**Active Directory (AD)**

**What is it?**

is a directory service developed by Microsoft, primarily for Windows Server environments. It provides a centralized system for managing access to networked resources such as users, computers, and printers within a secure and organized infrastructure.

**What is the meaning of directory and directory service?**

A directory is a hierarchical structure that stores information about objects on a network.

A directory service, such as Active Directory Domain Services (AD DS), provides the methods for storing directory data and making this data available to network users and administrators.

For example, AD DS stores information about user accounts, such as names, passwords, phone numbers, and so on. AD DS also provides a way for authorized users on the same network to access this information.

**At the simplest level:** you can think in **Active Directory like this:**

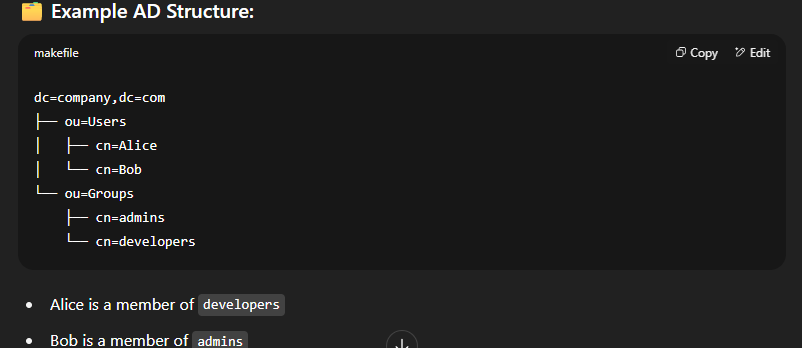
**Active Directory is a centralized database and authentication system** used in Windows-based networks to manage **users, groups, computers, and their access** to resources.

**Example Scenario:**

Imagine you're building an **internal admin tool** for a bank (as we do already):

* You don’t build your own login system.
* Instead, when a user logs in, your app asks **Active Directory**:
  + "Is this username and password, correct?"
  + "What is this user's role or group?"
  + "Is this user allowed to access this page?"

**Essential things and terminologies you must know in AD:**

* **AD is hierarchical, like a folder structure:**

**DC:** domain component: a part of the domain name.

**OU:** organization unit: a logical container (like folder)

**CN:** common name: the name of an object (user, device, ..)

**The LDAP path to Alice would be:**

cn=Alice, ou=users, dc=company,dc=com

**Starting for the leaf to the root.**

**Detailed Explanation:**

### ****dc = Domain Component:****

Used to define your **domain name** in parts.

For domain: company.com → LDAP: dc=company,dc=com

### ****ou = Organization Unit:****

Used to group your **objects, you can have nested OUs too**.

Example: ou=HR. ou=IT. And so on

### ****cn = common name****

The unique name of an individual **object** inside AD, usually the user or group name.

**so the dc contain OUs, and OUs contains CNs**

**Before starting integration with AD you must know:**

1. **Server hostname/IP and port**
2. **Base DN:** stands for **Base Distinguished Name**.

It defines the **starting point** (or root path) in the directory tree from which **LDAP searches** will begin.

1. **Bind DN (Service Account) And password: Distinguished Name of the user account** your application uses to "log in" (bind) to the LDAP/Active Directory server so it can perform operations like **searching for users**.

**Think of Bind DN as:** the key and secret when you try to integrate with a secured API.

1. **Search filters:**

A **search filter** is a string used to query objects in Active Directory (AD). It helps you find users, groups, or other objects **that meet specific conditions**.

**Think of it like SQL’s WHERE clause, but for AD**.

**Basic LDAP Filter Syntax: (attribute=value)**

**Example: (**sAMAcountName=Alice**).**

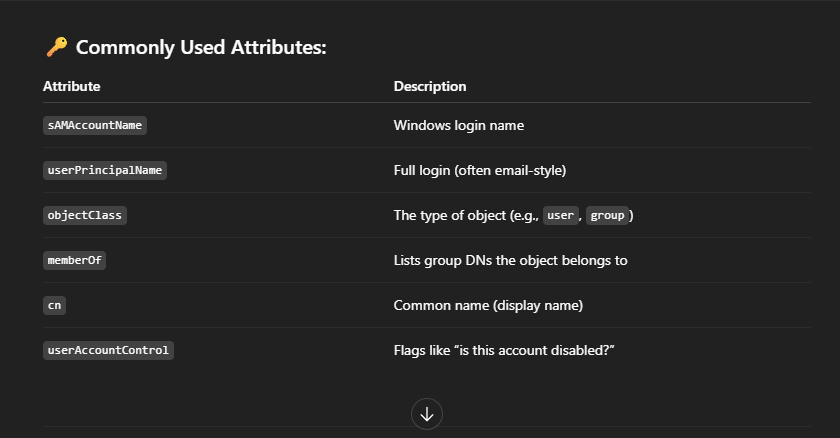
**This means:** Find a user with sAMAccountName (Windows login name) = Alice

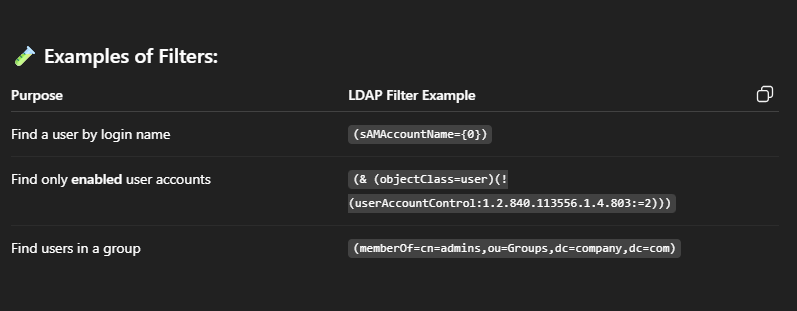
**You can combine multiple filters using prefix operators:**

**&: and 🡪 (&(filter1) (filter2)) 🡪 filter1 and filter2**

**`: or 🡪 (`(filter1) (filter2)) 🡪 filter1 or filter2**

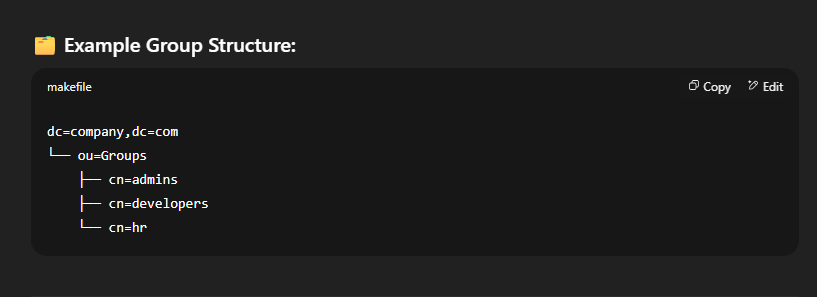
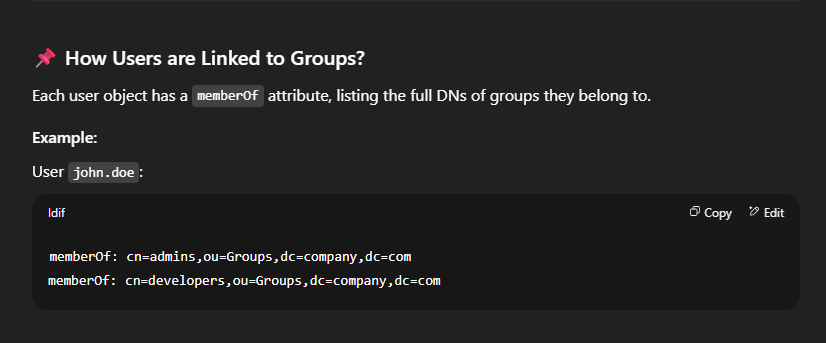
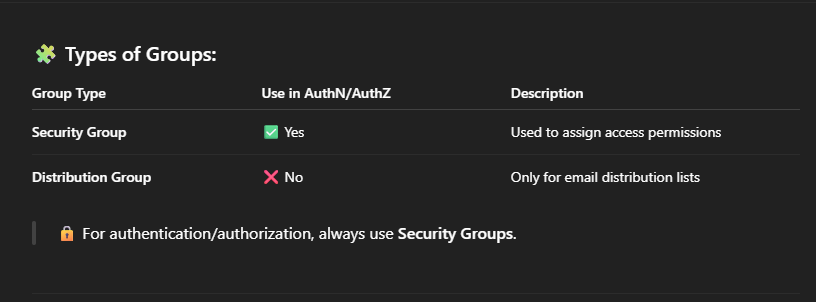
**!: not 🡪 (!(filter)) 🡪 excludes**

****

****

1. **group structure:**

A **group** in AD is a collection of user accounts used to manage **permissions, roles, or access**.

****

**Time to work:** lets build an AuthN system using spring boot and spring security LDAP.

**Steps:**

**1- Initialize your spring application**

* visit <https://start.spring.io>
* Select these dependencies: Spring web, Spring Security, Spring LDAP
* Generate

**2- Configure**

* In applicarion.yml:

spring:

ldap:

urls: ldap://ad.hostname:port

base: dc=company,dc=com

username: {Bind DN (service account)}

password: {service account password}

**3- Implement LDAP authentication provider**

**There are 3 main functions in the auth provider:**

* **filterChain:** This function defines how HTTP requests should be secured in the application—who can access what, and how login is handled.

**Or in other words:** it Secures HTTP requests (authenticatio& authorization)

* **contextSource:** This function sets up the connection details for your LDAP (Active Directory) server.
* **ldapAuthenticationManager:** Authenticates user credentials using LDAP bind.

**4- Future enhancement:**

Make your APIs token-based using JWT

