

Main Features

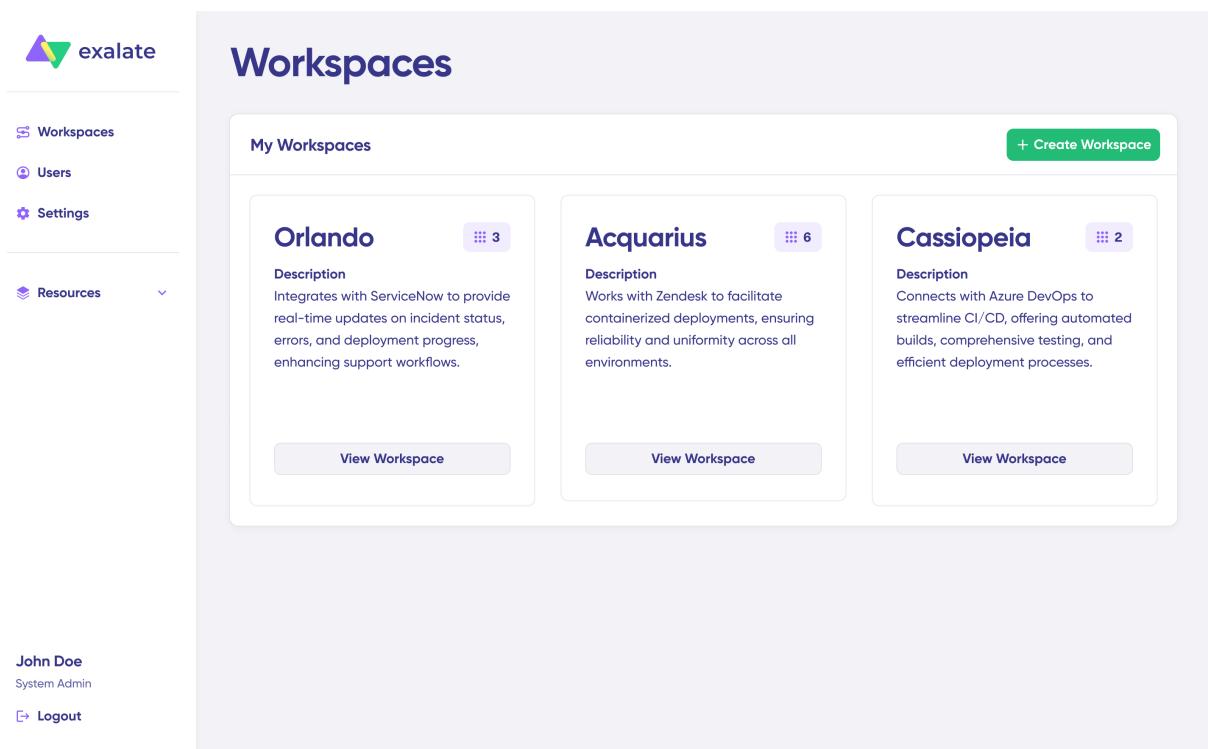
Last Modified on 01/15/2026 9:26 am EST

Formerly SyncRoom - now part of the Exalate Console Early Access.

Exalate Console provides a range of features to help streamline connection and node management:

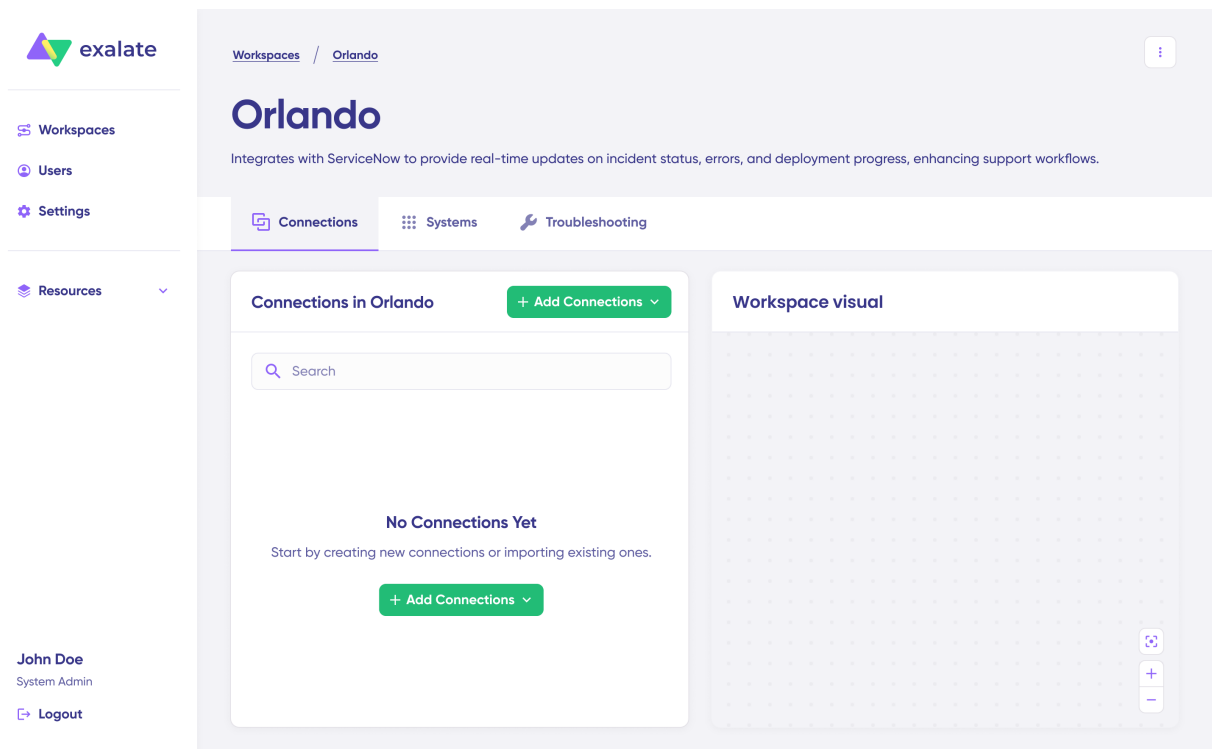
Create Workspaces

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



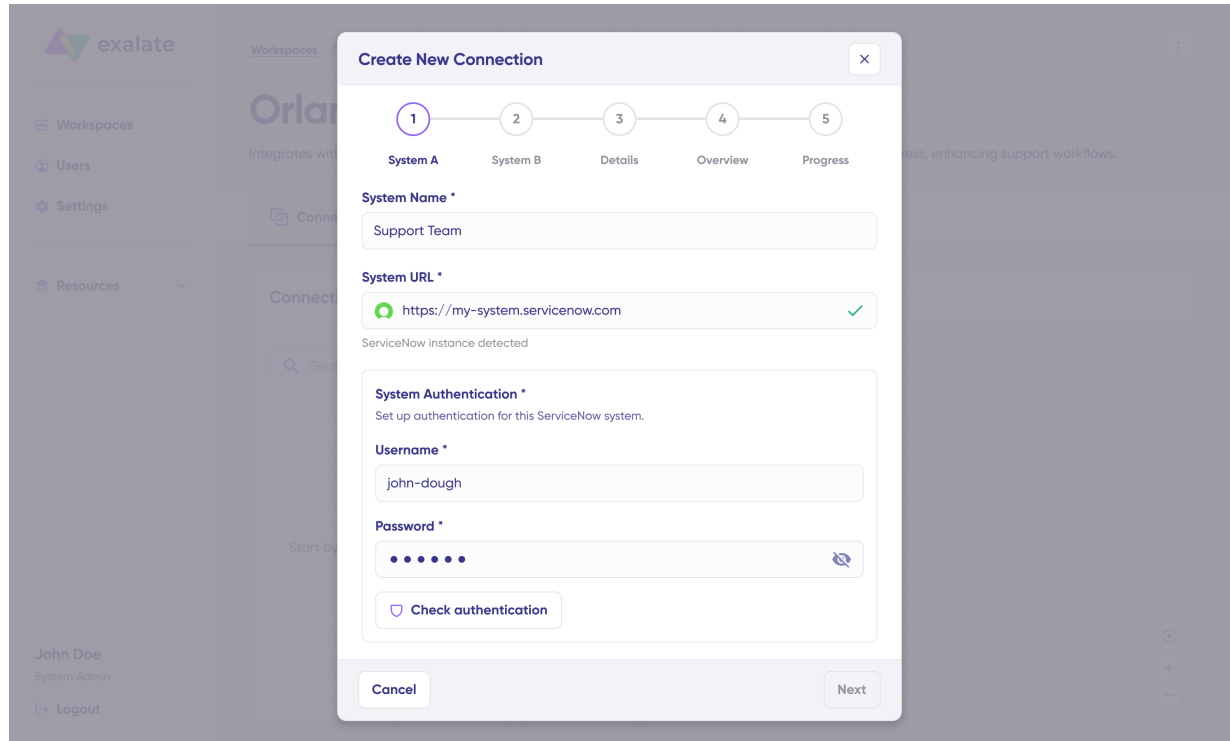
Import & Manage Connections

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



Streamlined Connection Creation

Create connections between systems without manual node installation or accessing each system individually. A smooth, guided wizard experience connects two systems in minutes.



Configure Scripts and Triggers with Versioning

You can create multiple configuration version drafts and publish a version from this Exalate Console. 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.

The screenshot shows the Exalate interface for configuring a connection between two teams: Dev Team and Support Team. The interface is divided into several sections:

- Header:** A notification bar at the top states, "You're looking at an active version of this connection. To edit, create a new version or select an existing draft."
- Breadcrumbs:** "Workspaces / Orlando / Support Team to Dev Team"
- Connection Name:** "Connection: Support Team to Dev Team"
- Team Diagram:** A visual representation of the connection: Dev Team (blue diamond) → Support Team (green circle).
- Version Control:** "Version 3 (Active)" with buttons for "+ New version" and "Open latest draft".
- Triggers:** A table listing triggers for different item types.

| Item type | Query | Status | Actions |
|-----------|---------------|--------------------------|------------|
| Issue | Project = EXA | <input type="checkbox"/> | Edit, More |
| Sprint | Project = SUP | <input type="checkbox"/> | Edit, More |
- Scripts:** Two script editors are shown side-by-side:
 - Outgoing script (From Dev Team):**

```
1 replica.type = issue.type
2 replica.summary = issue.summary
3 replica.description = issue.description
```
 - Incoming script (Into Support Team):**

```
1 if(firstSync){
2   issue.projectKey = nodeHelper.getProjectByKey("DT")
3   // Set type name from source issue, if not found set a default
```
- Footer:** User information for John Doe (System Admin) and a "Logout" button.

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.

This screenshot shows the "Scripts" editor in Exalate, Version 2. It features a side-by-side comparison of outgoing and incoming scripts.

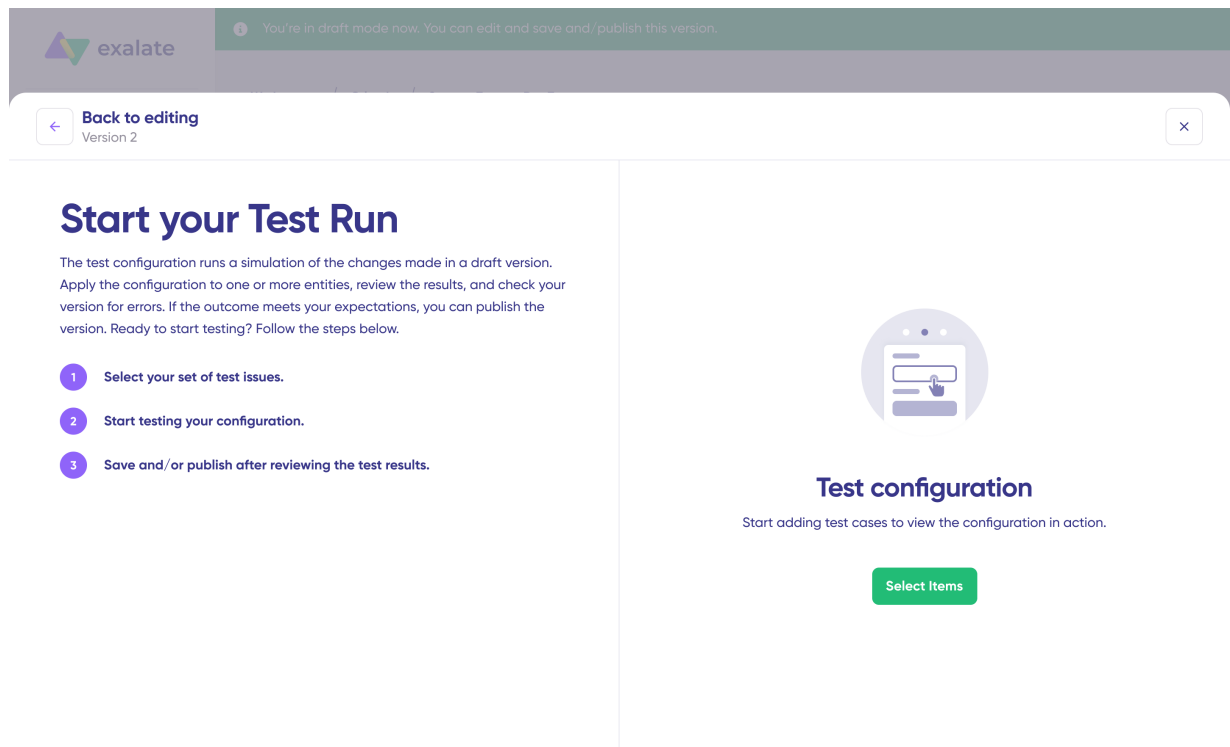
- Header:** A notification bar states, "You're in draft mode now. You can edit and save and/or publish this version."
- Buttons:** "Start Test Run" and "Save script" buttons are visible in the top right.
- Outgoing script (From Acturus):**

```
1 replica.type = issue.type
2 replica.summary = issue.summary
3 replica.description = issue.description
4 replica.priority = issue.priority
5 replica.attachments = issue.attachments
6 replica.status = issue.status
7
8 //Comment these lines out if you are interested in sending the full list of versions and
9 components of the source project.
10 replica.project.versions = [ ]
11 replica.project.components = [ ]
12
13 /*
14 Custom Fields
15
16 replica.customFields."CF Name" = issue.customFields."CF Name"
17 */
18
19
20
21
22
23
24
```
- Incoming script (Into Capella):**

```
1 if(firstSync){
2   issue.projectKey = nodeHelper.getProjectByKey("DT")
3   // Set type name from source issue, if not found set a default
4   issue.typeName = nodeHelper.getIssueTypeName(replica.type.name)
5   ?nodeHelper.getIssueTypeName("Task")
6   issue.summary = replica.summary
7   issue.description = replica.description
8   issue.comments = commentHelper.mergeComments(issue, replica) {
9     it.executor = nodeHelper.getUserByEmail(it.author?.email)
10  }
11   issue.attachments = attachmentHelper.mergeAttachments(issue, replica)
12   issue.priority = replica.priority
13   // Define the mapping between Jira Cloud statuses and
14   GitHub statuses
15   def statusMap = [ "To Do": "open",
16                     "Done": "closed" ]
17   // Apply the status mapping
18   def remoteStatusName = replica.status.name
19   issue.setStatus(statusMap[remoteForm.StatusName] ?: remoteStatusName)
20
21 /*
22 User Synchronization (Assignee/Reporter)
23
24 Set a Reporter/Assignee from the source side, if the user can't be found set a default
```
- Footer:** Two input fields for describing the scripts:
 - Outgoing: "Describe what to send from this system. e.g., Exclude attachments..."
 - Incoming: "Describe how to apply incoming data to this system. e.g., Map statuses..."

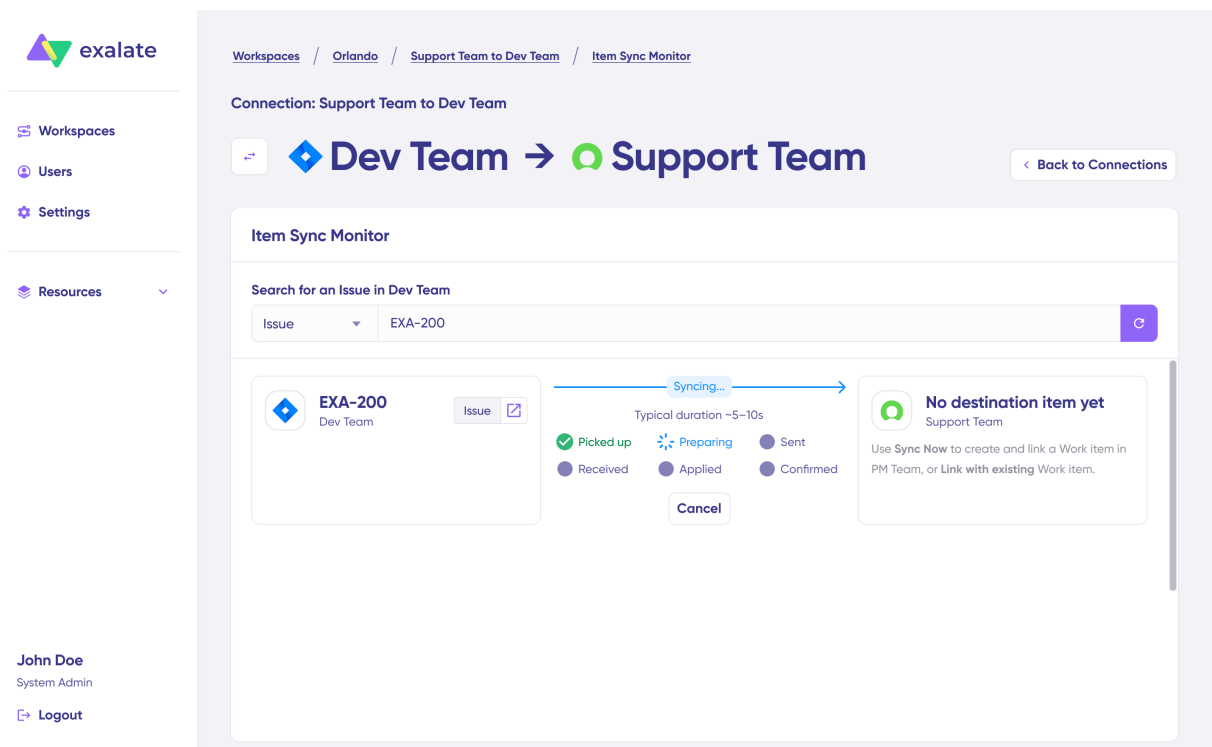
Test Run Your Configuration

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




Monitor Sync Status and synchronize manually

Check item sync status, Synchronize or Connect entities manually through the Item Sync Monitor.



AI-Powered Scripting and Troubleshooting

Get intelligent help writing sync scripts and resolving errors with AIDA, your AI assistant built into a new Exalate Console.



Workspaces

Users

Settings

Resources

John Doe

System Admin

Logout

Workspaces / Orlando

Orlando

Integrates with ServiceNow to provide real-time updates on incident status, errors, and deployment progress, enhancing support workflows.

Connections

Systems

Troubleshooting

Errors in Orlando

Search

| Occurred | Error | | | |
|----------------|---|-----------------------------------|------------|--------------------|
| Just now | Outgoing script error Cannot get property 'value' on null object Error line: 18 | Dev Team / DevTeam_to_Support | Conenction | <div>Resolve</div> |
| 25 minutes ago | Script execution failure Cannot get property 'value' on null object | Support / Support_to_CloudOps | Item | <div>Resolve</div> |
| 1 hour ago | TrackerResetException Cannot get property 'value' on null object | CloudOps / DevTeam_to_CloudOps | System | <div>Resolve</div> |

Aida Diagnosis

The error message you're seeing indicates a script-related issue: Cannot get property 'value' on null object. This means there's likely an attempt in your script to access a property on an object that doesn't exist or hasn't been initialized....

View Full Analysis

Tools

Resolve all

Next: Check the current **Limitations**.

ON THIS PAGE

- Create Workspaces
- Import & Manage Connections
- Streamlined Connection Creation
- Configure Scripts and Triggers with Versioning
- Product
- About Us
- Side by Side Script View
- Release History
- Test Run Your Configuration
- Glossary
- Monitor Sync Status and synchronize manually
- API Reference
- AI-Powered Scripting and Troubleshooting
- 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