1. What is Spring Boot?

Answer: Spring Boot is a Java-based framework for creating digital services that are testable,

easily maintainable and manageable by a small team, fill business needs and can be independently

deployed without a larger application.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2. What is Spring?

Answer: Spring is a powerful, lightweight application development framework for

enterprise Java development.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3. What is Spring vs Spring Boot?

Answer: Spring is a framework for web applications. The framework has tools and libraries

that you can use to create custom applications. Spring Boot is a module of Spring that you

can use to create a Spring application or project that can only run.

==> Spring:

\* Definition:

Spring is a powerful, lightweight application development framework for enterprise Java development.

\* Modules:

It is modular and consists of various modules like Spring Core, Spring MVC, Spring Security, etc.

\* Configuration:

Requires manual configuration and setup for different components.

\* Boilerplate Code:

Developers may need to write more boilerplate code for common tasks.

\* Project Type:

Suitable for large-scale enterprise applications with complex configurations.

==> Spring Boot:

\* Definition:

Spring Boot is a project within the Spring ecosystem that simplifies the development of Spring applications.

\* Objective:

Focuses on convention over configuration and providing defaults to get applications up and running quickly.

\* Configuration:

Emphasizes autoconfiguration, reducing the need for manual setup.

\* Boilerplate Code:

Significantly reduces boilerplate code through defaults and starter dependencies.

\* Project Type:

Ideal for microservices, web applications, and projects with rapid development needs.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. Discuss the advantages of using Spring Boot.

Answer: There are several advantages to using Spring Boot, including ease of use. It also helps reduce

the time and resources required for creating microservices, which increases your overall profitability.

In my experience with Spring Boot, I used it to convert ChemCal Corp to microservices architecture.

I handled all aspects of development, and using Spring Boot's autoconfigured starters allowed me

to do this quickly and resource efficiently.

1. Rapid prototyping: With Spring Boot's automatic configuration, you can focus more on

writing your business logic and less on configuration, making it faster to get a working prototype

up and running.

2. Stand-alone applications: Spring Boot applications are stand-alone, meaning they can be

started and run on their own without needing to be deployed on an external server, simplifying

the deployment process.

3. Embedded server: Spring Boot comes with embedded servers (like Tomcat or Jetty), so you

don't have to worry about setting up a server yourself.

4. Simplified dependency management: Spring Boot manages dependencies for you,

reducing the chances of version conflicts.

5. Actuator module: This provides production-ready features like health checks and metrics

gathering right out of the box.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5. What is autoconfiguration?

Answer: Autoconfiguration in Spring Boot automatically sets up your application by analyzing

dependencies, saving you from writing extensive configuration code. It provides sensible defaults

based on the project's classpath, making it quick and easy to start a new project.

or

Answer: Autoconfiguration is an important feature of Spring Boot that was created out of a need

to reduce the complexity of configuration in the Spring framework. Autoconfiguration does this

by offering different starters, and depending on which starter you use, it configures the application

differently. This saves developers time and makes Spring Boot easier to use than Spring.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6. How do you disable autoconfiguration in Spring Boot?

Answer: The exclude feature of the @EnableAutoConfiguration annotation allows you to

disable autoconfiguration. Do this by using it within an applicable string of code.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7. What is the process of registering a custom autoconfiguration with Spring Boot?

Answer: Custom autoconfiguration is a class in Spring Boot that can be registered. To do this,

the full name of the autoconfiguration should be entered into the META-INF/spring.factories

file in the EnableAutoConfiguration key. However, if it is in Maven, it should be registered in

the Maven directory file.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8. Can you list the available Spring Boot starters?

Answer: Some common starters are spring-boot-starter, spring-boot-starter-aop and

spring-boot-starter-data-jpa. Spring-boot-starter is a core starter that has autoconfiguration

support and more. Spring-boot-starter-aop is used for aspect-oriented programming, and

spring-boot-starter-data-jpa is compatible with Hibernate. These are a few top starters,

although there are more depending on your use case.

spring-boot-starter

spring-boot-starter-web

spring-boot-starter-data-jpa

spring-boot-starter-security

spring-boot-starter-test

spring-boot-starter-logging

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9. Explain your experience with Spring Boot.

Answer: I have four years of experience using Spring Boot to resolve bandwidth issues as

my company scaled to more users on its proprietary software platform.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10. What are the most important skills for a Spring Boot programmer to have?

Answer: I pride myself on a deep knowledge of architecture and am CompTIA Security+ certified.

My ability to create microservices architecture securely is an asset to the companies where I may work."

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

11. What is IOC container ?

Answer: Inversion of Control (IoC) is a principle where the control of object creation and lifecycle is

shifted from the application code to a container. The core of Spring provides the IoC container that

manages the objects and their dependencies.

It is the core of the spring framework, It manages & controls the lifycycle of java objects.

This is a principle where the control flow of a program is inverted: instead of the client controlling the

flow of control (what to do and when to do), this responsibility is given to an external entity.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

12. What is dependency injection ?

Answer: Dependency Injection (DI) is a design pattern in software development that focuses on

providing objects (dependencies) to a class rather than having the class create or manage

its own dependencies. The primary goal of dependency injection is to achieve loose coupling

between classes, making the system more modular, maintainable, and testable.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

13. What are some of the problems the Spring Framework solves?

Answer: Here are some of the problems that the Spring Framework helps to solve:

1. Boilerplate Code:

Spring reduces boilerplate code through features like dependency injection, declarative

configuration, and aspects, leading to more concise and maintainable code.

2. Tight Coupling:

Spring promotes loose coupling through features like dependency injection, allowing

components to be wired together without explicit dependencies. This enhances flexibility,

testability, and maintainability.

3. Aspect-Oriented Programming (AOP):

Spring's AOP support enables the modularization of cross-cutting concerns. Aspects can be defined

separately from the core business logic, reducing code duplication and improving maintainability.

4. Inversion of Control (IoC):

Spring's IoC container manages the lifecycle of objects and their dependencies. It promotes a more

flexible and modular design by allowing components to be wired together through configuration

rather than being instantiated within code.

5. Exception Handling:

Spring provides a comprehensive exception-handling mechanism, allowing developers to handle

exceptions using declarative annotations or configure global exception handling strategies.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

14. What is Spring Boot Actuator?

Answer: Spring Boot Actuator is a module that provides production-ready features to monitor

and manage your Spring Boot application. It offers various endpoints and metrics that can be

used for monitoring, health checks, auditing, and managing your application.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

15. What are Spring beans?

Answer: Spring Bean is nothing special, any object in the Spring framework that we

initialize through Spring container is called Spring Bean.

Spring beans are Java objects managed by the Spring framework. They are defined and configured

in the Spring IoC container, allowing for centralized control over their instantiation, configuration,

and lifecycle. Beans are the building blocks of a Spring application and are typically wired together

to form the application's architecture.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

16. Explain the differences between @RequestMapping and @RestController annotations.

Answer: @RequestMapping is used to map URLs to methods in Spring, handling various

HTTP methods. @RestController is a specialized version mainly for building RESTful APIs,

combining @Controller and @ResponseBody to simplify response handling. If you're building

an API, @RestController is often more convenient.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

17. What is the difference between @RequestMapping and @GetMapping?

Answer: @GetMapping is a specialized and more concise annotation specifically designed

for handling HTTP GET requests, whereas @RequestMapping is a more general-purpose

annotation that can handle requests for various HTTP methods.

@RequestMapping: Requires specifying the HTTP method explicitly using the method

attribute if you want to restrict it to a specific HTTP method.

@GetMapping: Focuses specifically on the GET method, and there is no need to explicitly

mention the HTTP method.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

18. What is dependency management?

Answer: Dependency management in Spring Boot is like having pre-packaged sets of

tools (starters) for common tasks. It makes adding and handling external pieces

(dependencies) in your project easy, saving you from dealing with version headaches.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

19. How do you set up a Spring Boot project?

Answer: There are several ways to set up a project in Spring Boot. One way is to use the Spring

Initialzr with Eclipse and Maven. First, launch the Initializr and choose the Group, Artifact and

dependencies. Then, select Generate Project. Finally, you can choose to import your existing

Maven project from Eclipse.