# **Partial Evaluation Exercises**

**Exercise 1: Partial Evaluation of Functions**

In this exercise you will use a!applyComponents and partial evaluation of functions to build a maintenance request for multiple parts on a car in multiple sections. You will create two interfaces--one for the individual section, and one to show multiple sections. Remember, when using more complicated expressions, a lot of the work is already done for you. For extra help, take a look at the SAIL Recipes in the documentation.

<https://forum.appian.com/suite/help/16.1/SAIL_Recipes.html#Add_and_Populate_Sections_Dynamically>

We’ll simplify the section created in the evaluation functions exercise and remove the under warranty checkbox. Your interfaces should adhere to the following guidelines:

1. Interface to build one section
   1. Change the original form interface to call a section layout instead (remove the button).
   2. This interface should have a rule input for the whole request that can collect multiple values.
   3. Add a rule input for the index of the section.
   4. Add a rule input called “parts” for the dropdown query and remove the load() function. (If you leave it here, the query will be executed every time a new section is built, versus just once).
   5. To test, populate ri!parts with the query rule.
   6. When saving and displaying values, each interface input should point to the index of the individual request. For instance:

value: dollar(ri!request[ri!index].repairCost),

saveInto: ri!request[ri!index].repairCost

1. Interface to build multiple sections
   1. This interface should call applyComponents to build multiple iterations of the individual section rule.
   2. Create a rule input that represents multiple requests.
   3. Use partial functions to build out the index.
   4. Use an enumeration on the total number of requests as the array parameter.
   5. Make sure the interface can handle if if the array has less than one item in it, or if it is null.
   6. Load the dropdown query here and pass it to the rule input in the child rule for applyComponents.
   7. Test the interface with an array of two requests, each with a different defaulted repairCost

STRETCH GOAL: Add the warranty checkbox back in