

DASHBOARD – LAB: HOW TO ADD OVERLAY SECTIONS TO YOUR INFOPATH FORM

Did you know you can make your InfoPath form look & feel like a modern web application?

Want to learn how to add modern web design approaches to your InfoPath form? Here's a cool technique of adding a semi-transparent overlay section to your InfoPath form.

Requirements:

- SharePoint Environment (Online or On-premise)
- InfoPath (2010 or higher)
- [qRules](#)
- [FormsViewer](#)

Total	Purpose	Manager	Status
\$9.00	Taxi Cab	Patrick Halstead	
\$78.00	Hotel expenses	Patrick Halstead	
\$78.00	Morning coffee to stay awake & early calls	Patrick Halstead	

Resource

Manager

Status

Approved

Paid

Rejected

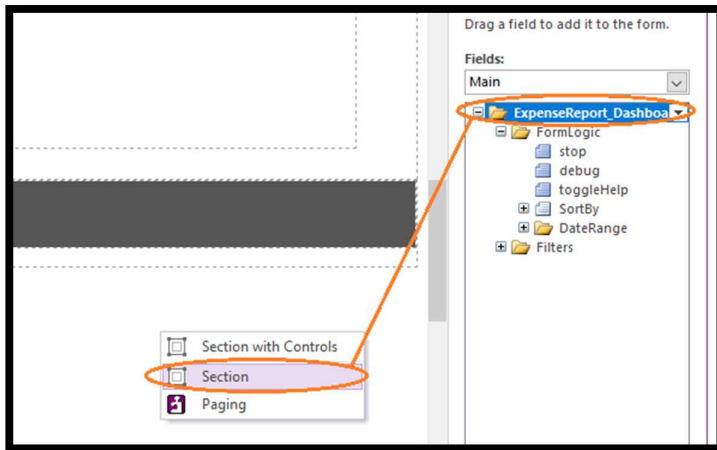
Submitted

ADD THE OVERLAY SECTION

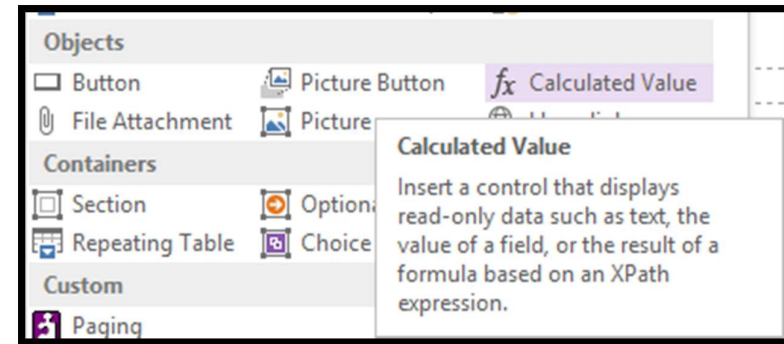
This guide assumes that you will be adding an overlay section to a pre-existing form injected with qRules.

Steps:

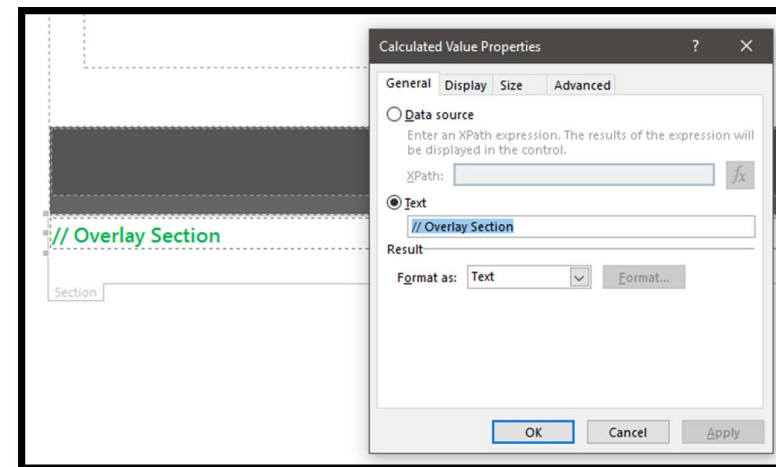
1. Open your form in design mode (right click on the form and click design) – We highly recommend keeping a backup of the form in case something doesn't turn out well, and sometimes it just feels like it's easier to start from scratch rather than fixing the issue, or if you would just want to revert back to an earlier form version.
2. Add the Root Section of your form as a Section to the bottom-most section of your form (right click and drag to canvas)



3. Inside this section, add a Calculated Value

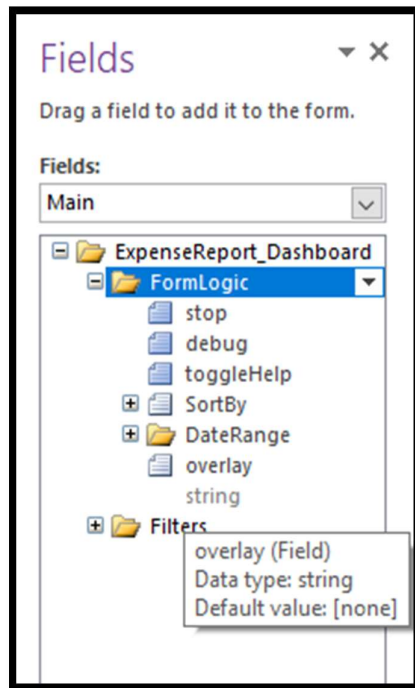


4. Insert the text "// Overlay Section"



This will help us locate the overlay section when we configure the XSL style sheet later.

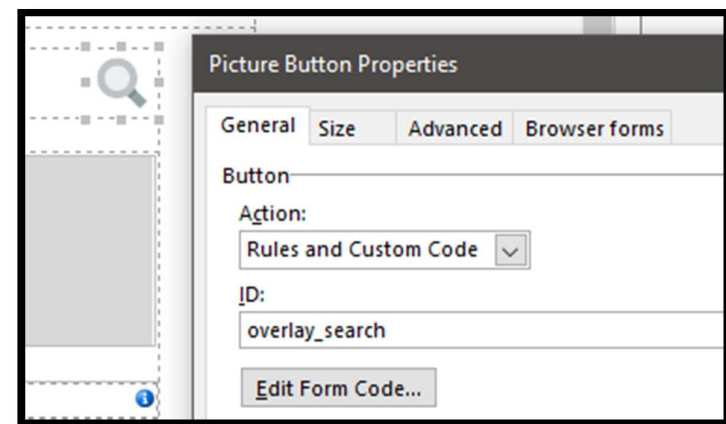
5. Inside the FormLogic group, add a hidden field called “overlay” inside this section



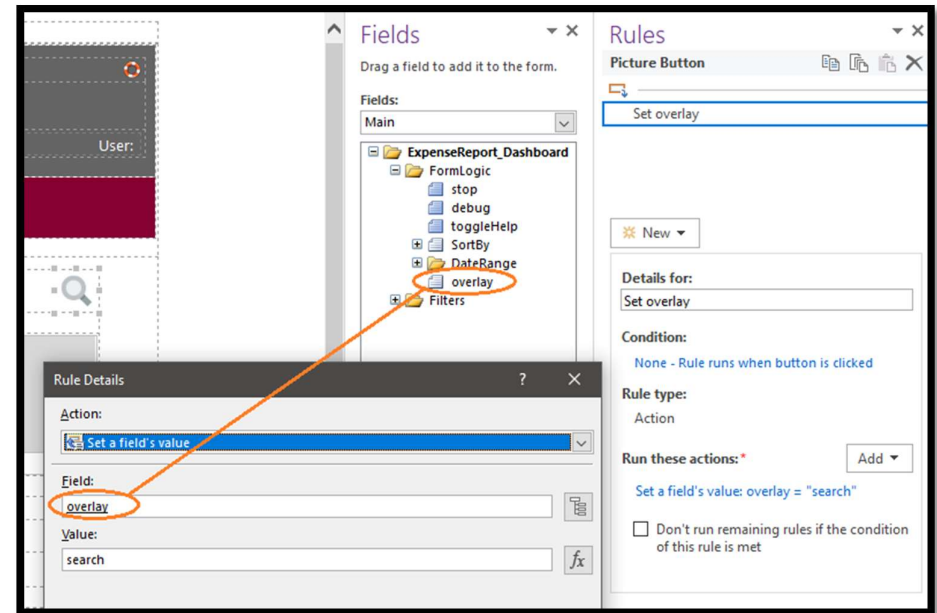
6. Next, add or determine which buttons should trigger the overlay display

For this guide, we'll add a  button for the search section and use the  date range button which was added previously – each button will open up a separate overlay section.

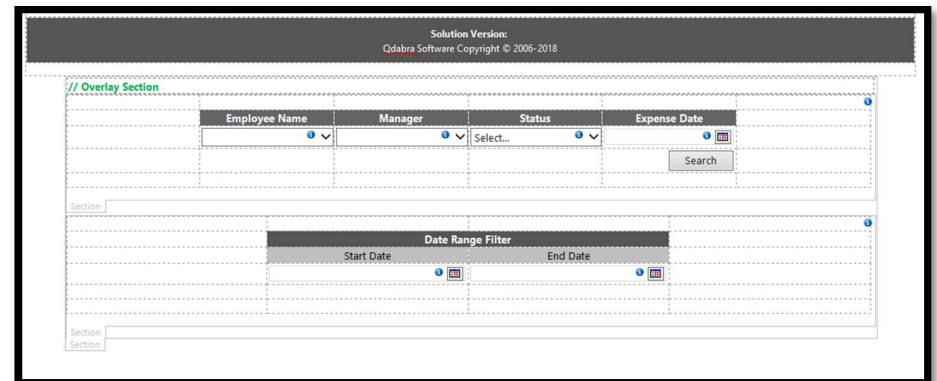
7. On each overlay button, rename the button ID with a meaningful name in the button properties – this will help us locate the buttons later when we edit the xsl stylesheet.



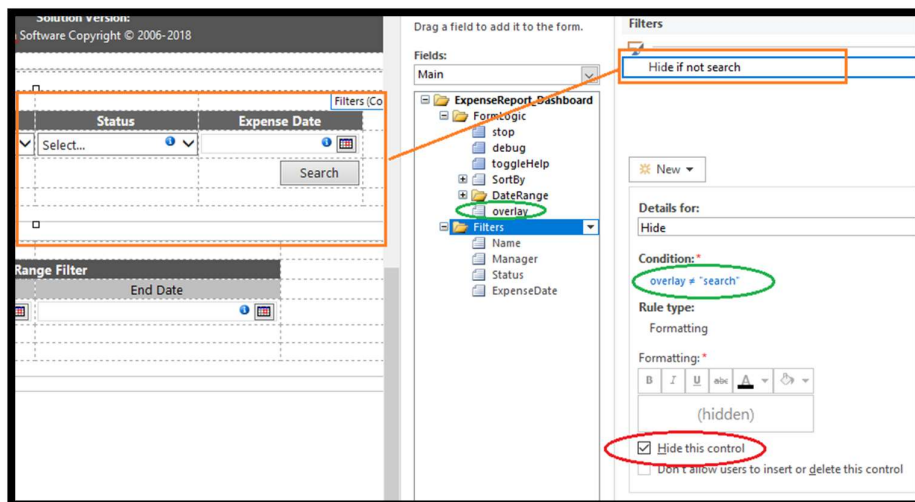
8. Add an action rule for each button to set the overlay field added in step 6 to meaningful or specific values for each overlay section:



9. In the overlay section on the form, drag the Search section and Date Range filter section.



10. Add a formatting rule that hides each section with a condition based off the value of the overlay field.

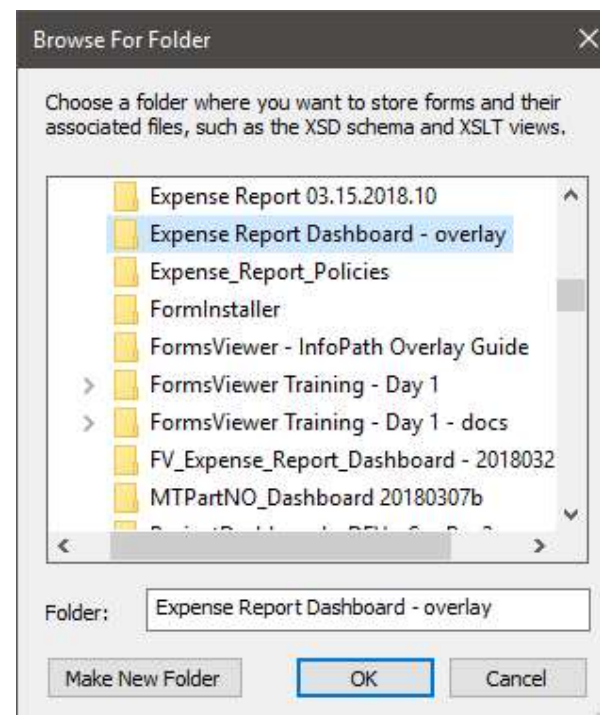


Note that the values being set by each overlay button must correspond to this hide/show section logic.

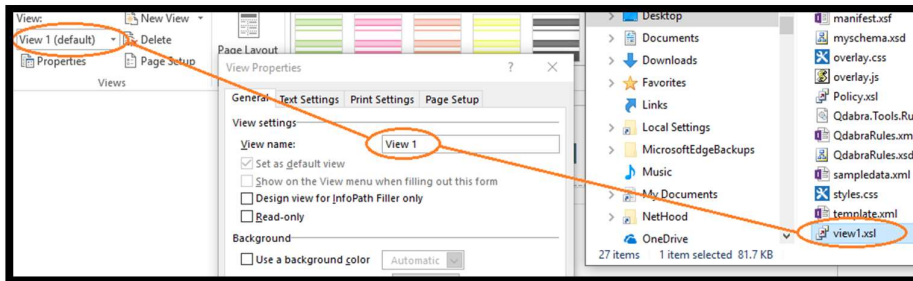
11. When done, Save your form.
12. Next, go to File tab > Publish > Export Source Files



Create a New Folder if you don't have one yet, and export the source files



13. Inside the Source File folder, locate the XSL file for the current view we are editing
– to determine this, at the InfoPath form’s Page Design tab look under View:



The initial view in any IP form is going to be view1.xsl, and most people just rename it w/out thinking.

In case you’ve renamed your view in InfoPath and you cannot find the XSL file which matches the view’s name, then it’s most probably the default view1.xsl which cannot be renamed.

14. Open the XSL file using a text editor (such as note pad)
15. Locate the “// Overlay Section” text created earlier ([step 4](#))



Take note of the mode value.

If you can’t find “// Overlay Section”, you may have opened the wrong XSL file. Check each view XSL until you find the “//Overlay Section” text.

16. Search for the <div> element which contains the mode value from the previous step. In this case, the mode value is _28

```
</tr>
</tbody>
</table>
</div>
<div align="center"><xsl:apply-templates select="." mode="_215"/>
</div>
<div></div>
</body>
</html>
</xsl:template>
```

17. Enclose the <div> element with the following IDs:

```
<div id="overlay">
<div id="overlaysection">
```

If you are unfamiliar with XSL (eXtensible Stylesheet Language), follow these steps:

- a. Align the div element enclosing the xsl apply template

```
</table>
</div>
<div align="center"><xsl:apply-templates select="." mode="_28"/>
</div>
<div><xsl:apply-templates select="my:FormLogic/@my:debug" mode="_23">
```

It should look similar to this:

```
<div align="center"><xsl:apply-templates select="." mode="_28"/></div>
```

- b. Add the <div> element IDs above the xsl apply template:

```
<div id="overlay">
<div id="overlaysection">
<div align="center"><xsl:apply-templates select="." mode="_28"/></div>
</div>
</div>
```

- c. Match each <div> element ID with a closing </div> tag below (see image above)

18. Locate the buttons which should trigger the overlay display (buttons from [step 6](#))

```
<td style="BORDER-BOTTOM-COLOR: ">
<div align="right"><button title="" class="xdPictureButton" contentEditable
="false" style="BORDER-TOP: medium none; HEIGHT: 32px; BORDER-RIGHT:
medium none; WIDTH: 32px; BORDER-BOTTOM: medium none; BORDER-LEFT: medium
none; MARGIN: 1px" xd:HideInPrintView="true" xd:CtrlId="overlay_search"
xd:xctname="PictureButton" tabIndex="0">
<img style="HEIGHT: 32px; WIDTH: 100%; POSITION: static" alt="" src
="btn_search.png" xd:HoverSRC=""/>
</button>
</div>
</td>
```

19. At the button element, insert an onclick event: onclick="on()"

```
<button onclick="on()" title="" class="xdPictureButton" contentE
```

Note: If you are adding an onclick event to a button, you might get confused by the type of element used. Here's how a Picture Button and a Regular InfoPath Button tags differ in the xsl:

```
Picture Button
<button title="" class="xdPictureButton" contentEditable="false" style="BORDER-TOP: medium none;
BORDER-LEFT: medium none; MARGIN: 1px" xd:CtrlId="overlay_pic_btn" xd:xctname="PictureButton"

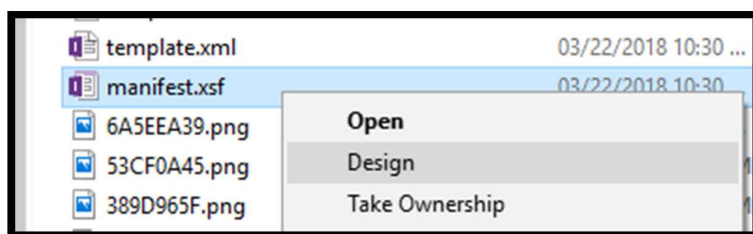

Regular Button
<input title="" class="langFont" style="MARGIN: 1px" type="button" value="Button" xd:CtrlId="o
xdBehavior_Select" size="1" tabIndex="0" xd:binding="my:field2" xd:CtrlId="CTRL4" xd:xctname="g
```

Add the onclick event to the other buttons – right before the title attribute.

20. Save the XLS stylesheet.

21. Locate the manifest.xsf file within the resource folder

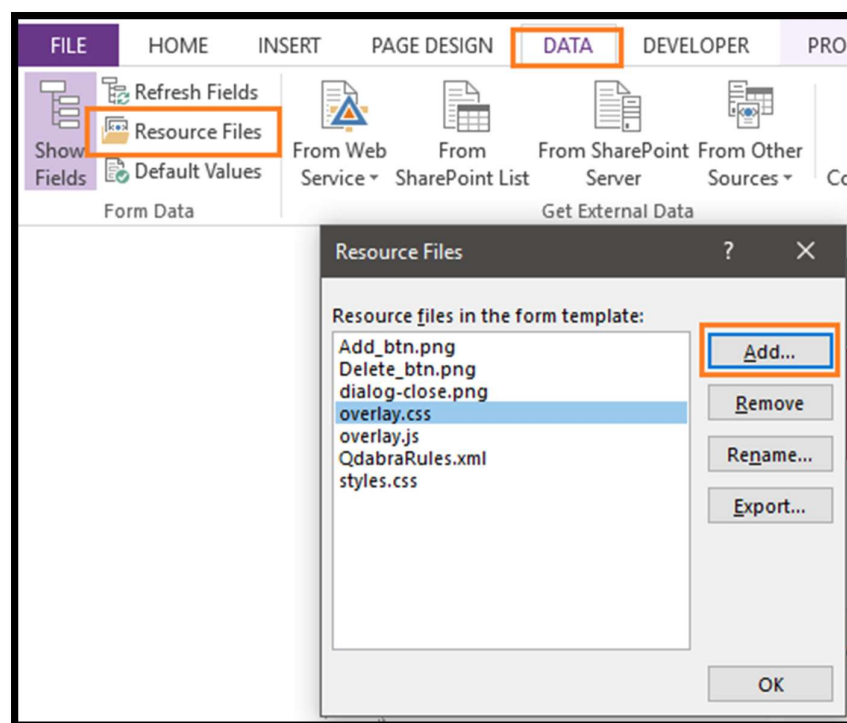
22. Right click > open in Design mode



23. Add the overlay.css and overlay.js files (included in the package) as resource files:

 overlay.css	03/22/2018 9:57 PM	CSS Document
 overlay.js	03/22/2018 9:57 PM	JavaScript File

Under the Data tab, click the Resource Files icon and add the files mentioned above:



24. Click OK to exit the Resource Files window
25. Load the overlay files on form load using the qRules command LoadResource

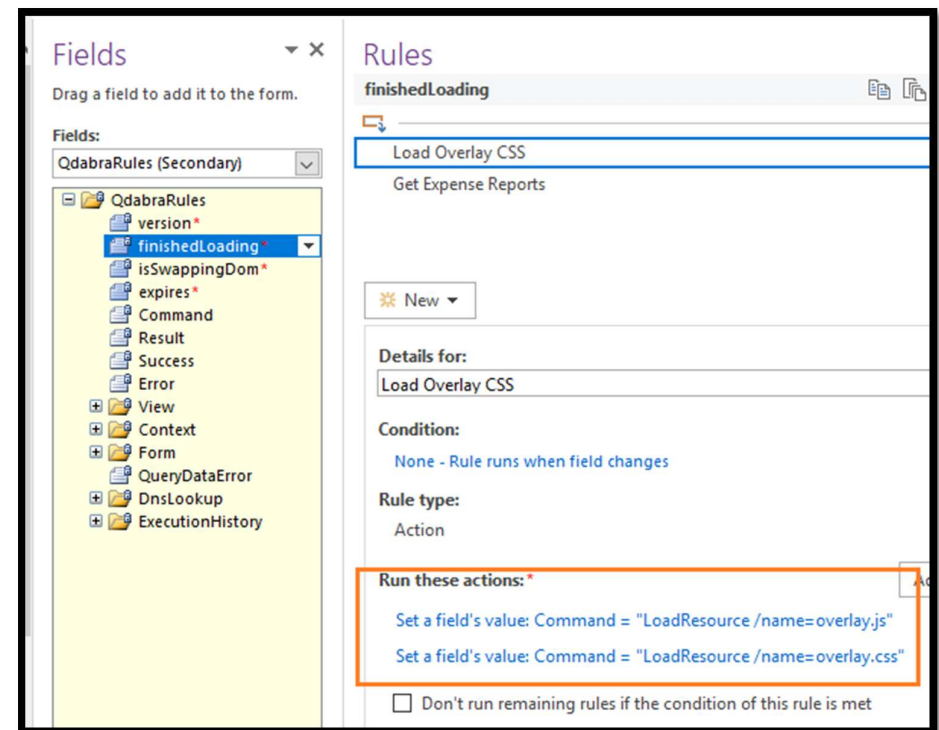
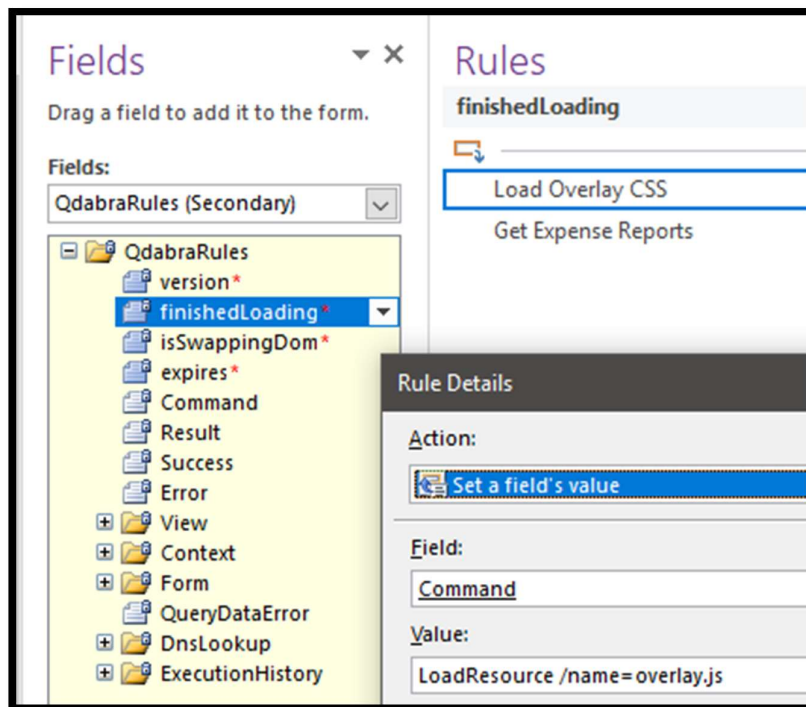
- a. On the finishedLoading node of the QdabraRules data connection, add a rule that sets the Command node to the value:

LoadResource /name=overlay.js

- b. Add another action to load the other overlay file:

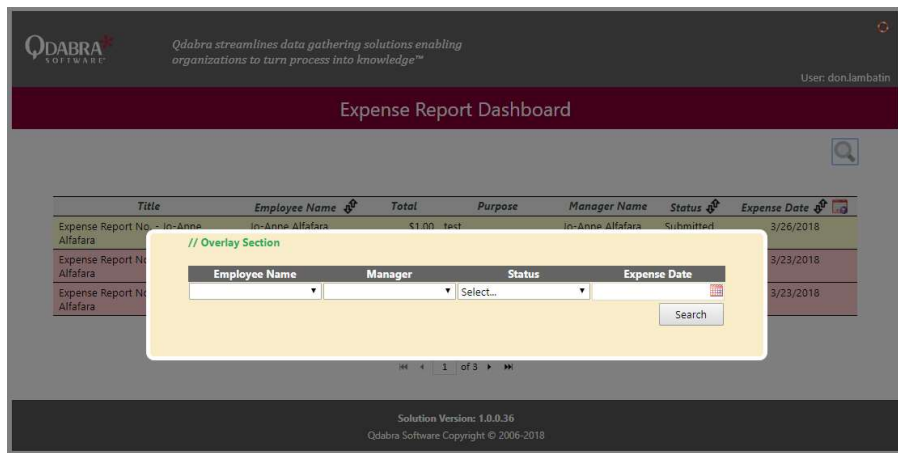
LoadResource /name=overlay.css

Your set of actions should look like this:



26. Save your form and upload to FormsViewer.

Test your form in FormsViewer and verify the buttons open each overlay section properly:

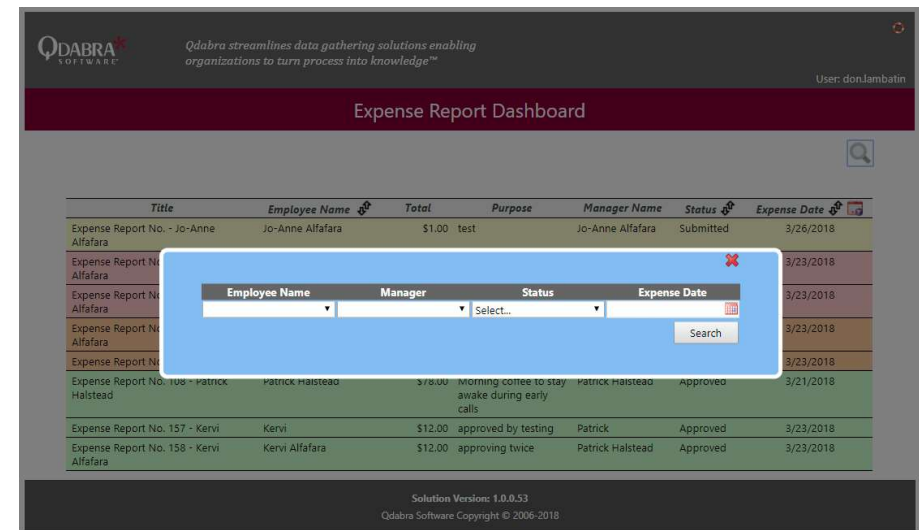


27. Back in the form's overlay section, Add a close dialog button  at the upper right most section, underneath the text // Overlay section.
28. Re-name the close dialog  button's control IDs (see step 8 for reference)
29. Save form and export source files (see [step 12](#) for reference)
30. Locate the close dialog buttons in the XSL stylesheet (see [step 18](#) for reference)

31. Insert an onclick event which closes the overlay section: onclick="off()"

```
align="right">
<font size="2" face="Calibri"><button onclick="off()" title="" class="
WIDTH: 16px; BORDER-BOTTOM: medium none; BORDER-LEFT: medium none; MAR
<img style="HEIGHT: 16px; WIDTH: 100%; POSITION: static" alt="
</button>
```

32. Save the XLS stylesheet.
33. Locate the manifest.xsf file within the resource folder
34. Right click > open in Design mode
35. Save form, re-upload to FormsViewer, and test your form:

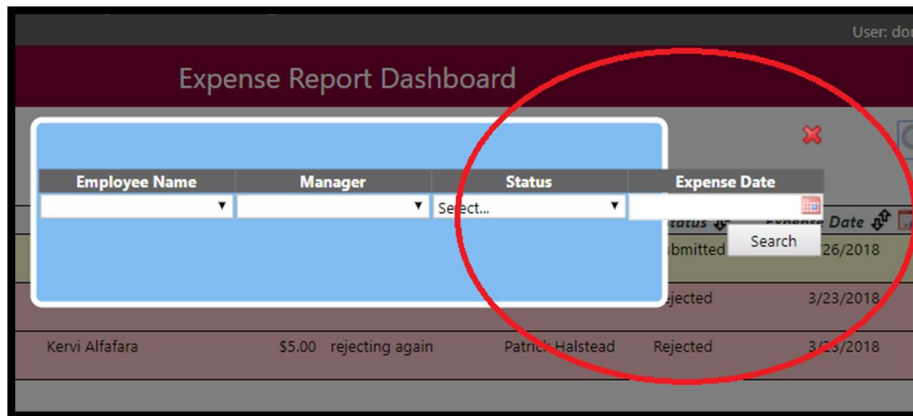


Optionally, you may choose to edit the overlay layout by editing the overlay.css stylesheet.

The first section is for the configuration of the [overlay](#) background – you can control the transparency here.

The second section is for the [overlay section](#) container – you can change the size, shape, and background color to your desire.

Adjust the section's width to suit your form's measurements and the background color to your desire:



```

/* Overlay background */
#overlay {
    position: fixed;
    display: none;
    width: 100%;
    height: 100%;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background-color: rgba(0,0,0,0.5);
    z-index: 2;
}

/* Overlay container section */
#overlaysection{
    position: absolute;
    top: 40%;
    left: 50%;
    width: 810px;
    background: #81bdf4;
    transform: translate(-50%,-50%);
    -ms-transform: translate(-50%,-50%);

    /* rounded corners */
    -moz-border-radius: 10px;
    -webkit-border-radius: 10px;
    border: 5px solid #ffffff;
    
```

CONTACT INFORMATION

Qdabra Software
218 Main Street,
Suite 731,
Kirkland, WA 98033

Phone: 877.544.2389
Email: Support@Qdabra.com
Website: <http://www.qdabra.com>
Community: <http://www.infopathdev.com>