


# [개정판] Spring Boot를 이용한 RESTful API 개발

{ REST API } +  Spring  
Boot

```
class Book {
    def setData(self, title, price, author):
        self.title = title
        self.price = price
        self.author = author
}

var fs = require('fs');
fs.readFile('./JONE.txt', '* 1 *',
    function (err, data) {
        console.log(data); // 3
    });

@interface NextInnovationDelegate : NSObject <UIApplicationDelegate>
```

- Section 0: Web Service & Web Application
- Section 1: Spring Boot로 개발하는 RESTful API
- **Section 2: User Service API 추가**
- Section 3: RESTful Service 기능 확장
- Section 4: Spring Boot API 사용
- Section 5: JPA 사용
- Section 6: REST API 설계 가이드

## 2. Section

# User Service API 추가

- User Domain 생성
- GET
- POST
- Exception Handling
- DELETE

# User Service

```
@Data
@AllArgsConstructor
public class User {
    private Integer id;
    private String name;
    private Date joinDate;
}
```

```
@Component
public class UserDaoService {
    private static List<User> users = new ArrayList<>();

    private static int usersCount = 3;

    static {
        users.add(new User( id: 1, name: "Kenneth", new Date()));
        users.add(new User( id: 2, name: "Alice", new Date()));
        users.add(new User( id: 3, name: "Elena", new Date()));
    }

    public List<User> findAll() {
        return users;
    }
}
```

사용자 수

```
public User save(User user) {
    if (user.getId() == null) {
        user.setId(++usersCount);
    }

    users.add(user);

    return user;
}

public User findOne(int id) {
    for (User user : users) {
        if (user.getId() == id) {
            return user;
        }
    }

    return null;
}
```

# Retrieve User resource

```
@RestController
public class UserController {
    private UserDaoService service;

    public UserController(UserDaoService service) {
        this.service = service;
    }

    @GetMapping("/users")
    public List<User> retrieveAllUsers() {
        return service.findAll();
    }

    @GetMapping("/users/{id}")
    public User retrieveUser(@PathVariable int id) {
        return service.findOne(id);
    }
}
```

localhost:8088/users

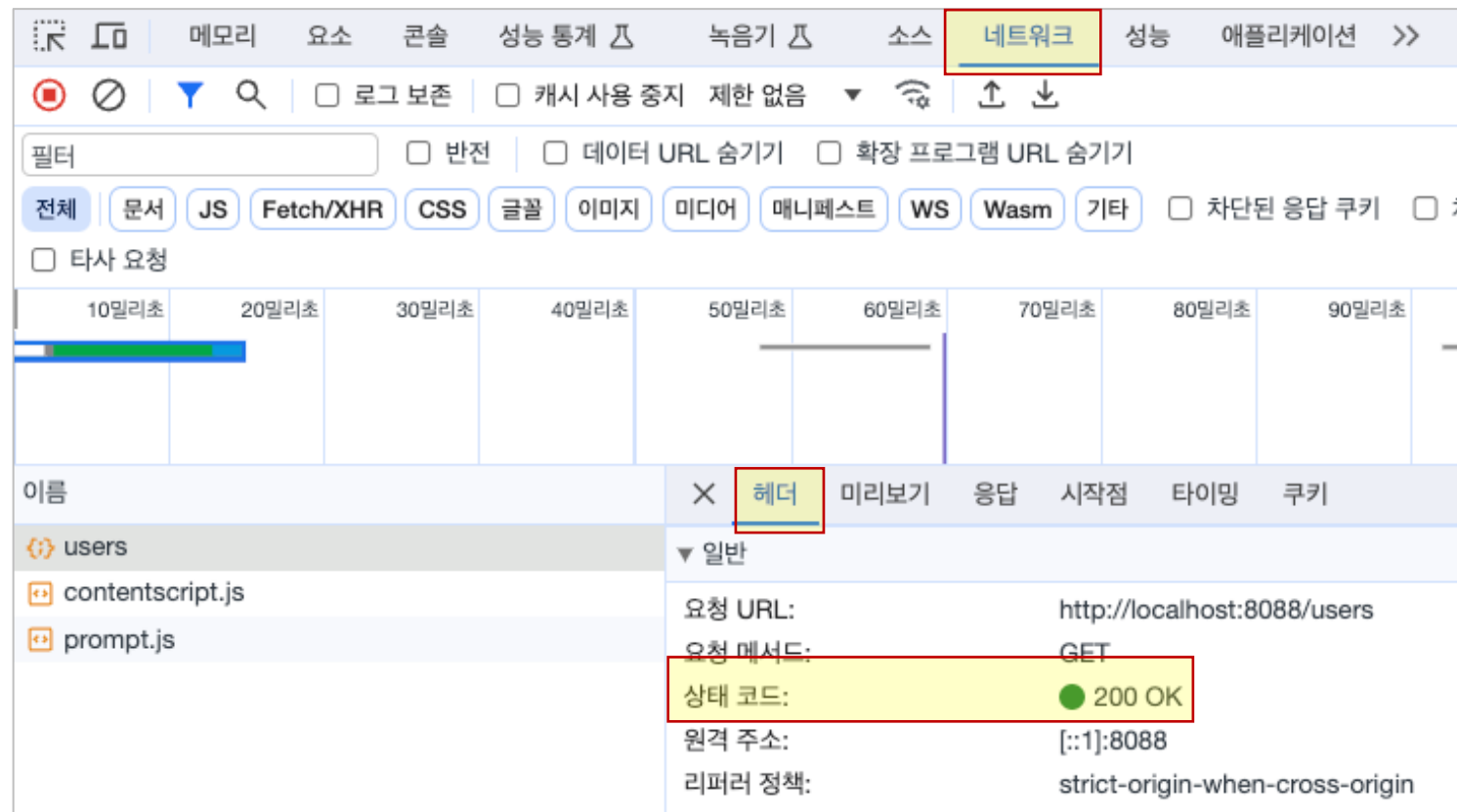
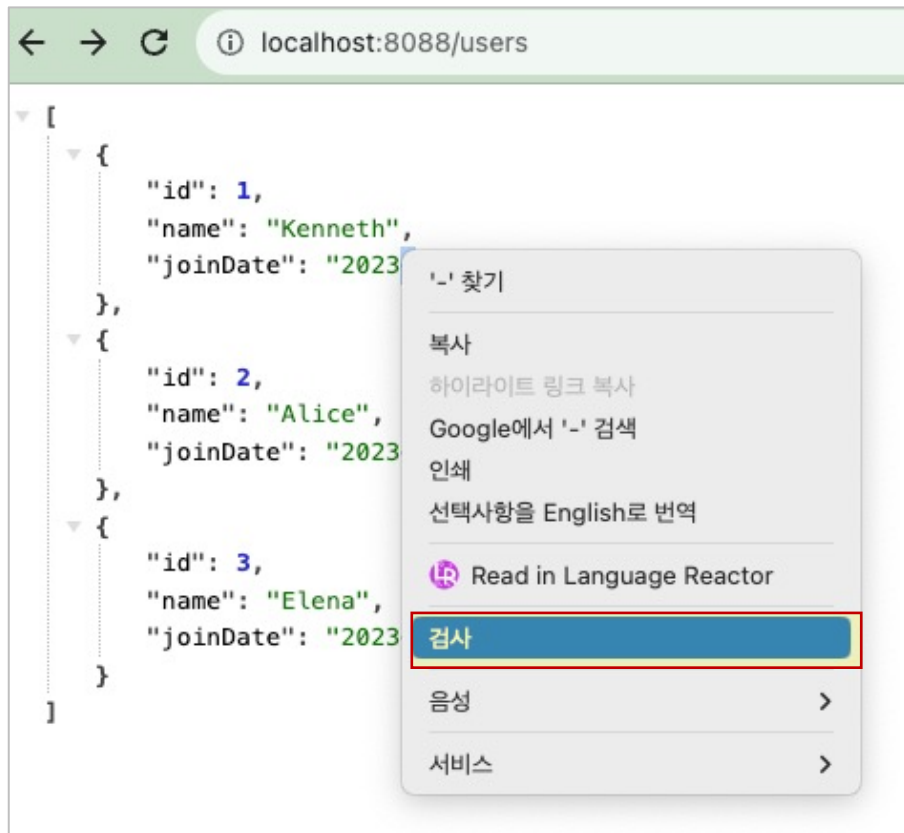
```
[
  {
    "id": 1,
    "name": "Kenneth",
    "joinDate": "2023-11-20T10:14:23.474+00:00"
  },
  {
    "id": 2,
    "name": "Alice",
    "joinDate": "2023-11-20T10:14:23.474+00:00"
  },
  {
    "id": 3,
    "name": "Elena",
    "joinDate": "2023-11-20T10:14:23.474+00:00"
  }
]
```

localhost:8088/users/2

```
{
  "id": 2,
  "name": "Alice",
  "joinDate": "2023-11-20T10:14:23.474+00:00"
}
```

# Create User resource

- <http://localhost:8080/users>



# Create User resource

## ■ @RequestBody

```
{
  "id": 1,
  "name": "Kenneth",
  "joinDate": "2023-11-20T10:14:23.474+00:00"
}
```

```
// CREATED
// input - details of user
// output - CREATED & Return the created URI
@PostMapping("/users")
public void createUser(@RequestBody User user) {
    User savedUser = service.save(user);
}
```

## ■ Post request 실행

- rest client 필요
- Postman or curl 명령어

# Create User resource

POST 127.0.0.1:8088/users

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON**

```
1 {  
2   "name": "Dowon Lee",  
3   "birthDate": "2000-11-15T03:21:32.896+0000"  
4 }
```

GET 127.0.0.1:8088/users/4

Params **Authorization** Headers (6) Body Pre-request Script Tests Settings

Query Params

	Key	Value	Des
	Key	Value	Des

Body Cookies Headers (5) Test Results ⌚ Status: 200 OK

Pretty Raw Preview Visualize **JSON**

```
1 {  
2   "id": 4,  
3   "name": "Dowon Lee",  
4   "joinDate": "2023-11-15T03:21:32.896+00:00"  
5 }
```

*status code: 200*



# HTTP Status Code

- ServletUriComponentsBuild → URI 생성
- ResponseEntity 반환

POST 127.0.0.1:8088/users

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {  
2   "name": "Dowon Lee",  
3   "joinDate": "2023-11-15T03:21:32.896+0000"  
4 }
```

Body Cookies Headers (5) Test Results

Status: 201 Created Time: 121 ms

Pretty Raw Preview Visualize Text

Body	Cookies	Headers (5)	Test Results	Status: 201 Created	Time: 121 ms
		Key		Value	
		Location		http://127.0.0.1:8088/users/4	

# Exception Handling

- user id 존재하지 않을 때도 200 OK 반환
- throw new UserNotFoundException 추가

The screenshot shows a REST client interface with the following details:

- Body** tab is selected.
- Status:** 500 Internal Server Error (highlighted in a yellow box).
- Time:** 89 ms
- Size:** 5.2 KB
- Save as Example** button is visible.
- JSON** view is selected.
- JSON Body:**

```
1 {
2   "timestamp": "2023-11-20T10:31:49.433+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "trace": "kr.co.joneconsulting.newmyrestfulservice.exception.UserNotFoundException: ID[4] not found\n\tat kr.co.jonecon",
6   "message": "ID[4] not found",
7   "path": "/users/4"
8 }
```

# Exception Handling

- @ResponseStatus 추가

```
@ResponseStatus(HttpStatus.NOT_FOUND)
public class UserNotFoundException extends RuntimeException {
    public UserNotFoundException(String message) {
        super(message);
    }
}
```

Body Cookies Headers (5) Test Results

Status: 404 Not Found Time:

Pretty

Raw

Preview

Visualize

JSON



1

{

2

"timestamp": "2023-11-20T10:33:51.126+00:00",

3

"status": 404,

4

"error": "Not Found",

5

"trace": "kr.co.joneconsulting.newmyrestfulservice.exception.UserNotFoundException:

6

"message": "ID[4] not found",

7

"path": "/users/4"

8

}

# Generic Exception Handling

- generic exception → ExceptionResponse class

```
@Data
@AllArgsConstructor
@NoArgsConstructor
public class ExceptionResponse {
    private Date timestamp;
    private String message;
    private String details;
}
```

- ResponseEntityExceptionHandler 검색

```
@RestController
@ControllerAdvice
public class CustomizedResponseEntityExceptionHandler extends ResponseEntityExceptionHandler {

    @ExceptionHandler(Exception.class)
    public final ResponseEntity<Object> handleAllExceptions(Exception ex, WebRequest request) {
        ExceptionResponse exceptionResponse =
            new ExceptionResponse(new Date(), ex.getMessage(), request.getDescription(includeClientInfo: false));

        return new ResponseEntity(exceptionResponse, HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

# Generic Exception Handling

GET

127.0.0.1:8088/users/4

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

	Key	Value	Description
	Key	Value	Description

Body

Cookies

Headers (4)

Test Results

Status: 500 Internal Server Error

Time: 99 ms

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "timestamp": "2023-11-20T10:36:33.148+00:00",
3   "message": "ID[4] not found",
4   "details": "uri=/users/4"
5 }
```

# Generic Exception Handling

```
@ExceptionHandler(UserNotFoundException.class)
public final ResponseEntity<Object> handleUserNotFoundException(Exception ex, WebRequest request) {
    ExceptionResponse exceptionResponse =
        new ExceptionResponse(new Date(), ex.getMessage(), request.getDescription(includeClientInfo: false));

    return new ResponseEntity(exceptionResponse, HttpStatus.NOT_FOUND);
}
```

GET 127.0.0.1:8088/users/4

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (5) Test Results

Status: 404 Not Found Time: 91 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2023-11-20T10:37:34.607+00:00",
3   "message": "ID[4] not found",
4   "details": "uri=/users/4"
5 }
```

# Delete User resource

```
public User deleteById(int id) {
    Iterator<User> iterator = users.iterator();

    while (iterator.hasNext()) {
        User user = iterator.next();

        if (user.getId() == id) {
            iterator.remove();
            return user;
        }
    }

    return null;
}
```

UserDaoService.java

```
@DeleteMapping("/users/{id}")
public void deleteUser(@PathVariable int id) {
    User user = service.deleteById(id);

    if (user == null) {
        throw new UserNotFoundException(String.format("ID[%s] not found", id));
    }
}
```

UserController.java

# Delete User resource

**DELETE** 127.0.0.1:8088/users/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (4) Test Results

1

Status: 200 OK

**GET** 127.0.0.1:8088/users

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

	Key	Value	Description
--	-----	-------	-------------

Body Cookies Headers (5) Test Results

123456789101112

Status: 200 OK Time:

```
[
  {
    "id": 2,
    "name": "Alice",
    "joinDate": "2023-11-20T10:40:45.781+00:00"
  },
  {
    "id": 3,
    "name": "Elena",
    "joinDate": "2023-11-20T10:40:45.781+00:00"
  }
]
```