

COMP9120 Database Management Systems**Assignment 2: Database Application Development****Group assignment (12%)****Introduction**

The objectives of this assignment are to gain practical experience in interacting with a relational database management system using an Application Programming Interface (API) (JDBC). This assignment additionally provides an opportunity to use more advanced features of a database such as functions.

This is a group assignment for teams of about 3 members, and it is assumed that you will continue in your Assignment 1 group. You should inform the unit coordinator as soon as possible if you wish to change groups.

Please also keep an eye on your email and any announcements that may be made on Ed.

Submission Details

The final submission of your database application is due at 11:59pm on Friday 12th Nov. You should submit the *items for submission* (detailed below) via Canvas.

Items for submission

Please submit your solution to Assignment 2 in the 'Assignment' section of the unit's Canvas site by the deadline, including EXACTLY THREE files:

- An assignment coversheet as a PDF document (.pdf file suffix) which is available for download from [this link](#) on Canvas.
- A SQL file (**BOSSchema.sql**) containing the SQL statements necessary to generate the database schema and sample data. This should contain the original schema and insert SQL statements, and any changes or additions you may have made.
- A Java file (**PostgresRepositoryProvider.java**) containing the Java code you have written to access the database.

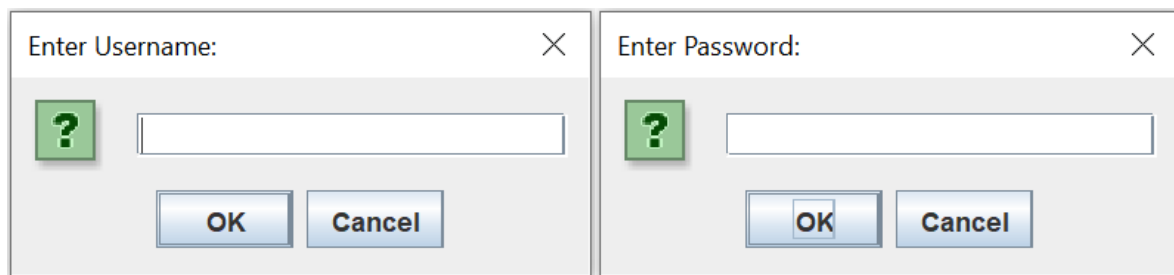
Section 1: Introducing the Brisbane Olympics System (BOS)

In this assignment, you will be working with the BOS System which is currently under development. The system still requires work in numerous areas, including the interaction with the database. Your main task in this assignment is to handle requests for reads and writes to the database coming from the user interface (UI). We first describe the main features that the BOS System should include from a UI perspective, and then discuss where the majority of your database code needs to be implemented.

Main Features***Logging In***

The first form a user is presented with when starting the BOS System is Login, as shown in Figure 1. This feature is still under development and currently requires that a user (Olympics sporting official) enters their

username and password to be validated prior to successfully log into the system. Security features such as password encryption/hashing will be implemented at a later stage (and is out of scope for this assignment). Once logged in, the user is taken to the Olympics Events List screen to view their associated events for the various Olympics sports.

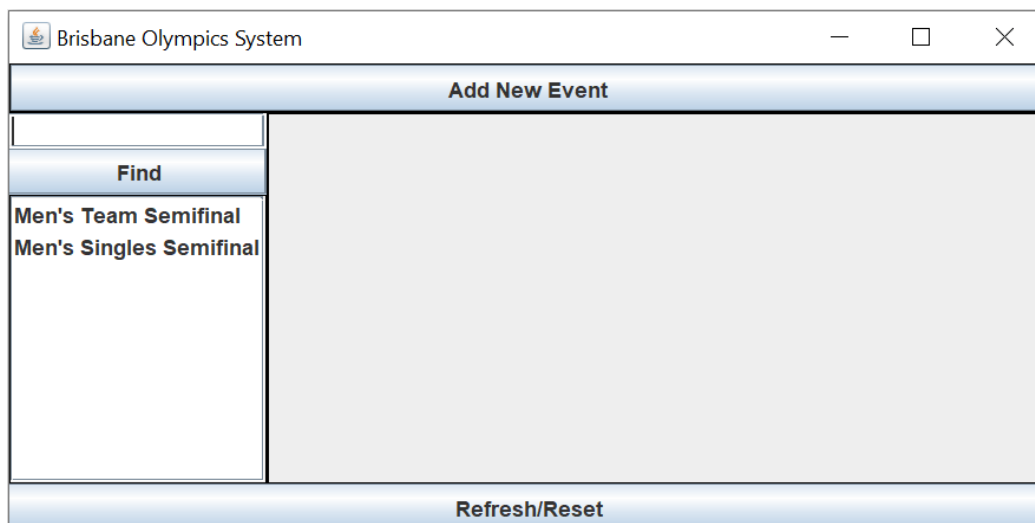


The login form consists of two side-by-side dialog boxes. The left dialog box is titled 'Enter Username:' and contains a green square button with a white question mark, a text input field, and 'OK' and 'Cancel' buttons. The right dialog box is titled 'Enter Password:' and contains a green square button with a white question mark, a text input field, and 'OK' and 'Cancel' buttons.

Figure 1 – Login form

Viewing Events List

Once a user has logged in, they are shown a list of all their associated events, as illustrated below in Figure 2. This events list must be ordered by sport name. Each event has an event name, sport name, and may or may not have a sporting official assigned as a referee, judge, and medal giver. An event is associated with a sporting official if the official is recorded as either the referee, judge, or medal giver.



The 'Brisbane Olympics System' window displays an 'Add New Event' dialog box. The dialog box has a search field at the top, a 'Find' button, and a list of events. The list contains two items: 'Men's Team Semifinal' and 'Men's Singles Semifinal'. At the bottom of the dialog box is a 'Refresh/Reset' button.

Figure 2 – Viewing Events List

Finding Events

A user can search through all events by entering a word or phrase (a 'keyword') in the field above the Find button, as shown in Figure 3, and then clicking on Find. When such a keyword is specified, then only events including this word or phrase in their sport names, event names, or usernames of the sporting officials will be retrieved and shown on the list. The search must also ignore case sensitivity. For example, given the search keyword 'semi', Find will return all events that include the word 'semi' in their sport names, event names, or sporting official usernames. Searching with a blank/empty keyword field will show all of the logged in user's associated events. Any search results returned must be ordered by sport name.

Figure 3 – Finding Events

Viewing Event's details

Selecting an event from the events list will present details for that event, as shown in Figure 4. The event name appears in the large text area in the bottom-right part of the form.

Figure 4 – Viewing Event's Details

Adding an Event

Users may also add a new event by clicking on the Add New Event button, entering event details in the popup dialog that appears, and then clicking on Add Event, as shown in Figure 5. User must enter a dash ('-') as either the referee, judge, or medal giver if there is no sporting official allocated to that event for a particular role.

Brisbane Olympics System

Sport:

Referee:

Judge:

Medal Giver:

Add Event

Figure 5 – Adding an Event

Updating an Event

Users can also update an event by modifying data in the event details screen as shown in Figure 6, and clicking on the Update Event button.

Brisbane Olympics System

Add New Event

Find	Event Id: 3
Men's Team Semifinal	Sport: Boxing
Men's Singles Semifinal	Referee: -
	Judge: GuoZ
	Medal Giver: ChrisP
	Men's Team Semifinal Tournament

Update Event

Refresh/Reset

Figure 6 – Updating an Event

The Refresh/Reset button can be used to refresh the event details from the database, allowing the user to check that their changes have been made, or to see if an event has been updated.

Database Interaction Code

The files that are needed for the Java version of assignment are as follows:

1. **BOSSchema.sql**: a file which contains SQL statements you need to run to create and initialise the BOS system database, before starting the application
https://canvas.sydney.edu.au/files/20044881/download?download_frd=1
2. **Assignment2_JavaSkeleton.zip**: a zip file encapsulating the Java project for the BOS system
https://canvas.sydney.edu.au/files/20100589/download?download_frd=1

To look through the BOS system code, you'll need to import the Eclipse project archive file into Eclipse. Please begin by trying to import the project into Eclipse using the steps below:

- 1) Right Click the zip file containing the assignment 2 archive > 7zip > Extract Here.
- 2) Start Eclipse and choose a new location for your workspace in a new folder.
- 3) Click on File > Import > General > Existing Projects into Workspace > Next.
- 4) Browse to the folder you extracted zip file to, and then click Select Folder.
- 5) Ensure the Assignment 2 Project is ticked in the Projects box, and click Finish.

If you experience any difficulties importing the eclipse project, ask your tutor or lecturer for assistance.

Once you import the project in Eclipse, you should notice that there are 3 main packages in the solution: Presentation, Business, and Data. The UI (in the Presentation package) is currently coded to invoke logic in the business layer (eg: see `EventProvider` class in the Business package), which in turn delegates responsibility for interacting with the database to an appropriate repository in the data layer (eg: see `PostgresRepositoryProvider` in the Data package). Notice that separating concerns in this way makes it easier to write a different `RepositoryProvider` in the data layer (eg: we could write a `MySQLRepositoryProvider`) if we wanted to change our database management system. In this assignment, you will mainly be working on writing the appropriate queries and SQL commands to fulfil the BOS system functionality described in Section 1; where these SQL commands and queries should be written in the `PostgresRepositoryProvider`. You should use the correct username/password details as specified in tutorial 7. The application's main method can be found in the `EventTrackerFrame` class. You can run this main method to run the entire application by right clicking the `EventTrackerFrame` file on the Project Explorer > Run As > Java Application.

Section 2: Your Task

Core Functionality

In this assignment, you are provided with a Java skeleton project that must serve as the starting point for your assignment. Your task is to provide a complete implementation for the file `PostgresRepositoryProvider.java`, as well as make any modifications necessary to the database schema (i.e., `BOSSchema.sql`). Specifically, you need to modify and complete these five functions:

1. `checkUserCredentials` (for login)
2. `findEventsByOfficial` (for viewing events list)
3. `findEventsByCriteria` (for finding events)
4. `addEvent` (for adding event)
5. `updateEvent` (for updating event)

Note that, for each function, the corresponding action should be implemented by issuing SQL queries to the database management system. If you directly output and/or manipulate the result without issuing SQL queries, you are considered as cheating, and you will get zero point for the assignment.

Marking

This assignment is worth 12% of your final grade for the unit of study. Your group's submission will be marked according to the attached rubric.

Group member participation

If members of your group do not contribute sufficiently you should alert the unit coordinator as soon as possible. The course instructor has the discretion to scale the group's mark for each member as follows:

Level of contribution	Proportion of final grade received
No participation.	0%
Full understanding of the submitted work.	50%
Minor contributor to the group's submission.	75%
Major contributor to the group's submission.	100%

Marking Rubric

Your submissions will be marked according to the following rubric, with a maximum possible score of 12 points.

	Part marks (0 – 1)	Full marks (1.5 – 2)
Login	Can correctly login the user 'ChrisP' and validate his username and password.	All valid users can be logged in successfully, and unsuccessful user logins should be rejected.
View Events List	Correctly list all events associated with user 'ChrisP' in the correct order (see Figure 2).	Correctly list all events associated with a user, in the correct order, for all possible username input from Figure 1.
Find Event	Correctly list events for keyword "semi" (see Figure 3).	Correctly list events for all possible keywords.
Add Event	Can correctly add an event to the database.	Can correctly add all valid events to the database. Events entered with invalid details should be rejected.
Update Event	Can correctly update the sport name of an event as shown in Figure 6.	Can correctly update details of all events, ensuring details entered for an event are valid.
Stored Procedure	A couple of stored procedures (functions) are correctly created in the submitted SQL file.	A couple of stored procedures (functions) are correctly created in the submitted SQL file, and correctly called in two of the five specified functions.