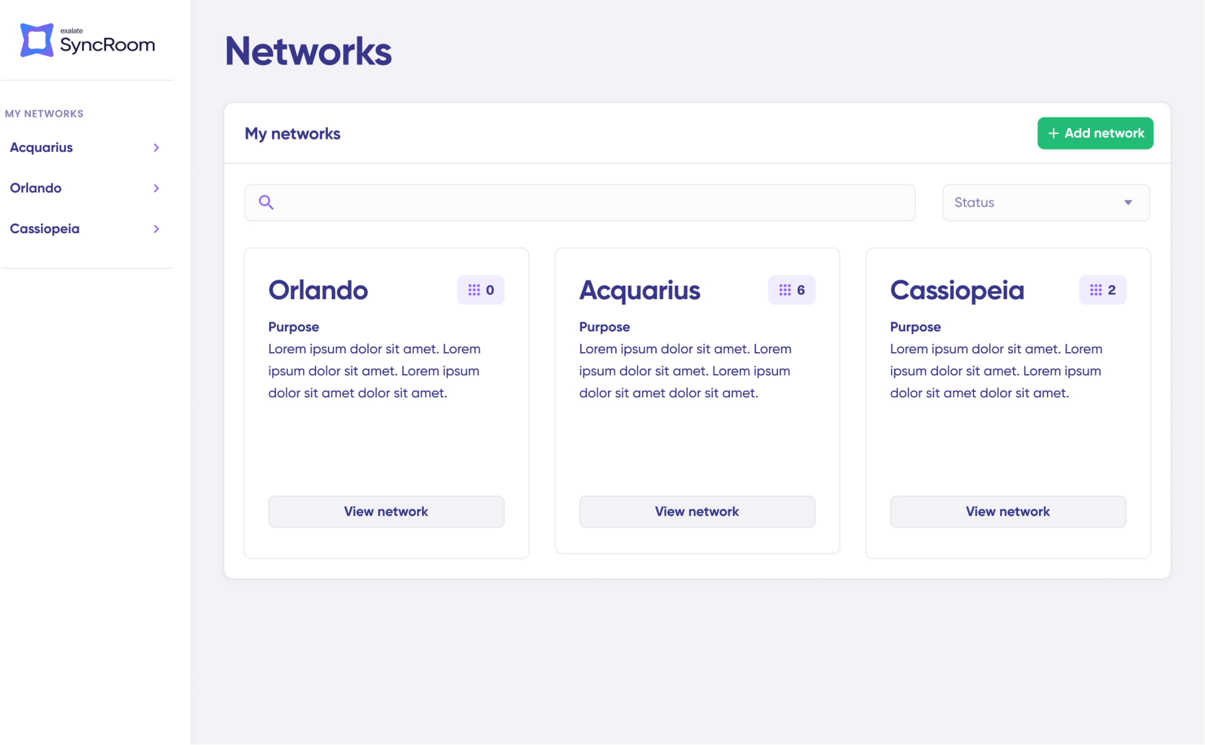
SyncRoom Main Features

Last Modified on 05/02/2025 7:15 am EDT

SyncRoom provides a range of features to help streamline network and node management:

Create Node Networks

You can create one or more networks to organize and manage your Exalate nodes and connections efficiently. The same nodes can belong to multiple networks, but connections are restricted within a single network. A connection cannot exist in more than one network.



Orlando 0

Purpose

Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet dolor sit amet.

Acquarius 6

Purpose

Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet dolor sit amet.

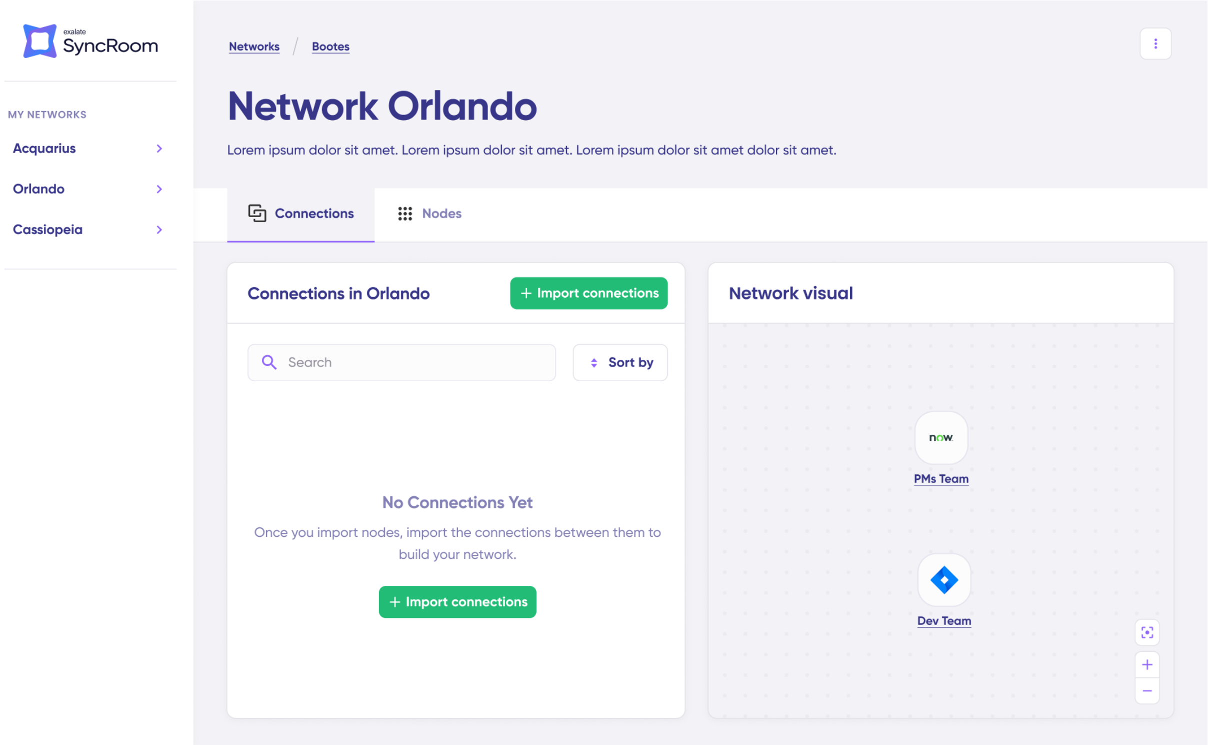
Cassiopeia 2

Purpose

Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet dolor sit amet.

Import & Manage Connections

You can choose which connections to manage within your network. It is also possible to temporarily unmanage a connection in SyncRoom, returning its management to the **Exalate Console** without removing it from the network. Similarly, you can regain management of the connection in the SyncRoom when needed.



Network Orlando

Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet dolor sit amet.

Connections in Orlando

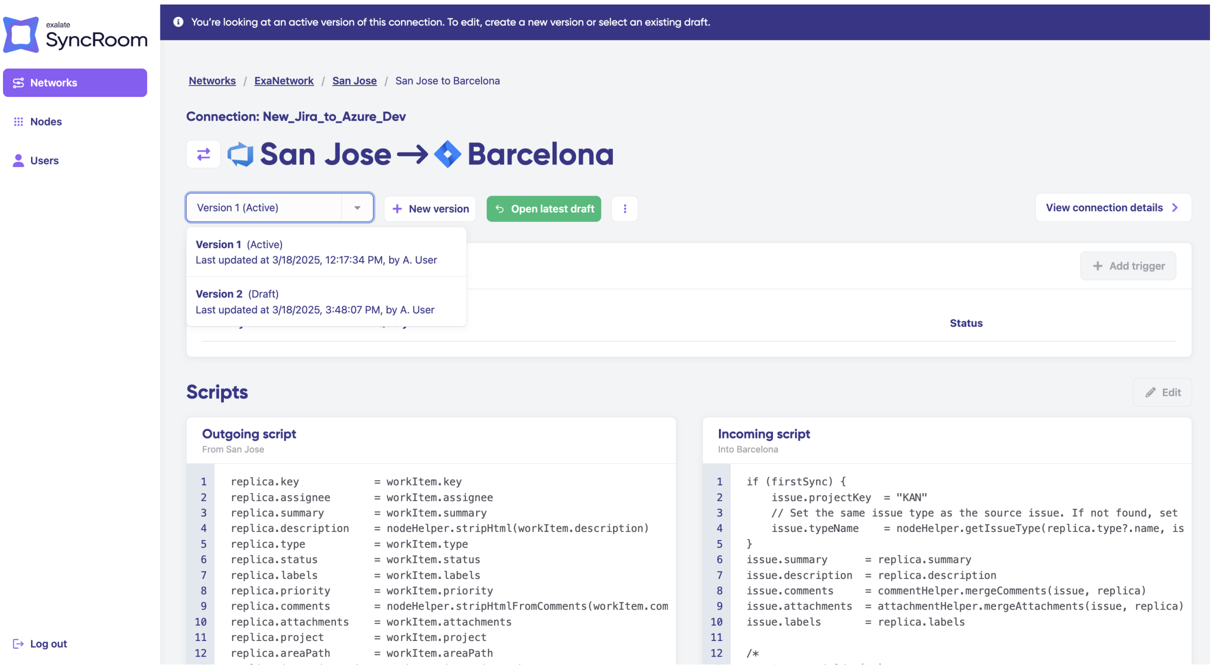
No Connections Yet

Once you import nodes, import the connections between them to build your network.

+ Import connections

Configure Scripts and Triggers with Versioning

You can create multiple configuration version drafts and publish a version from SyncRoom. Each version consists of two scripts—an outgoing script from the source side and an incoming script for the destination side—along with a trigger. It is also possible to switch the data flow direction and create another version to manage the backflow separately.



You're looking at an active version of this connection. To edit, create a new version or select an existing draft.

Connection: New_Jira_to_Azure_Dev

San Jose → Barcelona

Version 1 (Active)
Last updated at 3/18/2025, 12:17:34 PM, by A. User

Version 2 (Draft)
Last updated at 3/18/2025, 3:48:07 PM, by A. User

Scripts

Outgoing script
From San Jose

```
replica.key          = workItem.key
replica.assignee     = workItem.assignee
replica.summary      = workItem.summary
replica.description  = nodeHelper.stripHtml(workItem.description)
replica.type         = workItem.type
replica.status       = workItem.status
replica.labels       = workItem.labels
replica.priority     = workItem.priority
replica.comments     = nodeHelper.stripHtmlFromComments(workItem.com
replica.attachments  = workItem.attachments
replica.project      = workItem.project
replica.areaPath     = workItem.areaPath
```

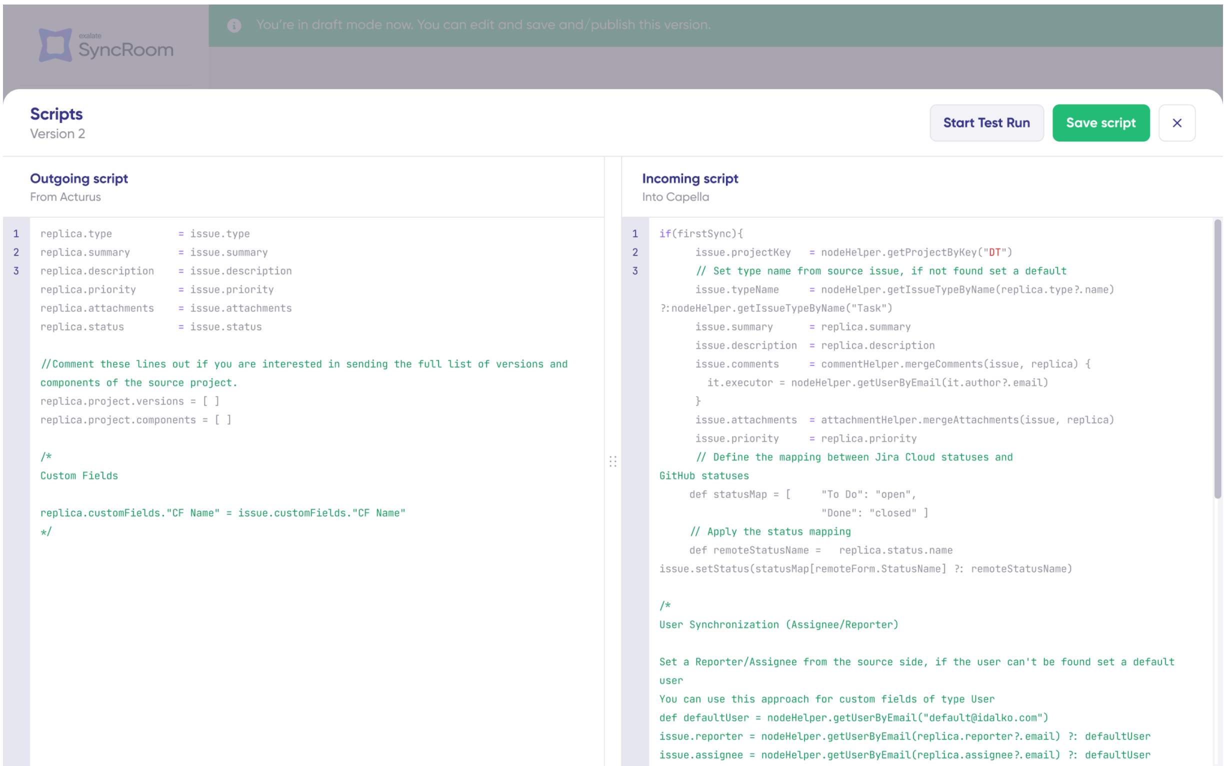
Incoming script
Into Barcelona

```
if (firstSync) {
    issue.projectKey  = "KAN"
    // Set the same issue type as the source issue. If not found, set
    issue.typeName    = nodeHelper.getIssueType(replica.type?.name, is
}
issue.summary      = replica.summary
issue.description  = replica.description
issue.comments     = commentHelper.mergeComments(issue, replica)
issue.attachments  = attachmentHelper.mergeAttachments(issue, replica)
issue.labels       = replica.labels

/*
```

Side-by-Side Script View

The outgoing script from the source side and the incoming script for the destination side are displayed side by side, making it easier to create, compare, and manage configurations.


exalate
SyncRoom
You're in draft mode now. You can edit and save and/publish this version.
Start Test Run
Save script
×

Scripts
Version 2

Outgoing script
From Acturus

```
replica.type            = issue.type
replica.summary         = issue.summary
replica.description     = issue.description
replica.priority        = issue.priority
replica.attachments     = issue.attachments
replica.status          = issue.status

//Comment these lines out if you are interested in sending the full list of versions and
components of the source project.
replica.project.versions = [ ]
replica.project.components = [ ]

/*
Custom Fields

replica.customFields."CF Name" = issue.customFields."CF Name"
*/
```

Incoming script
Into Capella

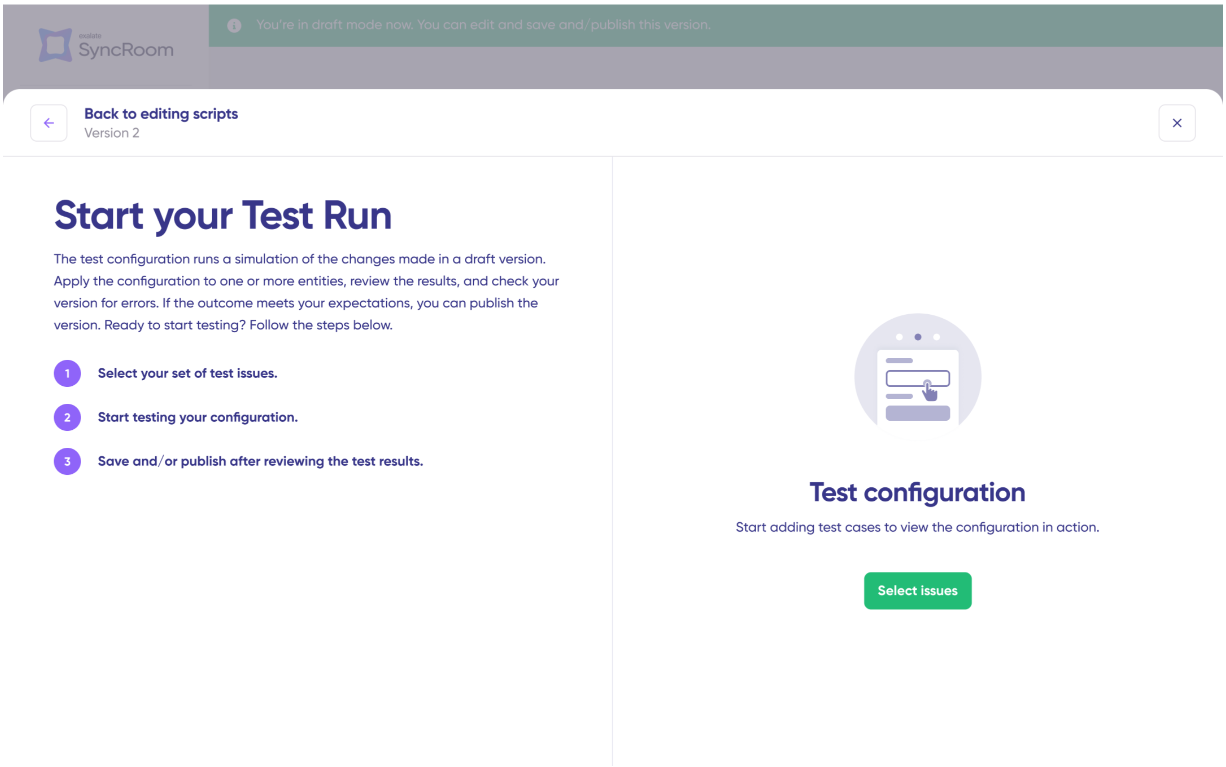
```
if(firstSync){
      issue.projectKey   = nodeHelper.getProjectByKey("DT")
      // Set type name from source issue, if not found set a default
      issue.typeName     = nodeHelper.getIssueTypeByName(replica.type?.name)
?:nodeHelper.getIssueTypeByName("Task")
      issue.summary      = replica.summary
      issue.description  = replica.description
      issue.comments     = commentHelper.mergeComments(issue, replica) {
        it.executor = nodeHelper.getUserByEmail(it.author?.email)
      }
      issue.attachments  = attachmentHelper.mergeAttachments(issue, replica)
      issue.priority     = replica.priority
      // Define the mapping between Jira Cloud statuses and
GitHub statuses
     def statusMap = [     "To Do": "open",
                           "Done": "closed" ]
     // Apply the status mapping
     def remoteStatusName =   replica.status.name
issue.setStatus(statusMap[remoteForm.StatusName] ?: remoteStatusName)

/*
User Synchronization (Assignee/Reporter)

Set a Reporter/Assignee from the source side, if the user can't be found set a default
user
You can use this approach for custom fields of type User
def defaultUser = nodeHelper.getUserByEmail("default@idalko.com")
issue.reporter = nodeHelper.getUserByEmail(replica.reporter?.email) ?: defaultUser
issue.assignee = nodeHelper.getUserByEmail(replica.assignee?.email) ?: defaultUser
```

Test Run Your Configuration

You can TestRun your scripts without creating an entity on the other side and compare the replica before making changes in production.


exalate
SyncRoom
You're in draft mode now. You can edit and save and/publish this version.
×
Select issues

Back to editing scripts
Version 2

Start your Test Run

The test configuration runs a simulation of the changes made in a draft version. Apply the configuration to one or more entities, review the results, and check your version for errors. If the outcome meets your expectations, you can publish the version. Ready to start testing? Follow the steps below.

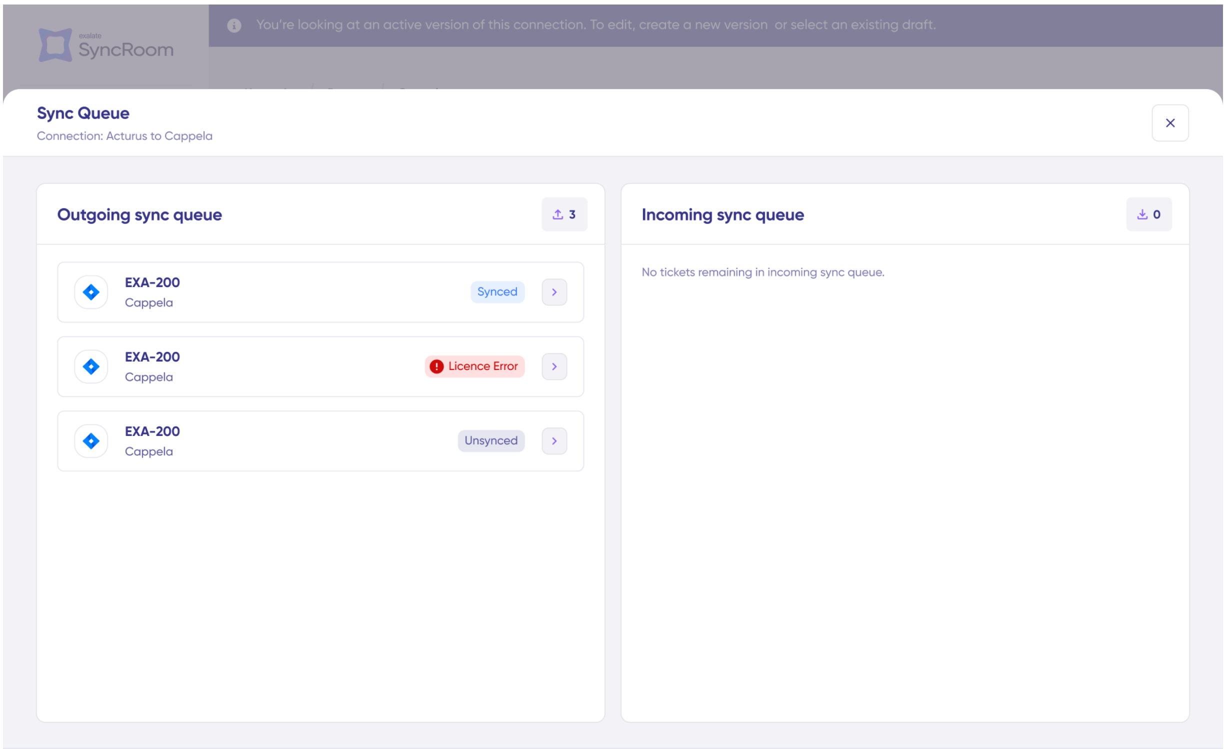
1. Select your set of test issues.
2. Start testing your configuration.
3. Save and/or publish after reviewing the test results.

Test configuration

Start adding test cases to view the configuration in action.

Monitor Sync Queue and Sync Status

Track ongoing incoming and outgoing synchronization messages, check sync status, and Exalate or Connect entities manually through the Entity Sync Status menu.



You're looking at an active version of this connection. To edit, create a new version or select an existing draft.

Sync Queue
Connection: Acturus to Cappela

Outgoing sync queue 3

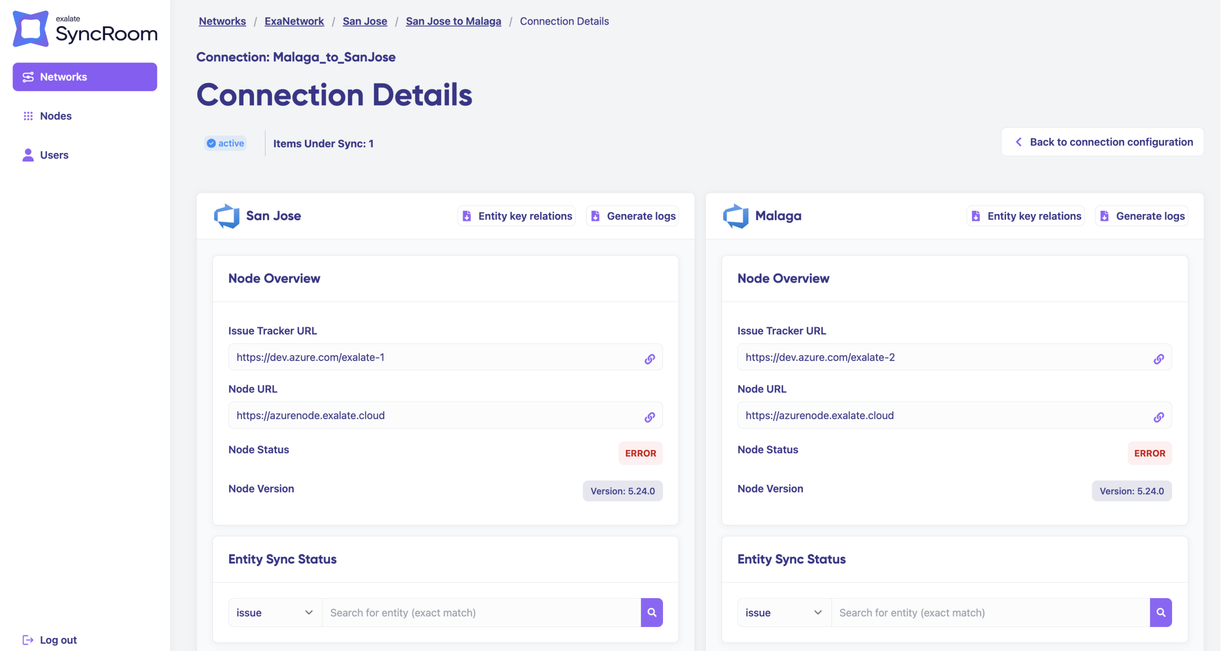
EXA-200 Cappela	Synced
EXA-200 Cappela	Licence Error
EXA-200 Cappela	Unsynced

Incoming sync queue 0

No tickets remaining in incoming sync queue.

Check Connection Details

View both nodes involved in a connection, including their status, version, and any reported errors. This helps in troubleshooting and ensuring smooth synchronization between nodes.



Networks / ExaNetwork / San Jose / San Jose to Malaga / Connection Details

Connection: Malaga_to_SanJose

Connection Details

active | Items Under Sync: 1

San Jose

Node Overview

Issue Tracker URL: https://dev.azure.com/exalate-1

Node URL: https://azurenode.exalate.cloud

Node Status: ERROR

Node Version: Version: 5.24.0

Entity Sync Status

issue | Search for entity (exact match)

Malaga

Node Overview

Issue Tracker URL: https://dev.azure.com/exalate-2

Node URL: https://azurenode.exalate.cloud

Node Status: ERROR

Node Version: Version: 5.24.0

Entity Sync Status

issue | Search for entity (exact match)

Next: Check the current SyncRoom **Limitations**.

ON THIS PAGE

Security

Pricing and Licensing

Resources

Subscribe for a weekly Exalate hack

Academy

Blog

YouTube Channel

Ebooks

Still need help?

Join our Community

Visit our Service Desk

Find a Partner