Spring Boot

Java Spring Boot is a tool that makes developing web application and microservices with Spring Framework faster and easier through three core capabilities:

1. AutoConfiguration
2. An opinionated approach to configuration
3. The ability to create standalone applications.
   These features work together to provide you with a tool that allows you to set up a Spring based application with minimal configuration and setup.

Why Spring boot?

You can choose Spring Boot because of the features and benefits it offers as given here-

- It provides a flexible way to configure Java Beans, XML configurations and Database Transactions.
- It provides a powerful batch processing and manages REST endpoints
- In Spring boot, everything is auto configured, no manual configurations are needed.
- It offers annotation-based spring application.
- Eases dependency management.
- It includes Embedded Servlet Container.

What are Starter Files>

Before spring Boot was introduced Spring Developers used to spend a lot of time on Dependency management. Spring Boot Starters were introduced to solve this problem so that the developers can spend more time on actual code than Dependencies. Spring Boot Starters are dependency descriptors that can be added under the **<dependencies>** section in pom.xml. There are around 50+ Spring Boot Starter for different Spring and related technologies. These starters give all the dependencies under a single name.

For example, if you want to use Spring Data JPA for database access, you include **spring-boot-starter-data-jpa** dependencies.

The advantages of using Starters are as follows:

- Increase productivity by decreasing the Configuration time for developers
- Managing the POM is easier since the number of dependencies to be added is decreased.
- Tested, Production-ready, and supported dependency configurations.
- No need to remember the name and version of the dependencies.

DIFFERENCE BETWEEN SPRING MVC AND SPRING BOOT

SPRING MVC	SPRING BOOT
1. Spring MVC is Model View, and Controller based web framework widely used to develop web applications.	1. Spring Boot is built on top of the conventional spring framework, widely used to develop REST API's.
2. If we are using Spring MVC, we need to build the configuration manually.	2. If we are using Spring Boot, there is no need to build the configuration manually,
3. Here a Deployment descriptor is required	3. Here there is no need for a deployment descriptor.
4. Spring MVC specifies each dependency separately.	4. It wraps the dependencies together in a single unit.
5. Spring MVC framework consists of four components Model, View, Controller, and Front controller.	5. There are 4 main layers in Spring Boot: Presentation Layer, Data Access Layer, Service Layer and Integration layer.
6. It takes more time in development.	6. Reduces development time and increases productivity.
7. Spring MVC do not provide powerful batch processing.	7. Powerful batch processing is provide by Spring Boot.

Important annotations in spring boot

ANNOTATION	DESCRIPTION
@Component	Application components and their variants are automatically reistered as Spring Beans, providing dependency injection, provided you use either @SpringBootApplication or @ComponentScan. @Repository, @Controller, and @Service are move specific alternatives to @component.
@EntityScan	This annotation is used to confure all the entity classes to spring
@Controller	@Controller is specialized @Component marked as a controller in MVC architecture
@EnableJPARepositories	This annotation is used to enable jpa repository and configure the sub-interface of jpa repository.
@RestController	@RestController combines the @Controller and @ResponseBody int oa single annotation. @RestController class return domains instead of views.

ANNOTATIONS(Method level)	DESCRIPTION
@PostMapping	It maps the **HTTP POST** requests on the specific handler method. It is used to create a web service endpoint that **persist data into database.**
@GetMapping	It maps the **HTTP GET** requests on the specific handler method. It is used to create a web service endpoint that **fetches data from database.**
@PutMapping	It maps the **HTTP PUT** requests on the specific handle method. It is used to create a web service endpoint that **creates** or **updates data.**
@DeleteMapping	It maps the **HTTP DELETE** requests on the specific handle method. It is used to create a web service endpoint that **deletes** a resource

ANNOTATIONS(Response Level)	Description
@RequestBody	It is used to **bind** HTTP request with an object in a method parameter. Internally it uses **HTTPMessageConverters** to convert the body of the request. When we annotate a method parameter with **@RequestBody,** the Spring framework binds the incoming HTTP request body to that parameter.(From JSON Body)
@RequestParam	It is used to extract the query parameters form the URL. It is also know an a **query parameter.** It is most suitable for web applications. It can specify default values if the query parameter is not present in the URL.