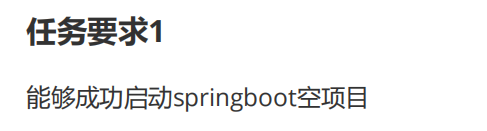
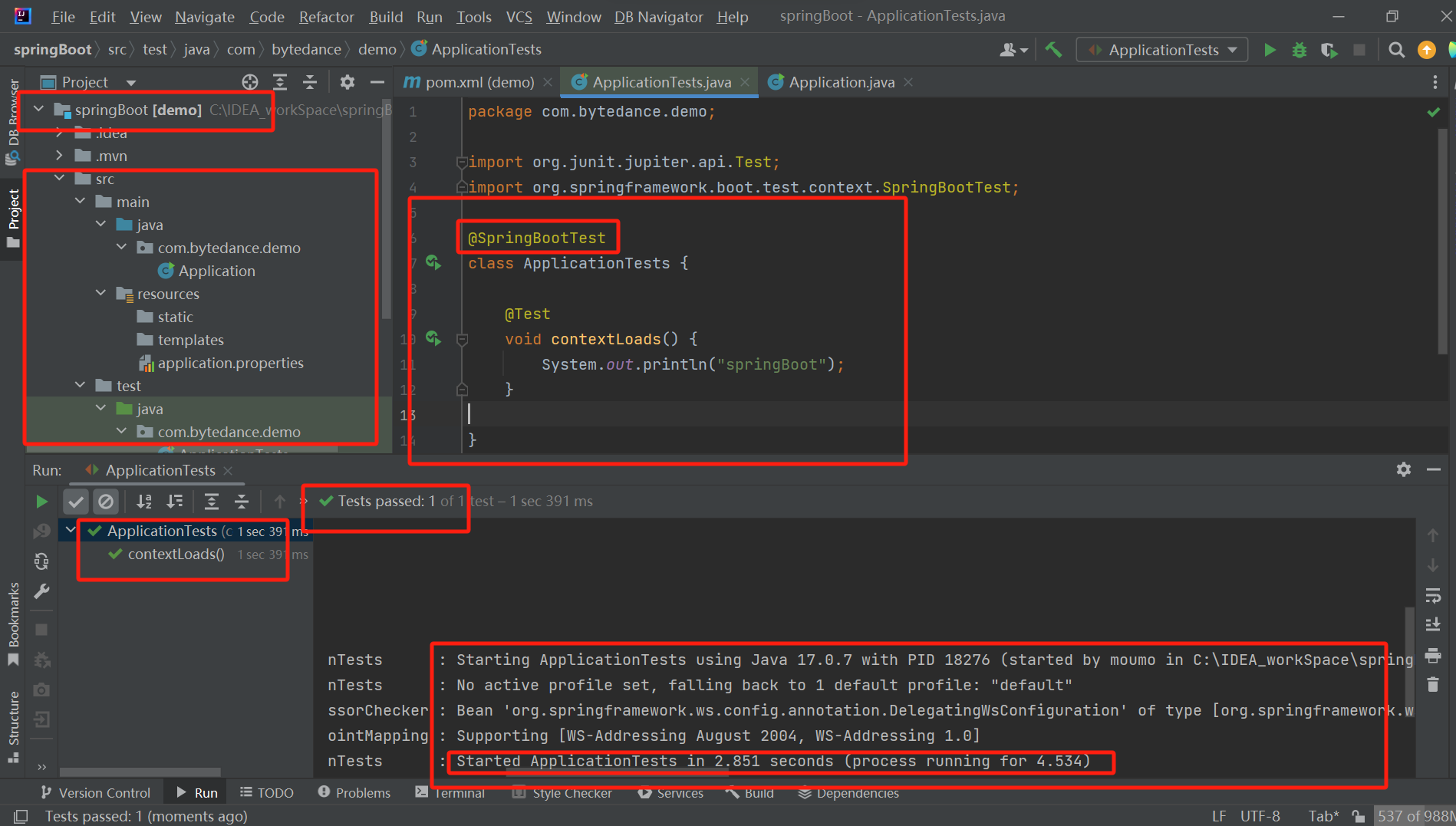
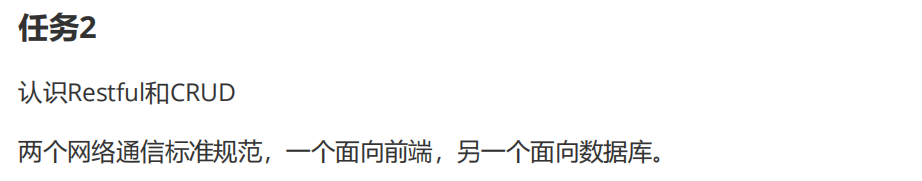
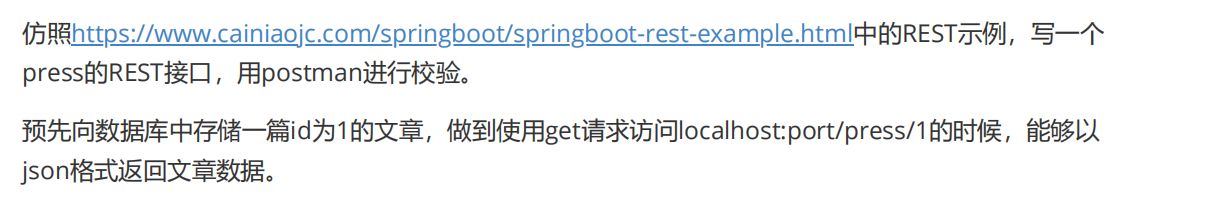
SpringBoot



1. 启动连接/生成初始项目包：<https://start.spring.io/>
2. SpringBoot 教程：<https://www.cainiaojc.com/springboot/springboot-tutorial.html>
3. 启动空springboot空项目:
   1. Output







1. RESTful 架构详解/...什么是RESTful? .../... ：

<https://www.runoob.com/w3cnote/restful-architecture.html>

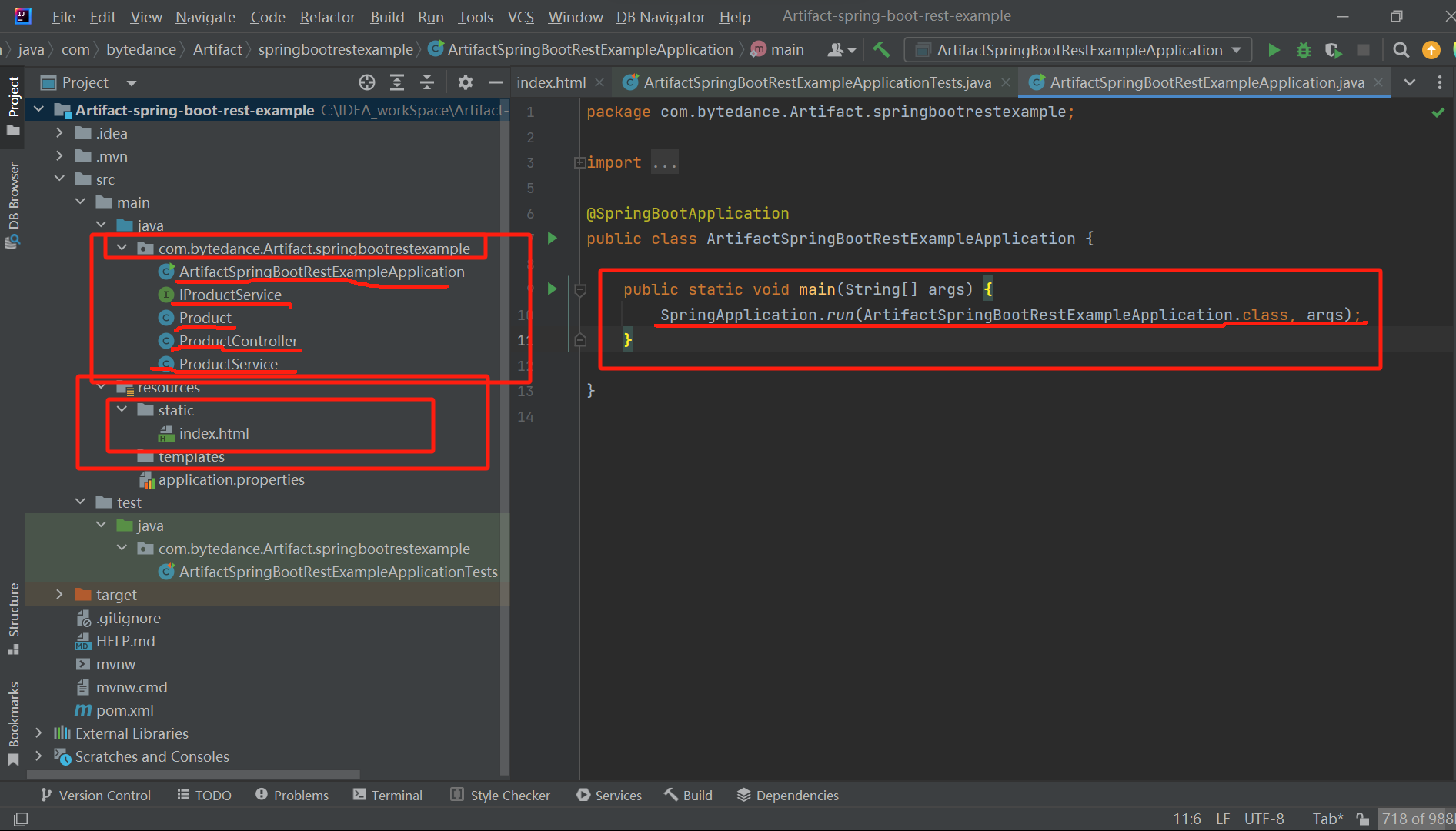
1. crud/CRUD - (Create, Read, Update, Delete):

<https://baike.baidu.com/item/crud/3606157?fr=aladdin>

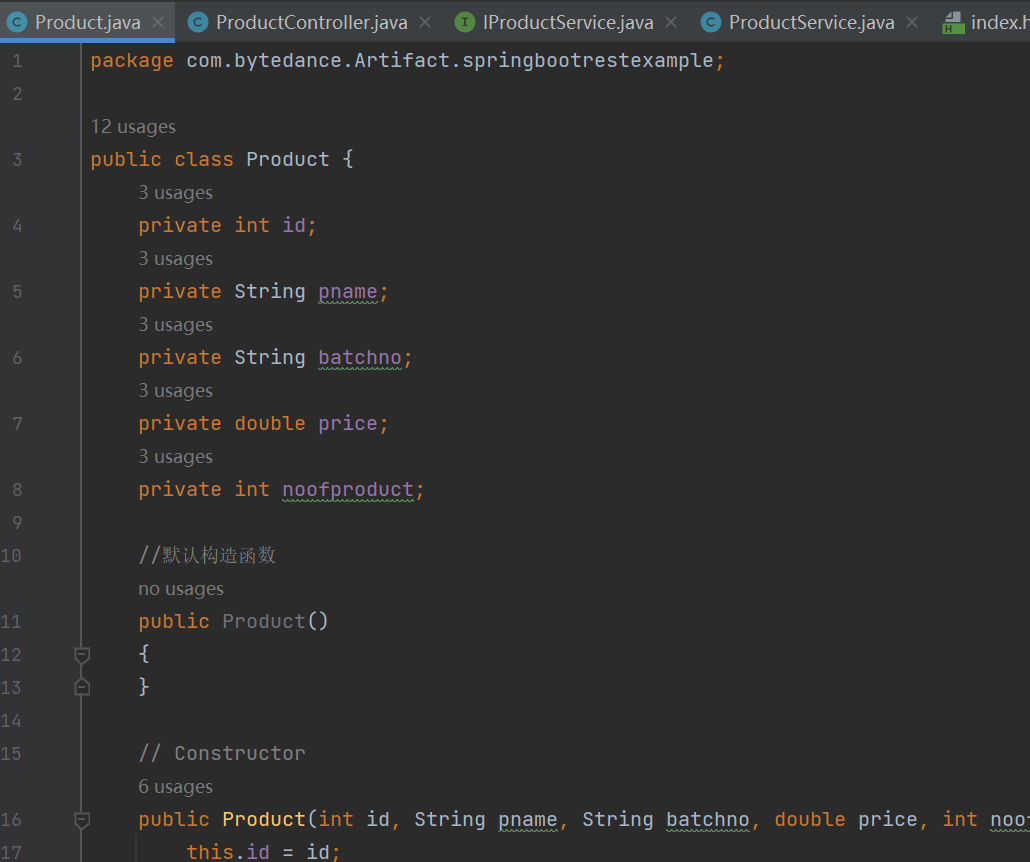
1. 复现“https://www.cainiaojc.com/springboot/springboot-rest-example.html”中的REST示例：

(Using using an in-memory list instead of using a database like MySQL)

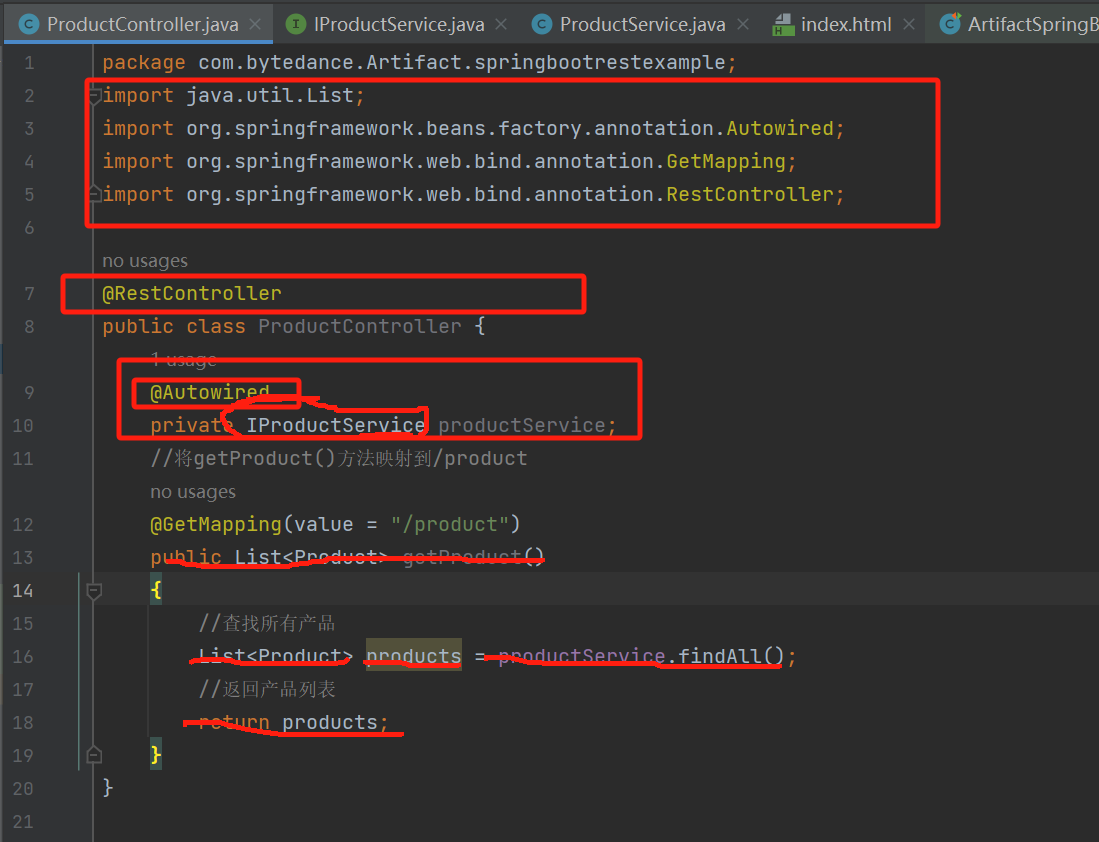
* 1. In ‘C:\IDEA\_workSpace\Artifact-spring-boot-rest-example’:



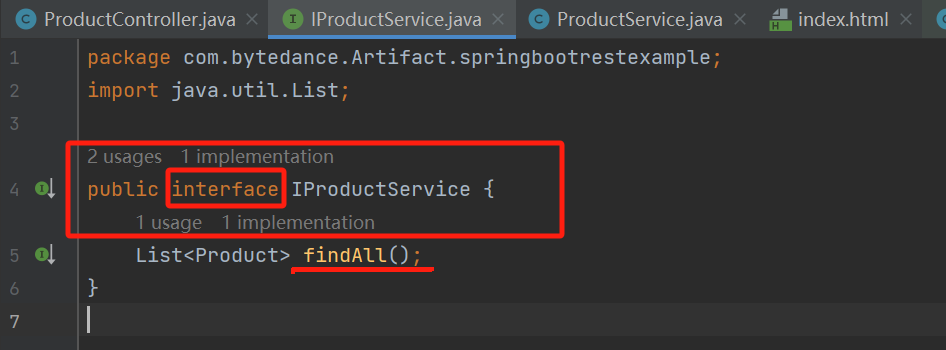
* 1. Product.java:



* 1. ProductController.java:

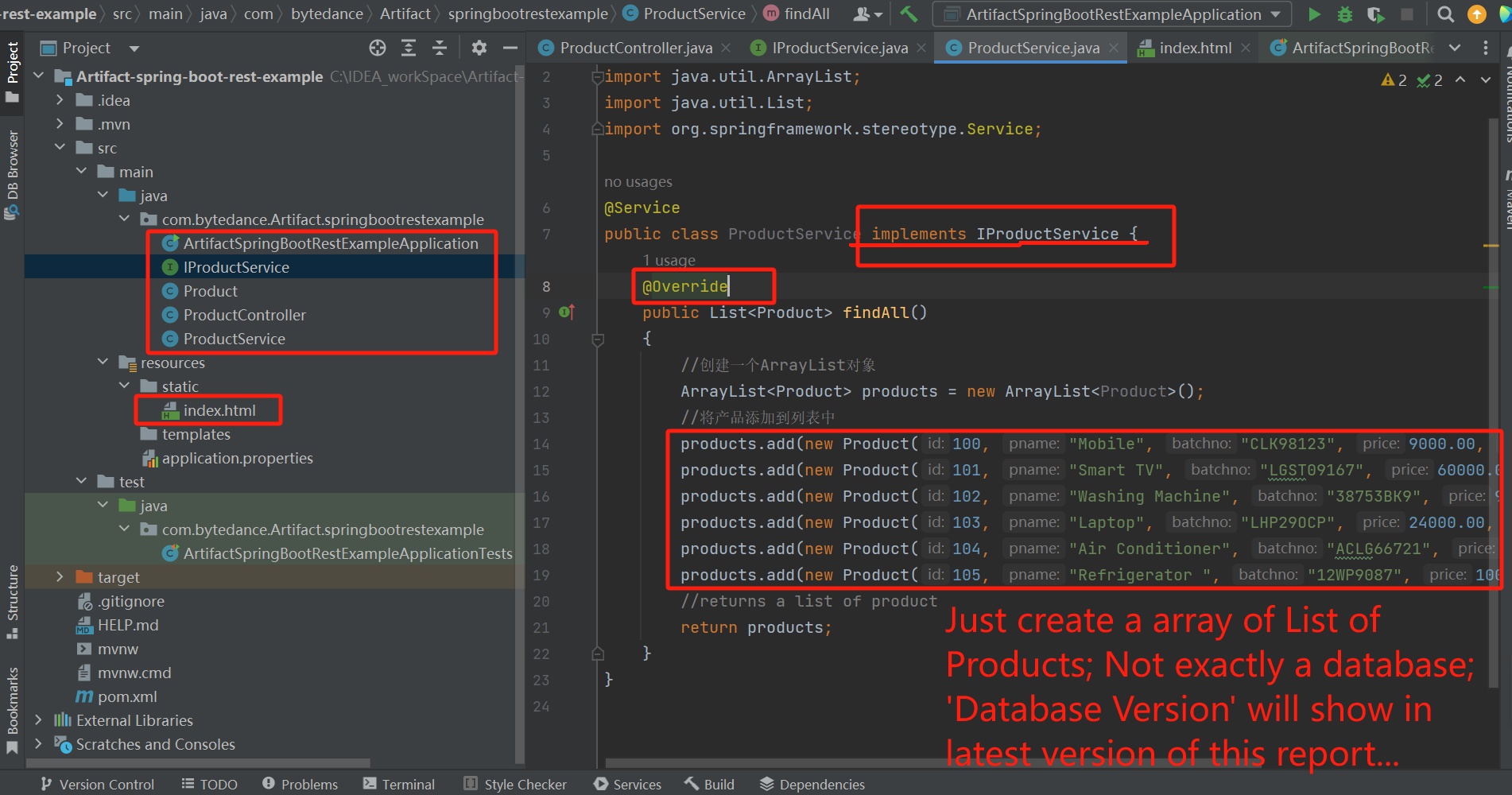


* 1. IProductService(Interface\_Product\_Service):

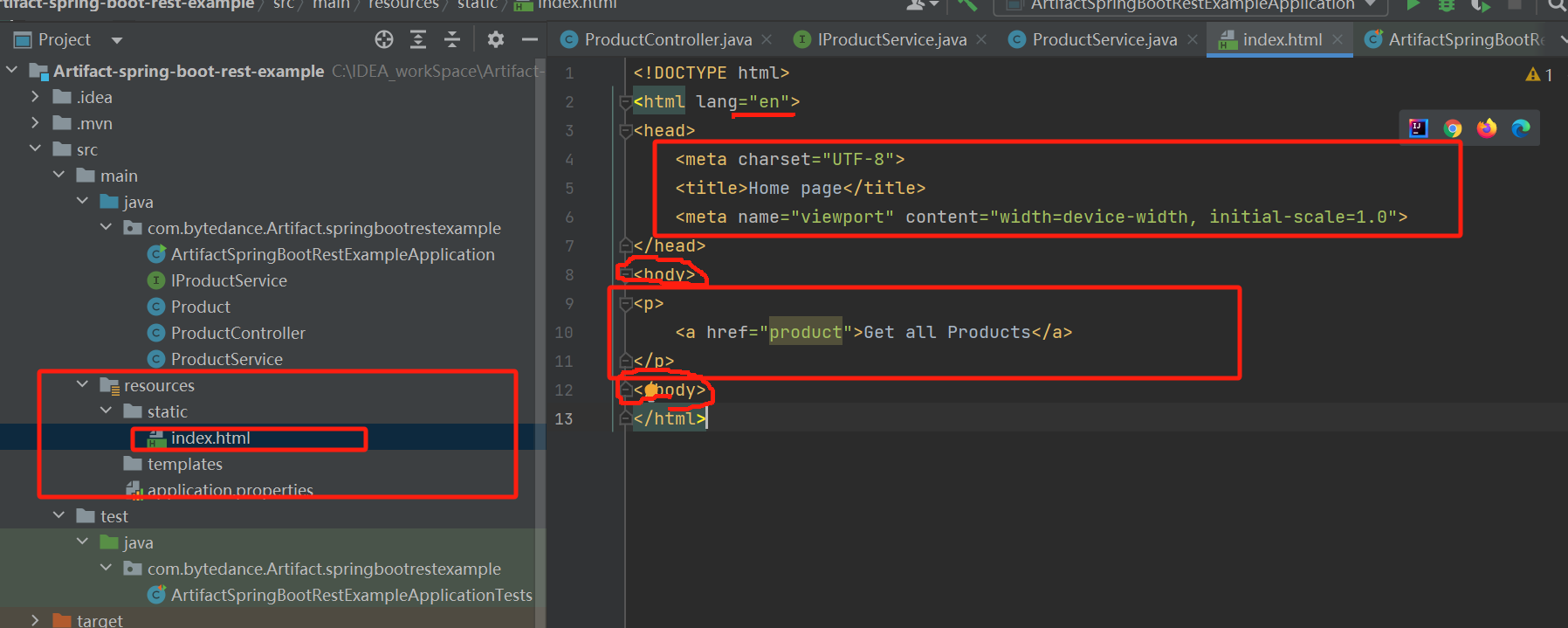


* 1. ProductService:

(Using using an in-memory list instead of using a database like MySQL)

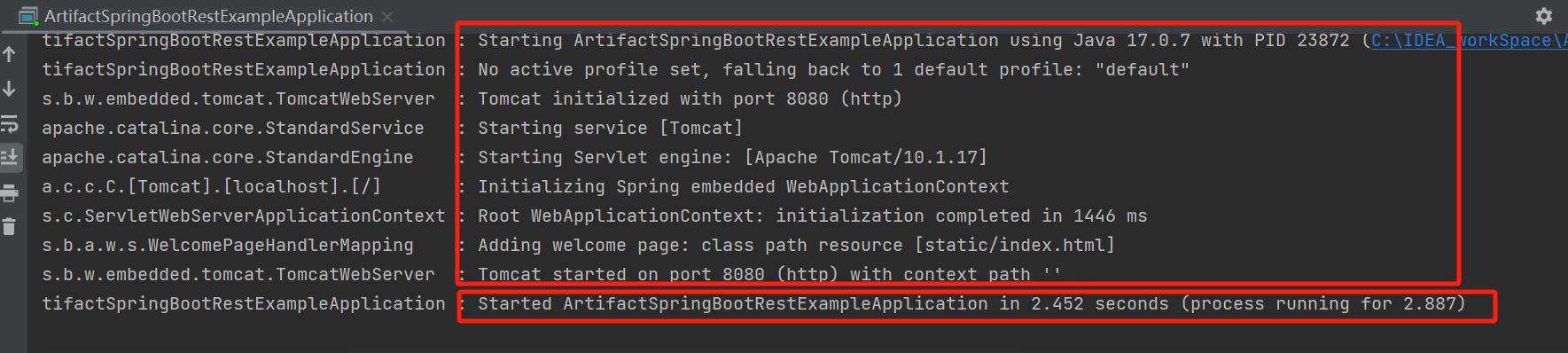


* 1. Index.html:



* 1. Output:

• Terminal



• 打开浏览器并调用 URL http://localhost:8080/index.html。它显示了 获取所有产品的链接，如下图所示:

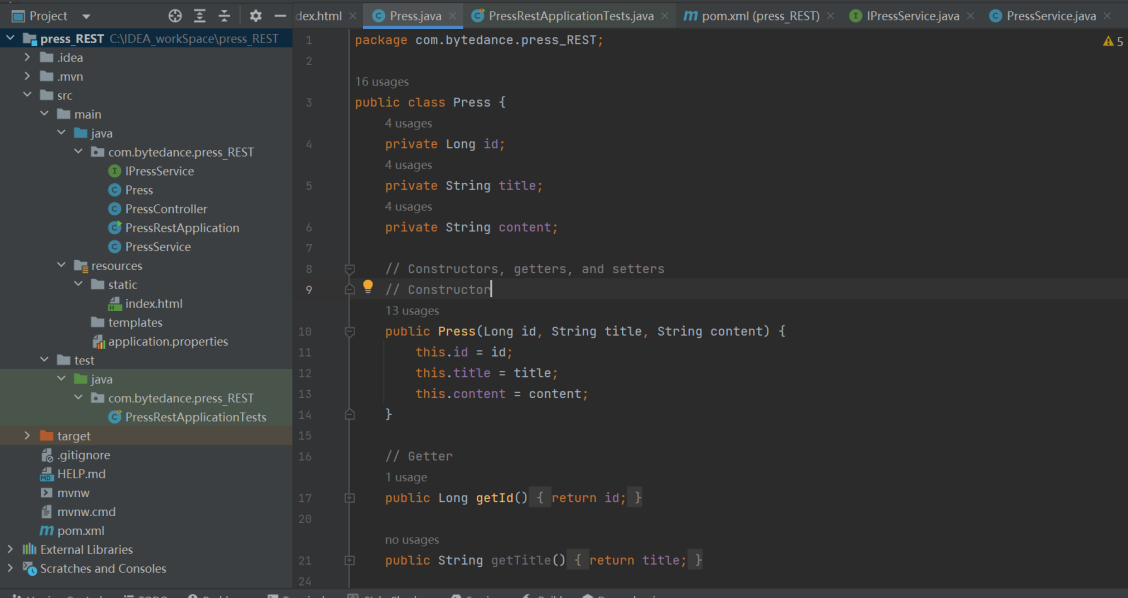


1. 仿照https://www.cainiaojc.com/springboot/springboot-rest-example.html中的REST示例，写一个press的REST接口，用postman进行校验(疑问：如何‘用postman进行校验？’，是类似于crul...命令吗？)：

(Using using an in-memory list instead of using a database like MySQL)

“C:\IDEA\_workSpace\press\_REST”:

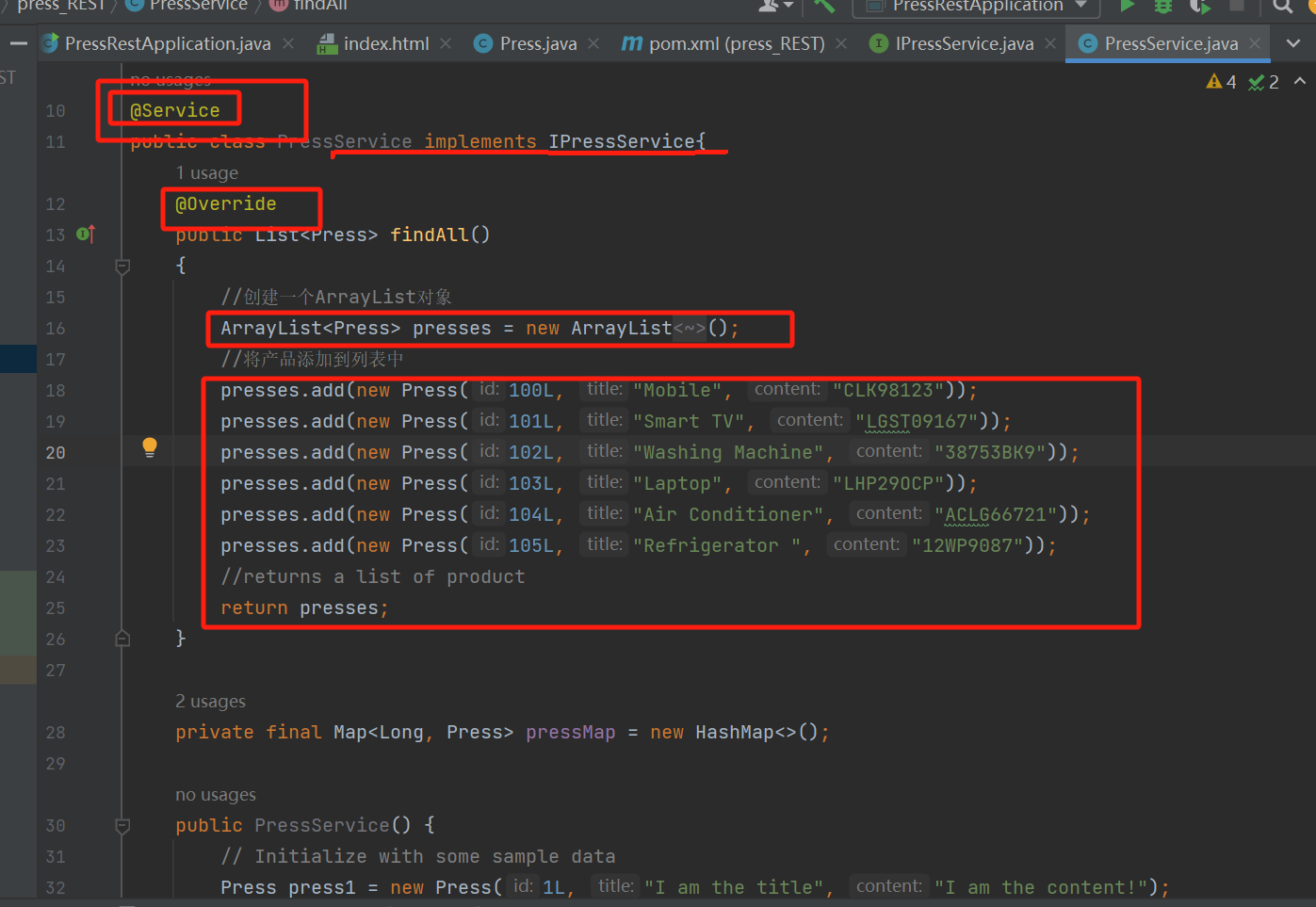
* 1. JavaBean creation: (Press.java)



...

• PressService.java:

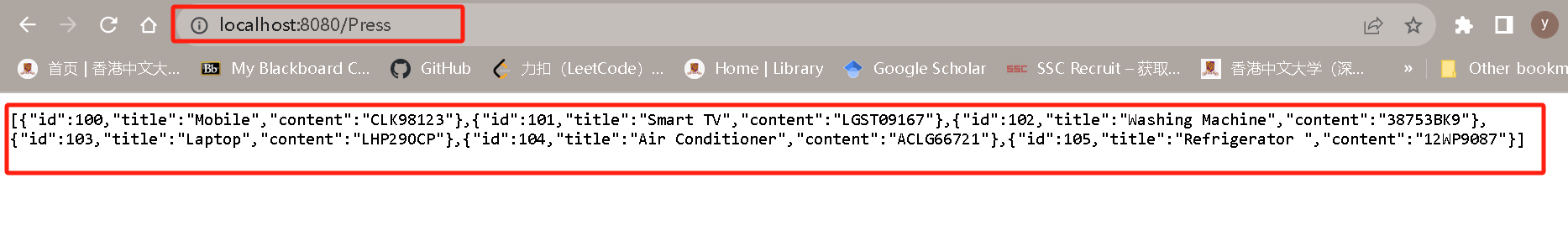
(Using using an in-memory list instead of using a database like MySQL)



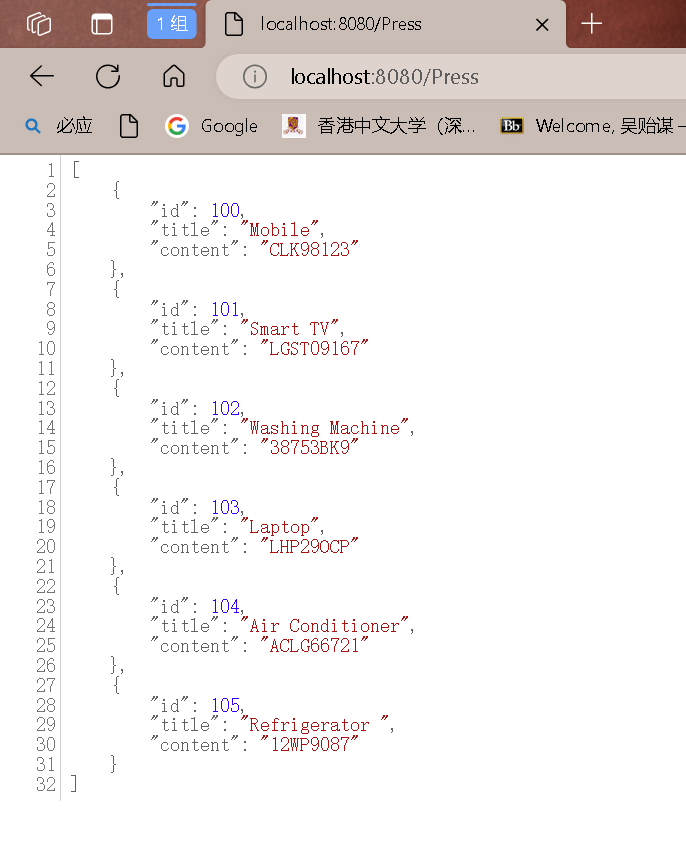
• 打开浏览器并调用 URL [http://localhost:8080/index.html (‘index.html’](http://localhost:8080/index.html(‘index.html’) here is a folder under “C:\IDEA\_workSpace\press-REST\_databaseVersion\src\main\resources\static\index.html” !)。它**显示了 获取所有产品的链接**，如下图所示:

- In Google Chrome:





- In Microsoft Edge:

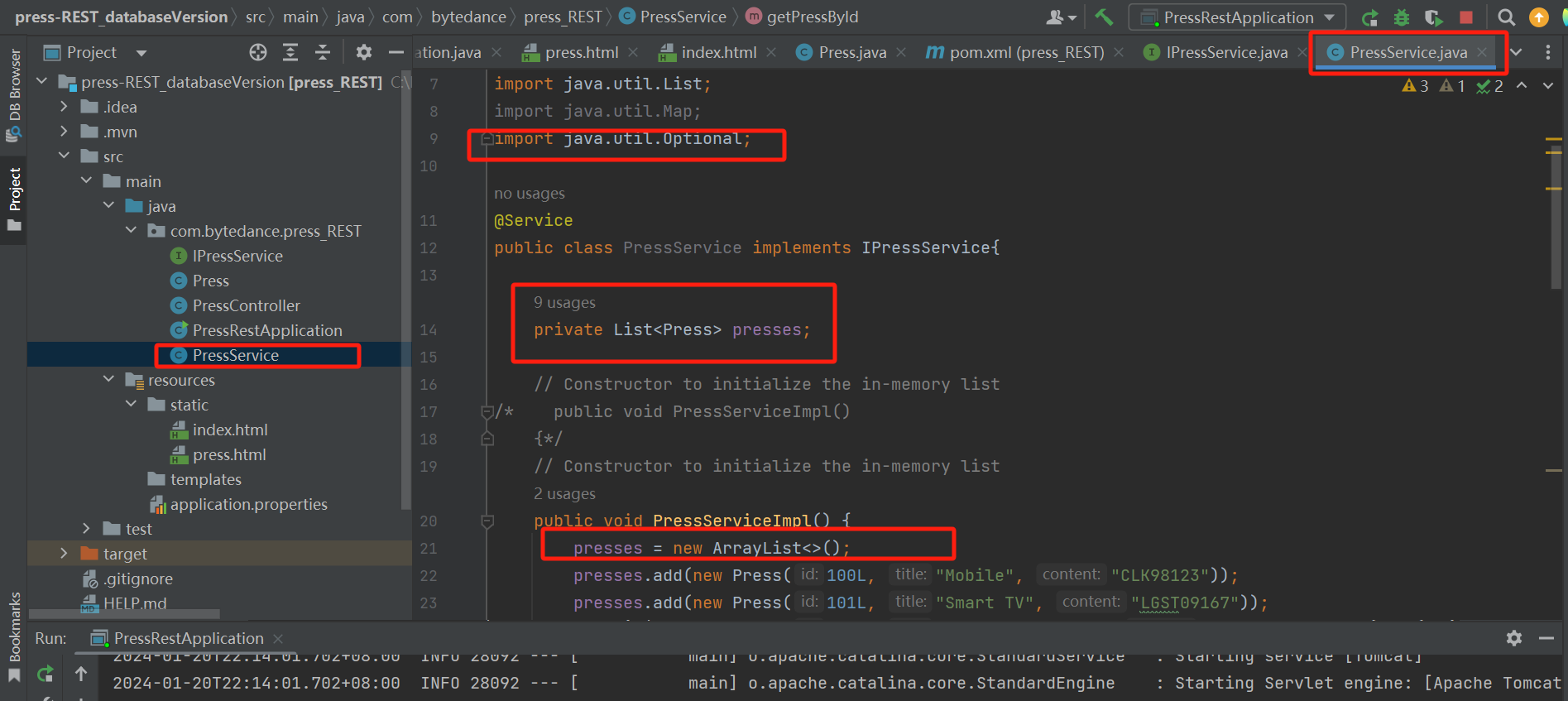


1. -仿照https://www.cainiaojc.com/springboot/springboot-rest-example.html中的REST 示例，写一个press的REST接口，用postman进行校验。

-预先向数据库中存储一篇id为1的文章，做到使用get请求访问localhost:port/press/1(localhost:8080/press/1 here, since )的时候，能够以json格式返回文章数据。

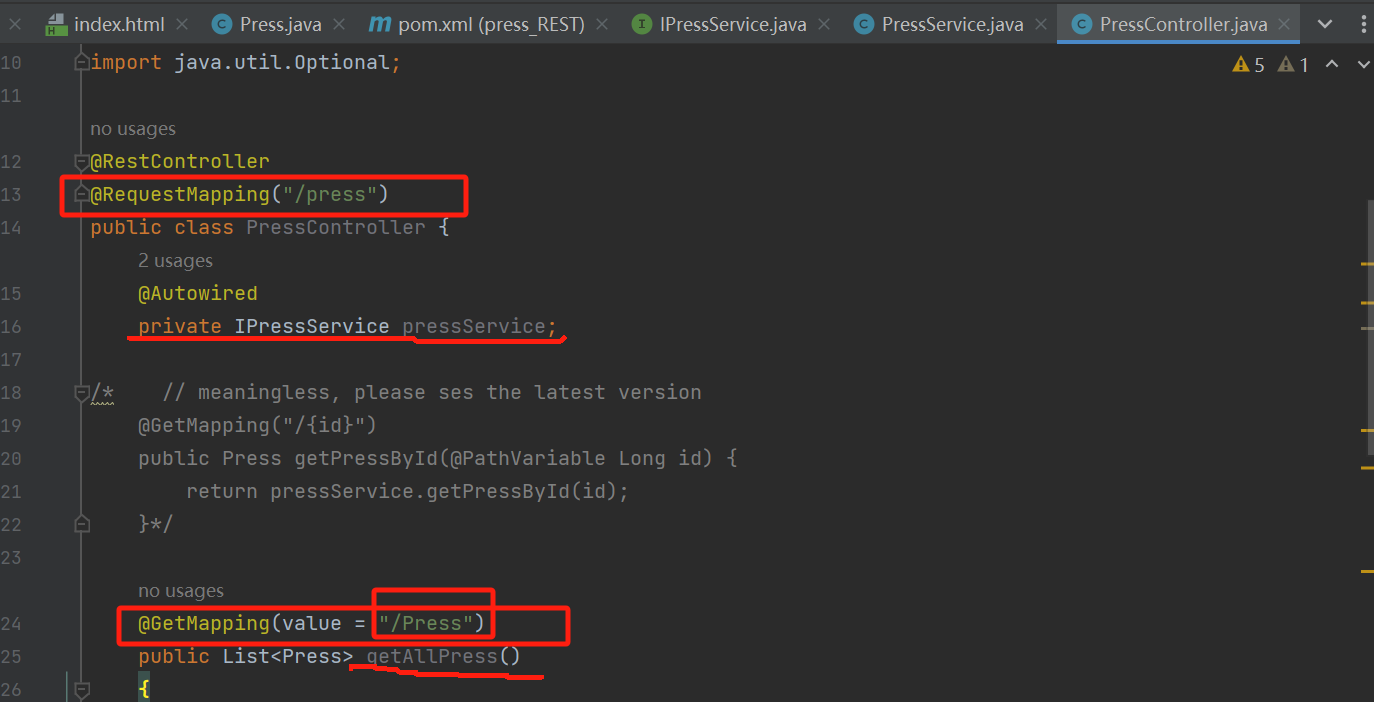
Modify parts:

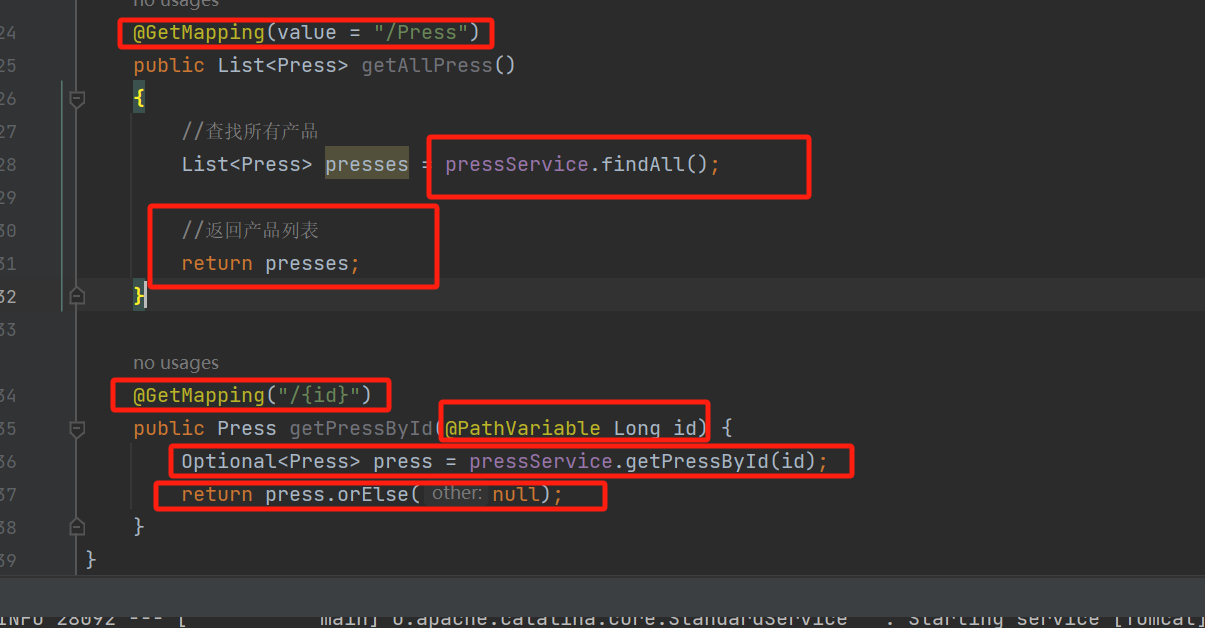
* 1. PressService.java:





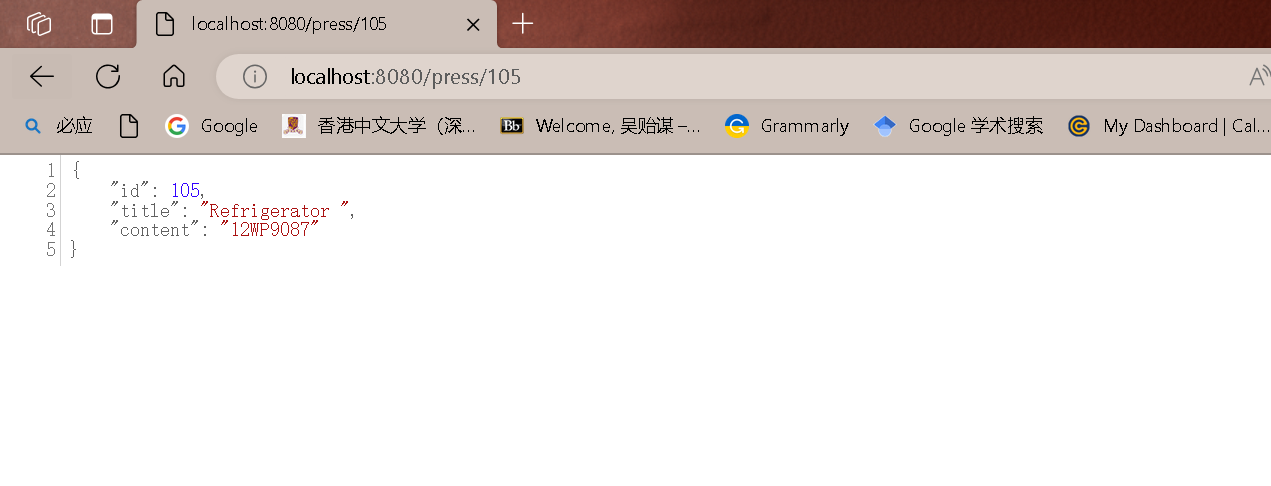
* 1. PressController.java:



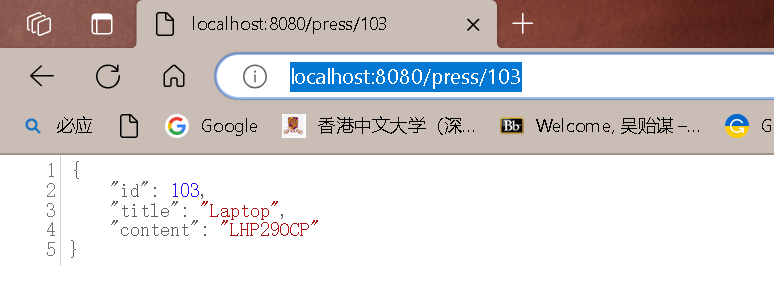


* 1. Output

"[localhost:8080/press/105](http://localhost:8080/press/105)":



“[localhost:8080/press/103](http://localhost:8080/press/103)”:



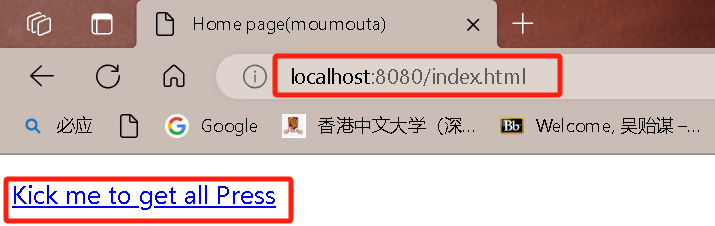
“[localhost:8080/press/100](http://localhost:8080/press/100)”:



“[localhost:8080/press/Press](http://localhost:8080/press/Press)”:



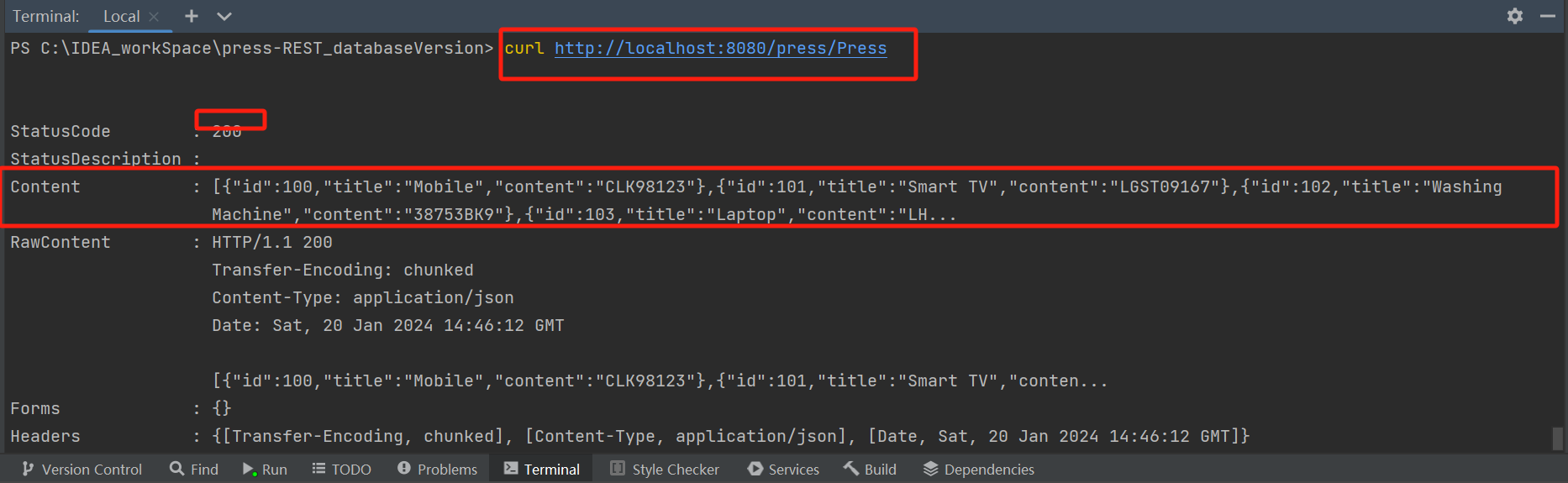
“[Home page(moumouta)](http://localhost:8080/index.html)” / “localhost:8080/index.html”



After Kick the link “Kick me to get all Press”:

Same as “[localhost:8080/press/Press](http://localhost:8080/press/Press)” page.

• 用postman进行校验



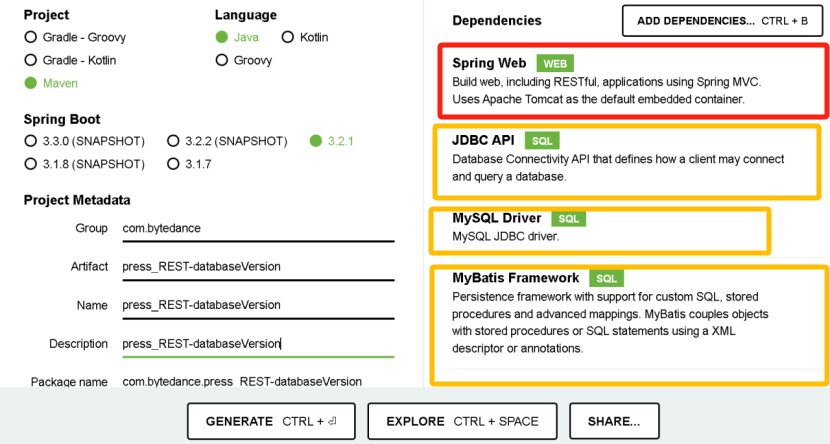
• 缺点/未完成的地方：

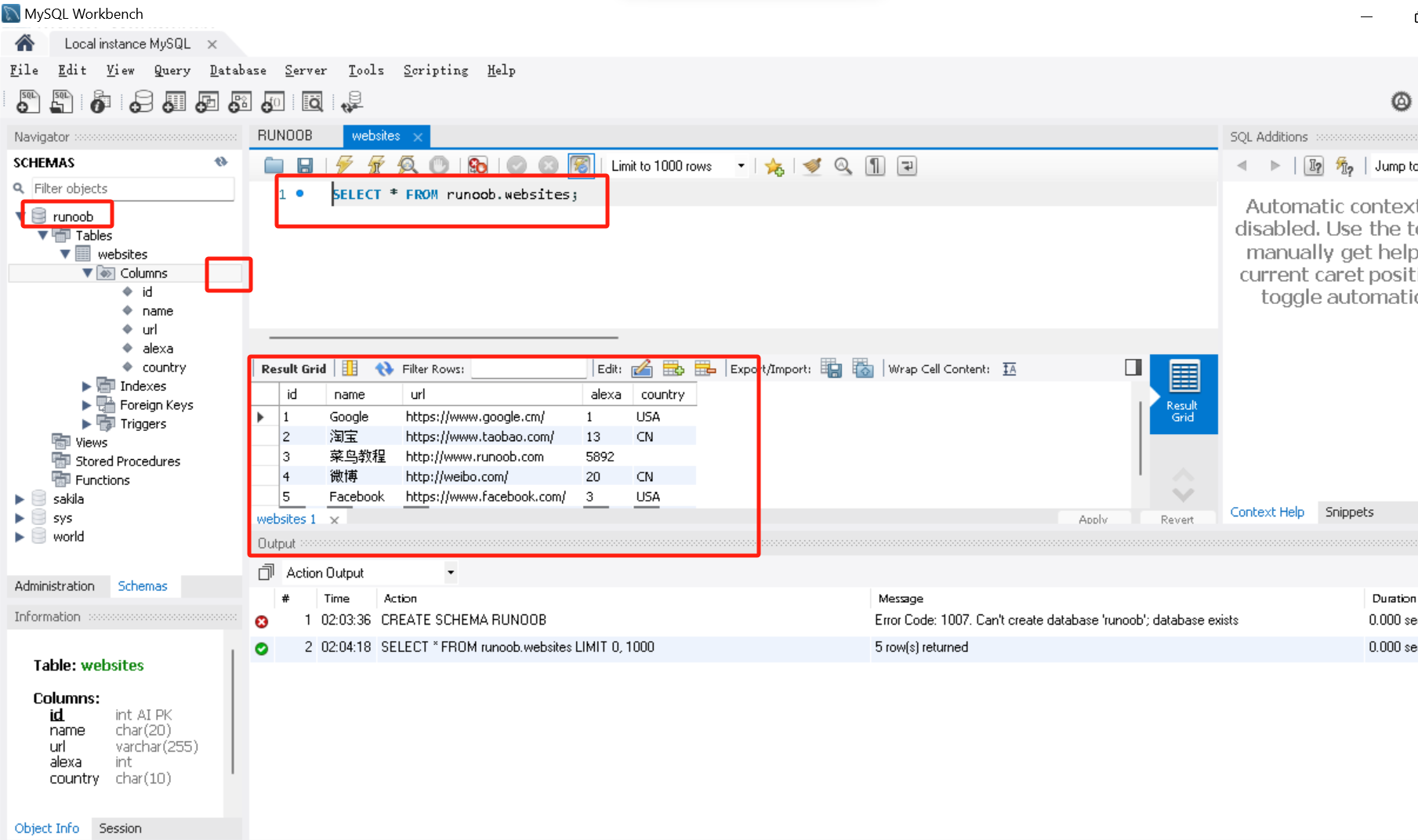
只能实现in-memory list 而不是 using a database like MySQL。在网上查了如何（远程）连接（绑定）SpringBoot和MySQL数据库，但实现过程中出现了较大阻碍。

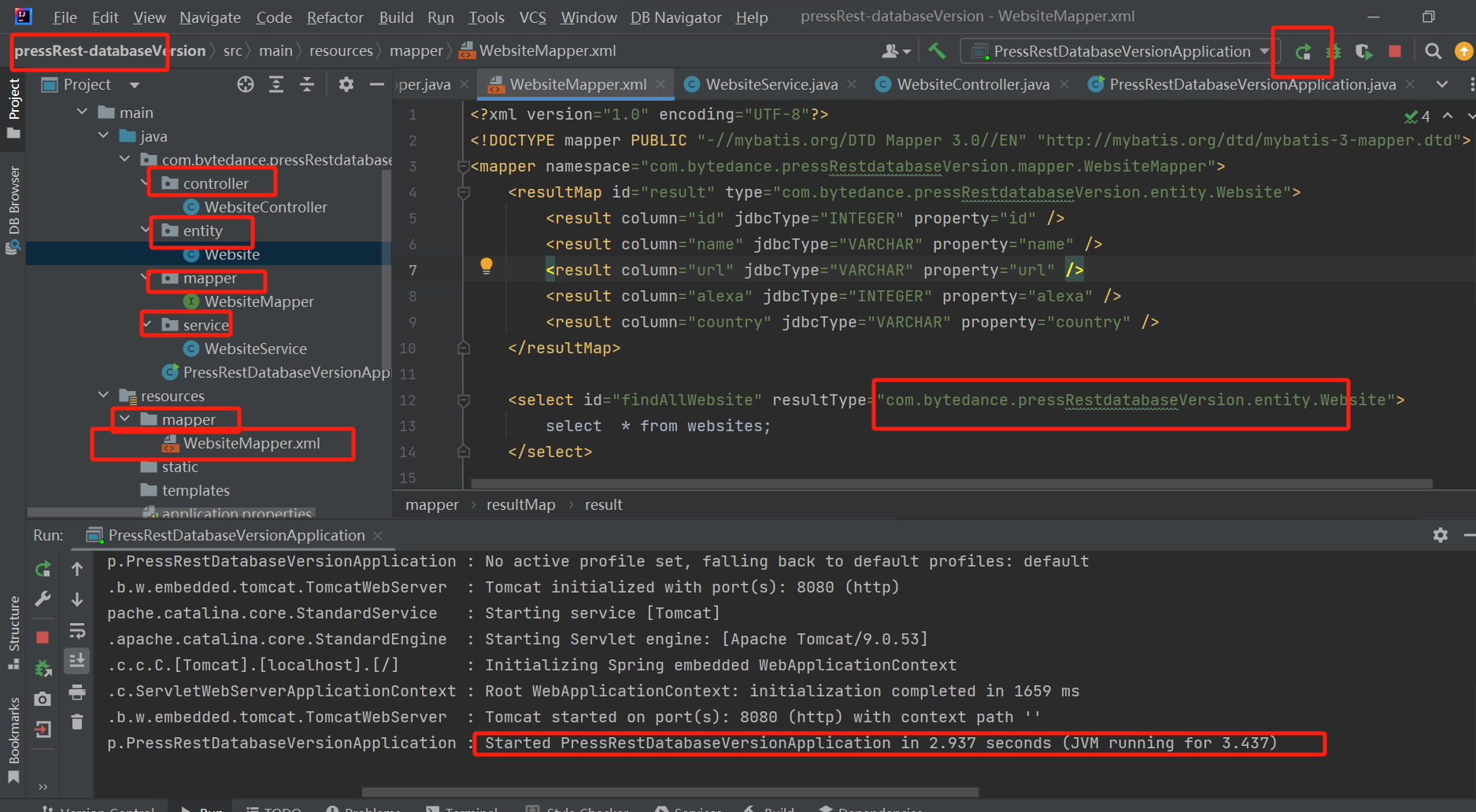
请老师帮忙指点下~ (已解决)

• Database Version:

1. Spring initializr





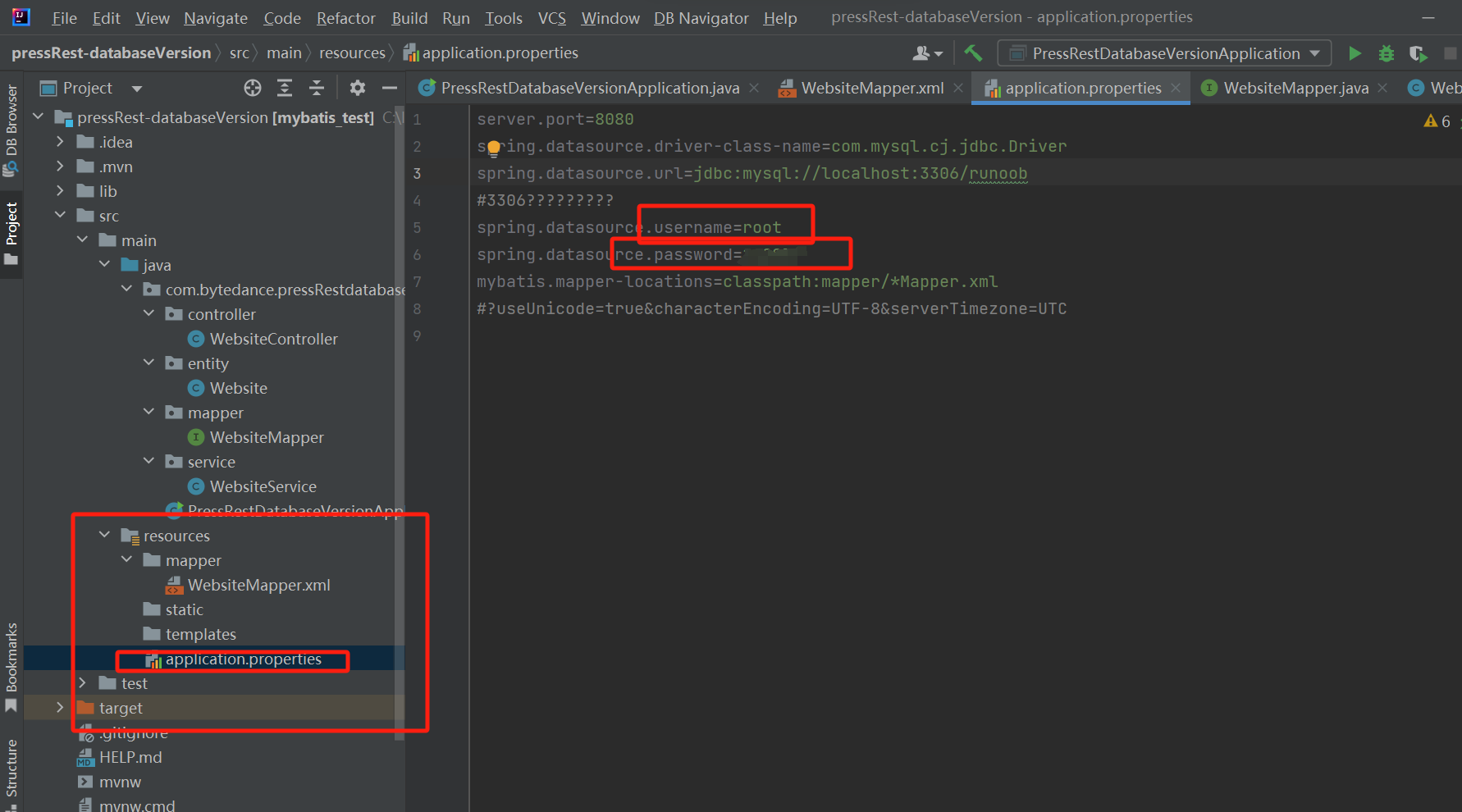


Reference:

<https://blog.csdn.net/YangMax1/article/details/120757964?spm=1001.2014.3001.5501>

<https://blog.csdn.net/YouthBlood9/article/details/120829154>

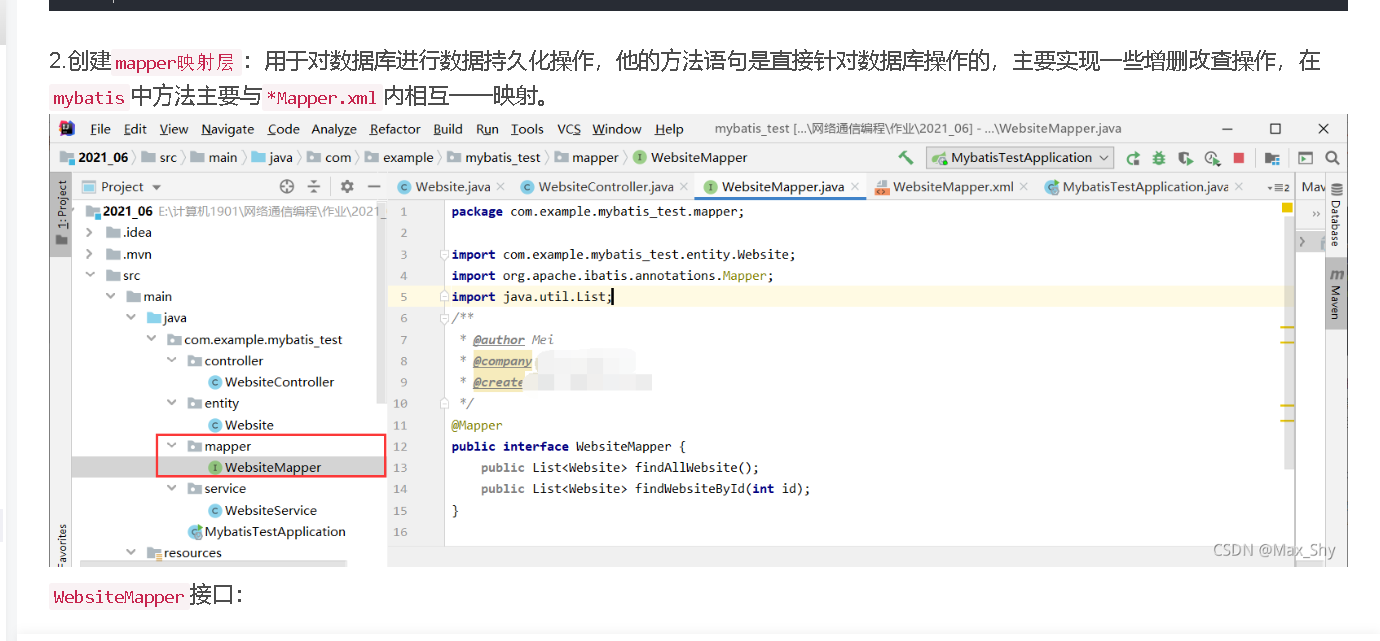
“application.properties”: (!!!)



mapper in ‘com.example’ and mapper in ‘resource.mapper’:

...

1. 创建mapper映射层：用于对数据库进行数据持久化操作，他的方法语句是直接针对数据库操作的，主要实现一些增删改查操作，在mybatis中方法主要与\*Mapper.xml内相互一一映射。



1. 创建Mapper映射对应的WebsiteMapper.xml文件

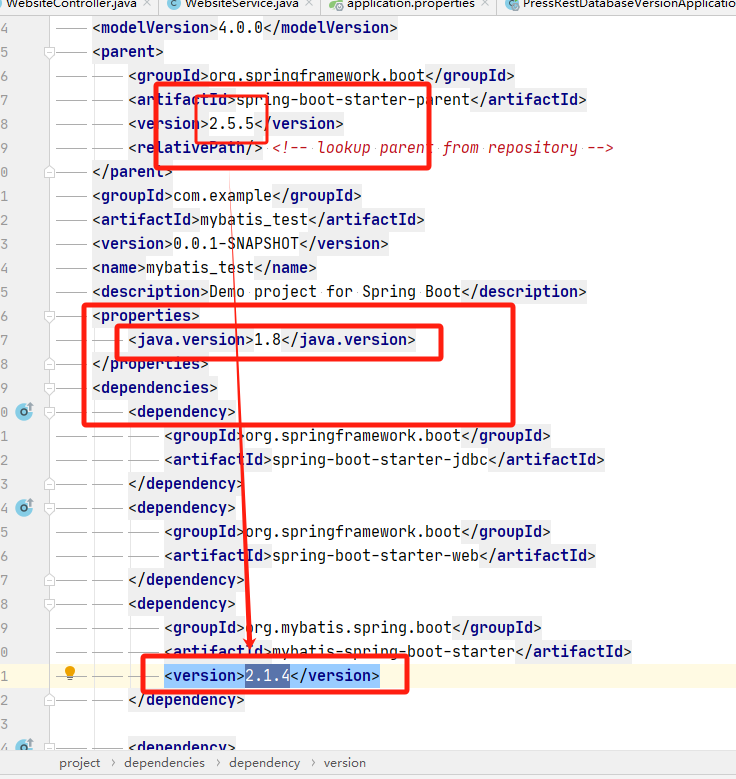
............

注意该文件放在resources目录下的mapper包中，具体包名位置namespace要和上边的映射类对应。  
WebsiteMapper.xml：

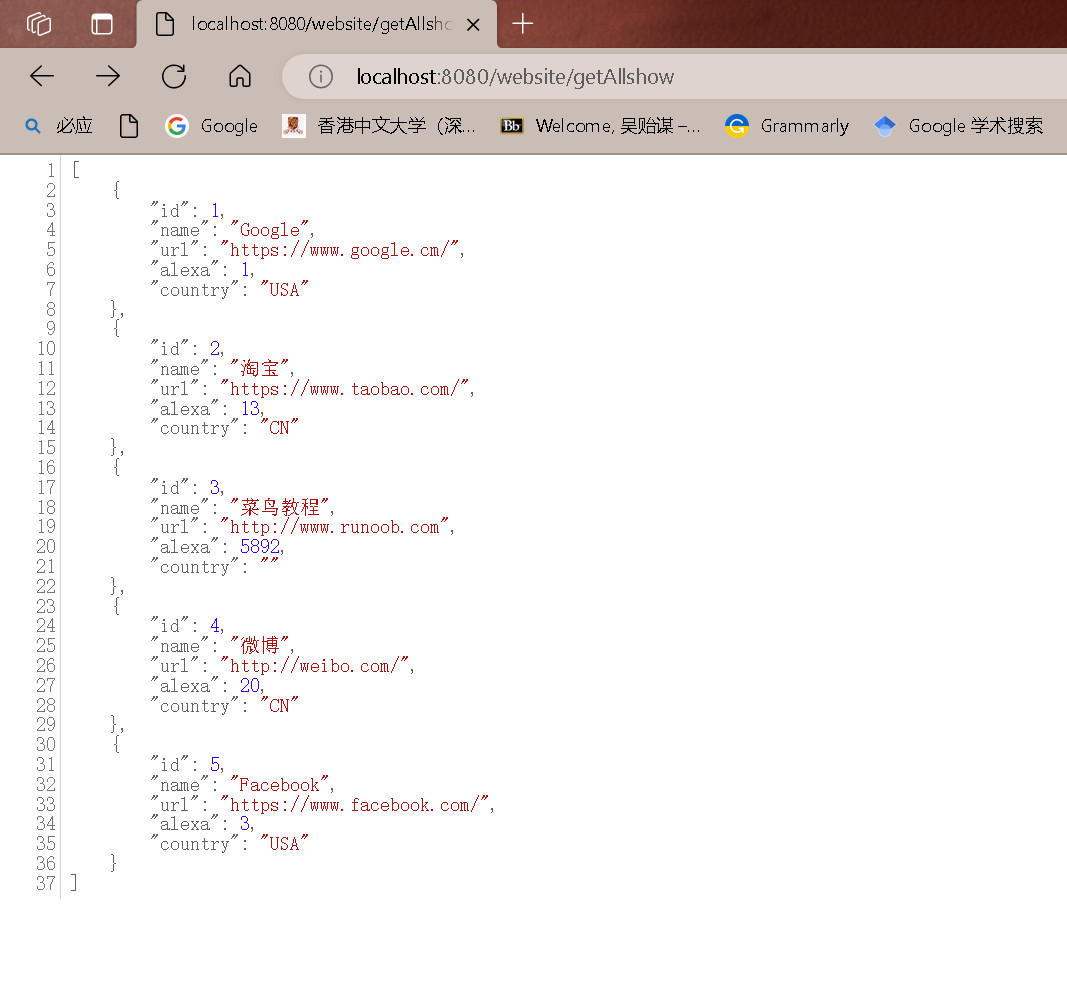


Output:

Mybytis, Java and springBoot’s Version!!! :



[localhost:8080/website/getAllshow](http://localhost:8080/website/getAllshow)



[localhost:8080/website//getWebsiteId/1](http://localhost:8080/website//getWebsiteId/1)

