CS166 Project Phase 3: Implementation

May 12, 2020

1 Description

Your final tasks for this project will be:

- Develop a client application using the Java Database Connector (jdbc) for psql.
- Use the client application to support specific functionality and queries for your online booking system.

In this phase, we will provide you with a create.sql that recreates the relational schema of phase 2. You will use this schema to test and demo your application to us. Additionally, we will give you a collection of .csv files containing dummy data that are compatible with the provided relational schema. You will have to create your own .sql scripts to insert the data from the given .csv files into the database.

Finally, we will give you a skeleton code for the client application. The code will be in Java and will contain some basic functionality that will help you to communicate with the database and issue various SQL statements. You will have to implement your own code for a certain number of functions, described in more detail in the next section.

For projects that include a user interface on top of a simple command line interface(e.g. web/mobile application), we will award up to 10% extra credit.

The submission for this phase must include all provided and modified source code as well as a short video demonstrating each query all zipped into one compressed folder. Each group member must participate evenly in the video creation. You can use any screen recording application, or even enter a private zoom meeting and record the meeting after activating screen-share. Please keep this video short(around 5 minutes). We don't need a full explanation of your code - just a video showing it running on each query. If your video is too large, even compressed, to upload to iLearn, please upload the video separately to a shareable media(e.g. Google Drive) and provide a link in your submission notes on iLearn. Be sure to test that the link can be viewed by anyone who has it!

This phase is due on **Tuesday, June 9, 2020, at 11:59pm**

2 Functions

1. Add User
    - Add a new user to the database. You should provide an interface that takes as input the information of a new user (i.e. first name, last name, email, phone) and checks if the provided information are valid based on the constraints of the database schema.
2. Add Booking
    - Add a new booking into the database. You should provide an interface that takes as input the information of a new booking (i.e. status, DateTime) and checks if the provided information is valid based on the constraints of the database schema. In order to add a new booking, you must draw information from an existing and valid user, show, movie, seating, theater, and cinema.
3. Add Movie Showing for an Existing Theater
    - Add a showing of a new movie using the Shows, Plays, and Movie tables for a given theater. You should provide an interface that takes as input the information of a new movie (i.e. title, duration) and show(i.e. start time) and checks if the provided information is valid based on the constraints of the database schema. Note: The order matters for this query. What happens when you try inserting in the wrong order? Think about why that happens and you'll know what the correct order should be.
4. Cancel Pending Bookings
    - Cancel all bookings that have a status of pending. This function should find all bookings that have a pending status and set their status to cancelled. Use case: this could be automatically executed every 15 minutes(the amount of time checkout could last) or once a day, so the system does not endlessly reserve seats that are not paid for.
5. Change Seats Reserved for a Booking
    - Replace the seats reserved for a given booking with different seats in the same theater. For example, a user changes their mind about where they want to sit. They have already booked seats 10 and 11 but would like to move back one row to seats 20 and 21. This should only work if the new seats are available and they are the same price.
6. Remove a Payment
    - A user cancelled their plans to see a show, so we must refund their payment and change their booking status to cancelled. Assume the refund has already been handled by the payment processor(e.g Square), so we should just delete the payment from our database. Cancelling the booking is similar with function 4 with different restrictions(e.g. update status based on Booking ID instead of status).
7. Clear Cancelled Bookings
    - Remove all Bookings with status of cancelled. Use case: this could be automatically executed once a day/week/month to clear unnecessary bookings and save on storage space.

8. Remove Shows on a Given Date

    - Remove all Shows on a given date at a specific Cinema. If there are any bookings on this day, they can be cancelled using the "Remove a Payment" method implemented above. Use case: the cinema is closed for a holiday or some unforeseen circumstance.

9. List all Theaters in a Cinema Playing a Given Show

10. List all Shows that Start at a Given Time and Date

11. List Movie Titles Containing "love" Released After 2010

12. List the First Name, Last Name, and Email of Users with a Pending Booking

    - List user information for users with a booking that has a status of pending. We do not want to allow users to make a new booking without finishing paying for another one. Assume we have a way to prevent this if we have the list of users.

13. List the Title, Duration, Date, and Time of Shows Playing a Given Movie at a Given Cinema During a Date Range

    - List the Movie Title, Movie Duration, Show Date, and Show Start Time of all Shows playing a given Movie at a given Cinema within a date range. This would be useful for users to find the best time for their schedule to book a showing of a movie at a given cinema.

14. List the Movie Title, Show Date & Start Time, Theater Name, and Cinema Seat Number for all Bookings of a Given User

    - List information related to a user's bookings so they can see the time and location for all shows they have booked.