

# **DOCUMENTATION ON SPRINGBOOT** **AND MICROSEVICES**

Spring Boot is a popular Java-based web application framework. It provides a number of features to assist developers in quickly and efficiently creating robust and scalable applications. Spring Boot has the following key features:

- 1) Spring Boot includes auto-configuration, which allows developers to configure the application based on the dependencies used in the project. This means that developers will not have to write a lot of boilerplate code to get their application up and running.
- 2) Spring Boot includes an embedded web server, allowing developers to easily create web applications without the need to configure an external server. Spring Boot works with a variety of web servers, including Tomcat, Jetty, and Undertow.
- 3) Spring Boot as an Actuator, Actuator is a set of tools that provides operational features for developers to monitor and manage their applications. Metrics, health checks, and endpoint mappings are among the features available.
- 4) Spring Data JPA: Spring Boot includes Spring Data JPA support, which is a high-level abstraction layer for working with databases. This enables developers to write database queries in Java code rather than SQL.
- 5) Spring Security: Spring Boot includes Spring Security support, which provides a robust and customizable web application security framework. Spring Security includes authentication, authorization, and session management capabilities.
- 6) Spring Boot Development Tools DevTools are a collection of tools that help developers streamline their development process. Hot reloading is one of these tools, which allows developers to see changes to their code without having to restart the application.
- 7) Spring Integration: Spring Boot includes Spring Integration support, which is a set of tools for developing enterprise integration solutions.

Message channels, message endpoints, and message routers are examples of these tools.

- 8) Spring Boot Test: Spring Boot Test is a set of tools and annotations that can be used to test Spring Boot applications. It simplifies the creation of unit tests, integration tests, and end-to-end tests for your application.
- 9) Spring Boot enables developers to externalise application configuration, which means that configuration settings can be saved in external files or the environment. This simplifies the process of configuring the application for various environments such as development, testing, and production.

10)

Simple deployment: Spring Boot applications are simple to install and run. They can be packaged as executable JAR files that can be run on any Java-enabled platform. This facilitates deployment of the application to cloud platforms such as AWS, Azure, or Google Cloud.

## **MICROSERVICES :**

Microservices is a software architecture style that emphasises small, independent services that work together to provide a larger application. Each service in a microservices architecture is designed to perform a single, specific function within the application.

Each microservice is self-contained, with its own codebase, database, and infrastructure, and communicates with other microservices via APIs or other lightweight mechanisms. Because each microservice can be developed, tested, and deployed independently of the others, developers can create highly scalable and flexible systems.

There are several advantages to using a microservices architecture, including:

**Scalability:** Because each microservice can be scaled independently, handling high traffic loads and providing better performance is simplified.

**Flexibility:** The microservices architecture enables developers to change and update services without disrupting the application as a whole.

Resilience: If one microservice fails, the other microservices can continue to function normally.

Diverse technology: Developers can use different technologies for different microservices, allowing them to select the best technology for each function.

Continuous delivery: Because each microservice can be developed and deployed independently, the microservices architecture enables continuous delivery and deployment.

Overall, microservices architecture is an effective method for developing complex and scalable software systems. It has significant benefits, but it also necessitates careful planning and design to be successful.