**Spring Boot**

What and Why Spring Boot

How Spring boot automatically done the configuration

Starter dependencies

Explore the main class annotation

Explore the component scan annotation

What is application.properties file

Explore the ResponseEntity

Explore the Service annotation

Explore the different types of request with postman

Explore the Logger and logging levels

Profiles in Spring boot application and some terminal commands

**Why Spring Boot**

1. Rapid Development: Spring Boot provides a streamlined and efficient way to develop Java-based applications. It comes with sensible defaults and auto-configuration, reducing the amount of boilerplate code needed to set up a project
2. Simplified Configuration: Spring Boot reduces the complexity of configuring your application by providing a set of default configurations. It also offers externalized configuration options, allowing you to easily customize settings without modifying code.
3. Embedded Server: Spring Boot includes an embedded servlet container (Tomcat, Jetty, or Undertow), eliminating the need to deploy your application to a separate server. This simplifies deployment and makes it easier to run your application locally for testing
4. Microservices Support: Spring Boot is well-suited for building microservices architectures. It provides features like RESTful web services support, easy integration with other Spring projects (e.g., Spring Cloud), and built-in support for service discovery, making it an ideal choice for developing microservices-based applications.
5. Production-Ready Features: Spring Boot includes several production-ready features out of the box, such as metrics, health checks, and monitoring. This helps you build applications that are reliable, scalable, and easy to manage in a production environment.
6. Externalized Configuration: Spring Boot allows externalizing configuration, enabling developers to configure their applications using properties files, YAML files, environment variables, or command-line arguments. This makes it easier to manage configurations across different environments without modifying the codebase

**Spring Boot Starters**

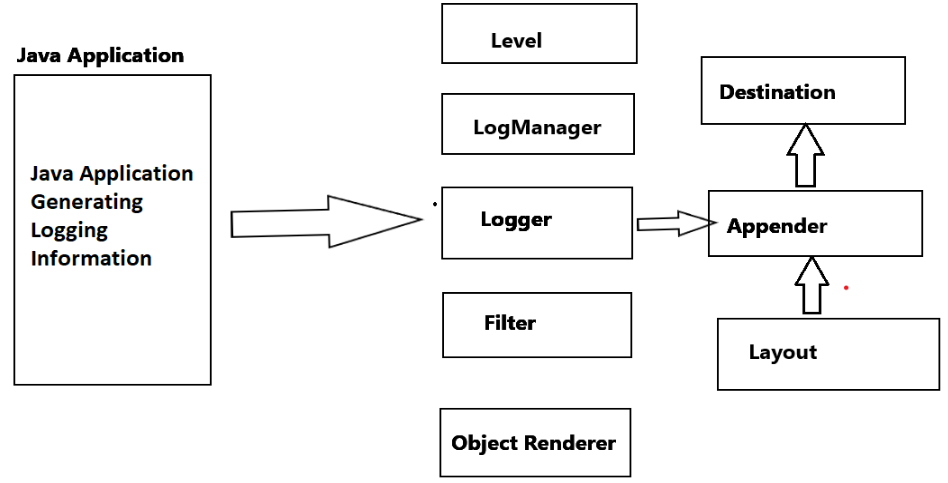
Spring Boot starters are a set of pre-configured dependencies that are commonly used together to accelerate the development of specific types of applications. They help streamline the dependency management process and provide a cohesive set of libraries for common tasks

Spring-boot-starter-web, spring-boot-starter-data-jpa, spring-boot-starter-test

**LOGGING**

**Benefits**

1. Logging Levels
2. Performance
3. Customization
4. Custom Layout



**LOGGING Levels**

ALL > TRACE > DEBUGG > INFO > WARN > ERROR > FATAL > OFF