NM1051 – SERVICENOW ADMINISTRATOR – SMART INTERNZ

OPTIMIZING USER,GROUP,AND ROLE MANAGEMENT ACCESS CONTROL AND WORKSFLOW

A PROJECT REPORT

Submitted by

SIVAKUMAR S (815422104052)
MARIMUTHU P (815422104027)
PERIYASAMY C (815422104033)
RAJESHWARAN P (815422104303)

BACHELOR OF ENGINEERING

IN

SEVENTH SEMESTER

COMPUTER SCIENCE AND ENGINEERING

SRI RAMAKRISHNA COLLEGE OF ENGINEERING
ANNA UNIVERSITY : CHENNAI 600 025

NOV/DEC 2025

BONAFIDE CERTIFICATE

Certified that this project report **"OPTIMIZHING USER GROUP,AND ROLE MANAGEMENT ACCESS CONTROL AND WORKFLOW"** is the bonafide work of **"SIVAKUMAR S,MARIMUTHU P,PERIYASAMY C,RAJESHWARAN P"** 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 CONTENTS

IDEATION PHASE

Problem Statement

During brainstorming sessions, multiple stakeholders — including IT administrators, security officers, and business process managers — were involved to gather diverse perspectives on the problem. This collaborative approach helped in understanding real-world scenarios such as delayed access requests, insufficient role transparency, and unauthorized privilege escalations. The discussions highlighted the need for a **centralized and intelligent system** that could manage users and permissions dynamically while maintaining strict security and audit standards.

Problem Definition

In modern organizations, managing users, groups, and roles effectively has become a critical aspect of maintaining security, compliance, and operational efficiency. As enterprises grow and adopt multiple systems and applications, user access management becomes increasingly complex. Many organizations still rely on **manual or semi-automated methods** for creating user accounts, assigning roles, and approving access requests, which often leads to **delays, human errors, and inconsistent permission levels**.

The absence of a **centralized access control system** makes it difficult to maintain a unified view of user privileges across departments or platforms. As a result, employees might have **excessive or outdated permissions**, exposing the organization to potential security breaches or data misuse. Furthermore, onboarding and offboarding processes are often time-consuming, with approvals and role assignments passing through multiple manual checkpoints, causing inefficiency and lack of accountability.

Another major issue arises from the **lack of structured workflows** for managing user access. Without automated approval processes or well-defined authorization hierarchies, tracking and auditing access requests becomes difficult. This not only impacts day-to-day productivity but also poses challenges during internal audits and compliance checks. IT administrators face difficulty ensuring that access rights align with employees' roles and responsibilities, leading to violations of the **principle of least privilege**.

Abstract

In today's digital environment, organizations rely on multiple systems and applications that require strict and efficient user access management. However, traditional approaches to managing users, groups, and roles are often **manual, time-consuming, and prone to errors**, leading to security vulnerabilities and administrative inefficiencies. This project focuses on **optimizing user, group, and role management** through the implementation of **centralized access control and automated workflows**.

The proposed system aims to simplify and automate critical identity management operations such as **user onboarding, role assignment, access approval, and deprovisioning**. By introducing

workflow automation, access requests can be processed with predefined approval hierarchies, reducing delays and ensuring policy compliance. Furthermore, the solution integrates **role analytics** to identify redundant or conflicting permissions and suggest optimal access structures based on actual usage data.

Empathy Map canvas

Section	Details
User Persona	**Primary Users:** IT Administrators, Department Managers, and Employees/End-users. **Goal:** To efficiently manage user access, roles, and permissions with better security, automation, and compliance.
What They Hear	- "Access approval is taking too long." - "We must ensure compliance during audits." - "Unauthorized access can cause major security issues." - "Employees are waiting for access activation or removal."
What They See	- Multiple systems used separately for user and role management. - Manual, repetitive access approval processes. - Frequent role conflicts and access mismatches. - Delayed onboarding and deprovisioning of users.
What They Say & Do	- "It's hard to track who has what access." - "Managing permissions manually is time-consuming." - "We need a single dashboard for access control." - Manually submit access requests or send emails for approval. - Regularly verify user roles across systems.
What They Think & Feel	**Frustrations:** - Manual tasks are error-prone and inefficient. - Concern about data breaches and excessive privileges. - Stress during audits due to poor visibility. **Needs/Desires:** - Centralized user and role management system. - Automated workflows for quick approval. - Real-time access visibility. - Secure, compliant, and simple-to-use system.
Pain Points	- Time-consuming manual processes. - Lack of real-time visibility and audit trails. - Redundant or conflicting roles. - Difficulty maintaining compliance.

	- Poor integration between HR and IT systems.
Gains / Opportunities	- Centralized and automated access control system. - Workflow-driven approvals and deprovisioning. - Reduced admin workload and faster onboarding.

Section	Details
	- Improved transparency and compliance reporting. - Role analytics to detect redundant permissions.
Empathy Insight	Users need an **automated, centralized, and workflow-driven access management platform** that enhances visibility, reduces manual effort, strengthens security, and ensures compliance across the organization.

Introduction

In the modern digital era, organizations operate across multiple platforms, applications, and cloud environments, each requiring secure and efficient user access management. As businesses expand, the number of users, roles, and permissions also grows rapidly, creating challenges in maintaining data security, regulatory compliance, and administrative efficiency. Managing users, groups, and roles manually often leads to inconsistencies, unauthorized access, and operational delays, making it essential to adopt an automated and intelligent approach to access control.

Effective user and role management is a critical component of **Identity and Access Management (IAM)** systems. It ensures that every user in an organization has the right level of access based on their job role and responsibilities. However, traditional systems often lack automation and workflow integration, resulting in repetitive tasks and an increased risk of security breaches. Without a centralized system, administrators struggle to track access permissions, leading to privilege misuse and compliance violations.

To address these challenges, this project focuses on **optimizing user, group, and role management** through the implementation of **automated access control mechanisms and workflow-based approvals**. The proposed solution aims to streamline operations such as user onboarding, role assignment, and access revocation by integrating rule-based workflows, audit trails, and compliance monitoring. This ensures that access privileges are granted or revoked automatically according to organizational policies and employee status.

By introducing a **centralized access management system**, organizations can achieve greater transparency, accountability, and operational efficiency. Additionally, the inclusion of **role analytics and periodic access reviews** helps identify redundant or conflicting roles, ensuring the principle of least privilege is consistently enforced. The automation of workflows reduces manual

errors, enhances data protection, and aligns access management with corporate governance standards.

Ultimately, the project aims to design a **secure, scalable, and workflow-driven access control solution** that empowers administrators, managers, and employees alike. This optimization not only improves the efficiency of IT operations but also strengthens overall information security and regulatory compliance within the organization.

Objectives

The main objective of this project is to design and implement a **centralized, automated, and workflow-driven system** for efficient user, group, and role management that enhances security, compliance, and administrative efficiency across the organization.

Specific Objectives:

1. **To centralize user, group, and role management** ○ Develop a unified platform that consolidates user and role information across
2. **To automate access control workflows** ○ Implement automated workflows for user onboarding, access approval, and deprovisioning processes to reduce manual effort and processing time.
3. **To ensure data security and compliance** ○ Enforce access control policies that align with organizational and regulatory standards, minimizing the risk of unauthorized access and policy violations.
4. **To enhance transparency and accountability** ○ Provide real-time tracking of user access requests, approvals, and role changes through detailed audit trails and reports.
5. **To optimize role structures using analytics** ○ Identify redundant, conflicting, or unused roles using data-driven analysis and suggest improvements to maintain the principle of least privilege.
6. **To improve operational efficiency** ○ Reduce the administrative burden on IT teams by automating repetitive tasks and providing self-service access request capabilities for users and managers.
7. **To integrate with existing systems and workflows** ○ Enable seamless integration with IT Service Management (ITSM), HR systems, and directory services (such as LDAP or Azure AD) for synchronized user lifecycle management.
8. **To support scalability and flexibility** ○ Design the solution to accommodate organizational growth and evolving access control requirements without significant reconfiguration.

Scope

The scope of this project covers the design, development, and implementation of an **automated access management system** that streamlines the administration of users, groups, and roles within an organization. The project focuses on integrating **access control policies** and **workflow automation** to enhance security, reduce manual workload, and ensure compliance with organizational standards.

This system will provide a **centralized platform** for managing user accounts, assigning roles, and defining permissions based on departmental hierarchy and job functions. It will automate routine processes such as **user onboarding, role assignment, access approval, and account deactivation**, ensuring that every change is tracked and authorized through a structured workflow.

The project also includes the development of **role analytics** to identify redundant, conflicting, or unused roles. By analyzing access patterns, the system can recommend optimization strategies to maintain proper segregation of duties and enforce the **principle of least privilege**.

The scope further extends to building a **self-service access portal** where users can request specific permissions, and managers can approve or reject requests through an automated workflow. This minimizes administrative intervention and enhances transparency and accountability across the organization.

Additionally, the project will incorporate **audit trails, reporting features, and compliance checks** to support periodic reviews and ensure that access rights remain aligned with organizational and regulatory requirements. Integration with existing IT systems such as **HR databases, directory services, or IT Service Management (ITSM) tools** will also be considered to enable seamless user lifecycle management.

However, the scope of the project is limited to the **design and prototype implementation** of the proposed access management model. It will not cover large-scale deployment or integration with commercial enterprise IAM systems. Future enhancements may include AI-driven role recommendations, risk-based access control, and multi-factor authentication integration.

PROJECT PLANNING PHASE

The **Project Planning Phase** serves as a blueprint for the successful execution of the project. It involves defining the project's goals, scope, deliverables, timelines, resources, and responsibilities. For this project, the planning phase focuses on establishing a structured approach to design, develop, and implement an **automated user, group, and role management system** integrated with **access control and workflows**.

The planning phase begins with an understanding of the **existing problems** in manual user and role management processes — such as inconsistent permissions, delayed approvals,

and lack of visibility. Based on this, a **clear roadmap** was created outlining each stage of development, from requirement gathering to deployment and evaluation.

Procedure or Implementation steps

Phase 1: Creation of a New Update Set

1. Navigate to **All → Local Update Sets** using the filter navigator.
2. Click on **New** to create a new update set.
3. Enter the following details:
    - **Name:** Family Expenses
4. Click on **Submit** to save the update set.

Phase 2: Creation of a New Update Set (and Make Current)

1. Go to **All → Local Update Sets** again and click on **New**.
2. Enter the following details:
    - **Name:** Family Expenses
3. Click on **Submit**.
4. After submission, click on **Make Current** to activate this update set as the working update set.

Phase 3: Creation of Table

Step 1: Creation of Family Expenses Table

1. Go to **All → Tables** using the filter navigator.
2. Click on **New** to create a new table.
3. Fill in the following details:
    - **Label:** Family Expenses ○ **Name:** (Auto-Populated) ○ **New menu name:** Family Expenditure
4. Go to the **Header**, right-click, and select **Save** to create the table.

Step 2: Creation of Columns (Fields)

1. Under the **Columns** section, double-click on **Insert a new row**.
2. Enter the details as:
    - **Column Label:** Number
    - **Type:** String
3. Again, double-click on **Insert a new row** to add another column.
4. Enter the details as:
    - **Column Label:** Date ○ **Type:** Date/Time

5. Continue adding additional columns as needed for the Family Expenses table (for example: Amount, Category, Description, etc.).

Phase 4 : Form Design and Workflow Configuration

Step 1: Form Design

1. Navigate to **All → Tables → Family Expenses**.
2. Open the newly created **Family Expenses** table.
3. From the table configuration, click on **Form Layout**.
4. In the **Available Fields** section, move the following fields to the **Selected Fields** area to appear on the form: ○ Number ○ Date ○ Expense Type
    - ○ Amount
    - ○ Description ○ Status
5. Click on **Save** to finalize the form layout.
6. Open the form to verify that all fields are displayed properly and in the desired order.

Step 2: Adding Auto Number for Expense Records

1. Navigate to **System Definition → Number Maintenance**.
2. Click on **New** to create a number prefix for Family Expenses.
3. Enter the following details:
    - ○ **Table:** Family Expenses ○ **Prefix:** FAMEXP
    - ○ **Number:** Auto-incrementing format (e.g., FAMEXP001, FAMEXP002, …)
4. Click **Submit**.
5. This ensures that each expense entry automatically receives a unique number.

Step 3: Workflow Creation

1. Navigate to **All → Workflow → Workflow Editor**.
2. Click on **New Workflow**.
3. Enter the details as follows:
    - ○ **Name:** Family Expenses Approval Workflow ○ **Table:** Family Expenses
4. Click on **Submit** to create the workflow.

Step 4: Designing the Workflow

1. In the Workflow Canvas, drag and drop the following activities:
    - **Begin Activity** – Marks the start of the workflow. ○ **Approval Activity** – Sends the record for manager approval. ○ **Update Record Activity** – Updates the record's status field (e.g., Approved / Rejected).
    - **End Activity** – Marks the completion of the workflow.
2. Connect the activities in the following order:
   Begin → Approval → Update Record → End.
3. Set approval conditions:
    - If **Approved:** Status changes to "Approved." ○ If **Rejected:** Status changes to "Rejected." 4. Click on **Publish** to activate the workflow.

Step 5: Testing the Workflow

1. Go to the **Family Expenses** module and create a new record.
2. Enter the details such as Date, Amount, Expense Type, and Description.
3. Click **Submit**.
4. The workflow automatically triggers an approval request to the assigned approver.
5. Upon approval or rejection, the system updates the **Status** field accordingly.
6. Verify workflow logs to confirm successful execution.

Step 6: Validation and Finalization

1. Test multiple records to ensure that the workflow runs smoothly for all users.
2. Validate that email notifications (if configured) are sent correctly.
3. Verify that access control rules restrict unauthorized users from modifying approved records.
4. Save and document all workflow configurations for future maintenance.

```
(function refineQuery(current, parent) {



// Add your code here, such as current.addQuery(field,
value); current.addQuery('u_date',parent.u_date);
current.query();



})(current, parent);
```

Testing Phase with Screenshots

Testing ensures that all modules — user management, access control, and workflows — function as expected and meet the system requirements.
Below is the recommended structure and description for your **testing screenshots section**.

1. Login Page Verification

Purpose:
To verify that users can log in securely and access their dashboards based on assigned roles.

Screenshot 1:

- Show the **ServiceNow Login Page**
- Entering valid credentials for Admin/User
- Click "Login" **Expected Result:**

- The system should navigate to the homepage based on the user's role.

2. Creating a New Record (Family Expenses Table)

Purpose:
To ensure that the **data entry form** works and saves records properly.

Screenshot 2:

- Show the form in **Family Expenses Table**
- Fields filled with data (Date, Amount, Description, Category, etc.)
- Click **Submit**

Expected Result:

- The new expense record should appear in the list view.

3. Workflow Trigger Verification

Purpose:
To confirm that a workflow is triggered automatically after record submission.

Screenshot 3:

- Show **Workflow Editor** or **Flow Designer**
- Workflow triggered for the new expense record

- Indicate the transition from “Submitted” → “Pending Approval” **Expected Result:**
- The request moves to the next approver automatically.

4. Manager Approval Screen

Purpose:
To validate that the assigned manager receives the request and can approve or reject it.

Screenshot 4:

- Show the manager’s interface
- Approval form with options “Approve” and “Reject” **Expected Result:**
- On approval, the record status changes to **Approved**.
- On rejection, status changes to **Rejected**.

5. Access Control (ACL) Test

Purpose:
To confirm that unauthorized users cannot access restricted records.

Screenshot 5:

- Log in as a **non-admin user**
- Attempt to open a record belonging to another group

Expected Result:

- Access Denied message or restricted visibility.

6. Notification Testing

Purpose:
To verify that email notifications are sent to users and approvers automatically.

Screenshot 6:

- Show an email notification screenshot (e.g., “Your request has been approved”)
- Include timestamp and workflow name.

Expected Result:

- Email or ServiceNow notification received successfully.

7. Audit Log Verification

Purpose:
To confirm that every activity (create, update, delete, approve) is logged.

Screenshot 7:

- Show **System Logs → All**
- Highlight records of the actions taken (Created, Updated by, Approved by)

Expected Result:

- All user activities appear correctly with timestamp and username.

8. Final Dashboard / List View

Purpose:
To validate that all approved records appear correctly in the system view.

Screenshot 8:

- Show the **Family Expenses** list view
- Columns like Number, Date, Amount, Status
- Status showing “Approved/Rejected”

PROJECT DESIGN PHASE

Project solution fit

In many organizations, managing **users, groups, and roles** within enterprise systems can be highly complex and error-prone. Common challenges include:

- Manual creation and maintenance of user accounts,
- Lack of standardized workflows for approvals and access requests,
- Overlapping or missing access permissions, leading to security risks,
- Difficulty in tracking who has access to what, and
- Delays in granting or revoking access, affecting productivity.

These issues result in **inefficient access control**, increased risk of **unauthorized access**, and a lack of **visibility and accountability** in user management processes.

Proposed Solution

The proposed solution — **Optimizing User, Group, and Role Management with Access Control and Workflows** — addresses these challenges through automation and rolebased access governance within a centralized platform (ServiceNow).

The key solution elements include:

- **Centralized User, Group, and Role Management:**
  All access-related configurations are managed in one place, improving visibility and reducing administrative effort.
- **Automated Workflows:**
  Approval and role assignment workflows are automated to ensure that every access request follows a defined and auditable process.
- **Role-Based Access Control (RBAC):**
  Each user is assigned specific roles (`expense_user`, `expense_manager`, `expense_admin`) that determine what actions they can perform — ensuring security and compliance.
- **Dynamic Table Design (Family Expenses):**
  A structured table stores all access and expense data, providing transparency and data integrity.
- **Access Control Rules (ACLs):**
  Restrict users from viewing or modifying records beyond their authorization level.
- **User-Friendly Interface:**
  Simplified forms and dashboards make it easy for users to submit requests and managers to approve them efficiently.

How the Solution Fits the Problem

Problem	Solution Fit
Manual and time-consuming user management reduces administrative effort.	Automated user and group creation via workflows
Lack of approval structure for access requests	Workflow-driven approval ensures each access request is reviewed and authorized.
Security risks due to undefined permissions on user role.	Role-based access control and ACLs limit actions based on user role.
Difficulty tracking who has what access	Centralized records and audit trails increase visibility and accountability.
Inconsistent access revocation	Automated workflows ensure timely updates and removal of roles when not required.

Conclusion

The project **"Optimizing User, Group, and Role Management with Access Control and Workflows"** successfully demonstrates how structured automation and access governance can enhance efficiency, security, and accountability within an organization.

Through the use of **ServiceNow**, the project integrates **user, group, and role management** with **workflow automation** to ensure that access requests are processed systematically and securely. The implementation of **Role-Based Access Control (RBAC)** and **Access Control Lists (ACLs)** ensures that users interact only with the data relevant to their responsibilities, preventing unauthorized access.

Solution Architecture

The **Solution Architecture** defines the overall structure, flow, and integration of components within the **User, Group, and Role Management System**.
It ensures that every process — from user authentication to workflow approval — is streamlined, secure, and scalable.

---

1. Architectural Overview

The system is built on the **ServiceNow Platform**, leveraging its **modular and workflowdriven architecture**.
It consists of **three key layers**:

Layer	Description
Presentation Layer (UI Layer)	Provides user interfaces such as forms, lists, and dashboards for interaction. Users can submit requests, managers can approve, and admins can monitor.
Application Logic Layer	Contains workflows, business rules, and scripts that define system behavior and automate processes like approvals and status updates.
	Stores all configuration and transaction data, including user details,

Data Layer (Database Layer)	group associations, roles, and expense records, ensuring secure and organized data management.

2. Major System Components

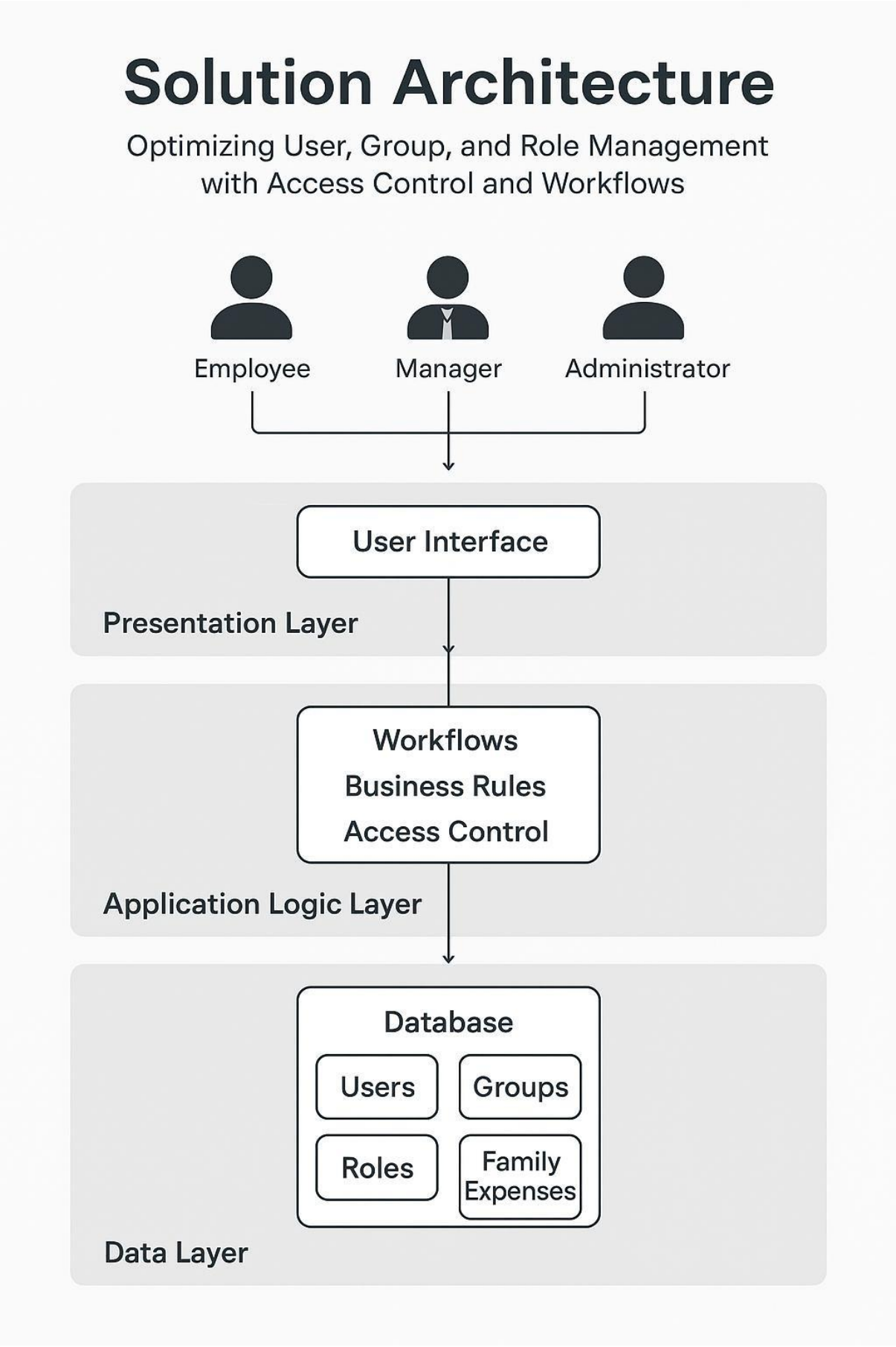
Component	Function
User Management Module	Handles creation, modification, and deactivation of user accounts.
Group Management Module	Organizes users into logical groups for easier approval routing and access tracking.
Role Management Module	Defines user privileges and ensures segregation of duties via Role-Based Access Control (RBAC).
Access Control (ACL)	Enforces table and record-level security, ensuring only authorized roles can read, write, or delete data.
Workflow Engine	Automates request submission, approval, and notification processes.
Notification System	Sends alerts to users and managers during key workflow events.
Reporting & Audit Logs	Tracks actions for compliance, accountability, and future auditing.

3. Logical Flow of the System

1. **User Request Initiation:**
   A user logs in and submits a new expense or access request through a form.
2. **Workflow Trigger:**
   The submission automatically triggers a workflow associated with the *Family Expenses* table.
3. **Approval Process:**
   The workflow routes the request to the appropriate **manager** (based on group or role).
   - If **approved**, the record status updates to "Approved." ○ If **rejected**, it updates to "Rejected."
4. **Access Enforcement:**
   ACLs and roles ensure that only authorized users (e.g., admin, manager) can view or modify certain records.
5. **Notification & Tracking:**
   Email notifications are sent to users about approval status. All transactions are logged for transparency.
6. **Data Storage:**
   Finalized data is securely stored in the **Family Expenses** table within the ServiceNow database.

Solution Architecture Description

The **Solution Architecture** is designed to provide a secure, automated, and role-based framework for managing users, groups, and access permissions.
It integrates workflow automation with access control policies to ensure that every operation — from request submission to approval — follows a structured and auditable process.



REQUIREMENT ANALYSIS

1. Project Timeline

Phase	Activity	Description	Duration
Phase 1	Requirement Gathering	Collect requirements related to user roles, access control, and workflow automation.	Week 1
Phase 2	System Design	Create data model, architecture diagram, and workflow design.	Week 2
Phase 3	Environment Setup	Create update sets, configure ServiceNow modules, and set up tables.	Week 3
Phase 4	Implementation	Develop tables, forms, and workflows for user, group, and role management.	Week 4–5
Phase 5	Testing & Validation	Perform unit testing and system integration testing.	Week 6
Phase 6	Documentation & Deployment	Prepare final documentation, conduct demonstrations, and deploy solution.	Week 7

---

2. Resource Planning

Resource Type	Details
Hardware	Computer with minimum 8GB RAM, stable internet connection
Software	ServiceNow Developer Instance, Browser (Chrome/Edge), Flow Designer
Human Resources	Project Lead, Developer, Tester, Documentation Analyst
Other Tools	Workflow Editor, Access Control Editor, Data Management Tools

---

3. Risk Management

Potential Risk	Impact	Mitigation Strategy
Incomplete requirements	Medium	Conduct detailed requirement sessions with

		stakeholders
Workflow errors	High	Validate workflows with test data before deployment
Access permission	High	Implement strict ACL and testing for role mapping conflicts
Time overrun	Medium	Track progress weekly and adjust workload as needed

4. Deliverables

Deliverable	Description
System Design Document	Includes architecture diagram, data model, and workflows
Functional Modules	User, Group, and Role management modules
Access Control Rules	Defined ACLs for secure operations
Workflow Automation	Approval and provisioning processes
Final Report & Presentation	Project summary, results, and demonstration

5. Expected Outcomes

- A fully functional **role-based access control system** in ServiceNow.
- Automated **workflow-driven user and group management**.
- Improved **security, compliance, and operational efficiency**.
- Reduced manual intervention through **automation and centralized control**.

User Stories

User stories describe the system requirements from the perspective of end-users — focusing on *who* needs a feature, *what* they need, and *why* they need it.
They help ensure the solution meets real user needs and aligns with business goals.

1. User Roles Identified

Role	Description
End User (Employee)	Submits access requests or data (e.g., family expense form).
Manager / Approver	Reviews and approves/rejects access or expense requests.
System Administrator	Manages users, groups, roles, and system configurations.
Auditor / Compliance Officer	Reviews system logs, audit trails, and user permissions for compliance.

2. User Stories Table

ID	As a (Role)	I want to...	So that I can...	Acceptance Criteria
US01	End User	Submit a new system access or expense record submission request	Get necessary access or expense record submission done easily	The form allows me to submit requests and notifies my manager for approval.

ID	As a (Role)	I want to...	So that I can...	Acceptance Criteria
US02	End User	View the status of my requests	Track whether they are pending, approved, or rejected	A dashboard shows all my request statuses in real time.
US03	Manager	Receive notifications for new requests	Quickly review and approve or reject them	Notification triggers whenever a new request is assigned to me.
US04	Manager	Approve or reject requests directly	Simplify approval workflows from the interface	Approval or rejection updates the record status automatically.
US05	Administrator	Create and assign roles to users	Control access based on responsibilities	Admin can create, edit, or delete roles within the system.
US06	Administrator	Define and manage access control rules (ACLs)	Secure system data and restrict unauthorized access	Only authorized users can view or modify sensitive records.
US07	Administrator	Automate workflows for access approval	Reduce manual work and delays	A workflow routes requests to the right approver automatically.
US08	Auditor	View audit logs of access and approval	Ensure compliance and accountability activities	The system logs all user actions and workflow decisions.
US09	Administrator	Integrate user management with consistency across HR or ITSM systems	Maintain data modules	Any new user or role updates sync automatically with other systems.
		Receive an email		

US10	End User	The system sends an notification once a request is approved	Stay informed about my request outcome automatic email upon or approval/rejection.

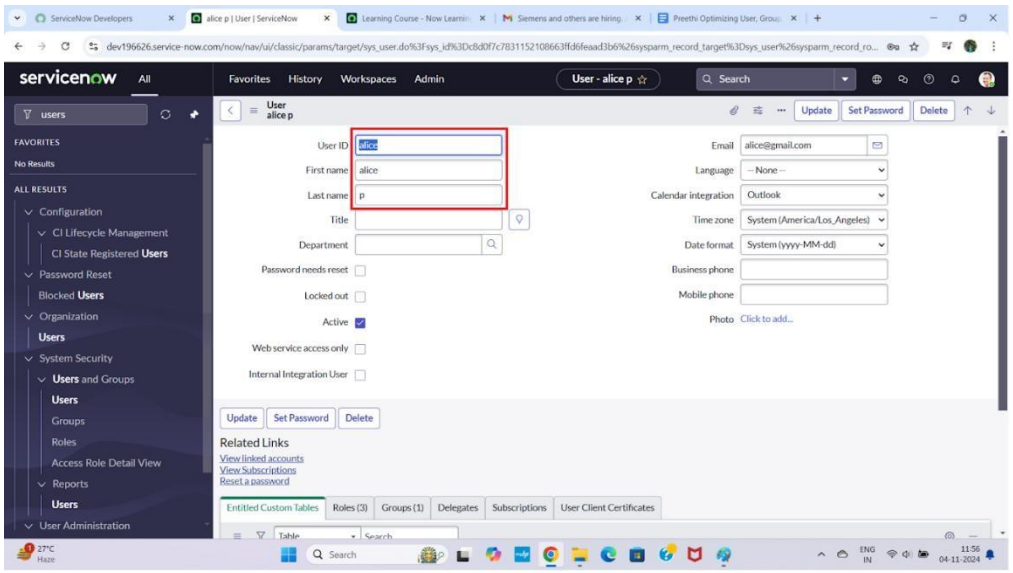
3. Additional Notes

- Each user story will have a corresponding **workflow and test case** to verify completion.
- **Priority levels** can be assigned (High, Medium, Low) based on organizational needs.
- Stories will be refined during **sprint planning** to ensure smooth implementation.

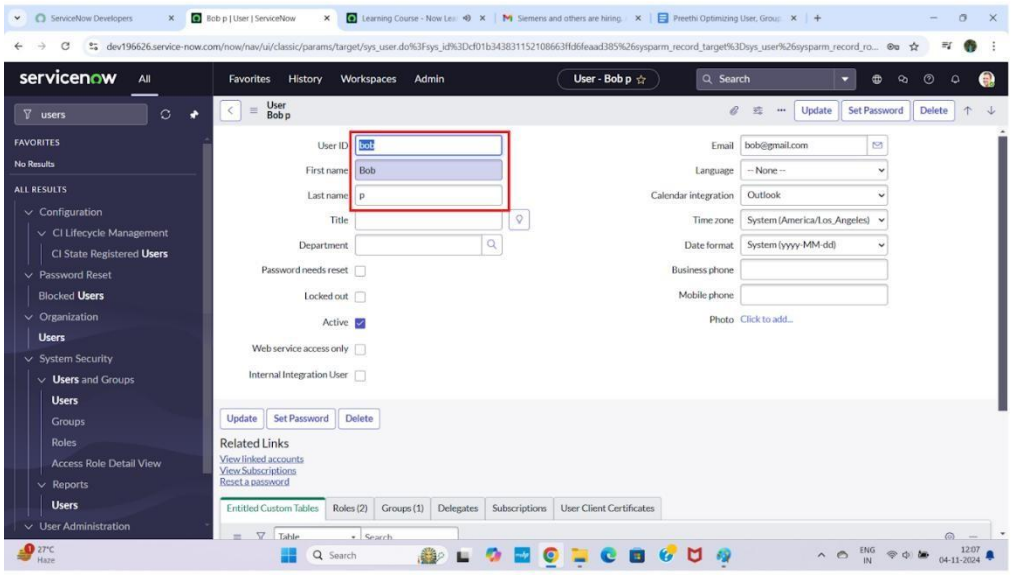
PERFORMANCE TESTING

Create Users

1. Open service now
2. Click on All >> search for users
3. Select Users under system security
4. Click on new
5. Fill the following details to create a new user 6. Click on submit



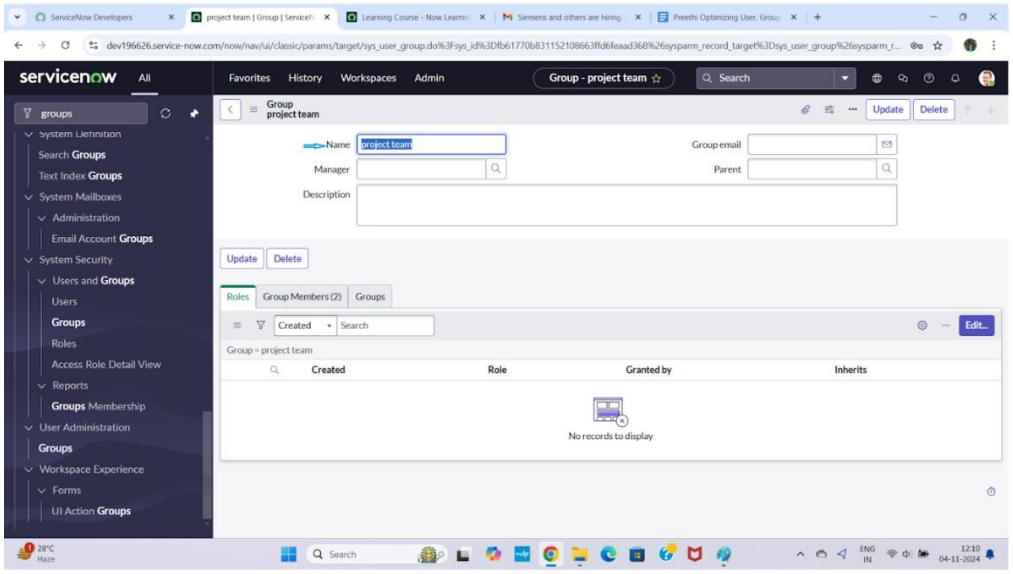
7.
8. **Create one more user:**
9. Create another user with the following details
10. Click on submit



11.

Create Groups 1. Open service now.

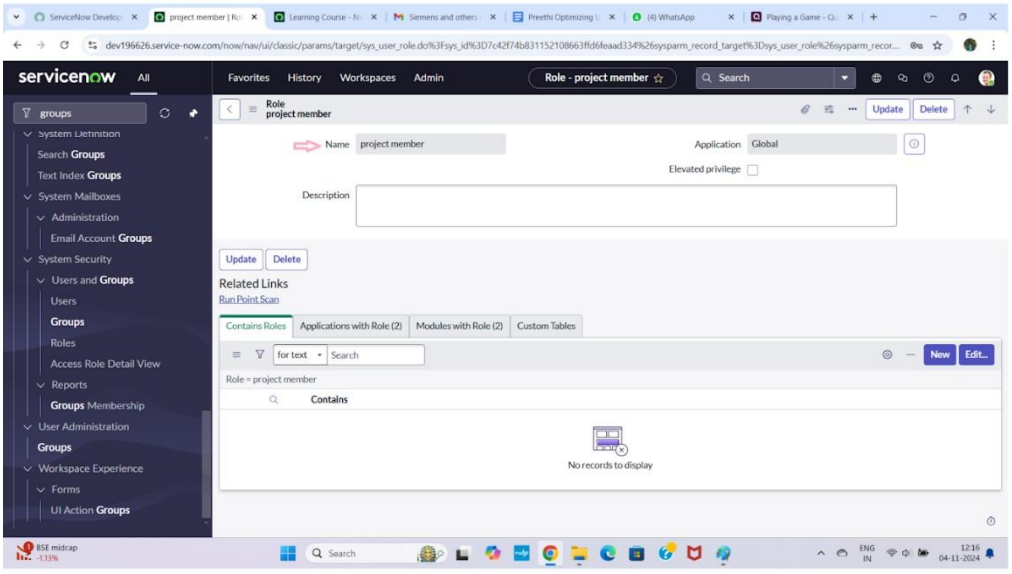
2. Click on All >> search for groups
3. Select groups under system security
4. Click on new
5. Fill the following details to create a new group
6. Click on submit



Create Roles

1. Open service now.
2. Click on All >> search for roles

3. Select roles under system security
4. Click on new
5. Fill the following details to create a new role
6. Click on submit



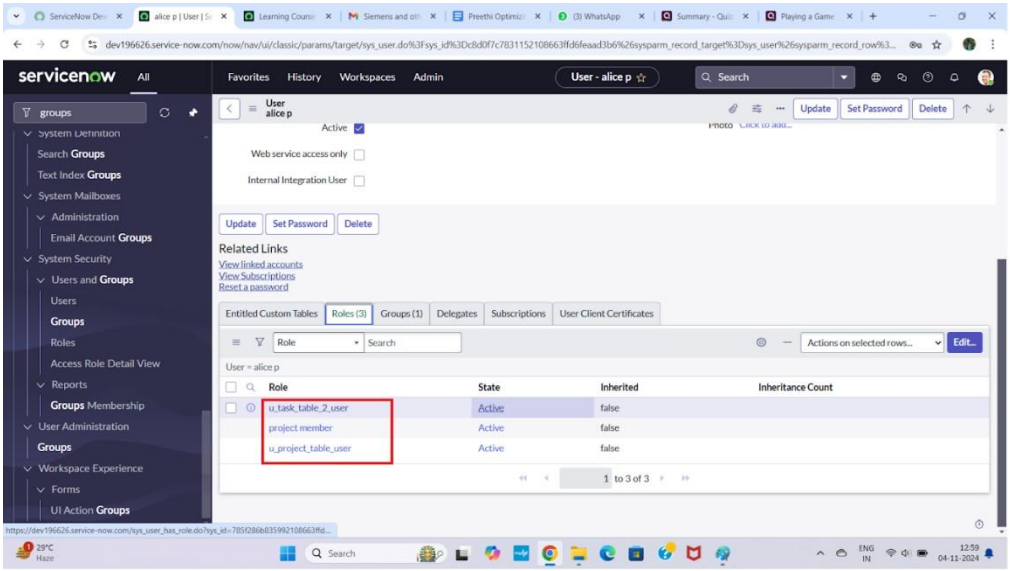
Create one more role:

7.Create another role with the following details

8.Click on submit

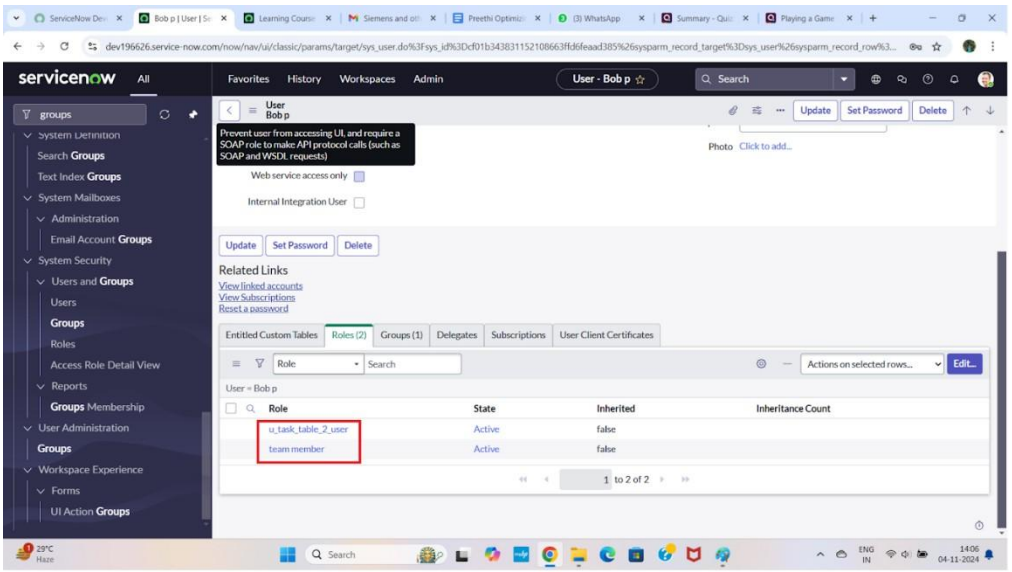
Assign roles to alice user

1. Open servicenow.Click on All >> search for user
2. Select tables under system definition
3. Select the project manager user
4. Under project manager
5. Click on edit
6. Select project member and save
7. click on edit add u_project_table role and u_task_table role
8. click on save and update the form.



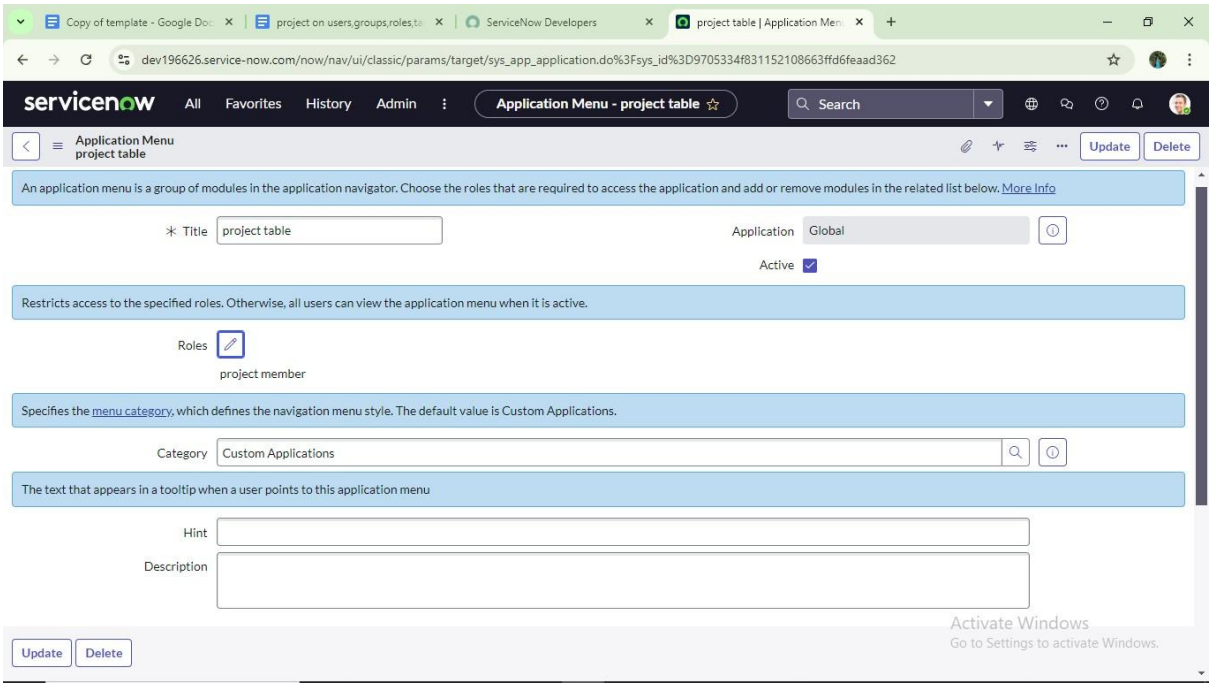
Assign roles to bob user

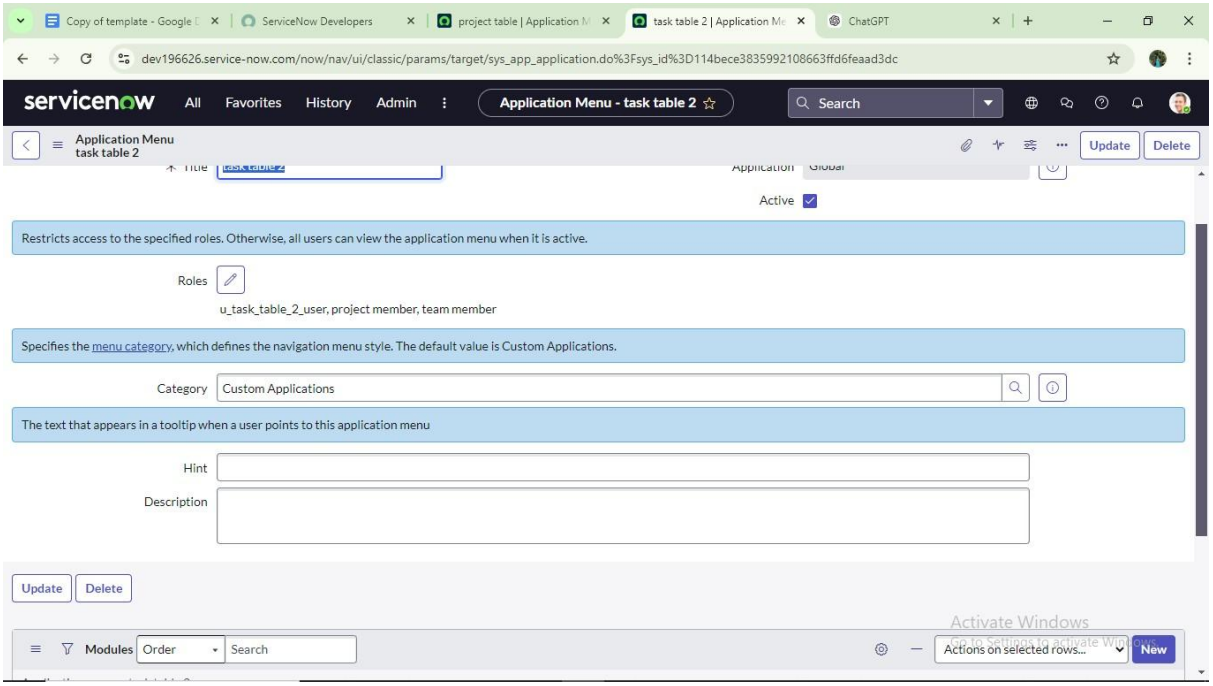
1. Open servicenow.Click on All >> search for user
2. Select tables under system definition
3. Select the bob p user
4. Under team member
5. Click on edit
6. Select team member and give table role and save
7. Click on profile icon Impersonate user to bob
8. We can see the task table2.



Assign table access to application

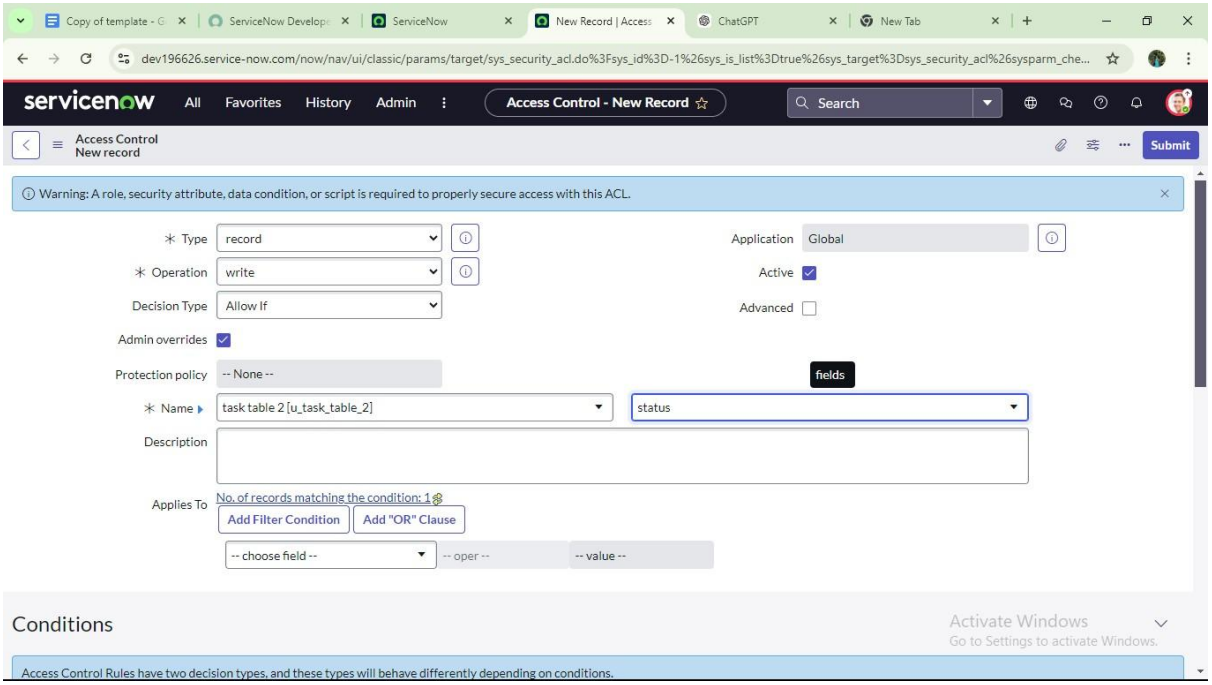
1. while creating a table it automatically create a application and module for that table
2. Go to application navigator search for search project table application
3. Click on edit module
4. Give project member roles to that application
5. Search for task table2 and click on edit application.
6. Give the project member and team member role for task table 2 application





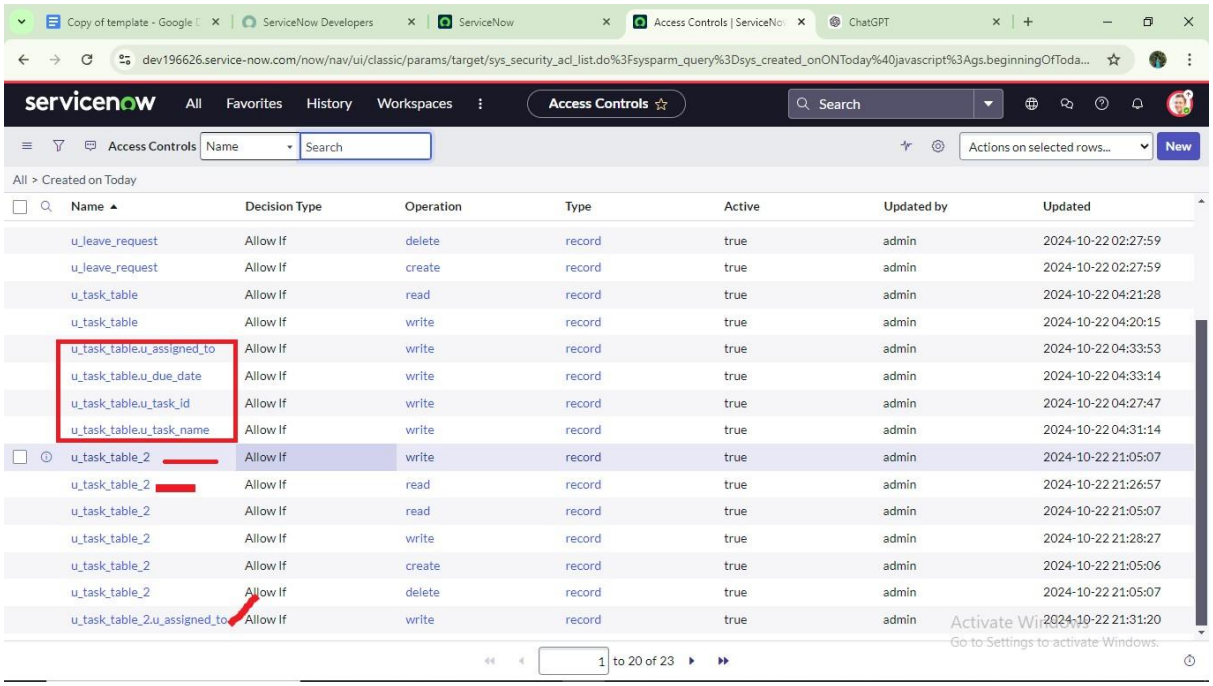
Create ACL

1. Open service now.
2. Click on All >> search for ACL
3. Select Access Control(ACL) under system security
4. Click on elevate role
5. Click on new
6. Fill the following details to create a new ACL

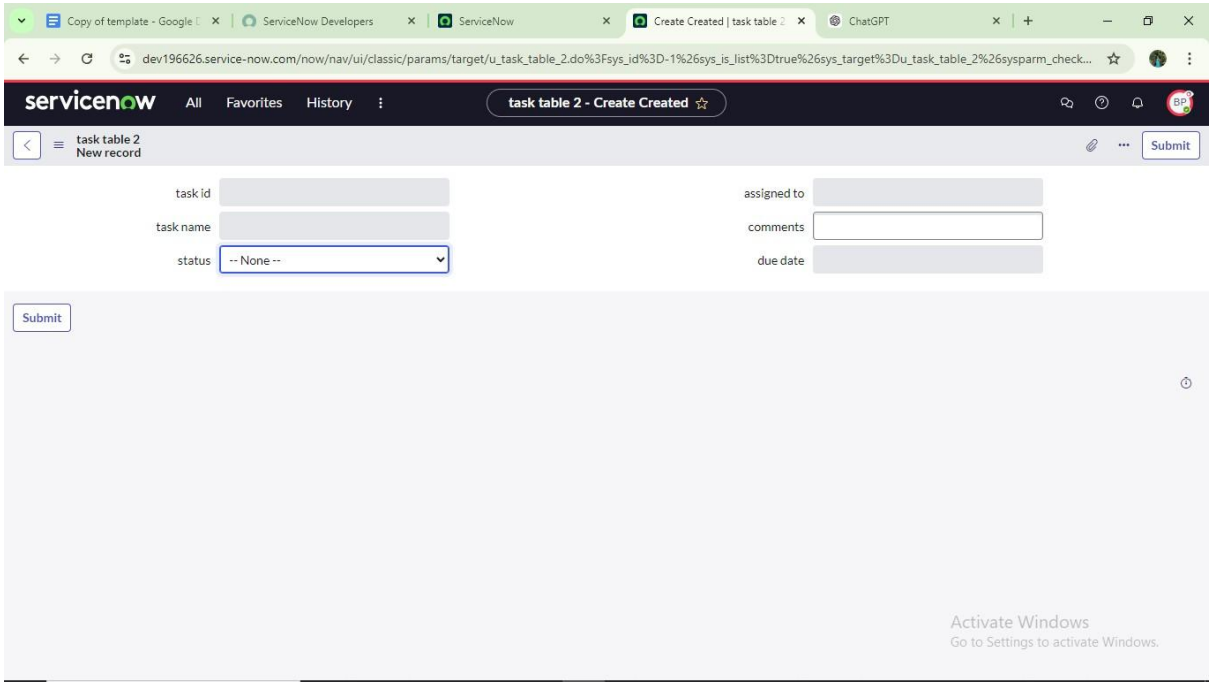


7. Scroll down under requires role
8. Double click on insert a new row

9. Give task table and team member role
10. Click on submit
11. Similarly create 4 acl for the following fields



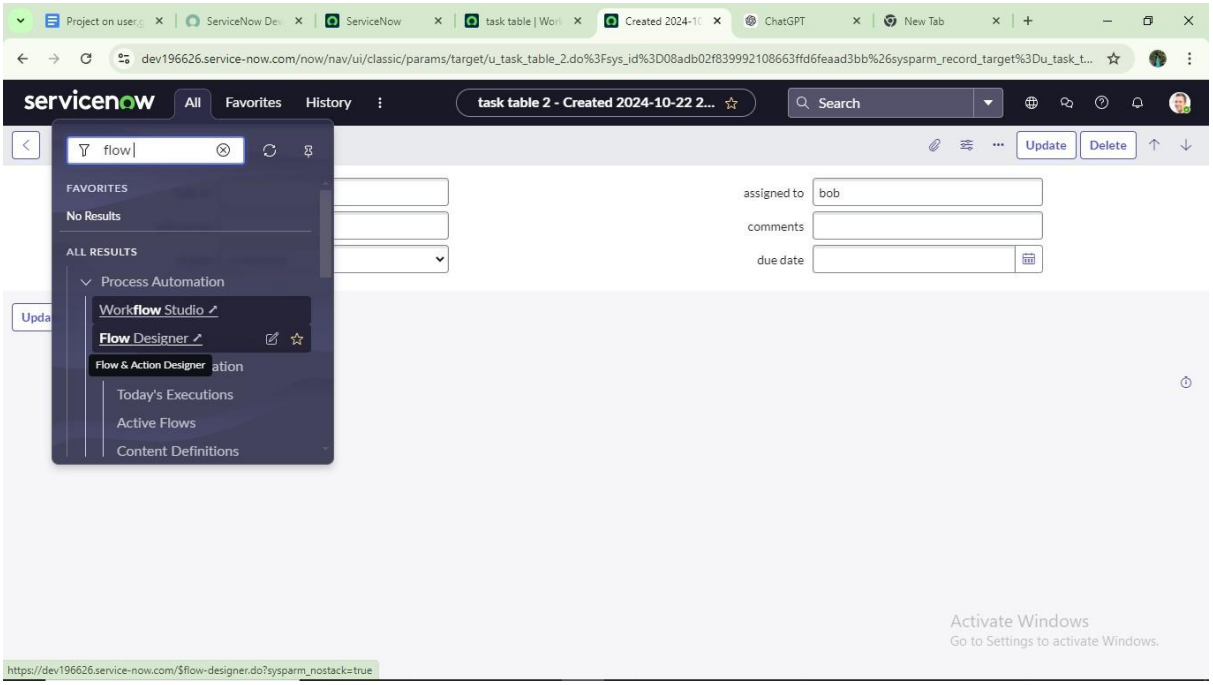
12. Click on profile on top right side
13. Click on impersonate user
14. Select bob user
15. Go to all and select task table2 in the application menu bar
16. Comment and status fields are have the edit access

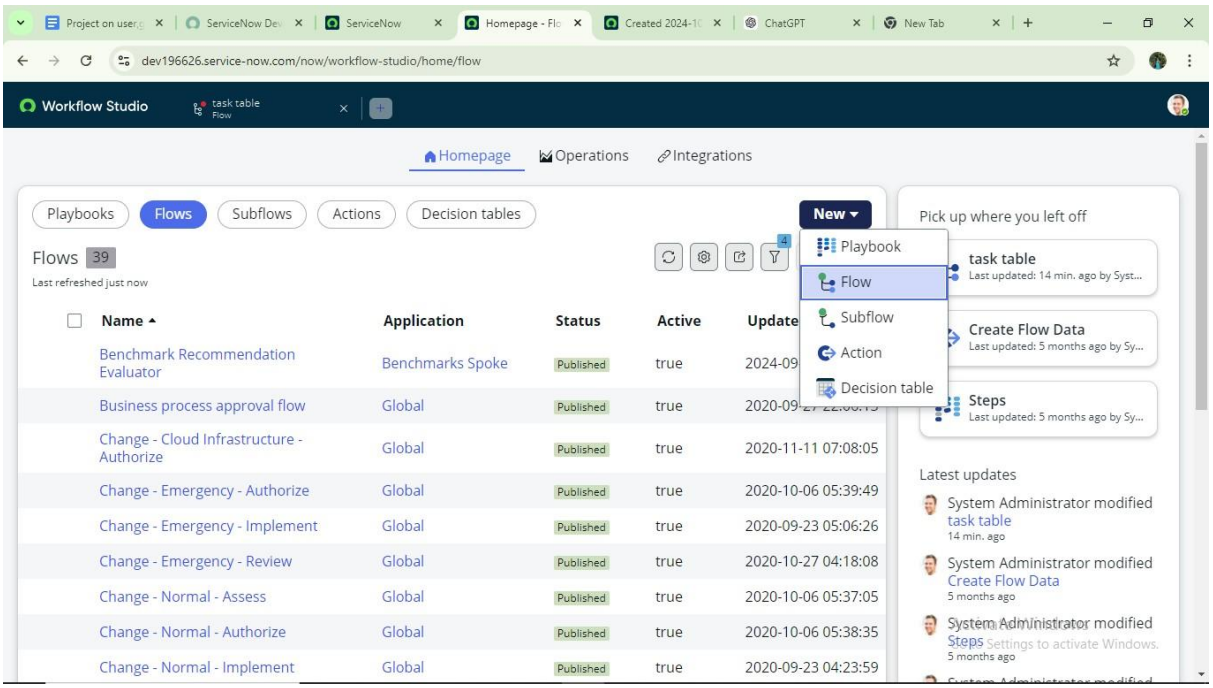


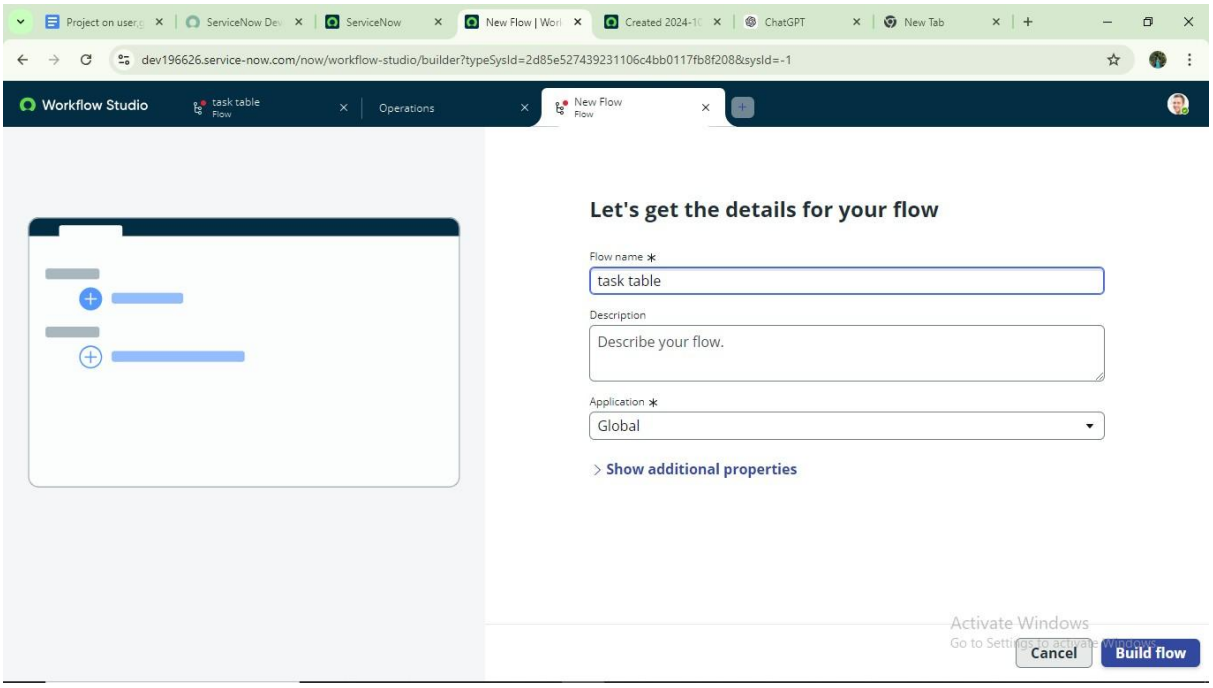
Create a Flow to Assign operations ticket to group

1. Open service now.

2. Click on All >> search for Flow Designer
3. Click on Flow Designer under Process Automation.
4. After opening Flow Designer Click on new and select Flow.
5. Under Flow properties Give Flow Name as " task table".
6. Application should be Global.
7. Click build flow.



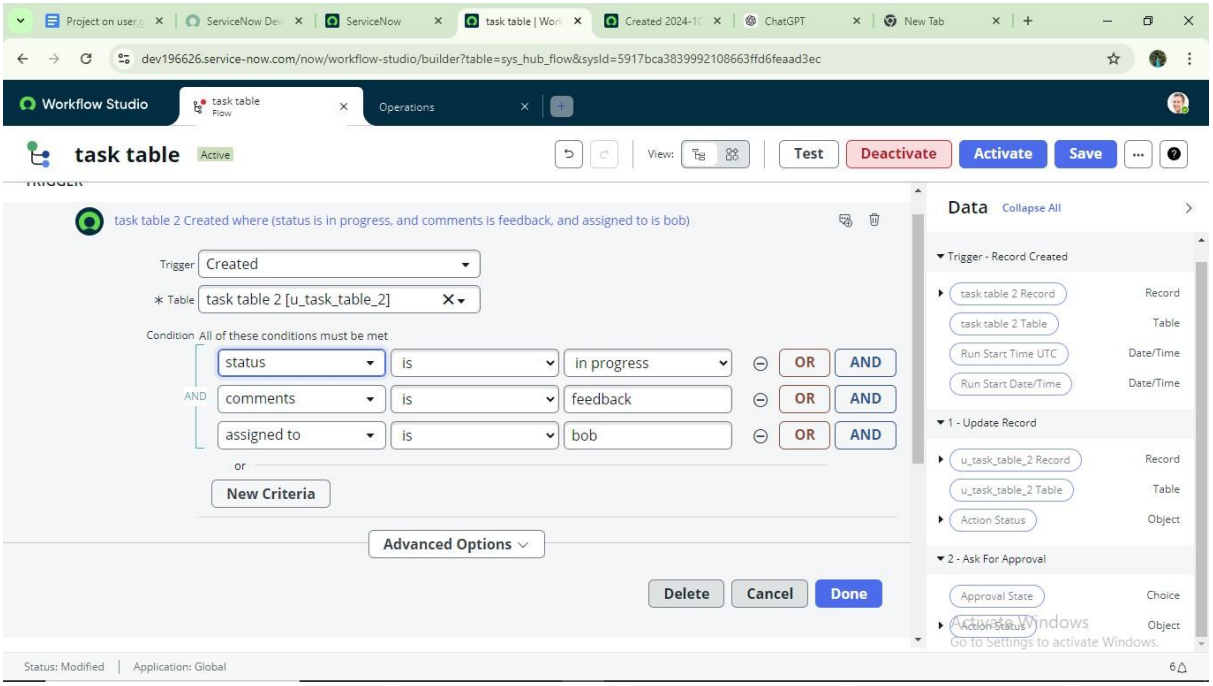




next step:

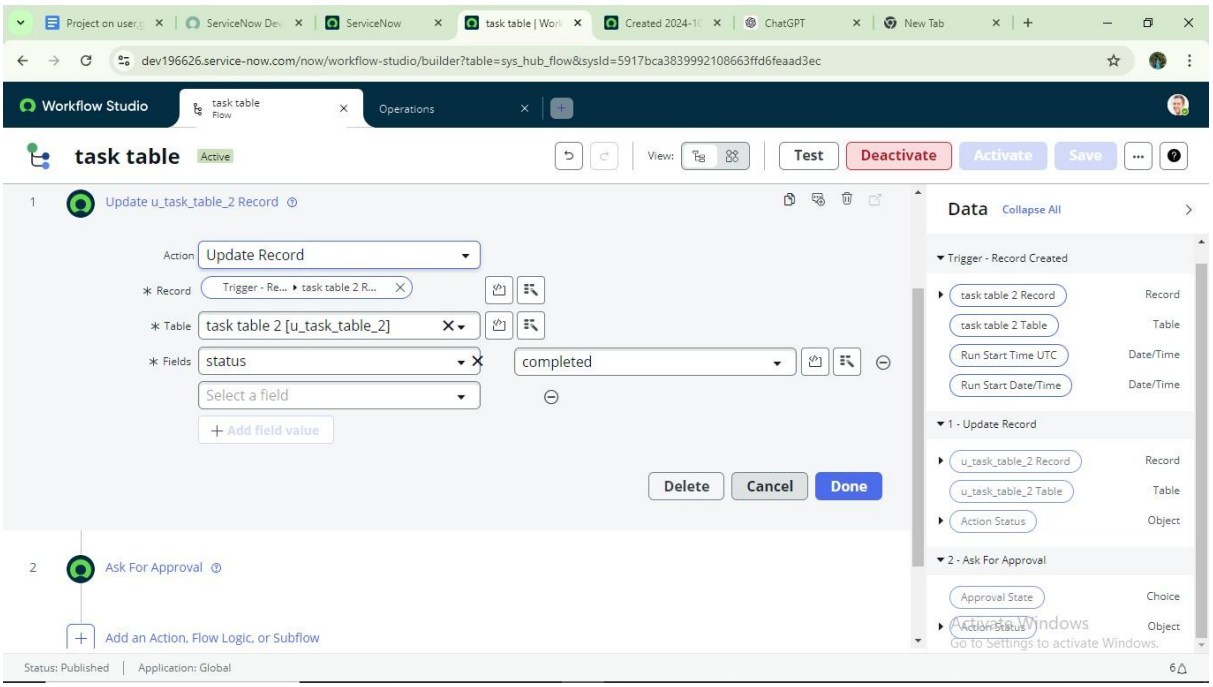
1. Click on Add a trigger
2. Select the trigger in that Search for “create record” and select that.
3. Give the table name as “ task table ”.
4. Give the Condition as Field : status Operator :is Value : in progress

   Field : comments Operator :is Value : feedback
   Field : assigned to Operator :is Value : bob
5. After that click on Done.



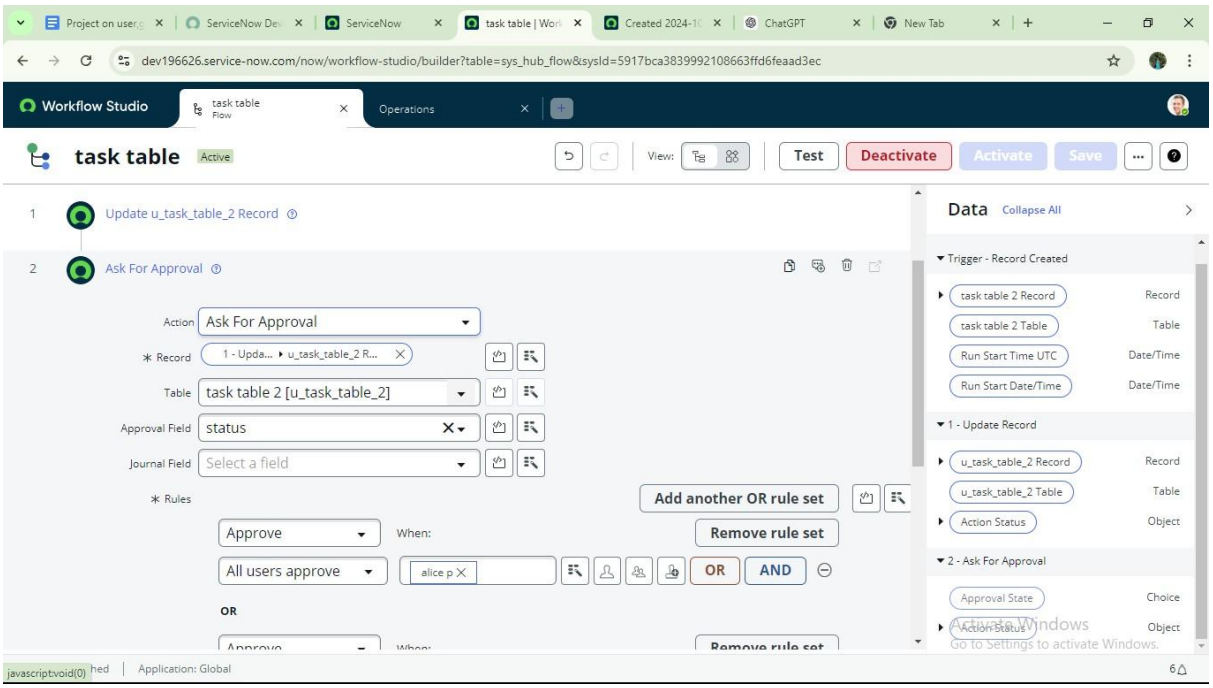
Next step:

1. Click on Add an action.
2. Select action in that ,search for " update records".
3. In Record field drag the fields from the data navigation from Right Side(Data pill)
4. Table will be auto assigned after that
5. Add fields as "status" and value as "completed"
6. Click on Done.

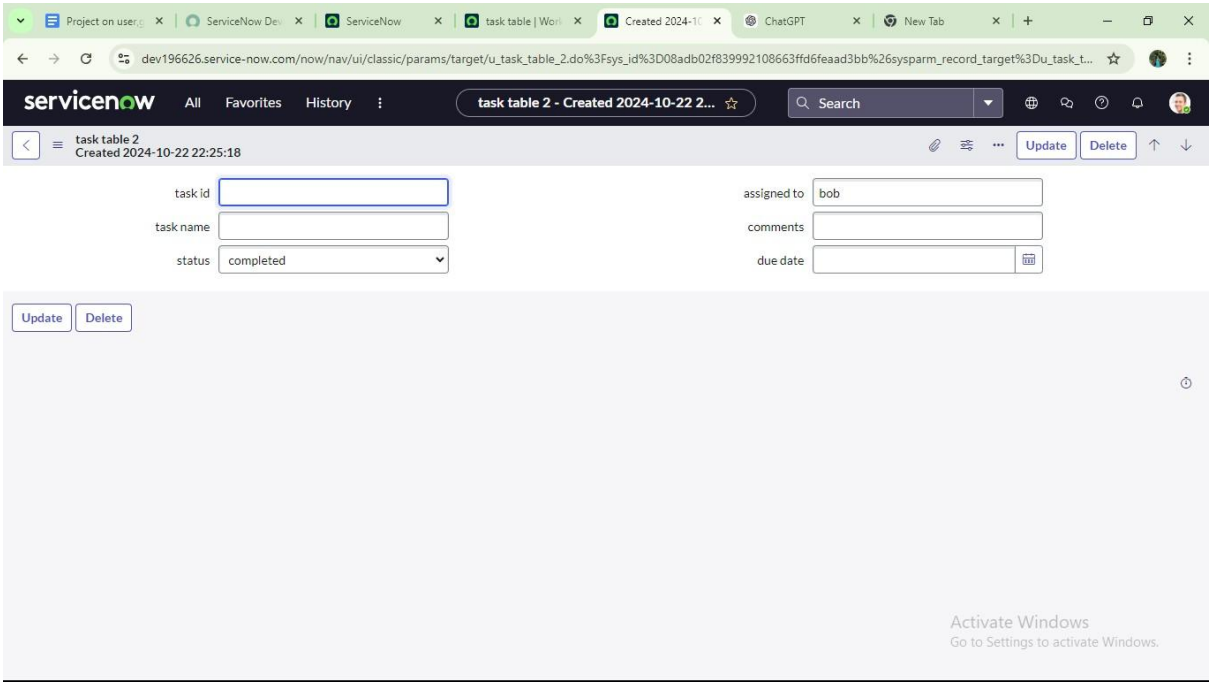


Next step:

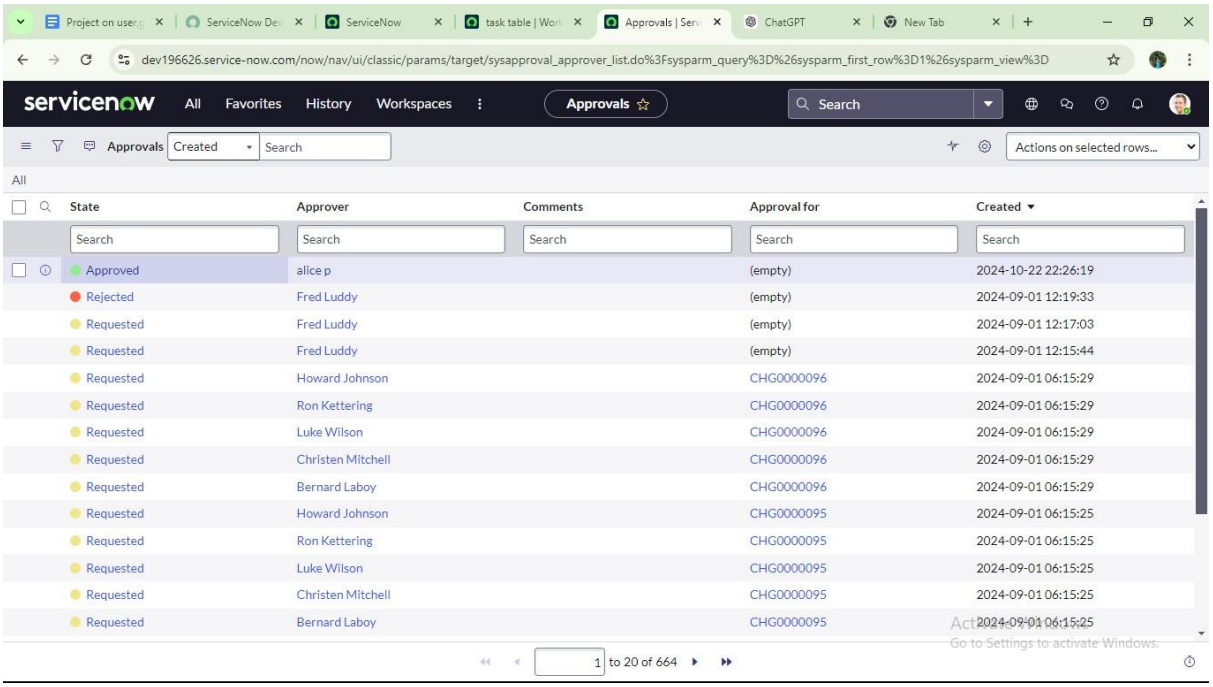
1. Now under Actions.
2. Click on Add an action.
3. Select action in that ,search for “ ask for approval ”.
4. In Record field drag the fields from the data navigation from Right side
5. Table will be auto assigned after that
6. Give the approve field as “ status”
7. Give approver as alice p
8. Click on Done.



1. Go to application navigator search for task table.
2. It status field is updated to completed



1. Go to application navigator and search for my approval
2. Click on my approval under the service desk.
3. Alice p got approval request then right click on requested then select approved



Conclusion :

This scenario highlights a structured approach to project management, showcasing the roles of Alice and Bob within a defined workflow. With Alice's oversight and Bob's execution, the team effectively collaborates to ensure project success. The use of tables organizes key information, facilitating easy tracking of projects, tasks, and progress updates. Overall, this system promotes accountability, enhances communication, and leads to the successful completion of projects.

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.

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
4. Arrange the widgets as needed for clarity.

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

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

2. Navigate to:

   System Security → Access Control (ACL)

3. Click New.

Fill in:

Field	**Value**

Type	record
Operation	read
Table	Your HR Case table
Active	True

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.
4. 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.

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);

}

}

- Click **Submit** or **Update** to save.

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
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

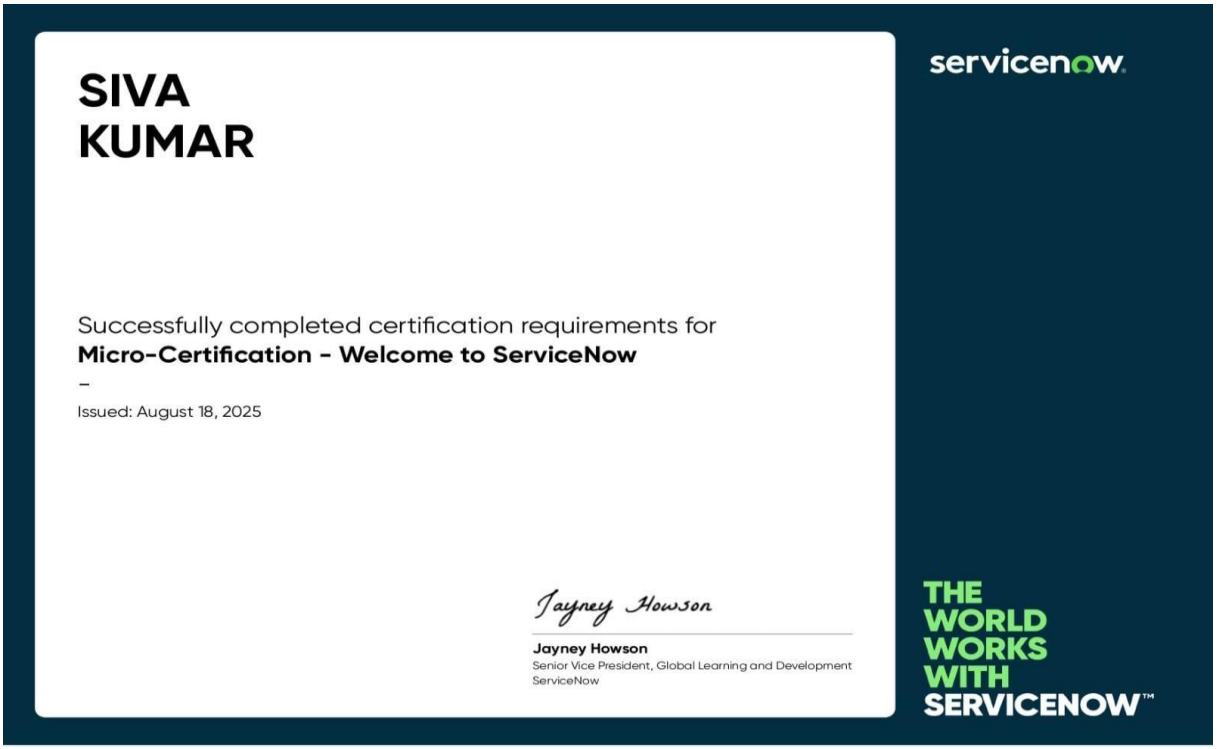
Description field.

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(curr


MARIMUTHU
.P
servicenow
Successfully completed certification requirements for
Micro-Certification - Welcome to ServiceNow
–
Issued: September 3, 2025
Jayney Howson
Jayney Howson
Senior Vice President, Global Learning and Development
ServiceNow
THE WORLD WORKS WITH SERVICENOW™


In recognition of the commitment to achieve professional excellence
Generative AI in Action
IBM SkillsBuild
IBM
Mari Muthu
Has successfully satisfied the requirements for:
Generative AI in Action
Issued on: Aug 06, 2025
Issued by: IBM SkillsBuild
Verify: https://www.credly.com/badges/4720a87d-0c21-4445-bd98-d809788fa096
IBM


SIVA
KUMAR
servicenow
Successfully completed certification requirements for
Micro-Certification - Welcome to ServiceNow
–
Issued: August 18, 2025
Jayney Howson
Jayney Howson
Senior Vice President, Global Learning and Development
ServiceNow
THE
WORLD
WORKS
WITH
SERVICENOW™


In recognition of the commitment to achieve professional excellence
Generative AI in Action
IBM
Siva Kumar.s
Has successfully satisfied the requirements for:
Generative AI in Action
Issued on: Aug 06, 2025
Issued by: IBM SkillsBuild
Verify: https://www.credly.com/badges/b352ca08-0949-4d18-bf74-df1d30843125
IBM

servicenow

periyasamy

Successfully completed certification requirements for
Micro-Certification - Welcome to ServiceNow
–
Issued: August 30, 2025

Jayney Howson

Jayney Howson
Senior Vice President, Global Learning and Development
ServiceNow

THE WORLD WORKS WITH SERVICENOW™

In recognition of the commitment to achieve professional excellence

periyasamy periyasamy

Has successfully satisfied the requirements for:

Generative AI in Action

Issued on: Aug 06, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/8bcea7a7-3e1e-43af-a261-625c3d5cb1ce

IBM

○


RAJESHWARAN
P
servicenow
Successfully completed certification requirements for
Micro-Certification - Welcome to ServiceNow
–
Issued: August 20, 2025
Jayney Howson
Jayney Howson
Senior Vice President, Global Learning and Development
ServiceNow
THE
WORLD
WORKS
WITH
SERVICENOW™


In recognition of the commitment to achieve
professional excellence
Generative AI
in Action
IBM
Rajeswaran R
Has successfully satisfied the requirements for:
Generative AI in Action
Issued on: Aug 30, 2025
Issued by: IBM SkillsBuild
Verify: https://www.credly.com/badges/5af14952-1147-46ec-a9b9-6ff986fa5e9e
IBM