CS 5200 PROJECT REPORT

Team Name: RiouxSGhumatkarP
Group Members: Seamus Rioux, Prajakta Ghumatkar
Presentation: https://youtu.be/JRC-mtZmwCg

BEATSTREAM: MUSIC STREAMING MANAGEMENT SYSTEM

README

Functionalities Available :

1. Register new user
2. Login and Logout for existing users
3. Create Playlist
4. Add songs to playlist
5. Delete songs from playlist
6. View information about songs
7. View profile information
8. Select Various payment plans

Prerequisites:

1. Python version 3.x (Link to download python--> https://www.python.org/downloads/)
2. MySQL Community Server 8.0 and above (Link to download mysql community server--> https://dev.mysql.com/downloads/mysql/)
3. Operating System: Windows 10/8/7, Linux, MacOS
4. Minimum 128 MB Ram
5. Minimum 10 MB of free storage
6. All necessary packages & modules mentioned above under the "Python Libraries to install" section should be installed on the system before hosting this program

Import Database Dump into MySQL Workbench

1. Open StreamingDatabaseDump.sql file in the root directory. Run these commands

Create new virtual environment variable

1. Ensure Python is installed on your machine and it is able to run through the command prompt or terminal. Check the installation using:
2. Windows Command Prompt:
   python --version
   Unix:
   python3 --version
3. Navigate into project directory
   Windows Command Prompt:
   cd RiouxSGhumatkarPProject\Streaming-Database\FlaskApp

   Unix:
   cd RiouxSGhumatkarPProject/Streaming-Database/FlaskApp

4. Run the following command *:*
   Windows Command Prompt:
   python -m venv <name_of_virtualenv>

   Unix:
   python3 -m venv <name_of_virtualenv>

5. Activate virtual environment
   Windows:
   <name_of_virtualenv> \Scripts\activate

   Unix:
   source <name_of_virtualenv> /bin/activate

Python libraries to install

1. Install pymysql - *pip install pymysql*
2. Install cryptography - *pip install cryptography*
3. Install flask - *pip install flask*
4. Install flask-pymysql - *pip install flask-pymysql*

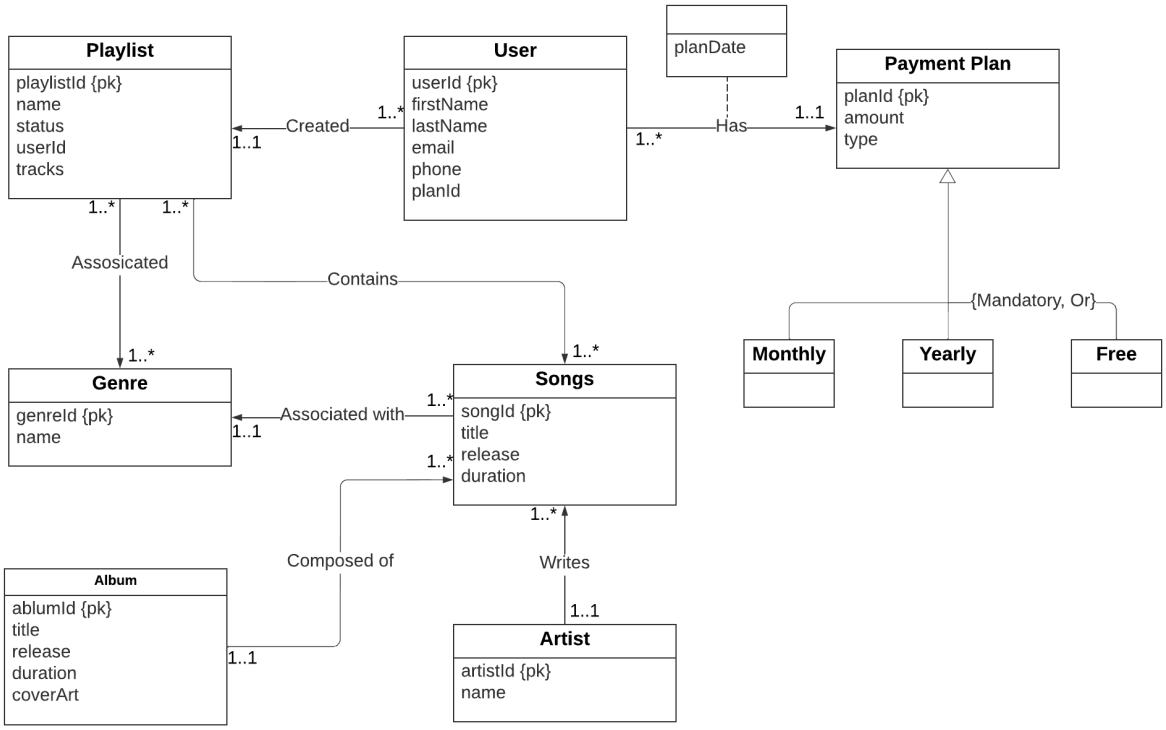
Run Application

1. Open db_config.py in the FlaskApp folder and change the DB_SERVER, DB_USER and DB_PASS according to your mysql credentials
2. Navigate to the FlaskApp folder in the terminal and run the following command :
   python app.py
3. Copy the link in the terminal and paste in a browser
4. Interact with the application

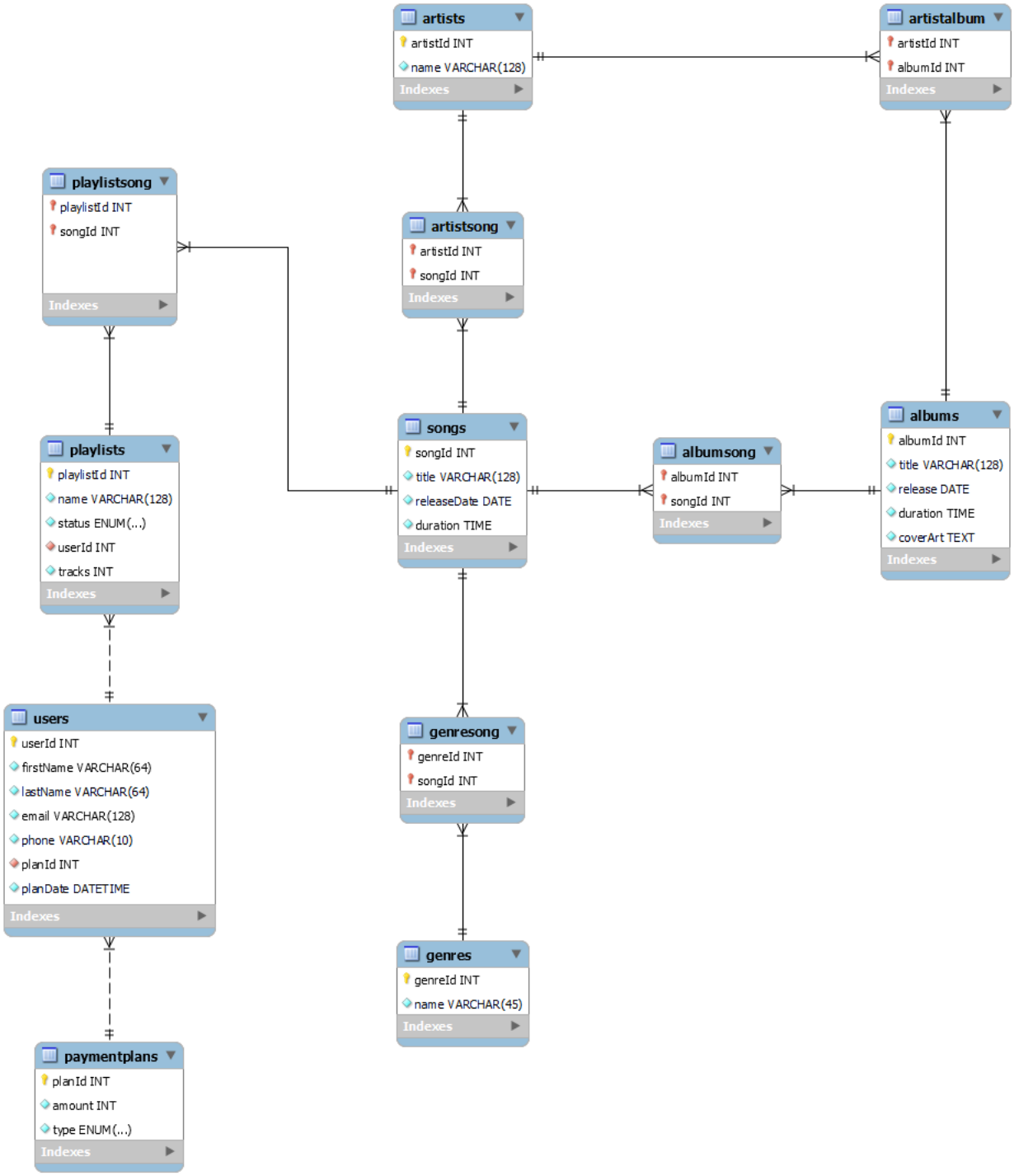
TECHNICAL SPECIFICATIONS

Software: MySQL Workbench we will be using SQL
Language: HTML, CSS and Javascript for developing the web pages. We have used the Flask framework for creating the interface. We have used Python in the backend. The main database application code is stored within the app.py file. Calls to the database are made using PyMySQL execute and callproc statements.
Flask: Flask will handle get and post requests from HTML files. When a user interacts with the application the request is passed to the flask function corresponding to the current page and decides what to do from there. These operations usually result in calling a SQL command and then rendering the current page or a new page to reflect the changes made to the database.
Templates: Flask uses html files rendered in the main python application. These html templates can be viewed in the template folder.
Data: Data for all the schemas in the database is stored using sql files for reference all information as well as procedures, functions and triggers will be composed in the database dump. Procedures and Functions are able to be run individually in their respective sql files.

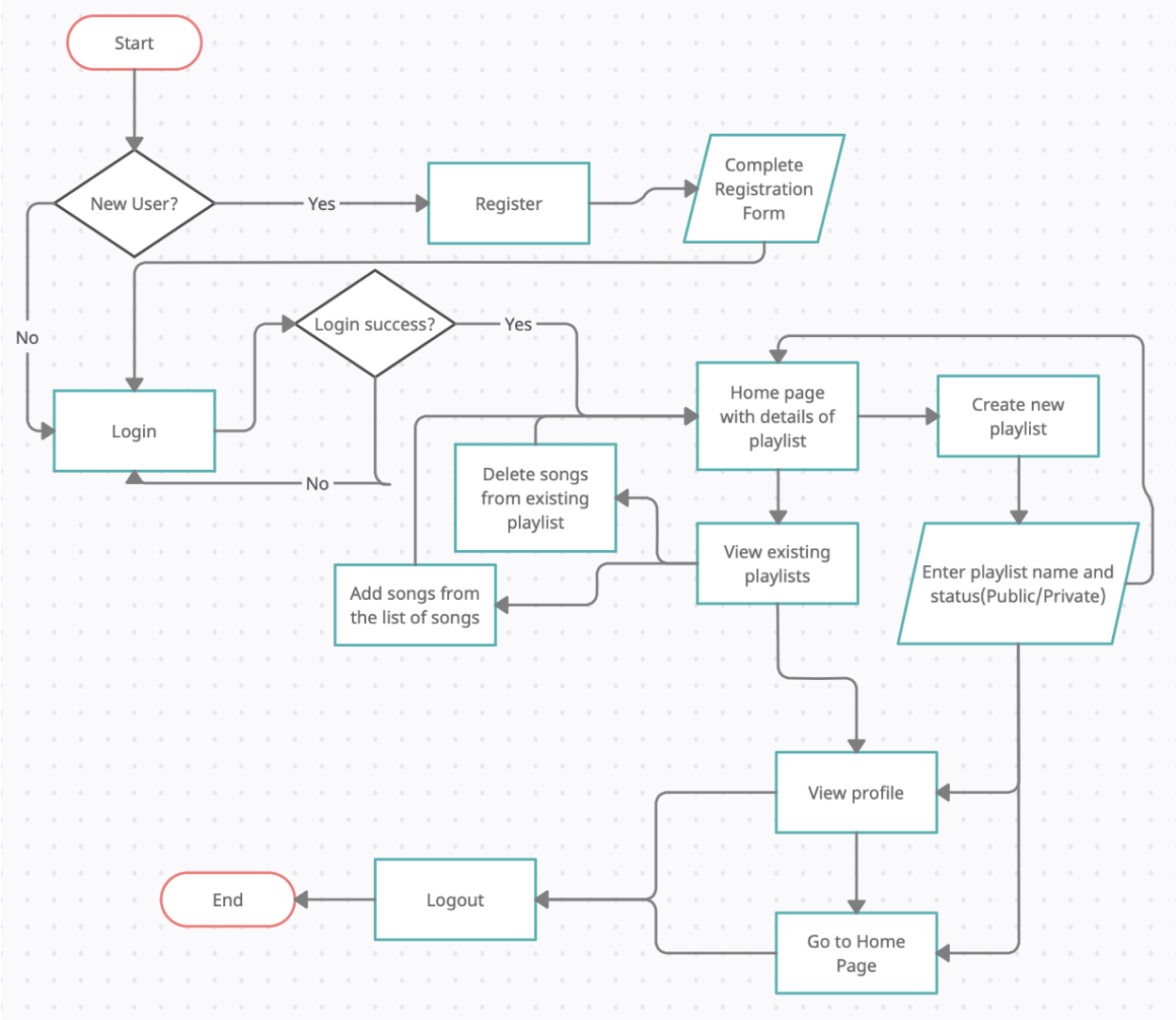
CONCEPTUAL DESIGN - UML



LOGICAL DESIGN (REVERSE ENGINEERED)



USER FLOW OF THE SYSTEM



LESSONS LEARNED

Technical Expertise gained

In this project we explored using Flask as the front end. It allowed us to integrate HTML code directly with python code. We also learned more about designing basic HTML pages and using CSS styles from Flask. This was not our first time using PyMySql, however we explored its uses in depth with its integration with procedures and functions. We also felt that we have learned more about SQL commands in procedures and functions since this was our first time designing our own architecture and integrating it with our own tables.

Time management insights and data domain insights

Designing a database from scratch and creating a domain can be a difficult task. Our topic revolved around an existing system, streaming platforms we studied existing applications and explored how the integration between songs, albums and playlists. We needed to include all the necessary information about songs, albums and playlists, however we focused mainly on the UI aspect of the project and felt that we put this before the complexity of the database. Having to redesign the schema several times wasted some necessary development time. We designed the application before finalizing the schema and felt that this should be less of the focus than the information and elements of the data.

Realized or contemplated alternative design

In our project we mainly stuck to the proposed schema and idea of the project. We kept the features of the streaming platform which were albums, songs, playlists and genres. We allowed a user to create an account, select payment plans and create playlists. We originally planned on using Java as the backend and Angular on the front end but due to time constraints we decided to use Python which we were both familiar with and decided to learn Flask instead. Our original plan was to also have song recommendations and use our knowledge and interest in machine learning but due to wanting a finalized schema, quality UI and time constraints we decided against this idea.

FUTURE WORK

Planned uses of the database

The idea of this application was to understand how streaming platforms worked and also learn how to create a database from scratch. We wanted to demonstrate our machine learning knowledge to create song recommendations based on genres, artists and more feature information about songs. This application could also be used to create mock playlists without having to buy a streaming platform subscription and eventually once song recommendations were added this could help show more useful recommendations with an algorithm we built. Most of the functionality we proposed is existing in the application; we will just need to add functionality for song recommendations.

Potential areas for added functionality

The main added functionality is song recommendation using existent fields in the database and adding feature information for each song and album. We could also factor genre information into our recommendation algorithm. The current database can be easily modified to support song recommendation, however we would need to add a page to support this potentially an existing table at the bottom of the song page which would list all the recommendations for the current playlist. Another feature we would have liked to add is multiple genre support for each song which would help with the song recommendation feature. Adding broad genre support would be a useful addition to our application. We also wanted to add a way to filter songs by the album. We would list albums within our song table and differentiate between them using a type variable. When you clicked on an album you could filter the songs within that album and see more information about it. We would also like to add featured artists to the songs table as well. This would allow for multiple artists to be connected to a song if there is a feature but for the scope of this project we chose to only have one artist per song. Instead of adding the song recommendation we chose to focus on the important aspects of the project such as database design and the UI but in the future we are planning to add a machine learning algorithm to support song recommendations.

Extra Credit Attempted

We attempted to create a complex schema with multiple joins and references between each table. We also created a GUI for our database using Flask to allow users to properly interact with our database.

References/Technology Resources

Technology Used

Flask: https://flask.palletsprojects.com/en/2.2.x/
Pymysql: https://pymysql.readthedocs.io/en/latest/
Flask-Pymysql: https://github.com/rcbensley/flask-pymysql
Font-Awesome: https://fontawesome.com/
Python: https://www.python.org/downloads
MySQL: https://www.mysql.com/
Python Virtual Environment: https://docs.python.org/3/library/venv.html

Album and Songs

https://en.wikipedia.org/wiki/Kids_See_Ghosts_(album)
https://en.wikipedia.org/wiki/Ye_(album)
https://en.wikipedia.org/wiki/Post_(Bj%C3%B6rk_album)
https://en.wikipedia.org/wiki/It%27s_Almost_Dry
https://en.wikipedia.org/wiki/Harry%27s_House