DevOps Center

Last Updated:2026.02.25

CONTENTS

DevOps Center

Salesforce DevOps Center provides an improved experience around change and release management that brings DevOps best practices to your development team, regardless of where you fall on the low-code to pro-code spectrum. All developers can work together to deliver value to customers in a repeatable and scalable way, whether you use declarative builders, write code in Salesforce Extensions for VS Code or Agentforce Vibes IDE, leverage the power and flexibility of Salesforce CLI, or all three.

REQUIRED EDITIONS

Available in: Lightning Experience in **Enterprise**, **Performance**, **Professional**, **Unlimited**, and **Developer** Editions
Available in: **Government Cloud Plus** as interoperable. Turning on DevOps Center in Government Cloud Plus orgs can send data outside the authorization boundary. Contact your Salesforce account executive for more details.
Not available in: **EU Operating** Zone. EU Operating zone is a special paid offering that provides an enhanced level of data residency commitment. DevOps Center *is* supported in orgs in the EU that aren't part of EU OZ, per standard product terms and conditions.

DevOps Center lets you choose whether to manage your releases from its point-and-click interface, or directly from the source control system, or a combination of both. Under the hood, we manage the source control branches so developers and builders can focus on development tasks.

Because all changes are captured in a source control system, you have a single source of truth for configuration and code, which improves collaboration across all functions: admins, developers, release managers, QA, and other business stakeholders.

Are you looking for an alternative to change sets based on modern development best practices? We thought so.

Check out this video, which explains how to securely manage and release changes in Salesforce using sandboxes and DevOps Center.
Watch the video: https://play.vidyard.com/o8du8LnFGjHrwoxwnQ2vcT

DevOps Center is available in two versions-DevOps Center (Managed Package) and next generation DevOps Center (Beta). While the change and release management features remain the same, the next generation of DevOps Center is built directly on the core Salesforce platform.

Next Generation DevOps Center (Beta)

Next Generation DevOps Center (Beta) is the new and improved DevOps Center, built with AI-powered capabilities and a modern, extensible UI. It delivers significant improvements over the current DevOps Center (Managed Package). If you're new to DevOps Center, start with DevOps Center (Beta). See Set Up DevOps Center (Beta).

Already Using DevOps Center (Managed Package)?

If you've already set up DevOps Center using the managed package, you can continue using it. However, we recommend you to explore next generation DevOps Center (Beta) to take advantage of AI-powered assistance and the improved user experience. See Salesforce Help: Install and Configure DevOps Center for details.

One Collaboration Tool to Support a Variety of Roles

Working on Salesforce customizations requires a variety of tasks by teams large and small. The degree of role specialization varies, and team members frequently have more than one role over the course of a release. The team roles we recommend include:

- Team Manager or Project Manager
- Salesforce admin
- Low-code developer (Builder)
- Pro-code developer
- Release Manager
- Business owner
- Environments manager
- Quality Assurance Specialist

To ensure that team members have access to the features they require to do their work, we provide

several DevOps Center permission sets. Your Salesforce admin (or team manager) assigns you the appropriate permissions to access DevOps Center functionality.

If you perform multiple roles and don't have access to functionality required to do your job, talk to your team manager.

Provide General Feedback

If you want to provide general feedback, request product enhancements, start discussions with other DevOps Center users or the product team, and share best practices, use the DevOps Center Trailblazer group.

Next Generation DevOps Center (Beta)

AI-powered, next-generation DevOps Center is built on principles similar to the existing DevOps Center and supports modern DevOps best practices through a click-based user interface (UI). It serves as a central hub for development and tracks all changes in a source control system.

Get Started with DevOps Center (Managed Package)

After your team manager or Salesforce admin has installed and configured DevOps Center, you'll be notified that the DevOps Center org is ready for you to log in. As part of the setup process, they created one or more projects, source control project repositories, and work items associated with each project. They also added each team member's source-tracked development environment.

Conflict Detection and Resolution

Conflicts can occur when you're working with multiple development environments and moving changes between multiple pipeline stages. Conflicts can take different forms, and DevOps Center provides functionality to help you identify potential conflicts early and resolve them.

DevOps Center MCP Tools

Accelerate your application lifecycle with DevOps Center MCP tools, built for DevOps Center. The Salesforce DX MCP server provides a set of DevOps Center tools that enable Large Language Models (LLMs) to securely and autonomously read, manage, and operate DevOps Center resources.

Troubleshoot DevOps Center Errors

Here are some tips if you encounter issues while using DevOps Center. If needed, get help from your team manager or Salesforce admin to resolve these issues.

See Also

Salesforce Help: Install and Configure DevOps Center
DevOps Center Developer Guide
Salesforce Help: Assign the DevOps Center Permission Sets

Next Generation DevOps Center (Beta)

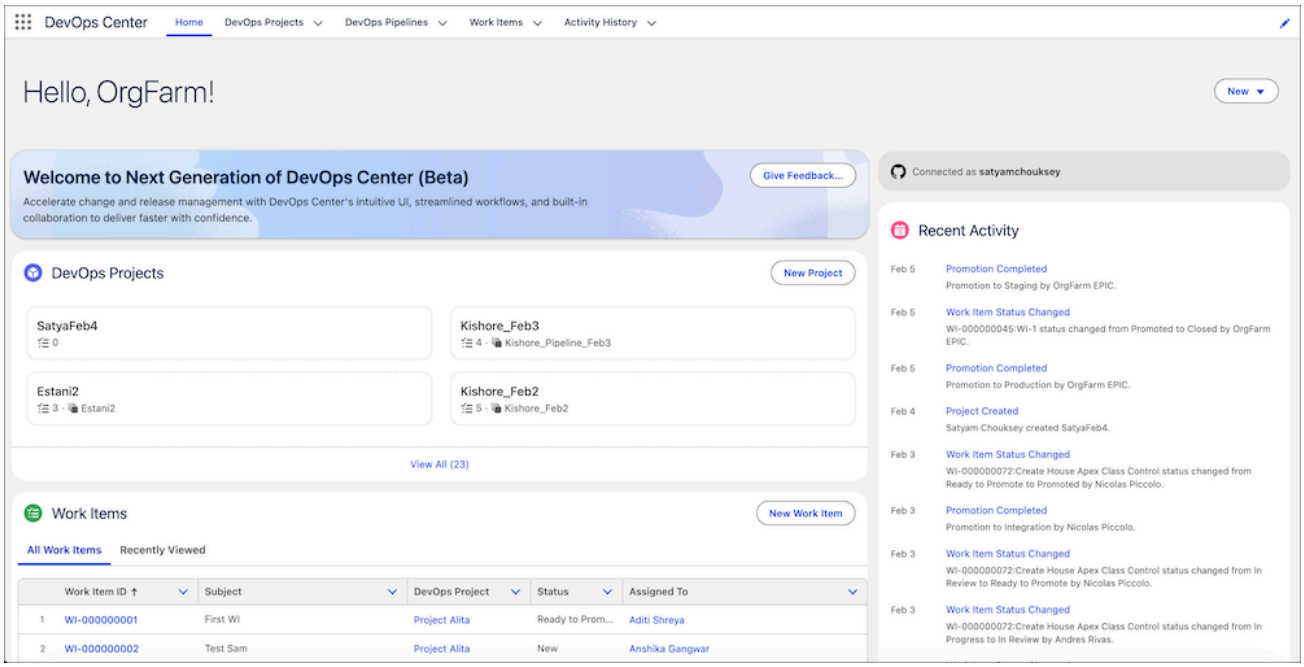
AI-powered, next-generation DevOps Center is built on principles similar to the existing DevOps Center and supports modern DevOps best practices through a click-based user interface (UI). It serves as a central hub for development and tracks all changes in a source control system.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Note DevOps Center is a pilot or beta service that is subject to the Beta Services Terms at Agreements - Salesforce.com or a written Unified Pilot Agreement if executed by Customer, and applicable terms in the Product Terms Directory. Use of this pilot or beta service is at the Customer's sole discretion.

DevOps Center creates a single source of truth and improves collaboration among developers, admins, release managers, and other stakeholders. It uses Salesforce DX MCP Server and DevOps Center MCP tools to provide AI-based recommendations for end-to-end work item management and conflict resolution.



Enhancements Over the Current DevOps Center

While the current DevOps Center provides essential capabilities for pipeline, change, and release management, the next generation DevOps Center introduces these capabilities:

- AI-powered assistance and resolution: Use natural language prompts to resolve complex DevOps issues, such as merge conflicts. This experience is powered by DevOps Center MCP tools in Salesforce DX MCP Server.
- Ease of use: DevOps Center doesn't require any package downloads and installs. After you turn on DevOps Center in your DevOps Center Hub org, you can instantly start using it.
- Modern and extensible UI: DevOps Center has a new, intuitive, and extensible UI built on Salesforce Lightning Design System (SLDS). This allows teams to customize DevOps Center with a flexible UI framework, advanced theming options, and integrate tools that match their specific governance models.
- Flexible environment support: Teams can build, track, and commit changes from non-source-tracked development environments. This is ideal for workflows with complex data configuration where source tracking isn't enabled.
- Integrated metadata tracking: DevOps Center integrates with DX Inspector to streamline tracking, committing, and reviewing metadata changes directly from your development orgs. This integration provides greater visibility and control over what's being promoted through your pipeline.

Provide Feedback

If you want to provide general feedback, request support or product enhancements, use the DevOps Center GitHub repository.

Application Lifecycle Management Journey with DevOps Center
Here's a high-level overview of your Application Lifecycle Management (ALM) journey. Start by setting up DevOps Center, then build your apps using DX Inspector, and finally, deploy your changes with either DevOps Center or metadata and data deployment feature.

Considerations to Set Up Next Generation DevOps Center (Beta)
Review these considerations before you set up next generation DevOps Center.

Set Up DevOps Center (Beta)
Configure DevOps Center to manage your release management lifecycle. Review the prerequisites, turn on the feature, manage users, and set up your projects and pipelines.

DX Inspector
Connect your development environment directly to your DevOps Center project to manage work items and track changes without leaving your org.

Review Changes with Team Members
After you open a pull request in your source control repository, your team members and collaborators can add comments about the changes.

Promote Work Items
Move your work items through a predefined sequence of pipeline stages, guiding them from development to production.

Custom Promotion
Select and promote individual work items that are ready for the next pipeline stage using custom promotion.

Keep Your Development Environment in Sync
Use back sync to update your development environment with the latest changes from the pipeline, which helps maintain compatibility and minimize merge conflicts.

Version Control System Synchronization
Maintain consistency between your pipeline and your source control repository by detecting external changes and merges.

DevOps Center Events in Activity History
View a comprehensive timeline of key DevOps events, including commits, promotions, and synchronizations, to audit activity and troubleshoot errors.

See Also
- Lightning Design Systems 2
- DevOps Center MCP Tools
- DX Inspector

Application Lifecycle Management Journey with DevOps Center

Here's a high-level overview of your Application Lifecycle Management (ALM) journey. Start by setting up DevOps Center, then build your apps using DX Inspector, and finally, deploy your changes with either DevOps Center or metadata and data deployment feature.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To set up DevOps Center:	DevOps Center Admin
To commit and review changes:	Customize Application in your development org and DevOps Center User in DevOps Center Hub org
To promote work items through the pipeline:	DevOps Center Deployment Manager

Set Up DevOps Center

DevOps Center is central to your ALM journey. If you're a new user, follow these steps to get started.

1. Turn on DevOps Center.
   See Enable Next Generation DevOps Center.
2. Add team members and assign the required user permissions.
   See Add Team Members as Users in the DevOps Center Hub Org and Assign DevOps Center Permission Sets.
3. Set up your source control.
   See Set Up Your Source Control.
4. Create a project.
   See Create a DevOps Center Project.
5. Build your pipeline.
   See Build Your Pipeline.
6. Create and assign a work item.
   See Create and Assign Work Items.
7. (Optional) Integrate Salesforce Agentforce Vibes IDE.
   See Integrate Agentforce Vibes IDE with DevOps Center.
8. Set up Salesforce DX MCP Server.
   See Access the DevOps Center MCP Tools.
   Interested in DevOps Center MCP tools in Salesforce DX MCP Server? You can use DevOps Center MCP tools to manage your DevOps Center tasks with natural language prompts. For example, state what you want to do, such as “Create a new feature branch for WI-0042.” See DevOps Center MCP Tools.

Work in Your Development Org

Automatically track app building and configuration changes in your development org using DX Inspector. DX Inspector provides quick access to metadata change tracking and visualization. It’s available at the top of a page or builder in your Salesforce sandbox or scratch org.

Note Committing changes and creating change requests are now available directly within your development environment instead of DevOps Center. This allows you to manage your metadata changes without leaving your development org.

1. Build apps, or make changes directly in your development org.
   Your changes are automatically tracked via source tracking and are listed on the Change Management page in DX Inspector. See Develop and Deploy with DX Inspector.
2. Commit or deploy changes.
   - With DevOps Center connection:

   a. Commit changes: Associate your changes with a work item and commit them to the source control repository.
      See Commit Changes to Source Repository.
   b. Start review: On the Change Management page in DX Inspector, start a review process.
      See Create a Change Request.
   c. Team collaboration: After your team member reviews and approves the work item, the work item moves into Ready to Promote status.
      See Review Changes with Team Members.

- Without a DevOps Center connection, solo Salesforce admins or smaller teams can move metadata and data together for direct deployment to a target org. See Deploy Metadata and Data from DX Inspector.

Promote Work Items in DevOps Center

After the work item is approved and ready, you can move it through your pipeline in DevOps Center.

1. When ready, promote the work items to the next stage in your pipeline.
   See Promote Work Items.
   Let the DevOps Center MCP tools in Salesforce DX MCP Server manage the promotion for you. For example, use the prompt, “Promote DevOps Center WI-0042 to the UAT stage.” Salesforce DX MCP Server runs all the necessary steps for pipeline deployment.
2. In case of merge conflicts, instead of resolving conflicts manually, you can use DevOps Center MCP tools in Salesforce DX MCP Server to simplify the process.
   Describe the outcome in natural language prompts, and Salesforce DX MCP Server analyzes and resolves the merge conflicts for you. See Resolve Merge Conflicts.

Considerations to Set Up Next Generation DevOps Center (Beta)

Review these considerations before you set up next generation DevOps Center.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Enable DevOps Center from Setup. This org, where DevOps Center is enabled, is called the DevOps Center Hub org. You can access DevOps Center from the App Launcher, then connect it to any environment that you use for development, testing, and final release.

- Turn on only one product at a time: either DevOps Center (Beta) or DevOps Center Managed Package. To enable DevOps Center (Beta), you must first disable DevOps Center Managed Package.
- Only a DevOps Center Admin can set up DevOps Center.
- You can't set up DevOps Center in a sandbox. Sandboxes can be connected to your pipeline as development or testing environments, but the DevOps Center Hub org itself can’t be a sandbox. See Evaluate DevOps Center in a Non-Production Org.
- Connect your development orgs, pipeline environments, and release orgs from within DevOps Center.
- Use GitHub.com cloud-based plans, including GitHub Enterprise Cloud, for source control.

See Also

Enable DevOps Center
Assign DevOps Center Permission Sets

Set Up DevOps Center (Beta)

Configure DevOps Center to manage your release management lifecycle. Review the prerequisites, turn on the feature, manage users, and set up your projects and pipelines.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To set up DevOps Center:	DevOps Center Admin

Before you begin, review the set up considerations and make sure that you have these resources:

- A GitHub account. See Creating an account on GitHub.
- (Optional) To use DevOps Center MCP tools in the Salesforce DX MCP Server:
  - An MCP-enabled IDE, such as Agentforce Vibes IDE or VS Code.
  - A user-facing MCP client, such as the Agentforce Vibes Extension.

1. Turn on DevOps Center (Beta) in your org and accept the terms and conditions.
   See Enable Next Generation DevOps Center.
2. Add team members as users in the DevOps Center Hub org.
   See Add Team Members as Users in the DevOps Center Hub Org.
3. Assign the required user permissions to your team members based on their roles.
   See Assign DevOps Center Permission Sets.
4. Set up and configure your source control and add team members as collaborators to the repository.
   See Set Up Your Source Control.
5. Create a DevOps Center project to serve as the workspace for your team's changes.
   See Create a DevOps Center Project.
6. Plan, build, and activate your pipeline.
   See Build Your Pipeline.
7. Create and assign the first work items to start tracking changes.
   See Create and Assign Work Items.
8. (Optional) Extend DevOps Center capabilities by integrating Agentforce Vibes IDE.
   See Integrate Agentforce Vibes IDE with DevOps Center.
9. (Optional) Set up Salesforce DX MCP Server.
   See Access the DevOps Center MCP Tools.

Enable DevOps Center

Turn on next generation DevOps Center in your Salesforce org to provide your teams with a collaborative release management tool that helps them adopt DevOps best practices.

Switch from DevOps Center Managed Package to Next Generation DevOps Center

If you currently use the DevOps Center managed package and want to try the next generation version, follow these steps to update your connection.

Add Team Members as Users in the DevOps Center Hub Org
Add any team members who need access to DevOps Center Hub org. For each team member, specify the appropriate license and profile based on their role.

Assign DevOps Center Permission Sets
Grant team members access to DevOps Center features and workflows by assigning the appropriate permission sets.

Set Up Your Source Control
Configure your GitHub repository to store project files and manage team access.

Create a DevOps Center Project
Create a project to serve as the central workspace for your team's change management collaboration.

Build Your Pipeline
Define the sequence of stages for your release lifecycle by building a pipeline.

Create and Assign Work Items
Use work items to track the progress of changes related to specific objectives, such as user stories or issue fixes. Work items help teams manage changes and releases by providing visibility into the status and progress of changes.

Integrate Agentforce Vibes IDE with DevOps Center
Agentforce Vibes IDE provides a modern developer experience and comes with everything you need to work with Salesforce DX MCP Server and DevOps Center tools. We recommend setting up Agentforce Vibes IDE if you want to experience the capabilities of the MCP server.

Enable DevOps Center

Turn on next generation DevOps Center in your Salesforce org to provide your teams with a collaborative release management tool that helps them adopt DevOps best practices.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To turn on DevOps Center:	DevOps Center Admin

Note Turn on only one product at a time: either DevOps Center (Beta) or DevOps Center Managed Package. To enable DevOps Center (Beta), you must first disable DevOps Center Managed Package.

If you use DevOps Center managed package, complete additional steps to associate your development environment with next generation DevOps Center. See Switch from DevOps Center Managed Package to

Next Generation DevOps Center.

1. From Setup, in the Quick Find box, enter *DevOps Center*, and then select **DevOps Center**.
2. Turn on **DevOps Center (Beta)**, and then review the DX Tools terms and conditions.
   This unified agreement also applies to other Salesforce DX tools including Agentforce Vibes IDE, DevOps Center, DX Inspector, and Scale Center.
3. Click **Accept**.
   DevOps Center is now enabled and available to users in your org who have the View All Data permission.

Switch from DevOps Center Managed Package to Next Generation DevOps Center

If you currently use the DevOps Center managed package and want to try the next generation version, follow these steps to update your connection.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To turn on DevOps Center:	DevOps Center Admin

If you're a new user, see Enable DevOps Center.

1. From Setup, in the Quick Find box, enter *DevOps Center*, and then select **DevOps Center**.
2. In the DevOps Center Setup page, turn off **DevOps Center Managed Package**.
3. Turn on **DevOps Center (Beta)**.
4. In your development environment, from DX Inspector, go to the Change Management page.
5. Click the dropdown arrow icon and select **Remove DevOps Center Connection**.

6. From the banner, click **Sign in to DevOps Center**.
7. In the window, enter your next generation DevOps Center credentials.

Add Team Members as Users in the DevOps Center Hub Org

Add any team members who need access to DevOps Center Hub org. For each team member, specify the appropriate license and profile based on their role.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To add team members to the DevOps Center Hub org:	DevOps Center Admin

Assign these minimum required licenses and profiles to team members based on their roles in DevOps Center.

Role	License	Profile
DevOps Center Admin, DevOps Center Deployment Manager (anyone who manages and adds environments to DevOps Center)	Salesforce	Standard User
Team Members	Salesforce	Standard User

1. Log in to the DevOps Center Hub org.
2. From Setup, in the Quick Find box, enter *Users*, and then select **Users**.
3. Click **New User** or **Add Multiple Users**.
4. Select the appropriate license type and profile based on the user's role.
5. Select **Generate passwords and notify user via email**.
6. Save your changes.

The team members receive an invitation. We recommend asking them to wait for your confirmation before logging in. Their access to DevOps Center is limited until you complete the setup.

Assign DevOps Center Permission Sets

Grant team members access to DevOps Center features and workflows by assigning the appropriate permission sets.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To assign DevOps Center permission sets:	DevOps Center Admin

Assign permission sets to everyone working on your project. When planning assignments, determine who must change project-level settings (such as adding environments) and who only needs access to work items.

Refer to this table to determine which permission sets to assign to your team members.

Permission Set	Description	Recommended Assignment
DevOps Center User	The base permission set. Provides data access and permissions to view pipelines and manage work items. This set does not overlap with the Admin set. Admins need this permission set, too.	All DevOps Center users, including DevOps Center admins.
DevOps Center Admin	Provides data access and permissions to set up projects, environments, and users.	Team managers, or project managers.
DevOps Center Deployment Manager	Provides permissions to promote changes through the pipeline stages.	Deployment managers and any team members authorized to promote changes.
Access DevOps Center Named Credentials	Provides access to the named credentials needed to authenticate to environments. Created and maintained automatically by DevOps Center.	All DevOps Center users

1. From Setup, in the Quick Find box, enter *Permission Sets*, and then select **Permission Sets**.
2. Select the **DevOps Center** permission set.
3. Click **Manage Assignments**, and then click **Add Assignments**.
4. Select the checkboxes next to the names of the users you want assigned to this permission set, and click **Assign**.
5. Click **Done**.

Repeat these steps to assign additional permission sets to users as needed. For example, a release manager needs both the DevOps Center User and DevOps Center Admin permission sets.

Set Up Your Source Control

Configure your GitHub repository to store project files and manage team access.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To set up a source control:	DevOps Center Admin

Structure the repository as a Salesforce DX project. You can identify a Salesforce DX project by the sfdx-project.json file in the root directory. Create the required repository in these ways:

- DevOps Center automatically creates the repository when you create a new pipeline.
- Use an existing repository or create one manually from the dx-empty template.

As a best practice, configure branch protection rules for branches like main, staging, and integration to prevent the source control from deleting them after a merge. For existing repositories, verify that protection rules are active and make sure they prevent the deletion of stage branches after a change request merge.

Each DevOps Center pipeline requires its own GitHub repository to act as the single source of truth. DevOps Center uses the GitHub REST API and OAuth 2.0 to ensure secure communication.

Add a Team Member to the Project Repository

If you're a team manager or Salesforce admin, complete these steps:

1. Ask each team member to create a GitHub account (if they don't have one) and to send you their GitHub username.
2. Log in to GitHub and update the project repository settings for access to invite the team member as a collaborator.
3. Confirm that the team member received the email invitation from GitHub and accepted it.

Join a Project Repository as a Collaborator

If you're a team member, complete these steps:

1. Create a GitHub account if you don’t have one.
2. Send your GitHub username to the team manager or Salesforce admin who’s setting up DevOps Center.
3. Check your email for an invitation to collaborate on the repository, and then accept it.
   If you haven’t received an email invitation, contact the person who set up DevOps Center.

Grant Access to Organization-Owned GitHub Repositories

Request access to GitHub repositories owned by an organization to make them available for your DevOps Center pipelines. GitHub repositories owned by an organization aren’t visible in DevOps Center until an organization account owner grants access. If you don't see your organization's repositories when setting up a pipeline, you must request access through your personal GitHub.

1. Log in to your personal GitHub account, and go to Settings.
2. Click **Applications**, then select Authorized OAuth Apps.
3. Click **Salesforce DX**.
4. Find the organization that owns the repository, then click **Request**.
5. Repeat the steps for other Salesforce integration applications.
   The GitHub repository owner receives a notification to approve your request.

After the GitHub repository owner approves the request, go to DevOps Center, and start creating your project.

Create a DevOps Center Project

Create a project to serve as the central workspace for your team’s change management collaboration.

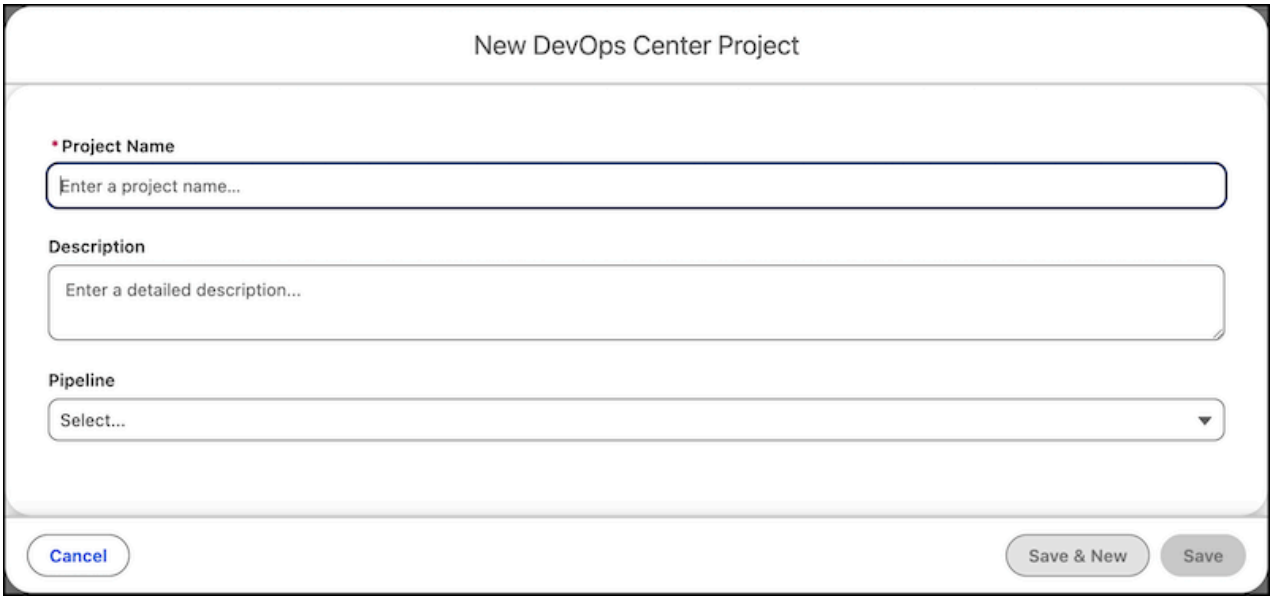
REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To create a project:	DevOps Center Admin

A DevOps Center project is the primary workspace where your team collaborates to manage changes for an application or customization. Each project is associated with a single pipeline.

1. From the DevOps Projects tab, click **New**.



2. Enter a unique name and a description to identify the purpose of the project.
3. Select a pipeline to associate with this project.

   Note If you haven't created and activated a pipeline yet, you can do so later. See Build Your Pipeline.

4. Save your project.

See Also

Install and Configure DevOps Center: Manage DevOps Center Projects

Build Your Pipeline

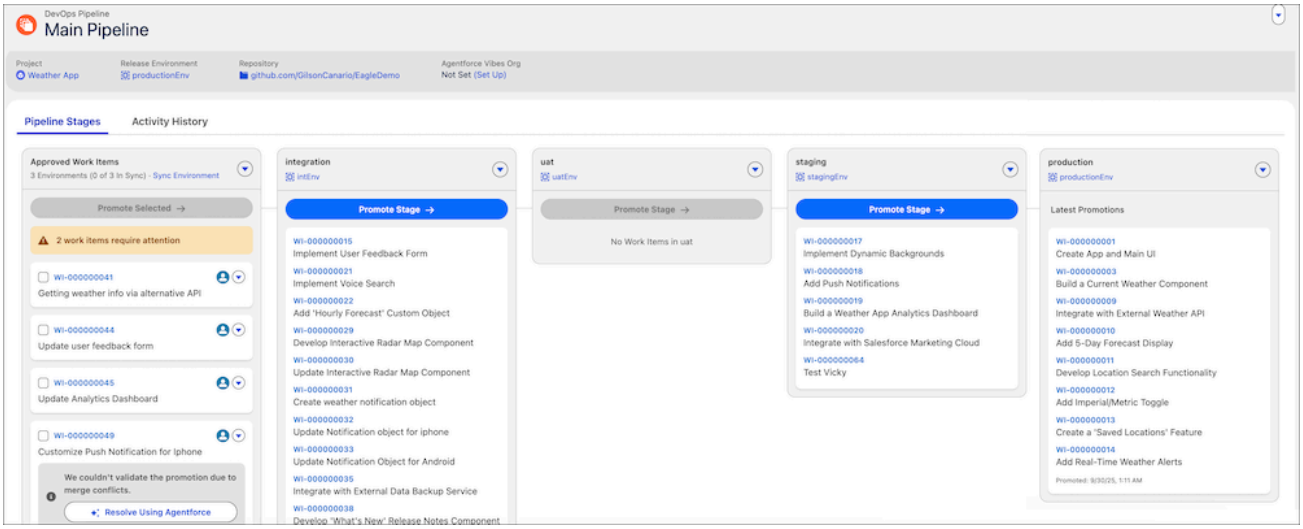
Define the sequence of stages for your release lifecycle by building a pipeline.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To build a pipeline:	DevOps Center Admin

A pipeline defines the sequence of stages that work items progress through as they go through the release lifecycle from development to production (or some other final release stage). For example, Dev → Integration → UAT → Staging → Production. The pipeline consists of pipeline stages. Each pipeline stage corresponds to an environment (currently a Salesforce org) and a branch in your source control repository.



Important Plan your pipeline carefully. You can't modify the pipeline after you promote changes through it.

The number of pipeline stages depends on your development and business processes. Besides the production and development environments, the default pipeline includes Integration, User Acceptance Testing (UAT), and Staging stages. If your team works in sandboxes and scratch orgs, you can connect them to these stages.

1. From the DevOps Pipelines tab, click **New**.
2. Enter a name for your pipeline.
3. Select your source control system.
   If you aren't authenticated to the source control system, follow the prompts to log in.
4. (Optional) Create a repository or select an existing one.
5. Click **Next**.
6. Select the projects to associate with this pipeline, and click **Create Pipeline**.
   Your default pipeline is ready.
7. For each stage in the new pipeline, complete these steps to finish the setup:
   a. Connect your environment.

      Note DevOps Center pipelines require a release environment (often your production org) and one development environment. Make sure that all team members are users in these environments if they require access to deploy within DevOps Center.

   b. Create a branch in the source control repository for this stage, or use an existing branch.
8. To associate a project with the pipeline, complete these steps:
   a. Click the icon next to the Activate Pipeline button.
   b. From the list, select **Edit Project Connections**, and associate a project with the pipeline.
9. Click **Activate Pipeline** to turn on the pipeline.

See Also

Install and Configure DevOps Center: Plan Your Pipeline

Create and Assign Work Items

Use work items to track the progress of changes related to specific objectives, such as user stories or issue fixes. Work items help teams manage changes and releases by providing visibility into the status and progress of changes.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

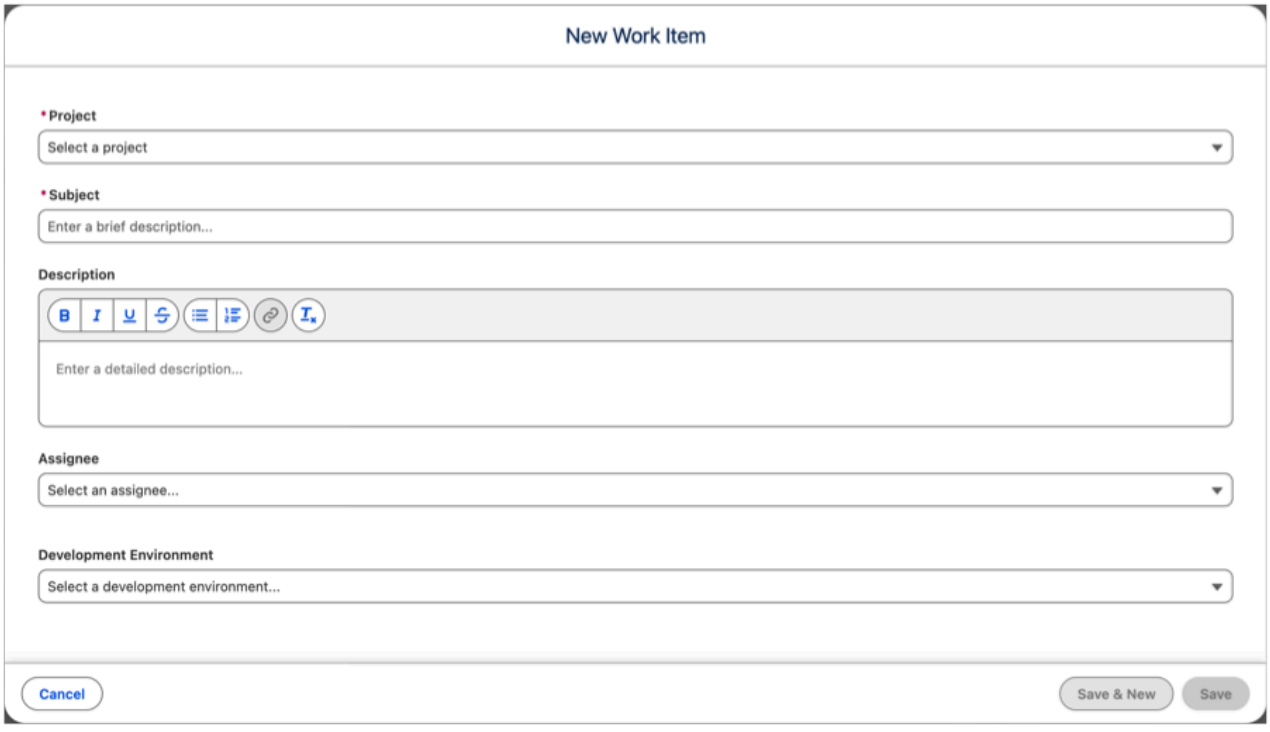
USER PERMISSIONS NEEDED	
To create, edit, and assign work items:	DevOps Center User

Before creating and assigning work items, create and activate your project pipeline. See Build Your Pipeline.

Alternatively, you can create a new work item directly from DX Inspector. See Develop and Deploy with DX Inspector.

1. From the Work Items tab, click **New**.

   You can also create a work item from the DevOps Projects or Home tab. Select the project for which you're creating the work item, then click **New Work Item**.



2. Select the DevOps Center project for which you're creating the work item.
3. In the Subject field, specify the objective or the problem.
4. In the Description field, enter additional information, including details about the changes and acceptance criteria.
   DevOps Center uses the first 255 characters of the description to identify this work item in the source control repository.
5. (Optional) Assign the work item to a team member.
6. (Optional) Select a connected development environment for this work item.
   If you don't select a development environment, the work item isn't assigned to any environment. However, when you commit this work item for the first time, the DX Inspector environment is automatically associated with it, and a branch is created. A branch is also created when the work item moves to the In Progress status. You can add a development environment from the DevOps Pipelines tab.
7. Save your work item or create another.
   The work item appears in the Work Items tab, with the status New.
8. On the Work Items tab, click **Move to In Progress** to inform your team members that you're working on this work item.

Both you and your team members can create additional work items as the project progresses.

The basic setup for DevOps Center is complete and you can start building in your development environment. See DX Inspector.

Interested in the agentic DevOps Center experience and how you can use DevOps Center MCP tools to simplify your manual tasks? First, integrate Agentforce Vibes IDE and then set up Salesforce DX MCP Server.

See Also

- Work Item Management
- Integrate Agentforce Vibes IDE with DevOps Center
- DevOps Center MCP Tools

Integrate Agentforce Vibes IDE with DevOps Center

Agentforce Vibes IDE provides a modern developer experience and comes with everything you need to work with Salesforce DX MCP Server and DevOps Center tools. We recommend setting up Agentforce Vibes IDE if you want to experience the capabilities of the MCP server.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To use Agentforce Vibes IDE:	View All Data

To set up Agentforce Vibes IDE, see Agentforce Vibes IDE.

Clone the Repository and Authorize

Clone your DevOps Center project repository into Agentforce Vibes IDE and authorize the connection to your Salesforce org.

1. Go to Setup in your Salesforce sandbox org.
2. Find and click Agentforce Vibes in the menu.
3. In Agentforce Vibes IDE, click the hamburger icon in the Activity Bar, and then select **Terminal** | **New Terminal**.
4. Go to the `/home.codebuilder/` directory.
5. Clone your DevOps Center Project GitHub repository into the Agentforce Vibes IDE environment. See Create a Project from a Repository.

   ```
   git clone https://github.com/<DevOps Center Project Path>.git
   ```

6. Enter your GitHub credentials when prompted.
7. Depending on the type of your GitHub repository, authorize and authenticate to GitHub.
   - Public repo: The folder is cloned automatically and appears in your Explorer panel.
   - Private repo with two-factor authentication (2FA) enabled: Authorize Agentforce Vibes IDE to access GitHub. Then, log in to GitHub or enter the 6-digit token shown in Agentforce Vibes IDE.
8. To switch to the new repository, click the hamburger icon in the Activity Bar and then click **File** | **Open Folder**.
9. Connect and authorize to a Salesforce org.

   ```
   sf org login web --set-default-dev-hub --alias DevHub
   ```

10. Enter your Salesforce org credentials in the browser that opens.
11. Open Agentforce Vibes Extension and accept the terms and conditions.

Select a Development Environment to Open Agentforce Vibes IDE

Define which environment opens by default when you launch the IDE. We recommend that you open Agentforce Vibes IDE from an environment that is not in production.

1. From the DevOps Pipelines tab, select your pipeline, then click **Default Vibes Org**.
2. On the Select Environment window, select the target org.
3. (Optional) If your target org isn't listed, connect to a new org and authenticate.
4. Save the environment as default.

To start using the AI capabilities in the next generation DevOps Center, set up the Salesforce DX MCP

Server. The MCP server is integrated with the Agentforce Vibes Extension, which serves as the MCP client. See Set Up Salesforce DX MCP Server.

DX Inspector

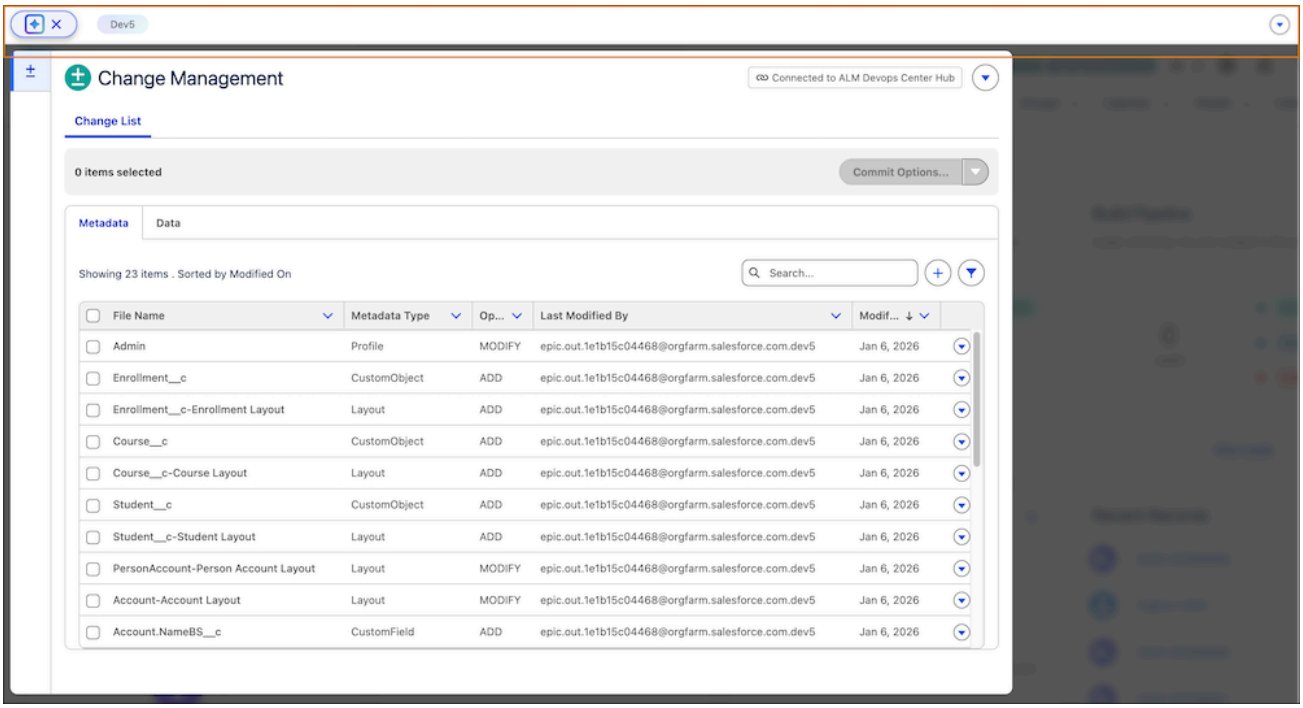
Connect your development environment directly to your DevOps Center project to manage work items and track changes without leaving your org.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

DX Inspector connects your development environment (sandbox, scratch org, or DE org) and your DevOps Center project. Instead of switching between multiple browser tabs to manage your DevOps tasks, you can track changes, create work items, commit metadata, and create change requests directly within the org where you're building.

The Change Management page in DX Inspector provides a comprehensive view of all source-tracked changes in your org. When your development work is complete, you can create a work item and commit your changes directly to the project repository. The real-time sync between DevOps Center and DX Inspector makes sure that every commit and status change is immediately reflected in DevOps Center.



When ready, you can start the peer review process by creating a change request (pull request) in DX Inspector. After the request is reviewed and approved in your source control, the work item moves to the Ready to Promote state and your deployment manager can then start the promotion process in DevOps

Center.

Add Metadata Manually
The Change Management page uses source tracking to automatically track metadata files in your org. However, when you require flexibility to include additional metadata files that source tracking hasn't pulled automatically, you can add metadata manually.

Commit Changes to Source Repository
Commit changes to your source repository directly from the Change Management page. After you review the changes, decide which changes to include for the work item and commit them to the DevOps Center project's source control repository. Seamlessly commit changes, and then go to DevOps Center to review, promote, and deploy your work items.

Create a Change Request
Start a peer review of your work item by creating a change request directly from DX Inspector.

Add Metadata Manually

The Change Management page uses source tracking to automatically track metadata files in your org. However, when you require flexibility to include additional metadata files that source tracking hasn't pulled automatically, you can add metadata manually.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

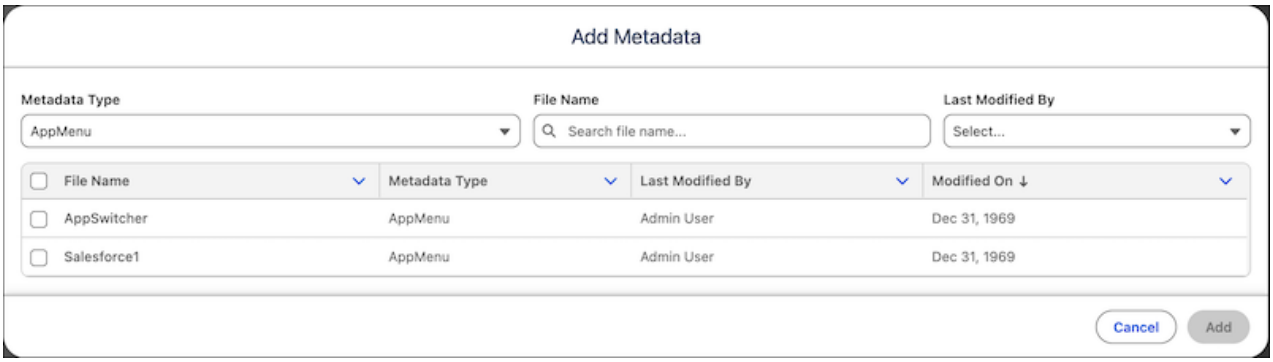
USER PERMISSIONS NEEDED	
To add metadata manually:	Customize Application in your development org

Consider adding metadata manually for these cases.

- Source tracking doesn't support the metadata file, so it isn't listed on the Change Management page. See Metadata Coverage for the list of metadata types supported by source tracking.
- Your development environment is shared, and another teammate has already committed a file that you also want to include in your work item commit. If the metadata hasn't changed in your environment since your teammate's commit, it doesn't automatically appear in your changes list.
- You want to commit files that were changed before source tracking was enabled in your environment.

Note You can't manually add metadata that's already on the Change Management page, and the original operation type (CHANGE or ADD) remains unchanged.

1. From the DX Inspector panel, open the **Change Management** page.
2. Click + .



3. Select a metadata type from the dropdown.
   Search by file name or last modified by to narrow your search results.
4. Select the metadata files that you want to add.
5. Click **Add**.

The file appears on the Change Management page with the operation type MANUAL, ready for you to include in your next commit.

Commit Changes to Source Repository

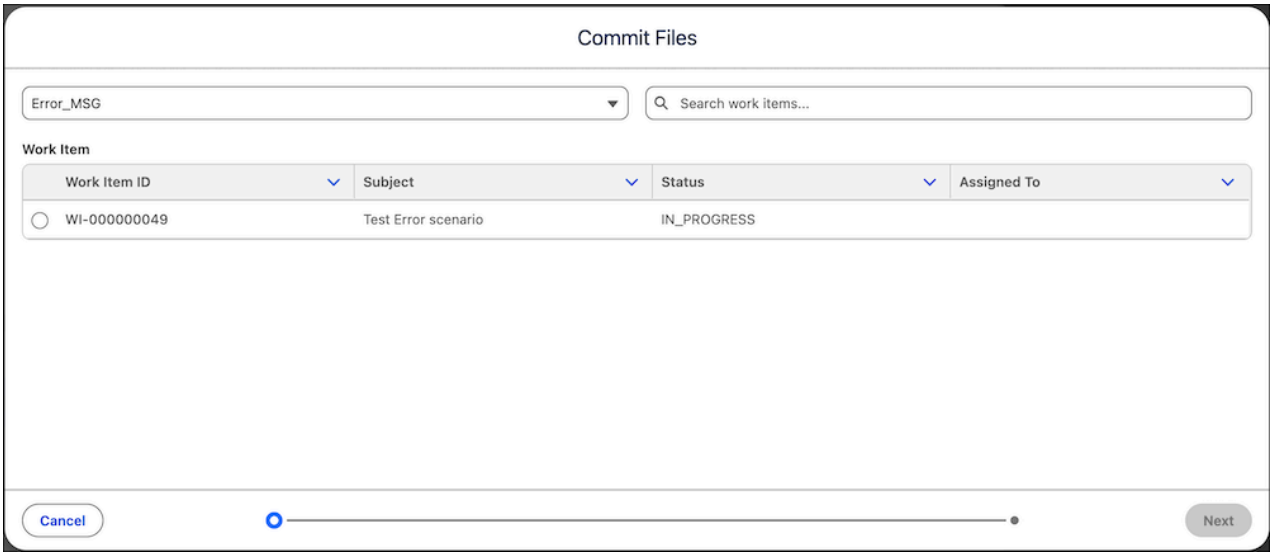
Commit changes to your source repository directly from the Change Management page. After you review the changes, decide which changes to include for the work item and commit them to the DevOps Center project's source control repository. Seamlessly commit changes, and then go to DevOps Center to review, promote, and deploy your work items.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

USER PERMISSIONS NEEDED	
To commit changes:	Customize Application in your development org
To commit changes:	DevOps Center User in DevOps Center Hub org

1. Log into your sandbox or scratch org.
2. Click on DX Inspector to open the DX Inspector panel.
3. From the DX Inspector panel, open the **Change Management** page.
4. Click , and then select **Connect DevOps Center**.
   This action authenticates the connection to the DevOps Center GitHub source repository. On successful authentication, all uncommitted changes are listed.
5. (Optional) Add metadata manually. See Add Metadata Manually.
6. Review and select the changes that you want to include for the work item. See Work Item Management.
7. Click **Commit Changes...**.



8. Select a DevOps Center project.
   Only work items in the project with the statuses In Progress, In Review, and Ready to Promote are displayed. If there are no work items, we recommend creating a new work item directly in DevOps Center.
9. Select a work item, and then click **Next**.
10. Add a commit comment that clearly describes the changes.
    This helps in identifying your commit, especially when the work item has multiple commits.
11. Review your changes, and then click **Commit**.

After a successful commit, create a review in DX Inspector.

Create a Change Request

Start a peer review of your work item by creating a change request directly from DX Inspector.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

USER PERMISSIONS NEEDED	
To create a change request:	Customize Application in your development org
To create a change request:	DevOps Center User in DevOps Center Hub org

Before you start the review process, make sure:

- Your work item is in the In Progress status.
- Commit all changes associated with the work item.

When you're ready to share your changes with project collaborators, use the **Create Review** button on

DX Inspector to start a review. Creating a review opens a change request (also called a pull request) in your source control system and moves the work item to the In Review status.

A pull request is a proposal to merge your committed changes with the main codebase. It allows team members to review, discuss, and approve the updates before they are merged.

1. From DX Inspector, click **Create Review**.
   DX Inspector opens a pull request for the changes in your work item branch and updates the work item status to In Review.
2. Click **Review Created** to open the pull request in your repository.
   You're redirected to your source control repository.

You or your team members can review and approve the changes in the source control repository.

Review Changes with Team Members

After you open a pull request in your source control repository, your team members and collaborators can add comments about the changes.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

Peer reviews help identify and fix potential issues such as bugs, coding standards violations, and performance issues.

After changes are reviewed, a team member approves the work item, and DevOps Center sets its status to Ready to Promote. Work items must be in this status to be available for promotion through the pipeline.

You can also manually mark a work item as ready for promotion. In DevOps Center, go to the Work Items tab, and click **Mark as Ready to Promote**.

The next step is to complete the promotion.

Promote Work Items

Move your work items through a predefined sequence of pipeline stages, guiding them from development to production.

REQUIRED EDITIONS

Available in: Lightning Experience

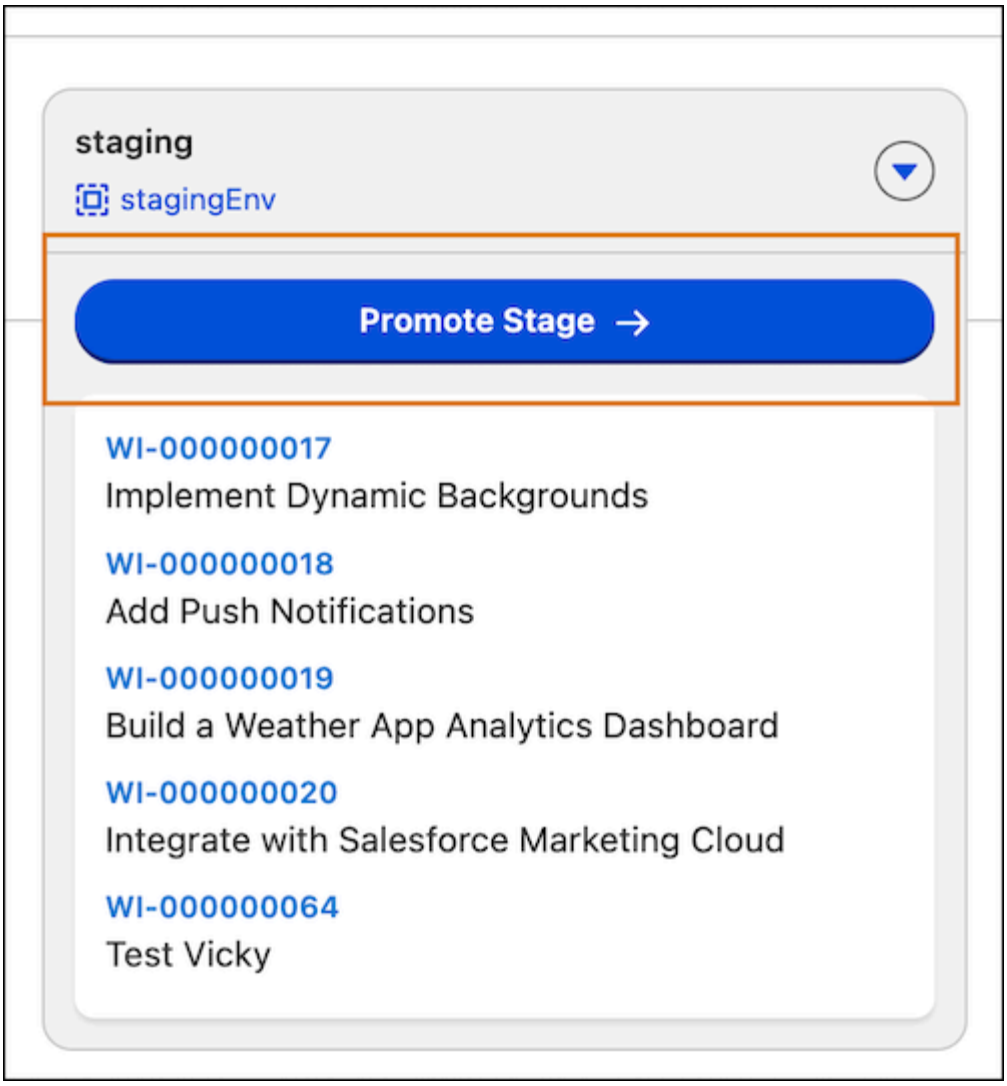
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

USER PERMISSIONS NEEDED	
To create a DevOps Center project:	DevOps Center Deployment Manager

When you promote work items, their associated changes are first merged from the current stage branch to the next, and then deployed to the org for that next stage.

DevOps Center supports only stage promotion (also known as work item bundle promotion in the current DevOps Center), which makes releases more stable and predictable. After a work item is considered ready to promote, it's added to the Approved Work Items list. You can promote individual, approved work items into the initial pipeline stage, where they are combined with other changes intended for the same release. Then, you promote the entire stage as a single, consistent unit to subsequent stages.

1. Go to the DevOps Pipelines tab and under Approved Work Items, select the work items to promote.
2. Click **Promote Selected**.
   The selected work items are promoted to the next stage and are now ready to be promoted as a unit. After all work items for a release are grouped together, promote them to the next stage in the pipeline.
3. In the pipeline view, click **Promote Stage**.
4. If you haven't logged in to the environment for the next pipeline stage from within DevOps Center, complete the login process, then click **Promote Stage**.



5. In the Promotion Options window, select which changes to promote and the Apex tests to run.
6. Click **Promote**.
   DevOps Center merge branches and deploys the work items to the environment associated with the next stage.
7. (Optional) If DevOps Center detects a merge conflict for the selected work items, it provides recommendations to resolve merge conflicts using DevOps Center MCP tools.
   See Resolve Merge Conflicts with DevOps Center MCP Tools.

Custom Promotion

Select and promote individual work items that are ready for the next pipeline stage using custom promotion.

REQUIRED EDITIONS

Available in: Lightning Experience

Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

While we recommend stage promotion for stable and predictable releases, custom promotion provides flexibility to immediately move approved work items forward, even if other work items in the same stage require additional work, such as testing.

Example You have work items, W1, W2, W3 in the Integration stage, and W1 and W2 are ready, but a related work item, W3 requires additional work. With custom promotion, you can promote work items, W1 and W2 to UAT, while W3 remains in Integration. You can promote W3 to UAT later when it's ready.

DevOps Center tracks the commit history for each work item. When you promote individual work items from a stage, DevOps Center extracts the changes specific to the selected work items. It then creates a change request that contains only the metadata from the selected work items and applies it to the target stage's branch. After the prepromotion validation completes, the selected work items move to the next stage. Remaining work items stay in the source stage and maintain their current status. This process ensures changes remain accurate while providing the flexibility needed for efficient release management.

Note Always check for dependencies and review the remaining work items in the stage for possible merge issues after promotion.

Promote Individual Work Items From a Stage
Move approved work items forward without waiting for the entire stage to be ready. Select specific work items for promotion, while others remain in their current stage until they're ready.

Promote Individual Work Items From a Stage

Move approved work items forward without waiting for the entire stage to be ready. Select specific work items for promotion, while others remain in their current stage until they're ready.

REQUIRED EDITIONS

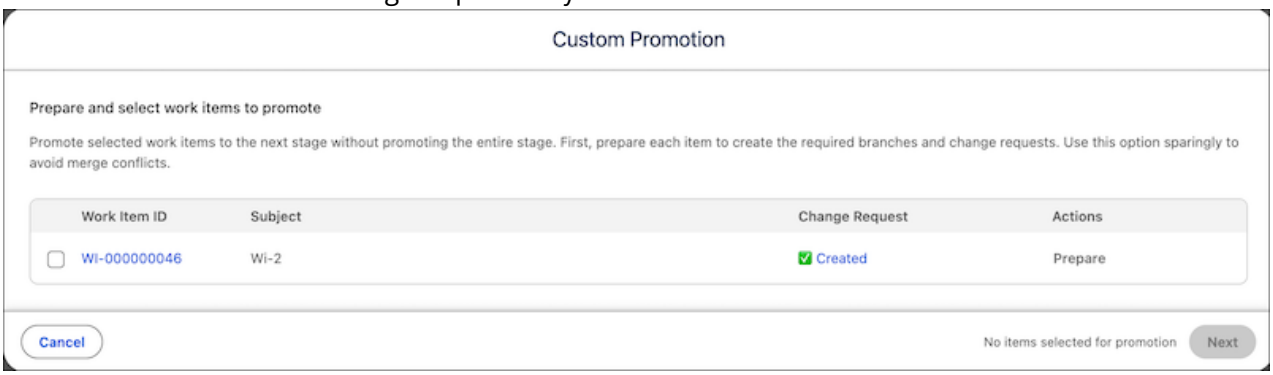
Available in: Lightning Experience

Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To build a pipeline:	DevOps Center Deployment Manager

1. Go to the DevOps Pipelines tab, and identify the stage that contains the work items you want to promote.

2. On that pipeline stage, click , and then select **Work Items to Promote**.
3. In the Custom Promotion window, click **Prepare Work Item** for each work item you want to promote. This creates a branch and change request for your work items.



4. Click **Promotion Options...**.
5. Select the changes and the Apex tests to run.
6. Click **Promote**.

Keep Your Development Environment in Sync

Use back sync to update your development environment with the latest changes from the pipeline, which helps maintain compatibility and minimize merge conflicts.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

When you work on a team with multiple developers, your individual development environment can drift out of sync with the source control repository. The back sync feature synchronizes your development environment with the first stage in your pipeline. This makes sure that you develop against the latest integrated source of truth, including changes from teammates. The process also confirms that your changes are compatible with other changes in the pipeline, which reduces potential conflicts when you promote changes.

DevOps Center tracks the differences between each development environment and the first pipeline stage's branch. When you connect a work item to a development environment, DevOps Center notifies you if the environment is out of sync. You can back sync at any time, but we recommend running a back sync whenever you begin your work or at a regular interval.

When you start a back sync, the process deploys all changes from the first pipeline stage branch to your development environment.

Note Back sync performs a full deployment that overwrites unpromoted changes in your development environment. Before you proceed, make sure your environment doesn't contain any changes that you want to keep.

Run Back Sync
Move changes from the first pipeline stage to your development environment to maintain consistency across all environments. Running a back sync makes sure that you're working with the most up-to-date, approved changes.

Run Back Sync

Move changes from the first pipeline stage to your development environment to maintain consistency across all environments. Running a back sync makes sure that you're working with the most up-to-date, approved changes.

REQUIRED EDITIONS

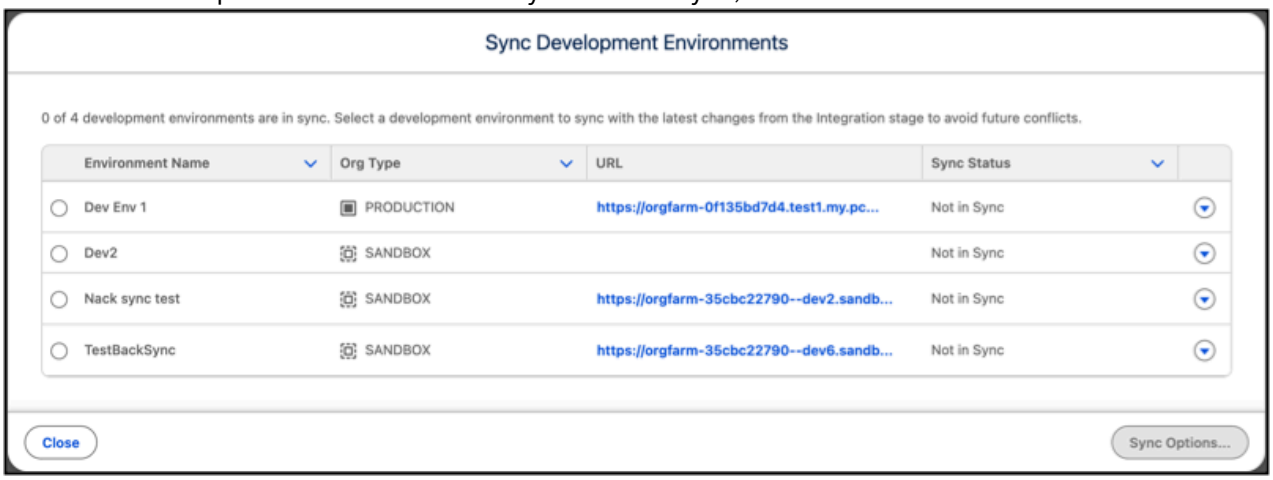
Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To run back sync:	DevOps Center Deployment Manager

Before you start, make sure that no other operations are in progress in the first stage or the development environment.

Note Back sync overwrites unpromoted work items with In Progress, In Review, or Approved statuses in the development environment. To avoid losing work, make sure your environment has no unpromoted work items before you proceed.

1. Go to the DevOps Pipelines tab, open your pipeline.
2. Under Approved Work Items, click **Sync Environment**.
3. Select the development environment that you want to sync, and then click **Next**.



4. On the Select sync options page, select which Apex tests to run.

- Default: No tests are run in sandboxes and scratch orgs.
- Run local tests: All local tests are run.
- Run all tests: All tests in your org are run.
- Run specified tests: Only the tests that you specify are run. Provide the names of test classes in a comma-separated list with no spaces. Example: TestDeleteData,TestDataFactory.createTestRecords

5. Click **Sync**.
   The synchronization process deploys all changes from the first pipeline stage branch into the development environment.

Version Control System Synchronization

Maintain consistency between your pipeline and your source control repository by detecting external changes and merges.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

The Version Control System (VCS) synchronization process maintains the version control system as the single source of truth for your work items, minimizing merge conflicts and environment inconsistencies. It makes sure that all changes, whether merged or committed directly to the source control repository, or performed using Git CLI tools remain consistent with DevOps Center.

With VCS synchronization, developers can continue to manage branches and change requests using external tools, such as Git CLI or the source control provider's UI. Meanwhile, deployment managers and DevOps Center admins maintain full visibility of those changes within the DevOps Center pipeline.

DevOps Center automatically runs VCS synchronization during user actions, such as promoting changes or viewing a work item. When you promote work items, DevOps Center checks the pipeline for external merges (changes made directly to the source control repository). In case of an external merge, DevOps Center analyzes the changes and updates the pipeline and work items with the latest changes.

DevOps Center detects specific events in the source control repository and maps them to DevOps Center actions.

VCS Event	DevOps Center Action
Create a work item branch	Maps the branch name to a work item and sets it as the development branch.
Open a change request (PR)	Updates the status of the corresponding work item or stage to indicate that a review is in progress.
Approve a change request	Marks the associated work item as Approved or

VCS Event	DevOps Center Action
	Ready to Promote.
Merge from work item to stage	Validates the merge sequence and tracks the merge.
Merge from stage to stage	Verifies that the source and target branches represent consecutive stages in the pipeline.

External Merges
DevOps Center detects external changes that you merge directly into a stage's source control branch.

External Merges

DevOps Center detects external changes that you merge directly into a stage's source control branch.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

While the external merge updates your source control repository, the changes are not yet deployed to the Salesforce environment associated with that pipeline stage. When an external merge occurs, your pipeline indicates that the stage is out of sync. To deploy the changes to the pipeline stage's associated environment, complete the promotion using the DevOps Center UI.

Note We don't recommend using external merges frequently. Use DevOps Center UI for your DevOps tasks, such as promotion.

Follow these best practices when working outside of DevOps Center:

- Use the **Squash and merge** option when merging changes directly into the source control repository. This option combines all commits into a single commit, which simplifies the history log and makes it easier to audit changes.
- If there are any merge conflicts during the external merge, resolve them using Agentforce or the source control provider's UI before DevOps Center can complete the deployment.

DevOps Center Events in Activity History

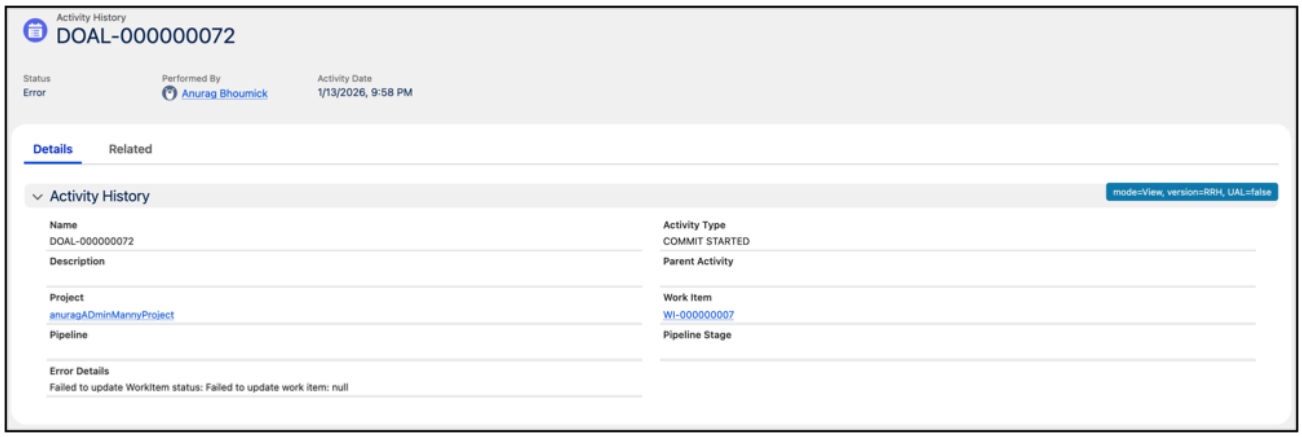
View a comprehensive timeline of key DevOps events, including commits, promotions, and synchronizations, to audit activity and troubleshoot errors.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

DevOps Center records every operation that occurs within your project pipeline. The Activity History tab provides a centralized audit trail, giving deployment managers and developers visibility into who performed an action, when it happened, and the outcome.

In the Activity History tab, DevOps Center lists activities in a table. To see granular details, such as the user who started the action or specific error messages, click the activity name. If an activity has a status of FAIL, the Error Details section within the activity record provides log information to help you pinpoint and resolve issues like merge conflicts or dependency failures.



Get Started with DevOps Center (Managed Package)

After your team manager or Salesforce admin has installed and configured DevOps Center, you'll be notified that the DevOps Center org is ready for you to log in. As part of the setup process, they created one or more projects, source control project repositories, and work items associated with each project. They also added each team member's source-tracked development environment.

Note This article provides information about the DevOps Center managed package. Starting in February 2026, Salesforce offers next generation DevOps Center (Beta). For information about the new version, see Next Generation DevOps Center.

DevOps Center Projects

A project is your team's central arena for work. The purpose of a project is to help you and your team collaborate and manage changes being developed for a particular application. A project encapsulates definitions and configurations of the many different things that managing a set of changes requires,

including:

- Work items that define the changes to be made
- A pointer to the source control repository that stores changes made for the project
- Which development environments are used to make changes (source-tracked Developer sandbox, Developer Pro sandbox, or scratch org)
- Environments used for pipelines stages, for example, integration, UAT, and staging
- A pipeline that defines how changes are deployed as they move from development to production

Basic Development Workflow

- Get ready to use a source control system.
- Open DevOps Center and select a project.
- Start on a work item.
- Pull changes from the development environment, or make changes outside of DevOps Center.
- Commit changes to the project repository.
- Share your changes for team members to review.
- Indicate work items are ready to promote.
- Promote work items through the pipeline.

Get Ready to Use a Source Control System

DevOps Center requires that you use it with a source control system. You can create a new account or use an existing account. Right now, we support GitHub and Bitbucket (beta).

Open DevOps Center and Select a Project

Are you ready to begin work? Log in to the DevOps center org, launch DevOps Center, and then select a project.

Work Item Management

A team uses work items to track the progress of changes created to achieve a specific objective or task. These tasks can include enabling a user story, creating a feature, or addressing a bug. Managing releases through work items makes it easier to track the progress of the overall release and identify areas of concern.

Are You Sharing a Development Environment?

Changes are associated in DevOps Center with the development environment within a project. Therefore, we recommend that all developers have their own development environment. A single development environment can be connected to multiple work items.

What Happens When a Sandbox Is Refreshed

Refreshing a sandbox creates a new copy of the sandbox based on the production or sandbox org you copy it from.

Determine Which Changes to Include for the Work Item

The work item Changes tab is where you can see the changes you made in the development environment. As you make changes in the development environment, the source-tracking feature keeps a record of the component files that have been added, deleted, or changed.

Commit Changes to the Source Control Repository

After you decide which changes to include for the work item, commit the changes to the project's

source control repository. When you commit changes, they're stored in the repository branch that was created when the work item moved to the In Progress stage.

View Work Item Events in Activity History

You can view the Activity History for each work item, which provides a comprehensive history of all key events, including promotions, commits, work item status changes, warnings, and errors (failures). This historical view enables you to maintain and view a record of events and associated details for auditing, troubleshooting, and general visibility purposes.

Synchronize Your Development Environment

If you're working on a team with multiple developers, it's likely that your dev environment is going to become out of sync with the source control repository. DevOps Center tracks the differences between each development environment and the first pipeline stage's branch.

Review Changes with Team Members

When changes for an In Progress work item are committed and ready to share with reviewers, creating a review opens a change request, also called a pull request.

Indicate That Work Items Are Ready to Promote

After the work item is reviewed, a team member approves the work item to be ready to promote. Work items aren't available for promotion through the pipeline until they are in the Ready to Promote status.

External Change Requests and Merges

You can create a change request and merge your changes directly in the source control system, which is the first half of the promotion process. You can complete the promotion by deploying all externally merged changes to the pipeline stage's associated environment within DevOps Center or by using Salesforce CLI.

Promote Work Items Through Your Pipeline

You promote work items through a pipeline, which defines the sequence of stages that work items progress as they go through the release lifecycle from development to production (or some other final stage). You can have any number of pipeline stages. Your team manager built the pipeline when configuring DevOps Center.

View Pipeline Events in Activity History

DevOps Center provides a comprehensive history of all key pipeline events, including promotions, synchronizations, and errors (failures). You can use these events and their associated details for auditing, troubleshooting, and general visibility purposes.

Update the Project's Source API Version Each Salesforce Release

Because projects in DevOps Center require a source control repository that uses the Salesforce DX project structure, each DevOps Center project contains a configuration file, sfdx-project.json. During each Salesforce release cycle, make a plan to update the source API version to avoid metadata type version mismatch errors when moving changes through your release pipeline.

Get Ready to Use a Source Control System

DevOps Center requires that you use it with a source control system. You can create a new account or use an existing account. Right now, we support GitHub and Bitbucket (beta).

Note DevOps Center Bitbucket Cloud support is a pilot or beta service that is subject to the Beta Services Terms at Agreements - Salesforce.com or a written Unified Pilot Agreement if executed by Customer, and applicable terms in the Product Terms Directory. Use of this pilot or beta service is at the Customer's sole discretion.

If you're new to source control, we recommend you get ready by reading the source control documentation. If you plan to use GitHub, see the Git and GitHub Basics module and the Work with the GitHub Workflow unit in Trailhead.

1. Create an account.

   Most source control system vendors offer a free account, if your company doesn't already use source control.

   - GitHub
   - Bitbucket (beta)
2. Send your username to the team manager or Salesforce admin who's setting up DevOps Center so they can add you as a collaborator in the project repository.
3. Accept the invitation to collaborate in the repository.

   If you haven't received an email invitation, contact the person who set up DevOps Center.

Open DevOps Center and Select a Project

Are you ready to begin work? Log in to the DevOps center org, launch DevOps Center, and then select a project.

1. Log in to the org where DevOps Center is installed.

   Contact your team manager or Salesforce admin for org information and user credentials if you don't have that information.

2. From the App Launcher, find and select **DevOps Center**.

   Your team manager created at least one project as part of the setup process.

3. Select the project that you've been assigned to.

If you don't see DevOps Center listed in the App Launcher, see Troubleshoot DevOps Center Configuration in Salesforce Help.

Work Item Management

A team uses work items to track the progress of changes created to achieve a specific objective or task. These tasks can include enabling a user story, creating a feature, or addressing a bug. Managing releases through work items makes it easier to track the progress of the overall release and identify areas of

concern.

A work item encapsulates information about an objective or task while it's being worked on and is assigned to one individual. The work item is a historical record if questions arise about those changes. How granular you make your work items per DevOps Center project is up to you. However, if you break down your project into too many work items, it can become unwieldy and complicated to manage all the associated work items.

Work items are automatically numbered, typically in a sequential manner as they are created. However, in some cases, work item numbers aren't consecutive. Record numbers can skip when multiple operations are running concurrently. Initially, DevOps Center uses a 6-digit auto-number format for work items, but can expand to 7 digits and beyond as required.

When a work item is reviewed and approved, you can change the status to Ready to Promote. The work item then appears in the Approved Work Items list. Now, the work item can be promoted to the first pipeline stage for testing and verification. You can have as many work items in the Approved Work Items list as you wish. However, review the guidelines regarding how many work items you can have in a pipeline stage and how many you can promote at one time. See Promote Work Items Through Your Pipeline for details.

What Can I Do to Manage My Huge Work Items List?

In a perfect world, all the changes associated with your team's work items move through the pipeline in a timely manner. In reality, you can have numerous work items for feature work that has been postponed or changes that you never plan to move through your pipeline. But these changes hang around and stack up in the Approved Work Items list or the Available Changes list. If you haven't yet promoted those changes to the first pipeline stage, you have options to reduce the number of pulled changes or work items in the list.

If you have multiple work items that fall into these categories, you can set the work item status to **Never** to make these changes available to commit in another work item. Then you can create different work items to group similar changes.

- For changes you never plan to release or move through your pipeline, create a work item to use as a "vault" for storing these changes.
- For changes with a future release date, you can create a work item to act as a "parking lot" for temporarily parking those changes. You can organize your parking lot work items by grouping all changes for a specific feature or app into a work item. When your release date approaches, you can change the parking lot work item status to **Never** to make the changes available to associate with one or more work items. You can break up the development work into multiple work items to assign to different team members to update, validate, and test.

You can continue to commit more changes to the work item vault or parking lot as needed.

Open a Work Item

When you open a project for the first time, it's likely to contain several work items that your release manager created when configuring DevOps Center. You and other team members can create

additional work items as the project progresses.

Modify Work Item Details

Keep work items up to date to provide full value to the team. For example, if new acceptance criteria is identified, edit the work item to ensure the assignee knows of the change. Or if you're ready to start on an unassigned work item, assign the work item to yourself, which lets other team members know you've picked it up.

Work Item Statuses

A work item has a progress bar that shows which stage of the development process it's in. The work item starts in the New stage, meaning that work hasn't begun to support the bug or user story that the work item describes.

Change Work Item Status to Never

Sometimes plans change. Sometimes you begin down one path and must start over. Sometimes a change conflicts with another change and the best strategy is to remove the files from the commit to allow another work item's promotion to succeed. For these reasons, you can change the status of a work item to Never.

Create a Work Item

All work begins with a work item. A work item captures a logical unit of work, such as a user story, task, or bug. Work items are organized by project and belong to the project where you created them.

See Also

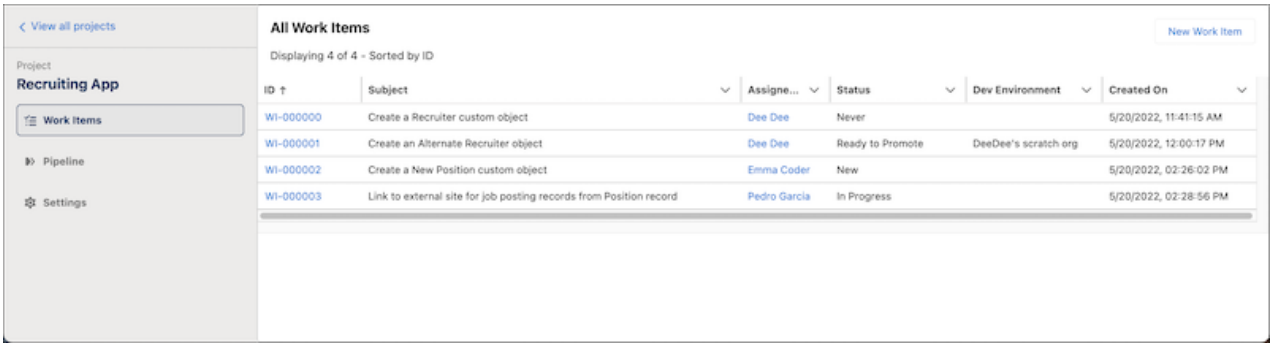
Available Changes List
Change Work Item Status to Never

Open a Work Item

When you open a project for the first time, it's likely to contain several work items that your release manager created when configuring DevOps Center. You and other team members can create additional work items as the project progresses.

1. From the Projects page, click the name of the project you're working on.

   The project opens to its All Work Items page. Look for any work items assigned to you.



2. Click the work item ID to open it.
   - If you're currently logged in to the source control system, DevOps Center opens the work item.
   - If you're not currently logged in to the source control system, log in and authorize access to DevOps

Center.

If the authorization is successful, you're returned to the work item. DevOps Center can make changes on your behalf in the source control repository.

3. Select how you're going to work.
   - Work in a Connected Dev Environment
   - Work Outside of DevOps Center

Modify Work Item Details

Keep work items up to date to provide full value to the team. For example, if new acceptance criteria is identified, edit the work item to ensure the assignee knows of the change. Or if you're ready to start on an unassigned work item, assign the work item to yourself, which lets other team members know you've picked it up.

1. From the Projects page, click the name of the work item's project.

   When you open a project, the Work Items tab is shown by default and lists all the current work items.

2. Locate the work item you want to edit, then click its ID to open it. Work items open in the Changes tab.
3. Edit the fields as needed.
   - In the Changes tab, choose a different development environment only if no changes have been committed for the work item from the current development environment.
   - Switch to the Details tab to edit the Subject, Description, or Assigned To fields.

Work Item Statuses

A work item has a progress bar that shows which stage of the development process it's in. The work item starts in the New stage, meaning that work hasn't begun to support the bug or user story that the work item describes.

As work items move through the release management cycle, the work item status is updated automatically to indicate how work is progressing.

The team can review a list of project work items and judge from the statuses how much work is done and how much is left to do.

STatus	Description
New	The initial status of a work item upon creation. The New status doesn't necessarily mean that no related work has occurred. Your team could be

STatus	Description
	planning, sizing, scheduling, designing, and so on, in support of this work item. When it's time to start building customizations, however, select a development environment. The work item moves to In Progress, and enables you to launch the environment directly from the work item.
In Progress	Work that the listed assignee is actively pursuing in the development environment. When in this status, another team member can't assign this item to themselves or start working on it. DevOps Center creates a branch directory for it in the project repository.
In Review	Work that's ready for your team members' review. When you click Create Review, the work item is moved to the In Review status, and a change (pull) request is created automatically.
Ready to Promote	The work item has been reviewed and approved, and is ready to be promoted. The work item doesn't have any merge conflicts and all mergeability rules and merge settings have been satisfied.
Promoted	This work item has been promoted to a pipeline stage. When it's promoted to the last pipeline stage, it's `Closed`.
Closed	Work that's complete, reviewed, fully tested and verified, and promoted through the entire pipeline.

See Also

Review Changes with Team Members

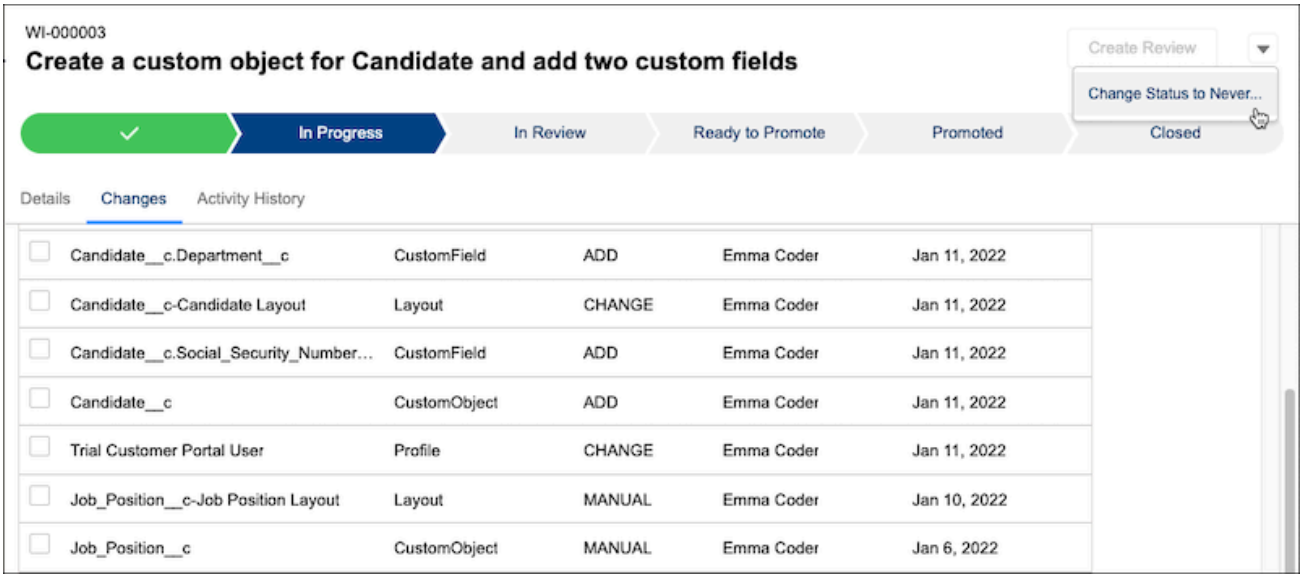
Change Work Item Status to Never

Sometimes plans change. Sometimes you begin down one path and must start over. Sometimes a change conflicts with another change and the best strategy is to remove the files from the commit to allow another work item's promotion to succeed. For these reasons, you can change the status of a work item to Never.

When you change the status to `Never`, the work item becomes inactive, and the committed component files are returned to the list of available changes. You can recommit any of these files to another work item.

- You can change a work item's status to `Never` before it's been promoted into a pipeline stage.
- Changing the status to `Never` is non-reversible.

From the work item menu, select **Change Status to Never**, then click **Confirm**.



Create a Work Item

All work begins with a work item. A work item captures a logical unit of work, such as a user story, task, or bug. Work items are organized by project and belong to the project where you created them.

Information in a work item includes:

- **ID**: An automatically generated unique identifier that has a WI- prefix (for example, WI-12345). The ID helps DevOps Center users differentiate between similar-sounding work items. It's also used in the project repository to identify the feature branch where changes for this work item are stored.
- **Subject**: Required. A brief description of the task to be tracked.
- **Description**: A more detailed description of the task to be tracked. If the work item is a bug, include an example of the problem to be solved. If the work item is a user story, include a hypothetical description of how the new functionality would benefit users.
  It's a good idea to include acceptance criteria that identifies what the successful completion of this work would look like. Make sure the person assigned this work item knows what's required or you could lose time to iterations that could have been avoided.
- **Assigned To**: DevOps Center user currently responsible for the completion of the task or objective tracked by this work item.

A team or release manager can create work items as needed to capture the complete scope of the project.

1. From Work Items, click **New Work Item**.
2. Enter a descriptive subject.
3. Use the Description field to provide information about the scope of work, acceptance criteria, and so on.
4. (Optional) Assign the work item to a team member.
5. Click **Save**.

The work item is displayed in the Work Items list.

6. Repeat this procedure to track and assign project work. All DevOps Center users can create additional work items as the project progresses.

When a team member is ready to begin work on a work item, they choose how they're going to perform the work:

- Develop and commit changes using a connected development environment (developer sandbox or scratch org)
- Develop and commit changes to a work item feature branch outside of DevOps Center (using their favorite IDE or other development tools)

Work in a Connected Dev Environment
When you choose to work in a connected development environment, you perform much of your development and configuration tasks directly in a Developer or Developer Pro sandbox with source tracking enabled. We also support scratch orgs, which have source tracking enabled by default.

Work Outside of DevOps Center
After a work item is created, you can choose to do your development work entirely outside DevOps Center (without connecting to a development environment). After changes are merged in the source control system, the promotion is completed in DevOps Center or by using Salesforce CLI.

Work in a Connected Dev Environment

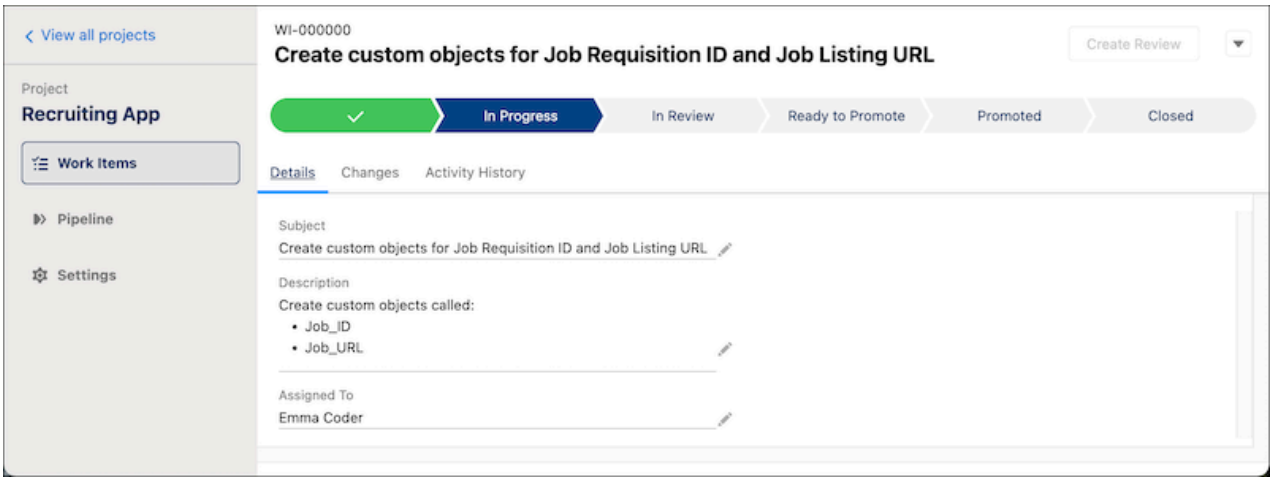
When you choose to work in a connected development environment, you perform much of your development and configuration tasks directly in a Developer or Developer Pro sandbox with source tracking enabled. We also support scratch orgs, which have source tracking enabled by default.

1. In Work Items, click the ID of a work item to open it.
2. Because you want to perform your development work in your developer org, select **I Want to Develop and Commit My Changes to the Work Item Feature Branch from DevOps Center Using a Connected Development Environment**, then select your development environment from the list.

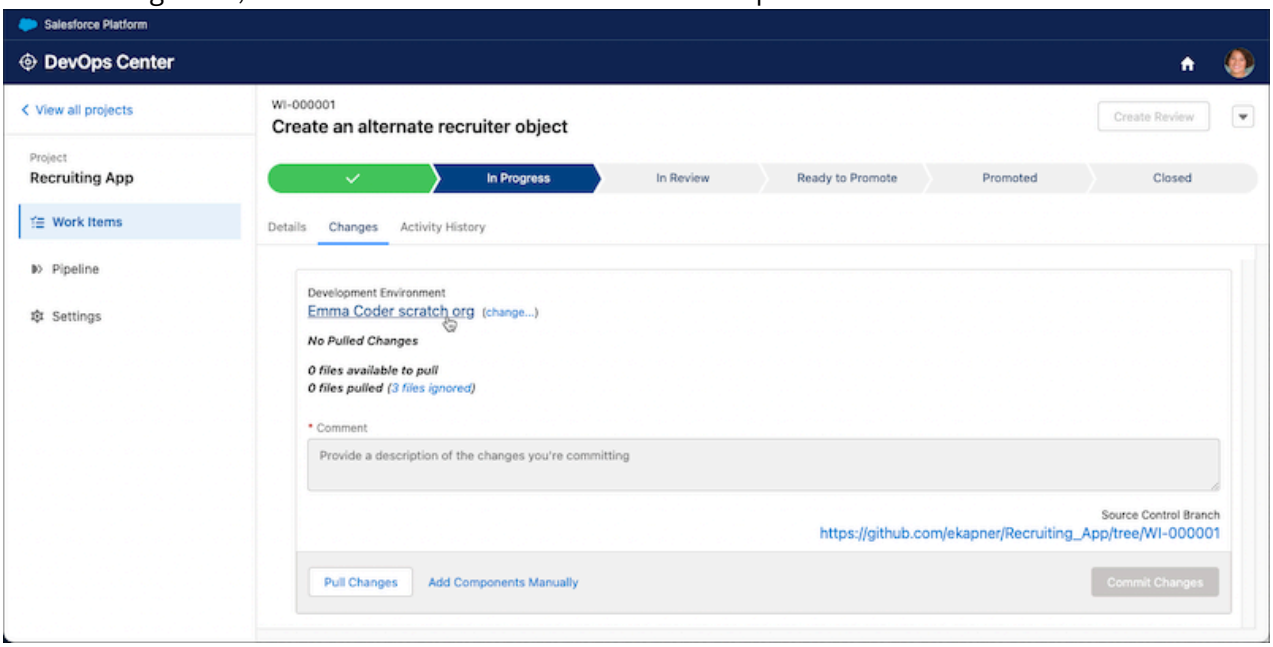
   Development environments were added to the project during setup to make them available for team members. During setup, each environment has a nickname to make it easier to recognize. To begin work, select the development environment you want to use for this work item. If you don't see your development environment listed, ask your team manager or Salesforce admin to add it.

3. Click **Proceed**.
4. Click the Details tab to view the description for the work item scope.

   If the work item is unassigned, assign it to yourself.



5. In the Changes tab, click the dev environment nickname to open it.



If you haven't yet logged into this development environment from within DevOps Center, complete the login process, then click the environment nickname link again to open the development environment.

After you complete your development work, pull changes from the development environment.

See Also

- Pull Changes from the Development Environment
- Work Outside of DevOps Center
- *Salesforce DX Developer Guide*: Enable Source Tracking in Sandboxes

Work Outside of DevOps Center

After a work item is created, you can choose to do your development work entirely outside DevOps Center (without connecting to a development environment). After changes are merged in the source control system, the promotion is completed in DevOps Center or by using Salesforce CLI.

After you select which work item to work on, DevOps Center can create the feature branch or you can create the feature branch using the tools of your choice. Be sure to create the feature branch from the first pipeline stage's branch, for example, Integration. Work item feature branches must follow this naming convention:

```
https://github.com/<GitHub-ID>/<Repo-Name>/tree/WI-000000
```

```
https://bitbucket.org/<Workspace>/<Repo-Name>/branch/WI-000000
```

Example:

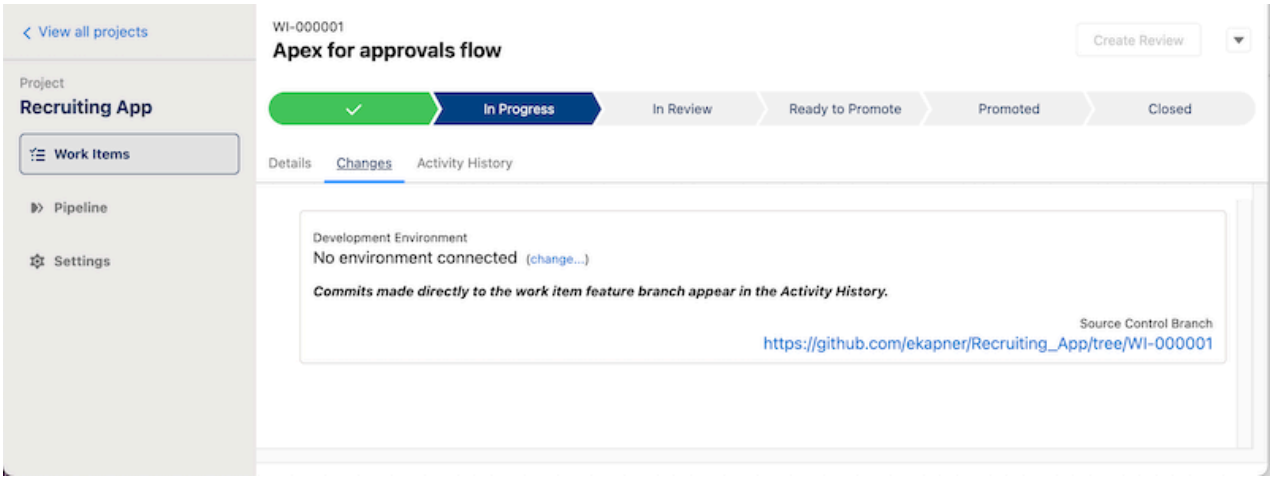
```
https://github.com/jdoe/Recruiting_App/tree/WI-000015
```

```
https://bitbucket.org/sfdc/Recruiting_App/branch/WI-000007
```

Important If you're working directly in the source control repo, delays can occur regarding when these actions happen and when they're reflected in DevOps Center. If you take subsequent actions on the related objects in DevOps Center before they're reflected, you're likely to see unexpected or inconsistent behavior. For metadata changes, we recommend that you wait for delayed events to be reflected in DevOps Center. For non-metadata changes, such as updates to the `.forceignore` file, you can continue with the promotion.

1. In DevOps Center, create a work item or select an existing one to associate with the work.
2. Decide how you want the feature branch created.
   - To have DevOps Center create the feature branch, click the work item ID. Select **I Want to Develop and Commit My Changes to the Work Item Feature Branch From Outside of DevOps Center**, then click **Proceed**.
   - Create the branch directly in the source control repository from the first pipeline stage's branch using the required naming convention.
   - Use another tool, such as Salesforce Extensions for VS Code, to create a feature branch from the first pipeline stage's branch using the required naming convention. Be sure to push the feature branch to the project repository.

   After the branch is created and the event is reflected in DevOps Center, the work item status moves to `In Progress`. You can start your development work.



3. After you complete your development work, commit your changes in the source control system.

   If you commit your changes locally, don't forget to push them to the repository. After you commit your changes, the changes in the feature branch are reflected in DevOps Center.

4. Decide how to create the change request and promote your changes to the first stage in the release pipeline.
   - Create the change request and promote your changes in DevOps Center. See Review Changes with Team Members and Promote Individual Work Items to continue.
   - Create a change (pull) request and merge your changes in the source control system, then complete the promotion in DevOps Center. See Promote Changes Merged Outside of DevOps Center to continue.
   - Create a pull request and merge your changes in the source control system, then deploy the changes to the pipeline stage's environment using Salesforce CLI. See Deploy Changes Using Salesforce CLI (Beta) for details.

See Also

Work in a Connected Dev Environment
Promote Changes Merged Outside of DevOps Center

Are You Sharing a Development Environment?

Changes are associated in DevOps Center with the development environment within a project. Therefore, we recommend that all developers have their own development environment. A single development environment can be connected to multiple work items.

Here are some considerations if you are sharing a development environment.

When you pull changes, the changed components are displayed in the changes list, but no files have been retrieved out of the development environment yet. The file isn't actually retrieved from the development environment until it's committed.

After a component is committed, it no longer appears in the changes list for any work item that is connected to that development environment, until the component is changed again in the development

environment.

If multiple developers are using one development environment (within the same project), actions by one developer can have ramifications on the other. For example:

- If two developers are changing the same component, after one developer commits the component, it no longer shows up in the other developer's change list (until the component is changed again in the development environment and pulled again). This disconnect can result in not all the needed files being included in a work item's commit and feature branch.
- If someone else changed the component since you pulled it, the committed file is included with those latest changes. Consequently, the file you're committing could be different than what you expect.
- When you pull changes, you see all the files changed by anyone who is making changes in the development environment, since the files' last commit.
- If multiple developers are making changes in a shared development environment, the Last Modified By column in the changes list indicates who made the last change. So if you're using that field to identify changes you've made, you could miss some if someone else has modified it more recently than you.

What Happens When a Sandbox Is Refreshed

Refreshing a sandbox creates a new copy of the sandbox based on the production or sandbox org you copy it from.

REQUIRED EDITIONS

	PERMISSION SETS NEEDED
To manage environments:	DevOps Center Manager
To access the named credentials required to authenticate to environments:	sf_devops_InitializeEnvironments

The sandbox refresh procedure creates a new org, copies the metadata and data from the source org, then deletes the old org after activation. Because the sandbox refresh results in deletion of the original org, DevOps Center loses connection with the original environment, but provides you the ability to swap the existing connected environment with the new environment. As part of the swap process, DevOps Center also ensures that the new environment is current with the latest source in the pipeline stage's branch.

The way you initiate the swap is different depending on if you're swapping a development environment or a pipeline environment.

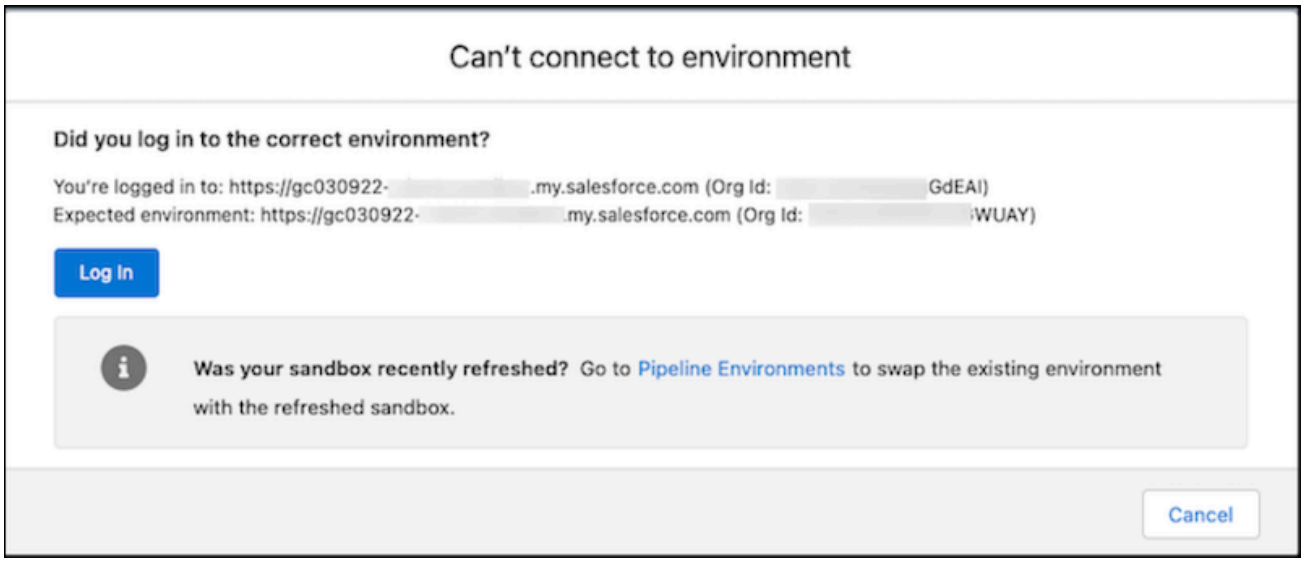
How to Tell That Your Dev Sandbox Was Refreshed

After a refreshed sandbox is activated, the connection is lost within DevOps Center because the refreshed sandbox has a new org ID. You'll notice some clues when you go to update a work item that

was connected to the sandbox.

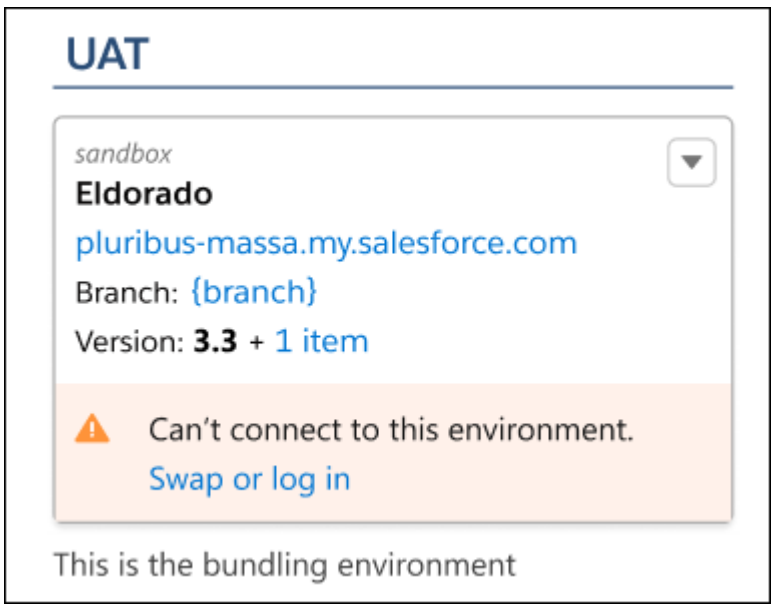
- In the work item Changes tab, you now see `Can't determine the files to pull - Log in to development environment`.
- You try to pull or commit changes, and you're asked to log in.

When you try to log in, you get confirmation that the sandbox you're trying to log into is no longer valid. Click the link to go to the Pipeline Environments page, where you can swap the sandbox and reconnect to it.



How to Tell That a Pipeline Sandbox Was Refreshed

In Pipeline Environments, you see a message on the pipeline stage telling you that DevOps Center can't connect to the environment.



Swap a Development Environment
Swap a development environment with another development environment. If your development environment is a Developer or Developer Pro sandbox that was recently refreshed, the connection to the existing environment is no longer valid. You must swap the sandbox to reestablish the connection to it in DevOps Center.

Swap a Pipeline Environment
Swap a pipeline environment with another environment. If the pipeline environment is a sandbox that was recently refreshed, you must swap it before anyone can promote changes to the associated pipeline stage. The DevOps Center connection to the current environment is no longer usable.

See Also
Salesforce Help: Refresh a Sandbox

Swap a Development Environment

Swap a development environment with another development environment. If your development environment is a Developer or Developer Pro sandbox that was recently refreshed, the connection to the existing environment is no longer valid. You must swap the sandbox to reestablish the connection to it in DevOps Center.

REQUIRED EDITIONS

PERMISSION SETS NEEDED	
To swap an environment:	DevOps Center Manager
To access the named credentials required to authenticate to the environment:	sf_devops_InitializeEnvironments

Important We recommend that all sandboxes are refreshed from your production (release environment), or cloned from a pipeline stage or an existing development environment.

Before you swap the dev environment, keep these things in mind:

- We recommend that you promote all work items before refreshing the sandbox.
- When you refresh the sandbox, you lose any changes in the current environment that weren't pulled and committed to work item feature branches in the source control repository.
- Any changes in the old sandbox that were pulled and committed, but not promoted, are in the work item's feature branch but aren't visible in the new sandbox. To view the committed changes in the new sandbox, promote the affected work items and then perform a sync from the first pipeline stage's branch back to the sandbox. See Synchronize Your Development Environment for details.
- The DevOps Center connection to the current environment is no longer usable.
- The new dev environment must have source tracking enabled.

1. In the Pipeline Environments tab, select **Swap Environment** from the environment's menu.
2. Click **I Understand** to acknowledge the considerations, then click **Continue**.

3. Select **Log In to New Environment**, then click **Log In**.
4. Complete the login process.

After the swap is complete, check to see if the development environment is out of sync. If so, we recommend synchronizing the environment before continuing work.

See Also

Salesforce DX Developer Guide: Enable Source Tracking in Sandboxes

Swap a Pipeline Environment

Swap a pipeline environment with another environment. If the pipeline environment is a sandbox that was recently refreshed, you must swap it before anyone can promote changes to the associated pipeline stage. The DevOps Center connection to the current environment is no longer usable.

REQUIRED EDITIONS

PERMISSION SETS NEEDED	
To swap an environment:	DevOps Center Manager
To access the named credentials required to authenticate to the environment:	sf_devops_InitializeEnvironments

Important We recommend that all sandboxes are refreshed from the production (release environment), or cloned from a downstream pipeline stage.

1. In the Pipeline Environments tab, select **Swap Environment** from the environment's menu.
2. To acknowledge the considerations, click **I Understand**, then **Continue**.
3. Select **Log In to New Environment**, then click **Log In**.
4. Complete the login process.

   The pipeline stage is disabled until the deployment is completed.

   - If DevOps Center can determine that the environment was descended from an environment in the project pipeline, all work items that are connected to the current pipeline environment are moved to the new pipeline environment.
   - If DevOps Center can't determine that the environment's ancestry, DevOps Center deploys all changes from the pipeline stage's branch to the new pipeline environment. This operation can take a while.

Determine Which Changes to Include for the Work Item

The work item Changes tab is where you can see the changes you made in the development environment. As you make changes in the development environment, the source-tracking feature keeps a record of the component files that have been added, deleted, or changed.

After making and testing your changes, pull them from your development environment, so you can see the changes before committing them to the repository for review. For the specific work item, you can select which changes you want to move forward and commit.

Exclude Metadata from the Changes List

When retrieving changes from your development environment, you likely have files or metadata types that you want to exclude from committing to the source control repository. You can exclude individual files or all files in a specific directory with a `.forceignore` file. DevOps Center provides you the ability to view information about these ignored files but they don't show up in the changes list.

If the project repository was created using the provided repository template, your project already includes a `.forceignore` file. This file was likely modified to ensure it contains all the files and metadata types your team wants to exclude when pulling changes from its development environments. If the project was created using Salesforce CLI or some other method, we recommend that you check this file into the source control system.

If you're familiar with using Salesforce CLI outside of DevOps Center, we honor excluding metadata when

running any of the commands to deploy or retrieve metadata, such as `sf project deploy start` or `sf project retrieve start`.

Some common files and metadata that are often excluded:

- `package.xml`
- Profiles
- App menu and app switcher
- Settings
- Object translations
- LWC configuration files
- Tests

See Salesforce Help: Determine What Metadata Types to Ignore During Development for sample syntax and considerations when updating the `.forceignore` file.

See Also

Salesforce DX Developer Guide: How to Exclude Source When Syncing or Converting

Supported Special Characters in Metadata Component Names

When naming metadata components, DevOps Center and its underlying technologies, such as Metadata API and Salesforce CLI, support a subset of special characters. To avoid issues when deploying or promoting metadata to a pipeline stage, we strongly suggest using only supported special characters in your component names.

The supported special characters are:

```
/\^$#+?()|[]{}!@$&:',"
```

If you get the deployment error, `The deployment of work items into the next stage failed`, where the explanatory text says that the component was not found in the zipped directory, see Error: Deployment of the work items into the next stage failed for details on how to fix it.

Pull Changes from the Development Environment

Because the development environment has source tracking enabled, DevOps Center knows what files have been added, changed, or removed. When you pull changes, DevOps Center displays a list of all files that it has identified as changed. These changes aren't associated with the work item until you select and commit them to the source control repository.

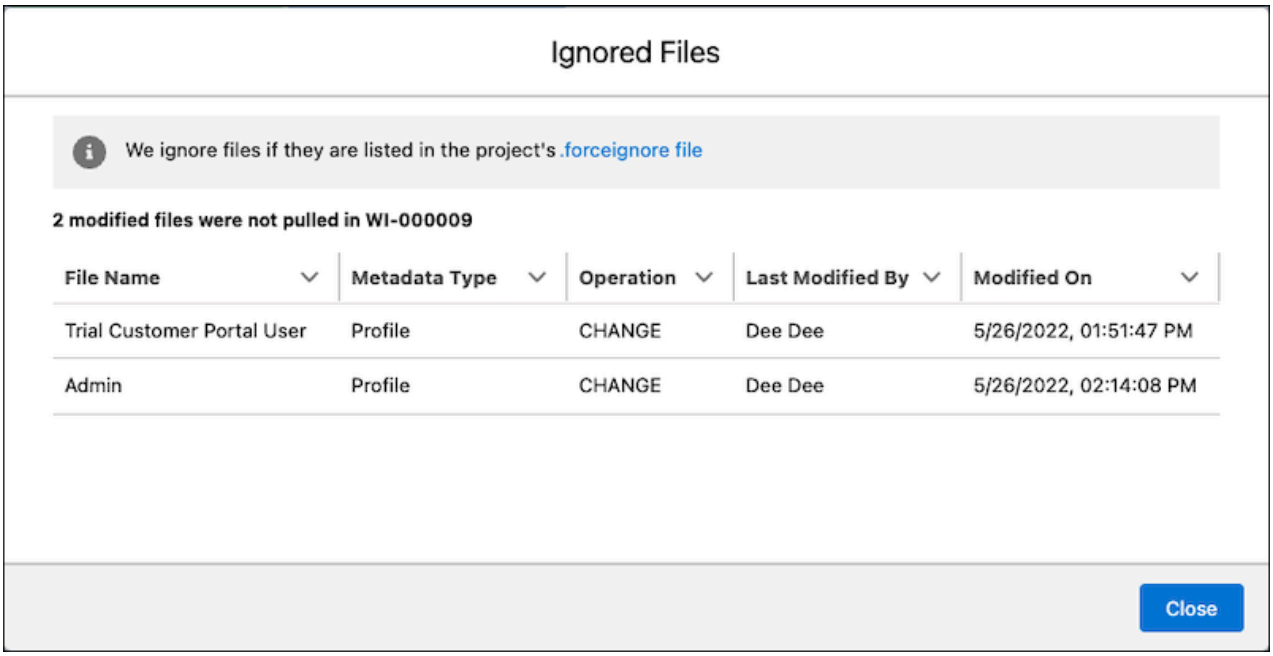
1. From within the Work Item Changes tab, click **Pull Changes**.

   DevOps Center displays how many files were pulled, how many files are available to pull, and how many files were ignored. See Available Changes List for details. DevOps Center can retrieve

approximately 40 changes at a time. You can keep clicking **Pull Changes** until all changes are listed.

2. (Optional) Click **n files ignored** to view which files were excluded as determined by the `.forceignore` file.

   In this example, two profiles were excluded and aren't available to commit.



3. Next, select the changes to commit to the source control repository.

See Also

Metadata API Guide: Deleting Components from an Organization

Available Changes List

When you connect a source-tracked development environment to a work item and pull changes, DevOps Center displays a list of all files that it has identified as changed. These changes aren't associated with the work item until you select and commit them to the source control repository. After a change is committed, it no longer appears in the Available Changes list.

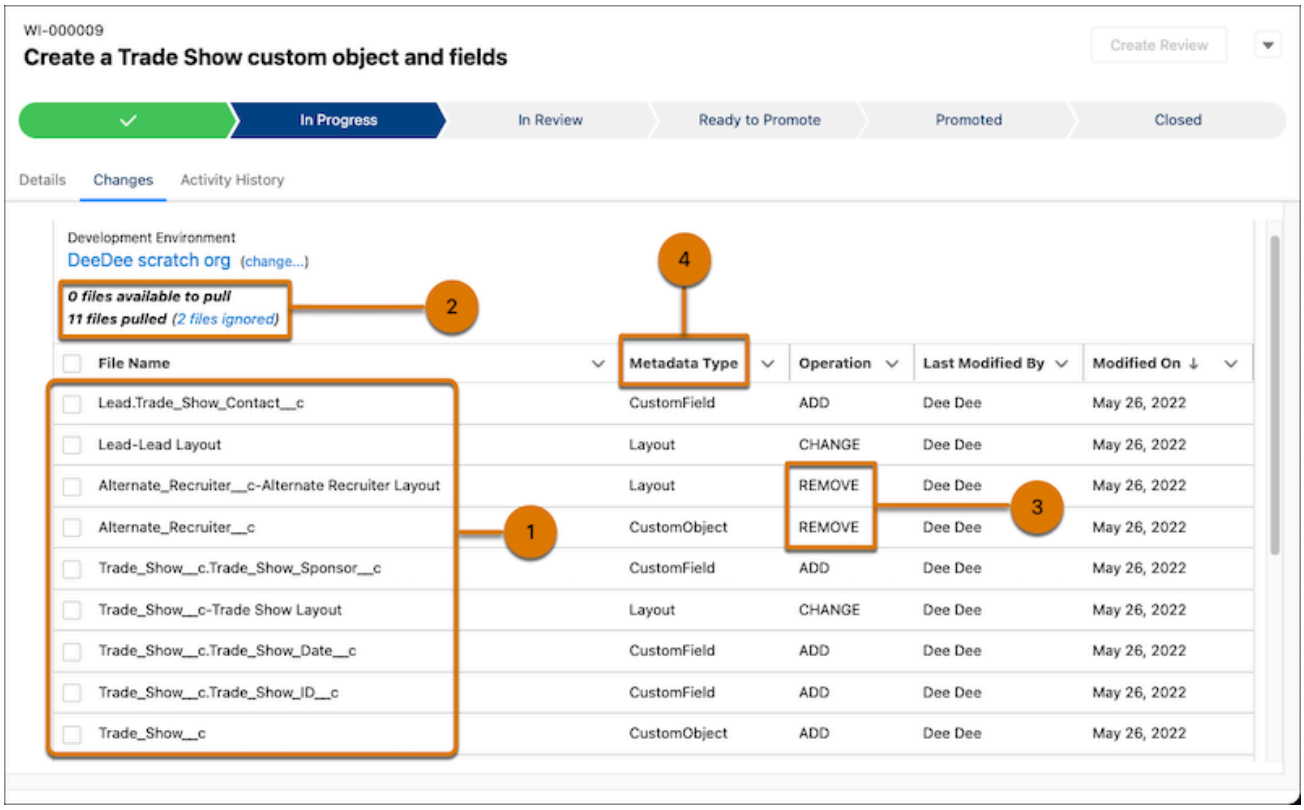
Any changes that you don't select and commit are again displayed in the Available Changes list. The changes are available to include in this work item (in another commit) or in another work item after you click the Pull Changes button.

In the work item Changes tab, DevOps Center lists all files that source tracking has identified as changed (1). DevOps Center also lists how many files were pulled and how many new files are available to pull (2). DevOps Center can retrieve approximately 40 changes at a time. You can keep clicking **Pull Changes** until all changes are listed.

If you excluded metadata from being pulled in the `.forceignore` file, we show you how many files were ignored during the pull operation (2). If you deleted files in your development environment, they're

marked as REMOVE (3).

If your changes list contains many items, all columns in the list are sortable. Click the column header to sort (4).



If You Change Your Mind After You Commit Changes

Let's say you commit changes for the work item but decide to remove some or all of the changes from the work item. In this case, you can change the Status of the work item to Never if you haven't yet promoted the changes to the first pipeline stage. When you change the status to Never, the source tracking for all the included changes is reset. The changes are returned to the available changes list, where they can be committed in another work item. See Change Work Item Status to Never for more details.

Connect to a Source-Tracked Sandbox

DevOps Center requires a source-tracked sandbox as your development environment, so we can identify what files and metadata have changed. If your current development environment doesn't have source tracking enabled, you have two options to enable it: directly in a specific sandbox, or in production for all developer sandboxes.

- If you enable the feature directly in the sandbox, you don't have to refresh the sandbox. Any new changes are source tracked.
- If you enable the feature in production, you must refresh the sandbox. After you refresh the sandbox,

you must reconnect it in DevOps Center. See Swap a Development Environment.

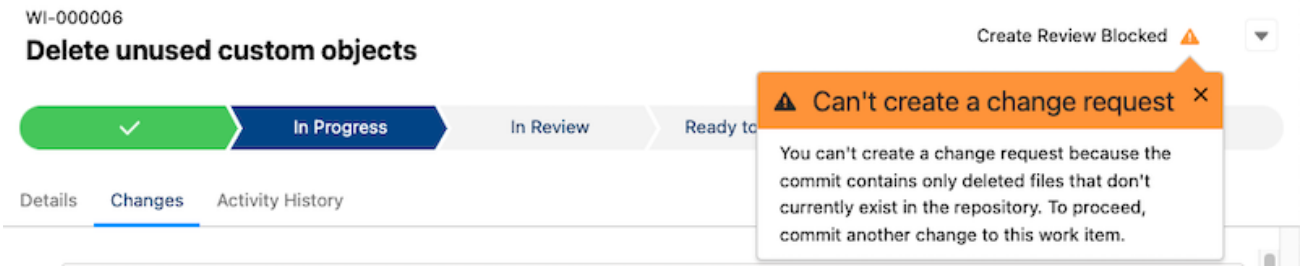
For more information on enabling source tracking, see Salesforce Help: Enable Source Tracking in Sandboxes.

If Your Changes Include Deleted Metadata

When you delete a component in the development environment and pull the change, it appears as a REMOVE change type. When you commit and subsequently promote this change, the component is removed from the target stage, leveraging the Metadata API destructive changes functionality.

If the files currently exist in the source control repo, you can commit deleted files by themselves. If the files don't exist in the source control repo, you must commit another change with the removed files before you can create a change request for the work item.

At the top of the work item, you see a warning message.



Add Any Metadata Component Manually

A key benefit of DevOps Center is that we track changes for you. However, for those times when you require flexibility to include more metadata than what was pulled automatically, you can select and commit additional metadata source files, not just the ones that DevOps Center determines have changed.

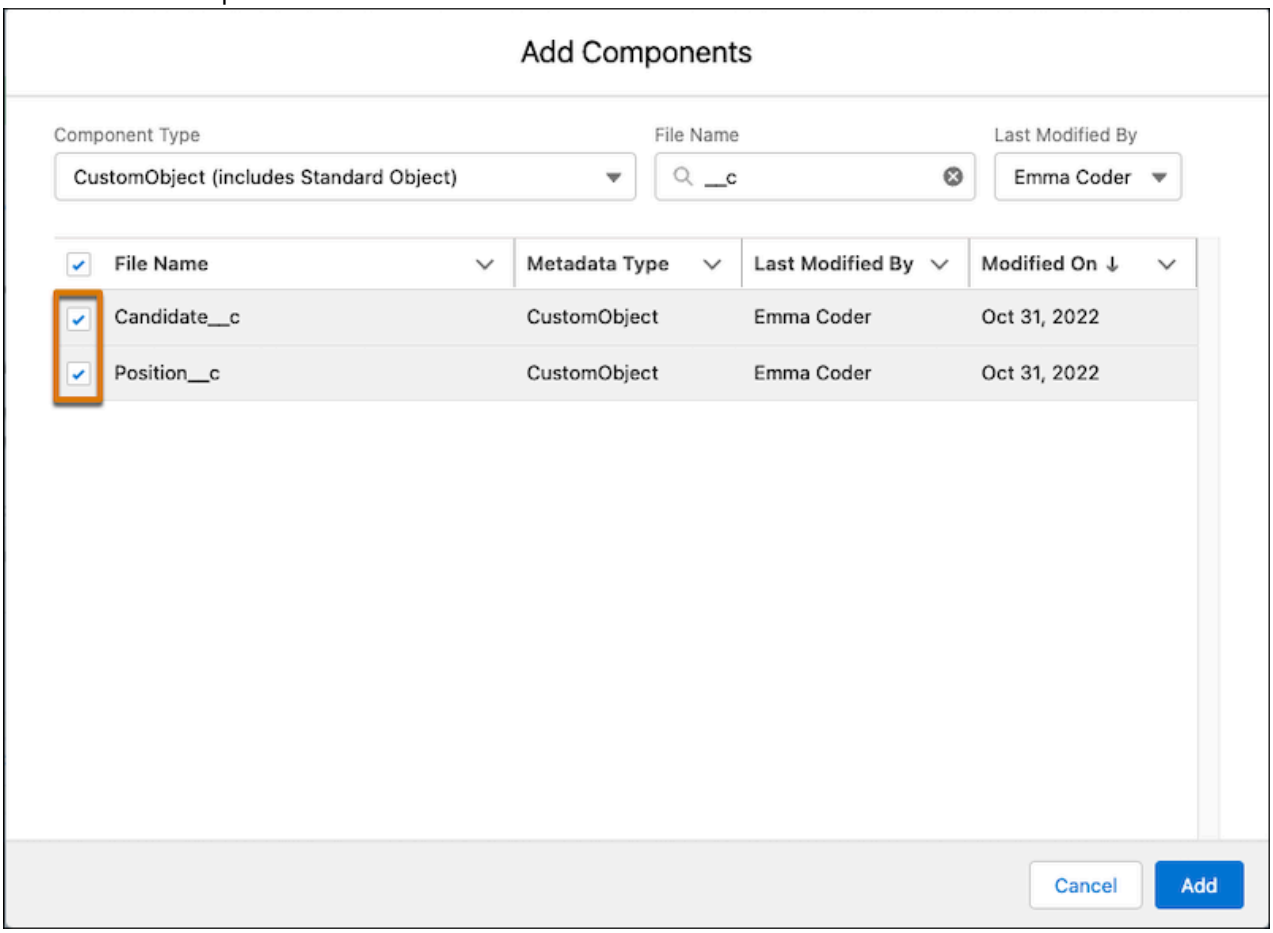
Some use cases include:

- Source tracking fails to identify that a file has changed.
- Your development environment is shared, and another teammate has already committed a file that you want to also include in your work item commit. If the component hasn't changed since the other teammate committed it, it doesn't appear in your changes list automatically.
- You want to commit files that were changed before source tracking was enabled in the development environment.

If you manually add a component that's already in the Changes list, the Operation type continues to reflect the original Operation type (CHANGE or ADD).

1. On the Work Item Changes tab, click **Add Components Manually**.
2. Select a component type from the list.

3. (Optional) Narrow the results by entering alphanumeric characters in the file name field, or selecting who last modified the component from the menu.
4. Select which components to add.



5. Click **Add**.

   After it's added, the component appears in the Changes list with an Operation type of MANUAL.

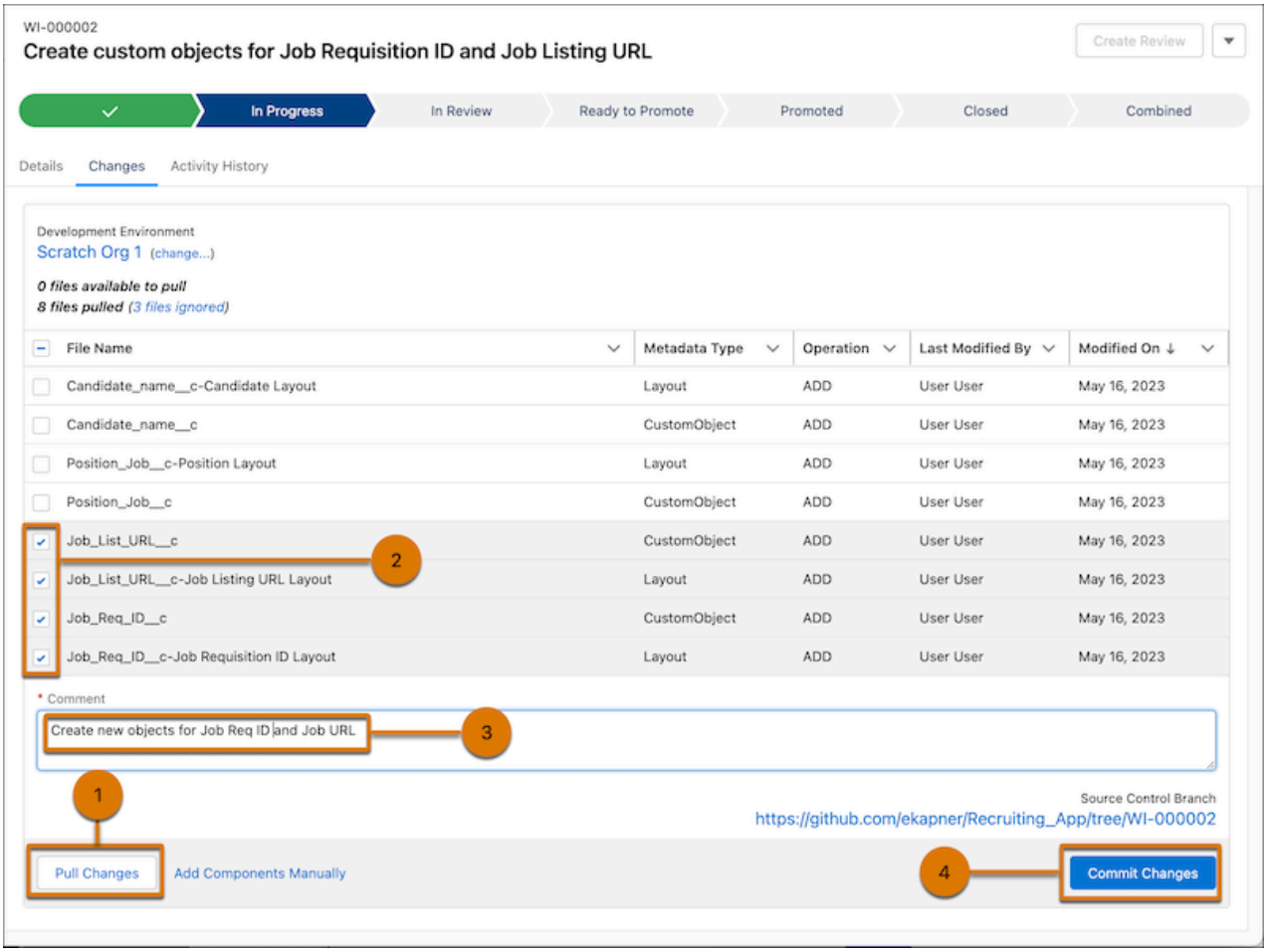
Commit Changes to the Source Control Repository

After you decide which changes to include for the work item, commit the changes to the project's source control repository. When you commit changes, they're stored in the repository branch that was created when the work item moved to the In Progress stage.

Before you can commit changes, you must pull changes from the development environment (1).

1. In the work item Changes tab, select the changes to commit (2).

   Only the changes you select and commit become part of the work item. Unselected changes are available to commit in another work item.



2. (Optional) Add metadata components manually.
3. Add a commit comment. (3)

   Include what's distinctive about the changes, so you can identify an individual commit by more than just its timestamp. A single work item can have multiple commits. You can continue to work in your development environment and commit a new group of changes for the same work item.

4. Click **Commit Changes** (4).

   The changes are committed to the work item's project repository branch.

   The branch is named after the work item, so it's clear where the changes it contains came from. For example, the branch for a work item WI-000002 looks something like:

   ```
   https://github.com/<username>/<repo-name>/tree/WI-000002
   ```

   ```
   https://bitbucket.org/<workspace>/<repo-name>/branch/WI-000002
   ```

   After you commit changes to the repository, the source tracking for these components in your development environment is reset. This reset ensures that changes that are already committed don't keep showing up when you pull changes from the development environment.

5. (Optional) To see the branch in the source control repository, click the Source Control Branch URL.

   The Code page for the branch is shown. Explore the changes you recently committed in the source control repository.

If you see the error `Ending position out of bounds: -1`, ask a Salesforce admin to turn off high assurance for authorization providers. See Troubleshoot DevOps Center Errors for details.

Stalled Commits
A stalled commit can occur when the updates from the external services back to DevOps Center are interrupted during the commit operation. After you determine when the failure occurred, you can fix it.

Stalled Commits

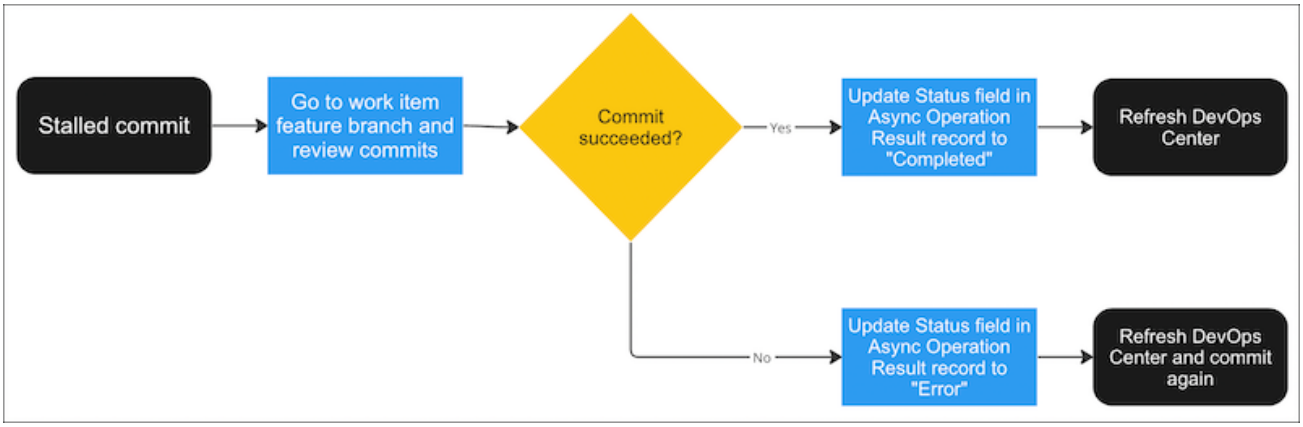
A stalled commit can occur when the updates from the external services back to DevOps Center are interrupted during the commit operation. After you determine when the failure occurred, you can fix it.

Here are the most common scenarios:

- Commit operation failed when pushing changes to the feature branch.
- Commit operation succeeded and changes were pushed to the feature branch, but the commit appears stalled in DevOps Center.

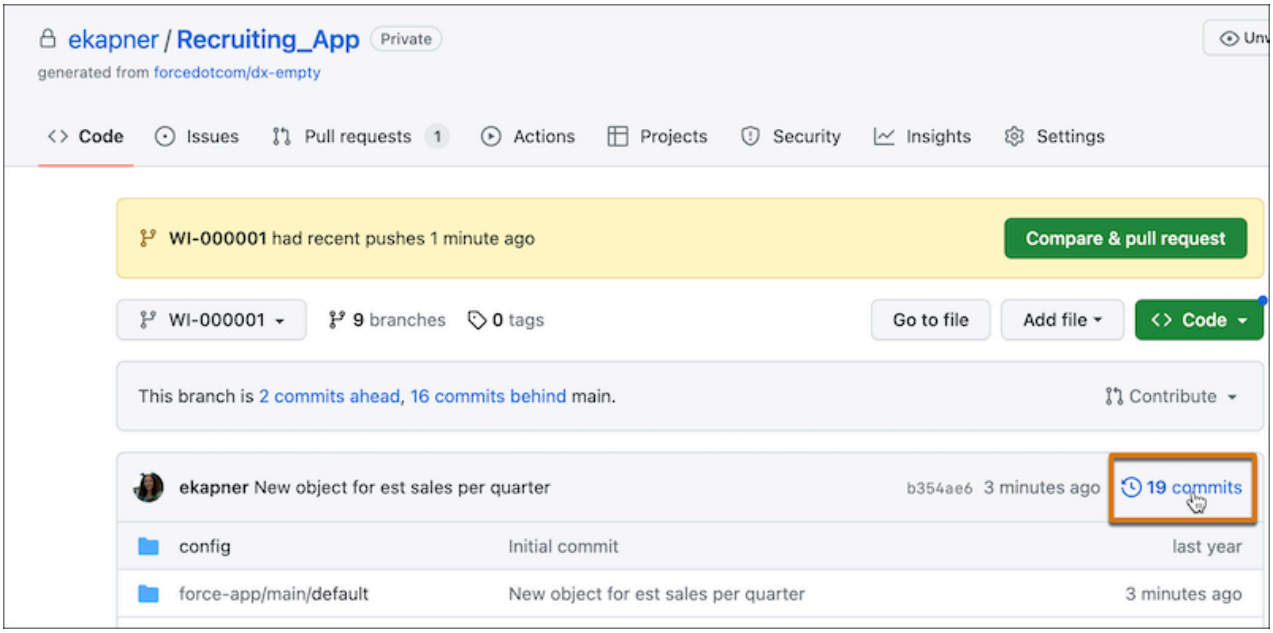
To determine how to proceed to fix the stalled commit, first troubleshoot to discover if the commit succeeded or failed. Then execute a query to find the associated Async Operation Result record and update its status.

Note Alternatively, you can change the Status of the work item to **Never**, and then start again with a new work item.



1. In the work item, click the link for the branch to view it in the source control repository.
2. Find the commits list in the source control repository.

   In GitHub, click the **<n> commits** link to view commits for the feature branch.



In Bitbucket, the list of commits is on the branch's page.

3. See if you can find the commit that contains the changes you selected. Look at the commit descriptions to identify the commit.
   - If you find the commit in the list, the commit succeeded.
   - If you don't see the commit in the list, the commit failed.

   Now you're ready to find the associated Async Operation Result record in DevOps Center.

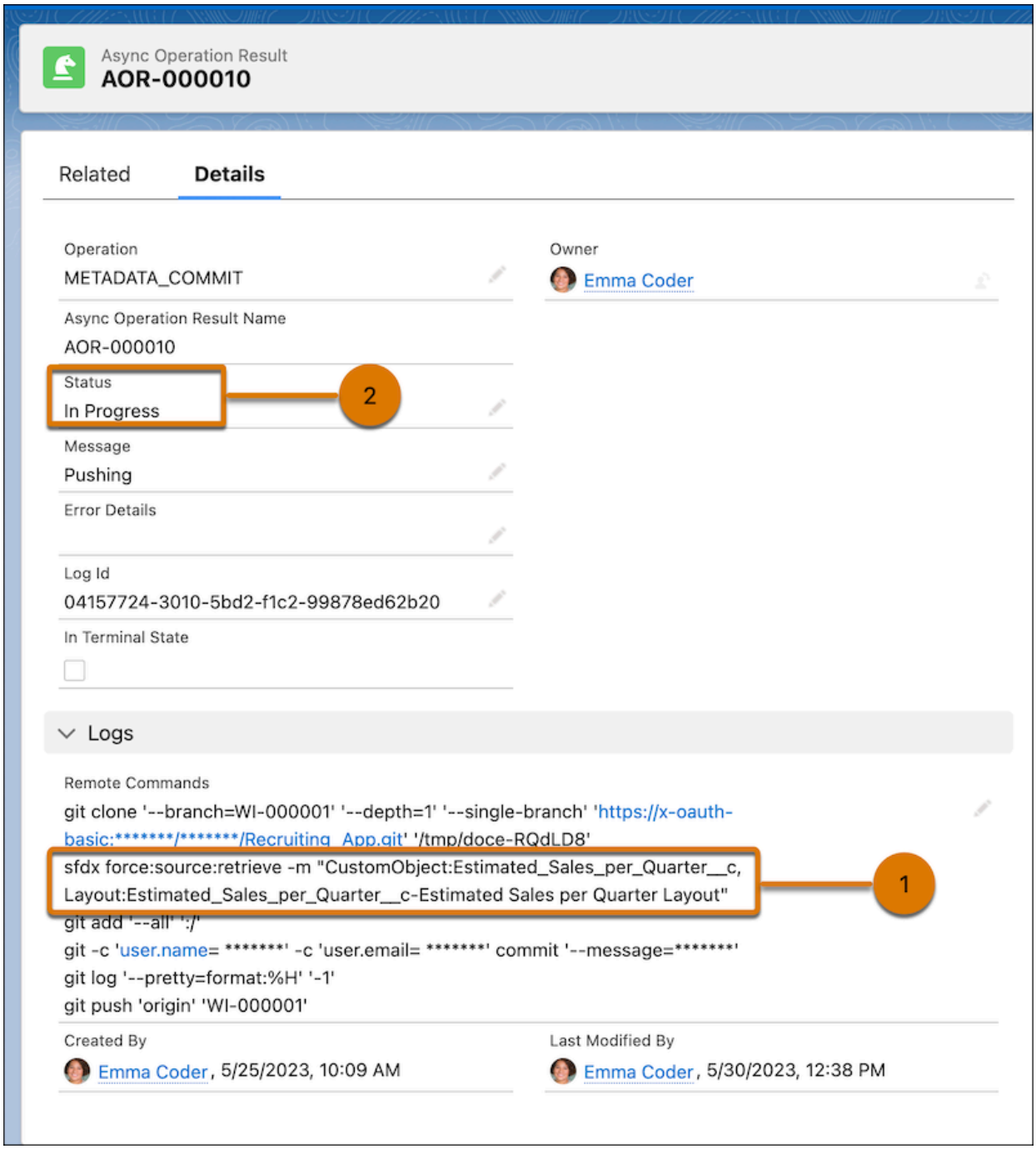
4. In the org in which DevOps Center is installed, launch Developer Console, then select the **Query Editor** tab.
5. Enter this SOQL query to find the associated commit that's still In Progress.

   In this example, line breaks were added for readability purposes. If you copy this query, remove the line breaks before running it.

```
SELECT Id FROM sf_devops__Async_Operation_Result__c
WHERE sf_devops__Operation__c = 'METADATA_COMMIT' AND
sf_devops__Status__c = 'In Progress'
```

   If you see only one record, select to highlight it, click **Open Detail Page**, and then go to the next step. If the query returns multiple records, look at each record to determine the correct one.

   a. Select a record to highlight it, then click **Open Detail Page**.
   b. In the Async Operation Result, look in the Logs section to see if the text contains references to the metadata that you're trying to commit (1). In the example, the referenced metadata is `CustomObject:Estimated_Sales_per_Quarter__c` and its layout.



c. After you find the correct record, go to the next step.

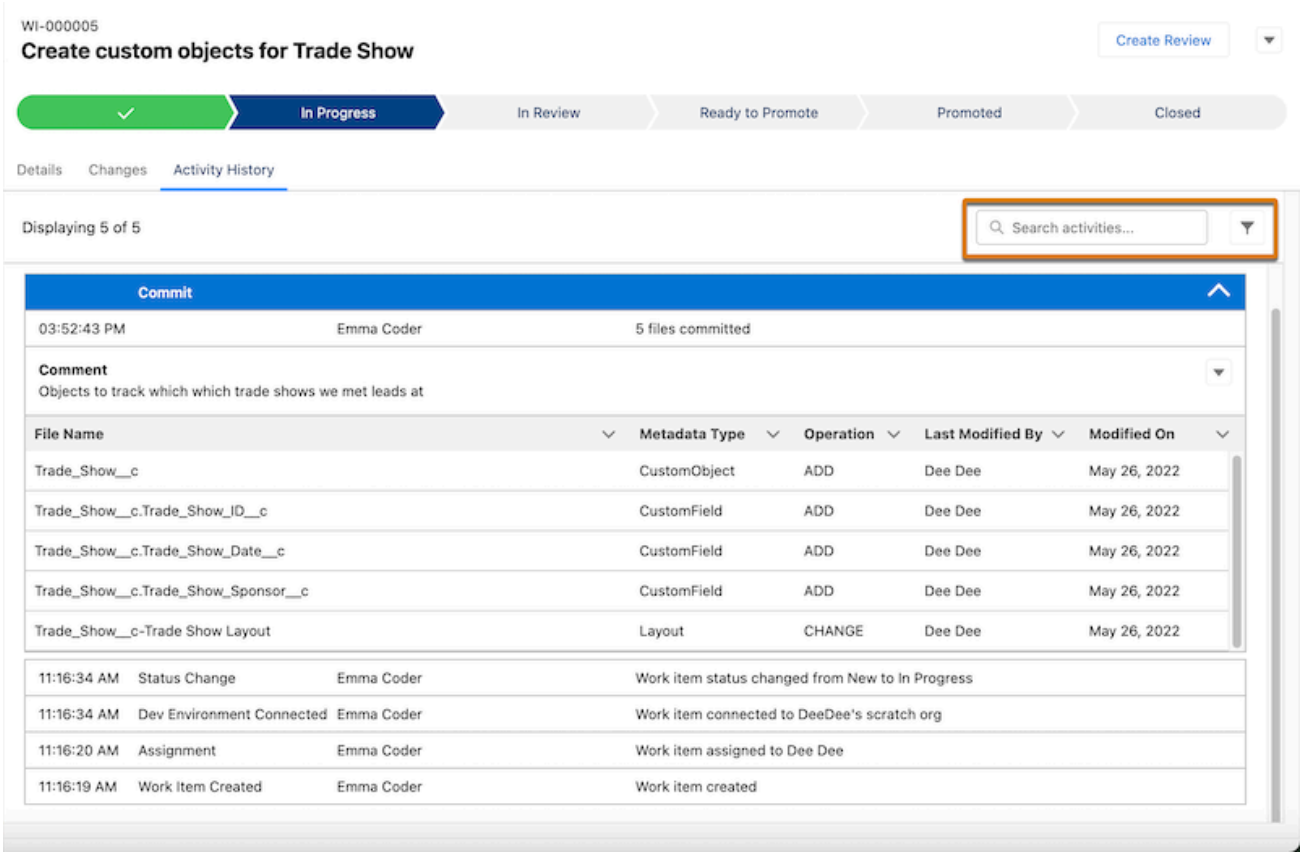
6. In the Async Operation Result record, change the Status (2) based on if the commit succeeded or failed, then click **Save**.
   - If the commit succeeded, change the Status to **Completed**.
   - If the commit failed, change the Status to **Error**.
7. Refresh DevOps Center, then proceed with the work item.
   - If the commit succeeded, the work item is ready for additional commits, or to move to the next stage, In Review.
   - If the commit failed, select the changes again and commit them.

View Work Item Events in Activity History

You can view the Activity History for each work item, which provides a comprehensive history of all key events, including promotions, commits, work item status changes, warnings, and errors (failures). This historical view enables you to maintain and view a record of events and associated details for auditing, troubleshooting, and general visibility purposes.

Activity histories can get long, so we give you the ability to narrow the list.

- To filter on User, Activity Type, and Date Range, use the filter.
- Use the Search Activities box to search using keywords.



Synchronize Your Development Environment

If you're working on a team with multiple developers, it's likely that your dev environment is going to become out of sync with the source control repository. DevOps Center tracks the differences between each development environment and the first pipeline stage's branch.

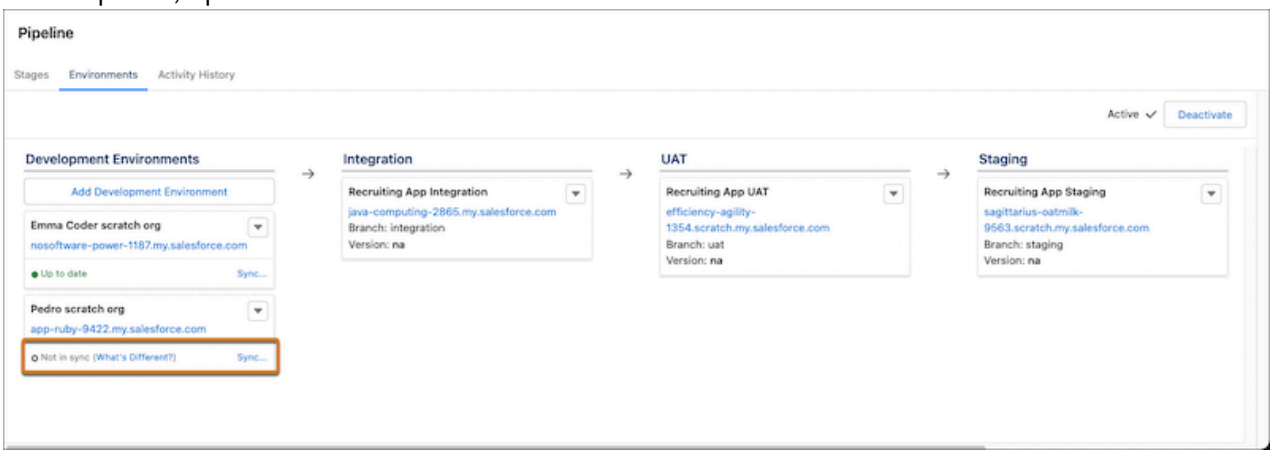
When a development environment is connected to a work item, you're notified if it's not up-to-date and given the option to synchronize it before starting new work. You can also choose to synchronize it at any time. This synchronization process ensures that you're developing against the latest integrated source of truth, including changes from other teammates. It also ensures that your changes are compatible with

other changes in the pipeline and reduces the possibility of conflicts when promoting changes.

From the Environments tab, you can see whether each development environment is in sync or not, see the differences if it's not in sync, and optionally synchronize. The synchronization process runs a deployment of the changes from the first pipeline stage branch into the development environment. For this reason, you're presented with deployment options when the synchronization process is initiated.

You can synchronize a development environment if it doesn't contain any In Progress work items.

1. From Pipeline, open the Environments tab.



   The status of the development environment, Pedro scratch org, indicates that it's out-of-sync.

2. (Optional) To see the differences between the dev environment and the next pipeline stage's branch, click **What's Different**.
3. Click **Sync**.
4. Select a synchronization option.

Changes Not in Dev Environment (recommended)	Deploy only the changed metadata (files) from the next stage's branch to your development environment.
All	Deploy everything in the next stage's branch to the development environment. Select this option if your deployments are failing due to dependent or missing metadata that doesn't exist in the What's Different list but does exist in the next stage's branch.

   The synchronization process runs a deployment of the changes from the first pipeline stage branch into the development environment. For this reason, you're presented with deployment options when the synchronization process is initiated.

   If the source environment exists in the project pipeline, then DevOps Center is aware of its ancestry and you can select **Changes Not in Dev Environment**. If you're synchronizing a scratch org or a sandbox where DevOps Center can't determine the ancestry, we recommend that you select **All** to

deploy all changes from the first pipeline stage's branch to this development environment.

5. Click **Start Sync**.

See Also
Promotion Options

Review Changes with Team Members

When changes for an In Progress work item are committed and ready to share with reviewers, creating a review opens a change request, also called a pull request.

A pull request is how you get your proposed changes reviewed and approved before they're merged with the rest of the code. The pull request pulls your changes from the work item branch so that they can be merged into the next branch in your pipeline. Before that happens, however, a team member reviews the work item changes. The pull request presentation format makes reviews and online discussion easier, because it highlights differences between the work item branch and the main branch.

Review Changes in GitHub
Create a change request (pull request) so that a team member can review and approve your changes.

Review Changes in Bitbucket (Beta)
Create a change request (pull request) so that a team member can review and approve your changes.

Review Changes in GitHub

Create a change request (pull request) so that a team member can review and approve your changes.

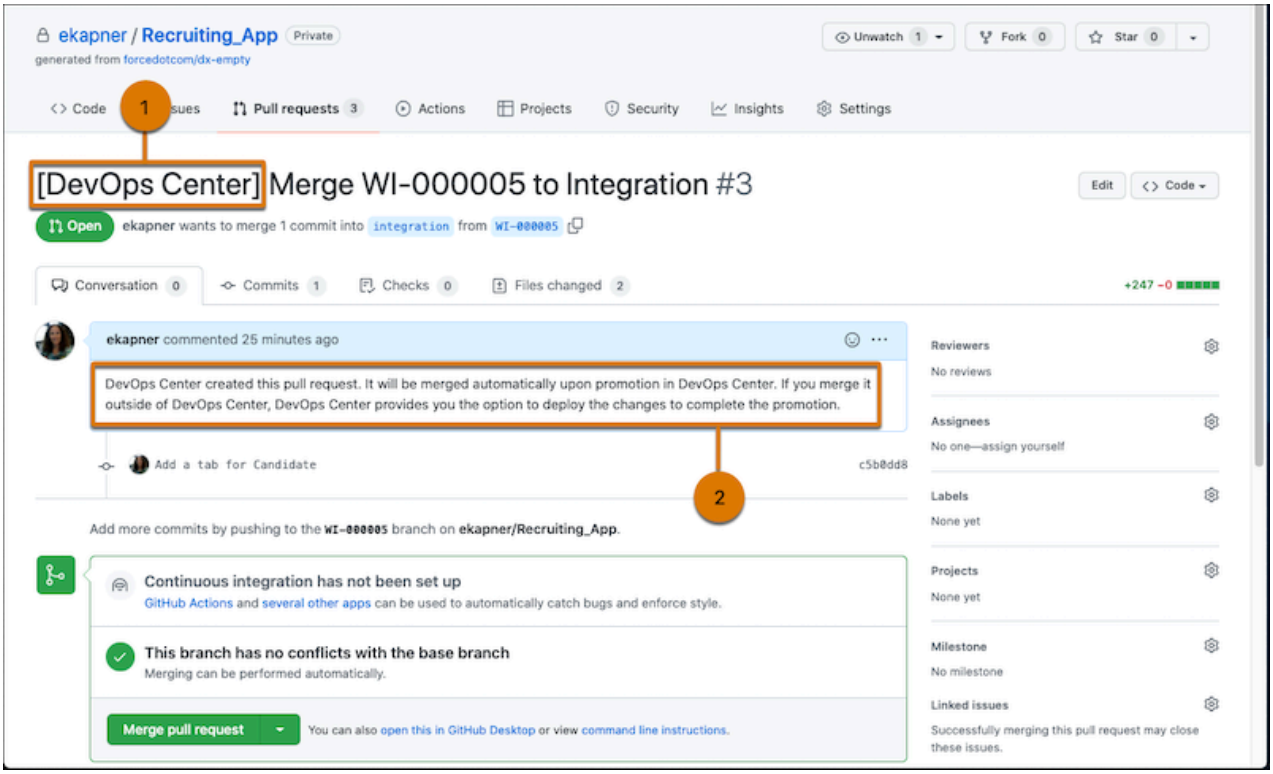
`main`.

1. From the work item, click **Create Review** to move the work item to In Review.

   DevOps Center creates a change request and automatically opens a pull request for the changes in your work item branch.

2. (Optional) Click View Change Request to open a browser tab showing the pull request in GitHub, where reviewers can see the file changes.

   You can identify that DevOps Center created the change request by the subject line (1) and auto-generated comment (2).



3. Team members can add comments and have discussion in the pull request about the changes.
4. Go back to DevOps Center.

   What you do next depends on whether you merged the pull request in GitHub.

   - If you didn't merge the pull request, indicate that the work item is ready to promote.
   - If you merged the pull request, the work item status is updated to **Ready to Promote**. The next step is to complete the promotion in DevOps Center or using Salesforce CLI.

To improve release quality, we recommend you set up and use DevOps Testing.

See Also

Indicate That Work Items Are Ready to Promote
Promote Work Items Through Your Pipeline

Review Changes in Bitbucket (Beta)

Create a change request (pull request) so that a team member can review and approve your changes.

Note DevOps Center Bitbucket Cloud support is a pilot or beta service that is subject to the Beta Services Terms at Agreements - Salesforce.com or a written Unified Pilot Agreement if executed by Customer, and applicable terms in the Product Terms Directory. Use of this pilot or beta service is at the Customer's sole discretion.

If you plan to perform most of your development work outside of DevOps Center, you can create the change (pull) request directly in the source control system. Be sure to select the pipeline stage's branch

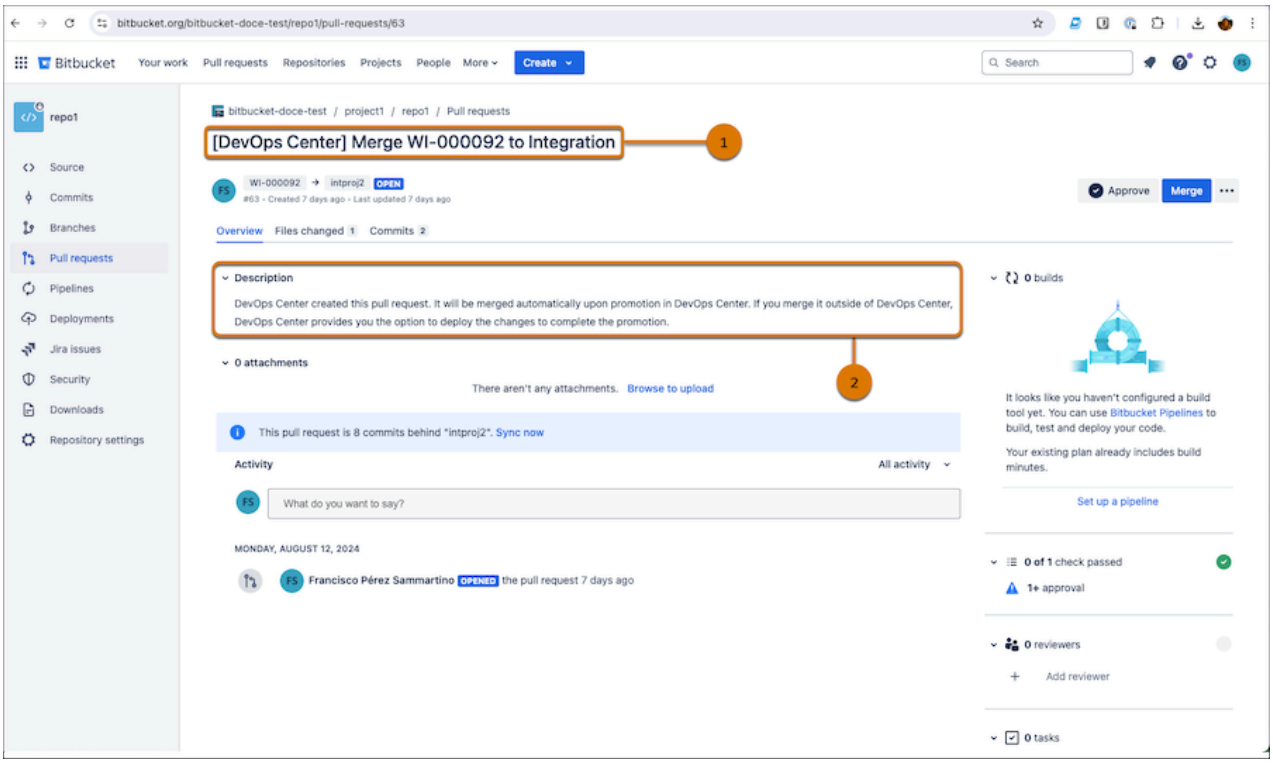
as the base branch instead of `main`.

1. From the work item, click **Create Review** to move the work item to In Review.

   DevOps Center creates a change request and automatically opens a pull request for the changes in your work item branch.

2. (Optional) Click View Change Request to open a browser tab showing the pull request in the source control repository, where reviewers can see the file changes.

   You can identify that DevOps Center created the change request by the subject line (1) and auto-generated description (2).



3. Team members can add comments and have discussion in the pull request about the changes.
4. Go back to DevOps Center.

   What you do next depends on whether you merged the pull request in Bitbucket.

   - If you didn't merge the pull request, indicate that the work item is ready to promote.
   - If you merged the pull request, the work item status is updated to **Ready to Promote**. The next step is to complete the promotion in DevOps Center or using Salesforce CLI.

Indicate That Work Items Are Ready to Promote

After the work item is reviewed, a team member approves the work item to be ready to promote. Work items aren't available for promotion through the pipeline until they are in the Ready to Promote status.

You can click the toggle again to indicate that the work item isn't yet ready to promote. This toggle is available until the work item has been promoted.

If you merged the pull request directly in the source control repository, the work item is automatically moved into the Ready to Promote status and can't be changed back. Because the change is now in the next stage's branch, your only option is to complete the promotion.

For guidelines regarding how many work items to have in a pipeline stage and how many to promote at one time, see Promote Work Items Through Your Pipeline.

External Change Requests and Merges

You can create a change request and merge your changes directly in the source control system, which is the first half of the promotion process. You can complete the promotion by deploying all externally merged changes to the pipeline stage's associated environment within DevOps Center or by using Salesforce CLI.

You can merge your changes in the source control repository or using `git` on the command line. To use `git` commands, see the Git documentation.

Merge Types

When you merge your changes directly in the source control repository, you have these merge options. For more information about merge types and how to merge, see the source control repository's documentation.

- Merge commit (default)–Enables you to see each commit that is included in the set of changes.
- Squash and merge (recommended)–Combines all commits into a single commit. You can look at a single commit to view your changes as a whole rather than sift through multiple commits.
- Rebase and merge (GitHub, don't use)–DevOps Center can't properly reflect the changes, which is likely to cause errors when these changes are promoted through the release pipeline.
- Fast forward (Bitbucket, don't use)–DevOps Center can't properly reflect the changes, which is likely to cause errors when these changes are promoted through the release pipeline.

Tips for Creating Change Requests and Merging Directly in the Source Control System

- Select the recommended merge option, Squash and Merge.
- Remember that all externally merged changes are deployed to the associated pipeline stage's environment. If you're unsure if you want to move a change forward, wait to merge it.
- Whether you perform your development work in an org or use an IDE, such as Salesforce Extensions for Visual Studio Code, you can create the change (pull) request directly in the source control repository. When merging work items in the first pipeline stage's branch, be sure to select the pipeline stage's branch as the base branch instead of `main`.

- If merging changes from one pipeline stage branch to the next branch in the pipeline, DevOps Center creates the change request for you. View the list of change requests, and then select the one you want to merge. Example: `[DevOps Center] Merge WI-000004 to UAT`.
- Never merge a change request with a title that looks like this: `[DO NOT MERGE] For DevOps Center use only - Integration to UAT`.

See Also

Mergeability Rules and Merge Settings
Promote Changes Merged Outside of DevOps Center
Deploy Changes Using Salesforce CLI (Beta)

Promote Work Items Through Your Pipeline

You promote work items through a pipeline, which defines the sequence of stages that work items progress as they go through the release lifecycle from development to production (or some other final stage). You can have any number of pipeline stages. Your team manager built the pipeline when configuring DevOps Center.

Each pipeline stage corresponds to an environment (currently a Salesforce org), and a branch in the source control repository. Depending on how your pipeline is configured, changes move through the pipeline when individual work items or a grouped set of work items (work item bundle) is promoted. Upon promotion, changes are merged from the current stage branch to the next stage branch, and then are deployed to the next stage org.

At a minimum, we recommend that your pipeline has one test stage and a production stage. However, it's common to have two to three test stages, often called something like integration, UAT (user acceptance testing), and staging.

Your team manager can configure your pipeline in one of two ways:

- Allow you to move work items individually through the entire pipeline.
- Allow you to move work items individually in early stages of the pipeline, and as a work item bundle in later stages. The point in the pipeline when you transition from the more flexible individually selectable work item promotion to the more predictable versioned promotion is referred to as the bundling stage. When changes are promoted from the bundling stage to the next stage, all work items that haven't yet been promoted are included in the versioned work item bundle and promoted as a unit. This versioned work item bundle continues to be promoted as a consistent unit through subsequent stages when you perform a promotion.

Your pipeline configuration determines how many stages allow you to promote individual work items.

- Stages that allow you to promote individual work items have the Promote Selected button at the top.
- Stages that allow you to promote a versioned work item bundle have the Promote Work Item Bundle or Promote Work Item Bundles button at the top.
- The bundling stage, where the versioned bundle is created, has the Promote as Work Item Bundle button at the top.

Special Characters in Metadata Component Names

When naming metadata components, DevOps Center and its underlying technologies support a subset of special characters. To avoid issues when deploying or promoting metadata to a pipeline stage, we strongly suggest using only supported special characters in your component names. See Supported Special Characters in Metadata Component Names for details.

Components in Managed Packages

Due to packaging manageability rules, you can't promote or modify most components that are installed in a managed package.

Best Practices for Non-Bundled Pipeline Stages

Ideally, it's best to keep work items moving through your pipeline so they don't stack up. If your team practices result in numerous work items in a stage, consider adding a bundling stage. To ensure that DevOps Center can efficiently process all the selected changes, follow these guidelines to avoid promotion errors.

- When promoting work items from the Approved Work Items list to the first pipeline stage, select up to approximately 20 work items at a time.
- When promoting work items from a non-bundled pipeline stage to the next pipeline stage, select up to approximately 30 work items at a time.
- Limit the total number of work items to approximately 50 or fewer for each non-bundled pipeline stage.

For work items in the Approved Work Items list, you have options to reduce the number of work items in your list. See Work Item Management for details.

Limit for Bundling Pipeline Stage

Limit the total number of work items to approximately 50 or fewer for the bundling pipeline stage.

Mergeability Rules and Merge Settings

DevOps Center blocks promotions if there's a conflict, if a test failed, or if a mergeability rule or merge setting isn't met.

Promotion Options

The available promotion options depend on if you're promoting individual work items, work item bundles, or performing a validate-only deployment.

Promote Individual Work Items

After a work item is considered "ready to promote," it's added to the Approved Work Items list. You can then promote individual work items to the first pipeline stage.

Promote Individual Work Items as a Work Item Bundle

During pipeline configuration, your project manager can define a stage as the bundling stage. Instead

of selecting individual work items to promote, you promote work items in this stage as a work item bundle. All unpromoted work items in this stage become part of the bundle.

Promote All Work Item Bundles

When you promote work item bundles, all unpromoted versioned bundles are promoted together to the next stage. The unpromoted work item bundles and corresponding work items are listed in this stage.

Run a Validate-Only Deployment

To ensure a successful deployment ahead of time, create a validate-only deployment. You can create a validation for only the bundling stage and versioned stages. You can run a validate-only deployment within DevOps Center or using Salesforce CLI.

Promote Changes Merged Outside of DevOps Center

If you merge the change request in the source control system, the changes are in a partially promoted state because they've been merged but not yet deployed to the associated pipeline environment. You can complete the promotion in DevOps Center or use Salesforce CLI.

Deploy Changes Using Salesforce CLI (Beta)

Use Salesforce CLI to deploy changes to a pipeline stage's environment or to perform a validate-only deployment. You must externally merge any changes in the source control repository before you can perform the deployment using the CLI. While the CLI command runs, you can see the progress of the promotion reflected in the DevOps Center Pipeline Stages tab.

Stalled Promotions

Stalled promotions can happen during several points in the promotion process. After you determine where the failure occurred, you can fix it.

View Promotion Status

When promotion completes, you see a success banner or a failure banner where you can see details of the error. You have several options to validate the promotion or dig into errors.

View Deployment Status to Dig Deeper

If the deployment error doesn't provide sufficient information, you can see more details in the Deployment Status in the org.

See Also

Salesforce Help: Plan Your Pipeline
Conflict Detection and Resolution

Mergeability Rules and Merge Settings

DevOps Center blocks promotions if there's a conflict, if a test failed, or if a mergeability rule or merge setting isn't met.

DevOps Center honors the source control system's mergeability rules and merge settings.

- Work items can be marked Ready to Promote only when the changes are deemed mergeable. Mergeability is determined either by merge conflicts or custom mergeability rules. If the source control system identifies merge conflicts with the changes, then the changes can't be promoted until the conflicts are resolved.

- Similarly, if custom mergeability rules or merge settings haven't been met for the changes, then the changes can't be promoted until those rules are met.

To improve release quality, we recommend you set up and use DevOps Testing.

See Also

GitHub documentation: Managing Protected Branches
Bitbucket documentation: Pull Request and Merge Settings

Promotion Options

The available promotion options depend on if you're promoting individual work items, work item bundles, or performing a validate-only deployment.

Merging Behavior for Promotions

- When a promotion runs, DevOps Center merges the branch from the originating stage into the target stage's branch. It then deploys the metadata from the target stage's branch to the target stage's org.
- If the deployment fails, we don't finalize the merge to the target stage's branch. All branches remain in the state they were before you attempted the promotion.
- If you perform a validate-only deployment, we don't merge any changes to the target stage's branch until you perform the Quick Deploy.

Deploy Changes Using Salesforce CLI

You can deploy externally merged changes to the associated pipeline environment or perform a validate-only deployment using Salesforce CLI. See External Change Requests and Merges and Deploy Changes Using Salesforce CLI (Beta) for details.

Promotion Options (Bundling and Versioned Pipeline Stages Only)

If you're promoting from the bundling or a versioned pipeline stage, we provide these promotion options.

- **Promote now**–Promote all versioned work item bundles to the next pipeline stage.
- **Run a validate-only deployment**–Run a validation to check the results of Apex tests and deploying components without saving any components in the org. See Run a Validate-Only Deployment for details.

Changes to Promote

When you promote individual work items through the pipeline, you can select whether to deploy only the items that changed or all the changes in the target stage's branch.

- **Changes not in the <stage-name>'s branch** (recommended)–Deploy only the files from that target stage's branch that aren't yet in the target stage's org. In effect, you're deploying only the files (changes) that recently came from the originating stage as part of this promotion. This behavior is the default, and is more efficient because we're deploying fewer files. In most circumstances, selecting this option is sufficient.
- **All metadata in the <stage-name>'s branch**–Deploys all files in the target stage's branch to the target stage's org, after the branch has been merged. Select this option if your deployments are failing due to dependent or missing metadata. In some cases, the metadata doesn't exist in the defined set of changes yet exists in the target stage branch. This option is useful when you or a colleague has committed and merged changes to the branch outside of DevOps Center but DevOps Center doesn't know about them.

Apex Test Options

When you promote work items through the pipeline, decide which Apex tests to run based on pipeline stage and to which environments you're deploying changes.

- **Default**–In sandboxes and scratch orgs, no tests are run. In production, all local tests are run if your changes contain Apex classes or triggers. This option excludes tests included in installed packages.
- **Run local tests**–All local tests are run, excluding tests included in installed packages.
- **Run all tests**–All tests in your org are run, including tests from installed packages.
- **Run specified tests**–Only the tests that you specify are run. Provide the names of test classes in a comma-separated list (no spaces between entries). Example: `TestDeleteData`, `TestDataFactory.createTestRecords`

See Also

Apex Developer Guide: Testing and Code Coverage

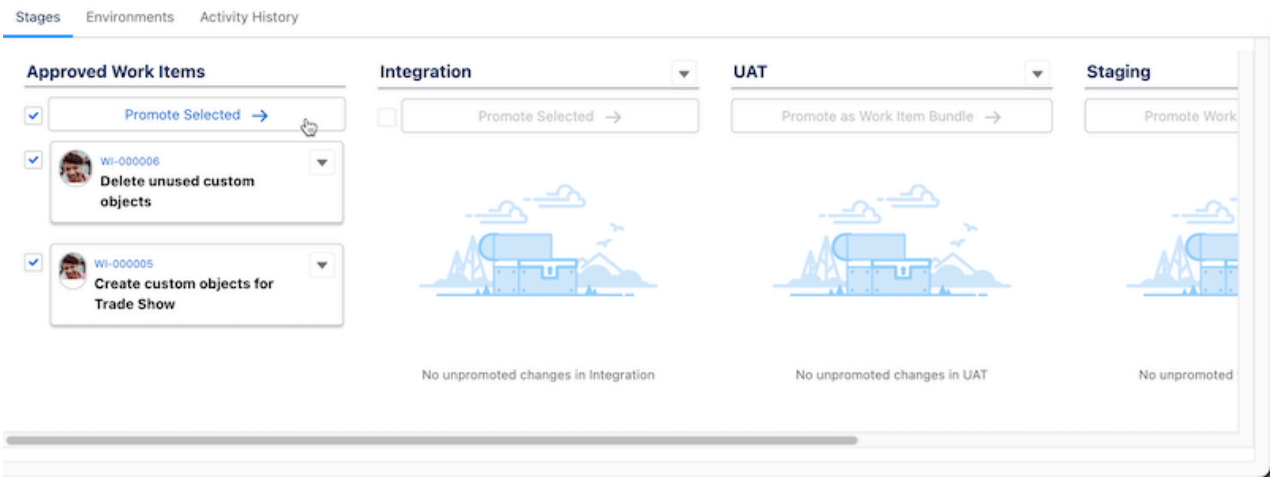
Promote Individual Work Items

After a work item is considered "ready to promote," it's added to the Approved Work Items list. You can then promote individual work items to the first pipeline stage.

To ensure DevOps Center can process all the changes during the promotion, follow these guidelines to avoid promotion errors.

- Limit the total number of work items to approximately 50 or fewer in each non-bundled pipeline stage.
- When promoting work items from the Approved Work Items list, select up to 15 work items at a time.
- When promoting work items from a non-bundled pipeline stage, up to 25 work items at a time.

1. Click **Pipeline**.
2. Under Approved Work Items, select the work items to promote to the next stage.
3. Click **Promote Selected**.



4. If you haven't logged in to the environment for the next pipeline stage from within DevOps Center, complete the login process, then reselect the items to promote and click **Promote Selected**.
5. In the Promotion Options dialog, select a promotion option and which Apex tests to run.
6. Click **Promote**.

   If DevOps Center doesn't detect any conflicts or shared components, the selected work items are merged and deployed to the environment associated with the next stage.

7. (Optional) If DevOps Center detects a dependency or shared components for the selected work items, DevOps Center provides you with the option to combine the work items to address the conflict.

   Proceed by choosing one of these options:

   - Combine the work items.
     For either shared components or a dependency, combine the work items. DevOps Center uses the latest versions of the conflicting components. See Prevent Potential Conflicts By Combining Work Items. Combining work items merges all changes into one feature branch so you avoid downstream merge conflicts.
   - Attempt to promote the originally selected work items.
     If shared components are detected, promote the selected work items. However, it's likely that the other work items, which you don't choose to promote, still contain conflicts. The promotion is blocked until you manually fix the conflict in the source control system. For GitHub, see Review and Resolve Conflicts in GitHub. For Bitbucket, see the Bitbucket documentation.
   - Promote the work item even though there's a dependency with another work item.
     If the promotion is blocked, promote the dependent work item at a later time. However, it's likely that the other work items still have conflicts. The promotion is blocked until you manually fix the conflict in the source control system. For GitHub, see Review and Resolve Conflicts in GitHub. For Bitbucket, see the Bitbucket documentation.

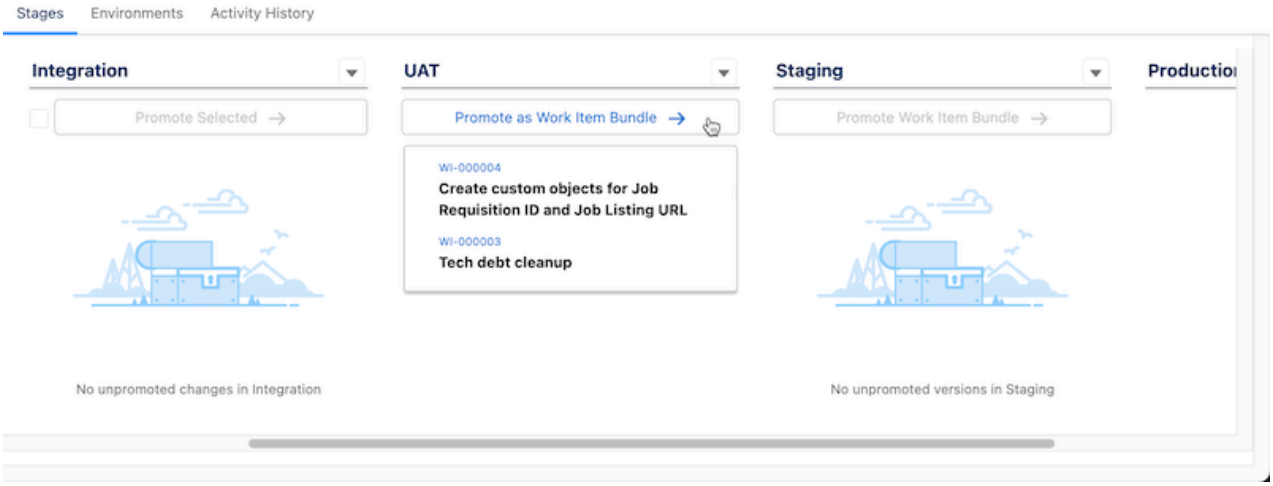
See Also

Promotion Options

Promote Individual Work Items as a Work Item Bundle

During pipeline configuration, your project manager can define a stage as the bundling stage. Instead of selecting individual work items to promote, you promote work items in this stage as a work item bundle. All unpromoted work items in this stage become part of the bundle.

1. In the Pipeline view, in the bundling stage, click **Promote as Work Item Bundle**.



2. If you haven't logged in to the environment for the next pipeline stage from within DevOps Center, complete the login process, then click **Promote as Work Item Bundle**.
3. In the Promotion Options dialog, select **Promote Now**, the changes to promote, and which Apex tests to run.

   Indicate the **Version** as an alphanumeric string of your choice. To perform a validate-only deployment, see Run a Validate-Only Deployment.

4. Select which changes to promote and which Apex tests to run.
5. Click **Promote**.

   The work item bundle is created with the version you specified, then merged and deployed to the environment associated with the next stage.

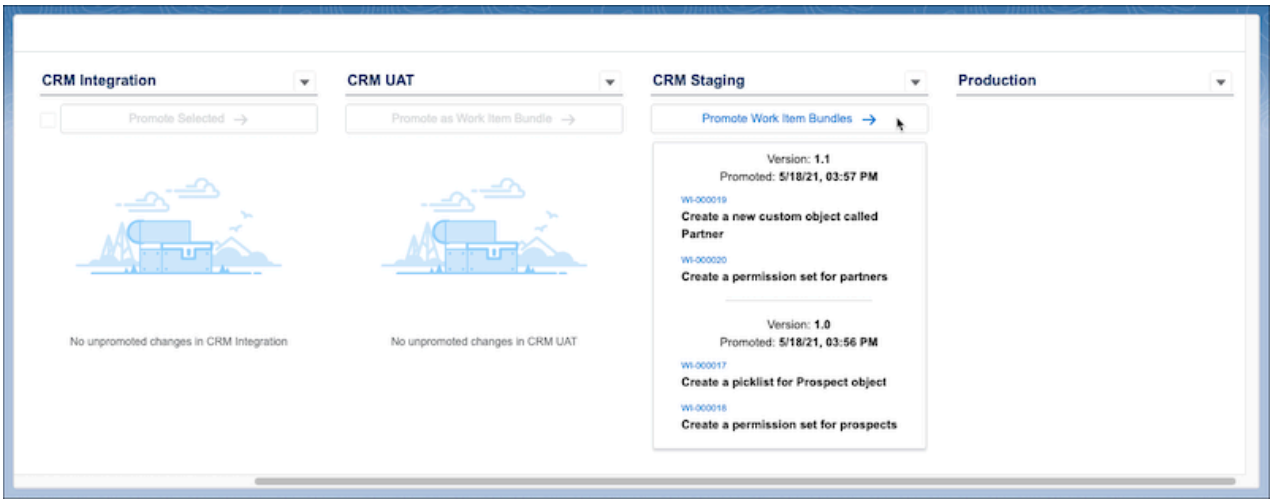
See Also

Promotion Options

Promote All Work Item Bundles

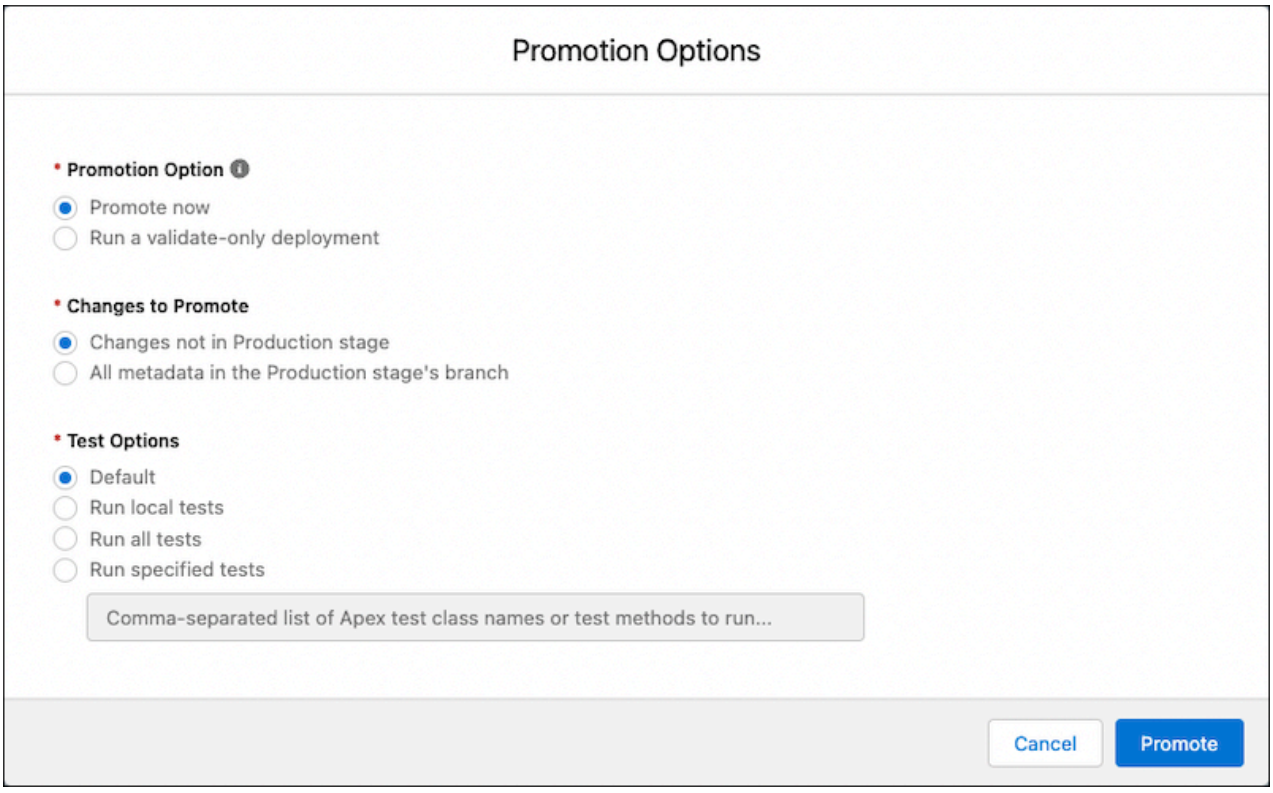
When you promote work item bundles, all unpromoted versioned bundles are promoted together to the next stage. The unpromoted work item bundles and corresponding work items are listed in this stage.

1. In the Pipeline view, in a versioned stage, click **Promote Work Item Bundle(s)**.



2. If you haven't logged in to the environment for the next pipeline stage from within DevOps Center, complete the login process, then click **Promote Work Item Bundle(s)**.
3. Select **Promote Now**, the changes to promote, and which Apex tests to run.

   To perform a validate-only deployment, see Run a Validate-Only Deployment.



4. Click **Promote**.

   Each versioned work item bundle is merged and deployed to the environment associated with the next stage. Only the latest work item bundle version is displayed, but all earlier work item bundles are also present in the stage.

See Also

Promotion Options

Run a Validate-Only Deployment

To ensure a successful deployment ahead of time, create a validate-only deployment. You can create a validation for only the bundling stage and versioned stages. You can run a validate-only deployment within DevOps Center or using Salesforce CLI.

Note To run a validate-only deployment using Salesforce CLI, you must first externally merge changes in the source control repository before running the CLI command. See Deploy Changes Using Salesforce CLI (Beta) for details.

A validation checks the results of Apex tests and deploying components but doesn't save any components in the org. A validation enables you to view the success or failure messages that you'd receive with an actual deployment.

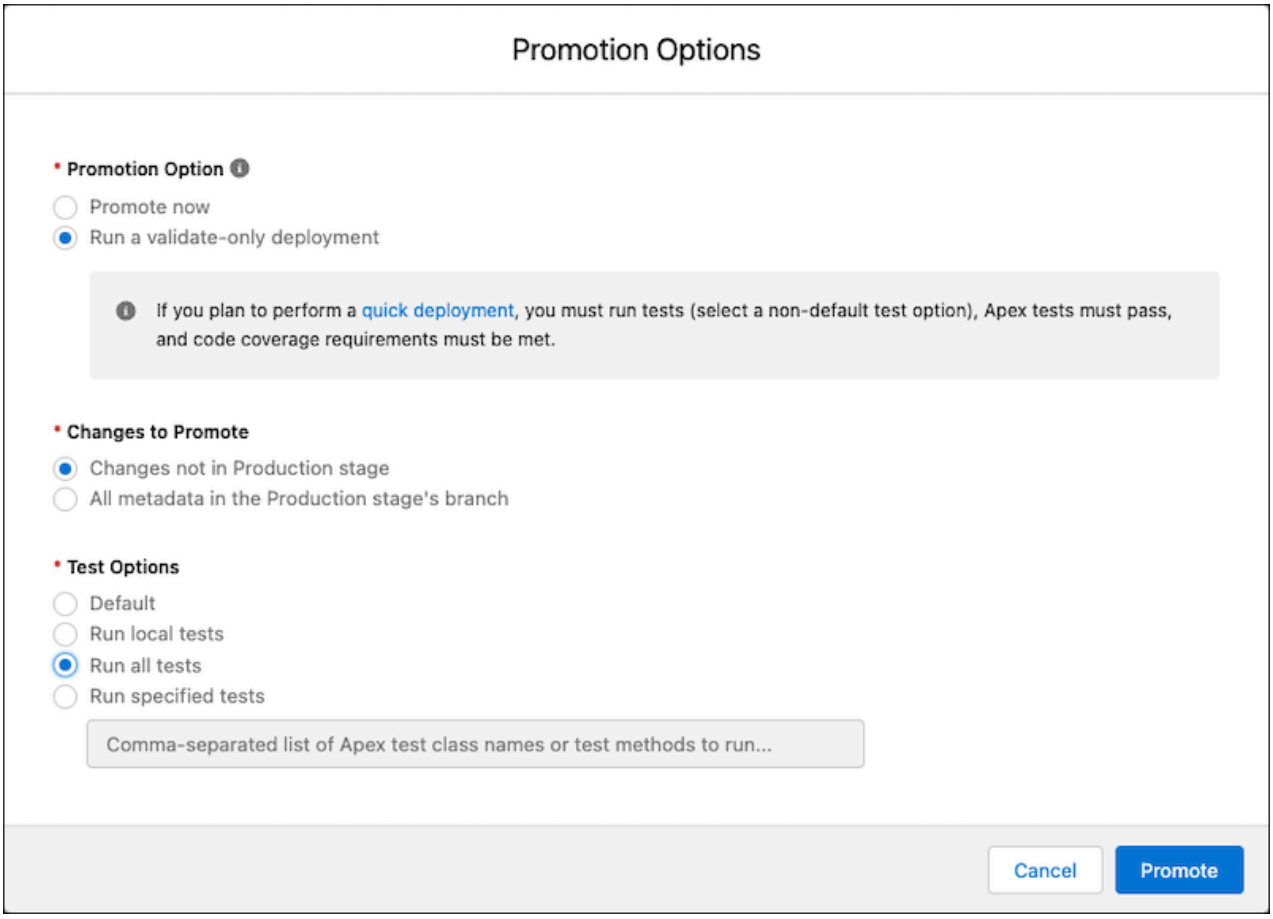
Later, you can perform a quick deployment to promote changes to a pipeline stage or to your release environment without having to run tests.

1. In DevOps Center, go to Pipelines Stages.
2. In the bundling or a versioned stage, click **Promote as Work Item Bundle**, **Promote Work Item Bundle**, or **Promote Work Item Bundles**, as applicable.

   If you have an existing validation, you see a warning indicating that any subsequent promotion disqualifies the existing validation for quick deployment.

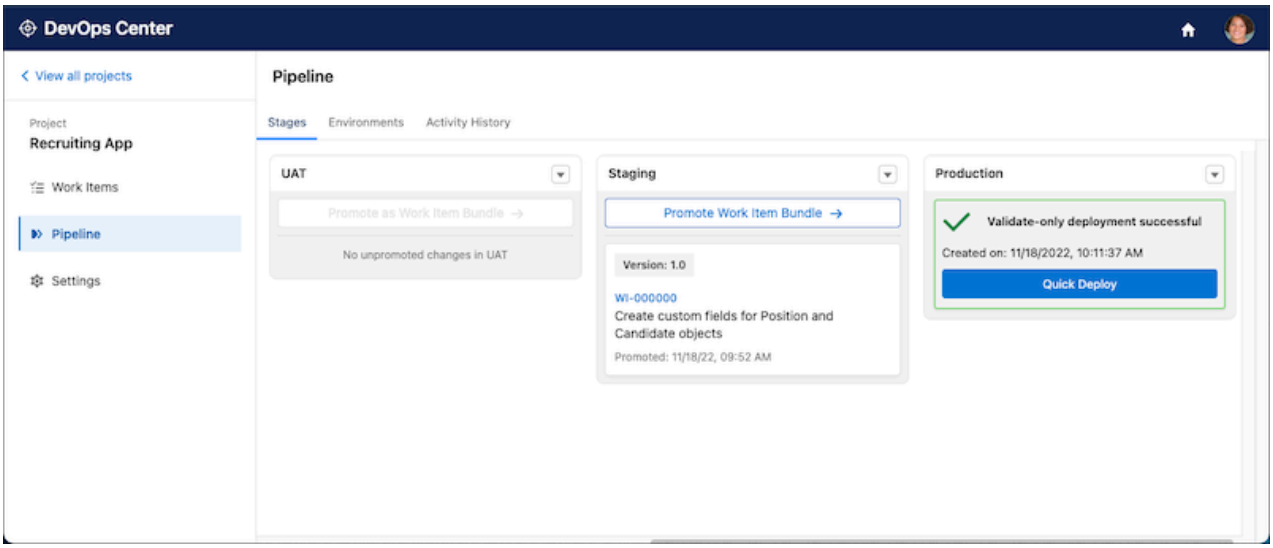
3. For Promotion Option, click **Run a Validate-Only Deployment**.
4. Select the changes to promote.
5. Select the appropriate test option.

   If you plan to run a quick deployment for this validation, you must run tests, the Apex tests must pass, and code coverage requirements must be met.



6. Click **Promote**.

   If you ran the validation without running tests, the Quick Deploy button isn't available. A successful validate-only deployment is valid for 10 days. If the validation fails, see Activity History for details.



7. Within 10 days, click **Quick Deploy** to complete the promotion.

See Also

Promotion Options

Salesforce Help: Monitor Deployments

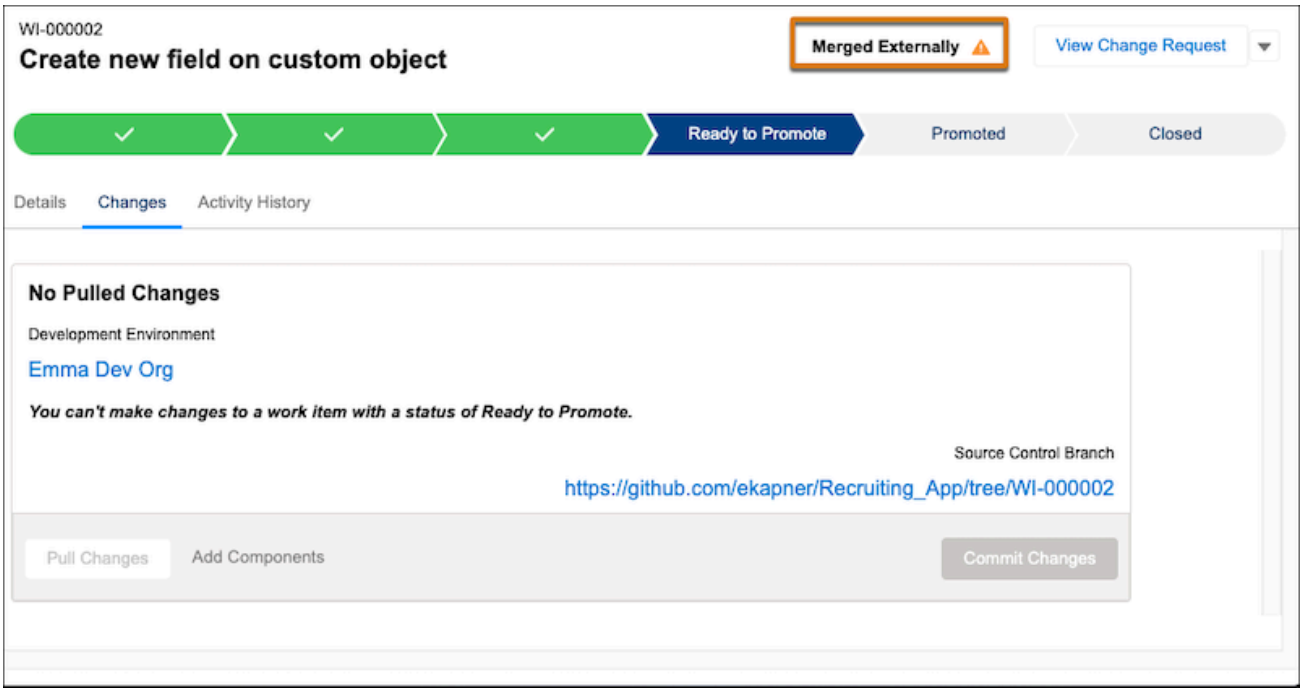
Promote Changes Merged Outside of DevOps Center

If you merge the change request in the source control system, the changes are in a partially promoted state because they've been merged but not yet deployed to the associated pipeline environment. You can complete the promotion in DevOps Center or use Salesforce CLI.

Note See External Change Requests and Merges for tips about creating the change request and merging the changes directly in the source control system.

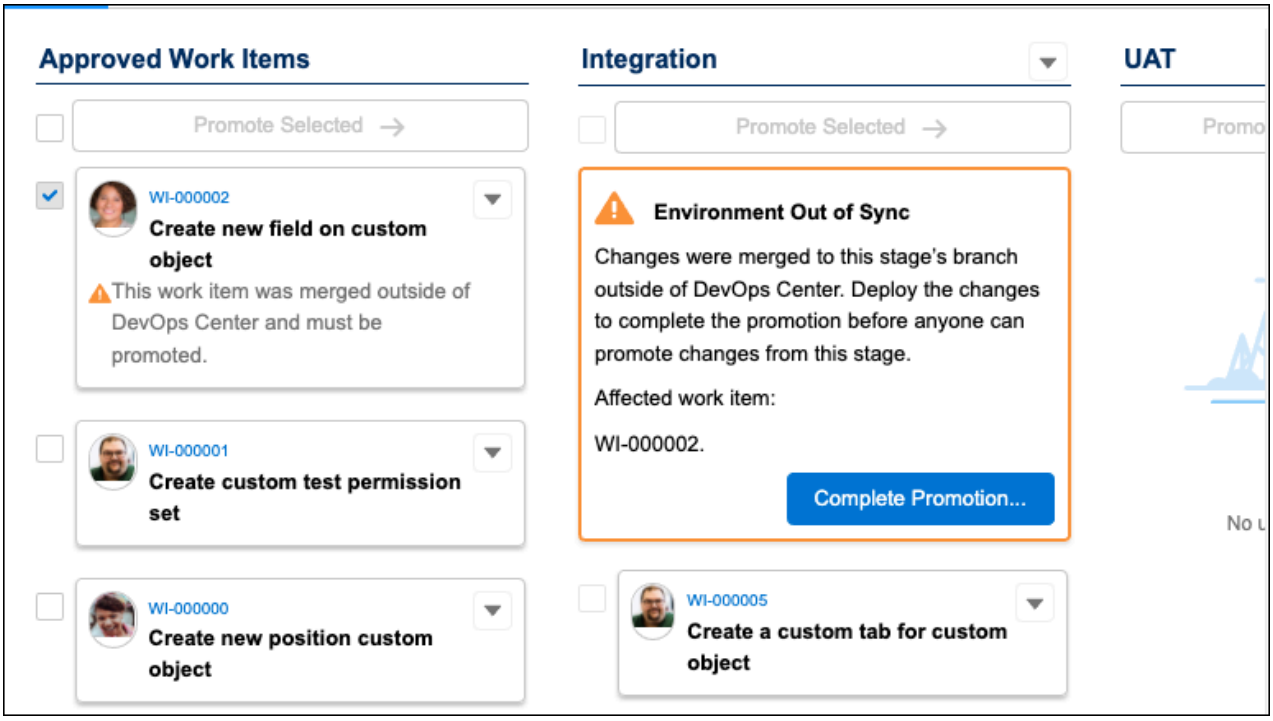
To keep everything in sync, whether you merged changes for a work item or changes in a downstream pipeline branch, you must finish the promotion before any more changes can be promoted to or from the stage.

- To complete the promotion using Salesforce CLI, see Deploy Changes Using Salesforce CLI (Beta).
- To complete the promotion in DevOps Center, go to Pipeline Stages and follow these steps.



Important If you're working directly in GitHub, delays can occur between when these actions happen and when they're reflected in DevOps Center. If you take subsequent actions on the related objects in DevOps Center before they're reflected, you're likely to see unexpected or inconsistent behavior. For metadata changes, we recommend that you wait for delayed events to be reflected in DevOps Center. For non-metadata changes, such as updates to the `.forceignore` file, you can continue with the promotion.

1. In DevOps Center, click **Pipeline**.
2. In the Pipeline Stages tab, click **Complete Promotion**.



Deploy Changes Using Salesforce CLI (Beta)

Use Salesforce CLI to deploy changes to a pipeline stage's environment or to perform a validate-only deployment. You must externally merge any changes in the source control repository before you can perform the deployment using the CLI. While the CLI command runs, you can see the progress of the promotion reflected in the DevOps Center Pipeline Stages tab.

Note This feature is a Beta Service. Customer may opt to try such Beta Service in its sole discretion. Any use of the Beta Service is subject to the applicable Beta Services Terms provided at Agreements and Terms.

The DevOps Center CLI plugin is compatible with DevOps Center package version 6.0 or later. The DevOps Center plugin (`plugin-devops-center`) is installed the first time you run a CLI command. To install and configure the CLI, see the Salesforce CLI Setup Guide.

Salesforce CLI releases on a weekly basis. See the Salesforce CLI Release Notes for information on the timing for the official release of the DevOps Center plugin.

The CLI commands for DevOps Center include:

Command	Description
`project deploy pipeline start`	Deploy changes from a branch to the pipeline stage's org.
`project deploy pipeline report`	Check the status of a pipeline deployment

Command	Description
	operation.
`project deploy pipeline resume`	Resume watching a pipeline deployment operation.
`project deploy pipeline validate`	Perform a validate-only deployment from a branch to the pipeline stage's org. You can create a validation for only the bundling stage and versioned stages.
`project deploy pipeline quick`	Quickly deploy a validated deployment to an org.

To see all command options, you can view help directly in a terminal or command window. Append `--help` to the command name. In this example, all `project deploy pipeline` commands are listed:

```
sf project deploy pipeline --help
```

To view condensed help for a command, use the `-h` flag. For detailed information on all CLI commands, see the Salesforce CLI Command Reference.

To use the DevOps Center CLI commands, you must:

- Be assigned the DevOps Center Release Manager permission set. This permission set enables you to perform promotions. Ask a Salesforce admin or project manager for assistance.
- Authorize the org in which DevOps Center is installed. The easiest way to authorize an org is with the `org login web` command. See Authorization in the *Salesforce DX Developer Guide* for details.

The following examples demonstrate how to deploy changes to a pipeline stage's environment using Salesforce CLI. First, changes must be merged to a pipeline stage branch directly in the source control repository. When the `pipeline deploy` command is run, all externally merged changes are deployed to the corresponding pipeline stage environment.

Tip To avoid having to indicate the `--devops-center-username` each time you run a DevOps Center CLI command, you can set it using a configuration variable. See `target-devops-center` in the *Salesforce CLI Setup Guide*: Configuration Variables for details.

In these examples, line breaks are added for readability purposes.

In this first example, several work item feature branches were merged to the Integration branch. To deploy the changes from the Integration branch to the Integration environment:

```
sf project deploy pipeline start --branch-name recruit-integration
--devops-center-username MyDevOpsCenterOrg --devops-center-project-name "Recru
iting App"
```

In this example, UAT is the bundling stage, which means that you indicate a version identifier. To deploy all merged changes in a version 1.0 bundle to the UAT environment:

```
sf project deploy pipeline start --branch-name recruit-uat --devops-center-use
rname
MyDevOpsCenterOrg --devops-center-project-name "Recruiting App" --bundle-versi
on-name 1.0
```

In this example, you're ready to validate the changes in the Staging branch so you can perform a quick deployment to production. When ready, you can perform the quick deployment in DevOps Center Pipeline Stages. To run a validate-only deployment:

```
sf project deploy pipeline validate --branch-name recruit-staging --devops-cen
ter-username
MyDevOpsCenterOrg --test-level RunLocalTests --devops-center-project-name "Rec
ruiting App"
```

See Also

Salesforce CLI Command Reference: sf project deploy pipeline

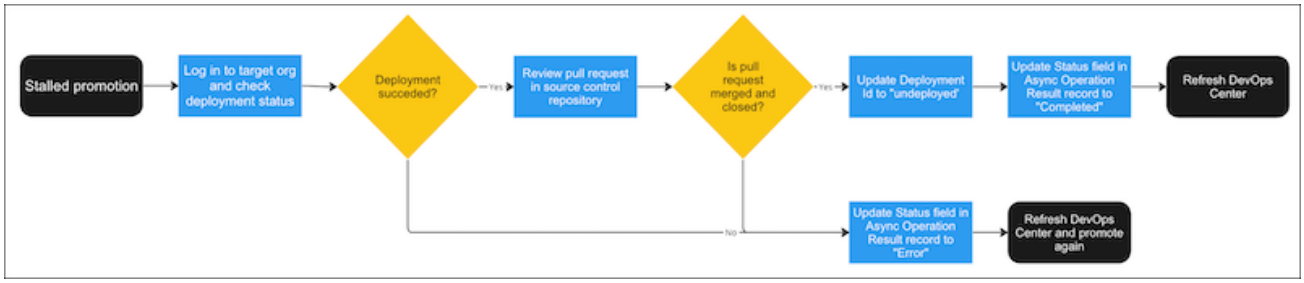
Stalled Promotions

Stalled promotions can happen during several points in the promotion process. After you determine where the failure occurred, you can fix it.

The promotion process comprises two main steps: deploying the selected metadata to the target stage's org, and pushing the merged metadata to the target stage's branch. A stalled promotion can occur when updates from external services back to DevOps Center get interrupted during one of these operations. This communication breakdown results in one of these types of stalled promotions:

- Promotion failed–the changes weren't promoted to either the target pipeline environment or the associated target pipeline branch. To fix the issue, execute a query to find the affected record, then change the Status to Error so you can try the promotion again.
- Promotion succeeded–the changes were promoted to both the target pipeline environment and the associated target pipeline branch, but weren't properly reflected in DevOps Center. To fix the issue, execute a query to find the affected record, then change to Status to Completed.
- Deployment succeeded but the branch wasn't merged–the changes exist in the target pipeline environment but don't appear in the associated target pipeline branch. To fix the issue, execute a query to find the affected record, then change the status to Error so you can try the promotion again.

To determine how to proceed to fix the stalled promotion, first troubleshoot to discover where the error occurred.



1. Log in to the target org and check the deployment status.

   From Setup, type *Deployment Status*, then select **Deployment Status**.

2. Check to see if the deployment failed.
   - If the deployment failed, reset the status of the work item and try again. See Fix a Stall Promotion When the Promotion Failed for instructions.
   - If the deployment succeeded, continue to the next step.
3. If the deployment succeeded, check to see if the pull (change) request in the source control system was merged.
   a. View the pull request in the source control repository. If you see the Stalled Promotion error, you can use the dropdown to view the change request. If the promotion message says “Pushing,” the dropdown menu is disabled. Go to the source control system directly and select the project repository to see the list of change requests.
   b. Click **Pull Requests**.
   c. If all the pull requests are closed, the promotion was successful but not properly reflected in DevOps Center. See Fix a Stalled Promotion when the Promotion Succeeded for instructions. If you see open pull requests, continue to the next step.
4. Look for the pull requests associated with your stalled promotion.
   - If you attempted to promote all work items in the stage, look for a pull request with the title that begins with `[DO NOT MERGE] For DevOps Center use only - <source stage> to <target stage>`.
   - If you selected a subset of items to promote, look at each pull request that contains the work item ID, for example, `[DevOps Center] Merge WI-000002 to <target stage>`.
   - If you attempted to promote a work item bundle, look for the pull request with the associated source and target stage, for example, `[DevOps Center] <source stage> to <target stage>`.
5. Determine what to do next.
   - If all the associated pull requests are merged (closed), see Fix a Stalled Promotion when the Promotion Succeeded for instructions.
   - If any of the associated pull requests weren’t merged (still open), update the status of the promotion record and try the promotion again. See Fix a Stalled Promotion When Deployment Succeeds But Merging to Branch Fails for instructions.

Fix a Stalled Promotion When the Promotion Failed

After you determine that your stalled promotion is due to a failure, update the status of the record and try the promotion again.

Fix a Stalled Promotion When the Promotion Succeeded
After you determine that the stalled promotion was successful but not property reflected in DevOps Center, update the status of the Async Operation Result record to update the results in DevOps Center.

Fix a Stalled Promotion When Deployment Succeeds But Merge to Branch Fails
After you determine that the deployment succeeded but the changes aren't reflected in the associated pipeline branch, update the status of the Async Operation Result record, then try the promotion again.

Fix a Stalled Promotion When the Promotion Failed

After you determine that your stalled promotion is due to a failure, update the status of the record and try the promotion again.

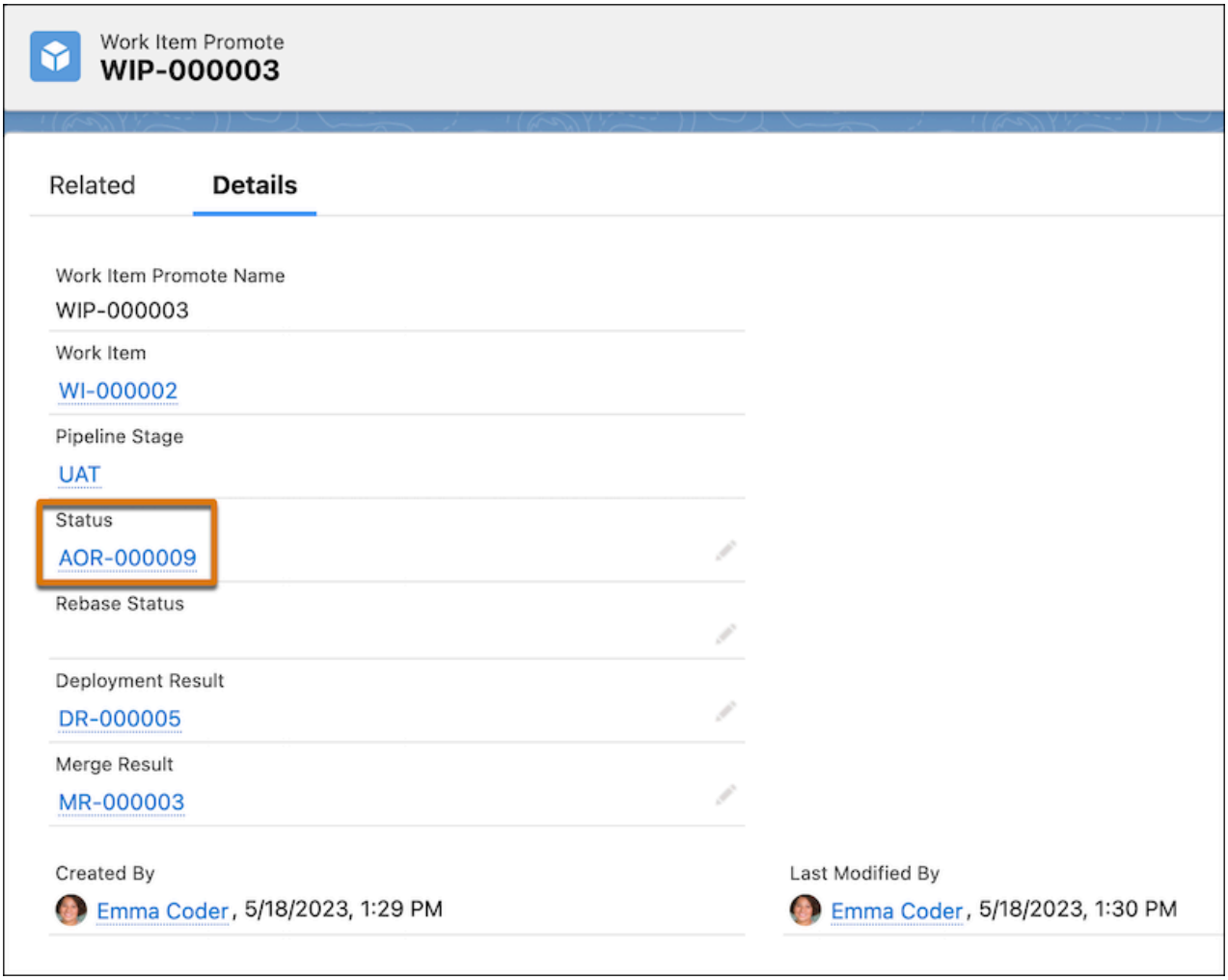
To determine the cause of the stalled promotion, see Stalled Promotions.

1. In the org in which DevOps Center is installed, launch Developer Console, then select the **Query Editor** tab.
2. To find the promotion record that's still In Progress, enter this SOQL query.

   In this example, line breaks were added for readability purposes. If you copy this query, remove the line breaks before running it.

   ```
   SELECT Id, sf_devops__Status__c FROM sf_devops__Work_Item_Promote__c
   WHERE sf_devops__Status__r.sf_devops__Status__c = 'In Progress'
   ```

3. Click to highlight the record, then click **Open Detail Page**.
4. In the Work Item Promote record, click the link in the Status field to view the associated Async Operation Result record.



5. In the Async Operation Result record, change the Status to **Error**, then click **Save**.
6. In DevOps Center, refresh the Pipeline Stages view to remove the Stalled Promotion error.
7. Promote your changes.

See Also

Fix a Stalled Promotion When the Promotion Succeeded
Fix a Stalled Promotion When Deployment Succeeds But Merge to Branch Fails

Fix a Stalled Promotion When the Promotion Succeeded

After you determine that the stalled promotion was successful but not property reflected in DevOps Center, update the status of the Async Operation Result record to update the results in DevOps Center.

To determine the cause of the stalled promotion, see Stalled Promotions.

1. In the org in which DevOps Center is installed, launch Developer Console, then select the **Query Editor** tab.
2. To find the promotion record that's still In Progress, enter this SOQL query.

   In this example, line breaks were added for readability purposes. If you copy this query, remove the

line breaks before running it.

```
SELECT Id, sf_devops__Status__c FROM sf_devops__Deployment_Result__c
WHERE sf_devops__Status__r.sf_devops__Status__c = 'In Progress'
```

If this query returns multiple records, query the Work Item object to determine which Deployment Result record you're interested in. Run this query for each record ID until you find the affected work item, and then use that Deployment Result record in later steps. In this example, the record ID is `a008N000002ImaKQAS`.

```
SELECT Id, Name FROM sf_devops__Work_Item__c WHERE sf_devops__Operation_Stat
us__c
= 'a008N000002ImaKQAS'
```

3. Click to highlight the Deployment Result record, then click **Open Detail Page**.
4. In the Deployment Result record, set the Deployment Id to `undeployed`, then click **Save**.
5. Next, click the link in the Status field, **AOR-<id>**, to go to the corresponding Async Operation Result record.
6. In the Async Operation Result record, change the Status field to **Completed**, then click **Save**.
7. In DevOps Center, refresh the Pipeline Stages view to remove the Stalled Promotion error.

See Also

Fix a Stalled Promotion When the Promotion Failed
Fix a Stalled Promotion When Deployment Succeeds But Merge to Branch Fails

Fix a Stalled Promotion When Deployment Succeeds But Merge to Branch Fails

After you determine that the deployment succeeded but the changes aren't reflected in the associated pipeline branch, update the status of the Async Operation Result record, then try the promotion again.

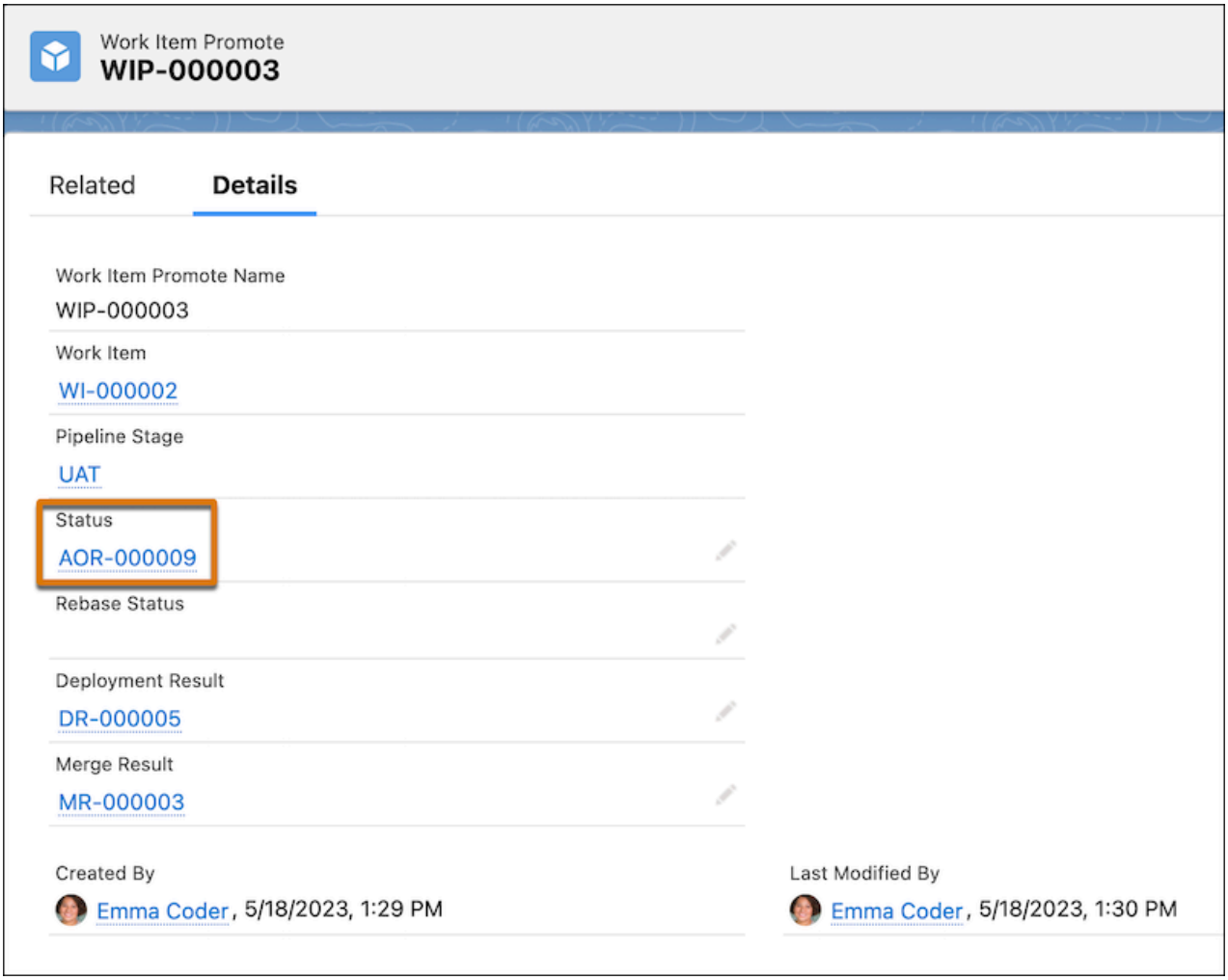
To determine the cause of the stalled promotion, see Stalled Promotions.

1. In the org in which DevOps Center is installed, launch Developer Console, then select the **Query Editor** tab.
2. To find the promotion record that's still In Progress, enter this SOQL query.

   In this example, line breaks were added for readability purposes. If you copy this query, remove the line breaks before running it.

```
SELECT Id, sf_devops__Status__c FROM sf_devops__Work_Item_Promote__c
WHERE sf_devops__Status__r.sf_devops__Status__c = 'In Progress'
```

3. Click to highlight the record, then click **Open Detail Page**.
4. In the Work Item Promote record, click the link in the Status field to view the associated Async Operation Result record.



5. In the Async Operation Result record, change the Status to **Error**, then click **Save**.
6. In DevOps Center, refresh the Pipeline Stages view to remove the Stalled Promotion error.
7. Promote your changes.

See Also

Fix a Stalled Promotion When the Promotion Failed
Fix a Stalled Promotion When the Promotion Succeeded

View Promotion Status

When promotion completes, you see a success banner or a failure banner where you can see details of the error. You have several options to validate the promotion or dig into errors.

From the pipeline stage dropdown menu (1):

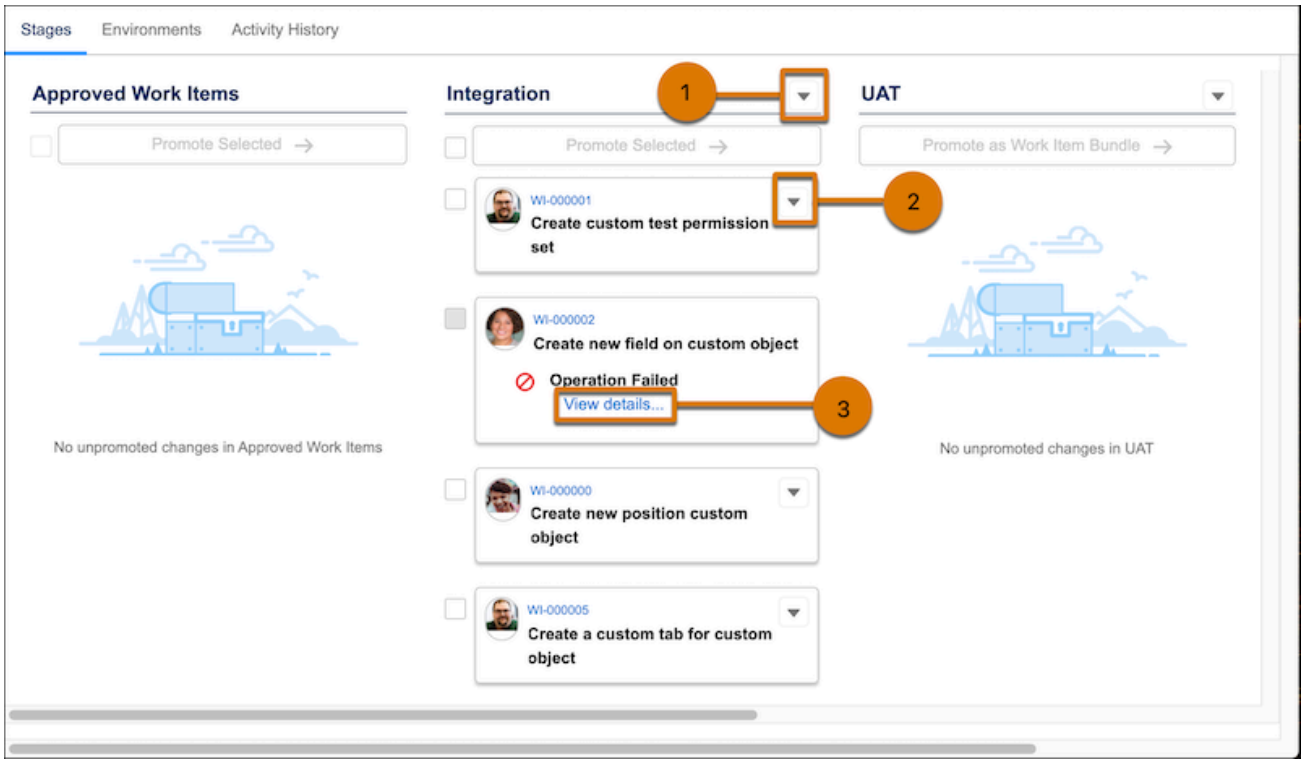
- Open Environment – opens the environment so you can validate changes.
- View Branch in Source Control – opens the branch in source control so you can view the files in this branch.
- View Last Commit in Source Control – opens a list of commits (individual changes to a set of files) in source control. Here, you can see what's changed between two branches.

From the Work Item:

- View Change Request from the work item dropdown menu (2) – opens the pull request in source control so you can view the changes and troubleshoot any merge conflicts.
- View details (3) – opens the Activity History tab for a failed promotion, where you can view more information.

From the Activity History tab:

- View the promotion details including work items that were part of the promotion.
- View whether the promotion succeeded or failed, failure details, and the date and time of completion.



See Also

Stalled Promotions
View Pipeline Events in Activity History

View Deployment Status to Dig Deeper

If the deployment error doesn't provide sufficient information, you can see more details in the Deployment Status in the org.

DevOps Center provides a menu option to quickly launch the Deployment Status setup page.

1. In Pipeline Stages, from the pipeline stage's context menu, select **View Deployment Status in Setup**.
2. From the list of Succeeded or Failed deployments, click **View Details**.
3. View and verify the actual changes in the org.

See Also

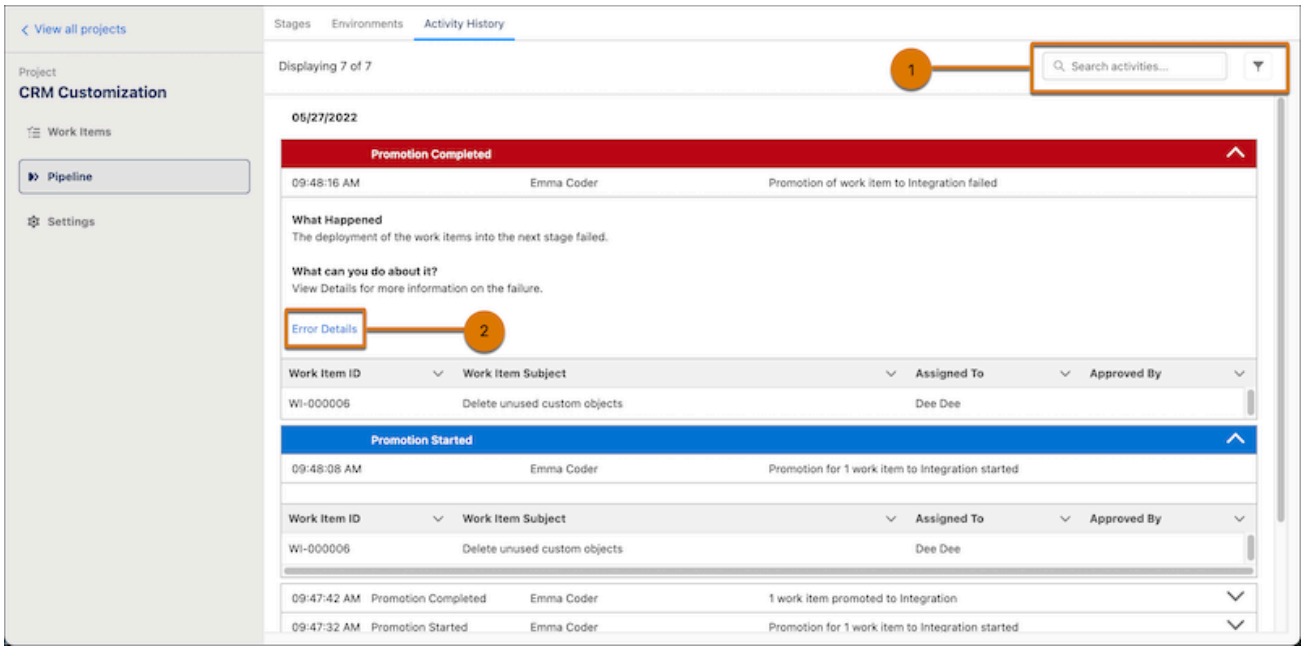
Salesforce Help: Monitor Deployments

View Pipeline Events in Activity History

DevOps Center provides a comprehensive history of all key pipeline events, including promotions, synchronizations, and errors (failures). You can use these events and their associated details for auditing, troubleshooting, and general visibility purposes.

Activity histories can get long, so we give you the ability to narrow the list (1).

- To filter on User, Activity Type, and Date Range, use the filter.
- Use the Search Activities box to search using keywords.



For promotion errors, click Error Details (2) to get more information to assist you in pinpointing and resolving the issue.

Update the Project's Source API Version Each Salesforce Release

Because projects in DevOps Center require a source control repository that uses the Salesforce DX project structure, each DevOps Center project contains a configuration file, sfdx-project.json. During each Salesforce release cycle, make a plan to update the source API version to avoid metadata type version mismatch errors when moving changes through your release pipeline.

When you create a new project source control repository, each pipeline and feature branch created from the main branch contains its own version of the DX project configuration file. However, this file isn't updated automatically during Salesforce release transitions. In most cases, the only value that requires updating is the **sourceApiVersion** option.

The **sourceApiVersion** option in `sfdx-project.json` determines the version of metadata that is retrieved when you commit changes to a work item feature branch, or when you promote changes through the release pipeline. This value is important if you're using a metadata type that has changed in a recent release or if you want access to new metadata types during or after a release transition.

For example, let's say a new field was added to the CustomTab object for API version 63.0. If the **sourceApiVersion** in your `sfdx-project.json` file is version 62.0, you see errors when promoting changes that include the new field.

If you're working in an existing DevOps Center project for a while, it's likely that the **sourceApiVersion** is outdated.

Update During Development When Working on a New Work Item

If you're actively working on changes, this method is the easiest for updating the source API version in your pipeline.

If the source API version isn't updated, some unexpected behaviors occur:

- When you try to add components manually to a work item, you don't see them in the Add Components list.
- You receive errors when you try to commit changes to the work item feature branch.
- You receive any metadata version mismatch errors when trying to promote changes to the first pipeline stage.

An easy way to resolve the issues is to update the **sourceApiVersion** in the `sfdx-project.json` file in the work item's feature branch. The work item must also contain metadata changes as well as the changes to the DX project file.

As you promote this work item through the release pipeline and eventually release it to production, the DX project file is updated in each pipeline stage's branch and the main branch.

Manually Update in Each Pipeline Branch and Any Required Feature Branches

To prepare for the next Salesforce release, or during the planning phase for your next project, you can proactively update the DX project files in every branch so team members don't run into issues when developing and testing new changes.

Manually update the **sourceApiVersion** in every branch's DX project file directly in the source control repository. To ensure successful promotions, update the main branch, each pipeline stage branch, and any feature branches.

Conflict Detection and Resolution

Conflicts can occur when you're working with multiple development environments and moving changes between multiple pipeline stages. Conflicts can take different forms, and DevOps Center provides functionality to help you identify potential conflicts early and resolve them.

Here are some common types of conflicts.

Multiple Work Items That Modify the Same Source File

If you have multiple work items that modify the same source file (particularly in cases where multiple developers are modifying the same metadata components in their own respective development environments), the changes can conflict or potentially overwrite each other when they're merged into the first integrated pipeline stage. During the promotion process, DevOps Center warns you about the potential conflict. In some cases, the source control repository doesn't know how to reconcile the changes to the same metadata component. Sometimes the promotion is blocked based on a dependency with a shared component.

Next generation DevOps Center (Beta) provides you with the option to resolve merge conflicts using Agentforce. Agentforce uses DevOps Center MCP tools in the Salesforce DX MCP Server, to analyze the conflict and recommend solutions based on natural language prompts. Alternatively, you can resolve the merge conflicts directly in your source repository. See Resolve Merge Conflicts with DevOps Center MCP Tools

When DevOps Center (Managed Package) detects a potential conflict due to work items that share components or contain dependencies, it provides you with the option to combine the work items so you can continue with the promotion. If you don't choose to combine the affected work items, you can choose to further analyze the potential conflict and fix it manually, or promote the dependent work item first.

Unresolvable Merge Conflict in Two Branches

When a work item or stage is promoted from one pipeline stage to the next, the source files contained in the work item are merged from the first stage's source control branch to the next stage's source control branch. The source control system, GitHub, helps to determine if the changes are mergeable before you attempt to promote them. If GitHub detects a conflict, the promotion is blocked.

You can see more information on the merge conflict in the error dialog in the Activity History, and you can view additional details of the conflict in the pull request in the source control system. You can resolve the conflict directly in the source control system or using DevOps Center MCP tools. Then, you can attempt the promotion in DevOps Center.

Your Component Version Is Different from Merged Component

When you have a stage that contains multiple work items that contain the same component, the stage contains a merged version of the component. If you then choose to promote only one of those work items from the stage to the next stage, it's possible that the version of the component that you promote is different from the merged component that was tested. DevOps Center warns you when this situation occurs and provides you the option to combine the changes using the latest version. We recommend that you promote all work items that contain the common component together, so that the next stage contains the same merged version of the component as what was tested in the previous stage.

For next generation DevOps Center, if you perform a custom promotion that results in a version mismatch, identify and resolve the issue directly in your source control repository.

Prevent Potential Conflicts By Combining Work Items

Prevent potential conflicts during the promotion process by combining work items that share metadata components. In some cases, the source control repository doesn't know how to reconcile the changes to the same metadata component. Sometimes the promotion is blocked based on a dependency with a shared component. But have no fear, DevOps Center detects the shared components and provides you with a way to promote your changes through the release pipeline with confidence.

Merge Conflicts

Merge conflicts can occur when multiple work items contain one or more of the same components, especially when multiple developers are making changes that impact the same components in different development environments. DevOps Center warns you about potential merge conflicts so you can investigate to ensure that you don't inadvertently overwrite desired changes.

Resolve Merge Conflicts in Next Generation DevOps Center (Beta)

Next generation DevOps Center uses Agentforce to streamline conflict resolution for both low-code admins and developers. By integrating Salesforce DX MCP Server tools with Agentforce, DevOps Center provides AI-assisted guidance whether you work in the UI or the IDE.

Review and Resolve Conflicts in GitHub

In cases where DevOps Center can't combine work items or you choose not to combine, DevOps Center blocks the promotion. In this case, you manually edit the files directly in GitHub to resolve the conflict in the affected branch.

Merge Conflict Resolution with DevOps Center MCP Tools (Managed Package)

The DevOps Center Model Context Protocol (MCP) tools in Salesforce DX MCP Server are a set of capabilities that simplify and accelerate the resolution of merge conflicts. The MCP server uses large language models (LLMs) to analyze conflicts, translate them into natural language explanations, and suggest solutions, so you can use natural language prompts instead of commands. Access these tools through Agentforce Vibes in your Integrated Development Environment (IDE).

Prevent Potential Conflicts By Combining Work Items

Prevent potential conflicts during the promotion process by combining work items that share metadata components. In some cases, the source control repository doesn't know how to reconcile the changes to

the same metadata component. Sometimes the promotion is blocked based on a dependency with a shared component. But have no fear, DevOps Center detects the shared components and provides you with a way to promote your changes through the release pipeline with confidence.

Note The combining work items feature is currently only supported in the DevOps Center (Managed Package). This information pertains to the managed package version. To learn about the next generation DevOps Center (Beta), available from February 2026, see Next Generation DevOps Center (Beta).

Important Combining work items can't be undone.

In DevOps Center, work items can move only forward through the pipeline. After a work item has been promoted to the first pipeline stage, you can't revert it or change the status to Never. Because you can't abandon a work item in a pipeline stage, all work items must eventually move through the pipeline to ensure all pipeline stage branches and their corresponding environments are in sync. Although it seems counterintuitive to move changes you don't want through the release pipeline, combining work items merges all the changes together to achieve the desired results.

Externally Merged Changes

If you already externally merged one of the work items with shared components to the target branch, we can't combine. Complete the promotion, resolve the conflict (if necessary), and then deploy the other work items.

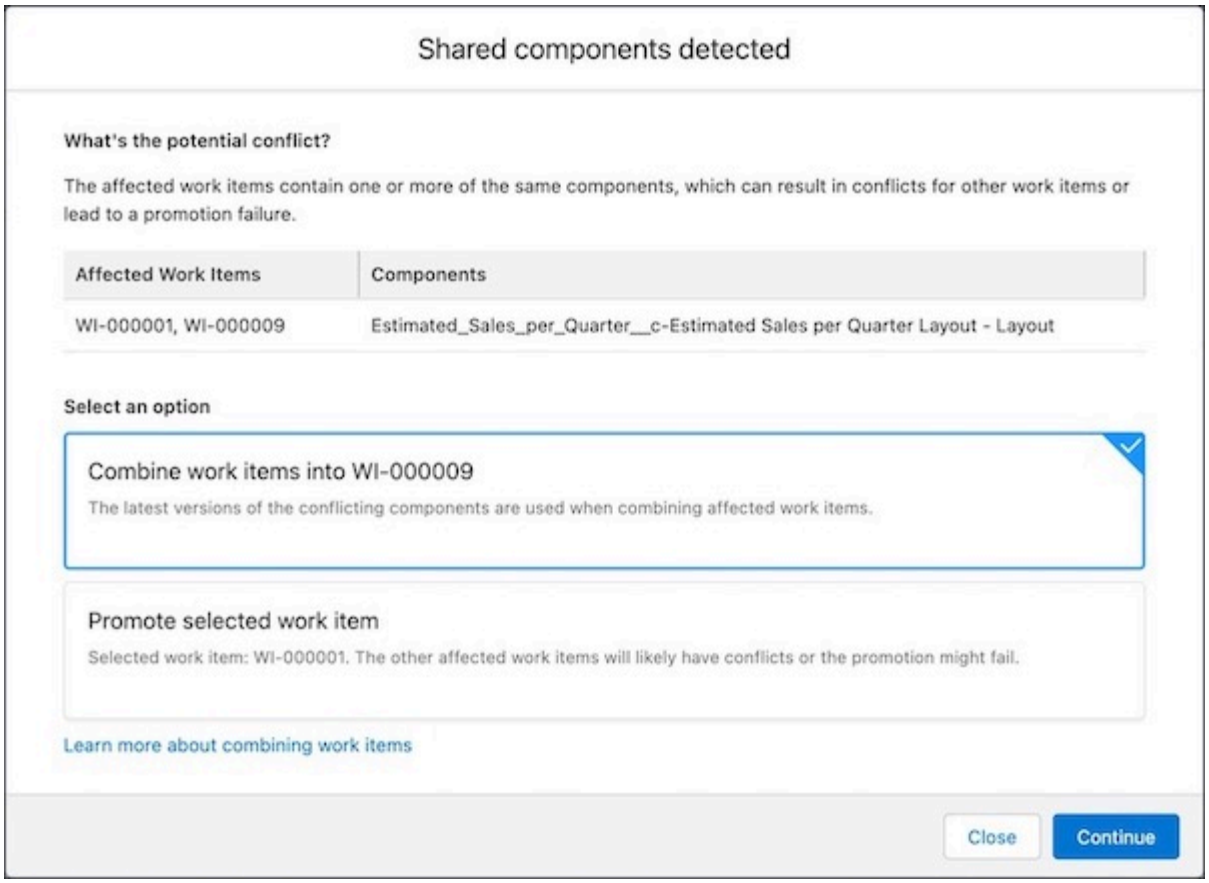
For information about handling external merges in next generation DevOps Center (Beta), see External Merges.

Common Scenarios

Let's say you discover a bug during testing or the requirements change. You create another work item that modifies the component just the way you want it, then promote this change through the pipeline. However, the source control repository doesn't know which change is the one you really want, which causes a conflict. To address the conflict and enable you to complete the promotion, you can combine work items that share components. During the combining process, the changes are merged. The latest version of the component with the required modifications "wins" and moves forward.

In another scenario, you have two work items: one that creates a new object, and another that changes the object, such as adding or deleting a field. If you attempt to promote the work item with the change before you promote the work item with the addition, the promotion is blocked due to the dependency between the work items. The next pipeline stage's branch doesn't know about the new component so it causes an error because it can't merge a change to something that doesn't already exist in its branch.

When DevOps Center detects a potential conflict due to work items that share components or contain dependencies, it provides you with the option to combine the work items so you can continue with the promotion.



Combining work items merges their feature branches into one feature branch, which avoids downstream conflicts. If you don't combine the work items, you'll likely have to manually resolve the conflicts or promote the dependent work items first to avoid promotion failures.

When work items are combined, all the metadata components from all work items are combined, not just the changes that affect shared components. For best results, follow DevOps best practices by keeping your work item scope as small as possible by including only related changes.

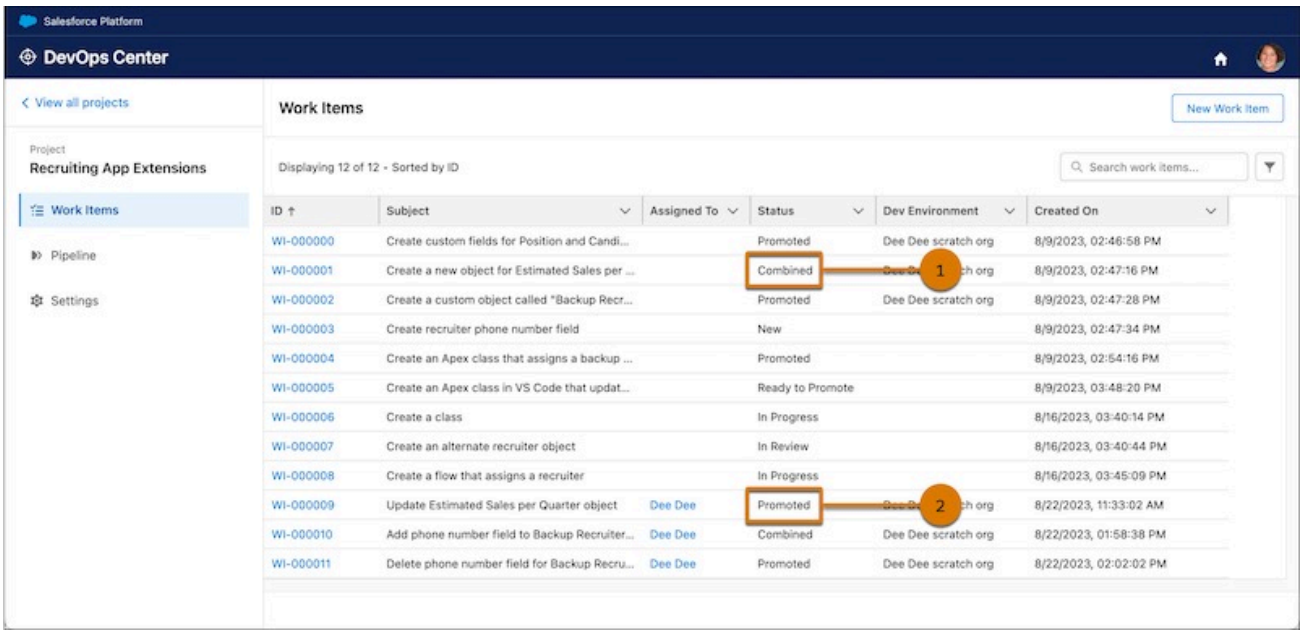
Scenario	What Happens When DevOps Center Combines Work Items
You're trying to promote multiple work items that share components. You selected all the work items that share components in the stage's branch.	DevOps Center combines the changes into the latest work item's feature branch. The latest version of the component is used when the changes are merged.
You're trying to promote multiple work items that share components, but you didn't select all the work items that share components in the stage's branch.	DevOps Center combines all the changes for the work items that contain shared components, even if you didn't initially select those work items for promotion. DevOps Center combines the changes into the latest work item's feature branch. The latest version of the component is used when the changes are merged.

Scenario	What Happens When DevOps Center Combines Work Items
You're trying to promote multiple work items that share components, and one of the work items has a dependency on another work item, so the promotion is blocked. This dependency causes a rebase error in the source control system.	DevOps Center combines the changes of the dependent work items into the latest work item's feature branch. The latest version of the component is used when the changes are merged.

Work items are combined in the order created. We start with the oldest work item, apply the changes from the second work item, and so on, until we merge the changes from the newest work item. For shared components, the resulting metadata is an accumulation of all the changes (commits), which are now reflected in the source control repository in a single work item's feature branch (the latest work item). DevOps Center removes the now obsolete work item feature branches for the older work items from the source control repository.

Work Items List

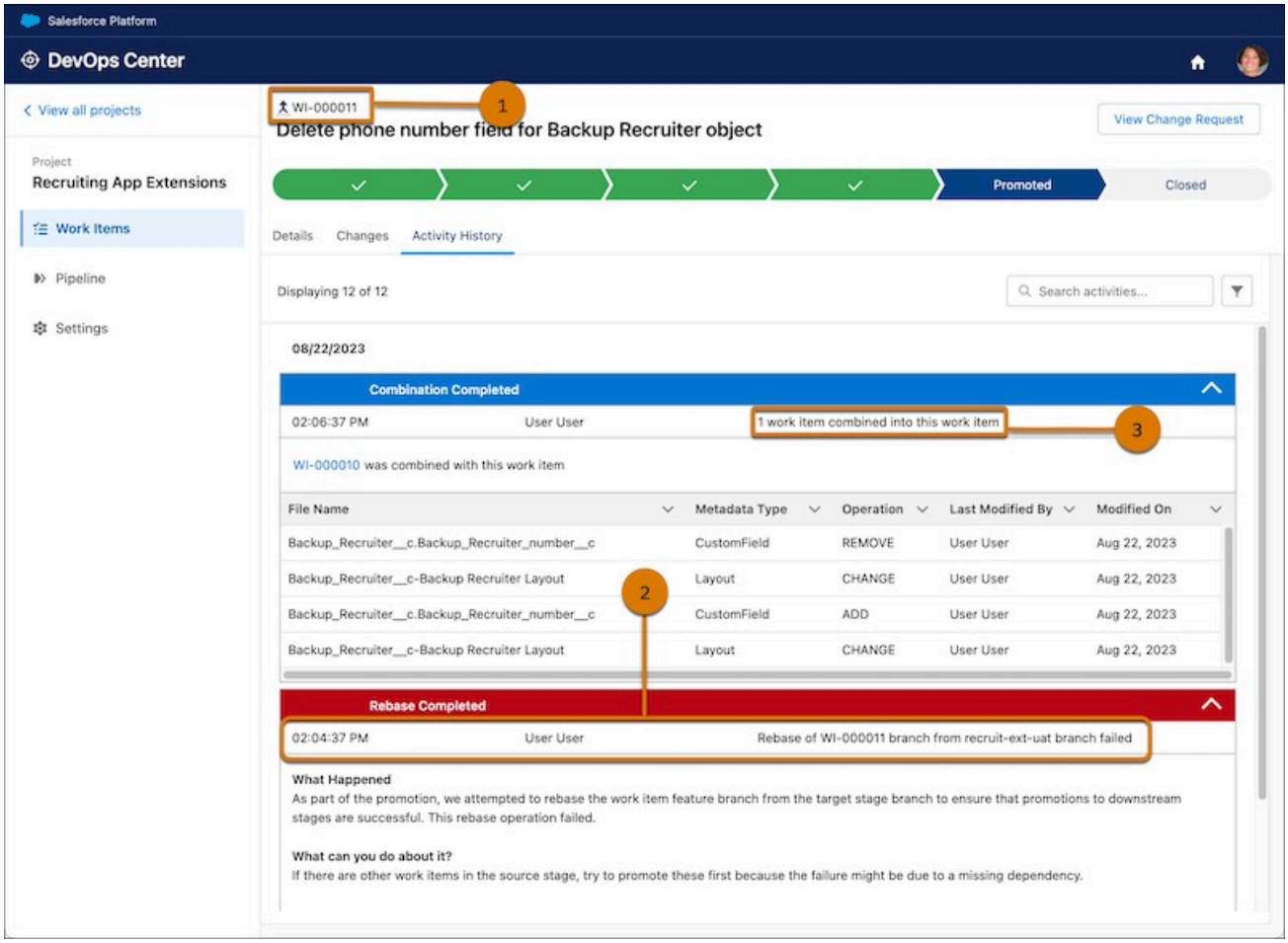
Work items that are combined with other work items have a status of Combined (1) in the Work Items list. The latest work item moves forward with the status of Promoted (2).



Work Item Activity History

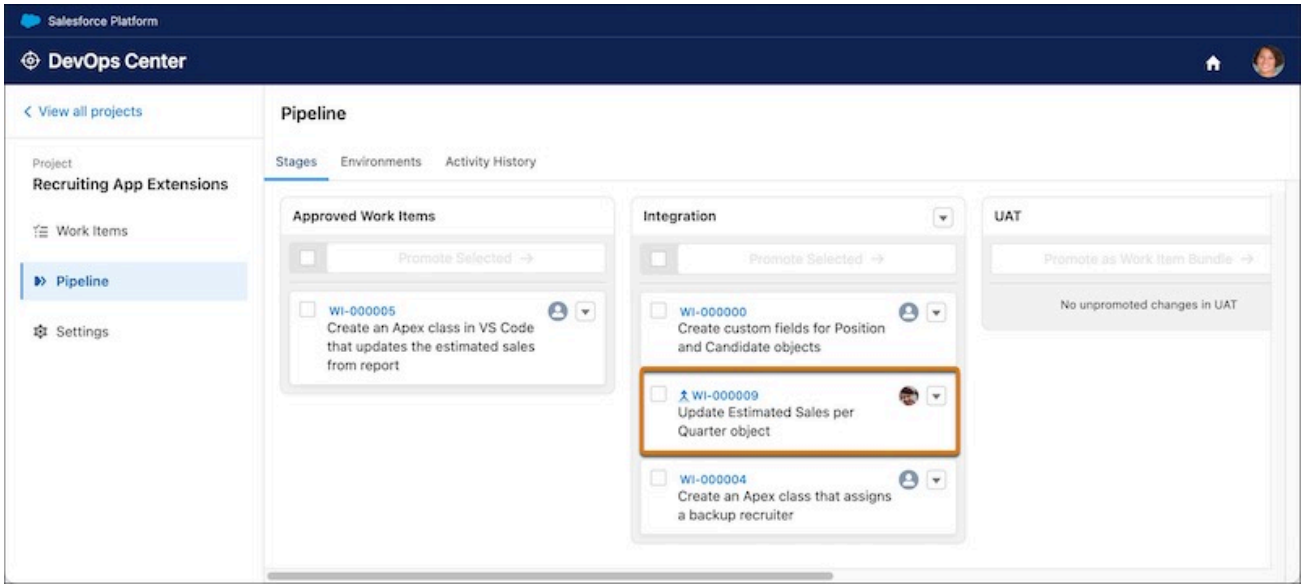
DevOps Center captures the details for combined work items in their activity history. In this example, WI-000010 was combined into WI-000011. The activity history for WI-000011 includes the Merge icon next to the work item number to identify that this work item was combined with one or more work items (1). You can see that:

- A dependency caused a rebase error that blocked its promotion (2).
- The work item was combined with WI-000010 (3).
- The combination and promotion were successful (3).



Pipeline Stages

After a successful combination and promotion, the work item moves forward to the first pipeline stage using the latest work item number. You can identify a combined work item by the Merge icon.



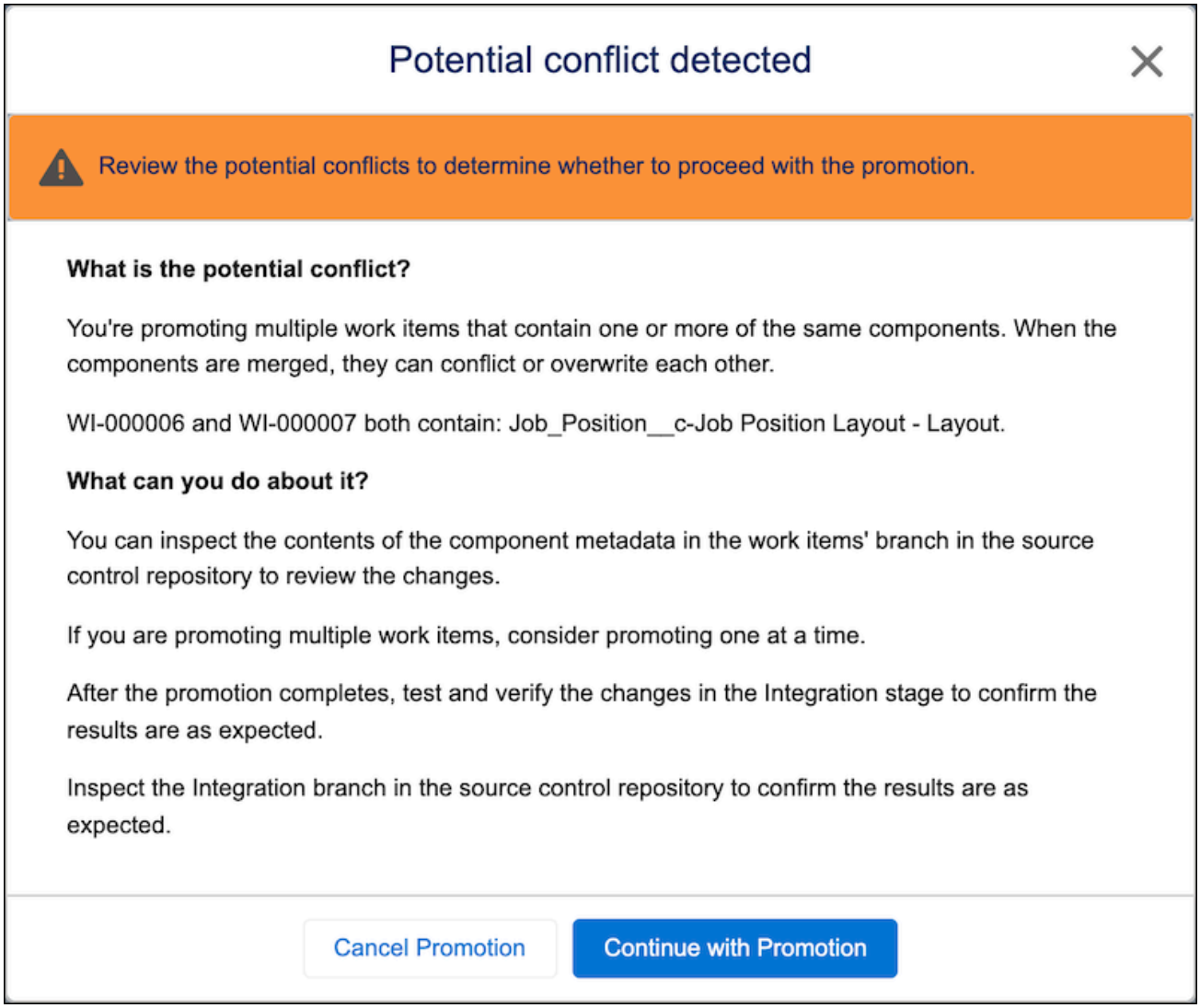
See Also

Promote Individual Work Items

Merge Conflicts

Merge conflicts can occur when multiple work items contain one or more of the same components, especially when multiple developers are making changes that impact the same components in different development environments. DevOps Center warns you about potential merge conflicts so you can investigate to ensure that you don't inadvertently overwrite desired changes.

In this example scenario, two developers are adding new fields to the same standard object, which creates potential merge conflicts in the page layouts.



Before you proceed, read the information about the potential conflict to determine what to do next.

Continue with Promotion

Based on the conflict description, you feel comfortable that the changes don't overwrite each other.

Cancel Promotion

Based on the conflict description, you're uncertain that the changes don't overwrite each other, or you'd like to look at the pull request for the work item, which indicates whether the source control repository has detected merge conflicts. After you complete your investigation, you either resolve the merge conflicts in the source control repo, sync the changes in the development environments, or continue with the promotion, which could fail if there's a merge conflict.

If the Promotion Fails

If the promotion fails, here are some options:

- Promote each work item individually.
  - If all individual work items are promoted to the next stage successfully, your work is done (for now).
  - If some work items are promoted successfully and some aren't, review the change request to resolve the merge conflicts manually in the branch in the source control system.
- Sync the development environments, or resolve the conflicts directly in the development environments, then try the promotion process again.
- Custom promotion in next generation DevOps Center (Beta): Promote individual work items from a stage. See Custom Promotion.

Resolve Merge Conflicts in Next Generation DevOps Center (Beta)

Next generation DevOps Center uses Agentforce to streamline conflict resolution for both low-code admins and developers. By integrating Salesforce DX MCP Server tools with Agentforce, DevOps Center provides AI-assisted guidance whether you work in the UI or the IDE.

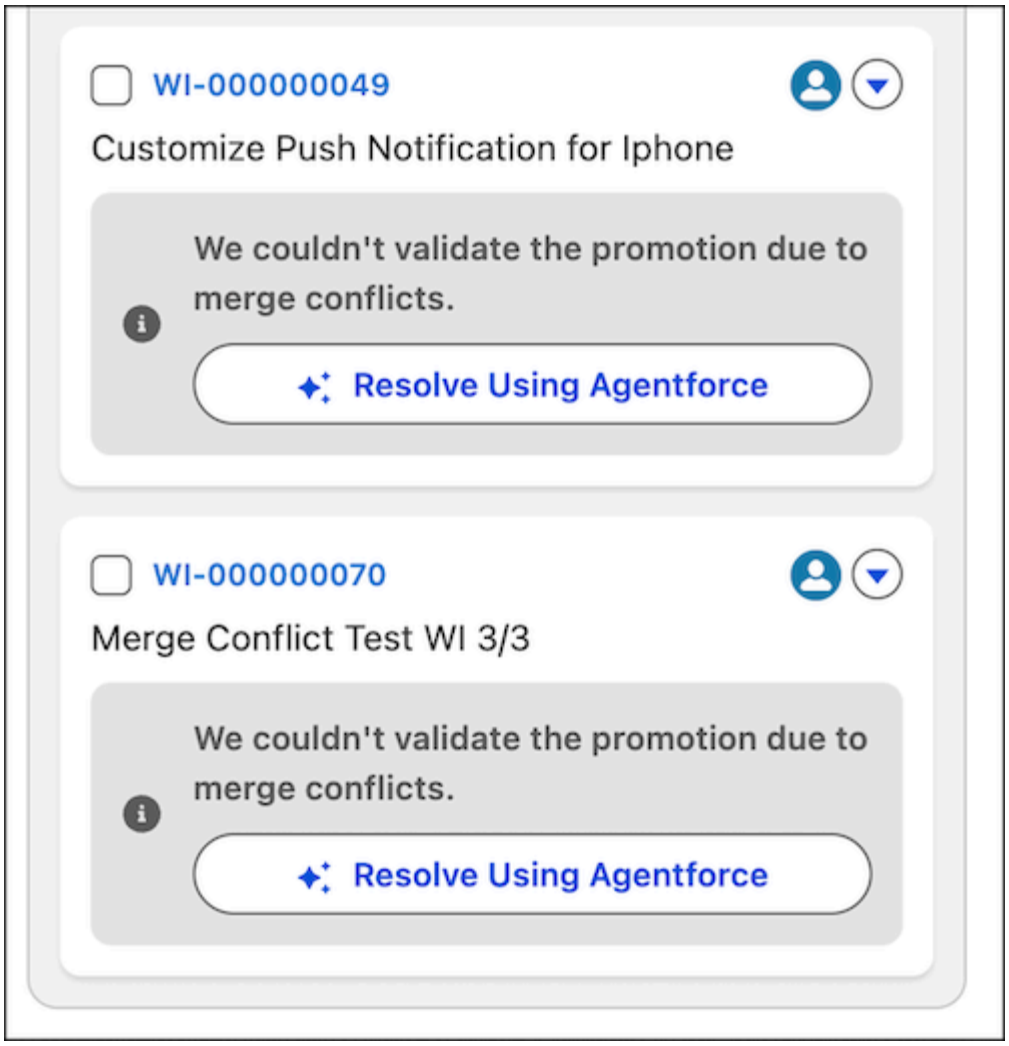
REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional**, **Enterprise**, **Performance**, and **Unlimited** Editions

Resolve Conflicts from the UI (Low-Code)

If you're a low-code developer or admin, use DevOps Center MCP tools to manage conflicts without writing commands. When DevOps Center detects a conflict during a promotion, the pipeline highlights the specific blocked work item with an error notification.

Click the **Resolve with Agentforce** button to automatically launch Agentforce Vibes IDE. DevOps Center preloads the specific error message and context into the prompt. You can interact with Agentforce to review the conflicting files and accept the recommended fixes.



Resolve Conflicts Using DevOps MCP Tools

Use DevOps MCP tools to detect, analyze, and resolve conflicts programmatically. When working within Agentforce Vibes IDE, Agentforce automatically detects the merge conflict. It adds the error details into the prompt. Ask the Agentforce Vibes to explain the differences between the files or suggest a solution.

See Also

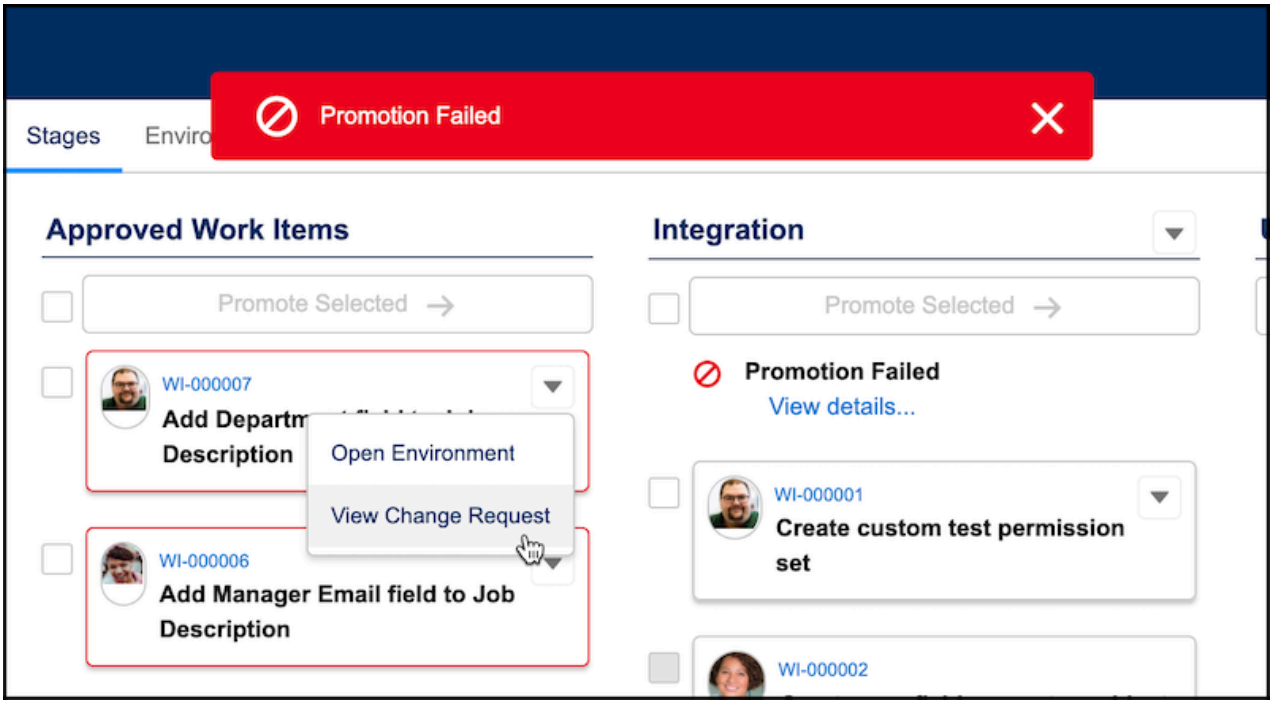
DevOps Center MCP Tools

Review and Resolve Conflicts in GitHub

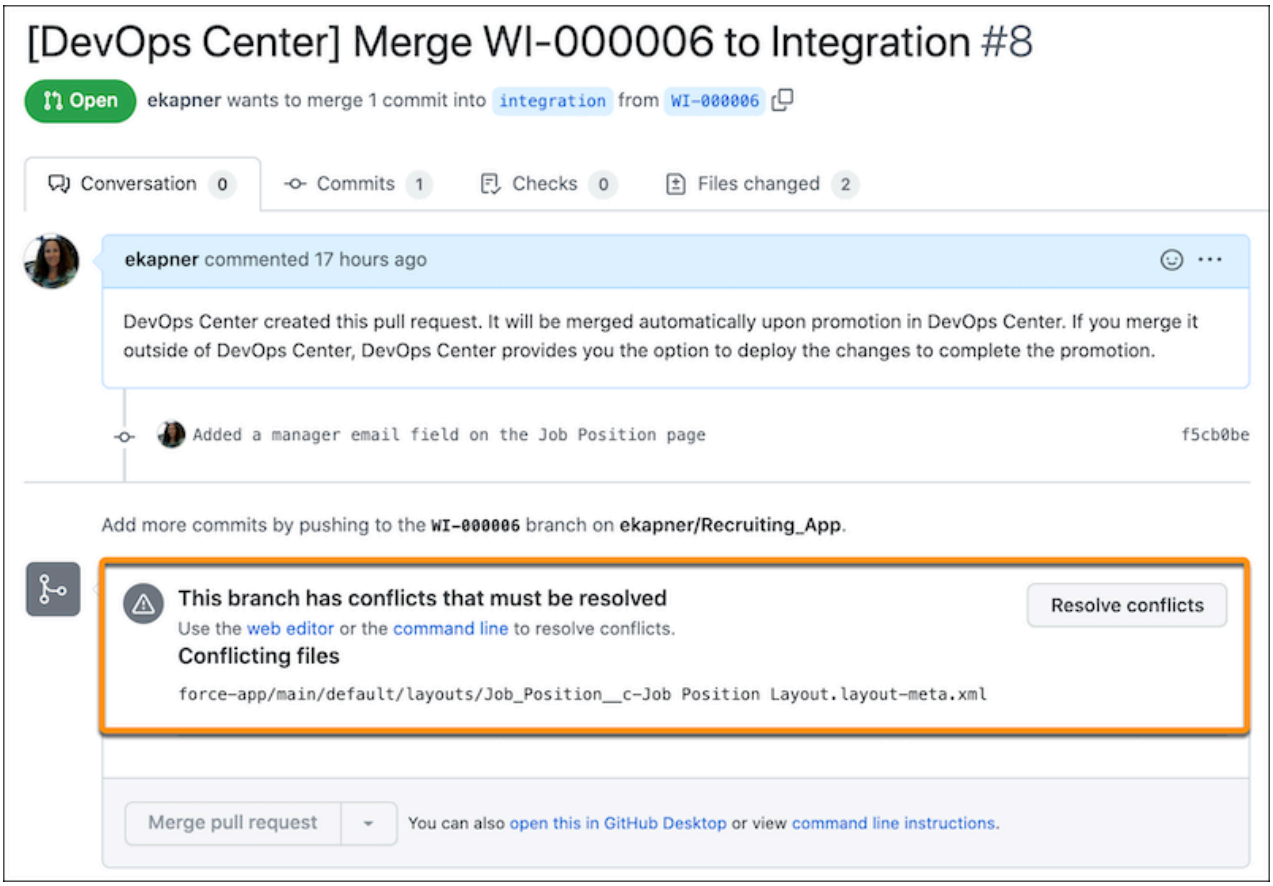
In cases where DevOps Center can't combine work items or you choose not to combine, DevOps Center blocks the promotion. In this case, you manually edit the files directly in GitHub to resolve the conflict in the affected branch.

1. From the work item dropdown (on the stage or work item you're promoting from), select **View Change Request** to go into GitHub to view the pull request and see the merge conflicts.

Repeat these steps for all work items where the promotion failed.

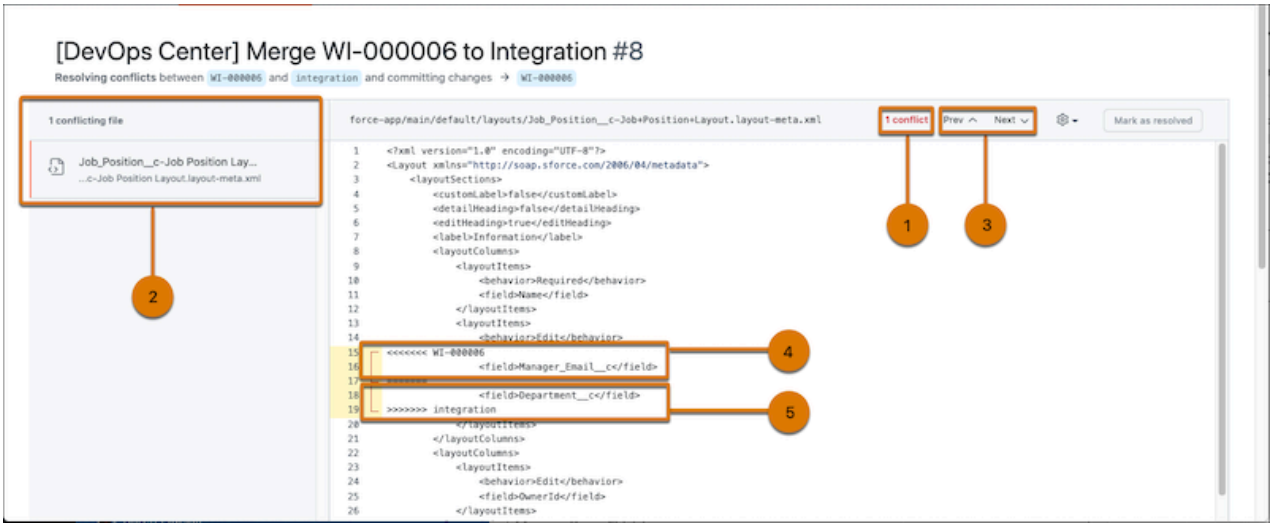


2. In GitHub, click **Resolve Conflicts**.



In our example scenario, GitHub has identified one conflict (1) in one file, the Job Position Layout (2). If GitHub detects multiple conflicts in a file, you can use the Prev and Next links (3) to address each

conflict one-by-one.



The format of the conflict as shown in GitHub is:

```
<<<<<<<<<< <branch 1 name>
<code in branch 1>
==========
<code in branch 2>
>>>>>>>>>> <branch 2 name>
```

In this example, the work item you attempted to promote (4) in branch WI-000006 is introducing a new field, while the CRM Integration branch doesn't contain that new field. Also, the CRM Integration branch contains a field that the work item branch doesn't contain (5), which was likely introduced by a different work item. In this case, you want to keep both these new fields, so you modify the file to eliminate the conflict.

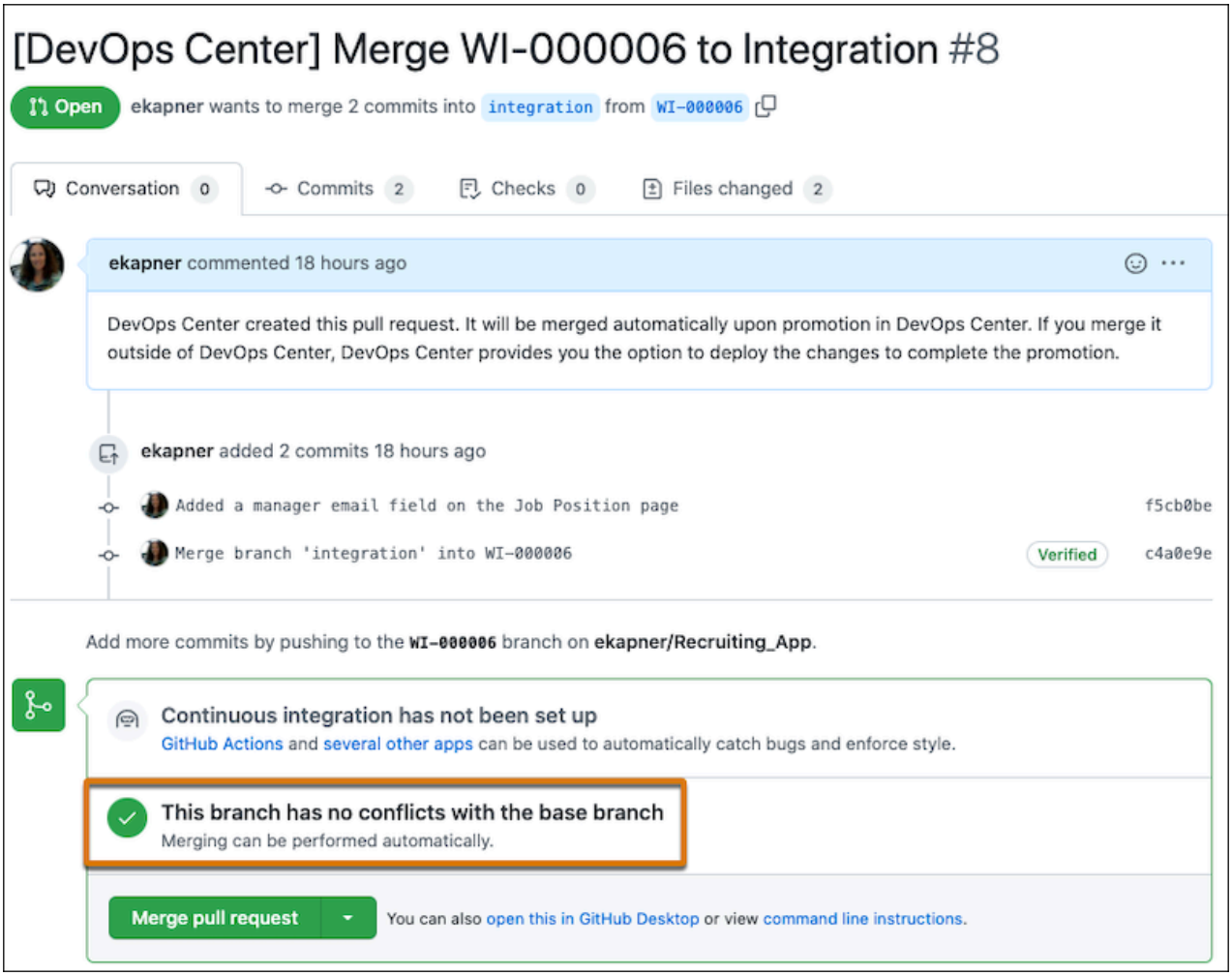
The Metadata API Developer Guide provides the schema for each metadata type to assist you with editing the files directly. In this case, we can look at "Layout" in the guide to view the schema for <layoutItems>.

The resulting changes now look like this:

```
<layoutItems>
<behavior>Edit</behavior>
       <field>Manager_Email__c</field>
</layoutItems>
<layoutItems>
<behavior>Edit</behavior>
       <field>Department__c</field>
</layoutItems>
```

3. At the top of the file, click **Mark as Resolved**.
4. Continue until you resolve all conflicts in the listed files.
5. Click **Commit Merge**, which saves your changes in the work item branch, W-000006.

   Notice that the conflicts are now resolved.



6. Return to DevOps Center to promote the work item, which merges the branch with the branch in the next stage, in this case, Integration.
7. In DevOps Center, promote the work items again.

   Alternatively, you can merge the change request in GitHub, then complete the promotion in DevOps Center or by using Salesforce CLI.

After you resolve conflicts, your development environments are out of sync with downstream pipeline environments, so we recommend that you synchronize your dev environment.

Merge Conflict Resolution with DevOps Center MCP Tools (Managed Package)

The DevOps Center Model Context Protocol (MCP) tools in Salesforce DX MCP Server are a set of

capabilities that simplify and accelerate the resolution of merge conflicts. The MCP server uses large language models (LLMs) to analyze conflicts, translate them into natural language explanations, and suggest solutions, so you can use natural language prompts instead of commands. Access these tools through Agentforce Vibes in your Integrated Development Environment (IDE).

Note This article provides information about the DevOps Center managed package. Starting in February 2026, Salesforce offers next generation DevOps Center (Beta). For information about the new version, see Next Generation DevOps Center.

Salesforce DX MCP Server is designed with multiple layers of security to protect your data and credentials. See Salesforce DX MCP Server and Tools (Beta).

For information about resolving conflicts in next generation DevOps Center (Beta) using MCP tools, see Resolve Merge Conflicts with DevOps Center MCP Tools.

Resolve Merge Conflicts by Using DevOps Center MCP Tools
Resolve merge conflicts by using natural language prompts. The provided prompts are examples that you can customize to suit your needs. The sample response varies based on the LLM that you use.

See Also
- Install and Configure the Salesforce DX MCP Server (Beta)
- MCP Solutions for Developers

Resolve Merge Conflicts by Using DevOps Center MCP Tools

Resolve merge conflicts by using natural language prompts. The provided prompts are examples that you can customize to suit your needs. The sample response varies based on the LLM that you use.

Access the DevOps Center MCP Tools

Note This article provides information about the DevOps Center managed package. Starting in February 2026, Salesforce offers next generation DevOps Center (Beta). For information about the new version, see Next Generation DevOps Center.

To resolve merge conflicts that occur during DevOps Center promotions, install and configure Salesforce DX MCP Server to use DevOps Center MCP tools.

Before you use the DevOps Center MCP tools, review the Salesforce DX MCP Server prerequisites and make sure that you have:

- An org with DevOps Center installed and configured
- A DevOps Center project connected to a version control system
- An IDE, such as Agentforce Vibes IDE or Visual Studio Code (VS Code), that your Salesforce admin has accepted the Terms and Conditions for
- A configured IDE project. To learn how, see Create Projects.

- An IDE installed in the same Salesforce org where your DevOps Center is installed. Connect and authenticate your IDE to this org.
- A cloned project repository open in your IDE's workspace
- An MCP client, such as the Agentforce Vibes Extension. For setup instructions, see Set Up Agentforce Vibes Extension.

Salesforce DX MCP Server is preconfigured in Agentforce Vibes Extension. Open your IDE and turn on Salesforce DX MCP Server in Agentforce Vibes Extension. See Enable Salesforce DX MCP Server.

You interact with the server by entering natural language prompts in Agentforce. Agentforce Vibes is available from the Activity Bar and includes a chat window where you enter your prompts. For examples of prompts that you can use, see Resolve Merge Conflicts.

To use VS Code as the MCP client, see Quick Start Using the VS Code With Copilot MCP Client (Beta).

1. Detect and Analyze a Specific Conflict

Prompt

"Help me understand the conflict in the `Admin.profile-meta.xml` file."

How It Works

The `detect_devops_center_merge_conflict` tool analyzes a specific merge conflict to provide intelligent resolution options. In three steps, the tool:

1. Checks out the corresponding work item branch in your local environment
2. Attempts to merge the work item branch into the target branch
3. Explains the conflicts and suggests a resolution if any are detected

Sample Response

"I analyzed the conflict in `force-app/main/default/profiles/Admin.profile-meta.xml`. Here are your options:

- **Take Incoming Changes**: Keep the changes from the target branch and discard the work item branch's changes.
- **Take Current Changes**: Keep the changes from the work item branch and override the target branch's changes wherever conflicts occur.

How would you like to proceed?"

2. Resolve the Merge Conflict

Prompt

"Resolve the conflict in `Admin.profile-meta.xml` by keeping the incoming changes."

How It Works

The `resolve_devops_center_merge_conflict` tool applies your chosen resolution strategy to the specified file. In three steps, the tool:

1. Reads the content of the conflicting file
2. Replaces the entire conflict block with the content from your chosen strategy–in this case, the incoming changes
3. Writes the modified content back to the file and stages the resolved file using `git add`

Important You must commit the AI-suggested changes as a final step.

Sample Response

"Success! The conflict in `force-app/main/default/profiles/Admin.profile-meta.xml` has been resolved by accepting the incoming changes from the dev branch. The file has been staged. Please review and commit when you're ready."

See Also

Agentforce Vibes IDE: Supported Editions

DevOps Center MCP Tools

Accelerate your application lifecycle with DevOps Center MCP tools, built for DevOps Center. The Salesforce DX MCP server provides a set of DevOps Center tools that enable Large Language Models (LLMs) to securely and autonomously read, manage, and operate DevOps Center resources.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

You can interact with the DevOps Center resources conversationally from Agentforce in your IDE. The MCP server acts as a context-aware backend agent that translates natural language prompts from your IDE into a sequence of API calls or CLI commands needed to perform complex DevOps tasks.

Interested in learning more about MCP servers? See MCP Solutions for Developers.

Benefits of DevOps Center MCP Tools

- Streamline manual DevOps processes. Manage work items, create pull requests, and deploy metadata using natural language prompts.
- Reduce deployment cycle time.
- Proactively identify risks and provide AI-assisted resolution for issues such as merge conflicts.
- Implement standardized workflows and DevOps best practices across all projects and users.

- Enhance security and data retention by using encrypted authentication files, preventing data leaks, and allowing only approved access to Salesforce orgs.

Access the DevOps Center MCP Tools
Set up Salesforce DX MCP Server for use with Agentforce Vibes IDE, integrated with Agentforce Vibes Extension as the MCP client. Use prompts in Agentforce, a part of Agentforce Vibes Extension, to do common DevOps tasks, like managing work items, and resolving merge conflicts.

Manage the Work Item Lifecycle with DevOps Center MCP Tools
Use DevOps Center MCP tools to manage your work item's lifecycle end-to-end. You can handle work item creation, user assignments, code changes, pull requests, and promotions directly through natural language prompts.

Resolve Merge Conflicts with DevOps Center MCP Tools
Detect, analyze, and resolve merge conflicts directly within your workflow. You can start this process using the Resolve with Agentforce option in the DevOps Center UI, which opens the Agentforce Vibes IDE with the Agentforce Vibes Extension and adds the conflict details to your prompt.

Tool Descriptions
Find detailed descriptions and sample prompts for each tool.

Access the DevOps Center MCP Tools

Set up Salesforce DX MCP Server for use with Agentforce Vibes IDE, integrated with Agentforce Vibes Extension as the MCP client. Use prompts in Agentforce, a part of Agentforce Vibes Extension, to do common DevOps tasks, like managing work items, and resolving merge conflicts.

Note The next generation DevOps Center MCP tools aren't generally available, so set the --allow-non-ga-tools flag to true in the a4d_mcp_settings.json file to enable them.

Although this example uses Agentforce Vibes IDE and Agentforce, you can configure other clients, such as Cursor or Claude Desktop. To use Visual Studio Code (VS Code) as the MCP client, see Quick Start Using the VS Code With Copilot MCP Client (Beta).

Before you use the DevOps Center MCP tools, review the Salesforce DX MCP server prerequisites and make sure that you have:

- An org with DevOps Center installed and configured
- A DevOps Center project connected to a version control system and a pipeline
- An IDE, such as Agentforce Vibes IDE or Visual Studio Code (VS Code), that your Salesforce admin has accepted the Terms and Conditions for
- A configured IDE project. To learn how, see Create Projects.
- An IDE installed in the same Salesforce org where your DevOps Center is installed. Connect and authenticate your IDE to this org.
- A cloned project repository open in your IDE's workspace
- An MCP client, such as the Agentforce Vibes Extension. For setup instructions, see Set Up Agentforce Vibes Extension.

Salesforce DX MCP Server is preconfigured in Agentforce Vibes Extension. Open your IDE and turn on Salesforce DX MCP Server in Agentforce Vibes Extension. See Enable Salesforce DX MCP Server.

You interact with the server by entering natural language prompts in Agentforce. Agentforce Vibes is available from the Activity Bar and includes a chat window where you enter your prompts. For examples of prompts that you can use, see Resolve Merge Conflicts with DevOps Center MCP Tools. When you enter a prompt, Salesforce DX MCP Server determines the appropriate DevOps Center MCP tool to run. It then shows it to you along with other information. You can click **Continue** to run the tool and see the results of your request.

See Also

- Install and Configure the Salesforce DX MCP Server (Beta)
- Agentforce Vibes IDE: Supported Editions
- Salesforce Extensions for Visual Studio Code

Manage the Work Item Lifecycle with DevOps Center MCP Tools

Use DevOps Center MCP tools to manage your work item's lifecycle end-to-end. You can handle work item creation, user assignments, code changes, pull requests, and promotions directly through natural language prompts.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Before you begin, set up the Salesforce DX MCP Server. See Access the DevOps Center MCP Tools.

Step 1: Start with a work item

Identify your assigned tasks and prepare your local environment for development.

Prompt

"Show all open work items assigned to me. "

How It Works

The `list_devops_center_work_items` tool returns a list of work items, including their titles, branches, environments, and statuses.

Sample Response

"Ok. Here are your open Work Items:

Work Item: WI-1234

Title: Bug Fix for Account Creation

Branch:`bugfix/account-create`

Environment: UAT

Status: In Progress"

Prompt

"Let's start working on WI-1234."

How It Works

The `checkout_devops_center_work_item` tool checks out the specified work item and prepares your local environment.

Sample Response

"Ok. Checking out Work Item WI-1234 and preparing your development environment."

Step 2: Commit changes

After making your code changes, commit them to the source control repository.

Prompt

"Commit changes: Fixed validation for Contact email."

How It Works

The `commit_devops_center_work_item` tool verifies the current Git branch, adds the work item ID to the commit metadata, and commits the changes.

Sample Response

"Acknowledged. Committing changes with the message 'Fixed validation for Contact email'. The work item ID will be added to the commit metadata."

Step 3: Create a pull request

Once your changes are committed, initiate the review process.

Prompt

"Create a PR for this feature to integration."

How It Works

The `create_devops_center_pull_request` tool first checks if your commits are pushed to the remote repository.

- If commits are not pushed, the tool blocks the action and prompts you to push them.
- If commits are pushed, the tool creates a pull request and returns the ID and URL.

Sample Response

"Understood. Creating a pull request. Once complete, I will provide the Pull Request ID and URL.

Here is the final output:

Pull Request ID: PR-102

Pull Request URL: `https://your-devops-platform.com/project-alpha/pull-requests/5678`"

Step 4: Review and approve pull requests

Open the URL provided in the previous step to review and approve the pull request directly in GitHub.

Step 5: Promote the work item

After the pull request is approved, deploy the changes to the next environment.

Prompt

"Promote WI-1234 to UAT."

How It Works

The `promote_devops_center_work_item` tool identifies the work item ID and target environment, merges the feature branch, and deploys the metadata.

Sample Response

"Promotion of WI-1234 to UAT is initiated. You can monitor the progress and view logs in DevOps Center."

Resolve Merge Conflicts with DevOps Center MCP Tools

Detect, analyze, and resolve merge conflicts directly within your workflow. You can start this process using the Resolve with Agentforce option in the DevOps Center UI, which opens the Agentforce Vibes IDE with the Agentforce Vibes Extension and adds the conflict details to your prompt.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Before you begin, set up the Salesforce DX MCP Server. See Access the DevOps Center MCP Tools.

Note If you are a DevOps Center (Managed Package) user and want to learn how to resolve merge conflicts using MCP tools, see Resolve Merge Conflicts by Using DevOps Center MCP Tools.

Step 1: Detect and summarize conflicts

Identify the files causing the conflict and understand the nature of the conflict.

Prompt

"Check if my current branch has any merge conflicts."

How It Works

The `detect_devops_center_merge_conflict` tool scans Git to determine the merge state. It compares the feature branch with the target branch, identifies conflicting metadata, and suggests resolution strategies. For example, merge suggested, accept incoming, or keep yours.

Sample Response

"Conflicts have been detected between the work item branch WI-000000036 and the target branch integration. The following files have conflicts:

- `force-app/main/default/classes/class_name.cls`

- Conflict Type: Both branches modified the same line.

- Explanation: The method has been modified in both branches with different debug messages.

- Conflict Excerpt: [Details]

- `force-app/main/default/classes/new_class_name.cls`

- Conflict Type: Both branches modified the same line.

- Explanation: The method has conflicting debug messages.

- Conflict Excerpt: [Details]

To proceed with resolving these conflicts, you can use the `resolve_devops_center_merge_conflict` tool for further assistance."

Step 2: Resolve the merge conflict

Apply a resolution strategy to the conflicting files.

Prompt

"Use AI to resolve the conflict."

How It Works

The `resolve_devops_center_merge_conflict` tool loads the specified file, applies the selected resolution strategy, and stages the file for commit.

Sample Response

"Task Completed

The merge conflicts between the work item branch WI-000000036 and the target branch integration have been resolved successfully. The following actions were taken:

1. Resolved conflicts in the files:

- `force-app/main/default/classes/class_name.cls`: Kept current changes from WI-000000036.

- `force-app/main/default/classes/new_class_name.cls`: Kept current changes from WI-000000036.

2. Committed the resolution with the message: 'Resolve merge conflicts between WI-000000036 and integration'.

The branch is now conflict-free and ready for further actions."

Step 3: Resume work item promotion

After conflicts are resolved, continue promoting your work items in DevOps Center.

Tool Descriptions

Find detailed descriptions and sample prompts for each tool.

Next Generation DevOps Center MCP Tools

Work Item

Tool	Description	Sample Prompts
`list_devops_center_work_items`	List all the work items for a specific DevOps Center project.	• “List my Work Items for this sprint along with their branches.” • “Give me a list of active Work Items with their environments and target branches.” • “Fetch all the Work Items I'm assigned to and show their promotion paths.”
`list_devops_center_projects`	List all DevOps Center projects in a specific org.	• “Can you list out all the projects in DevOps Center for me?” • “What are the DevOps Center projects currently in this org?” • “I'd like to see all the active DevOps Center projects.”
`checkout_devops_center_work_item`	Clone and open a DevOps Center work item’s branch in your local development environment.	• “Check out my WI-1234 and prepare the environment” • “Start work on WI-1234. Open the branch and show me the details.” • “Switch to the branch for WI-1234 and display the associated Work Item description.”
`commit_devops_center_work_item`	Commit local changes to a DevOps Center work item's feature branch.	• “Commit my changes with message 'Fix bug in account logic' and tie it to WI-1092.” • “Make a commit on the active feature branch and tie it to WI-9999, use message 'Initial

Tool	Description	Sample Prompts
		DevOps logic." • "Commit my changes to the work item."
`check_devops_center_commit_status`	Check the current status of a work item committed to DevOps Center.	• "What's the status of the commit for work item WI-001?" • "Can you check the commit request status for the latest work item?" • "Is the commit for work item #456 still pending?"

Pull Request

Tool	Description	Sample Prompts
`create_devops_center_pull_request`	Create a pull request from a DevOps Center work item's feature branch.	• "Create a pull request for Work Item and merge it into integration." • "Open a PR from my current feature branch to the integration branch." • "Create and register a PR to DevOps Center from my latest pushed commits."

Pipeline and Deployment

Tool	Description	Sample Prompts
`promote_devops_center_work_item`	Promote an approved work item to the next stage in the DevOps Center pipeline.	• "Deploy my current feature branch to the UAT sandbox." • "Promote the approved work item WI-1234 to the UAT environment." • "Release WI-1234 to UAT."

Conflict Resolution

Tool	Description	Sample Prompts
`detect_devops_center_merge_conflict`	Identify merge conflicts between a DevOps Center work item's feature branch and its target branch.	• "Analyze the merge conflict for my current branch against the dev branch." • "Help me understand the conflict that's preventing WI-1234 from being promoted?" • "Explain why my feature branch has a conflict?"
`resolve_devops_center_merge_conflict`	Apply a selected resolution method to a merge conflict.	• "Resolve the conflict in `profile-meta.yml` by merging both changes." • "Resolve the conflict by keeping current changes." • "Accept all incoming changes for all conflicting files."

DevOps Center (Managed Package) MCP Tools

Tool	Description	Sample Prompts
`detect_devops_center_merge_conflict`	Identify merge conflicts between a DevOps Center work item's feature branch and its target branch.	• "Show me a summary of the current merge conflicts for work item WI-007." • "Summarize the merge conflicts between my current branch and main." • "Why does my feature branch have a conflict?"
`resolve_devops_center_merge_conflict`	Apply a selected resolution method to a merge conflict.	• "Resolve the conflict in `profile-meta.yml` by merging both changes." • "Resolve the conflict by keeping my current changes."

Tool	Description	Sample Prompts
		• “Accept all incoming changes for all conflicting files.”

Troubleshoot DevOps Center Errors

Here are some tips if you encounter issues while using DevOps Center. If needed, get help from your team manager or Salesforce admin to resolve these issues.

Provide General Feedback

If you want to provide general feedback, use the DevOps Center Trailblazer group. You can also use the group to request product enhancements, start discussions with other DevOps Center users or the product team, and share best practices.

Issue: User interface doesn’t respond

You notice the DevOps Center user interface doesn’t change after you perform an action. For example, the Create Review button remains disabled after you perform a commit.

Cause: If you have Lightning Web Security (LWS) disabled, it can interfere with the user interface’s ability to respond to a user action.

Action: Refresh the DevOps Center browser page. Alternately, you can enable LWS to avoid this issue.

Error: Pushing Failed

When promoting changes, the promotion fails and you see the “Pushing Failed” error.

Cause: It’s likely that an access token timed out during the promotion process. In this case, the changes were deployed to the target org but weren’t merged to the pipeline stage’s branch.

Action: See if the changes were deployed to the target org. If you see that changes weren’t deployed on the Deployment Status Setup page, the promotion is stalled. See Stalled Promotions to continue.

- Launch the target org. From Setup, find and select **Deployment Status**.
- If the changes were deployed, go to the source control repository and merge the change (pull) request. The changes are now reflected in the pipeline stage’s branch.
- Back in DevOps Center, go to the Pipelines Stages page and complete the promotion. The changes are redeployed to the target org.

Issue: DevOps Center doesn't load

When you launch DevOps Center, you see a spinner but the Project page never loads.

Cause: If you installed the Salesforce Page Optimizer Chrome extension, it's not compatible with DevOps Center. We plan to fix this issue in a future release.

Action: Remove or disable the extension.

Error: Promotion Stopped Due to New Events dialog

The promotion was stopped and you see the Promotion Stopped Due to New Events dialog. Sometimes you see this dialog along with the "Too many SOQL queries" red banner.

Cause: DevOps Center can't proceed with the promotion because it's still processing events from the source control system. These events are required for this promotion to succeed.

Action: See Error: Promotion Stopped Due to New Events Dialog for step-by-step instructions.

Error: sf_devops:Too many callouts: 101

Cause: DevOps Center can't proceed with the promotion because it's still processing events from the source control system.

Action: See Error: sf_devops:Too many callouts: 101 for step-by-step instructions.

Error: sf_devops:Too many SOQL queries: 101

You see this error when trying to promote individual work items in a non-bundled stage:

```
sf_devops.AsyncOperationResultCleanup: System.LimitException: sf_devops:Too ma
ny SOQL queries: 101
```

Cause: You have many work items in a non-bundled stage, or you're trying to promote more than 30 individual work items in a pipeline stage or 20 work items from the Approved Work Items lists.

Action: Promote fewer work items at a time. If you haven't yet promoted the work items to the first pipeline stage, see Work Item Management. It provides tips on how to reduce the number of work items you're managing for changes you don't plan to promote in the near future.

Error: stalled commit

Cause: A stalled commit can occur when updates from external services back to DevOps Center are interrupted during the commit operation.

Action: Determine whether the commit succeeded or failed so you know how to fix it. See Stalled Commits for details. Alternatively, change the Status of the work item to Never, and then start again with a new work item.

Error: Stalled Promotion

Cause: A stalled promotion can occur when updates from external services back to DevOps Center are interrupted during deploy or merge operations.

Action: Determine where the error occurred so you can fix it. See Stalled Promotions for details.

Error: There was an error. Please reload the page...

You see this error when trying to add an environment to the project.

Cause: If Session Security "Level Required at Login" is set to `High Assurance`, it prevents the use of asynchronous processing.

Action: Ask a Salesforce admin to turn off high assurance for **Manage Auth. Providers**.

- From Setup, enter *`Identity Verification`* in the Quick Find box, then select **Identity Verification**.
- Under Session Security Level Policies, for Manage Auth. Providers, select **None**.
- Click **Save**.

Error: There was an internal data model error. Please reload the page...

You see a work item that indicates that it's externally merged but you didn't externally merge it. When you attempt to complete the promotion, you see a red banner with the "internal data model" error message.

Cause: Due to event processing delays, a duplicate Work Item Promote (WIP) record was erroneously created.

Action: Find and delete the duplicate Work Item Promote record. See Error: There was an internal data model error. Please reload the page... for step-by-step instructions.

Error: Ending position out of bounds: -1

You see this error when committing changes or performing some other operation.

Cause: If Session Security "Level Required at Login" is set to `High Assurance`, it prevents the use of asynchronous processing.

Action: Ask a Salesforce admin to turn off high assurance for **Manage Auth. Providers**.

- From Setup, enter *`Identity Verification`* in the Quick Find box, then select **Identity Verification**.

- Under Session Security Level Policies, for Manage Auth. Providers, select **None**.
- Click **Save**.

Error: Lightning Web Security: Resource loader error loading <...//devopsCenterResources/devopsCenterCSSOverride.css>

Cause: You see this error when launching a project or trying to access your work item list. Enabling Test Run for Secure Static Resources for Lightning Components is incompatible with DevOps Center.

Action: Disable test run:

- From Setup, enter `Release Updates` in the Quick Find box, then select **Release Updates**.
- Click **Get Started**.
- Click **Disable Test Run**.

Error: Bad_OAuth_Token

Cause: When you try to perform an action from within DevOps Center, such as committing or promoting changes, an error indicates that you have a bad OAuth token. Due to the underlying technologies that support creating sessions that allow DevOps Center to interact with multiple environments in a project's pipeline, IP addresses aren't static. Therefore, Lock sessions to the IP address from which they originated must be disabled.

Action: Ask a Salesforce admin to disable the setting in the org in which DevOps Center is installed. See Salesforce Help: Configure Session Settings for details.

Error: Did You Log In to the Correct Environment?

Cause: DevOps Center acts like you didn't log into the correct sandbox. In Winter '23, Salesforce launched Enhanced Domains, which changed sandbox My Domain URL formats to conform to new security standards. When you connect a sandbox in DevOps Center, DevOps Center uses a named credential to facilitate authentication. During the release transition, the named credential will likely require updating after the sandbox is upgraded to the Winter '23 release.

This issue surfaces when you try to perform an action using the sandbox. Actions include opening a connected sandbox, pulling changes from a connected environment, or promoting changes to the environment.

Action: A user with Salesforce admin privileges or the "Customize Application" user permission can update named credentials. After the named credentials are updated, you're prompted to reauthenticate when you perform an operation against any affected sandbox. See Troubleshoot DevOps Center Configuration for step-by-step instructions.

Error: Can't pull changes from a Developer or Developer Pro sandbox

Cause: Developer sandboxes require that source tracking is enabled.

- In the work item, when you select the non-source-tracked sandbox as your development environment, you see `Cannot determine files available to pull - Log in to development environment`. Nothing happens if you attempt to log in.
- When you click **Pull Changes**, you see an error message: `There was an error. Please reload the page or contact your system administrator if the problem persists.`

Action: See “If You Connect to a Non-Source-Tracked Sandbox” in Pull Changes from the Development Environment for options on how to work around this issue.

Error: Some DevOps Center fields aren’t viewable or editable

Cause: Check to see if you turned on encryption for custom fields in managed packages in the org in which DevOps Center is installed.

Action: Disable this option before installing the DevOps Center package. See *Salesforce Help*: Encrypt Custom Fields in Installed Managed Packages.

Error: “The request was invalid. Resource protected by organization SAML enforcement.”

Cause: GitHub repos owned by an organization aren’t visible in DevOps Center until an organization account owner provides access.

Action: See *Salesforce Help*: If an Organization Owns the GitHub Repo for instructions on working around this issue.

Error: “The request was invalid. URL does not reference a valid SFDX project. projectUrl”

Cause: A project repository must contain a Salesforce DX project to work with DevOps Center. The easiest way to create a repository is to use a template we provide in GitHub to create your project repository.

Action: Log in to GitHub and use the provided repository template at github.com/forcedotcom/dx-empty to create your project repository. See your team manager for assistance.

Error: no CustomObject named DevopsEnvironment found or no CustomObject named DevopsRequestInfo found

Cause: Metadata deployment from a sandbox to a production org fails because the DevopsEnvironment and DevopsRequestInfo objects are available only in your sandbox. These objects are used to manage commit, metadata, and data deployment features. You can only deploy them to a production org with the required permissions.

Action: Before you commit or deploy changes, manually remove the DevopsEnvironment and DevopsRequestInfo objects from the metadata XML file.

If you use DX Inspector, you can exclude these objects through the UI. Go to the Change Management page and deselect the DevopsEnvironment and DevopsRequestInfo objects from your change list.

Error: Deployment of the work items into the next stage failed

This error occurs when deploying or promoting metadata components with unsupported special characters to a pipeline stage. The error text displays the component name and indicates that it wasn't able to find the component in the zipped directory. To fix this error, replace the non-supported special characters with their URL-encoded versions.

Error: Promotion Stopped Due to New Events Dialog

This error can occur if you attempt to promote work items but DevOps Center is processing pending events from the source control system. You sometimes see this dialog in combination with the "Too Many Callouts" error message. To fix this error, you perform a SOQL query to determine if event processing is continuing as expected or has stalled.

Error: sf_devops:Too many callouts: 101

When attempting a promotion, a red banner displays the "too many callouts" error message.

Error: There was an internal data model error. Please reload the page...

A work item indicates that it was externally merged but you didn't externally merge it. When you attempt to complete the promotion, you see a red banner with the "internal data model" error message.

DevOps Center MCP Tools Error: No Project Repository Found

When you try to use the DevOps Center MCP tools, you receive an error message that no project repository is connected to your workspace.

DevOps Center MCP Tools Error: Incorrect Project or Org Is Active

You enter a command for a specific work item, but the tool responds with an error that the item can't be found.

Next Generation DevOps Center Error: Unresponsive Lightning Page

When you access a Lightning page in Lightning App Builder, you receive a warning message: "You can't add a runtime_alm_devops:{component_name} component there." If you dismiss the warning, the page becomes unresponsive, and you can't make any further edits.

Error: Deployment of the work items into the next stage failed

This error occurs when deploying or promoting metadata components with unsupported special characters to a pipeline stage. The error text displays the component name and indicates that it wasn't able to find the component in the zipped directory. To fix this error, replace the non-supported special characters with their URL-encoded versions.

DevOps Center and its underlying technologies, such as Metadata API and Salesforce CLI, support these special characters:

```
/\^$#+?()|[]{}!@$&:',"
```

If you try to promote a component with an unsupported special character, the promotion fails. From the pipeline, click **View Details** to view the error from the Activity History page.

To fix this error, first run a query to view the Submit Component record for the component name with the unsupported special character. Then substitute the unsupported character with its URL-encoded version.

In this example, the problematic component is a page layout named `Layout:Recruiters (LTAM) ver2` that uses full-width parentheses, which are not supported characters. These characters are commonly used in East Asian languages, particularly in Chinese, Japanese, and Korean (CJK) scripts.

1. In the DevOps Center org, run a SOQL query to identify the Submit Component record associated with the component with the unsupported special character.

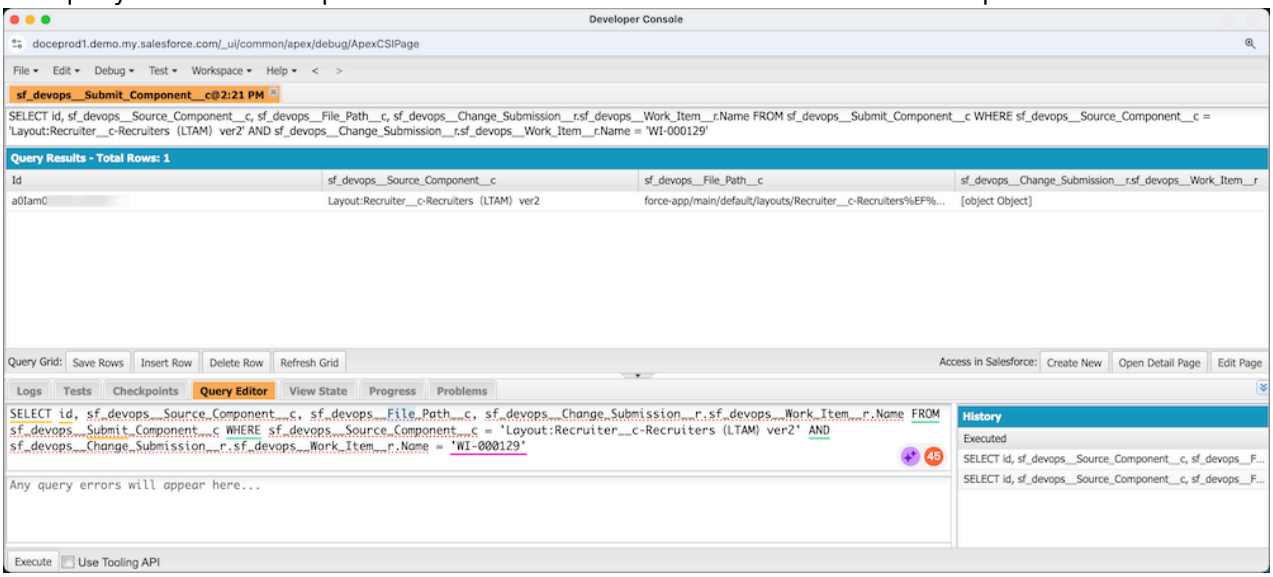
   ```
   SELECT id, sf_devops__Source_Component__c, sf_devops__File_Path__c,
   sf_devops__Change_Submission__r.sf_devops__Work_Item__r.Name FROM sf_devop
   s__Submit_Component__c
   WHERE sf_devops__Source_Component__c = '<source-component-name>' and
   sf_devops__Change_Submission__r.sf_devops__Work_Item__r.Name = '<work-item-n
   ame>'
   ```

   Substitute the name of the **<source-component-name>** and **<work-item-name>** with your specific details.

In our example, the SOQL query looks like this:

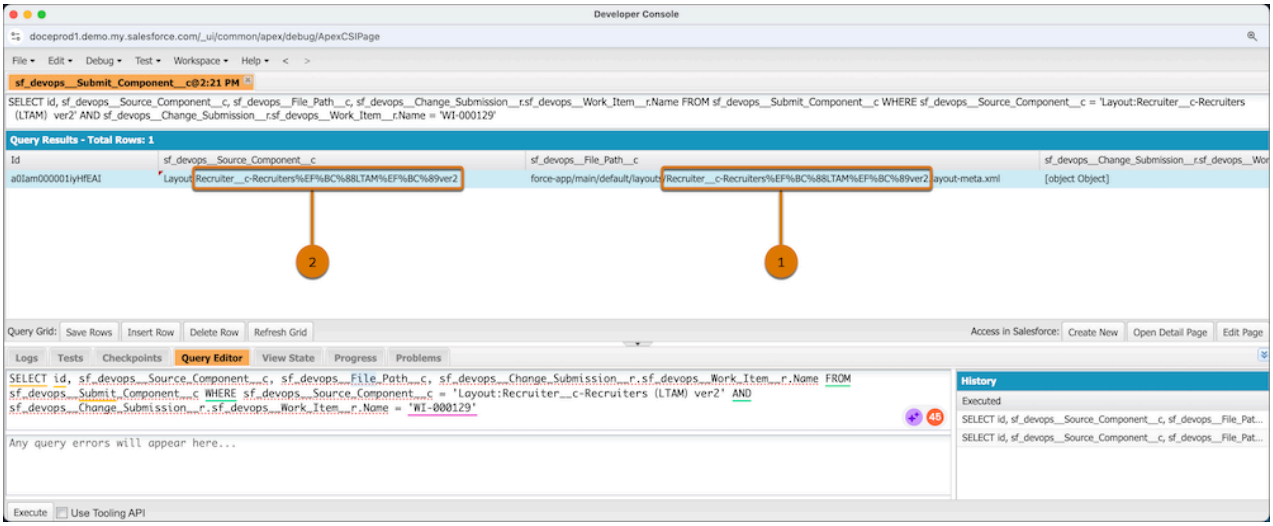
```
SELECT id, sf_devops__Source_Component__c, sf_devops__File_Path__c,
sf_devops__Change_Submission__r.sf_devops__Work_Item__r.Name FROM sf_devop
s__Submit_Component__c
WHERE sf_devops__Source_Component__c = 'Layout:Recruiter__c-Recruiters (LTA
M) ver2'
AND sf_devops__Change_Submission__r.sf_devops__Work_Item__r.Name = 'WI-00012
9'
```

The query returns the file path name that contains the encoded version of the special characters.



2. Update sf_devops__Source_Component__c to replace the special characters with its encoded version.

   In the sf_devops__File_Path__c field, copy the component name with the encoded special character (1), replace the non-encoded name in the sf_devops__Source_Component__c field (2). If using Developer Console as your query editor, you can double-click the fields when copying and pasting.



3. Repeat to fix any other affected source components.

4. Save the changes.
5. Back in DevOps Center, try the promotion again.

Error: Promotion Stopped Due to New Events Dialog

This error can occur if you attempt to promote work items but DevOps Center is processing pending events from the source control system. You sometimes see this dialog in combination with the “Too Many Callouts” error message. To fix this error, you perform a SOQL query to determine if event processing is continuing as expected or has stalled.

1. In DevOps Center, click the Home button.
2. Click the quick access menu (⚙) then click **Developer Console**.

   In this example, the target stage is UAT.

   a. Select the Query Editor tab.
   b. Execute this SOQL query to list the current commit information that DevOps Center is processing.

      Substitute the name of your target stage and DevOps Center project name.

      ```
      SELECT Id, Name, sf_devops__Branch__r.sf_devops__Remote_Reference__c, sf_d
      evops__Branch__r.sf_devops__Remote_Reference_Date__c
      FROM sf_devops__Pipeline_Stage__c WHERE Name = 'UAT' and sf_devops__Pipeli
      ne__r.sf_devops__Project__r.Name = 'Recruiting App'
      ```

   c. Note the values in the sf_devops_Remote_Reference__c and sf_devops_Remote_Reference_Date__c fields.
   d. Attempt the promotion again.

      If the promotion is successful, you can continue with your work. If the promotion is unsuccessful, go to the next step.

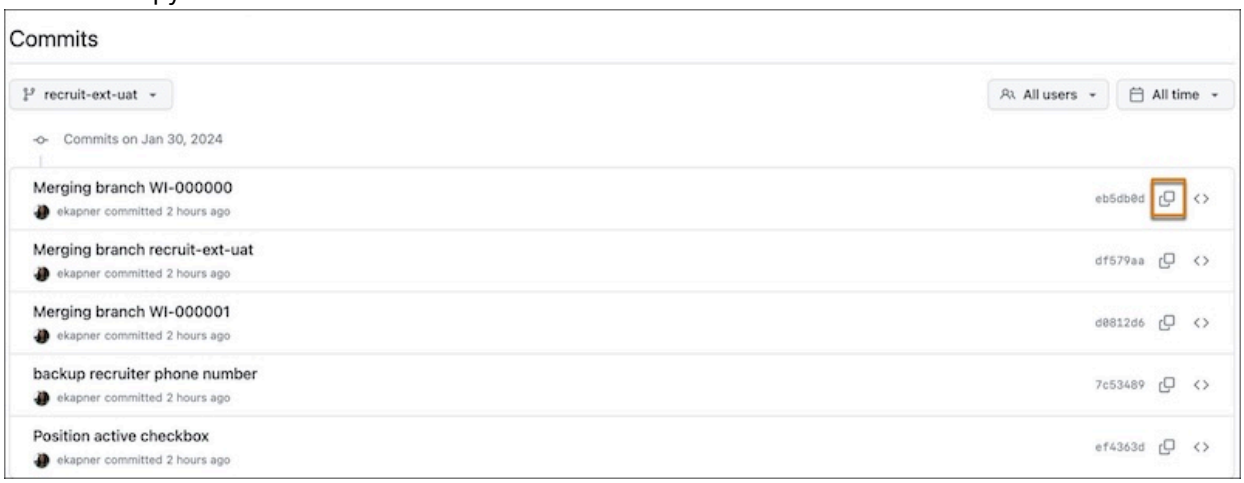
   e. In Developer Console, execute the query again.
   - If the values in the sf_devops_Remote_Reference__c and sf_devops_Remote_Reference_Date__c fields changed, DevOps Center is processing the events correctly. Keep trying the promotion until all external events are processed.
   - If the values in the sf_devops_Remote_Reference__c and sf_devops_Remote_Reference_Date__c fields didn’t change between executions, DevOps Center has encountered an error processing the events. Go to the next step.
3. In Pipeline Stages, click the dropdown menu in the target stage, then select **View Branch in Source Control**.

   In this example, the target stage is UAT.

4. In GitHub, find the unique ID (SHA) for the latest commit.
   a. Click the Commits history icon, which is a clock next to the total number of commits for this branch.

The most recent SHA is listed first.

b. Click the Copy Full SHA icon.



c. Keep the SHA in your clipboard, or paste it in a text file to save it for use in the next step.

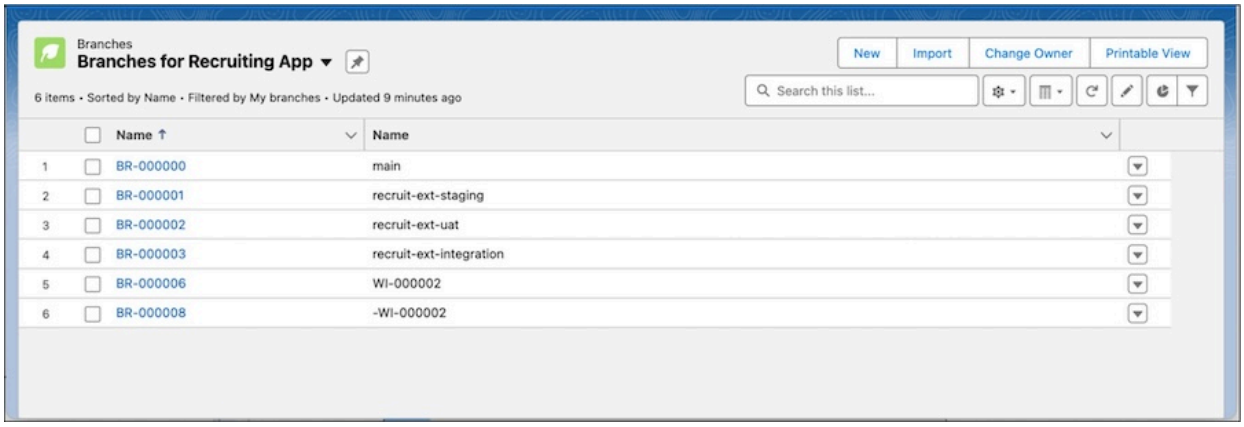
5. View the list of branches.

a. In DevOps Center, in the org URL, delete `sf_devops/DevOpsCenter.app` and replace it with `lightning/o/sf_devops__Branch__c/list`.

```
https://<my-domain>.lightning.force.com/lightning/o/sf_devops__Branch__c/list
```

b. Find the branch record that corresponds to the target branch, for example, recruiting_app_uat.

If you don't see any branches or the repo contains many branches, create a list view to display your branches. To find the exact branch, include the branch name.



c. Delete the existing value for Remote Reference and paste the SHA that you copied.
d. Update the Remote Reference Date field to the current date and time.
e. Click **Save**.

6. Promote the work items.

Error: sf_devops:Too many callouts: 101

When attempting a promotion, a red banner displays the “too many callouts” error message.

If you also see the Promotion Stopped Due to New Events dialog, see Error: Promotion Stopped Due to New Events Dialog instead for troubleshooting instructions.

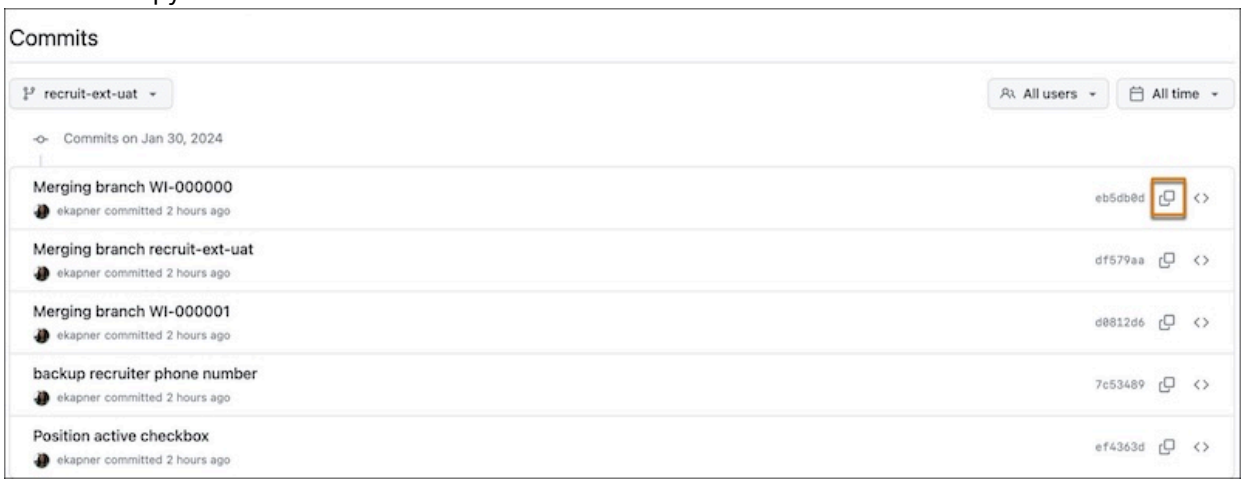
1. In Pipeline Stages, click the dropdown menu in the target stage, then select **View Branch in Source Control**.

   In this example, the target stage is UAT.

2. In GitHub, find the unique ID (SHA) for the latest commit.
   a. Click the Commits history icon, which is a clock next to the total number of commits for this branch.

The most recent SHA is listed first.

b. Click the Copy Full SHA icon.



c. Keep the SHA in your clipboard, or paste it in a text file to save it for use in the next step.

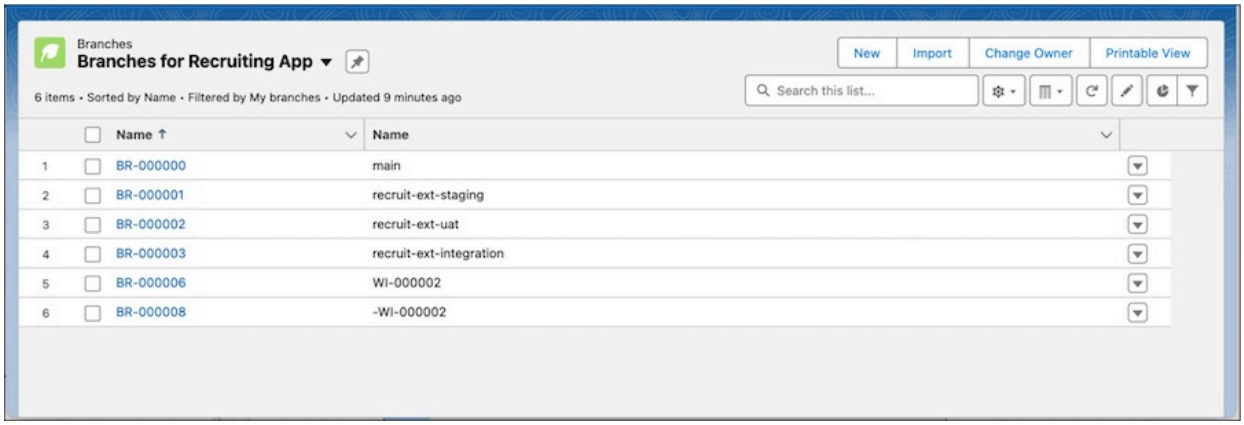
3. View the list of branches.

a. In DevOps Center, in the org URL, delete `sf_devops/DevOpsCenter.app` and replace it with `lightning/o/sf_devops__Branch__c/list`.

```
https://<my-domain>.lightning.force.com/lightning/o/sf_devops__Branch__c/list
```

b. Find the branch record that corresponds to the target branch, for example, recruiting_app_uat.

If you don't see any branches or the repo contains many branches, create a list view to display your branches. To find the exact branch, include the branch name.



c. Delete the existing value for Remote Reference and paste the SHA that you copied.

d. Update the Remote Reference Date field to the current date and time.

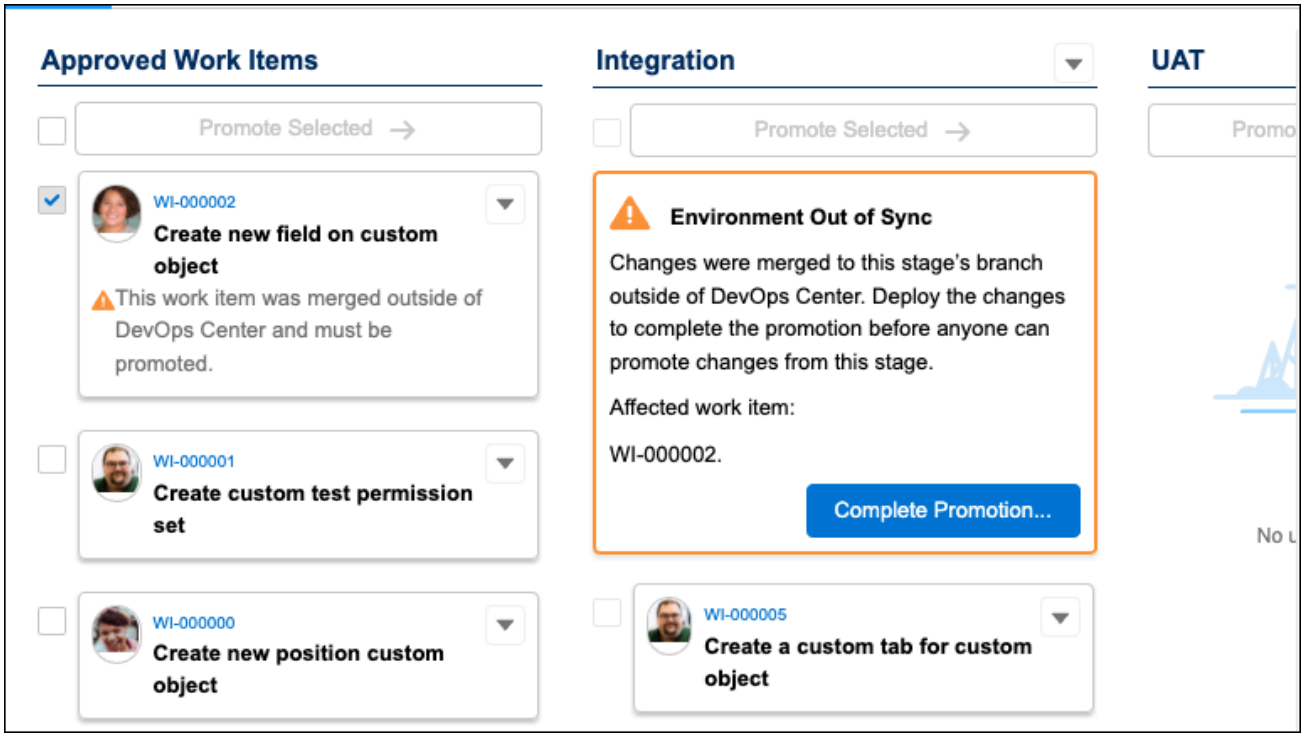
e. Click **Save**.

4. Promote the work items.

Error: There was an internal data model error. Please reload the page...

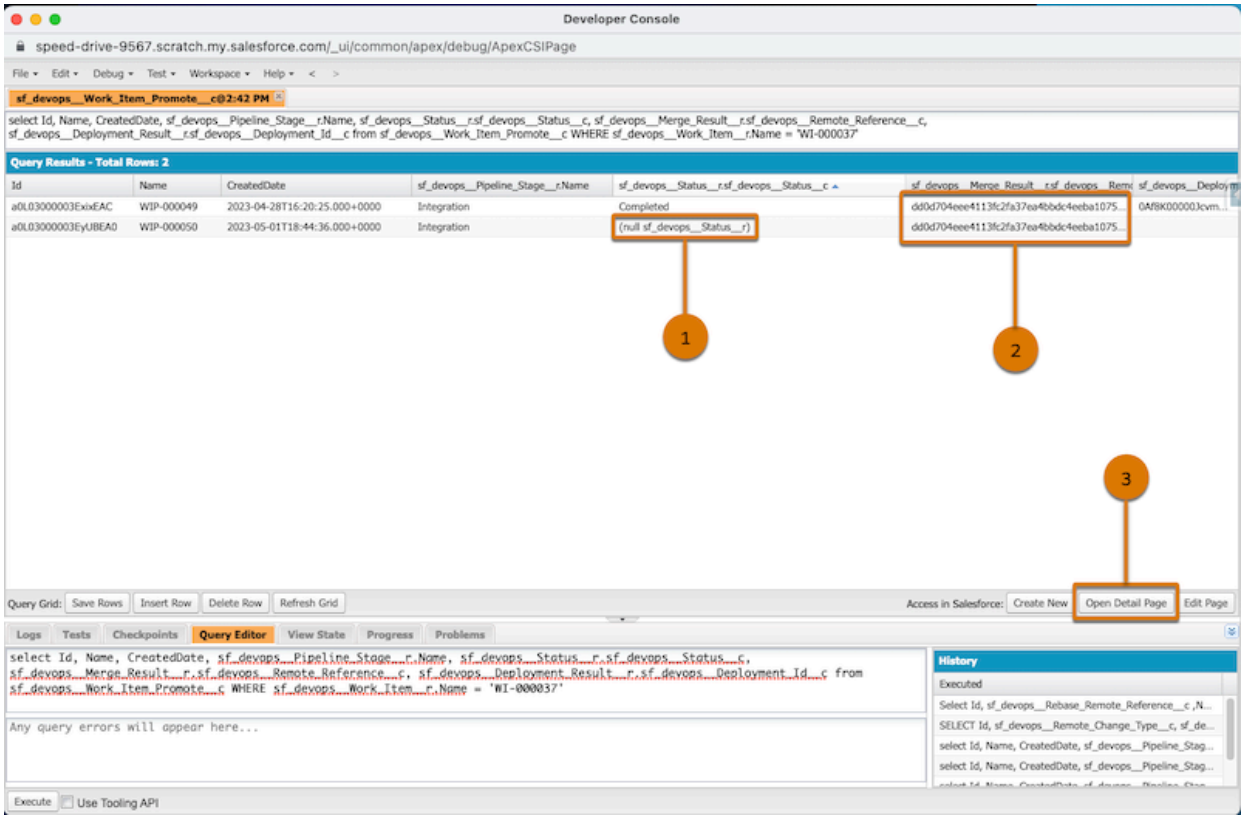
A work item indicates that it was externally merged but you didn't externally merge it. When you attempt to complete the promotion, you see a red banner with the "internal data model" error message.

Due to event processing delays, a duplicate Work Item Promote (WIP) record was erroneously created.



Find and delete the duplicate Work Item Promote record.

1. In the Pipeline Stages view, note the affected work item number, for example, WI-000002.
2. Find the duplicate Work Item Promote record.
   a. In Developer Console, select the Query Editor tab.
   b. Enter this SOQL query, substituting the example work item number you're trying to promote. In this example, line breaks were added for readability purposes. If you copy this query, remove the line breaks before running it.
   c. Look for the record with a null status for `sf_devops_Status__r` (1) and a duplicate value for Merge_Result/Remote_Reference for `sf_devops_Merge_Result__c`, `sf_devops_Remote_Reference__c` (2).



3. To open the record, click to highlight the null record, then click **Open Detail Page** (3).
4. Delete the WIP record.
5. Back in DevOps Center, refresh the Pipeline Stages view to remove the External Merge message. Now you can promote your changes.

DevOps Center MCP Tools Error: No Project Repository Found

When you try to use the DevOps Center MCP tools, you receive an error message that no project repository is connected to your workspace.

Cause

This error occurs when your IDE is open but isn't connected to a Git-based project repository. The tool needs access to the repository associated with your DevOps Center project to analyze conflicts.

Action

1. In your IDE, open the Command Palette by pressing `Cmd+Shift+P` (macOS) or `Ctrl+Shift+P` (Windows/Linux).
2. Enter `Git: Clone` and select it.
3. Paste the URL for your project's Git repository and press **Enter**.
4. Select a location on your local drive to store the project.
5. After the project is cloned, click **Open** in the notification window to open the repository in your workspace.

See Also

Agentforce Vibes IDE: Supported Editions

DevOps Center MCP Tools Error: Incorrect Project or Org Is Active

You enter a command for a specific work item, but the tool responds with an error that the item can't be found.

Cause

This error can occur if you have a different project open in your IDE than the one related to the work item that you're trying to resolve, or if your IDE is connected to the wrong Salesforce org.

Action

1. Make sure that you have the correct project open in your IDE. If not, from the File menu, select *Open Folder* and go to the correct repository.
2. From the status bar of your IDE, select the Salesforce org where your DevOps Center is installed.

See Also

Agentforce Vibes IDE: Supported Editions

Next Generation DevOps Center Error: Unresponsive Lightning Page

When you access a Lightning page in Lightning App Builder, you receive a warning message: "You can't add a runtime_alm_devops:{component_name} component there." If you dismiss the warning, the page becomes unresponsive, and you can't make any further edits.

Cause

This error appears because a `runtime_alm_devops` component, such as `runtime_alm_devops:activatePipeline` is no longer available in your org. If you had previously added this component to a page, editing that page now results in an error because there's no correct reference to the component.

Action

1. Run the command in your terminal to retrieve the metadata for the specific Lightning page. Replace `<Page_API_Name>` with the actual API name of your page.

   ```
   sfdx force:source:retrieve -m FlexiPage:<Page_API_Name>
   ```

2. Open the retrieved file and search the XML file for the name of the problematic component.

   For example, search for `your_namespace:componentName` in the file path `force-app/main/`

```
default/flexipages/<Page_API_Name>.flexipage-meta.xml .
```

3. Locate the `<componentInstance>` block that surrounds the component name. Delete the entire block from the opening tag to the closing tag.

   Make sure you delete only the specific `<componentInstance>` block. Don't leave empty lines or break the surrounding `<flexiPageRegions>` tags.

4. Save the .xml file.
5. Deploy the updated metadata back to your org by running this command.

   ```
   sfdx force:source:deploy -m FlexiPage:<Page_API_Name>
   ```

6. To verify the fix, open Lightning App Builder for the page you modified.
   The warning or error associated with the component is resolved.

DevOps Center

DevOps Testing extends the capabilities of DevOps Center by bringing AI-powered testing and quality control into your DevOps pipeline. Integrate DevOps Testing with testing tools, set quality gates, and run test suites to thoroughly test changes as they move through the DevOps pipeline. This reduces production issues and ensures higher-quality releases.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Application Lifecycle Management with DevOps Testing and DevOps Center

DevOps Testing works alongside DevOps Center, streamlining release management and ensuring error-free deployments.

Install and Configure DevOps Testing

Complete these one-time installation and setup tasks before you move on to the more specific tasks in the rest of this guide.

Get to Know the DevOps Testing UI

If you're new to DevOps Testing, here's a quick overview of the UI and what you can achieve on each tab.

Integrate Test Providers for Unified Testing

Centralize your test management, even when your test assets are in other testing tools and platforms. On the Test Providers tab, you can integrate both Salesforce testing tools and partner test providers. After you configure the test providers, manage and run test suites directly from DevOps Testing.

Tests and Test Suites

Tests and test suites are the building blocks of your software testing lifecycle. They ensure quality and reliability of your work items. With DevOps Testing, you can easily view and manage your tests and test suites across multiple test providers.

Quality Gate Rules

Quality gate rules are checkpoints that make sure only work items that meet your quality criteria move to the next pipeline stage. Use quality gate rules to maintain code quality, ensure reliable deployments, and identify issues early in the application lifecycle. When a quality gate rule fails, you can analyze the reason and resolve any issues before attempting to promote the work item again.

DevOps Pipeline Stage Assignments

Assign test suites and quality gate rules to your In Progress work items and pipeline stages on the Stage Assignments tab.

Run Your Test Suites

With DevOps Testing, you can run test suites for any test provider. You can run these test suites manually as needed, or automatically based on specific DevOps Center events. After each test suite run, DevOps Testing automatically generates comprehensive test results, which you can review.

Test Provider Sync History

See the outcome and details of your test provider sync process in a single, unified summary. It provides the sync status, timestamps, and reasons for any sync failures. Use this information for auditing and troubleshooting issues.

View DevOps Testing Events in DevOps Center Activity History

DevOps Center provides a comprehensive history of all key DevOps Testing events, including quality gate rule evaluation, test suite group run, stage assignments, and errors (failures). You can use these events and their associated details for auditing and troubleshooting.

Test Data Retention and Cleanup

In DevOps Testing, the data retention and cleanup processes automatically manage historical test data. A daily cleanup job automatically deletes data older than the set retention period, improving performance and data governance.

Troubleshoot DevOps Testing Issues

Here are some tips on troubleshooting common issues encountered while using DevOps Testing. If needed, get help from your test manager or Salesforce system administrator to resolve these issues.

DevOps Testing Glossary of Terms

Get familiar with the key terms and concepts related to DevOps Testing.

Application Lifecycle Management with DevOps Testing and DevOps Center

DevOps Testing works alongside DevOps Center, streamlining release management and ensuring error-free deployments.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To manage application lifecycle management with DevOps Testing:	DevOps Testing Manager OR DevOps Testing User
To manage application lifecycle management with DevOps Center:	DevOps Center OR DevOps Center Manager OR DevOps Center Release Manager

Note User permissions required for each step can vary.

Before you start, make sure that you completed these steps.

- Install and configure DevOps Center
  - Set up DevOps Center for GitHub.
  - Create a GitHub project.
  - Build and activate your pipeline.
- Install DevOps Testing

Let's consider a scenario where you've been using Salesforce DevOps Center for application lifecycle management. You've experienced the challenges of updating code changes to production and the occasional last-minute fixes due to unexpected bugs. Now, you're introduced to Salesforce DevOps Testing, a testing platform designed specifically for DevOps Center.

For example, you have a critical code update ready in your development environment. This update must be thoroughly tested and deployed to production. Here's how you can use DevOps Testing and DevOps Center together to manage your application lifecycle management process.

Let's begin your application lifecycle management journey.

1. Add Salesforce Code Analyzer v5 as a test provider in DevOps Testing. See Configure Test Providers for Active DevOps Center Projects.

   a. Sync tests and test suites from Salesforce Code Analyzer v5. Verify sync completion in DevOps Testing. See Tests and Test Suites.
   b. Go to the Test Suites and Tests tabs, and make sure that all tests and test suites from Code Analyzer v5 are visible.
2. Create quality gate rules in DevOps Testing to define the pass and fail criteria. See Create a Quality Gate Rule.
3. Assign Salesforce Code Analyzer v5 to the Work Item: Review stage of your DevOps Center pipeline. See DevOps Pipeline Stage Assignments.
4. Create a work item and commit your changes in DevOps Center. See Commit Changes to the Source Control Repository
5. Create a review for the work item in DevOps Center. This action starts the automatic run of the assigned test suites and quality gate rules. See Run Test Suites Triggered by DevOps Center Events.
6. Verify your test results.
   a. If all quality gate rules pass, the work item is promoted to the next stage in the pipeline.
   b. If any quality gate rules fail, go to DevOps Testing to identify the root cause of the failure. See Analyze Test Reports to Identify Failures.

See Also

Install and Configure DevOps Center: Assign the DevOps Center Permission Sets
Salesforce Code Analyzer Guide: Salesforce Code Analyzer v5 (Beta)

Install and Configure DevOps Testing

Complete these one-time installation and setup tasks before you move on to the more specific tasks in the rest of this guide.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To install and configure DevOps Testing:	Download AppExchange Packages

1. Install the latest version of DevOps Center.
   See Install and Configure DevOps Center.
2. Install the managed package from AppExchange.
   See Install the DevOps Testing Managed Package.
3. Assign permission sets to manage user access to DevOps Testing components such as test providers, test suites, and quality gate rules.
   See Assign Permission Sets to Users Working in DevOps Testing.
4. If you're using Salesforce Code Analyzer v5 as your test provider, complete the optional steps.

See Optional Setup Tasks to Integrate Code Analyzer v5 as Test Provider.

Install the DevOps Testing Managed Package
The managed package listing is available on AppExchange.

Assign Permission Sets to Users Working in DevOps Testing
DevOps Testing includes two permission sets based on user roles and responsibilities. The permission sets include individual user permissions and object permissions.

Optional Setup Tasks to Integrate Code Analyzer v5 as Test Provider
Make sure that your code adheres to best practices, and identify issues early in the development process with Salesforce Code Analyzer v5. To add Code Analyzer v5 as a test provider, follow these steps.

Install the DevOps Testing Managed Package

The managed package listing is available on AppExchange.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To install and configure DevOps Testing:	Download AppExchange Packages

1. Log in to your AppExchange account.
2. Go to DevOps Testing listing, and click **Get It Now**.
3. Choose a destination for the package, and agree to the terms and conditions.
4. Click **Confirm and Install**.
5. Log in to your Salesforce org.
6. Select **Install for Admins Only**.
7. Click **Install**.
8. Approve third-party access.
9. Click **Done**.
   The package can take a while to install. An email is sent when the installation is completed.
10. To confirm DevOps Testing installation, from Setup, enter `Installed Packages` in the Quick Find box, and then select **Installed Packages**.
    You see an entry for DevOps Testing.

Assign Permission Sets to Users Working in DevOps Testing

DevOps Testing includes two permission sets based on user roles and responsibilities. The permission

sets include individual user permissions and object permissions.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To assign permission sets to users:	DevOps Testing Manager

Permission Set	Description
DevOps Testing Manager	Provides read-write access to most of the objects. The DevOps_Testing_Manager permission set includes these user permissions. • Add and sync test providers. • Create, edit, enable, or disable quality gate rules. • Assign test suites and quality gate rules to specific DevOps events. • Run test suites both manually and through events in DevOps Center (for users with pipeline access in DevOps Center).
DevOps Testing User	Provides limited write access, but grants read access to most objects. The DevOps_Testing_User permission set includes these user permissions. • View test providers, assignments, and quality gate rules, but not change them. • View tests and test suites, but not change them. • Run test suites both manually and through events in DevOps Center (for users with pipeline access in DevOps Center). • View test results.

1. From Setup, enter `Permission Sets` in the Quick Find box, and then select **Permission Sets**.
2. Select **DevOps_Testing_Manager** or **DevOps_Testing_User** based on your requirements.
3. Click **Manage Assignments**, and then **Add Assignments**.
4. Select the checkboxes next to the names of the users that you want to assign to the permission set, and click **Assign**.
5. Click **Done**.

Optional Setup Tasks to Integrate Code Analyzer v5 as Test Provider

Make sure that your code adheres to best practices, and identify issues early in the development process with Salesforce Code Analyzer v5. To add Code Analyzer v5 as a test provider, follow these steps.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Choose one of these options based on whether you're setting up Code Analyzer v5 for a new or existing DevOps Center project.

Configure Code Analyzer v5 as a Test Provider for a New DevOps Center Project
Before configuring the pipeline stage environments for your new DevOps Center project, complete these steps if you're using Salesforce Code Analyzer v5 as your test provider.

Configure Code Analyzer v5 as a Test Provider For Existing DevOps Center Project
Sync the Git workflow across branches for existing DevOps Center projects with configured pipeline environments. Complete this optional task only if you're using Salesforce Code Analyzer v5 as your test provider.

Configure Code Analyzer v5 as a Test Provider for a New DevOps Center Project

Before configuring the pipeline stage environments for your new DevOps Center project, complete these steps if you're using Salesforce Code Analyzer v5 as your test provider.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To configure Code Analyzer v5 as a test provider:	DevOps Testing Manager

1. Go to the project repository on GitHub.
2. To sync the Code Analyzer v5 rules, create a file named `sync-code-analyzer-tests.yml`.
3. Copy the `sync-code-analyzer-tests.yml`file to the `.github/workflows/` directory on the main branch.

We recommend that you add this YAML file to the main branch of your project Git repository immediately after you create a new project. This file automatically copies the file to all pipeline stages when you configure your pipeline.

4. Paste the code in the `sync-code-analyzer-tests.yml` file.

```
name: Sync Code Analyzer Tests Workflow
on: workflow_dispatch
jobs:
  salesforce-code-analyzer-workflow:
    runs-on: ubuntu-latest
    steps:
      - name: Check out files
        uses: actions/checkout@v4

      - name: Install Salesforce CLI
        run: npm install -g @salesforce/cli@latest

      - name: Setup Code Analyzer 5
        run: sf plugins install code-analyzer

      - name: Get Code Analyzer V5 rules
        id: get-code-analyzer-rules
        run: |
          if [ -f .github/workflows/config/code-analyzer-config.yml ]; then
            echo "Config file exists"
            sf code-analyzer rules --config-file .github/workflows/config/co
de-analyzer-config.yml --rule-selector=all --output-file ca-v5-rules.json
          else
            echo "Config file does not exist -- Creating default test suite
s"
            sf code-analyzer rules --output-file ca-v5-rules.json
          fi

      - uses: actions/upload-artifact@v4
        with:
          name: code-analyser5-tests
          path: ./ca-v5-rules.json
```

5. To run Code Analyzer v5, create a file named `code-analyzer-v5.yml`.
6. Copy the `code-analyzer-v5.yml` file to the `.github/workflows/` directory on the main branch.
7. Paste the code in the `code-analyzer-v5.yml` file.

```
name: Run Code Analyzer 5 Workflow
on:
  workflow_dispatch:
```

```
    inputs:
      config_branch:
        description: "Config branch"
        required: false
        default: ""

jobs:
  salesforce-code-analyzer-workflow:
    runs-on: ubuntu-latest
    steps:
      - name: Setup Node
        uses: actions/setup-node@v4
        with:
          node-version: ">=20"

      - name: Check out files
        uses: actions/checkout@v4

      - name: Install Salesforce CLI
        run: npm install -g @salesforce/cli@latest

      - name: Setup Code Analyzer 5
        run: sf plugins install code-analyzer@latest

      - name: Check config file presence
        id: check-config-file
        run: |
          if [ -f .github/workflows/config/code-analyzer-config.yml ]; then
            echo "config_exists=true" >> $GITHUB_OUTPUT
          else
            echo "config_exists=false" >> $GITHUB_OUTPUT
          fi

      - name: Checkout config file from target branch
        if: ${{ inputs.config_branch != '' }}
        run: |
          cd $GITHUB_WORKSPACE
          if [ "${{ steps.check-config.outputs.config_exists }}" == "true"
]; then
            git fetch
            git checkout origin/${{inputs.config_branch}} .github/workflows/
config/code-analyzer-config.yml
          fi
```

```
      - name: Run Code analyzer
        uses: forcedotcom/run-code-analyzer@v2
        with:
          run-arguments: >-
            --workspace .
            ${{ steps.check-config.outputs.config_exists == 'true' && '--con
fig-file .github/workflows/config/code-analyzer-config.yml' || '' }}
            --output-file results.html
          results-artifact-name: salesforce-code-analyzer-v5-results
```

8. [Optional] Create a configuration file named `code-analyzer-config.yml`.
   a. Add the `code-analyzer-config.yml` file to the `.github/workflows/config/`directory.
   b. Run the command to retrieve the default configuration file, and copy it to the directory.

      ```
      sf code-analyzer config --output-file code-analyzer-config.yml
      ```

   c. Update the default configuration file for a specific pipeline stage as needed. See Customize the v5 Configuration (Beta).

Configure Code Analyzer v5 as a Test Provider For Existing DevOps Center Project

Sync the Git workflow across branches for existing DevOps Center projects with configured pipeline environments. Complete this optional task only if you're using Salesforce Code Analyzer v5 as your test provider.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To configure Code Analyzer v5 as a test provider:	DevOps Testing Manager

1. Create a work item in DevOps Center.
2. Go to the project Git repository, and perform these steps on the branch associated with the created work item ID.
   a. To sync the Code Analyzer v5 rules, create a file named `sync-code-analyzer-tests.yml`.
   b. Copy the `sync-code-analyzer-tests.yml`file to the `.github/workflows/` directory on the work item branch.
   c. Paste the code in the `sync-code-analyzer-tests.yml` file.

```
name: Sync Code Analyzer Tests Workflow
on: workflow_dispatch
jobs:
 salesforce-code-analyzer-workflow:
   runs-on: ubuntu-latest
   steps:
     - name: Check out files
       uses: actions/checkout@v4


     - name: Install Salesforce CLI
       run: npm install -g @salesforce/cli@latest


     - name: Setup Code Analyzer 5
       run: sf plugins install code-analyzer


     - name: Get Code Analyzer V5 rules
       run: sf code-analyzer rules --config-file .github/workflows/config/
code-analyzer-config.yml --rule-selector=all --output-file ca-v5-rules.jso
n


     - uses: actions/upload-artifact@v4
       with:
         name: code-analyzer5-tests
         path: ./ca-v5-rules.json
```

d. To run Code Analyzer v5, create a file named `code-analyzer-v5.yml`.
e. Copy the `code-analyzer-v5.yml` file to the `.github/workflows/` directory on the work item branch.
f. Paste the code in the `code-analyzer-v5.yml` file.

```
name: Run Code Analyzer 5 Workflow
on:
  workflow_dispatch:
    inputs:
      config_branch:
        description: "Config branch"
        required: false
        default: ""
```

```
jobs:
  salesforce-code-analyzer-workflow:
    runs-on: ubuntu-latest
    steps:
      - name: Setup Node
        uses: actions/setup-node@v4
        with:
          node-version: ">=20"


      - name: Check out files
        uses: actions/checkout@v4


      - name: Install Salesforce CLI
        run: npm install -g @salesforce/cli@latest


      - name: Setup Code Analyzer 5
        run: sf plugins install code-analyzer@latest


      - name: Check config file presence
        id: check-config-file
        run: |
          if [ -f .github/workflows/config/code-analyzer-config.yml ]; the
n
            echo "config_exists=true" >> $GITHUB_OUTPUT
          else
            echo "config_exists=false" >> $GITHUB_OUTPUT
          fi


      - name: Checkout config file from target branch
        if: ${{ inputs.config_branch != '' }}
        run: |
          cd $GITHUB_WORKSPACE
          if [ "${{ steps.check-config.outputs.config_exists }}" == "true"
]; then
            git fetch
            git checkout origin/${{inputs.config_branch}} .github/workflow
s/config/code-analyzer-config.yml
          fi
```

```
      - name: Run Code analyzer
        uses: forcedotcom/run-code-analyzer@v2
        with:
          run-arguments: >-
            --workspace .
            ${{ steps.check-config.outputs.config_exists == 'true' && '--c
onfig-file .github/workflows/config/code-analyzer-config.yml' || '' }}
            --output-file results.html
          results-artifact-name: salesforce-code-analyzer-v5-results
```

3. [Optiona] Create a configuration file named `code-analyzer-config.yml`.
   a. Add the `code-analyzer-config.yml` file to the `.github/workflows/config/`directory.
   b. Run the command to retrieve the default configuration file, and copy it to the directory.

      ```
      sf code-analyzer config --output-file code-analyzer-config.yml
      ```

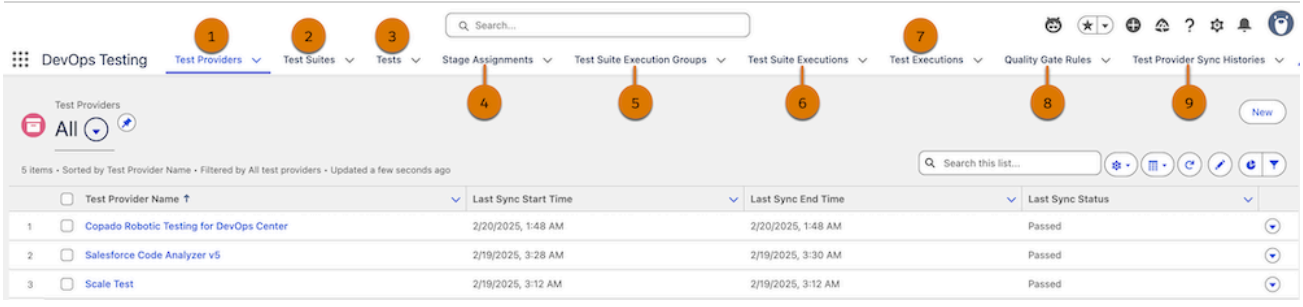
   c. Update the default configuration file for a specific pipeline stage as needed. See Customize the v5 Configuration (Beta).
4. Create a change request for the work item in DevOps Center.
5. Promote the work item to production through all the stages. This action copies the files to all pipeline environment branches.

Get to Know the DevOps Testing UI

If you're new to DevOps Testing, here's a quick overview of the UI and what you can achieve on each tab.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions



- Test Providers (1): View and manage your test providers. We recommend starting here and adding an available test provider.

- Test Suites (2): View and manage your test suites. Here, you can easily create and run test suites.
- Tests (3): View all your tests on a single page. Modify test severity, and mark them as essential for test suite runs.
- Stage Assignment (4): Assign test suites and quality gate rules to your DevOps Center pipeline stages.
- Test Execution Groups (5): View the outcomes of your test suite runs. If your quality gate rules fail, reevaluate them.
- Test Suite Executions (6): View the outcomes of your individual test suite runs.
- Test Executions (7): View detailed analysis of your test runs. See which tests passed and which ones failed.
- Quality Gate Rules (8): Create quality gate rules to prevent low-quality work items from moving through your pipeline stages.
- Test Provider Sync Histories (9): View the details of how your test provider sync performed.

Note If you're using a localized version of DevOps Testing and notice a small amount of user interface text in English, we're actively working to improve your experience and plan to address it in a future release.

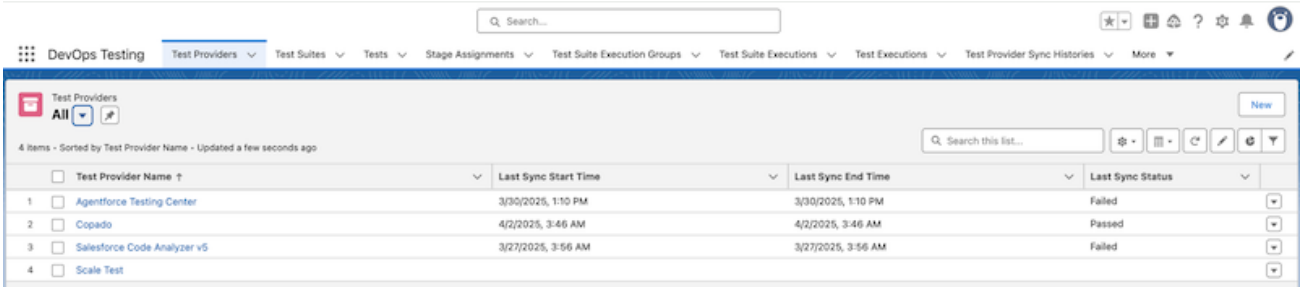
Important We recommend that you don't edit any fields on any tabs, even if they are editable. Changing field values causes unexpected product outcomes. Users with the DevOps Testing Manager permission set can only edit the **Severity** and **Essential** fields in the Tests tab.

Integrate Test Providers for Unified Testing

Centralize your test management, even when your test assets are in other testing tools and platforms. On the Test Providers tab, you can integrate both Salesforce testing tools and partner test providers. After you configure the test providers, manage and run test suites directly from DevOps Testing.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions



If you use multiple test providers and tools as part of your testing strategy, you can consolidate and run them automatically with DevOps Testing. In addition to Salesforce testing tools, you can find partner test

providers on AppExchange. AppExchange makes it easy to discover new test providers within the Salesforce ecosystem that fit your testing needs.

DevOps Testing integrates with multiple Salesforce and partner test providers. After installing the DevOps Testing package, you can install the partner test provider package in your Salesforce org.

DevOps Testing integrates with these partner test providers and tools:

- Copado Robotic Testing
- Provar Automation
- QualityClouds

DevOps Testing also supports test providers from Salesforce, such as Agentforce Testing Center, Apex Unit Testing, Code Analyzer v5, Flow Testing, and Scale Test.

You can check the availability of test providers as you add them. Their status indicates whether they are available, unavailable, already installed, or if a license is required.

The requirements for the Salesforce testing tools are listed in the table.

Test Provider	Testing Type	Notes
Agentforce Testing Center	Agent testing	• See Agentforce Testing Center. • Review Testing API considerations before using Agentforce Testing Center. See Considerations for the Testing API.
Apex Unit Testing	Unit testing for Apex classes and triggers	See Testing Apex.
Flow Testing	Flow tests	See Testing Your Flow Before Activation.
Salesforce Code Analyzer v5	Code analysis	• Free • Requires additional configuration. See Optional Setup Tasks to Integrate Code Analyzer v5 as Test Provider. • When you sync or run Code Analyzer v5 test suites manually without authenticating your source repository with DevOps

Test Provider	Testing Type	Notes
		Center, the test suite run or sync fails. We recommend that you connect your GitHub account through DevOps Center before syncing or running test suites.
Scale Test	Performance testing	• We currently support only lightweight Scale Test integration. With this integration, you can access Scale Test directly from DevOps Center. To get access, contact your customer success representative or account executive. See Scale Test. • Add Scale Test as a test provider in DevOps Testing. • After configuration, the Scale Test icon appears in the DevOps Center pipeline stages for environments with a Scale Test license. This icon provides direct access to the Scale Test setup for each environment. • Select **Manage Scale Tests** from the dropdown to go to the Test Scheduler page in Setup. • Select **View Scale Test Runs** from the dropdown to go to the Test Execution page in Setup. Facing issues? See Troubleshoot DevOps Testing Issues.

Test Provider Limitations

Before you integrate test providers with DevOps Testing, review these limitations and behaviors.

Configure Test Providers for Active DevOps Center Projects

Install and set up a test provider for your DevOps Center project.

Test Provider Limitations

Before you integrate test providers with DevOps Testing, review these limitations and behaviors.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Flow and Apex Unit Testing

- DevOps Testing syncs up to 10,000 Apex unit test records. It skips any additional records.
- Apex unit test and flow test executions with more than 5,000 test records may impact performance.
- Apex unit test or flow test executions fail if their runtime exceeds four hours.
- Flow class name (flowtesting.namespace.flowname) must not exceed 255 characters.
- Apex class name (namespacePrefix.apexClassName) must not exceed 255 characters.
- Flow method name, which is a combination of the test name and flow name, must not exceed 236 characters. If the combined length exceeds this value, DevOps Testing skips the record.
- Apex method name must not exceed 217 characters. If the method name exceeds this value, DevOps Testing skips the record.

Configure Test Providers for Active DevOps Center Projects

Install and set up a test provider for your DevOps Center project.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To configure test providers:	DevOps Testing Manager

Note DevOps Testing syncs up to 10 projects at a time.

1. If you're using a partner test provider, install the managed package from AppExchange.
2. Go to DevOps Testing and select the **Test Providers** tab.
3. Click **New** and select the project that you want to add the test provider to.
4. Click **Next** and select an available test provider.

5. Continue the configuration process. Click **Next** or **Save** based on your test provider.
   If the test provider requires additional details, you're redirected to their setup page. Provide the required information and complete the setup.

After the setup process completes, your tests and test suites are synced automatically. You can view the records on the Tests and Test Suites tabs. If a sync is in progress, wait for it to finish before starting another sync process.

Tests and Test Suites

Tests and test suites are the building blocks of your software testing lifecycle. They ensure quality and reliability of your work items. With DevOps Testing, you can easily view and manage your tests and test suites across multiple test providers.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

After you configure a test provider, its test and test suites are automatically synced with DevOps Testing. You can access them on the Tests and Test Suites tabs, respectively. Make sure your test assets remain up-to-date by syncing any changes made to tests or test suites in the test provider.

Note DevOps Testing syncs a maximum of 10,000 test suites and test records from a test provider.

Throughout this guide, we use the term *test suite execution groups*. A test suite execution group is a collection of test suites run for a particular pipeline stage in the DevOps process.

View Your Tests

View your tests in a single, unified summary. On the Tests tab, you can see the test details, including the type of test, severity, DevOps Center project, programming language, and the test provider from which it was synced. From the Tests tab, you can change the test severity and mark tests as essential. All other test changes must be made within the test provider.

Change the Test Severity Level

From the Tests tab, you can change the severity level of tests. Severity levels prioritize failing tests based on their potential impact. DevOps Testing evaluates the quality gate rules according to these severity levels. When creating a quality gate rule, you can set severity-level threshold criteria. If a test with a severity equal to or higher than the threshold fails, the quality gate rule also fails.

Mark a Test as Essential

From the Tests tab, you can mark tests as essential. When you mark a test as essential, the corresponding quality gate rule fails if the essential test fails. This helps to prioritize critical tests and make sure that essential tests are always validated.

View Your Test Suites

View your test suites in a single, unified summary. On the Test Suites tab, you can see the test suite details, including the last-run timestamp, status, DevOps Center project, environment, and the test provider from which it was synced. The Related tab on the test suite record page lists the tests included in the test suite.

Create a Test Suite

You can create new test suites directly from within DevOps Testing if your test provider supports it. After creating a new test suite, you can start a manual sync to make sure it's synced with your test provider.

See Also

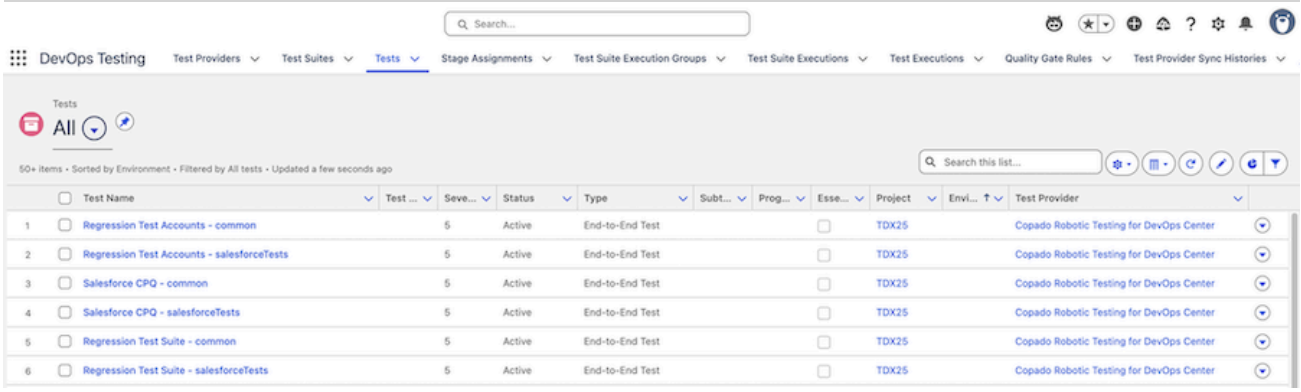
View Your Tests
View Your Test Suites

View Your Tests

View your tests in a single, unified summary. On the Tests tab, you can see the test details, including the type of test, severity, DevOps Center project, programming language, and the test provider from which it was synced. From the Tests tab, you can change the test severity and mark tests as essential. All other test changes must be made within the test provider.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions



Change the Test Severity Level

From the Tests tab, you can change the severity level of tests. Severity levels prioritize failing tests based on their potential impact. DevOps Testing evaluates the quality gate rules according to these severity levels. When creating a quality gate rule, you can set severity-level threshold criteria. If a test with a severity equal to or higher than the threshold fails, the quality gate rule also fails.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To update test severity:	DevOps Testing Manager

These are the severity levels defined in DevOps Testing.

- 1 - Critical
- 2 - High
- 3 - Medium
- 4 - Low
- 5 - Info

1. Go to the Tests tab and select the test for which you want to change the severity level.
2. In the Severity column, click  .
3. Change the severity level to meet your needs.

Mark a Test as Essential

From the Tests tab, you can mark tests as essential. When you mark a test as essential, the corresponding quality gate rule fails if the essential test fails. This helps to prioritize critical tests and make sure that essential tests are always validated.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To mark a test as essential:	DevOps Testing Manager

1. Go to the Tests tab, and select the test that you want to mark as essential.
2. In the Essential column, select the checkbox.

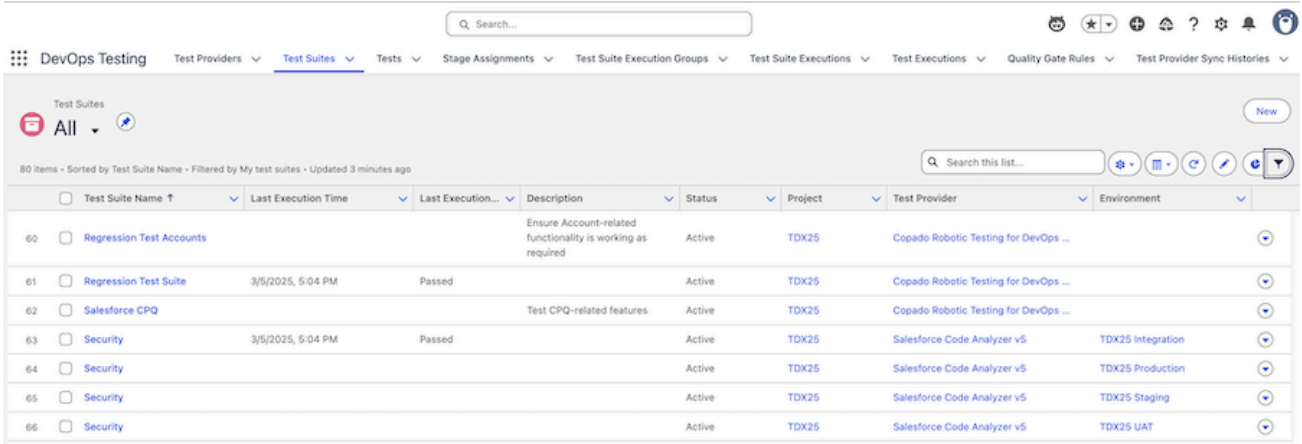
View Your Test Suites

View your test suites in a single, unified summary. On the Test Suites tab, you can see the test suite

details, including the last-run timestamp, status, DevOps Center project, environment, and the test provider from which it was synced. The Related tab on the test suite record page lists the tests included in the test suite.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions



Create a Test Suite

You can create new test suites directly from within DevOps Testing if your test provider supports it. After creating a new test suite, you can start a manual sync to make sure it's synced with your test provider.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To create test suites:	DevOps Testing Manager

Before you start, configure your test provider. See Configure Test Providers for Active DevOps Center Projects.

1. Go to the Test Suites tab, and then click **New**.
2. Select a test provider.
   DevOps Testing displays only test providers that support creating test suites from DevOps Testing.
3. Click **Next**.

4. Select the project for which you want to create a test suite.
   DevOps Testing displays only projects that use the selected test provider.
5. Click **Next**.
   You're redirected to the test provider.
6. Enter the test suite details and finish creating it in the test provider.
7. Open DevOps Testing and go to the Test Providers tab.
8. Go to the test provider record page, and then click **Sync**.
   DevOps Testing lists the DevOps Center projects that are available for sync.
9. Select the projects, and then click **Sync**.

Quality Gate Rules

Quality gate rules are checkpoints that make sure only work items that meet your quality criteria move to the next pipeline stage. Use quality gate rules to maintain code quality, ensure reliable deployments, and identify issues early in the application lifecycle. When a quality gate rule fails, you can analyze the reason and resolve any issues before attempting to promote the work item again.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Quality gate rules use any of these criteria:

- Severity level threshold – If any test with a severity equal to or higher than the threshold fails, the quality gate rule fails. For example, if the severity threshold is set to 3, test failures of severity 1, 2, or 3 cause this quality gate rule to fail.
- Test pass percentage – If the overall test pass rate is below the set percentage, the quality gate rule fails. For example, if the test pass percentage is 90% and fewer than 90% of the tests in the test suite pass, this quality gate rule fails.
- Check for essential test failures – If any test fails that's marked as essential, the quality gate rule fails.

You can assign quality gate rules to each test suite assigned to a pipeline stage. When a work item is promoted, DevOps Testing runs the test suites and evaluates the quality gate rules. If all quality gate criteria are met, the work items containing the changes are moved to the next pipeline stage. If any criterion isn't met, the test suite run automatically fails.

Create a Quality Gate Rule

Create a quality gate rule based on your testing goals. We recommend establishing criteria that are realistic yet strict enough to maintain quality.

See Also

Assign Quality Gate Rules to Work Items and Pipeline Stages

Create a Quality Gate Rule

Create a quality gate rule based on your testing goals. We recommend establishing criteria that are realistic yet strict enough to maintain quality.

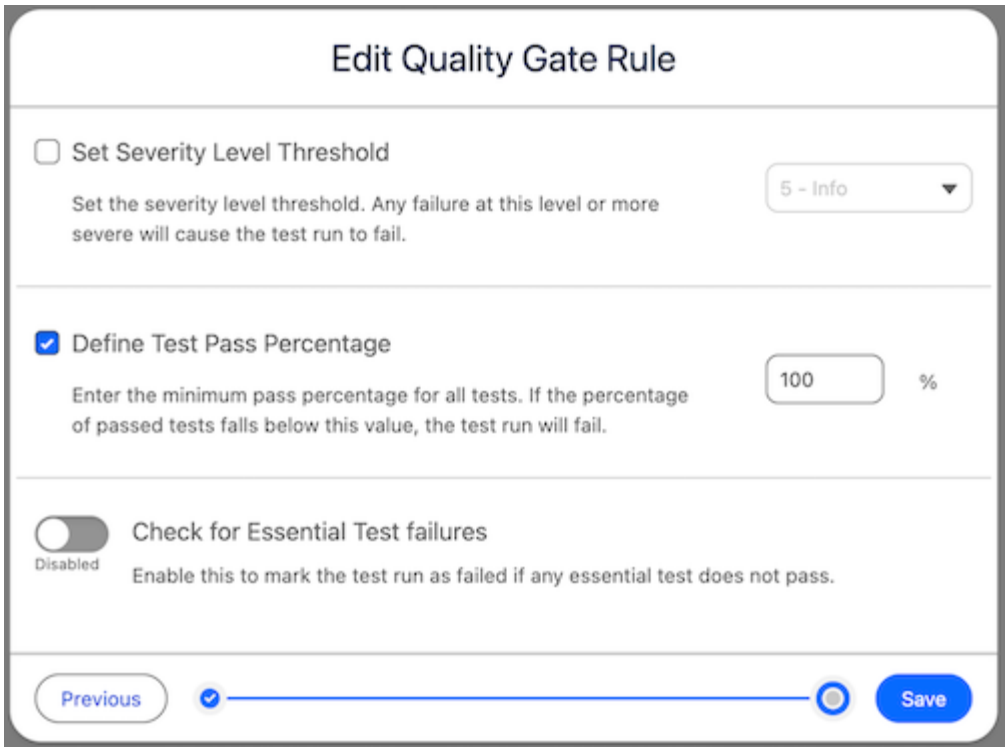
REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To create quality gate rules:	DevOps Testing Manager
To edit quality gate rules:	DevOps Testing Manager

1. From the Quality Gate Rules tab, click **New**.
2. Enter a name and description for the quality gate rule.
3. Click **Next**.
4. On the Criteria page, select the criteria that you want to set for the quality gate rule:
   - Severity level threshold– The quality gate rule fails if a test with a severity level equal to or higher than the threshold level fails.
   - Test pass percentage– The quality gate rule fails if the test pass percentage is less than this value.
   - Essential test failures– The quality gate rule fails if an essential test fails.

   See Quality Gate Rules.
5. Save your changes.
   View the details of the created quality gate rule in the Quality Gate Rules tab.
6. To edit a quality gate rule:
   a. Go to the Quality Gate Rules tab, and select the quality gate rule you want to edit.
   b. Click **Edit**.



See Also

Enable or Disable Quality Gate Rules for Test Suites

DevOps Pipeline Stage Assignments

Assign test suites and quality gate rules to your In Progress work items and pipeline stages on the Stage Assignments tab.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

When you create a review for a work item or promote a work item to the next pipeline stage in DevOps Center, DevOps Testing automatically runs the assigned test suites and quality gate rules. The work item moves to the next pipeline stage only if it meets the defined quality gate rules.

Assign Test Suites to Work Items and Pipeline Stages
Set up test suites for specific work items and events on pipeline stages.

Assign Quality Gate Rules to Work Items and Pipeline Stages
Set up quality gate rules for specific events on pipeline stages to ensure that only work items meeting the set quality criteria move to the next stage.

Enable or Disable Quality Gate Rules for Test Suites
Enable or disable quality gate rules after they're assigned to test suites. While we recommend that you do not disable your quality gate rules, you can temporarily disable them if tests are inconsistent or to speed up deployment. After disabling, reevaluate quality gate rules to promote the changes.

See Also
- Assign Test Suites to Work Items and Pipeline Stages
- Assign Quality Gate Rules to Work Items and Pipeline Stages
- Enable or Disable Quality Gate Rules for Test Suites

Assign Test Suites to Work Items and Pipeline Stages

Set up test suites for specific work items and events on pipeline stages.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To assign test suites to work items and DevOps pipeline stage:	DevOps Testing Manager

1. Go to the Stage Assignments tab.
2. For the pipeline stage record, click **View**.
3. Click **Assign Test Suites**.
4. Select the test suites to assign to the pipeline stage.
5. Save your work.

Assign Quality Gate Rules to Work Items and Pipeline Stages

Set up quality gate rules for specific events on pipeline stages to ensure that only work items meeting the set quality criteria move to the next stage.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To assign quality gate rules to work items and pipeline stages:	DevOps Testing Manager

Before you begin, make sure that you created quality gate rules. See Create a Quality Gate Rule.

1. Go to the Stage Assignments tab.
2. On the pipeline stage record, click **View**.
3. Click **Assign Quality Gate Rules**.
4. Select the quality gate rules to assign to the pipeline stage.
5. Select the test suites to which the quality gate rule is applied.
6. Save your work.
   The quality gate rules are enabled by default.

Enable or Disable Quality Gate Rules for Test Suites

Enable or disable quality gate rules after they're assigned to test suites. While we recommend that you do not disable your quality gate rules, you can temporarily disable them if tests are inconsistent or to speed up deployment. After disabling, reevaluate quality gate rules to promote the changes.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To enable or disable quality gate rules for test suites:	DevOps Testing Manager

1. Go to the Stage Assignments tab.
2. For the pipeline stage record, click **View**.
3. Click **Assign Quality Gate Rules**.
4. To enable or disable a quality gate rule for specific test suites:
   a. In the Enable Quality Gate Rule column, click  .
   b. To enable a quality gate rule, select the **Enable Quality Gate Rule** checkbox.
   c. To disable a quality gate rule, deselect the **Enable Quality Gate Rule** checkbox.
   d. Click **Apply**.
   e. Save your work.
5. To enable or disable a quality gate rule for all test suites:
   a. Go to the quality gate rules record page.
   b. Click the dropdown arrow next to the **Edit** button.
   c. To enable a quality gate rule for all test suites, select **Activate All Assignments**.

d. To disable a quality gate rule for all test suites, select **Deactivate All Assignments**.
e. Click **Confirm**.

Run Your Test Suites

With DevOps Testing, you can run test suites for any test provider. You can run these test suites manually as needed, or automatically based on specific DevOps Center events. After each test suite run, DevOps Testing automatically generates comprehensive test results, which you can review.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Note DevOps Testing runs a maximum of 10,000 records, including both test suites and tests. Records exceeding this limit are ignored.

You can run your test suites in one these ways:

Run Test Suites Manually

Run test suites manually against various pipeline stages in DevOps Center. You can manually run test suites and quality gate rules at any time, independently of when DevOps Center events occur. You can easily identify a manual test suite run from the Event column in the Test Suite Executions tab.

Run Test Suites Triggered by DevOps Center Events

Automatic test suite runs streamline the continuous delivery process by minimizing manual intervention. Test suites and quality gate rules assigned to a pipeline stage are automatically run when a create review or promote event occurs in that stage.

Reevaluate Quality Gate Rules

You can reevaluate failed quality gate rules without rerunning the test suites. Reevaluate these rules after you address an issue and want to quickly verify the quality gate rule results.

Analyze Test Reports to Identify Failures

Test reports show you which test suites passed and, more importantly, which ones failed. Use the insights from these reports to proactively address and fix root causes, and improve the quality of your work items.

See Also

Run Test Suites Manually
Run Test Suites Triggered by DevOps Center Events

Run Test Suites Manually

Run test suites manually against various pipeline stages in DevOps Center. You can manually run test suites and quality gate rules at any time, independently of when DevOps Center events occur. You can easily identify a manual test suite run from the Event column in the Test Suite Executions tab.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

USER PERMISSIONS NEEDED	
To run test suites manually:	DevOps Testing Manager OR DevOps Testing User

1. Go to the Test Suites tab.
2. Select a test suite to access its record page.
3. Click **Run Test Suite**.
4. Select the pipeline stages for which you want to initiate a run of the test suite, and then click **Run**.
5. To monitor the status of the newly created execution records, go to the Test Suite Execution Groups tab, where all the results are listed.

See Also

Run Test Suites Triggered by DevOps Center Events
Analyze Test Reports to Identify Failures

Run Test Suites Triggered by DevOps Center Events

Automatic test suite runs streamline the continuous delivery process by minimizing manual intervention. Test suites and quality gate rules assigned to a pipeline stage are automatically run when a create review or promote event occurs in that stage.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Create Review Event

When you start the create review process in DevOps Center, DevOps Testing creates a parent-level test suite execution group record. This record includes all associated test suites and their test execution

records, with the status set to **In Progress**.

DevOps Testing automatically runs tests for all assigned test suites and updates the corresponding execution records when the run completes. To view the most recent status, refresh the Work Item page in DevOps Center.

The **Ready to Promote** option is disabled while the tests are running. After all quality gate rules pass, refresh the page, and this option becomes available. If the latest test suite execution group record status isn't **In Progress** or if the quality gate rule status isn't **Passed**, a new run starts automatically when you reload the Work Item page.

If you make additional commits to an **In Review** work item, DevOps Testing cancels the previous test suite execution group run. When you refresh the Work Items page in DevOps Center, DevOps Testing automatically starts a new test suite execution group run to get the run details of the latest commit.

Promote Event

When you promote a work item in DevOps Center, DevOps Testing creates a new parent-level test suite execution group record. This record includes entries for the assigned test suites and individual tests, with the status set to **In Progress**.

Promotion of the selected work items is blocked until DevOps Testing validation is complete. To verify the current promotion status, click **Promote**.

When you promote work items, DevOps Testing runs all test suites assigned to the pipeline stage. The status is updated to **Passed**, **Failed**, **Error**, or **Canceled** after the test suite run is complete.

If a promotion attempt is canceled or results in an error, the next promotion attempt automatically starts a new run. If the run fails, details of the failing test suites are displayed. You can use the **Rerun Test Suites** button to run the test suites again.

You can also go to the test suite execution group record to reevaluate the failed quality gate rules.

See Also

- Reevaluate Quality Gate Rules
- Run Test Suites Manually

Reevaluate Quality Gate Rules

You can reevaluate failed quality gate rules without rerunning the test suites. Reevaluate these rules after you address an issue and want to quickly verify the quality gate rule results.

REQUIRED EDITIONS

Available in: Lightning Experience in **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions.

USER PERMISSIONS NEEDED	
To run test suites automatically:	DevOps Testing Manager OR DevOps Testing User

1. Go to the Test Suite Execution Groups tab.
2. Identify the failed test suite execution group from the Status and Quality Gate Rules Status columns.
3. Select the test suite execution group record that you want to analyze.
4. Click **Re-Evaluate Quality Gate**.
   DevOps Testing lists quality gate rules that are reevaluated.
5. Click **Confirm**.

See Also

Quality Gate Rules
Create a Quality Gate Rule

Analyze Test Reports to Identify Failures

Test reports show you which test suites passed and, more importantly, which ones failed. Use the insights from these reports to proactively address and fix root causes, and improve the quality of your work items.

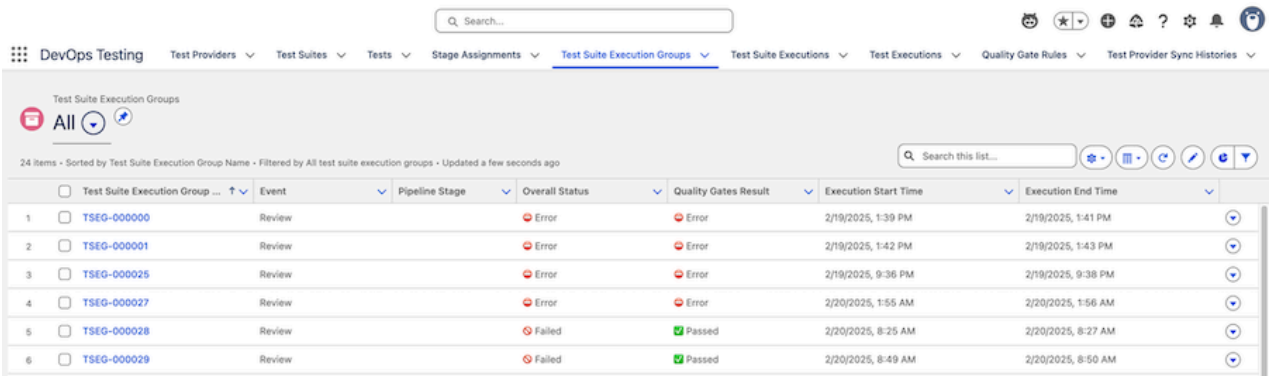
REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

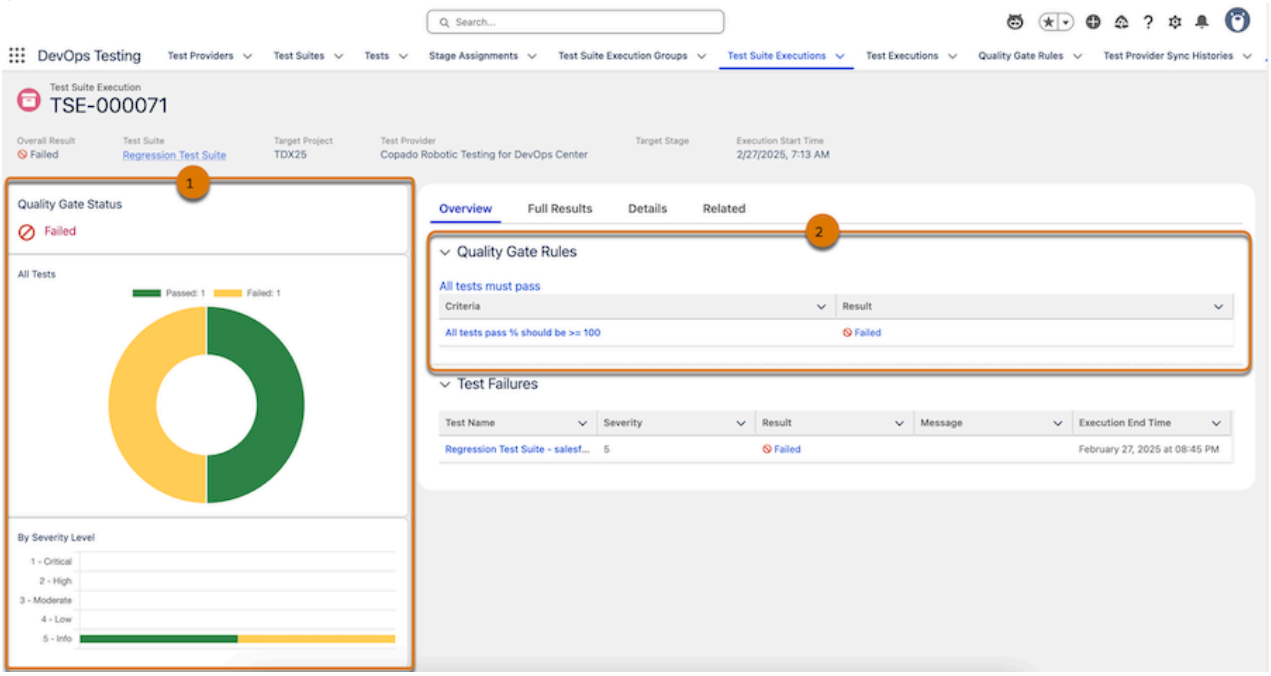
USER PERMISSIONS NEEDED	
To view test results:	DevOps Testing Manager OR DevOps Testing User

When you run test suites manually or automatically, the ideal scenario is that the test suites and quality gate rules pass, and your work item is promoted in DevOps Center. However, if a quality gate rule or test suite run fails, use the Test Suite Execution Groups, Test Suite Executions, and Test Executions tabs to identify and resolve issues.

1. Go to the Test Suite Execution Groups tab.
2. Identify failed test suite execution groups from the Status and Quality Gate Rules Status columns.



3. Select the test suite execution group record that you want to analyze, and then select the **Related** tab. This tab lists all test suites that are part of the test suite execution group.
4. Select the test suite that you want to analyze.
   You're redirected to the test suite executions record page.
5. From the Overview tab, review the Quality Gate Status (1) and Quality Gate Rules (2) to assess quality gate rule and test performance.



6. Analyze the donut chart to determine the overall pass and fail ratio of tests within the test suite, and assess the performance of your tests.
7. Analyze the bar chart to identify the number of tests that passed or failed, categorized by severity level. Use this information to prioritize fixing test failures.
8. Select the **Full Results** tab to view a list of all test results.
   Each test result includes a link to the individual test record page, where you can see detailed test run information.
9. Select the **Details** tab to view additional information about the test suite run, including code coverage (for Apex Unit Testing), pass and fail rate, start and end times, report URL, and result details.
10. Select the **Related** tab to view details such as test names, severity levels, and whether the tests are marked as essential.
11. To identify test-level issues, complete these steps.
    a. Select the test record under the Test Failures section.

You're redirected to the Tests tab.

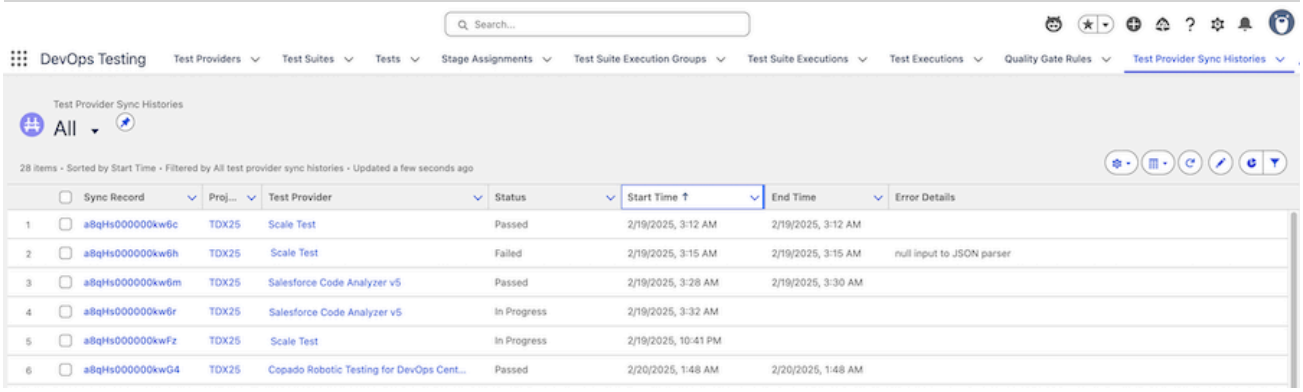
b. Select the **Details** tab to identify the exact point and reason for the test failure.

Test Provider Sync History

See the outcome and details of your test provider sync process in a single, unified summary. It provides the sync status, timestamps, and reasons for any sync failures. Use this information for auditing and troubleshooting issues.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions



Each time the test providers are synced, the details are tracked and logged in the Test Provider Sync Histories tab. If the sync fails, the tab provides a short summary of the cause. You can analyze the error details to quickly identify and address any issues, and then attempt to resync.

View DevOps Testing Events in DevOps Center Activity History

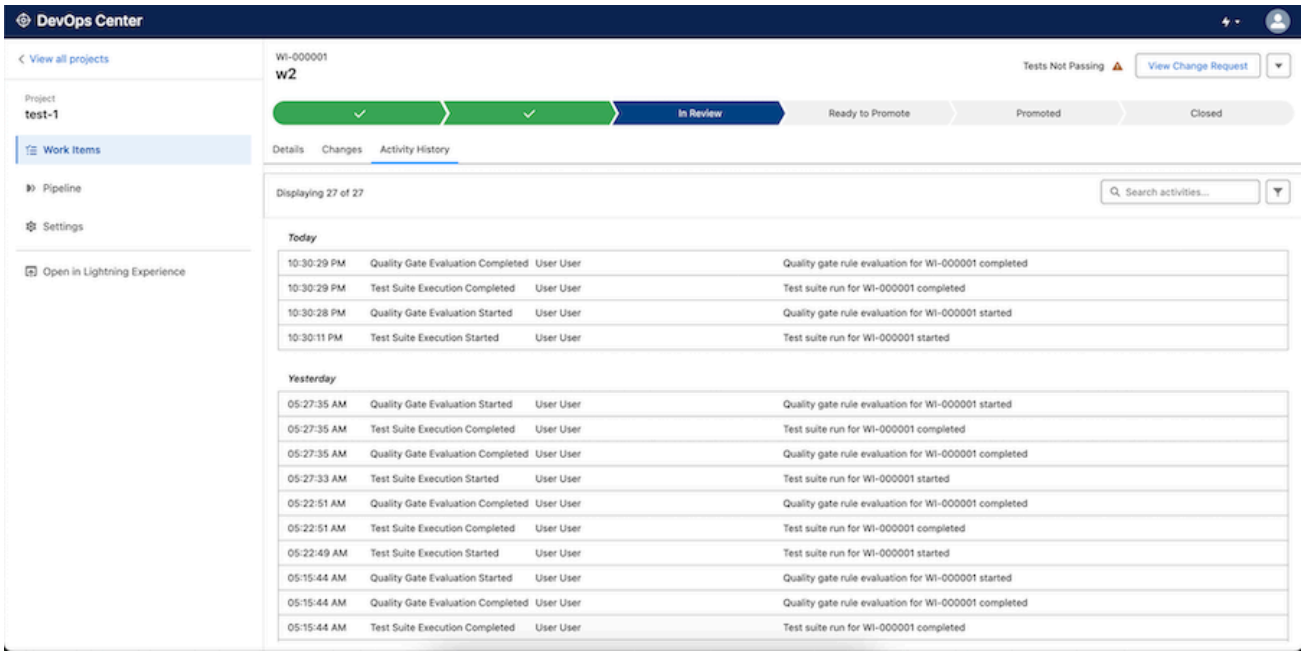
DevOps Center provides a comprehensive history of all key DevOps Testing events, including quality gate rule evaluation, test suite group run, stage assignments, and errors (failures). You can use these events and their associated details for auditing and troubleshooting.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Activity histories can get long, but you can filter or search the list.

- To filter on User, Activity Type, or Date Range, use the filter.
- Use the Search Activities box to search with keywords.



Test Data Retention and Cleanup

In DevOps Testing, the data retention and cleanup processes automatically manage historical test data. A daily cleanup job automatically deletes data older than the set retention period, improving performance and data governance.

REQUIRED EDITIONS

Available in: Lightning Experience

Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

Example A test suite execution record from June 1, 2025, has a 30-day data retention period. The cleanup job runs on July 1, 2025. On this day, the cleanup job determines that the test suite execution record from June 1, 2025, is 30 days old and automatically deletes it.

Considerations for Test Data Cleanup

Review these considerations before you change the test data retention period or create a cleanup job.

Change the Retention Period for Test Data

Change the test data retention period as needed to balance legal compliance, business requirements, and performance. The default retention period is 30 days.

Schedule and Run a New Cleanup Job
Customize the cleanup process to meet your specific needs by scheduling a new cleanup job. Review the details of the default cleanup job and create a cleanup job as needed. When you create a cleanup job, you can retain or delete the default cleanup job.

Considerations for Test Data Cleanup

Review these considerations before you change the test data retention period or create a cleanup job.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

- After you install DevOps Testing, it automatically schedules a default cleanup job. You can also create a custom cleanup job to run on a different schedule.
- The default retention period is 30 days, but you can change this value.
- The retention period calculation is based on the job's execution date, not its scheduling date.
- The daily job runs during periods of low-user activity, typically around 12 AM local time. It takes approximately 20 minutes to complete. We recommend that you don’t sync test data or run test suites during this time.
- The cleanup process applies to the Test Suite Executions, Test Executions, Test Suite Execution Groups, and Sync Histories tabs.
- If associated executions exist for disabled tests and suites, DevOps Testing follows their retention period. If no associated executions exist, DevOps Testing immediately deletes orphaned and disabled tests and test suites.

Change the Retention Period for Test Data

Change the test data retention period as needed to balance legal compliance, business requirements, and performance. The default retention period is 30 days.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To change the retention period:	DevOps Testing Manager

See Considerations for Test Data Cleanup.

1. From Setup, in the Quick Find box, enter *Custom Metadata Types*, and then select **Custom Metadata Types**.
2. On the All Custom Metadata Types page, in the row for the Test Data Retention custom metadata type, click **Manage Records**.
3. In the list of custom metadata records, in the row for the Data Retention Period custom metadata record, click **Edit**.
4. Update the **Days** field and then save your changes.

Schedule and Run a New Cleanup Job

Customize the cleanup process to meet your specific needs by scheduling a new cleanup job. Review the details of the default cleanup job and create a cleanup job as needed. When you create a cleanup job, you can retain or delete the default cleanup job.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Professional** (API access required), **Enterprise**, **Performance**, **Unlimited**, and **Developer** editions

USER PERMISSIONS NEEDED	
To schedule and run a new cleanup job:	DevOps Testing Manager

1. From Setup, in the Quick Find Box, enter *Scheduled Jobs*, and then select **Scheduled Jobs**.
2. On the All Scheduled Jobs page, click **Schedule Apex**.
3. Enter a job name.
4. Click the lookup icon next to the Apex Class field. From the list, select **CleanupService**.
5. Define a schedule to run the job.
6. Save your changes.
   The new cleanup job runs automatically on the scheduled start date.

Monitor your cleanup job to prevent unwanted data loss and maintain system health. To review cleanup jobs, from Setup, in the Quick Find box, enter *Scheduled Jobs*, and then select **Scheduled Jobs**. Review the job details, such as its start date and time and the next scheduled date and time.

Troubleshoot DevOps Testing Issues

Here are some tips on troubleshooting common issues encountered while using DevOps Testing. If needed, get help from your test manager or Salesforce system administrator to resolve these issues.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Issue: Test or test suite run remains in the In Progress status for an extended time, even after the run is complete in the test provider.

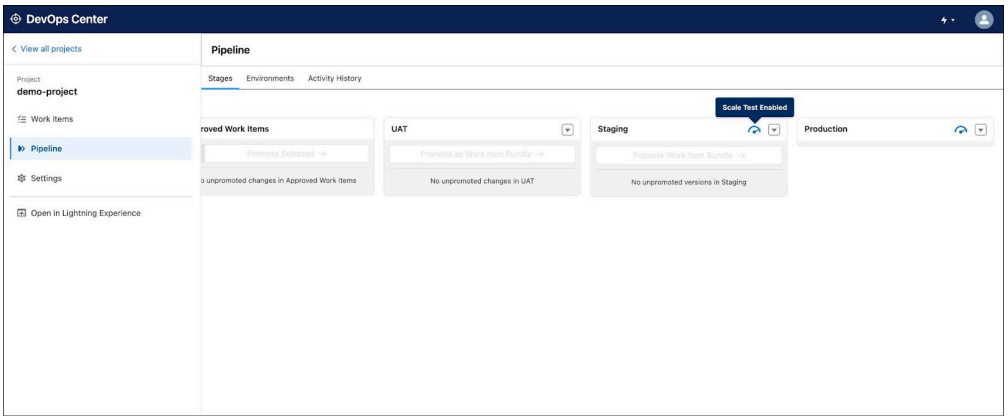
Action: Check for Apex job errors. Go to Setup, and then select **Apex Jobs**. Verify if any Apex jobs are incomplete. If there are no errors or incomplete jobs, request your Salesforce system administrator to manually update the run status in the Test Execution Groups, Test Suite Executions, or Test Executions tab. Rerun the test or test suites if needed.

Issue: Test provider sync remains in the In Progress status for an extended time.

Action: Check for Apex job errors. Go to Setup, and then select **Apex Jobs**. If there are no errors or incomplete jobs, request your Salesforce system administrator to manually update the sync status on the Test Providers tab. Start a new test provider sync if needed.

Issue: Scale Test is configured for a project, but the Scale Test icon isn't visible in the DevOps Center pipeline stage.

Action: Verify that you're logged into the pipeline environment. If not, click the environment links in DevOps Center to log in to the respective environments.



Issue: The Scale Test icon remains disabled even after you add the required license.

Action: This issue happens when the named credential is missing. We recommend that you log in to the environment associated with your DevOps Center project pipeline.

Issue: The create review and promote process fails in DevOps Center.

Action: After you install the DevOps Testing package, make sure that you have the required DevOps Testing permission sets assigned. Also, building and activating a new pipeline in DevOps Center requires

the necessary DevOps Testing permissions sets.

Issue: You update the project or stage names in DevOps Center, but the changes aren't visible in the DevOps Testing Stage Assignments tab.

Action: Make sure that you have the DevOps Testing Manager permission set before retrying to update the project and stage names.

Issue: Scale Test status for a project shows License Required, even though you've a license in the production org of the project pipeline.

Action: Verify that you're logged into the pipeline production environment. If not, use the environment link in DevOps Center to log in.

Issue: Code Analyzer v5 workflow remains in a queued state and doesn't proceed.

Action: This issue occurs if your organization's GitHub environment doesn't support the ubuntu-latest runner. To resolve this, choose one of these options.

- Set up a new GitHub runner with the listed software, and update the runner value in the workflow files.
  - Node version 20.9.0 or higher.
  - Java version 11 or higher.
  - Python version 3.10 or higher.
- Contact your GitHub administrator to enable the ubuntu-latest runner.

DevOps Testing Glossary of Terms

Get familiar with the key terms and concepts related to DevOps Testing.

REQUIRED EDITIONS

Available in: Lightning Experience
Available in: **Enterprise**, **Performance**, **Professional** (API access required), **Unlimited**, and **Developer** Editions

Code Coverage

The percentage of executable lines of code covered by a test suite. DevOps Testing calculates this value by dividing the total number of lines run by the number of lines in the classes covered by the tests, then multiplying the result by 100. For the default test suite, DevOps Testing fetches the org-wide coverage. Use this value to assess the quality of your test suite. See Testing and Code Coverage.

Quality Gate

The checkpoints that ensure only work items that meet quality criteria move to the next pipeline

stage.

Severity

A value ranging from 1 to 5 that helps prioritize failing tests according to their potential impact. DevOps Testing supports these severity levels.

- 1 - Critical
- 2 - High
- 3 - Medium
- 4 - Low
- 5 - Info

Test

A set of conditions under which you determine whether a system or change is working as intended.

Test Execution

The process of running a test that is part of a test suite.

Test Provider

A testing solution to automate and manage testing activities. For example, Copado Robotic Testing, Provar Automation, Salesforce Code Analyzer v5, and so on.

Test Suite

A collection of tests for the same test provider.

Test Suite Execution

The process of running a group of tests assigned to a pipeline stage to verify functionality and identify potential issues. A test suite can be run independently or as part of a test suite group.

Test Suite Execution Group

A collection of test suites run for a particular pipeline stage in the DevOps process.