Dynamically Load Field Values in ServiceNow Based on Another Field

This document explains how to dynamically load values into a form field in ServiceNow based on a previous field's selection. The implementation involves using a Client Script, a Script Include, or a Reference Qualifier.

Step 1: Add Fields to the Form

1. Navigate to Form Design for the desired table (e.g., Incident).

2. Add the required fields:

   - Field A: Dropdown or Reference field (e.g., Category).

   - Field B: Dropdown or Reference field (e.g., Subcategory).

Step 2: Use a Client Script for Dynamic Loading

1. Navigate to System Definition > Client Scripts.

2. Create a new Client Script:

   - Name: Dynamic Filter for Field B.

   - Table: Select the relevant table (e.g., Incident).

   - Type: onChange.

   - Field: Field A (e.g., category).

3. Add the following script:

```
(function executeRule(control, oldValue, newValue, isLoading, isTemplate) {

    if (isLoading || newValue == '') {

        return;

    }



    // Get the selected value from Field A
    var selectedValue = g_form.getValue('category'); // Replace 'category' with your field name

    // Clear the options in Field B
    g_form.clearOptions('subcategory'); // Replace 'subcategory' with your field name

    if (selectedValue) {
        // Use GlideAjax to fetch filtered options
        var ga = new GlideAjax('DynamicFieldOptions'); // Script Include Name
        ga.addParam('sysparm_name', 'getFilteredOptions'); // Function Name in Script Include
        ga.addParam('sysparm_category', selectedValue);
        ga.getXMLAnswer(function(response) {
            var options = JSON.parse(response);
            options.forEach(function(option) {
                g_form.addOption('subcategory', option.value, option.label); // Replace 'subcategory' with your field name
            });
        });
    }
})(g_form);
```

Step 3: Create a Script Include for Data Retrieval

1. Navigate to System Definition > Script Includes.

2. Create a new Script Include:

  - Name: DynamicFieldOptions.

  - Accessible from: This application scope only (or adjust as needed).

- Active: Checked.

3. Add the following code:

```
var DynamicFieldOptions = Class.create();
DynamicFieldOptions.prototype = {
    initialize: function() {},

    getFilteredOptions: function(category) {
        var options = [];

        // Query the database based on the selected value in Field A (e.g., Category)
        var gr = new GlideRecord('your_table'); // Replace with the table name
        gr.addQuery('category', category); // Replace 'category' with your actual field
        gr.query();
        while (gr.next()) {
            options.push({
                 value: gr.getValue('sys_id'), // Replace 'sys_id' with the desired field
for the value
                 label: gr.getValue('name') // Replace 'name' with the desired field for
the label
            });
        }

        return JSON.stringify(options);
    },
    type: 'DynamicFieldOptions'

};
```