

Despite knowing that you know how to use and run a Django project, I would rather let you some instructions here:

## 1. How to access the website

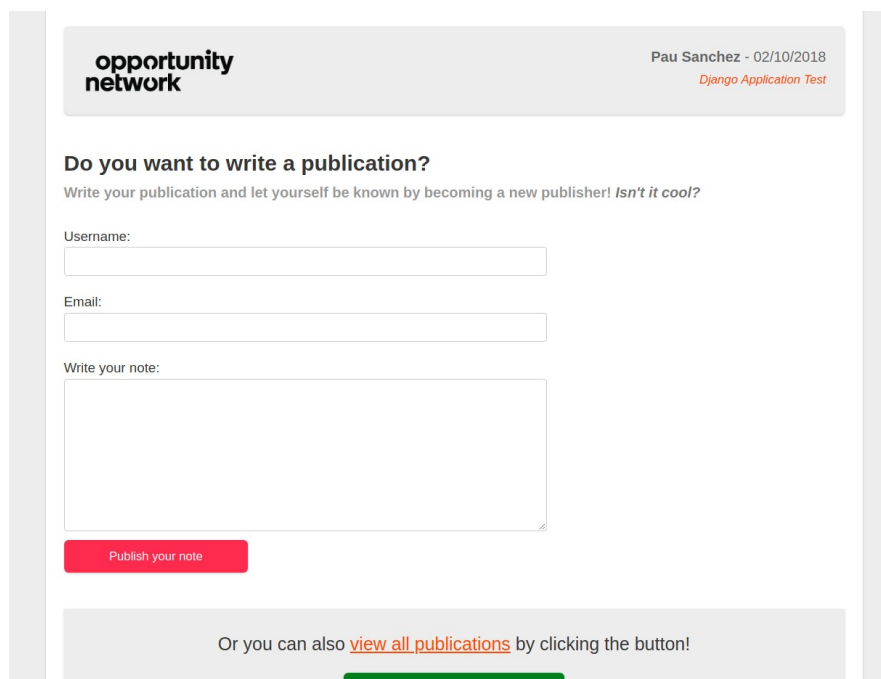
1) Go to this url and download or clone the project: <https://github.com/pausanchezv/opportunity-network-test>

2) Place yourself in the right directory  
>> cd opportunity-network-test/pausanchez

3) Run the Django Server  
>> python manage.py runserver 0.0.0.0:8000

4) Access the URL: <http://localhost:8000/test/>

- Now you should see the following website:



The screenshot shows a web application interface for 'opportunity network'. The header includes the site name and a user profile for 'Pau Sanchez - 02/10/2018' with a 'Django Application Test' status. The main content area prompts the user to 'Do you want to write a publication?' with the subtext 'Write your publication and let yourself be known by becoming a new publisher! Isn't it cool?'. Below this, there are input fields for 'Username:', 'Email:', and a larger text area for 'Write your note:'. A red button labeled 'Publish your note' is positioned below the text area. At the bottom, a grey bar contains the text 'Or you can also [view all publications](#) by clicking the button!' and a green button.

## 2. How to access Admin

A Django administration page has been created in order to view and manipulate the database objects. This admin can be accessed by following this link <http://localhost:8000/admin/>

Once there you will be asked to put an username and password. Use these:

Username: **test**

Password: **1qa2ws3ed**

### 3. How it works

The website is based on two pages (2 html templates) which represent the View of the MVC Framework. The first one, which is called *index.html*, is used as a landing page and here is where users can post their publications. This template is accessible via http GET or POST.

When the user first arrives at the website it is accessed via GET, but on the other hand, when the form is executed, this view is called via POST so as to make the changes in the database (in this case, it inserts a new publication into the database).

In this scenery, a page redirect is made to send the user to another template: 'publications.html', where it can be seen all the publications. And here is where theoretically this test would end.

Nevertheless, another functionality has been implemented: publications can be deleted without refreshing the page: using AJAX. To do that, a JavaScript function can be found inside the file 'publications.html', and it is through this file that an asynchronous call is made from client to server, passing the current ID as an argument and making the server capable of deleting a publication directly from the database. As a return, the client gets the publication's id to remove this piece of html from the DOM, in real time. Everything can be easily seen in the code.

Eventually, the file *views.py* depicts the controller of the MVC, and *models.py* is the model.

### 4. A little more

- Forms: a new file 'forms.py' has been added to handle the forms directly from Django
- Templates: django 'include' and 'extends' tags have been used so as not to repeat unnecessary code.
- Static: a new package called static has been created to store the static stuff such as css, images or JavaScript files
- CSS: a bit of CSS has been created to make the website a little more user friendly (Bootstrap has not been used, everything has been created from zero)
- jQuery: it has been imported to be able to use AJAX in a comfortable way

Should you require more information, do not hesitate to contact me.

*Pau Sanchez Valdivieso – 02-10-2018*