PUT Method – Detailed Notes (Beginner Friendly)

What is the PUT Method?

- • PUT is an HTTP method used in REST APIs.
- • It is mainly used to modify (update) existing data on the server.
- • In CRUD operations, it corresponds to the Update function.
- • It always requires a request body — the data you want to update must be sent with the request.
- • The data format should be specified using the Content-Type header (e.g., application/json).
- • PUT may or may not change the server state depending on the logic.

When is PUT Method Needed?

- • When you want to update an existing resource — not create something new.
- • The updated data should be accessed through a specific URL (endpoint).
- • It's often used in forms, e.g., updating profile info or resetting passwords.
- • PUT requests can be cached, though it's not common.
- • You can send data in the form of query parameters or as body parameters.

Examples of PUT Method in Real Life:

- • Updating cart items – e.g., increasing book quantity in an online cart.
- • Resetting passwords – when you fill a reset form and submit.
- • Updating reviews – editing your existing review for a book or product.
- • Form submissions – like updating your profile using a web form.
- • Book details update – changing book information on a page like Book Details.

Key Features of PUT Method:

- • Idempotent: Sending the same PUT request multiple times won't change the result after the first successful update.
- • It is not used to create a resource (that's what POST is for).
- • It replaces the entire resource unless partial update logic is used.
- • Needs complete data (usually) for the update — if you miss a field, it may get overwritten or lost depending on how the backend is set.

Summary: What You Learned Today

- • ✅ What HTTP PUT method is.
- • ✅ When the PUT method should be used.
- • ✅ Why it is necessary (for updates).
- • ✅ Features that make PUT unique and powerful for API design.

- ✅ **PUT Method - Practical Example (API Testing in Postman)**
- **CRUD Meaning:**
  Create → POST
  Read → GET
  Update → PUT
  Delete → DELETE

- ✅ **1. Free Test API for PUT Method Practice (No API Key Required)**
- Use this one:
- 🔗 **API**: https://jsonplaceholder.typicode.com/users/1
  🔧 **Method**: PUT
  📝 **Body** (raw → JSON):
- json
- CopyEdit
- {
- "name": "Shree",
- "username": "qa_shree",
- "email": "shreeqa@example.com"
- }
- 📌 **How to use in Postman**:
  - Set method to PUT
  - Paste the above URL
  - In the **Body tab**, choose raw → JSON
  - Paste the JSON body
  - Click **Send**
- 🟢 This will return a **200 OK response** with your updated data.
- 🔔 Note: This API doesn't actually update the server (it's fake), but it will show you how the response would look — perfect for practice.
- ---
- ✅ **2. QA Role in PUT Method**
- The PUT method in APIs is used to **update existing data** — and as a **QA engineer**, your job is to **verify and validate** that this works as expected.
- 🧪 **What a QA Does with PUT:**

QA Task	Description
✅ **Check request body**	Ensure required fields (like id, name, etc.) are included in the request.
✅ **Validation testing**	Test with **valid data**, **invalid data**, and **missing fields** to see if the API behaves correctly.

QA Task	Description
✅ **Status code testing**	Verify that correct status codes are returned: 200 OK for success, 400 Bad Request for missing fields, etc.
✅ **Data integrity**	Confirm that the updated data actually reflects in the response or database (in real APIs).
✅ **Negative testing**	What happens if you use a wrong user ID or send empty JSON? The API should return proper error messages like 404 or 400.
✅ **Security checks**	In some cases (like your earlier 401), the API should be protected and require a key/token. You test if the API is secure.

🔁 **In short:**

As a QA engineer, **your goal is to break things and report them** — especially when the API does not behave as expected on updates.

🔧 **WHAT WE DID:**

You have successfully tested a PUT **API request** in **Postman** to update a user's information using the dummy API:

✅ **API used:**

arduino

CopyEdit

https://jsonplaceholder.typicode.com/users/1

This is a **fake API endpoint** used for testing. It behaves like a real API but doesn't actually store the data permanently.

📩 **BODY SENT (What you sent to the server):**

You sent a **PUT** request (used to update existing data), with the following JSON data:

json

CopyEdit

```
{
  "name": "Shree",
  "username": "qa_shree",
  "email": "shreeqa@example.com"
}
```

This means you told the server:

"Please update user with ID = 1 to this new data."

- ✅ **SERVER RESPONSE (What you received back):**
- The response in the bottom section shows:
- json
- CopyEdit
- {
- "name": "Shree",
- "username": "qa_shree",
- "email": "shreeqa@example.com",
- "id": 1
- }
- This means the **server accepted your update** and responded with:
  - The updated user data (name, username, email)
  - id: 1 → Confirms that the user with ID 1 was targeted.
- Also, you got this:
  - **Status Code: 200 OK** → Means the request was successful.
  - Response Time: 892 ms → Less than 1 second.
- ---
- 📌 **SO, WHAT DID YOU LEARN HERE?**

🔍 Topic	✅ What Happened
PUT Method	You updated user data (replaced all fields).
Request Body	You sent JSON data to update user details.
Dummy API	You practiced safely without needing an account.
QA's Role	✓ To **verify** that the update API works correctly: ✓ Does it return the correct response? ✓ Is data correctly updated? ✓ Are error codes (like 400/401) handled?

- ---
- 💼 **As a QA, You Should Test:**

🔎 Test Case	Example
✅ Valid update	Update email → Check if it reflects in response
❌ Invalid data	Send missing field or invalid email → See if server gives error
🔐 Authorization	Check if API rejects request without login (401 error)

🔎 **Test Case**	**Example**
🧪 Boundary cases	Try empty strings or very long input