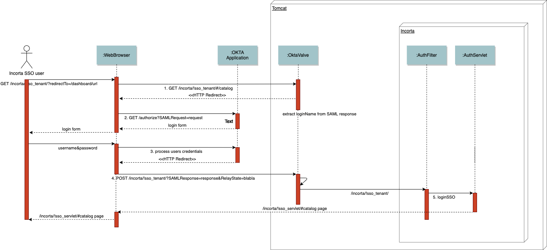
# SSO Okta valve overview

## quick overview:

the following sequence diagram shows okta workflow of authenticating users in Incorta using Okta



note that OktaValve is defined in tomcat so its not defined in the application level in tomcat hence it doesn’t have access to incorta classes because OktaValve exists in jar outside the jar of incorta which have advantages and disadvantages.

Advantages is that separated the SSO implementation from incorta so if there is any implementation change in customers you can send it to jar incorta/sso but you don't need to generate a new build.

Disadvantage is that it’s not able to use incorta classes in OktaValve for example you can’t get context in OktaValve.

## Okta login workflow:

1. The user enters incorta url attached with the tenant in the url to the browser then OktaValve checks if the user have an active session or not in current case the user doesn’t have session then checks for SAML response if exists in this case the SAML response doesn’t exist in request then OktaValve constructs the URL. and the SAML request that will be send to okta and redirects the browser to that URL with status code 302
2. Okta application checks if the user have an active session in okta in this case the user doesn’t have active session in okta then it returns to the browser by the login page of okta
3. The user submits the username and password to okta successfully then okta create a new session and then okta tells the browser to redirect to the SSO url configured in okta application which is the incorta URL with tenant attached and adding to it SAML response
4. When accessing incorta again then OktaValve is invoked and checks if it has an internal active session in incorta which in this case the user doesn’t have internal session in incorta so it checks if the request have SAML response then decodes the SAML response and extract from it the loginName of the user and adds it to the request and passes the Valve to proceed to incorta.
5. The first class in incorta that runs is AuthFilter the checks if the url containing the tenant then its from the SSO and checks if there is a session if not then checks if the tenant included in the URL exists and is SSO enabled then it calls AuthServlet.loginSSO which takes the request and response and gets the loginName from the request and creates a session for the user with the attached LoginName
6. The next time any request is passed to tomcat, OktaValve get invoked and checks if the user have internal active session then it passes the user to incorta.