

## Lesson 02 Demo 01

### Creating a RESTful Web Service with @RestController

**Objective:** To create a basic web service using Spring Boot to build and deploy RESTful APIs for various applications and use cases

**Tool required:** Eclipse IDE and Visual Studio Code

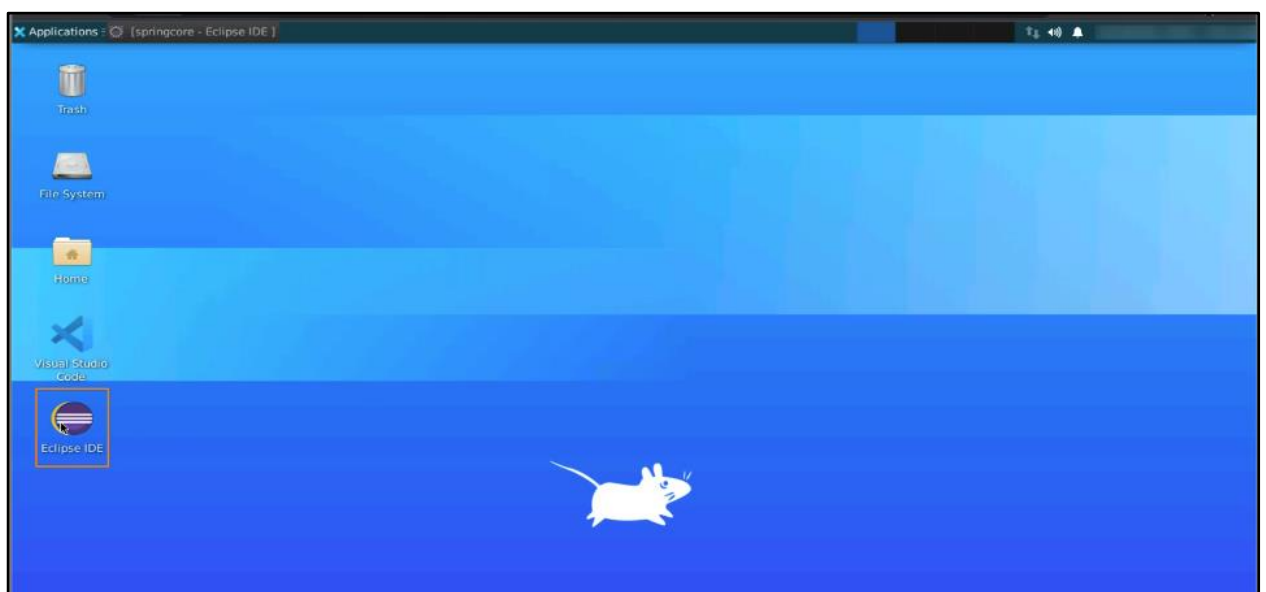
**Prerequisites:** None

#### Steps to be followed:

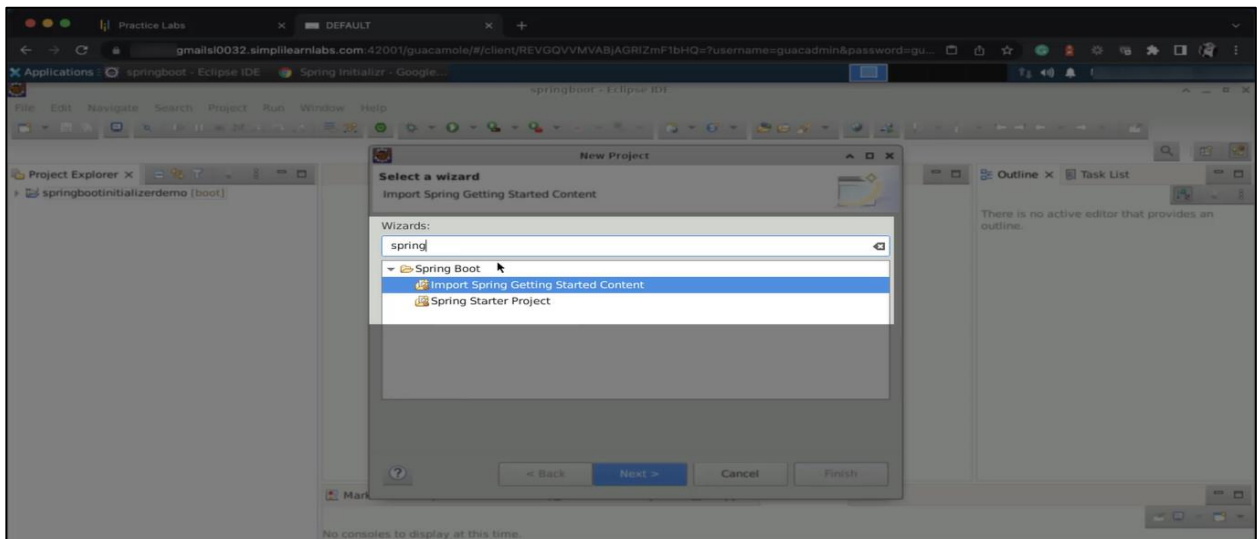
1. Implementing the web services that return a JSON file
2. Creating a Controller class
3. Implementing maps with key-value pair
4. Creating a News Model class

#### Step 1: Implementing the web services that return a JSON file

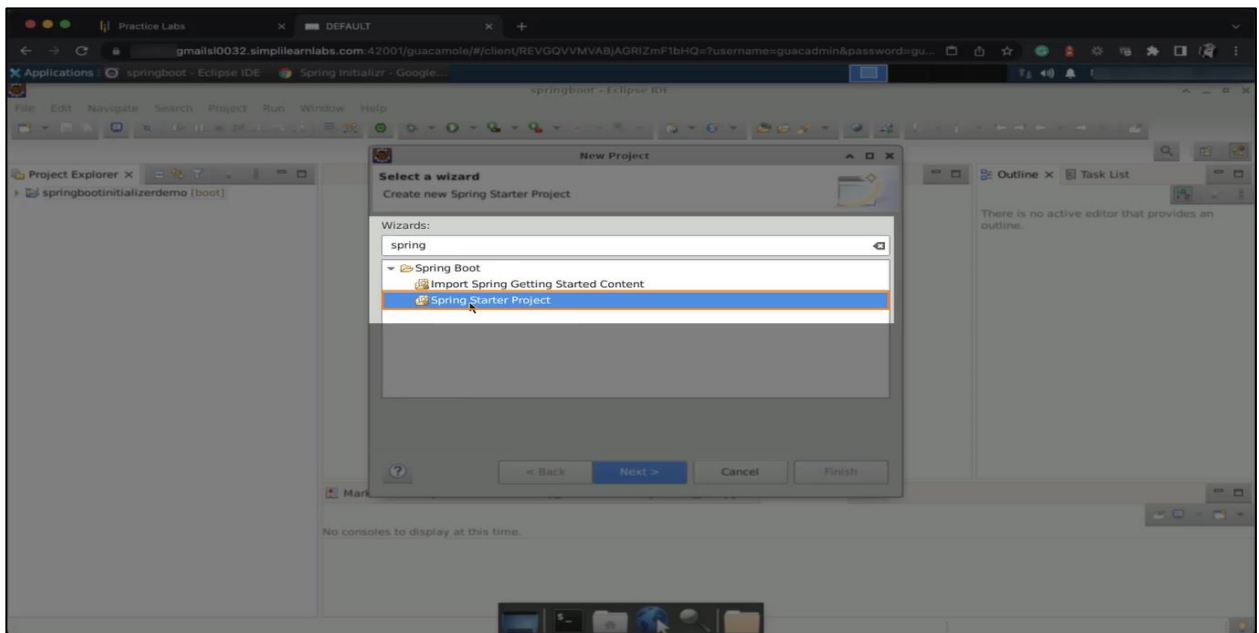
##### 1.1 Open Eclipse IDE



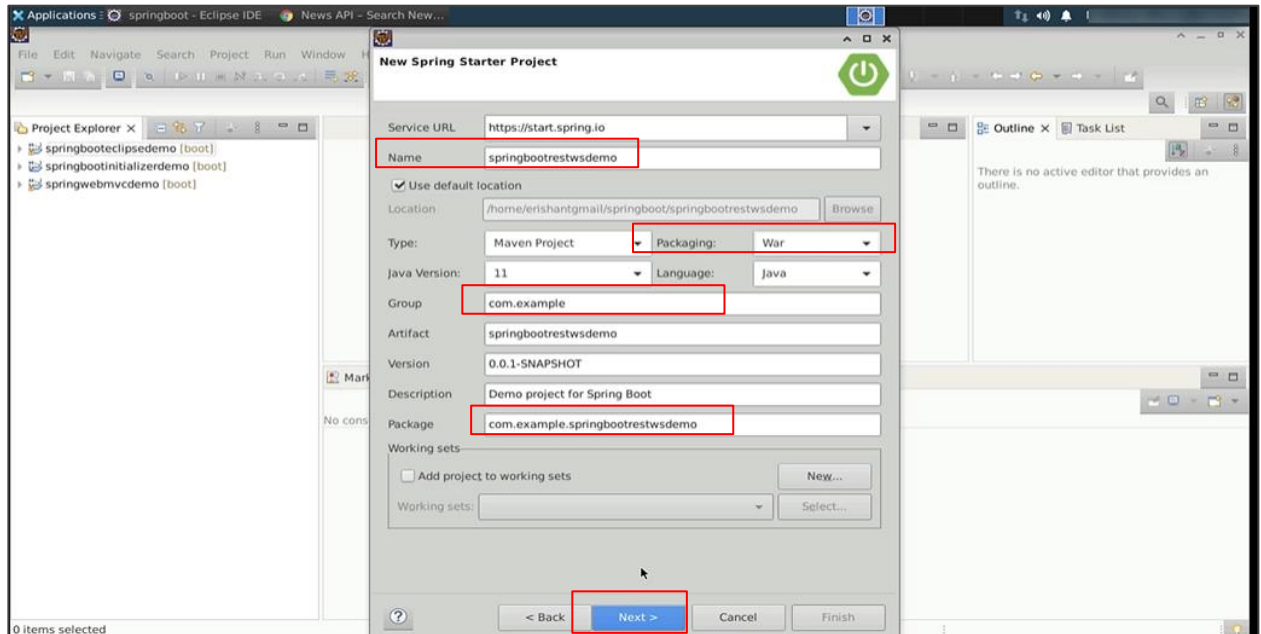
## 1.2 Create a Spring project, select **File > New > Project** and search for **spring**



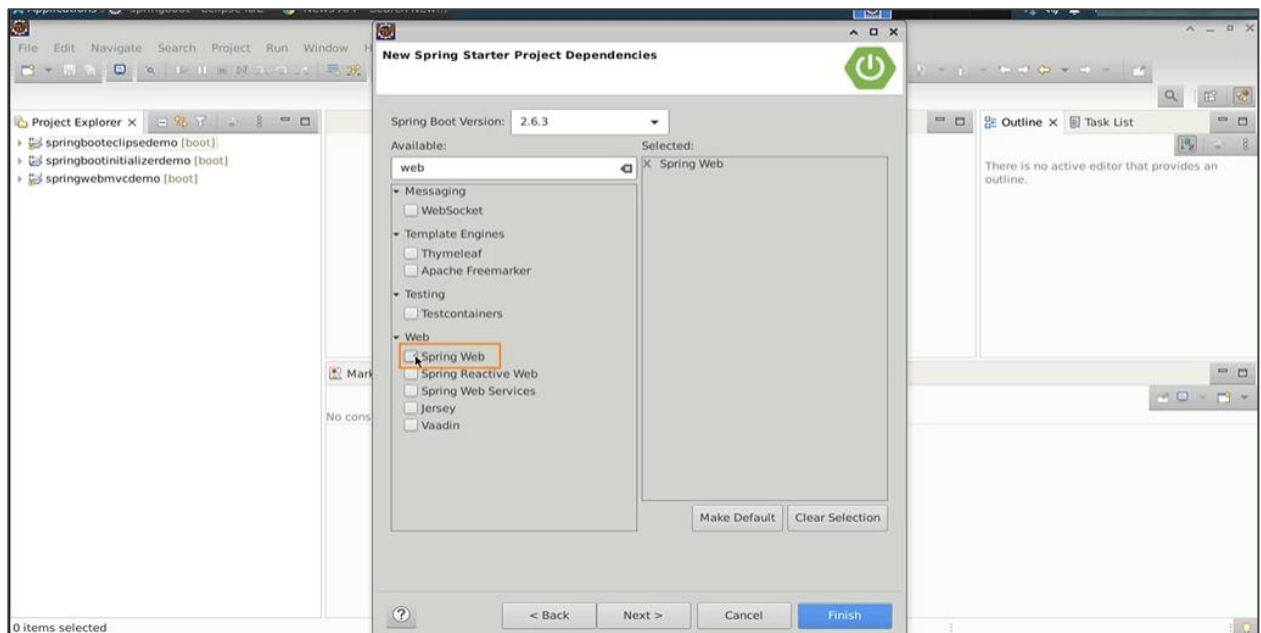
## 1.3 Select **Spring Starter Project** in the select wizard and click **Next**



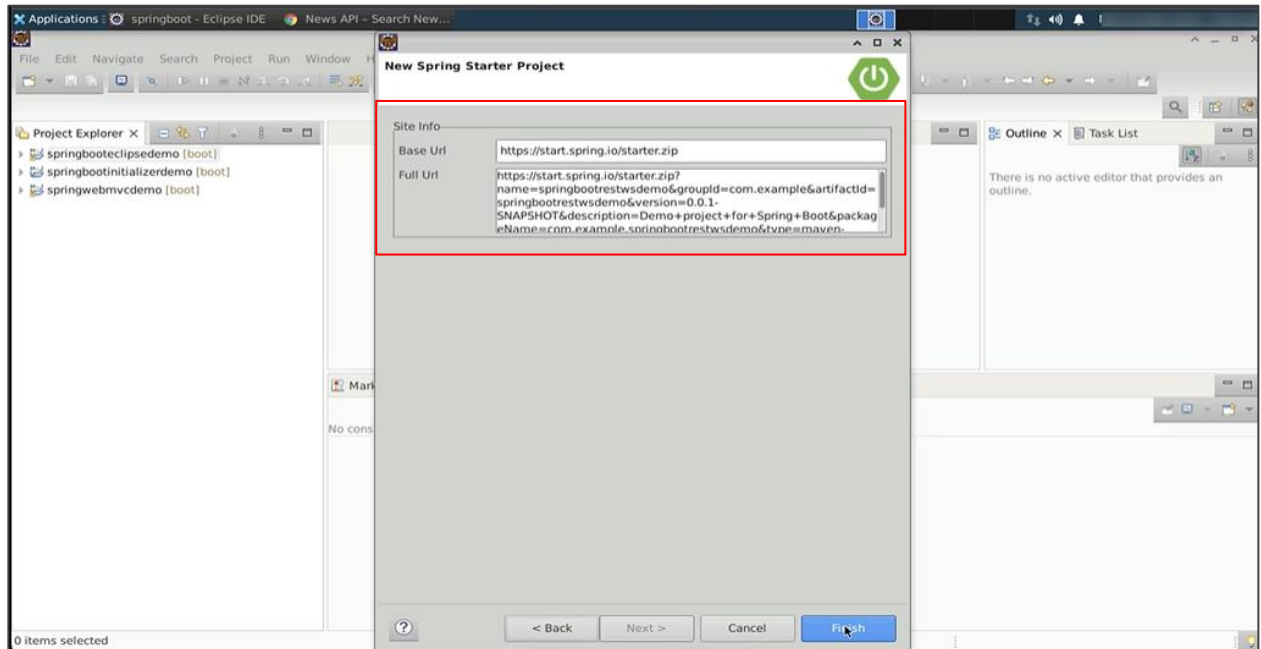
1.4 Provide a project name, such as **springbootrestwsdemo**, configure the project with Maven as the build tool, and choose the packaging as **War**



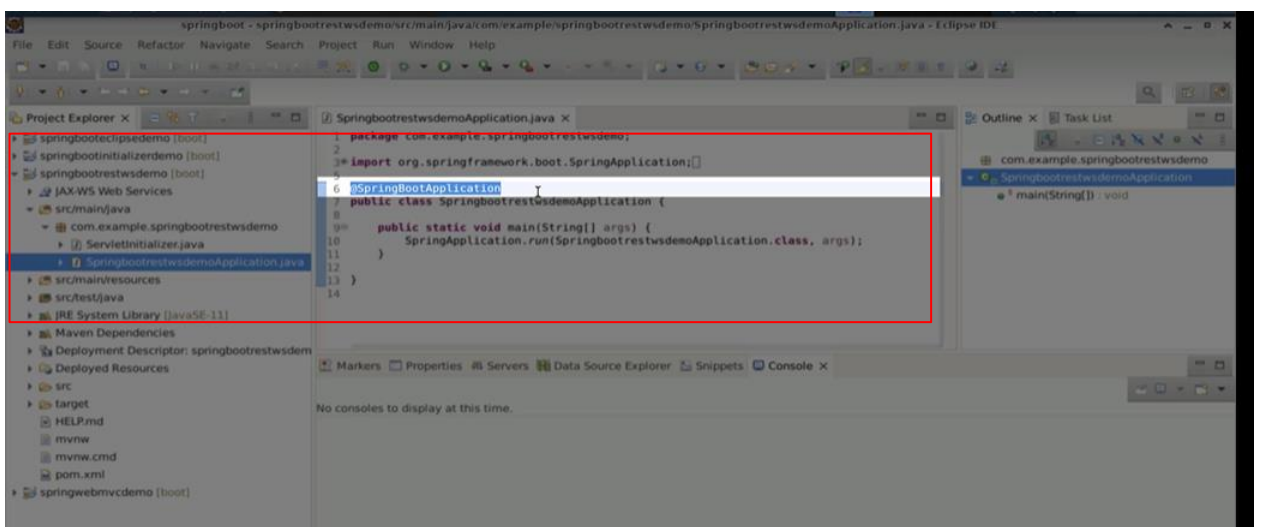
1.5 Search for **Spring Web**, add the dependency, and click **Finish**



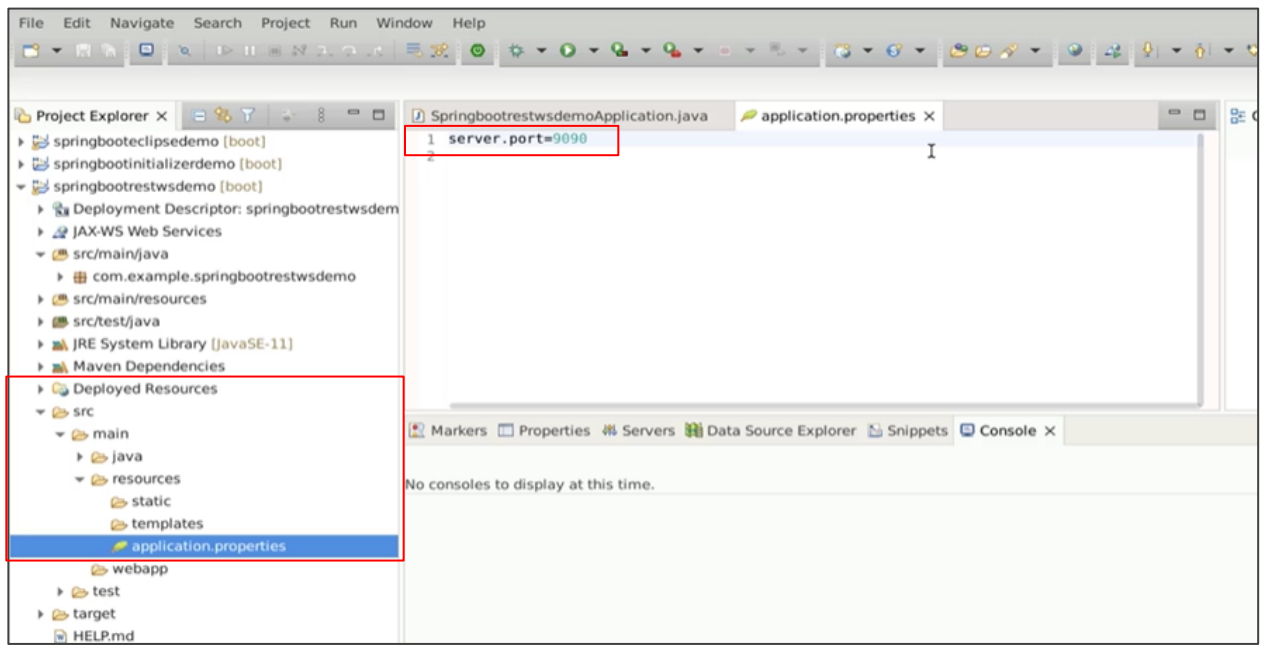
1.6 It will automatically sync your project from the Spring Starter website itself. Click **Finish**



Whenever you create a Spring Boot project, by default, you will get one application file that is annotated with **@SpringBootApplication**.

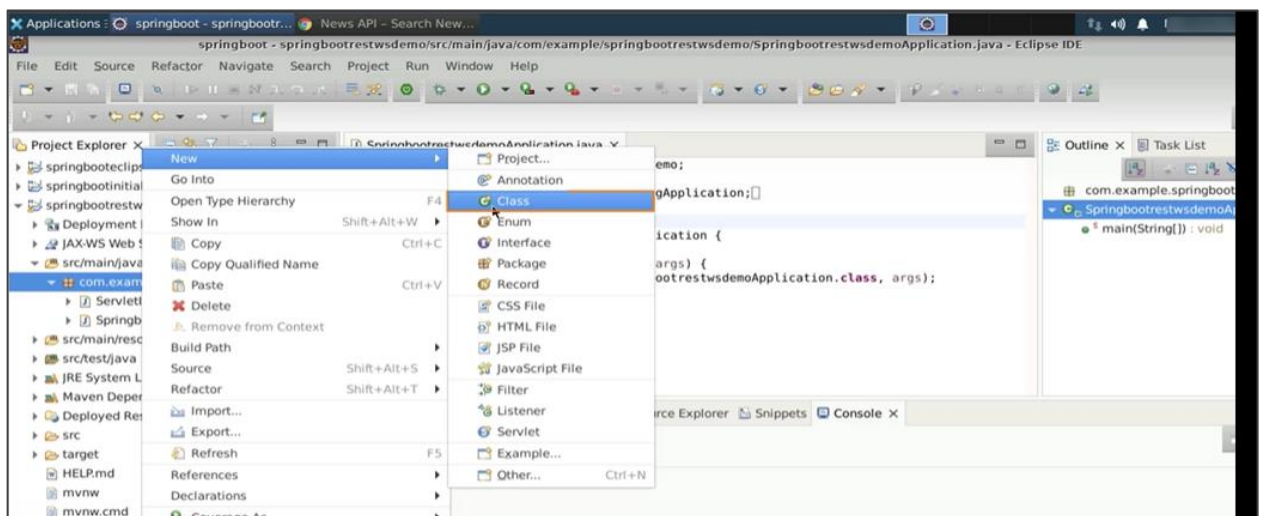


- 1.7 Open the **application.properties** file located in the **src/main/resources** directory and configure the server port by adding the property **server.port=9090** (choose any port number)

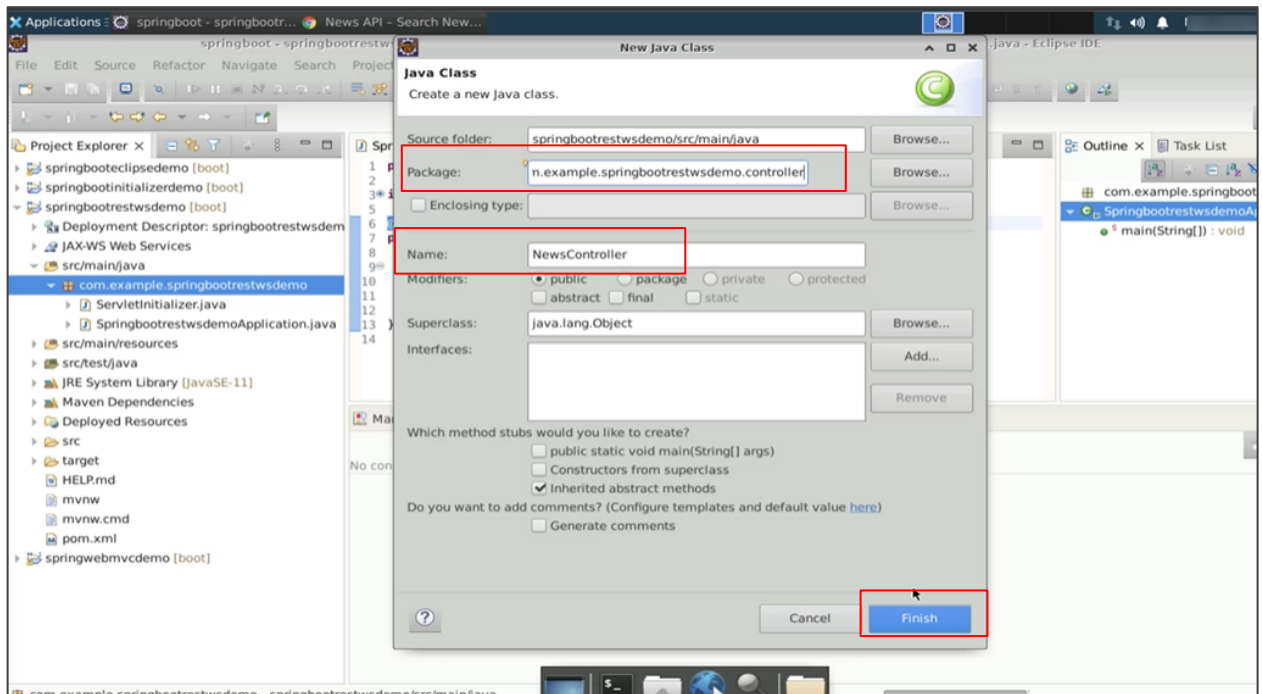


## Step 2: Creating a Controller class

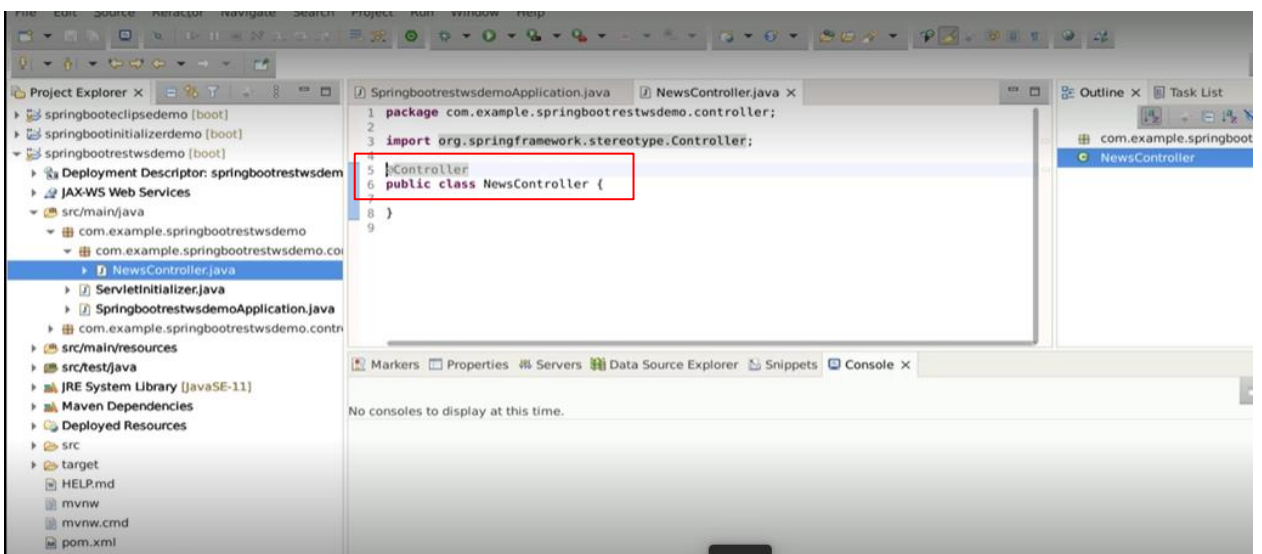
- 2.1 Right-click on **com.example.springbootrestwsdemo** package and click **New > Class**



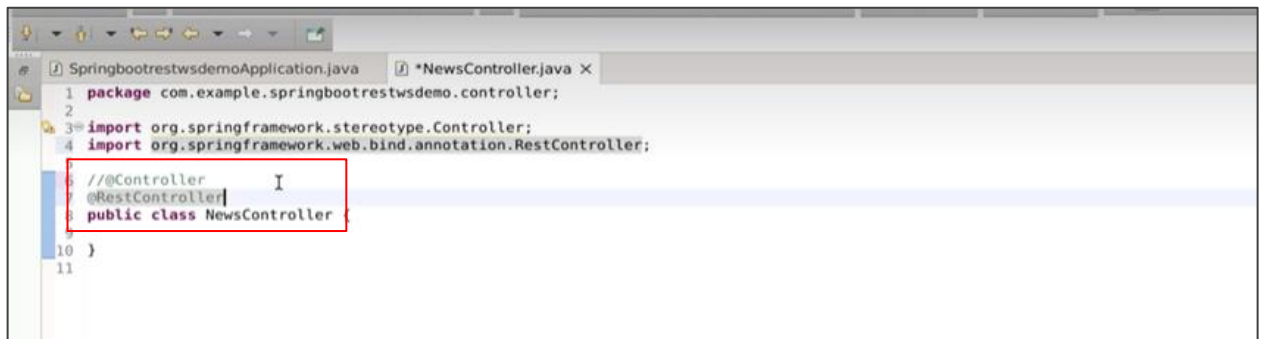
2.2 Name the class as **NewsController**, in the package field add **.controller** at the end, and click **Finish**



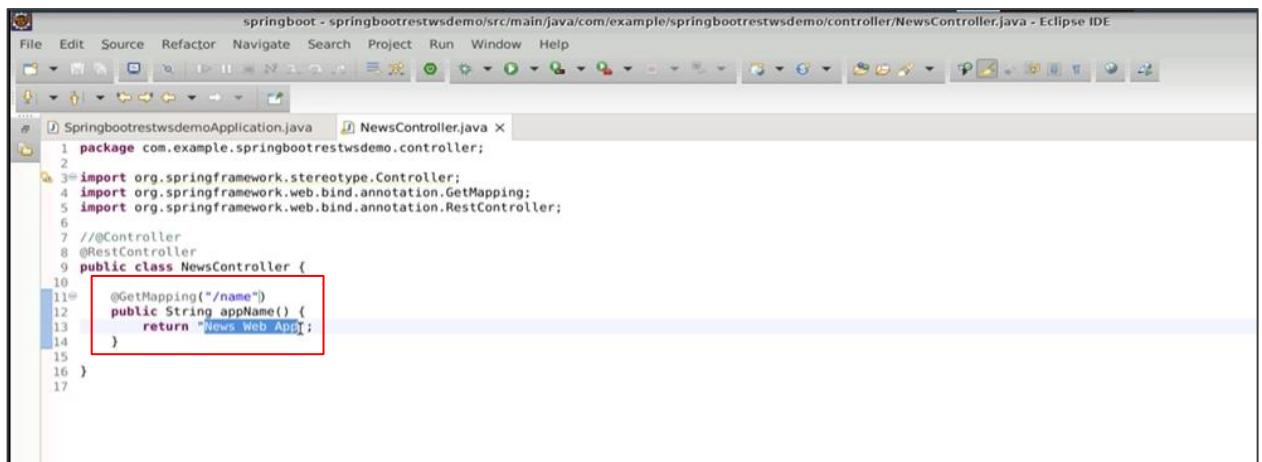
2.3 Annotate the class with **@controller**. When you mark your class as a controller using the annotation controller, you expect the class with the web method to return a template.



## 2.4 Add another annotation **@RestController**, which contains the response body

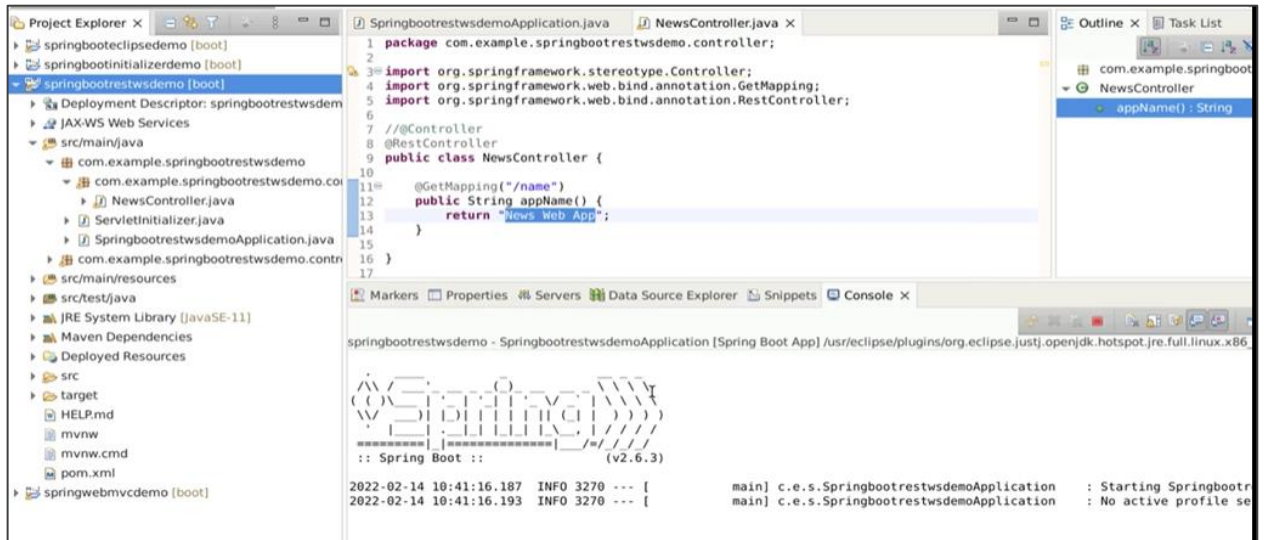


2.5 To create a RESTful API endpoint, start by annotating a **public** method with **@GetMapping("/name")** under the **@RestController** annotation. This method defines a public **String** method called **appName()** that returns the statement **News Web App**. Unlike a regular controller, where the response is typically a template, in a REST controller, the response is the actual text or HTML content by default.

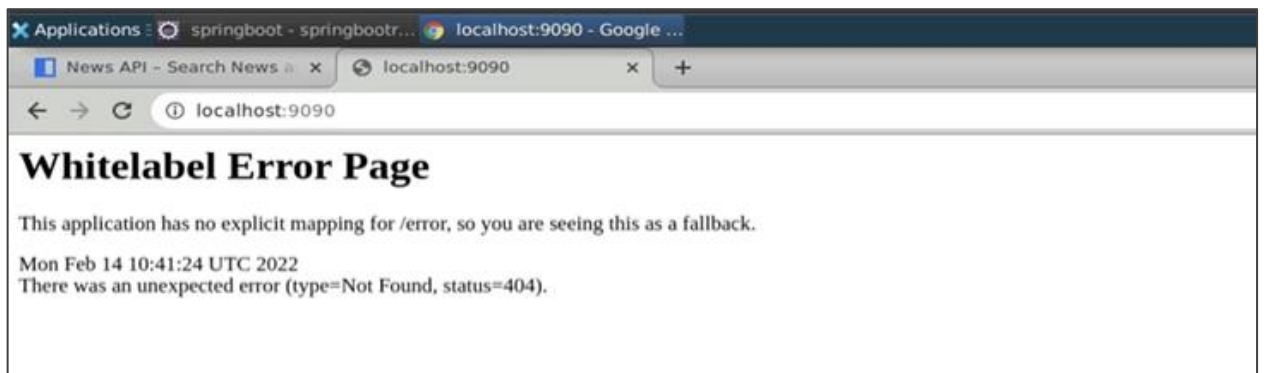




2.6 Run the file **springbootrestwsdemo** as **Spring Boot App** and the **Tomcat server** will start

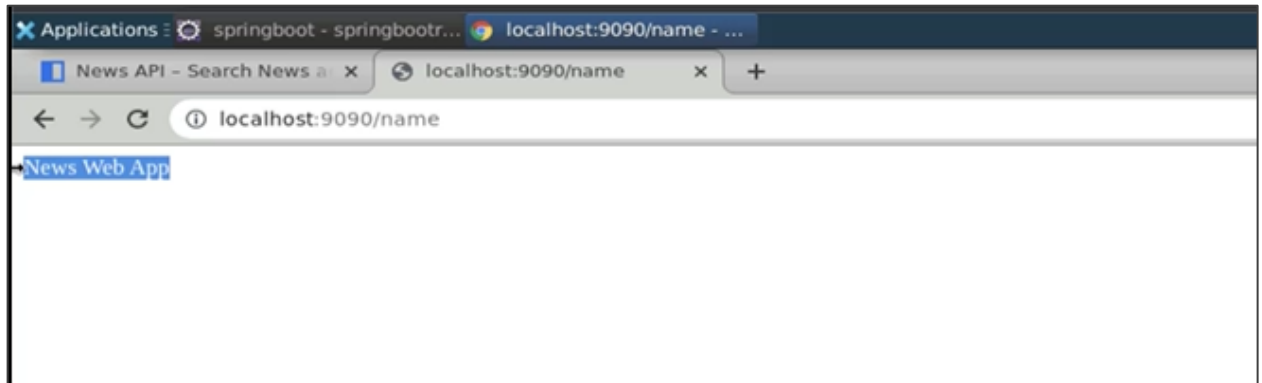


2.7 Search for **localhost 9090** in the browser. This will return an error because there is no default index page.

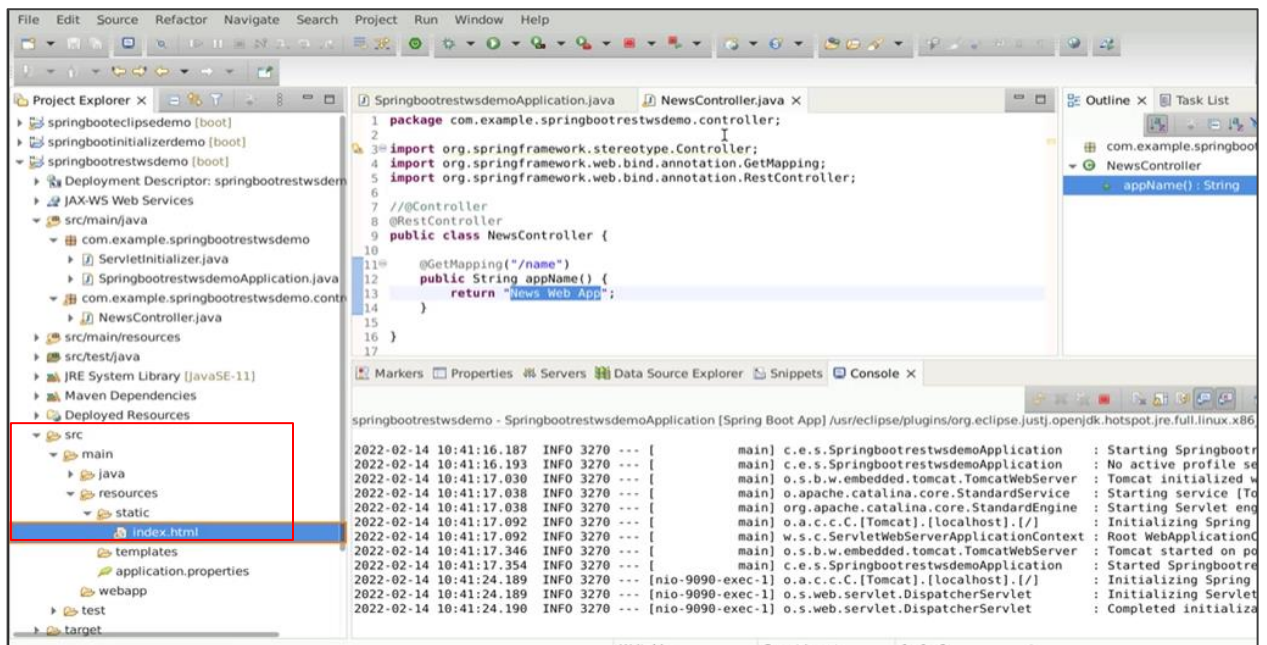




2.8 Now, add **/name** in the URL as **localhost:9090/name**. We have added this method to execute the return statement.



2.9 Take one of the Index pages from previously developed projects. You will have one static page **index.html** from a previous project.

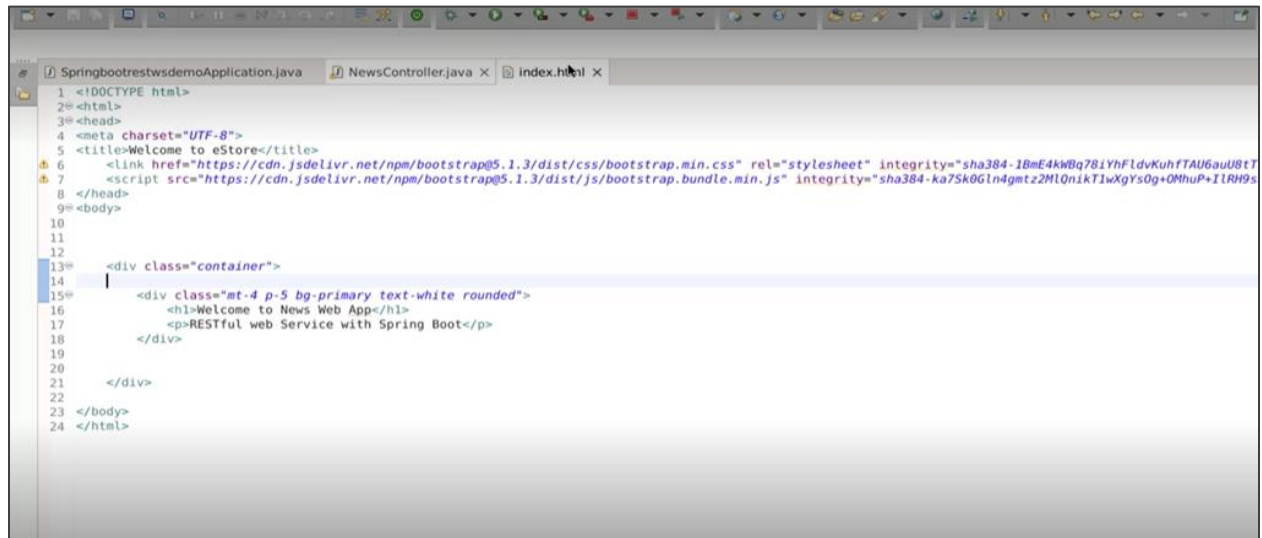


2.10 In the **index** page, remove the navigation bar and the login page.

```
File Edit Source Navigate Search Project Run Window Help
SpringbootrestwdsdemoApplication.java NewsController.java index.html x
7 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYs0g+OMhuP+IlR9H9sE
8 </head>
9 <body>
10
11
12
13 <div class="container">
14
15 <nav class="navbar navbar-expand-lg navbar-light bg-light">
16 <div class="container-fluid">
17 <a class="navbar-brand" href="#">eStore</a>
18 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-e
19 </button>
20 <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
21 <div class="navbar-nav">
22 <a class="nav-link active" aria-current="page" href="#">Home</a>
23 <a class="nav-link" href="register.html">Register</a>
24 <a class="nav-link" href="#">Products</a>
25 <a class="nav-link disabled">Account</a>
26 </div>
27 </div>
28 </div>
29 </nav>
30
31
32
33 <div class="mt-4 p-5 bg-primary text-white rounded">
34 <h1>Welcome to eStore</h1>
35 <p>Shop from Clothing to Electronics to Toys in Not Time !!</p>
36 </div>
37
38
39 <h3>Login Here</h3>
40
```

```
File Edit Source Navigate Search Project Run Window Help
SpringbootrestwdsdemoApplication.java NewsController.java *index.html x
6 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFIdvKuhfTAU6auU8tT9
7 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYs0g+OMhuP+IlR9H9sE
8 </head>
9 <body>
10
11
12
13 <div class="container">
14
15
16
17
18 <div class="mt-4 p-5 bg-primary text-white rounded">
19 <h1>Welcome to eStore</h1>
20 <p>Shop from Clothing to Electronics to Toys in Not Time !!</p>
21 </div>
22
23 <br><br>
24 <h3>Login Here</h3>
25
26 <form action="/app/login" method="post">
27 <div class="mb-3">
28 <label for="email" class="form-label">Enter Your Email</label>
29 <input type="email" class="form-control" id="email" name="email" aria-describedby="emailHelp" style="width: 300px">
30 </div>
31 <div class="mb-3">
32 <label for="password" class="form-label">Enter Your Password</label>
33 <input type="password" class="form-control" id="password" name="password" style="width: 300px">
34 </div>
35 <button type="submit" class="btn btn-primary">LOGIN</button>
36 </form>
37
38 </div>
39
40 </body>
41 </html>
```

- 2.11 Replace `<h1> Welcome to News Web App` and `<p>Restful web service with Spring Boot</p>`. Now that you have an index speed, whenever you run the application, you will not find your web server giving errors. Save the file

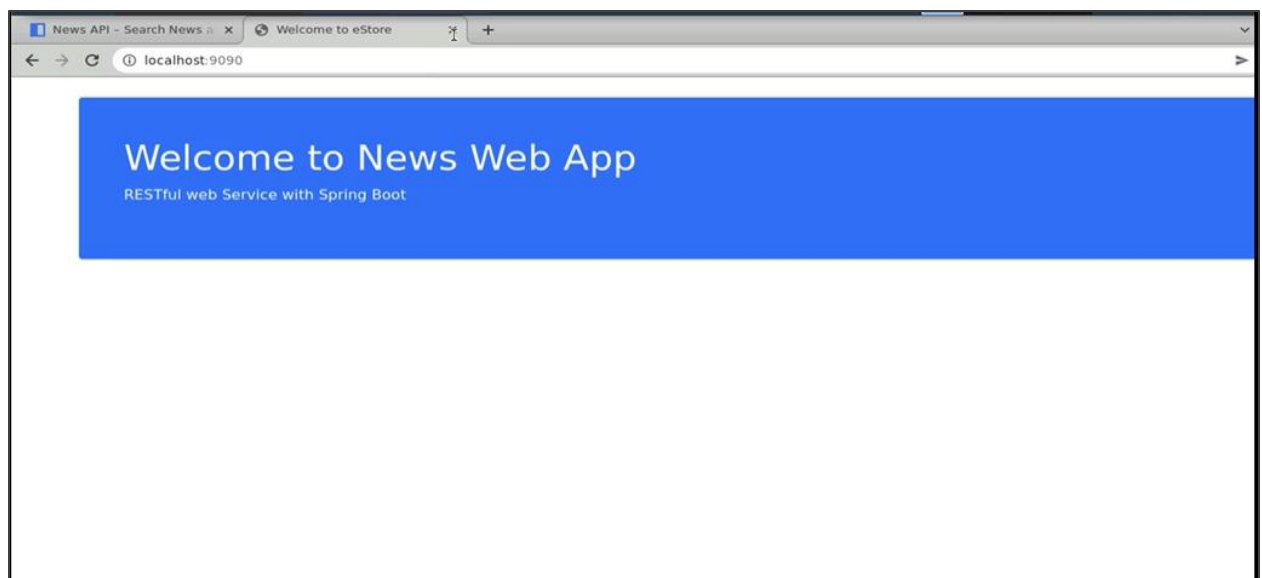


```

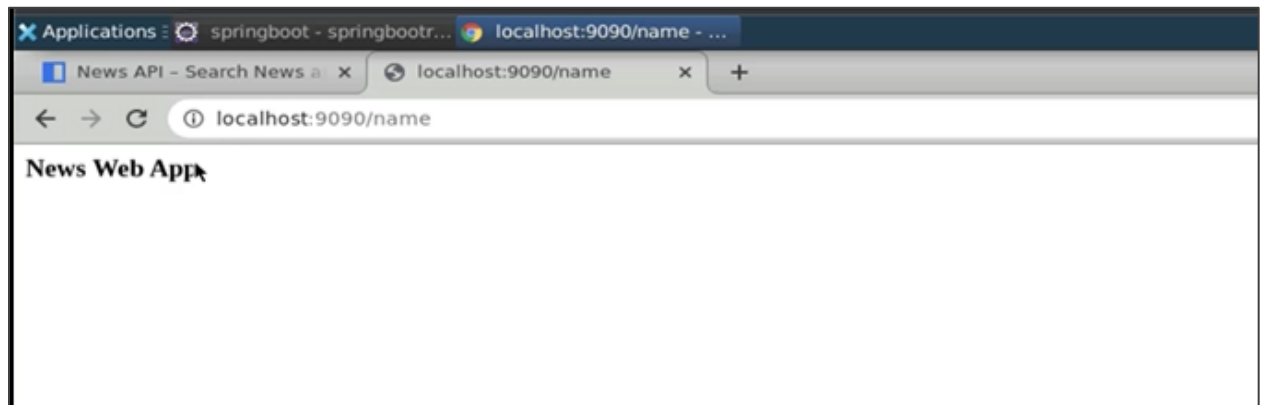
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset='UTF-8'>
5 <title>Welcome to eStore</title>
6 <link href='https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css' rel='stylesheet' integrity='sha384-1BmE4kWBq78iYhFtdvKuhfTAU6auu8tT
7 <script src='https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js' integrity='sha384-ka75k0GIn4gmtz2MlQn1kT1wXgYs0g+OMhuP+IlrH9s
8 </head>
9 <body>
10
11
12
13 <div class='container'>
14
15 <div class='mt-4 p-5 bg-primary text-white rounded'>
16 <h1>Welcome to News Web App</h1>
17 <p>RESTful web Service with Spring Boot</p>
18 </div>
19
20 </div>
21
22 </body>
23 </html>
24

```

- 2.12 With the configuration of this index page, return the HTML content in **NewController.java**. Stop the application and **rerun** the controller so that the changes are implied. Refresh the **localhost:9090** browser and the following output is shown:

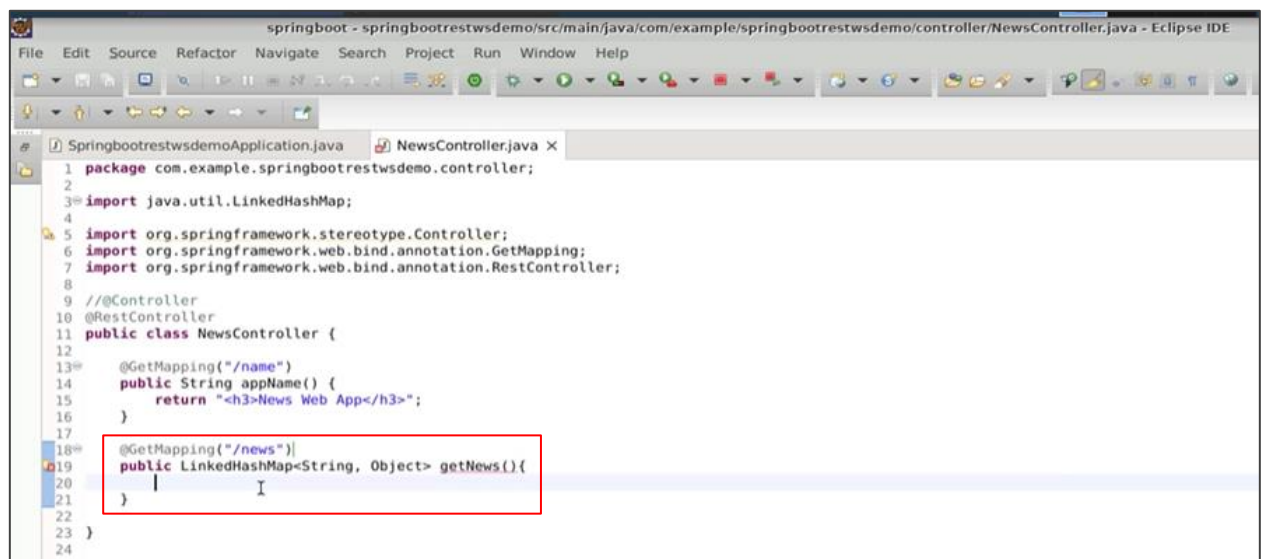


2.13 Add **/name** in the URL and the return statement in **h3 tag** is shown



### Step 3: Implementing maps with key-value pair

3.1 In the web method, add a **LinkedHashMap** method with the string, object, **getNews()**, and **@GetMapping** with **/news**



### 3.2 Create a **LinkedHashMap**, add a variable name **newsMap**, and start entering data

```

1 package com.example.springbootrestwsdemo.controller;
2
3 import java.util.LinkedHashMap;
4
5 import org.springframework.stereotype.Controller;
6 import org.springframework.web.bind.annotation.GetMapping;
7 import org.springframework.web.bind.annotation.RestController;
8
9 //Controller
10 @RestController
11 public class NewsController {
12
13     @GetMapping("/name")
14     public String appName() {
15         return "<h3>News Web App</h3>";
16     }
17
18     @GetMapping("/news")
19     public LinkedHashMap<String, Object> getNews(){
20
21         LinkedHashMap<String, Object> newsMap = new LinkedHashMap<String, Object>();
22         newsMap.put("status", "OK");
23         newsMap.put("total", 0);
24         newsMap.put("articles", new String[]{"Article1", "Article2", "Article3"});
25
26         return newsMap;
27     }
28 }
29
30

```

### 3.3 Use the **newsapi.org** website for mapping with the keys of **Top business headlines in the US**

News API | Get started | Documentation | Pricing | tuser6794@gmail.com

Locate articles and breaking news headlines from news sources and blogs across the web with our JSON API

[Get API Key →](#)

recent first →

2022-01-14&sortBy=pub 7

← Top business headlines in the US right now →

GET <https://newsapi.org/v2/top-headlines?country=us&category=business&apiKey=e2ee6e7d98724c8cbdf61865aebfbb7>

```

{
  status: "ok",
  totalResults: 70,
  articles: [
    {
      source: {
        id: null,
        name: "NDTV News"
      }
    }
  ]
}

```

← GET <https://newsapi.org/v2/top-headlines?country=us&category=business&apiKey=e7d98724c8cbdf61865aebfbb7>

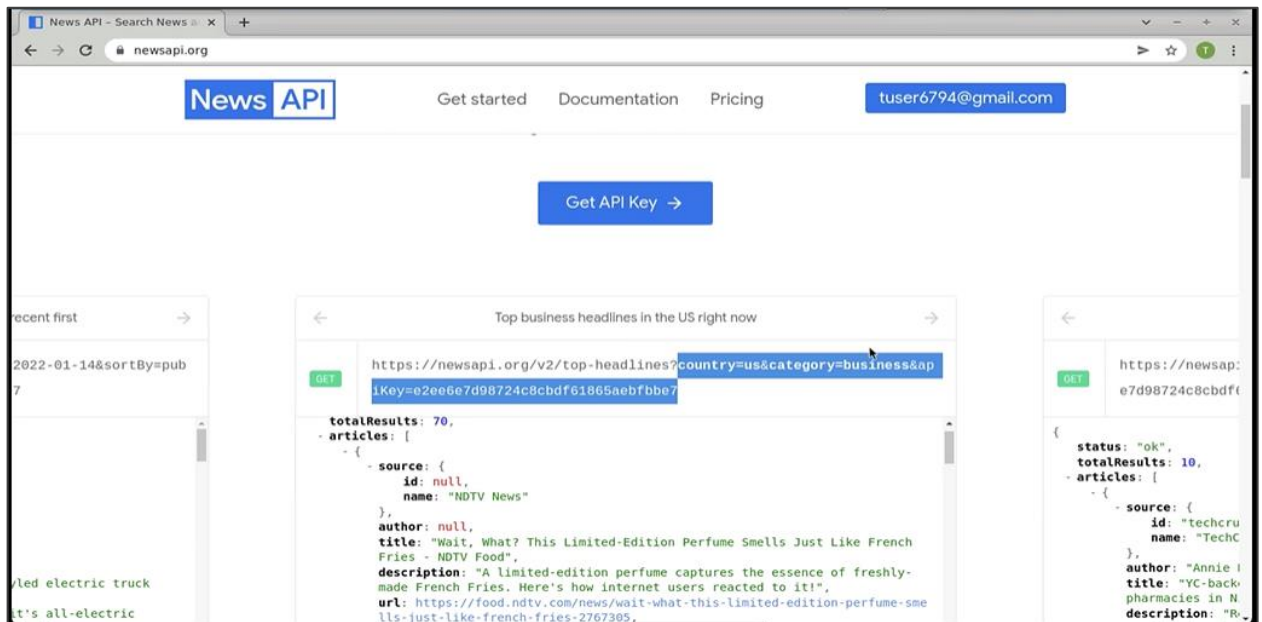
```

{
  status: "ok",
  totalResults: 10,
  articles: [
    {
      source: {
        id: "techcrunch",
        name: "TechCrunch"
      }
    }
  ]
}

```

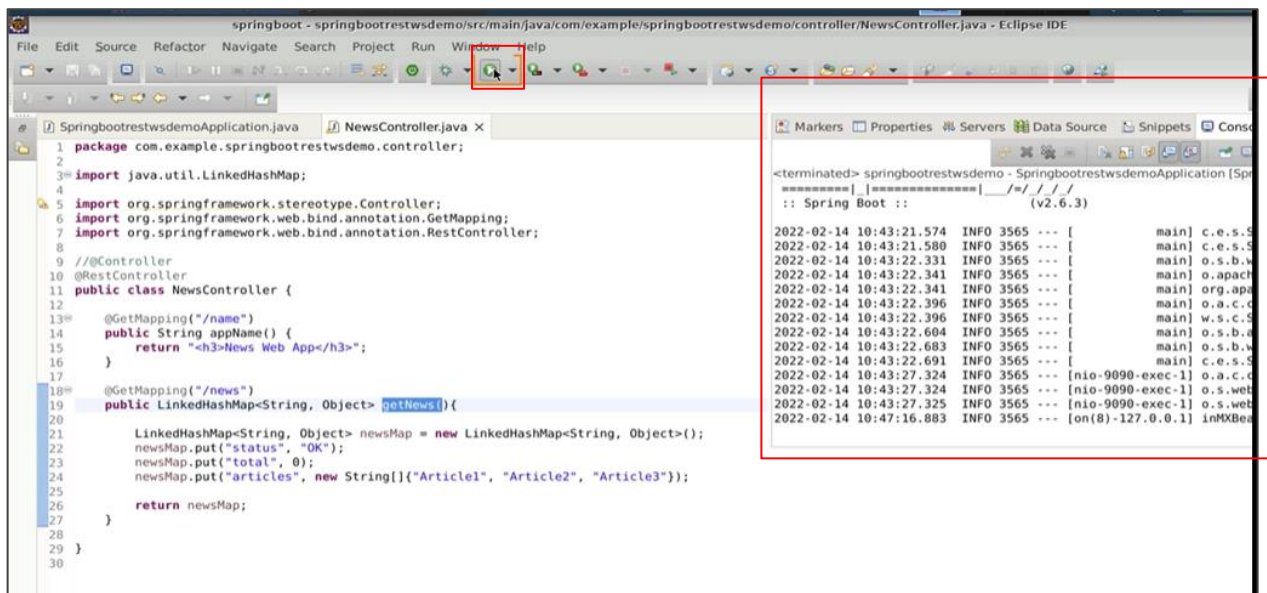


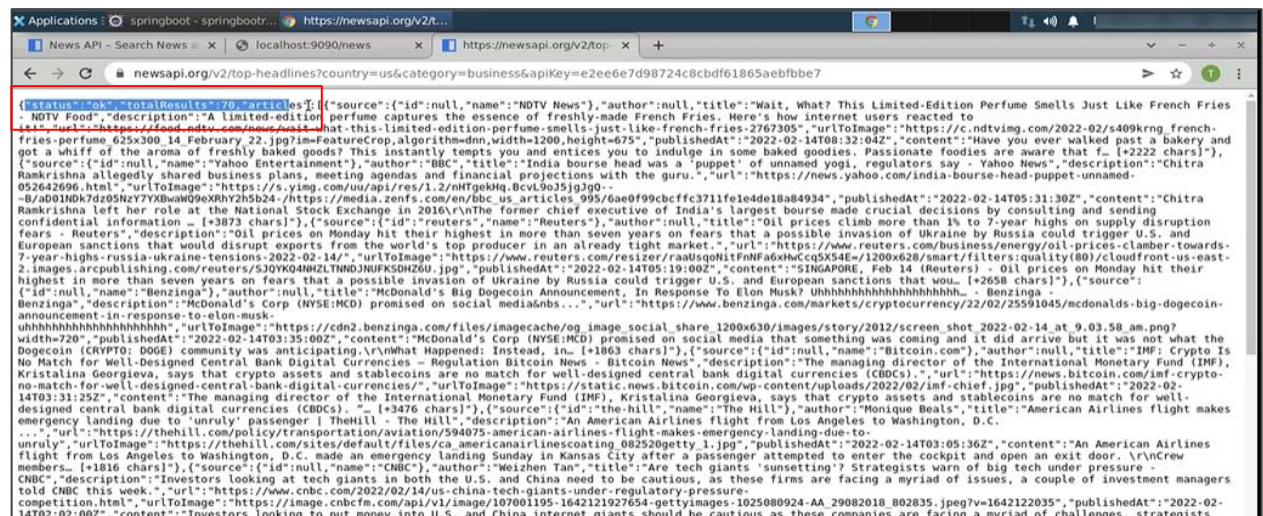
### 3.4 Use the **highlighted URL** to access the web service



Let us now map the keys from the headline section, like status, total, and articles.

### 3.5 Run the code added in step 3.2

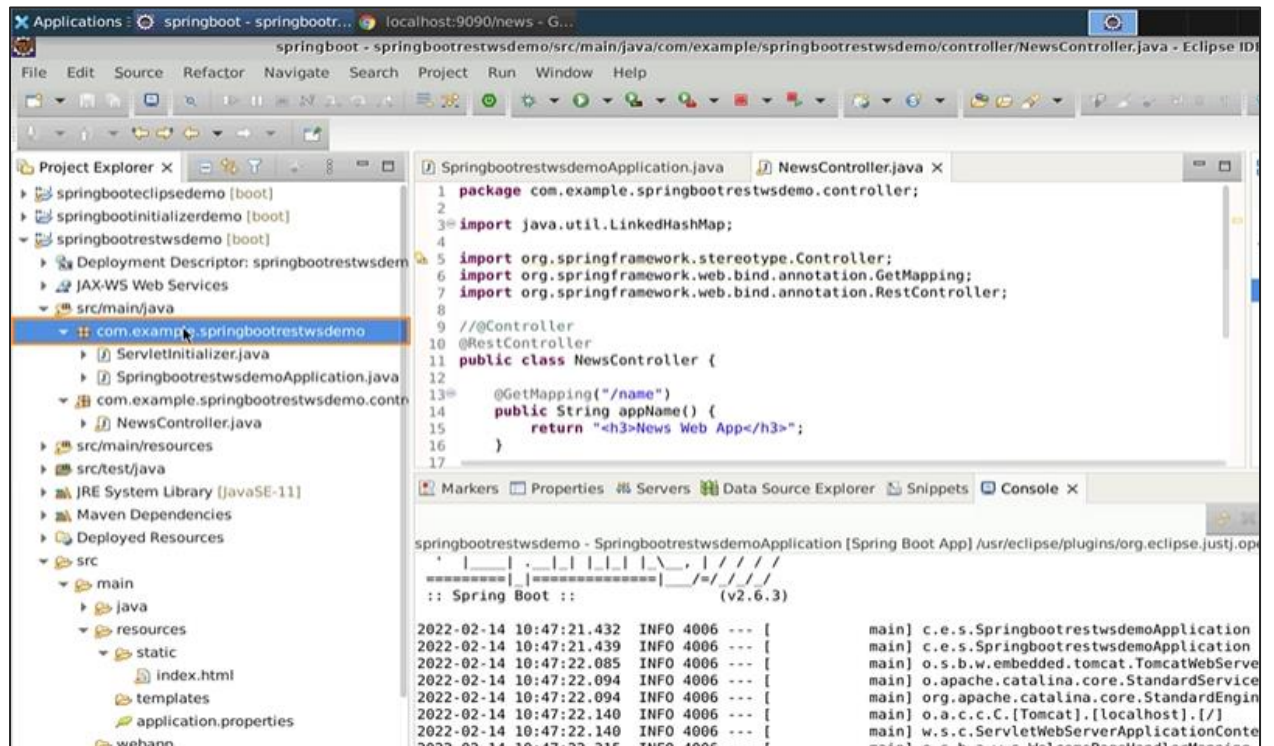




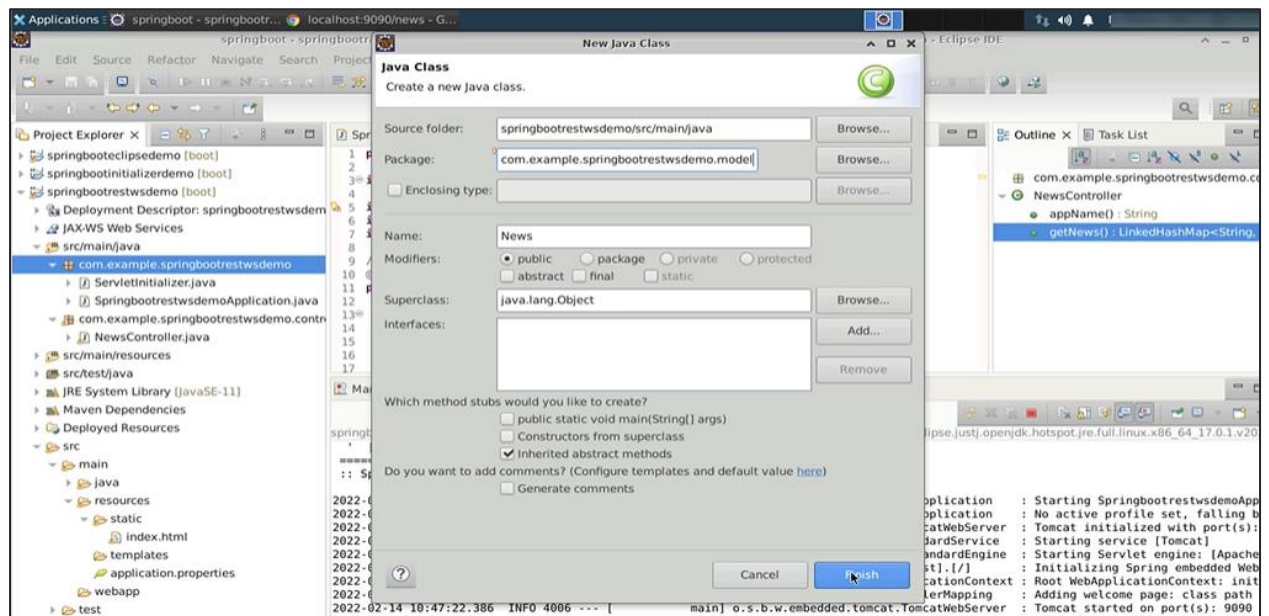


## Step 4: Creating a News Model class

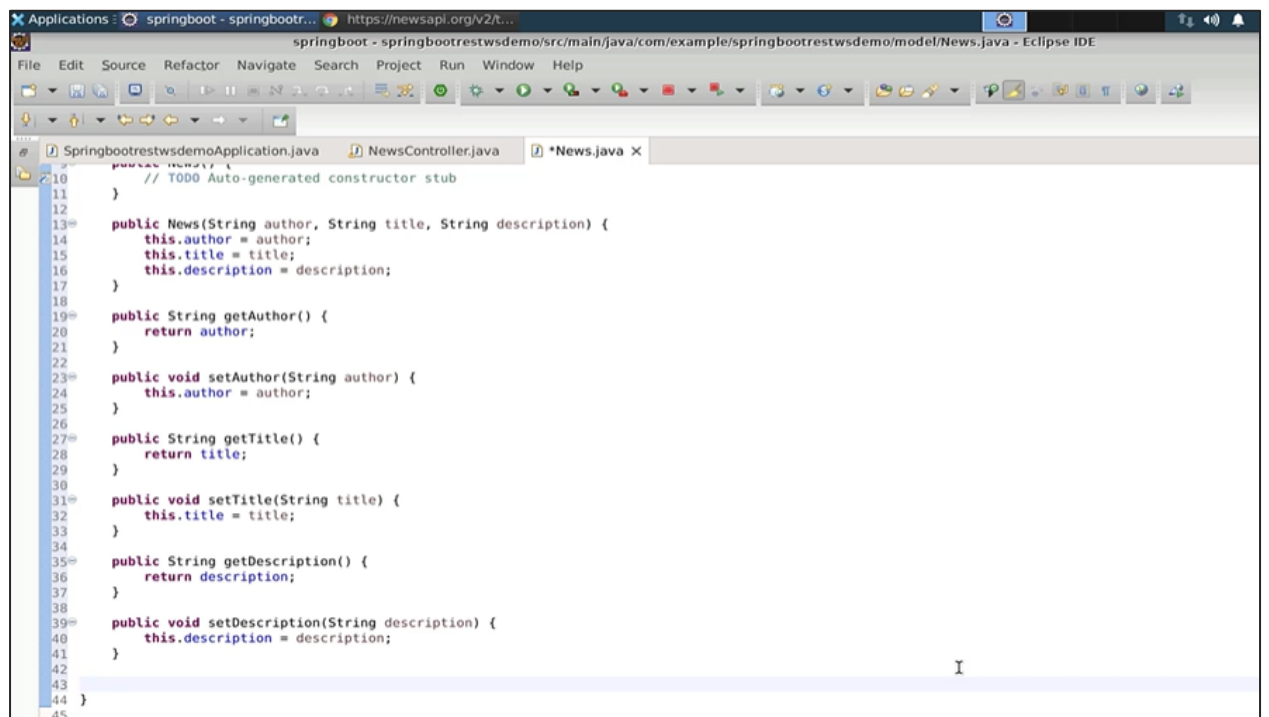
- 4.1 Create a new model class under the same parent package `com.example.springbootrestwsdemo` and **right-click** on the package > **New** > **Class**



#### 4.2 Name the class as **News** and add **.model** in the package



#### 4.3 Generate the default constructor, parameterized constructor, getters, setters, and a **toString()** method for the Address class



#### 4.4 Write `newsMap.put()` and add the key as articles and the value as the news list

```

17 public String appName() {
18     return "<h3>News Web App</h3>";
19 }
20
21 @GetMapping("/news")
22 public LinkedHashMap<String, Object> getNews(){
23
24     LinkedHashMap<String, Object> newsMap = new LinkedHashMap<String, Object>();
25     newsMap.put("status", "OK");
26     newsMap.put("total", 0);
27
28     //newsMap.put("articles", new String[]{"Article1", "Article2", "Article3"});
29
30
31     News news1 = new News("john", "News Article1", "Description of News Article1");
32     News news2 = new News("jennie", "News Article2", "Description of News Article2");
33     News news3 = new News("fionna", "News Article3", "Description of News Article3");
34     News news4 = new News("john", "News Article4", "Description of News Article4");
35     News news5 = new News("dave", "News Article5", "Description of News Article5");
36     News news6 = new News("john", "News Article6", "Description of News Article6");
37     News news7 = new News("jennie", "News Article7", "Description of News Article7");
38
39     ArrayList<News> newsList = new ArrayList<News>();
40     newsList.add(news1);
41     newsList.add(news2);
42     newsList.add(news3);
43     newsList.add(news4);
44     newsList.add(news5);
45     newsList.add(news6);
46     newsList.add(news7);
47
48     newsMap.put("total", newsList.size());
49     newsMap.put("articles", newsList);
50
51     return newsMap;
52 }

```

Ideally, the news objects can be pulled from the database.

#### 4.5 Return to the `localhost:9090/news`. You have the status, a total of seven, articles with the author, title, and description for the news, and other articles. This matches with the author, title, and the description kind of syntax.

```

{"status":"OK","total":7,"articles":[{"author":"john","title":"News Article1","description":"Description of News Article1"}, {"author":"jennie","title":"News Article2","description":"Description of News Article2"}, {"author":"fionna","title":"News Article3","description":"Description of News Article3"}, {"author":"john","title":"News Article4","description":"Description of News Article4"}, {"author":"dave","title":"News Article5","description":"Description of News Article5"}, {"author":"john","title":"News Article6","description":"Description of News Article6"}, {"author":"jennie","title":"News Article7","description":"Description of News Article7"}]}

```