Daniel R. Cender

, Grand Canyon University

CST-361: Design Patterns in Java

Charbel Elkhoury

April 4, 2021

**Write-Up**

**Patterns Questions**

* + - What is the Data Access Pattern?

The Data Access Pattern is a strategy of offloading the work of updating and querying a data store to a dedicated layer in an application. It isolates all persistence logic to one place in the app, which results in less code for teams to maintain over the evolution of a project.

* + - How did the use of Generics benefit the design we implemented as part of this activity?

The activity instructions made no mention of Generics to be used here. I’m confused exactly where Generic elements were used in this milestone.

* + - Why is it important to support Technical Encapsulation in the design of a Data Access Object?

The objects calling on the DAO should not have any knowledge of how the data is being fetched or where they need to look. Those technical aspects are handled and encapsulated neatly inside the Data Access layer of an application.

* + - Explain how Custom Checked Exceptions and Custom Unchecked Exceptions where used in this activity.

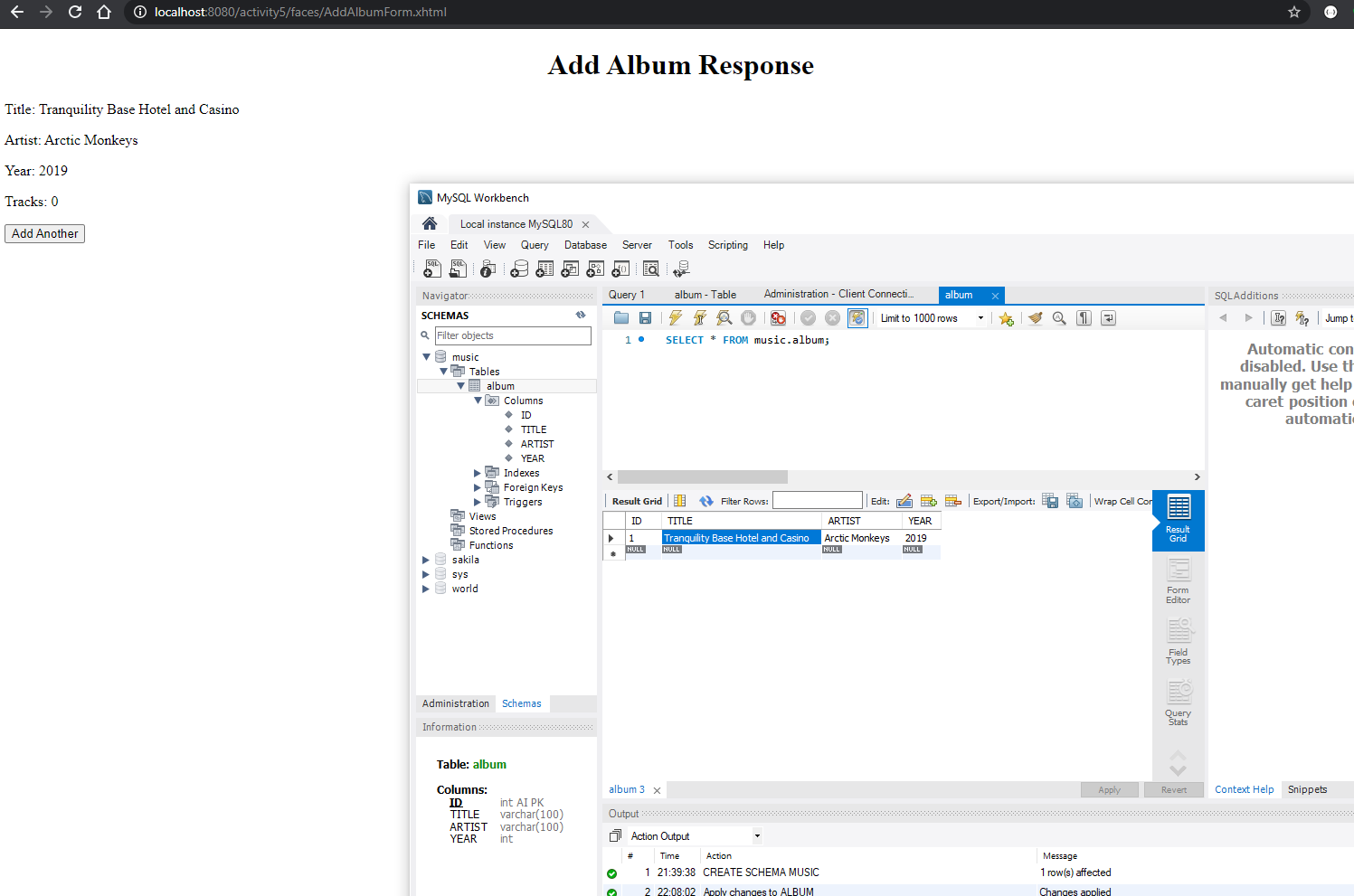
We used multiple custom exceptions to draw close attention to particular issues that might arise in the application during use. The custom exceptions tied into a custom exception handler factory, which processed them in such as way as to give the users and developers a better “error report” in the UI layer than JavaEE gives by default.

* + - Where besides in web application could you implement the MVC Design Pattern? Describe your application and solution to supporting the MVC Design Pattern.

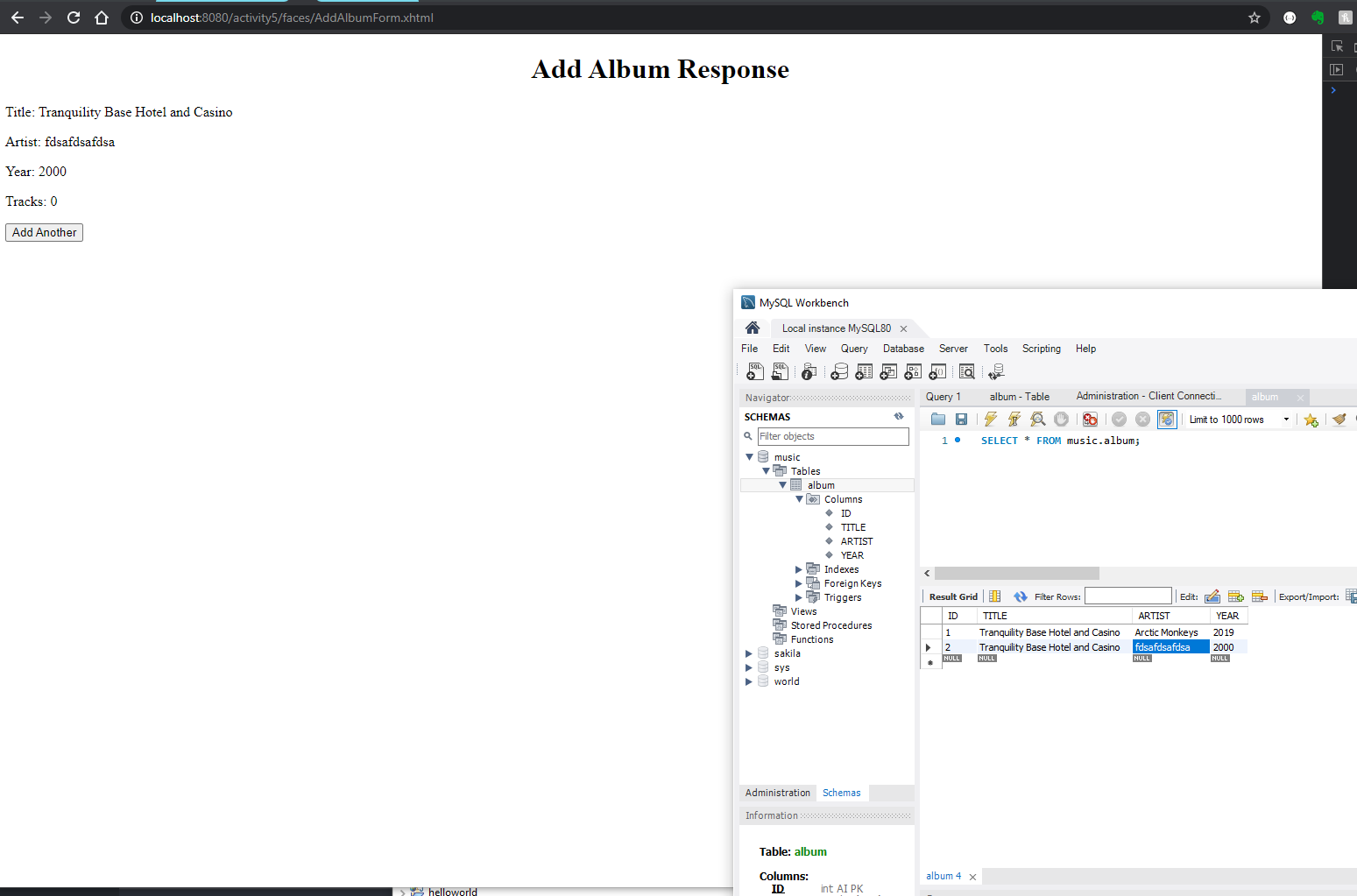
Another application type that could make use is some computer games. There is a view, usually drawn by a rendering engine. There are data models and necessities for data storage, either locally or in a remote database somewhere. The MVC model could be well upheld by containing all view and logic updates to a few, or one, central controller class. Games are often written anyways by creating controlling scripts or classes which contain event-bound methods for responding to user-view interactions.

**Deliverables**

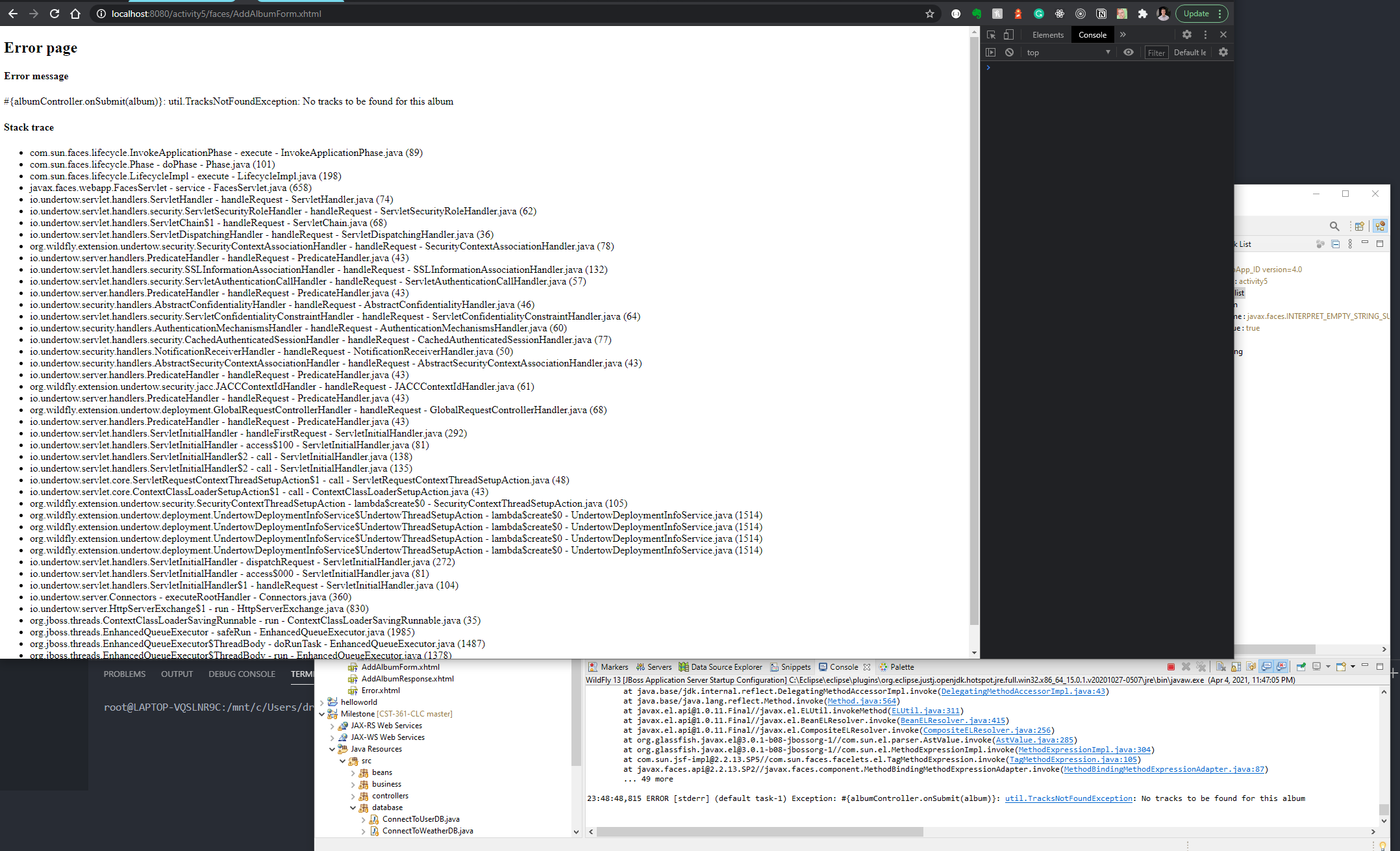
Result of adding a new album:



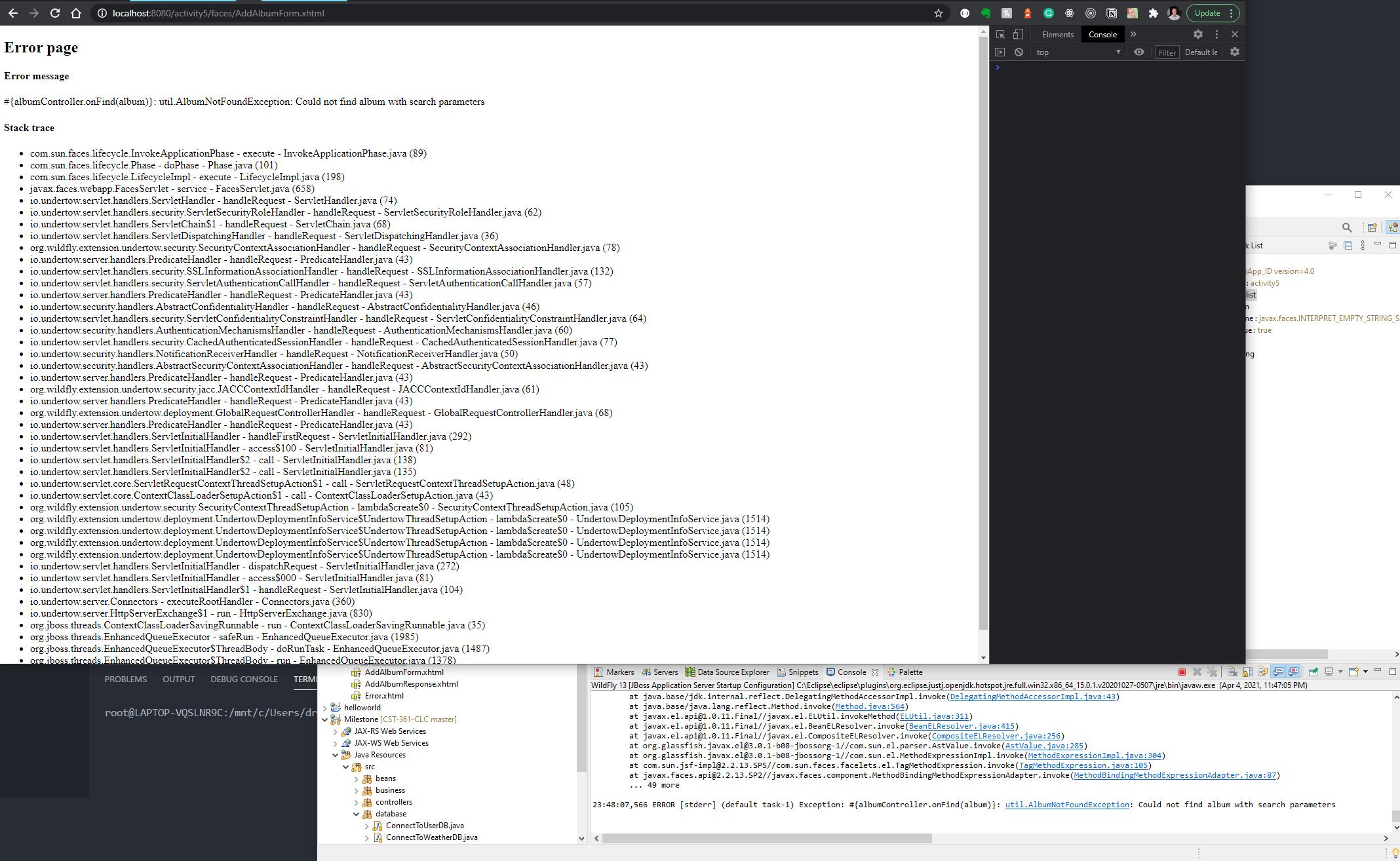
AddAlbumReponse when attempting to find an album in the database:



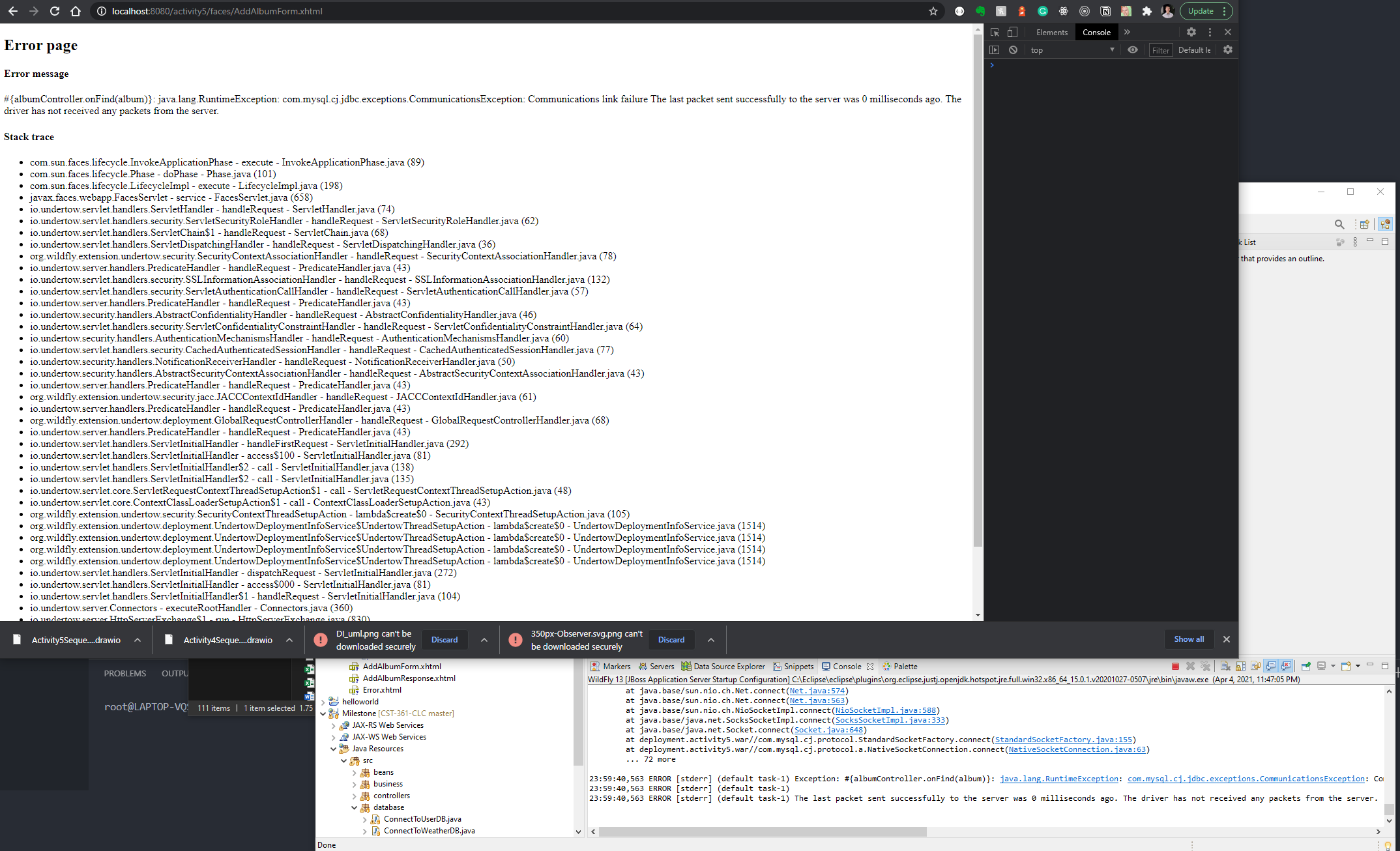
No Album found Exception, when trying to find an existing album:



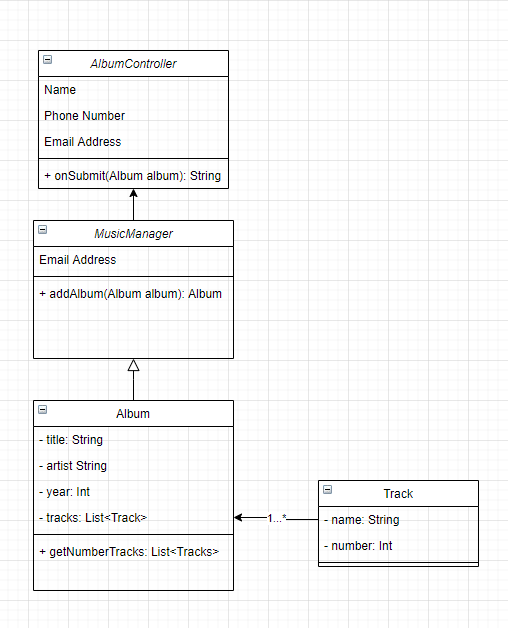
No Tracks Found Exception, when trying to create new album:



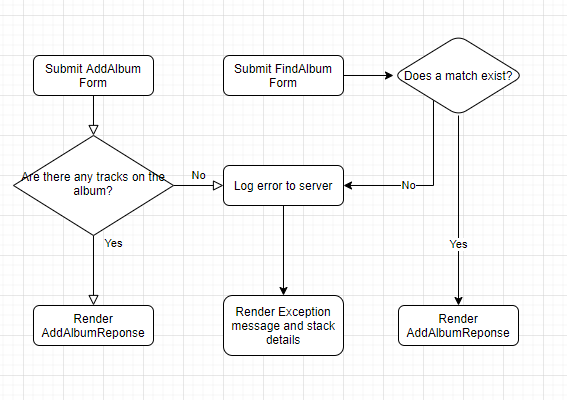
Database Exception, after MySQL server was stopped:



UML Class Diagram:



UML Sequence Diagram:



**References**