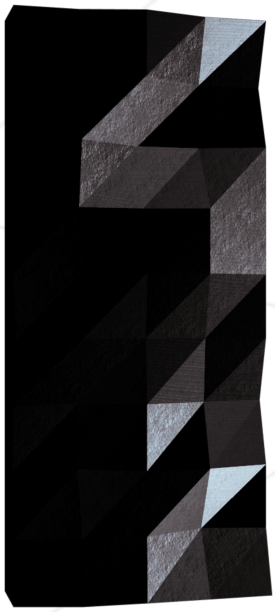

Training Python-Django - 1

Base Django

Summary: After Python, Django!

Version: 1.1

Contents

Chapter I

Preamble

Here is a list of monsters you may encounter wandering a dungeon in the land of Fangh:

- any kind of undead.
- giant spiders.
- orcs and goblins.
- trolls in the underground.
- sorcerers.
- cursed warriors.
- giant rats.
- an oil bottle.
- toilet paper.
- two sponges
- raviolis.

Chapter II

General rules

- Your project must be realized in a virtual machine.
- Your virtual machine must have all the necessary software to complete your project. These softwares must be configured and installed.
- You can choose the operating system to use for your virtual machine.
- You must be able to use your virtual machine from a cluster computer.
- You must use a shared folder between your virtual machine and your host machine.
- During your evaluations you will use this folder to share with your repository.
- Your functions should not quit unexpectedly (segmentation fault, bus error, double free, etc) apart from undefined behaviors. If this happens, your project will be considered non functional and will receive a `0` during the evaluation.
- We encourage you to create test programs for your project even though this work **won't have to be submitted and won't be graded**. It will give you a chance to easily test your work and your peers' work. You will find those tests especially useful during your defence. Indeed, during defence, you are free to use your tests and/or the tests of the peer you are evaluating.
- Submit your work to your assigned git repository. Only the work in the git repository will be graded. If Deepthought is assigned to grade your work, it will be done after your peer-evaluations. If an error happens in any section of your work during Deepthought's grading, the evaluation will stop.

Chapter III

Today's specific rules

- All paths relative to an application must be defined in a `urls.py` file located in a folder of this application.
- Any form (derived class of `django.forms.Form`) must be located in the `forms.py` application's file it is related to.
- Each displayed page must be properly formatted (a doctype, couples of `html` tags, `body`,`head` must be included), proper management of special characters, no strange display.
- Today, you will use Django's default development server provided with the `manage.py` utility.
- Only the specifically requested URLs must return a page without any error. Hence, if just a `/ex00` is requested, `/ex00foo` must return a 404 error.
- Requested URLs must work with or without end slash. Hence, if `/ex00` is requested, `/ex00` and `/ex00/` both must work.

Chapter IV

Exercise 00

	Exercise 00
Exercise 00: First static page.	
Turn-in directory : *ex*00/	
Files to turn in : `requirements.txt, files and folders of your project and application`	
Allowed functions : `All Django functionalities.`	

With this exercise, you will make your first static page with Django.

Create a `virtualenv` with `python3`, install `Django`, create a `requirements.txt` file containing the project's dependancies (what's installed in the virtualenv with `pip`) at the root of the repo.

Start a project `d05`.

Start an application `ex00`.

Create a page that will searchable at the following URL `/ex00` of your website. In this page, gather all the informations about the whole `Markdown` syntax and give this page the title "`Ex00: Markdown Cheatsheet.`".

The used template will be named `index.html`.

Chapter V

Exercise 01

	Exercise 01
Exercise 01: A few more pages.	
Turn-in directory : *ex*01/	
Files to turn in : `requirements.txt, files and folders of your project and application.`	
Allowed functions : `All Django functionalities.`	

Create the following pages in a second application `ex01`:

- **Title :** "Ex01: Django, framework web."

  URL: `/ex01/django`.

  Description: In this page, briefly introduce Django and its history.

- **Title:** "Ex01: Display process of a static page."

  URL: `/ex01/display`

  Description: In this page, describe the process causing the display of a static web page from a simple template going through a view, from the request to the answer.

- **Title:** "Ex01: Template engine."

  URL: `/ex01/templates`

  Description: In this page, describe the functioning of Django's default template engine as well as the functioning of:

  - Blocks.
  - Loops`for ... in`.
  - `if` control structures.
  - The display of the passing context variables.

The principle of the *don't repeat yourself* is part of the Django's philosophy. To respect this principle, create the required pages using a base template named `base.html`.

The `base.html` template must include:

- A `content` block in the `body`.
- A `style` block in the `head`.
- A `title` block in the `head`.

Also create a `nav.html` template containing a navigation bar listing the links to each of the exercise's pages.

All the pages of this exercise must be based on the `base.html` template. The `nav.html` template must be included in each of your pages.

Also create 2 files: `style1.css` and `style2.css`. The first one will define the text color as *blue*, the other one will be *rouge*. You will have to use `style1.css` in every page, except in the "Template engine" page, where you will use the `style2.css`.

`You must use each style sheet just once in all your templates for this exercise.`

`You will have to use collectstatic during evaluation.`

Chapter VI

Exercise 02

	Exercise 02
Exercise 02: First form.	
Turn-in directory : $ex02/$	
Files to turn in : `requirements.txt, files and folders of your project and application.`	
Allowed functions : `All Django functionalities.`	

Create a page accessible at the URL `/ex02` in a new `ex02` application.

This page will contain 2 parts:

- A form with a text field and a `Submit` button.
- A part that contains input history.

Here is the rule regarding the form:

- You must `NOT` hard code the form's field. You will use the `django.forms.Form` class provided by Django to create a form. You will pass it as `context` to the template's render function.

Here are the rules for the history:

- The input history will start empty.
- For each text submission in the form, you will have to:
  - List the input and its timestamp at the end of a `logs` file. If the file doesn't exist, it must be automatically created. This file must be created in the folder or the `ex02` application.
  - Add the input to the page's history with the submission's time and date.
- The path to the `logs` file (file name included) must be defined in the project's `settings.py` file. Hence, each submitted entry will be included in the page's history and in the `log` file, preceded by its timestamp.
- The data must also be persistent. If the development server must restart somehow, the data must not be lost.
  As a rule, if you redisplay the page, the data are redisplayed as long as they're saved.

Chapter VII

Exercise 03

	Exercise 03
Exercise 03: Fifty shades of bic.	
Turn-in directory : *ex*03/	
Files to turn in : `requirements.txt, files and folders of your project and application.`	
Allowed functions : `All Django functionalities.`	

Create a final application `ex03`.

Display a page containing a 4 columns x 51 lines table (one line will be dedicated to the column's name).

You will access this page at this URL `/ex03`.

Each table column will have a different color: *noir*, *rouge*, *bleu* and *vert*.

The table boxes must have the following specifications:

Height: 40 pixels.
Width: 80 pixels.
Background color: a color shade that matches the column.

The table must display the shade of each color as to obtain a 50 lines shading.

You must observe the following instructions:

- Your template must `NOT` include the colors in hard. The different shades must be generated in a view and must all be different.
- A total of 4 tag couples `<td></td>` are authorized in your templates.
- A total of 4 tag couples `<th></th>` are authorized in your templates.
- A table must be formatted as follows:
  - A tag couple `<th></th>` per box for the column names.
  - A tag couple `<tr></tr>` per color shade line.
  - A tag couple `<td></td>` per color shade box.

Chapter VIII

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.

`The evaluation process will happen on the computer of the evaluated group.`