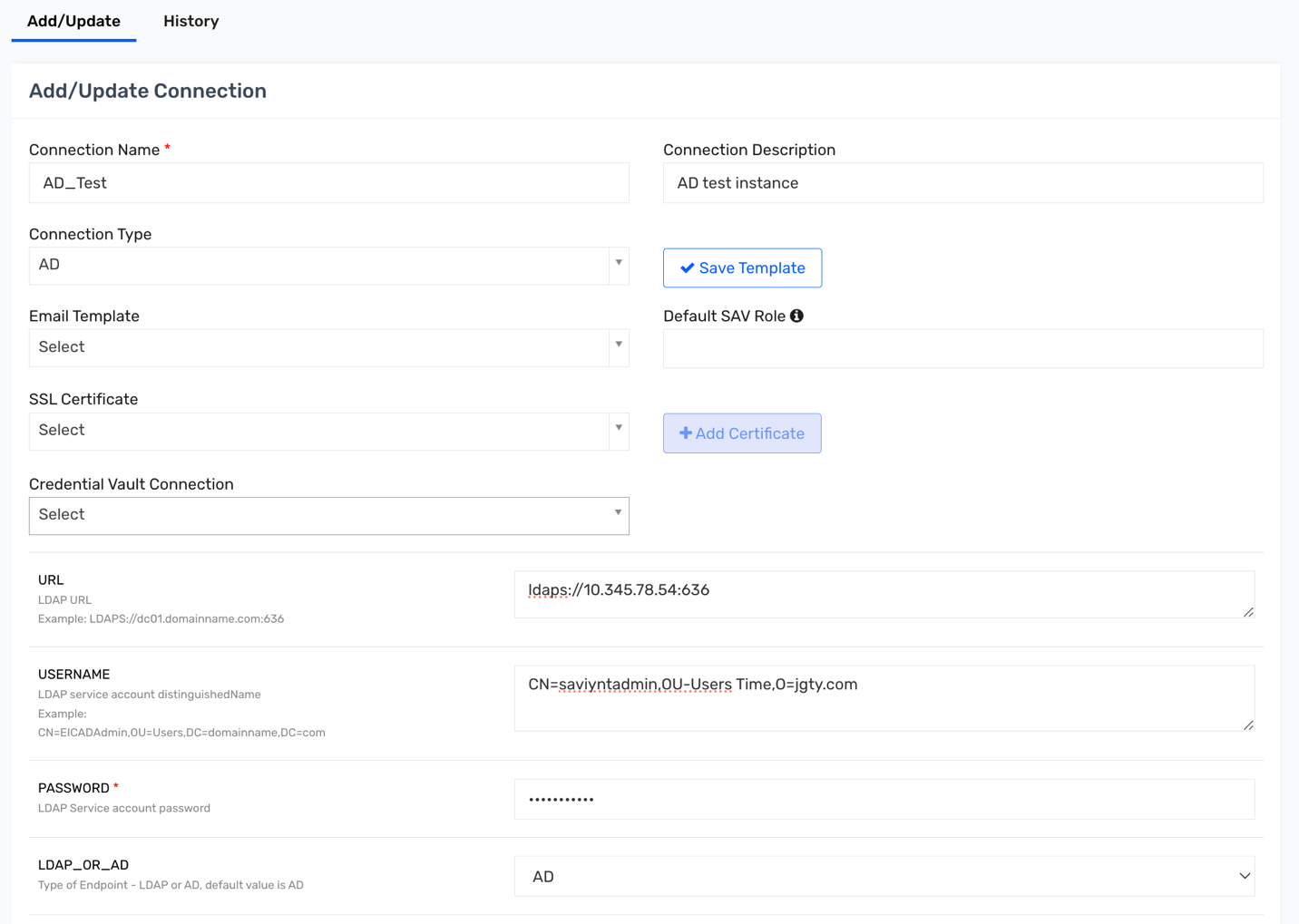
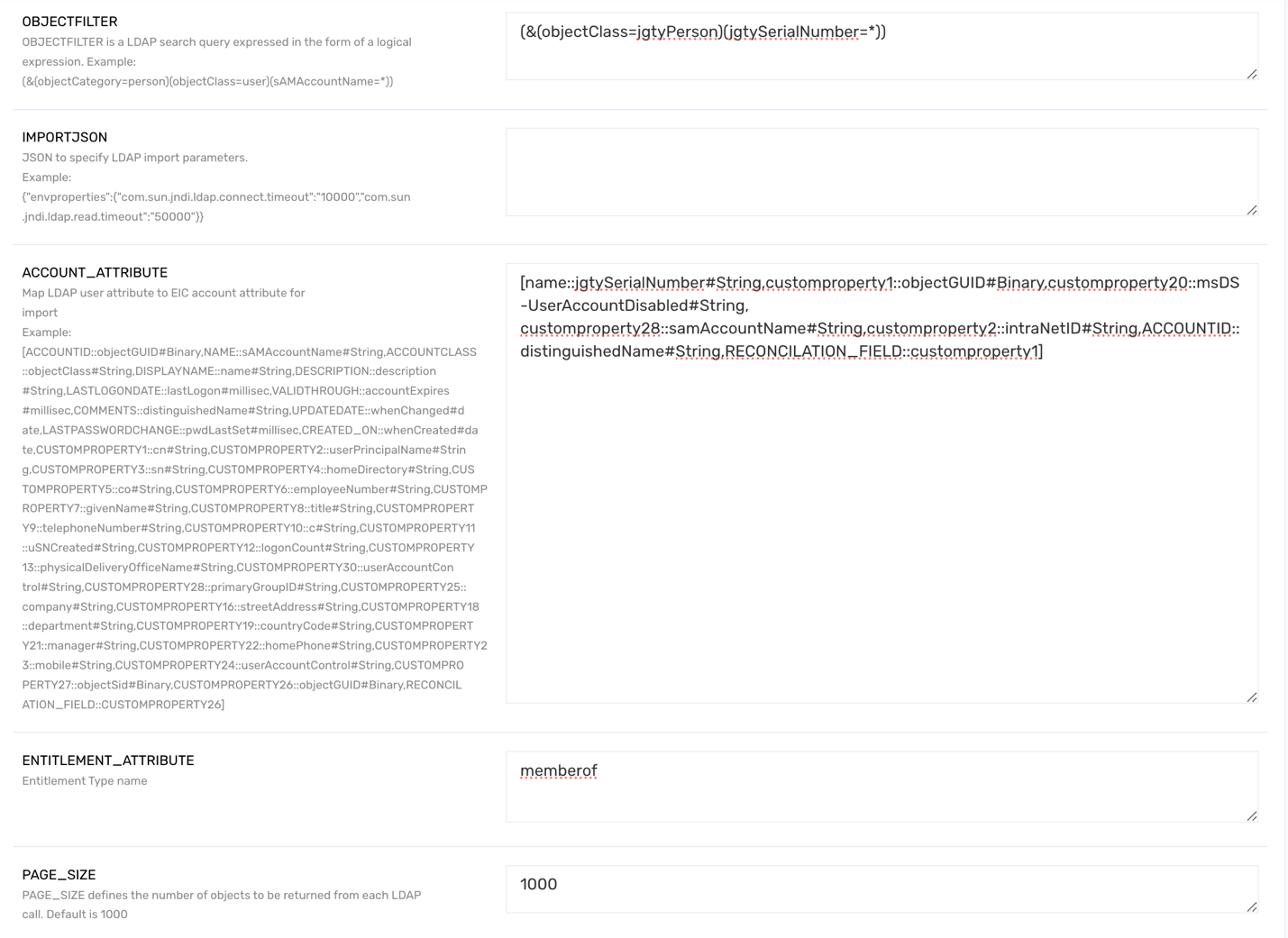
Sample AD connector

Introduction:  
Saviynt EIC offers OOB capability to integrate AD applications with Saviynt with the help of AD connectors. Saviynt Provides OOB templates that can be configured accordingly.  
  
  
How it works:  
  
Saviynt offers a connector module which can be configured to make a connection with the application.  
  
  
Prerequisites:   
  
Network connectivity should be established before we can begin operations like recon and provisioning. The network needs to be whitelisted, and firewall should allow the network to flow from Saviynt to EIC and vice versa.  
  
Integration Steps:  
  
Create an Ad connection. Let’s look at the integration steps .



1)Provide a Connection Name and Description.  
2) Select the Connection Type as AD, it will load the template below according to AD connection.  
3) Give the URL: This is the Instance of the AD to which we are trying to connect.  
4) Give the username and password.  
5) Select if its AD or LDAP  
   
  


6) Give the Search filter and Object Filter  
Search filter is the starting point in the directory from where inject will be imported, In this case it is   
o=jgty.com  
Object filter is a query In the form of Logical expression, example  
(&(objectClass=jgtyPerson)(jgtySerialNumber=\*))  
  
7) Give the Account Attribute which will map the LDAP attribute to EIC attribute  
example:  
  
[name::jgtySerialNumber#String,customproperty1::objectGUID#Binary,customproperty20::msDS-UserAccountDisabled#String,

customproperty28::samAccountName#String,customproperty2::intraNetID#String,ACCOUNTID::distinguishedName#String,RECONCILATION\_FIELD::customproperty1]  
  
**Account Name**: The jgtySerialNumber attribute is designated as the main identifier for the account and mapped to the **name** field in Saviynt.

* **Object GUID**: The objectGUID is a unique identifier for each account. It’s mapped to **customproperty1** and serves as the **Reconciliation Field** for matching existing records.
* **Account Disabled Status**: The attribute msDS-UserAccountDisabled shows whether an account is disabled. This information is stored in **customproperty20**.
* **SAM Account Name**: The samAccountName is a commonly used account identifier in Active Directory and is assigned to **customproperty28** in Saviynt.
* **Intranet ID**: This is identified by the intraNetID attribute and mapped to **customproperty2** for referencing each user’s intranet identifier.
* **Distinguished Name (DN)**: The distinguishedName attribute provides the full DN for each account, mapped to **ACCOUNTID** in Saviynt. This field is crucial for identifying accounts during synchronization.
* **Reconciliation field**: Unique field on the basis of which the accounts will be differentiated , which is Customproprty1 here and Cp1 is mapped to onjectGUID.

8) Entitlement Attribute: entitlement type name which will be reconned.  
  
9) Page\_size defines the number of records which will be returned for each call

A screenshot of a computer

Description automatically generated

10) Create Account Json: To create an account, Specfy the Json Paramters as below:

{ "objectClass": ["top", "person", "organizationalPerson", "user", "jgtyPerson"],

"jgtySerialNumber": "${user.username}",

"sn":"${user.lastname}",

"cn":"${user.firstname}",

"mail":"${user.email}",

"telephoneNumber":"${user.customproperty24}",

"serialNumber":"${user.customproperty10}",

"isManager":"${user.customproperty11}",

"employeeType":"${user.employeeType}",

"userPrincipalName":"${user.customproperty22}",

"workLocation":"${user.locationdesc}",

"workLoc":"${user.location}",

"samAccountName":"${user.customproperty1}",

"firstName":"${user.firstname}",

"lastName":"${user.lastname}",

"departmentName":"${user.departmentname}",

"tieLine":"${user.customproperty23}",

"intraNetId":"${user.customproperty22}",

"uid":"${user.username}",

"manager":"${managerAccount==null?'':managerAccount.accountID}",

"msDS-UserAccountDisabled":"${user.customproperty17!=null && user.customproperty17.toString().toUpperCase()=='A'?'FALSE':'TRUE'}" ,}

* **objectClass**: Specifies the classes associated with the user object, including standard classes like top, person, organizationalPerson, user, and a custom class jgtyPerson.
* **jgtySerialNumber**: Maps to ${user.username}, which assigns the unique username from Saviynt to this attribute.
* **sn (Surname)**: Maps to ${user.lastname}, pulling the user's last name from Saviynt.
* **cn (Common Name)**: Maps to ${user.firstname}, using the user's first name from Saviynt as the common name in Active Directory.
* **mail**: Maps to ${user.email}, assigning the user’s email address.
* **telephoneNumber**: Maps to ${user.customproperty24}, which likely holds the user's phone number stored in Saviynt.
* **serialNumber**: Maps to ${user.customproperty10}, indicating a unique serial number or ID specific to this user.
* **isManager**: Maps to ${user.customproperty11}, which might store a flag (or identifier) indicating if the user has a managerial role.
* **employeeType**: Maps to ${user.employeeType}, pulling the employee type information (e.g., full-time, contractor) from Saviynt.
* **userPrincipalName**: Maps to ${user.customproperty22}, which typically holds the user principal name (UPN) for logging into AD or other systems.
* **workLocation**: Maps to ${user.locationdesc}, providing a description of the user's primary work location.
* **workLoc**: Maps to ${user.location}, a more precise or internal code/location ID.
* **samAccountName**: Maps to ${user.customproperty1}, likely containing the SAM account name (AD-specific attribute).
* **firstName**: Maps to ${user.firstname}, defining the user’s first name.
* **lastName**: Maps to ${user.lastname}, defining the user’s last name.
* **departmentName**: Maps to ${user.departmentname}, specifying the department where the user works.
* **tieLine**: Maps to ${user.customproperty23}, which might hold a specific internal number or tie-line for the user.
* **intraNetId**: Maps to ${user.customproperty22}, duplicating the UPN as the intranet identifier.
* **uid**: Maps to ${user.username}, setting the user’s ID, often required in directories.
* **manager**: Uses a conditional mapping. If managerAccount is null, it leaves the field blank. Otherwise, it assigns the managerAccount.accountID, associating the user with their manager’s account.
* **msDS-UserAccountDisabled**: Uses a conditional expression. If ${user.customproperty17} equals 'A' (when converted to uppercase), it sets this attribute to 'FALSE', enabling the account. If ${user.customproperty17} does not equal 'A', it sets it to 'TRUE', disabling the account.

11) Update Account Json : To update an already existing account  
  
{

"jgtySerialNumber": "${user.username}",

"sn":"${user.lastname}",

"cn":"${user.firstname}",

"mail":"${user.email}",

"telephoneNumber":"${user.customproperty24}",

"serialNumber":"${user.customproperty10}",

"isManager":"${user.customproperty11}",

"employeeType":"${user.employeeType}",

"userPrincipalName":"${user.customproperty22}",

"workLocation":"${user.locationdesc}",

"workLoc":"${user.location}",

"samAccountName":"${user.customproperty1}",

"firstName":"${user.firstname}",

"lastName":"${user.lastname}",

"departmentName":"${user.departmentname}",

"tieLine":"${user.customproperty23}",

"intraNetId":"${user.customproperty22}",

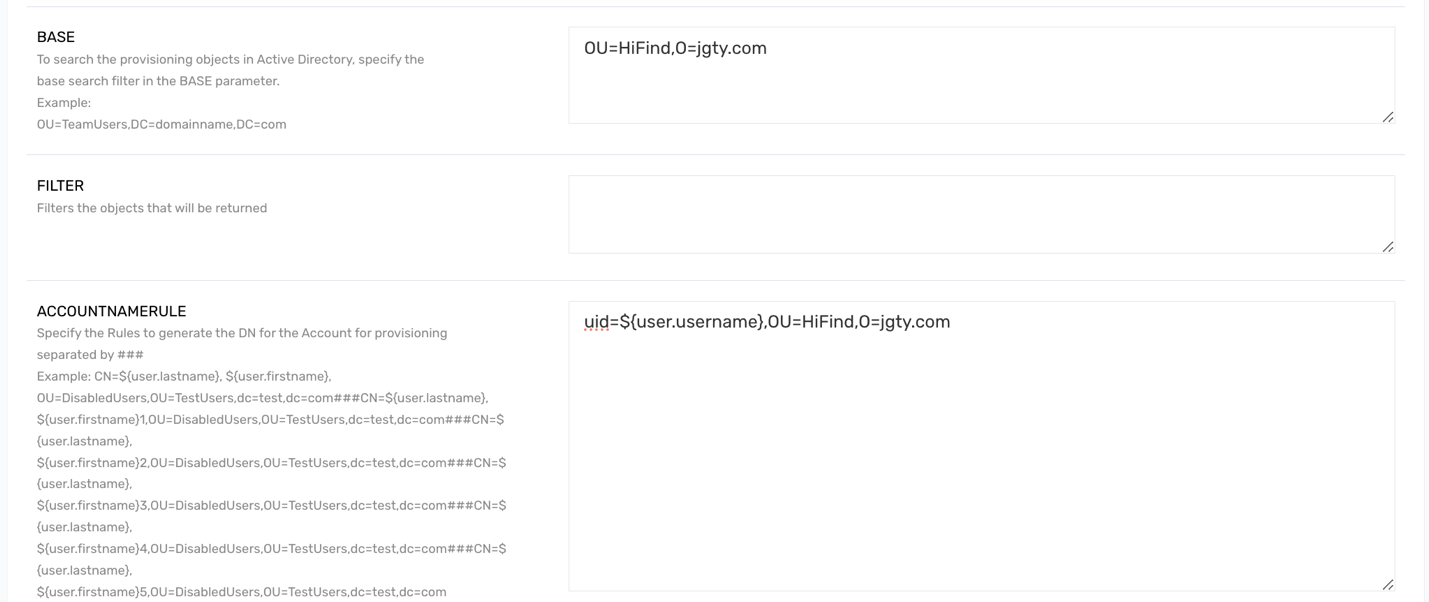
"uid":"${user.username}",

"manager":"${managerAccount==null?'':managerAccount.accountID}",

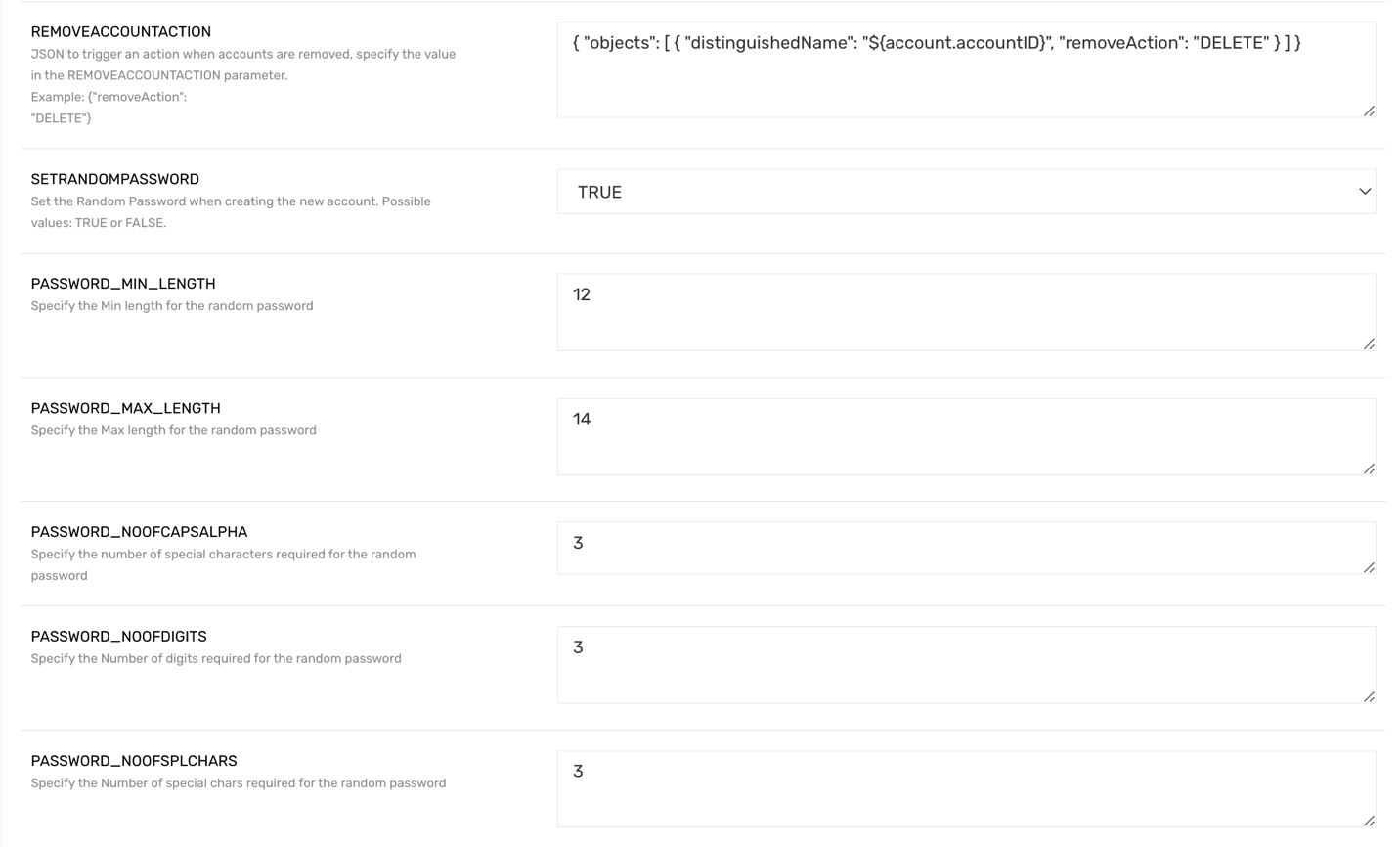
"msDS-UserAccountDisabled":"${user.customproperty17!=null && user.customproperty17.toString().toUpperCase()=='A'?'FALSE':'TRUE'}"

}

* **jgtySerialNumber**: Maps to the username (${user.username}) from Saviynt, used as a unique identifier.
* **sn (Surname)**: Maps to the user’s last name (${user.lastname}).
* **cn (Common Name)**: Maps to the user’s first name (${user.firstname}), serving as the display name.
* **mail**: Maps to the user’s email (${user.email}), ensuring email contact info is available.
* **telephoneNumber**: Maps to a custom property in Saviynt (${user.customproperty24}), likely containing the user’s phone number.
* **serialNumber**: Maps to another custom property (${user.customproperty10}), possibly a unique identifier specific to the user.
* **isManager**: Maps to a custom property (${user.customproperty11}) that may indicate managerial status.
* **employeeType**: Maps to employeeType in Saviynt, such as full-time, part-time, or contractor.
* **userPrincipalName**: Maps to ${user.customproperty22}, typically used for user login.
* **workLocation**: Maps to ${user.locationdesc}, describing the user’s work location.
* **workLoc**: Maps to ${user.location}, representing a specific internal location code.
* **samAccountName**: Maps to ${user.customproperty1}, often used for authentication in AD environments.
* **firstName**: Maps to ${user.firstname}, ensuring consistency across fields.
* **lastName**: Maps to ${user.lastname}, again maintaining consistency.
* **departmentName**: Maps to the user’s department (${user.departmentname}).
* **tieLine**: Maps to ${user.customproperty23}, potentially an internal number or direct line.
* **intraNetId**: Maps to ${user.customproperty22}, which may be the user’s intranet login ID.
* **uid**: Maps to ${user.username}, another identifier for the user.
* **manager**: Conditionally maps to managerAccount.accountID. If managerAccount is null, it remains empty; otherwise, it pulls the manager's ID.
* **msDS-UserAccountDisabled**: Uses a conditional expression based on ${user.customproperty17}. If it equals 'A', the account is set to 'FALSE' (enabled). Otherwise, it’s set to 'TRUE' (disabled).



12) Base: To search the provisioning objects in Active Directory, specify the base search filter  
example: OU=HiFind,O=jgty.com  
  
  
13) AccountNamerule: rule to generate the account name from Saviynt , This will take precedence over the one mentioned in the endpoint.  
example: uid=${user.username},OU=HiFind,O=jgty.com



14) RemoveAccountAction : This is the json to torgger an action when accounts are removed  
Example: { "objects": [ { "distinguishedName": "${account.accountID}", "removeAction": "DELETE" } ] }

* **objects**: An array holding items (in this case, only one) that define actions for specific accounts in the directory.
* **distinguishedName**: Maps to ${account.accountID}, which represents the unique identifier (DN) of the account to be removed. The distinguished name is an attribute that uniquely identifies an entry in directory services like Active Directory.
* **removeAction**: Set to "DELETE", indicating that this account should be removed from the directory system.

15) SETRANDOMPASSWORD: Set this to true if random password needs to be generated.

16) Password\_min\_Length : Defines the minimum length of the password.  
  
17)Password\_max\_length: Defines the maximum length of the password.  
  
18) PASSWORD\_NOOFCAPSALPHA: Defined the numver of capital characters required for password.  
  
19) PASSWORD\_NOOFDIGITS: Specifiy the number of digits required for password  
  
20) PASSWORD\_NOOFSPLCHARS: Specify the number of chars required for the random password

A screenshot of a computer

Description automatically generated

21) Status Threshold Config:   
{

"statusAndThresholdConfig":

{

"statusColumn":"customproperty30",

"activeStatus":["FALSE"],

"inactiveStatus":["true"],

"deleteLinks": true,

"accountThresholdValue" : 80000,

"correlateInactiveAccounts": true

}

}  
  
**statusAndThresholdConfig**: This is the main configuration object for handling account statuses and associated thresholds.

* **statusColumn**: Specifies "customproperty30" as the column or attribute that indicates the status of an account.
* **activeStatus**: Defines the value(s) that represent an active account. Here, "FALSE" indicates active status.
* **inactiveStatus**: Defines the value(s) that represent an inactive account. Here, "true" indicates inactive status.
* **deleteLinks**: Set to true, indicating that any links or references to accounts that meet certain conditions should be deleted.
* **accountThresholdValue**: Sets a threshold limit of 80000, which might be used to trigger certain actions or alerts if this threshold is exceeded (e.g., excessive account usage or count).
* **correlateInactiveAccounts**: Set to true, meaning that inactive accounts should be correlated or tracked for specific purposes, such as reporting or automated deprovisioning.

Troubleshooting:  
  
The Most common error in creating a connection would be the SSL error  
ldap.SaviyntGroovyLdapService - Checking for url = ldaps://[host:636](http://test.paysafecorp.net:636/)

2020-11-05 13:09:20,806 [https-jsse-nio-443-exec-79] ERROR ldap.SaviyntGroovyLdapService - Exception.. try next url

javax.naming.CommunicationException: simple bind failed: [host:636](http://test.paysafecorp.net:636/) [Root exception is javax.net.ssl.SSLHandshakeException: **sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target]  
  
Resolution:  
Ensure the certificate is properly placed in the SSL certificate of the connection and sometimes it might need all the certs including root,leaf and intermediate.**