CS374 – Intro to Database Management

Application Development Project

Rubric for First Deliverable

Group Member #1: __Jacob Janzen__

Group Member #2: __Noah Frost____

Group Member #3: __Ethan Nicolay__

Name	Requirements	Points	Awarded
Description of Application	• An overview of your application • System requirements (e.g. hardware, DBMS, other software) • A detailed description of your application	25	

Music Streaming DBMS

Application Overview:

We plan to build a music streaming database management system. This will involve a backend written in a combination of Python and C++, a front-end web application that will serve as a UI, and will connect to a RDBMS, which will likely be MySQL. The backend program will connect to the RDBMS through MySQL connector and the front end will be connected through exposed APIs.

System Requirements:

The described application is a full stack program, meaning it requires a front end, backend, and in this case a connected DBMS. The application will require user authentication software, and a login that limits the number of attempts to prevent brute force attacks. The application will also rely on a web hosting service, which will likely be Google Cloud.

Specific Application Description:

Front End:

- Will be created using custom HTML, CSS, and JS
    - Could also look at using WordPress if time constraint becomes an issue down the line.
- Will allow users to view their personal library songs, playlists, etc.
- Will allow users to sort their playlists in a variety of ways, available through a simple drop-down menu next to the playlist title.
- Could possibly create a generative algorithm to suggest music to users adjacent to their established preferences.
- Will connect to the web using a hosting service.
    - Will most likely use Google Cloud as it is free for us as students.

Back End:

- Will be mainly written in Python.
    - If we run into performance issues, we can use embedded C++.
- Will use MySQL Connector to connect the Python program to our DBMS.
- Will use Rest APIs to connect to the UI and query the web if we choose to add the predictive algorithm described above.
- Will implement consistent and frequent unit testing to ensure program works for all edge cases.
- Readable and abundant commenting will allow for future development and understanding of the backend program.

DBMS:

- Will use MySQL to create a relational database that includes several related tables.
    - Users and all their required information.
    - Playlists
        - Playlist Table (playlist_id, user_id(foreign key), name, date_created)
        - Songs Table (song_id, title, artist, album, genre, etc.)
        - Relationship Table (playlist_id(foreign key), song_id(foreign key))
- There will be a plethora of tables and complex relationships in our DBMS that are not explicitly mentioned here.
    - We are creating an EER diagram that will give detailed notations for every relationship that exists between tables.
    - We are also creating a UML that will depict the database schema at a high level.