# **Editable Grid Exercises**

**Exercise 1: Editable Grids**

In this exercise you will create an editable grid to collect requests to repair multiple parts on a single car. 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_Edit_and_Remove_Data_In_an_Inline_Editable_Grid>

Use a!gridLayout to create the grid and a!applyComponents to create the rows in the grid. The interface should adhere to the following guidelines:

1. Rule to create one row
   1. Use a!gridRowLayout() to create one row in an editable grid.
   2. Create the following rule inputs
      1. requests (VFM\_request array)
      2. index (integer)
      3. parts (VFM\_part array)
      4. token (V
      5. FM\_request array)
   3. In the contents parameter of the a!gridRowLayout add the following fields with similar configurations to Exercise 4.
      1. Part (dropdown)
      2. Cost (decimal)
      3. Date Needed (date)
      4. Type of Maintenance (dropdown)
      5. Reason (paragraph with height: “SHORT”)
      6. Remove icon (image field with a dynamic link)
   4. In the id parameter, call the rule input for index.
   5. Remember to not just map the saveInto and value to ri!requests.childField, but to add the index--e.g. ri!requests[ri!index].repairCost
   6. For the remove icon, make the image an “ICON” size and use the iconIndicator function to build the remove image.
   7. Make the image a dynamic link that removes the row from the entire requests variable. Use the a!save function.
2. Rule to create the editable grid
   1. Create an interface input
   2. Call the a!gridLayout function to build a grid. Add 6 header cells.
   3. Make the cost, date needed, and image columns narrow.
   4. Call applyComponents to build the rows.
      1. Use the interface rule that creates just one row.
         1. Pass in the rule input for requests.
         2. Use partial functions to defer evaluating the index.
         3. Use local variable for the parts.
         4. Use a local variable for the token.
      2. Create an array by enumerating over the number of requests
      3. Use a local token as the array variable
   5. Add a dynamic link to add more rows