Project Documentation

1.Introduction

- **Project Name**: Optimizing User, Group, and Role Management with Access Control and Workflows.

2. Objective

- **Purpose:** The purpose of this project is to build a secure and efficient system for managing users, groups, and roles with proper access control. Its goals are to simplify user onboarding and offboarding, automate workflows, improve accountability with audit logs, ensure compliance with standards, and enhance overall security and efficiency.
- **Features:**
    - **User Management** – Add, update, and remove users with self-service options like password reset.
    - **Group Management** – Create groups (department or project-based) and assign permissions to them.
    - **Role Management** – Define standardized roles (Admin, Manager, User, Guest) with least-privilege access.
    - **Access Control** – Enforce authentication (username, password, MFA) and ensure secure access rights.
    - **Workflow Automation** – Automate access requests, approvals, and provisioning/de-provisioning processes.
    - **Security & Compliance** – Maintain audit logs, support periodic access reviews, and align with standards (ISO, GDPR, HIPAA).
    - **Progress Monitoring** – Provide dashboards for task tracking, approvals, and real-time visibility.

3. System Architecture

The architecture of this project is designed around three main layers: **User Layer, Application Layer, and Data Layer**.

1. **User Layer (Frontend Access)**
    - End users interact with the system through portals or applications.
    - Project Manager (Alice) and Team Members (Bob) log in with secured authentication (username, password, MFA, SSO).
2. **Application Layer (Business Logic & Workflows)**
    - **Identity Management System** (Azure AD, Okta, or Keycloak) handles user, group,

and role management.
    - **Workflow Engine** (ServiceNow, Camunda, or custom workflows) automates access requests, approvals, task assignments, and provisioning/de-provisioning.
    - **Access Control** enforces RBAC (Role-Based Access Control) and ABAC (Attribute-Based Access Control, if needed).
3. **Data Layer (Storage & Logging)**
    - User, group, and role data stored in a relational or NoSQL database (MySQL, PostgreSQL, MongoDB).
    - Audit logs and compliance reports stored in monitoring systems (Splunk, ELK Stack).
4. **Security Layer (Across All Layers)**
    - Multi-Factor Authentication (MFA), encryption, and secure APIs ensure protection of data and workflows.
    - Regular access reviews maintain compliance with ISO, GDPR, and HIPAA standards.

4. Setup Instructions

1. Prerequisites

- ServiceNow Personal Developer Instance (PDI) account from developer.servicenow.com.
- A stable internet connection.
- Web browser (Google Chrome or Firefox recommended).

2. Environment Setup

- Sign up or log in to the ServiceNow Developer Portal.
- Request a Personal Developer Instance (PDI).
- Once assigned, log in to your instance using the provided URL (e.g., `https://devxxxx.service-now.com`).
- Use the default admin credentials shared with your instance.

3. Project Setup

- Navigate to System Applications → Studio in your PDI.
- Create a new application or open the existing one.
- Configure Tables, Roles, and Workflows as per project requirements.
- Test features such as user management, group assignment, and RBAC inside

your PDI.
- Save and publish the application to make it accessible within your PDI.

5. Folder Structure Description

The project is organized into logical folder to maintain clarity and ease of access, even though all configurations were performed within the ServiceNow Personal Developer Instance (PDI).

- **ServiceNow_Configuration/** – Holds all ServiceNow artifacts organized by type:
  - **Tables/** – Includes custom tables like `u_task_table2` and `u_user_roles` that store project and task data.
  - **Roles/** – Contains roles created for access control, such as Admin, Team Member, and Project Member.
  - **Workflows/** – Stores automated workflows for access requests, approvals, and onboarding processes.
  - **ACLs/** – Contains access control rules for tables and fields, enforcing role-based permissions.

6. Running the Application

The project application is deployed and executed within the ServiceNow Personal Developer Instance (PDI). To run the application:

1. **Login to PDI:** Open your PDI URL (e.g., `https://devxxxx.service-now.com`) and log in with your admin credentials.
2. **Access the Application:** Navigate to System Applications → Studio, and open the application created for the project.
3. **Interact with Tables:**
   - Create or view records in `Project Table` and `Task Table2`.
   - Assign tasks, manage users, and groups according to role-based access.
4. **Trigger Workflows:**
   - Create a record that meets the workflow conditions (e.g., status = “In Progress” and assigned to Bob).
   - ServiceNow automatically triggers the approval and update processes defined in Flow Designer.
5. **Monitor Approvals:**
   - The Project Manager (Alice) can review and approve tasks via My Approvals.
   - Task status is updated automatically after approval.
6. **Review Results:**
   - Check dashboards or table records to ensure workflows, roles, and access

control function as expected.

7. API Documentation

Endpoint	Description	Request Parameters	Example Response
/api/now/table/u_user	Retrieve all users	sysparm_limit (optional)	{ "result": [{ "name": "Alice", "role": "Project Manager" }, ...] }
/api/now/table/u_user	Create a new user	name, email, role	{ "result": { "sys_id": "12345", "name": "Bob", "role": "Team Member" } }
/api/now/table/u_task_table2	Get all tasks	assigned_to, status	{ "result": [{ "title": "Task 1", "status": "In Progress" }, ...] }
/api/now/table/u_task_table2	Create a new task	title, description, assigned_to, status	{ "result": { "sys_id": "67890", "title": "Task 1", "status": "In Progress" } }
/api/now/table/u_task_table2	Update a task	sys_id, status, comments	{ "result": { "sys_id": "67890", "status": "Completed" } }
/api/now/table/u_task_table2	Delete a task	sys_id	{ "result": "Task deleted successfully" }

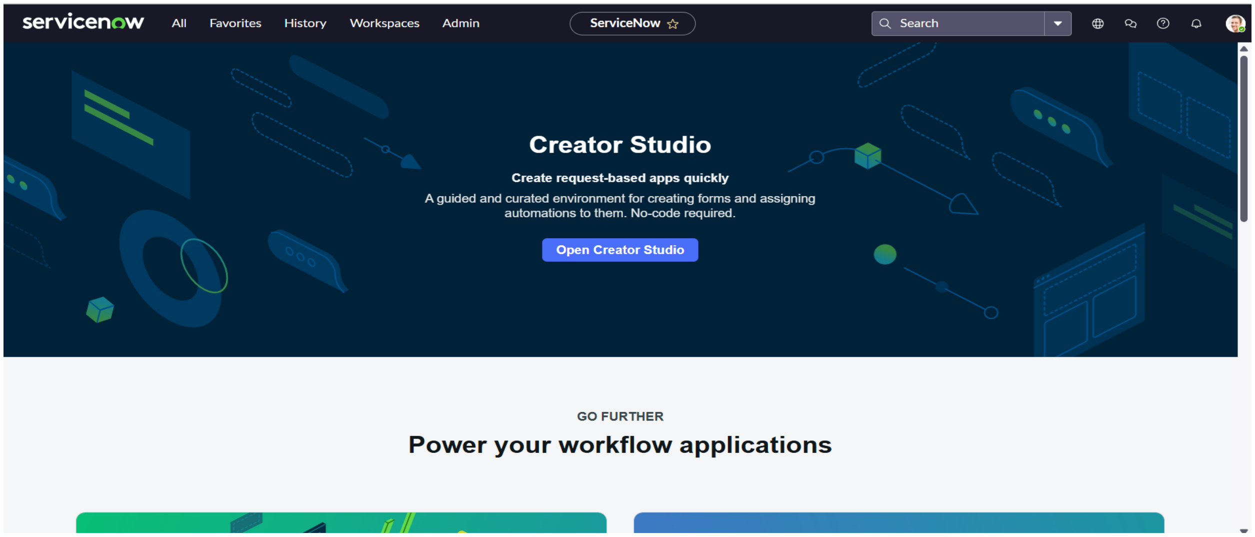
8. Authentication and Authorization

In this project, authentication and authorization are managed using ServiceNow's built-in security mechanisms within the Personal Developer Instance (PDI):

1. **Authentication**
    - Users log in using username and password.
    - Multi-Factor Authentication (MFA) can be enabled to provide an additional layer of security.
    - Sessions are maintained by ServiceNow, which generates a session token after successful login. This token is used to authenticate API requests and ensure secure access while the session is active.
2. **Authorization**
    - The system uses Role-Based Access Control (RBAC) to manage permissions. Each user is assigned specific roles such as Admin, Project Manager, Team Member, or Guest.
    - Access Control Rules (ACLs) enforce which tables, fields, and records each role can read, write, or delete.
    - Workflows and approvals respect the user's role. For example, a Team Member can update only their assigned tasks, while the Project Manager can assign tasks

and approve updates.

9. User Interfaces

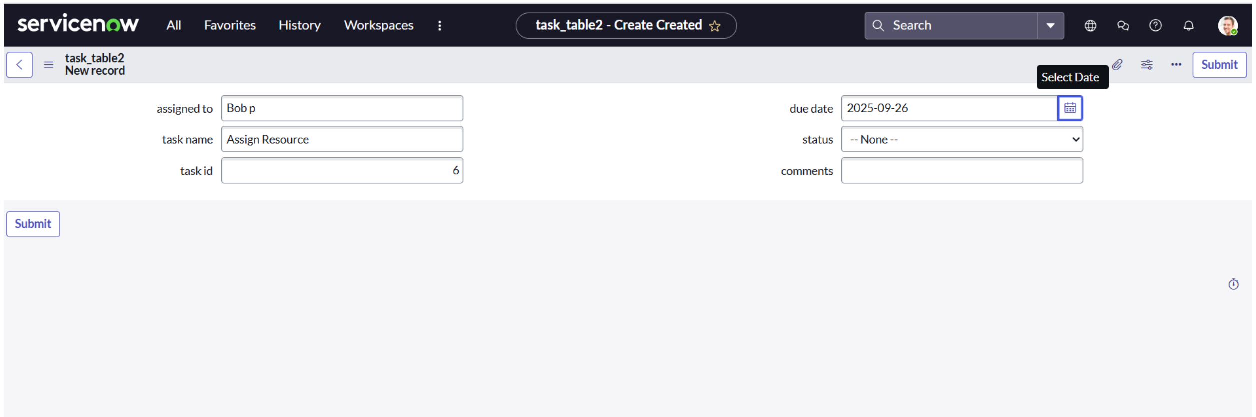


Creator Studio

Create request-based apps quickly

A guided and curated environment for creating forms and assigning automations to them. No-code required.

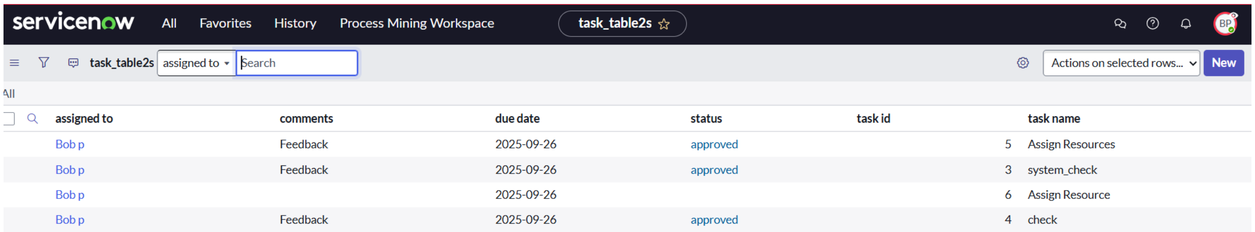
Power your workflow applications



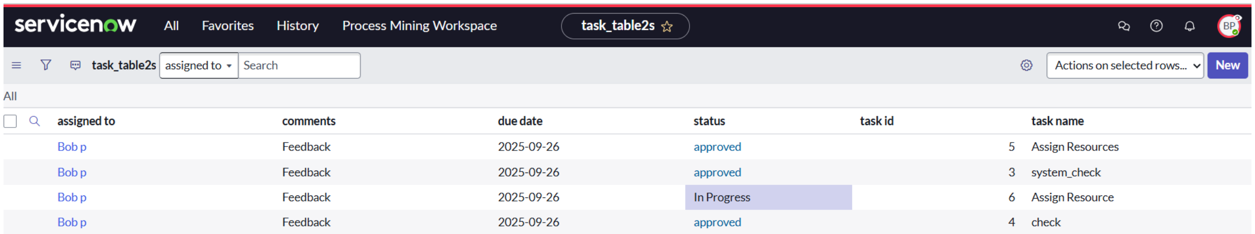
task_table2
New record

assigned to	Bob p	due date	2025-09-26
task name	Assign Resource	status	-- None --
task id	6	comments	

- ○ **Bob p**



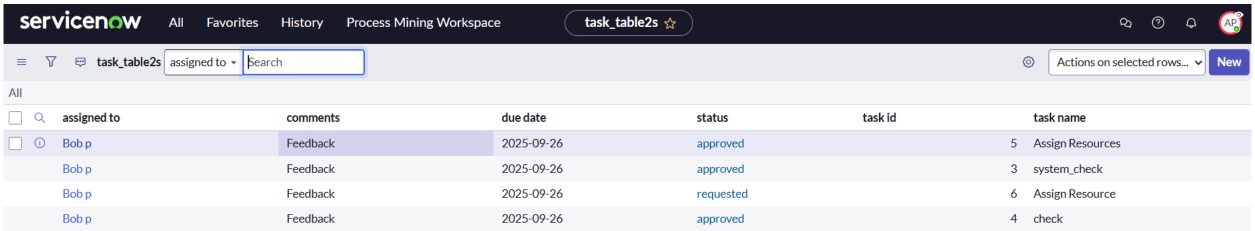
assigned to	comments	due date	status	task id	task name
Bob p	Feedback	2025-09-26	approved	5	Assign Resources
Bob p	Feedback	2025-09-26	approved	3	system_check
Bob p		2025-09-26		6	Assign Resource
Bob p	Feedback	2025-09-26	approved	4	check



All

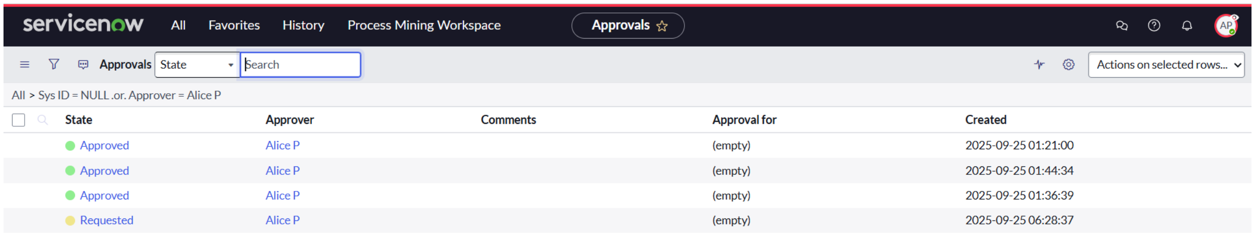
assigned to	comments	due date	status	task id	task name
Bob p	Feedback	2025-09-26	approved	5	Assign Resources
Bob p	Feedback	2025-09-26	approved	3	system_check
Bob p	Feedback	2025-09-26	In Progress	6	Assign Resource
Bob p	Feedback	2025-09-26	approved	4	check

○ Alice p



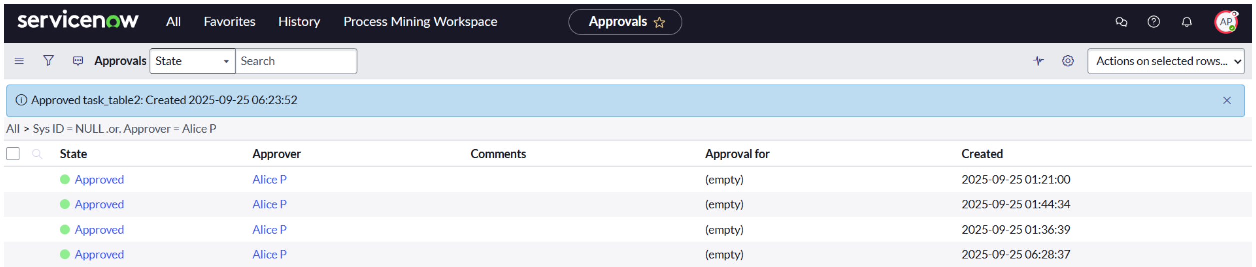
All

assigned to	comments	due date	status	task id	task name
Bob p	Feedback	2025-09-26	approved	5	Assign Resources
Bob p	Feedback	2025-09-26	approved	3	system_check
Bob p	Feedback	2025-09-26	requested	6	Assign Resource
Bob p	Feedback	2025-09-26	approved	4	check



All > Sys ID = NULL .or. Approver = Alice P

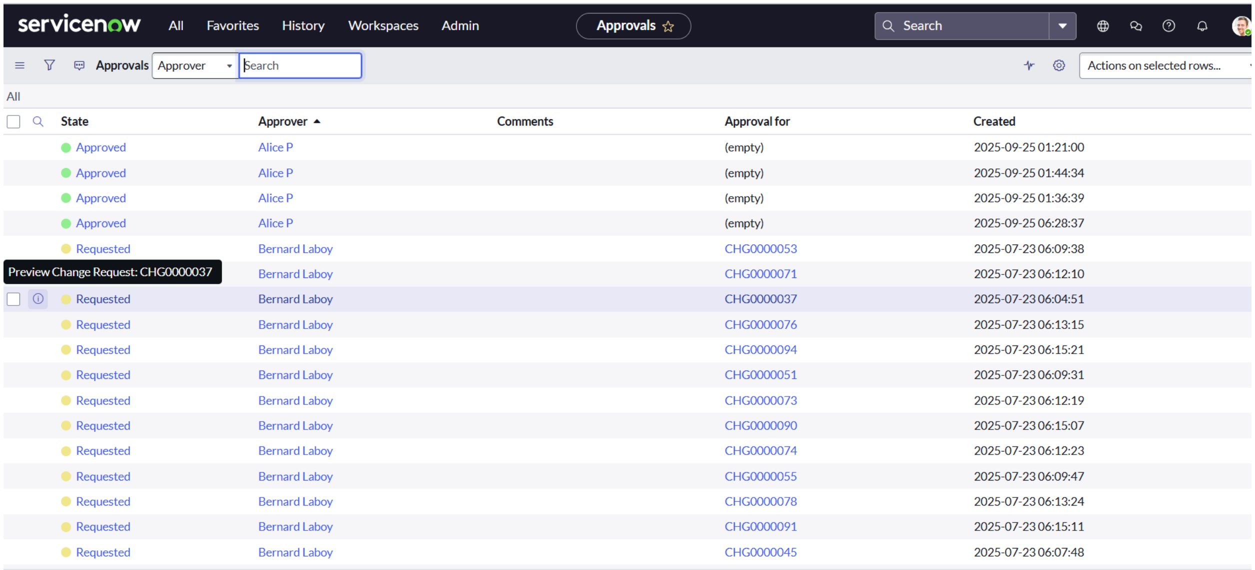
State	Approver	Comments	Approval for	Created
● Approved	Alice P		(empty)	2025-09-25 01:21:00
● Approved	Alice P		(empty)	2025-09-25 01:44:34
● Approved	Alice P		(empty)	2025-09-25 01:36:39
● Requested	Alice P		(empty)	2025-09-25 06:28:37



Approved task_table2: Created 2025-09-25 06:23:52

All > Sys ID = NULL .or. Approver = Alice P

State	Approver	Comments	Approval for	Created
● Approved	Alice P		(empty)	2025-09-25 01:21:00
● Approved	Alice P		(empty)	2025-09-25 01:44:34
● Approved	Alice P		(empty)	2025-09-25 01:36:39
● Approved	Alice P		(empty)	2025-09-25 06:28:37



State	Approver	Comments	Approval for	Created
● Approved	Alice P		(empty)	2025-09-25 01:21:00
● Approved	Alice P		(empty)	2025-09-25 01:44:34
● Approved	Alice P		(empty)	2025-09-25 01:36:39
● Approved	Alice P		(empty)	2025-09-25 06:28:37
● Requested	Bernard Laboy		CHG0000053	2025-07-23 06:09:38
	Bernard Laboy		CHG0000071	2025-07-23 06:12:10
● Requested	Bernard Laboy		CHG0000037	2025-07-23 06:04:51
● Requested	Bernard Laboy		CHG0000076	2025-07-23 06:13:15
● Requested	Bernard Laboy		CHG0000094	2025-07-23 06:15:21
● Requested	Bernard Laboy		CHG0000051	2025-07-23 06:09:31
● Requested	Bernard Laboy		CHG0000073	2025-07-23 06:12:19
● Requested	Bernard Laboy		CHG0000090	2025-07-23 06:15:07
● Requested	Bernard Laboy		CHG0000074	2025-07-23 06:12:23
● Requested	Bernard Laboy		CHG0000055	2025-07-23 06:09:47
● Requested	Bernard Laboy		CHG0000078	2025-07-23 06:13:24
● Requested	Bernard Laboy		CHG0000091	2025-07-23 06:15:11
● Requested	Bernard Laboy		CHG0000045	2025-07-23 06:07:48

10. Testing

Testing ensures that all functionalities of the project work correctly and securely. The following testing approaches were applied:

1. **Functional Testing**
    - Verified user management features: adding, updating, and deleting users.
    - Tested group and role management, ensuring proper RBAC enforcement.
    - Checked workflow automation, including task assignment, status updates, and approval processes.
2. **Access Control Testing**
    - Ensured that users could access only the tables and records permitted by their roles.
    - Verified that unauthorized actions were blocked by Access Control Rules (ACLs).
3. **API Testing**
    - Tested API endpoints for retrieving, creating, updating, and deleting records.
    - Validated request parameters and example responses to ensure data integrity.
4. **User Acceptance Testing (UAT)**
    - Project Manager (Alice) and Team Member (Bob) tested end-to-end task management, workflow approvals, and dashboards.
    - Confirmed that notifications, approvals, and automated updates worked as expected.

11. Future Enhancements

1. **Advanced Role Management**
    - Introduce dynamic roles based on attributes or project needs.
    - Support hierarchical roles for more granular access control.
2. **Integration with External Systems**
    - Connect with HR, ERP, or project management tools for automatic user provisioning and task updates.
    - Integrate with communication platforms (Slack, Teams) for notifications.
3. **Enhanced Workflow Automation**
    - Implement conditional workflows based on task priority, deadlines, or dependencies.
    - Add automated reminders and escalations for overdue tasks.
4. **Reporting and Analytics**
    - Build dashboards for monitoring task progress, approvals, and user activity.
    - Include trend analysis for productivity and workflow efficiency.
5. **Mobile Accessibility**
    - Develop a mobile-friendly interface or application for task management on the go.
6. **Scalability Improvements**
    - Optimize the system to handle larger teams and multiple projects without performance degradation.