

# WEB TECHNOLOGIES USING **JAVA**

## **COURSE 7 – REST WEBSERVICES.**

# AGENDA

---

- EXCEPTION HANDLING
- ERROR RESPONSES
- INPUT VALIDATION

# EXCEPTION HANDLING

---

- cross-cutting concern
- application-wide strategy that handles all errors in the same way
- `@ControllerAdvice`

# ERROR RESPONSES

---

- provide correct HTTP status code on the response
- provide helpful, fine-grained details regarding the error in the response body; the following are provided by Spring Boot:
  - timestamp - the time in milliseconds when the error happened
  - status - HTTP status code associated with the error; this is partly redundant as it is same as the response status code
  - error - the description associated with the status code
  - exception - the fully qualified path to the exception class resulting in this error
  - message - the message providing more details about the error
  - path - the URI that resulted in the exception

**NOTE:** Please see Chapter 3 “Interaction Design with HTTP”, Section “Response Status Codes” from *REST API Design Rulebook* for best practices on the error response codes.

# INPUT VALIDATION

---

- **JSR 303** validation: an API intended to simplify field validation in any layer of the application
- Bean Validation API:
  - built under the JSR 303/349 validation specifications
  - provides a metadata model for JavaBean validation via a set of standardized validation constraints
  - domain object properties are annotated with validation constraints:
  - **Hibernate Validator**, a popular JSR 303/349 implementation framework
- 400 Bad Request should be returned as a response status in case the validations fail

# INPUT VALIDATION

---

Constraint	Description
NotNull	Annotated field must not have null value
Null	Annotated field must be null
Max	Annotated field value must be an integer value lower than or equal to the number specified in the annotation
Min	Annotated field value must be an integer value greater than or equal to the number specified in the annotation
Past	Annotated field must be a date in the past
Future	Annotated field must be a date in the future
Size	Annotated field must match the min and max boundaries specified in the annotation. For a field that is a Collection, the size of the Collection is matched against boundaries. For a String field, the length of the string is verified against boundaries
Pattern	Annotated field must match the regular expression specified in the annotation

# BIBLIOGRAPHY

---

- Spring in Action, by Craig Walls
- Spring REST, by Balaji Varanasi, Sudha Belida
- REST API Design Rulebook, by Mark Masse
- [Architectural Styles and Design of Network-based Software Architectures](#)
- [JSR 303: Bean Validation](#)

# Q&A

---





**THANK YOU**

**DANIELA SPILCĂ**