**What needs to be done if there is another controller EmployeeController and similar validation needs to be done for Employee payload data?**

If you follow the **current approach** (manual validation inside the controller),  
then in EmployeeController, you would also have to:

* Create ValidatorFactory
* Create Validator
* Call validator.validate(employee)
* Loop over violations
* Throw ResponseStatusException if errors.

**SME to explain the disadvantage of the above solution:**

Problems with the current approach:  
Validation code is repeated in every controller — violates **DRY** (Don’t Repeat Yourself).  
Controller logic is cluttered with validation code, making it harder to read & maintain.  
 Any change in validation handling (like customizing error message format) would require editing all controllers.  
Not leveraging Spring’s built-in support for bean validation + centralized error handling.

**How will this be solved in the next hands-on?**

In the next hands-on, you will:  
Use @Valid annotation directly on the method parameter:

public Country addCountry(@Valid @RequestBody Country country) { … }

And let Spring automatically validate the bean based on annotations in the model (@NotNull, @Size, etc.).  
Write a **global exception handler** using @ControllerAdvice to catch validation errors and send appropriate response to the client.  
 No need to manually call Validator in every controller.