

ЛАБОРАТОРНА РОБОТА № 5

SSR-застосунок: опрацювання форм класичним методом (POST), завантаження файлів

Мета роботи: Навчитися будувати серверно-рендерений (SSR) застосунок, що приймає й опрацьовує дані форм класичним способом — через звичайне надсилання форми методом POST і повне перезавантаження сторінки, без JavaScript чи AJAX-запитів. Для кожної сутності програми передбачити HTML-форму додавання нового запису, причому щонайменше одна форма повинна містити поле завантаження файлу. Це остання лабораторна робота курсу, що використовує SSR-підхід — надалі акцент переміститься на клієнт-серверну взаємодію через API.

Теоретичні відомості

1. SSR: сервер формує готовий HTML

SSR (Server-Side Rendering) — підхід, за якого сервер сам формує повну HTML-сторінку у відповідь на кожен запит, і браузер отримує вже готову розмітку, без потреби довантажувати дані окремим запитом. Це протилежність підходу з лабораторної роботи № 4, де сторінка `demo.html` була статичною, а дані підвантажувалися окремим `fetch()`-запитом до JSON API вже *після* завантаження сторінки.

Ознака	SSR (ця робота)	Клієнтський рендеринг (лаб. № 4)
Що повертає сервер	Готовий HTML із вбудованими даними	Порожній HTML-каркас
Де формується розмітка	На сервері (PHP echo / вбудований PHP)	У браузері (JavaScript)
Оновлення після дії користувача	Повне перезавантаження сторінки	Частковий перемальовок без перезавантаження

SSR не є «застарілим» підходом — це свідомий, часто оптимальний вибір для форм адміністрування, внутрішніх систем обліку й будь-яких сторінок, де простота реалізації й надійність важливіші за миттєвість оновлення інтерфейсу. Розуміння SSR лишається фундаментом навіть для команд, що згодом переходять на клієнтський рендеринг.

2. HTML-форми: метод, дія, enctype

Тег `<form>` (тема 6) визначає, куди (атрибут `action`) і яким методом (атрибут `method`) браузер надішле введені користувачем дані. Для форми, що містить поле завантаження файлу, обов'язковий додатковий атрибут `enctype="multipart/form-data"` — без нього вміст файлу взагалі не потрапить на сервер, лише його ім'я.

```
<form method="post" action="backpack_form.php" enctype="multipart/form-data">
    <label>Назва: <input type="text" name="name" required></label>
    <label>Фото: <input type="file" name="photo" accept="image/*"></label>
    <button type="submit">Зберегти</button>
</form>
```

Атрибут `required` на боці клієнта — лише зручність для користувача (браузер не дасть надіслати порожнє поле) і жодним чином не є захистом: HTML можна редагувати чи обходити, тому кожне значення все одно обов'язково перевіряється на сервері (розділ 3) — довіряти клієнтській валідації не можна.

3. Опрацювання форми на сервері: `$_POST` і валідація

Дані, надіслані методом POST, доступні в PHP через суперглобальний масив `$_POST` (тема 6). Типовий скрипт-обробник форми перевіряє метод запиту, зчитує й перевіряє кожне поле, і лише за умови коректності всіх значень зберігає запис.

```
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name'] ?? '');

    if ($name === '') {
        $error = 'Назва обов'язкова.';
    } else {
        // ... зберегти запис ...
    }
}
```

4. Завантаження файлів: `$_FILES`

Файл, надісланий через поле `<input type="file">`, потрапляє в суперглобальний масив `$_FILES` (тема 6) — не як вміст напряму, а як тимчасовий файл на сервері, шлях до якого потрібно явно перенести в постійне місце функцією `move_uploaded_file()`.

Ключ <code>\$_FILES['photo'][...]</code>	Значення
<code>name</code>	Оригінальне ім'я файлу на комп'ютері користувача (ненадійне, не використовувати напряму)
<code>tmp_name</code>	Тимчасовий шлях, за яким PHP тимчасово зберіг завантажений файл
<code>error</code>	Код помилки завантаження (<code>UPLOAD_ERR_OK = 0</code> , якщо все гаразд)
<code>size</code>	Розмір файлу в байтах
<code>type</code>	MIME-тип, повідомлений браузером (ненадійний — перевіряти на сервері окремо)

```
<?php
if ($_FILES['photo']['error'] === UPLOAD_ERR_OK) {
    $allowedTypes = ['image/jpeg', 'image/png', 'image/webp'];
    $mimeType = mime_content_type($_FILES['photo']['tmp_name']);

    if (in_array($mimeType, $allowedTypes, true)) {
        $extension = pathinfo($_FILES['photo']['name'], PATHINFO_EXTENSION);
        $filename = uniqid('backpack_', true) . '.' . $extension;
        move_uploaded_file($_FILES['photo']['tmp_name'], __DIR__ .
        '/uploads/' . $filename);
    }
}
```

Перевірка MIME-типу функцією `mime_content_type()` (яка визначає тип за фактичним вмістом файлу) — обов'язкова, а не необов'язкова деталь: поле `$_FILES['photo']['type']` лише повторює те, що надіслав браузер, і зловмисник може підмінити його на будь-яке значення. Так само ім'я файлу ніколи не використовують у первісному вигляді (розділ «Завантаження файлів» теми 6) — новий, згенерований `uniqid()` ім'я файлу унеможливорює як конфлікти імен, так і спроби «протягнути» шкідливий шлях через оригінальну назву.

5. Патерн Post/Redirect/Get

Якщо після успішної обробки форми сервер одразу виводить сторінку результату методом `echo`, повторне натискання F5 (оновити сторінку) у браузері користувача повторно надішле ті самі POST-дані — і, наприклад, створить дублікат запису. Патерн **Post/Redirect/Get** (тема 9) вирішує це: після успішної обробки POST-запиту сервер не виводить HTML одразу, а надсилає HTTP-перенаправлення (заголовок `Location`) на іншу (чи ту саму) сторінку, яка вже завантажується звичайним GET-запитом.

```
<?php
// Після успішного збереження – перенаправлення, а не прямий вивід HTML
header('Location: index.php');
exit;
```

Завдання на лабораторну роботу

Продовжуйте працювати з предметною областю з лабораторних робіт № 3–4. Побудуйте SSR-застосунок за наведеними нижче кроками.

1. Для кожної сутності вашої предметної області створіть окрему HTML-форму додавання нового запису (метод POST, дія — сам файл форми чи окремий файл-обробник).
2. Забезпечте, щоб щонайменше одна форма містила поле завантаження файлу (`<input type="file">`) з відповідним атрибутом `enctype` у тегу `<form>`.
3. Реалізуйте обробку кожної форми на сервері класичним методом: перевірка `$_SERVER['REQUEST_METHOD']`, зчитування `$_POST` (і `$_FILES` для форми з

файлом), валідація обов'язкових полів, збереження в CSV (лабораторна робота № 4) через клас-репозиторій.

4. Для поля файлу передбачте перевірку MIME-типу завантаженого файлу та збереження його під новим, згенерованим ім'ям у окремій теці (наприклад, uploads/).
5. Застосуйте патерн Post/Redirect/Get: після успішного збереження перенаправляйте користувача на головну сторінку зі списком записів, а не виводьте результат напряму.
6. На головній сторінці виведіть актуальний список усіх записів кожної сутності (включно із завантаженим зображенням, якщо воно є) — сторінка повинна відображати щойно доданий запис одразу після перенаправлення.

Вимоги до звіту

Звіт повинен містити: тему й мету роботи; структуру файлів проєкту; лістинг коду всіх форм і обробників; знімки екрана кожної форми та головної сторінки до й після додавання запису (включно з відображенням завантаженим зображенням); опис реалізованої валідації; висновок.

Приклад виконання: «Спортивні рюкзаки»

Приклад продовжує предметну область із попередніх лабораторних робіт. Додано форми для всіх трьох сутностей: додавання категорії (без файлу), додавання рюкзака (з полем завантаження фото — крок 2 завдання) і додавання характеристики до наявного рюкзака.

Структура проєкту

```
lab5/
├── classes/
│   ├── Category.php
│   ├── Characteristic.php
│   ├── Backpack.php
│   └── BackpackRepository.php
├── data/
│   ├── categories.csv
│   └── backpacks.csv
├── uploads/
├── bootstrap.php
├── index.php
├── category_form.php
├── backpack_form.php
└── characteristic_form.php
```

(додано властивість photo)
(додано роботу з категоріями й фото)
(додано колонку photo)
(сюди зберігаються завантажені фото)
(головна сторінка зі списками)
(форма додавання категорії)
(форма додавання рюкзака, з файлом)
(форма додавання характеристики)

1. Оновлений клас Backpack (поле для фото)

```

<?php
// classes/Backpack.php
class Backpack {
    private array $characteristics = [];
    private ?string $photo = null;

    public function __construct(
        private int $id,
        private string $name,
        private Category $category,
        private float $price,
        private float $volumeLiters
    ) {}

    public function getId(): int { return $this->id; }
    public function getName(): string { return $this->name; }
    public function getCategory(): Category { return $this->category; }
    public function getPrice(): float { return $this->price; }
    public function getVolumeLiters(): float { return $this->volumeLiters; }
    public function getCharacteristics(): array { return $this-
>characteristics; }
    public function getPhoto(): ?string { return $this->photo; }

    public function setPhoto(string $photo): void { $this->photo = $photo; }
    public function addCharacteristic(Characteristic $c): void {
        $this->characteristics[] = $c;
    }

    public function __toString(): string {
        $chars = implode(' ', array_map(fn($c) => (string) $c, $this-
>characteristics));
        return sprintf(
            '#%d %s [%s] - %.1f л, %.2f грн (%s)',
            $this->id, $this->name, $this->category, $this->volumeLiters,
            $this->price, $chars
        );
    }
}

```

2. Оновлений репозиторій: категорії, фото, характеристики

```

<?php
// classes/BackpackRepository.php
class BackpackRepository {
    private array $backpacks = [];
    private array $categories = [];
    private int $nextBackpackId = 1;
    private int $nextCategoryId = 1;

    public function __construct(
        private string $categoriesFile,
        private string $backpacksFile
    ) {
        $this->loadCategories();
        $this->loadBackpacks();
    }

    private function loadCategories(): void {
        $handle = fopen($this->categoriesFile, 'r');
        fgetcsv($handle);
        while (($row = fgetcsv($handle)) !== false) {
            [$id, $name] = $row;
            $this->categories[(int) $id] = new Category((int) $id, $name);
            $this->nextCategoryId = max($this->nextCategoryId, (int) $id +
1);
        }
        fclose($handle);
    }

    private function loadBackpacks(): void {
        $handle = fopen($this->backpacksFile, 'r');
        fgetcsv($handle);
        while (($row = fgetcsv($handle)) !== false) {
            [$id, $name, $categoryId, $price, $volume, $charsRaw, $photo] =
$row;

            $backpack = new Backpack(
                (int) $id, $name, $this->categories[(int) $categoryId],
                (float) $price, (float) $volume
            );
            if ($charsRaw !== '') {
                foreach (explode('|', $charsRaw) as $pair) {
                    [$cName, $cValue] = explode(':', $pair, 2);
                    $backpack->addCharacteristic(new Characteristic($cName,
$cValue));
                }
            }
            if ($photo !== '') {
                $backpack->setPhoto($photo);
            }
            $this->backpacks[(int) $id] = $backpack;
            $this->nextBackpackId = max($this->nextBackpackId, (int) $id +
1);
        }
        fclose($handle);
    }

    public function getCategories(): array { return $this->categories; }
}

```

```

        public function findCategoryById(int $id): ?Category { return $this->categories[$id] ?? null; }
        public function findAll(): array { return $this->backpacks; }
        public function findById(int $id): ?Backpack { return $this->backpacks[$id] ?? null; }

        public function createCategory(string $name): Category {
            $category = new Category($this->nextCategoryId++, $name);
            $this->categories[$category->getId()] = $category;
            $this->saveCategories();
            return $category;
        }

        public function create(
            string $name, Category $category, float $price, float $volume,
            ?string $photo = null
        ): Backpack {
            $backpack = new Backpack($this->nextBackpackId++, $name, $category, $price, $volume);
            if ($photo !== null) {
                $backpack->setPhoto($photo);
            }
            $this->backpacks[$backpack->getId()] = $backpack;
            $this->save();
            return $backpack;
        }

        public function addCharacteristicToBackpack(int $backpackId, string $name, string $value): bool {
            $backpack = $this->findById($backpackId);
            if ($backpack === null) {
                return false;
            }
            $backpack->addCharacteristic(new Characteristic($name, $value));
            $this->save();
            return true;
        }

        private function saveCategories(): void {
            $handle = fopen($this->categoriesFile, 'w');
            fputcsv($handle, ['id', 'name']);
            foreach ($this->categories as $category) {
                fputcsv($handle, [$category->getId(), $category->getName()]);
            }
            fclose($handle);
        }

        private function save(): void {
            $handle = fopen($this->backpacksFile, 'w');
            fputcsv($handle,
                ['id', 'name', 'category_id', 'price', 'volume_liters', 'characteristics', 'photo']
            );
            foreach ($this->backpacks as $backpack) {
                $chars = implode('|', array_map(
                    fn(Characteristic $c) => $c->getName() . ':' . $c->getValue(),
                    $backpack->getCharacteristics()
                ));
            }
        }

```

```

    ));
    fputcsv($handle, [
        $backpack->getId(), $backpack->getName(),
        $backpack->getCategory()->getId(), $backpack->getPrice(),
        $backpack->getVolumeLiters(), $chars, $backpack->getPhoto()
    ], 'utf-8', 'r');
}
fclose($handle);
}
}

```

bootstrap.php не змінюється порівняно з лабораторною роботою № 4 — він так само підключає всі класи через `require_once` і створює один спільний об'єкт `$repository`, яким користуються всі сторінки застосунку (`index.php` та три файли форм).

3. Головна сторінка (`index.php`)

```

<?php
require_once __DIR__ . '/bootstrap.php';
?>
<!DOCTYPE html>
<html lang="uk">
<head><meta charset="UTF-8"><title>Каталог рюкзаків</title></head>
<body>
    <h1>Категорії</h1>
    <ul>
        <?php foreach ($repository->getCategories() as $category): ?>
            <li><?php echo htmlspecialchars($category->getName()) ?></li>
        <?php endforeach; ?>
    </ul>
    <p><a href="category_form.php">+ Додати категорію</a></p>

    <h1>Рюкзаки</h1>
    <ul>
        <?php foreach ($repository->findAll() as $backpack): ?>
            <li>
                <?php if ($backpack->getPhoto()): ?>
                    
                <?php endif; ?>
                <?php echo htmlspecialchars((string) $backpack) ?>
            </li>
        <?php endforeach; ?>
    </ul>
    <p><a href="backpack_form.php">+ Додати рюкзак</a></p>
    <p><a href="characteristic_form.php">+ Додати характеристику</a></p>
</body>
</html>

```

4. Форма без файлу (`category_form.php`)

```

<?php
require_once __DIR__ . '/bootstrap.php';

$error = null;

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name'] ?? '');

    if ($name === '') {
        $error = 'Назва категорії обов'язкова.';
    } else {
        $repository->createCategory($name);
        header('Location: index.php'); // PRG-патерн (тема 9, розділ 5)
        exit;
    }
}
?>
<!DOCTYPE html>
<html lang="uk">
<head><meta charset="UTF-8"><title>Додати категорію</title></head>
<body>
    <h1>Нова категорія</h1>
    <?php if ($error): ?>
        <p style="color:red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>
    <form method="post" action="category_form.php">
        <label>Назва: <input type="text" name="name" required></label>
        <button type="submit">Зберегти</button>
    </form>
    <p><a href="index.php">← Назад до каталогу</a></p>
</body>
</html>

```

5. Форма з полем файлу (backpack_form.php)

```

<?php
require_once __DIR__ . '/bootstrap.php';

$error = null;

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name'] ?? '');
    $category = $repository->findCategoryById((int) ($_POST['category_id'] ?? 0));
    $price = (float) ($_POST['price'] ?? 0);
    $volume = (float) ($_POST['volume'] ?? 0);

    if ($name === '' || $category === null || $price <= 0 || $volume <= 0) {
        $error = 'Заповніть усі поля коректно.';
    } else {
        $photoFilename = null;

        if (!empty($_FILES['photo']['name']) && $_FILES['photo']['error'] === UPLOAD_ERR_OK) {
            $allowedTypes = ['image/jpeg', 'image/png', 'image/webp'];
            $mimeType = mime_content_type($_FILES['photo']['tmp_name']);

            if (!in_array($mimeType, $allowedTypes, true)) {
                $error = 'Дозволені лише зображення JPEG, PNG чи WEBP.';
            } else {
                $extension = pathinfo($_FILES['photo']['name'], PATHINFO_EXTENSION);
                $photoFilename = uniqid('backpack_', true) . '.' . $extension;

                move_uploaded_file(
                    $_FILES['photo']['tmp_name'],
                    __DIR__ . '/uploads/' . $photoFilename
                );
            }
        }

        if ($error === null) {
            $repository->create($name, $category, $price, $volume, $photoFilename);
            header('Location: index.php');
            exit;
        }
    }
}

?>
<!DOCTYPE html>
<html lang="uk">
<head><meta charset="UTF-8"><title>Додати рюкзак</title></head>
<body>
    <h1>Новий рюкзак</h1>
    <?php if ($error): ?>
        <p style="color:red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>
    <form method="post" action="backpack_form.php" enctype="multipart/form-data">
        <label>Назва: <input type="text" name="name" required></label><br>
        <label>Категорія:

```

```

        <select name="category_id" required>
            <?php foreach ($repository->getCategories() as $category): ?>
                <option value="<?= $category->getId() ?>">
                    <?= htmlspecialchars($category->getName()) ?>
                </option>
            <?php endforeach; ?>
        </select>
    </label><br>
    <label>Ціна (грн): <input type="number" step="0.01" name="price"
required></label><br>
    <label>Об'єм (л): <input type="number" step="0.1" name="volume"
required></label><br>
    <label>Фото: <input type="file" name="photo"
accept="image/*"></label><br>
    <button type="submit">Зберегти</button>
</form>
    <p><a href="index.php">← Назад до каталогу</a></p>
</body>
</html>

```

Порядок дій у обробнику принциповий: спершу перевіряються текстові поля, потім — файл, і лише якщо жодної помилки не виникло, викликається `$repository->create()` і відбувається перенаправлення. Якщо MIME-тип виявився недозволеним, `$error` встановлюється, `$photoFilename` так і лишається `null`, і сторінка знову показує форму — тепер із повідомленням про помилку — замість збереження некоректного файлу.

6. Форма додавання характеристики (`characteristic_form.php`)

```

<?php
require_once __DIR__ . '/bootstrap.php';

$error = null;

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $backpackId = (int) ($_POST['backpack_id'] ?? 0);
    $name = trim($_POST['name'] ?? '');
    $value = trim($_POST['value'] ?? '');

    if ($name === '' || $value === '' || $repository->findById($backpackId)
    === null) {
        $error = 'Заповніть усі поля коректно.';
    } else {
        $repository->addCharacteristicToBackpack($backpackId, $name, $value);
        header('Location: index.php');
        exit;
    }
}
?>
<!DOCTYPE html>
<html lang="uk">
<head><meta charset="UTF-8"><title>Додати характеристику</title></head>
<body>
    <h1>Нова характеристика</h1>
    <?php if ($error): ?>
        <p style="color:red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>
    <form method="post" action="characteristic_form.php">
        <label>Рюкзак:
            <select name="backpack_id" required>
                <?php foreach ($repository->findAll() as $backpack): ?>
                    <option value="<?= $backpack->getId() ?>">
                        <?= htmlspecialchars($backpack->getName()) ?>
                    </option>
                <?php endforeach; ?>
            </select>
        </label><br>
        <label>Назва характеристики: <input type="text" name="name"
required></label><br>
        <label>Значення: <input type="text" name="value"
required></label><br>
        <button type="submit">Зберегти</button>
    </form>
    <p><a href="index.php">← Назад до каталогу</a></p>
</body>
</html>

```

Контрольні запитання

- Чим SSR принципово відрізняється від підходу з лабораторної роботи № 4, де html-сторінка підвантажувала дані через fetch()?
- Навіщо потрібен атрибут enctype="multipart/form-data" і що станеться з файлом у формі без нього?

- Чому атрибут `required` у HTML-полі не звільняє від перевірки того самого поля на сервері?
- Опишіть призначення кожного з ключів масиву `$_FILES['photo']`: `name`, `tmp_name`, `error`, `size`, `type`.
- Чому не можна довіряти значенню `$_FILES['photo']['type']` і як його правильно перевірити?
- Навіщо завантаженому файлу присвоюють нове, згенероване ім'я замість збереження оригінального?
- У чому саме полягає проблема, яку вирішує патерн `Post/Redirect/Get`, і як `header('Location: ...')` її вирішує?
- Чому обробники форм у прикладі спершу перевіряють усі умови й лише потім викликають методи репозиторію, а не навпаки?