Flask + SQLite File Upload App Tutorial

This document walks you through the process of building a Flask web application with SQLite that allows users to upload text files, store metadata (uploader, timestamp, message, genre), and view/search/filter uploaded files. We also structured the project with templates and added CSS for styling.

Project Structure

```
my_flask_app/
■■■ app.py
■■■ people.db
■■■ uploads/
■■■ templates/
■   ■■■ base.html
■   ■■■ index.html
■   ■■■ view_file.html
■■■ static/
■   ■■■ style.css
```

Backend: app.py

This is the main Flask application. It initializes the database, handles file uploads, saves metadata, implements search/filter, and renders templates.

```
import sqlite3
import os
import getpass
from datetime import datetime
from flask import Flask, render_template, request, redirect, url_for, flash, abort

app = Flask(__name__)
app.secret_key = "supersecretkey"
UPLOAD_FOLDER = "uploads"
ALLOWED_EXTENSIONS = {"txt"}

os.makedirs(UPLOAD_FOLDER, exist_ok=True)
app.config["UPLOAD_FOLDER"] = UPLOAD_FOLDER

def init_db():
    conn = sqlite3.connect("people.db")
    c = conn.cursor()
    c.execute('''CREATE TABLE IF NOT EXISTS uploads (
                    id INTEGER PRIMARY KEY,
                    filename TEXT,
                    username TEXT,
                    upload_time TEXT,
                    message TEXT,
                    genre TEXT)''')
    conn.commit()
    conn.close()

def allowed_file(filename):
    return "." in filename and filename.rsplit(".", 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST" and "file" in request.files:
        file = request.files["file"]
        message = request.form.get("message", "").strip()
        genre = request.form.get("genre", "Uncategorized")

        if file.filename == "":
            flash("No selected file")
            return redirect(request.url)

        if file and allowed_file(file.filename):
            filepath = os.path.join(app.config["UPLOAD_FOLDER"], file.filename)
            file.save(filepath)
            username = getpass.getuser()
            upload_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

            conn = sqlite3.connect("people.db")
            c = conn.cursor()
            c.execute("INSERT INTO uploads (filename, username, upload_time, message, genre) VALUES (?,
                      (file.filename, username, upload_time, message, genre))
            conn.commit()
            conn.close()
            flash(f"File '{file.filename}' uploaded by {username} at {upload_time}")
        else:
            flash("Invalid file type! Only .txt files are allowed.")
        return redirect(url_for("index"))

    search_query = request.args.get("search", "").strip()
    genre_filter = request.args.get("genre_filter", "")
    user_filter = request.args.get("user_filter", "")

    conn = sqlite3.connect("people.db")
    c = conn.cursor()
    query = "SELECT filename, username, upload_time, message, genre FROM uploads WHERE 1=1"
    params = []

    if search_query:
        query += " AND (filename LIKE ? OR message LIKE ?)"
        params.extend([f"%{search_query}%", f"%{search_query}%"])
    if genre_filter:
        query += " AND genre=?"
        params.append(genre_filter)
    if user_filter:
        query += " AND username=?"
        params.append(user_filter)

    query += " ORDER BY id DESC"
    c.execute(query, params)
    uploads = c.fetchall()

    c.execute("SELECT DISTINCT genre FROM uploads")
    genres = [row[0] for row in c.fetchall() if row[0]]

    c.execute("SELECT DISTINCT username FROM uploads")
    users = [row[0] for row in c.fetchall() if row[0]]
    conn.close()

    return render_template("index.html", uploads=uploads, genres=genres, users=users)

@app.route("/view/<filename>")
def view_file(filename):
    filepath = os.path.join(app.config["UPLOAD_FOLDER"], filename)
    if not os.path.isfile(filepath):
        abort(404, description="File not found")
    with open(filepath, "r", encoding="utf-8") as f:
        content = f.read()
    conn = sqlite3.connect("people.db")
    c = conn.cursor()
    c.execute("SELECT username, upload_time, message, genre FROM uploads WHERE filename=?", (filename,))
    row = c.fetchone()
    conn.close()
    if row:
        username, upload_time, message, genre = row
    else:
        username, upload_time, message, genre = ("Unknown", "Unknown", "", "Uncategorized")
    return render_template("view_file.html", filename=filename, content=content,
                           username=username, upload_time=upload_time, message=message, genre=genre)

if __name__ == "__main__":
    init_db()
    app.run(debug=True)
```

Templates

base.html

```
<!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>
```

index.html

```
{% 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"></texta
    <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="{{ request.ar
    <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 border="1" cellpadding="5">
    <tr>
        <th>Filename</th>
        <th>Uploaded By</th>
        <th>Upload Time</th>
        <th>Message</th>
        <th>Genre</th>
        <th>Action</th>
    </tr>
    {% 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 %}
</table>
{% endblock %}
```

view_file.html

```
{% 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 style="border:1px solid #ccc; padding:10px; background:#f9f9f9;">{{ content }}</pre>
<a href="{{ url_for('index') }}">■ Back</a>
{% endblock %}
```

CSS (static/style.css)

```
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;
}
form {
    margin-bottom: 20px;
    padding: 15px;
    background: white;
    border: 1px solid #ddd;
    border-radius: 8px;
}
input, textarea, select {
    padding: 8px;
    margin: 5px 0;
    width: 100%;
    max-width: 400px;
    border: 1px solid #ccc;
    border-radius: 5px;
}
button {
    padding: 8px 15px;
    background: #4a90e2;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}
button:hover {
    background: #357ab7;
}
table {
    width: 100%;
    border-collapse: collapse;
    background: white;
    margin-top: 15px;
    border-radius: 8px;
    overflow: hidden;
}
table th, table td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: left;
}
table th {
    background: #f1f1f1;
}
.flash {
    background: #ffdddd;
    border: 1px solid #ff8888;
    padding: 10px;
    border-radius: 5px;
    margin-bottom: 10px;
}
pre {
    background: #f4f4f4;
    padding: 15px;
    border-radius: 5px;
    white-space: pre-wrap;
}
```