**1. Integration with LDAP AD**

* **LDAP Authentication**: Integrate CODI MongoDB with LDAP AD for user authentication. This allows MongoDB to leverage existing user credentials and policies in the AD.

**2. Defining Roles and Privileges**

* **Role-based Access Control (RBAC)**: Use MongoDB’s built-in RBAC features to define roles and restrict access at different levels.
  + Create custom roles tailored to specific operational needs.
  + Assign roles based on the principle of least privilege, ensuring users have only the access required for their tasks.
* **Service Accounts for Automation**: Create service accounts for automated tasks and restrict their roles to the minimum required actions (e.g., readWrite on specific databases).
  + Avoid using admin-level roles for service accounts. Create roles like readWrite or dbAdmin specifically for their scope.

**3. Multi-Tenant Database Access Management**

* **Database-level Isolation**: Ensure that each tenant has access only to their respective database. Create LDAP groups corresponding to each tenant and map those groups to roles that provide access to specific databases.
  + Example: Map an LDAP group tenant1\_admins to a role in MongoDB that grants readWrite access to the tenant1\_db database.
* **Separate Service Accounts for Each Tenant**: Create unique service accounts for each tenant's automated processes, ensuring these accounts are restricted to accessing only their corresponding databases.