**TECHNICAL DOCUMENTATION**

Repo: <https://github.com/Bcsully3/CS3300_Music>

Kanban Board: <https://github.com/users/Bcsully3/projects/1/views/1>

Resources:

[Colors · Bootstrap (getbootstrap.com)](https://getbootstrap.com/docs/4.0/utilities/colors/)

<https://chat.openai.com/share/ef1db4e3-df54-49e8-944f-84845f6227a5>

[Django Project MVT Structure - GeeksforGeeks](https://www.geeksforgeeks.org/django-project-mvt-structure/)

[Django documentation | Django documentation | Django (djangoproject.com)](https://docs.djangoproject.com/en/4.2/)

[Models | Django documentation | Django (djangoproject.com)](https://docs.djangoproject.com/en/4.2/topics/db/models/)

Set up environment:

1. Set up and activate virtual environment (python -m venv djvenv)
2. Install django (pip install django)
3. Create django project (django-admin startproject django\_project)
4. Create portfolio app (django-admin startapp portfolio\_app)
5. Migrate (makemigrations and migrate)
6. Run the server (python manage.py runserver)

A computer screen shot of a program code

Description automatically generatedMusician Model:

Piece Model: A screen shot of a computer program

Description automatically generated

The models are setup so that each piece has a musician (one-to-many relationship). Each model also has various attributes, which are mostly just text fields. The piece model does have a mp3 file attribute, and the file, when uploaded, is stored at “music/mp3”, and in the settings.py file, the MEDIA\_URL is set to “/media” as you can see below.

A computer screen shot of a program code

Description automatically generated

VIEWS:

The detail and list veiws are fairly similar to the portfolio project. In urls.py file, you tell the program which view to render for each url pattern: A screen shot of a computer code

Description automatically generated

Each view then has instructions on what to do when called, and the list and detail views are the most simple because it just uses a built in function (generic):

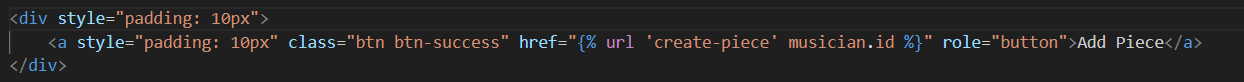
A computer screen shot of a program

Description automatically generated

The only extra part there is so that the musician detail page can also display each piece that is associated with that musician.

CREATE, EDIT, and DELETE

Throughout the page, there are buttons to create, edit and delete various models. Each button has a url that it calls, which is then associated with a view from the views.py file. For example, this button to create a piece calls the url “create-piece’ and sends the musician id to that url



Then that url calls the createPiece view



And the view takes in the musician\_id and renders the piece form for the user to input the new piece information

A computer screen shot of a program

Description automatically generated

This fucntionality is very similar for the other two methods, edit and delete.