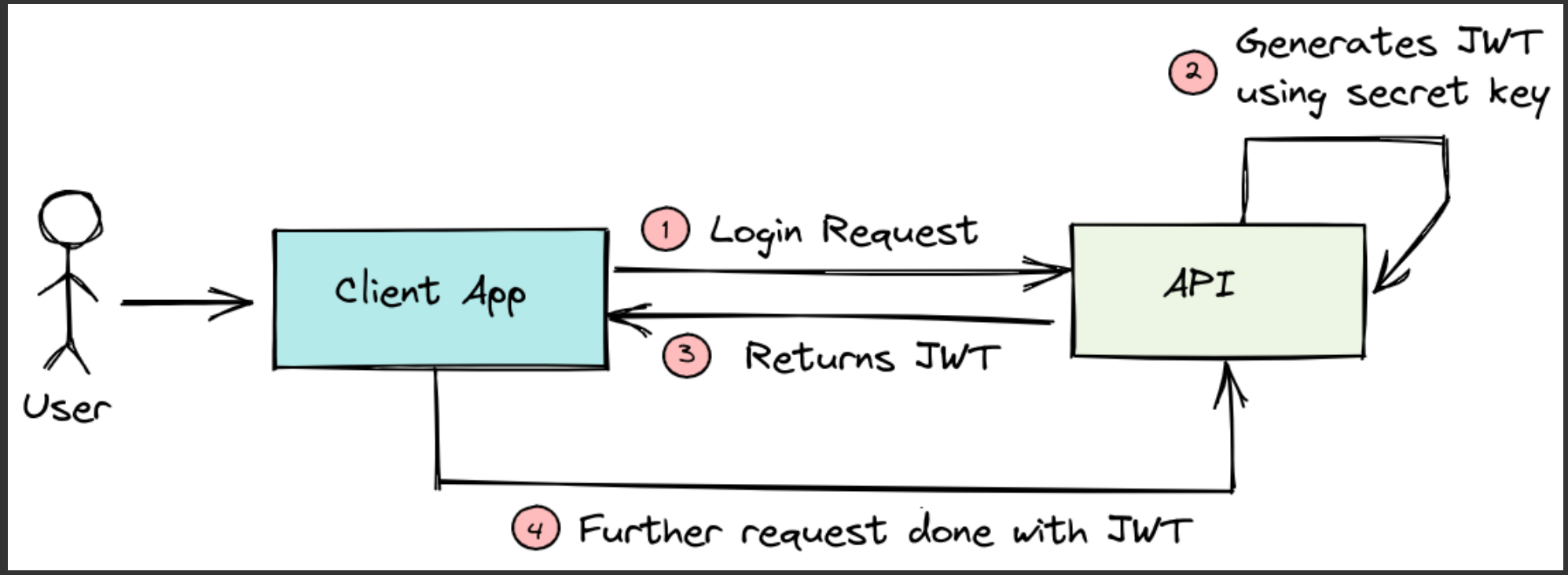
CODING SHUTTLE

Spring Security

JWT - Refresh Token and Access Token

Using JWT For Authentication



Issues with JWT Access Tokens

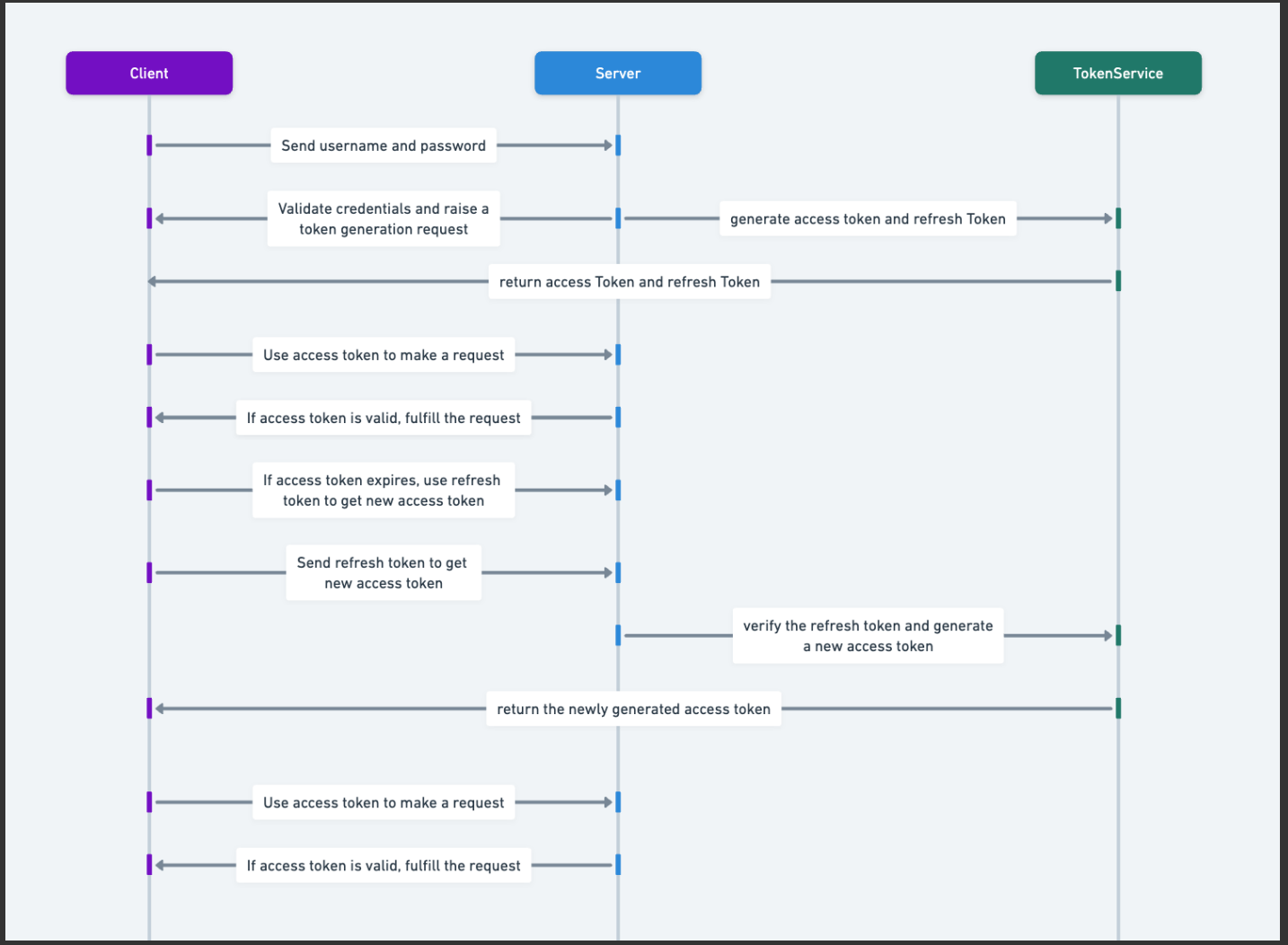
1. If an access token is used for an extended period, its exposure in requests increases the risk of it being intercepted or misused. Short-lived tokens reduce this risk by limiting how long a stolen token can be used.

2. Since access tokens are typically short-lived, users may need to reauthenticate frequently if a refresh token is not used. This can disrupt the user experience and reduce convenience.

The solution is to use Two Tokens:

1. Access Token
2. Refresh Token

Using Two Tokens instead of One



Using Two Tokens instead of One

Using two tokens, an access token and a refresh token, enhances both security and user experience.

Access tokens are short-lived, limiting the damage if compromised since they expire quickly. Refresh tokens are long-lived and only used to obtain new access tokens, reducing the exposure of long-term credentials and allowing them to be stored more securely, such as in HTTP-only cookies.

Additionally, even if the access token expires, the refresh token can be used to maintain session continuity without disrupting the user.

What if Refresh Token is Compromised?

1. Try to avoid this
   a. We can minimize the compromise by securely storing the Refresh tokens in Same-Site, HTTP only and Secure Cookies.
   b. Use HTTPS for transferring refresh tokens.

2. Use Database
   a. By maintaining sessions, We can invalidate the refresh tokens as soon as a new /login or /refresh request arrives. But this requires DB lookups as we will be storing the refreshToken in db.