Luigi De Russis, Alberto Monge Roffarello

LAB 5 – WEB IN PYTHON: FIRST APPROACH TO FLASK

GOAL

The goal of this lab is to write a web application, based on the Flask framework, to implement a web-based version of the task-management application developed in the fourth laboratory (AmITaskListBot.py).

PRELIMINARY

You should start with the basic structure of a Flask application (`app.py` file and `templates/` and `static/` directories). You may do that by creating a new project (PyCharm > File > New Project… > Flask), or by forking+cloning of the Lab5 repository (go to *https://github.com/AmI-2019/python-lab5*, Fork, then on PyCharm > VCS > Checkout from Version Control > Git). Remember to checkout (clone) your forked version of the project, not the one under AmI-2018.

We also suggest to checkout (clone) the Lab4 project (the one you developed last week) in a different window, to copy and re-use all the database access functions.

STEP 1 – INSERT AND "ID" FIELD IN THE DATABASE

Modify the database MySql schema starting from the one used in the Lab4.

1. Add a new integer column called "`id`" to the `task` table. This column should be set as *primary key*, *auto-increment*.
2. Check and modify the database access functions (especially `db_remove_task`) in order to use `id` instead of `todo` to identify a specific task. In this way we properly use *primary keys* to identify rows in a table.

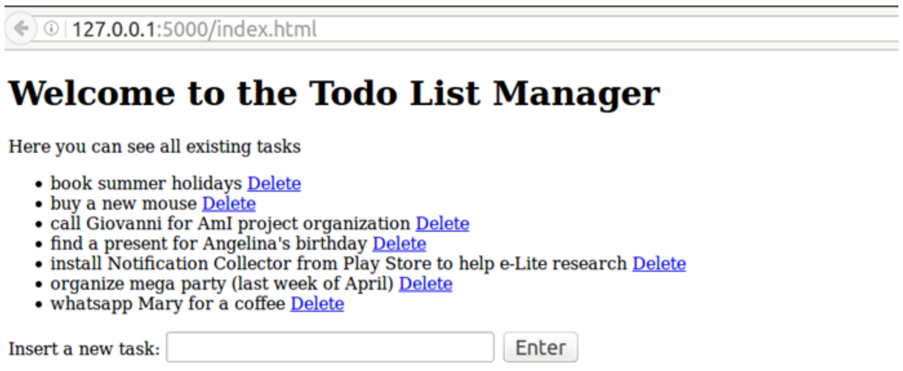
STEP 2 – TRANSFORM THE TELEGRAM BOT IN A SIMPLE WEB APPLICATION

1. Create a web application to display, in its main page (see Figure 1), a list of all existing tasks.
2. Add a one-line form to insert a new task, at the bottom of the page. Entering the new task should create it into the database. The form should call the `insert_task` page, with `description` and `urgent` parameters.
3. Add a "delete" button (or link) besides each task: it will delete the corresponding task. The link should call a `delete_task` page and embed the `id` parameter[1] of the task to be deleted.

The overall "site map" of the web application is shown in Figure 2.

[1] Suggestion: use `url_for('delete_page', id_task=id)`, combined with an `@app.route('/delete_page/<id_task>')`. See http://flask.pocoo.org/docs/1.0/blueprints/#building-urls

Figure 1 – Index



127.0.0.1:5000/index.html

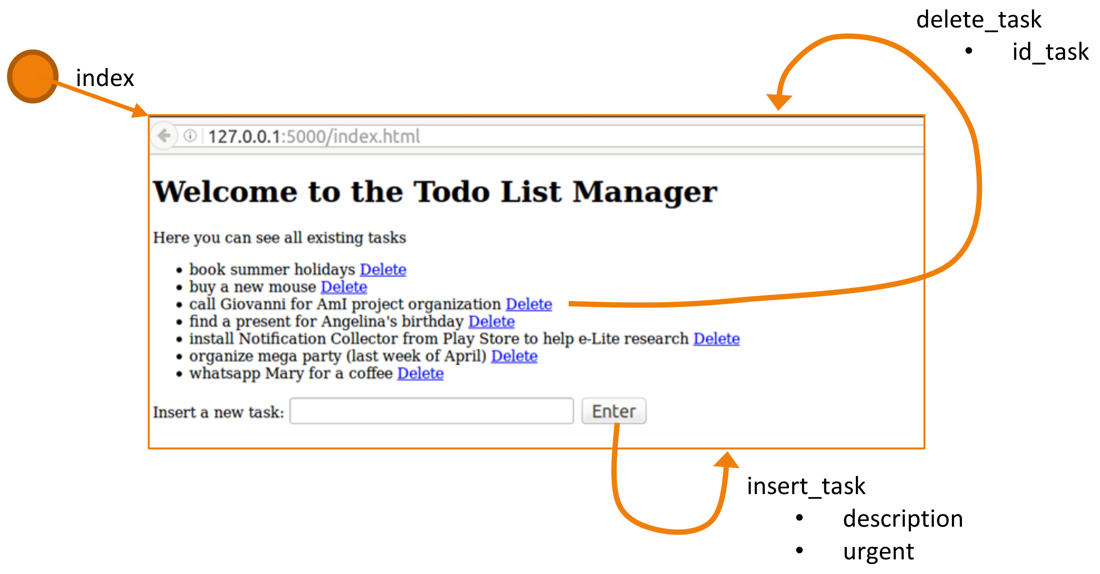
Welcome to the Todo List Manager

Here you can see all existing tasks

- book summer holidays Delete
- buy a new mouse Delete
- call Giovanni for AmI project organization Delete
- find a present for Angelina's birthday Delete
- install Notification Collector from Play Store to help e-Lite research Delete
- organize mega party (last week of April) Delete
- whatsapp Mary for a coffee Delete

Insert a new task: Enter

Figure 2 – Site Map


delete_task
• id_task
index
127.0.0.1:5000/index.html
Welcome to the Todo List Manager
Here you can see all existing tasks
• book summer holidays Delete
• buy a new mouse Delete
• call Giovanni for AmI project organization Delete
• find a present for Angelina's birthday Delete
• install Notification Collector from Play Store to help e-Lite research Delete
• organize mega party (last week of April) Delete
• whatsapp Mary for a coffee Delete
Insert a new task: Enter
insert_task
• description
• urgent