Ideation Phase

Empathize & Discover

Date: 02 November 2025

Team ID: NM2025TMID02609

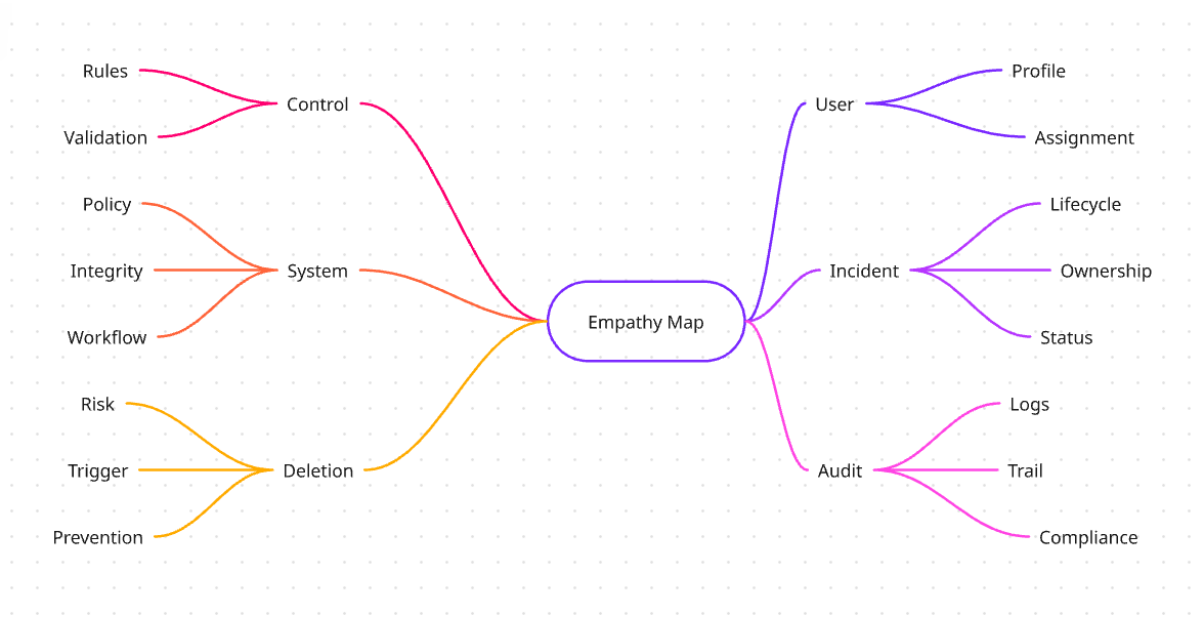
Project Name: Optimizing-User-Group-and-Role-Management-with-Access-Control-and-Workflows

Maximum Marks: 4 marks

🎯 Empathy Map Canvas

Project Title:

Optimizing User, Group, and Role Management with Access Control and Workflows



👤 User Persona (Who are we empathizing with?)

Primary Users:

- **System Administrator – Creates and manages users, groups, roles, and ACLs in ServiceNow**
- **Project Manager (Alice P) – Approves task requests and oversees project completion**
- **Team Member (Bob P) – Performs assigned tasks, updates status and comments**

Secondary Users:

- **Business Stakeholders – Rely on secure, efficient workflows and proper access management**

- **IT Security Team – Ensure compliance and data protection through proper ACL configuration**

💭 **THINKS**

- **"How do I ensure each user has the right level of access without over-permissioning?"**
- **"Will the workflow trigger correctly when Bob updates the task?"**
- **"I need to test this from each user's perspective to verify permissions"**
- **"Access control should be tight but not block legitimate work"**
- **"The approval flow should reduce manual follow-ups"**
- **"What happens if I assign the wrong role to a user?"**
- **"I need to understand role inheritance and how it affects permissions"**

👀 **SEES**

- **ServiceNow interface with User, Group, and Role creation forms**
- **System Security menus for managing users, groups, roles, and ACLs**
- **Project Table and Task Table2 applications in the navigation menu**
- **Flow Designer canvas showing trigger → update → approval workflow**
- **Impersonation view switching between admin, Alice, and Bob perspectives**
- **"Edit" and "Insert new row" options in ACL configuration**
- **My Approvals module showing pending approval requests**
- **Real-time status changes from "In Progress" to "Completed"**
- **Field-level restrictions (editable vs read-only) based on user role**

🗣️ **SAYS**

- **"I'll create Alice as Project Manager and Bob as Team Member"**
- **"Bob should only edit Comments and Status fields, nothing else"**
- **"Let me assign u_project_table and u_task_table roles to the Project Manager"**
- **"I need to impersonate Bob to verify he can only see Task Table2"**
- **"The flow should auto-complete tasks when Bob enters 'feedback' in comments"**

- **"Alice needs to approve before the task is finalized"**
- **"I must elevate my role to create ACLs"**
- **"Let me test this workflow end-to-end before going live"**

💔 FEELS

- **Confident when roles are properly assigned and users can access their required tables**
- **Relieved when the automated workflow triggers correctly without manual intervention**
- **Frustrated when ACL configurations block legitimate user access**
- **Anxious about security gaps if permissions are too broad**
- **Satisfied when Alice receives the approval notification automatically**
- **Accomplished when the entire flow works seamlessly from task creation to approval**
- **Secure knowing that Bob cannot access sensitive project data beyond his scope**
- **Overwhelmed initially by the number of steps required to set up proper access control**

👂 HEARS

- **"Why can't I see the Project Table?" – from Bob when testing access**
- **"The approval came through automatically!" – from Alice**
- **"We need better automation to reduce manual task updates" – from management**
- **"Make sure team members can't delete records" – from IT Security**
- **"Can you add more users to this group?" – from project leads**
- **"The workflow didn't trigger, check the conditions" – from testing team**
- **"Impersonate the user to see what they see" – from senior admin**
- **"We need audit trails for all status changes" – from compliance team**

🏁 DOES

Setup Phase:

- **Creates two users: Alice P (Project Manager) and Bob P (Team Member)**
- **Creates groups for organizing users by function**
- **Creates custom roles: Project Member, Team Member, with table-specific permissions**
- **Assigns roles to users through the User form**

Configuration Phase:

- **Assigns u_project_table and u_task_table roles to Project Manager role**
- **Assigns team member role with table access to Bob**
- **Configures application access for Project Table and Task Table2**
- **Creates 4+ ACLs for field-level access control (Comments, Status, etc.)**
- **Sets up "requires role" relationships in ACL definitions**

Testing Phase:

- **Uses "Impersonate User" feature to switch to Bob's view**
- **Verifies Bob can only see and edit authorized fields**
- **Checks that Bob cannot access Project Table**

Automation Phase:

- **Opens Flow Designer and creates a new flow named "task table"**
- **Configures trigger: Record Created on Task Table with specific conditions**
- **Sets up conditions: Status = "In Progress", Comments = "feedback", Assigned to = Bob**
- **Adds "Update Records" action to change status to "Completed"**
- **Adds "Ask for Approval" action with Alice P as approver**
- **Activates and tests the complete workflow**
- **Verifies Alice receives approval request in My Approvals**
- **Approves the request and confirms task completion**

🎯 Goals / Needs

- **Simplify user administration by creating clear roles and groups**
- **Enforce least privilege principle through granular ACL configuration**
- **Automate repetitive workflows to reduce manual task updates and approvals**
- **Ensure data security by restricting access based on roles**
- **Provide role-based visibility so users only see relevant applications**
- **Enable self-service approvals through automated notifications**
- **Maintain audit compliance with proper access controls**
- **Test permissions thoroughly using impersonation before deployment**
- **Create scalable access model that works as team grows**

⚡ Pain Points / Challenges

- **Complex role hierarchy – Understanding how roles inherit permissions**
- **ACL configuration errors – Wrong settings can block legitimate access or expose sensitive data**
- **Manual testing burden – Need to impersonate each user type to verify permissions**
- **Workflow conditions not triggering – Small mistakes in field values or operators break automation**
- **Role elevation requirements – Must remember to elevate role before creating ACLs**
- **Time-consuming setup – Multiple steps required for each user, role, and ACL**
- **Limited visibility during testing – Hard to see what users actually experience without impersonation**
- **Approval bottlenecks – If automation fails, approvals pile up manually**
- **Field-level ACL complexity – Must create separate ACLs for each field requiring restrictions**
- **No rollback mechanism – Mistakes in production require manual cleanup**

🌟 Key Outcomes

For System Administrators:

- ✅ **Centralized control over users, groups, and roles in one platform**
- ✅ **Field-level security through properly configured ACLs**
- ✅ **Ability to test permissions before deployment using impersonation**

For Project Managers (Alice):

- ✅ **Automated approval notifications reduce manual tracking**
- ✅ **Clear visibility into pending approvals in My Approvals module**
- ✅ **Faster task completion through automated workflows**

For Team Members (Bob):

- ✅ **Simple, focused interface showing only authorized tables**
- ✅ **Clear permissions on which fields can be edited**
- ✅ **Automated status updates reduce repetitive work**

For the Organization:

- ✅ **Enhanced security with role-based access control**
- ✅ **Improved efficiency through workflow automation**
- ✅ **Better compliance with audit trails and access restrictions**
- ✅ **Scalable user management model**
- ✅ **Reduced manual errors in task assignment and approval processes**
- ✅ **Clear separation of duties between roles**
