Java Spring (Non-Boot) Servlet + MVC TODO Project

Beginner-friendly guide to build a **Spring MVC + Servlet + Tomcat** TODO application **without Spring Boot**.

This includes: - Project structure - All required files (Servlet, Controller, Service, JSP, web.xml) - Full working code - How to run on Tomcat

---

✅1. Project Goal

Create a simple TODO App where you can: - View all todos - Add a todo

Using: - **Spring MVC (DispatcherServlet)** - **Servlet Container: Tomcat** - **JSP for UI** - **In-memory List** for storage

---

✅2. Project Structure

```
SpringTodoApp/
│
├── src/main/java/
│   └── com/example/todo/
│       ├── controller/
│       │   └── TodoController.java
│       ├── model/
│       │   └── Todo.java
│       └── service/
│           └── TodoService.java
│
├── src/main/webapp/
│   ├── WEB-INF/
│   │   ├── web.xml
│   │   └── todo-servlet.xml
│   ├── views/
│   │   ├── todo-list.jsp
│   │   └── add-todo.jsp
│
└── pom.xml
```

✅3. pom.xml (Spring MVC + JSP)

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://
maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>SpringTodoApp</artifactId>
    <version>1.0</version>
    <packaging>war</packaging>

    <dependencies>
        <!-- Spring MVC -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.3.30</version>
        </dependency>

        <!-- JSP -->
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jstl</artifactId>
            <version>1.2</version>
        </dependency>
    </dependencies>

    <build>
        <finalName>SpringTodoApp</finalName>
    </build>
</project>
```

✅4. web.xml (Configure DispatcherServlet)

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://
xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         version="3.1">

    <servlet>
        <servlet-name>todo</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</
servlet-class>
```

```
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/todo-servlet.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>todo</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

✅5. Spring MVC Config (todo-servlet.xml)

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
        http://www.springframework.org/schema/beans http://
www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context http://
www.springframework.org/schema/context/spring-context.xsd">

    <!-- Enable @Controller -->
    <context:component-scan base-package="com.example.todo" />

    <!-- JSP View Resolver -->
    <bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/views/" />
        <property name="suffix" value=".jsp" />
    </bean>

</beans>
```

✅6. Model: Todo.java

```
package com.example.todo.model;

public class Todo {
    private String title;

    public Todo() {}
```

```java
    public Todo(String title) {
        this.title = title;
    }

    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }
}
```

✅7. Service: TodoService.java

```java
package com.example.todo.service;

import com.example.todo.model.Todo;
import org.springframework.stereotype.Service;
import java.util.*;

@Service
public class TodoService {

    private List<Todo> todos = new ArrayList<>();

    public List<Todo> getAllTodos() {
        return todos;
    }

    public void addTodo(String title) {
        todos.add(new Todo(title));
    }
}
```

✅8. Controller: TodoController.java

```java
package com.example.todo.controller;

import com.example.todo.service.TodoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

@Controller
public class TodoController {
    @Autowired
    private TodoService service;

    @GetMapping("/")
    public String home(Model model) {
        model.addAttribute("todos", service.getAllTodos());
        return "todo-list";
    }

    @GetMapping("/add")
    public String addPage() {
        return "add-todo";
    }

    @PostMapping("/addTodo")
    public String addTodo(@RequestParam String title) {
        service.addTodo(title);
        return "redirect:/";
    }
}
```

✅9. JSP Pages

☸todo-list.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<html>
<head><title>Todo List</title></head>
<body>
<h1>All Todos</h1>
<ul>
    <c:forEach var="t" items="${todos}">
        <li>${t.title}</li>
    </c:forEach>
</ul>
<a href="add">Add New Todo</a>
</body>
</html>
```

☸add-todo.jsp

```
<html>
<head><title>Add Todo</title></head>
<body>
<h1>Add Todo</h1>
```

```
<form action="addTodo" method="post">
    <input type="text" name="title" placeholder="Todo title" />
    <button type="submit">Add</button>
</form>
</body>
</html>
```

✅10. Run the Project

1. Run `mvn clean package`
2. A WAR file gets created:

   ```
   target/SpringTodoApp.war
   ```

3. Copy WAR → Tomcat `webapps/`
4. Start Tomcat
5. Open:

   ```
   http://localhost:8080/SpringTodoApp/
   ```

‼️ Your Spring Servlet + MVC TODO Project is Ready!

If you want, I can also add: - JDBC database storage - CRUD (Update/Delete) - Filters + Interceptors - Login system with Sessions - Convert this into Spring Boot version - Create a downloadable ZIP template

Just tell me!