Homework 3

Submissions due: Wednesday, 28 February 2017, 23:59pm

Peer reviews due: Saturday, 03 March 2017, 23:59pm

Tools

Bottle: Python Web Framework

Install the Bottle Python web framework, following the instructions here:

- https://bottlepy.org/docs/dev/

Check to make sure your Python installation has **sqlite3** installed.

Python

If you need to install Python, consider using the Anaconda distribution or the Miniconda environment manager:

- Anaconda: https://www.anaconda.com/download/
- Miniconda: https://conda.io/miniconda.html

IDE

Use your favorite text editor or IDE. If you don't have one, give Visual Studio Code a try:

- https://code.visualstudio.com/download

You can point Visual Studio Code toward the Python environment you want to use for your application following the instructions here:

- https://code.visualstudio.com/docs/python/environments

Requirements

Build a website along the lines of the prior homework assignments using the Bottle framework.

Optimize your website for underpowered clients with limited power and computing resources; minimize your use of JavaScript use (client side) and maximize the work done on the server side.

Your Bottle website supports five *routes*:

/index: displays the home page

/list: displays the todo list

/new: displays a form for entering a new task (todo list item)

/edit/<id:int>: displays form for editing the task (todo list item) with key *id*, prepopulated with the current values

/delete/<id:int>: deletes the task (todo list item) with key *id*.

Database

Store your tasks in an *sqlite3* data base *todo*.db which consists of a table *task* with the following attributes:

Attribute	**Type & Constraints**
id	INTEGER primary key
title	TEXT
description	TEXT
posted	DATE
due	DATE
updated	DATE
status	BOOL (i.e. 0 or 1)

Instead of DATE you may also use DATETIME;

For SQL datatypes as defined by major relational databases, see

- https://www.w3schools.com/sql/default.asp

For sqlite3 specifics, see

- https://www.sqlite.org/datatype3.html

You may want to use a tool like DB Browser for SQLite to inspect and modify your database for test purposes.

- http://sqlitebrowser.org/

/index

The /index route displays the home page of the site. You may use your homepage from hw2. Make sure the home page allows for navigation to the todo list page (via the /list route).

/list

This route displays the *todo* list*,* showing for each task its *title*, *description*, and the *post*ed, last *updated*, and *due* date.

Associate with each list item an *Edit* and *Delete* button, which when clicked will generate a request along the /edit and /delete route, respectively.

Note: In both cases you will need to communicate the *task id* for the task to be edited or deleted to the backend. The *task id* should not be visible to the user, but it must be part of the todo list document as a hidden component.

This page also needs to provide an *New Task* button, which when clicked will display a form for entering a new task (via the /new route).

The user will be able to sort the list by *post*ed, last *updated*, and *due* date, and filter it for completed or incomplete tasks. By default, the list is ordered by *posted* date, and unfiltered, i.e. all tasks, completed or incomplete, are shown. Any re-display of the todo list preserves the most recent sort and filter settings.

In keeping with our design goal of optimizing for resource-constrained clients, all re-ordering and filtering is done on the server.

Hint: Consider including a hidden form in your document whose input fields record the sorting and filtering settings. Click event handlers can modify the input values of the form, set the action required, and submit the form to prompt the needed action on the server.

/new

The response to a request along this route displays a form with fields for the task attributes *title*, *description*, and *due date*.

When the form is submitted, the *posted* and *updated* date will be set to the current date/time, and the status is initially *false* (*0*). After storing the new task in the database, the todo list will be re-displayed (as per the most recent sort and filter settings).

Hint: You can have the database do the work for you by constructing a query that performs the sorting and filtering required.

/edit/<id:int>

The response to a request along this route displays a form in which all current task attributes are visible except for the task *id.* The *posted* and *updated* are visible but not modifiable.

When the form is submitted, the updated date will be set to the current time/date. After updating the task in the database, the todo list will be re-displayed (as per the most recent sort and filter settings).

/delete/<id:int>

This request will delete the task with key *id* from the database, and then re-display the todo list (as per the most recent sort and filter settings).

Header

Make the header read “cmps183: Homework 3”.

File organization

Folder structure:

a) Put the html files in folder called cmps183hw3.
b) Define your routes and endpoint functions in a file *controller.py*.
c) Include your sqlite3 database *todo.db*; it should contain at least 6 tasks.
d) Put your templates in a subfolder called *templates*.
e) Put your css files in a subfolder called *css*.
f) Put your javascript files in subfolder called *scripts*.
g) Put your pictures in a folder called *images*.

Keep this folder structure in mind when you define hyperlinks in your html files.

Submission instructions

1) Create a zip file for folder cmps183hw3.
2) If you registered for Homework 2 then you are already registered for Homework 3.
   Otherwise, self-register for the assignment at Crowdgrader.org with this sign-up link.
   (https://www.crowdgrader.org/crowdgrader/venues/join/3672/wuvate_mawona_wipavo_tanupo)
   Important: Please, use your UCSC email address to register.
   Then go to this CrowdGrader assignment.
   (https://www.crowdgrader.org/crowdgrader/venues/view_venue/3672)
   Note: You will need to log in with the email address that you signed up with.
3) Click on the *Submit* link at the top left.
4) Attach the zip file you created in step 1.
5) Add any comments, if you wish, by clicking on *Edit Content.* Be sure to specify any information that a user should be aware of.
6) That's it.

If you cannot log in, most likely you signed up with a different email account.

Review Instructions

Part of the assignment (20%) is to review 4 submissions of your peers. A grading rubric is available on Crowdgrader.

The primary purpose of the review is to learn from each other and to give honest feedback.

To do the reviews go to the assignment URL.

Collaboration

There is no issue with discussing the homework and possible solutions within your project team or other group. However, for the homework assignments each student needs to write his or her own code. This is the only way you will actually learn how to do it yourself. If you cheat, you primarily cheat yourself.