IBM Cloud Pak for Business Automation Demos and Labs 2021

Introduction to IBM Business Automation Application

Swapnil Agrawal
aswapnil@ca.ibm.com

V 1.0

NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Table of Contents

1 Introduction

1.1 IBM Business Automation Application

IBM Business Automation Application provides a way to create apps that automate repetitive and time-consuming work by quickly building user interfaces that integrate tasks, data and automations to drive efficiency across your business. Using a low-code application builder, IBM Business Automation Application Designer, users can create business applications (apps) that leverage the capabilities of the IBM Cloud Pak for Business Automation platform. The authored apps are deployed in the IBM Business Automation Application Engine and end users work with them in IBM Business Automation Navigator.

Additional information about IBM Business Automation Application can be found here.

1.2 Lab Overview

There are two types of users who use the Application Designer – Technical users and Business users. Technical users often create templates that are predefined starting points for business users to create their own applications. Templates can represent common-use case patterns and contain various artifacts that can be reused by business users in their applications. Applications & templates also contain toolkits that contain a collection of shared artifacts. These toolkits can be created by the technical user (e.g., a developer or a business partner), or contributed by another capability (e.g., Content or Workflow).

In this lab, you will perform the roles of both a technical user and a business user to create a toolkit, template and application that is a part of the overall Client Onboarding end-to-end solution.

Approximate Duration: 2 hours

1.3 Lab Setup Instructions

1. If you are performing this lab as a part of an IBM event, access the document that lists the available systems and URLs along with login instructions. For this lab, you will need to access **IBM Business Automation Studio**.
2. Download the **Focus-Corp-Logo.png** file from the **Lab Data** folder onto your computer.

2 Exercise: Creating the Client Onboarding toolkit

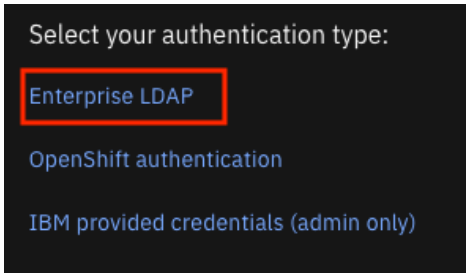
2.1 Introduction

In this exercise, we will create a reusable toolkit as a technical user. The toolkit will contain two reusable views that can be used in later exercises to create the **Client Onboarding** application.

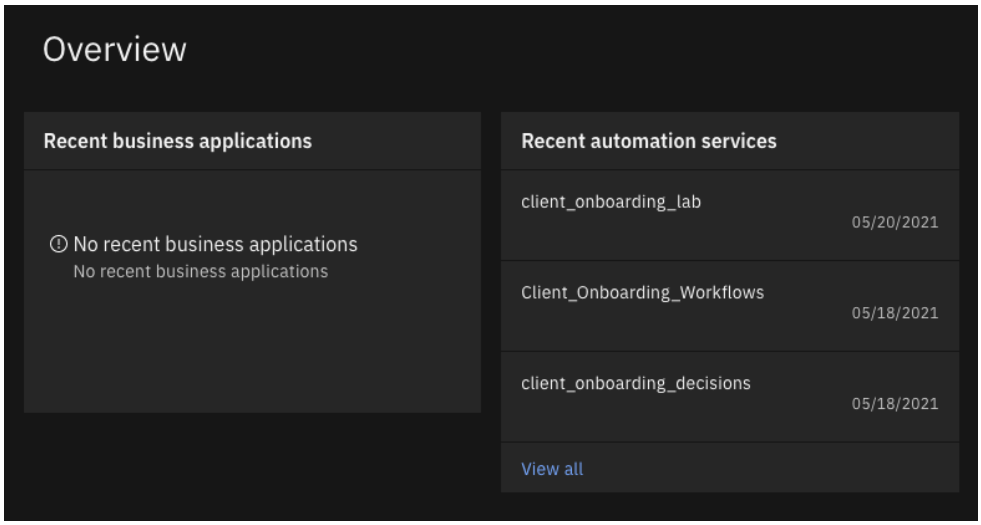
2.2 Exercise Instructions

2.2.1 Creating a toolkit

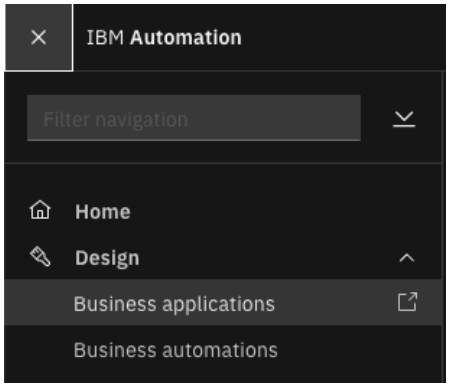
1. In your browser, login to IBM Business Automation Studio using the Enterprise LDAP option.



The homepage contains cards that showcase recent artifacts across all installed Cloud Paks in the system. For IBM Cloud Pak for Business Automation, the recent business applications and automation services are shown.

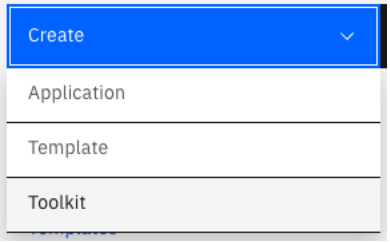


2. In the top-left corner, click on the menu icon and select **Design → Business applications** to access the application repository.

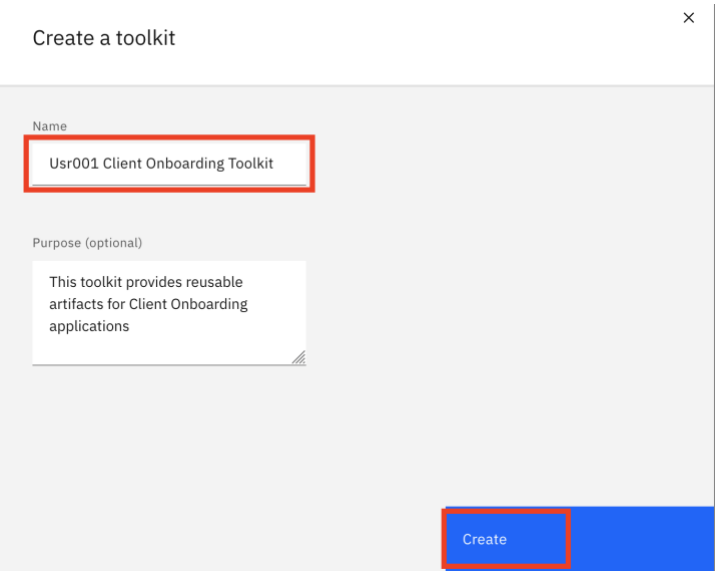


We will first create a toolkit that contains reusable Views (UI elements) and Actions (Services). These artifacts can also be added to the template directly but adding them to a toolkit ensures that they can be reused by multiple other toolkits and templates.

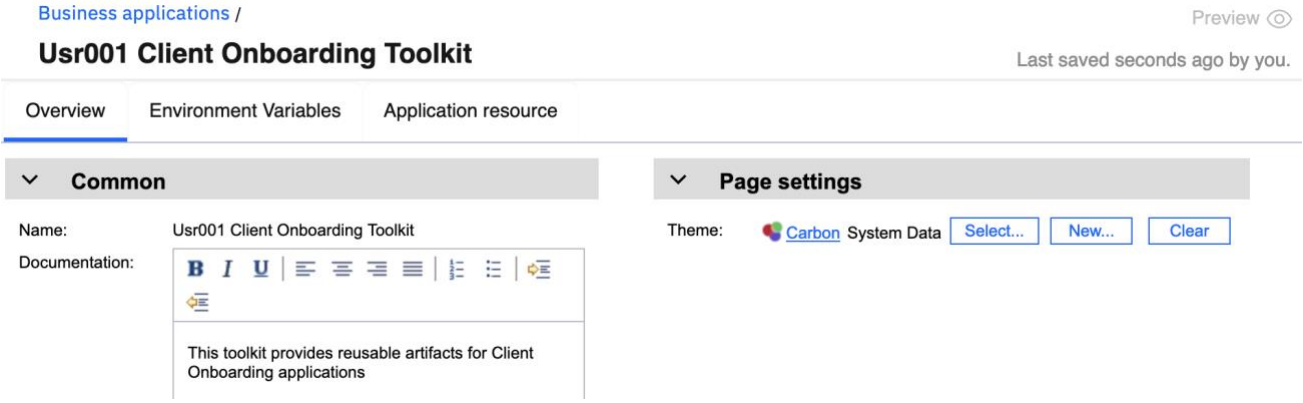
3. Click on **Create → Toolkit**.



4. For the **Toolkit name**, enter ***UsrNNN* Client Onboarding Toolkit** where *UserNNN* is your assigned username.
5. Provide an optional **purpose**.
6. Click on **Create**.

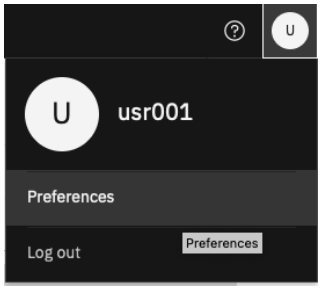


This will open the Application Designer Toolkit editor in a **Basic** view mode. The capabilities in the Basic view mode are scoped down for ease of use for business users.

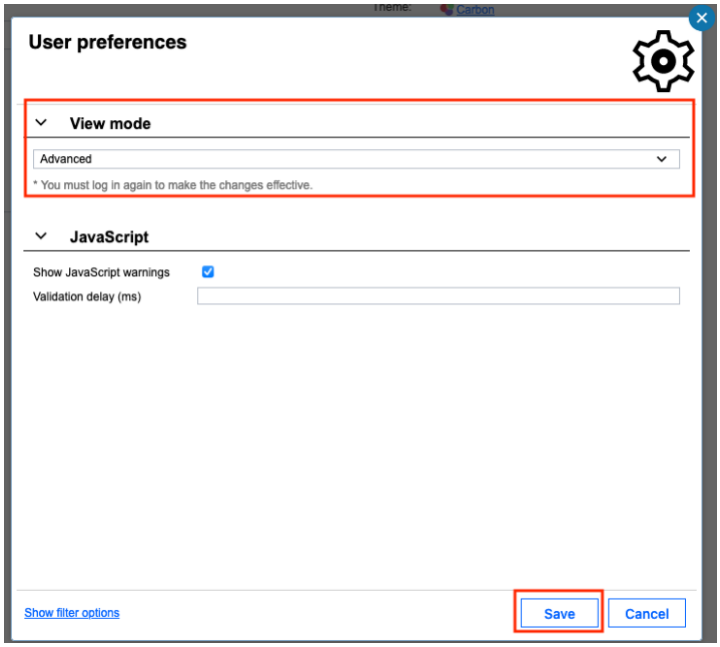


As we are now emulating a technical user, we will switch to the **Advanced** view mode.

7. In the top-right corner, click on the **user icon** and select **Preferences**.



8. In the **View mode** dropdown, select **Advanced**.
9. Click on **Save**.



10. For the changes to be effective, **refresh** the browser window.

2.2.2 Creating reusable views

Now, notice that a library pane is available on the left-hand side, and this can be used to create new artifacts and add other toolkits as dependencies.

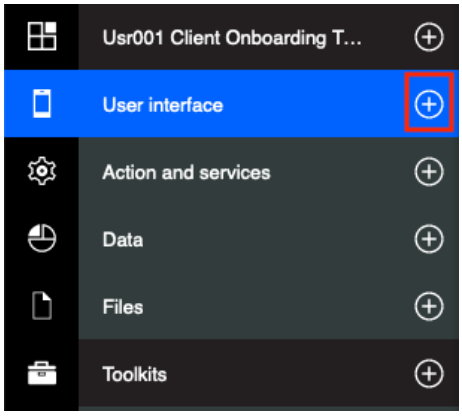
In this toolkit, we will add two views:

- A **Responsive Columns** view that contains two columns. It will show the two columns side-by-side on medium/large screens (e.g., tablets, desktops) but when the app is accessed on a small screen (e.g., mobile), the second column will appear below the first one. Users can then add other views within this view to create a responsive application.
- A **Custom Header** view that can be used by all templates & applications to display containing the logo of **Focus Corp**.

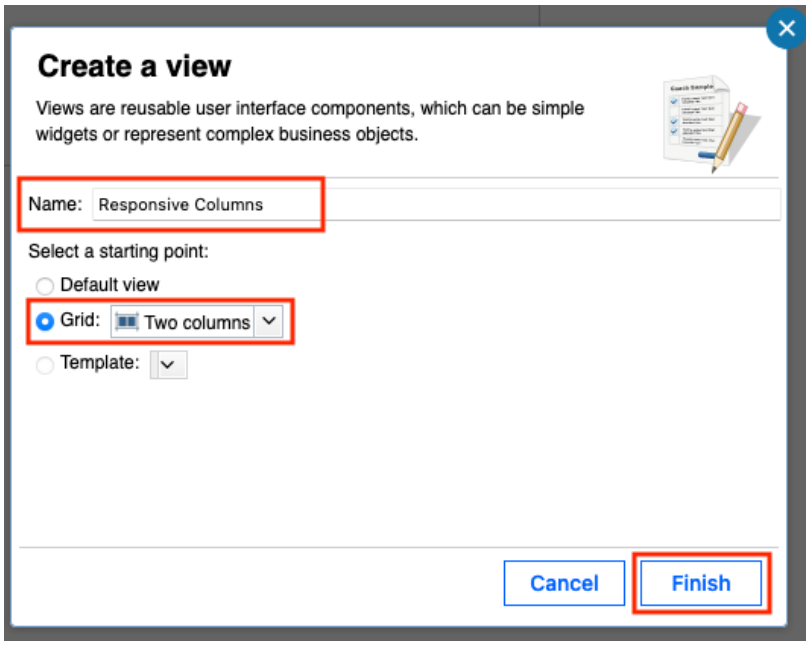
1. **Expand** the library pane by clicking on the **Library** icon in the bottom-left corner.

This shows the library menu which lists the different categories of artifacts you can create. The library pane also expands automatically if you click on one of the items in the collapsed panel.

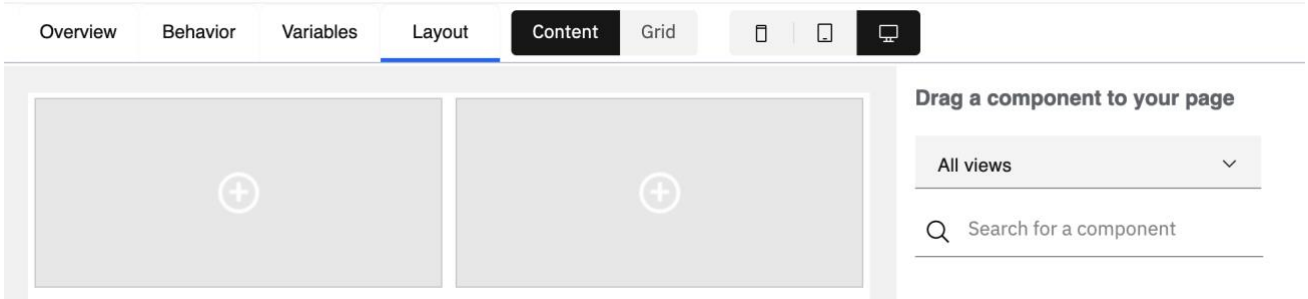
2. In the library pane, click on the **+** button next to **User interface**.



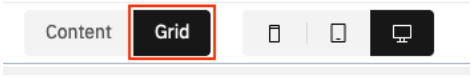
3. Select the **View** artifact.
4. In the **Name** field, enter **Responsive Columns**.
5. In the **Select a starting point** field, select **Grid** and choose the **Two Columns** option from the dropdown.
6. Click on **Finish**.



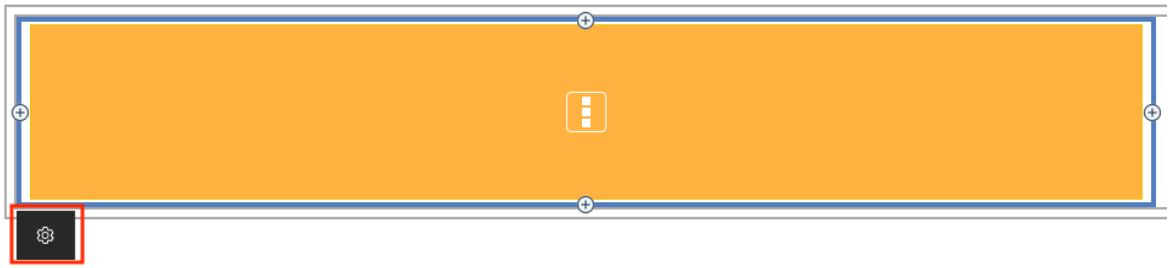
This opens the **View** editor with a grid of two columns.



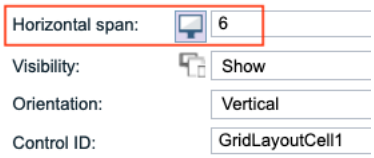
7. Click on **Grid** in the top toolbar.



8. Click on the yellow portion of one of the columns and click on the **Properties** icon to open the properties of that column.

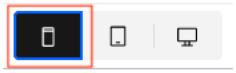


In the Properties pane, observe that the value in the **Horizontal span** is field is **6**.



A grid is a container that is **12 units** wide. This means that the column with a span of **6** will take half the width of the grid.

9. Click on **Cancel**.
10. In the top toolbar, select the **Small screen** option.

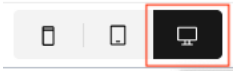


Observe how the editor now contains a mock mobile view to emulate a small screen and the columns automatically react to the change in screen size.

11. Open the **Properties** view of one of the columns**.**

You can see that the **Horizontal span** is now set to **12** which means that it will span the entire width of the container. The icon besides the value also displays a mobile icon showing that the value is set for mobiles only. Users can create complex applications that are responsive by using grids and modifying the horizontal span based on the design requirements of that application.

12. Switch back to the **Large screen** editor.

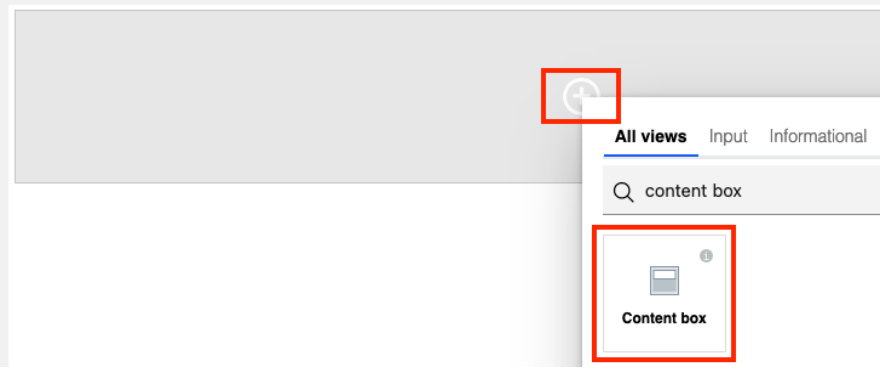


Next, we need to add the ability for the **Responsive Columns** view to contain other views when reused in an application. To do this, we need to add a **Content Box** view inside each column.

13. Click on **Content** in the top toolbar.

14. Click on **+** in one of the columns.

15. Select **Content Box** as the view to add to the column.

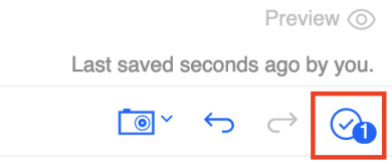


16. Repeat the previous step for the 2nd column. Note that you can also add the **Content Box** view by dragging and dropping it form the right-hand side palette onto the column.

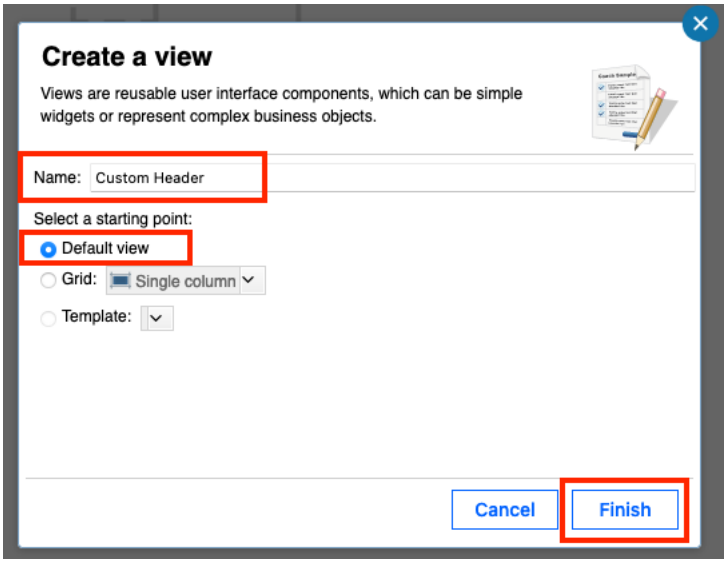
    Your layout should now look like the screenshot below:

    Next, we will add a **Custom Header** view.

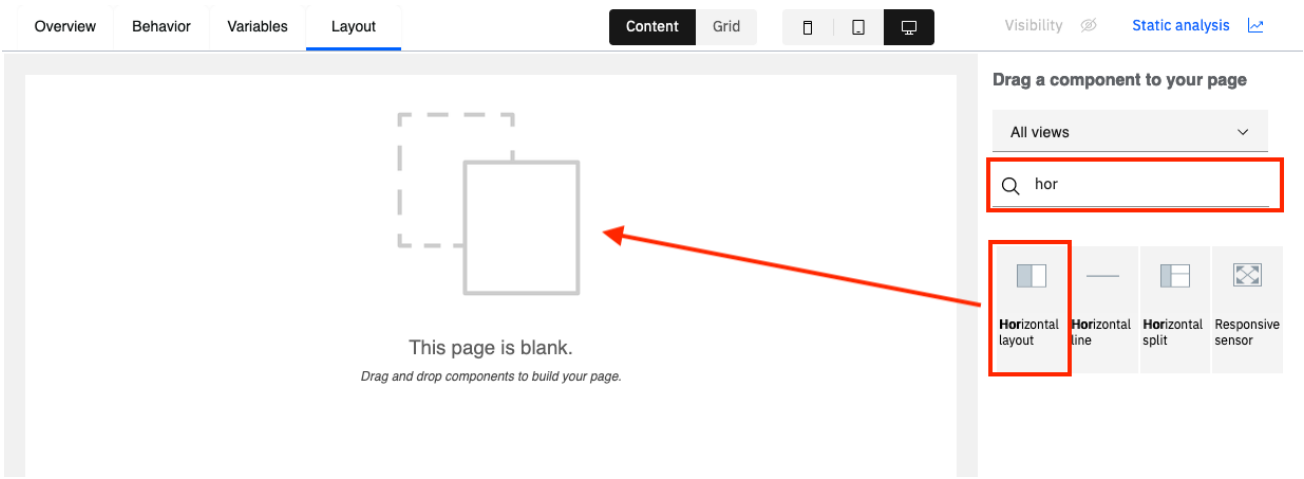
    Note: Your changes are automatically saved when you perform certain actions (e.g., preview an application or close an artifact editor). If at any point, you want to force a save, you can click on the **Finish Editing** button in the upper-right corner.



17. Create another view called **Custom Header**. During the creation of this view, the starting point selected can be the **Default view**.



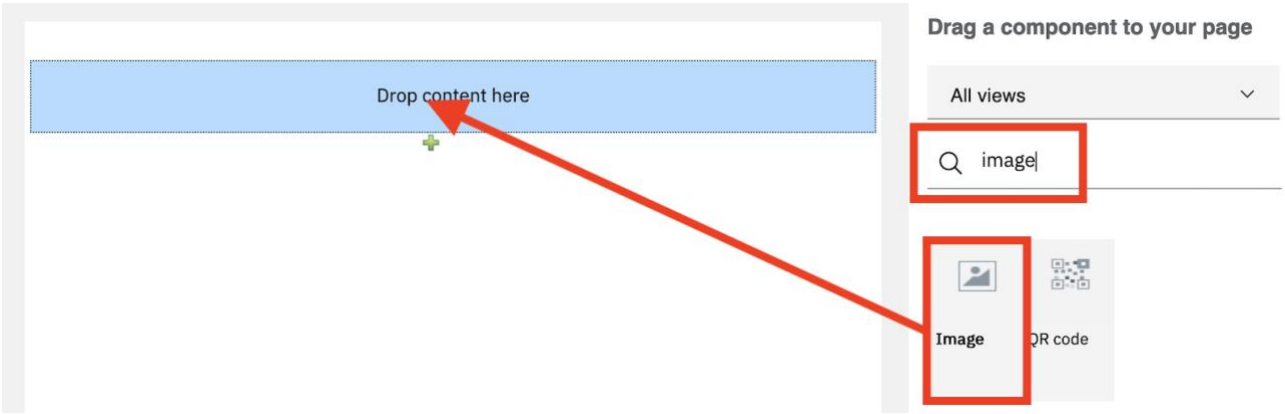
18. In the component palette on the right-hand side, enter **hor** in the **search** field.
19. Drag and drop the **Horizontal Layout** view onto the editor.



As the name suggests, the horizontal view can be used to house other views inside it horizontally.

Note: Hovering over the **info** icon on the view in the palette gives you more information about each view.

20. Next, look up the **Image** view by searching for **image** and drag and drop the view onto the just added horizontal layout.

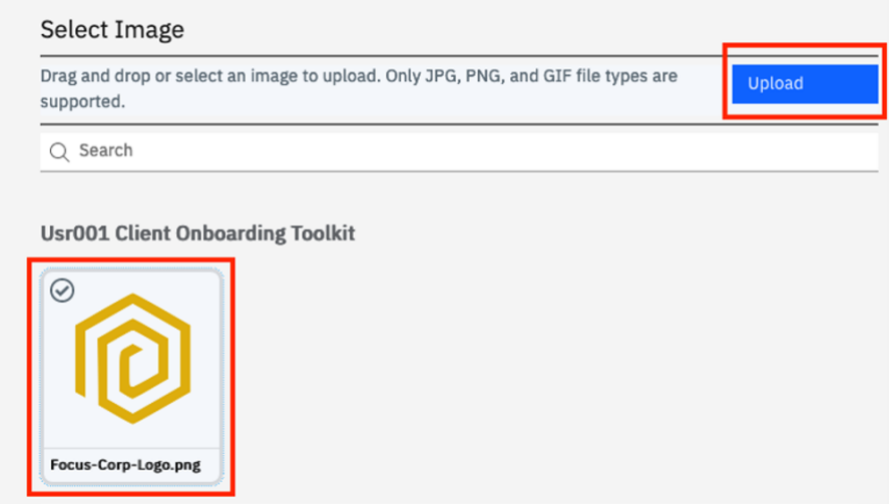


21. Select the **Image** view in the editor and click on the **Select image** icon.

This opens the editor to select and upload new images.

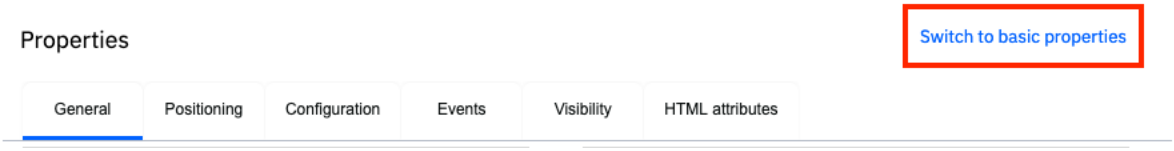
22. Click on **Upload**.
23. Upload the **Focus-Corp-Logo.png** file that you previously downloaded as a part of the lab setup.

24. Select the uploaded file **Focus-Corp-Logo.png** in the dialog.



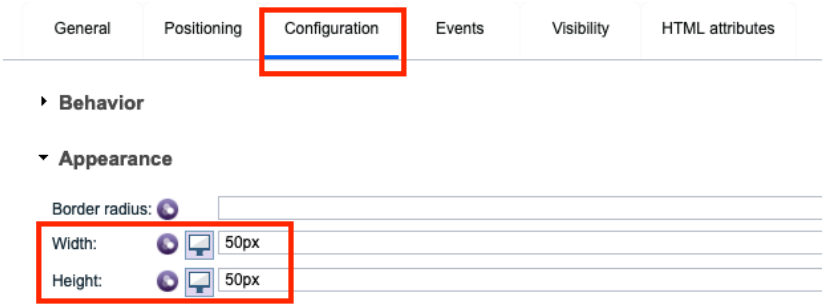
25. Click on **Select** to close the dialog.
26. To edit the size of the image, select the image and click on the **Properties** icon.

27. If the **Properties** pane shows multiple tabs such as **General, Positioning** and so on, click on **Switch to basic properties**.



This opens the user-friendly property editor that is specific to the **Image** view. Most out-of-the-box editors come with basic and advanced property editors to provide functionality to both business and technical users. Observe the design of the basic editor. Since we are acting as a technical user, we will switch back to the advanced properties editor.

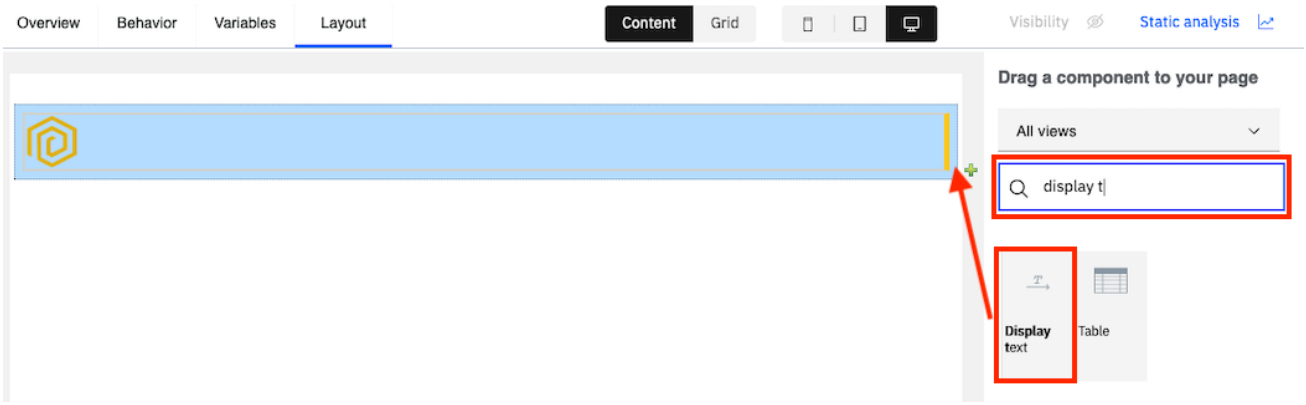
28. Click on **Switch to advanced properties**.
29. Click on the **Configuration** tab.
30. Expand the **Appearance** section.
31. For the **Width** and **Height** fields, enter **50px** as the values.



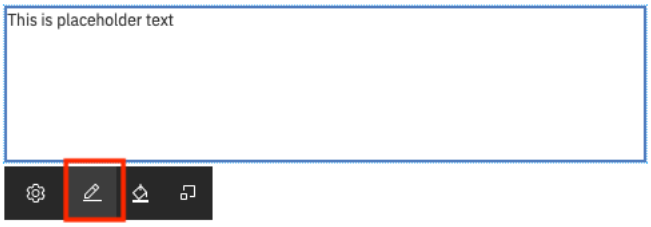
32. Click on **Done** to close the properties pane.

    Observe that the image size now reflects the specified width and height.

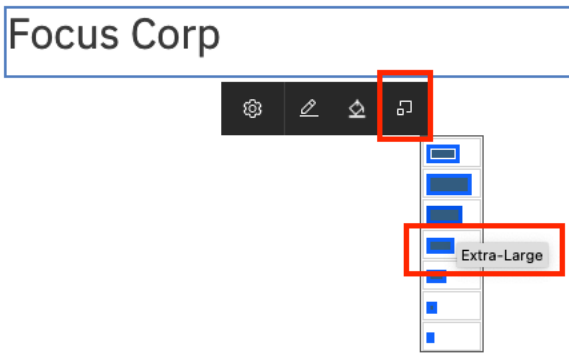
33. Add the **Display Text** view from the palette to the right of the logo within the horizontal layout.



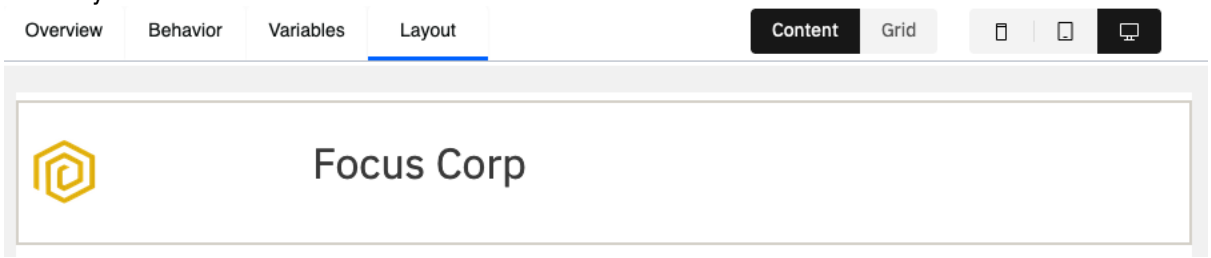
34. Type **Focus Corp** where it says **This is placeholder text**. If the text is not editable, click on the **Edit static text** button to edit it.



35. Click on the **Select size** icon on the displayed text and select the **Extra large** size to make the font size bigger.

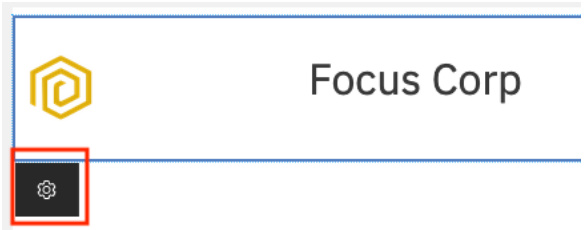


Your layout should now look as follows:

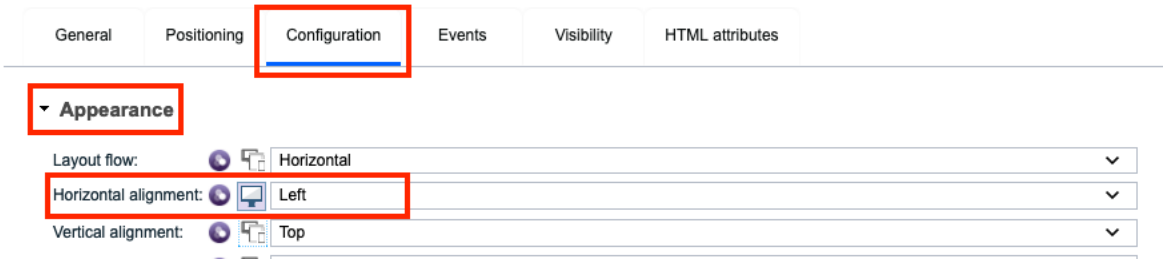


Next, we need to make the text appear closer to the logo. To do this, we will align the UI within the horizontal layout to the left.

36. Click on the **Properties** icon for the horizontal layout.

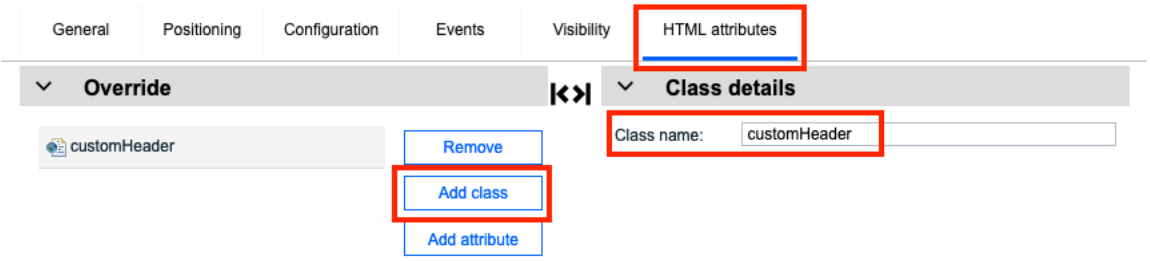


37. Click on the **Configuration** tab.
38. In the **Appearances** section, select **Left** as the value for the **Horizontal alignment** field.



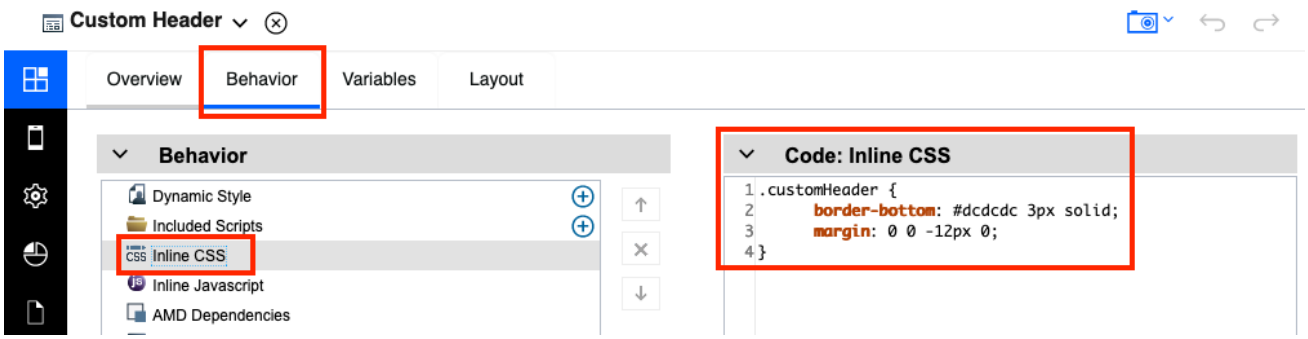
Next, we will see how users can add custom CSS classes and define the CSS for a custom view.

39. Click on the **HTML attributes** tab.
40. Click on **Add class**.
41. Enter **customHeader** as the **Class name**.



42. Click on **Done** to close the properties pane.
43. Click on the **Behavior** tab in the top bar.
44. Click on **Inline CSS** and enter the following CSS code:

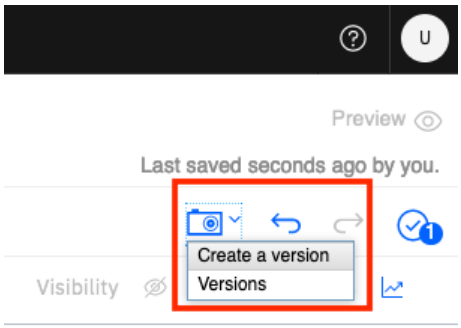
```
.customHeader {
  border-bottom: #dcdcdc 3px solid;
  margin: 0 0 -12px 0;
}
```



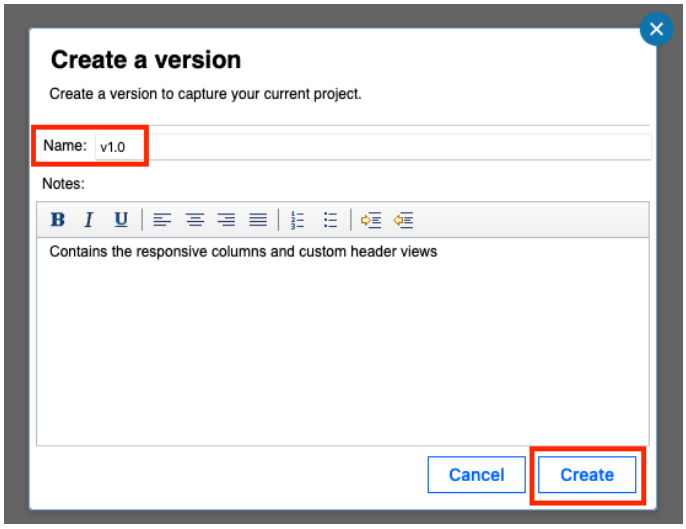
This will add a border to the bottom of the header and customize the margins. This is not required but just done as an example to show that even though the Application Designer is built for low-code, technical users can create highly customized views based on their design requirements.

Next, we will create a version of this toolkit so that it can be used in other toolkits, templates, and applications. You can also clone versions in case you want to create a copy. This way a developer can create a new template using an existing template.

45. Click on the **Version** icon  in the top-right corner.
46. Click on **Create a version**.

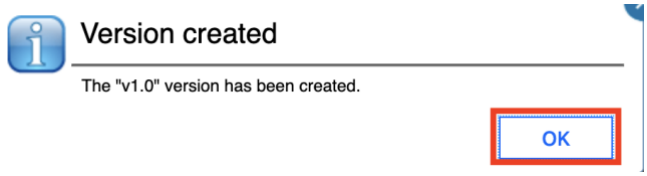


47. In the name field, enter **v1.0**, provide optional notes and click on **Create**.

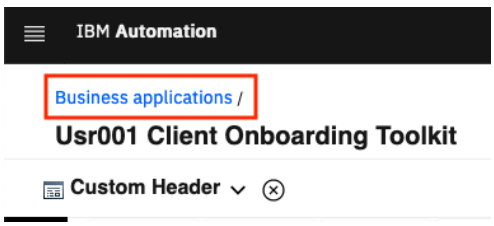


This automatically saves the artifacts and creates a new version of the toolkit.

48. Click on the **OK** button to confirm that the version was created.

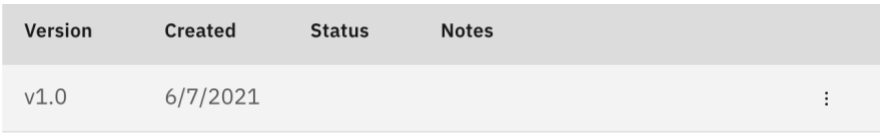


49. Click on **Business applications** in the top-left corner to go back to the repository.



50. Back in the repository, click on **Toolkits**.

51. Click on the **UsrNNN Client Onboarding Toolkit** tile. Do **NOT** click on the open button as it will reopen the toolkit. You should now see the **v1.0** version in the details on the right.



In the next exercise, we will use this toolkit in a **Client Onboarding** template. The template can then be used by application developers to create business applications based on client onboarding.

3 Exercise: Creating the Client Onboarding template

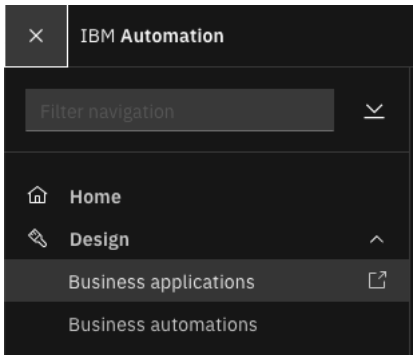
3.1 Introduction

In this exercise, we will create a template that will form as the start point for the **Client Onboarding** application in the next exercise.

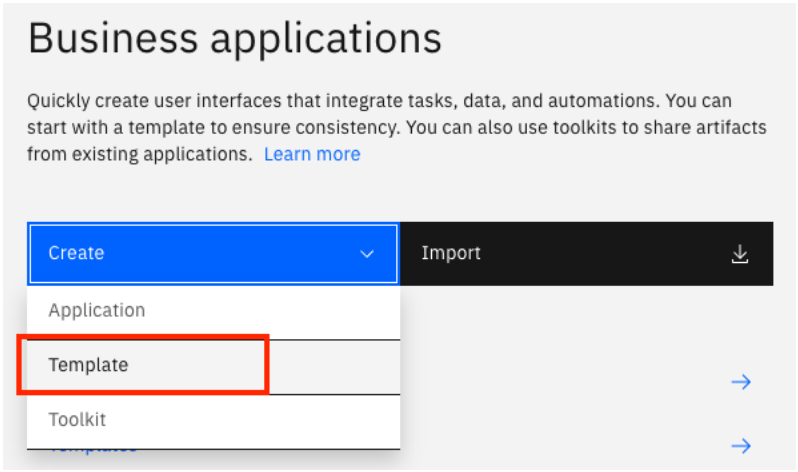
3.2 Exercise Instructions

3.2.1 Creating a template with toolkit dependencies

1. In your browser, open the IBM Automation page and login with the username & password assigned to you.
2. In the top-left corner, click on the menu icon and select **Design** → **Business applications** to access the repository.



3. Click on **Create** and select **Template**.

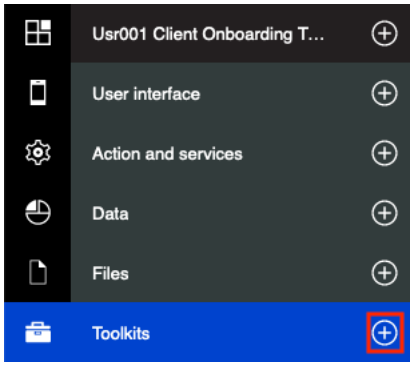


4. In the **Name** field, enter ***UsrNNN* Client Onboarding Template**.
5. Provide an optional **purpose**.
6. Click on **Create**.

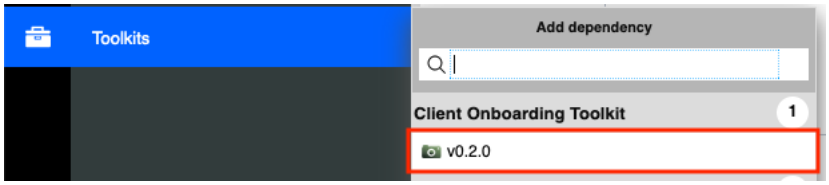
This opens the template editor at the Starting Page. When an application is launched, this is the first page that a user will see. An application can contain several pages.

Next, we will add two toolkit dependencies that are required to build this template. The first toolkit is the pre-built **Client Onboarding Toolkit** that contains some artifacts required for this lab. The second toolkit is the ***UsrNNN* Client Onboarding Toolkit** that we just created.

7. In the Library pane, click on the **+** button next to **Toolkits**.



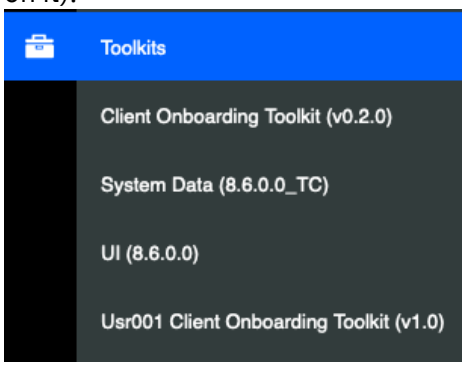
8. Select the only option under **Client Onboarding Toolkit**.



Note: The version number shown might differ from the screenshot.

9. Similarly, add the **v1.0** version of the ***UsrNNN*** **Client Onboarding Toolkit** you created in the previous steps.

   The toolkits section should look as follows when expanded (you can expand the section by clicking on it):



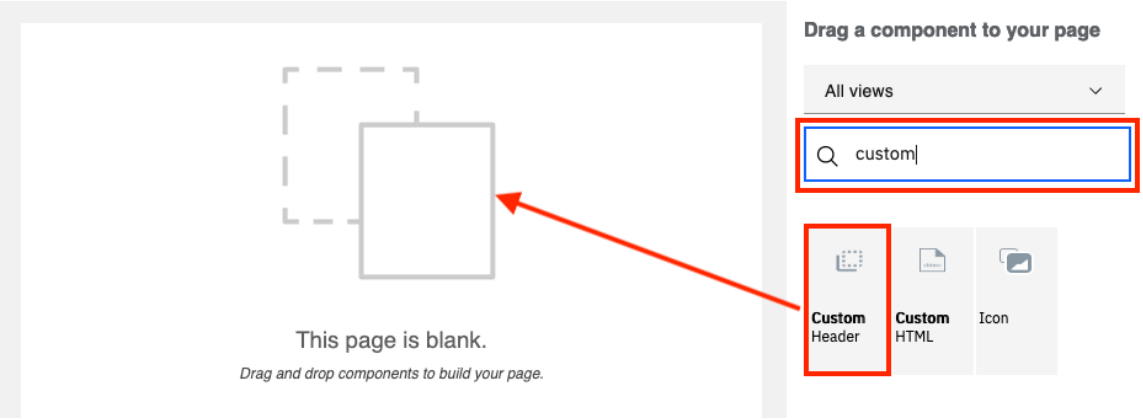
   The **System Data** and **UI** toolkits are added to each application, template and toolkit automatically.

   Next, we will update the **Starting Page** to include the UI necessary for any **Client Onboarding** application. The UI will call out to any required automation services. Automation services are services that are published by developers using other capabilities of the platform e.g., Workflow and Decisions.

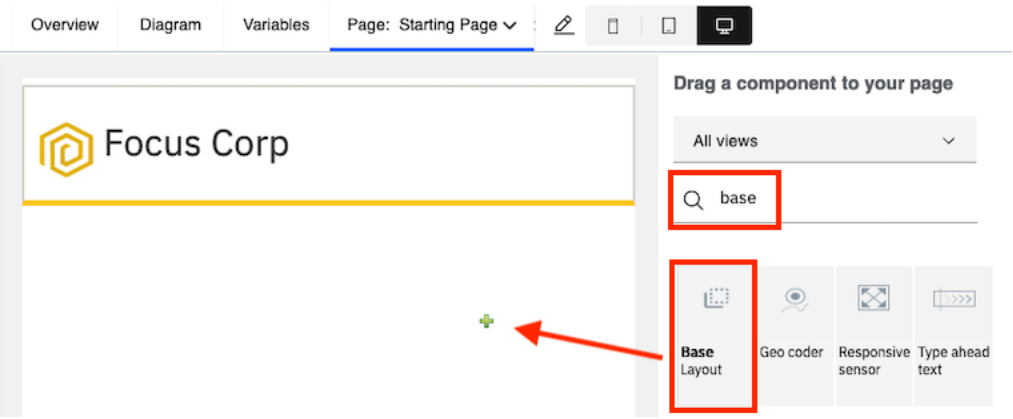
3.2.2 Creating UI that integrates with the Workflow capability

We will start by adding the **Custom Header** and **Base Layout** reusable views from the toolkits we just added to the template.

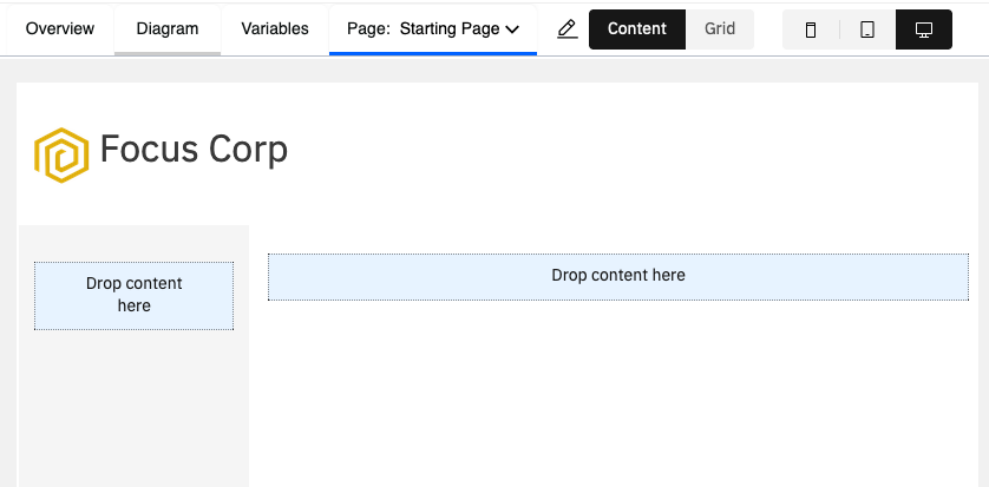
1. In the palette on the right-hand side, enter **custom** in the search field.
2. Drag and drop the **Custom Header** view onto the starting page.



3. Similarly, drag and drop the **Base Layout** view below the header.

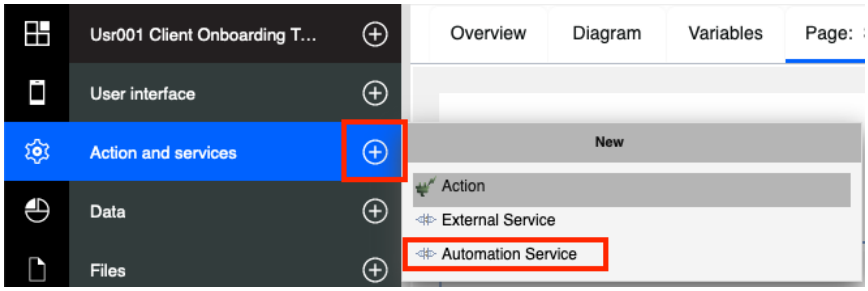


Your layout should now look as follows:

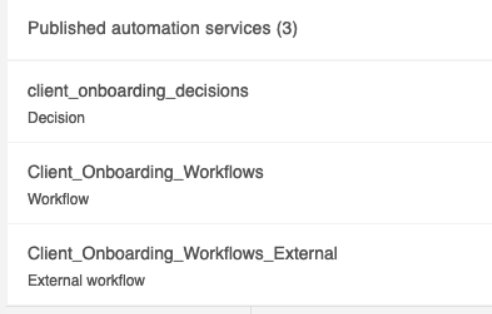


Next, we will add an **automation service** to the template that will allow us to get the details of a client that is being onboarded. This automation service is published by the developer of a Workflow application and provides the details of client based on the name of a known client. There are multiple ways to add an automation service to a template or an app. Let's look at the first one.

4. In the library pane on the left-hand side, click on the **+** button next to **Action and services** and select **Automation Service** as the artifact to create.



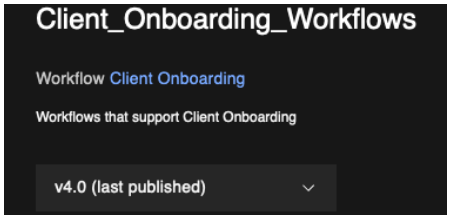
This launches the Automation Service discovery wizard that lets us add existing Automation Services to the template. The wizard shows all available Automation Services that are published from various capabilities of the Automation platform i.e., **Decision, Workflow** & **External Workflow**. The **External workflow** automation service is defined in an external Workflow system that can be both a traditional install or a containerized one. For the Client Onboarding scenario, the automation service to trigger a new Client Onboarding workflow is in an external Workflow system.



You may see a different list than the screenshot based on the system you are using.

5. Select the **Client_Onboarding_Workflows** automation service.

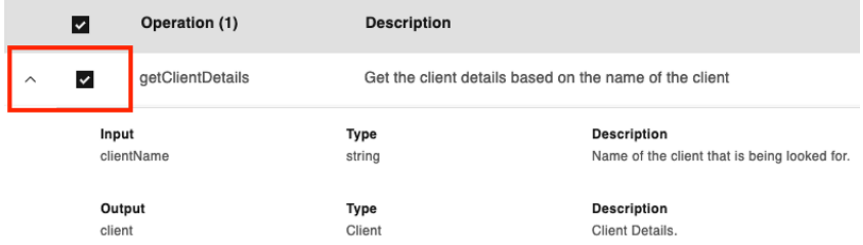
   This shows the available operations within this automation service which in this case is **getClientDetails**. An automation service could contain multiple operations. In the dropdown at the top, the last published version of the automation service is automatically selected.



6. Click on the **twisty** icon next to the **getClientDetails** operation to view its details.

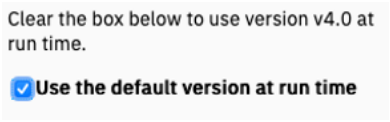
   Here you can see that the automation service requires the input **clientName** and provides the details of the **client** as the output.

7. Select the **getClientDetails** operation.

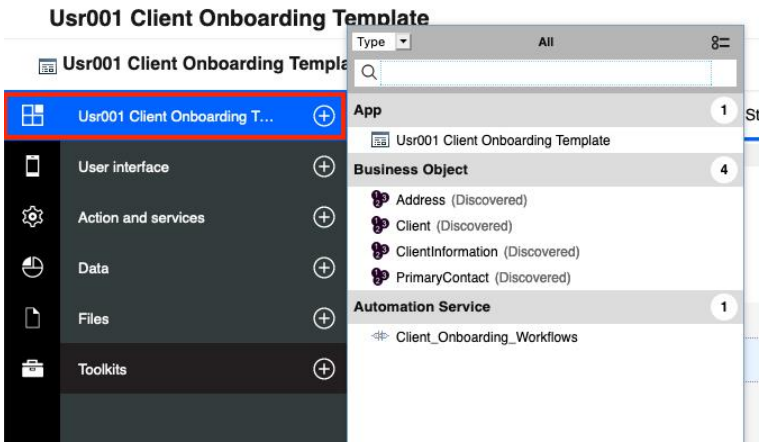


8. Click on the **Next** button.

   The next screen in the wizard gives us the option to select whether or not we want to use the default version at run time. What this means is that if a new version of the automation service is available, the application will automatically use that version during execution. We will keep that option selected.

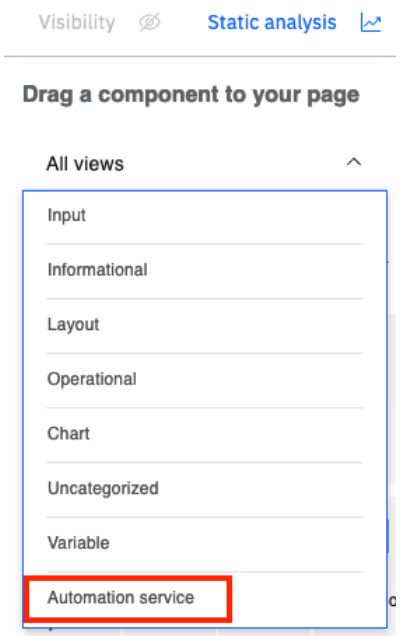


9. Click on **Submit** to add that automation service to the template.
10. Expand the first row in the library pane.



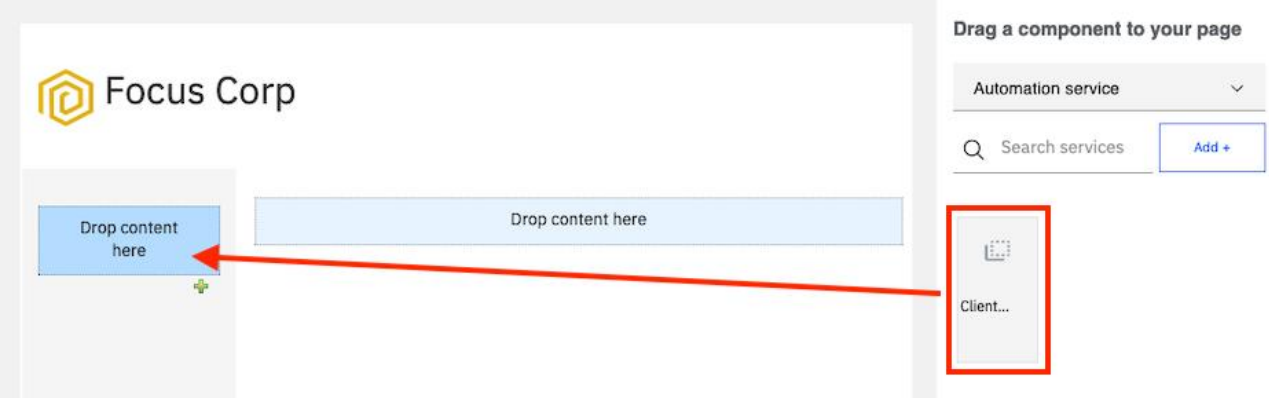
   The automation service **Client_Onboarding_Workflows** is added to the template along with all the business objects required to execute the service. Business objects are artifacts that encapsulate the business data just like Views encapsulate the UI. If you need to delete the automation service from an app/template, you can do it here as the advanced user as you have the library pane on the left-hand side available to you. Users in the basic view cannot see the library pane.

11. Back in the editor, in the palette on the right-hand side, switch to the **Automation service** option where it currently says **All views**.



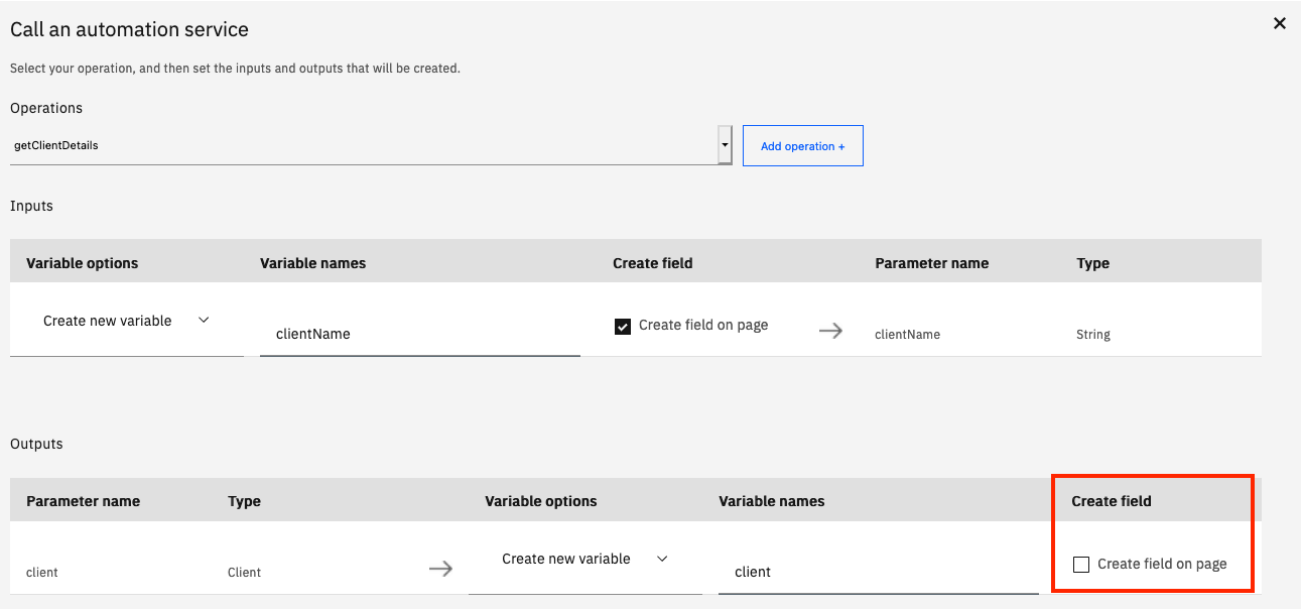
You will see that the **Client_Onboarding_Workflows** automation service is also shown here (you may have to clear any text present in the search field first). The right-hand side palette contains not only Views representing UI but also services and variables that can be dropped to the editor to create the relevant UI automatically.

12. Drag and drop the automation service from the palette to the grey column on the left where it says **Drop content here**.



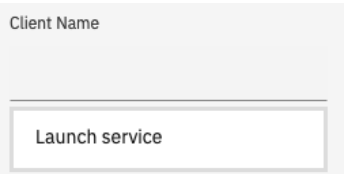
This pops up a wizard to call the automation service from the starting page. This wizard contains the list of operations available within the automation service and the inputs and outputs relevant to that operation. It also provides the users the ability to automatically create new variables and UI fields required to call this automation service.

13. Uncheck the **Create field on page** option for the **Output** as we will create the UI for the output in a different place on the page than the one we dragged the automation service to.

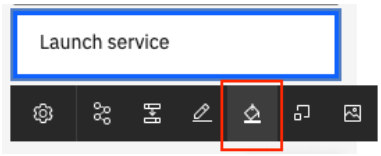


14. Click on **Done**.

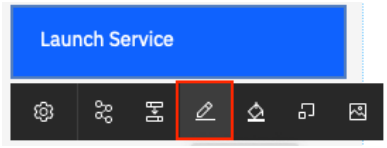
This adds the **Client Name** field on the page along with a **Launch service** button that will call the automation service when it is clicked.



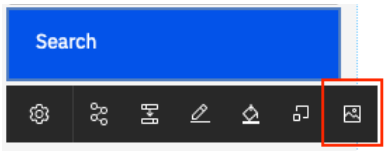
15. Click on the **Launch service** button and then click on the **Select color** icon.



16. Select the **Primary (dark blue)** color.
17. Click on the **Change label** icon to change the label of the button.



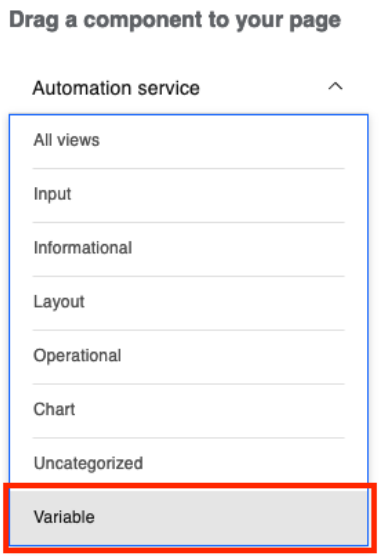
18. Update the label of the button to **Search**.
19. Click on **Select icon** to add an icon to the button.



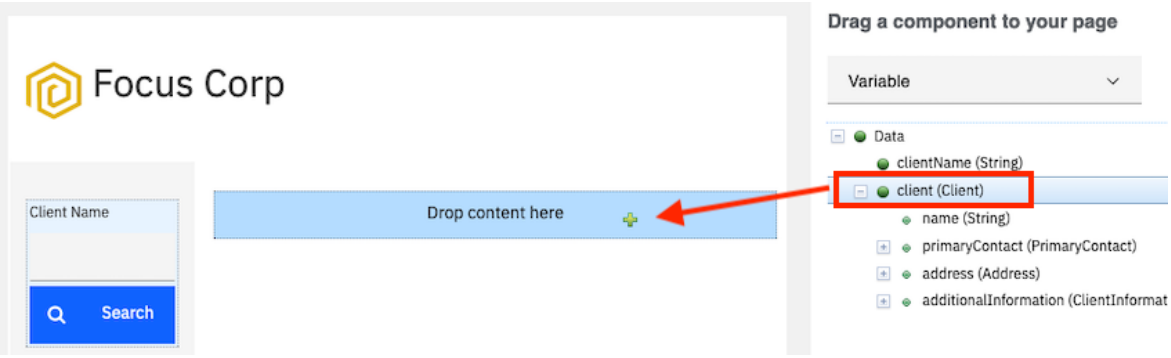
20. In the select icon wizard, enter **searc** and pick one of the icons to show on the button.

Next, we will add the UI fields for the client details on the page.

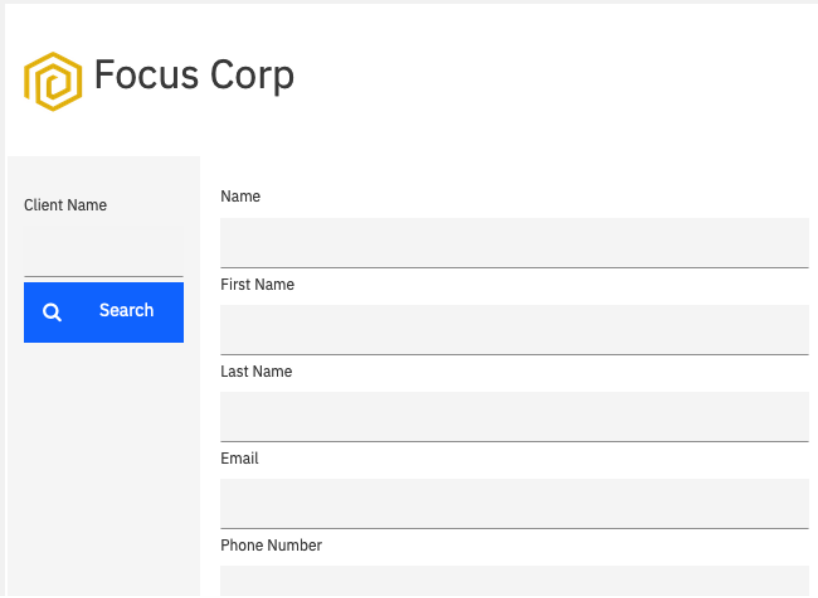
21. In the palette on the right-hand side, select the **Variable** option.



22. Drag and drop the **client** variable into the right column on the page.

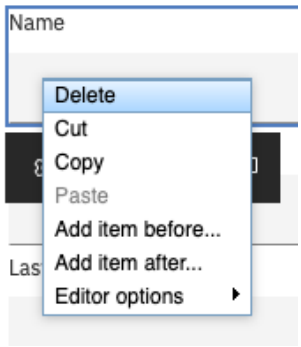


This automatically creates all the fields required to show the client details on the page. Scrolling down on the fields will show you that the fields created automatically point to the right view depending on the type of field. For example, the **Defaulted payment** field is a checkbox because the field is bound to a variable that is a Boolean.



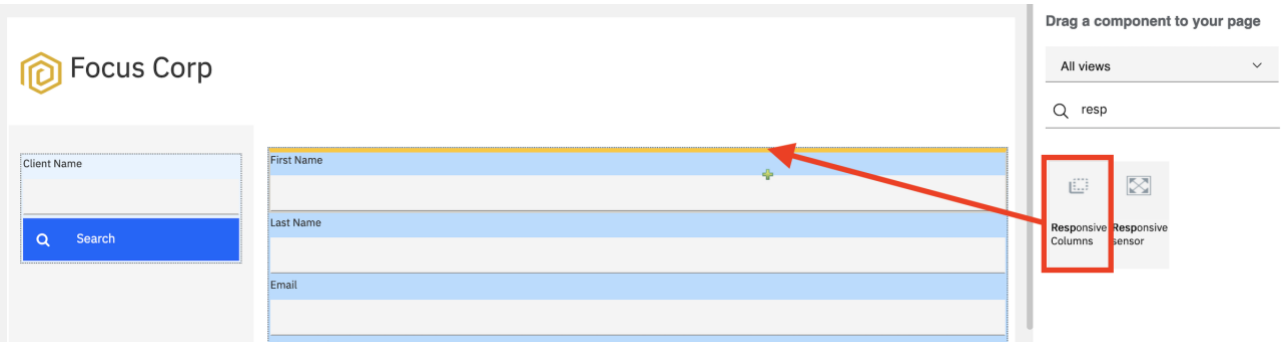
Next, we will customize the UI that was automatically created by dragging and dropping the variable.

23. Right click on the **Name** field and delete it as we already have the client name in the left column.



24. In the right-hand side palette, switch to the **All views** option and drop the **Responsive Columns** view above the **First Name** field.

    This view was created in the previous exercise to allow for responsive UIs.



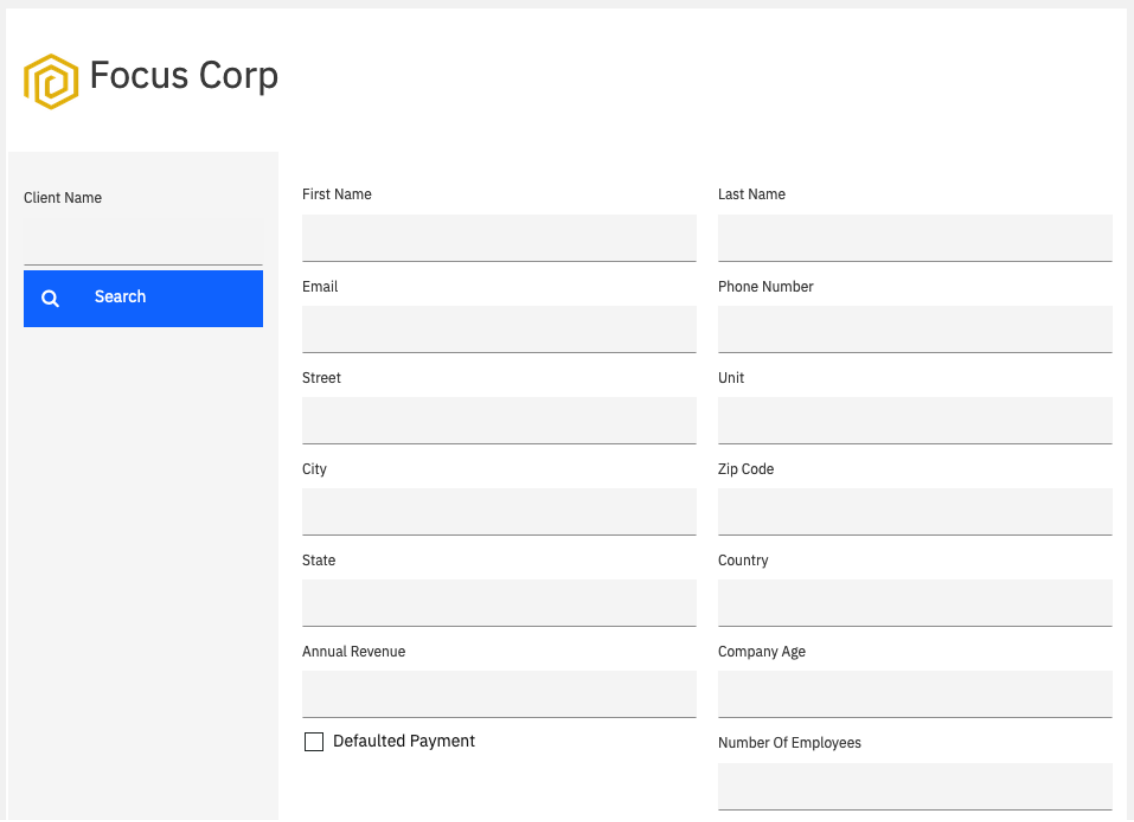
25. Drag and drop the **First Name** and **Last Name** fields to the left and right columns of the responsive columns view.

    They should now be side-by-side:

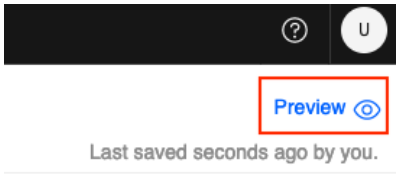
26. Optionally, customize the rest of the page using the **Responsive Columns** view.

    This will mean that on larger screens, more fields are displayed horizontally and on a smaller screen, they will be displayed vertically.

    If you do this for all fields, your page will look similar to the screenshot below:



27. Click on **Preview** in the top-right corner to preview the template built so far and test it.



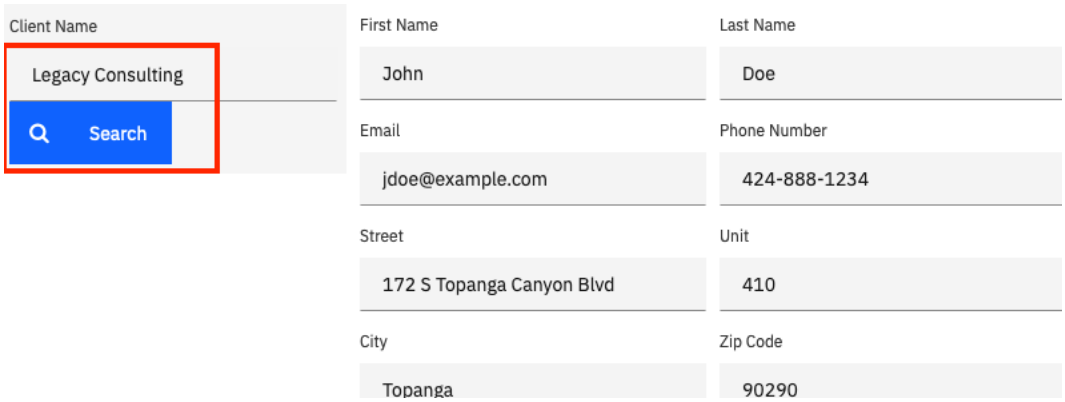
Clicking on the preview button deploys the template as an app to a **Playback Application Engine** that is separate from the **Business Automation Application Engine** used for apps published to production. Once the deployment is complete, it launches the app in a new window.

Note: You may have to allow browser pop-ups in order to see the previewed app.

28. Once the app launches, enter **Legacy Consulting** in the **Client Name** field on the left.
29. Click on **Search**.

This calls the automation service and fills in the fields on the page with the output of the service. As the automation service is published by a Workflow developer, the automation service in the background invoked the Workflow service to get the details of the client.

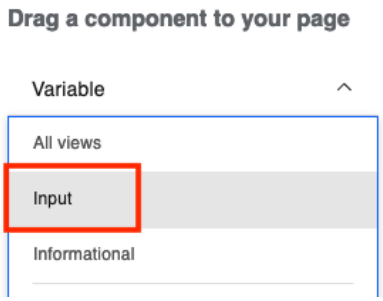
30. **Resize** the browser window in different sizes to verify the responsiveness of the app.



31. **Close** the browser window displaying the previewed app.

Next, we will add a button that takes us to a new page to review documents associated with the client being onboarded. This will allow you to see how a user can integrate with the Content capability using the out-of-the-box Content toolkit.

32. In the palette on the right, select the **Input** option in the dropdown.



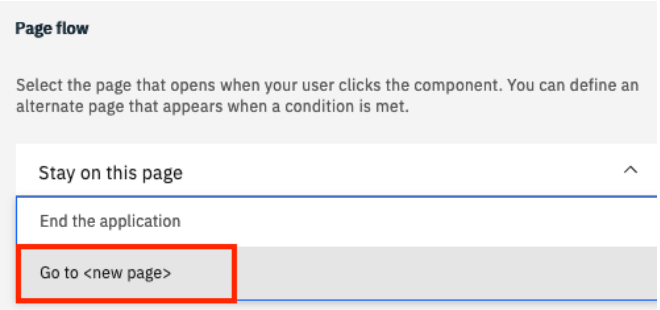
33. Drag and drop the **Button** view below the last field on the page.

This shows the **Next step** wizard that allows users to configure the steps to be performed when this button is clicked. The first page contains **Service** options i.e., which service should be executed when the button is clicked. In this case, we just want to go to a new page so we can leave the default option **Does not call a service** selected.

34. Click on **Next**.

    This shows the **Page flow** part of the wizard where users can select the page the app should navigate to on the click of the button.

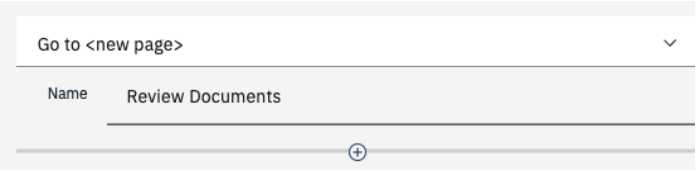
35. Click on the dropdown and select **Go to <new page>**.



    Note: A button can both call a service and navigate to another page on a click. The service will be called before navigating to the selected page.

36. In the **Name** field, enter **Review Documents**.

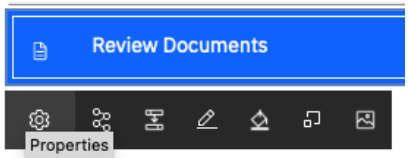
    The **+** button below the name field, allows users to conditionally navigate to a different page based on the value of a variable. For the purposes of this lab, we will not be doing that. If you did click on the button, you can remove the added option by clicking on the **x** icon next to it.



37. Click on **Done** to close the wizard.

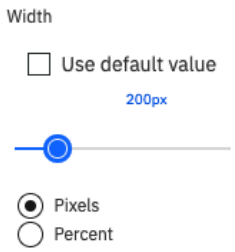
38. Style the button to call it **Review Documents**, with a **dark blue** color and a **document icon**.

39. To reduce the width of the button, click on the **Properties** icon for the button.



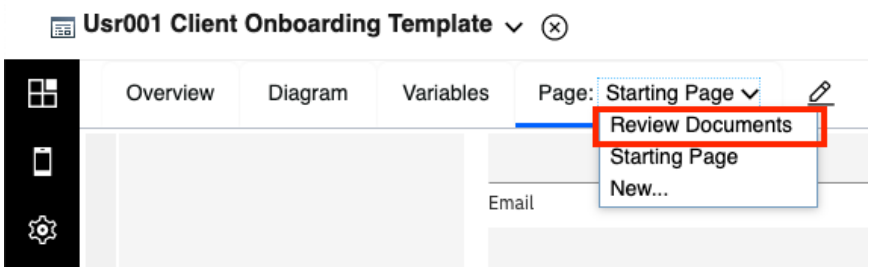
    This opens the property editor specific to the **Button** view. If you are shown the advanced property editor, switch to the basic one. If it says **Switch to basic properties** in the upper-right corner then you are in the advanced editor and vice versa.

40. In the **Appearance** section, uncheck the **Use default value** field and slide the slider to **200px**.

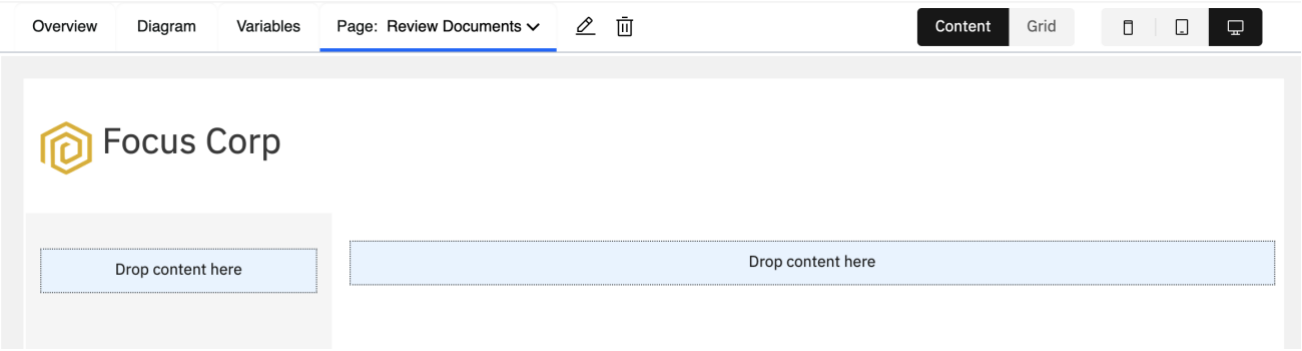


41. Click on **Done**.

42. In the page dropdown at the top, select the **Review Documents** page to edit its layout.



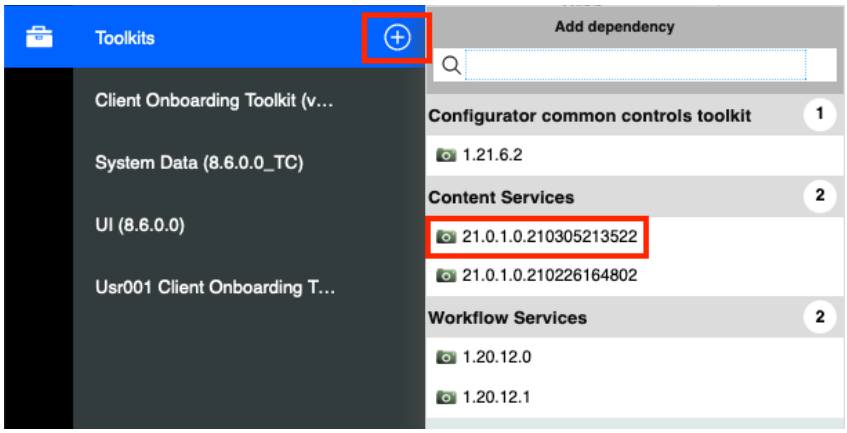
43. Add the **Custom Header** and **Base Layout** views to the editor like we did for the starting page.



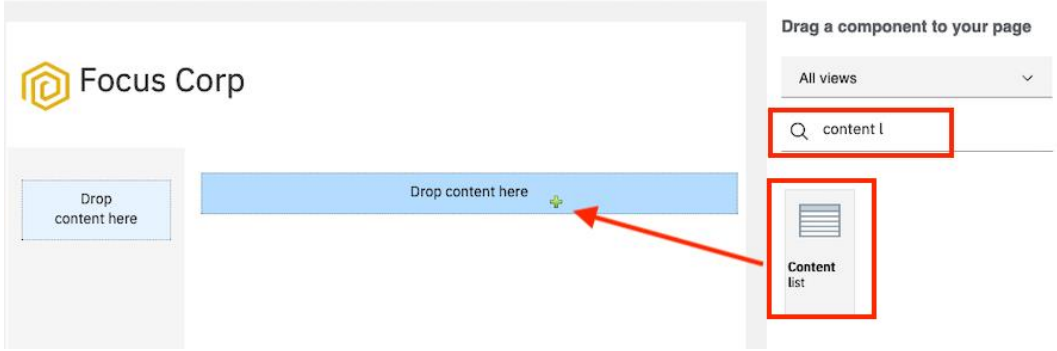
As we need to interact with the Content capability next, we will add the out-of-the-box **Content Services** toolkit to the template.

3.2.3 Creating UI that integrates with the Content capability

1. Add the **Content Services** toolkit as a dependency to the template. Pick the first version of the toolkit to add.

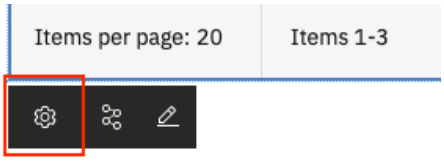


2. From the palette on the right-hand side, drag and drop the **Content list** view onto the editor.



This view is a part of the **Content Services** toolkit and provides a low-code way to interact with documents and folders in a content repository.

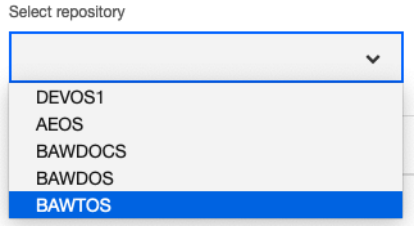
3. Open the **Properties** pane for the view.



4. If the properties pane opens in advanced mode, switch to the basic mode.

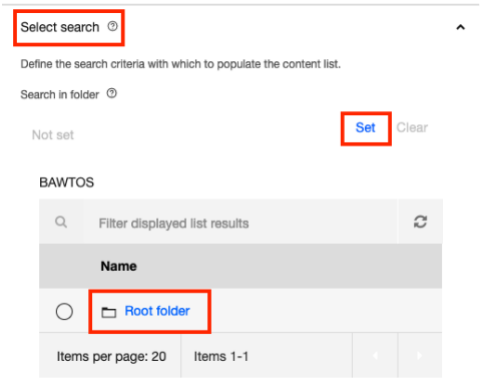
We will now need to select a repository that we want to view the list of documents from.

5. In the **Select repository** field, select the target object store of your system (**BAWTOS**).

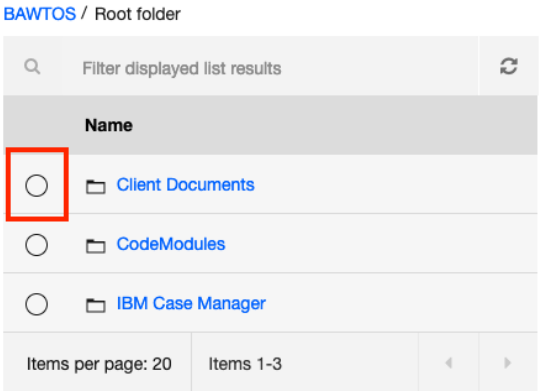


Next, we want to search for documents in the **Client Documents** folder of this repository that have a property **Client Name** that matches the name of the client being onboarded.

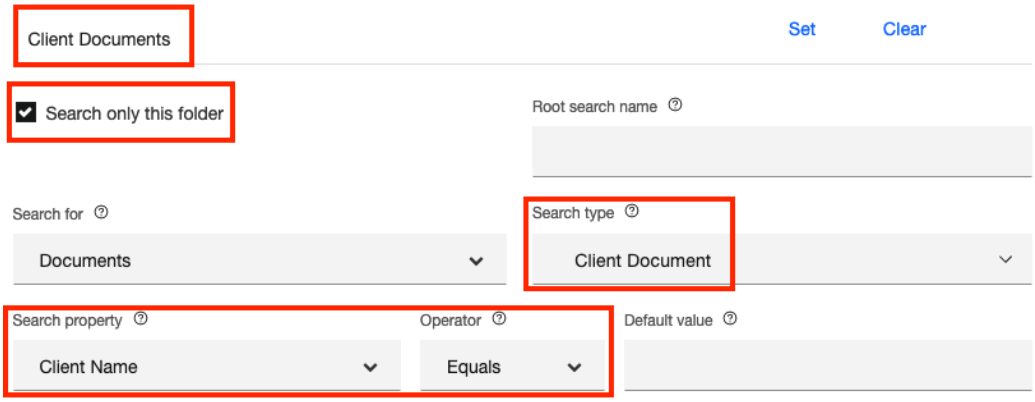
6. Expand the **Select search** section.
7. Click on **Set**.
8. Click on **Root Folder**.



9. Select the **Client Documents** folder.

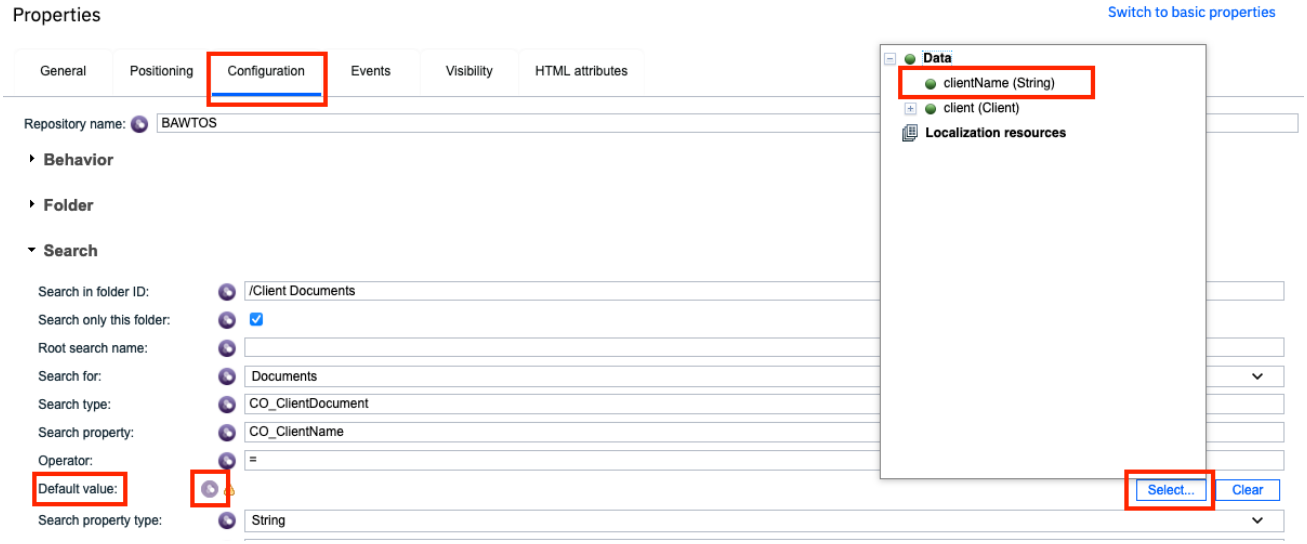


10. Select the **Search only this folder** checkbox to avoid checking subfolders.
11. In the **Search type** field, select the **Client Document** type.
12. In the **Search property** field, select the **Client Name** property.
13. In the **Operator** field, select the **Equals** operator.



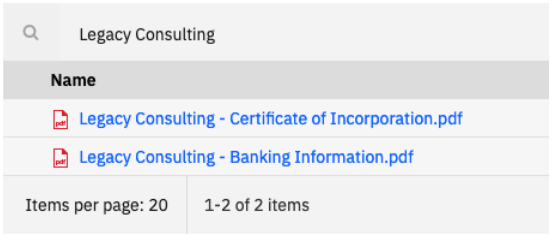
We want the **Default value** to be the name of the client. Since this is a variable, we will have to use the advanced mode.

14. At the top of the properties pane, click on **Switch to advanced mode**.
15. Click on the **Configuration** tab.
16. Expand the **Search** section.
17. For the **Default value** field, click the variable picker icon and **Select** the **clientName** variable.



18. Click on **Done** to close the properties pane.
19. Click on **Preview** in the top-right corner to preview the changes made to the template.
20. Once the app opens in a new window, enter **Legacy Consulting** in the **Client Name** field and click on **Search**.
21. Click on **Review Documents** to go to the next page.

Verify that the **Content List** does a search for **Legacy Consulting** and shows two documents



22. Close the preview window to go back to the Application Designer.

3.2.4 Persisting data within an application

If you preview the template again, the app starts fresh with no data persisted in it. For cases where users want to work on an app but come back to it later, the Application Designer provides the ability to persist the data so that it can be accessed again when an app is re-launched. Next, we will enable this persistence for the variables that were automatically created on the addition of the automation service.

Note: When a user starts an application, they "own" the instance of that application. A user can only run a single instance of an application at the time and cannot pass the ownership/control of that application to another user.

1. Click on the **Variables** tab of the template.

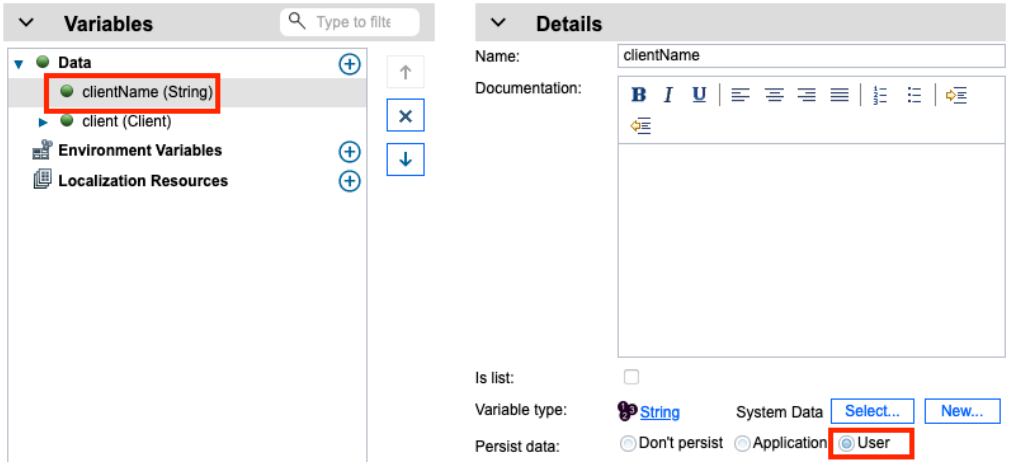
2. Click on the **clientName** variable.

   The **Persist data** field offers options to persist the data for an application or a user.

   - The application option persists the data in the context of an application and any user launching this app will see data persisted by other users.
   - The user option persists the data in the context of an application **only** for that user. Other users launching this app will work with their own data.

   In both cases, the persisted data remains even if a new instance of the application is launched. The user can choose to clear the persisted data using JavaScript on the end of an application. The JavaScript can use the low-code mechanisms provided by Application Designer as you will see later in this exercise.

3. For this lab, select the **User** option.



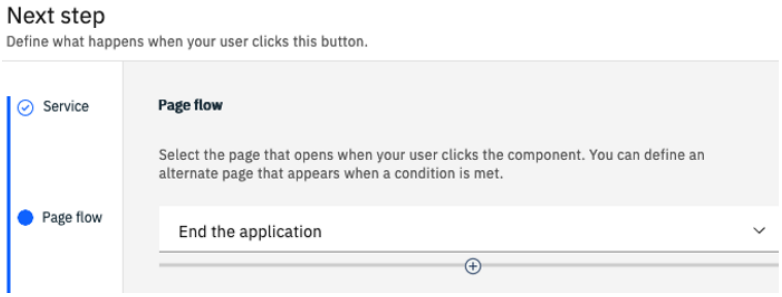
4. Repeat data persistence step for the **client** variable.
5. Preview the template.
6. Enter **Legacy Consulting** in the **Client Name** field.
7. Click on **Search**.
8. Click on **Review Documents** to go to the next page.
9. Close the preview window.
10. Preview the template again.

    You will see that the client data is now persisted. Next, let's see how the user can use low-code JavaScript to clear the persisted data when the application ends. For this we will first need to add a button that ends the application.

11. Close the preview window and go back to the Application Designer.
12. In the **Review Documents** page, add a **Button** below the **Content List**.

This brings up the **Next step** wizard. At this point, in the Client Onboarding end-to-end scenario, we call an automation service from an external Workflow system to launch a new Client Onboarding Workflow. For this lab, as we have already learnt how to call an automation service, we will simply end the application.

13. Click **Next**.
14. In **Page flow**, select **End the application**.

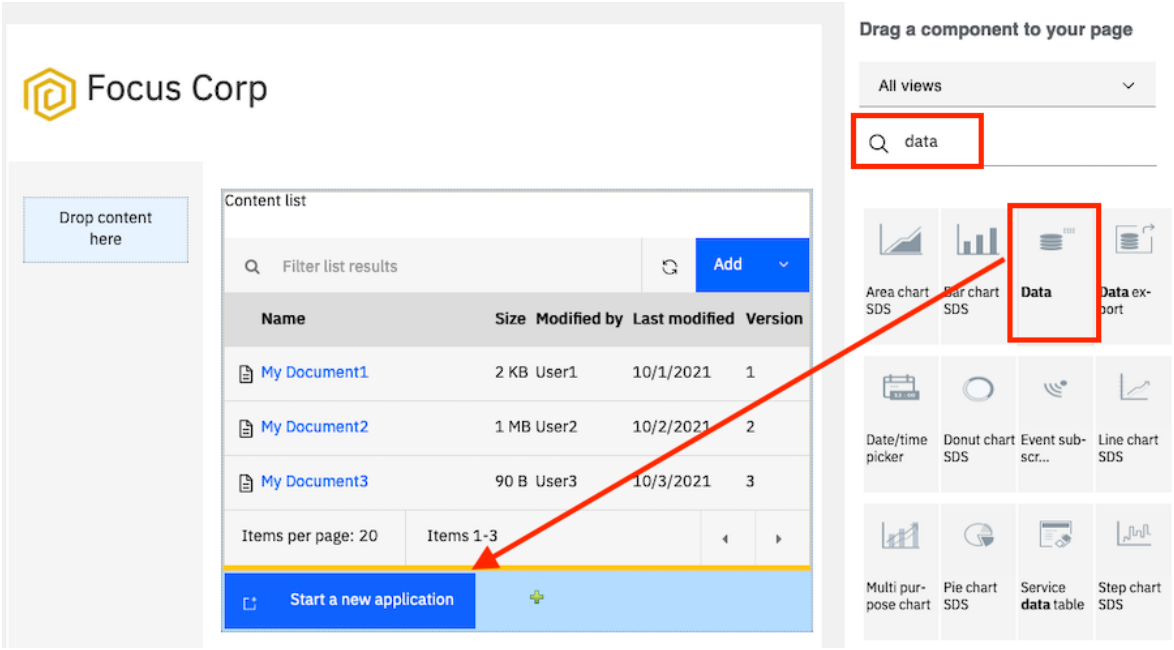


    When you end an application, a new instance of the application is automatically started.

15. Click on **Done** to close the wizard.
16. Customize the button by calling it **Start a new application**, changing the color to **dark blue**, adding an icon. You can optionally reduce the width of the button.

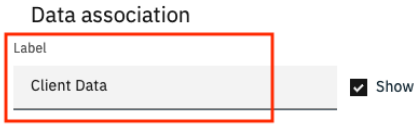
    Next, we want to clear the persisted data when the button is clicked. For this we will need to access the data defined in the **client** variable. To do this, we can use the **Data** view.

17. Drag the **Data** view from the palette on the right above the button.



This brings up the **Data association** wizard where you can bind data to the view.

18. Update the label to **Client Data**.

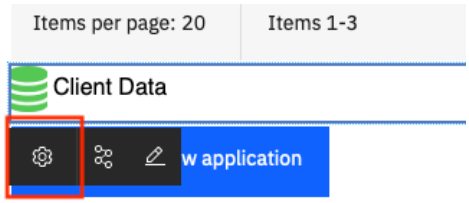


19. Click on **Select existing**.

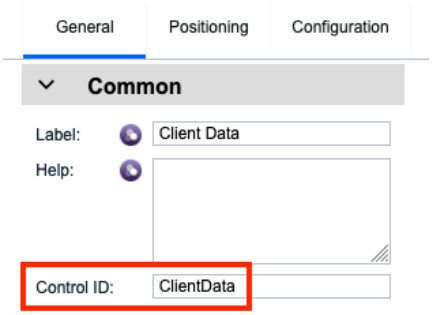
20. In the **Select existing variable** field, select the **client** variable.

21. Click on **Done** to close the data association pane.

22. Open the properties pane for the **Client Data** view.

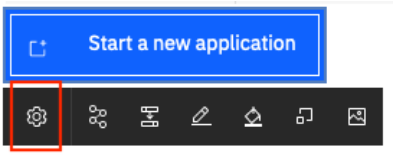


23. In the properties pane, update the **Control ID** field to **ClientData**.



The **Control ID** can be used in low-code JavaScript to access that view. Each out-of-the-box view comes with its own methods that can be invoked once the view is accessed. We will access the **Data** view when the **Start a new application** button is clicked and clear the data stored within in to delete the persisted data from the system.

24. Click on **Done** to close the properties pane.
25. Open the properties pane for the **Start a new application** button.

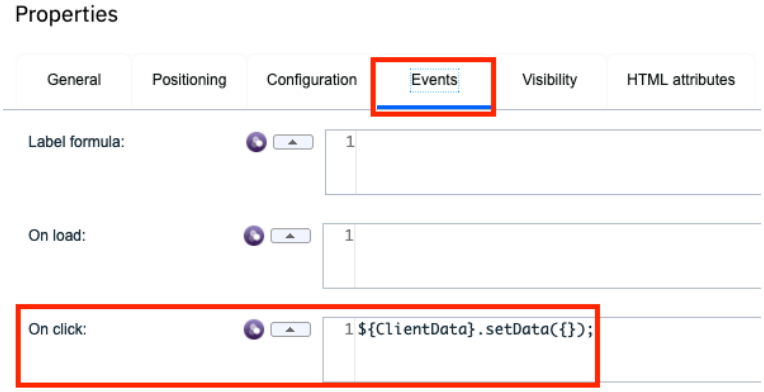


26. Click on **Switch to advanced properties** in the top-right corner.
27. Click on the **Events** tab.

The event tab lists events specific to a view where users can add their code. In this case, we will update the **On Click** event to add code that accesses the **Client Data** view and clears the data associated with it.

28. In the **On Click** event handler, add the following code:

```
${ClientData}.setData({});
```



- **${ClientData}** accesses the **Client Data** view previously added.
- **setData** is a method available for the **Data** view that sets data to the input of the method.
- **{}** is an empty object provided to the input of the **setData** method to clear the persisted data.

29. Click on **Done** to close the properties pane.
30. Preview the template.
31. If the persisted data is not already shown, enter **Legacy Consulting** as the name of the client, do a search for the client details.
32. Go to the **Review Documents** page.
33. Click on **Start a new application**.
34. The client details should now be cleared.

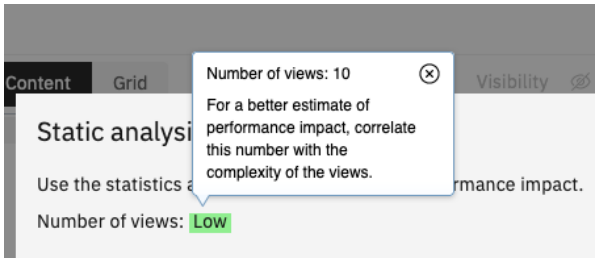
    You will notice that the name of the client is still there as we put that into a different variable, **clientName**, that was not cleared. You can optionally choose to clear the persisted data for it.
35. **Close** the previewed application and go back to Application Designer.

3.2.5 Analyzing the performance of an application

Before making this template available, we want to analyze it from a performance perspective. There is a static analysis which looks at the number of views on the page and a runtime analysis which analyses the page in real-time. For this lab, we will focus on the static analysis.

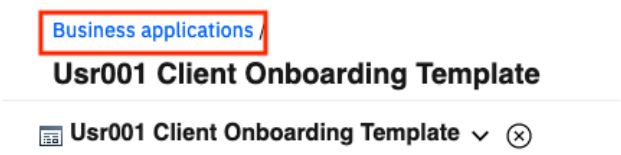
1. In the top-right corner, click on **Static Analysis**.

2. Hover over the **Low** label to show more details on the page:



   This gives us an estimate of the performance impact. 10 views with a high complexity can cause performance issues as well. If the page had repeatable views such as a table, the static analysis would also list those views and warn the developer that large lists can cause performance issues.

3. Create a new version of the template called **v1.0** using the version icon in the top-right corner.
4. Click on **Business Applications** in the top-left corner to go back to the repository.



With that you have successfully created the Client Onboarding template that can be used by business users to create Client Onboarding applications. You will do that in the next exercise.

4 Exercise: Creating the Client Onboarding application

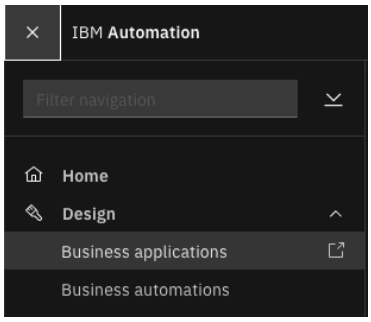
4.1 Introduction

In this exercise, we will create the **Client Onboarding** application based on the template created in the previous exercise. The application developer will use the template as a starting point and expand on it by adding an automation service (published by a developer using the Decisions capability) along with relevant UI that allows the client to sign up for services offered by Focus Corp. We will also use the **Basic view mode** to emulate the development experience of a business user.

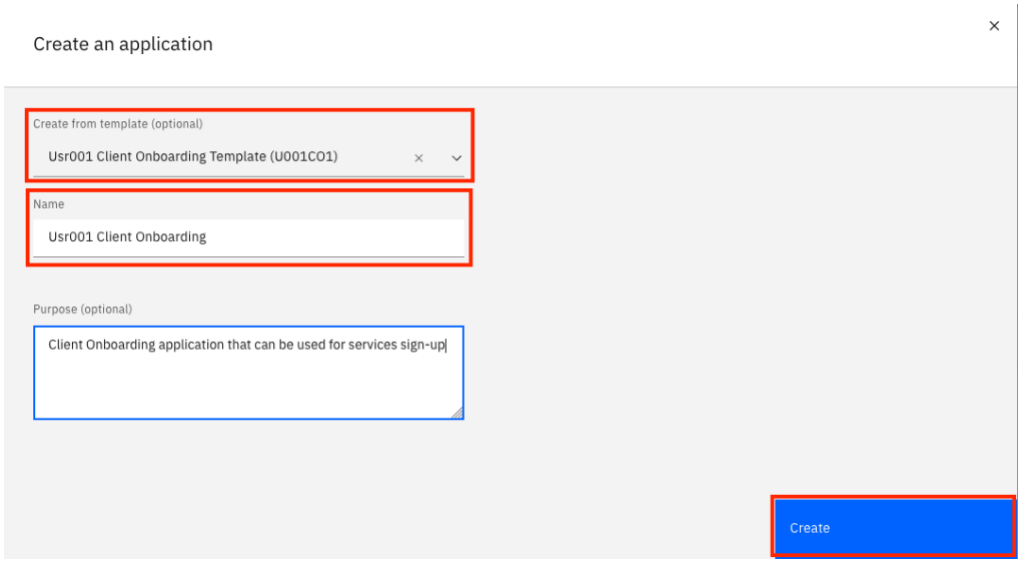
4.2 Exercise Instructions

4.2.1 Creating an application from a template

1. In your browser, open the IBM Automation page and login with the username & password assigned to you (if not done so already).
2. In the top-left corner, click on the menu icon and select **Design → Business applications** to access the application repository.

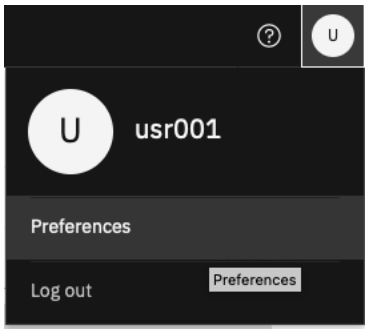


3. Click on **Create** and select **Application**.
4. In the **Create from template** field, select the template created in the previous exercise i.e. ***UsrNNN* Client Onboarding Template**.
5. In the **Name** field, enter ***UsrNNN* Client Onboarding**.
6. Provide an optional purpose.
7. Click on **Create**.

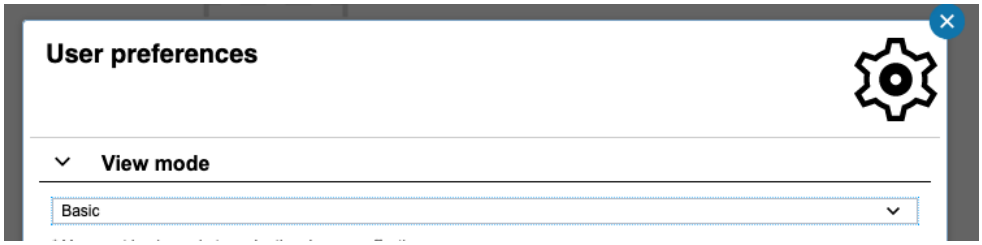


A new application will now be created based on the Client Onboarding template. This means that all toolkit dependencies, views, pages, etc. from the template will be copied to the application. If a template is updated, the application developer can choose to update the application to the latest version of the template. This will update all the toolkit dependencies as defined in the template but keep everything else as-is in the application.

8. In the top-right corner, click on the **user icon** and select **Preferences**.



9. In the **View mode** dropdown, select **Basic**.
10. Click on **Save**.



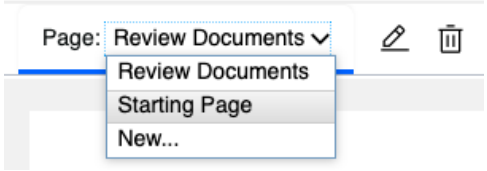
11. For the changes to be effective, **refresh** the browser window.

    As you can see, the UI is a lot simpler to use now. For example, the library pane on the left is no longer visible, the tabs on the top (Variable, Layout, etc. are also hidden). This allows the application developer to focus purely on the application and build UIs using existing services.

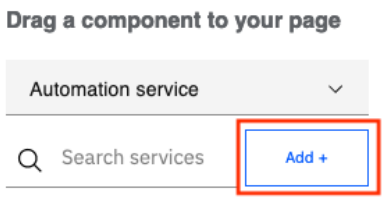
    The application developer wants to now add UI to the first page of the application that can be used to sign up for services and show a fee associated with those services.

4.2.2 Creating UI that integrates with the Decisions capability

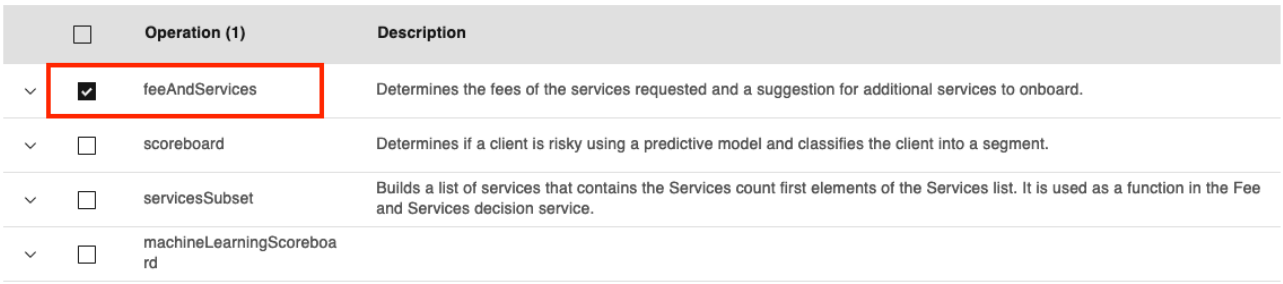
1. Switch to the **Starting Page** in the top-left corner.



2. In the palette on the right-hand side, select the **Automation Service** option.
3. Click on **Add +**.



4. Select the **client_onboarding_decisions** automation service.
5. Select the **feesAndServices** operation.



6. Click **Next**.

   The automation service is now available to use in the application. Previously, we added an automation service to the page by dragging and dropping it to the page. While that is still possible (and the easier way), in this exercise, we will try a different approach and add a button first and then configure it to call the automation service. This is helpful because there are other views (e.g., Navigation Event) that can be used to call automation services that use the same principle.

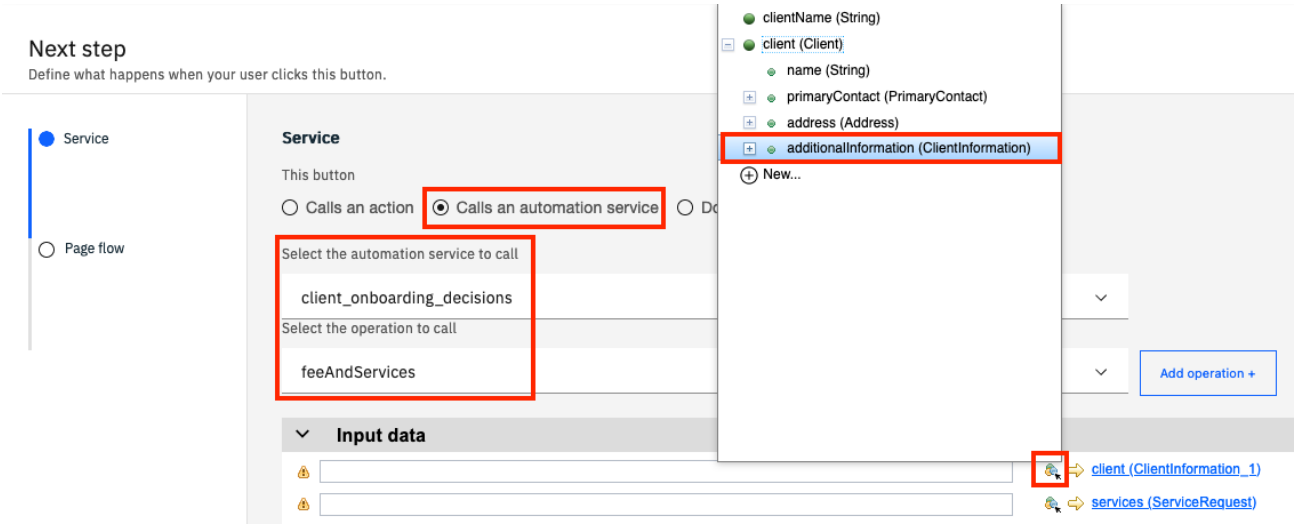
7. Drag and drop a **Button** view above the **Review Documents** button.

8. In the **Next Step** wizard, select **Calls an automation service**.
9. In the **Select automation service to call** field, select **client_onboarding_decisions**.
10. In the **Select operation to call** field, select **feeAndServices**.

    The **Input data** and **Output data** now show in the wizard. These show the inputs and outputs defined for the **feesAndServices** operation for the **client_onboarding_decisions** automation service. As you can see, the inputs required to get the fee and services i.e. **client** with data type **ClientInformation_1** and **services** with data type **ServiceRequest**. The **_1** is appended as another business object with the name **ClientInformation** already exists in this application (from the discovery of the first automation service).

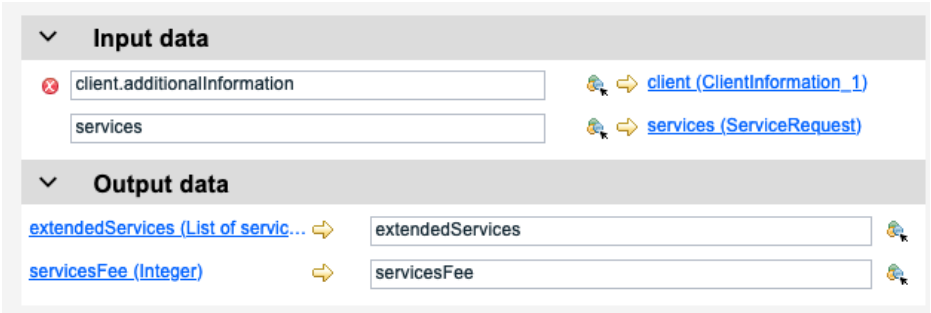
    Next, we will map variables to the fields in these sections. In some cases, we will use existing variables that were created on the creation of the first automations service in the template and in other cases, we will create a new variable.

11. For the input data section, select the existing **client → additionalInformation** variable using the variable picker icon .



As you can see the input **client** and selected variable **client.additionalInformation** have similar data types, **ClientInformation** & **ClientInformation_**1. These were automatically created on creation of the two automation services in the app but contain the same attributes within them. You will see an error icon at this point telling us that the two data types do not match but we can ignore this because the attributes within the two defined types are the same.

12. For the **services** input, click on the variable picker icon and select **New...**
13. Repeat the previous step for the output data variables.

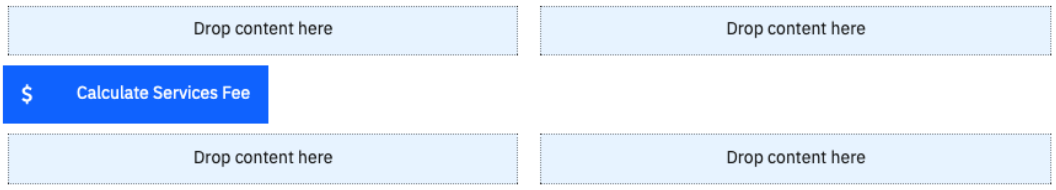


14. Click **Next**.

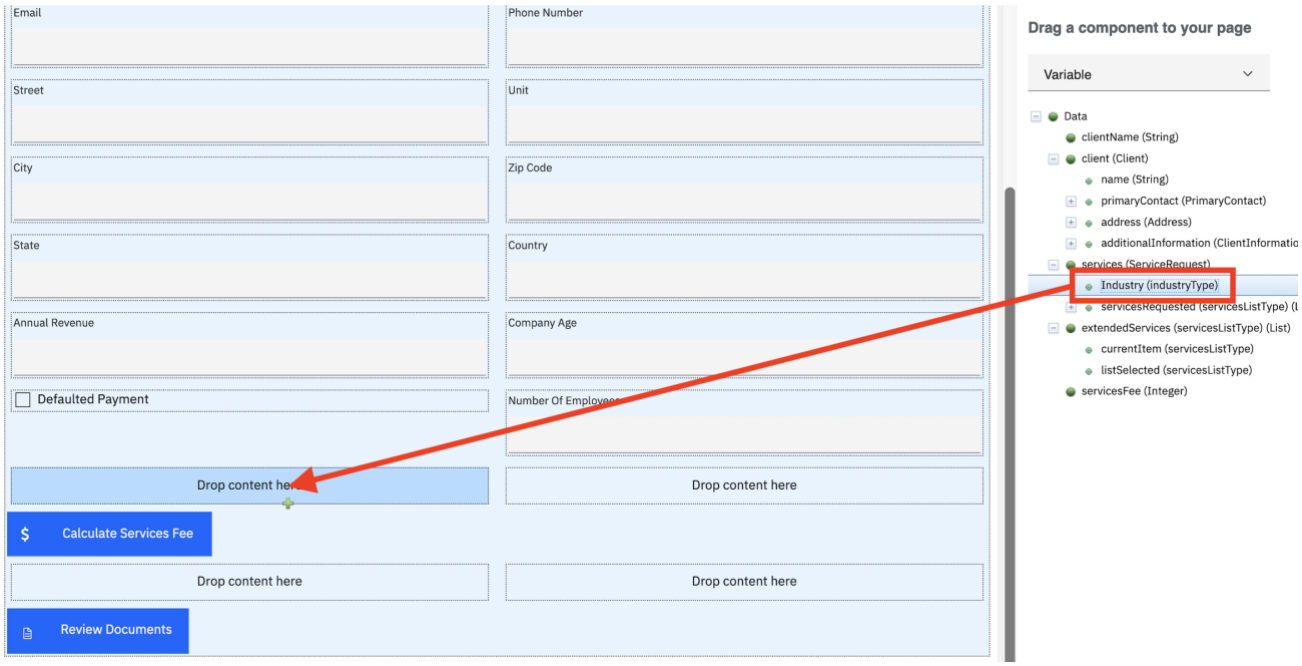
    We want to stay on the current page when this button is clicked so we will make no changes to the page flow.

15. Click **Done**.
16. Customize the button by calling it **Calculate Services Fee**, changing the color to **dark blue,** adding an icon and reducing the width.

17. Drag and drop a **Responsive Columns** view above the **Calculate Services Fee** button.
18. Drag and drop another **Responsive Columns** view below the **Calculate Services Fee** button.



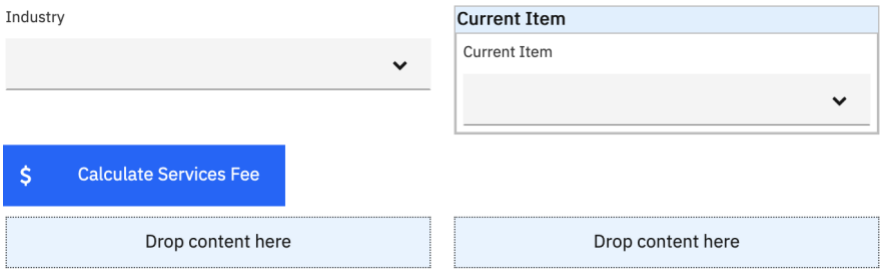
19. In the palette on the right-hand side, switch to **Variable**.
20. Drag and drop the **services → Industry** variable in the left column of the first responsive columns view.



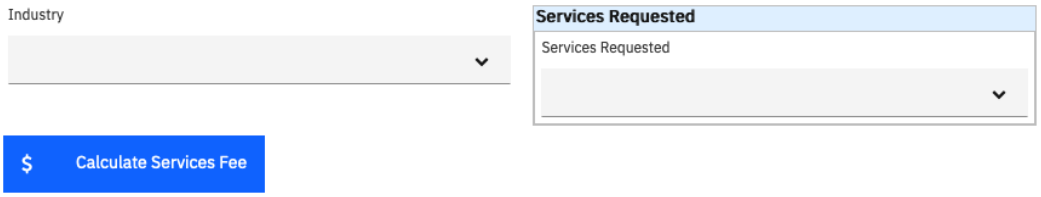
You will see that a dropdown is automatically created with values pre-filled from the automation service discovery.

21. Drag and drop the **services → servicesRequested** variable in the right column of the responsive columns view above the **Calculate Services Fee** button.

You will see that a table is automatically created as the **servicesRequested** is a list**.** The table contains a drop down with a list of services that was discovered as a part of the automation service discovery.

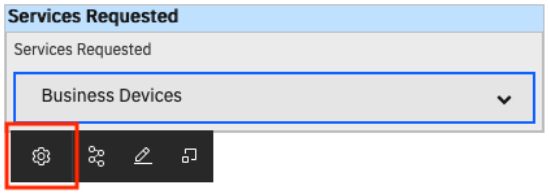


22. Update the label of the dropdown to **Services Requested.**



Next, we only want to show the services that are relevant to the chosen industry.

23. Open the properties for the **Services Requested** dropdown.

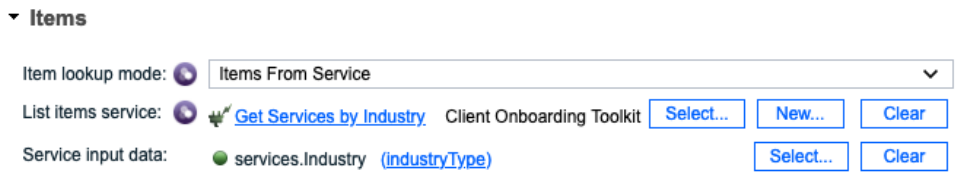


If you scroll down, you will see a list of all the services that is easily configurable by the application developer. The view provides different item lookup modes. One of the options is to dynamically look up items to show in the dropdown based on a specified input. This can be done using an action and as a part of this lab, a pre-built action is provided contains the services to display based on the chosen industry.

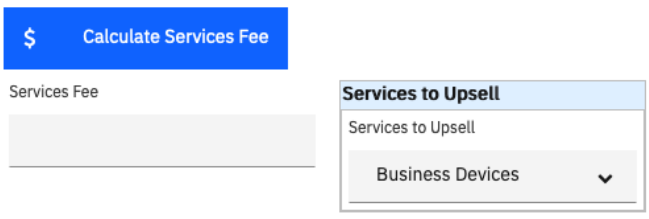
24. In the top-right corner, click on **Switch to advanced properties**.

Observe that the available tabs here are reduced as we are in the basic view mode.

25. Click on the **Configuration** tab.
26. Expand the **Items** section.
27. For the **Item lookup mode** field, select **Items from Service**.
28. For the **List items service** field, select the **Get Services by Industry** action.
29. For the **Service input data** field, select the **services → Industry** variable.



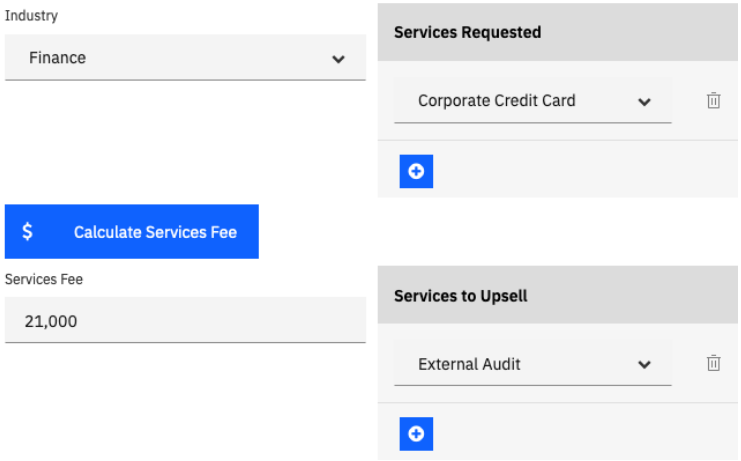
30. Click on **Done**.
31. Drag and drop the **servicesFee** variable from the right-hand side palette onto the left column of the responsive columns under the **Calculate Services Fee** button.
32. Drag and drop the **extendedServices** variable on the right column of the responsive columns under the **Calculate Services Fee** button.
33. Change the label of the extended services dropdown to **Services to Upsell**.



With that, we have successfully updated the application and the UI required to call the automation service.

Time to test it.

34. Click on **Preview**.
35. Enter **Legacy Consulting** as the name of the client.
36. Click on **Search**.
37. Choose **Finance** as the **industry**.
38. In the **Services Requested** field, click on the **+** button and select **Corporate Credit Card** as a selected service. Verify that the dropdown only shows financial services.
39. Click on **Calculate Services Fee**.
40. Verify that the **Services Fee** is **21,000** and the **Services to Upsell** shows an additional service.



41. **Close** the preview window.

As you can see, the developer of an application can integrate with multiple services built within the IBM Cloud Pak for Business Automation platform using automation services and the experience remains the same from an application development perspective.

The administrator can export the application as a .zip file and publish it to an IBM Business Automation Navigator desktop. The published app will then execute on the IBM Business Automation Application Engine.

With that you have successfully completed the creation of the Client Onboarding application.

Congratulations on completing the lab!