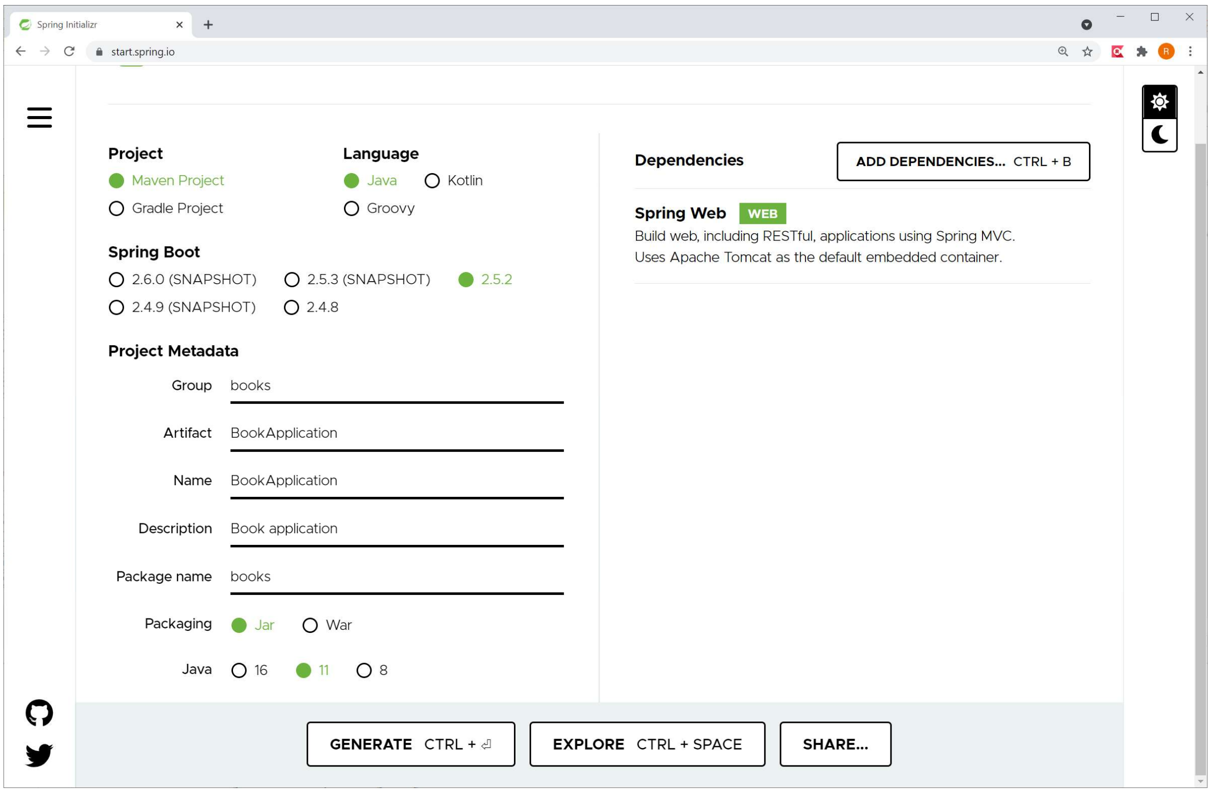
Lab 2

Part 1

In the browser go to https://start.spring.io/


Project: Maven Project [✓], Gradle Project
Language: Java [✓], Kotlin, Groovy
Spring Boot: 2.6.0 (SNAPSHOT), 2.5.3 (SNAPSHOT), 2.5.2 [✓], 2.4.9 (SNAPSHOT), 2.4.8

Project Metadata

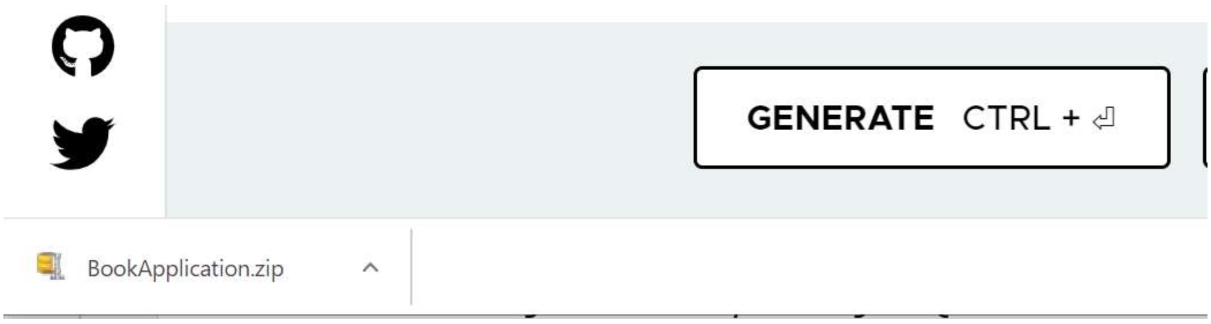
Group	books
Artifact	BookApplication
Name	BookApplication
Description	Book application
Package name	books
Packaging	Jar [✓], War
Java	16, 11 [✓], 8

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Fill in the fields as shown above.

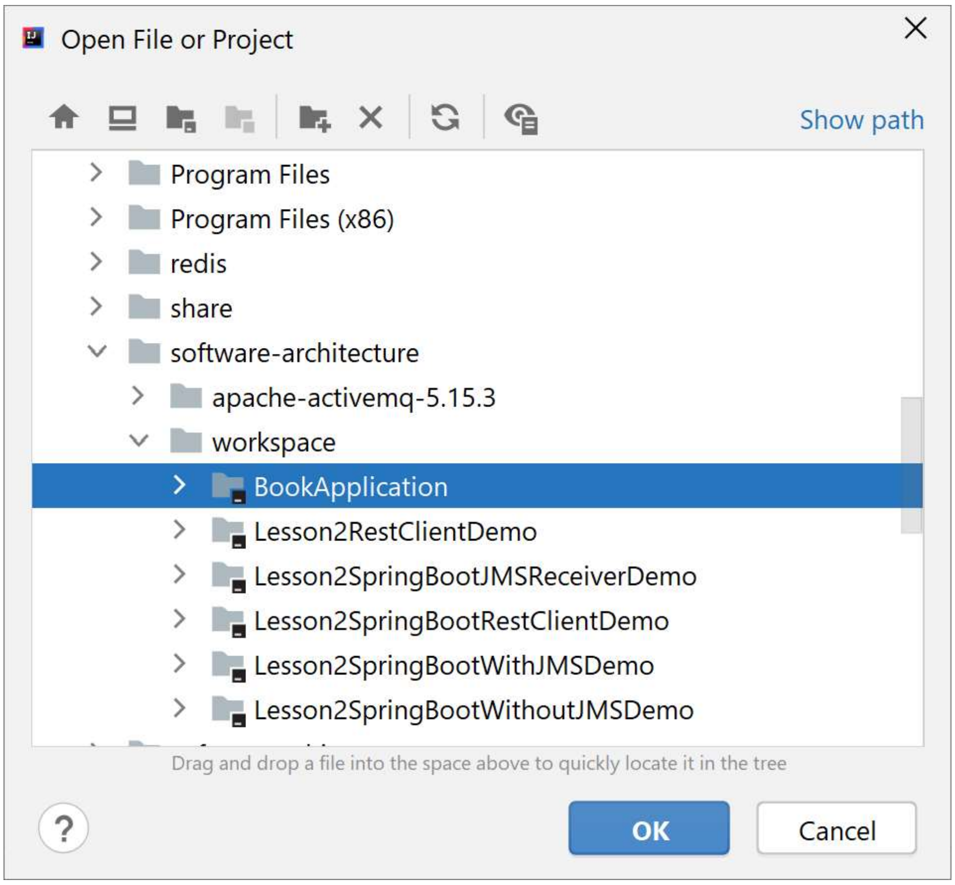
To get the web dependency, click the **Add Dependencies** button and select **Spring Web**.



Click the **Generate** button and see that **BookApplication.zip** is now downloaded to your computer.

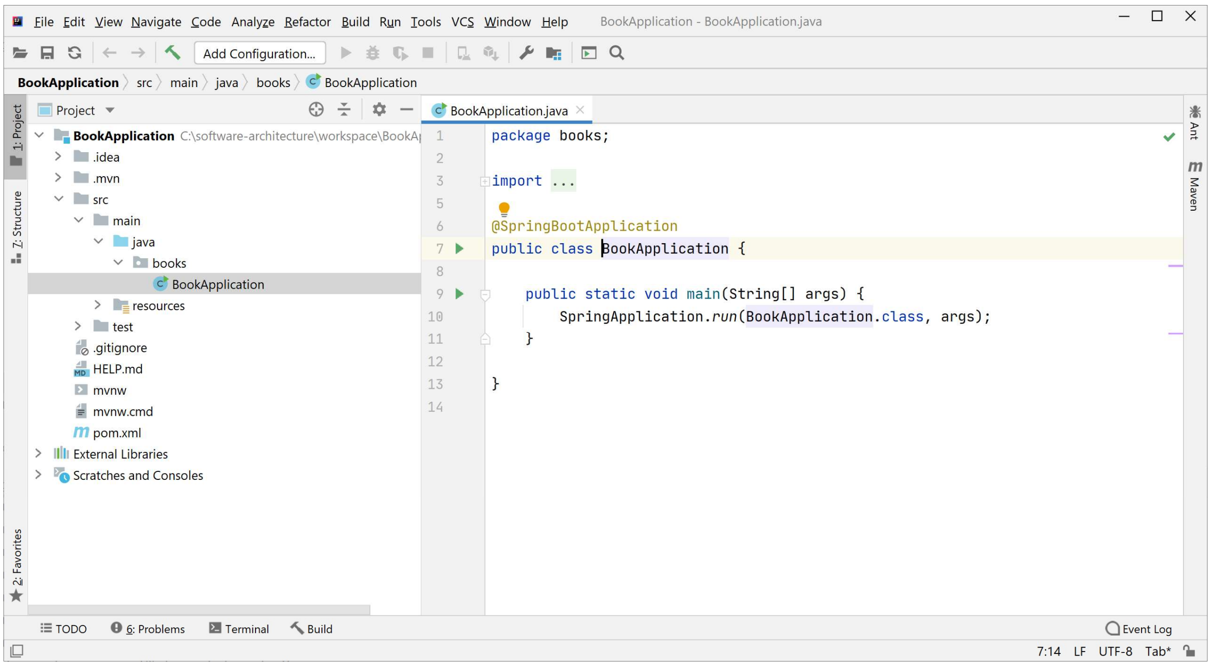
Unzip the content of this file, for example to c:\software-architecture\workspace

In IntelliJ, open now the folder **BookApplication** from the location where you unzipped the file.


Open File or Project
Show path
Program Files
Program Files (x86)
redis
share
software-architecture
apache-activemq-5.15.3
workspace
BookApplication
Lesson2RestClientDemo
Lesson2SpringBootJMSReceiverDemo
Lesson2SpringBootRestClientDemo
Lesson2SpringBootWithJMSDemo
Lesson2SpringBootWithoutJMSDemo
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel

Click **OK**.

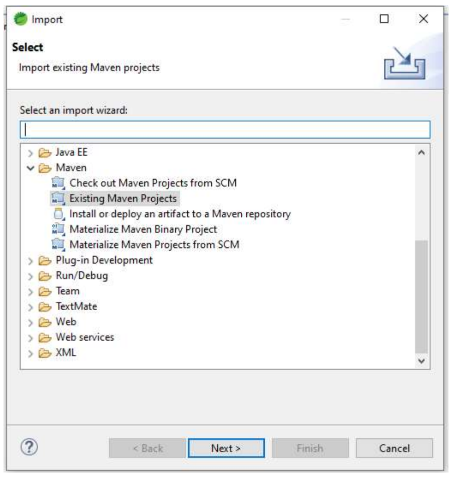
You see now a basic Spring boot application.



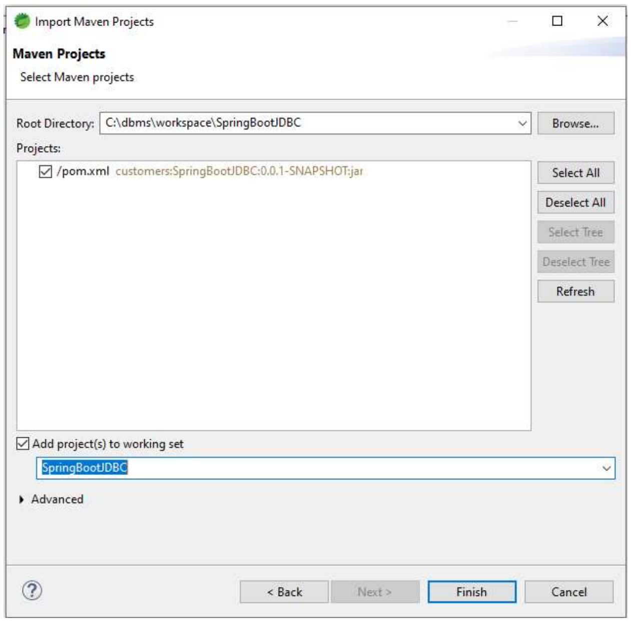
```
package books;

import ...

@SpringBootApplication
public class BookApplication {

    public static void main(String[] args) {
        SpringApplication.run(BookApplication.class, args);
    }

}
```

If you want to use Eclipse instead of IntelliJ, do the following:

In Eclipse select **File-> Import**


Import
Select
Import existing Maven projects
Select an import wizard:
Java EE
Maven
Check out Maven Projects from SCM
Existing Maven Projects
Install or deploy an artifact to a Maven repository
Materialize Maven Binary Project
Materialize Maven Projects from SCM
Plug-in Development
Run/Debug
Team
TextMate
Web
Web services
XML
< Back
Next >
Finish
Cancel

Choose **Maven-> Existing Maven Project** and click **Next**



Root Directory: C:\dbms\workspace\SpringBootJDBC

☑ /pom.xml customers:SpringBootJDBC:0.0.1-SNAPSHOT:jar

☑ Add project(s) to working set

SpringBootJDBC

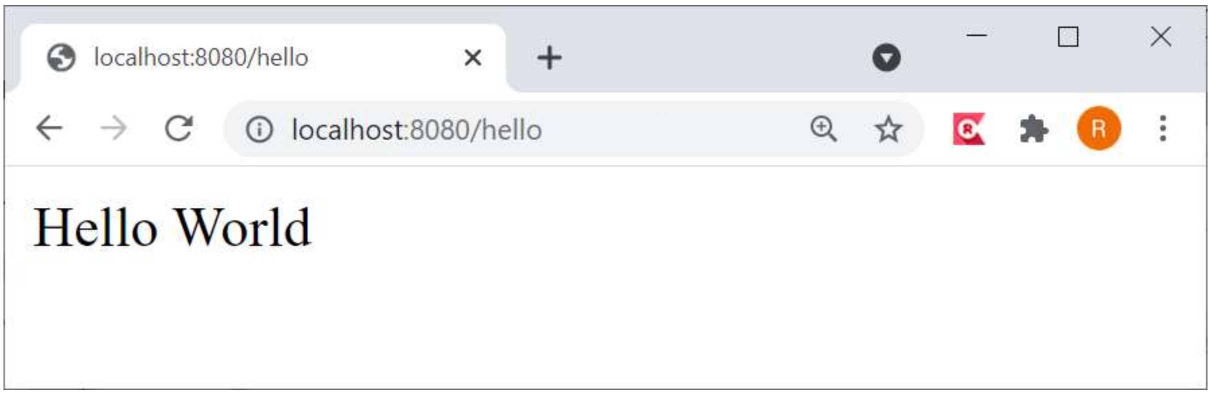
Select the location of you project and check the **Add project to working set** checkbox.

Then click **Finish** and you are done.

Now add the following class to the BookApplication project:

```
package books;

import org.springframework.web.bind.annotation.RestController;

@RestController
public class GreetingController {
    public String sayHello(){
        return "Hello World";
    }
}
```

Then run the file BookApplication.java
Op the browser with the URL: **http://localhost:8080/hello**

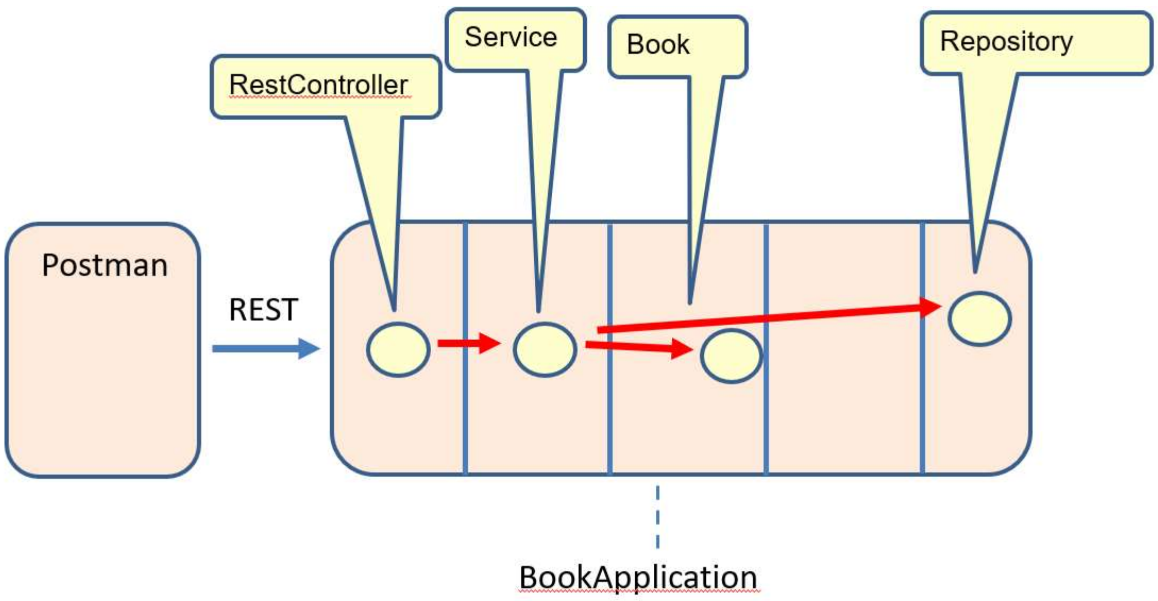


```
Hello World
```

Now write a Book application using REST with the following functionality:

addBook(Book book);
updateBook(Book book);
deleteBook(String isbn);
getBook(String isbn);
getAllBooks();

The Book class has the following properties: isbn, author, title, price
The application should have a controller class, a service class and a repository class.

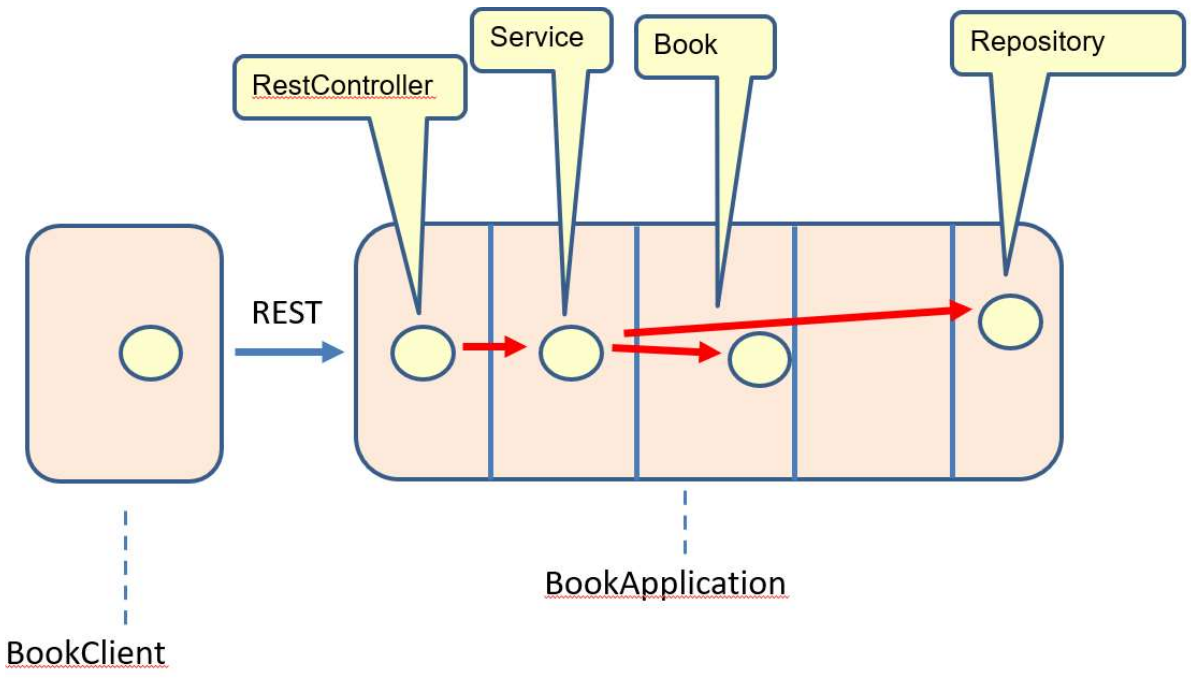


Download and install **postman** and check if your application works correctly

Part 2

Copy and paste the given **Lesson2SpringBootRestClientDemo** to a new project with the name BookClient.

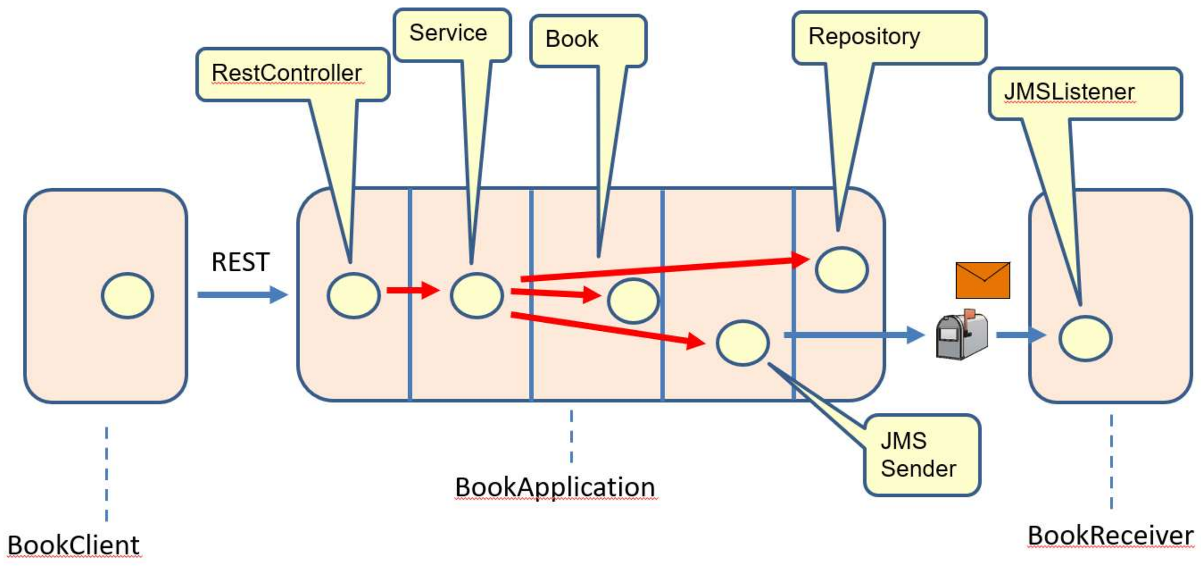
Modify the application so that this client application calls the REST interface of the BookApplication using a RestTemplate.



<u>Part 3</u>

Modify the BookApplication so that every time a book is added, deleted or updated, the application sends a JMS message with the corresponding book.

Copy and paste the given **Lesson2SpringBootJMSReceiverDemo** to a new BookReceiver project. Modify this project so that it receives all messages send by the BookApplication.



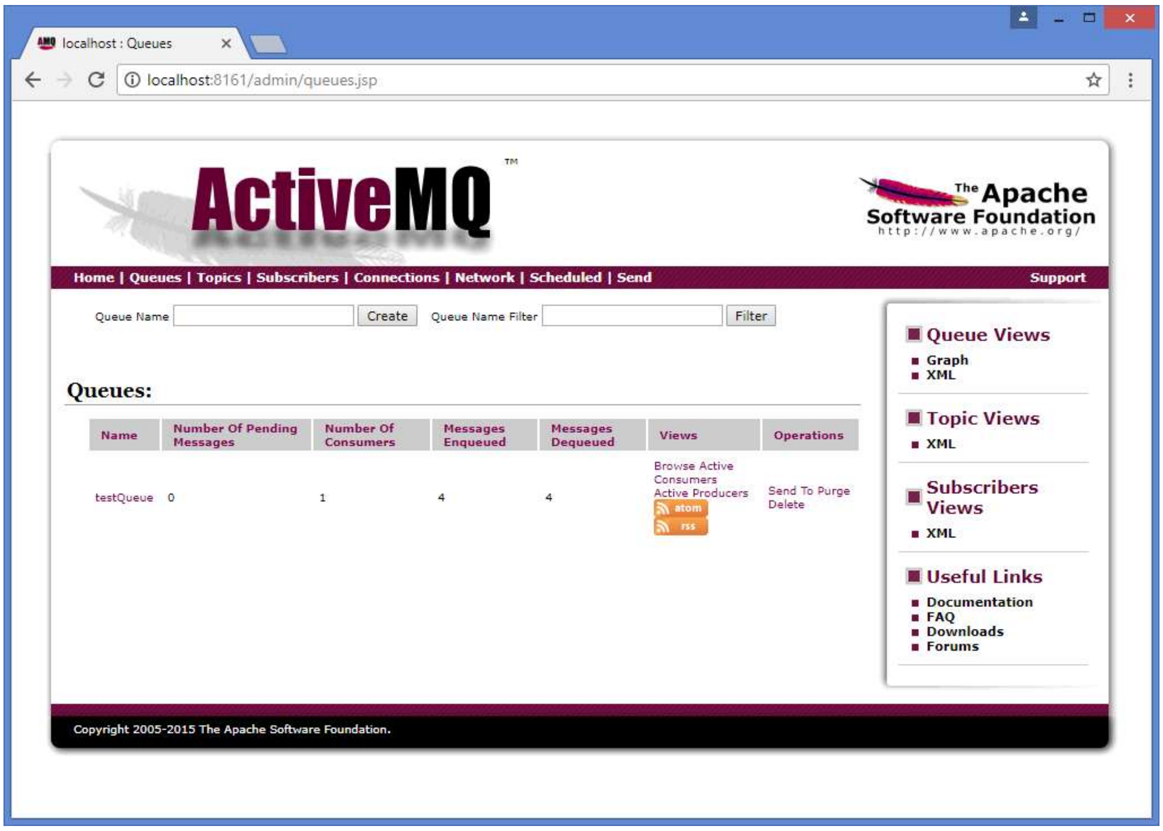
For this part you need to run ActiveMQ.

Go to the folder C**\apache-activemq-5.15.3\bin** and click the file **startactivemq.bat**

Once ActiveMQ is running you can open the ActiveMQ console at http://localhost:8161/admin.

You can login with username **admin** and password **admin**

Select the Queues page from the menu:


localhost : Queues
localhost:8161/admin/queues.jsp
ActiveMQ
The Apache Software Foundation http://www.apache.org/
Home | Queues | Topics | Subscribers | Connections | Network | Scheduled | Send
Support
Queue Name
Create
Queue Name Filter
Filter

Queues:

Name	Number Of Pending Messages	Number Of Consumers	Messages Enqueued	Messages Dequeued	Views	Operations
testQueue	0	1	4	4	Browse Active Consumers Active Producers atom rss	Send To Purge Delete

Queue Views
Graph
XML
Topic Views
XML
Subscribers Views
XML
Useful Links
Documentation
FAQ
Downloads
Forums
Copyright 2005-2015 The Apache Software Foundation.

Here you see the queues and other data.

Part 4

Modify the BookApplication so that every time a book is added or deleted, the application does the following using spring events:

1. It keeps track of when a certain book is added or deleted. So we need a new class that contains a list of all modifications.
2. It prints the data of the book to the console.

What to hand in?

1. A separate zip file for part1, part 2, part 3 and part 4