Best practices in Django Rest Framework

glemmaPaul

Published with GitBook

Table of Contents

Best practices in Django Rest Framework

This books gives you a clear guide in making RESTful API's through Django and Django Rest Framework.

How this started

Hi there. I'm Paul Oostenrijk, a Back-end developer located in New York. Originally I'm from Holland, but have chosen to persue a bigger dream in New York.

I'm always happy to learn people what I have learned in the years that I have been developing. This brought me on the idea for creating an open sourced book about Django Rest Framework and creating RESTful API's.

For whom is this book?

This book is meant for people that currently work as a Back-end developer or are just starting to learn. This book is also as a reference in discussions. My ultimate dream will be that people are going to contribute to this book with their experiences and knowledge. Also my grammar can be off, so please help me make this book a lesson for my bad grammar skills.

Creating RESTful API's looks like an easy thing, but many people underestimate it a lot. This is why we should continue to standardize practices together. To come with one unified solution to most of our REST related issues.

Who can contribute?

Anyone! Create a Pull Request and we all will review!

Getting everything up and running

Most of you know that Django REST Framework is made in Python. A lot of you also know that DRF is made on top of Django. If you are familiar with setting up a DRF project you can just skip this chapter. If you are not, you should definitely try to follow these instructions.

Setting up your project structure

The first thing we are going to discuss is the project structure. Like any other Django app we can work in apps. The project structure I prefer is fairly easy.

```
- repository
    - project
        - app1
            __init__.py
            models.py
            etc.py
        - app2
        - config
        - manage.py
    - .gitignore
    - README.md
```

This project structure keeps all the README's and gitignores away from the code itself.

Create the project with cookiecutter

Have you ever used it? This magic thing called `cookiecutter`, which makes bootstrapping a project feel like a breeze? With cookiecutter you don't have to worry about manually creating a project.

To install cookiecutter you can read the instructions here:
https://github.com/audreyr/cookiecutter

After having installed cookiecutter on your machine you can create a new project structure with one of the thousands cookiecutter project setups in the opensource world. I personally love to use the cookiecutter made by my buddy Andrew Conti https://github.com/agconti/cookiecutter-django-rest. Big ups to Andrew for creating such a lean but useful cookiecutter setup!

Creating your Virtual Machine

Python applications normally run sandboxed in a Virtual Machine. This to prevent other projects from having problems with each others dependencies. Look, when we have multiple projects. For example, project `Cheese` has a dependency called `Milk` and uses the version 1.0. On the other hand we have `Oatmeal` and it uses `Milk` as well, but `Oatmeal` is way more recent and uses version 2.0.

When we didn't have any Virtual machine we would have a problem, because we cannot have multiple versions installed on one machine. That's why we use a virtual machine to `sandbox` our dependencies. `TODO actually write about creating the virtual machine`

Run the app

Well here we go, let's spin up this monster. `TODO write about running the app`

What the H!@#!ck am I building?

Wow wow wow, hold on there soldier, you thought you were starting already? Not yet! We're first going to figure out what we're actually building. But no worries, it's not that hard and it doesn't take that long. It's just good to know beforehand what you are building.

Step 1: Figure out how to document

Yes, it sounds strange. Figuring out how you are going to document your documentation. There are million ways of creating, even billion ways of publishing. What you want to have is an easy readable and accessible place for your docs. It can be a word doc on Dropbox to a Github Page with Markdown

Step 2: Know what you are building

We all know from G.I. Joe that knowing is half the battle. This is why we ask ourselves, what are we exactly building? What are we achieving or what do we want to solve with our API? By listing out what your application is able to do, you have a better and clearer insight of how you would want your end result to be like. Do this as high level as you can.

Step 3: Figure out your models

True, this doesn't have anything to do with Django Rest framework. It is good to know that you can use those insights from step 2 and take that into your model structure.

Step 4: App structure

App structure is an important decision in the process of creating

Step 5: Review with the team