



**Wyższa Szkoła Bankowa
w Gdańsku**

Java Server Pages

Konfiguracja projektu

Katarzyna Duchowska



JSP – konfiguracja projektu

- UWAGA!!
- Żeby uruchomić projekt bez problemów będzie potrzebny IntelliJ w wersji **Ultimate**

JSP – konfiguracja projektu

- Na stronie <https://start.spring.io/> można skonfigurować sobie wstępnie projekt w Spring Boocie (oczywiście w zależności od potrzeb można dobrać sobie więcej zależności):



Project
☒ Maven Project ☐ Gradle Project

Language
☒ Java ☐ Kotlin ☐ Groovy

Spring Boot
☐ 2.5.0 (SNAPSHOT) ☐ 2.4.2 (SNAPSHOT) ☒ 2.4.1 ☐ 2.3.8 (SNAPSHOT)
☐ 2.3.7

Project Metadata
Group
Artifact
Name
Description
Package name
Packaging ☒ Jar ☐ War
Java ☐ 15 ☐ 11 ☒ 8

Dependencies ADD DEPENDENCIES... CTRL + B

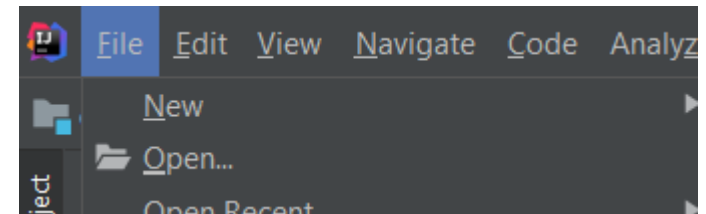
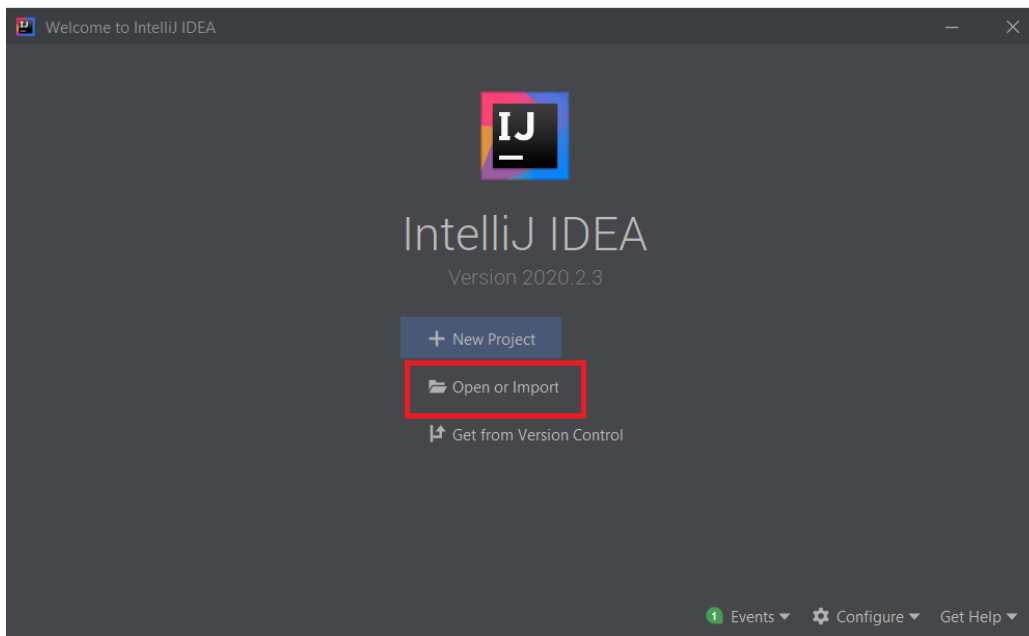
Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

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

GENERATE CTRL + G EXPLORE CTRL + SPACE SHARE...

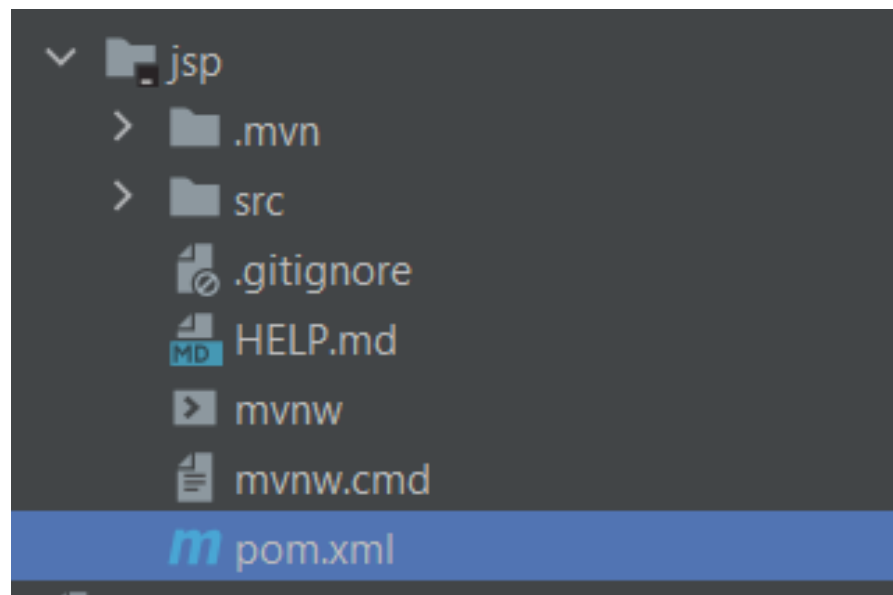
JSP – konfiguracja projektu

- Klikamy „*Generate*”, rozpakowujemy projekt na dysku
- Otwieramy IntelliJ – wybieramy opcję „*Open or Import*” (albo *File* → *Open*):



JSP – konfiguracja projektu

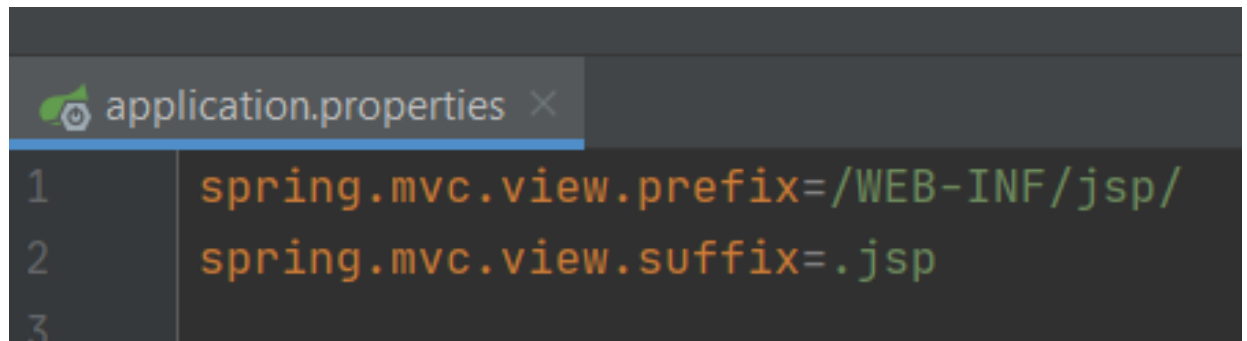
- W okienku dialogowym wyszukujemy nasz pobrany projekt i wybieramy do otwarcia plik *pom.xml*, klikamy „OK”, a następnie „Open as Project”:



JSP – konfiguracja projektu

- W pliku *src/main/resources/application.properties* podajemy ścieżki, w których będą przechowywane pliki JSP:

```
spring.mvc.view.prefix=/WEB-INF/jsp/  
spring.mvc.view.suffix=.jsp
```



```
application.properties x  
1 spring.mvc.view.prefix=/WEB-INF/jsp/  
2 spring.mvc.view.suffix=.jsp  
3
```

JSP – konfiguracja projektu

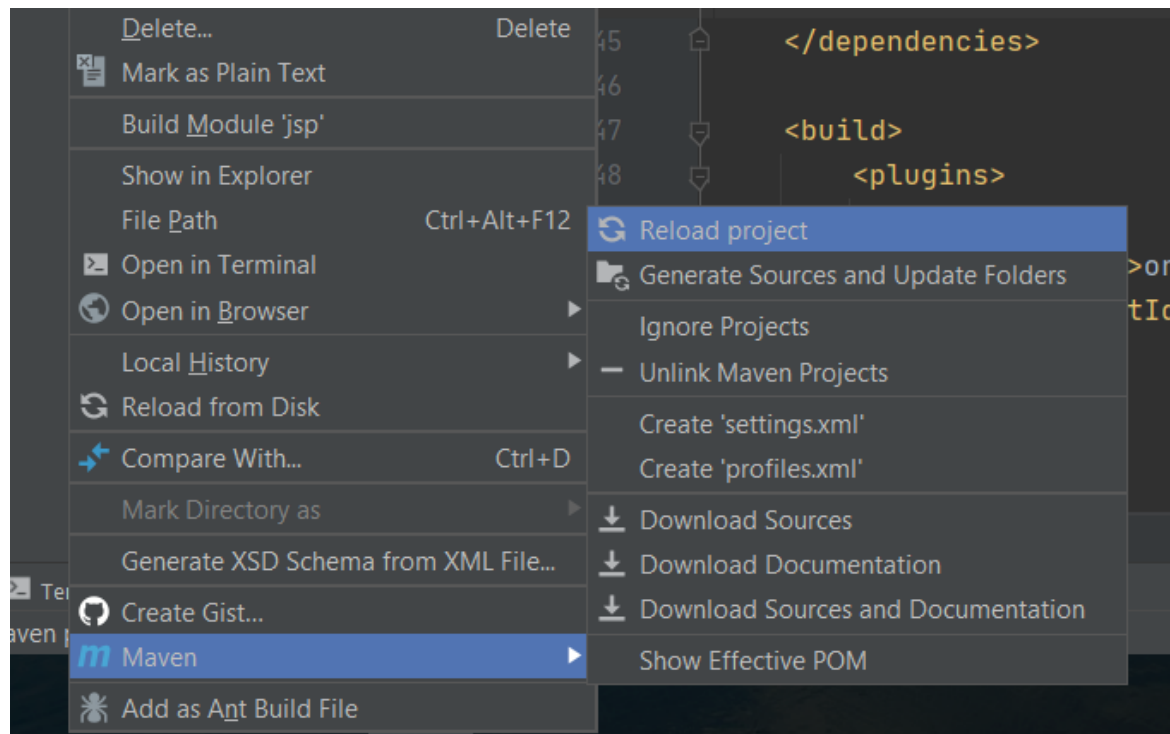
- W pliku *pom.xml* dodajemy kolejną zależność w węźle *<dependencies>*:

```
<dependency>  
  <groupId>org.apache.tomcat.embed</groupId>  
  <artifactId>tomcat-embed-jasper</artifactId>  
  <scope>provided</scope>  
</dependency>
```

```
<dependency>  
  <groupId>org.apache.tomcat.embed</groupId>  
  <artifactId>tomcat-embed-jasper</artifactId>  
  <scope>provided</scope>  
</dependency>  
</dependencies>
```

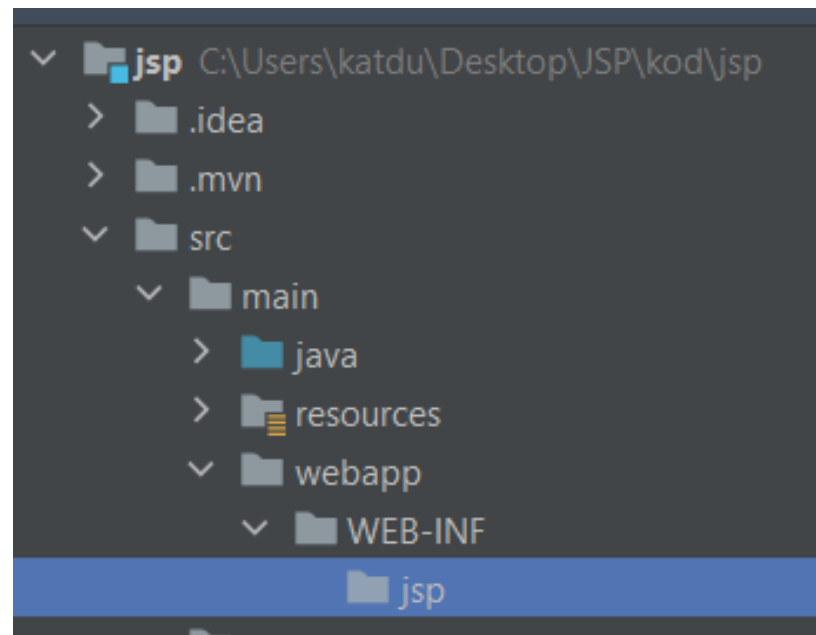
JSP – konfiguracja projektu

- Klikamy PPM na plik *pom.xml* i wybieramy opcję *Maven* → *Reload project*:



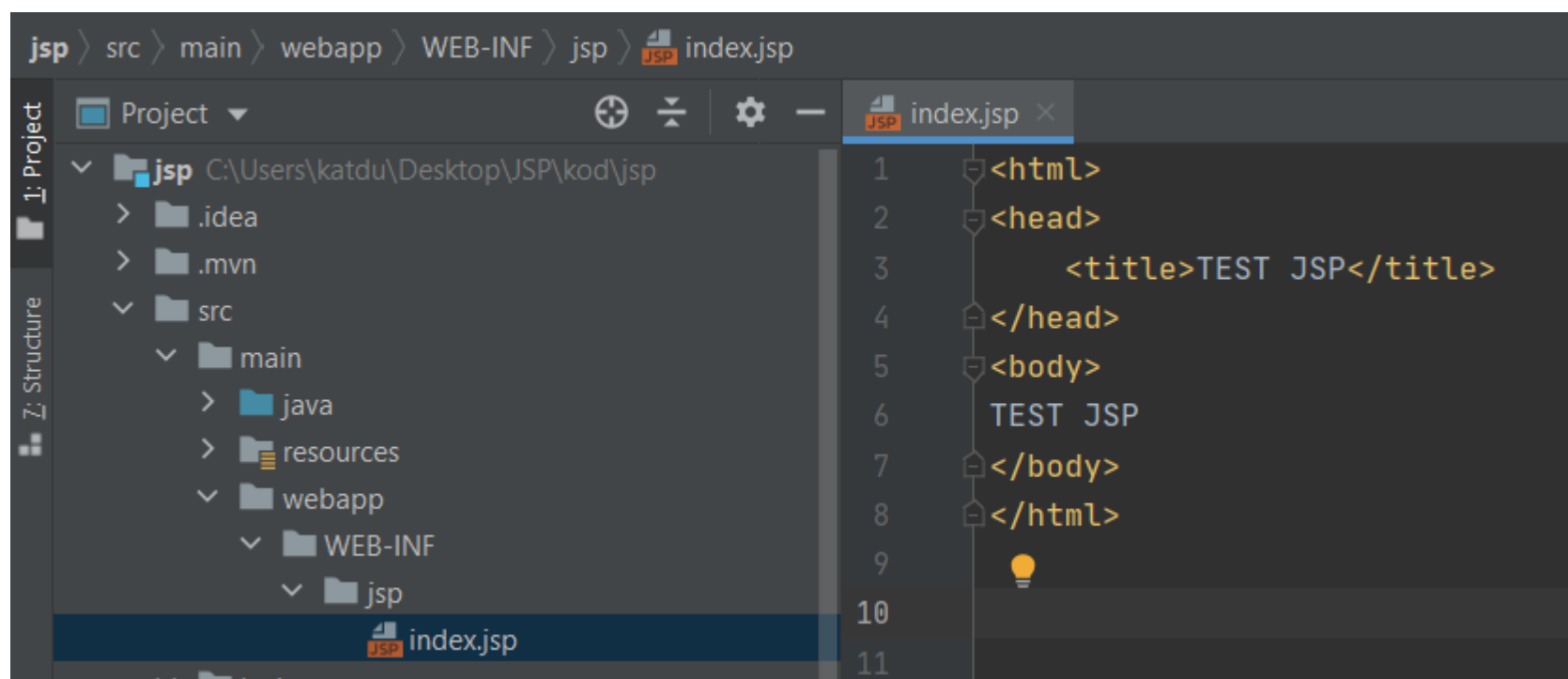
JSP – konfiguracja projektu

- Tworzymy foldery *main/webapp*, *main/webapp/WEB-INF* oraz *main/webapp/WEB-INF/jsp*:



JSP – konfiguracja projektu

- W folderze *main/webapp/WEB-INF/jsp* tworzymy testowy plik *.jsp* – może to być jakiś prosty HTML:



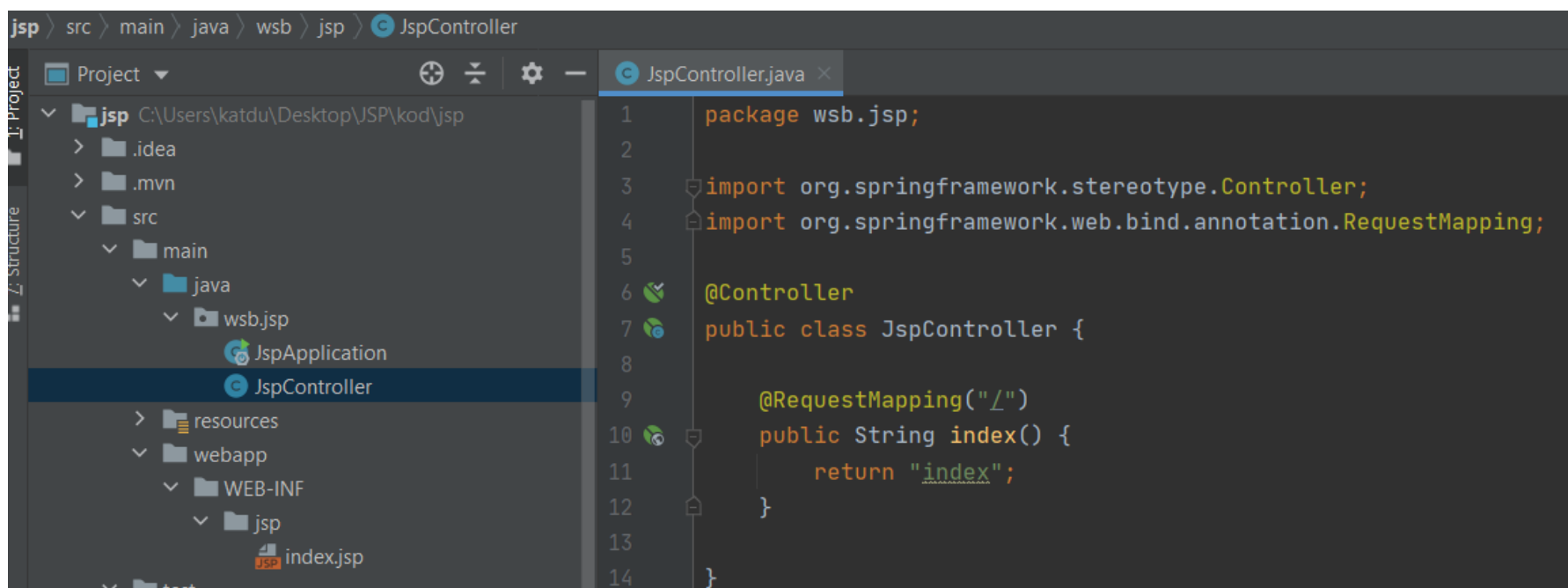
The screenshot shows an IDE with the following components:

- Project Explorer (Left):** Displays the project structure. The path `src > main > webapp > WEB-INF > jsp` is expanded, and the file `index.jsp` is selected.
- Editor (Right):** Shows the content of `index.jsp`, which is a simple HTML document.

```
1 <html>
2 <head>
3     <title>TEST JSP</title>
4 </head>
5 <body>
6     TEST JSP
7 </body>
8 </html>
9
10
11
```

JSP – konfiguracja projektu

- Tworzymy kontroler, który będzie zwracał nam stworzony testowy plik *.jsp*:



The screenshot shows an IDE with the following structure:

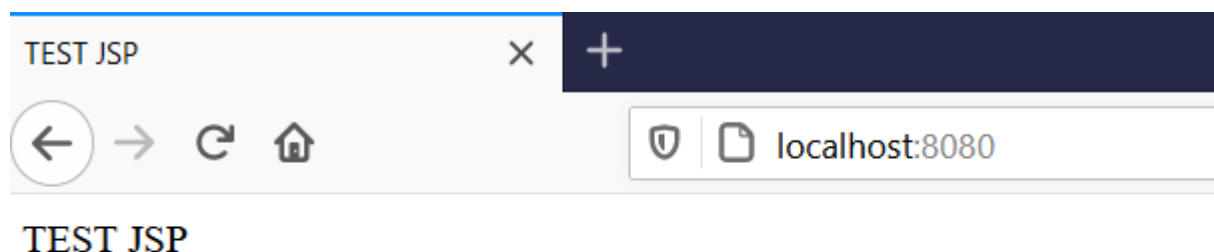
- Project Structure:**
 - Project: jsp (C:\Users\katdu\Desktop\JSP\kod\jsp)
 - .idea
 - .mvn
 - src
 - main
 - java
 - wsb.jsp
 - JspApplication
 - JspController** (selected)
 - resources
 - webapp
 - WEB-INF
 - jsp
 - index.jsp
 - test

- JspController.java Code:**

```
1 package wsb.jsp;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5
6 @Controller
7 public class JspController {
8
9     @RequestMapping("/")
10    public String index() {
11        return "index";
12    }
13
14 }
```

JSP – konfiguracja projektu

- Po uruchomieniu projektu i przejściu na *localhost:8080* powinniśmy zobaczyć stworzoną stronę testową:



Dodanie JSTL

- Dodajemy do pliku *pom.xml* wpis:

```
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```

```
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>

</dependencies>
```

Dodanie JSTL

- Ponownie klikamy na PPM na plik *pom.xml* i wybieramy opcję *Maven → Reload project*
- Modyfikujemy plik *index.jsp* i uruchamiamy projekt ponownie, żeby sprawdzić, czy działa JSTL:

```
index.jsp
1  <%@ taglib prefix="c"
2      uri="http://java.sun.com/jsp/jstl/core" %>
3
4  <html>
5  <head>
6      <title>TEST JSP</title>
7  </head>
8  <body>
9      TEST JSP
10
11     <c:if test="${3 == 3}">
12         <div>
13             JSTL JEST OK!
14         </div>
15     </c:if>
16
17 </body>
18 </html>
```

