Отчет по аудиту безопасности веб-приложения

Ссылка на репозиторий: *https://github.com/fastyrer/fastyrer.github.io*

🔓 1. XSS (Cross-Site Scripting) — *2 балла*

❗Описание:

XSS позволяет злоумышленнику внедрить скрипты в сайт (например, через поля формы), что может привести к краже cookies и перехвату сессий.

✅ Меры защиты:

- Применение htmlspecialchars() при выводе данных пользователя:

php

КопироватьРедактировать

<input type="text" name="name" value="<?= htmlspecialchars($form_data['name']) ?>">

- В тексте сообщений:

<?php if ($success): ?>

<?= htmlspecialchars($_SESSION['user_login']) ?>, ваши данные сохранены

<?php endif; ?>

□ 2. Information Disclosure — *1 балл*

❗Описание:

Утечка внутренней информации (пути файлов, сообщения об ошибках, дампы БД и пр.) может привести к дальнейшим атакам.

✅ Меры защиты:

- Отключена директива показа ошибок:

ini_set('display_errors', 0);

error_reporting(0);

- Ошибки логируются отдельно:

error_log("Ошибка подключения к БД: " . $e->getMessage());

- Страница ошибки не содержит технической информации (например, error.php).

🛡 3. SQL Injection — *2 балла*

❗Описание:

Возможность внедрения вредоносного SQL-запроса через поля ввода.

✅ Меры защиты:

- Использование подготовленных выражений (PDO):

```
$stmt = $pdo->prepare("SELECT * FROM users WHERE login = :login AND password = :password");
$stmt->execute([
  'login' => $_POST['login'],
  'password' => md5($_POST['password'])
]);
```

- Отсутствие прямой вставки переменных в SQL.

□ 4. CSRF (Cross-Site Request Forgery) — *2 балла*

❗Описание:

Позволяет злоумышленнику отправлять поддельные запросы от имени пользователя без его ведома.

✅ Меры защиты:

- Генерация и проверка CSRF-токена:

```
// Генерация
if (empty($_SESSION['csrf_token'])) {
  $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}

// В форме
<input type="hidden" name="csrf_token" value="<?= $_SESSION['csrf_token'] ?>">

// На сервере
if ($_POST['csrf_token'] !== $_SESSION['csrf_token']) {
```

```
    die("Недопустимый запрос (CSRF)");
}
```

🗁 5. Include / File Inclusion — *0.5 балла*

❗Описание:

Через include или require можно подключить вредоносный файл, особенно при использовании данных из URL (?page=).

✅ Меры защиты:

- Исключение динамического подключения файлов от пользователя:

```
require_once 'config.php'; // Без $_GET
```

- Или ограничение списка разрешенных значений:

```
$pages = ['home', 'form', 'about'];
if (in_array($_GET['page'], $pages)) {
    include "pages/{$_GET['page']}.php";
} else {
    include "pages/404.php";
}
```

📤 6. Upload — *0.5 балла*

❗Описание:

Пользователь может загрузить и выполнить вредоносный файл (например, .php).

✅ Меры защиты:

- Ограничение по MIME-типу и расширению:

```
$allowed = ['image/png', 'image/jpeg'];
if (!in_array($_FILES['file']['type'], $allowed)) {
    die("Недопустимый формат файла");
}
```

- Генерация уникального имени файла:

```
$filename = uniqid() . '.' . pathinfo($_FILES['file']['name'], PATHINFO_EXTENSION);
```

move_uploaded_file($_FILES['file']['tmp_name'], "/uploads/$filename");

- Запрет на выполнение PHP в /uploads (через .htaccess):

```
<FilesMatch "\.php$">
  Deny from all
</FilesMatch>
```

✅ Вывод

Приложение успешно прошло аудит безопасности. Были реализованы следующие меры:

Уязвимость	Статус	Метод защиты
XSS	✓ Защищено	htmlspecialchars()
Information Disclosure	✓ Защищено	отключен вывод ошибок, error_log()
SQL Injection	✓ Защищено	PDO + prepare/execute
CSRF	✓ Защищено	Токен сессии + скрытое поле + проверка
Include	✓ Защищено	whitelist страниц, запрет include через URL
Upload	✓ Защищено	MIME-проверка, уникальные имена, .htaccess