OPTIMIZING USER, GROUP, AND ROLE MANAGEMENT WITH ACCESS IN SERVICENOW

Submitted by

Team Leader

VAIRAPRAKASH.V (910022104039)

Team Members

BARATH.M (910022104004)

SARAVANA GOUTHAM. A (910022104056)

In partial fulfilment for the award of the degree

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

NAAN MUDHALVAN LAB

ANNA UNIVERSITY REGIONAL CAMPUS MADURAI-625-019

ANNA UNIVERSITY: CHENNAI 600 025

NOVEMBER 2025

Supervised by

Dr. Srie Vidhya Janani, M.E., Ph.D.,

BONAFIDE CERTIFICATE

This is to certify that the project report titled **"OPTIMIZING USER, GROUP AND ROLE MANAGEMENT WITH ACCESS IN SERVICENOW"** is the Bonafide work of **VAIRAPRAKASH.V (910022104039), BARATH.M (910022104004), SARAVANA GOUTHAM.A (910022104056)** who carried out the project work under my supervision in the Naan Mudhalvan Lab.

30/10/25

SIGNATURE
HEAD OF THE DEPARTMENT

30/10/25

SIGNATURE
FACULTY

Department of Computer Science and Engineering,
Anna University Regional Campus Madurai-625-019

ACKNOWLEDGEMENT

I extend my heartfelt gratitude to **Dr. Srie Vidhya Janani**, **M.E., Ph.D.,** Faculty Incharge of Naan Mudhalvan Lab, for her guidance and support throughout this project. I also thank my peers and family for their encouragement, without which this project would not have been possible.

I am deeply grateful to **Dr. V. Sasikala, M.E., Ph.D.,** Head of the Department, for her constant support and guidance.

I extend my sincere thanks to all teaching and non-teaching staff of the Department of Computer Science and Engineering and my peers for their support and encouragement.

Finally, I thank my family and friends, whose encouragement and patience motivated me to complete this project successfully.

Team Leader

VAIRAPRAKASH.V (910022104039)

Team Members

BARATH.M (910022104004)

SARAVANA GOUTHAM.A (910022104056)

ABSTRACT

In modern enterprise systems, efficient access management is crucial to maintaining security, accountability, and operational flow. This project, “Optimizing User, Group, and Role Management with Access Control”, focuses on implementing a structured access control mechanism within the ServiceNow platform. The objective is to establish a secure and automated environment for user management, group assignment, and task workflow operations.

The project involves creating users and groups, assigning appropriate roles to users such as Alice and Bob, and configuring application access by defining table-level permissions. To ensure data security, Access Control Lists (ACLs) are implemented to regulate read, write, and update permissions based on user roles. Additionally, a workflow (Flow) is designed using ServiceNow Flow Designer to automatically assign operated tickets to relevant groups, reducing manual intervention and enhancing efficiency.

Through these implementations, the project achieves a well-defined and role-based access control system that enhances transparency, simplifies administrative management, and strengthens organizational security. This structured approach to user and group management not only optimizes operations but also aligns with best practices for ServiceNow governance and automation.

Title: Optimizing User, Group, and Role Management with Access in ServiceNow

1.Project Overview

The project *"Optimizing User, Group, and Role Management with Access"* focuses on implementing and improving the access control structure within the **ServiceNow platform**. It aims to streamline how users, groups, and roles are created, managed, and assigned to ensure secure and efficient operations.

The project enhances platform governance by enabling **role-based access control (RBAC)**, **application access management**, and **automated ticket assignment workflows**. By defining clear user responsibilities and system permissions, this solution ensures compliance, reduces security risks, and improves operational productivity.

Through this project, the configuration of ServiceNow's access model—including user creation, group structuring, ACL configuration, and automation through flows—is optimized for better system management, scalability, and accountability.

2.Objectives

- To design and implement a secure and scalable **role-based access control (RBAC)** model.
- To create and configure **users**, **groups**, and **roles** for effective task management and ownership.
- To enable **application-level access** by defining and assigning permissions at the table and module level.
- To develop and automate workflows using **ServiceNow Flows** for assigning tickets dynamically to appropriate groups.

- To ensure that every action performed in the system aligns with organizational policies and access governance standards.

3.Student Outcomes

- Understand and implement **ServiceNow user and role administration** principles.
- Gain hands-on experience in configuring **Access Control Lists (ACLs)** and **Flow Designer automation**.
- Learn how to manage **application-level access** and ensure proper data security in enterprise systems.
- Develop **problem-solving and process optimization** skills relevant to ITSM and enterprise workflow design.
- Strengthen collaboration and documentation abilities following real-world ServiceNow practices.

4.System Requirements

4.1 Hardware Requirements

- Computer with minimum 8 GB RAM, Intel i5 Processor or above
- Stable internet connection
- Modern web browser (Google Chrome / Edge)

4.2 Software Requirements

- ServiceNow Developer Instance (Personal Instance)
- Flow Designer
- Studio Editor for Script Configuration (if applicable)

- Access Control List (ACL) Module
- Role and Group Management Modules
- Visual Diagram Tools (for ER & Flow diagrams)

5.Skills Required

- ServiceNow Platform Administration
- Role and Group Management
- Access Control Configuration (ACLs)
- Application Access & Table Permission Management
- Workflow and Flow Designer Automation
- Basic Scripting in ServiceNow (optional)
- Documentation and Reporting

6.Project Duration

300 Hours

7. Project Phase Overview

Phase No.	Phase Name	Description	Page Nos.
1	Requirement Analysis & Planning	Understanding the business need for structured access management and defining security requirements.	8-15
2	Design Phase	Designing user, group, and role structure; defining ACL strategy and flow logic.	15-17
3	Development Phase	Creating users and groups, assigning roles (Alice, Bob), setting application access, building flows for ticket routing.	18-20
4	Testing Phase	Testing ACL rules, verifying ticket automation flow, and validating access permissions.	20-25

8.Project Main Overview

8.1 Overview

The **Optimizing User, Group, and Role Management with Access** project simplifies and secures access control management in **ServiceNow**. It addresses key challenges such as inefficient access permissions, unorganized role assignments, and lack of automation in ticket management.

By implementing **user creation**, **group formation**, **role-based permissions**, and **ACL-driven access rules**, the project enforces secure boundaries across applications. Furthermore, **Flow Designer automation** ensures that operated

tickets are dynamically assigned to the right groups—reducing manual work and ensuring accountability.

This implementation serves as a strong foundation for organizations aiming to improve **security compliance**, **workflow efficiency**, and **data governance** within their ServiceNow ecosystem.

Phase1: Requirement Analysis & Planning

1.1 Business Problem

A small project team lacked clear role definitions and a structured workflow. Tasks were not consistently assigned or tracked, and there was ambiguity about who could edit task data.

i)Objective:

The objective of this project is to enhance accountability and streamline task management within a small project team using the ServiceNow platform. The system must define clear roles, enforce access control policies, and enable a structured workflow to eliminate confusion in task assignments and progress tracking.

ii)Approach:

- Identified the key users (Project Manager and Team Member) and their responsibilities.
- Defined access levels, group memberships, and workflow dependencies.
- Determined the need for a structured task assignment flow with automatic status updates and approvals.
- Planned to utilize ServiceNow modules such as **Users, Groups, Roles, Access Control Lists (ACLs), and Flow Designer** to build the solution.

iii)Project Scope

The scope of this project, *"Optimizing User, Group, and Role Management with Access Control and Workflows,"* focuses on designing a secure and efficient

structure within the ServiceNow platform for managing user access, group assignments, and automated workflows in a project management environment.

The system is implemented for a small project team comprising a Project Manager (Alice) and a Team Member (Bob). The primary objective is to streamline access management and task workflow to ensure transparency, accountability, and efficiency in handling project tasks.

The key scope areas include:

- **User and Group Management:** Creating users and groups in ServiceNow to define project team hierarchies.
- **Role Assignment:** Assigning appropriate roles to users (e.g., Project Manager and Team Member) to control permissions and access levels.
- **Application Access Configuration:** Granting specific table and module access to roles, ensuring secure and role-based data visibility.
- **Access Control List (ACL) Implementation:** Establishing access control rules to protect sensitive information and restrict data modification based on roles.
- **Workflow Automation:** Designing a Flow in ServiceNow to automatically assign operation tickets to groups and update task status based on predefined conditions.

By defining these components, the project delivers a robust access management and task automation framework that promotes operational efficiency, data security, and controlled collaboration within ServiceNow.

1.2 Functional Requirements

- FR1: Create users for team members and define roles.
- FR2: Group users logically (project/team groups).

- FR3: Assign role-based application/table access so only authorized users view or edit records.
- FR4: Implement ACLs to restrict field-level edit permissions on task records.
- FR5: Automate assignment/approval process using Flow Designer based on task attributes.

1.3 Non-functional Requirements

- Usability: Simple UI flows for request, assignment and approval.
- Security: Role-based access and ACL enforcement.
- Reproducibility: Clear setup steps to replicate the configuration in another instance.

Phase 2: System Design

2.1 High-level Components

- **Users**: Alice (Project Manager), Bob (Team Member) — stored in sys_user.
- **Groups**: Project/Team group(s) — stored in sys_user_group.
- **Roles**: u_project_table, u_task_table, plus role labels like *Project Member* and *Team Member* — stored in sys_user_role.
- **Applications/Modules**: Project Table Application, Task Table2 Application (automatically created when custom tables were created).
- **Access Control**: ACLs applied to Task Table2 and its fields (Comment, Status).
- **Automation**: Flow Designer flow named **Task Table** that triggers on task creation and routes approval to Alice.

2.2 Object Schema (tables & key fields)

Below are the primary objects you worked with (ServiceNow standard + custom placeholders):

- **sys_user (Users)**
    - user_name — login id
    - name — full name (Alice / Bob)
    - email — user email
- **sys_user_group (Groups)**
    - name — group name (Project Management Team)
    - members — link to sys_user
- **sys_user_role (Roles)**
    - name — role name (u_project_table, u_task_table, Project Member, Team Member)
- **project_table (Custom Project table)** — auto-created with table
    - u_project_name, u_owner, u_start_date, u_status
- **task_table2 (Custom Task table)**
    - number — task id
    - short_description — summary
    - assigned_to — reference to sys_user (Bob)
    - status — work status (e.g., New, In Progress, Completed)

- comments — work comments

Note: Replace project_table and task_table2 with the exact internal table names used in your instance if different.

2.3 Concepts Implemented

- **User Record (sys_user):** A standard ServiceNow record representing a person. You created Alice and Bob here.
- **Group (sys_user_group):** A container for users; simplifies assigning module or role access to many people at once. You created a project group and added members.
- **Role (sys_user_role):** A permission set that grants access to modules, tables, and actions. You created u_project_table and u_task_table and assigned them to users.
- **Application Module:** When custom tables are created, ServiceNow auto-generates an application/module. You edited module access to limit visibility based on role.
- **Access Control List (ACL):** Security records that enforce who can read, write, or create records and even field-level access. You created ACLs for Task Table2 and for specific fields comments and status.
- **Flow Designer:** A low-code automation tool. You built a flow that triggers on task record creation under specific conditions, updates status, and asks for approval from Alice.

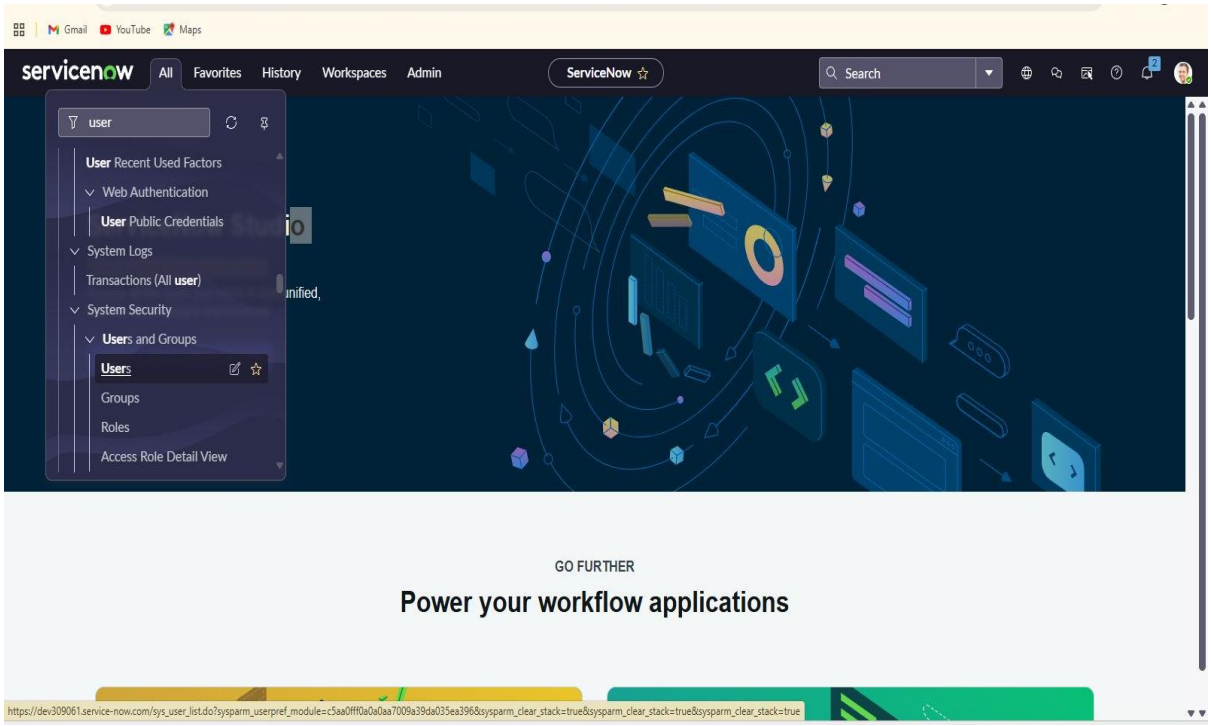
Phase 3: Development — Milestones & Activities

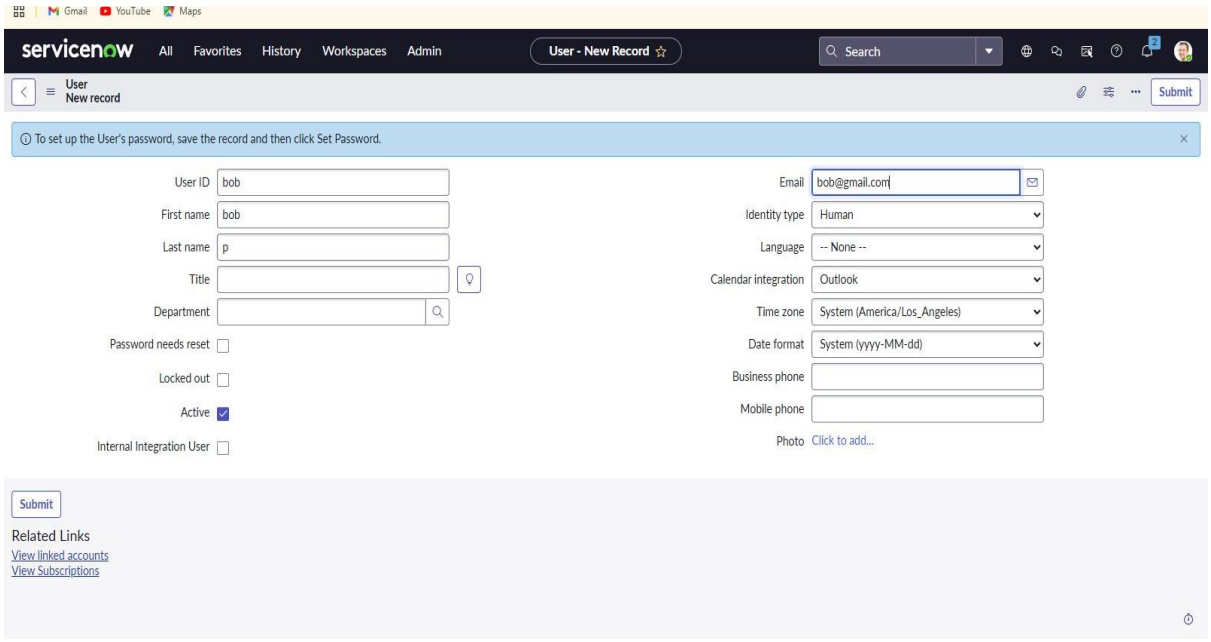
Milestone 1 — Users & Groups

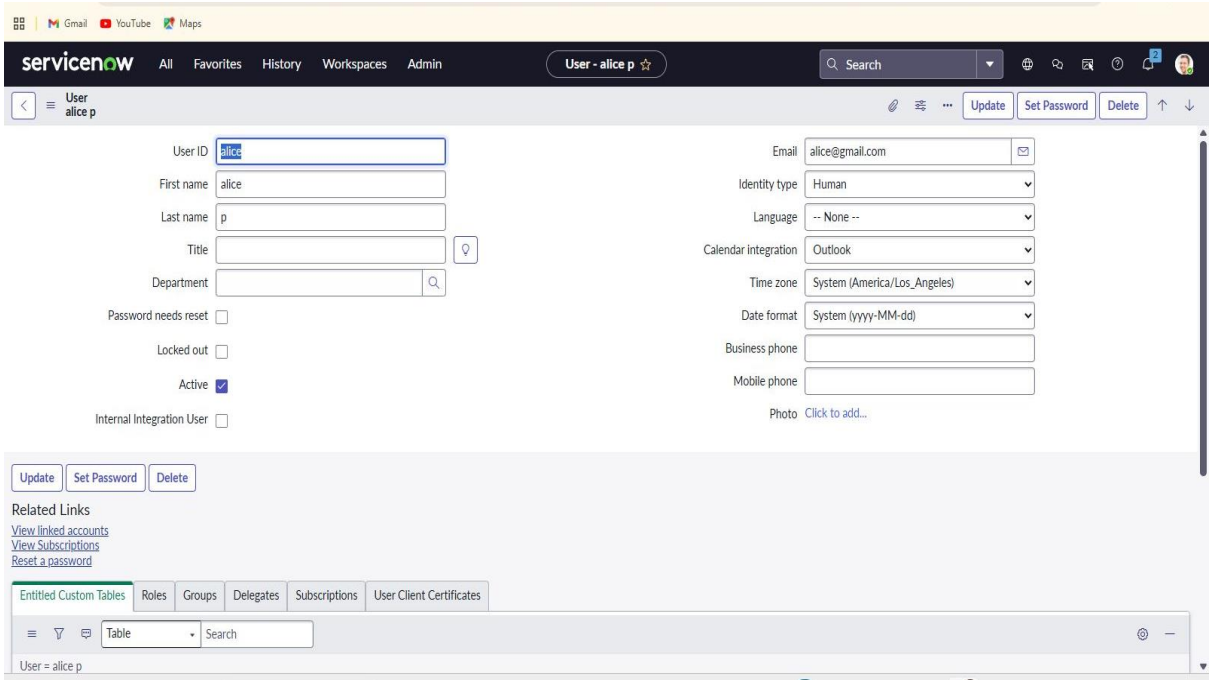
Activity 1.1 — Create Users

- Navigated: **All → Users** (System Security).

- Created Alice (Project Manager) and Bob (Team Member) with required details and clicked **Submit**.

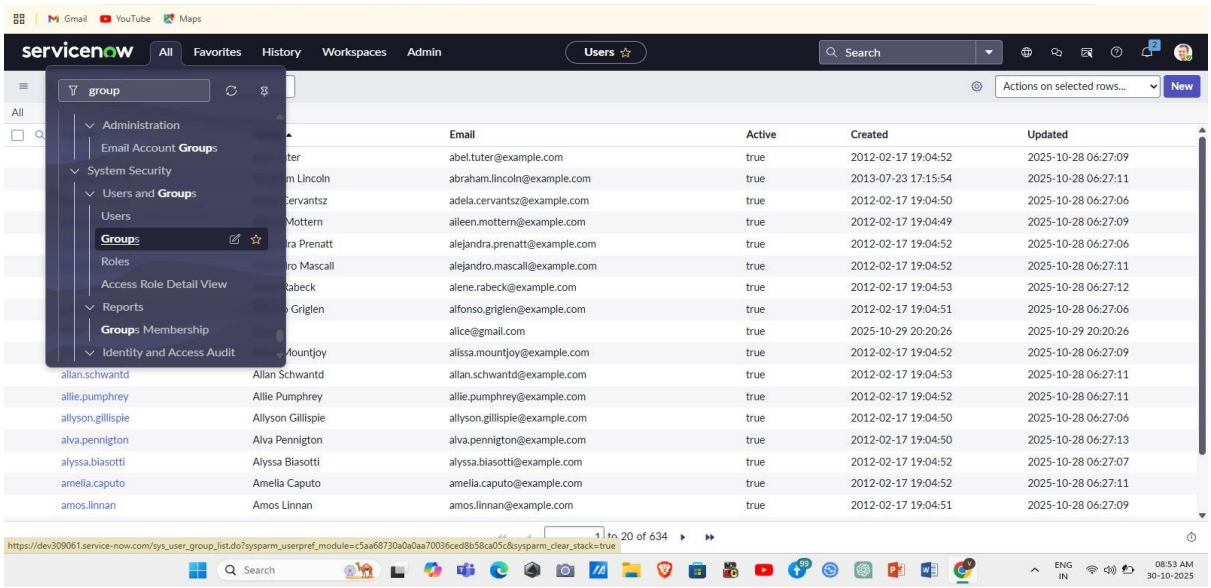






Activity 1.2 — Create Groups

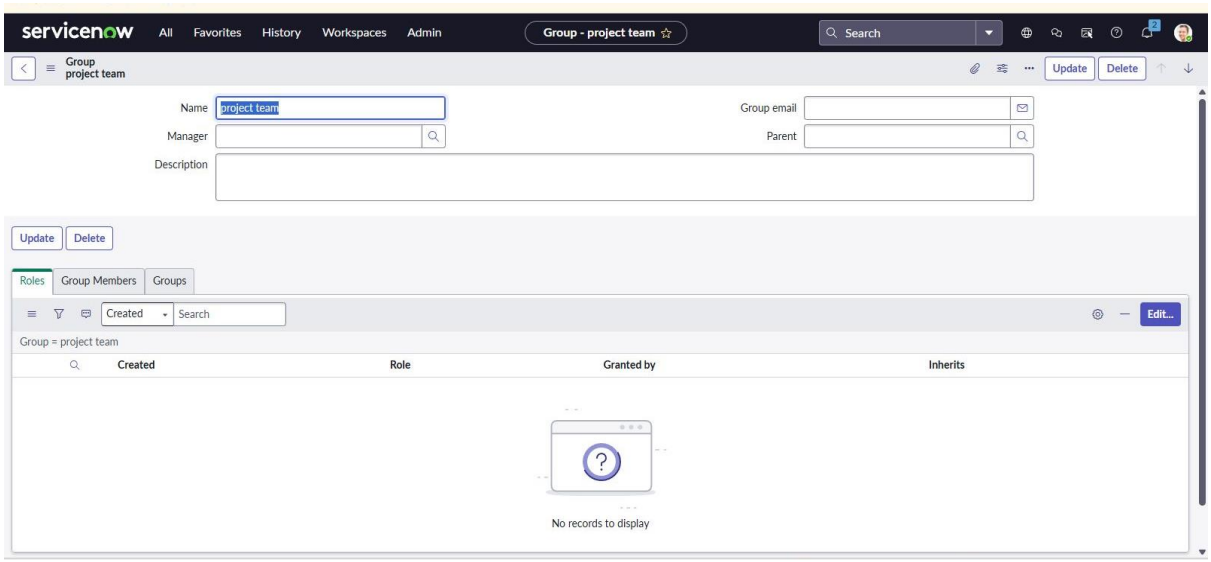
- Navigated: **All → Groups** (System Security).
- Created a group (e.g., *Project Management Team*).
- Added group description and saved.

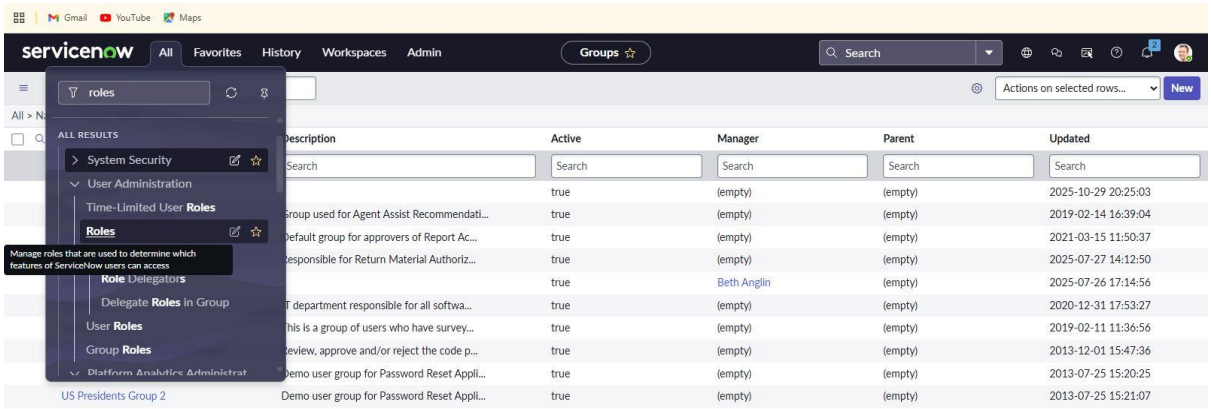


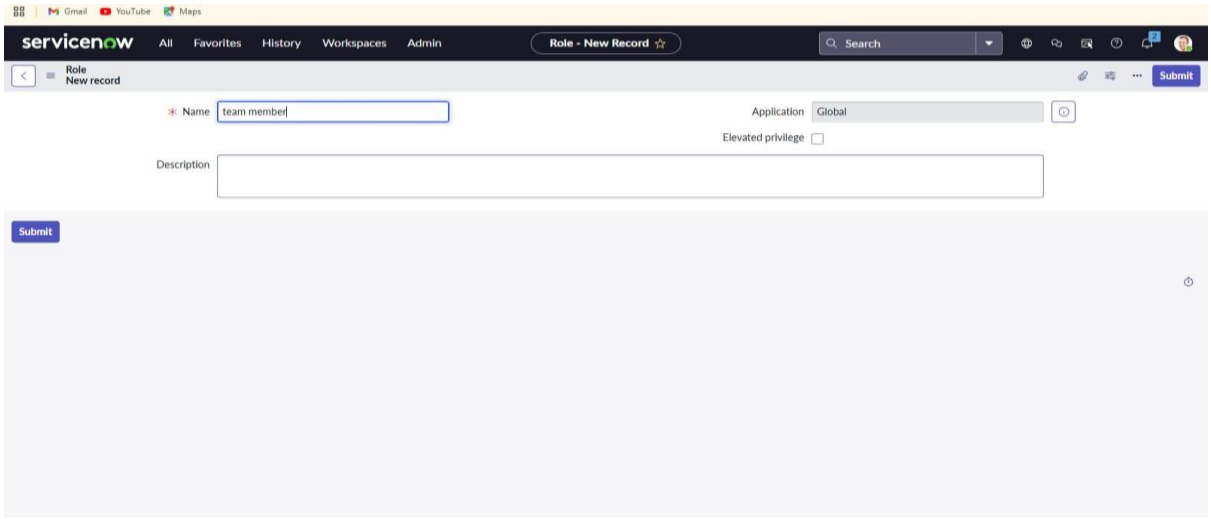
Milestone 2 — Roles & Role Assignment

Activity 2.1 — Create Roles

- Navigated: **All → Roles** (System Security).
- Created roles:
    - u_project_table — intended for users accessing project records
    - u_task_table — intended for users accessing task records
- Saved roles.

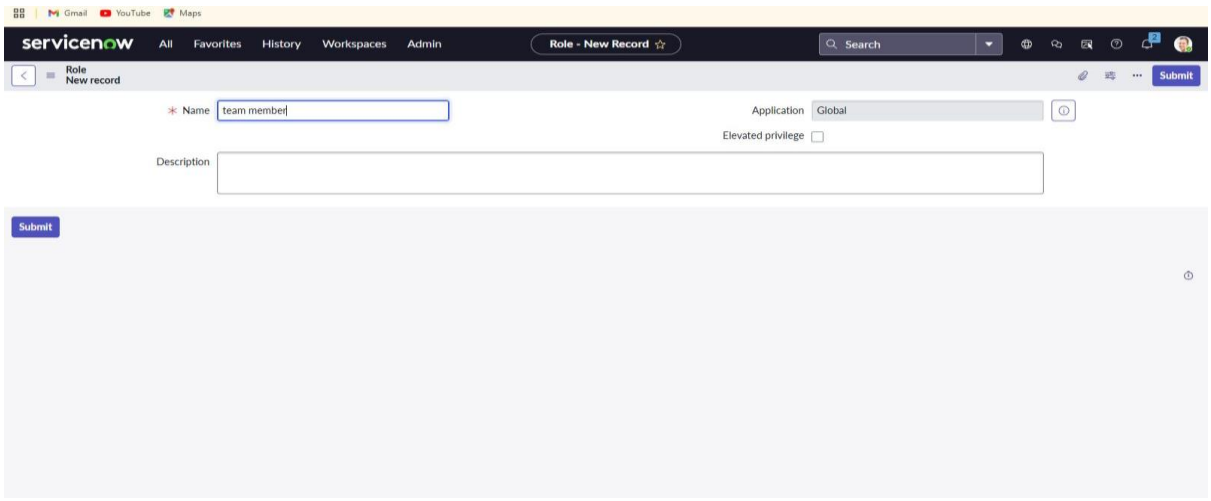






Activity 2.2 — Assign Roles to Users (associate to groups where required)

- For **Alice (Project Manager)**:
    - Opened Alice's user record → Roles → Edit → selected: *Project Member*, u_project_table, u_task_table. → Save/Update.
- For **Bob (Team Member)**:
    - Opened Bob's user record → Roles → Edit → selected: *Team Member*, u_task_table. → Save/Update.
- Verified roles are listed in each user's Roles tab.

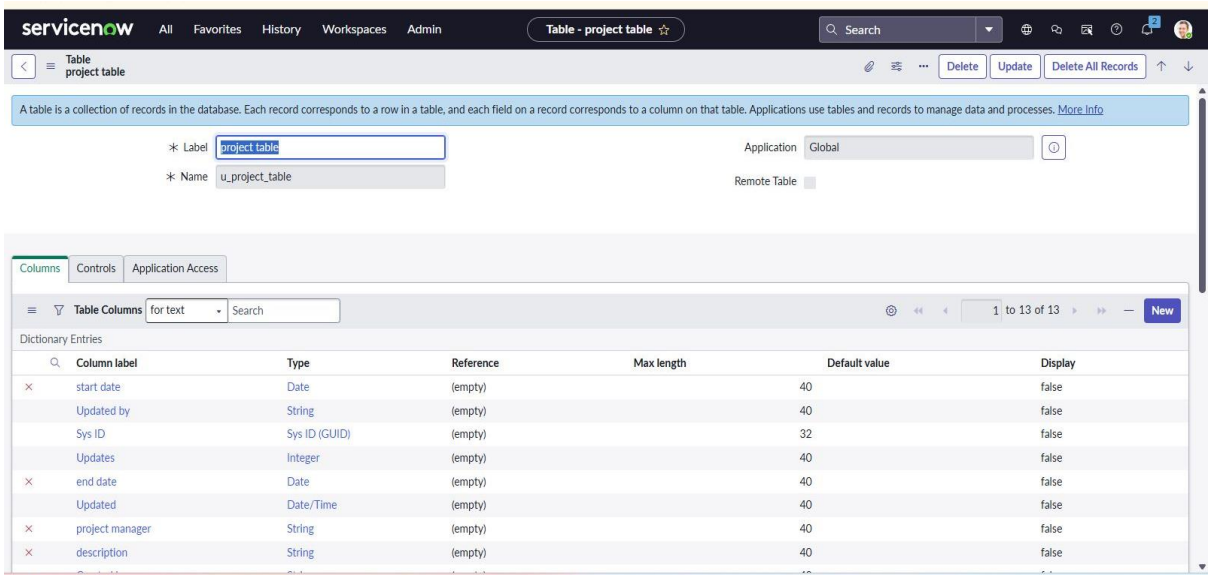


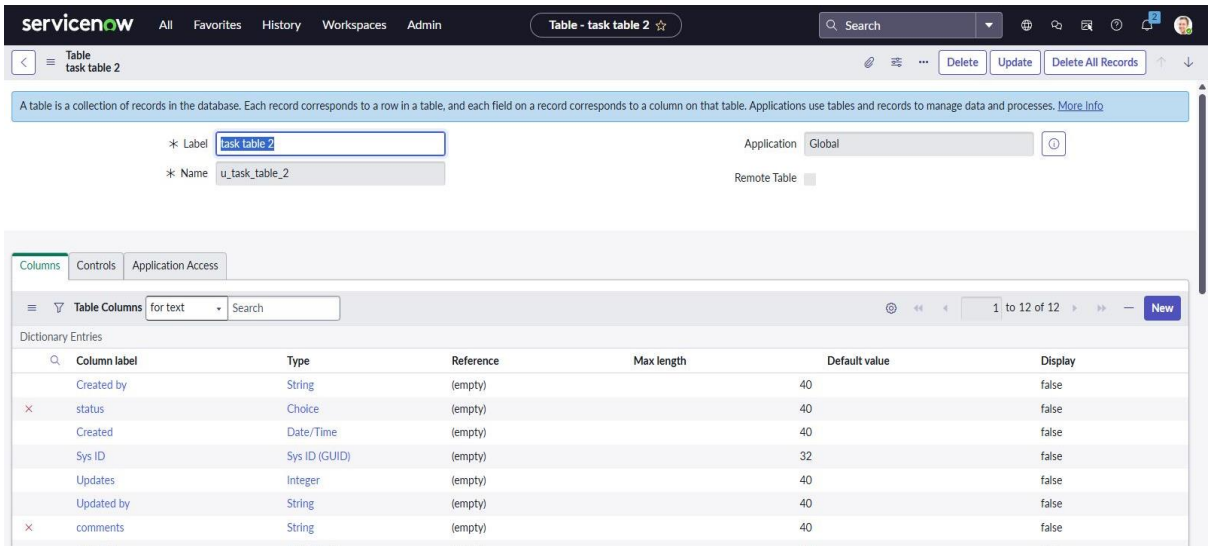
Milestone 3 — Application / Table Access

Activity 3.1 — Assign Table Access to Application Modules

- Found the auto-created **Project Table Application** and edited its module properties to give access to Project Member role.

- Edited **Task Table2 Application** module and added Project Member and Team Member roles so users with those roles can see the application/module in the navigator.

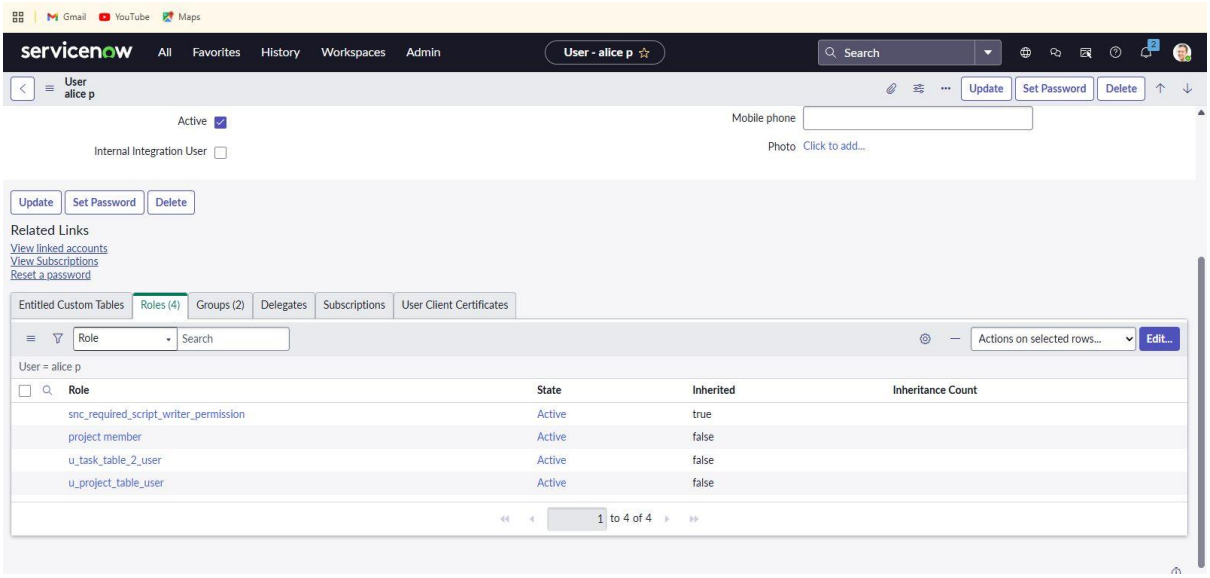




Activity 3.2 — Verification by Impersonation

- Used the **Profile → Impersonate User** function to impersonate Bob: confirmed Bob sees **Task Table2** module only.

- Impersonated Alice and confirmed access to both Project and Task modules.



Milestone 4 — Access Control Lists (ACLs)

Activity 4.1 — Create ACLs

- Navigated: **All → Access Control (ACL)** (System Security).

- Clicked **Elevate Role** to gain elevated privileges to create security records.

- Created ACL records for **Task Table2** and specific fields:

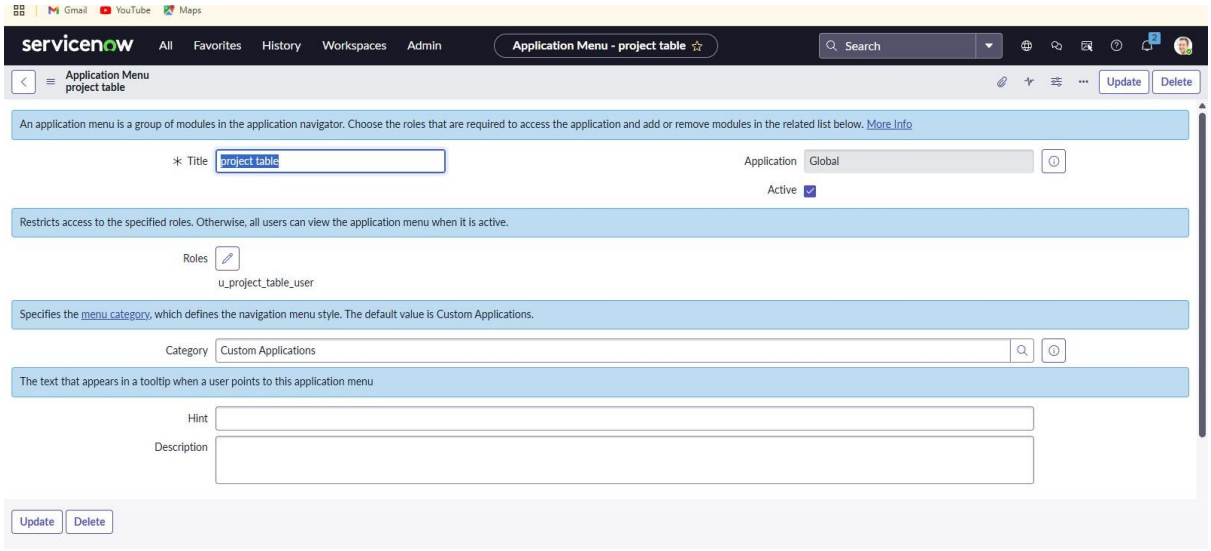
    - Table-level ACL for Task Table2 (create/read/write as needed).

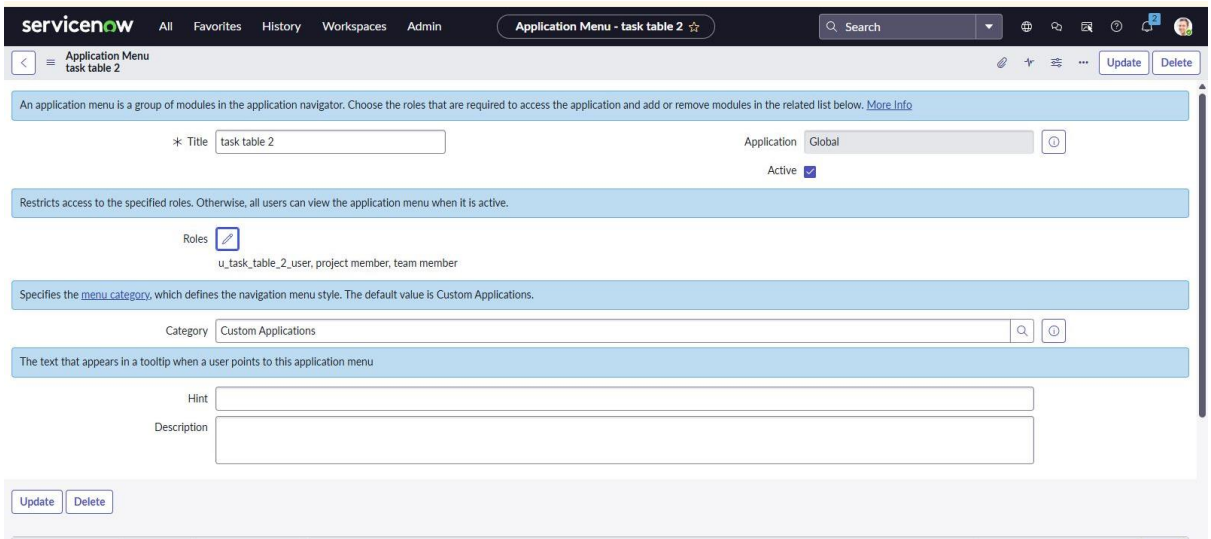
    - Field-level ACLs specifying that only users with Team Member role can edit comments and status fields.

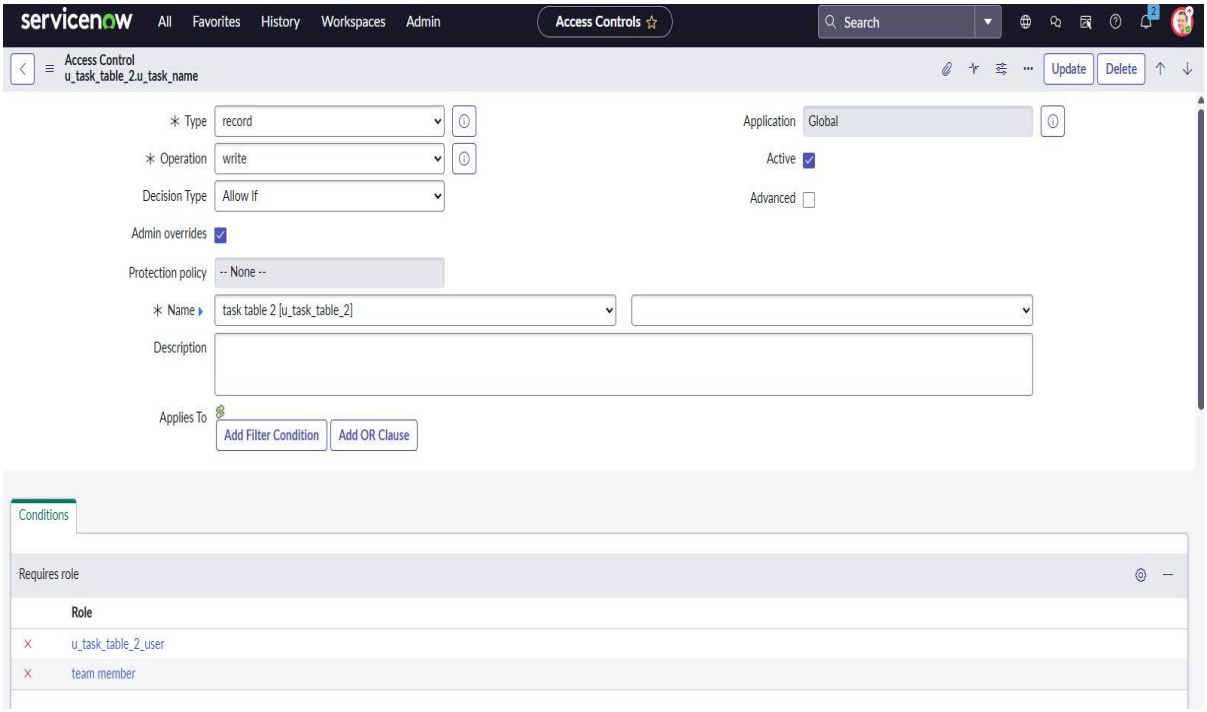
Activity 4.2 — ACL Validation

- Impersonated Bob and attempted to edit comments and status in Task Table2 — edits allowed per role.

- Verified that other unauthorized users (without u_task_table role) could not access or edit Task Table2 records.



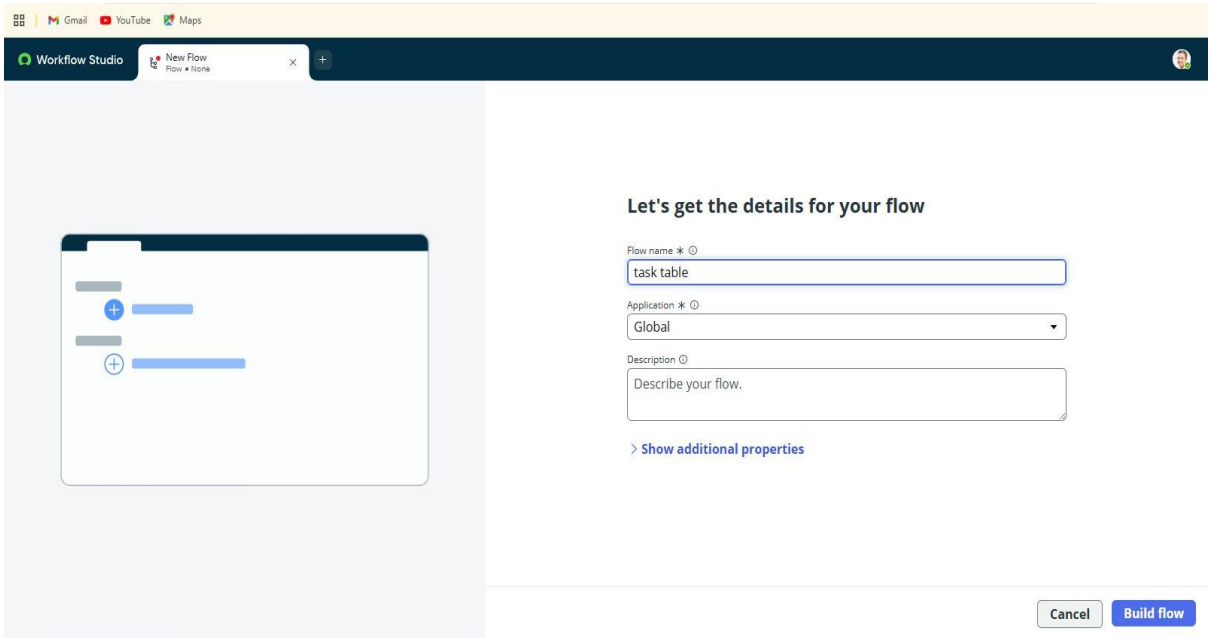


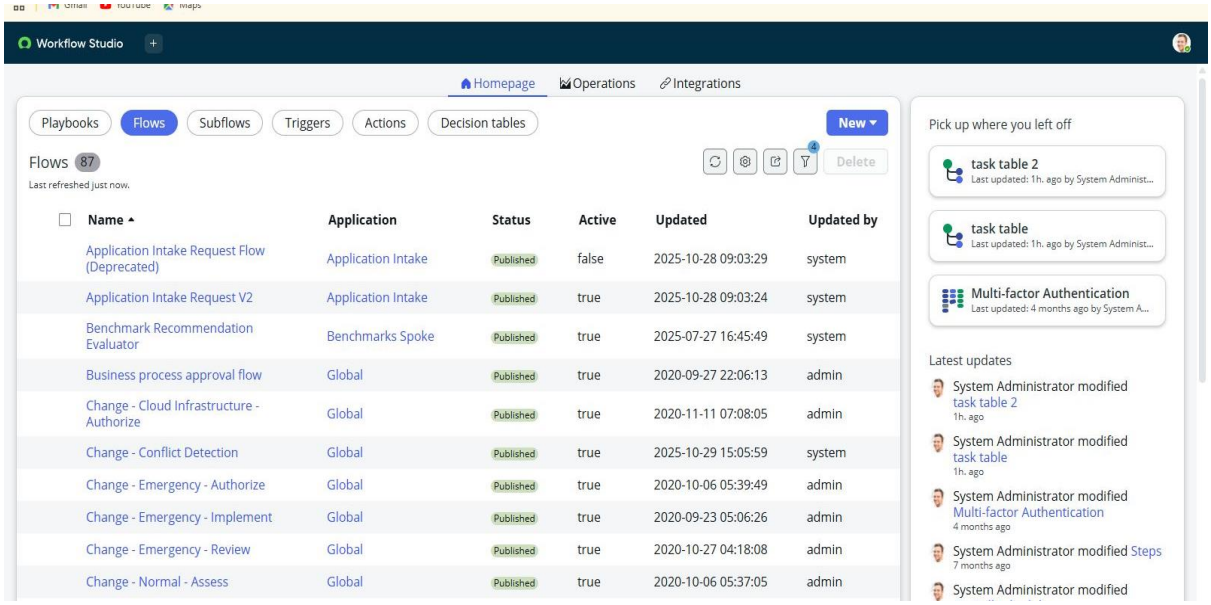


Milestone 5 — Flow Designer: Task Flow & Approval

Activity 5.1 — Create Flow

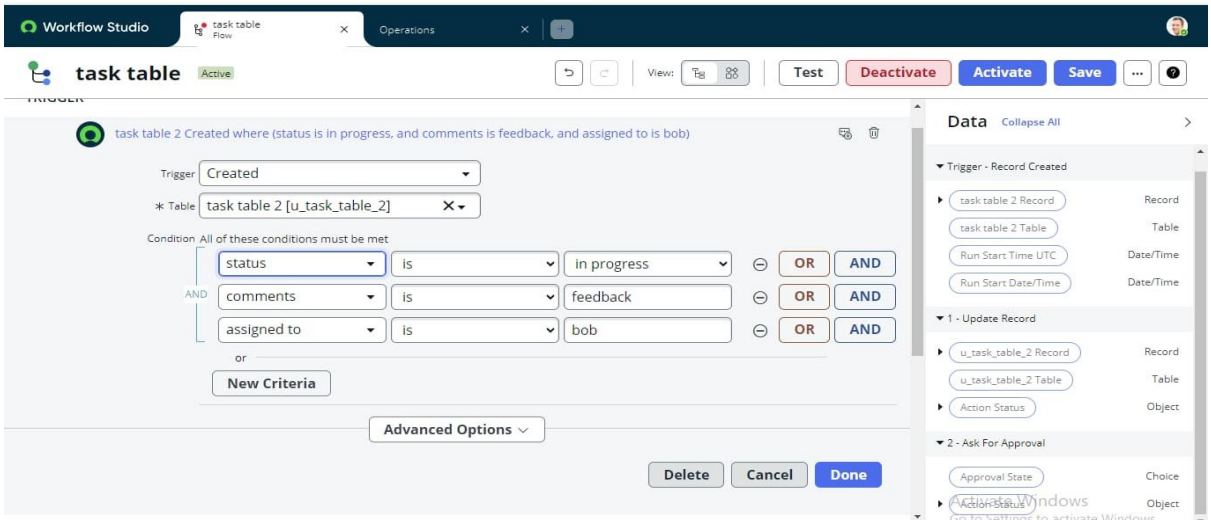
- Navigated: **All → Flow Designer** (Process Automation).
- Created a new Flow named **Task Table** in the Global application scope.

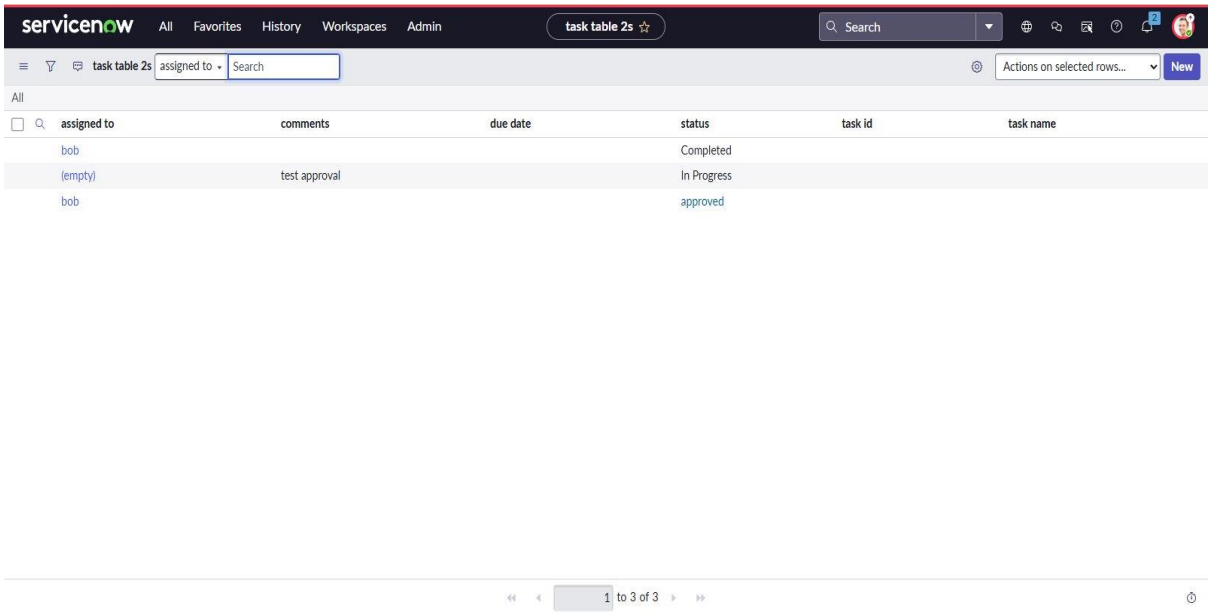




Activity 5.2 — Add Trigger

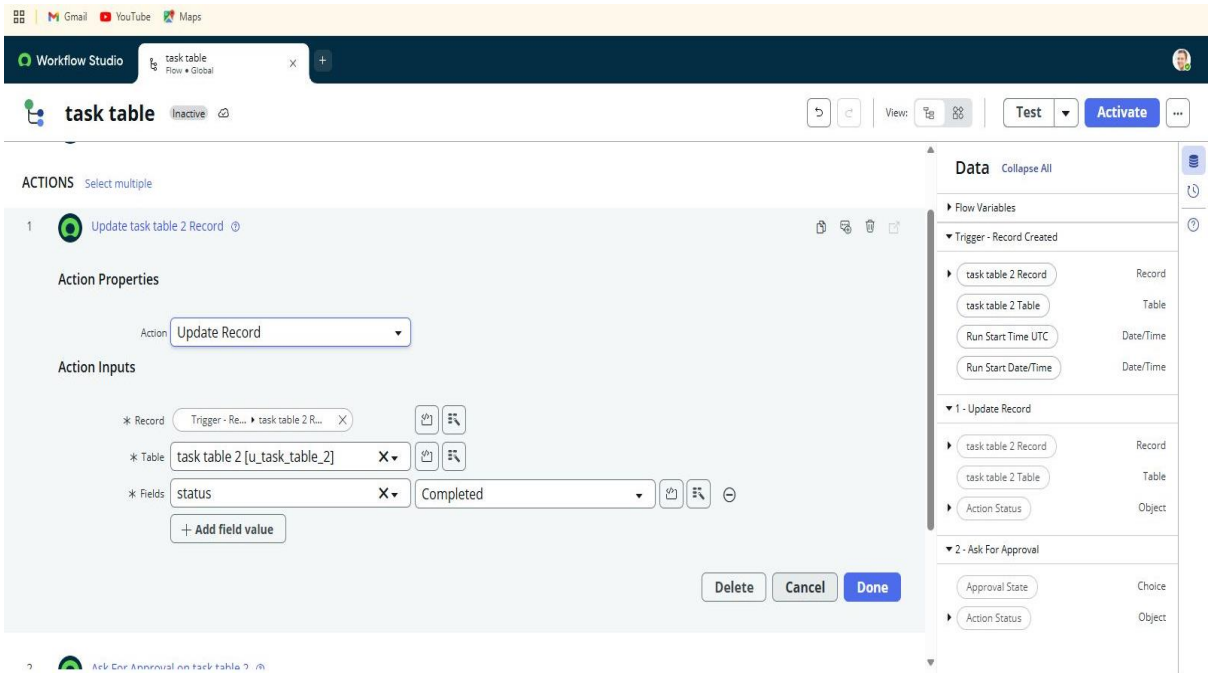
- Trigger: **Record Created** on **Task Table2**.
- Added trigger conditions:
  - status is In Progress
  - comments is Feedback
  - assigned_to is Bob

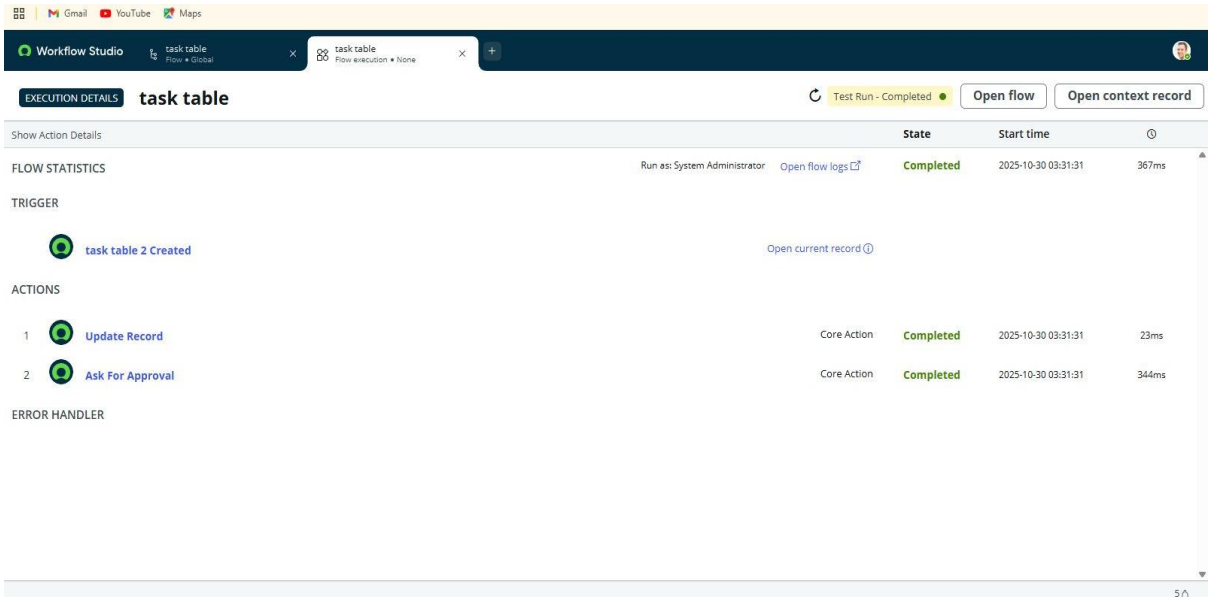




Activity 5.3 — Add Actions

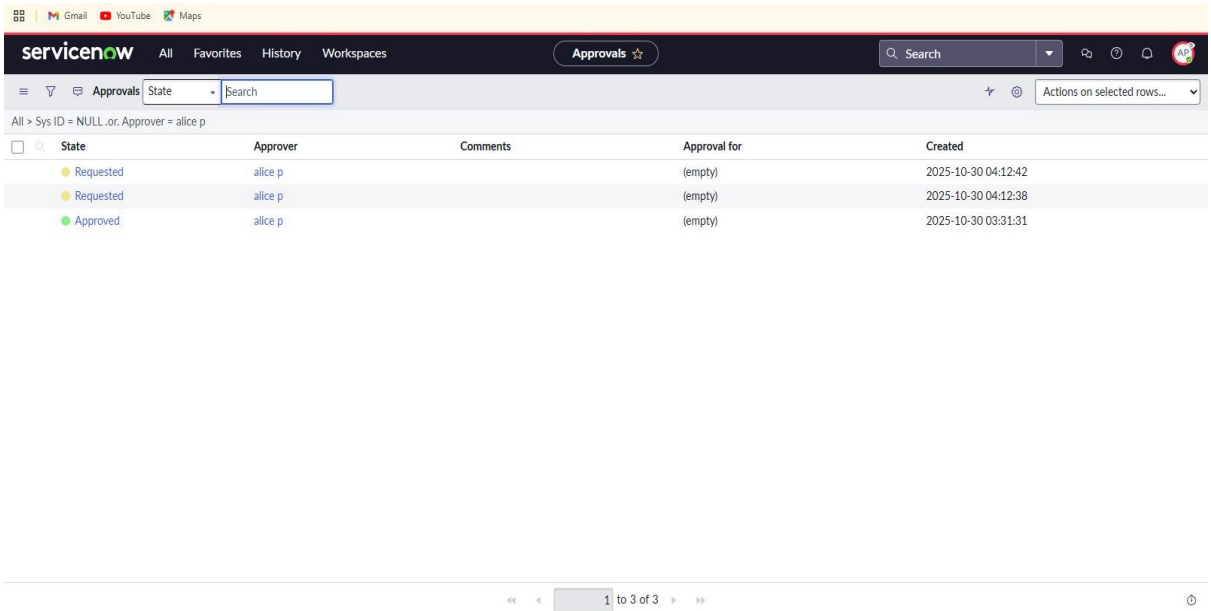
- **Action 1 — Update Record**: set status = Completed.
- **Action 2 — Ask for Approval**: configured approver as Alice (Project Manager); approval linked to the same task record.





Activity 5.4 — Flow Testing

- Created a task with conditions that match the trigger and assigned it to Bob.
- Verified Flow updated status to Completed.
- Verified Alice received approval request under **My Approvals** and could approve the task.



Phase 4: Testing

4.1 Test Strategy

- **Role Assignment Tests** — verify users have proper roles listed in their records.
- **Impersonation Tests** — verify visibility and edit rights from a user's perspective.
- **ACL Tests** — verify field-level and table-level ACLs restrict/allow access correctly.
- **Flow Execution Tests** — validate Flow triggers only when conditions match and actions complete successfully.
- **End-to-End Scenario** — create a record in Task Table2 → assign to Bob → Flow updates status → Alice receives approval → Alice approves.

5.2 Sample Test Cases

ID	Test Case	Expected Result	Status
TC1	Create Alice & Bob	Users created successfully	Pass
TC2	Assign roles to Alice/Bob	Roles listed in user record	Pass
TC3	Impersonate Bob	Bob sees Task Table2 only	Pass
TC4	Field-level ACL on comments/status	Bob can edit; others cannot	Pass
TC5	Flow trigger conditions met	Flow updates status & sends approval	Pass

Phase 5: Deployment & Setup Instructions

Follow these exact steps to reproduce the configuration on another ServiceNow instance:

5.1 Pre-requisites

- Admin or equivalent rights on target instance (to create users, roles, groups, ACLs, and flows).

- Have target custom tables project_table and task_table2 created (or create them via Table Builder).

5.2 Step-by-step Setup

1. **Create Users**
    - Navigate: **All** → **Users** → **New** → fill fields for Alice and Bob → **Submit**.

2. **Create Group**
    - **All** → **Groups** → **New** → Name: *Project Management Team* → **Submit**.

3. **Create Roles**
    - **All** → **Roles** → **New** → Create u_project_table and u_task_table → **Submit**.
4. **Assign Roles to Users**
    - Open Alice → Roles → **Edit** → add u_project_table, u_task_table, Project Member → **Save**.

    - Open Bob → Roles → **Edit** → add u_task_table, Team Member → **Save**.

5. **Configure Application Module Access**
    - Locate Project and Task application modules → Edit module → Add Project Member/Team Member roles to **Roles** field → **Save**.

6. **Create ACLs**

- **All → Access Control (ACL) → New** → Select table: task_table2 → define read/write/create as per policy → in **Requires role** add Team Member for field-level ACLs like comments and status → **Submit**.

7. **Create Flow in Flow Designer**

   - **All → Flow Designer → New Flow** named *Task Table* → Add **Trigger: Record Created** on task_table2 → set conditions (status, comments, assigned_to) → Add **Update Record** action (set status = Completed) → Add **Ask for Approval** (approver = Alice) → **Activate** flow.

8. **Validate**

   - Create a task that meets trigger conditions and follow the approval step as Alice.

9. Diagrams

9.1 ER Diagram

The ER diagram represents the relationships between the main entities in the ServiceNow Access Control project. It focuses on Users, Groups, Roles, Applications, and Access Control Lists (ACLs). The relationships define how users are assigned to groups, groups have roles, and roles determine access permissions for applications and tables.

Entities:

- User**:** Represents individual users (e.g., Alice, Bob) who access the ServiceNow platform.
- Group**:** A collection of users with common responsibilities or privileges.
- Role**:** Defines the permissions assigned to a user or group (e.g., itil, admin).
- Application**:** Represents the specific ServiceNow application to which access is granted.
- ACL (Access Control List)**:** Defines security rules for tables and records.

Relationships:

- A User can belong to one or more Groups.
- A Group can have multiple Roles.
- Roles determine access rights to Applications.
- ACLs restrict or permit access to data within Applications.

9.2 Flow Diagram

The process flow diagram illustrates how user access and ticket assignment flow within the ServiceNow environment. It covers user creation, group assignment, role mapping, ACL enforcement, and automated ticket assignment via Flow Designer.

Steps:

1. User Creation**:** Admin creates users such as Alice and Bob.
2. Group Assignment**:** Users are added to specific groups (e.g., IT Support, Service Desk).
3. Role Assignment: Appropriate roles are assigned to users/groups to define access privileges.
4. Application Access Configuration**:** Access to applications and tables is controlled based on assigned roles.
5. ACL Creation**:** Access Control Lists are configured to secure table-level and record-level access.
6. Flow Design**:** Automated flow assigns operated tickets to the correct group based on role and group mapping.
7. Ticket Operation**:** The assigned group operates and resolves tickets based on permissions granted.

10. Validation Rules

10.1 Validation Rules Implemented / Recommended

- **Implemented (by configuration):**
    - Role membership controlled via Roles tab in user records (manual assignment).
    - Application visibility restricted by module Roles (configured).
    - ACLs enforcing field-level restrictions for comments and status.
- **Recommended (not scripted in your run — optional future work):**
    - Validate assigned_to is not empty on active tasks (UI Policy or Client Script).
    - Ensure status transitions follow allowed states (e.g., only In Progress → Completed via flow) — can be enforced via Business Rule or UI Action if needed.

- Prevent overlapping assignments for the same task (if multiple assignment logic arises).

11. User Guide

11.1 For Project Manager (Alice)

- **View tasks:** Use navigator → *Task Table2* (visible because of u_project_table / Project Member role).
- **Approve tasks:** Go to **My Approvals** → view pending approvals → right-click and **Approve**.
- **Manage team:** Open **Groups** and **Users** (requires elevation) to review memberships.

11.2 For Team Member (Bob)

- **View and edit tasks:** Use navigator → *Task Table2*. You can edit comments and update status if ACL allows.
- **Work on tasks:** Update comments field to give status or progress notes.

11.3 For Admin (Replication / Maintenance)

- Use the **Roles** tab in user records to assign/unassign roles.
- Use **Flow Designer** to edit/activate/deactivate the Task Table flow.
- Modify ACLs under **All → Access Control (ACL)** if permission changes are required.

12. Limitations

- All role assignments were done manually through the User record — there is no self-service role request form implemented in this iteration.

- No time-based automatic revocation of roles (temporary elevation) was implemented.

- Validation rules that restrict advanced state transitions were not implemented as Business Rules; these are left as recommendations.

- No custom logging/audit table was created as part of this project (only standard platform logs are available).

13. Future Enhancements

- Add a **Service Catalog** item for self-service role requests and approval flows.

- Implement **temporary role elevation** with automatic expiry and email notifications.

- Add stricter **state transition validation** for tasks via Business Rules.

- Add a small **dashboard** summarizing tasks by status, pending approvals, and role-based access statistics.

14. Conclusion

This project implements a practical, minimal-but-complete ServiceNow configuration to manage users, groups, roles, ACLs, and an automated task assignment/approval flow between Alice (Project Manager) and Bob (Team Member). The implementation improves accountability, reduces confusion about access rights, and demonstrates how ServiceNow constructs (Users, Groups, Roles, ACLs, Flow Designer) work together to create a governance-backed task management process.