บทที่ 3

Routing & Middleware in Spring Boot

หัวข้อที่จะเรียนรู้

- แนวคิดของ Routing ใน Backend
- Spring Boot Routing ด้วย `@RestController`
- Path Variables & Query Parameters
- Middleware/Interceptor/Filter
- การจัดการ Error ด้วย Global Exception Handling

Routing คืออะไร?

- กำหนด **เส้นทาง (URL Path)** สำหรับรับ Request
- แต่ละ Route เชื่อมโยงกับ **Controller Method**
- รองรับ HTTP Method (GET, POST, PUT, DELETE)

Spring Boot Routing

- ใช้ Annotation:
  - `@RestController` → ระบุ class ที่เป็น API controller
  - `@RequestMapping` → ระบุ path หลัก
  - `@GetMapping`, `@PostMapping`, `@PutMapping`, `@DeleteMapping`

ตัวอย่าง:

```
@RestController
@RequestMapping("/users")
public class UserController {
    @GetMapping("/{id}")
    public String getUser(@PathVariable int id) {
        return "User ID = " + id;
    }
}
```

Path Variable vs Query Parameter

Path Variable: `/users/1`

```
@GetMapping("/{id}")
public String getUser(@PathVariable int id) { ... }
```

Query Parameter: `/users?role=admin`

```
@GetMapping
public String getUsers(@RequestParam String role) { ... }
```

Middleware Concept

- ทำงานก่อน/หลังการประมวลผล Request
- ใช้สำหรับ:
    - Logging
    - Authentication/Authorization
    - Error Handling
    - Request Modification

Middleware ใน Spring Boot

- **Filter**
  - ทำงานในระดับ Servlet
  - ใช้ `OncePerRequestFilter`
- **Interceptor**
  - ทำงานใน Spring MVC
  - ใช้ `HandlerInterceptor`

ตัวอย่าง Filter (Logging)

```
@Component
public class LogFilter extends OncePerRequestFilter {
    @Override
    protected void doFilterInternal(HttpServletRequest req,
                                    HttpServletResponse res,
                                    FilterChain chain)
                                    throws IOException, ServletException {
        System.out.println("Request URI: " + req.getRequestURI());
        chain.doFilter(req, res);
    }
}
```

Global Exception Handling

- ใช้ `@ControllerAdvice` + `@ExceptionHandler`
- รวมศูนย์การจัดการ Error
- ส่ง Response ในรูปแบบมาตรฐาน

ตัวอย่าง:

```
@ControllerAdvice
public class GlobalExceptionHandler {
    @ExceptionHandler(Exception.class)
    public ResponseEntity<String> handleException(Exception ex) {
        return ResponseEntity.status(500).body("Error: " + ex.getMessage());
    }
}
```

Lab

1. สร้าง Controller `/products`
    - GET `/products/{id}` → คืนข้อมูลสินค้า
    - GET `/products?category=...` → คืนสินค้าตามหมวดหมู่
2. เขียน Interceptor สำหรับ Logging
    - log path และ method ของทุก request

Assignment

- เขียน API `/users` ที่มี:
  - GET `/users/{id}`
  - GET `/users?role=...`
  - POST `/users` (สร้างผู้ใช้ใหม่)
- เพิ่ม Interceptor ที่ log:
  - เวลา request
  - path
  - method
- เพิ่ม Global Exception Handler ที่คืน JSON:

```
{ "status": 500, "error": "Internal Server Error", "message": "..." }
```