|  |  |
| --- | --- |
|  |  |
| A green and white sign  Description automatically generated with low confidence | TraininG |

## Alfresco Process Services

## Building a Claims Process

### lab workbook (attendee)

­­

# Table of Contents

|  |  |
| --- | --- |
| Lab 1. Create an Intake Task | Page 3 |
| Lab 2. Create a Process Application | Page 6 |
| Lab 3. Testing Your Intake Process | Page 7 |
| Lab 4. Generating and saving a document | Page 8 |
| Lab 5. Create a data model | Page 9 |
| Lab 6. Create a Store Entity Task / Save Values to Database | Page 10 |
| Lab 7: Create a Lookup Process | Page 11 |
| Lab 8: Create a Service Task w/ Java Delegate | Page 13 |
| Lab 9: Add a Parallel Gateway and Splitting Paths | Page 14 |
|  |  |
|  |  |

## Class Prework:

* **\*Only do this IF** you did not sign into the share connector on each user account.
* Select the **Identity Management** tile.
* Scroll to the bottom of profile and select the **alfresco-1** repository.
* Sign in with provided credentials.

|  |  |
| --- | --- |
| **Scenario – 9 Second Insurance: New Customer Process** |  |
| As a systems architect working for 9 Second Insurance, you are tasked with creating a process that will allow agents to create reimbursement claims. Your process will include:   * Data intake mechanism to capture claim information. * Data review mechanism to determine accuracy of information. * Document generation and saving to content management system. * A REST call to retrieve customer values from a custom database. * A Decision Table to perform logic and edit values. | |

## Lab 1. Create an Intake Task

1. From the Alfresco home page, launch the **Activiti App** (Process Services) by clicking on the **Activiti App** hyperlink.
   1. Sign in with your provided username and password. You will be directed to the Activiti App home page.
2. Select the **App Designer** tile to navigate to the **Business Process Models** page.
3. To create a new process, press the **Create Process** button in the top right of the page.
4. In the **Create a new business process model** popup, enter the following information:
   1. **Model name:** [Your user #] New Claim (*ex:* ***U1 New Claim***)
   2. **Description**: Create a new claim.
   3. **Editor Type**: Leave as BPMN editor
   4. **Stencil**: Default BPMN
   5. Press the **Create new model** button.
5. With nothing selected on the stage, select the **Variables** attribute in the bottom configuration panel.
6. Select the “+” icon below the chart to add new variables. We’ll add a few variables that will be used in this process. Create the following variables with this configuration:

|  |  |
| --- | --- |
| **Variable Name** | **Variable Type** |
| cAddress | String |
| cCity | String |
| cFirstName | string |
| cLastName | String |
| cState | String |
| cZip | String |
| deductible | integer |
| lu\_lastname | string |
| recordCount | integer |
| recordList | string |

1. Create a new **User Task** connected to the start event. 
2. Give the user task a name by double-clicking on the task to open a text field. Name this task ***Customer Search***.
3. With the user task selected, notice that the **Referenced form** value in the bottom configuration window is *No reference selected*. To create an intake form for this task, click on the *No reference selected* value.
4. In the **Form reference** popup window, select the **New Form** button.
5. In the **Create a new form** window, enter the following values:
   1. **Form name:** Customer Lookup
   2. **Description:** Gather customer information to retrieve records.
   3. **Stencil:** Default form
   4. Select the **Create form** button.
6. Follow these steps to create the form you’ll need to intake a new hire employee.
   1. From the left object menu, drag a **Header** onto the canvas. To edit, click on the pencil icon that appears when you hover your mouse over the header object. In the **Label** field, name it ***Enter the Customer’s last name*** and click the **Close** button.
   2. Drag a **Text** object and dop it into the Header object.
   3. Click on the pencil icon on the Display Value object to open the edit prompt. Configure the text field with the following options:
      1. **Label:** Last Name:
      2. **Override ID:** check the box
      3. **ID:** lookup\_lastname
      4. **Required:** check the box
      5. *Close the edit prompt.*
   4. Save and close the form editor by clicking on the save button. 
   5. On the **Save form** popup window, click the **Save and close editor** button to return to your process model.
7. Create a new script task connected to the start event. In the bottom configuration panel, configure the following attributes:
   1. **Name:** Get DB Values
   2. **Script format:** groovy
   3. **Script:** *enter this code in the popup script window:*

|  |
| --- |
| import groovy.sql.Sql;  import groovy.json.\*;  import groovy.json.JsonBuilder;  import com.activiti.security.SecurityUtils;  class Record {  String recId  String firstname  String lastname  String address  String city  String state  String zip  }  def ln = execution.getVariable("lookup\_lastname");  execution.setVariable("lu\_lastname", ln);  def url = 'jdbc:oracle:thin:@//aps-custom-oracle-db.cp58lgpzkwpy.us-east-1.rds.amazonaws.com/ORCL';  def user = 'admin';  def password = 'administrator';  def driver = 'oracle.jdbc.driver.OracleDriver';  def sql = Sql.newInstance(url, user, password, driver);  rowNum = 0;  def recordList = [];  sql.eachRow("SELECT ID, FIRSTNAME, LASTNAME, ADDRESSLINE1, CITY, STATE, ZIPCODE FROM CUSTOMERS WHERE LASTNAME = ${lu\_lastname}") { row ->    def r = new Record( recId: row.id, firstname:row.firstname, lastname:row.lastname, address:row.addressLine1, city:row.city, state:row.state, zip:row.zipcode)  execution.setVariable("cFirstName", row.firstname);  execution.setVariable("cLastName", row.lastname);  execution.setVariable("cAddress", row.addressLine1);  execution.setVariable("cCity", row.city);  execution.setVariable("cState", row.state);  execution.setVariable("cZip", row.zipcode);  recordList.add(r);    }  execution.setVariable("recordCount", recordList.size);  println new JsonBuilder( recordList ).toPrettyString();  execution.setVariable("recordList", new JsonBuilder( recordList ).toPrettyString()); |

1. Create a new **User Task** and connect it to the script task. Name it **Display DB Values**.
2. In the bottom configuration panel, select the **Referenced Form** attribute.
3. In the form popup window, select the **New Form** button.
4. In the **Create a new form** window, enter the following values:
   1. **Form name:** Display DB Values
   2. **Description:** Displays values from customer DB lookup.
   3. **Stencil:** Default form
5. Select the **Create form** button.
6. Follow these steps to create the form:
   1. From the left object menu, drag a **Header** onto the canvas. To edit, click on the pencil icon that appears when you hover your mouse over the header object. In the **Label** field, name it ***Query*** and click the **Close** button.
   2. Drag a **Display Text** object and dop it into the Header object. Select the pencil icon to edit.
      1. In the **Text to display** field enter the following text:

|  |
| --- |
| Found ${recordCount} records with the Last Name of "${lu\_lastname}". |

* + 1. Close the edit text field prompt.
  1. From the left panel drag a **Dynamic Table** onto the stage adding it below the Header object. Click on the pencil icon to edit.
  2. Give it a Label of: **CUSTOMERS**
  3. Select the **Override ID** check box.
  4. Give it an ID of: **recordList**
  5. Select the **Table Columns** tab at the top of the prompt.
  6. Press the “+” icon button to create new property mappings with the following values (**Note:** for each property check all of the boxes ***Required, Editable, sortable, show in table***):
     1. Column 1:
        1. **Property ID:** recId
        2. **Property Name:** ID
        3. **Property Type:** string
     2. Column 2:
        1. **Property ID:** firstname
        2. **Property Name:** First name
        3. **Property Type:** string
     3. Column 3:
        1. **Property ID:** lastname
        2. **Property Name:** Last Name
        3. **Property Type:** string
     4. Column 4:
        1. **Property ID:** address
        2. **Property Name:** Address
        3. **Property Type:** string
     5. Column 5:
        1. **Property ID:** state
        2. **Property Name:** State
        3. **Property Type:** string
     6. Column 6:
        1. **Property ID:** city
        2. **Property Name:** City
        3. **Property Type:** string
     7. Column 7:
        1. **Property ID:** zip
        2. **Property Name:** Zip
        3. **Property Type:** string
  7. Close the edit prompt.
  8. From the left panel drag a **Display text** object onto the stage, adding it below the dynamic table. Select the pencil icon to go into edit mode:
     1. In the **Text to display** field, add the following text:

|  |
| --- |
| Verify that this is the correct customer and proceed.  Press Back if customer does not appear. |

* + 1. Close the edit text field prompt.
  1. Save and close the Form editor and return to your process.

1. Add a connected end event to the process, connected to the *Display DB Values* task. 
2. Save the process model by clicking on the **Save** icon in the top left of the page. 
3. In the **Save model** popup window, press the **Save and close editor** button.

|  |  |
| --- | --- |
| **Scenario – Process Application** |  |
| To deploy and place your processes into use, you need to create a Process Application. A process application can deploy multiple processes and are usually grouped by function. Because an insurance company might have more than one claims process, it would make sense to place all of your claims related processes into one Process Application titled ***9SecondInsurance***. Let’s create that now. | |

## Lab 2. Create a Process Application

1. **Note:** If you already have a Process Application that you’d like to add your new process to, then skip this lab and perform **Lab 2a** instead.
2. From the Activiti home page, select the **App Designer** tile to navigate to the **Business Process Models** page.
3. Select the **Apps** hyperlink in the top blue banner.
4. Select the **Create App** button.
5. Give your application the App definition name: *[Your User #]* **9SecondInsurance** (*ex:* ***U1 9SecondInsurance***)*.*
   1. **Optional**: enter a description of your application describing it will do, i.e.: **Deploys our claims processes**.
6. Click the **Create new ap definition** button.
7. **OPTIONAL**: On the **App definition details** page, you can change the color and icon of your application tile. Use the **Icon** and **Theme** drop downs to customize your tile.
8. To add the process model you created, click on the **Edit included models** button below the tile.
9. In the **Models included in app definition** popup window, select the process model you want to include. **NOTE:** The selections toggle on and off when you click them, so be sure to click the one you want **once** and notice the blue + icon will appear to ensure it is selected. A blue square with a white cross in the center

   Description automatically generated with medium confidence
10. Click the **Close** button to close the popup window.
11. Save the App definition by clicking on the save icon located in the top left of the page. 
12. In the **Save app definition** popup window, ensure you select the **Publish?** Checkbox, then click on the **Save and close editor** button.
13. Navigate back to the home page by clicking the home button in the top, blue banner: 
14. If your application already appears on the home page as a tile you are done.
    1. **If not**, deploy your new application by selecting the blank tile, depicted with a plus sign “+” (“Add a new app” appears when you hover your mouse over it. Select this tile, then select your application in the **Add app to landing page** popup window. Press the **Deploy button** on that window. Your application is now deployed.

## Lab 2a. Add Claims Process to Application

1. **Perform this lab if you already have a Process Application and need to add your new process.**
2. From the Activiti home page, select the **App Designer** tile to navigate to the **Business Process Models** page.
3. Select the **Apps** hyperlink in the top blue banner.
4. Select the process application (in edit mode by using the pencil icon in the top, right corner of the tile) that you want to add your model to.
5. Select the **Edit Included** **Models** button.
6. In the Models included popup window, select the process you want to add. **Note:** a selected process will show a blue “+” icon in the corner of the thumbnail. Close the popup window once your process is selected.
7. Click the save icon to save the App definition, then publish your application.

## Lab 3. Testing Your New Hire Intake Process

1. From the Alfresco home page, launch the Alfresco Digital Workspace and sign in with your provided credentials.
2. To start your process, select the **Start Process** button found in the top right of the page.
3. In the **New Process** popup window, select your **9SecondInsurance** application from the Application selector.
4. Select your **New Claims** process.
   1. OPTIONAL: You can change the name of the process that will run by editing the **Process Name** field.
5. Click on the **START PROCESS** hyperlink at the bottom right corner of the page.
6. Select **My Tasks** found under the **Workflow** dropdown on the left side of the page.
7. Your **Customer Search** task should appear. Click on it to perform the task.

## Lab 4. Creating a Form Outcome with Exclusive Pathing

1. Access the App Designer tile from the homepage of the Activiti App (Process Services).
2. Enter your *New* *Claims* process in **edit mode** by selecting the edit icon when hovering your mouse over its tile. 
3. Select the *Display DB Values* task and open the **Referenced Form** in edit mode.
4. **Add an Outcome to the Review Form:**
   1. Select the **Outcomes** tab from the top of the form.
   2. Select the **Use form outcomes for this form** radio button.
   3. Under **Possible outcomes**,enter a new outcome in the provided field: **Proceed**.
   4. Press the **Add outcome** button and enter another outcome: **Back**.
      1. **Note:** Do not press the add outcome button again or you will add a third option.
   5. Press the **Save** button and close the form.
5. Remove the **End event** and the line connecting to it by selecting each one and clicking the trash can icon that appears.
6. Add an **Exclusive Gateway** task to the process found under the **Gateways** drop down.
7. Connect the gateway task to the *Display DB Values* task.
8. With the gateway task selected, click and drag a **Sequence flow line** back to the *Customer Search* task. Your process should now look something like this:

A diagram of a flowchart

Description automatically generated

1. Select the **Sequence flow line** that routes back to the *Customer Search* task. Click the value to open the **Sequence flow condition** popup window.
   1. Select **Simple** as the condition type.
   2. Select **Form outcome** as the Depends on selector.
   3. In the first dropdown menu, select the form that you added the form outcome to: the *Display GB Value* form.
   4. In the second dropdown menu, select **equal**.
   5. In the last dropdown menu, select **Back**.
   6. Select the **Save** button on the popup window.
2. Select the exclusive gateway event, then select the **User Task** icon to create a new user task stemming from the gateway event.
3. Select the **Sequence flow line** that **ends** at the **User Task.** In the bottom configuration panel, check the check box for the **Default flow** attribute. This will signify that the default traffic will go this route.
4. Select the new user task and give it a name of: **Create Claim**.
5. Select the **Referenced form** attribute from the bottom configuration panel, open the referenced form popup window and choose the **New Form** button.
6. Give the form a name of: **New Claim** and select the **New Claim button.**
7. In the form editor, perform the following steps to create a new claim form:
   1. Add a Header object to the page and select the pencil icon to go into edit mode. Give it a label of **Customer Information:** Close the edit popup.
   2. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** First Name:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cFirstName** variable.
      4. Close the edit popup window.
   3. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** Last Name:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cLastName** variable.
      4. Close the edit popup window.
   4. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** Address:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cAddress** variable.
      4. Close the edit popup window.
   5. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** City:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cCity** variable.
      4. Close the edit popup window.
   6. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** State:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cState** variable.
      4. Close the edit popup window.
   7. Add a **Display Value** object into the Customer Information Header object. Give it the following configuration:
      1. **Label:** Zip Code:
      2. Select the **Variable** button.
      3. In the drop-down menu select the **cZip** variable.
      4. Close the edit popup window.
   8. Add another Header object to the page and select the pencil icon to go into edit mode. Give it a label of **Claim Information:** Close the edit popup.
   9. Add a **Date** object to the Claim Information header and go into edit mode. Give it the following configuration:
      1. **Label:** Incident Date:
      2. **Required:** checked
      3. Select the **Advanced** tab.
      4. Enter **MM-DD-YYYY** into the Date display format field.
      5. Close the edit prompt.
   10. Add a **Dropdown** object to the Header and go into edit mode. Give it the following configuration:
       1. **Label:** Incident Date:
       2. **Required:** checked
       3. Select the **Options** tab. Enter the following configuration:

|  |  |
| --- | --- |
| **Label** | **ID** |
| Damage | Damage |
| Lost | Lost |
| Stolen | Stolen |
| Other | Other |

* + 1. Close the edit prompt.
  1. Add an **Amount** object to the Header and go into edit mode. Give it the following configuration:
     1. **Label:** Claim Amount:
     2. Select the **Override ID** checkbox.
     3. **ID:** value
     4. Check the **Required** checkbox.
     5. Close the edit prompt.

1. Save and close the form editor and return to your process.
2. Add an end event to your process connected to the *Create Claim* task.
3. Save and close your process.
4. Navigate to the Apps page and republish your application.
5. Open the Digital Workspace and test your updated process.

## Lab 5. Adding Decision Logic (Decision Table)

1. Access the App Designer tile from the homepage of the Activiti App (Process Services).
2. Enter your *New* *Claims* process in **edit mode** by selecting the edit icon when hovering your mouse over its tile. 
3. Delete the end event from the process.
4. From the left panel, add a **script** task to the process and connect it in place of the end event deleted in the previous step. Using the bottom config panel, give the task the following configuration:
   1. **Name:** Assign Deductible
   2. **Script Format:** groovy
   3. Enter the following text into the **Script** popup window:

|  |
| --- |
| **execution.setVariable("deductible", 200);** |

1. From the left panel, drag and place a **Decision Task** after the Script task and connect it.
2. Name the Decision task: **Configure Deductible**
3. In the bottom configuration panel, select the **Referenced decision table** attribute to open the decision table popup window. Select the **New Decision Table** button.
4. In the top blue header, click the **[** **Undefined ]** title, which opens the Edit input popup. Configure the popup with the following information:
   1. **Column Label:** Equipment Value
   2. **Variable Type:** Form field
   3. **Form Field:** Claim Amount – value
   4. Save the popup.
5. In the top green header, click the **[ Undefined ]** title, which opens the Edit output popup. Configure the popup with the following information:
   1. **Column label:** Deductible
   2. **Column type:** Existing
   3. **Variable Type:** Variable
   4. **Variable:** deductible
   5. Save the popup.
6. In the blue cell underneath the blue header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
   1. **Operator:** Less than
   2. **Variable type:** Number
   3. **Number:** 100
   4. Click **OK.**
7. In the green cell underneath the green header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
   1. **Variable type:** Variable
   2. **Variable:** deductible
   3. **Method:** Subtract
   4. **Number:** 200
   5. Click **OK.**
8. Create a new Rule by pressing the **Add Rule** button.
9. In the second blue cell underneath the blue header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
   1. **Operator:** Less than
   2. **Variable type:** Number
   3. **Number:** 500
   4. Click **OK.**
10. In the second green cell underneath the green header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
    1. **Variable type:** Variable
    2. **Variable:** deductible
    3. **Method:** Subtract
    4. **Number:** 150
    5. Click **OK.**
11. Create a new Rule by pressing the **Add Rule** button.
12. In the third blue cell underneath the blue header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
    1. **Operator:** Less than
    2. **Variable type:** Number
    3. **Number:** 1000
    4. Click **OK.**
13. In the third green cell underneath the green header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
    1. **Variable type:** Variable
    2. **Variable:** deductible
    3. **Method:** Subtract
    4. **Number:** 100
    5. Click **OK.**
14. Create a new Rule by pressing the **Add Rule** button.
15. In the fourth blue cell underneath the blue header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
    1. **Operator:** Greater than or equal
    2. **Variable type:** Number
    3. **Number:** 1000
    4. Click **OK.**
16. In the fourth green cell underneath the green header, select the pencil icon to open the Edit rule expression popup window. Configure with the following information:
    1. **Variable type:** Variable
    2. **Variable:** deductible
    3. **Method:** Add
    4. **Number:** 200
    5. Click **OK.**
17. Save and close the decision table to return to your process.
18. Add a new **User task** to your process and connect it to the decision table task.
19. Name the user task: **Verify Claim**
20. Select the **Referenced form** attribute from the bottom configuration panel, open the referenced form popup window and choose the **New Form** button.
21. Give the form a name of: **Verify Claim** and select the **New Claim button.**
22. In the form editor, perform the following steps to create a new claim form:
    1. Add a Header object to the page and select the pencil icon to go into edit mode. Give it a label of **Verify Claim Information:** Close the edit popup.
    2. Add a **Display Text** field to the Header and select the pencil icon to go into edit mode.
    3. In the **Text to display** field, enter the following text:

|  |
| --- |
| Mr. or Ms. ${cLastName}, your claim has been submitted with the peril type of ${incidenttype}.  The approximate claim value is $${value}. Due to this amount the cost of your deductible will be $${deductible}.  Thank you for being a loyal customer of 9 Second Insurance! |

* 1. Close the prompt.

1. Save and close the form editor to return to your process.
2. Save and close the process editor.
3. Navigate to the applications page and publish your application.
4. Test the new process changes.
   1. The decision table logic should scale the deductible down based on the value entered into the claim form. Test by submitting different values.

------------------------------------------|| THIS IS THE END OF THE LABS ||---------------------------------------

## Publishing & Testing Your Edited Process Application

1. **NOTE:** Before you can test your edited process, you must **republish** your application. Follow these steps now and in the future when you need to republish your edited models.
   1. Select the **Apps** button in the top blue banner.
   2. Select your application tile.
   3. Click the **publish** button in the top right of the page.
   4. Click the red **Publish app definition** button on the popup window.
2. Refer to Lab 5 steps in order to test your edited process. Your process should now be creating and saving a document to your Content Services site. One you have completed the new process, perform these steps to ensure the new document functionality is working.
   1. If not already open, launch the Digital Workspace from the Alfresco home page and sign in.
   2. From the left navigation menu, select the **File Libraries** drop-down menu, then select **My Libraries**.
   3. Choose the site you created for this class, then choose the **HR Documents** folder you created for this process.
   4. You should have a saved document (or multiples if you completed your process multiple times) in this folder.

## Testing Your Process Application

At any time, you may refer to these steps in order to test your process application. **Note:** If you have made edits to any models that are deployed in your application, you need to re-publish your application.

* Re-Publishing your application:
  1. From the home / landing page, navigate to the App Designer by selecting its tile.
  2. Navigate to the Apps page by clicking on the **Apps** button in the top, blue banner.
  3. Find and click on your process application tile; do not select the edit icon, just the tile.
  4. Select the **Publish** button found in the top, right corner of the page.
  5. Select the red **Publish app definition** button on the Publish pop-up window.
* Testing your application:
  1. Follow the steps under **Testing Your Process Application** found on page 12.

## Alfresco Process Services

Lab Workbook (Attendee)