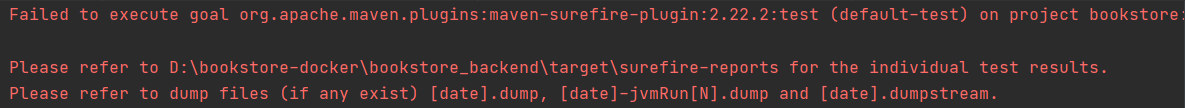
使用maven package进行打包

打包时发现报错：



于是在pom.xml的plugins里面设置跳过测试

文本

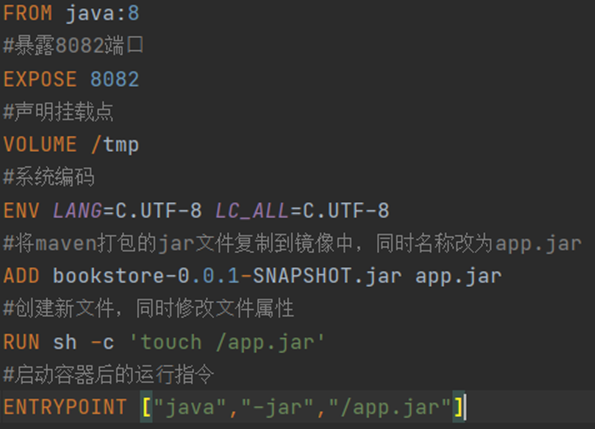
描述已自动生成

成功将bookstore项目打包

文本

中度可信度描述已自动生成

在pom.xml相同目录下新建一个Dockerfile：

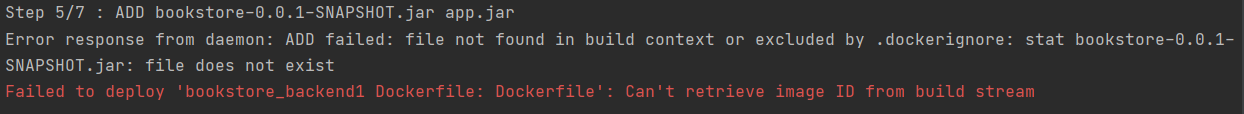


在Edit Configurations中配置：

图形用户界面, 应用程序

描述已自动生成

运行后发现报错：



发现Dockerfile中jar文件名字与实际打包出来的jar文件名字不一致，于是修改dockerfile：

文本

描述已自动生成

成功放进docker：

图形用户界面, 应用程序

描述已自动生成

使用postman测试，可以成功处理请求：

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

**Redis：**

docker pull redis 拉取redis镜像

配置redis.conf

注释掉bind 127.0.0.1

protected-mode yes改为no

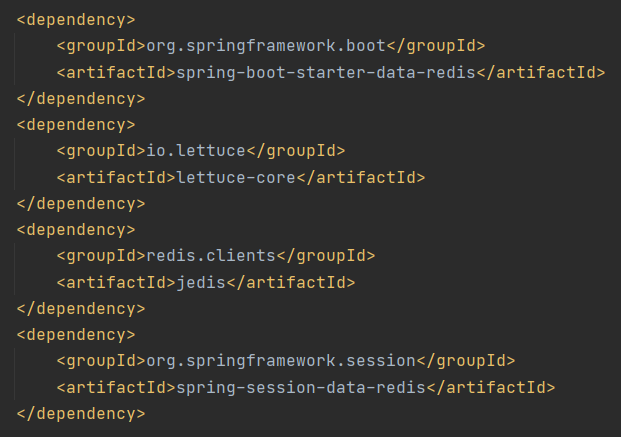
appendonly yes

docker run -p 6379:6379 --name redis -v D:/docker/redis/redis.conf:/etc/redis/redis.conf -v D:/docker/redis/data:/data -d redis redis-server /etc/redis/redis.conf --appendonly yes 在docker中创建一个redis实例

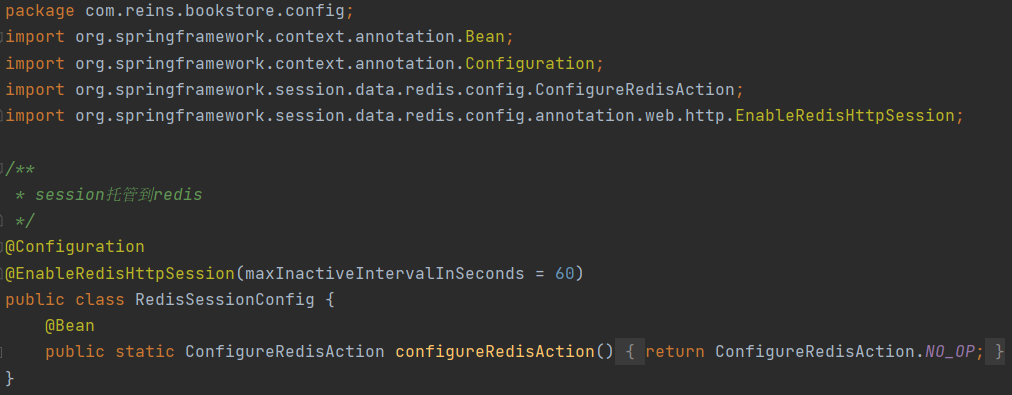


使用docker inspect redis 查看其IPaddress

在bookstore的pom文件里引入：



增加配置类：



在application.properties中增加：

文本

描述已自动生成

（172.17.0.2为redis在docker中的ip地址）

重新打包bookstore并将其部署到docker中。

**nginx：**

docker pull nginx 拉取nginx镜像

新建一个nginx.conf配置文件，重点配置：

（通过docker inspect bookstore查看两个实例对应的容器中的ip，后面跟的是映射到容器中的端口）

文本, 信件

描述已自动生成

docker run -p 8080:8080 --name nginx -v D:/docker/nginx.conf:/etc/nginx/nginx.conf -d nginx 在docker中创建一个nginx实例。

图形用户界面, 应用程序

描述已自动生成

此时bookstore1跑在8081，bookstore2跑在8082，nginx跑在8080。

通过8080端口访问test接口可以看到nginx进行了默认的负载均衡：

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

（可以看到，再次发送请求时，转由bookstore2处理请求，但是sessionid并没有改变。）