**Spring MVC**

**Request Processing Flow**

1. Request comes to DipatcherServlet
2. DispatcherServlet delegates to one of the mvc controller’s mapping (method)
3. Request is processed in that method, the method returns model and view hint to DispatcherServlet
4. DispatchServlet uses ViewResolver like InternalViewResolver to map it to the page which should be rendered as response , also the model object is passed to the page.

**DispatcherServlet**

1. It is the inbuilt class in Spring MVC
2. it is a servlet
3. its job is to delegate the request to a Spring MVC controller’s mapping ie. the method of class annotated with @Controller and have mapping like @GetMapping etc , DispatcherServlet for that reason is known as a front controller or in other words it is the job of front controller to decide which MVC controller(@Controller) will serve the request.

**Two ways of returning model and view, both does same task**

1)

@GetMapping(**"/findemployee"**)  
**public** ModelAndView details(@RequestParam(**"id"**) **int** id) {  
 Employee e = employeeService.findEmployee(id);  
 ModelAndView mv = **new** ModelAndView(**"employeedetails"**, **"employee"**, e);  
 **return** mv;  
}

2)

@GetMapping(**"/findemployee"**)  
**public** String details(@RequestParam(**"id"**) **int** id,Model model) {  
 Employee e = **service**.findEmployeeById(id);  
 model.addAttribute(**"employee"**,e);// adding model object with name with which it will be  
 **return "employeedetails"**;//returning view name  
}

**Different request mapping options in spring mvc**

@PostMapping(value = **"/employee/add"**)

**Or**

@RequestMapping(method = RequestMethod.POST,value = **"/employees/add"**)

Similarly for @GetMapping, @DeleteMapping etc

**Inform spring to scan packages for components**

1. <context:component-scan base-package="com.veri" /> specified in xml(old way)

2) @ComponentScan(“"com.veri") in java configuration

Or @SpringBootApplication(scanBasePackages = "com.veri")

In Spring Beans Factory, beans are registered with id or name

But fetched either by bean’s name or bean’s type(class) from bean factory or ApplicationContext

<context:annotation-config> gives this information to spring that annotation support is enabled ie support for @Component , @Bean, @Autowired

<mvc:annotation-driven /> when mentioned in xml provides this information to spring that annotation support is enable for Spring mvc ie. support for @Controller, @RestController, @GetMapping,@RequestMapping

**Session scope**

Get bean session scoped , bean object/instance will be kept in http session and will be accessible accross multiple requests from same user

Use @Scope(“session”)

Or

@Scope(WebApplicationContext.***SCOPE\_SESSION***)

@Component

@Scope(“session”)

public class Employee{

}

**Request Scope**

@Scope(WebApplicationContext.***SCOPE\_REQUEST***)

**Or** @Scope(“request”)

@Component

@Scope(“request”)

public class Employee{

}

This scope will inform spring to keep bean object in request scope ie. the object/instance can be accessed any number of time till request finishes , for new request new instance will be created