**Serialization**Java object->JSON(Post,Put,Patch)  
Serialization is the process of converting a Java object into a format that can be easily transmitted over a network,  
such as JSON or XML. This allows the object’s state to be saved or sent across the network.  
  
**Deserialization**  
JSON->Java object(Get)  
Deserialization is the reverse process, where the serialized JSON or XML data is converted back into a Java object,  
enabling you to work with the data in your Java application.  
  
 **Confirms API behaviour need to chk**  
It tells you if the API call was successful or failed (e.g., 200 OK, 404 Page Not Found, 500 Internal Server Error).  
  
Ensures correct test results  
If you're doing automated testing (e.g., using RestAssured, Postman, or any framework), status code checks ensure  
 your test passes or fails appropriately.  
  
Guards against false positives  
Sometimes an API may return a well-formed error message (with JSON), but if the status code is wrong  
(e.g., 500 or 400), it still means something broke — even if the response looks fine.  
  
**The HTTP 100**-series status codes are **informational responses**. They indicate that the request has been received and the  
 process is continuing — but no final response has been sent yet.  
  
 **3xx — Redirection (the “300 series”)**  
 Meaning: Client must take additional action to complete the request.  
  
 The server is telling the client: “Here’s something else you need to do or know.”  
  
  
 **HTTP Status Code Series Overview  
 1xx — Informational**  
 Request received, continuing process  
  
 Examples:  
  
 100 Continue — Client should continue with request  
  
 101 Switching Protocols — Server switching protocols as requested  
  
 **2xx — Success  
  
 The request was successfully received, understood, and accepted**  
 Examples:  
  
 **200 OK — Standard success response  
  
 201 Created — Resource successfully created  
  
 204 No Content — Success but no content returned**  
 **3xx — Redirection (the “300 series”)  
  
 Meaning: Client must take additional action to complete the request.  
  
 The server is telling the client: “Here’s something else you need to do or know.”  
  
 Common 3xx Status Codes  
 Code Name Meaning  
 300 Multiple Choices Multiple options for the resource, client decides  
 301 Moved Permanently Resource has moved permanently; use new URL  
 302 Found (Temporary Redirect) Resource temporarily at a different URL  
 303 See Other Redirect to a different resource (GET)  
 304 Not Modified Cached version is still valid (no need to resend)  
 307 Temporary Redirect Temporary redirect, method and body not changed  
 308 Permanent Redirect Permanent redirect, method and body not changed** When to use 300 Multiple Choices  
  
 When a resource has multiple possible representations (formats, languages, versions)  
  
 Client must pick one from the given options  
  
 Less common in practice—most APIs use content negotiation or direct redirects instead  
  
 Example 300 workflow recap:  
  
 Client requests a resource  
  
**HTTP 400 and 500 Series — Client vs Server Errors**

**400 Series — Client Errors**

**Meaning:** The client did something wrong — the request is invalid or can’t be processed.

| **Code** | **Name** | **Description** |
| --- | --- | --- |
| 400 | Bad Request | Request malformed or invalid syntax |
| 401 | Unauthorized | Authentication required or failed |
| 403 | Forbidden | Client is authenticated but not allowed access |
| 404 | Not Found | Resource doesn’t exist |
| 405 | Method Not Allowed | HTTP method not supported on the resource |
| 409 | Conflict | Request conflicts with current state of the resource |
| 422 | Unprocessable Entity | Request well-formed but semantic errors (validation) |

**Examples:**

* Sending invalid JSON in a POST → **400 Bad Request**
* Trying to access a protected resource without login → **401 Unauthorized**
* Accessing a forbidden page → **403 Forbidden**
* Requesting a non-existing URL → **404 Not Found**

**500 Series — Server Errors**

**Meaning:** The server failed to fulfill a valid request due to an internal problem.

| **Code** | **Name** | **Description** |
| --- | --- | --- |
| 500 | Internal Server Error | Generic server failure |
| 501 | Not Implemented | Server does not support the functionality |
| 502 | Bad Gateway | Server received invalid response from upstream |
| 503 | Service Unavailable | Server overloaded or down for maintenance |
| 504 | Gateway Timeout | Server upstream timed out |

**Examples:**

* Unhandled exception in backend code → **500 Internal Server Error**
* Server overloaded, can’t handle requests → **503 Service Unavailable**
* Proxy server gets invalid response → **502 Bad Gateway**