Spring Security JWT secures APIs by enabling stateless authentication using JSON Web Tokens. Here's how it works:

1. **Token Issuance**:
   The client sends credentials to an authentication endpoint. Upon successful authentication, the server generates a signed JWT containing user information and permissions.
2. **Token Usage**:
   The client includes the JWT in subsequent requests' Authorization header (Bearer <token>).
3. **Token Validation**:
   A custom filter (e.g., JwtAuthenticationFilter) intercepts incoming requests, validates the token, and loads user details into the SecurityContext.

Advantages:

- Stateless, suitable for distributed systems.
- Scalable, no session storage required.
- Flexible, allowing custom claims in tokens.

Best Practices:
Use short-lived tokens, implement HTTPS, and secure signing keys, and include refresh tokens for extended sessions.