Spring Boot Interview Questions & Answers

Chapter 10: Spring Boot – Building RESTful Web Service

Q: What is a RESTful web service in Spring Boot?

A RESTful web service is an API that follows REST principles, where clients communicate with the server using HTTP methods like GET, POST, PUT, and DELETE. Spring Boot makes it very easy to create REST APIs by using annotations and auto-configuration.

Use-Case / Example: A student management system could expose APIs like GET /students to fetch all students and POST /students to add a new student.

Q: How do you create a simple REST endpoint in Spring Boot?

You create a class annotated with @RestController and define request-handling methods using annotations like @GetMapping, @PostMapping, etc.

Use-Case / Example:

```
@RestController
public class HelloController {
@GetMapping("/hello")
public String sayHello() {
return "Hello, Spring Boot!";
}
}
```

Q: What is the difference between @Controller and @RestController?

@Controller is used for MVC applications where views (like JSP or Thymeleaf) are returned, while @RestController is a shorthand that combines @Controller and @ResponseBody, directly returning data (like JSON or XML).

Use-Case / Example: For a REST API returning JSON data, you should use @RestController. For a traditional web app returning HTML pages, use @Controller.

Q: How do you handle HTTP methods in Spring Boot?

Spring Boot provides specific annotations:
- @GetMapping → for GET requests
- @PostMapping → for POST requests
- @PutMapping → for PUT requests
- @DeleteMapping → for DELETE requests

Use-Case / Example: In an online store API, @PostMapping("/orders") can be used to create a new order, while @GetMapping("/orders/{id}") retrieves an existing order.

Q: How do you pass path variables in REST APIs?

You can use @PathVariable in method parameters to capture values from the URL.

Use-Case / Example:
@GetMapping("/students/{id}")
public Student getStudent(@PathVariable int id) {
return studentService.findById(id);
}

Q: How do you pass query parameters in REST APIs?

You can use @RequestParam to capture query parameters.

Use-Case / Example:
@GetMapping("/search")
public List searchStudents(@RequestParam String name) {
return studentService.findByName(name);
}

Q: How do you handle request bodies in Spring Boot?

For POST and PUT requests, you can use @RequestBody to map the request body to a Java object.

Use-Case / Example:
@PostMapping("/students")
public Student addStudent(@RequestBody Student student) {
return studentService.save(student);
}

Q: How do you return JSON responses in Spring Boot?

Spring Boot automatically converts Java objects into JSON using the Jackson library, as long as you use @RestController or @ResponseBody.

Use-Case / Example: Returning a Student object from a controller will automatically return JSON like:
{ "id": 1, "name": "Alice" }

Q: How do you test a RESTful web service in Spring Boot?

You can use tools like Postman, curl, or Spring Boot's built-in MockMvc for testing endpoints.

Use-Case / Example: A developer can send a POST request with JSON data using Postman to check if the API correctly saves a student record.

Q: Why is Spring Boot widely used for REST API development?

Spring Boot provides simplicity, auto-configuration, and embedded servers, allowing developers to focus on writing APIs instead of configuring frameworks. It also integrates

easily with databases, security, and cloud tools.

Use-Case / Example: A fintech company building microservices for payments, transactions, and notifications can quickly create REST APIs using Spring Boot, reducing development time.