Homework 4

Submissions due: Tuesday, 13 March 2018, 23:59pm
Peer reviews due: Friday, 16 March 2018, 23:59pm

Tools

Make sure you have a Python3 environment with the Bottle framework installed, including Squlite3 (see HW3 for details). The *hw4 todospa* app posted on Piazza (Resources > Homework) runs in such an environment. Use the running app as a reference to follow the instructions below.

Read up on the *Fetch API* for asynchronous communication between the browser and the server.

Requirements

Important note: You may write your own app or complete *hw4 todospa* to create a single-page app with the functionality specified below. In any case, it may be helpful to take a good look at the *hw4 todospa* application source code.

Unlike the app in hw3, which was server-heavy, the app in hw4 loads only one html document initially and then incrementally updates this document using the *fetch API*. The *fetch API* is a recent API that replaces *XMLHttpRequest*, also known as *AJAX*.

HW4 todospa

The *hw4 todospa* app consists of a single page with the following components:
An *unchanging frame* that includes a

1. Header
2. Left sidebar
3. Right sidebar
4. Footer

and a center panel that shows the items of the todo list.

The app currently has the following functionality:

Initial page (routes /, /index, /home):

The initial page loads the page frame, which upon loading calls a script that loads all todo items currently in the database.

New task button (right sidebar)

Pressing the New task button opens a task form at the top of the center panel for entering new tasks.

This form includes a text area for the task description, a checkbox indicating whether the task has been *done* or remains to be done (*tbd*), a *Save* button and a *Cancel* button. Note that one may press the *New task* button repeatedly; each click will open a new task edit form without reloading the page.

Save button (route /task/new)

Clicking the *Save* button stores the new task in the database *todo.db* by making a *fetch* call to send a POST request along the */task/new* route. The task data is transmitted as JSON data in the body of the request.

Cancel button

Clicking the *Cancel* button removes the task edit form from the center panel.

Todo List Item Checkbox

Clicking the checkbox in a todo list item updates the status of the item in the database by making a fetch POST request along the */update/status* route.

Complete the app

Complete the app by adding the following functionality:

Filter button

The filter panel in the right sidebar allows the user to filter the task list in the center panel. Implement the filtering functionality so that clicking the filter will show the tasks with the selected task status. You may do this by either manipulating the visibility of the todo items (e.g. by setting attribute *style="display: none"*) or by making a fetch request along the *route/task/<filter>*. The endpoint of this route is used for initially loading the task list.

Todo List Item

Implement the functionality for the Edit and Delete buttons.

Edit button

Clicking an *Edit* button shall open a task edit form in place of the todo item in the task list; the form shall be prepopulated with the current values for the task description and checkbox.

Save button

Clicking the Save button on the form (displayed by clicking *Edit*) shall update the task in the database by making a fetch POST request. You may transmit the task data as a JSON string or by using a FormDict (see Fetch API).

After receiving confirmation that the task has been updated in the database, the task will be displayed in the todo list in its original position but with the updated task description and checkbox setting.

Hint: You can manipulate the task visibility via the display setting in the style attribute.

Cancel button

Clicking the Cancel button on the task edit form will redisplay the task in its previous position with the task description and checkbox values prior to the cancelled edit.

The user should be able to edit several tasks simultaneously.

Delete button

Clicking the *Delete* button of a todo item shall delete the task from the database and, upon confirmation of deletion from the database, remove the task from the todo list in the browser.

Bonus Features

Use the left sidebar in your app to indicate to the reviewers which bonus features you implemented.

Post Date/Time and Due Date [50 points]

Add a post date and due date to tasks. The task posting date/time is set upon storing a new task in the database and displayed in the todo item. The user can edit the due date in the task edit form.

Update Due Date individually [25 points]

The user can update the Due date directly without going through the edit form.

Sort Todo List by date [25 points]

The user can sort the todo list ascending or descending by posting date or due date. The user can specify the order by using a selection panel in the right sidebar.

File organization

Put your app in a folder called cmps183hw4. Use the same structure as *hw4 todospa* if you are completing this app, or continue with the structure of your previous app.

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

Submission instructions

1) Create a zip file for folder cmps183hw4.
2) If you registered for Homework 3 then you are already registered for Homework 4.
   Otherwise, self-register for the assignment at Crowdgrader.org with this sign-up link.
   (https://www.crowdgrader.org/crowdgrader/venues/join/3689/cegiru_bonuge_vasyfa_firicy)
   Important: Please, use your UCSC email address to register.
   Then go to this CrowdGrader assignment.
   (https://www.crowdgrader.org/crowdgrader/venues/view_venue/3689)
   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.

CMPS183 WINTER 2018 HOMEWORK 4

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.