Spring Boot Annotation Interview Cheat Sheet

1■■ CORE & CONFIGURATION ANNOTATIONS

@**SpringBootApplication**: Main entry point of a Spring Boot app. Combines @Configuration, @EnableAutoConfiguration, and @ComponentScan.

Example:
@SpringBootApplication
public class App { public static void main(String[] args){ SpringApplication.run(App.class, args); } }

@**Component /** @**Service /** @**Repository /** @**Controller**: Marks a class as a Spring Bean and registers it in the container.

Example:
@Service
class UserService { void register(){} }

@**Configuration &** @**Bean**: Used to define Spring Beans manually in a configuration class.

Example:
@Configuration
class AppConfig { @Bean MyService myService(){ return new MyService(); } }

@**Autowired**: Injects dependencies automatically via constructor, field, or setter.

Example:
@Autowired private UserService userService;

@**Qualifier**: Used when multiple beans of the same type exist to specify which one to inject.

Example:
@Autowired @Qualifier("emailService") private MessageService service;

@**Value**: Injects values from application.properties or environment variables.

Example:
@Value("${app.name}") private String appName;

@**Primary**: Marks a bean as the default choice when multiple candidates exist.

Example:
@Primary @Component class EmailService {}

@**Scope**: Defines the scope of a bean (singleton, prototype, etc.).

Example:
@Scope("prototype") @Component class Connection {}

@**Lazy**: Initializes a bean only when it’s needed.

Example:
@Lazy @Component class HeavyService {}

2■■ WEB & REST ANNOTATIONS

@**RestController**: Marks a controller that returns data (JSON/XML) directly instead of a view.

Example:
@RestController class UserController { @GetMapping("/users") List list(){ return ...; } }

@**RequestMapping**: Maps web requests to specific handler methods or classes.

Example:
@RequestMapping("/api") class ProductController {}

@**GetMapping / @PostMapping / @PutMapping / @DeleteMapping**: Shortcut annotations for specific HTTP request methods.

Example:
@GetMapping("/items") List getAll(){}

@**RequestBody / @ResponseBody**: Bind JSON request body to Java object, or return object as JSON.

Example:
@PostMapping("/add") String add(@RequestBody User u){}

@**PathVariable / @RequestParam**: Extracts data from URL path or query string.

Example:
@GetMapping("/user/{id}") String get(@PathVariable int id){}

@**ExceptionHandler / @ControllerAdvice**: Handle exceptions globally or locally within a controller.

Example:
@ControllerAdvice class GlobalHandler { @ExceptionHandler(Exception.class) ... }

3■■ DATA & DATABASE ANNOTATIONS

@**Entity / @Table**: Marks a class as a JPA entity and maps it to a database table.

Example:
@Entity @Table(name="users") class User { @Id Long id; }

@**Id / @GeneratedValue**: Defines primary key and its generation strategy.

Example:
@Id @GeneratedValue(strategy=GenerationType.IDENTITY) Long id;

@**Column**: Specifies column details for a field.

Example:
@Column(name="username") private String name;

@**Transactional**: Marks a method or class as transactional (commits or rolls back automatically).

Example:
@Transactional void processOrder(){}

4■■ TESTING & UTILITY ANNOTATIONS

@**SpringBootTest**: Used for integration testing, loads the full Spring context.

Example:
@SpringBootTest class MyAppTests {}

@**Test**: Marks a method as a unit test (JUnit).

Example:
@Test void shouldReturnTrue(){}

@**MockBean**: Creates and injects mock beans in Spring tests.

Example:
@MockBean private UserService userService;

@**EnableScheduling /** @**Scheduled**: Used for scheduling periodic tasks.

Example:
@Scheduled(fixedRate=5000) void runTask(){}