极客时间

Django 开发实战 - 第五章

极客时间

扫码试看/订阅

《 Django 快速开发实战》视频课程

遗留系统集成：为已有数据库生成管理后台

- 问题

  已经有内部系统在运行了，缺少管理功能，希望能有一个权利后台。

  比如 人事系统，CRM，ERP 的产品，缺少部分数据的维护功能

- 诉求

  - 3分钟生成一个管理后台；
  - 可以灵活定制页面；
  - 不影响正在运行的业务系统。

为已有数据库生成管理后台

- 创建项目: $ django-admin startproject empmanager
- 编辑 settings.py 中的数据库配置， vim ~/settings.py

```
DATABASES = {

'default’: {

   'ENGINE': 'django.db.backends.mysql',    'NAME': 'mydatabase',

   'USER': 'mydatabaseuser',    'PASSWORD': 'mypassword',    'HOST': '127.0.0.1', 'PORT': '5432',

   }

}
```

- 生成 model 类: ./manage.py inspectdb > models.py

Django 的中间件（Middleware）

- 什么是中间件 Middleware？
  - 注入在 Django **请求/响应** 处理流程中的钩子框架，能对 request/response 作处理
- 广泛的使用场景
  - 登录认证，安全拦截
  - 日志记录，性能上报
  - 缓存处理，监控告警....
- 自定义中间件的 2 种方法
  - 使用函数实现
  - 使用类实现

Django的中间件（Middleware）：函数实现

```python
def simple_middleware(get_response):
    # One-time configuration and initialization.

    def middleware(request):
        # Code to be executed for each request before
        # the view (and later middleware) are called.

        response = get_response(request)

        # Code to be executed for each request/response after
        # the view is called.

        return response

    return middleware
```

Django的中间件（Middleware）：类实现

- Django 提供的 get_response 方法：
- 可能是一个真实的视图，也可能是请求处理链中的下一个中间件

```python
class SimpleMiddleware:
    def __init__(self, get_response):
        self.get_response = get_response
        # One-time configuration and initialization.

    def __call__(self, request):
        # Code to be executed for each request before
        # the view (and later middleware) are called.

        response = self.get_response(request)

        # Code to be executed for each request/response after
        # the view is called.

        return response
```

创建一个请求日志，性能日志记录中间件

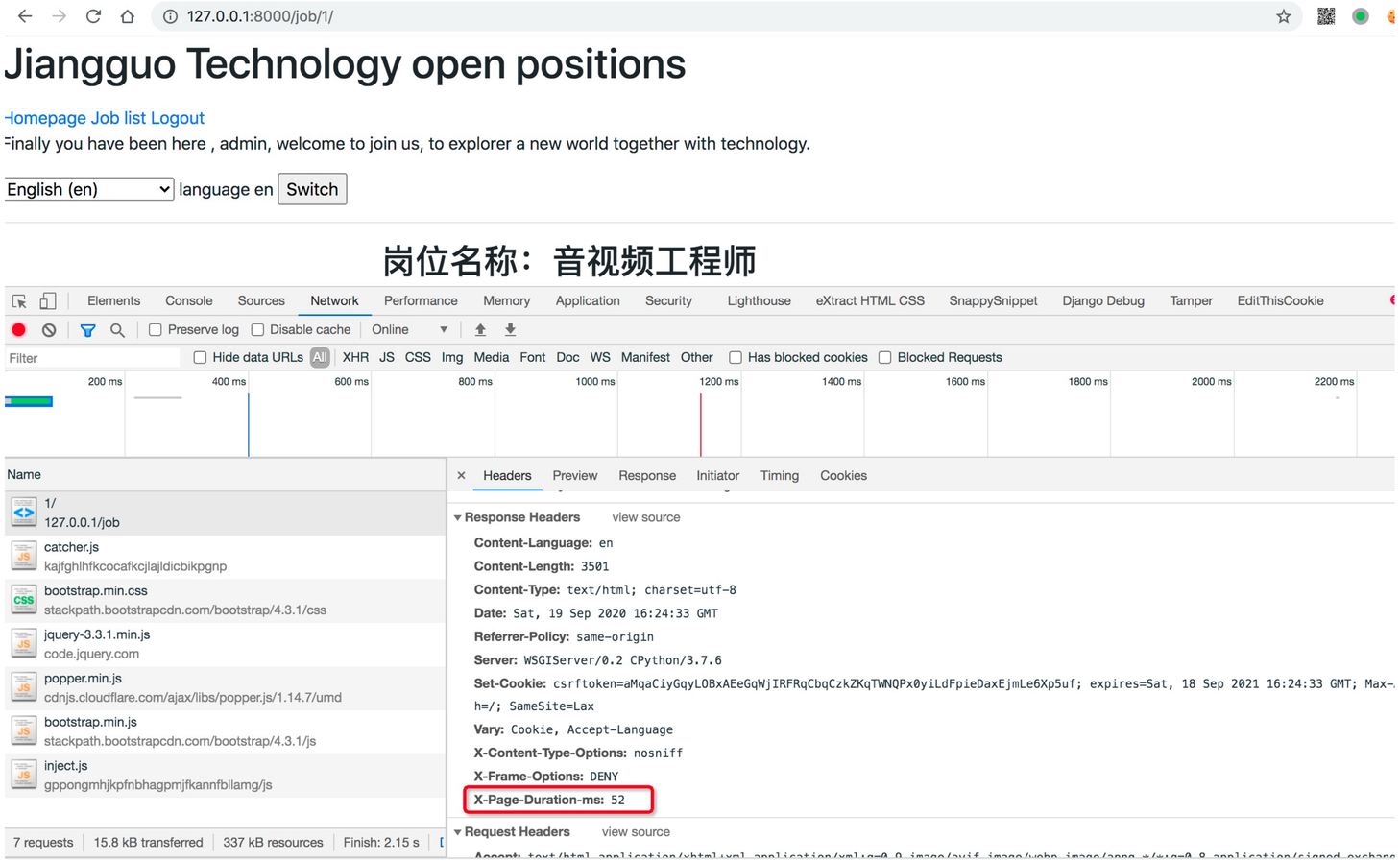
- 定义实现中间件: def performance_logger_middleware(get_response)
- 记录请求 URL， 参数， 响应时间
- 注册 middleware 到 settings 中
- 配置 日志文件路径

performance.py

```python
import time
import logging

logger = logging.getLogger(__name__)


def performance_logger_middleware(get_response):
    def middleware(request):
        start_time = time.time()
        response = get_response(request)
        duration = time.time() - start_time
        response["X-Page-Duration-ms"] = int(duration * 1000)
        logger.info("%s %s %s", duration, request.path, request.GET.dict())
        return response

    return middleware
```

性能记录

- 输出到 Response Header 里面的 X-Page-Ducration-ms



极客时间

扫码试看/订阅

《 Django 快速开发实战》视频课程