нижче ж розглянуто основні елементи:

Визначення типу файлу та його розмірів:

```

async function getImageDimensions(filePath) {
  try {
    const metadata = await sharp(filePath).metadata();
    return {
      width: metadata.width || 0,
      height: metadata.height || 0
    };
  } catch (err) {
  }
}
    
```

```

        console.error('Error reading image dimensions:', err);
        return { width: 0, height: 0 };
    }
}

function getVideoDetails(filePath) {
    return new Promise((resolve, reject) => {
        ffmpeg.ffprobe(filePath, (err, metadata) => {
            if (err) return reject(err);

            const videoStream = metadata.streams.find(s => s.codec_type === 'video');
            if (!videoStream) {
                return resolve({ width: 0, height: 0, length: 0, sound: false });
            }
            const audioStream = metadata.streams.find(s => s.codec_type === 'audio');
            resolve({
                width: parseInt(videoStream.width, 10) || 0,
                height: parseInt(videoStream.height, 10) || 0,
                length: parseFloat(metadata.format.duration) || 0,
                sound: !!audioStream
            });
        });
    });
}

```

Даний код перевіряє тип файлу, чи є він фото або відео. Також, даний код визначає висоту та ширину файлу, а у випадку з відео – перевіряє наявність звуку та довжину.

Визначення шляху файлу:

```

const storage = multer.diskStorage({
    destination: (req, file, cb) => {
        cb(null, path.join(__dirname, 'public', 'posts'));
    },
    filename: (req, file, cb) => {
        cb(null, file.originalname);
    }
});

const upload = multer({ storage: storage });

```

Даний код за допомогою бібліотеки `multer` для прийому файлів та їх коректного завантаження на сайт з внесенням відповідних даних до бази даних.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Отримання постів:

```
app.get('/api/posts', async (req, res) => {
  try {
    const query = `
      SELECT
        p.post_id,
        c.name,
        c.path_to_file,
        c.size,
        c.width,
        c.height,
        c.upload_time,
        u.name AS uploader_name,
        c.description,
        p.likes,
        CASE
          WHEN p.rating = false THEN 'General'
          WHEN p.rating = true THEN 'Sensitive'
        END AS rating,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 0 THEN t.tag END)
        AS artist_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 1 THEN t.tag END)
        AS copyrights_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 2 THEN t.tag END)
        AS character_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 3 THEN t.tag END)
        AS general_tags,
        COALESCE(
          JSON_AGG(
            DISTINCT jsonb_build_object('author', u2.name, 'text', co.comment)
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				13
Змн.	Арк.	№ докум.	Підпис	Дата		


```

        ) FILTER (WHERE co.comment IS NOT NULL), '['
    ) AS comments,
    ARRAY_AGG(DISTINCT jsonb_build_object('tag', t.tag, 'description',
t.description)) AS tag_descriptions,
    u.reg_date,
    CASE
        WHEN u.status = true THEN 'Online'
        WHEN u.status = false THEN 'Offline'
    END AS status,
    CASE
        WHEN u.is_admin = true THEN 'Admin'
        WHEN u.is_admin = false THEN 'User'
    END AS role,
    u.description AS user_description,
    u.uploads,
    u.reputation
FROM post p
JOIN content c ON p.content_id = c.content_id
LEFT JOIN post_tags pt ON p.post_id = pt.post_id
LEFT JOIN tags t ON pt.tag_id = t.tag_id
LEFT JOIN post_comments pc ON p.post_id = pc.post_id
LEFT JOIN comments co ON pc.comment_id = co.comment_id
LEFT JOIN uploader u ON c.uploader_id = u.uploader_id
LEFT JOIN uploader u2 ON co.uploader_id = u2.uploader_id
GROUP BY p.post_id, c.name, c.path_to_file, c.size, c.width, c.height,
c.upload_time,
        u.name, c.description, p.likes, p.rating, u.reg_date, u.status, u.is_admin,
        u.description, u.uploads, u.reputation
ORDER BY p.post_id DESC;
\;

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    const result = await pool.query(query);
    res.json(result.rows);
  } catch (err) {
    console.error(err);
    res.status(500).send('Error fetching posts');
  }
});

```

Даний код посилає SQL-запит до бази даних для отримання з неї даних з усіх таблиць для подальшого створення посту. Код для отримання даних про користувачів та тегів працює аналогічно з різницею у самому SQL-запиті.

Завантаження посту на сайт:

```

app.post('/api/upload', upload.single('file'), async (req, res) => {
  console.log('Request received at /api/upload');

```

```

  console.log('Request Body:', req.body);

```

```

  console.log('Uploaded File:', req.file);

```

```

  const { uploader, likes, sensitive, tags } = req.body;

```

```

  const file = req.file;

```

```

  if (!file) {

```

```

    console.log('No file uploaded');

```

```

    return res.status(400).send('No file uploaded');

```

```

  }

```

```

  const { size, path: filePath, originalname } = file;

```

```

  const extension = filePath.split('.').pop().toLowerCase();

```

```

  console.log('File Extension:', extension);

```

```

  const allowedExtensions = ['jpg', 'jpeg', 'png', 'gif', 'webm', 'mp4', 'avi', 'mkv'];

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (!allowedExtensions.includes(extension)) {
    console.log('Invalid file extension');
    return res.status(400).send('Invalid file extension');
}

let width, height, length, sound;
if (['mp4', 'avi', 'mkv', 'webm'].includes(extension)) {
    ({ width, height, length, sound } = await getVideoDetails(filePath));
} else {
    ({ width, height } = await getImageDimensions(filePath));
    length = null;
    sound = false;
}

try {
    console.log('Inserting content into database');
    const contentResult = await pool.query(
        `INSERT INTO content
        (name, path_to_file, size, width, height, type, extension, length, sound,
upload_time, uploader_id)
        VALUES ($1, $2, $3, $4, $5, $6, $7, $8, $9, NOW(), $10) RETURNING
content_id`,
        [
            originalname,
            filePath,
            size,
            width,
            height,
            // if it's a supported video format or not
            ['mp4', 'avi', 'mkv', 'webm'].includes(extension),

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        extension,
        length,
        sound,
        uploader
    ]
);

const contentId = contentResult.rows[0].content_id;
console.log('Content inserted with ID:', contentId);

console.log('Inserting post into database');
const postResult = await pool.query(
    `INSERT INTO post (content_id, likes, rating, sensitive)
    VALUES ($1, $2, $3, $4)
    RETURNING post_id`,
    [contentId, likes, false, sensitive]
);
const postId = postResult.rows[0].post_id;
console.log('Post inserted with ID:', postId);

const tagList = tags
    ? tags.split(',').map(t => t.trim()).filter(t => t !== '')
    : [];

for (const tag of tagList) {
    const tagResult = await pool.query('SELECT tag_id FROM tags WHERE tag = $1', [tag]);
    if (tagResult.rows.length > 0) {
        const tagId = tagResult.rows[0].tag_id;
        await pool.query('INSERT INTO post_tags (post_id, tag_id) VALUES ($1,

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$2)', [postId, tagId]);
    }
}

res.status(200).send('Post uploaded successfully');
} catch (err) {
    console.error('Error during upload process:', err);
    res.status(500).send('Error uploading post');
}
});

```

Даний код відповідає за завантаження посту на сайт (кнопка «Upload Post»). Він проводить різноманітні перевірки на відповідність даних, а потім виконує SQL-запит до бази даних на вставлення даних. Також даний код веде журнал, щоб у випадку, коли щось піде не так, можливо було знайти причину. Код для завантаження коментарів, тегів та користувачів працює аналогічно з відмінностями у SQL-запиті.

Висновки до другого розділу

У даному розділі було проаналізовано інформаційні процеси у базі даних, її структуру та алгоритми роботи з нею. Було описано кожен елемент проекту, як-от методи роботи з даними у базі даних та алгоритми завантаження даних. Особлива увага була приділена підрозділу 2.3, де описуються алгоритми взаємодії сайту з базою даних. У результаті було створено сайт-медіахостинг, який задовольняє усі поставлені вимоги та коректно працює з базою даних.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ОБРОБКИ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ РОБОТИ

3.1 Проектування інтерфейсу обробки даних

Інтерфейс (див. рис. 2.1) надає користувачу можливість додавати користувачів, теги, коментарі та пости, а також переглядати інформацію про користувачів та теги шляхом кліку по ним. Тепер пройдемося по кожному з пунктів більш детально:

1. Після натискання кнопки «Add Tag» (див. рис. 2.2) у правому верхньому кутку відкривається інтерфейс додавання тегів до бази даних, що містить: поле «Tag Name» для назви тегу, поле «Description» для його опису, список «Tag Type», де користувач може обрати тип тегу зі списку: Artist, Copyright, Character, General, кнопка «Add Tag», що вносить задані користувачем дані до бази даних та кнопка «Close», що закриває дане вікно.
2. Після натискання кнопки «Add Comment» у правому верхньому кутку відкривається інтерфейс додавання коментарів до бази даних, що містить: поле «Comment» для написання коментаря, список «Author», який містить у собі імена вже наявних користувачів у базі даних, список «Post», який містить у собі назви вже наявних постів у базі даних, до якого потрібно прикріпити коментар, кнопки внесення інформації до бази даних та закриття вікна.



Рис. 3.1 Приклад внесення інформації про коментар до БД через інтерфейс

3. Після натискання кнопки «Add Uploader» користувач бачить інтерфейс з наступними полями: текстові поля «Name», «Description» та «Reputation», які відповідають за ім'я користувача, опис профілю та числове значення репутації відповідно, чекбокси «Online» та «Admin», що відповідають за мережевий статус користувача та наявність прав адміністратора і кнопки внесення змін до бази даних та закриття вікна.

Рис. 3.2 Внесення інформації про користувача до БД через інтерфейс

4. Після натискання кнопки «Upload Post» користувач бачить інтерфейс завантаження посту: кнопка вибору файлу для завантаження, що відкриває провідник для вибору файлу, список «Choose uploader» зі списком існуючих у БД користувачів, поле «Likes» для кількості лайків на посту, чекбокс «Sensitive», що встановлює рейтинг посту (General, якщо не обрано, Sensitive, якщо обрано), список «Tags» з існуючими у БД тегами, які при

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

вибору користувачем будуть з'являтися у вигляді списку та кнопки завантаження посту та закриття вікна.

Рис. 3.3 Внесення інформації про пост до БД через інтерфейс

5. Перегляд інформації про тег здійснюється шляхом кліку по ньому у пості, що відображається на сайті (див. рис. 2.3).
6. Перегляд інформації про користувача здійснюється шляхом кліку по його імені у існуючому пості у полі «Uploader» або у секції коментарів.

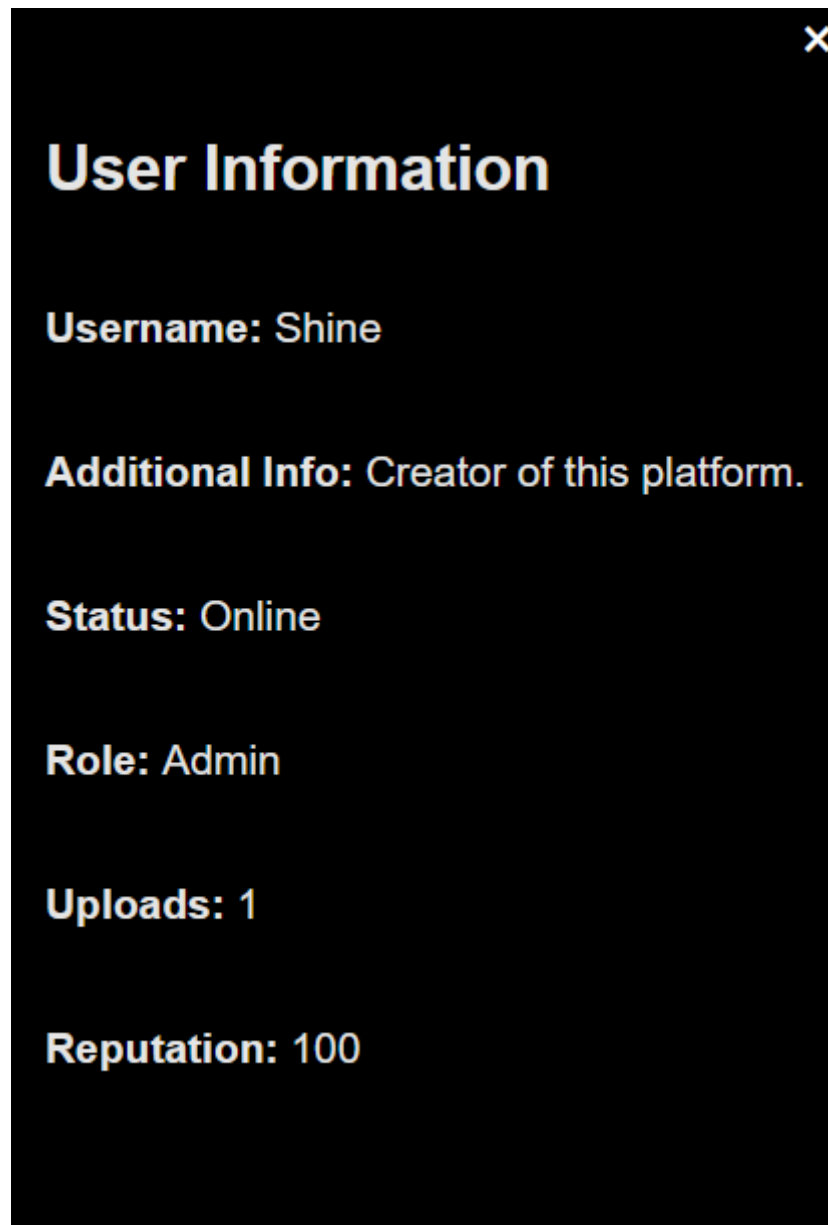


Рис. 3.4 Виведення інформації про користувача через інтерфейс

3.2 Реалізація операцій обробки даних в БД за напрямом курсової роботи

В даному проекті було використано СУБД PostgreSQL для створення та управління базою даних, середовище розробки WebStorm для розробки веб-сайту, що буде використовувати наявну БД з використанням наступних мов: HTML – для написання веб-сторінки, CSS – для зовнішнього вигляду веб-сайту, JavaScript – для frontend розробки та взаємодії з backend, Node.JS – для backend частини сайту та взаємодії з базою даних, а також бібліотеки pg для роботи з PostgreSQL, path для визначення шляху, express для розробки серверної частини, sharp для аналізу зображень, fluent-ffmpeg для аналізу відео, multer для роботи з файлами. Для

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

зчитування даних використовується SQL-запит SELECT FROM з умовами, для внесення даних до БД використовується SQL-запит INSERT INTO з послідуною перевіркою коректності даних, наприклад:

```
app.post('/api/tags', async (req, res) => {
  const { tag, description, tag_type } = req.body;
  try {
    const checkQuery = 'SELECT tag_id FROM tags WHERE tag = $1';
    const checkResult = await pool.query(checkQuery, [tag]);

    if (checkResult.rows.length > 0) {
      return res.status(400).send('Tag already exists');
    }

    const insertQuery = 'INSERT INTO tags (tag, description, tag_type) VALUES ($1, $2, $3)';
    await pool.query(insertQuery, [tag, description, tag_type]);
    res.status(201).send('Tag added successfully');
  } catch (err) {
    console.error('Error adding tag:', err);
    res.status(500).send('Error adding tag');
  }
});
```

В даному коді у змінну checkQuery заноситься запит 'SELECT tag_id FROM tags WHERE tag = \$1, потім у змінну checkResult заноситься результат перевірки тегу на унікальність, і якщо тег вже наявний у БД, то його не буде внесено і виведеться помилка. Якщо ж тег пройшов перевірку, то у змінну insertQuery вноситься запит INSERT INTO tags (tag, description, tag_type) VALUES (\$1, \$2, \$3), де замість \$число – відповідні значення полів з бази даних. Якщо тег було успішно внесено, виводиться повідомлення про успіх, якщо ж ні – повідомлення про помилку. Для

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

створення посту виконується об'єднання відповідних даних з усіх таблиць бази даних для коректного відображення постів на сайті за допомогою даного коду:

```
app.get('/api/posts', async (req, res) => {
  try {
    const query = `
      SELECT
        p.post_id,
        c.name,
        c.path_to_file,
        c.size,
        c.width,
        c.height,
        c.upload_time,
        u.name AS uploader_name,
        c.description,
        p.likes,
        CASE
          WHEN p.rating = false THEN 'General'
          WHEN p.rating = true THEN 'Sensitive'
        END AS rating,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 0 THEN t.tag END) AS
artist_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 1 THEN t.tag END) AS
copyrights_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 2 THEN t.tag END) AS
character_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 3 THEN t.tag END) AS
general_tags,
        COALESCE(
          JSON_AGG(
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				24
Змн.	Арк.	№ докум.	Підпис	Дата		

```

DISTINCT jsonb_build_object('author', u2.name, 'text', co.comment)
) FILTER (WHERE co.comment IS NOT NULL), '['
) AS comments,
ARRAY_AGG(DISTINCT jsonb_build_object('tag', t.tag, 'description',
t.description)) AS tag_descriptions,
u.reg_date,
CASE
WHEN u.status = true THEN 'Online'
WHEN u.status = false THEN 'Offline'
END AS status,
CASE
WHEN u.is_admin = true THEN 'Admin'
WHEN u.is_admin = false THEN 'User'
END AS role,
u.description AS user_description,
u.uploads,
u.reputation
FROM post p
JOIN content c ON p.content_id = c.content_id
LEFT JOIN post_tags pt ON p.post_id = pt.post_id
LEFT JOIN tags t ON pt.tag_id = t.tag_id
LEFT JOIN post_comments pc ON p.post_id = pc.post_id
LEFT JOIN comments co ON pc.comment_id = co.comment_id
LEFT JOIN uploader u ON c.uploader_id = u.uploader_id
LEFT JOIN uploader u2 ON co.uploader_id = u2.uploader_id
GROUP BY p.post_id, c.name, c.path_to_file, c.size, c.width, c.height,
c.upload_time,
u.name, c.description, p.likes, p.rating, u.reg_date, u.status, u.is_admin,
u.description, u.uploads, u.reputation
ORDER BY p.post_id DESC;

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

```

\;

const      result      =      await      pool.query(query);
res.json(result.rows);
}          catch          (err)          {
  console.error(err);
  res.status(500).send('Error fetching posts');
}
});

```

В даному кодi у змiнну `query` вноситься SQL-запит, який вибирає потрібні поля з відповідних таблиць БД з деякими замінами даних (наприклад, для поля `rating` використовується тип `Boolean` зі значенням `true` або `false`, які замінюються на `Sensitive` та `General` відповідно для розуміння користувача. Потім виконується об'єднання даних за допомогою `JOIN` та `LEFT JOIN` і їх групування через `GROUP BY` з послідуочим сортуванням у спадаючому порядку через `ORDER BY DESC`. Потім запит посилається до БД сервером, і завдяки даним, отриманим у відповіді, відбувається виведення посту. Якщо ж постів немає, або з якихось причин щось пішло не так, даний API видасть помилку «Error fetching posts» з кодом 500.

3.3 Організація звітності системи

Даний проект надає різноманітні звітності, як-от інформацію про файл, інформацію про теги та інформацію про користувачів для користувача сайту. Адміністратор завдяки запитам до СУБД має можливість бачити більш детальну інформацію, як-от кількість постів, завантажених певним користувачем або кількість тегів з певним типом тегу. Також програмний код серверу веде логи та у випадку проблем у консоль виводиться лог, по якому можна знайти причину виникнення проблеми.

Висновки до третього розділу

У даному розділі було розглянуто роботу з базою даних через сайт, описано кожен елемент інтерфейсу та за що відповідає кожен елемент. Також було описано реалізацію роботи з базою даних через сайт шляхом надсилення на сервер запиту зі

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

сторони користувача і його подальшою обробкою вже у базі даних. Було наведено види звітностей, які надає проект.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У першому розділі було розглянуто предмет дослідження та структуру бази даних для медіахостингу, також було обгрунтовано засоби, що будуть використуватись для роботи над проектом.

У другому розділі було проаналізовано інформаційні процеси у базі даних, її структуру та алгоритми роботи з нею. Було описано кожен елемент проекту, як-от методи роботи з даними у базі даних та алгоритми завантаження даних. Особлива увага була приділена підрозділу 2.3, де описуються алгоритми взаємодії сайту з базою даних. У результаті було створено сайт-медіахостинг, який задовольняє усі поставлені вимоги та коректно працює з базою даних.

У третьому розділі було розглянуто роботу з базою даних через сайт, описано кожен елемент інтерфейсу та за що відповідає кожен елемент. Також було описано реалізацію роботи з базою даних через сайт шляхом надсилання на сервер запиту зі сторони користувача і його подальшою обробкою вже у базі даних. Було наведено види звітностей, які надає проект.

У результаті роботи над курсовим проектом було створено базу даних для сайту-медіахостингу та сам сайт, які задовольняють усім вимогам. Сайт містить у собі зрозумілий для користувача інтерфейс, які надає легкі можливості з внесення даних до бази даних та простий але водночас ефективний код з великим потенціалом розвитку, що позитивно позначається на розумінні коду в цілому, що відкриває можливості до легкого оновлення та доробки додатку. Також в ході курсової роботи було значно підвищено навички роботи з мовою SQL, СУБД PostgreSQL та frontend та backend розробкою для створення сайтів, що реалізують всі можливості баз даних.

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				28
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. danbooru.donmai.us. (2005). Danbooru: Anime Image Board. [Електронний ресурс] – Режим доступу до ресурсу: <https://danbooru.donmai.us/>
2. Chatham, Mark (2012). Structured Query Language By Example - Volume I: Data Query Language., 2012. – 8 с.
3. postgresql.org (2011). "PostgreSQL server programming". PostgreSQL 9.1 official documentation. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.postgresql.org/docs/9.1/static/server-programming.html>
4. Chamberlin, Donald. "Early History of SQL". IEEE Annals of the History of Computing., 2012. – 78 с.
5. postgresql.org (2018). "4.1. Lexical Structure". PostgreSQL documentation. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.postgresql.org/docs/current/sql-syntax-lexical.html#SQL-SYNTAX-IDENTIFIERS>

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				29
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				30
Змн.	Арк.	№ докум.	Підпис	Дата		

Код програми

SQL-запит на створення бази даних

```

CREATE DATABASE booru;

\c booru;

CREATE TABLE uploader
(uploaders_id SERIAL PRIMARY KEY,
name VARCHAR(50) NOT NULL,
reg_date TIMESTAMP NOT NULL,
status BOOLEAN NOT NULL,
is_admin BOOLEAN NOT NULL,
description VARCHAR(255),
uploads INT NOT NULL,
reputation INT NOT NULL);

CREATE TABLE content
(content_id SERIAL PRIMARY KEY,
name VARCHAR(255) NOT NULL,
path_to_file VARCHAR(255) NOT NULL,
size INT NOT NULL,
width INT NOT NULL,
height INT NOT NULL,
description VARCHAR(255),
type BOOLEAN NOT NULL,
extension VARCHAR(4) NOT NULL,
length INT,
sound BOOLEAN,
upload_time TIMESTAMP NOT NULL,
uploaders_id INT NOT NULL,
FOREIGN KEY (uploaders_id) REFERENCES uploader(uploaders_id));

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				31
Змн.	Арк.	№ докум.	Підпис	Дата		

```

CREATE TABLE tags (tag_id SERIAL PRIMARY KEY,
tag_type SMALLINT NOT NULL CHECK (tag_type BETWEEN 0 AND 3),
tag VARCHAR(50) UNIQUE NOT NULL,
description VARCHAR(255));
CREATE TABLE comments (comment_id SERIAL PRIMARY KEY,
comment VARCHAR(255) NOT NULL,
uploader_id INT NOT NULL,
FOREIGN KEY (uploader_id) REFERENCES uploader(uploader_id));
CREATE TABLE post (post_id SERIAL PRIMARY KEY,
content_id INT NOT NULL,
likes INT NOT NULL,
rating BOOLEAN NOT NULL,
FOREIGN KEY (content_id) REFERENCES content(content_id));
CREATE TABLE post_tags (post_id INT,
tag_id INT,
PRIMARY KEY (post_id, tag_id),
FOREIGN KEY (post_id) REFERENCES post(post_id),
FOREIGN KEY (tag_id) REFERENCES tags(tag_id));
CREATE TABLE post_comments (post_id INT,
comment_id INT,
PRIMARY KEY (post_id, comment_id),
FOREIGN KEY (post_id) REFERENCES post(post_id),
FOREIGN KEY (comment_id) REFERENCES comments(comment_id));

```

Код функції update_uploads_on_insert

```

CREATE FUNCTION update_uploads_on_insert()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE uploader
    SET uploads = uploads + 1
    WHERE uploader_id = NEW.uploader_id;

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				32
Змн.	Арк.	№ докум.	Підпис	Дата		

```

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_update_uploads_insert
AFTER INSERT ON content
FOR EACH ROW
EXECUTE FUNCTION update_uploads_on_insert();
                                Код функції update_uploads_on_delete
CREATE OR REPLACE FUNCTION update_uploads_on_delete()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE uploader
    SET uploads = uploads - 1
    WHERE uploader_id = OLD.uploader_id;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_update_uploads_delete
AFTER DELETE ON content
FOR EACH ROW
EXECUTE FUNCTION update_uploads_on_delete();
                                Код серверної частини (Node.JS)
const express = require('express');
const cors = require('cors');
const { Pool } = require('pg');
const path = require('path');
const sharp = require('sharp');
const ffmpeg = require('fluent-ffmpeg');
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				33
Змн.	Арк.	№ докум.	Підпис	Дата		

```
const multer = require('multer');
```

```
const pool = new Pool({  
  user: 'postgres',  
  host: 'localhost',  
  database: 'booru',  
  password: '2828137137',  
  port: 5432  
});
```

```
async function getImageDimensions(filePath) {  
  try {  
    const metadata = await sharp(filePath).metadata();  
    return {  
      width: metadata.width || 0,  
      height: metadata.height || 0  
    };  
  } catch (err) {  
    console.error('Error reading image dimensions:', err);  
    return { width: 0, height: 0 };  
  }  
}
```

```
function getVideoDetails(filePath) {  
  return new Promise((resolve, reject) => {  
    ffmpeg.ffprobe(filePath, (err, metadata) => {  
      if (err) return reject(err);  
  
      const videoStream = metadata.streams.find(s => s.codec_type === 'video');  
      if (!videoStream) {
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				34
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        return resolve({ width: 0, height: 0, length: 0, sound: false });
    }
    const audioStream = metadata.streams.find(s => s.codec_type === 'audio');
    resolve({
        width: parseInt(videoStream.width, 10) || 0,
        height: parseInt(videoStream.height, 10) || 0,
        length: parseFloat(metadata.format.duration) || 0,
        sound: !!audioStream
    });
});
});
});
}

const app = express();
const PORT = 3000;

app.use(cors());
app.use(express.json());
app.use(express.static(path.join(__dirname, 'public')));

const storage = multer.diskStorage({
    destination: (req, file, cb) => {
        cb(null, path.join(__dirname, 'public', 'posts'));
    },
    filename: (req, file, cb) => {
        cb(null, file.originalname);
    }
});

const upload = multer({ storage: storage });

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				35
Змн.	Арк.	№ докум.	Підпис	Дата		

```

app.get('/', (req, res) => {
  res.send('Server is up and running!');
});

app.get('/api/posts', async (req, res) => {
  try {
    const query = `
      SELECT
        p.post_id,
        c.name,
        c.path_to_file,
        c.size,
        c.width,
        c.height,
        c.upload_time,
        u.name AS uploader_name,
        c.description,
        p.likes,
        CASE
          WHEN p.rating = false THEN 'General'
          WHEN p.rating = true THEN 'Sensitive'
        END AS rating,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 0 THEN t.tag END)
      AS artist_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 1 THEN t.tag END)
      AS copyrights_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 2 THEN t.tag END)
      AS character_tags,
        ARRAY_AGG(DISTINCT CASE WHEN t.tag_type = 3 THEN t.tag END)
      AS general_tags,

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				36
Змн.	Арк.	№ докум.	Підпис	Дата		

```

COALESCE(
    JSON_AGG(
        DISTINCT jsonb_build_object('author', u2.name, 'text', co.comment)
    ) FILTER (WHERE co.comment IS NOT NULL), '[]'
) AS comments,
ARRAY_AGG(DISTINCT jsonb_build_object('tag', t.tag, 'description',
t.description)) AS tag_descriptions,
u.reg_date,
CASE
    WHEN u.status = true THEN 'Online'
    WHEN u.status = false THEN 'Offline'
END AS status,
CASE
    WHEN u.is_admin = true THEN 'Admin'
    WHEN u.is_admin = false THEN 'User'
END AS role,
u.description AS user_description,
u.uploads,
u.reputation
FROM post p
JOIN content c ON p.content_id = c.content_id
LEFT JOIN post_tags pt ON p.post_id = pt.post_id
LEFT JOIN tags t ON pt.tag_id = t.tag_id
LEFT JOIN post_comments pc ON p.post_id = pc.post_id
LEFT JOIN comments co ON pc.comment_id = co.comment_id
LEFT JOIN uploader u ON c.uploader_id = u.uploader_id
LEFT JOIN uploader u2 ON co.uploader_id = u2.uploader_id
GROUP BY p.post_id, c.name, c.path_to_file, c.size, c.width, c.height,
c.upload_time,
u.name, c.description, p.likes, p.rating, u.reg_date, u.status, u.is_admin,

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				37
Змн.	Арк.	№ докум.	Підпис	Дата		


```

        u.description, u.uploads, u.reputation
    ORDER BY p.post_id DESC;
`;
    const result = await pool.query(query);
    res.json(result.rows);
  } catch (err) {
    console.error(err);
    res.status(500).send('Error fetching posts');
  }
});

app.get('/api/uploaders', async (req, res) => {
  try {
    const result = await pool.query('SELECT uploader_id, name FROM uploader');
    res.json(result.rows);
  } catch (err) {
    console.error(err);
    res.status(500).send('Error fetching uploaders');
  }
});

app.get('/api/tags', async (req, res) => {
  try {
    const query = 'SELECT tag_id, tag FROM tags ORDER BY tag';
    const result = await pool.query(query);
    res.json(result.rows);
  } catch (err) {
    console.error('Error fetching tags:', err);
    res.status(500).send('Error fetching tags');
  }
}

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				38
Змн.	Арк.	№ докум.	Підпис	Дата		

```

});

app.post('/api/upload', upload.single('file'), async (req, res) => {
  console.log('Request received at /api/upload');

  console.log('Request Body:', req.body);
  console.log('Uploaded File:', req.file);

  const { uploader, likes, sensitive, tags } = req.body;
  const file = req.file;

  if (!file) {
    console.log('No file uploaded');
    return res.status(400).send('No file uploaded');
  }

  const { size, path: filePath, originalname } = file;
  const extension = filePath.split('.').pop().toLowerCase();
  console.log('File Extension:', extension);

  const allowedExtensions = ['jpg', 'jpeg', 'png', 'gif', 'webm', 'mp4', 'avi', 'mkv'];
  if (!allowedExtensions.includes(extension)) {
    console.log('Invalid file extension');
    return res.status(400).send('Invalid file extension');
  }

  let width, height, length, sound;
  if (['mp4', 'avi', 'mkv', 'webm'].includes(extension)) {
    ({ width, height, length, sound } = await getVideoDetails(filePath));
  } else {

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				39
Змн.	Арк.	№ докум.	Підпис	Дата		

```

({ width, height } = await getImageDimensions(filePath));
length = null;
sound = false;
}

try {
  console.log('Inserting content into database');
  const contentResult = await pool.query(
    `INSERT INTO content
      (name, path_to_file, size, width, height, type, extension, length, sound,
upload_time, uploader_id)
      VALUES ($1, $2, $3, $4, $5, $6, $7, $8, $9, NOW(), $10) RETURNING
content_id`,
    [
      originalname,
      filePath,
      size,
      width,
      height,
      ['mp4', 'avi', 'mkv', 'webm'].includes(extension),
      extension,
      length,
      sound,
      uploader
    ]
  );

  const contentId = contentResult.rows[0].content_id;
  console.log('Content inserted with ID:', contentId);

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

```

console.log('Inserting post into database');
const postResult = await pool.query(
  `INSERT INTO post (content_id, likes, rating, sensitive)
  VALUES ($1, $2, $3, $4)
  RETURNING post_id`,
  [contentId, likes, false, sensitive]
);
const postId = postResult.rows[0].post_id;
console.log('Post inserted with ID:', postId);

const tagList = tags
  ? tags.split(',').map(t => t.trim()).filter(t => t !== "")
  : [];

for (const tag of tagList) {
  const tagResult = await pool.query('SELECT tag_id FROM tags WHERE tag =
$1', [tag]);
  if (tagResult.rows.length > 0) {
    const tagId = tagResult.rows[0].tag_id;
    await pool.query('INSERT INTO post_tags (post_id, tag_id) VALUES ($1,
$2)', [postId, tagId]);
  }
}

res.status(200).send('Post uploaded successfully');
} catch (err) {
  console.error('Error during upload process:', err);
  res.status(500).send('Error uploading post');
}
});

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				41
Змн.	Арк.	№ докум.	Підпис	Дата		

```

app.post('/api/comments', async (req, res) => {
  const { comment, uploader_id, post_id } = req.body;
  try {
    const insertCommentQuery = 'INSERT INTO comments (comment, uploader_id)
VALUES ($1, $2) RETURNING comment_id';
    const commentResult = await pool.query(insertCommentQuery, [comment,
uploader_id]);
    const commentId = commentResult.rows[0].comment_id;

    // Insert the new comment into the post_comments table
    const insertPostCommentQuery = 'INSERT INTO post_comments (post_id,
comment_id) VALUES ($1, $2)';
    await pool.query(insertPostCommentQuery, [post_id, commentId]);

    res.status(201).send('Comment added successfully');
  } catch (err) {
    console.error('Error adding comment:', err);
    res.status(500).send('Error adding comment');
  }
});

```

```

app.post('/api/tags', async (req, res) => {
  const { tag, description, tag_type } = req.body;
  try {
    const checkQuery = 'SELECT tag_id FROM tags WHERE tag = $1';
    const checkResult = await pool.query(checkQuery, [tag]);

    if (checkResult.rows.length > 0) {
      return res.status(400).send('Tag already exists');
    }
  }
});

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    const insertQuery = 'INSERT INTO tags (tag, description, tag_type) VALUES ($1,
    $2, $3)';

    await pool.query(insertQuery, [tag, description, tag_type]);
    res.status(201).send('Tag added successfully');
  } catch (err) {
    console.error('Error adding tag:', err);
    res.status(500).send('Error adding tag');
  }
});

app.post('/api/uploaders', async (req, res) => {
  const { name, description, reputation, status, is_admin, reg_date } = req.body;
  try {
    const query = 'INSERT INTO uploader (name, description, reputation, status,
    is_admin, reg_date, uploads) VALUES ($1, $2, $3, $4, $5, $6, 0)';
    await pool.query(query, [name, description, reputation, status, is_admin,
    reg_date]);
    res.status(201).send('Uploader added successfully');
  } catch (err) {
    console.error('Error adding uploader:', err);
    res.status(500).send('Error adding uploader');
  }
});

app.listen(PORT, () => {
  console.log(`Server is running at http://localhost:${PORT}`);
});

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

Код frontend частини (JavaScript)

```
document.addEventListener('DOMContentLoaded', async () => {  
    const postContainer = document.getElementById('post-container');  
    const infoModal = document.getElementById('tag-info-modal'); // Модальне вікно  
    для опису тегу або користувача  
    const infoText = document.getElementById('tag-info-text'); // Текстовий елемент  
    для опису  
    const closeInfo = document.getElementById('close-tag-info'); // Кнопка закриття  
    модального вікна  
  
    // Перевірка існування кнопки закриття для модального вікна  
    if (closeInfo) {  
        closeInfo.addEventListener('click', () => {  
            infoModal.style.display = 'none';  
        });  
    }  
  
    try {  
        // Виконуємо запит до API  
        const response = await fetch('/api/posts');  
        if (!response.ok) {  
            throw new Error(`HTTP error! Status: ${response.status}`);  
        }  
  
        const posts = await response.json();  
  
        if (posts.length === 0) {  
            postContainer.innerHTML = '<p>No posts found.</p>';  
            return;  
        }  
    }  
}
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				44
Змн.	Арк.	№ докум.	Підпис	Дата		

```

}

// Підраховуємо кількість постів для кожного тегу
const tagCount = {
  artist: {},
  copyrights: {},
  character: {},
  general: {}
};

posts.forEach(post => {
  post.artist_tags?.forEach(tag => tagCount.artist[tag] = (tagCount.artist[tag] || 0)
+ 1);
  post.copyrights_tags?.forEach(tag => tagCount.copyrights[tag] =
(tagCount.copyrights[tag] || 0) + 1);
  post.character_tags?.forEach(tag => tagCount.character[tag] =
(tagCount.character[tag] || 0) + 1);
  post.general_tags?.forEach(tag => tagCount.general[tag] =
(tagCount.general[tag] || 0) + 1);
});

// Додаємо пости
posts.forEach(post => {
  const postElement = document.createElement('div');
  postElement.classList.add('post');

  // Шлях до зображення
  const imgPath = post.path_to_file.replace(/^.*[\\\/]/, 'posts/'); // Замінюємо
ШЛЯХ

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				45
Змн.	Арк.	№ докум.	Підпис	Дата		


```
// Функція для обробки тегів без останньої коми та null значень
const formatTags = (tags, tagClass) => {
  tags = tags.filter(tag => tag !== null);
  if (tags.length === 0) return "";

  return tags
    .slice(0, -1)
    .map(tag => `<span class="${tagClass}" data-tag="${tag}">${tag}
    (${tagCount[tagClass][tag] || 0})</span>`)
    .join(', ') + (tags.length > 1 ? ', ' : '') + `<span class="${tagClass}" data-
    tag="${tags[tags.length - 1]}">${tags[tags.length - 1]}
    (${tagCount[tagClass][tags[tags.length - 1]] || 0})</span>`;
};

postElement.innerHTML = `
  <div class="image-container">
    
  </div>
  <div class="post-details">
    <h3>${post.name}</h3>
    <p>${post.description}</p>
    <p><strong>Likes:</strong> ${post.likes}</p>
    <p><strong>Uploader:</strong> <span class="user-name" data-
    username="${post.uploader_name}">${post.uploader_name}</span></p>
    <p><strong>Size:</strong> ${post.size} kB</p>
    <p><strong>Dimensions:</strong> ${post.width} x ${post.height}</p>
    <p><strong>Upload Time:</strong> ${new
    Date(post.upload_time).toLocaleString()}</p>
    <p><strong>Rating:</strong> ${post.rating}</p>
  </div>
`;
```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

```

<!-- Теги: 4 колонки -->
<div class="tags">
    <div class="tags-column">
        <h4>Artist</h4>
        <p>${formatTags(post.artist_tags, 'artist')}</p>
    </div>
    <div class="tags-column">
        <h4>Copyrights</h4>
        <p>${formatTags(post.copyrights_tags, 'copyrights')}</p>
    </div>
    <div class="tags-column">
        <h4>Character</h4>
        <p>${formatTags(post.character_tags, 'character')}</p>
    </div>
    <div class="tags-column">
        <h4>General</h4>
        <p>${formatTags(post.general_tags, 'general')}</p>
    </div>
</div>

<div class="comments">
    <strong>Comments:</strong>
    <ul>
        ${post.comments.map(comment => `<li class="user-name" data-
username="${comment.author}">${comment.author}: ${comment.text}</li>`).join("")}
    </ul>
</div>
</div>
`;

postContainer.appendChild(postElement);

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// Обробка кліку на ім'я користувача в пості
const userNameElement = postElement.querySelector('.user-name');
if (userNameElement) {
    userNameElement.addEventListener('click', () => {
        const username = userNameElement.getAttribute('data-username');
        // Відображаємо модальне вікно з інформацією про користувача
        const user = posts.find(p => p.uploader_name === username); // Знахо-
        димо користувача

        if (user) {
            infoText.innerHTML = `
                <h2>User Information</h2>
                <p><strong>Username:</strong> ${username}</p>
                <p><strong>Additional Info:</strong> ${user.user_description}</p>
                <p><strong>Status:</strong> ${user.status}</p>
                <p><strong>Role:</strong> ${user.role}</p>
                <p><strong>Uploads:</strong> ${user.uploads}</p>
                <p><strong>Reputation:</strong> ${user.reputation}</p>
            `;

            // Позиціонуємо модальне вікно справа від курсора
            const rect = userNameElement.getBoundingClientRect();
            infoModal.style.left = `${rect.left + window.scrollX +
            userNameElement.offsetWidth + 5}px`;

            infoModal.style.top = `${rect.top + window.scrollY}px`;

            infoModal.style.display = 'block';
        }
    });
}

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    // Обробка кліку на ім'я користувача в коментарях
    const commentUserElements = postElement.querySelectorAll('.comments .user-
name');
    commentUserElements.forEach(commentUserElement => {
        commentUserElement.addEventListener('click', () => {
            const username = commentUserElement.getAttribute('data-username');
            // Відображаємо модальне вікно з інформацією про користувача
            const user = posts.find(p => p.uploader_name === username); // Знахо-
димо користувача

            if (user) {
                infoText.innerHTML = `
                    <h2>User Information</h2>
                    <p><strong>Username:</strong> ${username}</p>
                    <p><strong>Additional Info:</strong> ${user.user_description}</p>
                    <p><strong>Status:</strong> ${user.status}</p>
                    <p><strong>Role:</strong> ${user.role}</p>
                    <p><strong>Uploads:</strong> ${user.uploads}</p>
                    <p><strong>Reputation:</strong> ${user.reputation}</p>
                `;

                // Позиціонуємо модальне вікно справа від курсора
                const rect = commentUserElement.getBoundingClientRect();
                infoModal.style.left = `${rect.left + window.scrollX +
commentUserElement.offsetWidth + 5}px`;
                infoModal.style.top = `${rect.top + window.scrollY}px`;

                infoModal.style.display = 'block';
            }
        });
    });

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				49
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
  });
});

// Обробка наведення на тег
const tags = postElement.querySelectorAll('.tags span');
tags.forEach(tagElement => {
  tagElement.addEventListener('click', () => {
    const tagName = tagElement.getAttribute('data-tag');
    const tagInfo = post.tag_descriptions.find(tag => tag.tag === tagName);
    if (tagInfo) {
      infoText.innerHTML =
`<strong>${tagName}</strong><br>${tagInfo.description}`;

      // Позиціонуємо модальне вікно справа від курсора
      const rect = tagElement.getBoundingClientRect();
      infoModal.style.left = `${rect.left + window.scrollX +
tagElement.offsetWidth + 5}px`;
      infoModal.style.top = `${rect.top + window.scrollY}px`;

      infoModal.style.display = 'block';
    }
  });
});
});
} catch (err) {
  console.error('Failed to fetch posts:', err);
  postContainer.innerHTML = '<p>Error loading posts. Please try again later.</p>';
}
});

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

```

document.addEventListener('DOMContentLoaded', async () => {
  const openPopupButton = document.getElementById('open-popup');
  const closePopupButton = document.getElementById('close-popup');
  const popup = document.getElementById('upload-post-popup');
  const form = document.getElementById('upload-post-form');
  const fileInput = document.getElementById('file');
  const uploaderSelect = document.getElementById('uploader');
  const tagSelect = document.getElementById('tag-select');
  const selectedTagsContainer = document.getElementById('selected-tags');
  const sensitiveCheckbox = document.getElementById('sensitive');

  // Show popup
  openPopupButton.addEventListener('click', () => {
    popup.style.display = 'block';
  });

  // Hide popup
  closePopupButton.addEventListener('click', () => {
    popup.style.display = 'none';
  });

  // Fetch and populate uploader options
  try {
    const response = await fetch('/api/uploaders');
    if (!response.ok) {
      throw new Error(`Error fetching uploaders: ${response.status}`);
    }
    const uploaders = await response.json();
    uploaders.forEach(uploader => {

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    const option = document.createElement('option');
    option.value = uploader.uploader_id;
    option.textContent = uploader.name;
    uploaderSelect.appendChild(option);
  });
} catch (err) {
  console.error('Failed to fetch uploaders:', err);
}

// Fetch and populate tag options
try {
  const tagResponse = await fetch('/api/tags');
  if (!tagResponse.ok) {
    throw new Error(`Error fetching tags: ${tagResponse.status}`);
  }
  const tags = await tagResponse.json();
  tags.forEach(tag => {
    const option = document.createElement('option');
    option.value = tag.tag_id;
    option.textContent = tag.tag;
    tagSelect.appendChild(option);
  });
} catch (err) {
  console.error('Failed to fetch tags:', err);
}

// Handle tag selection
tagSelect.addEventListener('change', () => {
  const selectedTag = tagSelect.options[tagSelect.selectedIndex];
  if (selectedTag.value) {

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				52
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    addTag(selectedTag.textContent, selectedTag.value);
    tagSelect.value = "";
  }
});

function addTag(tagName, tagId) {
  const tagElement = document.createElement('div');
  tagElement.className = 'selected-tag';
  tagElement.innerHTML = `${tagName}</span><button type="button"
data-tag-id="${tagId}">x</button>`;
  selectedTagsContainer.appendChild(tagElement);

  // Handle tag removal
  const removeBtn = tagElement.querySelector('button');
  removeBtn.addEventListener('click', () => {
    selectedTagsContainer.removeChild(tagElement);
  });
}

form.addEventListener('submit', async (event) => {
  event.preventDefault();

  // Debug log: ensures this event is firing
  console.log('Form submission initiated...');

  const formData = new FormData();
  const selectedTags = Array.from(
    selectedTagsContainer.querySelectorAll('.selected-tag button')
  ).map(button => {
    // We'll append the button's text, not just the ID

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				53
Змн.	Арк.	№ докум.	Підпис	Дата		


```

    // because our server expects the actual tag text for insertion.
    return button.parentNode.querySelector('span').textContent;
});

// Full console output for debugging
console.log('Selected tags:', selectedTags);

formData.append('file', fileInput.files[0]);
formData.append('uploader', uploaderSelect.value);
formData.append('likes', document.getElementById('likes').value);
formData.append('sensitive', sensitiveCheckbox.checked);
// Join tags with commas to match server code
formData.append('tags', selectedTags.join(','));

try {
    const response = await fetch('/api/upload', {
        method: 'POST',
        body: formData
    });
    console.log('Upload request sent, awaiting response...');

    if (response.ok) {
        alert('Post uploaded successfully!');
        console.log('Post upload success');
        popup.style.display = 'none';
        // Optionally refresh or fetch the new posts list here
    } else {
        alert('Failed to upload post.');
        console.log('Upload failed with status:', response.status);
    }
}

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				54
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    } catch (err) {
        console.error('Error during fetch /api/upload:', err);
    }
});

});

document.addEventListener('DOMContentLoaded', async () => {
    const openAddTagPopupButton = document.getElementById('open-add-tag-popup');
    const closeAddTagPopupButton = document.getElementById('close-add-tag-popup');
    const addTagPopup = document.getElementById('add-tag-popup');
    const addTagForm = document.getElementById('add-tag-form');

    // Show Add Tag popup
    openAddTagPopupButton.addEventListener('click', () => {
        addTagPopup.style.display = 'block';
    });

    // Hide Add Tag popup
    closeAddTagPopupButton.addEventListener('click', () => {
        addTagPopup.style.display = 'none';
    });

    addTagForm.addEventListener('submit', async (event) => {
        event.preventDefault();

        const tagName = document.getElementById('tag-name').value;
        const tagDescription = document.getElementById('tag-description').value;
        const tagType = document.getElementById('tag-type').value;

        const response = await fetch('/api/tags', {

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

```

method: 'POST',
headers: {
  'Content-Type': 'application/json'
},
body: JSON.stringify({
  tag: tagName,
  description: tagDescription,
  tag_type: tagType
}))
});

if (response.ok) {
  alert('Tag added successfully!');
  addTagPopup.style.display = 'none';
} else {
  alert('Failed to add tag.');
}
});
});

document.addEventListener('DOMContentLoaded', async () => {
  const openAddCommentPopupButton = document.getElementById('open-add-comment-popup');
  const closeAddCommentPopupButton = document.getElementById('close-add-comment-popup');
  const addCommentPopup = document.getElementById('add-comment-popup');
  const addCommentForm = document.getElementById('add-comment-form');
  const commentAuthorSelect = document.getElementById('comment-author');
  const commentPostSelect = document.getElementById('comment-post');

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// Show Add Comment popup
openAddCommentPopupButton.addEventListener('click', () => {
    addCommentPopup.style.display = 'block';
});

// Hide Add Comment popup
closeAddCommentPopupButton.addEventListener('click', () => {
    addCommentPopup.style.display = 'none';
});

// Fetch and populate author options
const authorResponse = await fetch('/api/uploaders');
const authors = await authorResponse.json();
authors.forEach(author => {
    const option = document.createElement('option');
    option.value = author.uploader_id;
    option.textContent = author.name;
    commentAuthorSelect.appendChild(option);
});

// Fetch and populate post options
const postResponse = await fetch('/api/posts');
const posts = await postResponse.json();
posts.forEach(post => {
    const option = document.createElement('option');
    option.value = post.post_id;
    option.textContent = post.name;
    commentPostSelect.appendChild(option);
});

```

```

addCommentForm.addEventListener('submit', async (event) => {
    event.preventDefault();

    const commentText = document.getElementById('comment-text').value;
    const commentAuthor = commentAuthorSelect.value;
    const commentPost = commentPostSelect.value;

    const response = await fetch('/api/comments', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            comment: commentText,
            uploader_id: commentAuthor,
            post_id: commentPost
        })
    });

    if (response.ok) {
        alert('Comment added successfully!');
        addCommentPopup.style.display = 'none';
    } else {
        alert('Failed to add comment.');
    }
});

document.addEventListener('DOMContentLoaded', async () => {
    const openAddUploaderPopupButton = document.getElementById('open-add-

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

```

uploader-popup');

const closeAddUploaderPopupButton = document.getElementById('close-add-
uploader-popup');

const addUploaderPopup = document.getElementById('add-uploader-popup');
const addUploaderForm = document.getElementById('add-uploader-form');

// Show Add Uploader popup
openAddUploaderPopupButton.addEventListener('click', () => {
  addUploaderPopup.style.display = 'block';
});

// Hide Add Uploader popup
closeAddUploaderPopupButton.addEventListener('click', () => {
  addUploaderPopup.style.display = 'none';
});

addUploaderForm.addEventListener('submit', async (event) => {
  event.preventDefault();

  const uploaderName = document.getElementById('uploader-name').value;
  const uploaderDescription = document.getElementById('uploader-
description').value;
  const uploaderReputation = document.getElementById('uploader-
reputation').value;
  const uploaderStatus = document.getElementById('uploader-status').checked;
  const uploaderAdmin = document.getElementById('uploader-admin').checked;
  const regDate = new Date().toISOString();

  const response = await fetch('/api/uploaders', {
    method: 'POST',

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

```

headers: {
    'Content-Type': 'application/json'
},
body: JSON.stringify({
    name: uploaderName,
    description: uploaderDescription,
    reputation: uploaderReputation,
    status: uploaderStatus,
    is_admin: uploaderAdmin,
    reg_date: regDate
})
});

if (response.ok) {
    alert('Uploader added successfully!');
    addUploaderPopup.style.display = 'none';
} else {
    alert('Failed to add uploader.');
}
});
});

```

		Лебідь Е.Ю.			ДУ «Житомирська політехніка».23.121.12.000 – ПЗ	Арк.
		Коротун О.В.				60
Змн.	Арк.	№ докум.	Підпис	Дата		