

Team CodeBlooded

Daniel Park, Aditya Anand, Jackie Zeng, Jayden Zhang

SoftDev

P04

TARGET SHIP DATE: {2025-04-25}

Description

Create an interactive Flask web application that helps users explore and visualize Spotify song trends from 2000–2019 using data analysis and rich, dynamic visualizations. Using D3.js, we will craft interactive data visualizations that reveal relationships between musical features like danceability, energy, popularity, tempo, and more.

Program Components

1. Frontend (Tailwind + D3.js + HTML templates)

- **home.html**: Landing page with summary stats, intro, and nav.
- **visualize.html**: Scatter plot of danceability vs. popularity
- **coorelationmatrix.html**: Correlation matrix
- **histo.html**: Histogram (e.g., average energy by year or genre)
- **topcharts.html**: Shows the top artists and songs.
- **about.html**: Info page and links to dataset/API used.
- **login.html**: Allows user sign in to account
- **register.html**: Allows user to register an account

2. Middleware (Flask)

- **__init__.py**: App entrypoint, Flask routing.
- **app.py**: Routing logic.
- **d3_utils.py**: Serve data as JSON
- **plot_utils.py**: Functions to return graphs (plotly/seaborn).
- **filter_utils.py**: Functions to filter dataset based on user input.
- **spotify_data.csv**: Preprocessed version of songs_normalize.csv file from dataset.
- **templates/**: HTML Jinja templates
- **static/js/**: D3.js scripts

3. Backend (SQLite)

- One central table (songs) with all tracks and metadata.

Dataset:

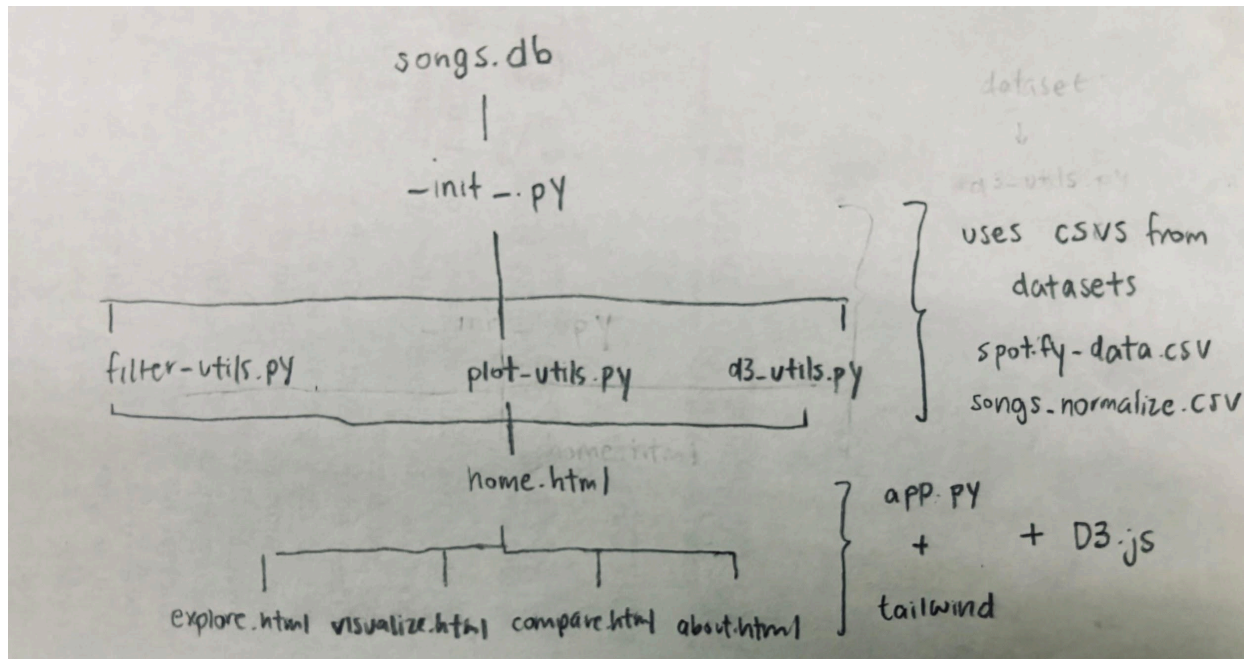
- Spotify Dataset:
- Top Hits Spotify from 2000-2019
- <https://www.kaggle.com/code/varunsaikanuri/spotify-data-visualization>
- Data Cleaning Algorithm (PANDAS)

Database Schema

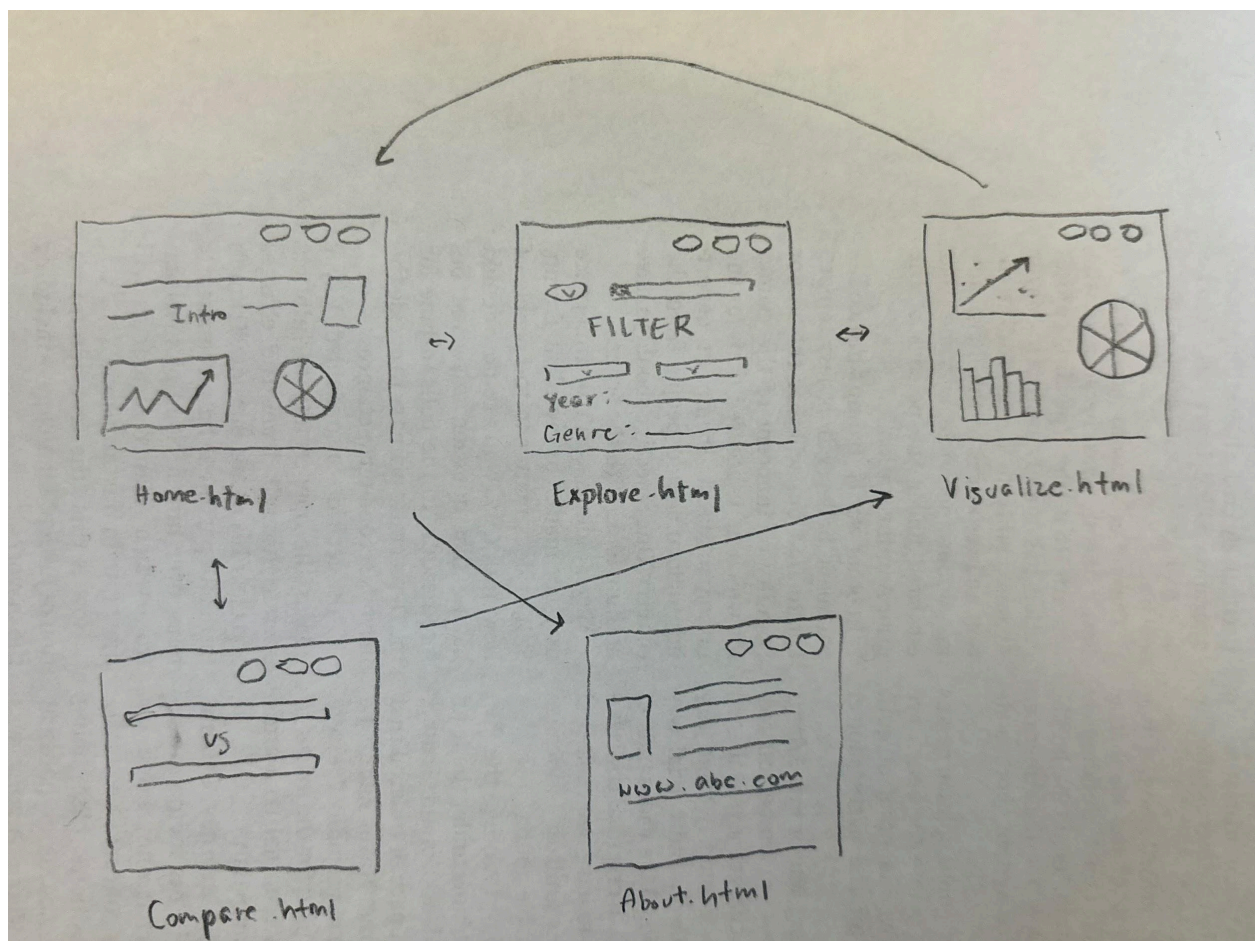
songs.db		
artist	TEXT	Song artist
song	TEXT	Song title
year	INTEGER	Year released
popularity	FLOAT	0–100
danceability	FLOAT	0–1
energy	FLOAT	0–1
speechiness	FLOAT	Song artist
acousticness	FLOAT	0–1
instrumentalness	FLOAT	0–1
liveness	FLOAT	0–1
tempo	FLOAT	BPM
genre	FLOAT	Genre label
valence	FLOAT	0–1 (happy/sad scale)

Component Relationships

- Frontend (Tailwind + D3.js) communicates with Flask middleware, which serves data from the SQLite backend.
- Middleware processes user input and queries the database.
- D3.js visualizes processed data dynamically on the frontend.




Site Map



Why Tailwind?

- Lightweight and utility-first, enabling faster styling.
- Ensures responsive and consistent UI across all pages.
- Layout (grid, flexbox), responsive breakpoints
- Hover states, transitions
- Consistent navigation bar and visual identity

Why D3?

- Cause its #gudfam 
- Very customizable
- Easy-to-visualize graphs for time-stamped data like music and comparison of listening preferences of Spotify users

Task Assignments

TASK	Daniel Park	Aditya Anand	Jackie Zeng	Jayden Zhang
Set up Flask and SQLite3 environment		X		X
Build User Authentication and Management Backend	X		X	
Build D3 Component	X	X	X	X
Build Database	X		X	
Frontend (HTML Templates)	X	X	X	X
Frontend (Tailwind)		X		X
Final Testing and Bug Fixing	X	X	X	X