**GET() and POST()**

**@webServlet**

@WebServlet is an annotation used to define a **Servlet** and its **URL mapping** directly in the servlet class, without needing to configure it in the web.xml file.

**✅ Purpose:**

* To define a servlet and map it to a URL.
* Reduces the need for web.xml configuration.
* Introduced in **Servlet 3.0** (Java EE 6).

**1.service() Method**

* Defined in **HttpServlet** class.
* Called automatically by the **Servlet container** when a request comes in.
* Determines the HTTP request type (GET, POST, PUT, DELETE, etc.).
* Calls the corresponding doGet(), doPost(), doPut(), doDelete() method.

**✅ Example:**

@Override

protected void service(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.getWriter().println("Service method called");

super.service(req, res); // Calls doGet() or doPost()

}

**2. doGet() Method**

* Handles **HTTP GET** requests.
* Used to **retrieve data** (read-only operation).
* Called when a request type is GET.

**✅ Example:**

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.getWriter().println("GET request received");

}

**🔎 When to Use:**

✅ When fetching or reading data (e.g., search query).

✅ When showing a form or webpage.

**3. doPost() Method**

* Handles **HTTP POST** requests.
* Used to **send data** to the server (write operation).
* Called when a request type is POST.

**✅ Example:**

@Override

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.getWriter().println("POST request received");

}

**🔎 When to Use:**

✅ When submitting data (e.g., form submission).

✅ When creating or updating resources.

✅ For securely sending the data.

If you're working with **REST APIs** in Spring MVC or Servlets, you **should** use PUT and DELETE where appropriate, but if it's a web form, it's better to stick with POST and handle it accordingly.