JWT in Spring Boot

What is JWT?

A quick and safe method of sending data between parties as a JSON object is JWT (JSON Web Token). Stateless authentication is frequently handled by Spring Boot, particularly in RESTful APIs.

Real-Life Example (Analogy)

Consider JWT to be similar to a movie ticket. Your ticket includes the movie name, time, and seat number when you purchase it. You enter using that ticket; you don't have to present your ID each time. In a similar vein, JWT is used to access restricted routes without repeatedly logging in and contains user information.

How JWT Works in Spring Boot

1. A user logs in with a username and password.
2. If credentials are valid, a JWT token is generated and sent back.
3. The client stores the token (usually in local storage or cookies).
4. For every subsequent request, the token is sent in the Authorization header as:
   Authorization: Bearer <token>
5. A JWT Filter checks the token before processing the request.
6. If valid, Spring Security authenticates the user and allows access.

Why Use JWT?

- Stateless: No need to store sessions on the server.
- Scalable: Ideal for microservices and distributed systems.
- Secure: Tokens are signed to prevent tampering.
- Flexible: Can include custom claims such as roles or permissions.