CS 165: Phase 5

Updated Documentation, Technical Specification, How-to-use Guide

Beltran Database of Selected Movies

by Sean Ryan Chan, Katrina Kopio, John Matthew Villaluz

Technical Specifications

The Beltran Database of Selected Movies application is designed to run locally on a computer with the said computer's command line interface as the server whereas the GUI can be accessed through a web browser. The BDSM app has been tested on the following specifications:

1. Google Chrome - Stable version, 70.0.3538.110
    a. (Mozilla Firefox should also work)
2. Python - 3.6.5
    a. (any version of Python 3 should work however)
3. sqlite3 Python module - 2.6.0
4. Flask Python module - 1.0.2

Setting Up

1. **Open** your terminal (or command-line interface of choice which has Python 3 installed), then **navigate** to the local directory of our app which should look like this:
```
a. Bootstrap
b. HUV_Kopio_Katrina_phase3.sql
c. __pycache__
d. database.db
e. sqldiff
f. static
g. Flask-0.12.2-py3.6.egg-info     README.md
h. app.py
i. flask
j. sqlite3
k. templates
l. Flask_Cors-3.0.3-py3.6.egg-info
m. Werkzeug-0.12.2-py3.6.egg-info
n. data.py
o. flask_cors
```

    p. `sqlite3_analyzer`
    q. `werkzeug`

2. **Run** the following code in your terminal: `python app.py`
3. If you get the following output in your terminal:

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 231-054-904
```

4. The application is now live and working! Please open Google Chrome and type in this in the address bar: `localhost:5000/`
5. You should see BDSM, our home page.

Project Description

This project is built from Katrina Kopio’s original proposal for a movie application. The movie database is structured based on the movie industry today, with entities representing actors and directors, movie soundtrack, songs, and reviews from professionals. This database also allows the user to add their own personal reviews.

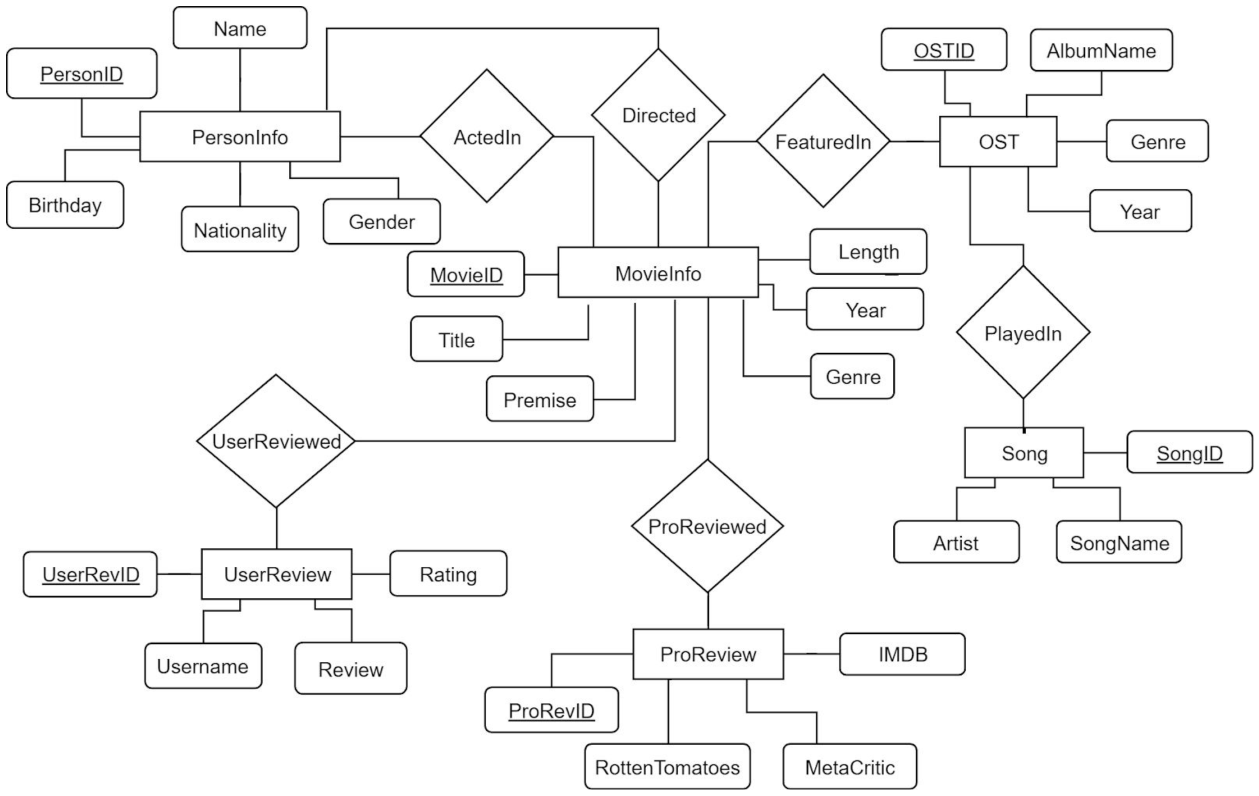
The objective of this project is to allow users to take note and rate their own movies but it’ll be a more personalized list. It also acts as an offline database for movies. This database is for personal use which is why it does not support several users, or logging in and out.

The audience of our proposed application are movie junkies, cinephiles, and etc.

Schema

```
MovieInfo (MovieID integer, Title varchar (20), Premise varchar
(300), Genre varchar (20), Year integer, Length integer)

PersonInfo (PersonID integer, Name varchar (20), Birthday varchar
(20), Nationality varchar (20), Gender varchar (10))

OST (OSTID integer, AlbumName varchar (20), Genre varchar (20),
Year varchar (5))

Song (SongID integer, SongName varchar (20), Artist varchar (20))

UserReview (UserRevID integer, UserName varchar (20), Review
varchar (300) Rating varchar (4))

ProReview (ProRevID integer, RottenTomatoes varchar (10),
MetaCritic varchar (10), IMDB varchar (10))

ActedIn (MovieID integer, PersonID integer)

Directed (MovieID integer, PersonID integer)

FeaturedIn (MovieID integer, OSTID integer)

PlayedIn (OSTID integer, SongID integer)

ProReviewed (ProRevID integer, MovieID integer)

UserReviewed (UserRevID integer, MovieID integer)
```

Entity-Relationship (E-R) Model Diagram



App Functionalities (CRUD)

*REVISED FROM PHASE 4, SQL Statements are found in the next section.

1. **CREATE -** Users will be able to create entries in the following tables listed in the schema:
    a. MovieInfo
    b. PlayedIn
    c. Directed
2. **READ -** Users will be able to read/view all information from all the tables said above. It will be displayed in this format below:

Movie Info	Title Premise Genre Year released Length in minutes
Actor/Director Info	Name Birthday Nationality Gender *and the information on the movie and the associated actor/director.
OST	Album name Year released Genre of music *and information on its associated movie.
Song	Song name Artist *and information on its associated OST

3. **UPDATE -** Users will be able to update/edit entries in the following tables (from the schema):
    a. MovieInfo
    b. Song
    c. PersonInfo
4. **DELETE -** Users will be able to delete entries in the following tables (from the schema):
    a. MovieInfo
    b. PlayedIn

c. Directed

Implemented SQL Statements

The following list of SQL Statements are implemented in the application provided to you. Note that this has been changed since the previous documentation.

CREATE Statements

```
insert into PlayedIn values (<OSTID>,<SongID>);

insert into MovieInfo values  (<title>, <premise>, <genre>, <year>,
<length>);

insert into Directed values (<MovieID>,<PersonID>);
```

READ Statements

```
select * from OST where OSTID=<givenOSTID>;

select * from Song where SongID=<givenSongID>;

select * from PlayedIn, OST o where p.OSTID=o.OSTID and
p.SongID=<givenSongID>;

select * from FeaturedIn f, MovieInfo m where f.MovieID=m.MovieID
and f.OSTID=<givenOSTID>;

select * from MovieInfo;

select * from MovieInfo where MovieID=<givenMovieID>;

select * from PersonInfo;

select * from PersonInfo where PersonID=<givenPersonID>;
select * from Directed d, MovieInfo m where d.MovieID=m.MovieID and
d.PersonID=<givenPersonID>;
```

UPDATE Statements

```
update Song
	set
	SongName=<newSongName>,
	Artist=<newArtist>
	where
	SongID=<modifiedSongID>;

update MovieInfo
	set
	Title=<newTitle>,
	Premise=<newPremise>,
	Genre=<newGenre>,
	Year=<newYear>,
	Length=<newLength>
	where
	MovieID=<modifiedMovie>;

update PersonInfo
	set
	name=<newName>,
	gender=<newGender>
	where
	personid=<modifiedPersonInfo>;
```

DELETE Statements

```
delete from PlayedIn
	where ostID=<givenOSTID> and songID=<givenSongID>;
```

```
delete from Directed
      where PersonID=<deletedPersonInfo> and
      MovieID=<deletedMovieID>;

delete from MovieInfo
      where MovieID=<deletedMovieID>;
```

Video Demos / Walkthroughs

This Google Drive link contains the videos of our team members' giving a guide and demo on their implemented CRUD functionalities.
https://drive.google.com/drive/folders/1V70bg0WD7EzJzIhWeXxsHf1asymq46Pb?usp=sharing

Katrina Kopio's video segment may not be viewed properly, but OneDrive's online viewer will let us do so:
https://onedrive.live.com/?authkey=%21AJdNqKIQdcozuT0&cid=AEE14EE4FD18BE77&id=AEE14EE4FD18BE77%2170147&parId=AEE14EE4FD18BE77%2170131&o=OneUp

or on YouTube:
https://www.youtube.com/watch?v=c9glrPGtoG4&feature=youtu.be&fbclid=IwAR1_h6siWIboE-ZVY4St8rTupVQcM_u7CVw3C44u0_8HiaEwUoJZFH600zQ
https://youtu.be/c9glrPGtoG4