บทที่ 6

Authentication & Authorization

with Spring Security + JWT

หัวข้อที่จะเรียนรู้

- ความแตกต่างระหว่าง Authentication และ Authorization
- แนวคิด Session-based vs Token-based Auth
- JWT (JSON Web Token) คืออะไร
- Spring Security Basics
- Implement JWT Authentication ใน Spring Boot

Authentication vs Authorization

- **Authentication**
  - การตรวจสอบตัวตนผู้ใช้
  - เช่น Login ด้วย Username/Password
- **Authorization**
  - การกำหนดสิทธิ์การเข้าถึง Resource
  - เช่น Admin เข้าถึง `/admin/*` แต่ User ธรรมดาเข้าไม่ได้

Session-based Authentication

- Server เก็บ Session state (user info)
- Client เก็บ Session ID (เช่นใน Cookie)
- ปัญหา:
  - Scalability → ต้องแชร์ session ระหว่างหลาย server
  - ไม่เหมาะกับระบบ distributed/microservices

Token-based Authentication

- Server ออก Token หลัง Login
- Client ส่ง Token มาพร้อมทุก request
- Stateless → ไม่ต้องเก็บ session ที่ server
- Token มักใช้รูปแบบ **JWT**

JWT (JSON Web Token)

- ข้อมูลเข้ารหัส Base64 แบ่งเป็น 3 ส่วน

```
Header.Payload.Signature
```

ตัวอย่าง Payload:

```
{
  "sub": "alice",
  "role": "ADMIN",
  "exp": 1717305600
}
```

- มีวันหมดอายุ (Expiration)
- ใช้ตรวจสอบสิทธิ์ฝั่ง server ได้

Spring Security Basics

- Framework ความปลอดภัยใน Spring Boot
- ใช้ **Filter Chain** ตรวจสอบทุก request
- สามารถกำหนด Role-based Access Control

ตัวอย่าง:

```
http
  .authorizeHttpRequests(auth -> auth
      .requestMatchers("/admin/**").hasRole("ADMIN")
      .anyRequest().authenticated()
  );
```

ขั้นตอน JWT Auth ใน Spring Boot

1. **User Login** → ส่ง username/password
2. **Server ตรวจสอบ** → ถ้าถูกต้อง ออก JWT
3. **Client เก็บ JWT** → ใน LocalStorage/SessionStorage
4. **Client ส่ง JWT** ใน Header:

```
Authorization: Bearer <token>
```

5. **Server ตรวจสอบ JWT** → ถ้า valid ให้เข้าถึง API ได้

Spring Boot: JWT Dependencies

Maven:

```
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt-api</artifactId>
  <version>0.11.5</version>
</dependency>
```

ตัวอย่างโค้ด: JWT Utility

```
public class JwtUtil {
    private String secret = "mysecretkey";

    public String generateToken(String username) {
        return Jwts.builder()
            .setSubject(username)
            .setExpiration(new Date(System.currentTimeMillis() + 1000*60*60))
            .signWith(SignatureAlgorithm.HS256, secret)
            .compact();
    }
}
```

Lab

1. ติดตั้ง Spring Security + JWT
2. สร้าง API `/auth/login` → คืน JWT เมื่อ login สำเร็จ
3. เพิ่ม Security Filter ตรวจสอบ JWT
4. สร้าง Endpoint:
    - `/users/me` → คืนข้อมูลผู้ใช้จาก JWT
    - `/admin/dashboard` → ให้สิทธิ์เฉพาะ Admin

Assignment

- พัฒนา Authentication System:
  - `/auth/login` → รับ username/password และออก JWT
  - `/auth/register` → ลงทะเบียนผู้ใช้ใหม่
  - `/users/me` → คืนข้อมูลผู้ใช้ (ต้องส่ง JWT)
- เพิ่ม Role-based Access:
  - `/admin/*` → เฉพาะ Admin
  - `/users/*` → เฉพาะ User ที่ login แล้ว
- ส่งโค้ด + ตัวอย่าง JWT + Screenshot การทดสอบด้วย Postman