Assignment 1

Overview

The goal of assignment 1 is to create a graphical user interface that will allow users to read information from a MySQL database and display it in a TableView and as chart data. The information should be ***anything you are interested in***. For example, it could be comparing aspects of video games, weather data, processor capabilities, etc.... Each student will need to register a unique data set prior to building their program on Java - Assignment 1 Ideas.xlsx. Do not start building your idea until it is approved.

The MySQL database should be remotely accessible on the AWS platform. Your program must be built using Intellij and stored in a PRIVATE GitHub repository.

When the application is launched, it should show a graph of information on a styled JavaFX application.

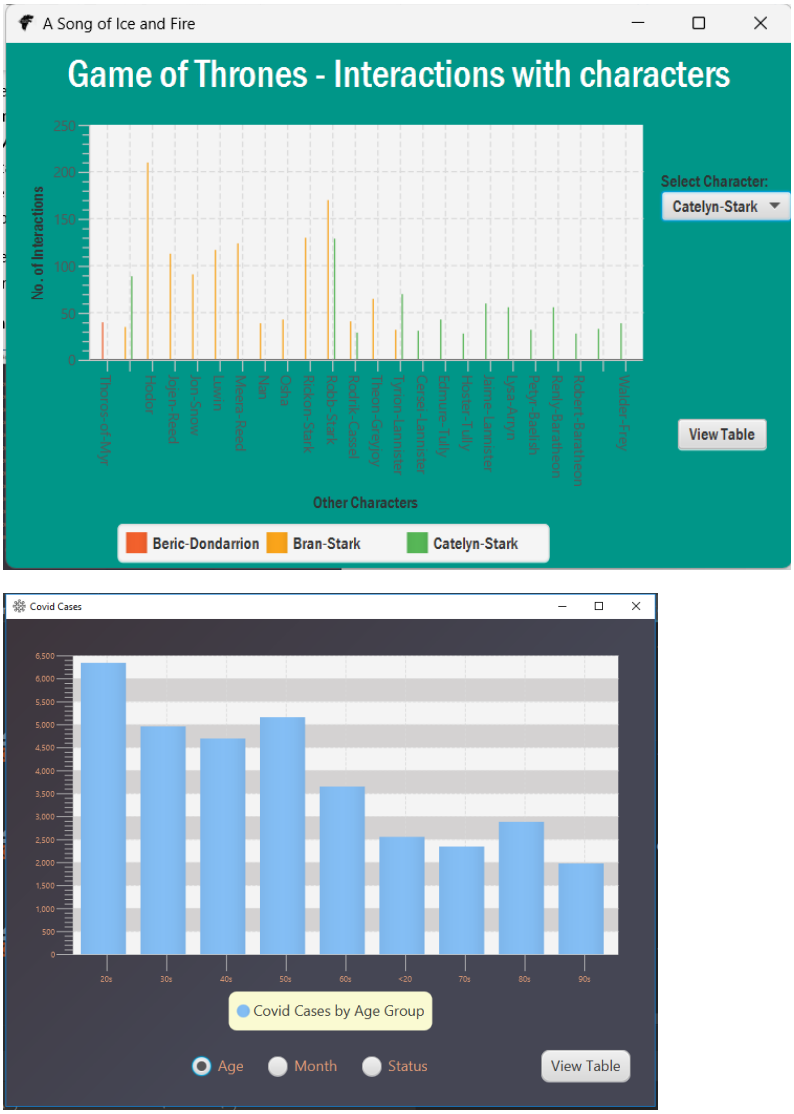
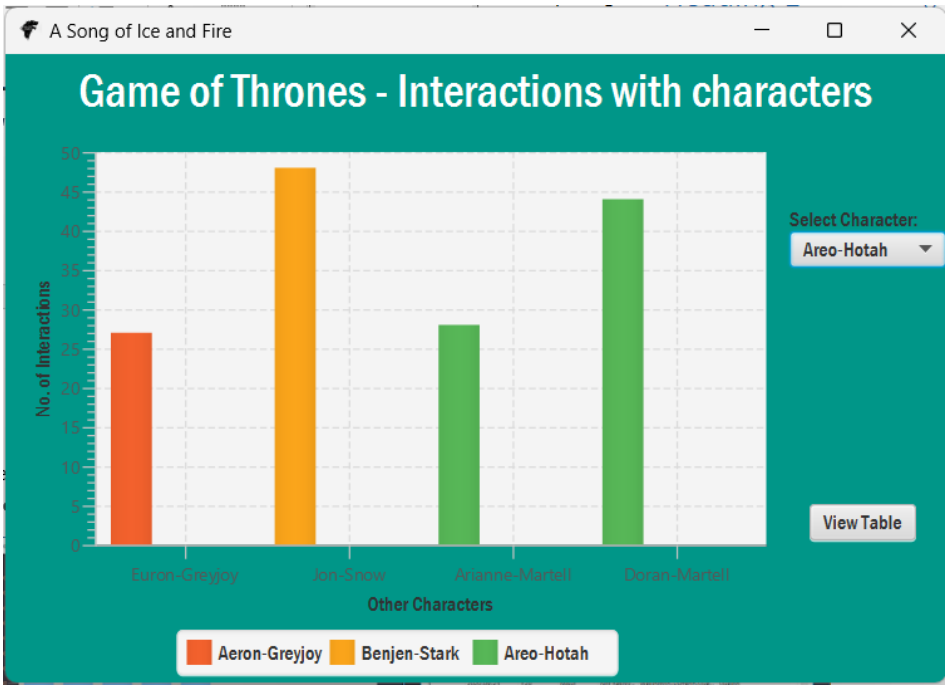


Figure 1 - Initial launch of project shows a graph

The application must support at least 1 graph and have the ability to change scenes to a TableView object that displays all the data from the database.



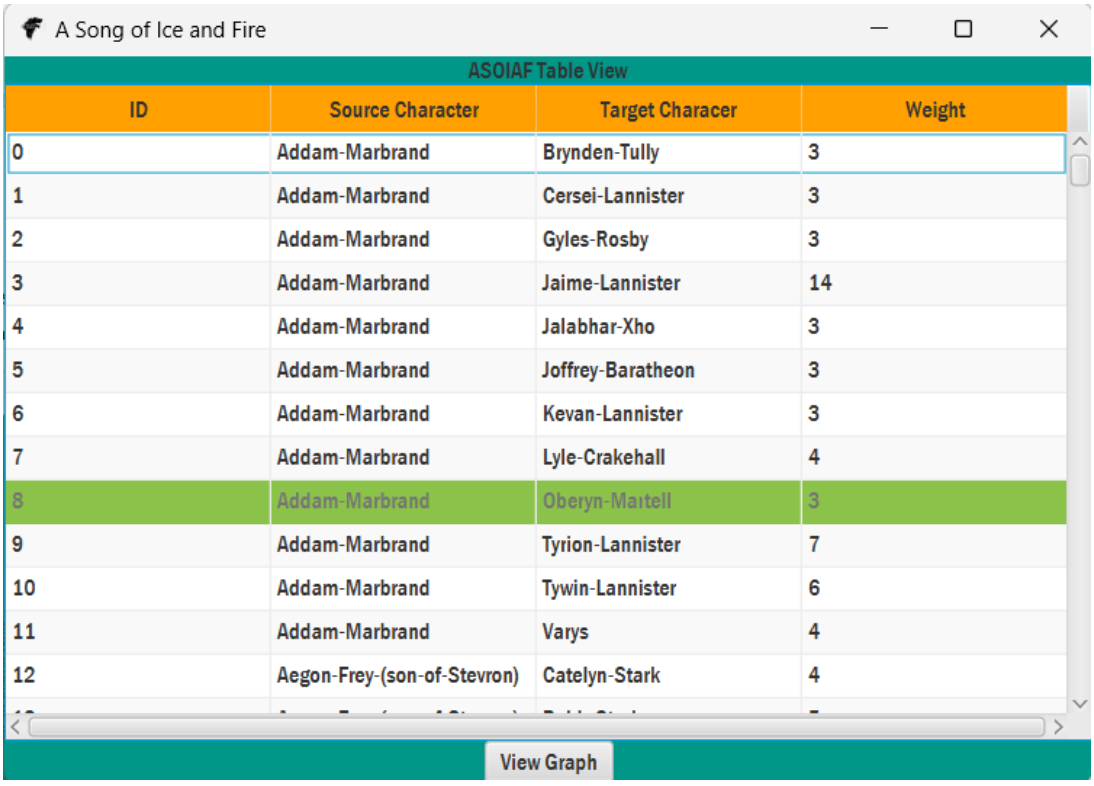


Figure 2 - A graph and a TableView Scene

What to Do:

1. Register your unique project idea by going to Java - Assignment 1 Ideas.xlsx. You can see what other students are planning to build. ***No duplicate projects will be allowed, so do not invest significant energy in your project until you receive approval from me***.
2. When the application launches, it should display a scene with a Chart object populated from information in a MySQL database.
3. There should be a utility to change scenes.
4. All scenes should be styled using CSS. Do not use the default grey background for everything. Add some color, round some corners, add images to buttons, change some fonts… have fun with it!!
5. The icon on the stage should be something unique that aligns with the information your project is displaying.

Figure 2 - Image showing changed icon

Grading

All of your marks will be based on the rubric defined below (and visible in Blackboard).

Criteria	Level 0	Level 1	Level 2	Level 3
Code Style	The code does not follow typical Java programming style. I.e. a capital letter to start all class names, lower case letters start variable and method names	The code is indented and has proper upper case/lower case conventions	All of level 1, plus each method has a Javadoc style comment prior to the method describing what it does	Level 2 plus different directories/packages are used for the model and controller classes to keep everything organized. The views should be in the resources folder.
User Experience	When the program launches, an exception is thrown or the Chart/TableView is not populated.	The program launches and the chart launches with some data, but the overall look and feel does not have a professional look to it. i.e. objects are not aligned or extreme colour choices are made.	Level 1 plus a professional look and feel is apparent.	
When launched, a Chart object is populated with data	The first scene does not contain a chart.	The first scene has a Chart object, but it is not populated from a MySQL DB.	Level 1 plus it is populated based off a DB query.	Level 2 plus the labels and legend are easy to read/visible.
Change to TableView or a different Chart scene	There is no utility to change from the first graph.	There is a functional object (i.e. button) that will change the scene to show a new graph or TableView object		
CSS styling	There is no CSS stylesheet and/or it is not connected to the view object.	The CSS is connected and styles up to 3 elements.	Level 1, plus it styles up to 6 elements.	Level 2, plus it styles more than 6 elements.
Change the icon	The default icon is showing	There is a new icon.	The new icon is somewhat related to the graph content.	

Criteria	Level 0	Level 1	Level 2	Level 3
The Model class(es) is well structured	The model class(es) is either not present or is not used.	The model class has poor naming conventions and/or poor instance variable data types. All instance variables must be private.	Level 1 plus all data types are logical. All method and variable names are logical and follow camelcase style. The constructor uses the set methods.	Level 2 plus all "set" methods contain ***useful*** validation. Do not use isBlank() (or equivalent) for validation in all set methods. Validate against known lists, number ranges, etc…
Database	There is no remote database connectivity in the project and/or the necessary SQL was not provided for a local DB	There is database access, but it is local and the necessary SQL statements are missing	There is local database access and the necessary SQL scripts are provided to create a local DB. Local DB access must use the user student and pw of student	All the necessary scripts have been run on the AWS MySQL server, code connects to that and there is over 50 lines of DB records.
Database Query for table	A DB query does not exist or creates an error	The DB query returns a valid ResultSet, but it is not used.	Level 1 plus it is used to populate a TableView object and connection, statements and resultSet objects are closed.	
Submission	A link to your private GitHub account was not submitted on Blackboard.	A private Github link was sent. PriyanshT is listed as a collaborator AND all project files including the build info are present for Intellij.	Level 1 plus there are at least 2 commits per week and the source URL for the data series is included.	Level 2 plus there are a total of over 6 commits with meaningful changes. In other words, do not submit a series of commits at the last minute with updates to comments. I want to see you working on your project over time.