The Backend: Django and REST

Reindert-Jan Ekker

@rjekker www.rjekker.nl

Summary

Starting a Django App

Models and a Database

Django Admin

Django-Rest-Framework

A Django App and Models

Demo

Creating a Django App

Adding Model Classes

Running Database Migrations

```
$ python manage.py startapp scrumboard


in settings.py:

INSTALLED_APPS = ( ..., ‘scrumboard’ )
```

Adding a Django App

Creates a new Python package “Scrumboard”

App has its own models, urls, views

Add name of app to `INSTALLED_APPS` in `settings.py`

```
in models.py:

from django.db import models

class Card(models.Model):

    title = models.CharField(max_length=100)

    description = models.TextField(blank=True)
```

Django Model Classes

Inherit from django.db.models.Model

Model classes are mapped to database tables, instances to rows

Fields are mapped to columns

Field classes: https://goo.gl/ef55le

```
$ python manage.py makemigrations

$ python manage.py migrate
```

Running a Django Migration

Will update the database according to changes in Model classes

Generate script to update database

Run the script

The Django Admin Interface

Demo

Creating a Superuser

Adding Model Classes to Admin

Using the Admin Interface to Add Data

```
$ python manage.py createsuperuser


In admin.py:

admin.site.register(Card)
```

Enabling the Admin Interface

Superuser is needed for logging into admin

Call admin.site.register in `admin.py` for each Model class

Admin interface is located under `/admin`

Adding a REST Service

Demo

Installing django-rest-framework

Adding Serializers

Adding API Views

Configuring URLS

```
$ pip install djangorestframework


In settings.py:

add ‘rest_framework’ to INSTALLED_APPS
```

Installing django-rest-framework

```
class CardSerializer(serializers.ModelSerializer):

    class Meta:
        model = Card
```

Serializers

Encode Python objects as JSON (JavaScript Object Notation)

ModelSerializer autogenerates JSON based on the Model fields

```
class CardApi(ListAPIView):
    queryset = Card.objects.all()
    serializer_class = CardSerializer
```

REST API Views

Will handle requests for data from the client

Inherit from generic view classes

We need lists of objects so we use ListAPIView

```
in scrumboard/urls.py:


urlpatterns = [
    url(r'^lists$', ListApi.as_view()),
    url(r'^cards$', CardApi.as_view())
]
```

Configuring API URLS 1

Function `url()` maps views to urls

First argument is a Regular Expression

Second argument is the view. For view classes we call `.as_view()`

These URL are relative to the URL for the app

```
in djangular/urls.py:


urlpatterns = [

    url(r'^admin/', admin.site.urls),
    url(r'^scrumboard/', include('scrumboard.urls')),

]
```

Configuring API URLS 2

Include the URLs for an app at a specific path

Match start of url: “`scrumboard/`”. Rest is matched by `urls.py` in app.

Use `include()` with the name of the url module as a string.

Summary

Starting a Django App

Models and a Database

Django Admin

Django-Rest-Framework