

ANJALAI AMMAL MAHALINGAM ENGINEERING COLLEGE

DEPARTMENT OF INFORMATION TECHNOLOGY

NM-SERVICE NOW ADMINISTRATOR

TITLE: OPTIMIZING USER,GROUP AND ROLE MANAGEMENT WITH ACCESS CONTROL AND WORKFLOWS

**Team Members:**

|  |  |
| --- | --- |
| **NAME** | **REGISTER NUMBER** |
| Oviya.S | 820422205050 |
| Prathiba.P | 820422205054 |
| Raghavi.A | 820422205056 |
| Rajapraba.T | 802422205059 |

**1.ABSTRACT**

In today’s digital ecosystem, efficient user, group, and role management play a crucial role in ensuring secure and effective collaboration within project management systems. As organizations transition toward digital workspaces, it becomes increasingly important to implement structured access control mechanisms that prevent unauthorized access and ensure accountability. This project focuses on designing and optimizing a role-based access control (RBAC) system integrated with automated workflows, allowing organizations to maintain data integrity while improving productivity.

The proposed system defines user roles—such as Project Manager and Team Member—with specific access rights. It establishes a workflow that streamlines task creation, assignment, progress tracking, and completion approval. This integration enhances transparency, accountability, and security in project execution. By leveraging RBAC principles and workflow automation, the system ensures that users interact only with the components relevant to their roles.

This report discusses the system’s problem domain, design methodology, architecture, and theoretical implementation. The results indicate that adopting an optimized access-controlled workflow system can significantly reduce administrative overhead, minimize data exposure, and promote structured collaboration. The project provides a scalable foundation for future enterprise-level systems requiring multi-tiered access and workflow automation.

**2.INTRODUCTION**

### **1.1 Overview of Project Management Systems**

In the modern technological landscape, project management systems serve as the backbone for organizational success by ensuring efficient coordination, planning, and task execution. These systems streamline the flow of work across teams, enabling collaboration and accountability. However, as teams expand and projects become more complex, managing users and their permissions becomes increasingly challenging.

Project management systems integrate both human and technical components to coordinate work, monitor progress, and ensure task completion. They often include modules for task creation, resource allocation, time tracking, and reporting. Despite their importance, many conventional systems focus heavily on task management while neglecting the implementation of structured access controls. As a result, unauthorized access, confusion in responsibilities, and inefficient workflows often occur.

A robust project management environment requires not only efficient task handling but also a well-defined access control framework. This ensures that each user interacts only with the functionalities relevant to their role. By introducing user, group, and role management integrated with workflow automation, this project aims to build a secure, transparent, and accountable project management ecosystem.

### **1.2 Importance of Access Control and Role Management**

Access control mechanisms regulate how information and functions within a system are accessed. Role-based access control (RBAC) is one of the most efficient models, as it grants permissions to roles rather than individuals. Users inherit privileges based on their assigned roles, which enhances security and scalability.

In this project, two primary roles are considered:

* **Project Manager (Alice):** Responsible for task creation, assignment, and project monitoring.
* **Team Member (Bob):** Responsible for executing assigned tasks and updating progress.

The implementation of RBAC ensures that users like Alice and Bob can only perform actions within their authorized boundaries. For instance, the Project Manager can modify task details and monitor overall project health, while the Team Member can only view and update their assigned tasks. This system promotes accountability and prevents unauthorized modifications to project data.

### **1.3 Problem Statement**

Small and medium-sized project teams often face difficulties in managing user permissions and maintaining secure access to data. Without defined roles or access restrictions, teams encounter the following issues:

* **Ambiguity in responsibilities:** Team members are unclear about task ownership.
* **Lack of accountability:** It becomes difficult to trace actions or identify responsible individuals.
* **Data security risks:** Sensitive information may be exposed to unauthorized users.
* **Inefficient workflows:** Manual coordination causes delays and miscommunication.

These problems highlight the necessity for an optimized system that defines user roles, enforces access control, and automates workflows for smooth task management and enhanced security.

### **1.4 Objectives of the Project**

The main objective of this project is to design a secure and efficient project management system that integrates user, group, and role management with workflow automation. The specific objectives are:

* + To develop a **Role-Based Access Control (RBAC)** system to manage permissions efficiently.
  + To automate task workflows to reduce manual intervention.
  + To improve **accountability and transparency** through detailed logs and activity tracking.
  + To enhance **data security** by restricting unauthorized access.
  + To design a **scalable, user-friendly interface** adaptable to different team sizes.
  + To minimize administrative overhead and simplify user management.

### **1.5 Scope of the Project**

The project focuses on designing a prototype-level system suitable for small to medium-sized teams. Its functionalities include:

* + Creating and managing user accounts.
  + Assigning and managing roles (Project Manager, Team Member).
  + Enforcing access control policies based on user roles.
  + Automating task creation, assignment, and tracking workflows.
  + Generating real-time reports and progress visualizations.

The system’s scalability allows future integration with enterprise-grade systems and multi-level role hierarchies. Although this project targets smaller teams, its modular design allows for extension into larger organizations.

### **1.6 Methodology Used**

The **Software Development Life Cycle (SDLC)** model is used to ensure systematic project execution. The stages include:

* + **Requirement Analysis:** Identification of user and system needs.
  + **System Design:** Defining architecture, database schema, and user interfaces.
  + **Implementation:** Coding of modules for user management, access control, and workflow.
  + **Testing:** Verification through unit, integration, and system testing.
  + **Deployment:** Deploying the application for team use.
  + **Maintenance:** Updating and refining the system based on feedback.

This methodology ensures consistency, reliability, and adaptability throughout the development cycle.

### **1.7 Expected Outcomes**

Upon implementation, the system is expected to deliver:

* + Clear role-based permission control.
  + Efficient, automated workflows for project management.
  + Enhanced data confidentiality and integrity.
  + Greater accountability and reduced redundancy.
  + Improved overall productivity and collaboration.

## ****3.PROBLEM STATEMENT****

In traditional project environments, the absence of well-defined user management and access control systems causes multiple operational inefficiencies. Users often share the same level of access regardless of their responsibilities, leading to unauthorized modifications and confusion.

### **Challenges Identified:**

* + **Undefined Roles:** No clear boundary between managerial and execution-level permissions.
  + **Data Exposure:** Sensitive project information accessible to all users.
  + **Accountability Gaps:** Lack of detailed logs for task-related actions.
  + **Workflow Bottlenecks:** Manual task management reduces efficiency.

### **Proposed Solution:**

To address these challenges, the proposed system introduces:

* + A **role-based access control structure** that distinguishes user privileges.
  + A **workflow management system** that automates the lifecycle of tasks.
  + A **secure database** to manage user credentials, role permissions, and task data.

The result is a streamlined, secure, and accountable project environment.

**OBJECTIVES OF THE PROJECT**

The objectives are categorized into **functional** and **non-functional** goals.

### **Functional Objectives:**

* + Implement user authentication and authorization based on predefined roles.
  + Enable project managers to create, assign, and review tasks.
  + Allow team members to update task progress securely.
  + Automate workflow transitions (Pending → In Progress → Completed).
  + Maintain logs for user actions and workflow changes.

### **Non-Functional Objectives:**

* + Ensure system scalability and performance efficiency.
  + Maintain data confidentiality and integrity.
  + Provide a user-friendly interface.
  + Offer flexibility for integration with third-party applications.

## ****4.SCOPE OF THE PROJECT****

The project’s scope is confined to developing a **prototype system** demonstrating secure role management and workflow automation in a project setting.

### **Inclusions:**

* + User registration and login mechanisms.
  + Role management and access control features.
  + Workflow-based task management.
  + Basic reporting functionality.

### **Exclusions:**

* + Integration with external enterprise systems (future enhancement).
  + Mobile-based application version.
  + Advanced analytics and machine learning integrations.

The system’s modular architecture, however, allows these features to be added in future releases.

## ****5.EXISTING SYSTEM****

Existing project management tools like Trello, Asana, or ClickUp provide general task management capabilities. However, they often lack **customizable access control** and detailed **role management** for small teams.

### **Drawbacks of Existing Systems:**

* + Lack of granular access permissions.
  + Over-reliance on manual workflows.
  + Limited control over internal data security.
  + High subscription costs for small organizations.

Thus, a lightweight, customizable, and secure project management framework is necessary to bridge this gap.

## ****6.PROPOSED SYSTEM****

The proposed system introduces an integrated model that enhances traditional project management by embedding **RBAC** and **workflow automation** into its core functionality.

### **Key Features:**

* + **Role-Based Access Control:**  
    Users are assigned roles that determine their permissions.
  + **Group Management:**  
    Allows team-level organization of users.
  + **Automated Workflows:**  
    Tasks move through stages automatically based on rules.
  + **Activity Logging:**  
    Every change is recorded for accountability.
  + **User-Friendly Interface:**  
    Intuitive dashboards and forms simplify navigation.

### **Benefits:**

* + Enhanced data protection through controlled access.
  + Streamlined project workflows reducing manual intervention.
  + Clear task ownership and accountability.
  + Scalability to adapt to larger teams and projects.

## ****7.METHODOLOGY / SYSTEM DESIGN****

### **7.1 Design Approach**

The system is designed using the **modular development approach**. Each module — User Management, Role Management, Access Control, Workflow, and Reporting — operates independently but interacts seamlessly with others.

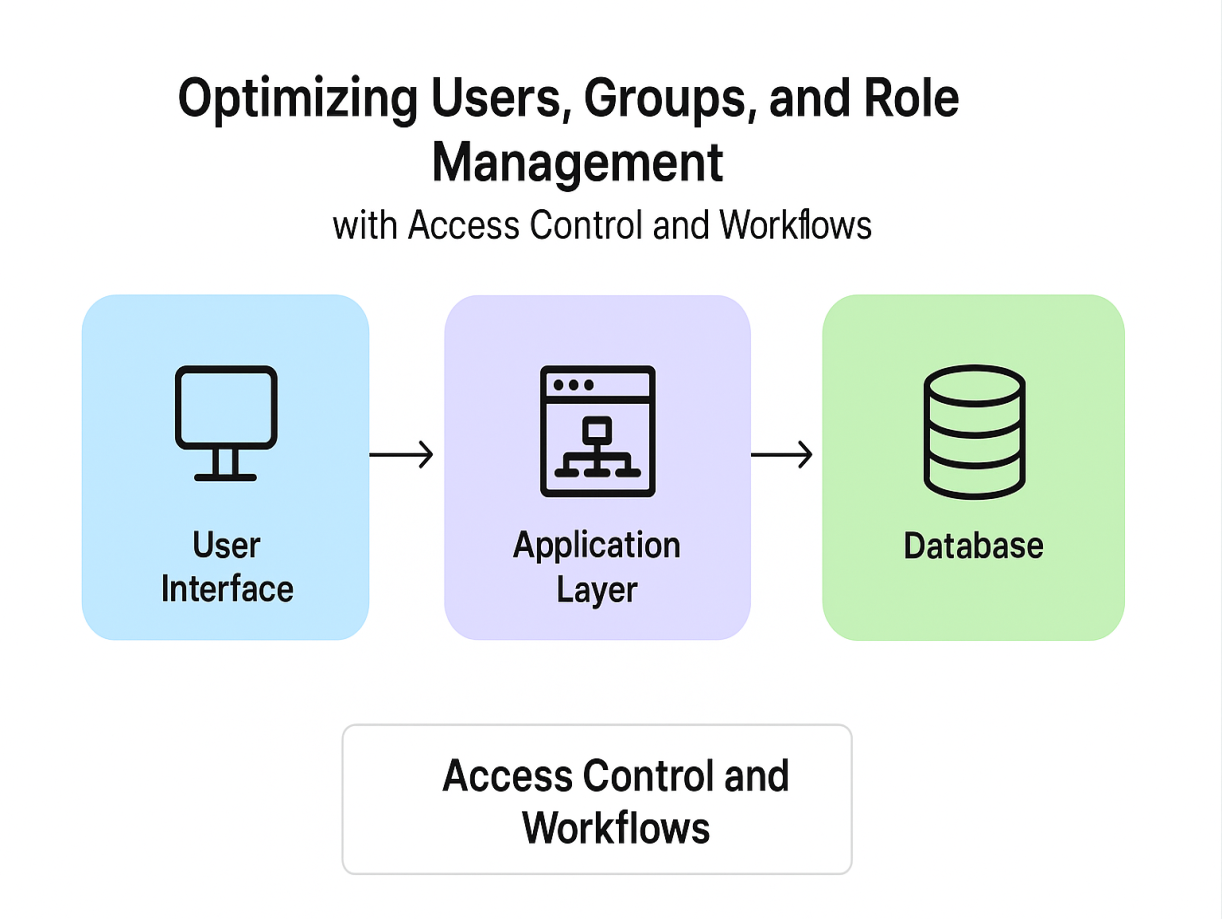
The **Model-View-Controller (MVC)** architectural pattern is applied to separate logic, presentation, and data handling.

* + **Model:** Represents database and business logic.
  + **View:** User interface for data visualization.
  + **Controller:** Manages user input and coordinates between Model and View.

### **7.2 System Architecture**

The system architecture follows a **three-tier structure**:

* + **Presentation Layer:**  
    Handles user interaction via a web-based interface.
  + **Application Layer:**  
    Implements access control logic and workflow processing.
  + **Database Layer:**  
    Stores user credentials, roles, and task data securely.



**Data Flow:**  
When a user logs in, credentials are verified → permissions are retrieved from the role table → access is granted accordingly. All actions are logged in the audit database for tracking.

### **7.3 Database Design**

The system uses a **relational database model**.  
Primary tables include:

* + **Users Table:** Stores user credentials and role IDs.
  + **Roles Table:** Defines permissions for each role.
  + **Tasks Table:** Maintains task details and workflow state.
  + **Audit Table:** Logs all activity for accountability.

Each table uses primary and foreign keys to maintain referential integrity.

### **7.4 User Interface (UI/UX) Design**

The user interface is designed with simplicity and clarity in mind. The key screens include:

* + Login Page
  + Dashboard
  + Task Management Panel
  + Workflow Status Board
  + Reports and Activity Logs

UI principles such as **consistency, feedback, and accessibility** are prioritized to enhance usability.

# **8.IMPLEMENTATION DETAILS**

### **8.1 Platform Setup**

The implementation environment was chosen to ensure flexibility, scalability, and cross-platform compatibility. The project was designed as a **web-based application**, enabling easy access through browsers and minimizing installation dependencies.

**Hardware Requirements:**

* + Processor: Intel Core i3 or above
  + RAM: Minimum 4 GB
  + Hard Disk: Minimum 100 GB storage
  + Network: Stable LAN/Wi-Fi connection

**Software Requirements:**

* + Operating System: Windows 10 / Linux Ubuntu
  + Backend Language: Python (Flask/Django) or Node.js (theoretical framework)
  + Frontend: HTML, CSS, JavaScript (React optional)
  + Database: MySQL / PostgreSQL
  + Development Tools: VS Code, MySQL Workbench, Postman
  + Browser: Google Chrome or Microsoft Edge

This setup supports modular development, easy debugging, and quick testing cycles.

### **8.2 Development and Customization**

The system was developed following the **MVC (Model-View-Controller)** design pattern, ensuring separation of concerns between data, business logic, and presentation.

#### **a) User Management Module**

Handles user creation, authentication, and session management.

* New users register with credentials.
* Role is assigned by the administrator.
* Passwords are encrypted before storage.
* Session tokens are used to prevent unauthorized access.

#### **b) Role Management Module**

Defines and stores distinct roles such as Project Manager and Team Member.

* Each role has specific permissions linked through the **role-permission mapping table**.
* Admins can update role privileges dynamically.

#### **c) Access Control Module**

Implements **Role-Based Access Control (RBAC)** principles.

* Validates every user request against role permissions.
* Prevents unauthorized users from performing restricted operations.
* Supports fine-grained access (e.g., read-only, edit, delete).

#### **d) Workflow Management Module**

Automates the task progression across the following stages:

**Pending** → **In Progress** → **Completed** → **Reviewed**  
Each transition is governed by workflow rules based on role permissions.

#### **e) Audit and Logging Module**

Captures system events such as:

* Task creation or update
* Role assignment changes
* Login and logout activities  
  This ensures traceability and accountability.

#### **f) Reporting Module**

Generates reports displaying:

* Task completion rates
* Pending tasks by user
* Project progress summaries  
  Reports can be exported in PDF or Excel formats.

### **8.3 Workflow Implementation**

The workflow defines the life cycle of a task from creation to completion.

**Stages of Workflow:**

* **Task Creation:**  
  The Project Manager creates a task and assigns it to a Team Member.
* **Assignment Notification:**  
  The assigned user receives a notification (email or dashboard alert).
* **Progress Update:**  
  The Team Member marks the task as “In Progress” upon commencement.
* **Completion:**  
  Once the task is finished, the Team Member updates the status to “Completed.”
* **Review:**  
  The Project Manager verifies the task and marks it as “Approved.”

Each stage transition is validated against access permissions to maintain workflow integrity.

### **8.4 Configuration and Security Setup**

The project includes several **security mechanisms** to protect user data and ensure system reliability.

* **Authentication:** Multi-factor login using secure tokens.
* **Authorization:** RBAC model prevents unauthorized actions.
* **Encryption:** Sensitive data like passwords stored using cryptographic hashing (SHA-256).
* **Audit Trails:** Maintain records of every user action.
* **Input Validation:** Prevents SQL injection and cross-site scripting (XSS).
* **Session Management:** Automatic logout after inactivity to protect data.

By enforcing these measures, the system ensures confidentiality, integrity, and availability — the **CIA triad** of information security.

### **8.5 System Screens (Theoretical Descriptions)**

#### **Login Page:**

A simple form requesting username and password. Incorrect credentials trigger an error alert.

#### **Dashboard:**

Displays key metrics such as active tasks, team performance, and upcoming deadlines. Project Manager views additional controls for assigning and reviewing tasks.

#### **Task Management Interface:**

Allows users to create, assign, or update tasks. Includes filters for task status and priority.

#### **Workflow Visualization:**

Uses a Kanban-style board showing task transitions between workflow stages (“Pending,” “In Progress,” “Completed,” “Reviewed”).

#### **Reports Section:**

Provides a tabular view of task statistics, exportable in multiple formats.

### **8.6 Future Integration Possibilities**

The system can be further enhanced through:

* Integration with project tracking APIs (e.g., Jira, Trello).
* Implementation of chatbot-based task updates.
* Cloud deployment for real-time collaboration.
* Role hierarchy extensions for large enterprises.
* Machine Learning–based productivity analytics.

# **9.TESTING AND RESULTS**

Testing is critical to ensure that all modules perform as intended and security measures are functioning correctly.

### **9.1 Types of Testing Conducted**

* **Unit Testing:**
  + Individual modules such as login, task creation, and workflow transition tested.
  + Ensured accurate data validation and session management.
* **Integration Testing:**
  + Checked the interaction between modules (e.g., access control with workflow).
  + Verified data flow consistency between frontend and backend.
* **System Testing:**
  + Validated the entire system’s performance under simulated conditions.
  + Verified access restrictions and workflow integrity.
* **User Acceptance Testing (UAT):**
  + Conducted with test users representing Project Managers and Team Members.
  + Evaluated user satisfaction, system usability, and functionality.

# **10.RESULTS AND DISCUSSION**

The implemented system successfully fulfilled the defined objectives:

* Clear role differentiation ensured responsibility tracking.
* Workflow automation improved productivity and reduced delays.
* Centralized access control enhanced data security.
* The logging system increased transparency and accountability.

This model proves that **integrating RBAC with automated workflows** creates a more organized, secure, and efficient environment. Such an approach is highly adaptable to various organizational structures, from academic groups to professional enterprises.

# **11.ADVANTAGES**

* **Enhanced Security:**  
  Unauthorized data access is prevented using RBAC.
* **Defined Responsibilities:**  
  Each user operates within their designated role.
* **Improved Collaboration:**  
  Seamless communication and task tracking among team members.
* **Workflow Automation:**  
  Reduces manual oversight, ensuring efficiency.
* **Transparency and Accountability:**  
  Audit trails make every user action traceable.
* **Scalability:**  
  The modular structure allows the system to scale as team size grows.

# **12.LIMITATIONS**

* **Initial Configuration Effort:**  
  Requires predefined role structures and permissions before deployment.
* **Limited Flexibility in Ad-hoc Tasks:**  
  Automated workflows may not suit highly dynamic environments.
* **Maintenance Requirement:**  
  System roles and user data must be updated periodically.
* **Theoretical Prototype:**  
  The current model is conceptual; full implementation requires additional resources and time.

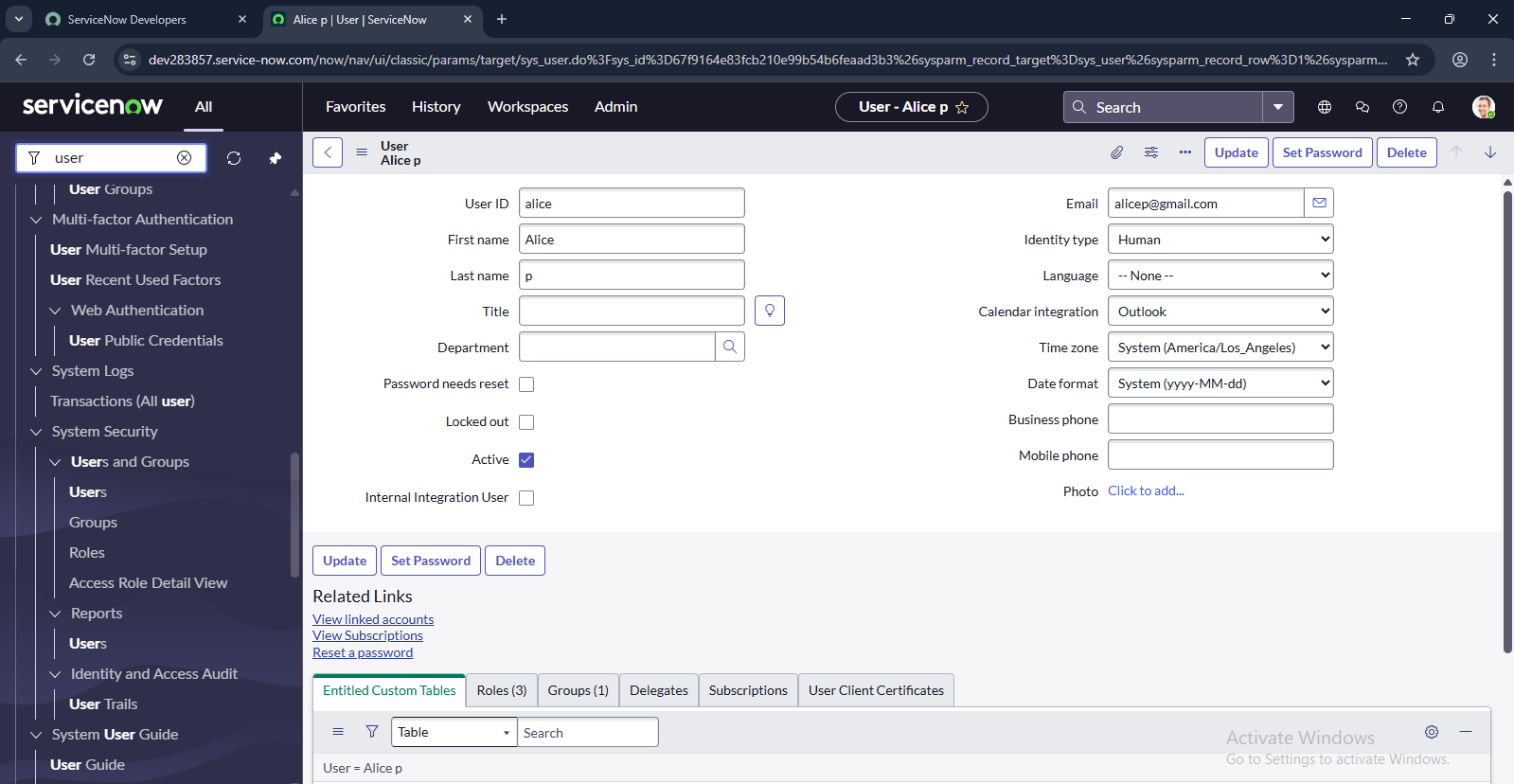
# **13.FUTURE ENHANCEMENTS**

The system can be expanded with advanced functionalities:

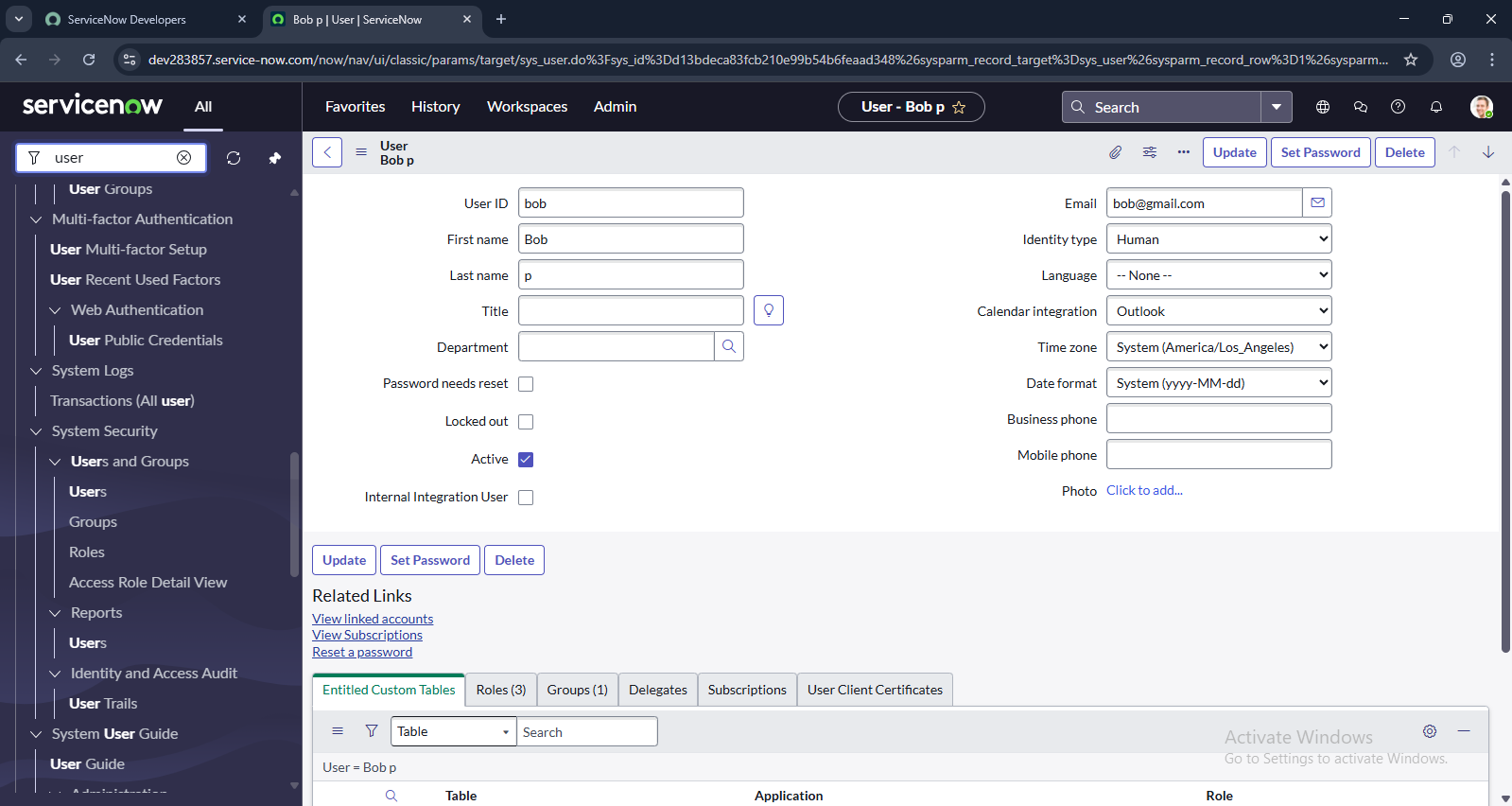
* **Artificial Intelligence Integration:**  
  AI algorithms can predict task delays and suggest workload distribution.
* **Cloud Integration:**  
  Deploying the system on cloud platforms for accessibility and scalability.
* **Mobile Application:**  
  Android/iOS versions to manage tasks on the go.
* **Data Analytics Dashboard:**  
  Visual analytics for tracking productivity trends.
* **Multi-level Role Hierarchies:**  
  Support for complex organizational structures with nested permissions.
* **Real-time Notifications:**  
  Push notifications for task updates and status changes.

**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

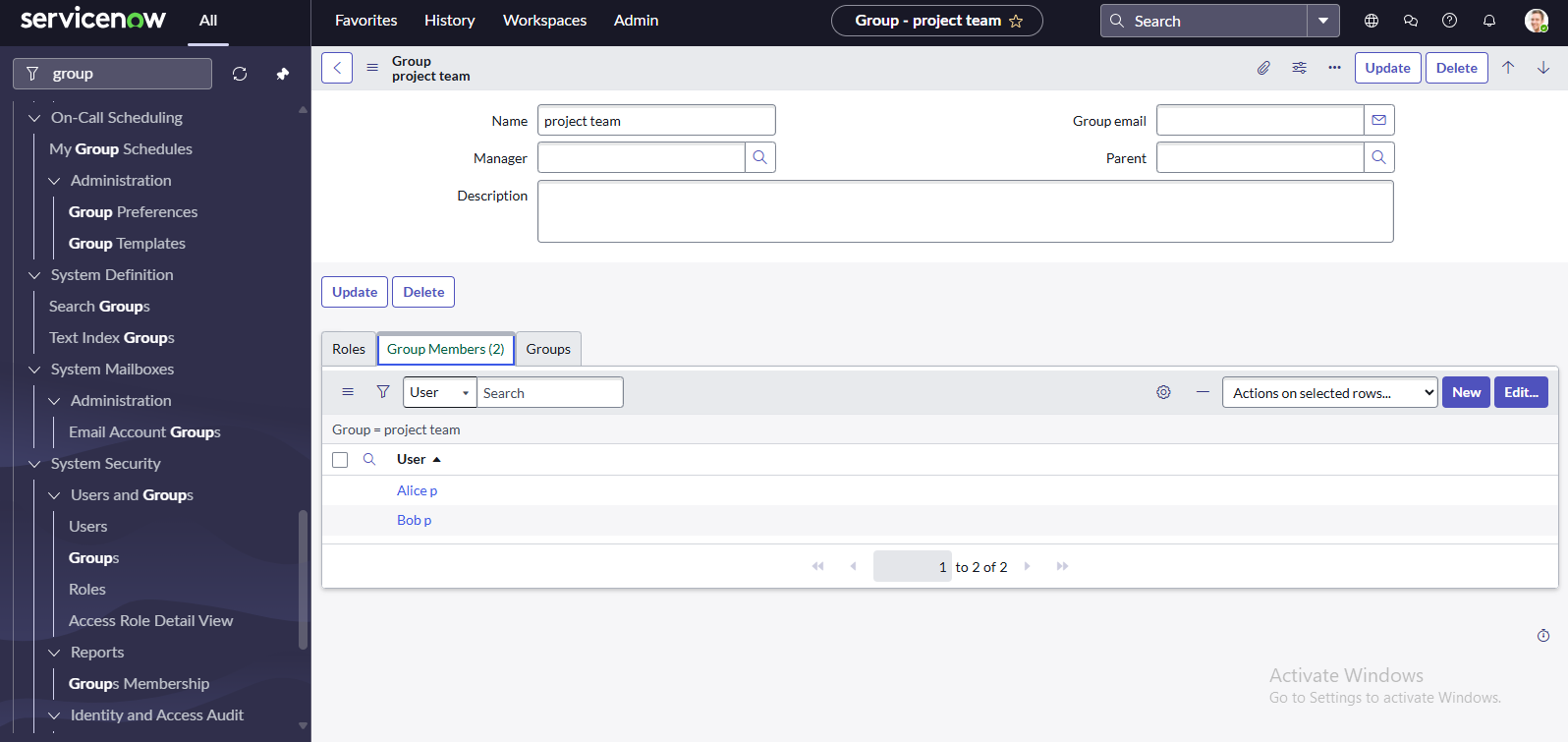


1. Create one more user
2. Create another user with the following details
3. Click on submit



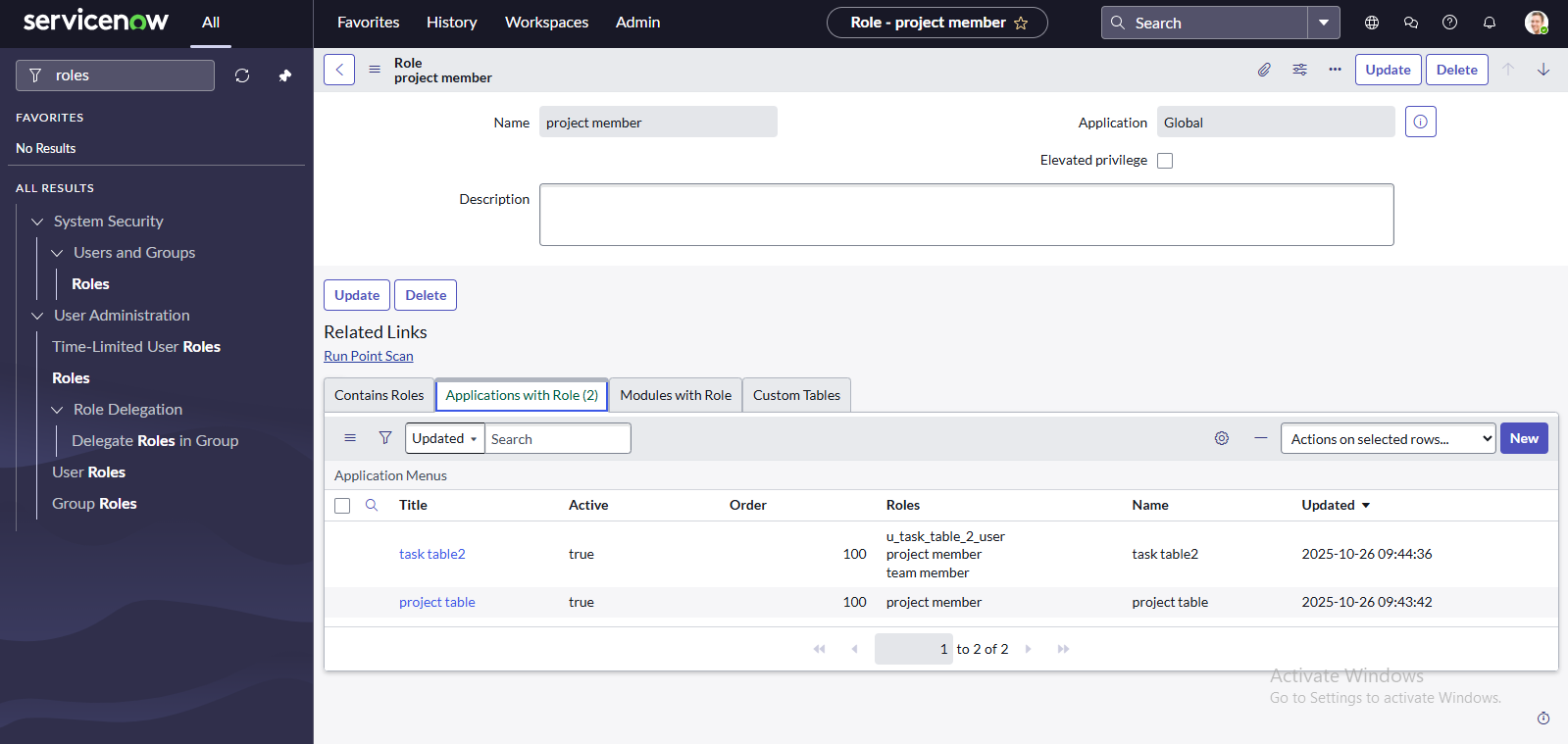
**Create Groups**

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**

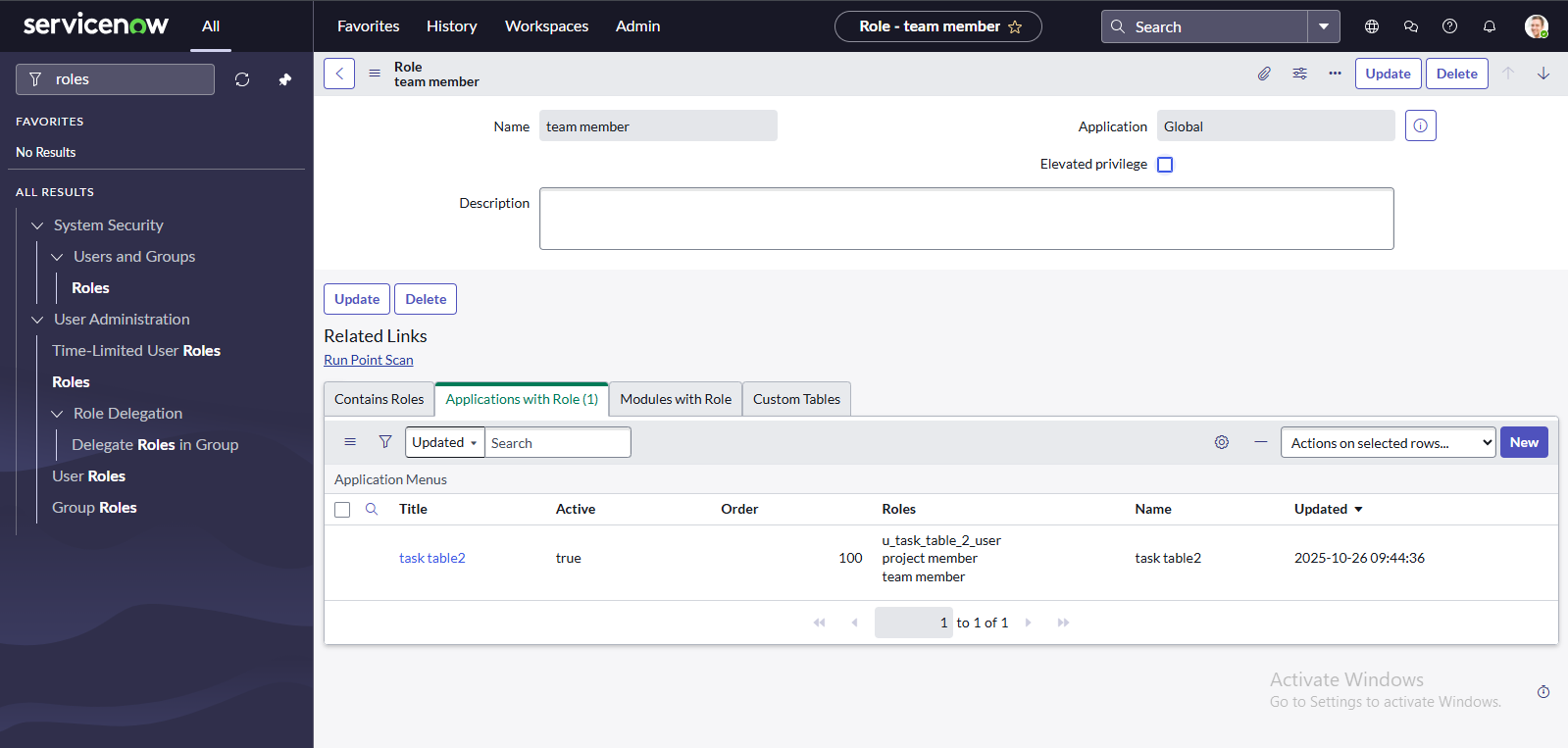
1. Open service now.
2. Click on All  >> search for roles
3. Select roles under system security
4. Click on new
5. Fill the following details to create a new role
6. Click on submit



**Create one more role:**

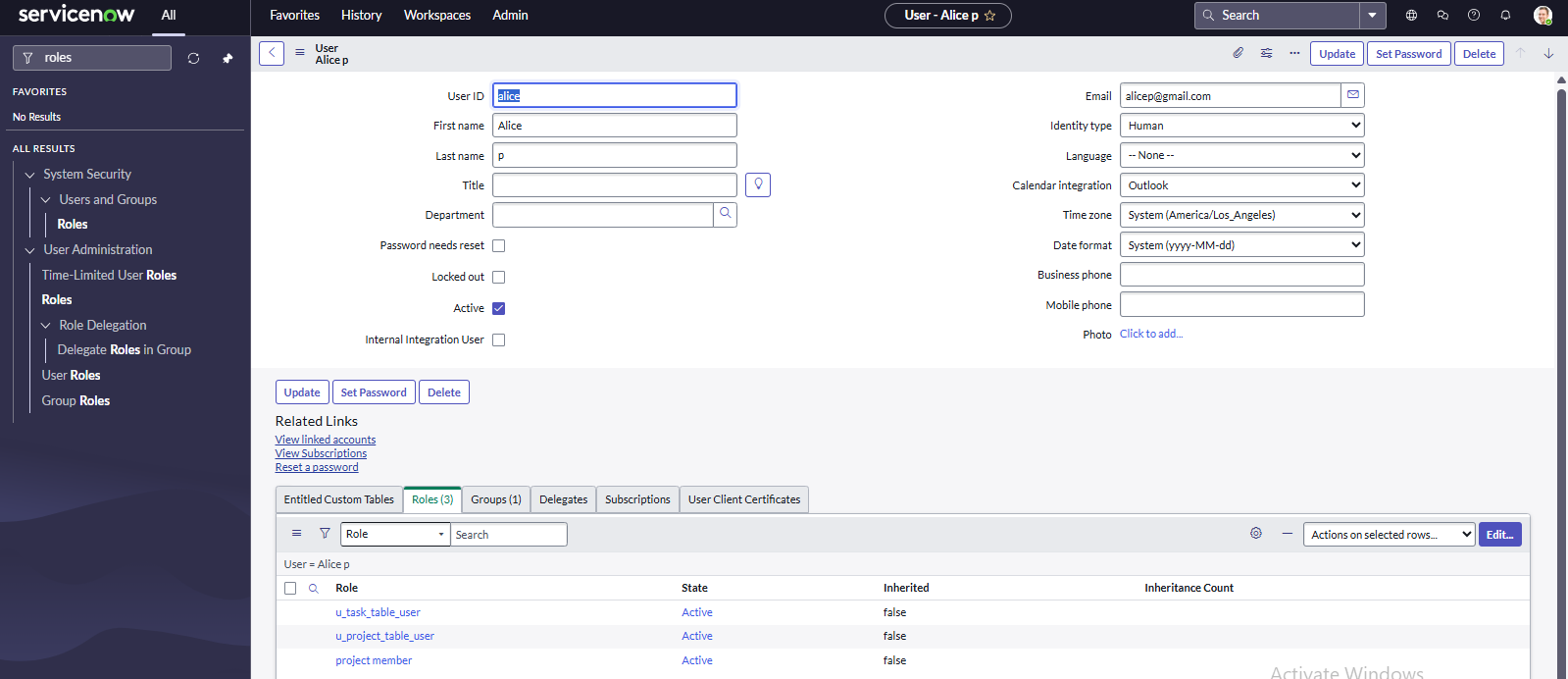
             7.Create another role with the following details

             8.Click on submit



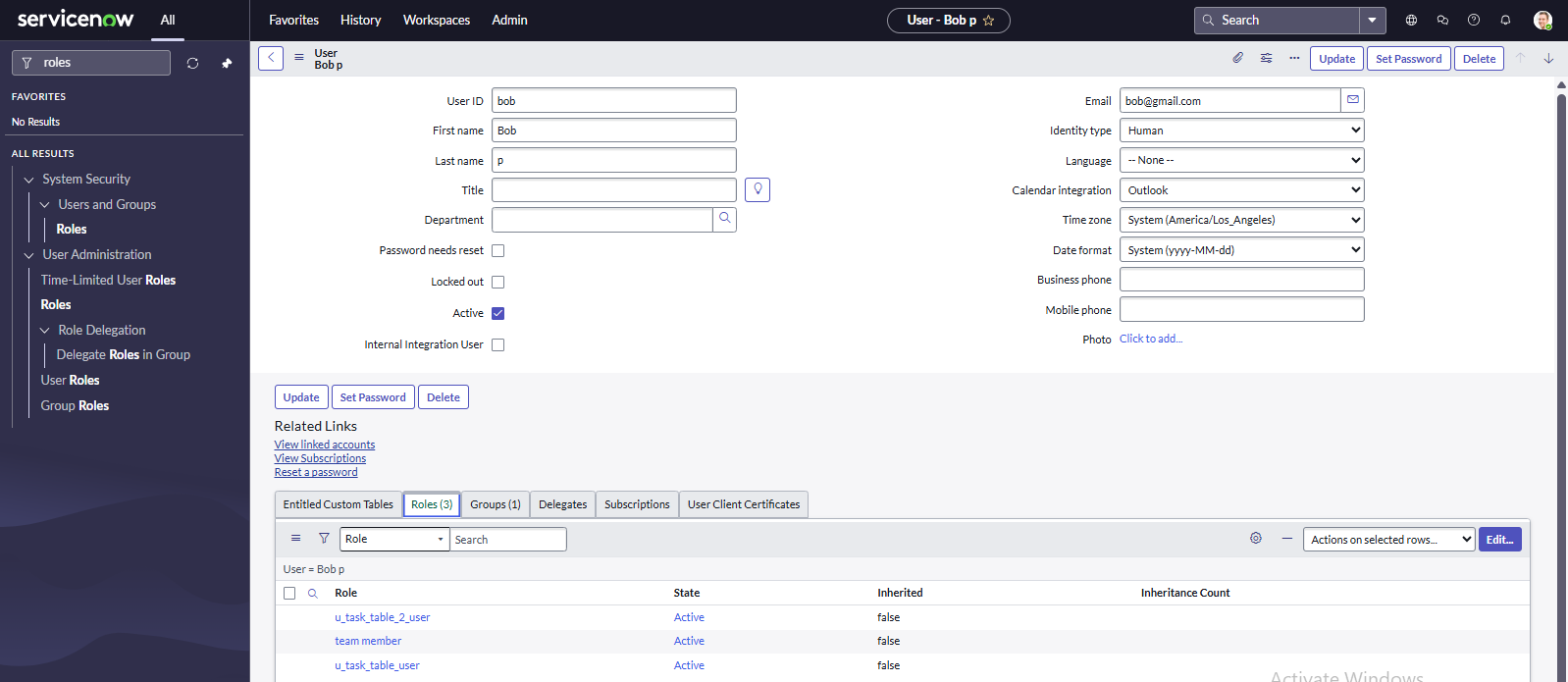
**Assign roles to alice user**

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.



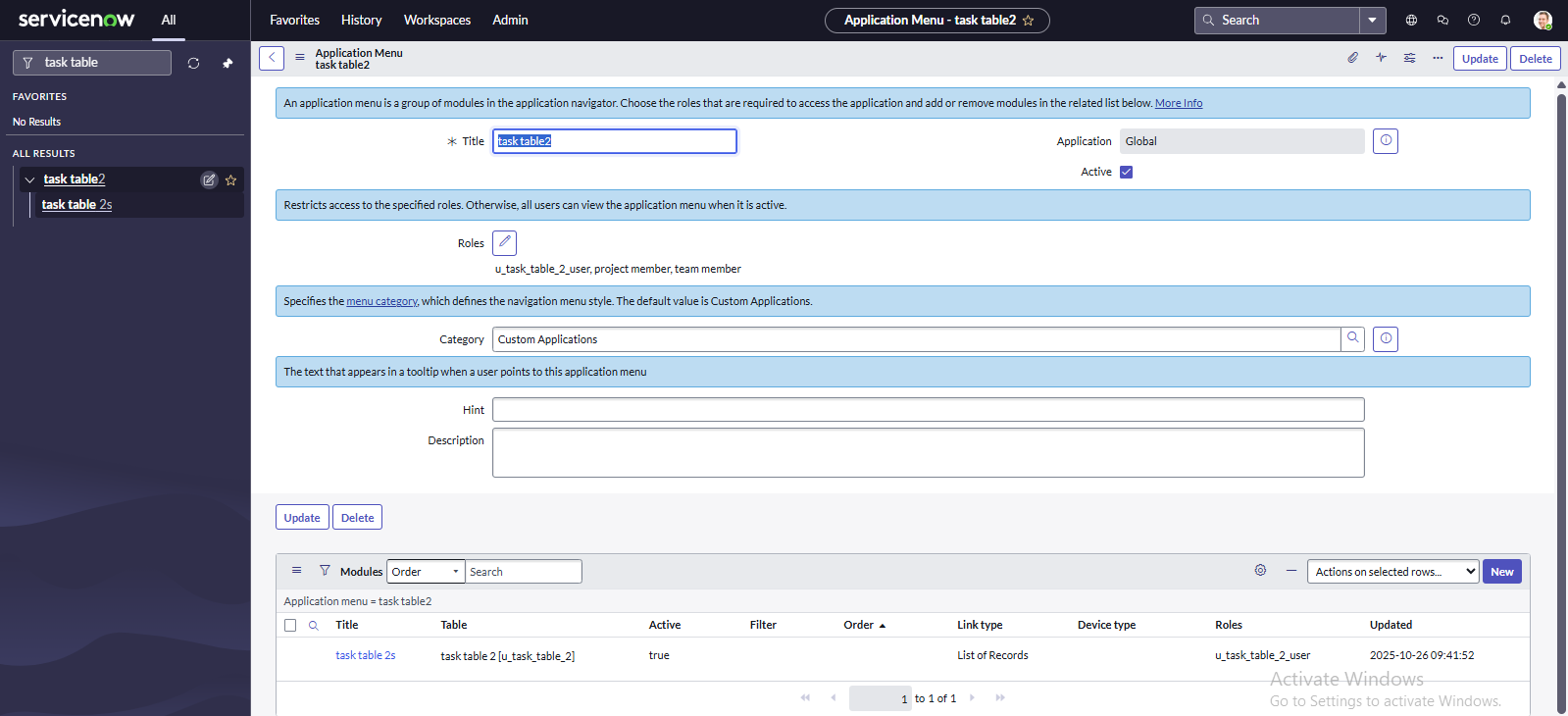
**Assign roles to bob user**

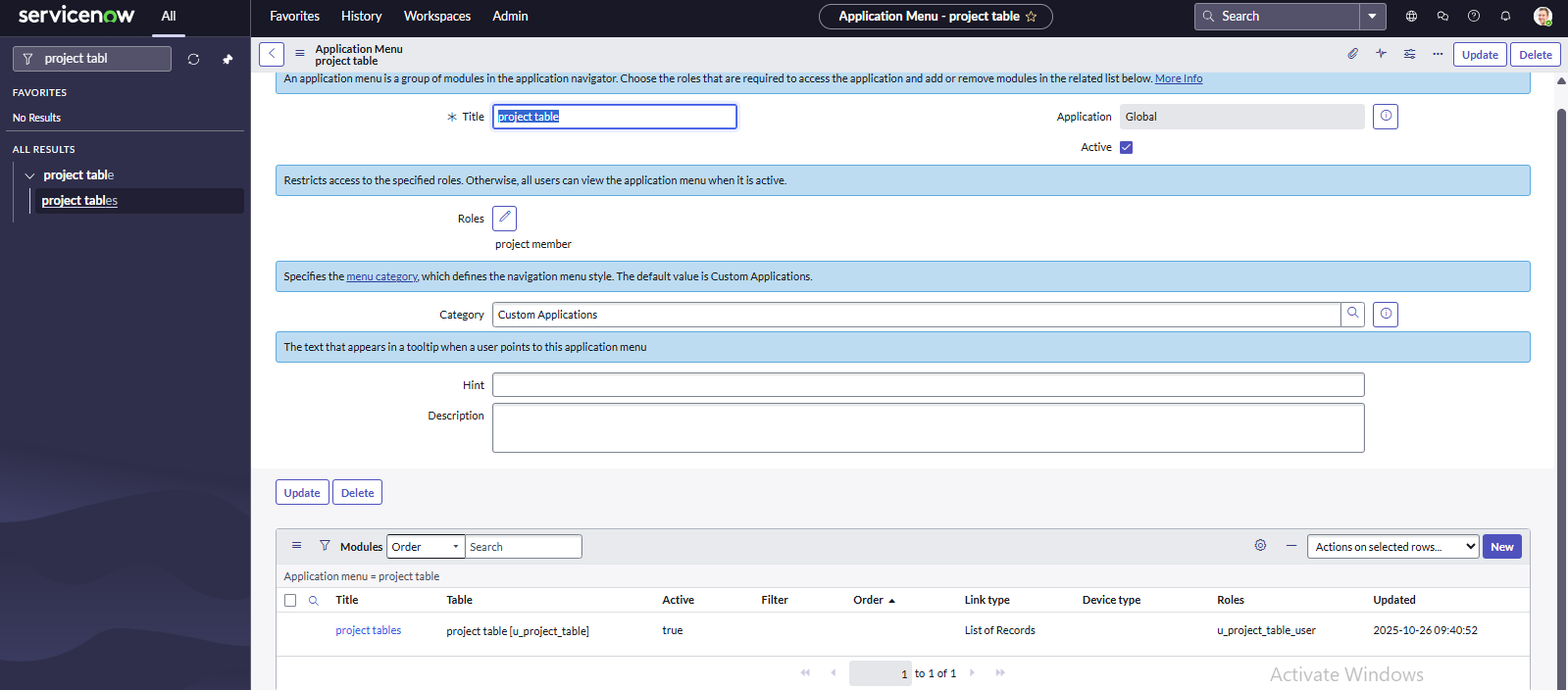
1. Open servicenow.Click on All  >> search for user
2. Select tables under system definition
3. Select the bob p user
4. Under team member
5. Click on edit
6. Select team member and give table role  and save
7. Click on profile icon Impersonate user to bob
8. We can see the task table2.



**Assign table access to application**

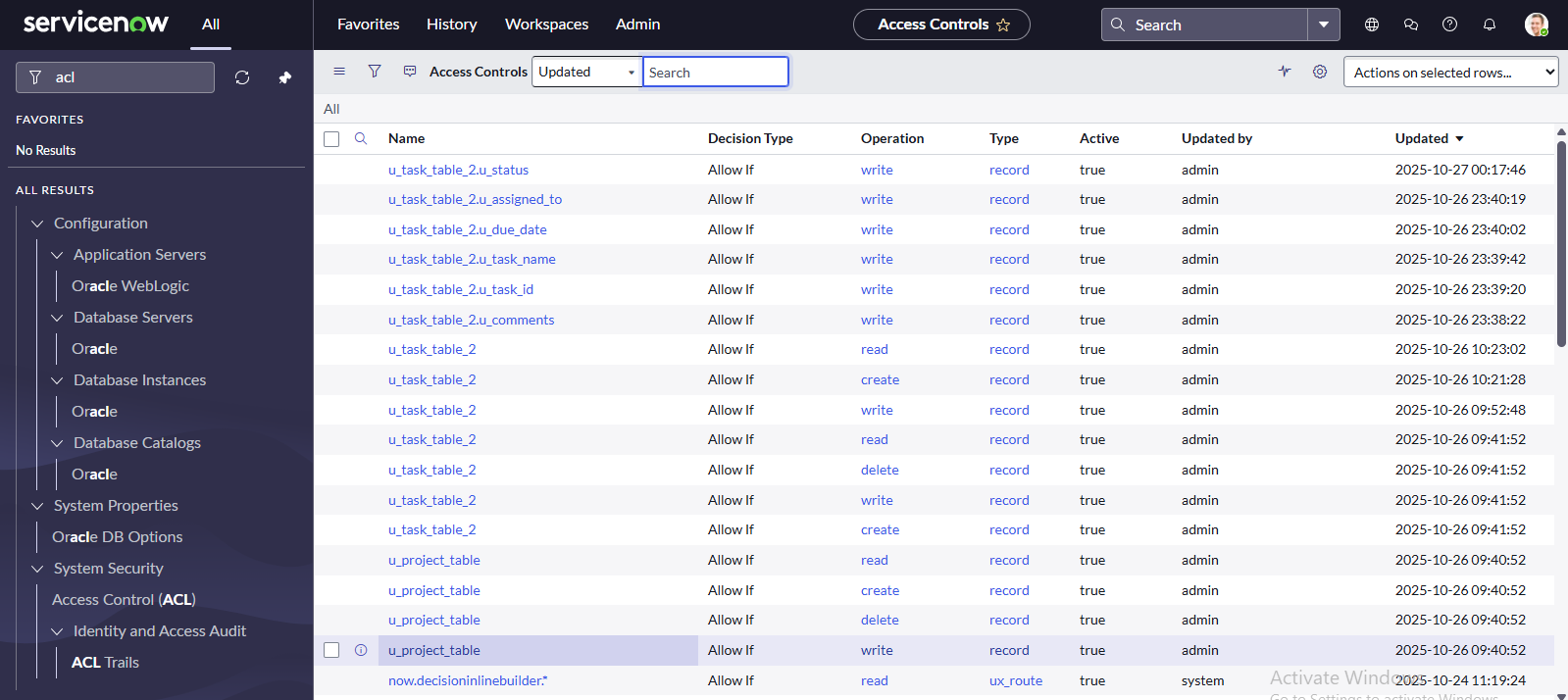
1. while creating a table it automatically create a application and module for that table
2. Go to application navigator search for search project table application
3. Click on edit module
4. Give project member  roles to that application
5. Search for task table2 and click on edit application.
6. Give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



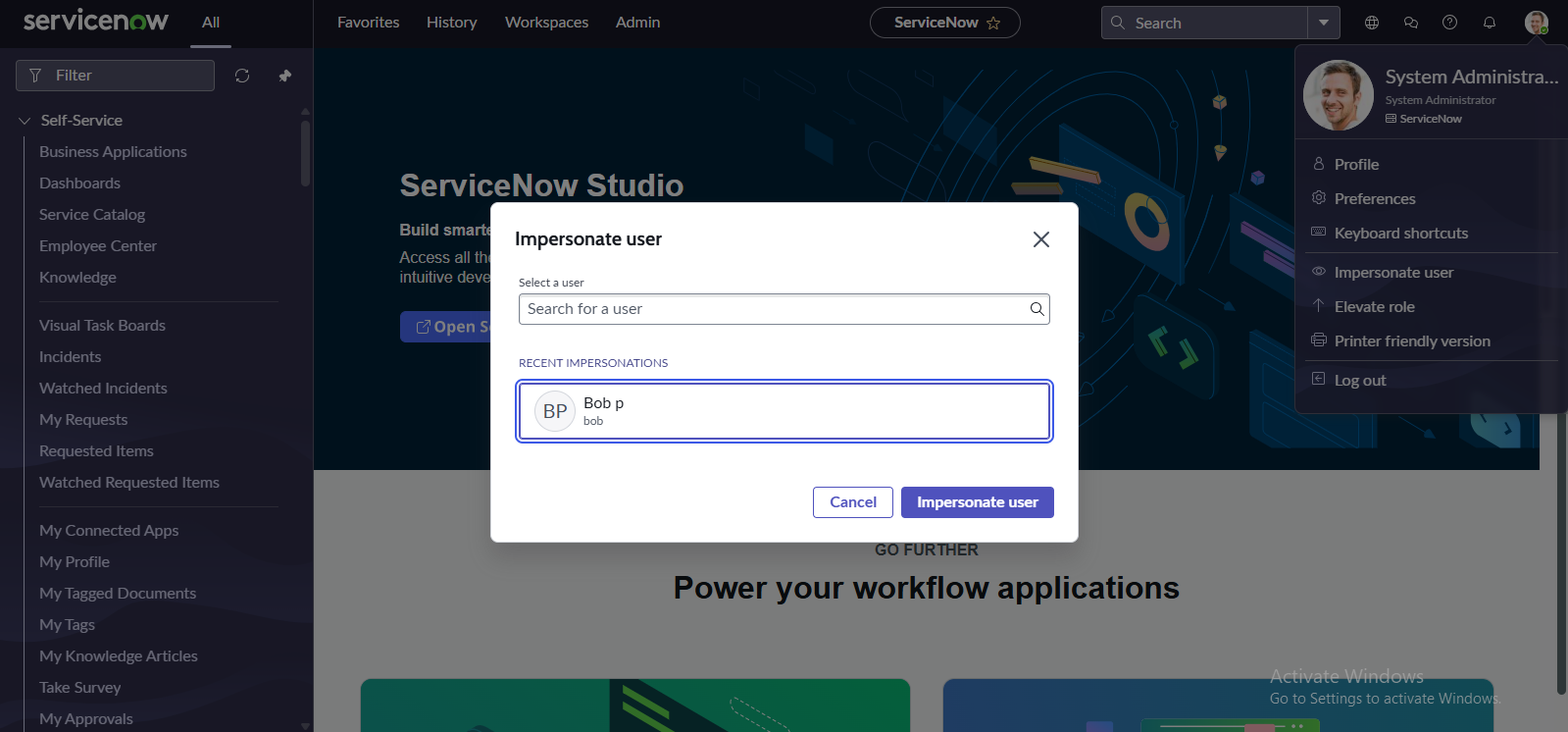


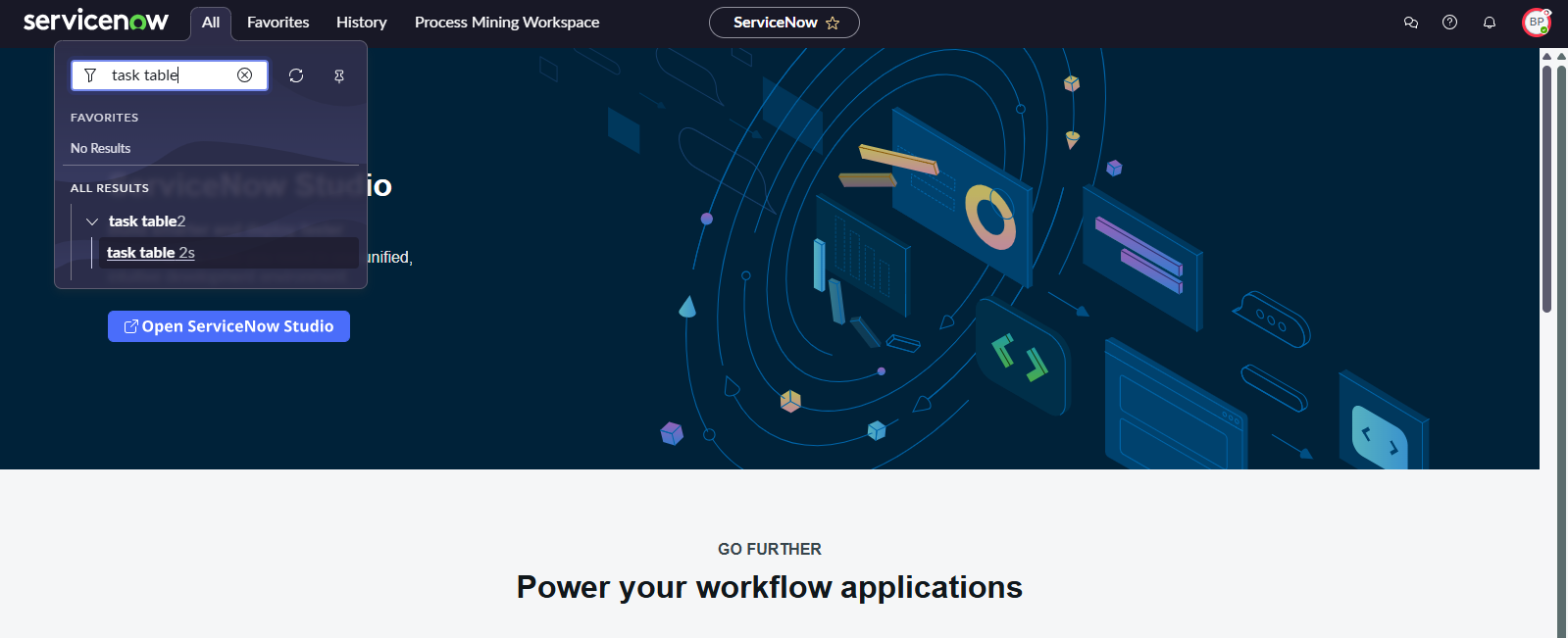
**Create ACL**

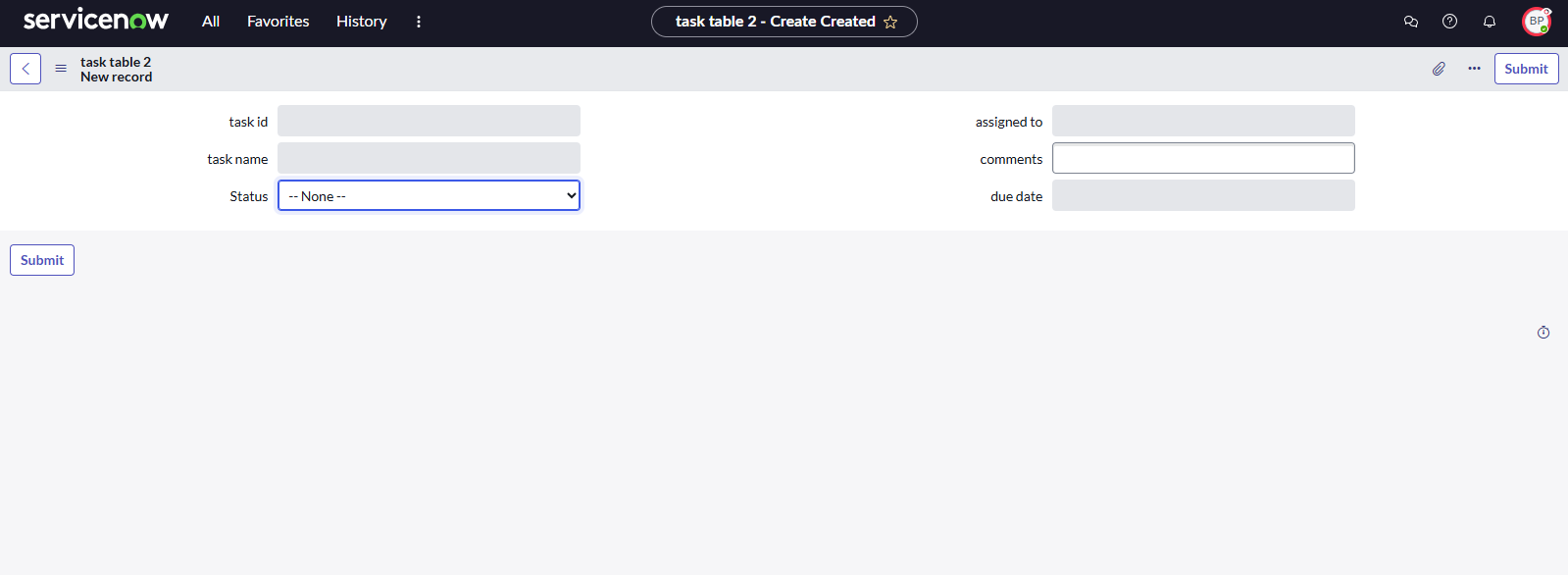
1. Open service now.
2. Click on All  >> search for ACL
3. Select Access Control(ACL) under system security
4. Click on elevate role
5. Click on new
6. Fill the following details to create a new ACL
7. Scroll down under requires role
8. Double click on insert a new row
9. Give task table and team member role
10. Click on submit
11. Similarly create 4 acl for the following fields



1. Click on profile on top right side
2. Click on impersonate user
3. Select  bob user
4. Go to all and select task table2 in the application menu bar
5. Comment and status fields are have the edit access

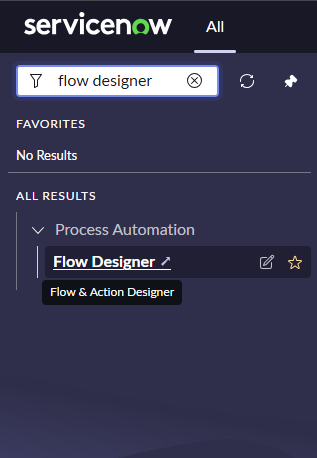


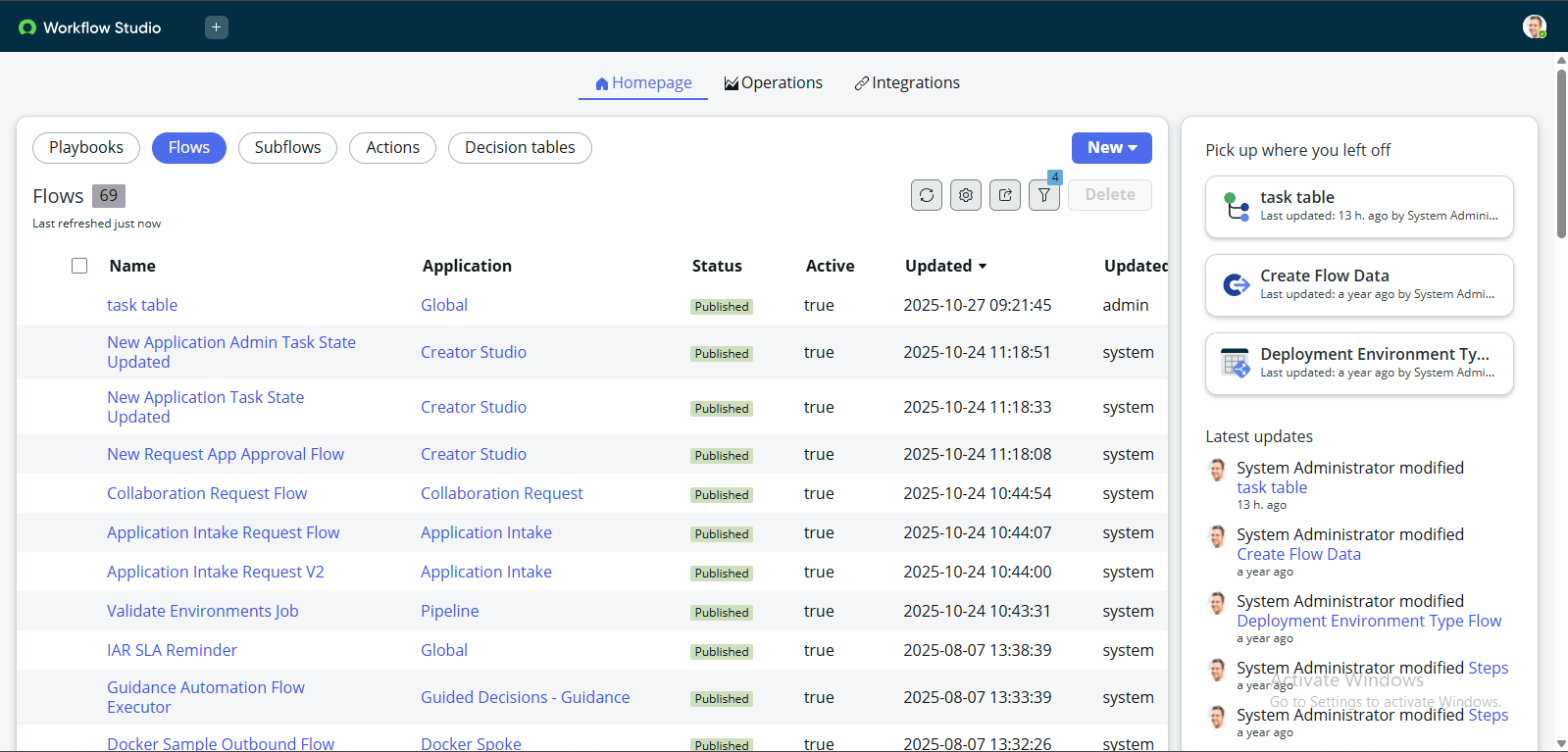


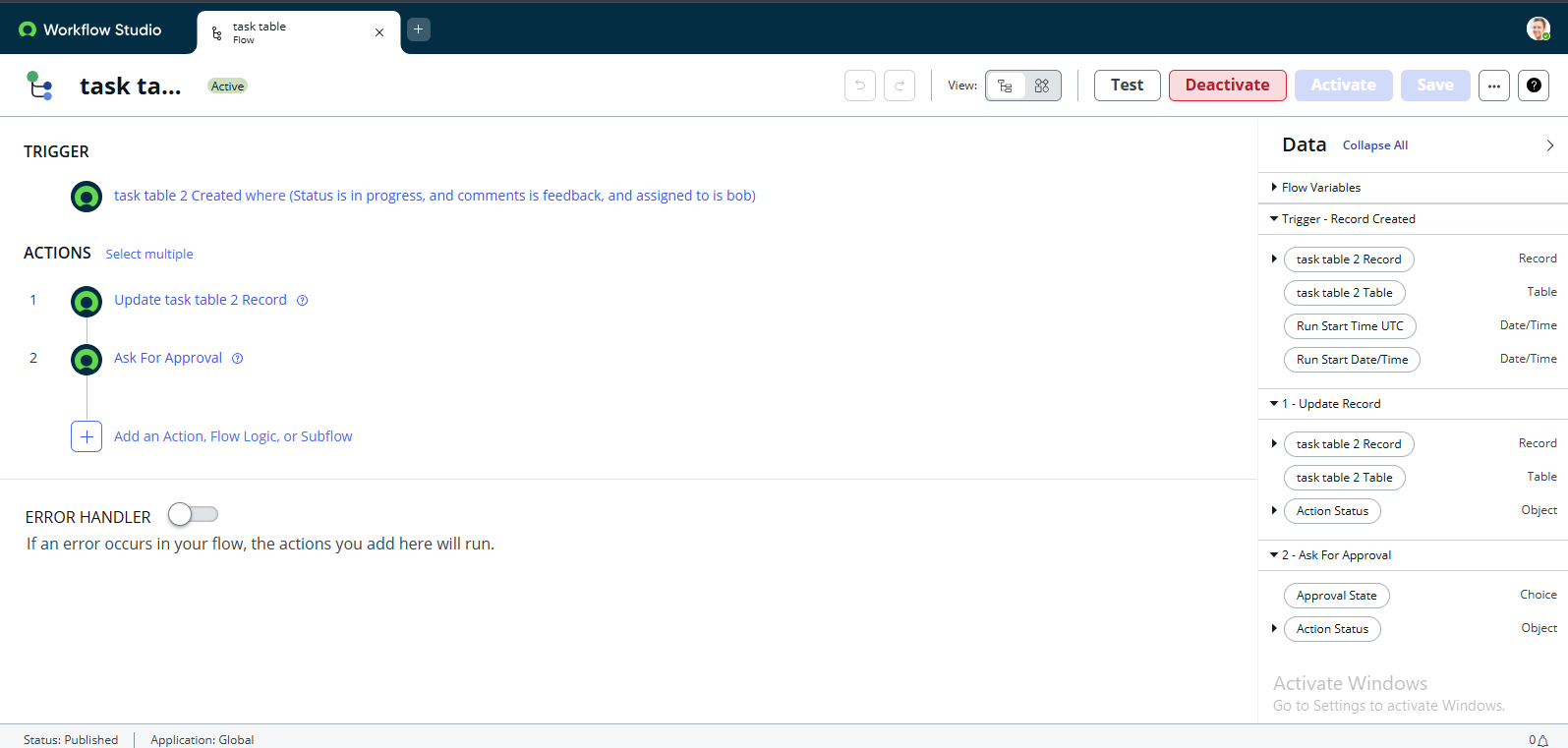
 

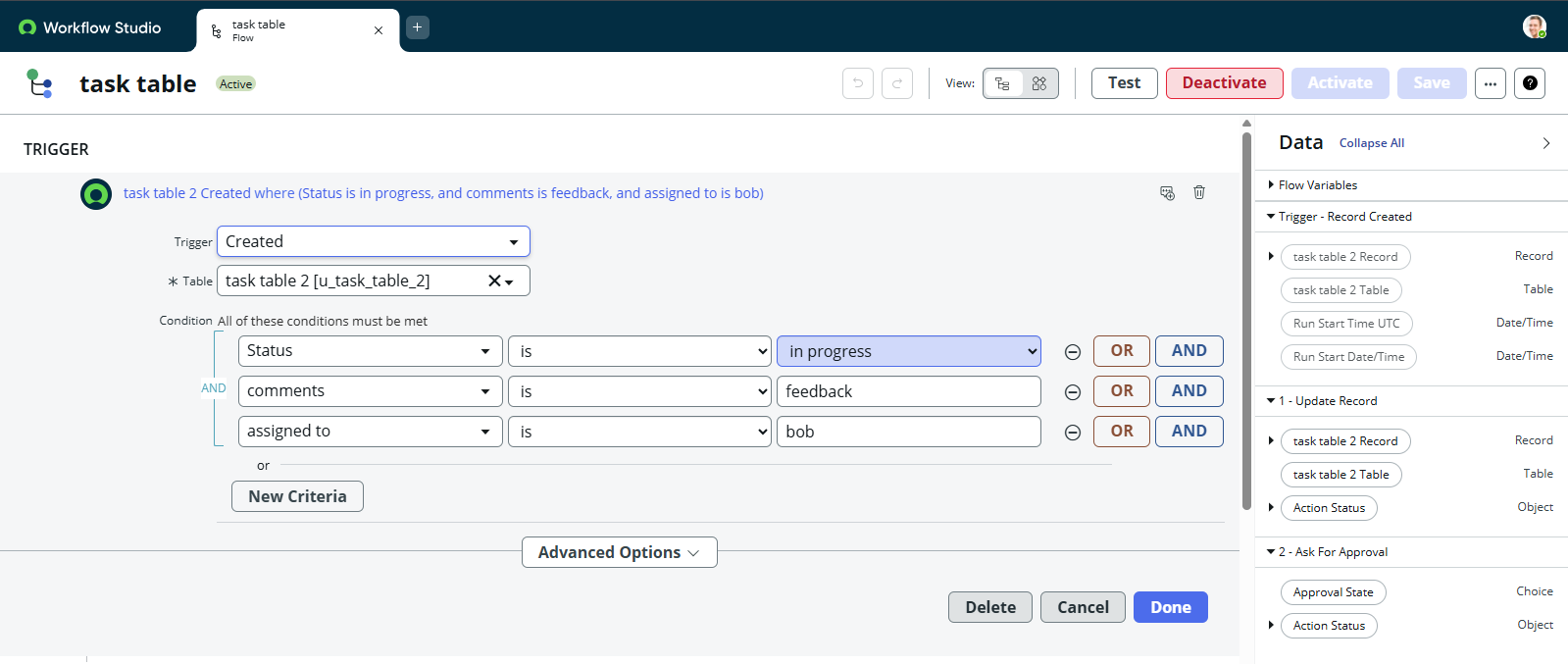
**Create a Flow to Assign operations ticket to group**

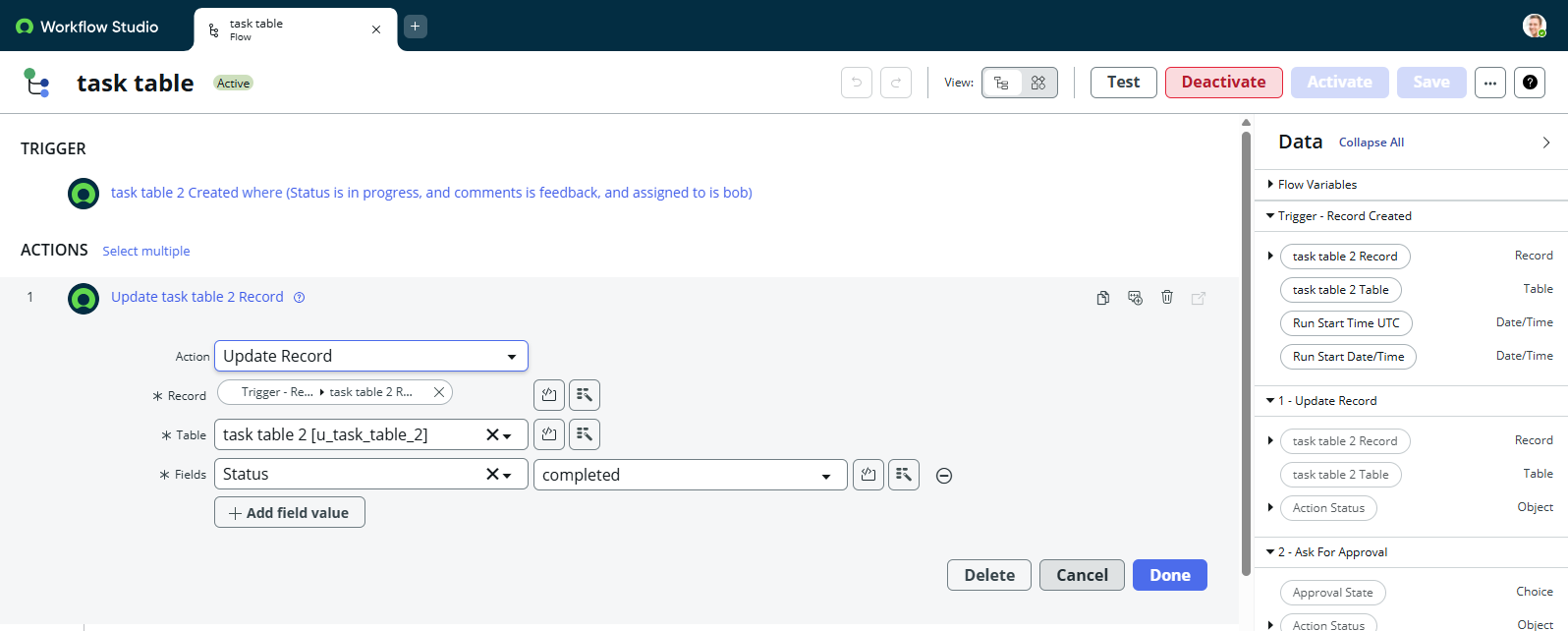
1. Open service now.
2. Click on All  >> search for Flow Designer
3. Click on Flow Designer under Process Automation.
4. After opening Flow Designer Click on new and select Flow.
5. Under Flow properties Give Flow Name as “ task table”.
6. Application should be Global.
7. Click build flow.

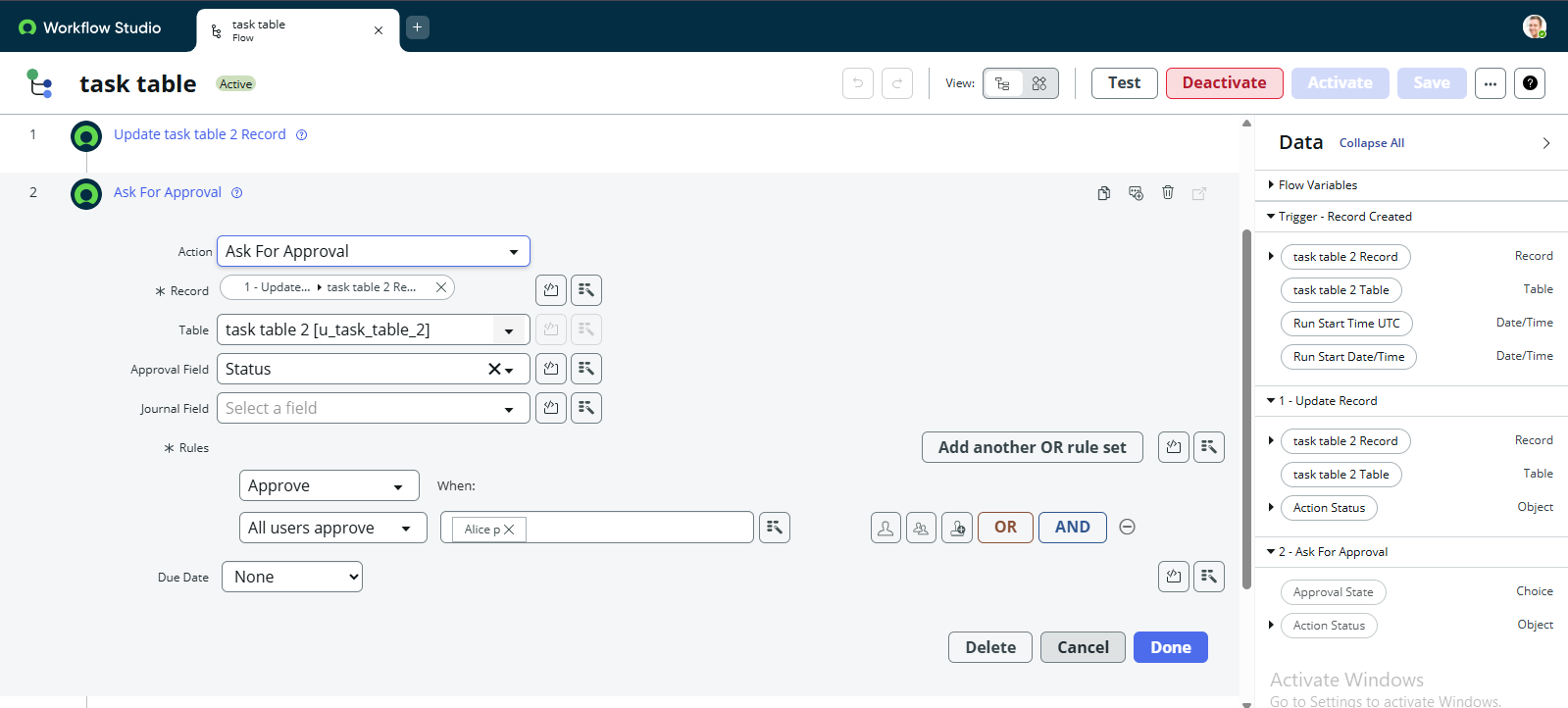


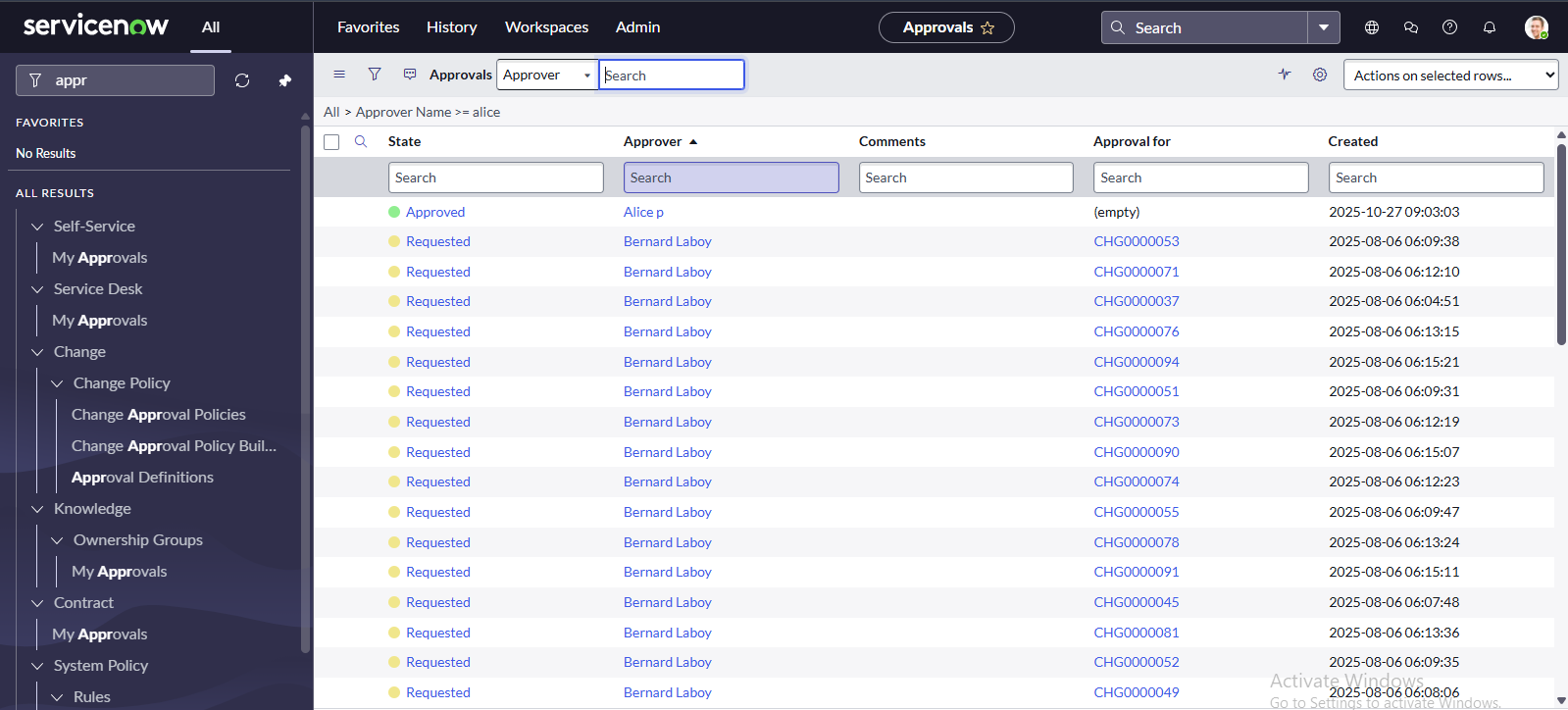


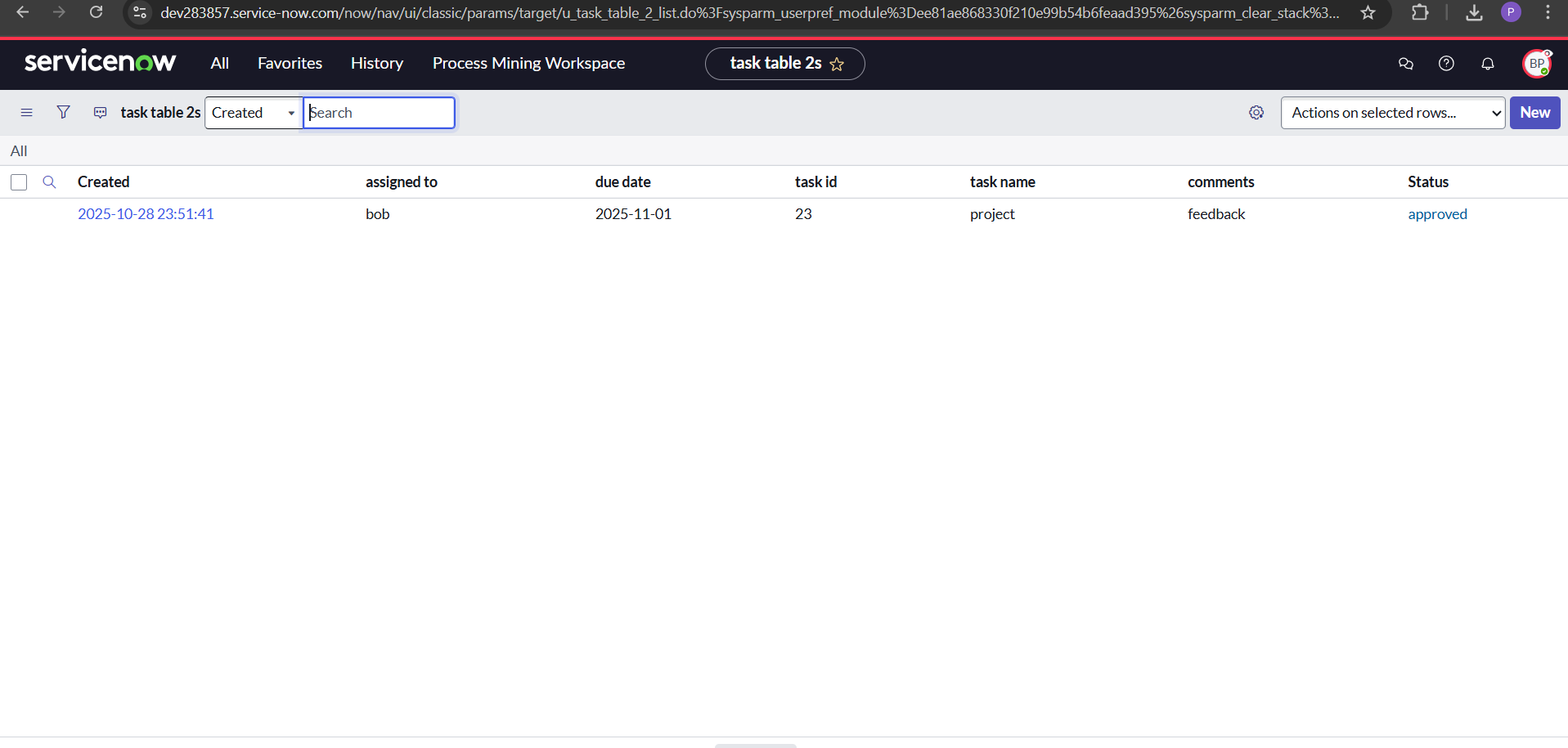


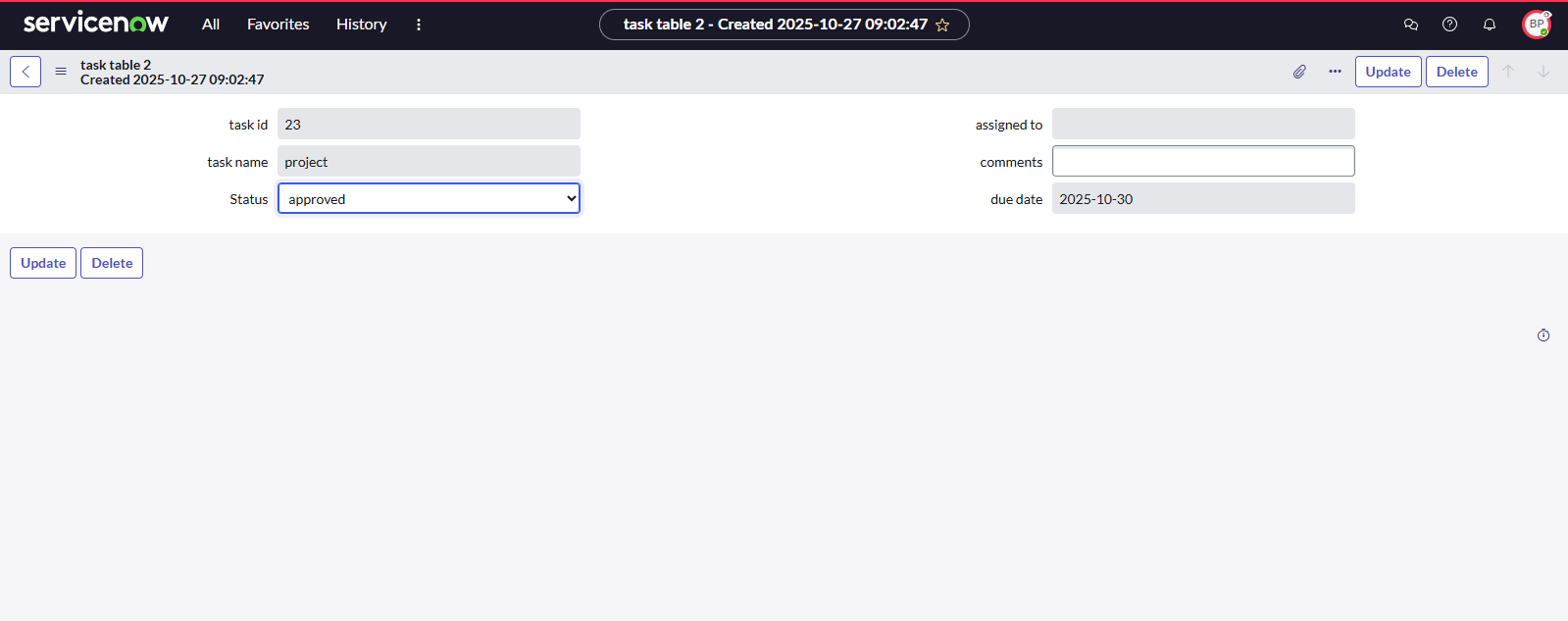












# **14.CONCLUSION**

# This project demonstrates the necessity of structured user, group, and role management integrated with access control and automated workflows. Through the implementation of **Role-Based Access Control (RBAC),** the system ensures that users perform only the operations permitted by their role, maintaining security and accountability.Automated workflows streamline task management, reducing manual dependency and enhancing team coordination. The modular design ensures scalability and adaptability, making the system suitable for various organizational sizes.In conclusion, “Optimizing User, Group, and Role Management with Access Control and Workflows” provides a theoretical foundation for developing secure, efficient, and transparent project management systems. It addresses core challenges such as unauthorized access, unclear responsibilities, and workflow inefficiencies — paving the way for intelligent, access-controlled collaboration environments in the future.