Flask + SQLite File Upload App — Fully Explained (with Templates & CSS)

This edition adds full listings for index.html, view_file.html, and the complete style.css, each with line numbers and brief explanations.

Context Recap

The app lets users upload .txt files, stores metadata (uploader, time, message, genre) in SQLite, supports search/filter, and displays file contents. Frontend is separated using Flask templates.

Template: templates/base.html

Purpose: Provides a shared layout, loads CSS, shows flash messages, and yields a content block.

```
<!DOCTYPE html>
<html>
<head>
    <title>{{ title or "Flask App" }}</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <header>
        <h1>{{ title or "Flask App" }}</h1>
    </header>
    <main>
        <p class="flash-messages">
            {% with messages = get_flashed_messages() %}
                {% if messages %}
                    {% for msg in messages %}
                        <div class="flash">{{ msg }}</div>
                    {% endfor %}
                {% endif %}
            {% endwith %}
        </p>
        {% block content %}{% endblock %}
    </main>
</body>
</html>
```

Template: templates/index.html

Purpose: Upload form (file + message + genre), search/filter controls, and the uploads table with a View link.

```
{% extends "base.html" %}
{% block content %}

<h2>Upload a Text File</h2>
<form method="POST" enctype="multipart/form-data">
    <input type="file" name="file" accept=".txt" required><br><br>

    <textarea name="message" placeholder="Enter a message about this file..." rows="3" cols="40"></

    <label for="genre">Select a genre:</label>
    <select name="genre" required>
        <option value="Analysis">Analysis</option>
        <option value="Development">Development</option>
        <option value="Programming">Programming</option>
        <option value="Other">Other</option>
    </select><br><br>

    <button type="submit">Upload</button>
</form>

<h2>Search & Filter</h2>
<form method="GET" action="/">
    <input type="text" name="search" placeholder="Search by filename or message..." value="{{ reque

    <select name="genre_filter">
        <option value="">All Genres</option>
        {% for g in genres %}
            <option value="{{ g }}" {% if request.args.get('genre_filter') == g %}selected{% endif
        {% endfor %}
    </select>

    <select name="user_filter">
        <option value="">All Users</option>
        {% for u in users %}
            <option value="{{ u }}" {% if request.args.get('user_filter') == u %}selected{% endif %
        {% endfor %}
    </select>

    <button type="submit">Filter</button>
    <a href="{{ url_for('index') }}">Clear</a>
</form>

<h2>Upload History</h2>
<table>
    <thead>
        <tr>
            <th>Filename</th>
            <th>Uploaded By</th>
            <th>Upload Time</th>
            <th>Message</th>
            <th>Genre</th>
            <th>Action</th>
        </tr>
    </thead>
    <tbody>
    {% for f, u, t, m, g in uploads %}
        <tr>
            <td>{{ f }}</td>
            <td>{{ u }}</td>
            <td>{{ t }}</td>
            <td>{{ m }}</td>
            <td>{{ g }}</td>
            <td><a href="{{ url_for('view_file', filename=f) }}">View</a></td>
        </tr>
    {% endfor %}
    </tbody>
</table>

{% endblock %}
```

Template: templates/view_file.html

Purpose: Displays file content plus its metadata (uploader, time, message, genre).

```
{% extends "base.html" %}
{% block content %}

<h2>Viewing File: {{ filename }}</h2>

<p><b>Uploaded by:</b> {{ username }}<br>
<b>Upload time:</b> {{ upload_time }}<br>
<b>Message:</b> {{ message }}<br>
<b>Genre:</b> {{ genre }}</p>

<pre class="file-view">{{ content }}</pre>
```

```

<p><a href="{{ url_for('index') }}">■ Back</a></p>

{% endblock %}
```

Stylesheet: static/style.css (FULL)

Purpose: Basic clean styling for header, forms, tables, flash messages, and file viewer.

```
/* Reset and base */
body {
    font-family: Arial, sans-serif;
    margin: 20px;
    background: #f8f9fa;
    color: #333;
}

header {
    background: #4a90e2;
    color: white;
    padding: 15px;
    border-radius: 8px;
    margin-bottom: 20px;
}

h1, h2 {
    margin-top: 0;
}

/* Forms */
form {
    margin-bottom: 20px;
    padding: 15px;
    background: white;
    border: 1px solid #ddd;
    border-radius: 8px;
}

input[type="file"],
input[type="text"],
textarea,
select {
    padding: 8px;
    margin: 5px 0;
    width: 100%;
    max-width: 500px;
    border: 1px solid #ccc;
    border-radius: 5px;
    box-sizing: border-box;
}

button {
    padding: 8px 15px;
    background: #4a90e2;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background: #357ab7;
}

/* Table */
table {
    width: 100%;
    border-collapse: collapse;
    background: white;
    margin-top: 15px;
```

```css
    border-radius: 8px;
    overflow: hidden;
}

table th, table td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: left;
    vertical-align: top;
}

table th {
    background: #f1f1f1;
}

/* Flash messages */
.flash {
    background: #ffdddd;
    border: 1px solid #ff8888;
    padding: 10px;
    border-radius: 5px;
    margin-bottom: 10px;
}

/* File viewer */
pre,
pre.file-view {
    background: #f4f4f4;
    padding: 15px;
    border-radius: 5px;
    white-space: pre-wrap; /* allow wrapping for long lines */
    word-wrap: break-word;
}
```