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) (Python DB-API). 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 Python file (`database.py`) containing the Python code you have written to access the database.

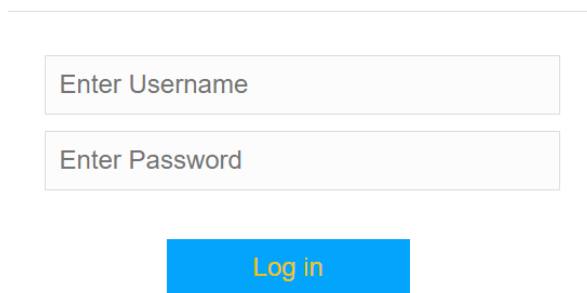## Section 1: Introducing the Brisbane Olympics System (BOS)

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 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.

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 in to 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 page to view their associated events for the

various Olympics sports.

Figure 1 – Login form
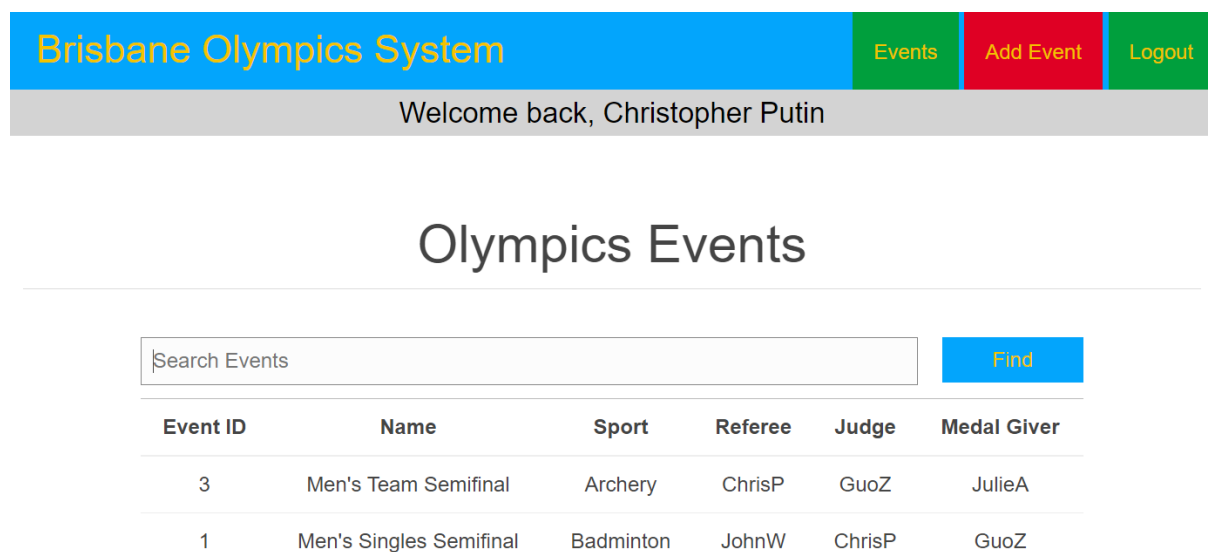
Viewing Events List

Once a user has logged in, they are shown a list of all their <u>associated</u> events, as shown 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 <u>associated with a sporting official</u> if the official is recorded as either the referee, judge, or medal giver.

Event ID	Name	Sport	Referee	Judge	Medal Giver
3	Men's Team Semifinal	Archery	ChrisP	GuoZ	JulieA
1	Men's Singles Semifinal	Badminton	JohnW	ChrisP	GuoZ

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 next to 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 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.

Olympics Events

Event ID	Name	Sport	Referee	Judge	Medal Giver
3	Men's Team Semifinal	Archery	ChrisP	GuoZ	JulieA
1	Men's Singles Semifinal	Badminton	JohnW	ChrisP	GuoZ
4	Men's Tournament Semifinal	Basketball	-	JohnW	MaksimS

Figure 3 – Finding Events

Adding an Event

Users may also add a new event by clicking on the *Add Event* tab in the title bar, entering event details in the popup dialog that appears, and then clicking on Add Event button. 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.

New Event

Event Name:

Sport:

Referee:

Judge:

Medal Giver:

Add Event

Figure 4 – Adding an Event

Updating an Event

Users can also update an event by modifying data in the event details screen as shown in Figure 5, and clicking on Update Event button. You can access this update screen by clicking on an event from the list of events in the Events tab.

Figure 5 – Updating an Event

Database Interaction Code

The files that are needed for the Python 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_PythonSkeleton.zip`**: a zip file encapsulating the Python project for the BOS system https://canvas.sydney.edu.au/files/20099339/download?download_frd=1

To inspect the BOS system code, you need to unzip the ZIP archive first, which will create a folder that includes the name **`Assignment2_PythonSkeleton.`** If you experience any difficulties installing and exploring the project, ask your tutor or lecturer for assistance.

The skeleton code uses a number of Python modules to implement a simple browser-based GUI for the BOS system. The main modules are the Flask framework for the GUI and the psycopg2 module for the PostgreSQL database access. Similar to tutorial 7, **you will need to install the Psycopg2 module and the Flask module**. The skeleton code follows the structure described below:

- The main program starts in the **`main.py`** file. You need to use the correct username/password details as specified in tutorial 7, and then implement the missing database access functions – including any necessary SQL code statements required – in the data layer **`database.py`**.
- The presentation layer is done via a simple HTML interface that can be accessed from a web browser. The corresponding page templates are located in the **`templates/`** subdirectory, their CSS style file is **`static/css`**.
- The transition between the different GUI pages and the initialisation of the Flask framework is done in the **`routes.py`** file. It currently just invokes the pages, but there is no further business logic implemented yet.

You can run the code by running "**`python main.py`**". This starts a local web server and prints out some

debug messages in the terminal; the GUI can then be accessed with any web browser on the same computer via the local URL http://127.0.0.1:5000/ (If that doesn't work you can also try http://0.0.0.0:5000/). Please note that, to terminate the application, you will need to stop the local web server which is running in the background.

Section 2: Your Task

Core Functionality

In this assignment, you are provided with a Python skeleton project that must serve as the starting point for your assignment. Your task is to provide a complete implementation for the file **database.py**, 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 in line with the rubric that follows.

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 5.	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.