PathVariable

👉 Fetching a Single Job Post

In our previous API, we used /jobPosts to fetch all job posts. But what if we want to fetch just one specific job post by its ID?

- For example, we might want to access a job post with ID 3 using a URL like: http://localhost:8080/jobPost/3

- To handle this, we need to create a new API endpoint that:
  - Takes an ID from the URL
  - Returns only the matching job post

👉 Using @PathVariable

- The @PathVariable annotation allows us to capture values from the URL path.

Here's how we implement it:

```
@GetMapping("jobPost/{postId}")
public JobPost getJob(@PathVariable("postId") int postId){
    return service.getJob(postId);
}
```

- {postId} in the URL path defines a variable part of the URL
- @PathVariable("postId") extracts that variable and passes it to the method as postId
- Spring automatically converts the string from the URL to an integer
- Path variables are defined with curly braces in the URL pattern: {postId}
- The @PathVariable annotation connects the URL variable to a method parameter
- You can have multiple path variables in a single URL (e.g., /category/{categoryId}/job/{jobId})

Example:

👉 **Controller**

```java
package com.telusko.spring_boot_rest.controller;

import java.util.List;

import com.telusko.spring_boot_rest.model.JobPost;
import com.telusko.spring_boot_rest.service.JobService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;


@RestController
@CrossOrigin(origins = "http://localhost:3000")
public class JobRestController {

    @Autowired
    private JobService service;


    @GetMapping("jobPosts")
    public List<JobPost> getAllJobs() {
        return service.getAllJobs();
    }

    @GetMapping("jobPost/{postId}")
    public JobPost getJob(@PathVariable("postId") int postId){
         return service.getJob(postId);
    }
}
```

👉 **Service**

```java
package com.telusko.spring_boot_rest.service;

import com.telusko.spring_boot_rest.model.JobPost;
import com.telusko.spring_boot_rest.repo.JobRepo;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class JobService {
    @Autowired
    public JobRepo repo;


    // method to add a jobPost
    public void addJob(JobPost jobPost) {
        repo.addJob(jobPost);

    }


    //method to return all JobPosts
    public List<JobPost> getAllJobs() {
        return repo.getAllJobs();

    }

    // method to return single JobPost by postId
    public JobPost getJob(int i){
        return repo.getJob(i);
    }


}
```

👉 Repository

```java
package com.telusko.spring_boot_rest.repo;

import com.telusko.spring_boot_rest.model.JobPost;
import org.springframework.stereotype.Repository;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

@Repository
public class JobRepo {


  // ArrayList to store JobPost objects
  List<JobPost> jobs = new ArrayList<>(Arrays.asList(

    new JobPost(1, "Java Developer", "Must have good experience in core Java and advanced Java", 2,
                List.of("Core Java", "J2EE", "Spring Boot", "Hibernate")),


    new JobPost(2, "Frontend Developer", "Experience in building responsive web applications using React", 3,
                List.of("HTML", "CSS", "JavaScript", "React")),


    new JobPost(3, "Data Scientist", "Strong background in machine learning and data analysis", 4,
                List.of("Python", "Machine Learning", "Data Analysis")),

    new JobPost(4, "Network Engineer", "Design and implement computer networks for efficient data communication", 5,
                List.of("Networking", "Cisco", "Routing", "Switching")),


    new JobPost(5, "Mobile App Developer", "Experience in mobile app development for iOS and Android", 3,
                List.of("iOS Development", "Android Development", "Mobile App"))
    ));

  // method to return all JobPosts
  public List<JobPost> getAllJobs() {
       return jobs;
  }

  // method to save a job post object into arrayList
  public void addJob(JobPost job) {
       jobs.add(job);
       System.out.println(jobs);

  }

  // method to return single JobPost by postId
  public JobPost getJob(int i){

      for(JobPost job: jobs){

          if(job.getPostId()==i)
               return job;
      }

       return null;
  }

}
```

- When we test http://localhost:8080/jobPost/3 in the postman, we get only the job post with ID 3: