01UDFOV/01TXYOV – WEB APPLICATIONS I

BIGLAB 2C: COMPLETE APIS INTEGRATION

WHAT ARE WE BUILDING THIS WEEK?

We will continue to **update the front-end** of the web-based task manager to **use all the APIs** designed in the first part of this BigLab. In particular, we will use the APIs for filtering tasks, and for deleting and updating them.

STEP-BY-STEP INSTRUCTIONS:

Modify the front-end of the web-based task manager in the following ways.

- Minimize the "n-clients problem" shown during the lecture, by reloading the application state every time a filter change. In particular, when the user clicks on a filter, the list of **filtered tasks is retrieved from the server** by invoking the proper API.
- Make the delete operations persistent: when the user deletes a task, the **task is removed from the server-side database**, and the list of tasks is updated and displayed accordingly. To do so, invoke the proper API for deleting tasks, and then retrieve the updated list of tasks from the server.
- Make the update operations persistent: when the user updates a task, the **task is modified on the server-side database**, and the list of tasks is updated and displayed accordingly. To do so, invoke the proper API for modifying a task, and then retrieve the updated list of tasks from the server.
  Beware. A task can be updated in two different ways:
  - by updating its fields through the dedicated modal;
  - by marking it as *completed/not completed*. Tasks can be marked as *completed/not completed* by using the checkboxes that are displayed near each task in the list. A checkbox is checked if the corresponding task is completed, unchecked otherwise.

Hints:

1. *The general specification of the second BigLab can be found at:* *https://github.com/polito-WA1-AW1-2021/course-materials/blob/main/labs/BigLab2/BigLab2.pdf*
2. *Trust the server! It shall be always up-to-date, and every operation rely on it, not on the internal state of the webapp.*