Appian
Academy

Exercise 5: Create a Wizard

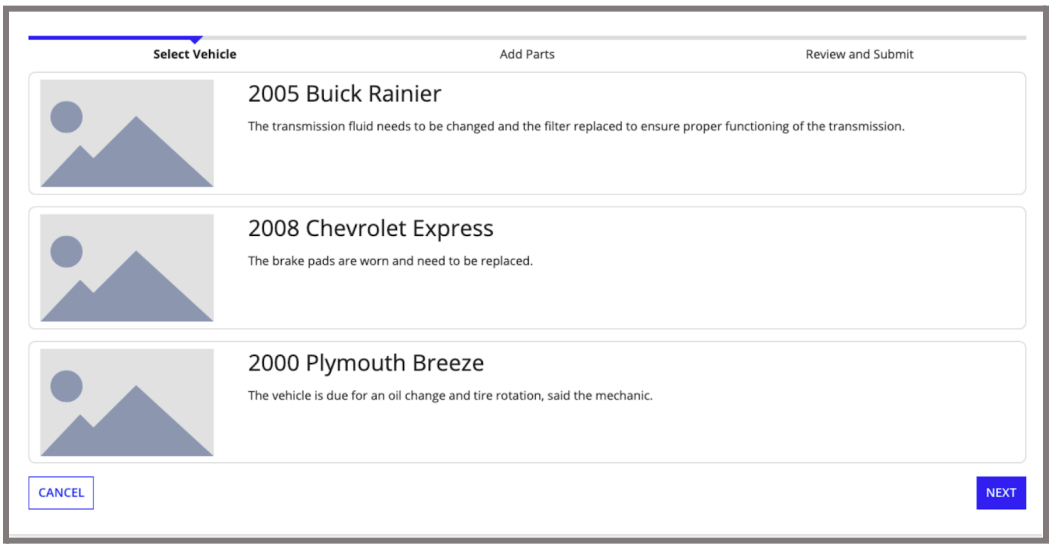
Exercise to Accompany
Create a Wizard and Save Your Data

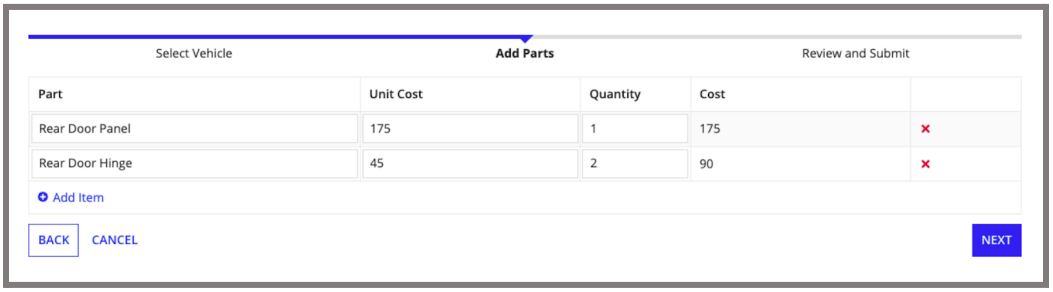
The course Interfaces 102: Building Dynamic Interfaces contains 4 exercises. These exercises build upon each other. Complete exercises in order and keep the app and all objects until you are done with the course.

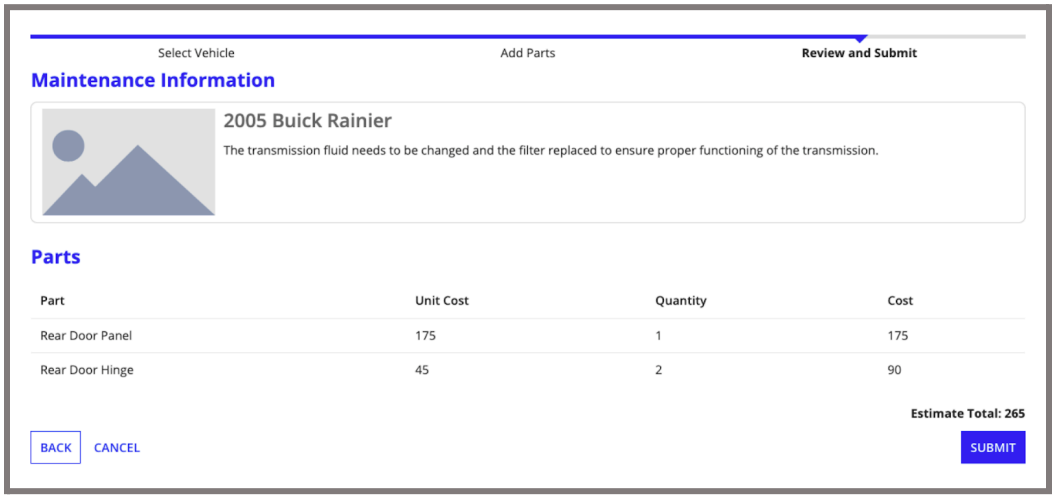
Exercise 5: Create a Wizard

In the previous exercises, you built an interface that allows mechanics to select a vehicle maintenance request, view its details, and create a list of parts needed to complete the maintenance. In this exercise we will create a wizard that will enhance the usability of the interface by breaking the process into three steps: Select a Vehicle, Add Pards, and Review & Submit.

At the end of this exercise, your interface should look something like this:







Create a Wizard

In this step, you will use the SAIL expression below to create a wizard for your interface consisting of three steps.

1. Copy the text below.
2. Paste it into your interface expression just above the a!forEach() function you added in a previous exercise.


```
a!localVariables(
  local!milestoneBarSteps: {
    /*Replace the text below with the step titles.*/
    "Step One Title",
    "Step Two Title",
    "Step Three Title"
  },
  local!currentMilestoneBarStep: 1,
  {
    a!milestoneField(
      steps: local!milestoneBarSteps,
      active: local!currentMilestoneBarStep,
    ),
    choose(
      local!currentMilestoneBarStep,
      {
        /*Replace the a!richTextDisplayField() below
          with the contents of the first step.*/
        a!richTextDisplayField(
          value: "The contents of the first step go here.",
          align: "CENTER"
        )
      },
      {
        /*Replace the a!richTextDisplayField() below
          with the contents of the second step.*/
        a!richTextDisplayField(
          value: "The contents of the second step go here.",
          align: "CENTER"
        )
      },
      {
        /*Replace the a!richTextDisplayField() below
          with the contents of the third step.*/
        a!richTextDisplayField(
          value: "The contents of the third step go here.",
```

```
      align: "CENTER"
    )
  }
),
a!columnsLayout(
  columns: {
    a!columnLayout(
      contents: {
        a!buttonArrayLayout(
          buttons: {
            a!buttonWidget(
              label: "Back",
              value: local!currentMilestoneBarStep - 1,
              saveInto: local!currentMilestoneBarStep,
              style: "OUTLINE",
              showWhen: local!currentMilestoneBarStep > 1
            ),
            a!buttonWidget(
              label: "Cancel",
              style: if(
                local!currentMilestoneBarStep = 1,
                "OUTLINE",
                "LINK"
              )
            )
          },
          align: "START"
        )
      }
    ),
    a!columnLayout(
      contents: {
        a!buttonArrayLayout(
          buttons: a!buttonWidget(
            label: if(
              local!currentMilestoneBarStep=length(local!milestoneBarSteps),
              "Submit",
              "Next"
            ),
            saveInto: if(
              local!currentMilestoneBarStep=length(local!milestoneBarSteps),
              {
                /*Use a!save() to save values to rule inputs
                  when form is submitted.*/
              },
              a!save(
                local!currentMilestoneBarStep,
                local!currentMilestoneBarStep + 1
```

```
                )
              ),
              style: "SOLID"
            )
          )
        }
      )
    }
  )
}
)
```

Populate the Wizard Steps

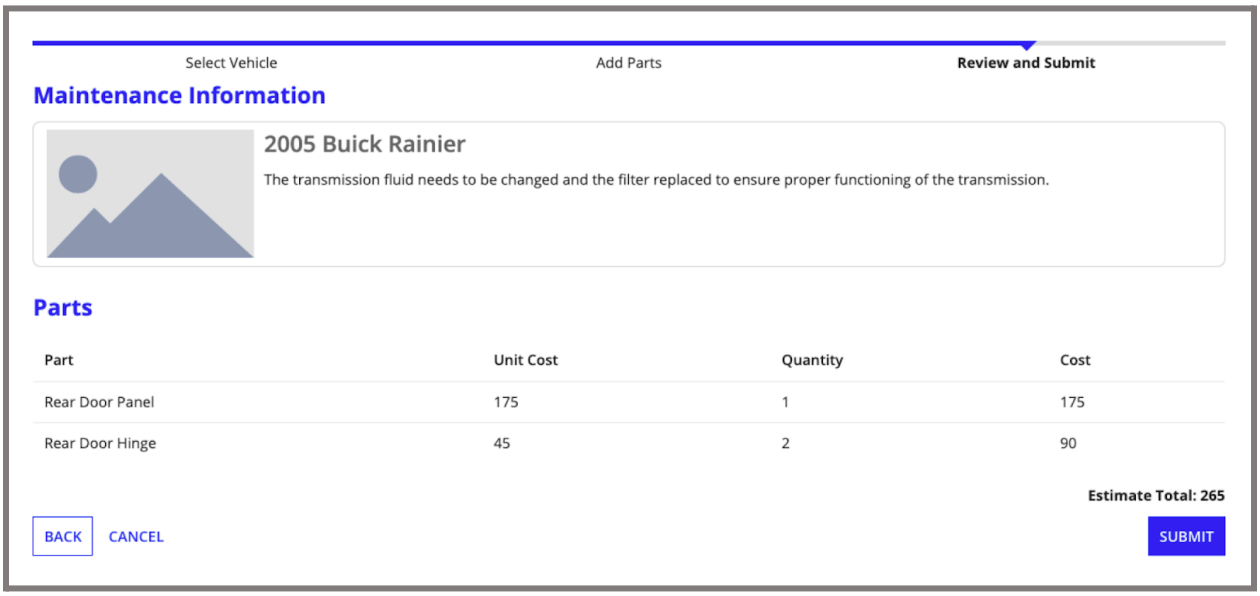
In this step, you will move the components that you have previously created into the correct steps of the wizard, and create content for the third step: Review and Submit

1. Locate the choose() function and delete the contents of the first step. Replace these with the following:
    a. The **a!forEach()** function you created earlier that generates the maintenance request cards.
    b. The **sectionLayout()** component that displays the maintenance summary view.
2. Delete the contents of the second step and replace with the **a!gridLayout()** component for adding parts that you created in the previous exercise.
3. Replace the contents of the third step with:
    a. An **a!sectionLayout** component labeled “Maintenance Information” that displays details from the **local!selectedMaintenanceRequest** variable.
    In the sample expression provided at the end of this step, the maintenance information is presented in a card similar to those created in Exercise 2. The existing **a!cardLayout** was copied with references to **fv!item** replaced with references to **local!selectedMaintrenanceRequest**.
    b. An **a!sectionLayout** component labeled “Parts” that contains a read-only grid displaying the information contained in the **local!parts** variable.
        i. Set the *data* parameter for the grid to **local!parts**.
        ii. Set the *columns* parameter to a list of four **a!gridColumn** functions labeled “Part”, “Unit Cost”, "Quantity", and “Cost”.
        iii. Set the values for the columns by indexing into **fv!row** using the square bracket notation for record type fields.
    c. An **a!richTextDisplayField** component that displays the total cost of all parts stored in the **local!parts** variable.
    In the sample expression provided at the end of this step, the rich text field

contains an **if()** statement to handle instances where the **local!parts** variable is empty.

4. Replace the placeholder values for the step labels in the local variable local!milestoneBarSteps with the following:
   `{"Select Vehicle","Add Parts", "Review and Submit"}`

The third step of your wizard should now look something like this:



Below are examples of what the contents of the **choose()** function should look like at this stage.

The first two steps are populated with the **a!forEach()**, **a!sectionLayout()**, and **a!gridLayout()** created in previous exercises:

```
choose(
  local!currentMilestoneBarStep,
  {
    a!forEach(↔),
    a!sectionLayout(↔)
  },
  {
    a!gridLayout(↔)
  },
```

The third step is populated with two **a!sectionLayout()** components and an **a!richTextField()**. The first section should look like this:

```
a!sectionLayout(
  label: "Maintenance Information",
  contents: a!cardLayout(
    contents: {
      a!sideBySideLayout(
        items: {
          a!sideBySideItem(
            item: a!imageField(
              label: "Image",
              labelPosition: "COLLAPSED",
              images: {
                a!webImage(
                  source: local!selectedMaintenanceRequest[ W0000AA Maintenance.aaVehicle.imageUrl ]
                )
              },
              size: "MEDIUM",
              isThumbnail: false,
              style: "STANDARD"
            ),
            width: "MINIMIZE"
          ),
          a!sideBySideItem(
            item: a!richTextDisplayField(
              labelPosition: "COLLAPSED",
              value: {
                a!richTextHeader(
                  text:
                  local!selectedMaintenanceRequest[ W0000AA Maintenance.aaVehicle.year ]&" "&
                  local!selectedMaintenanceRequest[ W0000AA Maintenance.aaVehicle.make ]&" "&
                  local!selectedMaintenanceRequest[ W0000AA Maintenance.aaVehicle.model ],
                  size: "MEDIUM"
                ),
                a!richTextItem(
                  text: {local!selectedMaintenanceRequest[ W0000AA Maintenance.issue ]}
                )
              }
            )
          )
        }
      )
    },
    height: "AUTO",
    style: "TRANSPARENT",
    shape: "ROUNDED",
    marginBelow: "STANDARD"
  )
),
```

The second section should look like this:

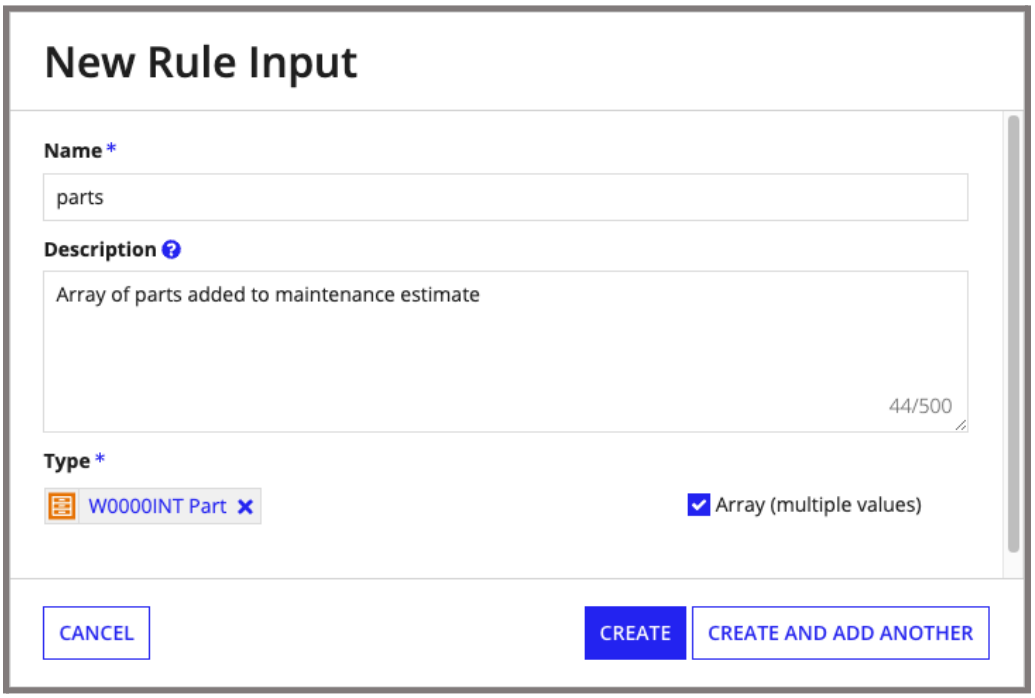
```
a!sectionLayout(
  label: "Parts",
  contents: {
    a!gridField(
    data: local!parts,
    columns: {
      a!gridColumn(
        label: "Part",
        value: fv!row[ W0000INT Part.name ]
      ),
      a!gridColumn(
        label: "Unit Cost",
        value: fv!row[ W0000INT Part.unitCost ]
      ),
      a!gridColumn(
        label: "Quantity",
        value: fv!row[ W0000INT Part.qty ]
      ),
      a!gridColumn(
        label: "Cost",
        value: fv!row[ W0000INT Part.unitCost ]*fv!row[ W0000INT Part.qty ]
      )
    }
  ),
    a!richTextDisplayField(
      align: "RIGHT",
      value: a!richTextItem(
        style: "STRONG",
        text: "Estimate Total: "&
        a!currency(
          isoCode: "USD",
          value: if(
            condition: a!isNotNullOrEmpty(local!parts),
            valueIfTrue: sum(local!parts[ W0000INT Part.qty ]*local!parts[ W0000INT Part.unitCost ]),
            valueIfFalse: 0
          )
        )
      )
    )
  }
)
```

Save Your Data to Rule Inputs

In this step, you will configure the Submit button on the final step of the wizard to save the data from your local variables into rule inputs so that it can be passed to a process model.

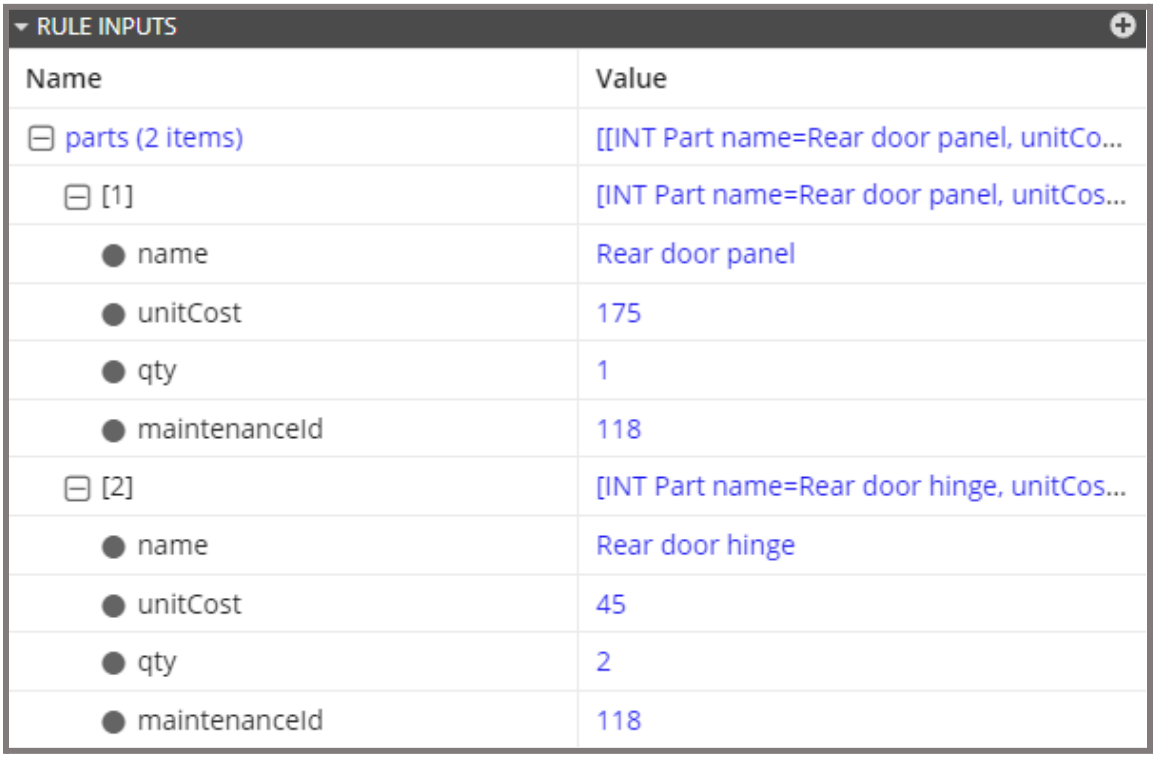
1. Create a rule input.
    a. Name the rule input "parts".
    b. Set the type to the Part record type created in Exercise 1.

c. Check the **Array** box since there may be multiple Part records generated with each maintenance estimate.



2. Locate the **Next/Submit** button in the interface expression. It should be near the bottom. The *saveInto* parameter of this button is set to an **if()** function. The first argument of this function determines whether the user is on the last step of the wizard. If so, the next argument will execute. Currently this argument contains the comment:
   `/*Use a!save() to save values to rule inputs when form is submitted.*/`
3. In the second argument of the **if()** function, create an **a!forEach** loop that will set the *maintenanceId* field for each item in **local!parts** to the *requestId* of **local!selectedMaintenanceRequest**. This ensures the Part records, when created, are associated with the correct Maintenance record.
   a. Set the *items* parameter to **local!parts**.
   b. Set the *expression* parameter to:
   ```
   a!save(
      target: fv!item['W#INT Part.maintenanceId'],
      value: local!selectedMaintenanceRequest[W#AA Maintenance.id']
   )
   ```
4. Add another **a!save** function after the **a!forEach()** loop to save the values from **local!parts** to the rule input **ri!parts**.

5. Test that clicking **Submit** updates the rule input and that the *maintenanceId* field is set.



Here is an example of what your interface expression should look like at this stage:

```
a!localVariables(
  local!scheduledMaintenanceRequests: rule!W6565INT_getScheduledMaintenance(),
  local!selectedMaintenanceRequest,
  local!parts: {},
  {
    a!localVariables(
      local!milestoneBarSteps: {↔},
      local!currentMilestoneBarStep: 1,
      {
        a!milestoneField(↔),
        choose(↔),
        a!columnsLayout(
          columns: {
            a!columnLayout(↔),
            a!columnLayout(
              contents: {
                a!buttonArrayLayout(
                  buttons: a!buttonWidget(
                    label: if(
                      local!currentMilestoneBarStep=length(local!milestoneBarSteps),
                      "Submit",
                      "Next"
                    ),
                    saveInto: if(
                      local!currentMilestoneBarStep=length(local!milestoneBarSteps),
                      {
                        a!forEach(
                          items: local!parts,
                          expression: a!save(
                            fv!item[ W5656INT Part.maintenanceId ],
                            local!selectedMaintenanceRequest[ W5656AA01 Maintenance.id ]
                          )
                        ),
                        a!save(
                          ri!parts,
                          local!parts
                        )
                      },
                      a!save(
                        local!currentMilestoneBarStep,
                        local!currentMilestoneBarStep + 1
                      )
                    ),
                    style: "SOLID"
                  )
                )
              }
            )
          }
        )
      }
    )
  }
)
```