07 - Summary

1. Project Setup:

- Start by creating a Maven Project.
- Add the required dependency for **Spring Web MVC** in the pom.xml file to support Spring MVC functionality.

2. Controller Creation:

- Define a controller class that contains various methods to handle HTTP requests.
- Use annotations such as @Controller and @RequestMapping to map URLs to specific methods.

3. DispatcherServlet Configuration:

- In Spring MVC, the **DispatcherServlet** acts as the Front Controller.
- To use the DispatcherServlet, configure it in the **web.xml** file by mapping it to a URL pattern, which intercepts requests.

4. DispatcherServlet File Setup:

- Create a Spring configuration file, named **[DispatcherServletName]-servlet.xml** (e.g., telusko-servlet.xml), inside the WEB-INF folder.
- This file configures the Spring MVC container and instructs it to scan for components and annotations.

5. Component Scan and Annotation Config:

- Use <ctx:component-scan> to specify which package(s) should be scanned for Spring components.
- Use <ctx:annotation-config> to indicate that annotations (such as @Controller and @RequestMapping) will be used for configuration and mapping.

6. View Resolver Setup:

- Configure **InternalResourceViewResolver** to define where the views (e.g., JSP pages) are stored and what file type they are.
- Set the prefix to the folder path and the suffix to the file extension (e.g., .jsp) to resolve the view names returned by controllers.

👉**WorkFlow:**

➢ A request comes to DispatcherServlet, which routes it to the appropriate controller method.

➢ The controller processes the request and returns a view name.

➢ The InternalResourceViewResolver resolves the view name to a JSP file located in a specific directory and renders it to the client.