

**Project Report: Laptop Request Catalog Item**

### 

### 

### **Team Members:**

* **BHARATH KUMAR D**(Team Leader)
* **AJAY KUMAR S**
* **AJITH KUMAR B**
* **AKASH S**

## **1. Abstract**

The Laptop Request Catalog Item project is designed to simplify and automate the laptop request process within an organization using ServiceNow. The previous system relied on manual handling, which was slow, inefficient, and prone to data inconsistencies.  
This project introduces a dynamic Service Catalog item that allows employees to conveniently submit laptop requests through an intuitive, automated interface. The catalog form intelligently adapts to user inputs with features like conditional field visibility and a reset option for better usability.  
By utilizing ServiceNow’s Catalog UI Policies, UI Actions, and Update Sets, the team developed a scalable and reusable solution that significantly improves process accuracy, operational efficiency, and overall user experience in IT service management.

## **2. Tasks Done with Explanation**

### **Task 1: Understanding the Problem**

The initial phase involved analyzing the existing manual process for laptop requests.  
 **Challenges Identified:**

* Manual and time-consuming process.
* Lack of dynamic form behavior.
* High potential for data entry errors.
* No mechanism for governance or change tracking.

The goal was to create a **ServiceNow Catalog Item** with automation and dynamic features to address these challenges.

### **Task 2: Creation of Update Set**

Before starting the configuration, a **local Update Set** was created and activated to ensure that all changes (Catalog Items, UI Policies, UI Actions) were tracked.  
 **Update Set Name:** *Laptop Request*

This practice allows proper version control, governance, and easy migration between ServiceNow environments (Development → Testing → Production).

### **Task 3: Creating the Service Catalog Item**

A new catalog item was created under the **Hardware** category.

* **Name:** Laptop Request
* **Catalog:** Service Catalog
* **Category:** Hardware
* **Short Description:** Use this item to request a new laptop

This item acts as the main form where users enter all necessary details for a laptop request.

### **Task 4: Implementing Dynamic Form Behavior using Catalog UI Policy**

Dynamic behavior was added using a **Catalog UI Policy** to make the form responsive to user inputs.

* **UI Policy Name:** Show Accessories Details
* **Condition:** Applies when the *additional\_accessories* variable is set to *true*.
* **UI Policy Action:**
  + *accessories\_details* → Visible = True
  + *accessories\_details* → Mandatory = True

**Outcome:** The form automatically displays accessory details fields only when the user selects that they need additional accessories.

### 

### **Task 5: Adding User Experience Enhancement – UI Action (Reset Form)**

A **client-side UI Action** was developed to allow users to reset the form instantly.

**UI Action Details:**

* **Name:** Reset Form
* **Table:** Shopping Cart (*sc\_cart*)
* **Client:** True

**JavaScript Script:**

function resetForm() {

g\_form.clearForm(); // Clears all fields in the form

alert("The form has been reset.");

}

**Result:** Users can clear all entered data and start over without reloading the page.

### **Task 6: Deployment and Validation**

The developed catalog item was exported via an **Update Set** and imported into a testing instance for validation.

**Steps followed:**

1. Exported the Update Set from the source instance as an XML file.
2. Imported the XML into the target instance.
3. Previewed the Update Set to check for errors or conflicts.
4. Successfully committed the Update Set for deployment.

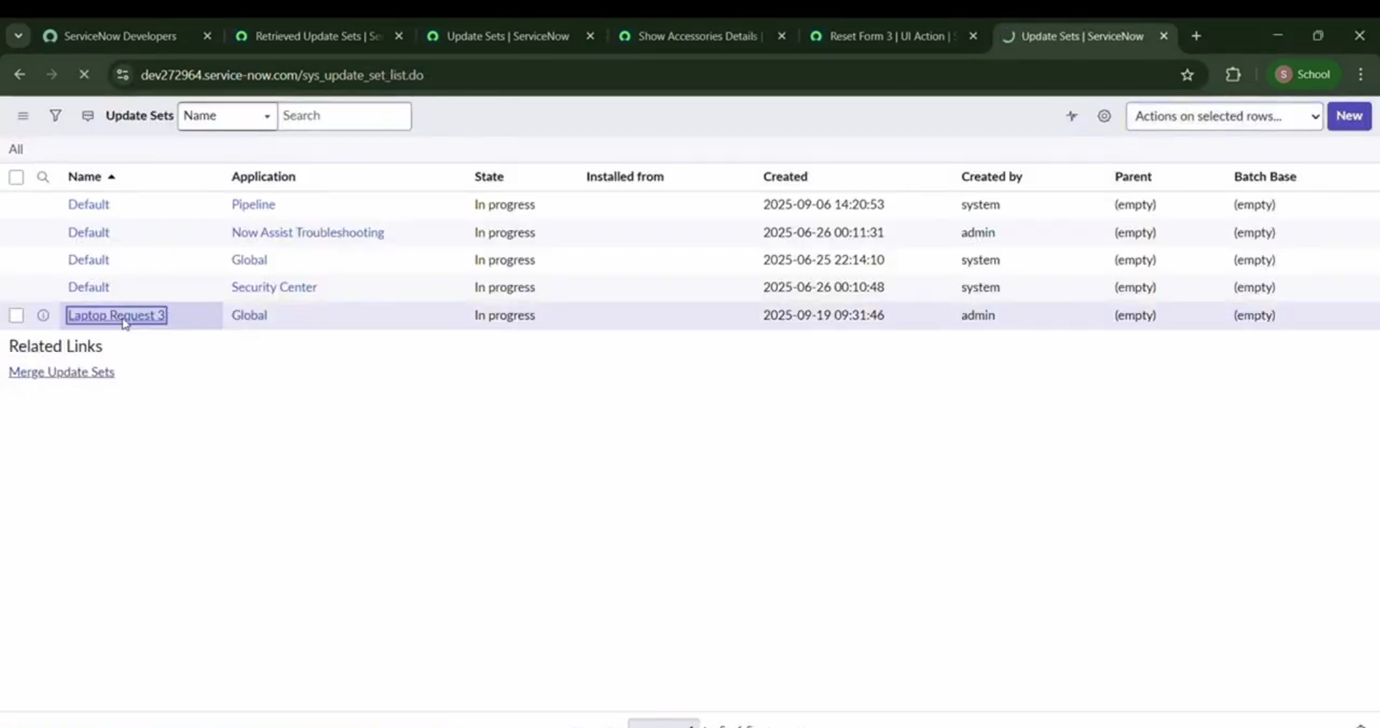
**Testing Results:**

* Dynamic form behavior worked correctly.
* Reset button cleared all fields as expected.
* Form submission and workflow integration functioned properly.

## **3. Screenshots**

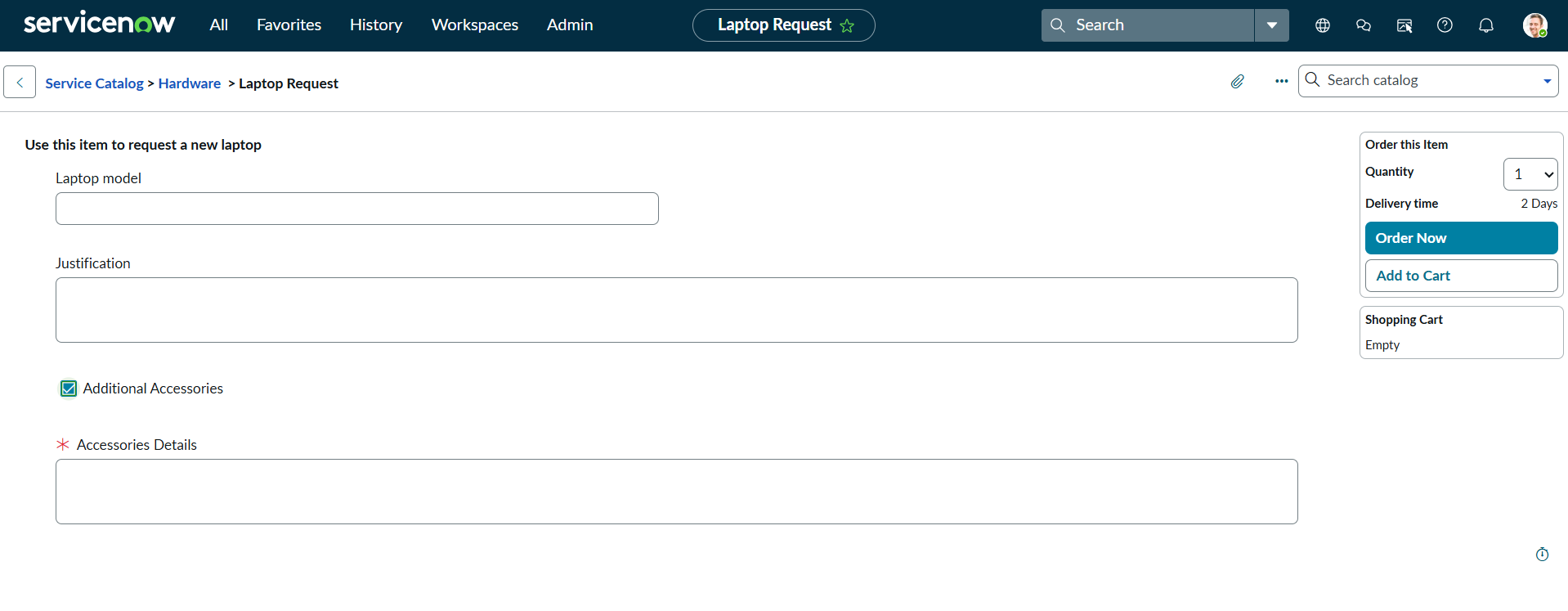
Below are example screenshots representing key parts of the project (you can insert actual screenshots when printing or submitting):

1. **Service Catalog Item Configuration:**

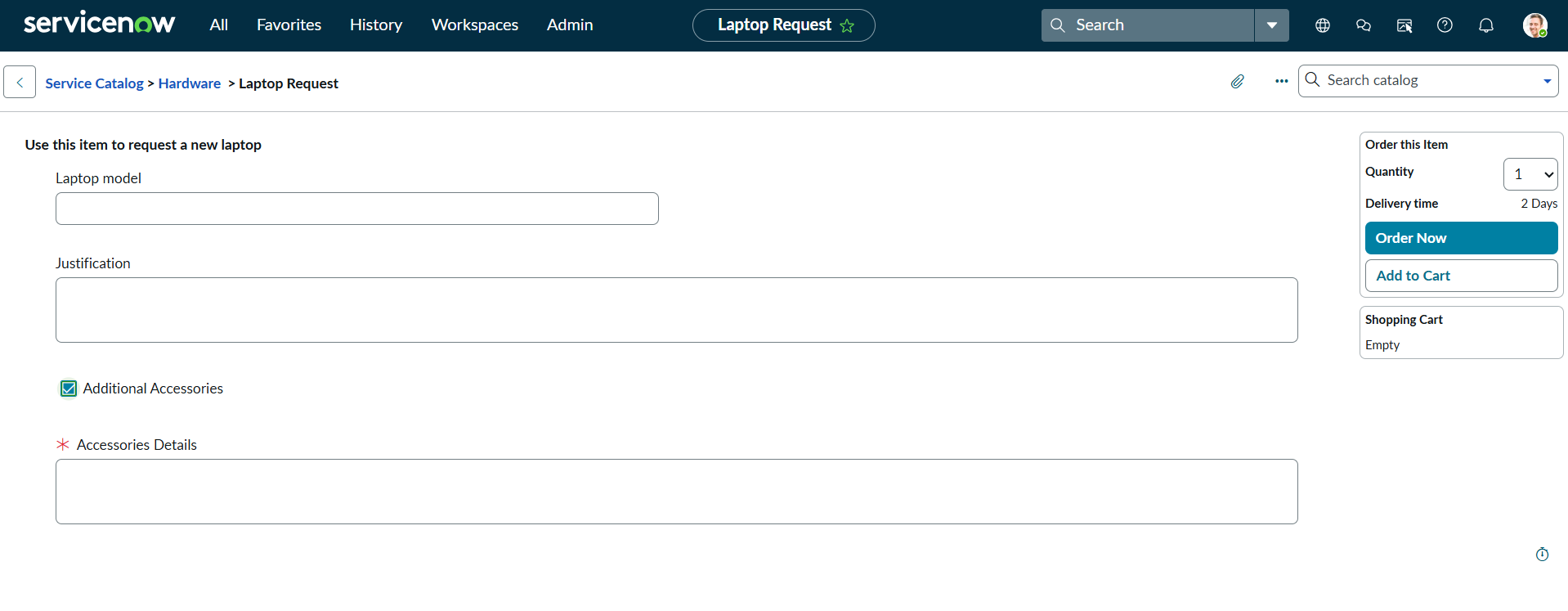


**2.Dynamic Field Visibility (Before and After)**

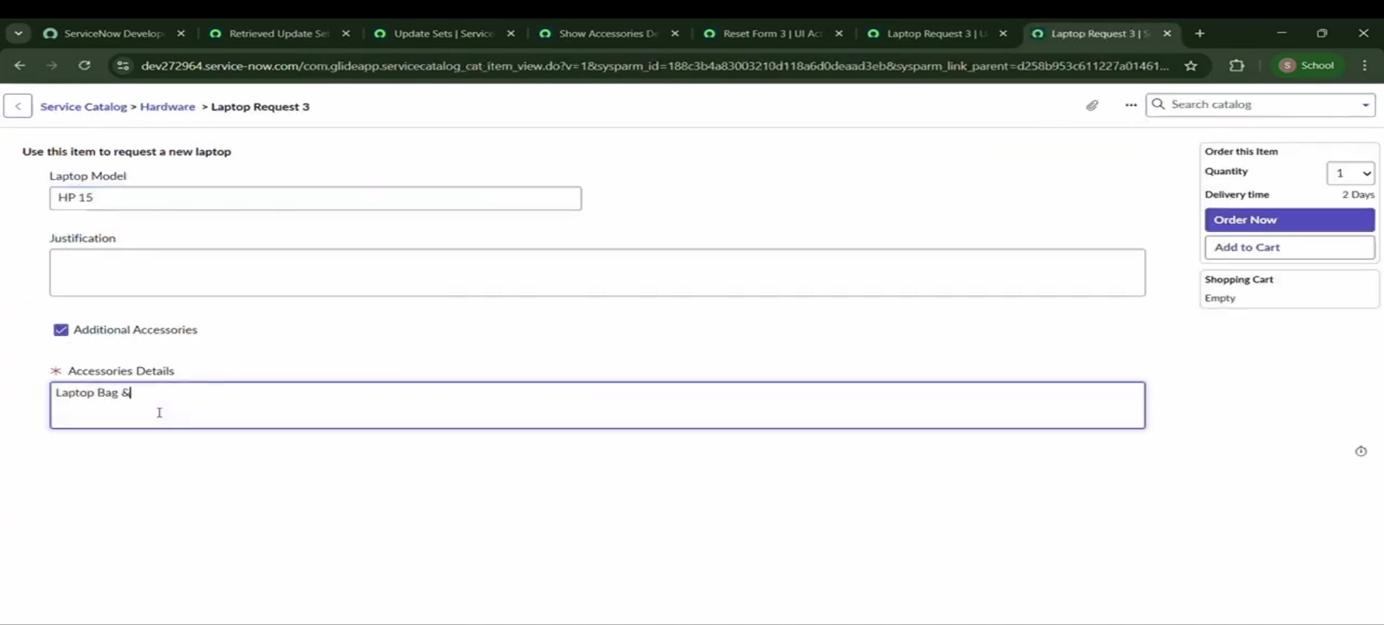
* + Before checking *Additional Accessories*: field hidden
  + After checking *Additional Accessories*: field visible and mandatory.



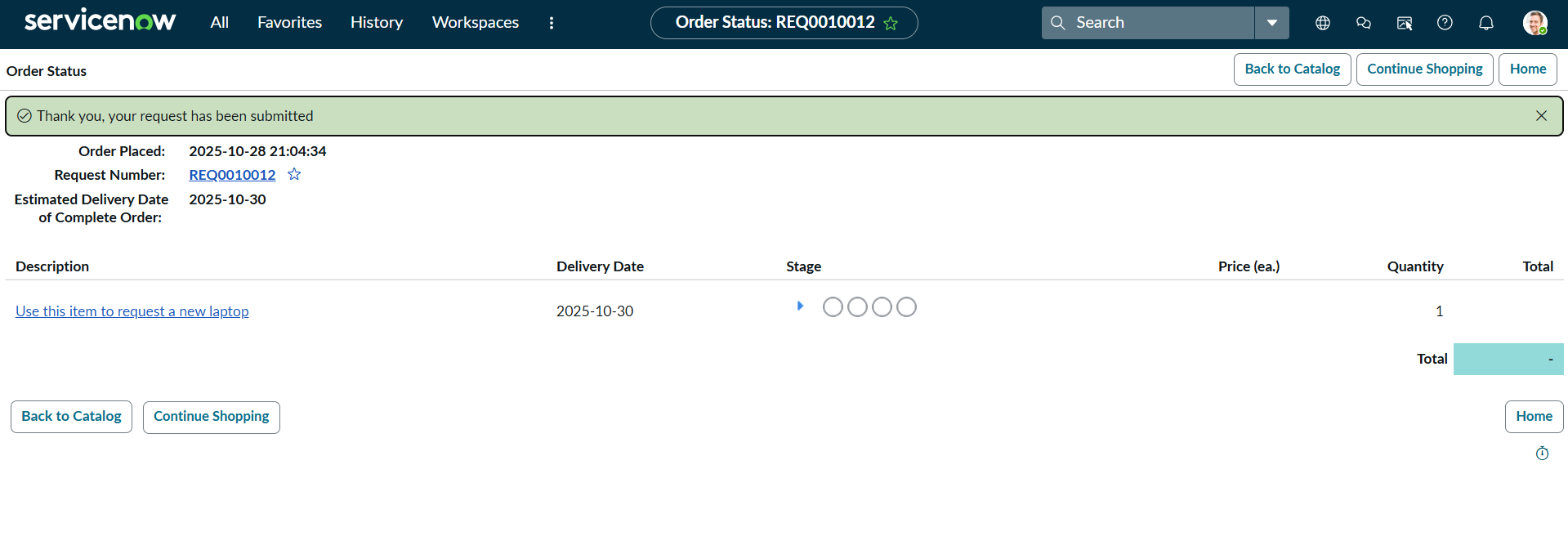
**3.UI Action Script Editor** *(Screenshot showing the “Reset Form” script configuration)*

**

**4.Preview and Commit of Update Set** *(Screenshot showing the Update Set commit process)*



**5.Final Laptop Request Form Displayed in Service Catalog** *(Screenshot showing the published and working catalog form)*

**

## **4. Result**

The **Laptop Request Catalog Item** was successfully implemented and tested in ServiceNow.

### **Key Results:**

* Reduced manual effort and errors in the laptop request process.
* Improved user experience through dynamic form logic.
* Ensured governance through Update Set tracking.
* Provided reusability for similar ServiceNow catalog items.

### **Business Impact:**

* **Efficiency:** The automated system saves time for both requesters and IT staff.
* **Accuracy:** Dynamic form rules ensure only valid and complete data is submitted.
* **Scalability:** The same model can be extended for other hardware or software requests.

## 

## **5. Conclusion**

The **Laptop Request Catalog Item** project achieved its primary objective of transforming a manual, error-prone process into an automated, efficient, and user-friendly digital solution using ServiceNow.

By utilizing **Catalog UI Policies**, **UI Actions**, and **Update Sets**, the team successfully enhanced user interaction, reduced processing time, and ensured maintainability and governance.

This project demonstrates the power of **ServiceNow automation** in improving IT service management workflows and stands as a strong example of how digital transformation can optimize organizational operations.

## **6. Future Enhancements**

1. Integrate **Approval Workflows** for managerial authorization.
2. Add **automatic email notifications** for request status updates.
3. Link with **asset management modules** for real-time inventory tracking.
4. Include **analytics dashboards** to monitor request trends and performance.