Software Development – Cloud Development

Year 3

2023/2024 Academic Year

Assignment 3 – A database-enabled webapp (with a single-page option).

Motivation for Assignment

While you've been working on the previous version of your webapp, the Coach has run another training session for his young swimmers. A new ZIP file of swimmer times (called `swimdata2.zip`) has been created as a result of this session occurring[1]. Note the following about this new data: (1) there are a lot less swimmers in this file (as a bunch of them are off at a competition) and (2) a few new swimmers have joined the club, including a seven year old called "Darius"…[2]

Although it is tempting to remove the sixty datafiles in your webapp's `swimdata` folder and replace them with the new datafiles in `swimdata2.zip`, such a solution is problematic in that the Coach wants to be able to call up any prior training session's data via his webapp (without having to swap out datafiles). A better solution is to use a database backend to store the data for all of the Coach's training sessions, then provide a way (within the webapp) to easily switch between them.

The major goal of this assignment is to add database support to your webapp in order to allow the Coach to work with any training session's dataset. A minor goal is to convert your multi-page webapp into a single-page webapp (using the HTMx framework).

Assignment Details

Part #1: Install either MySQL or MariaDB onto your computer[3]. Create a database and any associated tables to hold all of the training datasets. Note: A suggested structure/design for the database tables is provided at the end of this PDF. This is only a suggestion as you can use whichever database structure you desire. Create a file called `schema.sql` which contains instructions for recreating your database structure on any other computer.

Part #2: With your database schema created and ready, create a Jupyter Notebook which uses Python/DBcm to take the data in any of the Coach's ZIP datafiles and add the training dataset to your database. Call your notebook `ingest_data.ipynb`.

---

1 The original datafile (`swimdata.zip`) and this new one are available for download on Teams, in the `SwimData` folder.

2 Being only seven, young Darius trains with the Under-8 group.

3 But **not** both! Misery ensues when you try to install *both* MySQL and MariaDB. Pick one and stick to it, but note that MariaDB comes with a strong recommendation.

Part #3: Add a new opening page to your webapp which allows the Coach (or anyone else) to select an individual training session's dataset to use[4]. The values displayed on this page's drop-down list must come from your database backend, not from your filesystem.

Part #4: With Part #3 complete, adjust your webapp code to work with the selected dataset, taking the training dataset from your database as opposed to directly from the filesystem. Only the data from the selected training session is charted at any one time. FYI: the swim club never has more than one training session per day.

Part #5: Visit the HTMx website (https://htmx.org/) to learn about this JavaScript-replacing framework. Use the facilities of HTMx to convert your multi-page webapp to a single-page webapp. That is, rather than having multiple webpages with an individual drop-down list on each page, reconfigure your webapp to place all of your drop-downs and your bar chart on a single page. As the Coach interacts with each of your drop-downs, your webpage needs to update accordingly[5].

Part #6: With your single-page webapp ready (Part #5), deploy it to your PythonAnywhere account, replacing the previous version of your webapp from CA #2.

Marks Allocation

- **50%** of this year's total marks with the following breakdown:

  up-to 5 marks for Part #1,
  up-to 10 marks for Part #2,
  up-to 5 marks for Part #3,
  up-to 10 marks for Part #4
  up-to 15 marks for Part #5, and
  up-to 5 marks for Part #6.

Note: Your submission needs to clearly identify the work you are submitting for each part. For Part #1, submit your `schema.sql` file. For Part #2, submit your `ingest_data.ipynb` notebook. For Part #3, submit your webapp (as an appropriately-named ZIP) updated with the additional drop-down page. For Part #4, submit another ZIP for your updated webapp, and for Part #5 submit yet another ZIP for your HTMx-enabled webapp. Part #6 has no submission requirement (other than deployment to PythonAnywhere). The marks for Part #6 are for the HTMx-enabled version of your webapp (so don't bother deploying any earlier version of your webapp to PythonAnywhere).

To submit your work, ZIP the entire contents of your folder containing all of the requirements for this assignment, then copy the ZIP to your University OneDrive, before sharing the created OneDrive link with **paul.barry@setu.ie**. For Part #6, it is assumed you'll deploy to your existing PythonAnywhere account. Note that each of your webapp versions will be executed on your lecturer's computer, so be sure **NOT** to hard-code any values in your code which refer to specific locations/resources on your computer).

The due date/time for this assignment is: **5:00pm on Friday December 8th 2023**.

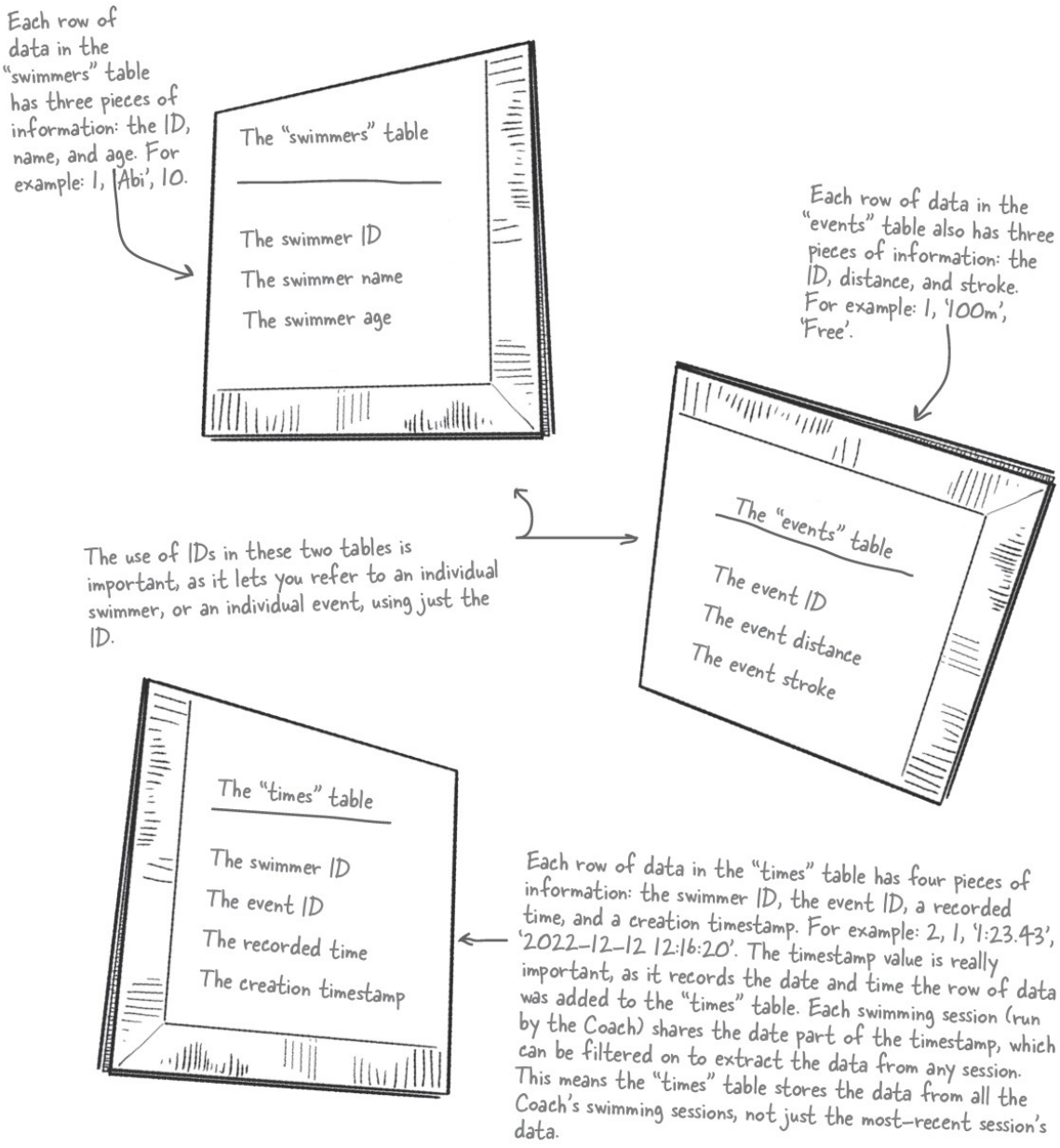
This is an individual assignment: you are expected to work on your own, and that the work you submit is written/created by you. You must declare if this is not the case. If you collaborate with anyone else, this must be declared.

---

4 See the https://barryp.pythonanywhere.com/swims for an example of what one such web page might look like.

5 Notice the "examples" link near the right-hand-bottom corner of the HTMx main page which provides access to many, many example uses of HTMx. The "Value Select" example is of particular interest.

A Suggested "Swim Club" Database Schema (rough draft)



The napkin structure + data

The `swimmers` table contains a row of data for each of the Coach's swimmers. In addition to the unique ID for each row, each swimmer can be identified by the combination of their name and age group.

Assuming the `swimmers` table is populated with each swimmer's data, here's what the first five rows of data might look like:

You'll get to populating your tables with data in a little bit.

id	name	age
1	Abi	10
2	Hannah	13
3	Darius	13
4	Owen	15
5	Mike	15

Each ID value uniquely identifies a swimmer.

Each name and age group combination appears once in the "swimmers" table.

Similarly, the `events` table, when populated with each event's data, might have the following data in its first five rows:

id	distance	stroke
1	100m	Free
2	100m	Back
3	100m	Fly
4	50m	Back
5	200m	Free

Each ID value uniquely identifies an event.

Each distance and stroke combination appears once in the "events" table.

The `times` table, when populated with timing data, might have the following data in its first ten rows:

"ts" is short for "timestamp".

swimmer_id	event_id	time	ts
2	1	1:21.43	2022-12-12 12:16:20
2	1	1:21.40	2022-12-12 12:16:20
2	1	1:21.62	2022-12-12 12:16:20
2	1	1:25.38	2022-12-12 12:16:20
3	2	1:22.57	2022-12-12 12:16:20
3	2	1:29.64	2022-12-12 12:16:20
3	2	1:20.39	2022-12-12 12:16:20
3	2	1:23.83	2022-12-12 12:16:20
4	1	1:15.57	2022-12-12 12:16:20
4	1	1:14.40	2022-12-12 12:16:20

All these rows have the same timestamp, so they were all recorded by the Coach during the same swim session.

Swimmer ID #3 is "Darius", and Event ID #2 is "100 Back". Note the four recorded times for this swimmer/event combination.