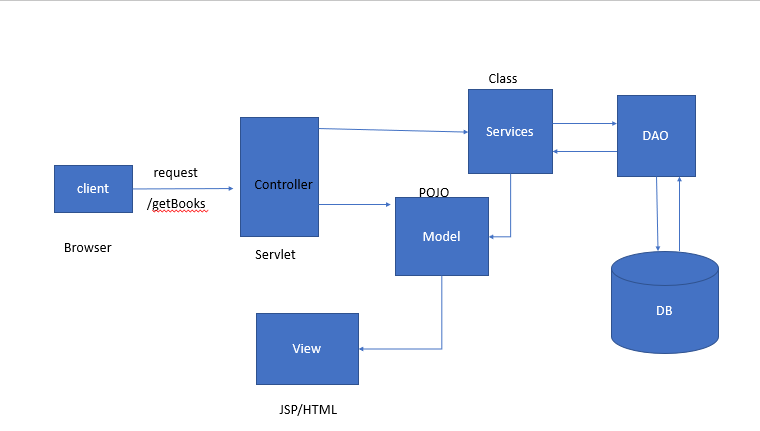


Java MVC

MVC using Servlet and JSP

HTML is the static page

JSP is dynamic view



Servlet(server components) : In Java MVC servlet will act as controller, is web component with resides on server , exeute on server and send response to client.

Request mapping in JAVA MVC

1. Add the server to web.xml file to get it identified by web container.
2. Or add the annotation @webServlet(UrlPatter= { “/Add “ }) to add in web.xml

Public class <Servlet\_name> extends HttpServlet{

}

public void init() throws ServletException {

// Initialization code...

}

public void service(ServletRequest request, ServletResponse response)

throws ServletException, IOException {

int k = Integer.ParseInt(request.getParameter(“t1”);

//writing into the html file

PrintWriter out = response.getWriter();

Out.println(k);

}

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Servlet code

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Servlet code

}

Mapping configuration will be present in web.xml.

POJO : Plain Old Java Object

DAO : Data Access Object

HTML : Hyper Test Markup Language

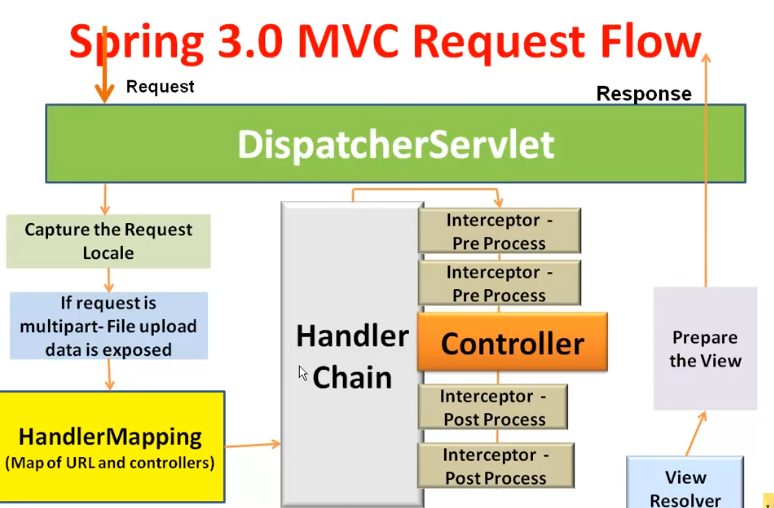
JSP : Java Server Page

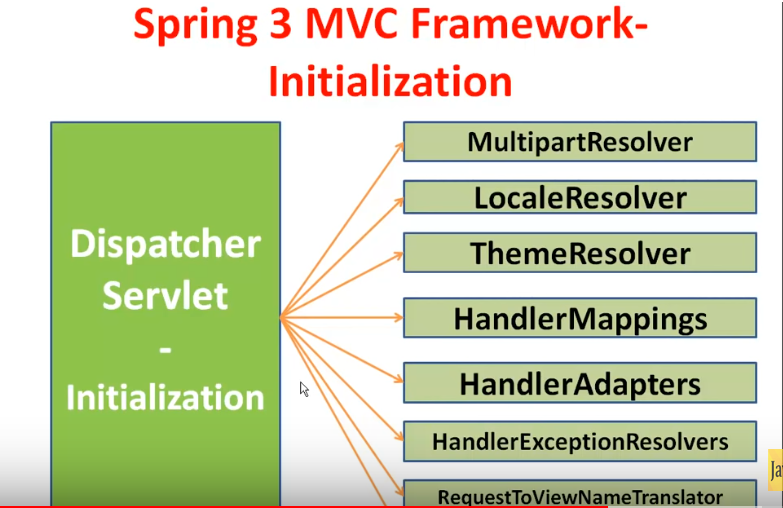
**Spring MVC**

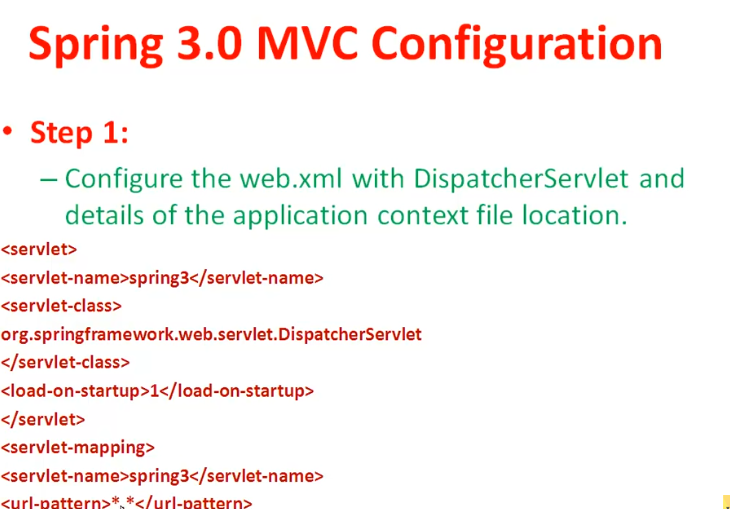
Advantage

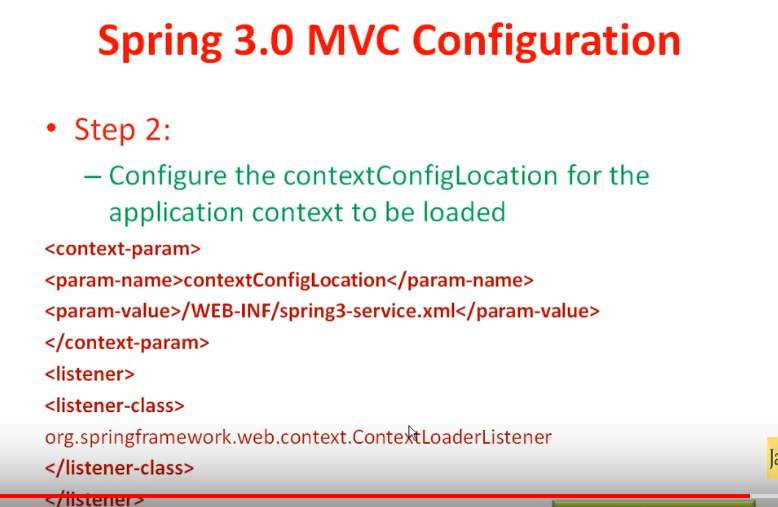


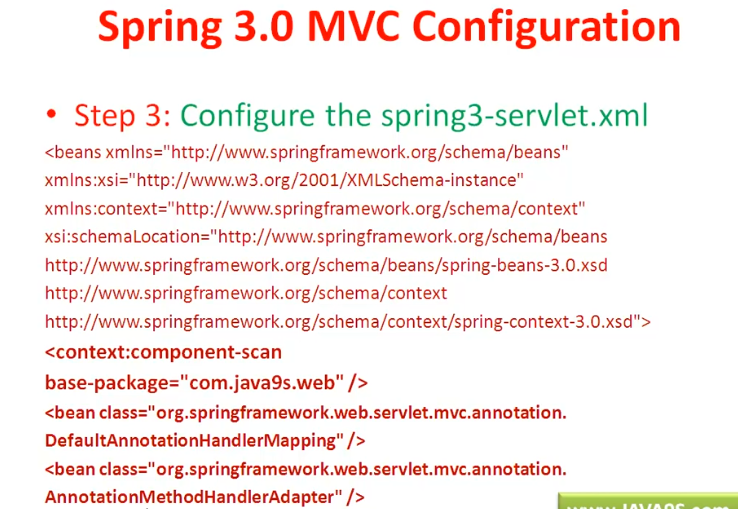
****

****

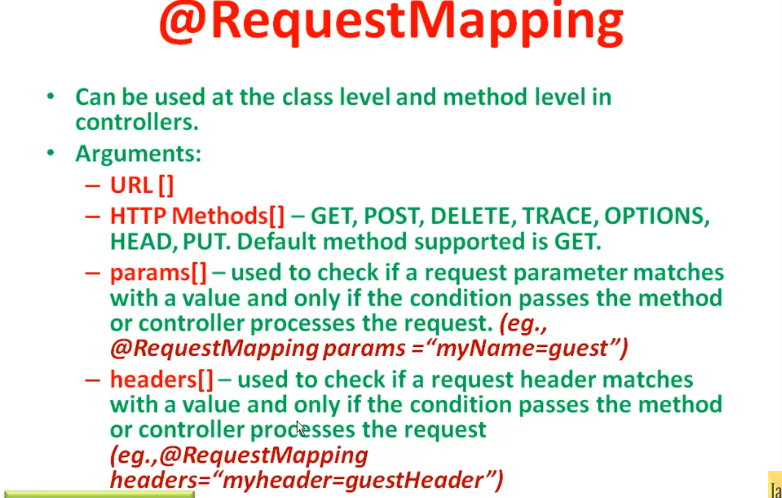
****

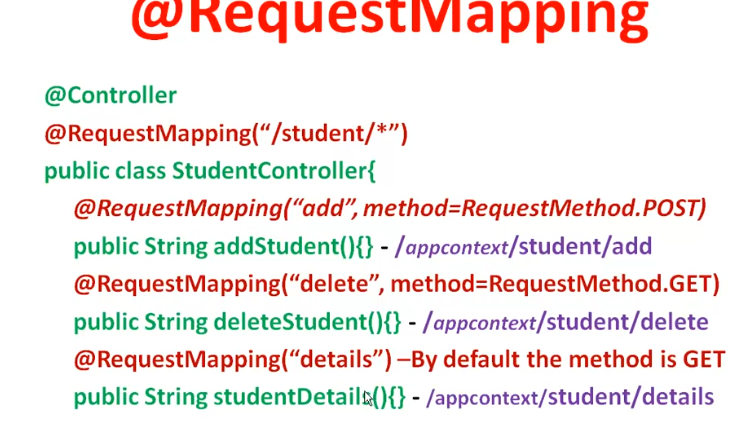
****

****

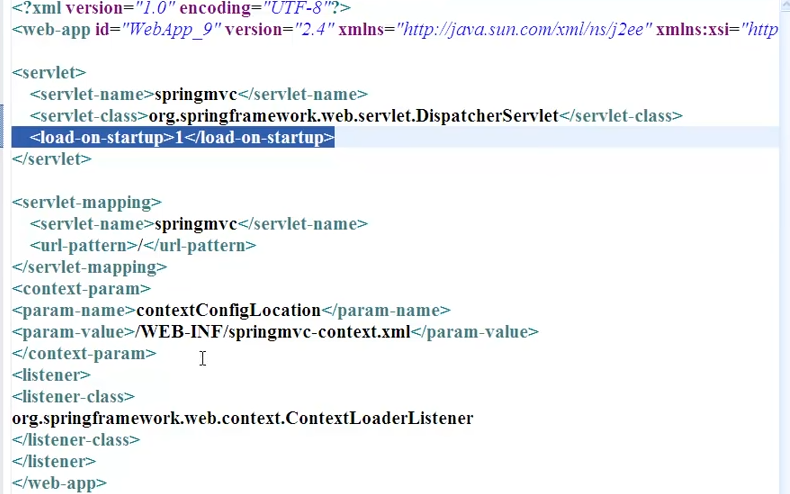
****

****

****

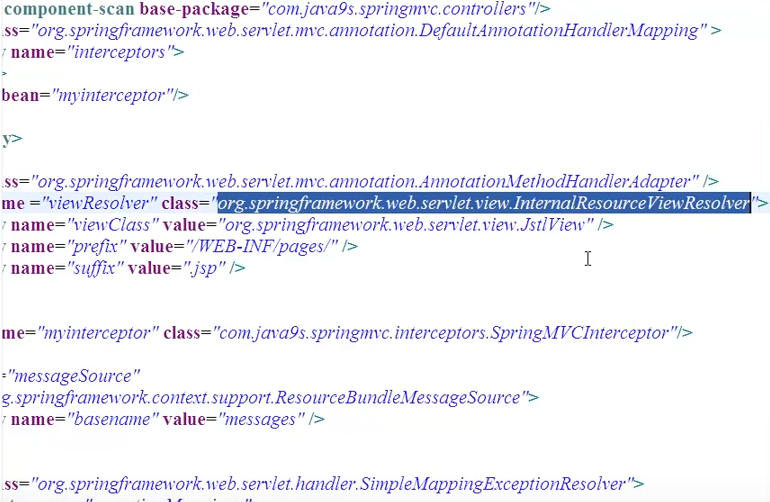
****

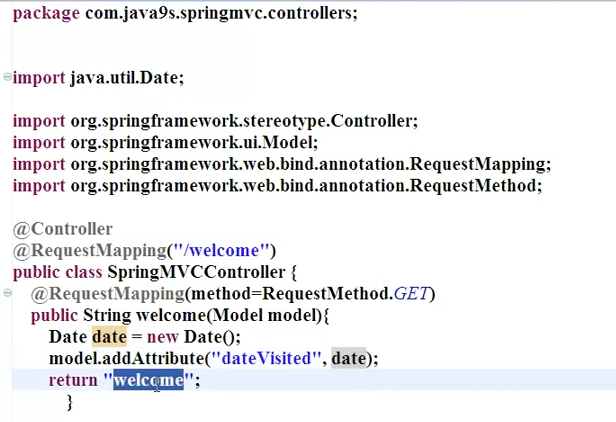
**Web.xml**

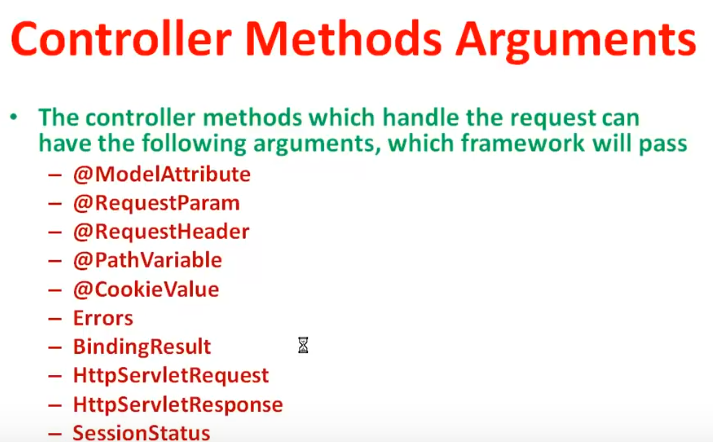
****

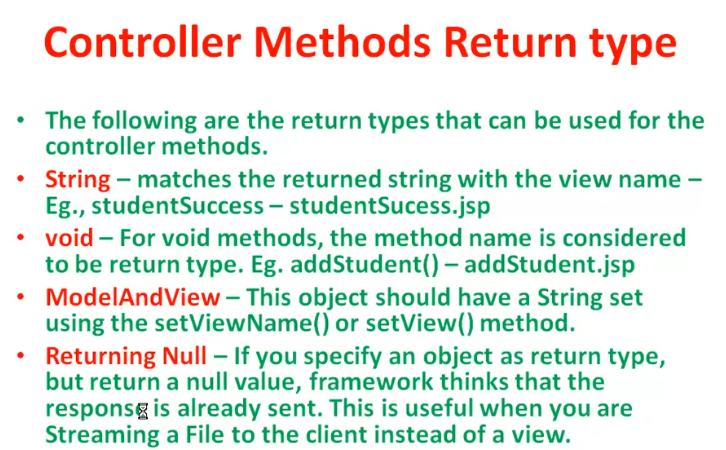
**Servlet.xml**

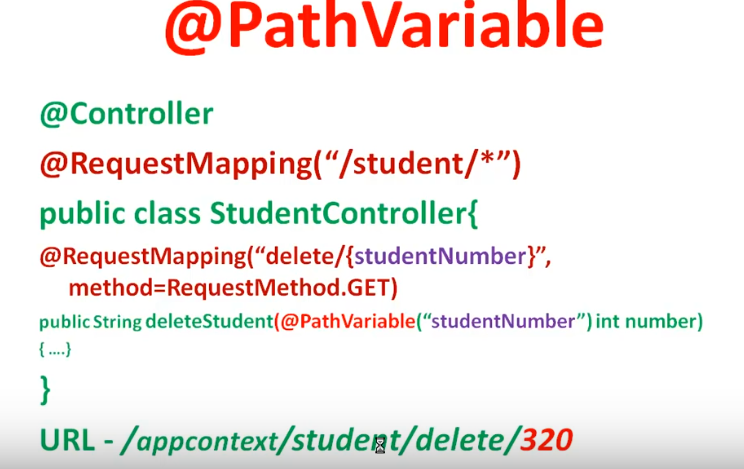
****

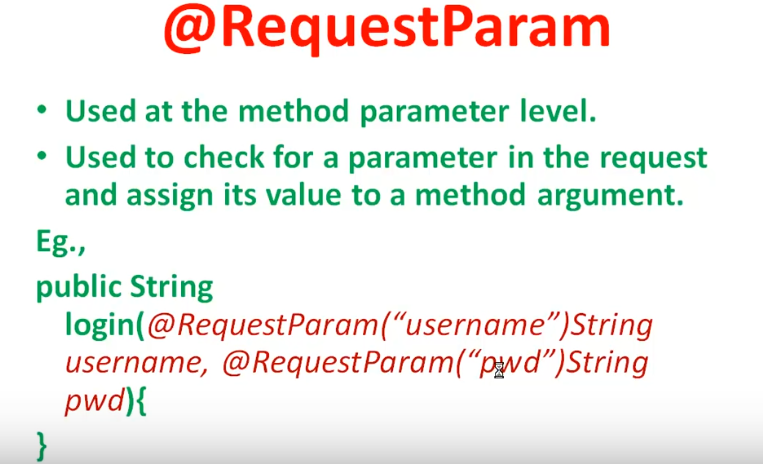
****

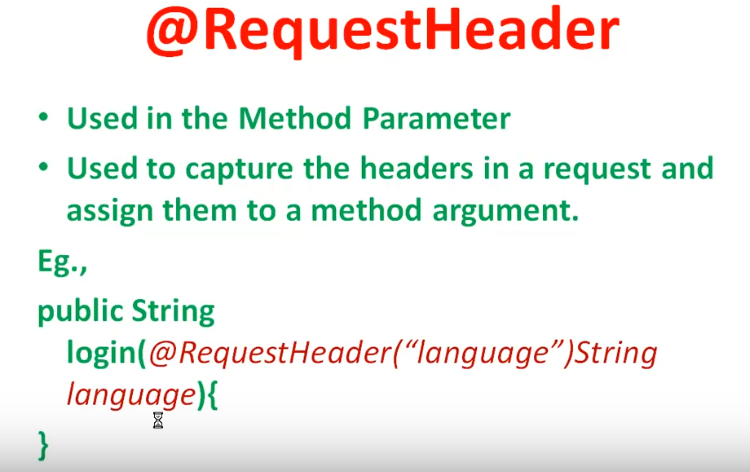
****

****

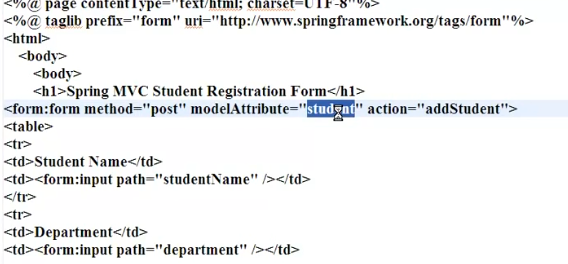
****

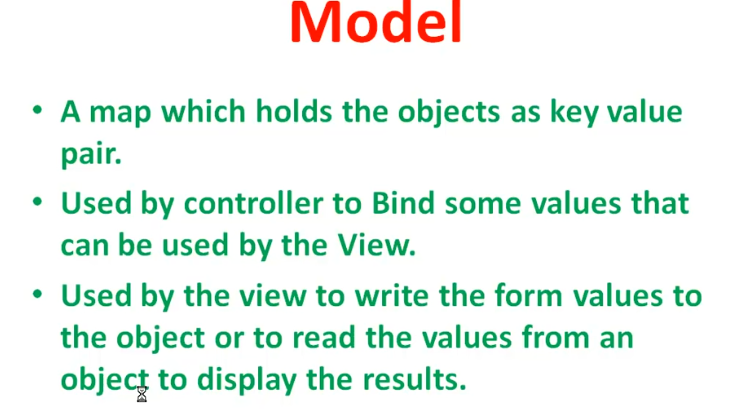
****

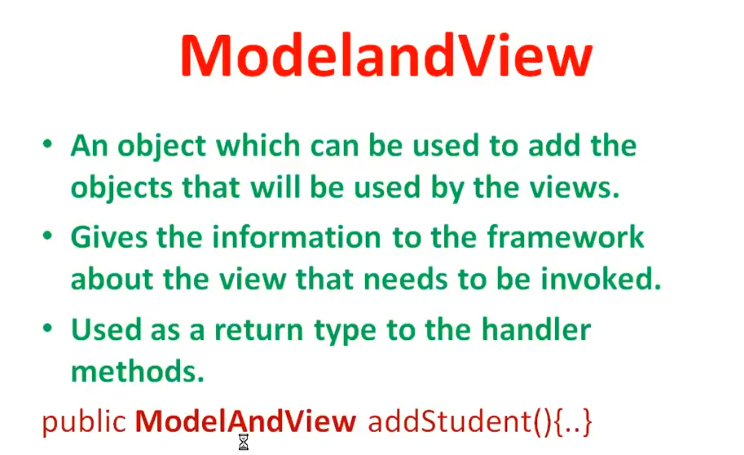
****

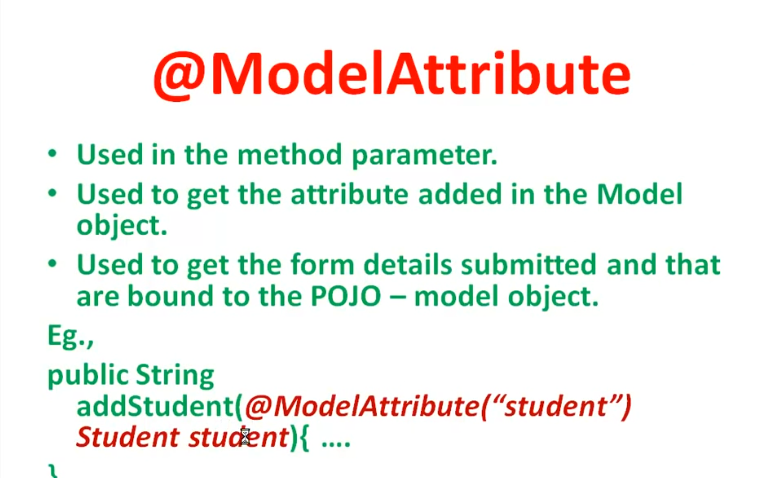
****

****

****

****

****

****

**Spring View Resolver**

****

****

**Spring MVC Configuration with Class.**

Add all the dependences in pom.xml and run the “mvn clean install” . All the dependences are added to web application.

**Web.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app version=*"2.4"* xmlns=*"http://java.sun.com/xml/ns/j2ee"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd"*>

<display-name>SpectreDashboard\_Sharan</display-name>

<context-param>

<param-name>contextClass</param-name>

<param-value>

<!-- org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer -->

org.springframework.web.context.support.AnnotationConfigWebApplicationContext

</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<servlet>

<servlet-name>SpringDispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextClass</param-name>

<param-value>

org.springframework.web.context.support.AnnotationConfigWebApplicationContext

<!-- org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer -->

</param-value>

</init-param>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>net.codejava.spring</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>default</servlet-name>

<url-pattern>/pages/\*</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>SpringDispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<session-config>

<session-timeout>30</session-timeout>

</session-config>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

</web-app>

Create the class as below

@Configuration

@ComponentScan(basePackages="net.codejava.spring")

/\*@EnableWebMvc\*/

@PropertySource("classpath:application.properties")

**public** **class** MvcConfiguration **extends** WebMvcConfigurerAdapter{

@Autowired

**private** Environment env;

@Bean

**public** ViewResolver getViewResolver(){

/\*InternalResourceViewResolver resolver = new InternalResourceViewResolver();

resolver.setPrefix("/WEB-INF/views/");

resolver.setSuffix(".jsp");

return resolver;\*/

UrlBasedViewResolver viewResolver=**new** UrlBasedViewResolver();

viewResolver.setViewClass(TilesView.**class**);

**return** viewResolver;

}

@Override

**public** **void** addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("/resources/\*\*").addResourceLocations("/resources/");

}

@Bean

**public** DataSource getDataSource() {

DriverManagerDataSource dataSource = **new** DriverManagerDataSource();

dataSource.setDriverClassName("com.mysql.jdbc.Driver");

dataSource.setUrl("jdbc:mysql://" +env.getProperty("database.ip") + ":" + env.getProperty("database.port") +"/spectre2");

dataSource.setUsername(env.getProperty("database.username"));

dataSource.setPassword(env.getProperty("database.pass"));

**return** dataSource;

}

@Bean

**public** DataBaseConfig getDBConfig(){

DataBaseConfig db = **new** DataBaseConfig();

String username =env.getProperty("database.username");

String pass=env.getProperty("database.pass");

String ip = env.getProperty("database.ip");

String port =env.getProperty("database.port");

db.setIp(ip);

db.setPass(pass);

db.setUsername(username);

db.setPort(port);

**return** db;

}

@Bean

**public** UserDAO getUserDAO() {

**return** **new** UserDAOImp(getDataSource(),getDBConfig() );

}

}