

Phase – 2: Backend Development & Configurations Data Architecture

1. Phase & Task Reference

Phase 2 focuses on backend automation and configuration, enabling the system to process software installation requests automatically after submission.

Modules covered:

- Data Architecture & Table Relationships
 - Business Rule Configuration
 - Flow Designer Automation
 - Approval Handling
 - Catalog Task Creation
 - Exception Handling (License Issues)
-

2. Objective

The objective of Phase 2 is to implement backend logic that:

- Automates request approvals and fulfillment
- Handles license unavailability scenarios
- Ensures correct data flow between ServiceNow tables
- Reduces manual intervention through Flow Designer and Business Rules

This phase transforms business requirements into **working automation logic**.

3. Business Requirement

The organization requires a backend system that can:

- Automatically process software installation requests
 - Route approvals to the correct approvers
 - Create fulfillment tasks for IT teams
 - Detect license issues and raise incidents when software cannot be fulfilled
 - Maintain data consistency across request-related tables
-

4. Configuration Details

4.1 Data Architecture

The request lifecycle is managed using standard ServiceNow tables:

Data Flow Description:

1. User (Employee / Requestor)
Submits a Software Installation Request via Service Catalog.
2. Service Catalog Item (sc_cat_item)
Captures request details from the catalog form.
3. Request (sc_request)
A parent request record is created to track the overall request.
4. Requested Item (sc_req_item)
Each catalog item creates a Requested Item (RITM) containing item-specific details.
5. Approval (sysapproval_approver)
Manager or Network Admin approves or rejects the request.
6. Catalog Task (sc_task)
Upon approval, fulfillment tasks are created and assigned to the Network / Software Support team.

4.2 Business Rule

Purpose:

To automatically create an **Incident** when a requested software cannot be fulfilled due to license unavailability.

Configuration Summary:

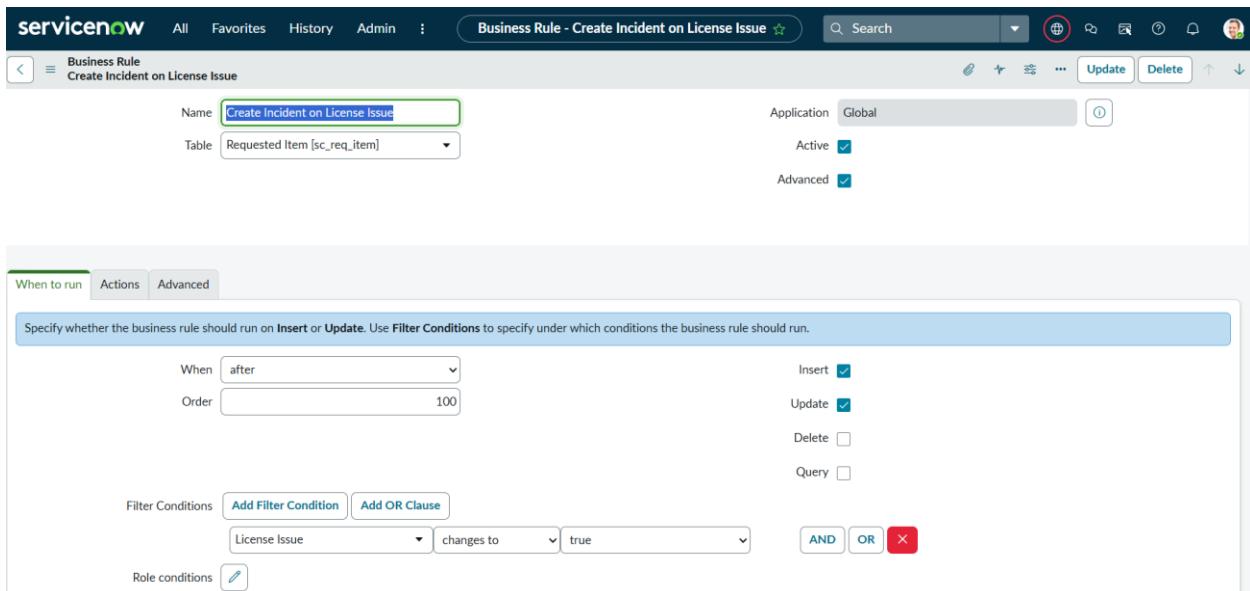
- **Table:** Requested Item (sc_req_item)
 - **Trigger:** On Update
 - **Condition:**
 - State changes to *On Hold*
- OR**

- Custom field License Issue = true

Business Rule Script Logic:

- Checks whether the license issue flag is set
- Creates an Incident record automatically
- Assigns the incident to Software Support
- Sets priority and provides a clear description

 **Screenshot: Creation of Business rule**

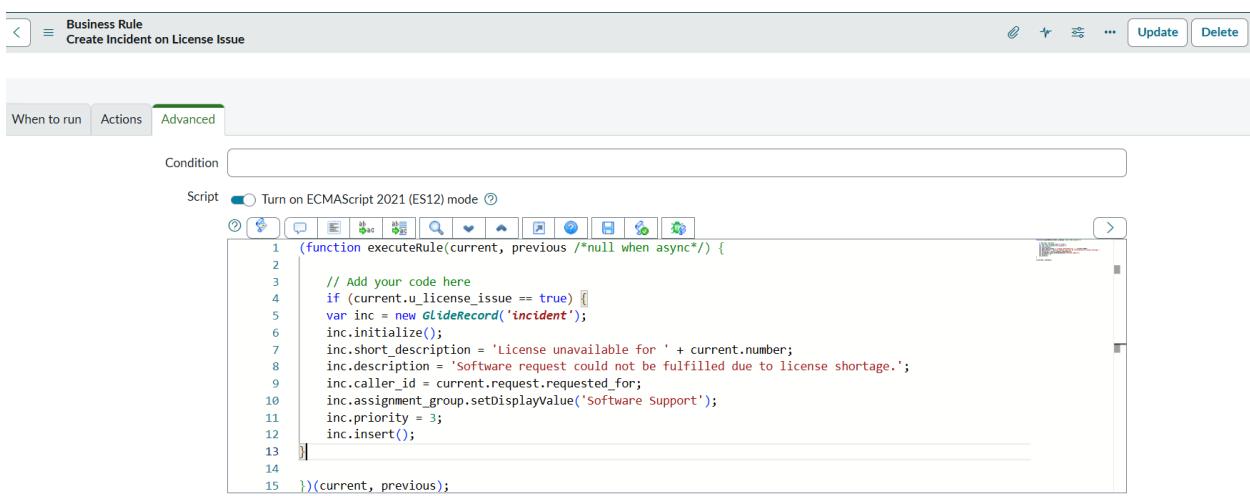


The screenshot shows the 'Business Rule - Create Incident on License Issue' configuration page. The 'Name' is 'Create Incident on License Issue'. The 'Table' is 'Requested Item [sc_req_item]'. The 'Application' is 'Global'. The 'Active' and 'Advanced' checkboxes are checked. The 'When to run' tab is selected, showing the following conditions:

- When: after
- Order: 100
- Insert: checked
- Update: checked
- Delete: unchecked
- Query: unchecked

Filter Conditions: License Issue changes to true. Role conditions: None.

 **Screenshot: Business rule script**



The screenshot shows the 'Script' tab of the business rule configuration. The ECMAScript 2021 (ES12) mode is turned on. The script code is as follows:

```

(function executeRule(current, previous /*null when async*/) {
    // Add your code here
    if (current.u.license_issue == true) {
        var inc = new GlideRecord('incident');
        inc.initialize();
        inc.short_description = 'License unavailable for ' + current.number;
        inc.description = 'Software request could not be fulfilled due to license shortage.';
        inc.caller_id = current.request.requested_for;
        inc.assignment_group.setDisplayValue('Software Support');
        inc.priority = 3;
        inc.insert();
    }
})(current, previous);

```

Outcome:

- Ensures license issues are tracked as incidents
 - Enables faster resolution and auditability
 - Prevents silent request failures
-

5. Configuration Steps(High Level)

Activity 1: Flow Design Creation (Flow Designer)

Objective: Create a Flow that automates approvals and task creation.

Steps:

1. Open ServiceNow
2. Navigate to **All → Flow Designer**
3. Click **New**
4. Select **Flow**
5. Enter Flow Name (e.g., *Software Installation Request Flow*)
6. Choose Trigger: **Service Catalog → Catalog Item Requested**
7. Submit and open the Flow for configuration

Activity 2: Flow Automation Logic

Objective: Define approval and fulfillment logic.

High-Level Flow Design:

1. **Trigger:** Catalog Item Requested
2. **If Condition:** Validate the catalog item (Software Installation Request)
3. **Ask for Approval:**
 - Approval routed to Manager / Network Admin
4. **If Approved:**
 - Create Catalog Task (sc_task)
 - Assign to Network / Software Support Team
5. **If Rejected:**
 - Update Requested Item state to *Closed Incomplete*
 - Add rejection notes
6. **End Flow**

Key Actions Used:

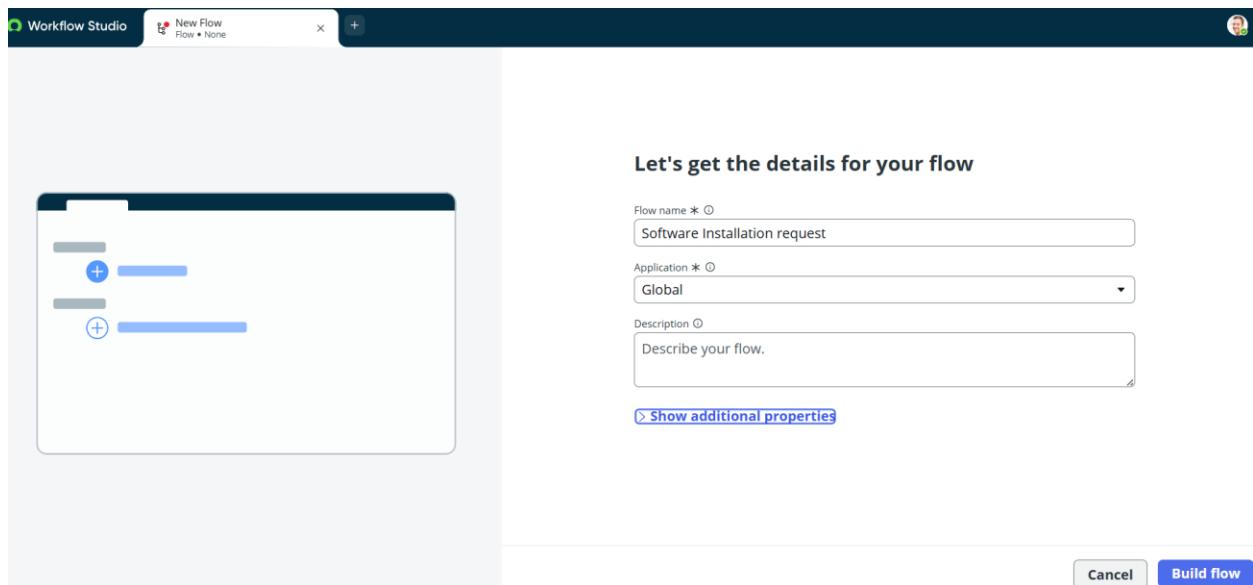
- If Condition
- Ask For Approval
- Create Record (Catalog Task)
- Update Record

- End

Final Step:

Activate the Flow after successful testing.

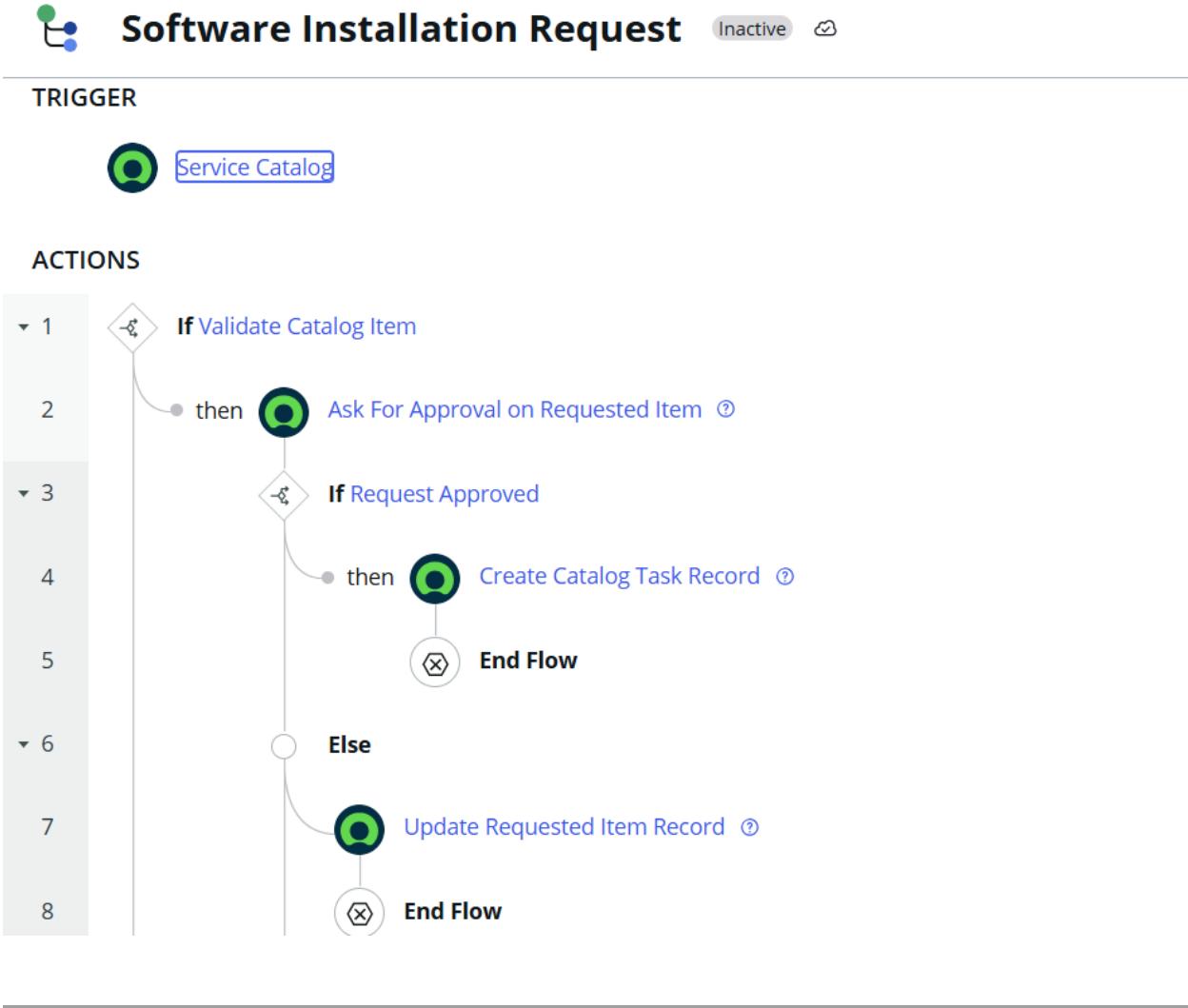
 **Screenshot: Flow Creation**



 **Screenshot: Trigger Configuration**

The screenshot shows the configuration for a 'Software Installation Request' trigger. At the top, it says 'Software Installation Request' with status 'Inactive' and a lock icon. Below this is a 'TRIGGER' section. It shows a 'Service Catalog' icon and the text 'Trigger Service Catalog'. A dropdown menu is open over this text. At the bottom of the trigger configuration is a 'Advanced Options' button.

 **Screenshot:** Completed Flow



6. Outcome

After completing Phase 2:

- Backend automation is fully implemented
- Requests flow seamlessly from submission to fulfillment
- Approvals are enforced through Flow Designer
- License issues automatically generate incidents
- Manual coordination is minimized
- Data integrity across REQ, RITM, and SCTASK is ensured