What is `@ControllerAdvice`?

`@ControllerAdvice` is an annotation used to define a **global handler** that applies to **all controllers** in your Spring Boot application.

It allows you to:

- Handle exceptions from anywhere in your app (global exception handling)
- Bind data globally for all controllers
- Apply model attributes or init binder methods across multiple controllers

---

🧠 Why do we need it?

Without `@ControllerAdvice`, every time an exception occurs in a controller, you would need to handle it manually in each method like this:

```
@GetMapping("/user/{id}")
public ResponseEntity<?> getUser(@PathVariable Long id) {
    try {
        User user = userService.findById(id);
        return ResponseEntity.ok(user);
    } catch (UserNotFoundException ex) {
        return
ResponseEntity.status(HttpStatus.NOT_FOUND).body("User not found");
    }
}
```

This makes your code repetitive and messy.

Instead, you can move all this logic to one centralized place using `@ControllerAdvice`.

⚙️ How it Works

When an exception is thrown from any controller method, Spring looks for a matching `@ExceptionHandler` method inside any class annotated with `@ControllerAdvice`.

🧱 Example — Global Exception Handler

```
package com.example.digitalWalletDemo.exception;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;

import java.time.LocalDateTime;
import java.util.HashMap;
import java.util.Map;

@ControllerAdvice
public class GlobalExceptionHandler {

    @ExceptionHandler(Exception.class)
    public ResponseEntity<Map<String, Object>>
handleAllExceptions(Exception ex) {
        Map<String, Object> errorDetails = new HashMap<>();
        errorDetails.put("timestamp", LocalDateTime.now());
        errorDetails.put("status",
HttpStatus.INTERNAL_SERVER_ERROR.value());
        errorDetails.put("error", "Internal Server Error");
        errorDetails.put("message", ex.getMessage());

        return new ResponseEntity<>(errorDetails,
HttpStatus.INTERNAL_SERVER_ERROR);
    }

    @ExceptionHandler(UserNotFoundException.class)
    public ResponseEntity<Map<String, Object>>
handleUserNotFound(UserNotFoundException ex) {
        Map<String, Object> errorDetails = new HashMap<>();
        errorDetails.put("timestamp", LocalDateTime.now());
        errorDetails.put("status", HttpStatus.NOT_FOUND.value());
        errorDetails.put("error", "User Not Found");
        errorDetails.put("message", ex.getMessage());
```

```
        return new ResponseEntity<>(errorDetails,
HttpStatus.NOT_FOUND);
    }
}
```

🧩 Example Exception Class

```
package com.example.digitalWalletDemo.exception;

public class UserNotFoundException extends RuntimeException {
    public UserNotFoundException(String message) {
        super(message);
    }
}
```

🧩 Example Controller

```
@GetMapping("/users/{id}")
public ResponseEntity<User> getUser(@PathVariable Long id) {
    User user = userService.findById(id)
        .orElseThrow(() -> new UserNotFoundException("User with ID "
+ id + " not found"));
    return ResponseEntity.ok(user);
}
```

Now — if the user is not found, the `UserNotFoundException` is thrown,
and the `@ControllerAdvice` automatically catches it and sends a structured JSON response.

✅ Advantages

Feature	Description
Centralized error handling	One place to handle all exceptions
Clean controller code	Controllers only contain business logic
Custom responses	You can send consistent JSON error responses

Extensible	You can add multiple handlers for different exceptions

---

💡 Bonus: Works with `@RestControllerAdvice`

If your app is purely REST-based (i.e., only returns JSON), you can use:

`@RestControllerAdvice`

instead of

`@ControllerAdvice`

The difference is that `@RestControllerAdvice` automatically adds `@ResponseBody` to all methods — so you don't need to manually wrap responses.

✅ Summary

Concept	Description
Annotation	`@ControllerAdvice`
Purpose	Global exception handling and configuration
Used with	`@ExceptionHandler`, `@InitBinder`, `@ModelAttribute`
Alternative	`@RestControllerAdvice` for REST APIs