These are the commands for the follow-along exercises in Lesson 6.2.

6.202: Setting up an image store

terminal

```
workon advanced_web_dev
cd topic6
django-admin startproject image_store
cd image_store
python manage.py startapp store
mkdir store/templates
touch store/templates/index.html
touch image_store/celery.py
```

celery.py

```
from __future__ import absolute_import

import os
import time

from celery import Celery
from django.conf import settings

os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'image_store.settings')
app = Celery('image_store', broker='redis://localhost/',
backend='redis://localhost/')
app.config_from_object('django.conf:settings', namespace='CELERY')
app.autodiscover_tasks()
```

/image_store/__init__.py

```
from .celery import app as celery_app
__all__ = ('celery_app',)
```

create

```
image_store/store/tasks.py
image_store/store/api.py
image_store/store/serializers.py
image_store/store/urls.py
mkdir image_store/images/
```

terminal

```
celery --app=image_store.celery:app worker --loglevel=INFO --
pidfile=celery.pid
```

6.204: Building the image app skeleton

models.py

```
class Image(models.Model):
    name = models.CharField(max_length=256, unique=True,
db_index=True)
    image = models.FileField(blank=False)

    def __str__(self):
        return self.name
```

views.py

```
def index(request):
    return render(request, 'index.html')
```

api.py

```
from rest_framework import generics
from rest_framework import mixins
from .models import *
from .serializers import *
from django.http import HttpResponseRedirect


class ImageDetail(mixins.CreateModelMixin,
generics.GenericAPIView):
    queryset = Image.objects.all()
    serializer_class = ImageSerializer

    def create(self, request, *args, **kwargs):
        response = super(ImageDetail, self).create(request, *args,
**kwargs)
        return HttpResponseRedirect(redirect_to='/test')

    def post(self, request, *args, **kwargs):
        return self.create(request, *args, **kwargs)


class ImageList(generics.ListAPIView):
    queryset = Image.objects.all()
    serializer_class = ImageListSerializer
```

serializers.py

```
from rest_framework import serializers
from .models import *


class ImageSerializer(serializers.ModelSerializer):
    class Meta:
        model = Image
        fields = ['name', 'image']


class ImageListSerializer(serializers.ModelSerializer):
    class Meta:
        model = Image
        fields = ['name', 'image']
```

image_store/urls.py

```
from django.urls import path, include
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('', include('store.urls')),
    path('admin/', admin.site.urls),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

/store/urls.py

```
from django.urls import include, path
from . import views
from . import api

urlpatterns = [
    path('', views.index, name='index'),
    path('api/image/', api.ImageDetail.as_view(),
name="image_api"),
    path('api/images/', api.ImageList.as_view(),
name="image_api"),
]
```

settings.py

```
MEDIA_ROOT = 'images'
MEDIA_URL = '/images/'

INSTALLED_APPS = [
      'store.apps.StoreConfig',
      'rest_framework',

ALLOWED_HOSTS = [
    'localhost',
    '127.0.0.1',
    '.coursera-apps.org',
]
```

terminal

```
python manage.py makemigrations
python manage.py migrate
```

6.205: Frontend and Celery task

store/templates/index.html

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <div id="images"></div>
  <div id="input">
```

```
    <form action="/api/image/" method="post" class="form"
enctype="multipart/form-data">
        {% csrf_token %}
        <label for="name">Name:</label><br>
        <input type="text" id="name" name="name"><br><br>
        <label for="image">Upload Image:</label><br>
        <input type="file" id="image" name="image"><br><br>
        <input type="submit" value="Submit">
    </form>
  </div>
</body>
</html>
```

store/templates/index.html

```
<head>
  <script>
  setInterval(function() {
    console.log("Hey");
    const req = new XMLHttpRequest();
    req.onreadystatechange = function(){};
    req.open("GET", "/api/images/");
    req.send();
    }, 5000);
  </script>
</head>
```

store/templates/index.html

```
req.onreadystatechange = function(){
  if (req.readyState === 4) {
    console.log("Got Data")
    html = "<table><tr>";
    data = JSON.parse(req.response)
    data.forEach(function(obj)
    { html += "<td>"+obj.name+"<br />";
      html += "<img src=\""+obj.image+"\"></td>" });
    html += "</tr></table>";
    document.getElementById("images").innerHTML = html;
  }}
```

terminal

```
python manage.py runserver 127.0.0.1:8080
python manage.py flush
rm images/*
```

6.206: Adding Celery tasks to the image store

models.py

```
    thumbnail = models.FileField(null=True)
```

terminal

```
python manage.py makemigrations
python manage.py migrate
```

apis.py

```
from .tasks import *

def perform_create(self, serializer):
    record = serializer.save()
    make_thumbnail.delay(record.pk)
```

tasks.py

```
from celery import shared_task
from .models import *
from PIL import Image as img
import io
from django.core.files.uploadedfile import SimpleUploadedFile

@shared_task
def make_thumbnail(record_pk):
    record = Image.objects.get(pk=record_pk)

    image = img.open('images/'+str(record.image))
    x_scale_factor = image.size[0]/100
    thumbnail = image.resize((100,
int(image.size[1]/x_scale_factor)))
    thumbnail.save("test.jpg")

    byteArr = io.BytesIO()
    thumbnail.save(byteArr, format='jpeg')
    file = SimpleUploadedFile("thumb_"+str(record.image),
byteArr.getvalue())
    record.thumbnail = file
    record.save()
```

serialisder.py

```
class ImageListSerializer(serializers.ModelSerializer):
    class Meta:
        model = Image
        fields = ['name', 'image', 'thumbnail']
```

index.html

```
html += "<img src=\""+obj.thumbnail+"\"></td>" });
```