REST Assured PUT and DELETE Requests

🔹 PUT and DELETE Requests in Web Development (HTTP + Examples)

In HTTP, **`PUT`** and **`DELETE`** are two of the most common methods used in **RESTful APIs** to modify or remove resources on a server.

✅ PUT Request

🔸 Purpose:

- Used to **update** a resource or **create** a resource if it doesn't exist (depending on implementation).
- It is **idempotent**, meaning calling it multiple times with the same data results in the same state.

🔸 Syntax:

```http
PUT /users/123 HTTP/1.1
Host: example.com
Content-Type: application/json

{
  "name": "John Doe",
  "email": "john@example.com"
}
```

🔸 Example in Java (using RestAssured):

```java
given()
  .header("Content-Type", "application/json")
  .body("{ \"name\": \"John\", \"email\": \"john@example.com\" }")
.when()
  .put("https://example.com/users/123")
.then()
  .statusCode(200);
```

❌ DELETE Request

🔸 Purpose:

- Used to **delete** a resource.
- It is also **idempotent** – deleting the same resource multiple times results in the same state.

🔸 Syntax:

```http
DELETE /users/123 HTTP/1.1
Host: example.com
```

🔸 Example in Java (using RestAssured):

```java
given()
.when()
  .delete("https://example.com/users/123")
.then()
  .statusCode(200); // or 204 No Content
```

🔄 Difference Between PUT and POST:

Method	Use	Idempotent?	Resource Location
`PUT`	Create or Update existing resource	✅ Yes	Specified in URL (**`/users/123`**)
`POST`	Create new resource	❌ No	Usually auto-generated by server

---

Would you like a **mini-project or hands-on lab** for PUT/DELETE using Postman, Java, or Python?

To **validate** whether a resource was successfully **updated (PUT)** or **deleted (DELETE)**, you typically verify two things:

✅ 1. Validating PUT (Update)

🔸 HTTP Status Code:

- **`200 OK`** – Resource updated.
- **`201 Created`** – If PUT created a new resource.
- **`204 No Content`** – Update successful, no body returned.

🔸 Body Content Validation (Optional):

Check if the updated data is returned and matches what you sent.

🔸 Example (RestAssured - Java):

```java
java

// Step 1: Update the user
given()
  .header("Content-Type", "application/json")
  .body("{ \"name\": \"Koti\", \"email\": \"koti@abc.com\" }")
.when()
  .put("https://api.example.com/users/123")
.then()
  .statusCode(200) // or 204
  .body("name", equalTo("Koti"))
  .body("email", equalTo("koti@abc.com"));
```

🔸 Step 2 (Optional): GET the updated resource

```java
java

given()
.when()
  .get("https://api.example.com/users/123")
.then()
  .statusCode(200)
  .body("name", equalTo("Koti"));
```

❌ 2. Validating DELETE

🔸 HTTP Status Code:

- **`200 OK`** – Resource deleted, may return confirmation.
- **`204 No Content`** – Deleted successfully, no response body.
- **`404 Not Found`** – If you try to delete it again.

🔸 Example (RestAssured - Java):

```
java

// Step 1: Delete the user
given()
.when()
  .delete("https://api.example.com/users/123")
.then()
  .statusCode(204); // or 200
```

🔸 Step 2 (Optional): Confirm deletion via GET

```
java

given()
.when()
  .get("https://api.example.com/users/123")
.then()
  .statusCode(404); // Not Found means deletion successful
```

🧪 Summary of Validation

Operation	Status Code	Additional Validation
PUT	200 / 204	Re-GET and check values
DELETE	200 / 204	Re-GET returns 404

Would you like the same example using **Postman**, **Python**, or for a live dummy API like **`reqres.in`** or **`jsonplaceholder`**?