**Authentication:** Authentication is the process of verifying the identity of a user. It answers the question, "Who are you?"

**Authorization:** Authorization determines what an authenticated user is allowed to do. It answers the question, "What can you do?"

**2. Authentication in Spring Boot**

**Overview:** In Spring Boot, authentication can be implemented using Spring Security, which provides a comprehensive and customizable framework for securing applications.

**Steps and Components:**

1. **AuthenticationFilter:** Intercepts login requests and extracts user credentials.
2. **AuthenticationManager:** Processes the authentication request.
3. **AuthenticationProvider:** Validates the credentials.
4. **UserDetailsService:** Loads user-specific data from a data source.
5. **PasswordEncoder:** Encodes and matches passwords.
6. **SecurityContext:** Stores authentication information.

**3. Authorization in Spring Boot**

**Overview:** Authorization in Spring Boot using Spring Security involves defining roles and permissions and restricting access to resources based on these roles.

**Steps and Components:**

1. **Roles and Authorities:** Define roles and authorities in the database or as constants.
2. **Method Security:** Use annotations to secure methods.
3. **URL Security:** Configure HTTP security to restrict access to URLs based on roles.

### 1. ****Introduction to JWT****

**Authentication:** Authentication with JWT involves generating a token after verifying the user's credentials. This token is then used for subsequent requests to identify the user.

**Authorization:** Authorization determines what resources a user can access based on the claims encoded in the JWT token.

### 2. ****Authentication Process with JWT in Spring Boot****

#### Steps:

1. **User Login:** User submits their credentials (username and password).
2. **Authenticate User:** The application verifies the credentials.
3. **Generate JWT:** Upon successful authentication, the application generates a JWT.
4. **Return Token:** The JWT is sent back to the client.
5. **Token Usage:** The client includes the JWT in the **Authorization** header of subsequent requests.
6. **Token Validation:** The application validates the JWT for every request to ensure it is valid and has not expired.