Below is the **exact mechanism Spring uses to decide which method to call**, expressed concisely and in the correct conceptual order. This is core Spring MVC knowledge.

High-Level Answer (One Line)

> Spring matches an incoming HTTP request to a controller method using **URL path + HTTP method + annotations**, via the **DispatcherServlet** and **HandlerMapping**.

Step-by-Step (What Actually Happens)

1. Request Enters Spring

```
HTTP Request → DispatcherServlet
```

- `DispatcherServlet` is the **front controller**
- Every HTTP request goes through it

---

2. Spring Scans Controller Mappings (At Startup)

When the application starts, Spring:

- Scans all classes annotated with `@RestController` / `@Controller`
- Scans all methods annotated with:
    - `@GetMapping`
    - `@PostMapping`
    - `@PutMapping`
    - `@DeleteMapping`
    - `@RequestMapping`

Spring builds an **internal lookup table** like:

```
(GET,  /api/employees)       → EmployeeController.getAll()
(POST, /api/employees)       → EmployeeController.create()
(GET,  /api/employees/{id})  → EmployeeController.getById()
(PUT,  /api/employees/{id})  → EmployeeController.update()
```

This happens **once at startup**, not per request.

---

3. Request Matching (Runtime)

Example request:

```
POST /api/employees
```

Spring checks:

1. **URL path** → `/api/employees`
2. **HTTP method** → `POST`
3. **Controller base path** → `@RequestMapping("/api/employees")`
4. **Method mapping** → `@PostMapping`

✅ Match found → call that method.

---

4. Method Invocation

Spring now:

- Converts JSON → Java object (`@RequestBody`)
- Injects path variables (`@PathVariable`)
- Injects query params (`@RequestParam`)
- Calls the matched method

Example:

```
@PostMapping
public EmployeeDTO create(@RequestBody EmployeeDTO dto)
```

Spring:

- Parses JSON body
- Creates `EmployeeDTO`
- Calls `create(dto)`

How Spring Resolves Ambiguity (Important)

If **multiple methods could match**, Spring applies priority rules:

1. HTTP method match (`GET`, `POST`, etc.)
2. Exact path > path variables > wildcards
3. Consumes / Produces (`application/json`)
4. Request parameters (if specified)

If still ambiguous: ❌ Spring throws `AmbiguousMappingException`

Why Wrong Method Is Never Called

Spring **does not guess**.

If:

- Path does not match → 404
- HTTP method wrong → 405 (Method Not Allowed)
- Body invalid → 400

Spring fails **explicitly**, not silently.

Internal Components (Minimal but Correct)

```
DispatcherServlet
   ↓
HandlerMapping   (finds method)
   ↓
HandlerAdapter   (invokes method)
   ↓
Controller Method
```

You do not interact with these directly.

One Example Fully Explained

Controller

```
@RestController
@RequestMapping("/api/employees")
public class EmployeeController {

    @GetMapping("/{id}")
    public EmployeeDTO getById(@PathVariable Long id) { }
}
```

Request

```
GET /api/employees/10
```

Spring logic:

- `/api/employees` → controller match
- `/{id}` → method match
- `GET` → correct HTTP verb
- `10` → converted to `Long`
- Method invoked

Mental Model (Remember This)

> **Spring never "calls a method." Spring matches a request to a method.**

Interview-Grade Summary (Use This)

> "Spring MVC uses the DispatcherServlet to intercept requests and relies on HandlerMapping to match the HTTP method and URL to a controller method annotated with mapping annotations, then invokes it after resolving request parameters and request bodies."

If you want next:

- How Spring converts JSON → DTO
- How `@PathVariable` and `@RequestBody` are resolved
- How validation happens before method execution

Just say the topic name.