This past week, I wrote the code for the visualizer and linked it to the audio .js file. I expanded upon one of the visualizers from the Austin Zhang collection, and edited it to work with the loaded audio instead of the mic. I used universal functions (window.setCurrentAnalyzer) to fetch the analysers from the audio tab and run the visualizers on two groups of tracks: frequency-based visualizers and rhythm-based visualizers. For either frequency or time domain, I averaged values in the data array and divided them by 255 to normalize them into a value between 0 and 1, to simplify things. I then edited the drawing function to convert polar coordinates into Cartesian positions on the canvas. I also changed the normal vertex to curveVertex, changed the frame count, smoothed the audio level for the rhythm visualizers, and slowed it down because it was a bit too chaotic and pointy for my taste. I also smoothed the visualizer in other places, added more points on the axes and added Perlin noise to make the shapes more organic. I also matched the audio level to stroke color so the visualizer changes color throughout the song based on the gain! In the audio file, I also added a function to automatically stop any song that's playing when I start a new one. This upcoming week, since the visualizer is working pretty much the way I want it to, I'll work on the html and visual aspect of the website, add links to the music on streaming platforms, and write a small blurb explaining the project (EP and visualizer) and give a credit to Austin Zhang.