SPRING BOOT IMPORTANT ANNOTATIONS

Ashan Indrajith

@SpringBootApplication

This annotation is used to mark the main class of a Spring Boot application. It combines three other annotations: @Configuration, @EnableAutoConfiguration, and @ComponentScan.

```java
@SpringBootApplication
public class MyApplication {
    public static void main(String[] args) {
        SpringApplication.run(MyApplication.class, args);
    }
}
```

Ashan Indrajith

@RestController

`@RestController` marks a class as a RESTful controller, simplifying API development by handling HTTP requests and serializing responses automatically.

```
@RestController
public class MyController {
    @GetMapping("/hello")
    public String hello() {
        return "Hello, World!";
    }
}
```

Ashan Indrajith

@Autowired

This annotation is used for automatic dependency injection. It injects a Spring-managed bean into another bean.

```
@Service
public class MyService {
    public String getMessage() {
        return "Hello from MyService!";
    }
}

@RestController
public class MyController {
    @Autowired
    private MyService myService;

    @GetMapping("/message")
    public String getMessage() {
        return myService.getMessage();
    }
}
```

Ashan Indrajith

@Service

This annotation is used to mark a class as a service in Spring, indicating that it holds the business logic. Typically, services handle the application's business logic, acting as a middle layer between controllers and repositories .

```
@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;

    public User getUserById(Long userId) {
        return userRepository.findById(userId);
    }

    public void saveUser(User user) {
        userRepository.save(user);
    }
}
```

Ashan Indrajith

@Repository

This annotation is used to indicate that a class serves as a repository, responsible for data access operations. Repositories interact with the database or any other data store.

```
@Repository
public interface UserRepository extends JpaRepository<User, Long> {
    User findByUsername(String username);
}
```

Ashan Indrajith

@Controller

This annotation is used to mark a class as a controller in Spring MVC, responsible for handling incoming HTTP requests, processing them, and returning an appropriate response. Controllers interpret user input, interact with services to process data, and determine the appropriate view to render.

```java
@Controller
public class UserController {
    @Autowired
    private UserService userService;

    @PostMapping("/users")
    public ResponseEntity<Void> createUser(@RequestBody User user) {
        userService.saveUser(user);
        return ResponseEntity.status(HttpStatus.CREATED).build();
    }
}
```

Ashan Indrajith

@RequestMapping

@RequestMapping is a Spring annotation that connects HTTP requests to specific methods in your controller. It allows you to define various attributes such as the request path, HTTP method (GET, POST, etc.), request parameters, headers, and more.

```java
@Controller
public class HelloController {

    @RequestMapping("/hello")
    @ResponseBody
    public String hello() {
        return "Hello, World!";
    }
}
```

Ashan Indrajith

@RequestBody

The @RequestBody annotation in Spring MVC is used to indicate that a method parameter should be bound to the body of the HTTP request. It's commonly used to extract JSON or XML data from an incoming HTTP request and convert it into a Java object.

```java
@RestController
public class ExampleController {

    @PostMapping("/example")
    public String handleRequest(@RequestBody String requestBody
   {
        // Process the request body
        return "Received request body: " + requestBody;
    }
}
```

Ashan Indrajith

@PathVariable

The @PathVariable annotation in Spring MVC is used to extract values from URI templates in the request URL and map them to method parameters in a controller handler method.

```java
@RestController
public class ExampleController {

    @GetMapping("/hello/{name}")
    public String hello(@PathVariable String name) {
        return "Hello, " + name + "!";
    }
}
```

Ashan Indrajith