Daniel Avery Nisbet

Search Engine for Software Packages

5/1/2018

User Interface Documentation – The Essentials

Setting up the User Interface:

1. The easiest way to setup the user interface locally is to use Anaconda. You can do so at https://conda.io/docs/user-guide/install/download.html.
2. The interface uses Django 2.0 which runs on Python 3. To create a python 3.6 environment that has the required dependencies, enter the following commands into your console:
    a. conda create --name django_20env python=3.6
    b. conda activate django_20env
    c. conda install -n django_20env -c conda-forge Django
    d. source activate django_env

    These commands create a virtual environment named django_20env, and installs Django 2.0. for more information about managing virtual environments using anaconda, see https://conda.io/docs/user-guide/tasks/manage-environments.html. After all, it may be slightly different depending on your operating system.
3. Clone the repository from https://github.com/vivianbuan/Search-Engine-for-Software.
4. Switch to the user_interface branch
5. Navigate to /mysite and run the command:
    a. python manage.py runserver

    If you're on your local machine, the default address that it will start the server on is http://127.0.0.1:8000/. More details can be found at https://docs.djangoproject.com/en/2.0/intro/tutorial01/
6. Now you should be able to access the site at http://127.0.0.1:8000/ from your web browser.

Brief Walkthrough the Codebase:

1. The codebase can be found at https://github.com/vivianbuan/Search-Engine-for-Software, under the branch user_interface.
2. Search-Engine-for-Software/mysite contains the manage.py file needed to run the server.
3. Search-Engine-for-Software/mysite/search_engine contains pretty much all the files one would want to work with on a regular basis. In this directory you can find:
    a. views.py, which contains the endpoints and helper functions for loading data from the indexer and rendering the webpage.
    b. /templates/search_engine, which contains all the html template files. Any file preceded by an underscore, for example _information_pane.html, is a partial template – it holds just a single component on the page. The page is modularized in this way that results.html contains several partial templates.
    c. /static/, which contains the modules that are used, most of which are used site-wide

    d. /static/search engine, which has the CSS documents matching to each template, as well as results.js which contains the JavaScript code for the main page. The JavaScript code is not modularized like the HTML and CSS. The images used on the site, such as the SwaGo logo, are also found here.
4. Search-Engine-for-Software/mysite/~ contains another path to the static files. I'm not sure why, but this got Apache to link the static files to the site so that Google Compute Engine could render the pages with static files.
5. Search-Engine-for-Software/mysite/conf contains Apache configuration files for Google Compute Engine.
6. Search-Engine-for-Software/mysite/mysite contains a few important files. urls.py is needed to add any more pages to the site, and settings.py contains all Django relevant settings, such as what URLs it will run from, and the directory to access static files.