Programming assignment - Python Backend Developer

Your task is to create a music management/sharing/streaming platform. A user should be able to upload/download/play/delete/search songs. All songs go into a common playlist (which shows up when the user wants to view all songs).

Detailed requirements:

* Upload a song
  + Has metadata fields title, artist, album
  + Song should have a unique URL which can be shared with anyone publically.
* Delete an uploaded song
* View all songs
* Search for any song via album/title/artist
* Stream song
  + Each song should have a page where you can play the song from the browser itself. Use HTML5 for this, not flash.
  + Clicking on the unique URL of a song should lead to this page.
  + Clicking on a song entity from either search or view all pages should also lead to this page.
  + There should be an option to download this song.

Important things to keep in mind

* Allow only mp3 files for simplicity.
* Focus on functionality and ease of use, not on the fanciness of the interface.
  + Also on backend scalability if possible.
* UX should be easy and intuitive.
* You do not need to use any complicated CSS (or frameworks) for this. Simple HTML buttons/links/tables should be enough. You can if you want to and have free time, but will not be graded for look and feel.
  + You will be graded better for a better user experience, keeping aside the look and feel.
* **Keep in mind the security of your application.**
  + Keep your code as secure as possible.
  + Identify (if not fix) points in your code which are vulnerable to security attacks.

Technology to use

* Python3/Flask for the backend.
* SQL for database. You may use SQLite.
* **No framework for frontend**
  + Simple, separate HTML pages using Jinja is enough. Do not write any complicated javascript code, this website doesn’t require it. Some simple auto fill / onclick / other event triggers are okay.
  + No need to optimise javascript and/or DOM loading.
* **Keep CSS separate if you have any**
  + This is very important. I will not be evaluating the CSS files. Keep them separate so that I can read the rest of the code properly.
* **Keep the uploading functionality separate**
  + You may use 3rd party libraries if you want. I will not consider this while scoring this as long as it works.
  + You can store the code on your local filesystem, or on cloud somewhere, for example
* **Do not use Apache or NGINx or any webserver.**
  + You should run your app using *flask run app* or a similar command.
  + **If** your application needs any extra commands to be run while initializing, add a shell script called **run.sh** which does that.
    - The shell script should run on Ubuntu 18.04 LTS.
    - *You may ship a Python script (****run.py)*** *instead of a shell script if you want to make it cross platform. This will not fetch you extra marks.*
  + You **may** have an **install.sh** which installs the app.
    - *You may ship a Python script (****install.py)*** *if you want to make it cross platform. This will not fetch you extra marks.*

Sending over the code.

* Create a public GitHub repository and share the link with us.
  + Commit a readme with instructions to deploy/run the code and basic documentation about the code if required.
  + Commit your code regularly so that I can see the workflow, with sensible commit messages.
    - **A single commit of the entire code at the end will be rejected.**
      * Even if you have just two sensible commits I will consider it. ***One extra fake commit just to satisfy this criteria does not count.***
* **Give proper instructions on how to make this app run on my machine.**
  + You can make the following assumptions
    - I know how to run a **pip install -r requirements.txt** and run a flask app
    - I use Ubuntu 18.04 LTS.
  + Any other instructions should be copy-pasteable from your readme, and work on Ubuntu 18.04 LTS.
  + Keep the instructions as simple and concise as possible.