**Handling Multiple Parameters in a Spring Boot REST API**

If you have **7 fields** that need to be passed to a method in a POST request and then inserted into a database, the best approach is to **use a DTO (Data Transfer Object)**.

**✅ Best Approach: Use a DTO (Recommended)**

Instead of passing 7 parameters in the URL, create a **DTO class** and send the data as a JSON object in the request **body**.

**1️⃣ Create a DTO (Data Transfer Object)**

java

CopyEdit

public class TaskRequestDTO {

private String title;

private String description;

private String priority;

private LocalDateTime dueDate;

private String assignedUser;

private boolean completed;

private String category;

// Getters and Setters (or use @Data from Lombok)

}

**2️⃣ Update the Controller to Accept the DTO**

java

CopyEdit

@RestController

@RequestMapping("/tasks")

public class TaskController {

@Autowired

private TaskService taskService;

@PostMapping("/create")

public ResponseEntity<String> createTask(@RequestBody TaskRequestDTO taskDTO) {

taskService.createTask(taskDTO);

return ResponseEntity.status(HttpStatus.CREATED).body("Task created successfully!");

}

}

* @RequestBody TaskRequestDTO taskDTO → Spring automatically converts the JSON request body into a **Java object**.
* The request must be sent as **JSON**.

**3️⃣ Implement the Service Layer**

java

CopyEdit

@Service

public class TaskService {

@Autowired

private TaskRepository taskRepository;

public void createTask(TaskRequestDTO taskDTO) {

Taskuri task = new Taskuri();

task.setTitle(taskDTO.getTitle());

task.setDescription(taskDTO.getDescription());

task.setPriority(taskDTO.getPriority());

task.setDueDate(taskDTO.getDueDate());

task.setAssignedUser(taskDTO.getAssignedUser());

task.setCompleted(taskDTO.isCompleted());

task.setCategory(taskDTO.getCategory());

taskRepository.save(task);

}

}

**4️⃣ Example JSON Request (Postman)**

**🔹 URL: POST http://localhost:8080/tasks/create**

**🔹 Headers: Content-Type: application/json**

**🔹 Body (JSON)**

json

CopyEdit

{

"title": "Finish Spring Boot API",

"description": "Complete the REST API development",

"priority": "HIGH",

"dueDate": "2025-03-05T10:00:00",

"assignedUser": "Leonid",

"completed": false,

"category": "Development"

}

**Alternative: Use Path Variables or Request Parameters (Not Recommended)**

If you **must** use URL parameters (not recommended for long data):

java

CopyEdit

@PostMapping("/create/{title}/{desc}/{priority}/{dueDate}/{user}/{completed}/{category}")

public ResponseEntity<String> createTask(

@PathVariable String title,

@PathVariable String desc,

@PathVariable String priority,

@PathVariable @DateTimeFormat(pattern = "yyyy-MM-dd'T'HH:mm:ss") LocalDateTime dueDate,

@PathVariable String user,

@PathVariable boolean completed,

@PathVariable String category) {

// Call service layer

return ResponseEntity.ok("Task Created!");

}

**Call this API as:**

http

CopyEdit

POST http://localhost:8080/tasks/create/TaskTitle/Description/HIGH/2025-03-05T10:00:00/Leonid/false/Development

🚨 **This method is harder to maintain & not flexible**.

**✅ Best Practice Summary**

| **Approach** | **Pros** | **Cons** |
| --- | --- | --- |
| **DTO + JSON (@RequestBody) ✅** | Clean, structured, scalable | Requires sending JSON |
| Path Variables (@PathVariable) | Simple for short params | URL gets too long, hard to maintain |
| Request Parameters (@RequestParam) | Works for simple filters | Not ideal for inserting objects |

For inserting **new rows in a database**, **always prefer DTOs with JSON requests**. 🚀