Project Outline

Track: Data Engineering
Dataset: Video Games

We want to engineer a database of thousands of video games, and use that database to find hidden gems of data. We want to show the following: most popular video games by rating, least popular by rating, average rating of each gaming platform/console, and (anything else? Most popular genre etc?).

1. Data must be stored in a SQL or NoSQL database (PostgreSQL, MongoDB, SQLite, etc) and the database must include at least two tables (SQL) or collections (NoSQL). PostgreSQL or SQLite?
2. The database must contain at least 100 records. Done (12k records)
3. Your project must use ETL workflows to ingest data into the database (i.e. the data should not be exactly the same as the original source; it should have been transformed in some way). Remove duplicates, remove null values, etc
4. Your project must include a method for reading data from the database and displaying it for future use, such as:
    a. Pandas DataFrame
    b. Flask API with JSON output Pandas DF's and should do the trick
5. Your project must use one additional library not covered in class related to data engineering. Consider libraries for data streaming, cloud, data pipelines, or data validation. Spark(PySpark)?
6. Your GitHub repo must include a README.md with an outline of the project including:
    - An overview of the project and its purpose, instructions on how to use and interact with the project; Documentation of the database used and why (e.g. benefits of SQL or NoSQL for this project); ETL workflow with diagrams or ERD; At least one paragraph summarizing efforts for ethical considerations made in the project; References for the data source(s); and References for any code used that is not your own.
7. OPTIONAL: add user-driven interaction, either before or after the ETL process. e.g.:

a. BEFORE: provide a menu of options for the user to narrow the range of data being extracted from a data source (e.g. API or CSV file, where fields are known in advance).

b. AFTER: Once the data is stored in the database, add user capability to extract filtered data from the database prior to loading it in a Pandas DataFrame or a JSON output from a Flask API. ??????