NM1051 – SERVICENOW ADMINISTRATOR – SMART INTERNZ

LAPTOP REQUEST CATALOG ITEM

A PROJECT REPORT

Submitted by

CHANDRU G (815422104011)
BASKAR M (815422104007)
SANJAY S (815422104046)
SHANMUGAM R (815422104050)

BACHELOR OF ENGINEERING

IN

SEVENTH SEMESTER

COMPUTER SCIENCE AND ENGINEERING

SRI RAMAKRISHNA COLLEGE OF ENGINEERING

ANNA UNIVERSITY : CHENNAI 600 025

NOV/DEC 2025

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"LAPTOP REQUEST CATALOG ITEM"** is the bonafide work of **" CHANDRU G,BASKAR M,SANJAY S,SHANMUGAM R"** who carried out the project work under my supervision. No part of the dissertation has been submitted for any degree or any other academic award anywhere before.

SIGNATURE
Mrs. C.SURYA M.E.,
ASSISTANT PROFESSOR
Department of CSE
Sri Ramakrishna College of Engineering
Perambalur

SIGNATURE
Mr. R.DINESH RAJ M.E.,
HEAD OF THE DEPARTMENT
Department of CSE
Sri Ramakrishna College of Engineering
Perambalur

Submitted for the University Practical Examination held on…….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENT

CHAPTER NO.	TITLE	PAGE NO.
1	IDEATION PHASE	1
2	PROJECT PLANNING PHASE	4
3	PROJECT DESIGN PHASE	6
4	REQUIREMENT ANALYSIS	9
5	PERFORMANCE TESTING	11

1. IDEATION PHASE

1.1 Introduction :

The ideation phase represents the creative foundation of the project. It focuses on identifying organizational challenges and conceptualizing innovative solutions that leverage modern technology. In this project, the goal is to address the inefficiencies in the laptop request process within an organization by developing a ServiceNow-based solution. Employees often face delays and confusion due to manual laptop request workflows, lack of dynamic form behavior, and inconsistent data capture. Through ideation, the team aimed to create a digital service catalog item that streamlines laptop requests, automates approvals, and ensures governance and traceability**.**

1.2 Problem Identification :

In most organizations, laptop requests are still processed through emails or offline forms. This causes several issues such as delays, errors, missing information, and a lack of transparency in tracking requests. IT departments often struggle to manage multiple requests efficiently, leading to prolonged approval cycles and unstructured record-keeping. The absence of dynamic form guidance also means employees may fill incorrect details, causing additional rework. Identifying this gap led to the central idea of developing a ServiceNow Laptop Request Catalog Item that simplifies and automates the entire process.

1.3 Objectives :

The main objectives defined in this phase include:

- To automate the laptop request process through ServiceNow.
- To implement dynamic form behaviors based on user selections.
- To provide a reset option for clearing form data.
- To ensure that all changes and requests are tracked for governance.
- To create an intuitive and user-friendly interface.

1.4 Significance of the Solution :

The ideation process highlighted the organizational need for an automated system that saves time, enhances productivity, and reduces human error. A self-service portal empowers employees to request resources independently, while IT administrators can monitor and fulfill requests efficiently. The integration of ServiceNow ensures compliance with ITSM standards and provides scalability for future improvements.

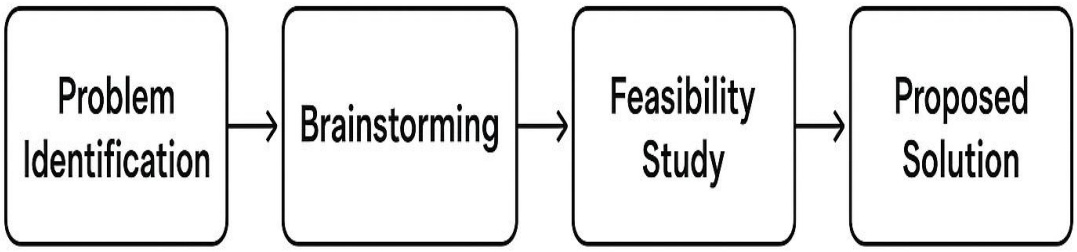
The proposed ***Laptop Request Catalog Item*** solution significantly improves the efficiency, transparency, and accuracy of the laptop provisioning process within an organization. By replacing the traditional manual method with an automated ServiceNow catalog item, employees can easily submit their requests through a guided, dynamic form that ensures proper data entry and eliminates common human errors.

1.5 Expected Outcomes :

The expected outcomes include a functional, dynamic, and interactive catalog form that supports multiple request types, ensures accurate data collection, and reduces manual workload. It will serve as a model for other hardware and software request processes within the organization.

The Laptop Request Catalog Item — which simplifies and automates the entire workflow through the ServiceNow platform. This outcome includes a detailed understanding of user needs, workflow requirements, and dynamic field behaviors to enhance the request experience.

1.6 Block Diagram :



2. PROJECT PLANNING PHASE

2.1 Scope Definition :

The project scope focuses on designing and implementing a Service Catalog item in ServiceNow that automates laptop requests. It includes creating catalog variables, implementing dynamic behaviors using UI policies and client scripts, developing approval workflows, and validating the form through performance testing. The scope also involves ensuring the form's scalability and usability across departments.

2.2 Resource Allocation :

Key resources for this project include:

- Platform: ServiceNow (developer instance).
- Scripting Language: JavaScript (for client and UI scripts).
- Team Roles: Developer, Tester, and Administrator.
- Tools: Workflow Editor, Variable Editor, and Catalog UI Policy Manager.

2.3 Timeline and Milestones :

The project follows a structured timeline:

- Step 1: Ideation and requirement gathering
- Step 2: Catalog item creation and variable design
- Step 3: Dynamic behavior scripting
- Step 4: Workflow development

2.4 Risk Management :

Potential risks include misconfigured scripts, workflow delays, or incomplete data capture. These are mitigated by version control, iterative testing, and peer reviews. Backup copies of scripts and workflows are maintained to ensure smooth recovery.

2.5 Governance Strategy

Governance is maintained by using ServiceNow’s change management module. Every modification to the catalog item is logged for accountability. Access permissions are assigned based on user roles, ensuring that only authorized personnel can modify configurations.

3. PROJECT DESIGN PHASE

3.1 System Overview

The project design phase translates ideas into structured implementation. It focuses on the architectural layout of the ServiceNow catalog item and its components variables, forms, workflows, and scripts.

The Laptop Request Catalog Item *System* is a structured ServiceNow-based application designed to automate and streamline the process of laptop requisition within an organization. The system integrates employee self-service functionality, automated approval workflows, and real-time tracking to ensure efficiency, transparency, and accountability in hardware allocation.

3.2 Form Design:

The form includes fields such as:

- Employee Name (auto-fetched or selectable)
- Department
- Laptop Type (Standard / High-Performance / Developer)
- Accessories (optional checkboxes)
- Justification
- Required Date

3.3 Dynamic Logic Implementation

Dynamic behavior enhances user experience by showing or hiding fields based on selections. For instance:

- When “High Performance” is selected, the justification field becomes mandatory.
- When “Accessories = Yes,” additional checkboxes (keyboard, mouse, headset) appear.

  This is achieved through UI Policies and Client Scripts.

  Example script snippet:

```
function onChange(control, oldValue, newValue) {
  if (newValue == 'High Performance') {
     g_form.setMandatory('justification', true);
  } else {
     g_form.setMandatory('justification', false);
  }
}
```

3.4 Workflow Design

A workflow routes the request from the employee to their manager and then to the IT fulfillment team. The workflow includes approval and task creation stages. Each transition is tracked through ServiceNow’s workflow editor.

3.5 Client Script and UI Policy Design

Client scripts manage the dynamic interactivity of the form, while UI policies handle visibility and mandatory field logic.

A “Reset Form” button is added with the following script:

```
function onResetForm() {
  g_form.clearValue('employee_name');
  g_form.clearValue('department');
  g_form.clearValue('laptop_type');
  g_form.clearValue('justification');
  g_form.clearValue('required_date');
  alert('Form has been reset successfully!');
}
```

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements :

- Users can submit, view, and track laptop requests.
- The system dynamically displays relevant fields.
- Approval routing through manager workflows.
- Audit logs maintained for all actions.

4.2 Non-Functional Requirements

- High system availability.
- Secure access control.
- Scalability for multiple departments.
- Consistent response time under load.

4.3 Hardware and Software Requirements

- Software: ServiceNow Developer Instance, Web Browser
- Hardware: Laptop/PC with Internet Connection

4.4 Use Case Scenarios

1. Employee requests a standard laptop → auto-approval.
2. Employee requests high-performance laptop → routed to manager.
3. Accessories added → additional approval from IT.

4.5 Use Case Name: Laptop Request Submission :

- Primary Actor: Employee
- Precondition: Employee must be logged into the ServiceNow portal.
- Main Flow:

1. Employee navigates to the *Laptop Request Catalog Item*.
2. The system displays a dynamic form with fields such as *Employee Name*, *Department*, *Laptop Type*, and *Justification*.
3. Employee completes the form and clicks Submit.
4. The system validates the input data and creates a request record in the *Service Request* table.
5. The system automatically routes the request to the respective manager for approval.
6. A confirmation message and email notification are sent to the employee.

4.5 Data Flow

Data flows from the user input form to the Request (REQ) and Requested Item (RITM) tables in ServiceNow. Approvals and tasks are recorded in Workflow History.

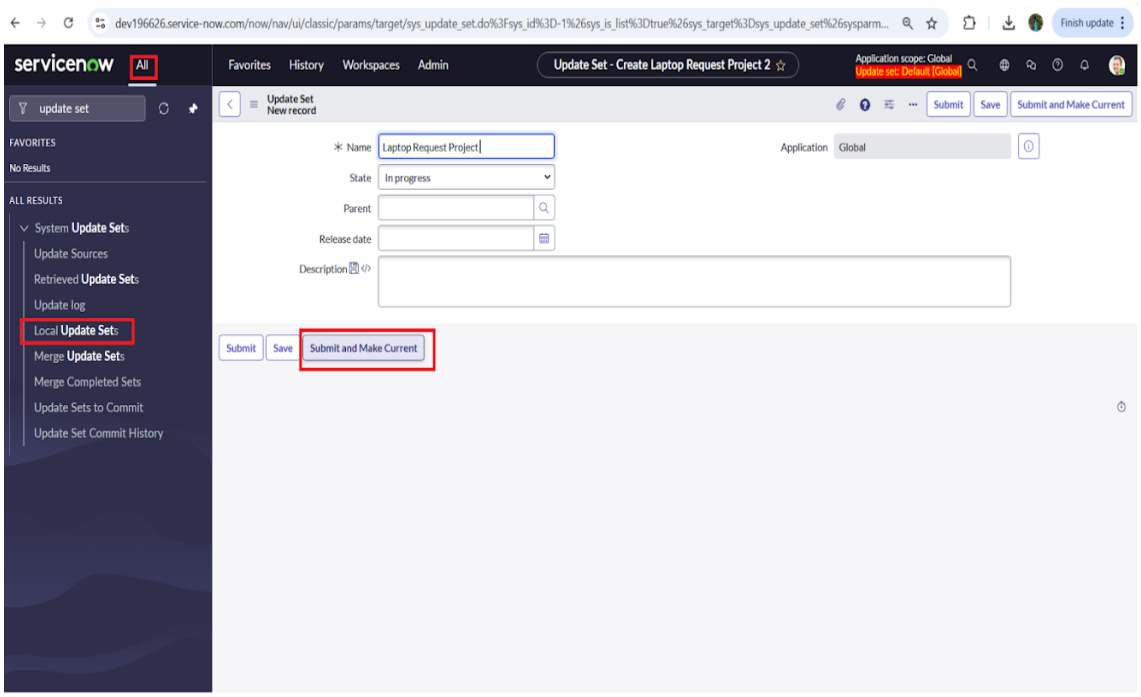
- The **Employee** enters request data through the *ServiceNow Catalog Form*.
- The system validates input fields and stores the data in the **Request Database**.

.

5. PERFORMANCE TESTING

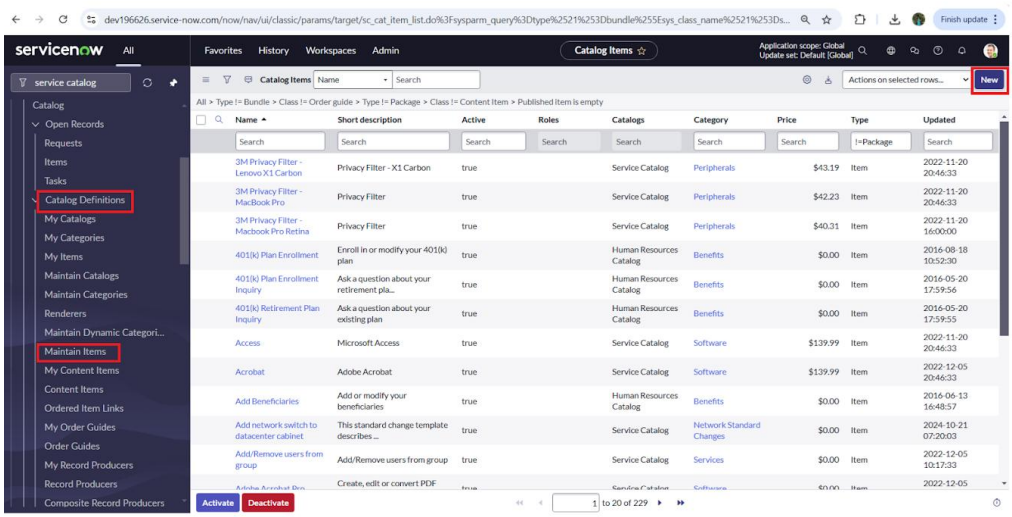
5.1 Create Local Update set :

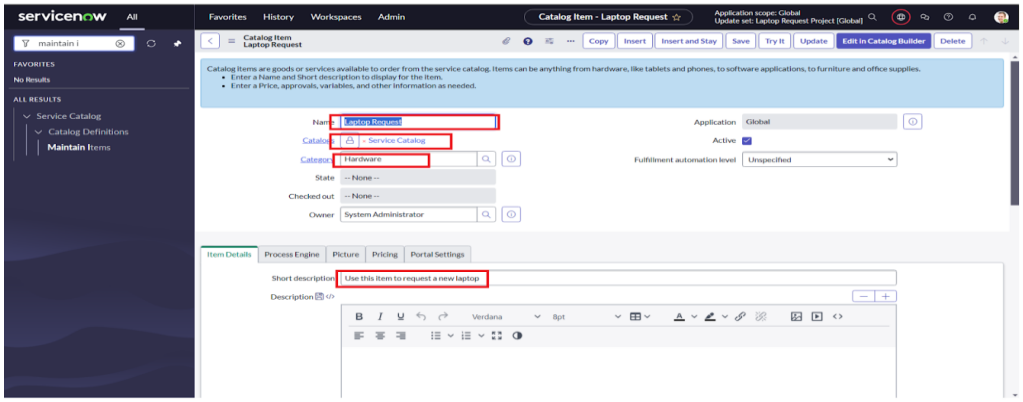
1. Open service now.
2. Click on All >> search for update sets
3. Select local update sets under system update sets
4. Click on new
5. Fill the following details to create a update set as: “Laptop Request”
6. Click on submit and make current
7. By clicking on the button it activates the update set .



5.1 Create Local Update set :

1. Open service now.
2. Click on All >> service catalog

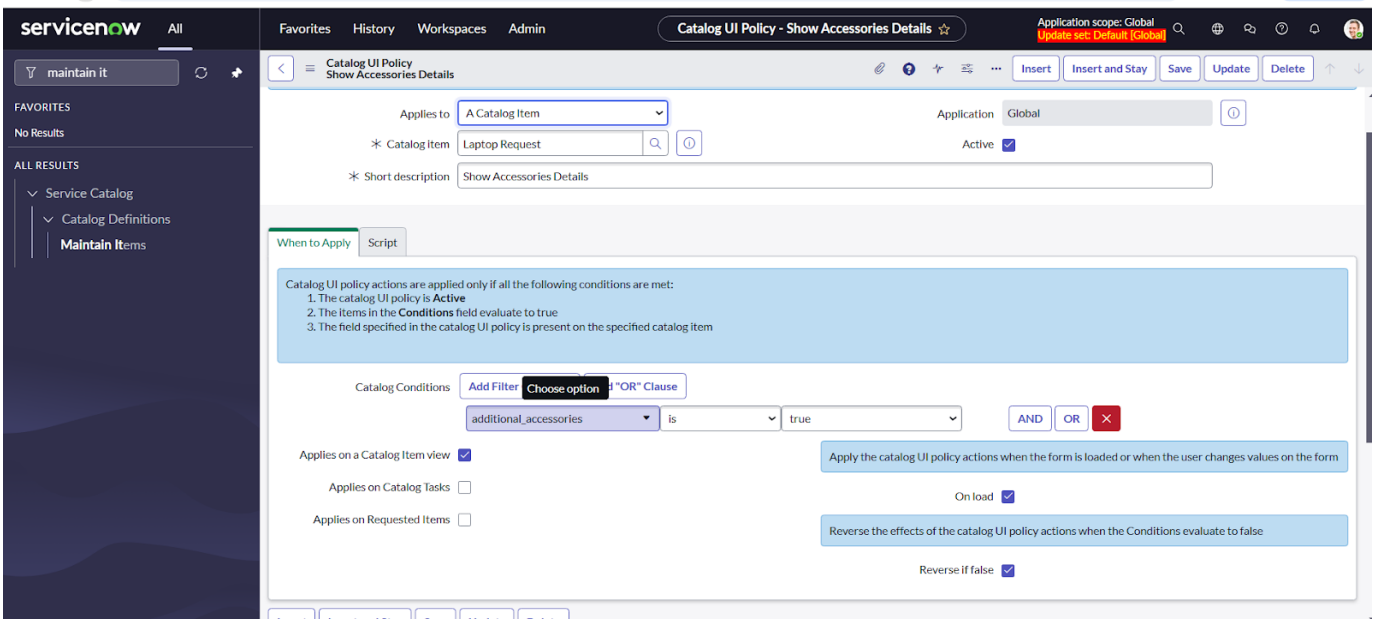




3. Select maintain items under catalog definitions
4. Click on New.

5.3 Create Catalog Ui policies :

1. Click on all>> search for service catalog
2. Select maintain item under catalog definition
3. Search for ‘laptop request’ which is created before
4. Select ‘laptop request’ and scroll down click on “Catalog Ui policies”
5. In the catalog ui policies related list tab click on new
6. Give short description as: show accessories details
7. Set the



8. Click on save.(do not click on submit)
9. Scroll down and select ‘catalog ui action’
10. Then click on new button
11. Select variable name as: accessories_details

Order:100

Mandatory: True

Visible : True

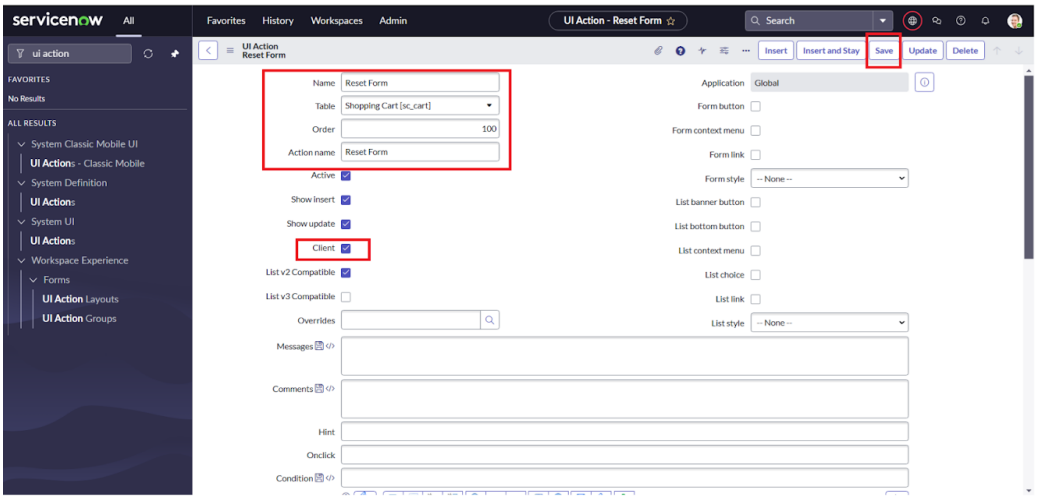
5.4 Create ui action :

1. Open service now.
2. Click on All >> search for ui action
3. Select ui actions under system definition
4. Click on new
5. Fill the following details to create ui action

Table: shopping cart(sc_cart)
Order:100
Action name: Reset form
Client : checked

Script:

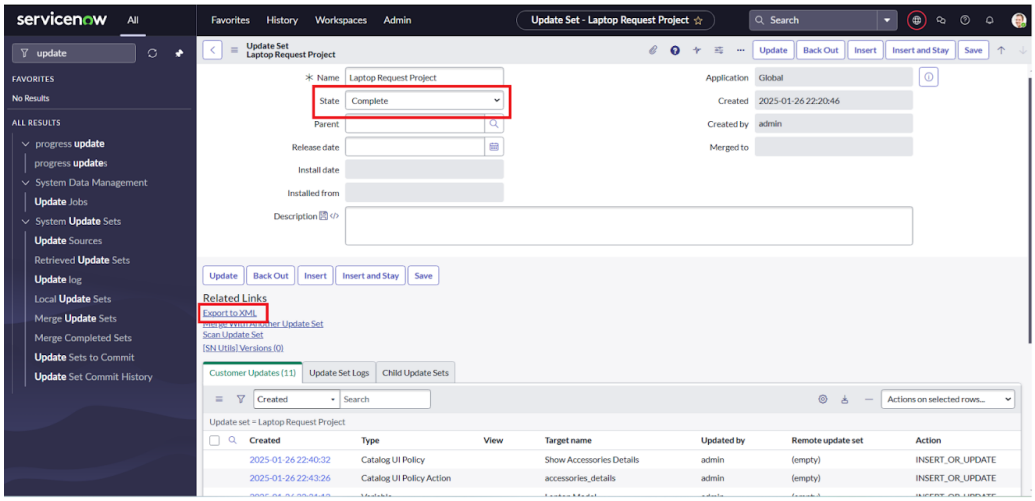
```
function resetForm() {
g_form.clearForm(); // Clears all fields in the form
alert("The form has been reset.");
}
```



Click on save

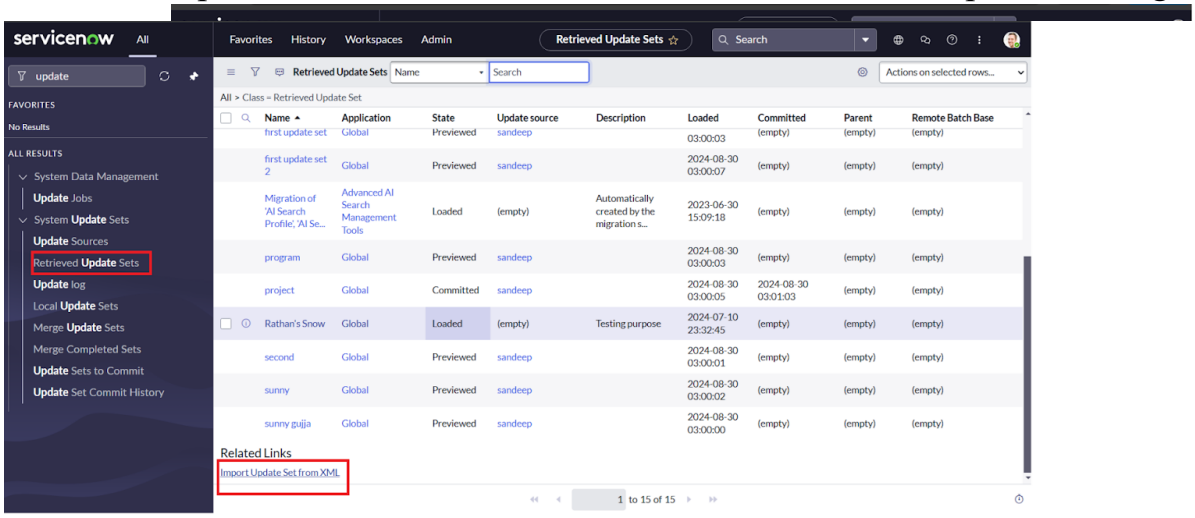
5.5 Explore Update Set :

1. Click on All >> search for update sets
2. Select local update set
3. Select created update set i.e. ‘Laptop Request Project’
4. Set the state to ‘Complete’
5. In the related list Update tab, updates are visible which we perform under this update set.
6. Click on export to XML ,it download one file



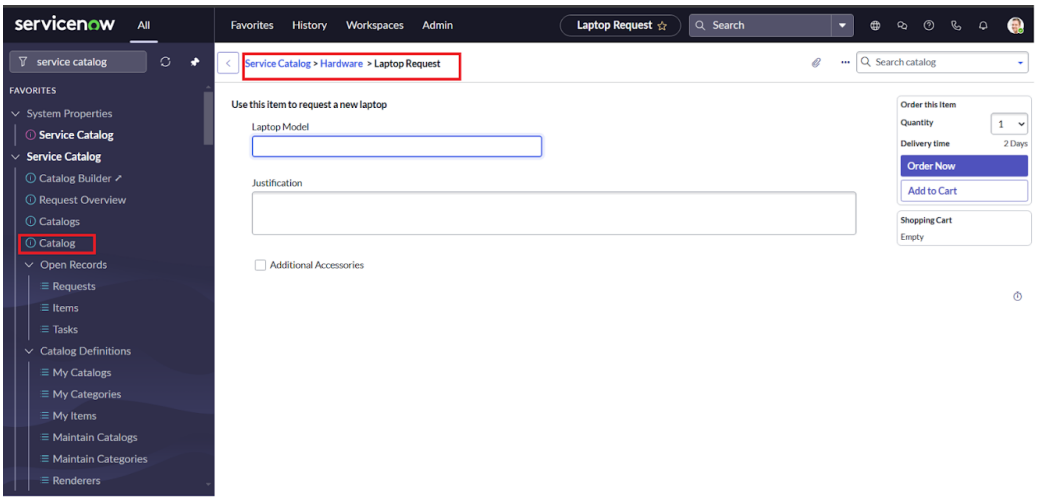
5.6 Retrive The Update Set :

1. Open another instance in incognito window
2. Login with credentials
3. Click on all>> search for update sets
4. Select “Retrieved update set” under system update set
5. It open retrieved update set list and scroll down
6. Click on Import update set from XML
7. Upload the downloaded file in XML file8.Click on Upload and it gets



5.7 Test Catalog :

1. Search for service catalog in application navigator in target instance
2. Select catalog under service catalog
3. Select hardware category and search for ‘laptop request’ item
4. Select laptop request item and open it
5. It shows three variables
6. As per our scenario, when we click on additional accessories checkbox then accessories details fields is visible and that should.



Conclusion :

The Laptop Request Catalog Item project successfully streamlines the process of requesting laptops within the organization by leveraging ServiceNow's Service Catalog capabilities. Through the implementation of a dynamic catalog item, the project ensures that users have an intuitive and user-friendly interface, reducing errors and improving efficiency.This project demonstrates how ServiceNow can be used to replace manual, error-prone processes with automated, efficient, and user-centric solutions. It not only improves service delivery but also enhances employee satisfaction by providing a modern and streamlined request experience.

PRACTICE SCENARIOS FOR SERVICENOW ADMIN

1. **Create a new user for a contractor, assign them to an "IT Support" group, and ensure they can only access the *Incident* application.**
   Solution:

   - **Create the Contractor User**

     - Navigate to **Users** → *User Administration* > *Users*.
     - Click **New**.
     - Fill in details:
       - **User ID**: contractor1
       - **First name / Last name**: Contractor User
       - **Email**: contractor1@gmail.com
       - **Active**: Checked.
     - Save.

   - **Assign the User to the "IT Support" Group**

     - On the user record, scroll to **Groups** (related list).
     - Click **Edit**.
     - Add to the **IT Support** group.
     - Save.

   - **Restrict Access to Only the Incident Application**

     Now we need to make sure this contractor can only work with **Incident**.

     Option A: Role-Based Control (Mostly Preferred)

     - By default, Incident application requires **itil** role.
     - Instead of giving full **itil** access (which gives too much), do the following:
       - Create a **new custom role**, ex: **incident_contractor**.
       - Assign this role only to permissions needed for Incident (using ACLs).
       - Assign the new role to your contractor user.
       - Do **not** give **itil** or other broad roles.

Option B: Application Menu Restriction

- Go to **System Definition > Application Menus**.
- Open the **Incident** application menu.
- In the **Roles** field, add your custom role (**incident_contractor).**
    - This ensures only users with this role can see the Incident.
    - Verify Access
- **Impersonate** the contractor user.
- Check:
    - They should only see the **Incident application** in the left nav.
    - They can open/create/edit incidents (based on the ACLs you configured).
    - They cannot access other apps (like Change, Problem, etc.).

2. Assign a role to a new group so members can read *Knowledge Articles* but cannot create or edit them.

- **Create a New Group**

  - Navigate to User Administration > Groups.
  - Click New.
  - Enter a Name for the group (e.g., Knowledge Readers).
  - Optionally, add a Description.
  - Click Submit.

- **Assign the Appropriate Role**

To allow read-only access to Knowledge Base articles, assign the **knowledge** role:

  - Open the newly created group.
  - Scroll to the Roles related list.
  - Click Edit.
  - Add the role: knowledge
    - This role allows users to view published articles.
  - Click Save.

****Do NOT assign roles like **knowledge_admin or knowledge_manager,** which grant create/edit permissions.

- **Add Users to the Group**

  - In the group record, scroll to the Group Members related list.
  - Click Edit.
  - Select users you want to add.
  - Click Save.

- **Verify Access**

  - Log in as one of the group members.
  - Navigate to Knowledge > Articles.
  - Confirm they can view articles.
  - Try creating or editing an article — they should not have access.

3. **Configure a UI Policy that hides the "Work Notes" field unless the state is "In Progress"**.
Solution:

- **Navigate to UI Policies**
- Go to Application Navigator → type UI Policies → click System UI > UI Policies.
- Create a New UI Policy
- Click New.
- Select the Table → e.g., *Incident* (or whichever table you're working on).
- Provide a Name (e.g., *Hide Work Notes unless In Progress*).
- In the Conditions section, set:
    - Field = *State*
    - Operator = *is*
    - Value = *In Progress*.
- Check the box Active.
- Save the record.
- **Add a UI Policy Action**
- In the same UI Policy record, scroll to UI Policy Actions (Related List).
- Click New.
- Configure the action:
    - Field name = *Work notes*
    - Visible = *True* (since you want it visible only when the condition is met).
- Submit the action

4. **Configure a UI Policy to hide Notes section in incident, when state is In Progress.**

Solution:

- **Navigate to UI Policies**
- Go to Application Navigator → type UI Policies → click System UI > UI Policies.
- Create a New UI Policy
- Click New.
- Select the Table → e.g., *Incident* (or whichever table you're working on).
- Provide a Name (e.g., *Hide Work Notes unless In Progress*).
- In the Conditions section, set:
    - Field = *State*
    - Operator = *is*
    - Value = *In Progress*.
- Check the box Active.
- Save the record.
- **Make Run Script box True**
- Just write one line of code:
- g_form.setSectionDisplay('notes',false);
- Submit the action.

5. **Configure a response SLA, the SLA should pause, when the incident state is in On Hold vice versa**.

Create or Modify an SLA Definition

- Navigate to **Service Level Management > SLA Definitions**.
- Click **New** or open an existing SLA (e.g., "Response SLA").
- Fill in the basic details:
    - **Name**: Response SLA
    - **Table**: **Incident**
    - **Type**: Response
    - **Duration**: Set your desired time (e.g., 1 hour)
    - **Set SLA Conditions**
- Under the **Start Condition**:
    - Example: **State is New**
- Under the **Stop Condition**:
    - Example: State is Resolved or Closed
- Under the **Pause Condition**:
    - Add: **State is On Hold**

This ensures the SLA timer **pauses** when the incident is moved to **On Hold**, and **resumes** when it returns to another **New** state

- **Test the SLA Behavior**
- Create a test incident.
- Confirm SLA starts when an incident is created.
- Change state to **On Hold** — SLA should pause.
- Change back to **Active** — SLA should resume.
- Resolve the incident — SLA should stop.

6.**Configure an email notification that alerts the assigned group whenever a new *Change Request* is created.**

Solution:

- **Navigate to Notifications**
- In the **Application Navigator**, type **Notifications**.
- Go to **System Notification > Email > Notifications**.
- **Create a New Notification**

1. Click **New**.
2. Fill in the basic details:
    a. **Name**: *New Change Request Assigned Group Alert*
    b. **Table**: *Change Request [change_request]*
    c. **Active**: Checked

- **Define When to Send**

1. Under **When to send**, configure:
    a. **When to send**: *Insert* (since you want this when a new record is created).

- **Define Who Will Receive**

1. In the **Recipients** tab:
    a. Under **Users/Groups in fields**, choose **Assigned to group** (or the field name for assigned group).
    b. This ensures the entire assigned group gets the email.

- **Define What Will Contain**
- In the **What it will contain** tab:

Please review and take necessary action.

- **Save & Test**
- Save the Notification.
- Create a new **Change Request** record, assign it to a group.
- Verify that the email goes out to all members of the Assigned Group.

7. **Create a report showing the number of incidents opened by each department in the last 30 days.**

- **Navigate to Reports**
- Go to Reports > Open Reports Modules.
- Click Create a Report.
- **Define Report Source**
- Name: **Incidents by Department - Last 30 Days**
- Source Table: **Incident**
- **Set Conditions**
- **Under Filter, add:**
    - Opened At → on or after → Today - 30 days
    - Department → is not empty ***(optional, to exclude unassigned)***
- **Choose Report Type**
- Select Type: **Bar Chart** or **Pie Chart** (or **List** if you prefer tabular view)
- **Configure Grouping**
- Under Group By, select: **Department**
- Under Aggregation, choose: **Count**
- **Save and Run**
- Click Save.
- Click Run to view the report.

8. **Build a dashboard for Service Desk Managers showing KPIs like incidents by priority, created within a week, state wise also.**

Step 1: Create Individual Reports

You'll need to create three separate reports first:

- **Incidents by Priority**
- Go to: Reports > Create New
- Name: Incidents by Priority
- Type: Bar Chart or Pie Chart
- Group By: Priority
- Filter: Opened At → on or after → Today - 30 days
- **Incidents Created Within a Week**
- Name: Incidents Created - Last 7 Days
- Source Table: Incident
- Type: Time Series or Bar Chart
- Filter: Opened At → on or after → Today - 7 days

Step 2: Create a Dashboard

- Go to Self-Service > Dashboards.
- Click Create New Dashboard.
- Name: **Service Desk Manager KPIs**
- Add a Proper Description
- Click Submit.

Step 3: Add Reports to the Dashboard

1. Open the newly created dashboard.
2. Click Edit Content.
3. Use Add Reports to include:
    - **Incidents by Priority**
    - **Incidents Created - Last 7 Days**
    - **Incidents by State**

9. **Restrict the ability to delete records in the *Change Request* table so only users with the "admin" role can do so.**

- **Navigate to Access Control (ACLs)**
- In the **Application Navigator**, type **Access Control**.
- Go to **System Security > Access Control (ACL)**.
- **Create a New ACL Rule**
- Click **New**.
- Fill in details:
    - **Type**: *record*
    - **Operation**: *delete*
    - **Table**: *Change Request [change_request]*
    - **Name**: *(auto-populates when you pick table + operation)*
- **Define the Condition / Role**

In the **Requires role** field, add: **admin**

    - This ensures only users with the **admin** role can delete records.

- **Save & Test**
- Save the ACL.
- Test with a non-admin user → they should **not** see the delete option (or get a permission error if they try via URL).
- Test with an admin user → delete should work normally.

10. **Create a custom table and create two reference fields (ex: assignment group and assigned to). Display the users based on selection of assignment group.**

- **Create a Custom Table**

1. In the Application Navigator, type **Tables**.
2. Go to **System Definition > Tables**.
3. Click **New**.
    - Name: *u_custom_task*
    - Label: *Custom Task*.
    - Save.

- **Add Fields**

1. Open your table and go to the **Columns** tab.
2. Add two reference fields:
    - **Assignment Group** → Type = *Reference*, Table = *sys_user_group*.
    - **Assigned To** → Type = *Reference*, Table = *sys_user*.

- **Configure Reference Qualifier on "Assigned To"**
- We need to filter "Assigned To" users based on the selected Assignment Group.

Using Reference Qualifier

- Right click on the **Assigned To** field, click on **Configure Dictionary.**
- Go to **Dependent** Section, give the name of the Assignment Group(ex: u_ass_group)
- Update and Test the functionality.

11.**How to auto assign incidents when user selects a category as network, the same incident be assigned to Network group.**

Solution:

1. Go to Flow Designer → Designer.
2. Click New Flow.
    - Name: Assign Incident by Category
    - Trigger: Created or Updated → Table = Incident
3. Add a If action (Condition) with expression:
    - Select Trigger Record Category is Network
4. Under the If branch, add Action → Update Record:
    - Record: Trigger → Incident(Trigger Record)
    - Set field Assignment group → Network
5. Save and Activate the flow.
6. Test the Flow.

12. HR Groups members are only able to see HR Related Records in servicenow?

Solution:

Step 1: Create a Role for HR Access

Navigate to:

User Administration → Roles → New

1. Enter:
    - Name: hr_access
    - Description: Role to allow access to HR Cases
2. Click Submit.

Step 2: Assign the Role to HR Group

1. Navigate to:
   User Administration → Groups
2. Open your HR group record.
3. In the Roles tab → click Edit.
4. Move hr_access from Available → Selected.
5. Click Save.

Now all members of the HR group have the hr_access role.

Step 3: Create Access Control (ACL) for Viewing HR Cases

1. Navigate to:
   System Security → Access Control (ACL)
2. Click New.

Fill in:

Field	Value
Type	record
Operation	read
Table	Your HR Case table
Active	True

Step 4: Define Access Condition (No Script)

Scroll down to the Requires role section:

- Add the Role hr_access.

This means only users with the hr_access role can read/view HR Case records.

Step 5: Save and Test

1. Click Submit or Update to save the ACL.
2. Impersonate a non-HR user:
    - Go to your profile → click Impersonate User → choose a user *not in the HR group*.
    - Try opening an HR Case record → You should see a "Security constraints prevent access to requested page" message.
3. Now impersonate an HR group member:

They should be able to open HR Cases normally

13. When the Incident state changes to In Progress, Child incident related list should be hidden.

Solution:

1. Navigate to System UI → UI Policies → New.
2. Fill the header:
    - Name: Hide related lists when State is In Progress
    - Table: Incident
    - Active: checked
    - Global: checked
3. Condition: **State is In Progress**

(Use the exact label used in your instance for the In Progress state.)

4. Submit the UI Policy record.
5. In the UI Policy record click **New** under **UI Policy Actions**.

Set:

- **Field name:** select the related list–Child incident
- **Visible:** false
- **Read only:** optional
- Save and Test the UI Policy Action.

14.How to Display Incident number while loading the incident form

Solution:

1. Navigate to System UI → Client Scripts → New.
2. Fill the header:
    - Name: Show Incident Number on Load
    - Table: Incident
    - Type: onLoad
    - Active: True
3. Add this script:

```
function onLoad() {
  // Get the Incident number field value
  var incNum = g_form.getValue('number'); // 'number' is the field name
  alert('Incident Number: ' + incNum);
}
```

15. When the Incident state changes to In Progress, description should be hidden and short description should be mandatory.

Solution:

Step 1 — Navigate to Client Scripts

1. Go to:

   System UI → Client Scripts → New
2. Fill the header:
   - Name: Hide Description and Make Short Description Mandatory
   - Table: Incident
   - Type: onChange
   - Field name: state
   - Active: checked

Step 2 — Add the Client Script Code

```
function onChange(control, oldValue, newValue, isLoading) {
  if (isLoading) return;
  if (newValue === '2') {
     g_form.setDisplay('description', false);
     g_form.setMandatory('short_description', true);
  } else {
      g_form.setDisplay('description', true);
     g_form.setMandatory('short_description', false);
  }
```

16. Users can not change the state field values in the incident list.

Solution:

Step 1 — Navigate to Client Scripts

3. Go to:

   System UI → Client Scripts → New

4. Fill the header:
   - Name: Prevent State Inline Edit
   - Table: Incident
   - Type: onCellEdit
   - Field name: state
   - Active: checked

Step 2 — Add the Client Script Code

```
if(newValue==2){

alert('You can not edit this value');

saveAndClose==false;

}else{

saveAndClose==true;

}
```

17. How to set the Caller to Logged in user automatically in the incident table.

Solution:

1. Navigate: System Definition → Business Rules → New
2. Fill the details:
    - Name: Set Caller on Incident Create
    - Table: Incident
    - When: before
    - Insert/update: checked
    - Advanced: true
3. **Script**:

```
current.caller_id = gs.getUserID();
```

18.When a user updates an incident record, priority should change to Critical automatically.

Solution:

1. Navigate: System Definition → Business Rules → New

2. Settings:
    - Name: Set Priority field
    - Table: Incident
    - When: before
    - Update:checked
3. Script:

```
current.impact = 1;

current.urgency = 1;
```

19.Create a button on the Incident form that allows users to mark an Incident as Resolved with a single click.

Solution:

1. Navigate: System UI → UI Actions → New
2. Settings:
    - Name: Resolve Incident
    - Table: Incident
    - Action type: Form button
    - Active: checked
3. Script:
    - current.state = 6;
    - current.update();
    - action.setRedirectURL(current);

20.Create a button on the incident table that copies the Short Description value into the Description field.

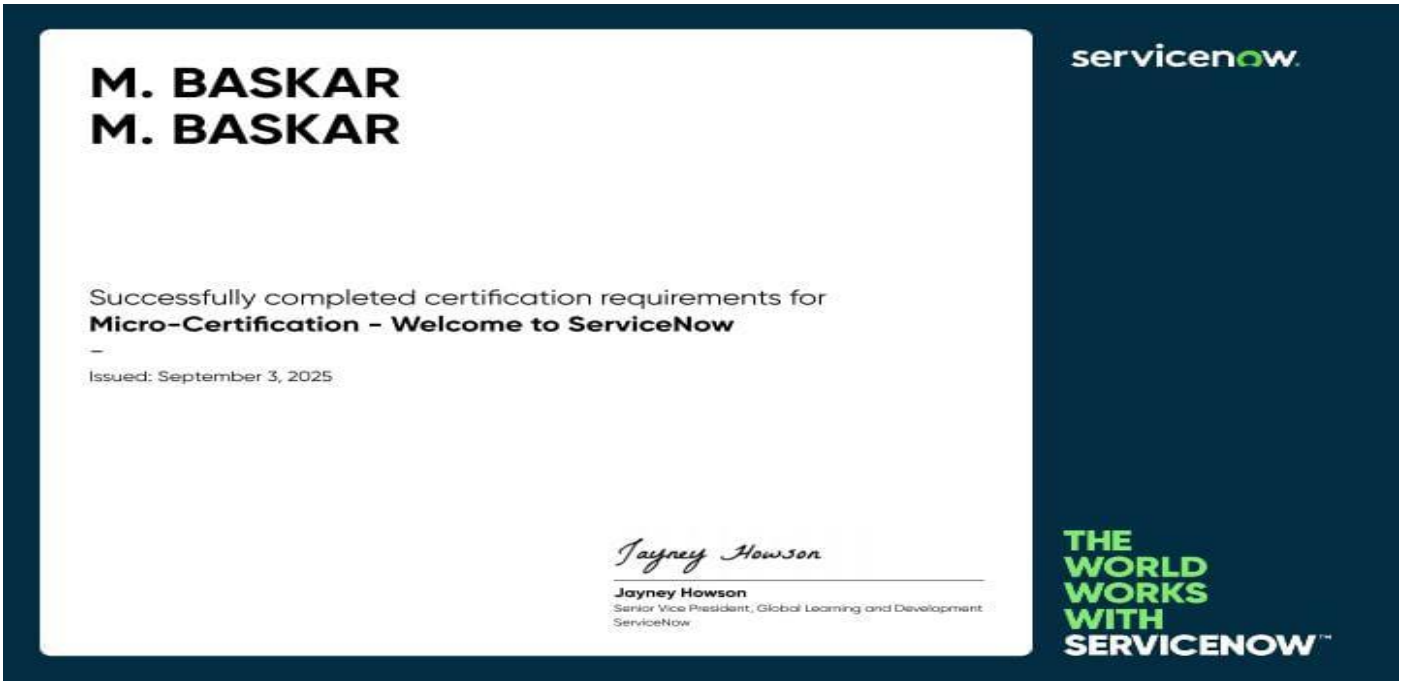
Solution:

1. Navigate: System UI → UI Actions → New
2. Settings:
    - Name: Copy Short Description
    - Table: Incident
    - Action type: Form button
    - Active: checked
3. Script:
    - current.description = current.short_description;
    - current.update();
    - action.setRedirectURL(current);

CERTIFICATES

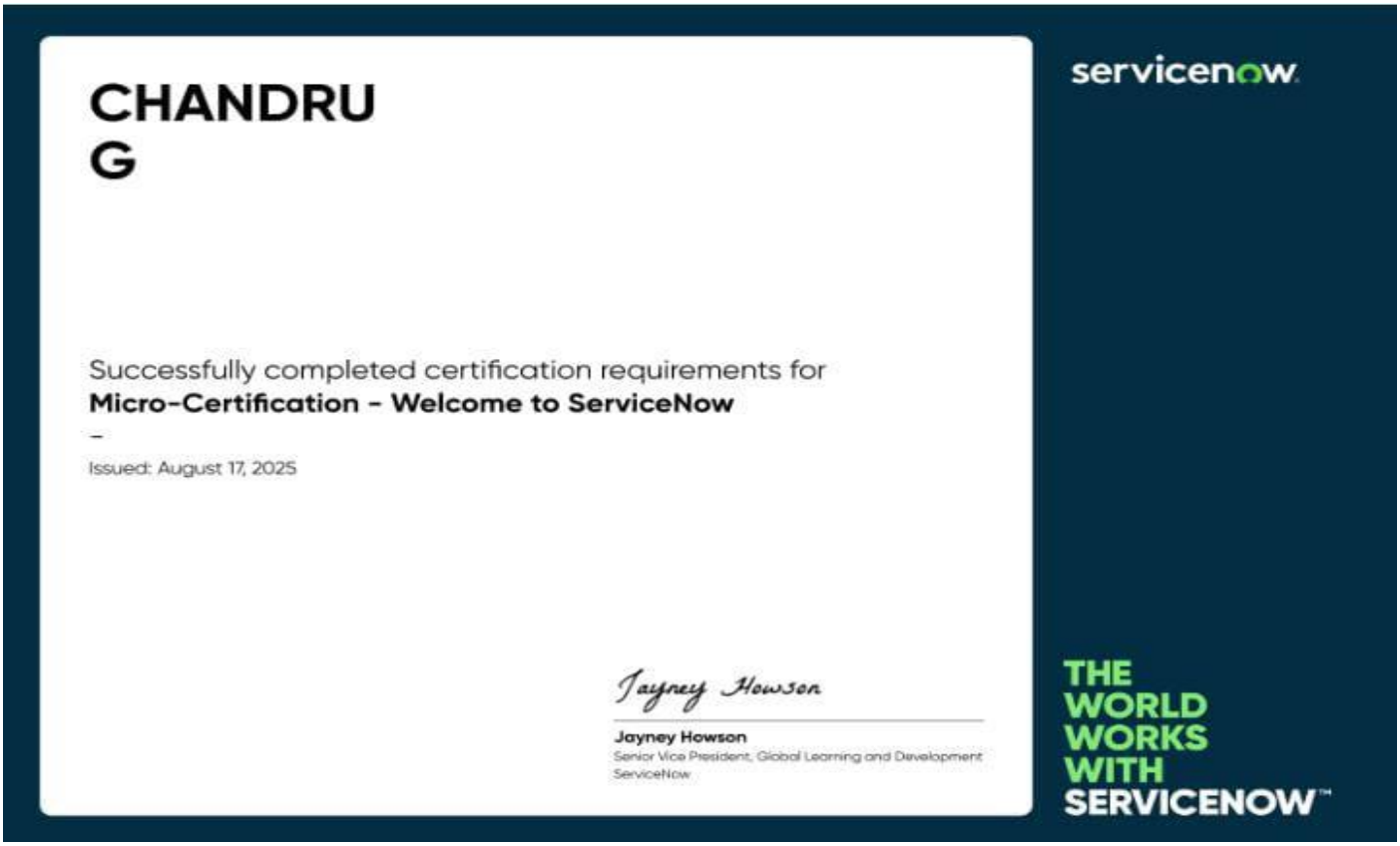
1) Generative AI in Action :

2) Welcome to Service Now – Micro Certification :



1) **Generative AI in Action :**

2) Welcome to Service Now – Micro Certification :



1) **Generative AI in Action :**

2) Welcome to Service Now – Micro Certification :

1) **Generative AI in Action :**

2) Welcome to Service Now – Micro Certification :