# Sending CSRF Token

## Purpose of the getCsrfToken Method:

- This method is used to fetch the Cross-Site Request Forgery (CSRF) token from the HttpServletRequest object.
- It retrieves the token stored under the attribute "_csrf" in the request.

## CSRF Token Retrieval:

- CSRF tokens are a security mechanism to prevent unauthorized or malicious actions on behalf of authenticated users.

In this implementation, the token is extracted using:

```
return (CsrfToken) request.getAttribute("_csrf");
```

## Endpoint:

- The endpoint "/csrf-token" is mapped to this method, which allows clients (e.g., frontend applications) to retrieve the CSRF token as needed for secure communications.

## Practical Use Case:

- The retrieved token can be sent to the client (e.g., in the response body or headers) and is expected to be included in subsequent requests requiring CSRF protection.

## Spring Security Integration:

- This method assumes that Spring Security is configured to handle CSRF tokens automatically, where tokens are generated and managed for each session or request.
- The attribute "_csrf" is a standard key used by Spring Security to store the CSRF token.

## Usage in Client Requests:

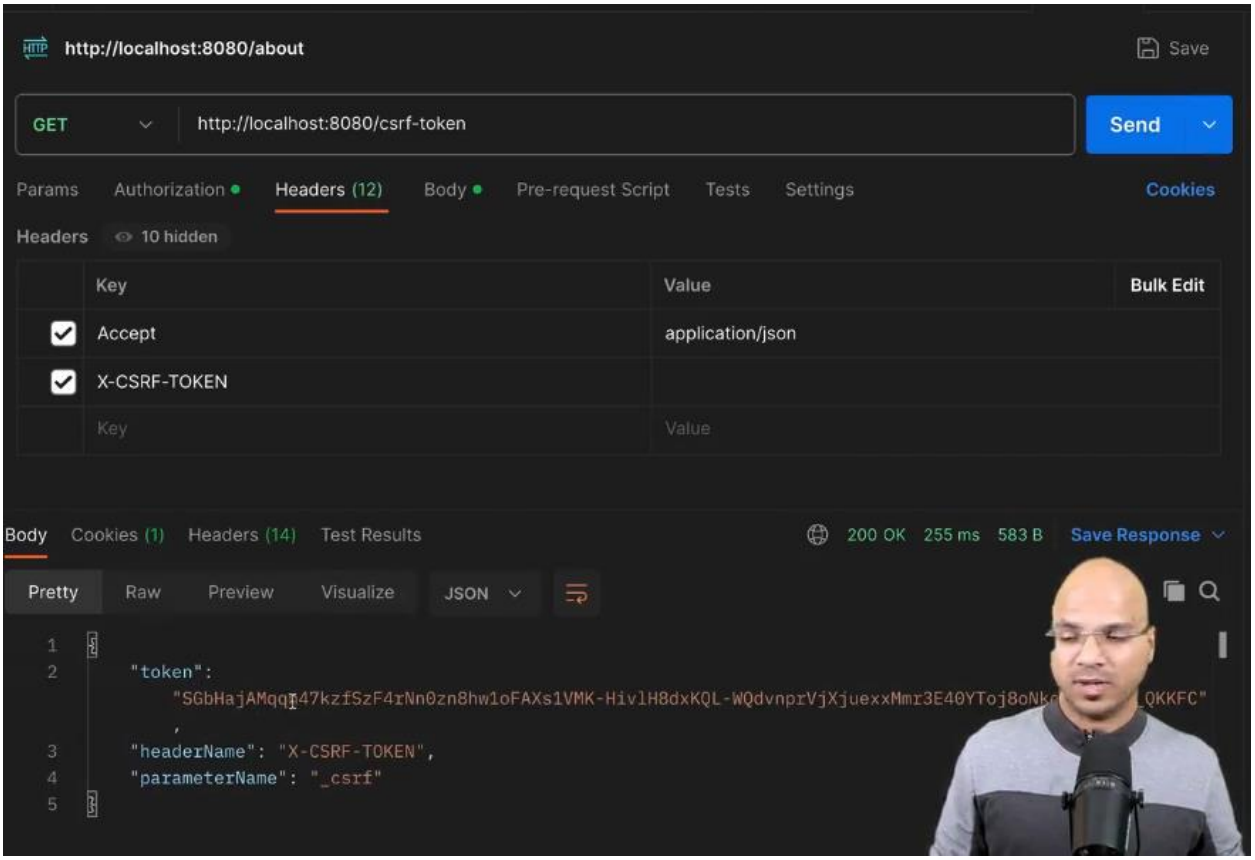
- When performing actions like POST, PUT, or DELETE, the client must include this token in the request headers (e.g., X-CSRF-Token) or as a hidden form field.

## Security Best Practices:

- Ensure that this endpoint ("/csrf-token") is not exposed to unauthorized access.
- Use HTTPS to protect the transmission of the CSRF token.
- Avoid caching responses from this endpoint, as CSRF tokens are session-specific.

**StudentsController.java**

```
@RestController
public class StudentController {

    @Autowired
    private StudentService studentService;

    @GetMapping("/csrf-token")
    public CsrfToken getCsrfToken(HttpServletRequest request) {
        return (CsrfToken) request.getAttribute("_csrf");
    }

    @GetMapping("/students")
    public ResponseEntity<List<Student>> getStudents(){
        return ResponseEntity.ok(studentService.getStudents());
    }

    @PostMapping("/students")
    public ResponseEntity<Student> addStudent(@RequestBody Student student ){
        if(studentService.addStudent(student)) {
            return ResponseEntity.ok(student);
        }else {
            return ResponseEntity.internalServerError().build();
        }
    }

}
```



http://localhost:8080/about

GET http://localhost:8080/csrf-token

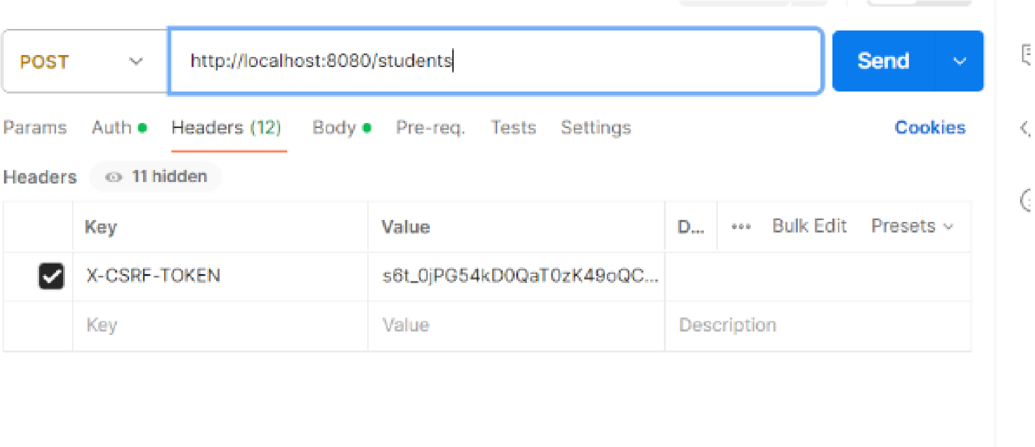
Headers (12)

| Key | Value |
|---|---|
| ☑ Accept | application/json |
| ☑ X-CSRF-TOKEN | |

200 OK 255 ms 583 B

```
{
    "token":
        "SGbHajAMqqn47kzfSzF4rNn0zn8hw1oFAXs1VMK-HivlH8dxKQL-WQdvnprVjXjuexxMmr3E40YToj8oNk[illegible]QKKFC"
        ,
    "headerName": "X-CSRF-TOKEN",
    "parameterName": "_csrf"
}
```

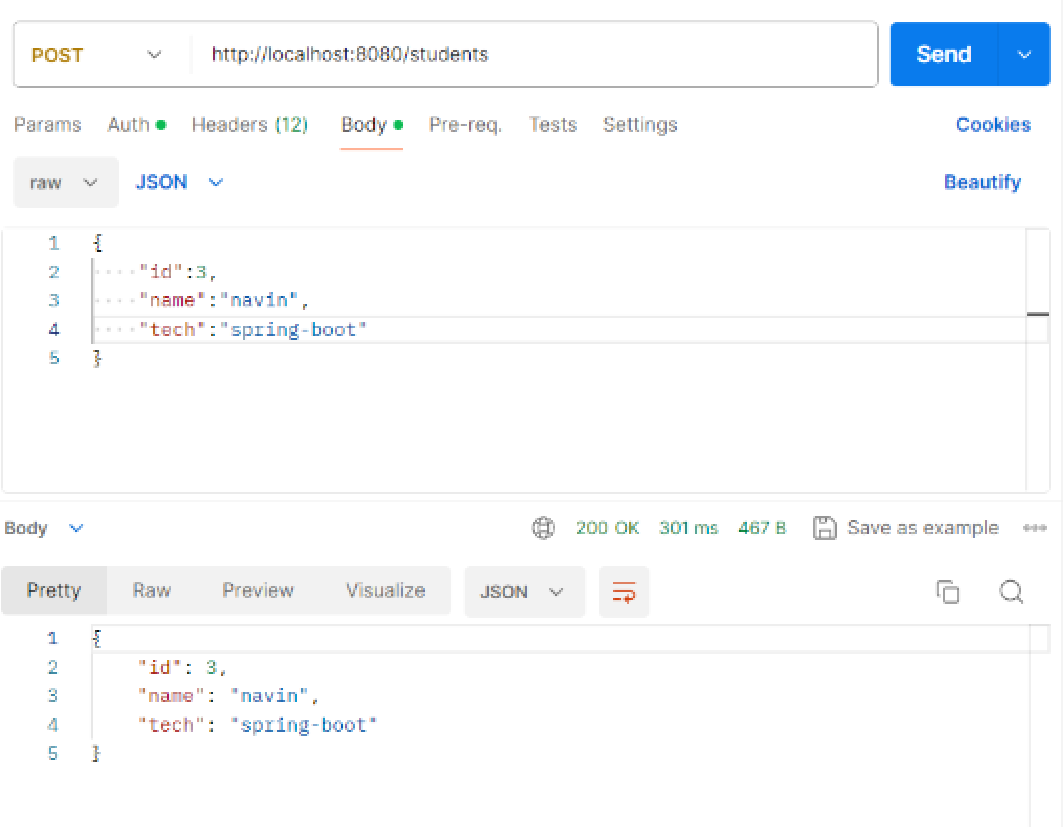
## Now after adding csrf token:



POST http://localhost:8080/students

Headers (12)

| Key | Value | D... |
|---|---|---|
| ☑ X-CSRF-TOKEN | s6t_0jPG54kD0QaT0zK49oQC... | |



http://localhost:8080/students

```
{
    "id":3,
    "name":"navin",
    "tech":"spring-boot"
}
```

200 OK 301 ms 467 B

```
{
    "id": 3,
    "name": "navin",
    "tech": "spring-boot"
}
```