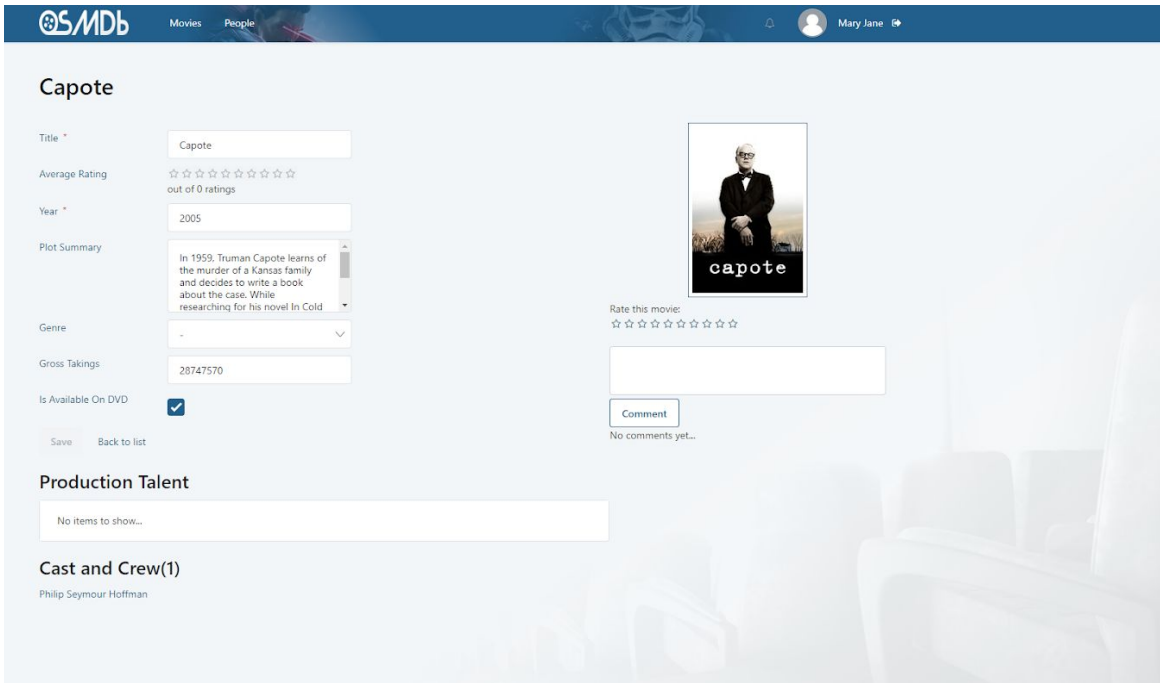
Themes and Styling Exercise



Introduction

In this exercise lab, we will do some visual tweaks to the application. Despite the icon chosen at the beginning influences the look and feel of the application, it is in fact possible to change and adjust how the application looks to our needs.

The first tweak we are going to do is to add a border to the movie poster, similar to a frame of a picture. This will consist of using some CSS and defining a new Style Class in the MovieDetail Screen.

Then, we will change the colors of the stars to gold, by creating another Style Class. However, since the stars are defined in a Web Block, the Style Class will be defined in the Block.

Finally, we will modify the Theme of the module, which will impact all Screens. We will apply some changes to the background and header of the pages, as well as the look and feel of the application menu.

In summary, in this specific exercise lab, we will:

- Add a border to the movie poster
- Make the rating stars gold
- Tweak the look and feel of the module's Theme

Table of Contents

Create a border for the movie poster

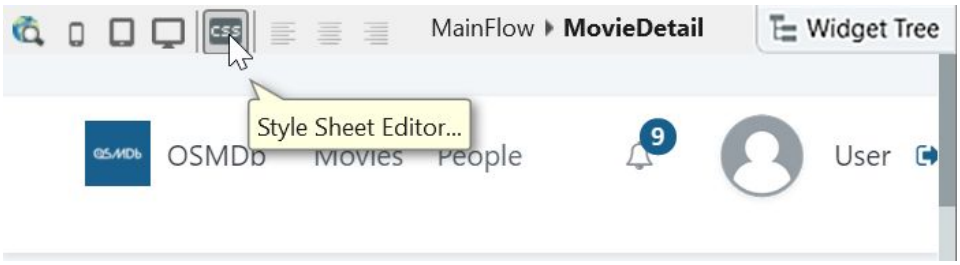
In this part of the exercise, we will create a new Style Class to customize the poster image. This Style Class will allow adding a border, like a frame of a picture, around a Widget.

The new Style Class will be created in the **MovieDetail** Screen and applied to the **Image** Widget with the movie poster.

1) Define a new Style Class to create a border around Widgets. This Class will be defined using CSS, and will add to the Widget a *white* background color and a border of *1px*, solid, with the *#185f8d* color.

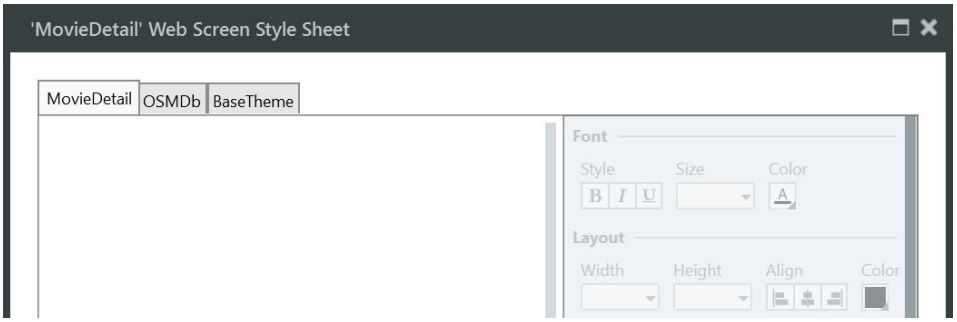
    a) Open the **MovieDetail** Screen.

    b) Click the **CSS** icon from the toolbar.



NOTE: It is also possible to access the Style Sheet editor by double-clicking the Style Sheet property of the **MovieDetail** Screen, in the properties area.

    c) Notice that the Style Sheet editor now has three tabs.



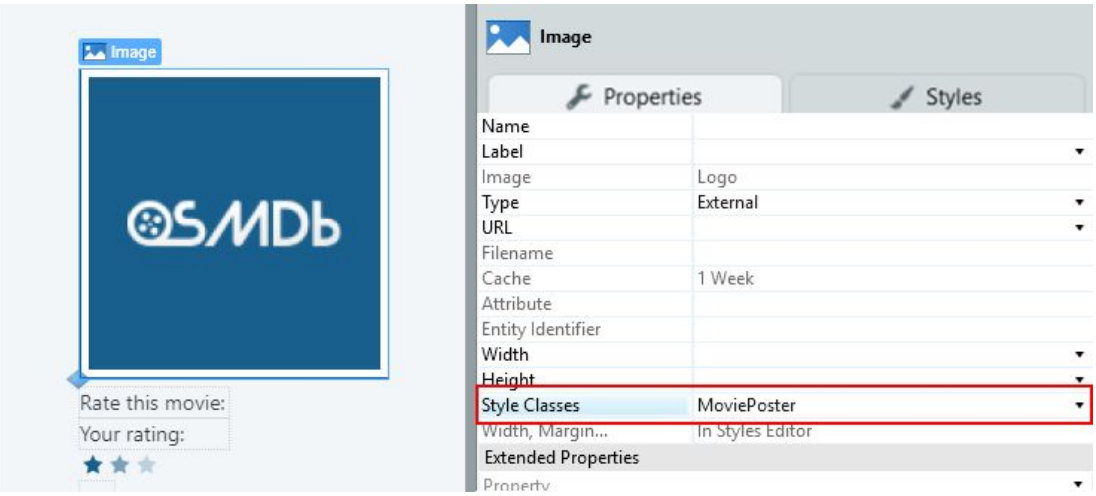
NOTE: Besides the two previous tabs (**OSMDb** and **Base Theme**), since you are in the context of a Screen, a new tab with the Screen name (**MovieDetail**) is also available.

All styles defined in the Screen tab only affect the respective Screen, and are only available for usage in the context of the same Screen.

d) Add the following CSS code to the **MovieDetail** tab, to create a *1px* border around Widgets.

```
.MoviePoster {
        background-color: white;
        padding: 5px;
        border: 1px solid #185f8d;
}
```

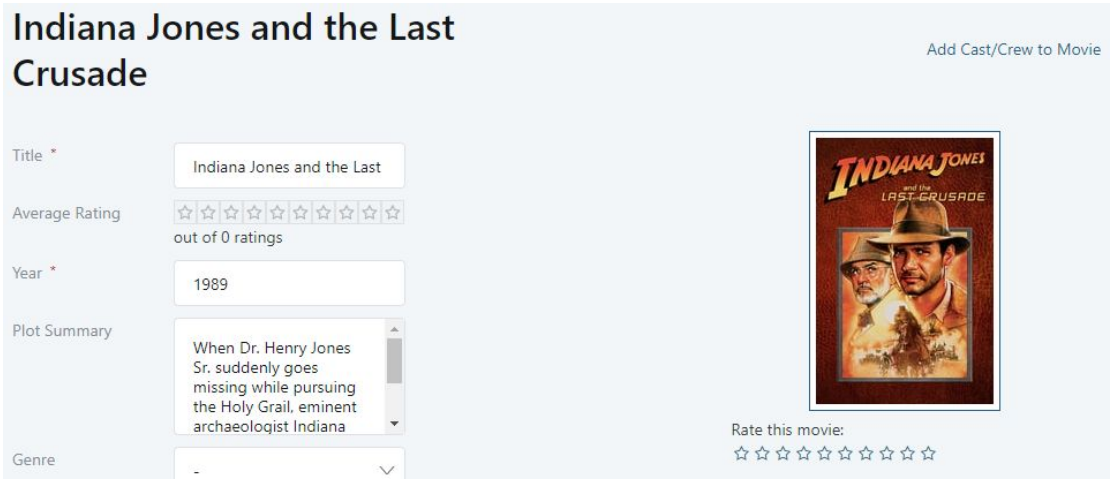
e) Select the poster image on the sidebar, and set the **Style Classes** property to *MoviePoster.*



2) Publish and test the look and feel changes made to the OSMDb application.

a) Click the **1-Click Publish** button to publish the application, and access it.

b) Navigate to **MovieDetail** Screen, by selecting an existing movie. The Screen should look like this



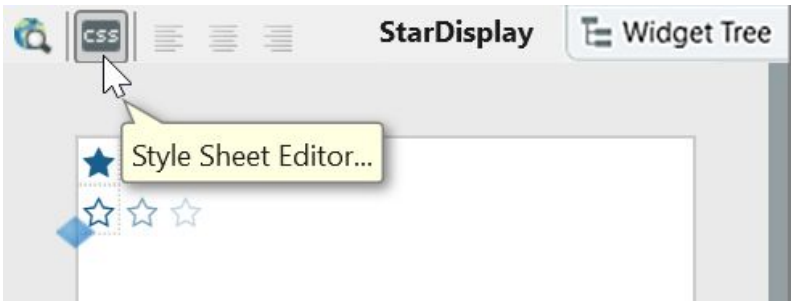
Create golden stars

In this part of the exercise, we will create a Style Class within the **StarDisplay** Web Block to customize the filled rating stars with a golden color.

1) Define a **GoldStar** Style Class in the **StarDisplay** Web Block. Use the color *#FFD700* to get the stars look gold.

   a) Open the **StarDisplay** Web Block

   b) Click the **CSS** icon in the toolbar to open the Web Block's Style Sheet editor



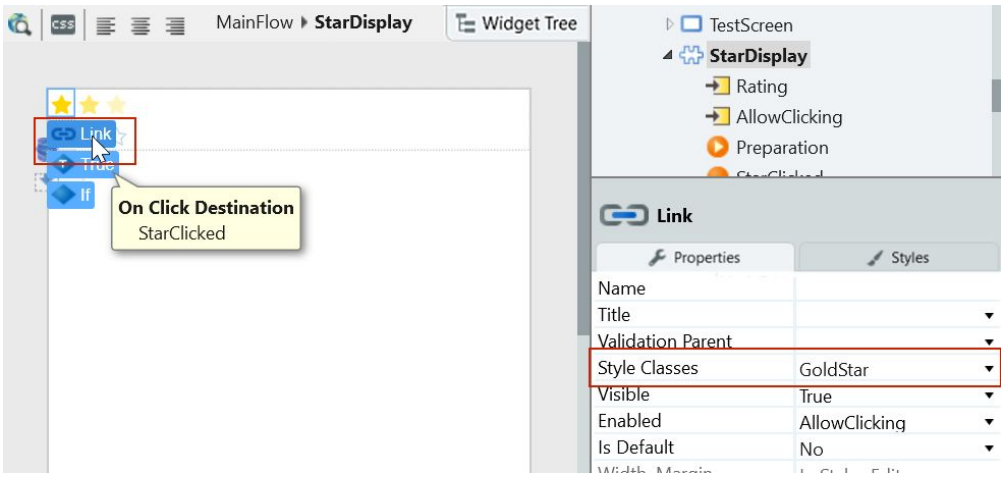
NOTE: In this case, since we are in the context of a Web Block, all styles defined in the first tab are specific to the current Web Block.

---

   c) Add the following CSS code to the **StarDisplay** tab, to define the *GoldStar* Style Class. Close the Stylesheet editor.

```
a.GoldStar {
    color: #FFD700;
}
```

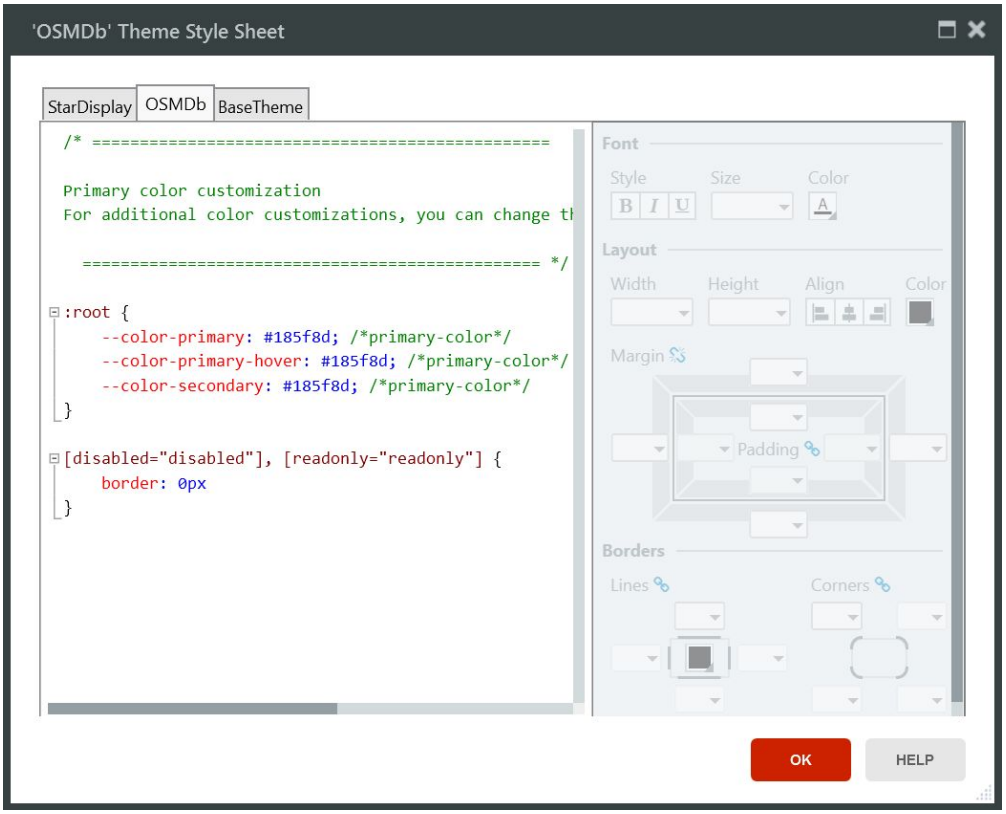
d) Select the filled star Link, and set the Style Classes property to GoldStar.



e) Open again the StarDisplay Web Block's Style Sheet editor and select the **OSMDb** tab in the **CSS** window.

f) Add the following code to the bottom of the CSS to ensure the star list records does not have a border and click the **OK** button.

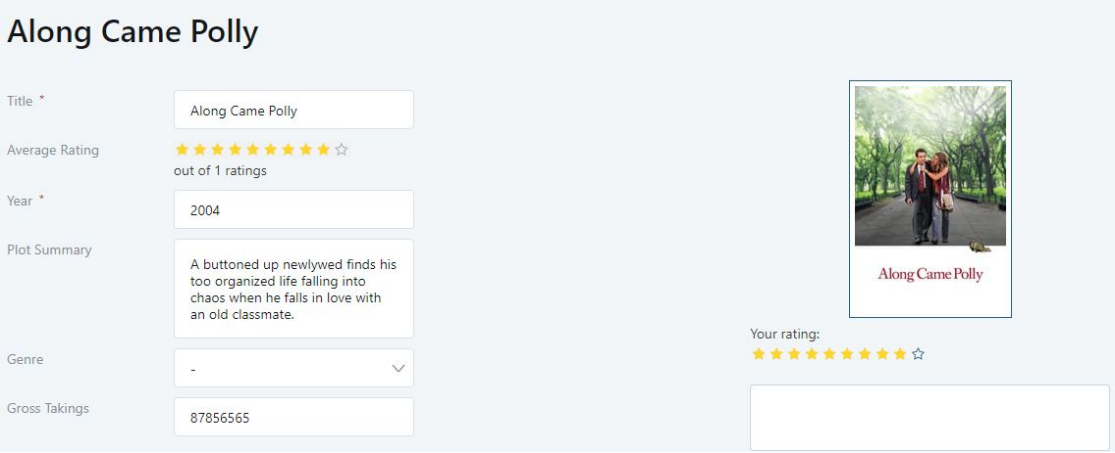
```
[disabled="disabled"], [readonly="readonly"] {
    border: 0px;
}
```



2) Publish and test the look and feel changes made to the OSMDb application.

g) Click the **1-Click Publish** button to publish the application, and access it.

h) Navigate to the **MovieDetail** Screen by selecting an existing movie. If the movie is not rated yet, rate it. Notice that the filled stars in the average rating and user rating are now golden, while the hollowed stars remain the same. The Screen should look like this



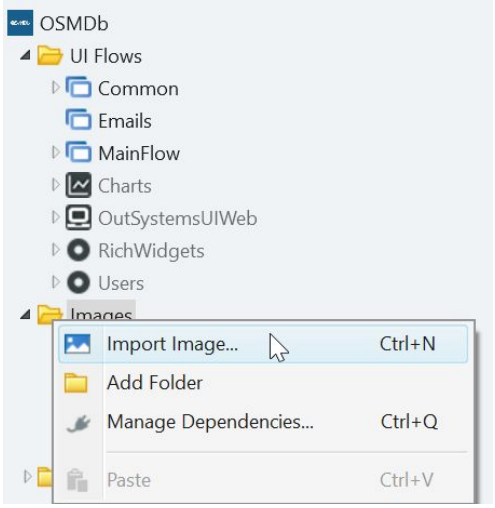
Modify the module's Theme

In this part of the exercise, we will modify the generated Theme. We will start by changing the Header to have a custom background image, and change the look and feel of the Menu items, so that they can be adjusted to look nice with the new image.

Also, we will add a new icon to appear in the application, instead of the name of the app, and a new background image to the main area of all Screens.

1) Load the new Header and Background images as resources to the module. These images can be found in the exercise **Resources** folder.

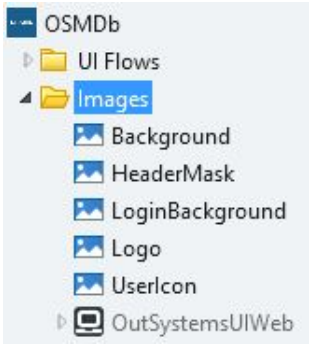
    a) Switch to the **Interface** tab, right-click the **Images** folder and select *Import Image...*



    b) Locate the *HeaderMask.png* file and import it.

    c) Repeat the same steps to import the *Background.png* image file.

    d) The **Images** folder should look like this



2) Change the Header background to use the image that was just uploaded to the module (*HeaderMask.png*). Since this changes the look and feel of the header, adjust the Menu options and the user login information to make sure it looks good with the new image. Change their color to white, without forgetting when an end-user hovers the menu options with the mouse. Otherwise, they may "disappear" in the background. We need to also adapt the tablet view of the application, to make sure that the Menu appears with a color (dark blue) that matches the new Header background.

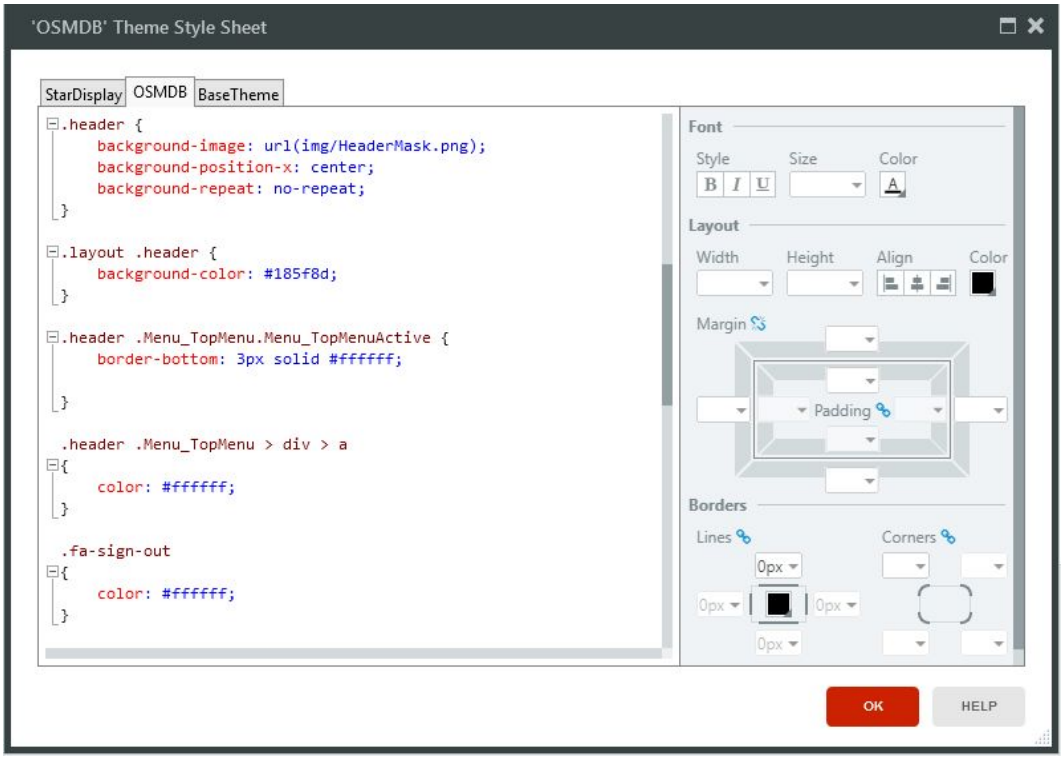
   a) Open the Style Sheet editor, and switch to the **OSMDb** tab.

   NOTE: Since we will modify the module's styles, we can open the Style Sheet editor using any **Style Sheet** property, or via the toolbar. Module Styles are accessible in the context of all Screens and Web Blocks, in the appropriate tab.

   ---

   b) After the existing css code, modify the **header** and **Menu_TopMenuActive** Classes as follows, to improve the look and feel of their background.

```
.header {
    background-image: url(img/HeaderMask.png);
    background-position-x: center;
    background-repeat: no-repeat;
}
.layout .header {
    background-color: #185f8d;

}

.header .Menu_TopMenu.Menu_TopMenuActive {
    border-bottom: 3px solid #ffffff;
}

.header .Menu_TopMenu > div > a{
    color: #ffffff;
}

.fa-sign-out{
    color: #ffffff;

}
```



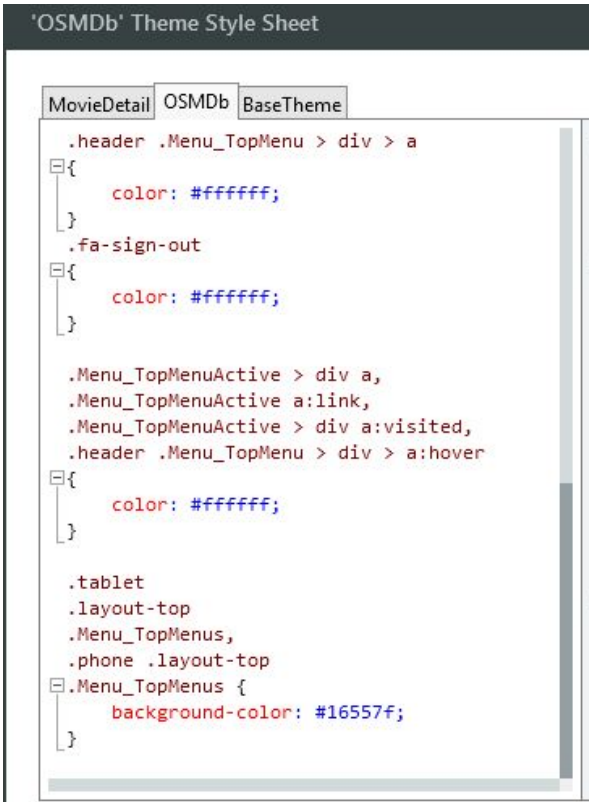
NOTE: When a module is published, image resources are deployed to the **img** folder. In this case, the **HeaderMask.png** image will be available at *http://<your_server>/OSMDb/img/HeaderMask.png*

c) Change the background color of the Top Menu Classes to white, when the menu options are selected and hovered, to make sure they combine well with the previous changes. Otherwise, the menu options would appear faded out, when hovered, with the new background.

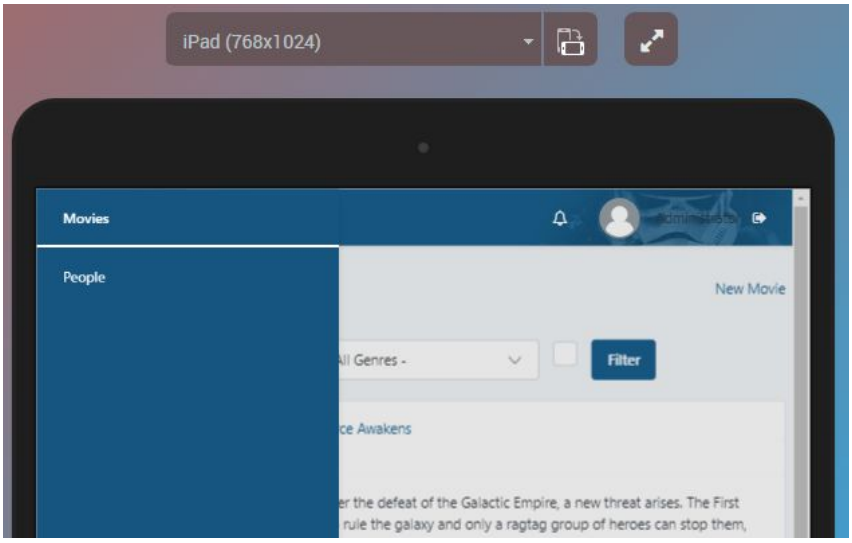
```
.Menu_TopMenuActive > div a,
.Menu_TopMenuActive a:link,
.Menu_TopMenuActive > div a:visited,
.header .Menu_TopMenu > div > a:hover
{
   color: #ffffff;
}
```

d) Set the Tablet Menu Style background color to dark blue (*#16557f)* to match the theme's color.

```
.tablet
.layout-top
.Menu_TopMenus,
.phone .layout-top
.Menu_TopMenus {
    background-color: #16557f;
}
```

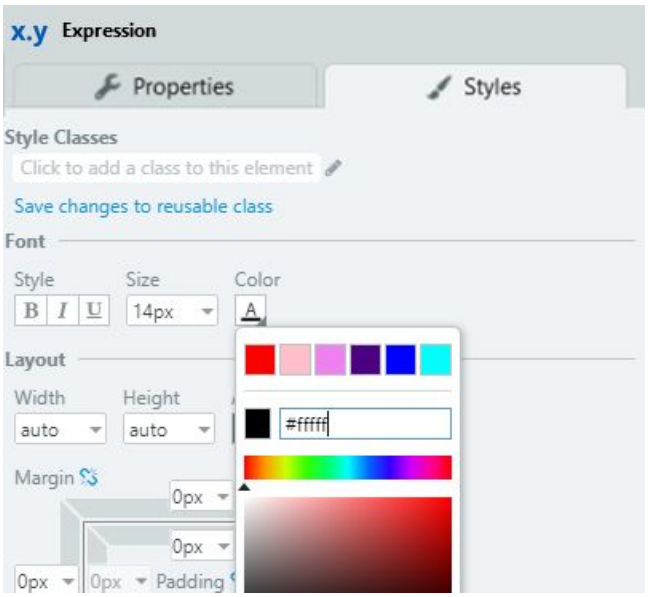


NOTE: Setting the Menu background color to dark blue will allow the menu to also be visible, when viewing the application on a tablet. Publish the module and preview the application with the tablet view to see these changes.



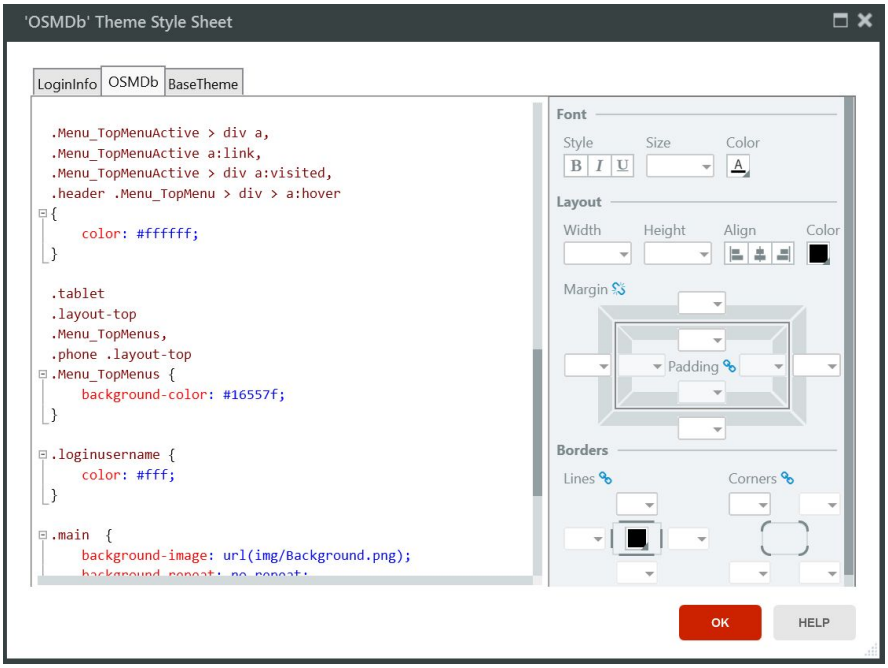
e) Open the **LoginInfo** Web Block from the Common flow and select the expression that contains the user name.

f) Use the **Styles Editor** to update the font color of this expression text to white.

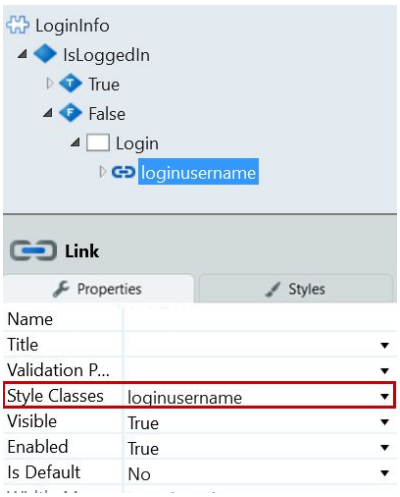


g) Click the *Save changes to reusable class* link and give your new style the name of *loginusername* and click the **OK** button.

This creates a new Style Class loginusername in the OSMDb module Style Sheet, so that this class can be reused throughout the module.



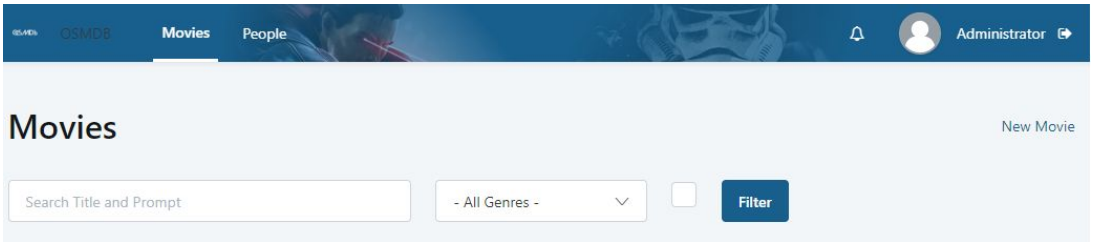
h) Apply the Style Class to the Login link as well, so that it is visible with the new Header background.



3) Publish and test the look and feel changes made to the OSMDb application.

a) Click the **1-Click Publish** button to publish the application, and access it.

i) Navigate through the application Screens. The application Header and Menu should look like this



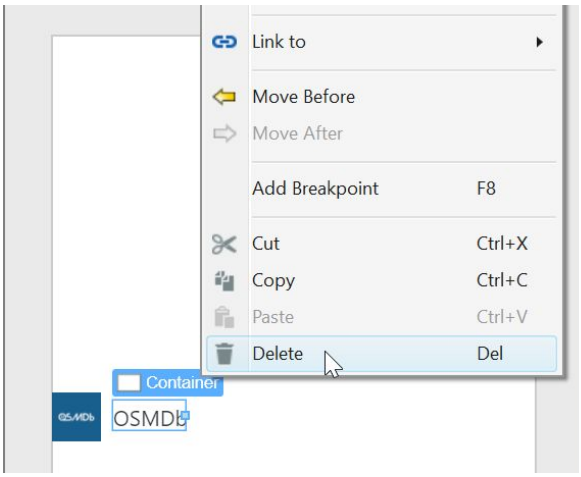
4) Change the application logo to the image *Logo-Transparent.png*, present in the Resources folder of the exercise. Set the width and height to *125px* and *35px* respectively.

a) Open the **ApplicationTitle** Web Block, in the **Common** flow.

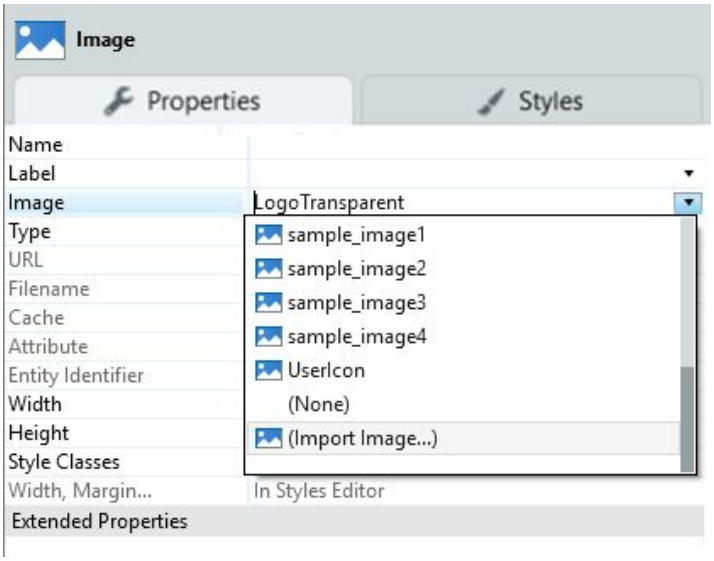
NOTE: If you are in a Screen, you can double-click a Web Block to open it directly. For instance, in the **Movies** Screen, double-click the **Header** Web Block to open it.

---

b) Delete the OSMDb expression text and the container that holds this text.



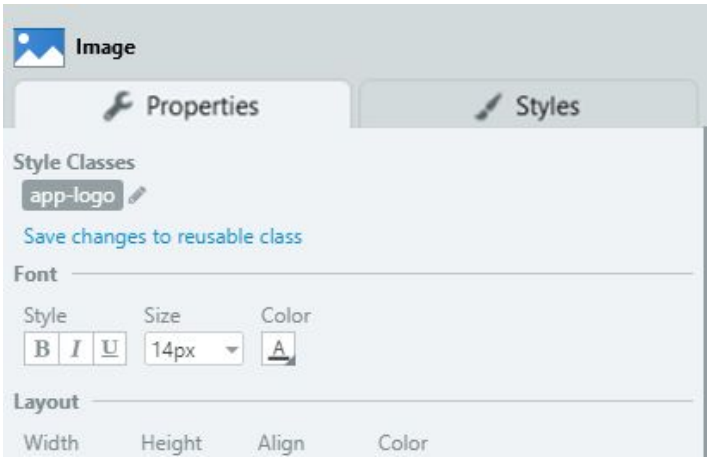
c) Select the OSMDb logo, and in the dropdown for **Image** property select '(Import Image...)'. Choose the logo with the transparent background provided in the Resources folder of the exercise.



NOTE: This is an alternative way to import resources to your module. With this method, the imported image will also be added to the **Images** resources folder under the **Interface** tab.

---

d) Change the width and height of the image to *125px* and *35px*



5) Now we will define a background image for all Screens, using the Background.png image. Also, we will change the label color for Input Widgets to dark blue (*#185f8d)* and add spacing (*20px*) between the **Save** Buttons and **Back** Links.

a) In the module (**OSMDb)** tab of the Style Sheet editor, add the following CSS to the bottom, to change the background image of the **MainContent**.

```
.main  {
    background-image: url(img/Background.png);
    background-repeat: no-repeat;
    background-position: bottom right;
}
```

NOTE: Since we want to define the background image for all Screens, it is important that this Style is defined in the **OSMDb** module tab. Since we are extending an already existing **Style Class,** there is no need to apply this Style to any Widget. Recall that the background image has already been imported.

---

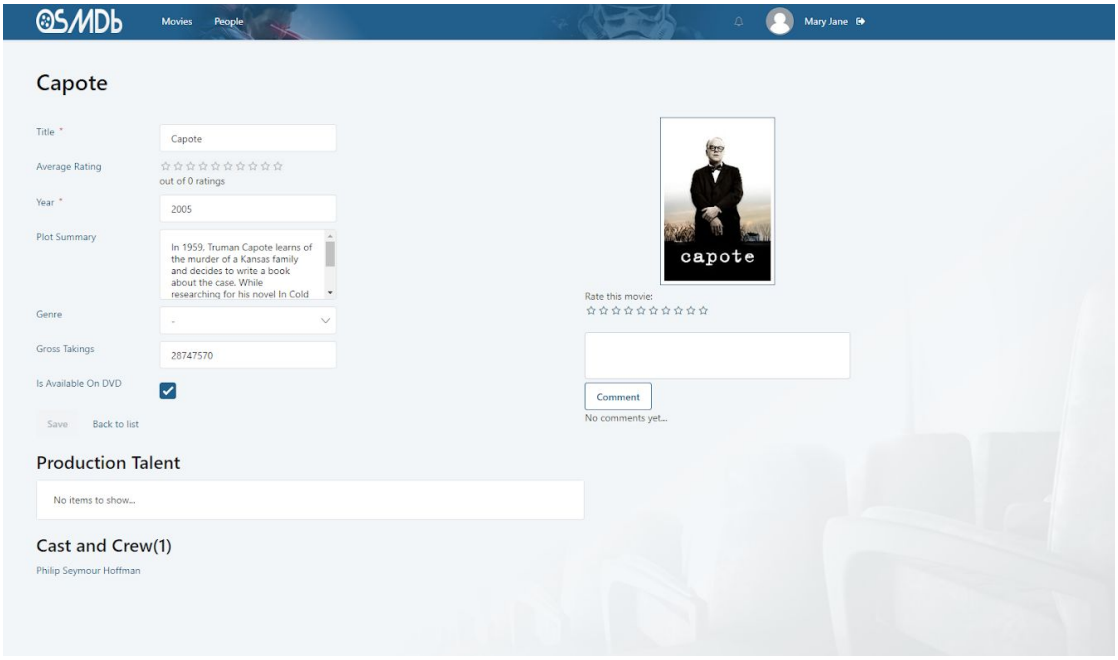
b) After the **.main** Class, add the following CSS Style Classes to change the color of the Form Input Labels and to add a spacing between the Buttons and the Links next to them.

```
.Form label {
    color: #185f8d;
}
input[type="submit"] + a {
    margin-left: 20px;
}
```

6) Publish and test the look and feel changes made to the OSMDb application.

   a) Click the **1-Click Publish** button to publish the application, and access it.

   b) Navigate through the application Screens. If you enter the details of a movie, you should be able to see all changes: the background image, the label colors, and the margin between the Button and the Link. The **MovieDetail** Screen should look like this



End of Lab

In this exercise lab, we used CSS to customize the look and feel of our application. These changes were made at different levels of the module: on a Screen, on a Web Block and on the module's Theme.

First, we created a border to the posters in the MovieDetail. Then, we changed the stars for rating movies to gold, in the respective Web Block. Finally, we added some CSS Styles to the module's Theme, which included some pictures for the background and header.

This lab did not have the objective of teaching CSS. We wanted to show examples of usage of CSS in OutSystems, and how flexible it is to use it.