1. **Server Side Validations**

**- Presentation Logic (P.L)**

- To be performed in Rest Controller

- **B.L**

- In service layer

**Dev steps** for adding server side P.L validations

1. Add spring boot validation starter dependency

- pom.xml (done already in spring boot backend template)

2. To validate request body

- Add **@Valid** annotation along with **@RequestBody** (de-serial)

- Add validation rules (annotations - jakarta, spring , hibernate-validator) on DTO fields.

eg - **@NotNull , @NotEmpty , @NotBlank , @Email , @Pattern, @Length,@Min , @Max**

**@Future ,@Past, @DateTimeFormat** ....

More details -

@NotEmpty

=>Strings cannot be null and must have at least one character including whitespace

for collections must contain at least 1 element

@NotBlank

strings not null n must contain some content after whitespace trimming

@NotNull => simply not null (use it typically on non string fields - eg LocalDate , enum , double ...)

- In case of P.L failure , Spring throws

- **MethodArgumentNotValidException**

- add new exception handler method(@ExceptionHandler) in global exception handler class(@RestControllerAdvice)

3. To validate pathvariable | request parameter

- Add @Validated annotation

- on rest controller class

- use same validation annotations with @RequestParam | @PathVariable

- In case of P.L failure , Spring throws

- ConstraintViolationException

3.1 Add it for User sign up end point n test.

2. **Spring Boot Testing**

- unit testing

- integration testing

Solve

Add unit test – for Data Layer

* getRestaurantAndMenu(Long id);
* findTop3ByUserRoleOrderBySubscriptionAmountDesc(UserRole role);
* Write in DAO layer
* Get list of user’s last name from specified city
* Input - city
* Write its test case

3. **Spring AOP**