Tight Integration

The easiest way to make use of Yellowfin's interactivity is to convert the platform as a re-branded, tightly integrated portal. The aim is to create a seamless link between the parent application and Yellowfin, so the end-user is unaware that they have navigated from one application to another. Using a development language other than Java should not increase the difficulty of implementing tight integration.

Main Steps

The main requirements for implementing tight integration are:

- A custom header and footer, usually replicating the top / bottom navigation elements from the third-party application.
- An authentication bridge connecting the parent third-party application and Yellowfin via a Single Sign-On (SSO) link or a custom login page.
- Adding links in the header to navigate to the third-party application. See section below.

Add Links

The custom header should contain links that take a user to different pages within the third-party application and a “Reports” link that will take the user to Yellowfin. The action associated with clicking the “Reports” link will launch an SSO call to Yellowfin's web services allowing the user to be taken directly into Yellowfin (add Redirecting to Yellowfin with SSO token). A user can return to a particular page in the third-party application by clicking the other links in the header.

With SSO call to Yellowfin, you can determine where a user is taken after the log in has completed. This can be anything from a report or dashboard to the browse page or a time line. For more details on this, see our page on Session Options.

Multiple Entry Points

Individual links can be used for separate entry points into the application. For instance, one link could take you to the dashboard, another could take you to the Browse page. These different entry-points allow for a pseudo tight-integration implementation where the third-party application could load a list of the reports available for a particular user via web services. The list of reports could be rendered within the application natively and a user could view these reports which would be executed by performing SSO directly to that report in Yellowfin.

The process that performs the Single Sign-On into Yellowfin may need to perform other functions such as creating users and determining the correct group access depending on the nature of the implementation. For more details refer to our page on Single Sign On.

Rebranding

Some implementations will also require re-branding on a Client Org level. This can be done in different ways. Below are a few examples:

- Using the internal re-branding infrastructure. This allows for a customized header and footer for each Client Org and customized colors and fonts. These are configured through the Yellowfin Configuration and Report Styles user interface.
- It is possible to incorporate a dynamic header that will inspect the session information to determine which client header should be displayed. It essentially has an IF statement and can render different HTML or images based on the session's current Client Org.
- Another option is to configure Apache or IIS to deliver static content. This allows different hosts/domain names to deliver different styles and login pages but still run off a single instance of Yellowfin. For example: *client_A.myYellowfin.com* will look different to *client_B.myYellowfin.com* but will share the same instance of Yellowfin. Users going to those different domains will have a custom login page, which can be used to automatically log them in to a particular Client Org.

Limitations

An issue faced with Tight Integration is session timeouts. Timeouts occur when the third-party application doesn't receive a request within a specified period while a user is using Yellowfin. This can be fixed by embedding an image from the host application in the custom header used by Yellowfin. This will cause the user's browser to make a request to the host application each time a page is loaded in Yellowfin.