


```
*/
$files = array_filter(scandir($uploads_dir), function($f) use
($uploads_dir) {
    return is_file($uploads_dir . '/' . $f) && in_array(pathinfo($f,
PATHINFO_EXTENSION), ['js', 'php', 'json', 'html', 'css']);
});

/**
 * Обработка загрузки файлов
 */
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_FILES['files'])) {
    foreach ($_FILES['files']['name'] as $i => $name) {
        if ($_FILES['files']['error'][$i] === UPLOAD_ERR_OK) {
            move_uploaded_file($_FILES['files']['tmp_name'][$i],
$uploads_dir . '/' . basename($name));
        }
    }
    header('Location: ./index.php');
    exit;
}
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Llama Safe Editor</title>
    <style>
        body { font-family: sans-serif; margin: 30px; background:
#f4f6f9; color: #333; }
        .container { display: flex; gap: 20px; }
        .panel { background: white; padding: 20px; border-radius: 8px;
box-shadow: 0 2px 5px rgba(0,0,0,0.1); flex: 1; }
        textarea { width: 100%; height: 100px; box-sizing: border-box;
padding: 10px; margin-bottom: 10px; }
        button { background: #007bff; color: white; border: none;
padding: 10px 15px; border-radius: 4px; cursor: pointer; font-size:
16px; }
        button:hover { background: #0056b3; }
        .btn-success { background: #28a745; display: none; margin-top:
10px; }
        .btn-success:hover { background: #218838; }
        pre { background: #272822; color: #f8f8f2; padding: 15px;
border-radius: 5px; overflow-x: auto; max-height: 250px; }
    </style>
</head>
<body>
    <h1>🔒 Безопасный редактор кода через Qwen/Llama</h1>
    <div class="container">
        <!-- Слева: файлы -->
        <div class="panel" style="max-width: 280px;">
            <h3>1. Загрузка исходников</h3>
```

```

        <form method="post" enctype="multipart/form-data">
            <input type="file" name="files[]" multiple
required><br><br>
            <button type="submit">Загрузить</button>
        </form>
        <h4>Файлы в проекте:</h4>
        <ul>
            <?php foreach ($files as $f): ?>
                <li>□ <b><?= htmlspecialchars($f) ?></b></li>
            <?php endforeach; if (empty($files)) echo '<li>Нет
файлов</li>'; ?>
        </ul>
    </div>

    <!-- Справа: работа с AI -->
    <div class="panel">
        <h3>2. Что нужно сделать?</h3>
        <textarea id="prompt" placeholder="Например: 'Добавь
валидацию'"></textarea>
        <button onclick="askLlama()">Сгенерировать изменения</button>

        <div id="status" style="margin-top:10px; font-
weight:bold;"></div>

        <!-- Блок предпросмотра -->
        <div id="preview-area" style="display:none; margin-
top:20px;">
            <h3 style="color: #007bff;">□□ Код от нейросети
(Предпросмотр)</h3>
            <div id="ai-code-output"></div>
            <button id="apply-btn" class="btn-success"
onclick="applyChanges()">Применить эти изменения в файлы</button>
        </div>

        <h3 style="margin-top:30px;">Текущий код на сервере:</h3>
        <div>
            <?php foreach ($files as $f): $content =
file_get_contents($uploads_dir . '/' . $f); ?>
            <h5><?= htmlspecialchars($f) ?></h5>
            <pre><code><?= htmlspecialchars($content)
?></code></pre>
            <?php endforeach; ?>
        </div>
    </div>
</div>

<script>
    let lastAiResult = null;

    async function askLlama() {
        const prompt = document.getElementById('prompt').value;

```

```
const status = document.getElementById('status');
const previewArea = document.getElementById('preview-
area');

const aiOutput = document.getElementById('ai-code-output');
const applyBtn = document.getElementById('apply-btn');

if(!prompt) return alert('Введите задачу!');

status.style.color = '#28a745';
status.innerText = '🔄 Модель думает над кодом... Подождите.';
previewArea.style.display = 'none';
applyBtn.style.display = 'none';

try {
    const res = await fetch('./edit.php', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ action: 'ask', prompt })
    });

    // Получаем чистый текст ответа бэкенда
    const rawText = await res.text();

    // Пытаемся распарсить его вручную
    let data;
    try {
        data = JSON.parse(rawText);
    } catch(jsonErr) {
        status.style.color = '#dc3545';
        status.innerHTML = '❌ Ошибка PHP на сервере:<br><div
style="text-align:left; background:#fff; padding:10px; border:1px solid
#ccc; font-family:monospace; margin-top:10px; color:#000;">' + rawText
+ '</div>';

        return;
    }

    if(data.success && data.files.length > 0) {
        status.innerText = '✅ Код сгенерирован! Проверьте его
ниже.';

        lastAiResult = data.files;

        aiOutput.innerHTML = '';
        data.files.forEach(f => {
            const div = document.createElement('div');
            div.innerHTML = `<h4>Файл:
${f.name}</h4><pre><code>${f.content.replace(/&/g,
"&amp;").replace(/</g, "&lt;").replace(/>/g, "&gt;")}</code></pre>`;
            aiOutput.appendChild(div);
        });

        previewArea.style.display = 'block';
    }
}
```

```
        applyBtn.style.display = 'block';
    } else {
        status.style.color = '#dc3545';
        status.innerHTML = '❌ Ошибка: ' + (data.error ||
'Model не выдала разметку файлов.');
```

Файл edit.php

[edit.php](#)

```
<?php
/**
*
```

```

=====
=====
*   edit.php — ОБРАБОТЧИК ЗАПРОСОВ ДЛЯ РЕДАКТИРОВАНИЯ ФАЙЛОВ И РАБОТЫ С
AI
*
=====
=====
*
*   УЧЕБНОЕ НАЗНАЧЕНИЕ:
*   Обработывает запросы от index.php для генерации изменений в коде с помощью
AI
*   и сохранения обновленных файлов на сервер.
*
*   КАК РАБОТАЕТ:
*   1. Принимает JSON-запросы с действием 'ask' или 'save'.
*   2. Для действия 'ask':
*       - Получает задачу для AI из запроса.
*       - Собирает содержимое всех файлов в директории uploads.
*       - Отправляет задачу и контекст файлов на сервер Ollama.
*       - Парсит ответ от Ollama, разделяя его по маркерам [START_FILE].
*       - Возвращает обновленные файлы клиенту.
*   3. Для действия 'save':
*       - Принимает массив файлов с содержимым из запроса.
*       - Сохраняет обновленное содержимое каждого файла в директорию uploads.
*
*   ВНИМАНИЕ: этот файл только для обучения. На боевом сервере его нужно защищать.
*
=====
=====
*/

set_time_limit(0);
ini_set('display_errors', 1);
error_reporting(E_ALL);

header('Content-Type: application/json');

$uploads_dir = __DIR__ . '/uploads';
$input = json_decode(file_get_contents('php://input'), true);
$action = isset($input['action']) ? $input['action'] : '';

if ($action === 'ask') {
    $prompt = isset($input['prompt']) ? $input['prompt'] : '';

    if (!is_dir($uploads_dir)) {
        echo json_encode(['success' => false, 'error' => 'Шаг 1: Папка
uploads не найдена']);
        exit;
    }

    $files = array_filter(scandir($uploads_dir), function($f) use

```

```
($uploads_dir) {
    return is_file($uploads_dir . '/' . $f);
});

$projectContext = "";
foreach ($files as $f) {
    $projectContext .= "=== НАЧАЛО ФАЙЛА: $f ===\n" .
file_get_contents($uploads_dir . '/' . $f) . "\n=== КОНЕЦ ФАЙЛА: $f
===\n\n";
}

$systemPrompt = "Ты веб-разработчик. Задача: $prompt\nТекущие
файлы:\n$projectContext\nИзмени код и верни файлы ПОЛНОСТЬЮ. Строго
используй теги:\n[START_FILE: имя]\nкод\n[END_FILE: имя]";

$postData = json_encode([
    'model' => 'qwen2.5-coder:14b',
    'prompt' => $systemPrompt,
    'stream' => false,
    'options' => [
        'temperature' => 0,
        'num_ctx' => 8192
    ]
]);

$fp = @fsockopen('127.0.0.1', 11434, $errno, $errstr, 10);
if (!$fp) {
    echo json_encode(['success' => false, 'error' => "Не удалось
подключиться к Ollama на 127.0.0.1:11434. Ошибка: $errstr ($errno)"]);
    exit;
}

// Запрашиваем через HTTP/1.0, чтобы исключить chunked transfer
encoding на бэкенде
$out = "POST /api/generate HTTP/1.0\r\n";
$out .= "Host: 127.0.0.1\r\n";
$out .= "Content-Type: application/json\r\n";
$out .= "Content-Length: " . strlen($postData) . "\r\n";
$out .= "Connection: close\r\n\r\n";
$out .= $postData;

fwrite($fp, $out);

$response = '';
while (!feof($fp)) {
    $response .= fgets($fp, 4096);
}
fclose($fp);

// Гарантированно отрезаем заголовки по двойному переносу строки
$parts = explode("\r\n\r\n", $response, 2);
```

```
$body = isset($parts[1]) ? trim($parts[1]) : '';

if (empty($body)) {
    // Пробуем отрезать по одиночным \n на случай Unix-формата заголовков
сервера
    $parts = explode("\n\n", $response, 2);
    $body = isset($parts[1]) ? trim($parts[1]) : '';
}

// Разбор JSON
$responseData = json_decode($body, true);
$aiText = isset($responseData['response']) ?
$responseData['response'] : '';

if (empty($aiText)) {
    echo json_encode([
        'success' => false,
        'error' => 'Шаг 4: Не удалось распарсить JSON от Ollama.',
        'debug_raw_body' => substr($body, 0, 300)
    ]);
    exit;
}

// Шаг 5: Парсинг маркеров [START_FILE]
$parsedFiles = [];
preg_match_all('/\[START_FILE:\s*([^\s]+)\]([^\s\S]*?)\[END_FILE:\s*\1
\]/', $aiText, $matches, PREG_SET_ORDER);

foreach ($matches as $match) {
    $parsedFiles[] = [
        'name' => trim($match[1]),
        'content' => trim($match[2])
    ];
}

if (empty($parsedFiles)) {
    echo json_encode([
        'success' => false,
        'error' => 'Шаг 5: Модель проигнорировала маркеры [START_FILE]
и выдала обычный текст.',
        'ai_raw_text' => substr($aiText, 0, 300)
    ]);
    exit;
}

echo json_encode(['success' => true, 'files' => $parsedFiles]);
exit;
}

if ($action === 'save') {
    $filesToSave = isset($input['files']) ? $input['files'] : [];
}
```



```
foreach ($filesToSave as $f) {  
    $fileName = basename(trim($f['name']));  
    file_put_contents($uploads_dir . '/' . $fileName,  
$f['content']);  
}  
echo json_encode(['success' => true]);  
exit;  
}  
  
echo json_encode(['success' => false, 'error' => 'Неверное действие']);
```

From:

<https://wwoss.ru/> - **worldwide open-source software**

Permanent link:

https://wwoss.ru/doku.php?id=tmp_ollama&rev=1785135512

Last update: **2026/07/27 09:58**

