Appian Academy

Exercise 2: Examine and Configure a Process Model

Introduction to Process Modeling

Introduction

How Can I Practice?

You should practice in Appian Community Edition. Keep in mind that this is a community environment not suitable for production workloads or sensitive information.

Appian Version

Appian Community Edition is on the latest Appian version. If you are following the exercises from a previous Appian version, go to academy.appian.com to download the latest version.

Naming Conventions

When you register for Appian Community Edition, you gain access to a workspace, which is your personal area within the community environment. When your workspace has been assigned, you will receive a confirmation email with a workspace ID. Pre-loaded apps in your workspace include the workspace ID.

Use this workspace ID when creating new applications and objects. Otherwise, you could run into naming conflicts with objects created by other users. In the example below, the application prefix for the Acme Auto Solution app is W0000AS. So, the workspace ID is W0000, which would be included in any new app's prefix.

Save Often

Appian does not automatically save updates, so save your objects frequently.

Additional Resources

Appian provides a number of training resources for Appian developers. The following resources are particularly popular with our learners:

- Academy Online - Appian's online courses provide useful survey courses, step-by-step tutorials, and practice exercises. Explore these resources at your own pace. Survey courses will help you develop a better grasp of the topics you need to learn. Video and print tutorials will help you with getting hands-on experience with Appian.
- Community Discussions for New Users - Check out the **New to Appian** thread in Community. Join our community of experts to ask questions and find answers from past discussions.
- Appian Documentation - Appian's product documentation will provide you with an overview of key Appian features, newest release information, additional tutorials, and helpful patterns and recipes to implement in your app.

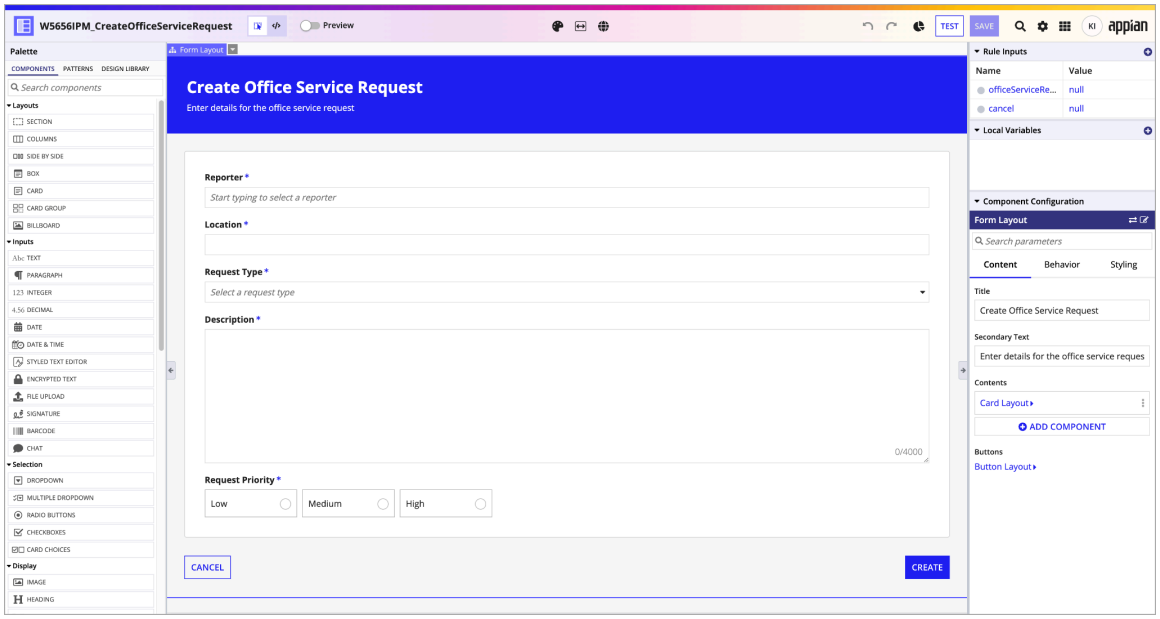
Examine Generated Objects

In the last exercise, you created a record type with an associated record action. When the record action was generated, Appian created two objects:

- CreateOfficeServiceRequest interface
- Create Office Service Request process model

In this exercise, you'll examine these objects to see how data moves from the interface and into the process model.

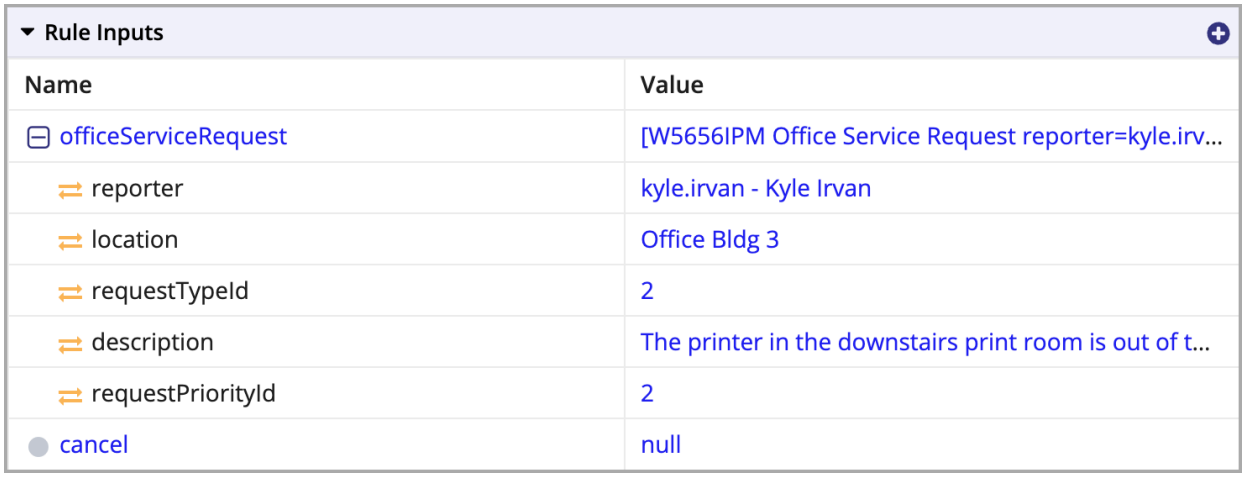
1. In the application Introduction to Process Modelling that you created in the previous exercise, open the interface **W#IPM_CreateOfficeServiceRequest**. You'll see an interface similar to this:



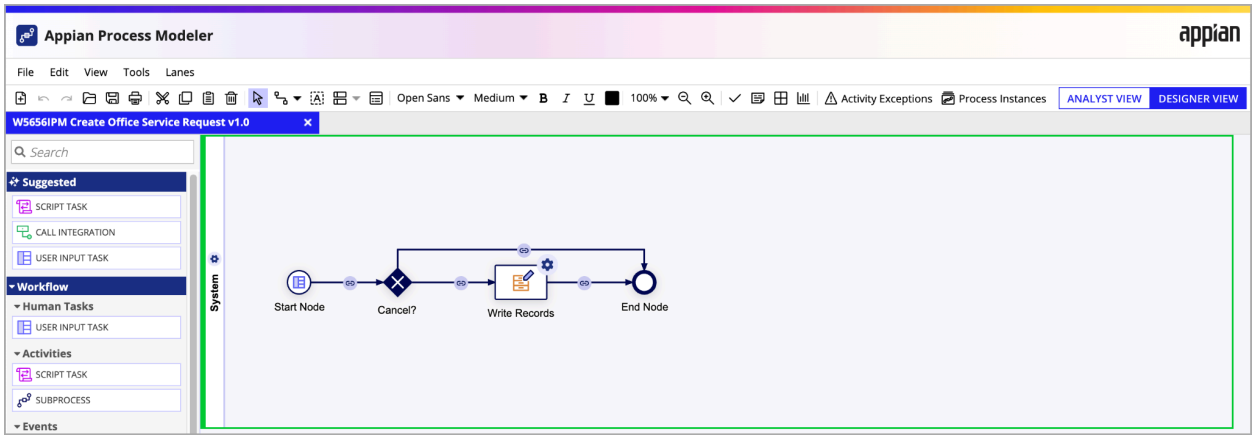
   In the **Rule Inputs** pane, notice the two rule inputs that have been created: **officeServiceRequest** and **cancel**.

2. Fill out any values for the fields in the form and click **CREATE**.

3. Expand the **officeServiceRequest** rule input to verify that all entered values have been saved to the rule input.

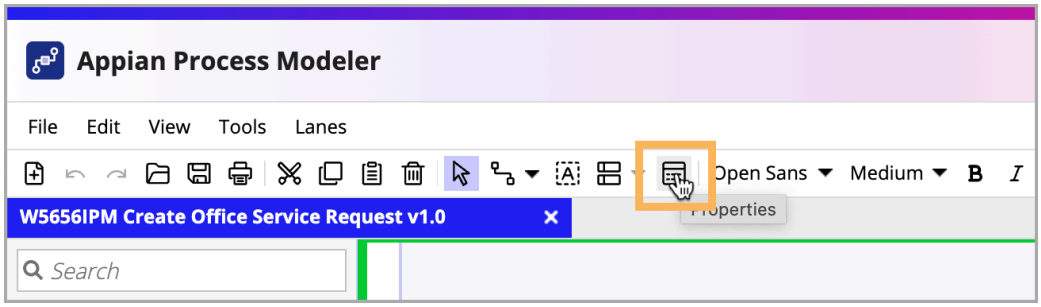


4. From the application **Build** view, open the process model **W#IPM Create Office Service Request**.

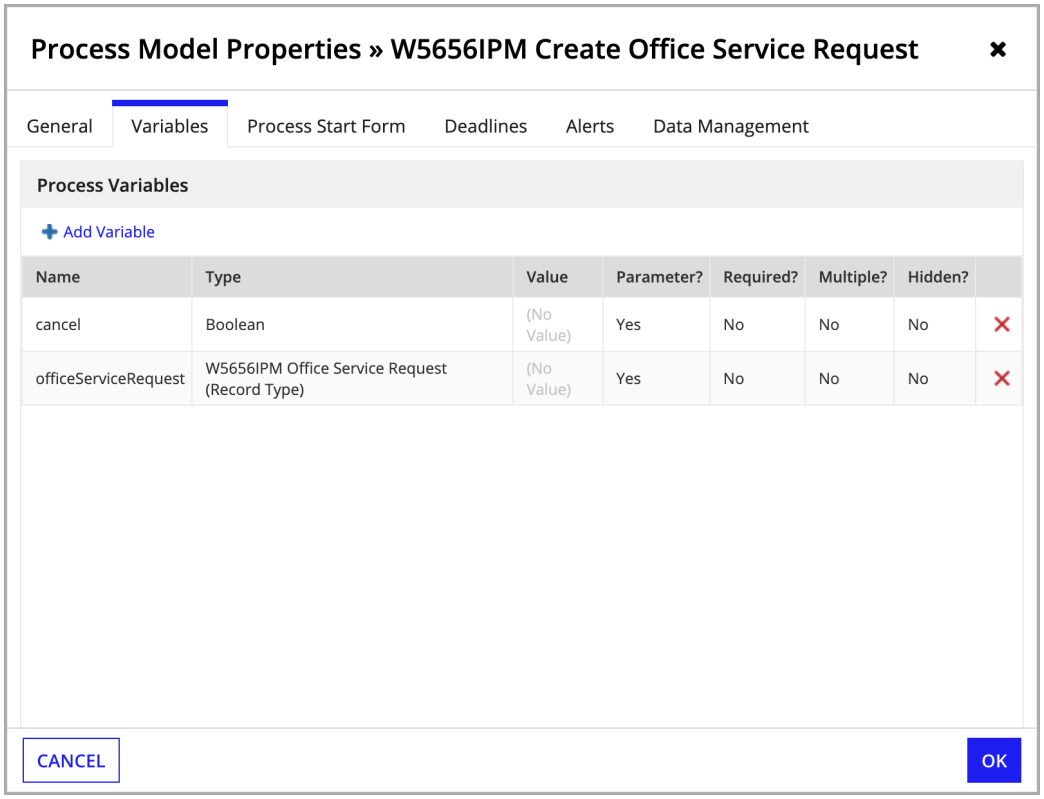


Here you can see the nodes of this simple process model.

5. From the toolbar, open the process model properties.

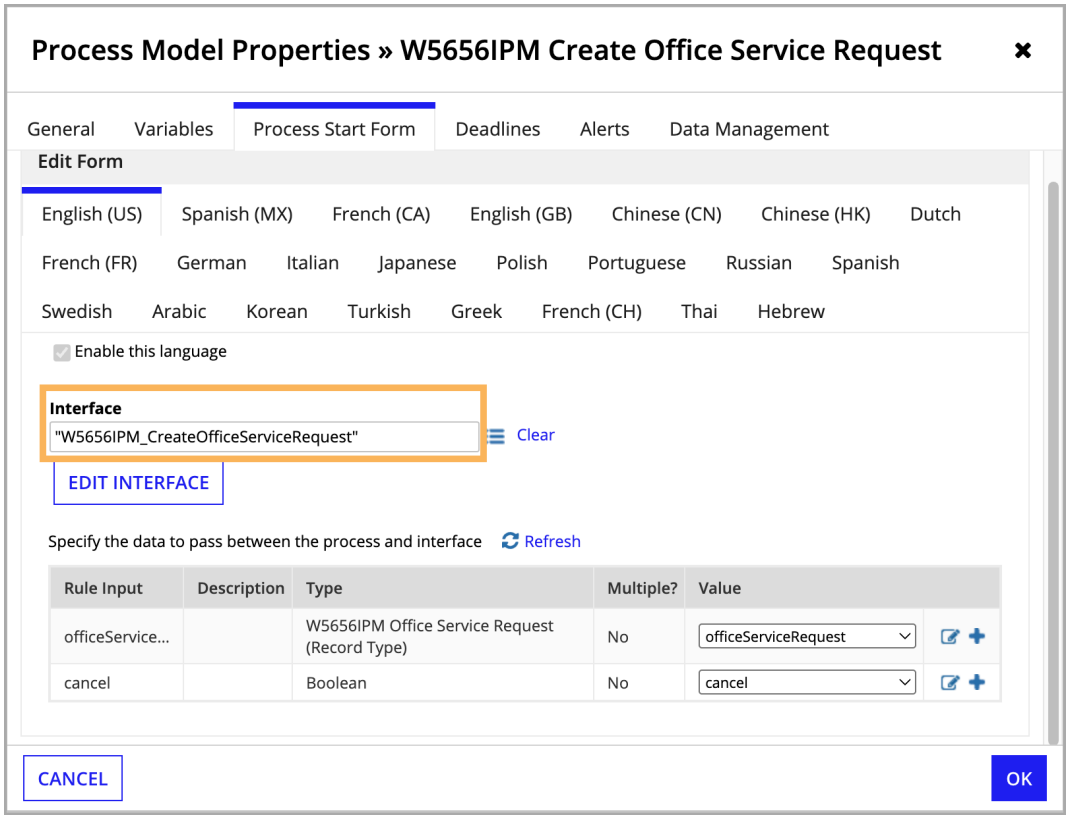


6. In the **Process Model Properties** dialog, navigate to the **Variables** tab.



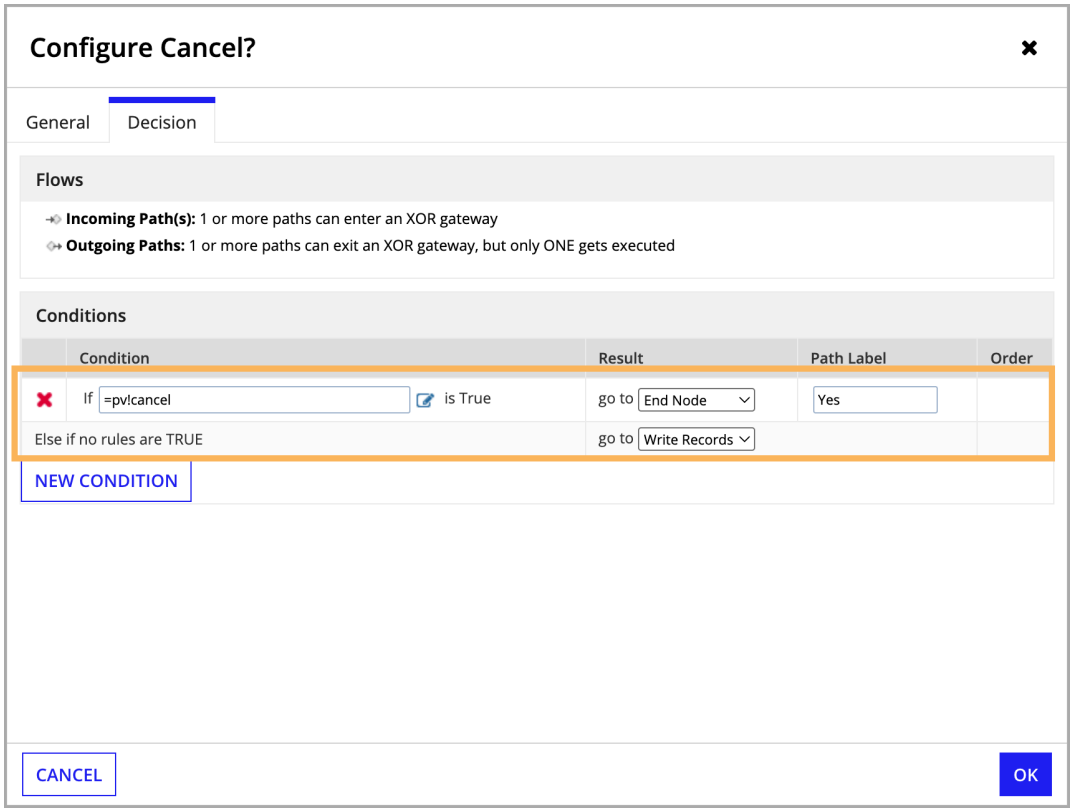
Here you can see the two process variables that were created and their data types. These align to the rule inputs on the **W#IPM_CreateOfficeServiceRequest** interface. Notice that both of these variables are marked as parameters, meaning they can accept values when the process starts.

7. Navigate to the **Process Start From** tab.



Here you can see the input form specified as well as how the rule inputs from that form map to the process variables.

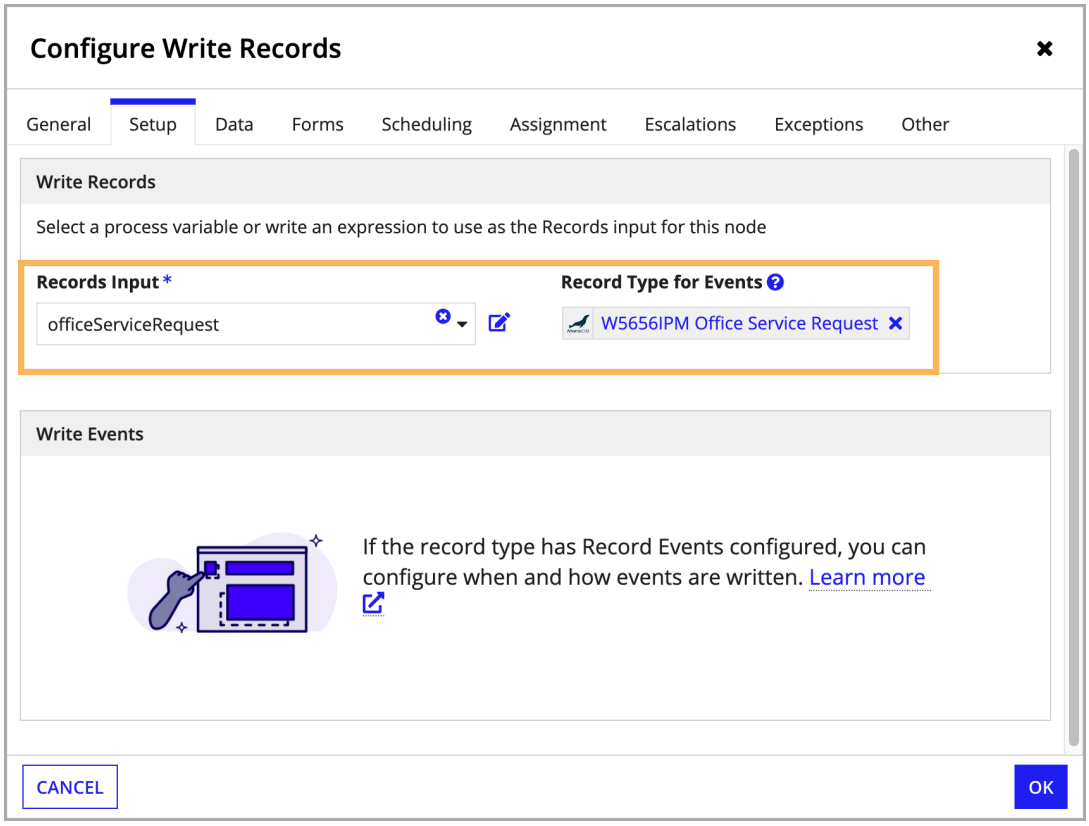
8. Click **OK** to close the dialog.
9. Double click on the **Cancel?** node in the process model and navigate to the **Decision** tab.



Here, under **Conditions** you can see that the node checks the value of the process variable **pv!cancel**. If the value is true, the process advances to the **End Node**. Otherwise, the process advances to the **Write Records** smart service.

10. Click **OK** to close the dialog.

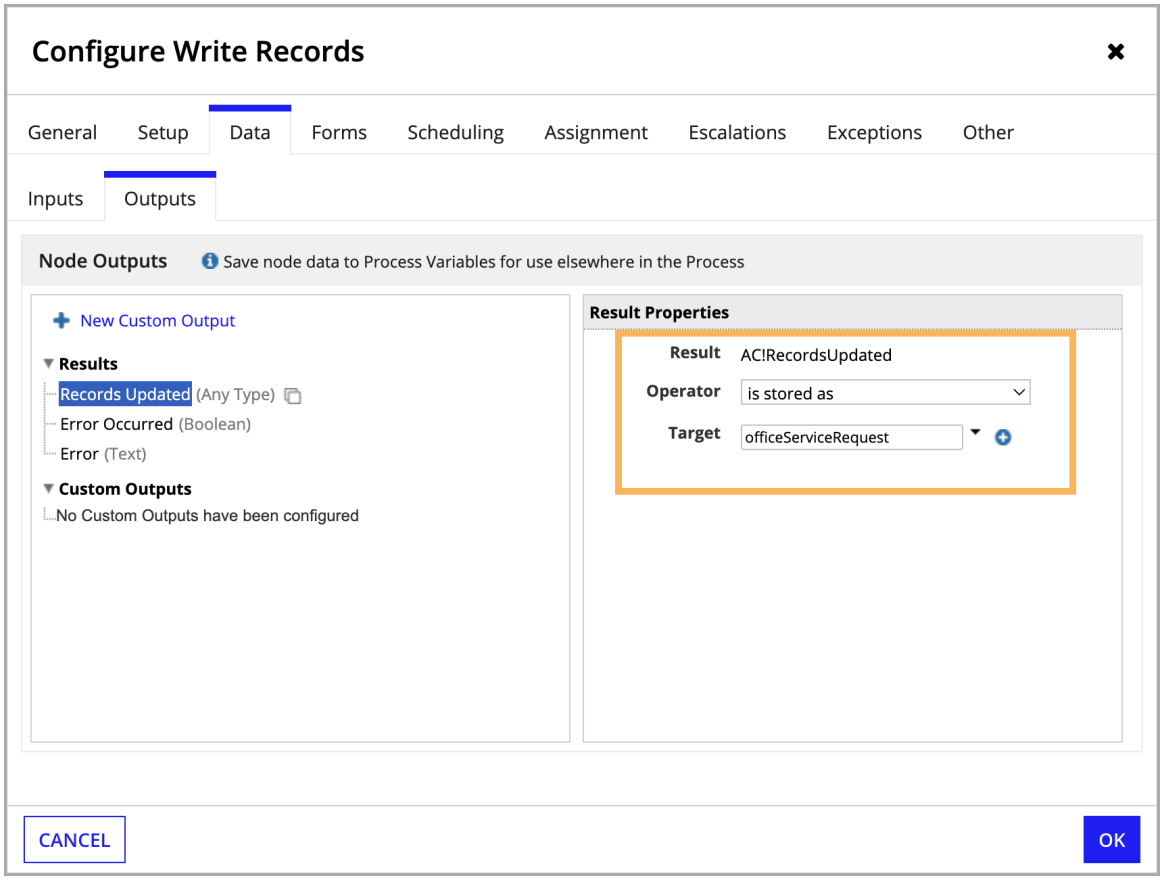
11. Double click on the **Write Records** smart service and navigate to the **Setup** tab.



Here you can see that the process variable **officeServiceRequest** is specified as the **Records Input**. The smart service writes data to the corresponding record type.

Notice that the **Office Service Request** is specified in the **Record Type for Events** field. This field is only needed if you are using the smart service to generate record events for you. In this case we're not using record events.

12. Navigate to the **Data** tab and open the **Outputs** sub-tab. Then, under **Results**, click **Records Updated**.



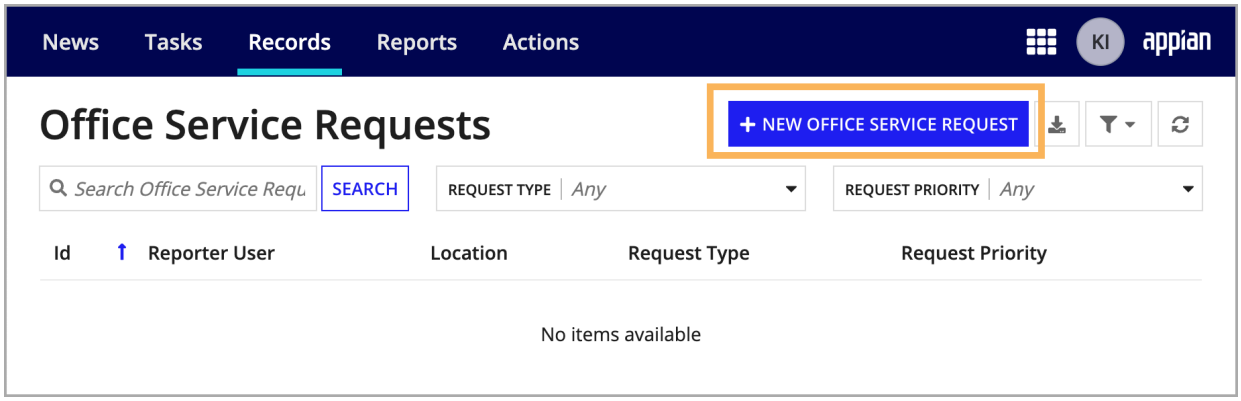
Under **Result Properties**, you can see that the activity class parameter **AC!RecordsUpdated** is saved to the process variable **officeServiceRequest**. The **RecordsUpdated** activity class parameter is built into the smart service and contains the values of the records after they have been written to the record type.

Often, when generating a new record, you will not provide a value for the primary key (notice that there was no field for id on the **CreateServiceRequest** form), but instead have the database generate a value for this field automatically. When the smart service executes, it stores the full new record (with the primary key) in the activity class parameter **AC!RecordsUpdated**. Remember though that activity class parameters are unavailable once a node completes. To make this data available to other nodes, later in the process model, it must be written to a process variable. In this case, it is written to **pv!officeServiceRequest**.

13. Click **OK** to close the dialog.

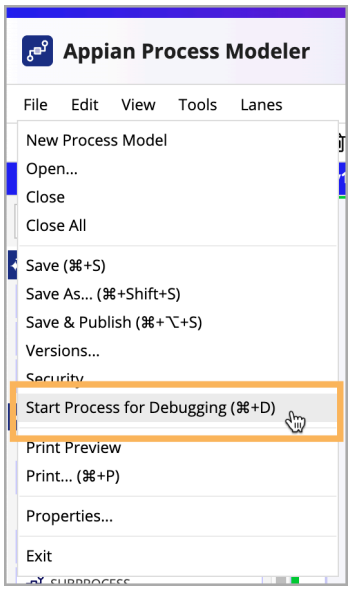
Create an Office Service Request

Now that you've seen how data moves from fields in a form into process variables to be used by process nodes, you can create an Office Service Request and see the process in action. In this simple application, there are currently two ways to do this. One is to open the record list from the **W#IPM Office Service Request** record type and click **NEW OFFICE SERVICE REQUEST**.



The other is to start the process from within the process model object itself. It's this method you will use in this exercise.

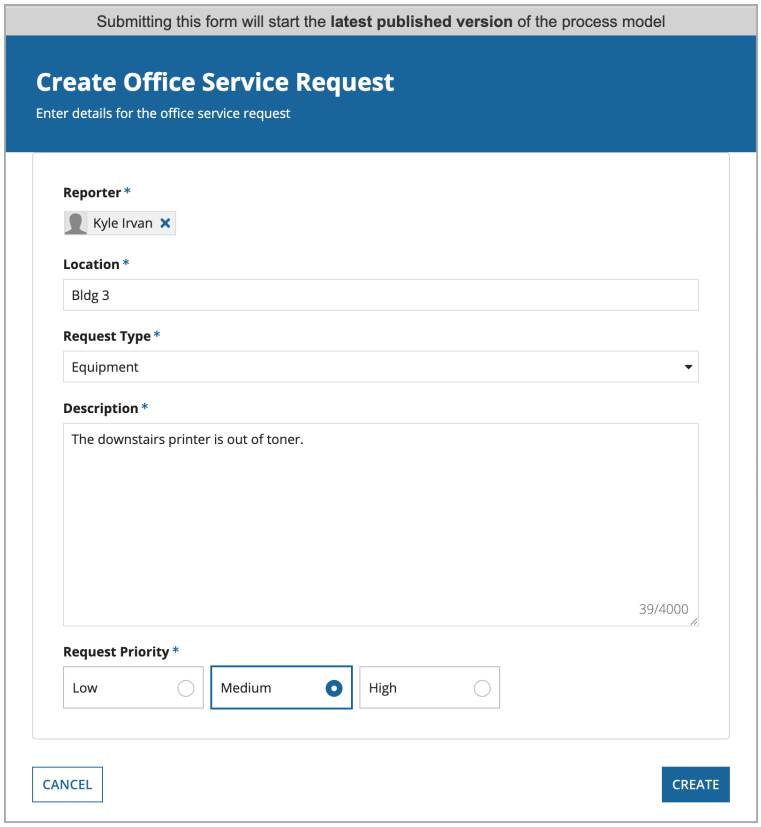
1. In the **W#IPM Create Office Service Request** process model, click **File**, then **Start Process for Debugging**.



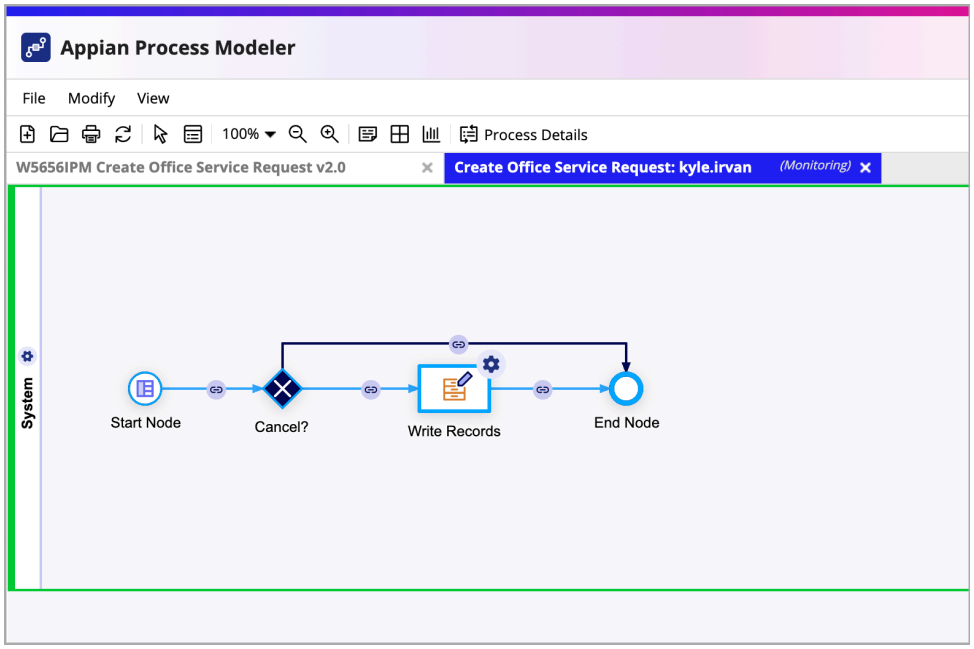
This will open a dialog window with the process start from. If you are prompted to save any changes, click **YES**.

Note: If the start form does not open, you may need to adjust your browser pop-up settings and try again.

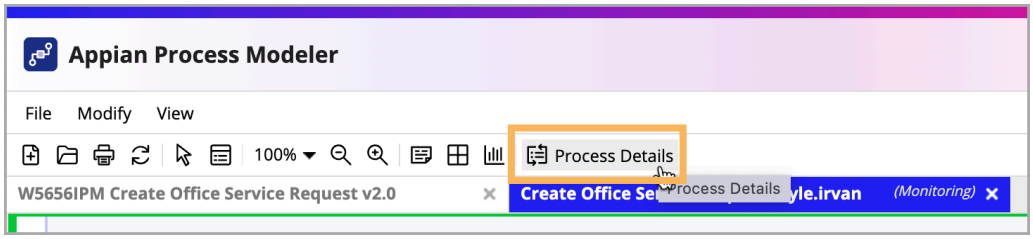
2. Fill out the form and click **CREATE**.



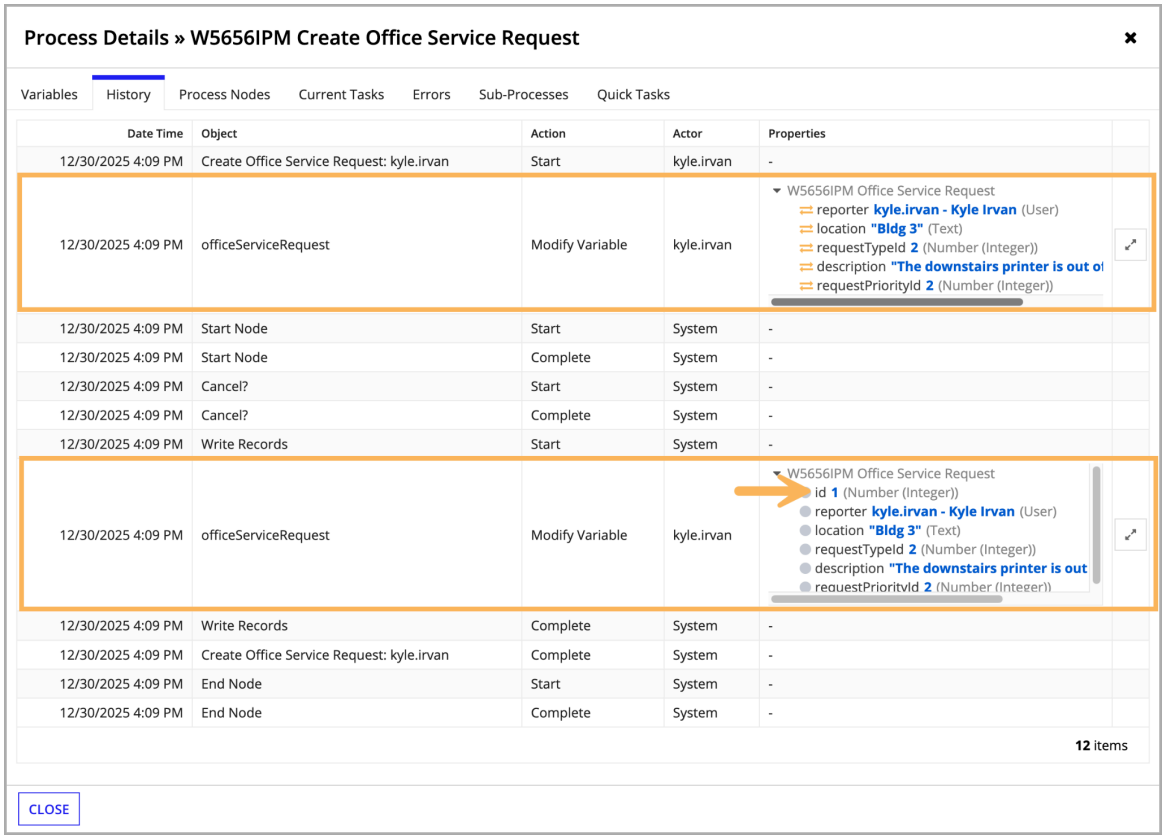
3. You will now see a new tab in the process model object. This tab represents a process instance. A process instance is a single active or completed execution of the process model. Notice how the nodes and connectors are highlighted in blue. This indicates the path that the process followed and that all nodes completed successfully.



4. From the toolbar, open the **Process Details**.

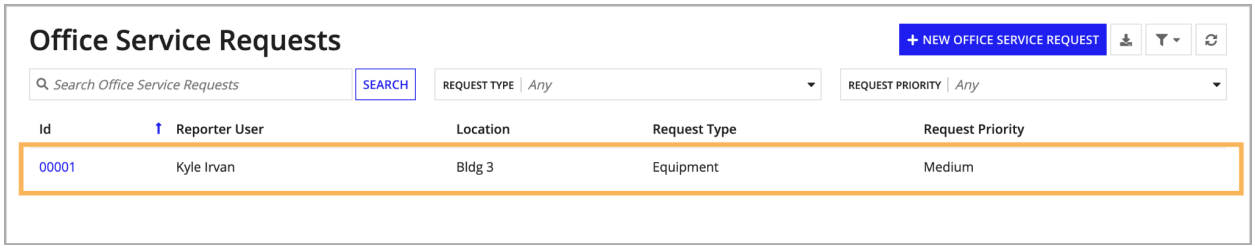


5. Navigate to the **History** tab.



Here you can see the value of the **officeServiceRequest** process variable updated twice. It is initially populated with the data you filled out in the form. Later, in the **Write Records** node, it is updated after the data has been written to the record type and assigned a value for id.

6. Navigate to the record list for **W#IPM Office Service Request**, and you will see the new record represented.

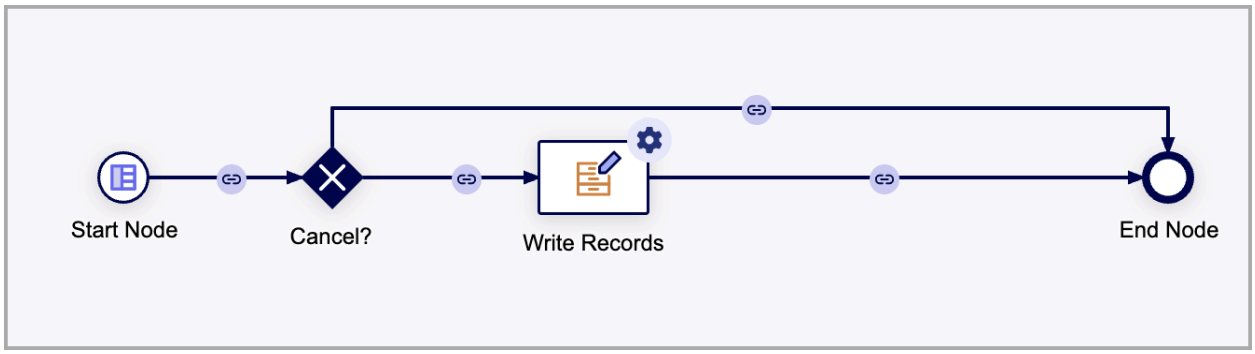


Edit a Process Model

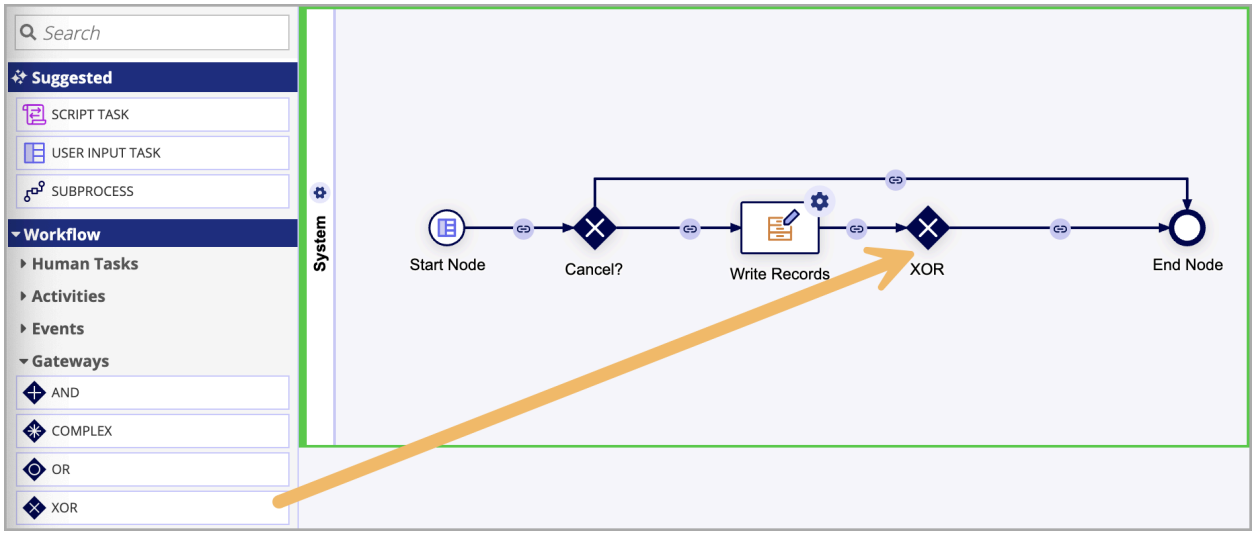
In the last exercise, you examined a process model for creating a new service request. You saw how data moved from rule inputs in the process start form to process variables in the process model. You also saw how the process variable data was used by the nodes in the process model.

In this exercise, you will edit that process model so that high priority requests will trigger an email notification to administrators.

1. Open the **W#IPM Create Office Service Request** process model.
2. Click and drag the **End Node** to the right to create space for then new nodes to be added in later steps.



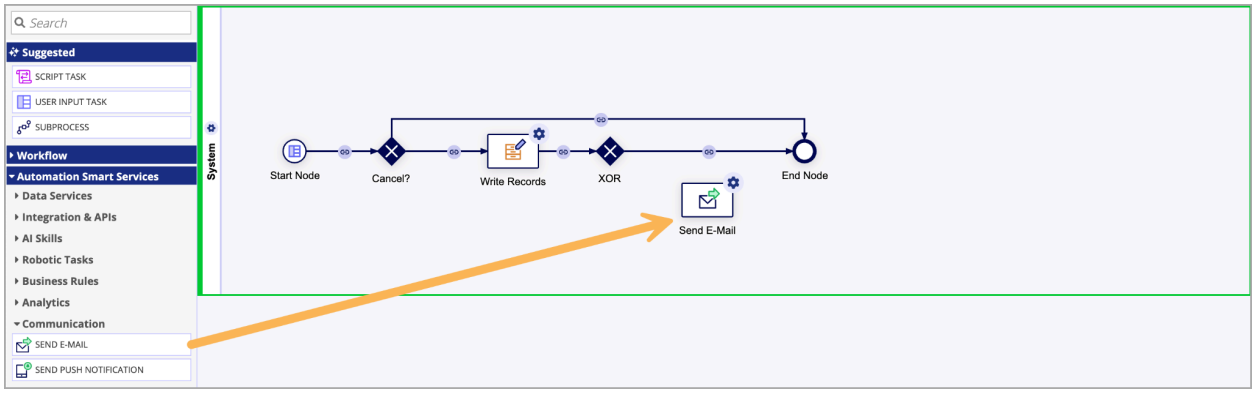
3. Click and drag an **XOR** gateway from the palette into the process model between the **Write Records** smart service and the **End Node**.



Note: When dragging a node into a process model, look at the connector before dropping the node in place. If the connector is highlighted in blue (as shown below) the node is being connected into the flow of the process model. If the connector isn't highlighted, it may look like the node has been added, but it will not be integrated into

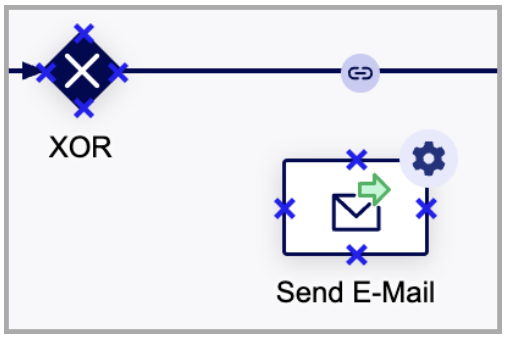
the process flow.

4. Click and drag a **Send E-Mail** smart service into an empty area below the process, between the **XOR** gateway and the **End Node**.



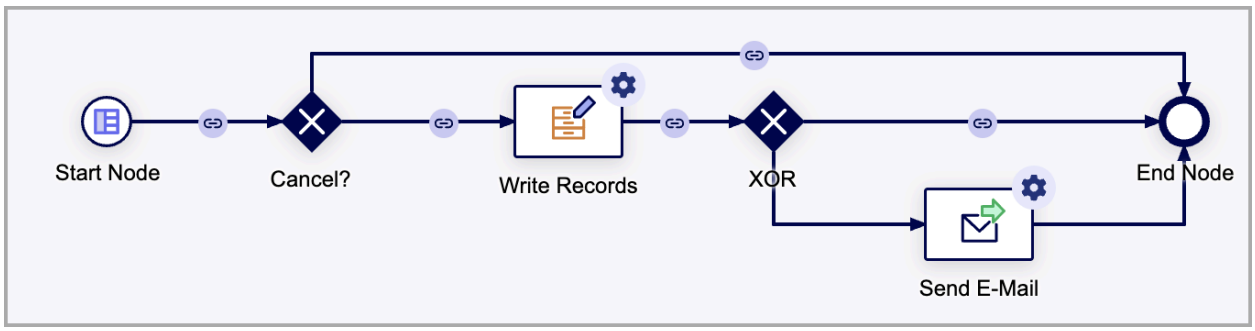
5. Add a connector from the **XOR** gateway to the **Send E-Mail** node.

   You can add connectors by either clicking the **Connect** icon in the tool bar or by holding the shift key. Either way, you will see small **X**s indicating where you can add connectors.

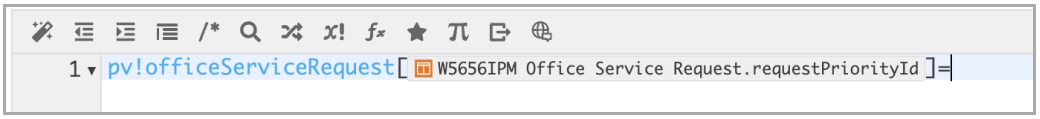


   Click and hold on an **X** on the **XOR** gateway (where the connector will start), and drag to an **X** on the **Send E-Mail** node (where the connector will end).

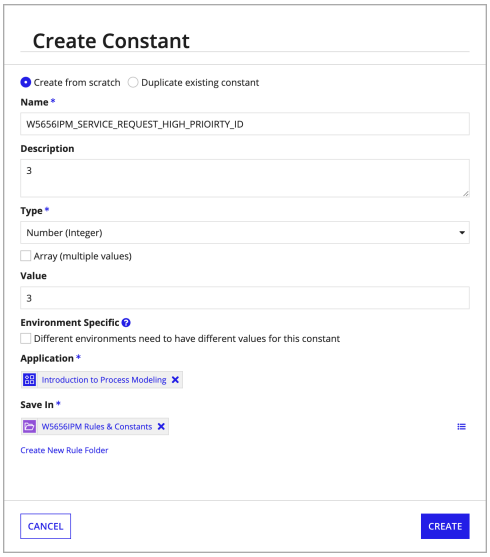
6. Repeat the process to add a connector from the **Send E-Mail** node to the **End Node**.



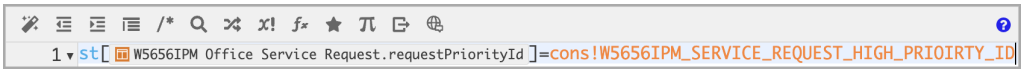
7. Double click the **XOR** gateway to configure it.
    a. On the **General** tab, change the Name to `High Priority?`.
    b. On the **Decision** tab, click **NEW CONDITION**.
    c. In the new condition line, click the expression edit icon .
    d. In the **Expression Editor** enter `pv!officeServiceRequest[W#IPM Office Service Request.requestPriorityId]`.
    You can enter this manually by typing it out, or you can expand the data pane on the left to select the correct field from the process variables. This can help avoid typos. Just expand **Process Variables**, then **officeServiceRequest**, then **requestPriorityId**.
    e. Type and equals sign `=` at the end of the expression.



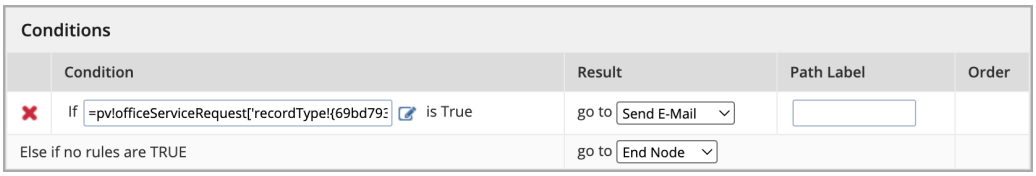
    You could type the number "3" after the equals sign to specify the id corresponding to the request priority **High** in the **W#IPM Request Priority** record type and this node would function properly. However, it is not best practice to directly enter in numerical values like this (often called "hard coding"). If, for some reason, the id for high priority were to change, this node would not behave as expected since it would depend on the value being exactly 3. For that reason, the best practice is to use a constant instead of hard-coding values.
    f. Place your cursor at the expression (after the equals sign) and click the **Create Constant** button in the toolbar above the expression and set the following values:
        - **Name**: `W#IPM_SERVICE_REQUEST_HIGH_PRIORITY_ID`.
        - **Description**: `3`.
        - **Type**: `Number (Integer)`.
        - **Value**: `3`.
        - Leave the **Application** and **Save In** fields with the default values.



g. Click **CREATE**.
h. Your new constant will now appear in your expression with the prefix `cons!`.

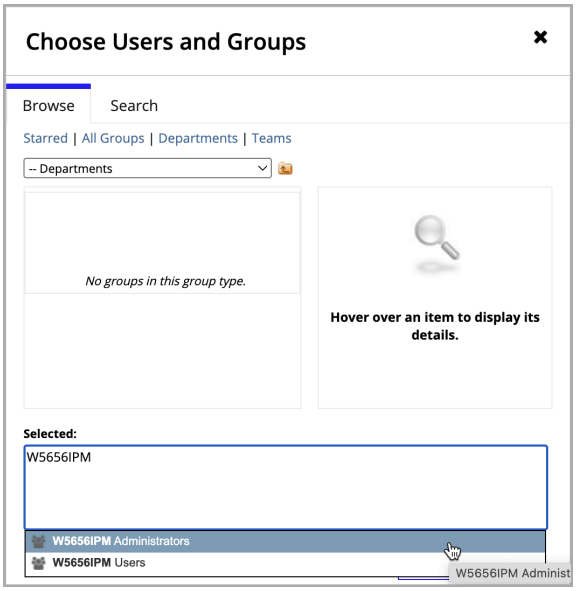


Click **SAVE AND CLOSE**.
i. Now you need to specify which node the process should go to when the request is high priority. In the dropdown, select **Send E-Mail**.
j. In the line below (Else if no rules are TRUE), in the dropdown, select **End Node**.

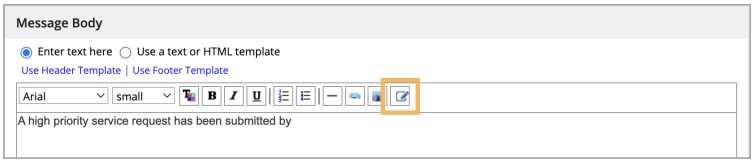


k. Click **OK**.

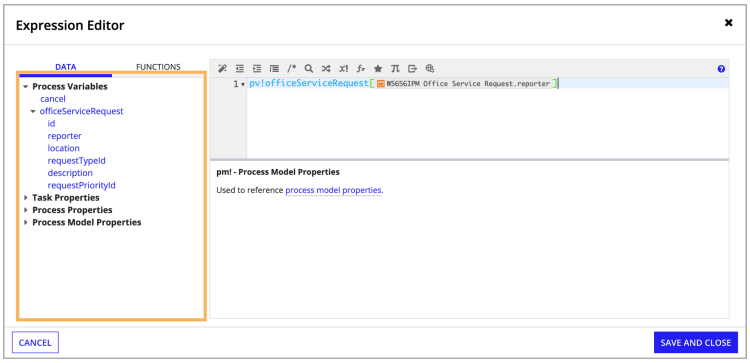
8. Double click the **Send E-Mail** node to configure it.
   a. On the **Setup** tab, under **Email Configuration**, set the following values:
      - **From**: Select `Process`.
      - **To**: Click the **Directory** icon and select the administrators group for your application. Start by typing your app prefix **W#IPM.**



      - **Subject**: Enter `High Priority Request Submitted`.
   b. Under **Message Body**, enter the following:
   `A high priority service request has been submitted by `.
   c. Click the **Insert an Expression** button in the toolbar.



d. In the **Expression Editor**, use the drill down options on the left to select the **reporter** field of the **officeServiceRequest** process variable.

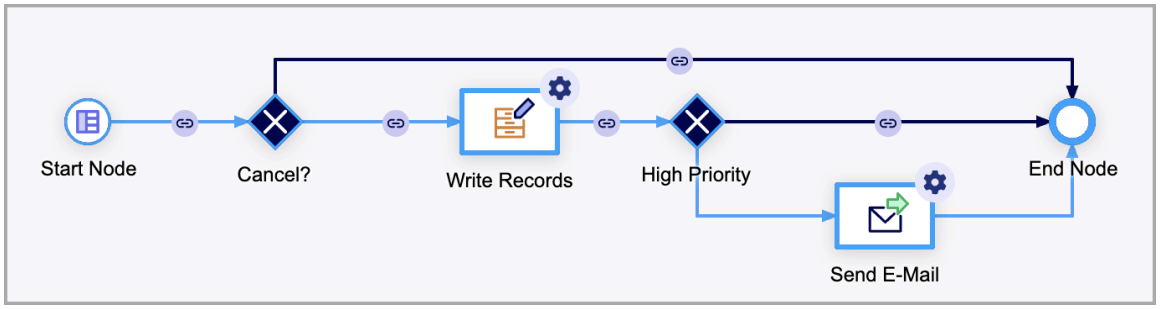


e. Click **SAVE AND CLOSE**.
f. Click **OK**.

9. In the main toolbar, select **File**, then **Save and Publish**.

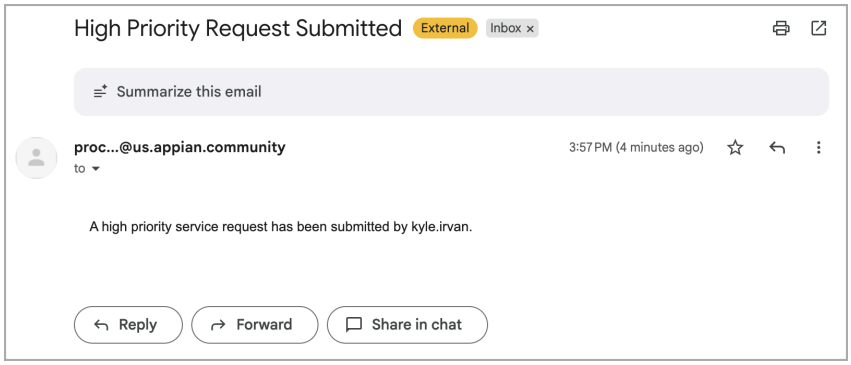
Your process model is now saved and the new version has been published. Whenever a process model is updated, it is important to test it and verify that it is working as expected.

10. In the toolbar, select **File**, then **Start Process for Debugging**.
11. Fill out the form, making sure to set the **Request Priority** to **High**, then click **CREATE**.
12. In the process instance verify that the process followed the intended path, through the **Send E-Mail** smart service.



Note: You may need to hit the **Refresh** button in the toolbar.

13. Since you are a member of the **W#IPM Administrators** group, you should also have received an email message. Check the inbox associated with your Appian Community account.



Note: You may need to check your spam folder.

14. Close the process instance tab.
15. Start the process for debugging again, this time with the **Request Priority** set to either **Low** or **Medium**.
16. Check the process followed the expected path in the process instance.