DevOps Center Setup

Last Updated:2026.02.25

CONTENTS

DevOps Center Setup

A Salesforce admin or user with the appropriate permissions can install DevOps Center. A Salesforce admin or user who is assigned the DevOps Center Manager permission set can configure DevOps Center.

REQUIRED EDITIONS

Available in: Lightning Experience in **Enterprise**, **Performance**, **Professional** (API access required), **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.

USER PERMISSIONS NEEDED	
To install and configure DevOps Center:	"Download AppExchange Packages"

Important You can't install DevOps Center in a sandbox. To install DevOps Center in a Professional Edition org, the org must have API access, which enables DevOps Center to access data from the source control system. If you attempt to install DevOps Center in a Professional Edition org without API access, an installation error occurs. Contact your Account Executive to request the API add-on.

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

Install the DevOps Center package in a supported org edition from the DevOps Center Setup page. You can install DevOps Center in the org that is the final release org in your pipeline, or in another org that's unrelated to the orgs in the pipeline. After you install it, you connect DevOps Center to any environments that you use for development, testing, and final release.

As part of the configuration process, you add DevOps Center users in the org in which it's installed. You can provide minimum access to the org by assigning users either the Identity Only or Salesforce Limited Access - Free license. Then you assign users to DevOps Center permission sets that determine what tasks they can perform in the DevOps Center app.

Why Can't I Install DevOps Center in a Sandbox?

DevOps Center is a managed package that is installed in an org that acts as a "hub" for the application. From within DevOps Center, you connect to your development orgs, pipeline environments, and release orgs. DevOps Center doesn't interact with the existing data in the org, yet generates data in the form of custom object records for the custom objects delivered with the package.

We don't provide the ability to install DevOps Center in sandboxes because:

- Sandbox refreshes result in the loss of DevOps Center projects and their associated records. All your DevOps Center data will be gone.
- It's designed so that all team members log in to the same org to access DevOps Center. Cloning a sandbox would result in multiple instances of the managed package, which can lead to confusion and synchronization issues for teams expecting to be working against the same projects.
- Unlike some other packages, DevOps Center has no interaction with existing data or metadata in your org.

To evaluate DevOps Center, enable and install it in a Developer Edition org or a scratch org. However, remember that scratch orgs expire.

What Source Control Repositories Can I Use with DevOps Center?

Although we've built a framework for DevOps Center to eventually integrate with multiple third-party source control systems, we now support these source control systems:

- GitHub.com Cloud-based plans, including GitHub Enterprise Cloud. Locally hosted versions of GitHub, including GitHub Enterprise Server, aren't currently supported. See Set Up DevOps Center for GitHub to get started.
- Bitbucket Cloud (beta). See Set Up DevOps Center for Bitbucket to get started.

Provide General Feedback

To provide general feedback, request product enhancements, start discussions with other DevOps Center users or the product team, and share recommended practices, use the DevOps Center Trailblazer group.

DevOps Center Releases

Based on your feedback and our drive for innovation, the DevOps Center team releases new features, product enhancements, and bug fixes on a regular basis. If you want to know all the goodness included in each package version, you've come to the right place.

Evaluate DevOps Center in a Non-Production Org

So you want to evaluate DevOps Center, but prefer to install it in an org other than your production org. Have no fear. You have options. You can completely evaluate DevOps Center using a combination of Developer Edition (DE) orgs and scratch orgs.

Enable and Install DevOps Center in the Org

Enable DevOps Center to provide the permissions needed to install the DevOps Center package in the org. You can install DevOps Center in the org that is the final release org in your pipeline, or in another org that is unrelated to the orgs in the pipeline.

Upgrade to the Latest DevOps Center Version

In most cases, you don't have to do anything to upgrade to the latest version of the DevOps Center. When we release a new package version, we update your org automatically.

Set Up DevOps Center for GitHub

To use GitHub with DevOps Center as your source control repository, some tasks are performed within GitHub, such as creating a repository and adding team members. And some tasks are performed in DevOps Center, such as setting up the associated DevOps Center project, adding development and pipeline environments, configuring your pipeline, and creating and assigning work items to team members.

Set Up DevOps Center for Bitbucket (Beta)

Set up DevOps Center to use a Bitbucket source control repository. Currently, we support only Bitbucket Cloud.

Manage DevOps Center Projects

Your team's central arena for work in DevOps Center is the project. The purpose of a project is to help you and your team manage changes being developed for a particular application.

Complete Optional Setup Tasks

Complete these optional setup tasks, as needed.

Manage Environments

When first configuring DevOps Center, identify the environments that you're going to use for each project. After environments are defined in the Pipeline Environments tab, you can manage each environment from its drop-down menu.

Plan Your Pipeline

A pipeline defines the sequence of stages that work items progress as they go through the release lifecycle from development through to production (or some other final release stage).

Create and Assign Project Work Items

Create work items so that when your team members open DevOps Center for the first time, project work is already identified and assigned to them.

Build an Extension Package for DevOps Center

Salesforce partners and ISVs can use scratch orgs as their dev environments when building managed second-generation (2GP) extension packages that expand the functionality of DevOps Center. An extension is any package, component, or set of components that adds to the functionality of a managed package. An extension requires that the base managed package is installed in the org.

Uninstall DevOps Center

You can uninstall DevOps Center, if necessary. If you plan to reinstall DevOps Center, don't delete the DevOps Center auth providers. However, if you deleted the auth providers, you can recreate them by re-enabling the DevOps Center preference in Setup.

Troubleshoot DevOps Center Configuration

Here are some tips if you encounter issues when installing or configuring DevOps Center.

See Also

Salesforce Help: Manage and Release Changes Easily and Collaboratively with DevOps Center
DevOps Center Developer Guide
Uninstall DevOps Center
Trailhead: Git and GitHub Basics

DevOps Center Releases

Based on your feedback and our drive for innovation, the DevOps Center team releases new features, product enhancements, and bug fixes on a regular basis. If you want to know all the goodness included in each package version, you've come to the right place.

Package Version 11.0.0

Release date: June 25, 2025

This release contains service updates, bug fixes, and performance improvements.

Release notes: DevOps Center Roadmap: v11.0 (June 2025)

Package Version 10.1.0

Release date: March 25, 2025

This release contains service updates, bug fixes, and performance improvements.

Release notes: DevOps Center Roadmap: v10.1 (March 2025)

Package Version 9.1.0

Release date: February 21, 2025

This release contains service updates, bug fixes, and performance improvements.

Release notes: DevOps Center Roadmap: v9.1 (February 2025)

Package Version 8.2.0

Release date: November 20, 2024

This release includes changes to support DevOps Testing pilot, under-the-hood bug fixes, and performance improvements. This release is required if you're part of the DevOps Testing pilot.

Release notes: DevOps Center Roadmap: v8.2 (November 2024)

Package Version 7.8.0

Release date: October 24, 2024

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v7.8 (October 2024)

Package Version 7.7.0

Release date: October 8, 2024

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v7.7 (October 2024)

Package Version 7.6.0

Release date: September 13, 2024

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v7.6 (September 2024)

Package Version 7.5.0

Release date: September 3, 2024

This release contains feature enhancements, performance improvements, and bug fixes.

Release notes for new features: Winter ’25 Release Notes (DevOps Center)

Release notes for bug fixes: DevOps Center Roadmap v7.5 (September 2024)

Package Version 6.4.0

Release date: July 22, 2024

This release contains feature enhancements, performance improvements, and bug fixes.

Release notes: DevOps Center Roadmap: v6.4 (July 2024)

Service Update

Release date: March 2024

This release contains service updates.

Release notes for updates: Update backend library versions

Service Update

Release date: February 2024

This release contains service updates.

Release notes for updates: Use API version from sfdx-project.json

Package Version 6.3.0

Release date: November 30, 2023

This release contains new features, performance improvements, and bug fixes.

Release notes for new features: Winter ’24 Release Notes (DevOps Center)

Release notes for bug fixes: DevOps Center Roadmap: v6.3 (November 2023)

Package Version 6.1.0

Release date: October 12, 2023

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v6.1 (October 2023)

Package Version 6.0.0

Release date: August 31, 2023

This release contains new features, performance improvements, and bug fixes.

Release notes for new features: Winter ’24 Release Notes (DevOps Center)

Release notes for bug fixes: DevOps Center Roadmap: v6.0 (August 2023)

Package Version 5.8.0

Release date: July 27, 2023

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v5.8 (July 2023)

Package Version 5.7.0

Release date: June 15, 2023

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v5.7 (June 2023)

Package Version 5.5.0

Release date: April 11, 2023

This release contains performance improvements and bug fixes.

Release notes: DevOps Center Roadmap: v5.5 (April 2023)

Package Version 5.3.0

Release date: December 9, 2022

DevOps Center is generally available, and includes some significant new features and product enhancements since we released the beta version.

Release notes for new features: Spring '23 Release Notes (DevOps Center)

See Also

DevOps Center Roadmap: Tell Us What's Important to You

Evaluate DevOps Center in a Non-Production Org

So you want to evaluate DevOps Center, but prefer to install it in an org other than your production org. Have no fear. You have options. You can completely evaluate DevOps Center using a combination of Developer Edition (DE) orgs and scratch orgs.

We support these configurations; they're listed in the order in which we recommend them.

- Developer Edition orgs for your installation, release, and other pipeline environments, and a source-tracked Developer or Developer Pro sandbox for your development environment.
- Developer Edition orgs for your installation, release, and other pipeline environments, and a scratch org for your development environment. You can enable Dev Hub in the Developer Edition installation org so you can create scratch orgs.
- Developer Edition org as your Dev Hub, and scratch orgs for all your environments.

Not all combinations are listed here. This information provides you with a basis to determine which options work for your particular situation.

If you use a Developer Edition org as your Dev Hub to create scratch orgs, you're limited to 3 active scratch orgs per day. If you already have Dev Hub enabled in your production org with plenty of scratch orgs to spare, we recommend using it instead of a Developer Edition org as your Dev Hub org.

Note You can also use Trailhead Playgrounds for the installation, release, and pipeline environments. A Trailhead Playground is a Developer Edition org with Trailhead-specific data and tools.

When determining which type of org to use, think about how long you need for the evaluation period.

Developer Edition Orgs

- You can sign up for as many Developer Edition orgs as you like.
- They don't expire as long as you log in to them regularly.
- You can install DevOps Center in them.
- You can create as many DE orgs as required to mimic your release pipeline.
- You can create a DE org for the installation org, release environment, and pipeline environments.
- You can enable Dev Hub in a DE org if you plan to use scratch orgs.

Developer or Developer Pro Source-Tracked Sandboxes

- If you have a spare source-tracked developer sandbox that you can use as your development environment, this option is the most desirable choice.
- The sandbox already mirrors your release environment.
- Sandboxes don't expire as long as you log in to them regularly.

Note Sandboxes can't be used as the installation org. See Why Can't I Install DevOps Center in Sandbox? for more information.

Scratch Orgs

- You can install DevOps Center in them.
- Scratch orgs have a maximum duration of 30 days.
- A Dev Hub is required to create scratch orgs. The Dev Hub edition determines the maximum number of active scratch orgs.
- Scratch orgs have source-tracking enabled by default so you can use them as development or pipeline environments.
- You create scratch orgs using Salesforce CLI. After installing and configuring it, you can create scratch orgs quickly.

Install DevOps Center in a Developer Edition Org

To evaluate DevOps Center, you can install it in a Developer Edition (DE) org. If you don't see DevOps Center as an option under Setup, your Developer Edition org was likely created before DevOps Center was introduced and doesn't have the correct permissions. To continue, create another Developer

Edition org.

Install DevOps Center in a Scratch Org
To evaluate DevOps Center, you can install it in a scratch org. Scratch orgs have a maximum duration of 30 days.

See Also

Salesforce CLI Setup Guide: Install Salesforce CLI
Salesforce DX Developer Guide: Enable Dev Hub Features in Your Org
Salesforce DX Developer Guide: Scratch Orgs
Salesforce DX Developer Guide: Enable Source Tracking in Sandboxes

Install DevOps Center in a Developer Edition Org

To evaluate DevOps Center, you can install it in a Developer Edition (DE) org. If you don't see DevOps Center as an option under Setup, your Developer Edition org was likely created before DevOps Center was introduced and doesn't have the correct permissions. To continue, create another Developer Edition org.

1. Sign up for a Developer Edition org.
2. Enable and install DevOps Center in the org.
3. Complete the DevOps Center app configuration.

Next, create more Developer Edition orgs as pipeline environments, and decide if you're going to use a sandbox or scratch org as your development environment.

Although you can use a Developer Edition org as your permanent installation org, we don't recommend it because DE orgs have very low limits that can affect the long-term use of the DevOps Center.

See Also

Install DevOps Center in a Scratch Org

Install DevOps Center in a Scratch Org

To evaluate DevOps Center, you can install it in a scratch org. Scratch orgs have a maximum duration of 30 days.

- Install Salesforce CLI. See Install Salesforce CLI in the *Salesforce CLI Setup Guide*.
- Enable Dev Hub in an org. If you don't already have a Dev Hub org with larger scratch org allocations, you can use a Developer Edition org, which provides a maximum of 3 active scratch orgs. See Enable Dev Hub Features in Your Org in the *Salesforce DX Developer Guide*.

If you already use scratch orgs and have Salesforce CLI installed, skip to the information regarding the scratch org definition file.

1. Enable DevOps Center in the Dev Hub org without installing it in the org, which verifies that you acknowledge the terms and conditions.

You can disable DevOps Center after you complete your evaluation.

2. Create a Salesforce DX project.

   If you don't currently have a source control repository with a Salesforce DX project, you can create an empty DX project as a starting point. See Create a Salesforce DX Project in the *Salesforce DX Developer Guide*.

3. Update the scratch org definition file, `sfdx-project.json`, to enable DevOps Center.

```
{
    "orgName": "Acme",
    "edition": "Enterprise",
    "features": ["DevOpsCenter"],
    "settings": {
        "devHubSettings": {
            "enableDevOpsCenterGA": true
            }
        }
    }
```

   If creating a scratch org based on an org shape, you still have to include the DevOps Center feature and setting in the scratch org definition file for legal reasons as part of the DevOps Center terms and conditions.

```
    "orgName": "Acme",
    "sourceOrg": "00DB1230400Ifx5",
    "features": ["DevOpsCenter"],
    "settings": {
        "devHubSettings": {
            "enableDevOpsCenterGA": true
            }
        }
    }
```

4. Log in to the Dev Hub org using Salesforce CLI.

   Logging into the Dev Hub org authorizes Salesforce CLI to run commands that require it. In this example, the Dev Hub org is given the alias `MyDevHub`.

```
sf org login web -a MyDevHub
```

5. Create a scratch org using Salesforce CLI, referencing the scratch org definition that you previously updated.

For example, this scratch org is created with the alias `devops-center-scratch`, with a duration of 30 days, using the Dev Hub org with the alias `MyDevHub`:

```
sf org create scratch -a devops-center-scratch -f config/project-scratch-def.json -v MyDevHub -d 30
```

6. After you create a scratch org in which to install DevOps Center, enable and install DevOps Center in the org, then complete the DevOps Center app configuration.

See Also

Salesforce DX Developer Guide: Build Your Own Scratch Org Definition File
Salesforce CLI Reference Guide: sf org create scratch
Install DevOps Center in a Developer Edition Org

Enable and Install DevOps Center in the Org

Enable DevOps Center to provide the permissions needed to install the DevOps Center package in the org. You can install DevOps Center in the org that is the final release org in your pipeline, or in another org that is unrelated to the orgs in the pipeline.

REQUIRED EDITIONS

Available in: Lightning Experience in **Enterprise**, **Performance**, **Professional** (API access required), **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.

Important You can't install DevOps Center in a sandbox.To install DevOps Center in a Professional Edition org, the org must have API access, which enables DevOps Center to access data from the source control system. If you attempt to install DevOps Center in a Professional Edition org without API access, an installation error occurs. Contact your Account Executive to request the API add-on.

USER PERMISSIONS NEEDED	
To install and configure DevOps Center:	"Download AppExchange Packages"

1. From Setup, enter *DevOps Center* in the Quick Find box, then select **DevOps Center**.

   If you don't see DevOps Center as an option under Setup in your Developer Edition org, it's likely that the DE org was created before DevOps Center was introduced and doesn't have the correct permissions. To continue, create another Developer Edition org.

2. Enable DevOps Center, and then review and accept the opt-in terms.

   You can disable the preference at any time. Disabling the preference means that you can't install the app or updates. If DevOps Center is already installed, disabling the preference prevents access to the DevOps Center application.

3. Click **Install Package**.

   After the installer launches, you're guided through the installation process to install the latest version of the DevOps Center managed package. You can come back to this Setup page to reinstall or upgrade the package.

4. Select **Install for Admins Only**, and then click **Install**.
5. Approve third-party access to `login.salesforce.com` and `test.salesforce.com`.

Next, create an external client app so that DevOps Center appears in App Launcher, then be sure to assign the appropriate DevOps Center permission sets to each team member.

Confirm DevOps Center Package Installation

When installation is complete, you receive a confirmation email. You can confirm the installation on the Installed Packages Setup page.

Create an External Client App

External client apps provide single sign-on (SSO) and use OAuth protocols to authorize third-party applications. When you configure the app and assign appropriate permission sets, DevOps Center appears in the App Launcher.

Add Team Members as Users in the DevOps Center Org

Add any team members who aren't already users in the DevOps Center org. For each team member, specify the appropriate license and profile based on their role.

Assign the DevOps Center Permission Sets

Assign permission sets to everyone working on your project in DevOps Center. Consider who must change project-level settings (such as adding another work environment) and who needs access to only work items.

Configure Session Settings

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, you must disable the session setting, Lock sessions to the IP address from which they originated, to avoid seeing errors when attempting to commit and promote changes.

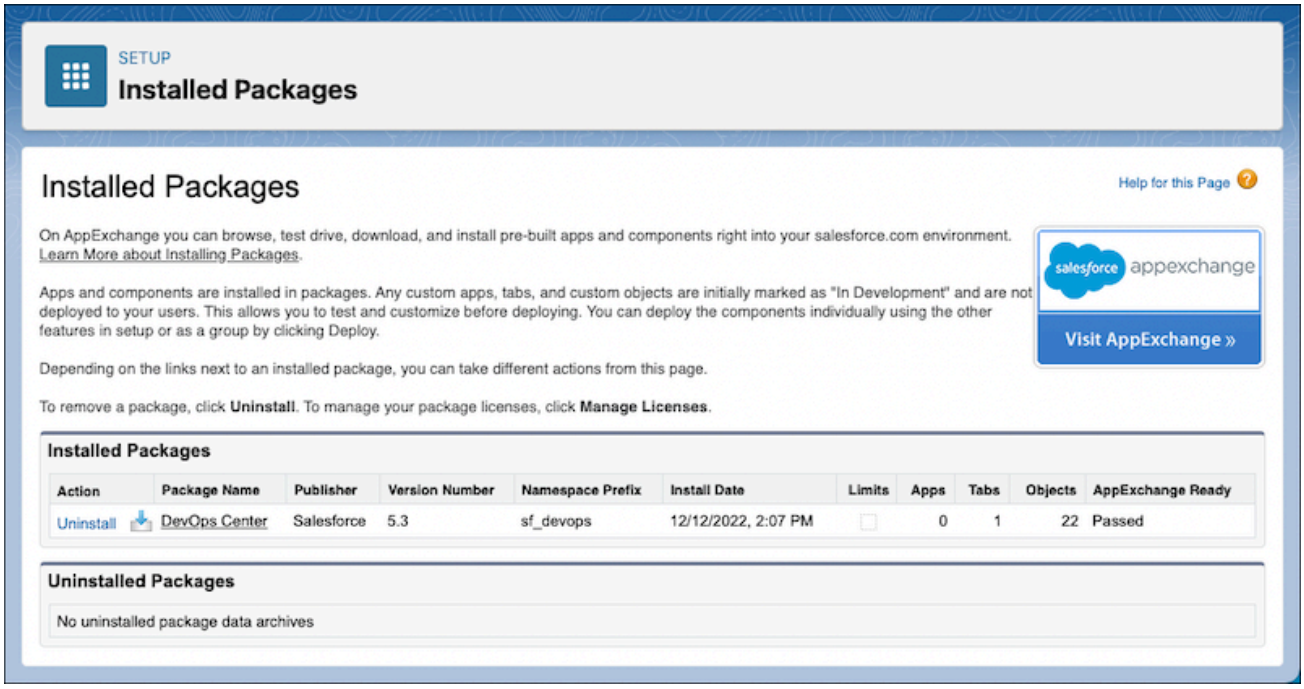
Disable API Access Control Setting for Allowlisted Connected Apps

For orgs with API Access Control settings enabled by Salesforce Support, you must disable the "For admin-approved users, limit API access to only allowlisted connected apps" because this setting blocks DevOps Center from interacting with required APIs.

Confirm DevOps Center Package Installation

When installation is complete, you receive a confirmation email. You can confirm the installation on the Installed Packages Setup page.

From Setup, enter *Installed Packages* in the Quick Find box, then select **Installed Packages**. You see an entry for DevOps Center that looks something like this. If you get the message that the app is taking a long time to install, you get automatically redirected to this page after you click **Done**.



Create an External Client App

External client apps provide single sign-on (SSO) and use OAuth protocols to authorize third-party applications. When you configure the app and assign appropriate permission sets, DevOps Center

appears in the App Launcher.

REQUIRED EDITIONS

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

USER PERMISSIONS NEEDED	
To create local external client apps:	Create, edit, and delete External Client Apps

Before you start, identify and save your DevOps Center URL. Go to My Domain in Setup. Copy the Current My Domain URL value under My Domain Details. Replace the <my_url> placeholder with this value to format your custom URL: `https://<my_domain>/sf_devops/DevOpsCenter.app`.

Note Starting with the 260 release, DevOps Center is transitioning from connected apps to external client apps.

1. From Setup, in the Quick Find box, enter *External Client Apps*, and then select **External Client App Manager**.
2. Click **New External Client App**.
3. On the Settings tab, enter these details in the Basic Information section.
   - External Client App Name: DevOps Center
   - API Name: DevOps_Center
   - Contact Email: support@salesforce.com
   - Distribution State: Local
4. Set any other parameters based on your business requirements, and save your changes.
5. On the Policies tab, click **Edit**, and then configure the App Policies section.
6. In the Start Page field, select **Custom**.
7. In the Custom Start URL field, enter the DevOps Center URL that you saved.
8. Save your changes.

See Also

Salesforce Help: External Client Apps

Add Team Members as Users in the DevOps Center Org

Add any team members who aren't already users in the DevOps Center org. For each team member, specify the appropriate license and profile based on their role.

The listed licenses and profiles are the minimum required to use DevOps Center for the associated roles. Users can also have a more fully featured license or profile that provides them access beyond the minimum requirements for DevOps Center. However, be aware of what access you are delegating to your users in your org.

Minimum Required Licenses and Profiles

Role	License	Profile
Project Manager, Release Manager (anyone who manages and add environments to DevOps Center)	Salesforce	Standard User
Team members	Salesforce Limited Access - Free	Limited Access User

Tip This procedure generates an email inviting the new users into the org. But until you're finished setting up DevOps Center, there's not much for them to do in the org. We recommend that you let your team know that you're setting up DevOps Center and to wait until they hear from you before logging in.

1. Log in to the DevOps Center org.
2. From Setup, enter `Users` in the Quick Find box, 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 the **Generate passwords and notify user via email** checkbox.
6. Click **Save**.

See Also

Salesforce Help: View and Manage Users
Salesforce Help: Standard Profiles
Salesforce Help: User Licenses
Salesforce DX Developer Guide: Free Limited Access License

Assign the DevOps Center Permission Sets

Assign permission sets to everyone working on your project in DevOps Center. Consider who must change project-level settings (such as adding another work environment) and who needs access to only work items.

Important If you create your own custom permission sets, make sure that they mirror the access of the required DevOps Center permission sets. As we add more features to DevOps Center with each new package version, the access that our permission sets grant is likely to change over time.

Permission Set	Description
DevOps Center	The base permission set for DevOps Center. Provides the data access and permissions needed to manage customizations for DevOps Center work items. Ability to view all connected

Permission Set	Description
	environments and pipelines. Assign to: all DevOps Center users, including team members also assigned the DevOps Center Manager permission set. The two permission sets don't overlap.
DevOps Center Manager	Provides the data access and permissions needed to set up DevOps Center projects, environments, and users. Assign to: team/project managers
DevOps Center Release Manager	Provides permissions to perform promotions through the pipeline, including deployments using Salesforce CLI. Assign to: release manager and any team members who promote changes through the pipeline
sf_devops_InitializeEnvironments	Allows managers of DevOps Center projects to manage the connections to work environments. Includes the Modify Metadata Through Metadata API Functions and Customize Application user permissions, so the manager can create NamedCredential records. Assign to: team/project managers
sf_devops_NamedCredentials	Grants access to the named credentials needed to authenticate to environments. Created and maintained automatically by DevOps Center. Assign to: all DevOps Center users

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

6. Repeat the procedure to assign the sf_devops_NamedCredentials permission set to your team members.
7. Add the DevOps Center Manager and DevOps Center Release Manager permission sets to team members who need permissions to configure projects, build pipelines, and promote changes through the pipeline.

See Also

Salesforce Help: Permission Sets
Salesforce Help: Named Credentials

Configure Session Settings

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, you must disable the session setting, Lock sessions to the IP address from which they originated, to avoid seeing errors when attempting to commit and promote changes.

1. From Setup, in the Quick Find box, enter *`Session Settings`*, then select **Session Settings**.
2. Deselect **Lock sessions to the IP address from which they originated**.

Disable API Access Control Setting for Allowlisted Connected Apps

For orgs with API Access Control settings enabled by Salesforce Support, you must disable the "For admin-approved users, limit API access to only allowlisted connected apps" because this setting blocks DevOps Center from interacting with required APIs.

If you have this setting enabled, the behavior you see depends on if it was enabled before or after you installed DevOps Center.

- If this setting was enabled before DevOps Center was installed, you can't add a release environment to a project.
- If this setting was enabled after DevOps Center was installed, you can't perform common operations, such as commits and promotions.

1. From Setup, in the Quick Find box, enter *`API Access Control`*, then select **API Access Control**.
2. Deselect **For admin-approved users, limit API access to only allowlisted connected apps**.

Next, continue configuring DevOps Center.

Upgrade to the Latest DevOps Center Version

In most cases, you don't have to do anything to upgrade to the latest version of the DevOps Center. When we release a new package version, we update your org automatically.

However, sometimes the push upgrade doesn't work. If you don't have the latest version, we recommend that you manually install the package through the DevOps Center Setup page. If a newer version doesn't exist, the package installer informs you that you already have the latest version.

The first time a project is opened after the new package version has been installed, a migration process runs automatically to update the project data to be compatible with the new version. This migration process requires that the user is a collaborator on the project's repository.

Missing Buttons in Page Layouts

Sometimes after an upgrade, some buttons aren't appearing in some page layouts. These buttons are important when configuring and using DevOps Center. If you find that these buttons are missing, you can easily add them to the affected page layouts.

- View in DevOps Center button in Project layout
- View in DevOps Center button in Work Item layout
- Connect to Repository button in VCS layout

Add View in DevOps Center Button to Project Layout

If you upgraded from a previous version of DevOps Center, the View in DevOps Center button doesn't always show up in the Project page layout. If this navigational button is missing, add it back so you can easily navigate between the Lightning Experience record pages and the DevOps Center app when creating projects. You can alternatively access the DevOps Center app using App Launcher.

Add View in DevOps Center Button to Work Item Layout

If you upgraded from a previous version of DevOps Center, the View in DevOps Center button doesn't always show up in the Work Item page layout. If this navigational button is missing, add it back so you can easily navigate between the Lightning Experience record pages and the DevOps Center app when creating work items. You can alternatively access the DevOps Center app using App Launcher.

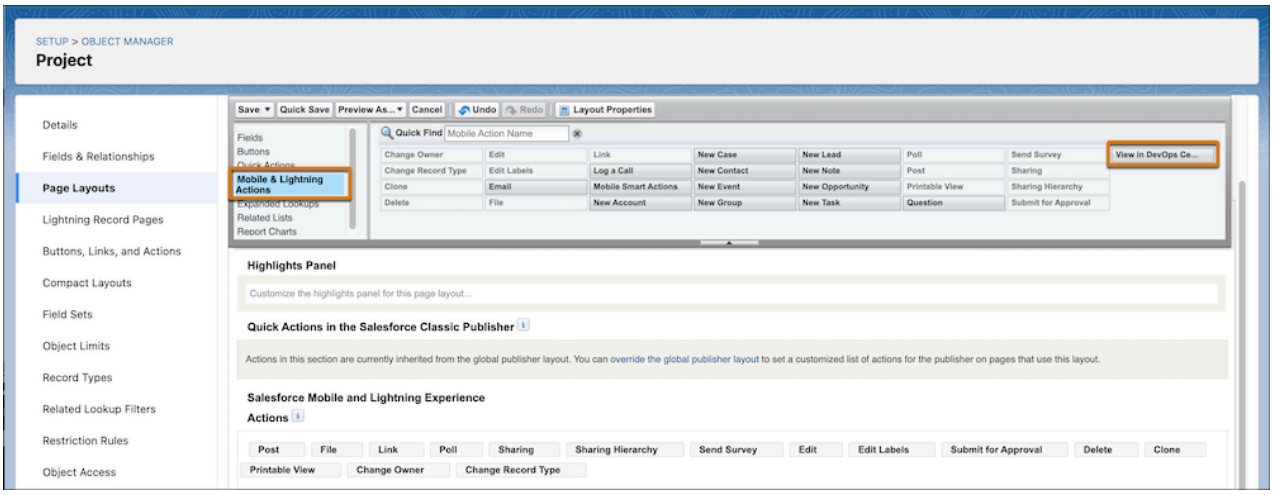
Add Connect Repository Button to VCS Layout

If you upgraded from a previous version of DevOps Center, the Connect Repository button doesn't always show up in the Source Control System (VCS) page layout. This button is required when initially setting up a source control system vendor, such as Bitbucket.

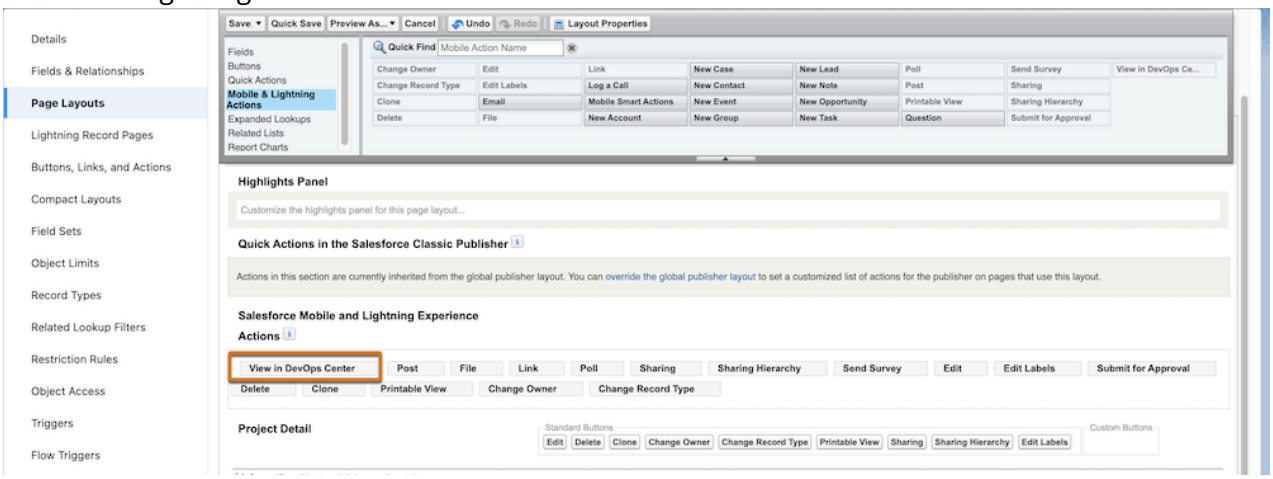
Add View in DevOps Center Button to Project Layout

If you upgraded from a previous version of DevOps Center, the View in DevOps Center button doesn't always show up in the Project page layout. If this navigational button is missing, add it back so you can easily navigate between the Lightning Experience record pages and the DevOps Center app when creating projects. You can alternatively access the DevOps Center app using App Launcher.

1. In Setup, find and select **Object Manager**.
2. Find and click **Project** (sf_devops_Project__c).
3. From Page Layouts, click **Project Layout**.
4. From the side panel, click **Mobile & Lightning Actions**, then locate the View in DevOps Center button tile.



5. Under Salesforce Mobile and Lightning Experience Actions, drop the View in DevOps Center button tile at the beginning of the list.



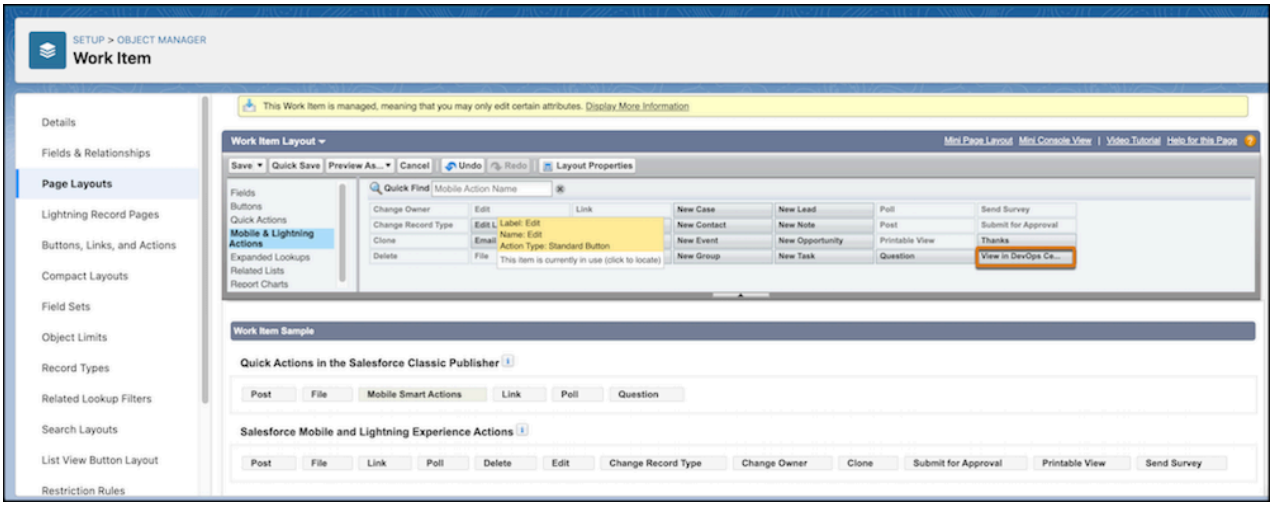
If you don't see a list or the red No Symbol icon appears when you attempt to drag the tile, click the **Override the Predefined Actions** link to make the section editable. The text link is "override the global publisher layout" when the page layout is editable.

6. Save your changes.

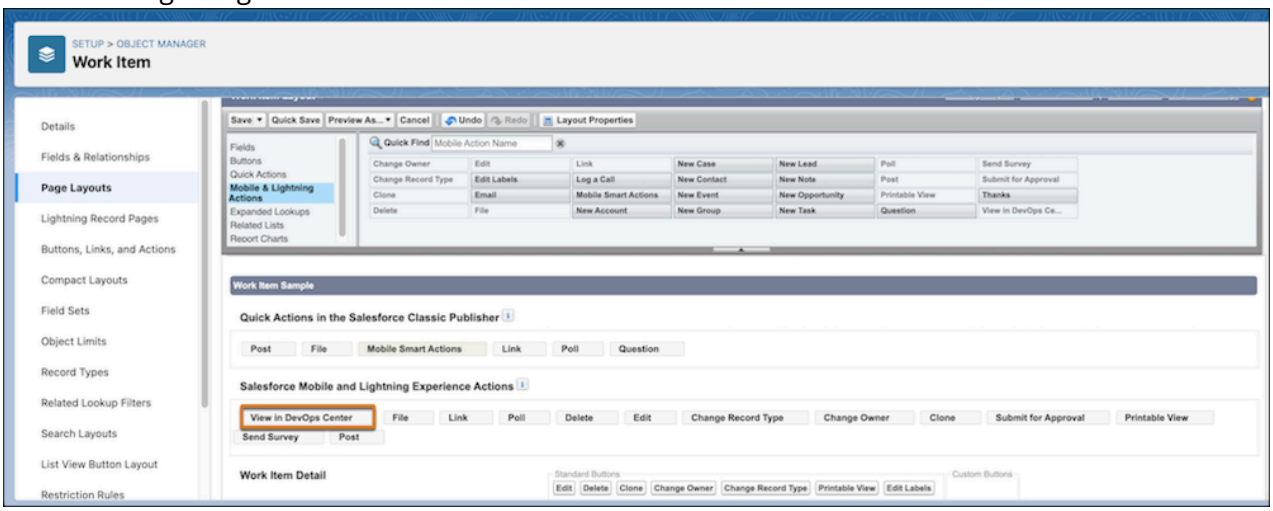
Add View in DevOps Center Button to Work Item Layout

If you upgraded from a previous version of DevOps Center, the View in DevOps Center button doesn't always show up in the Work Item page layout. If this navigational button is missing, add it back so you can easily navigate between the Lightning Experience record pages and the DevOps Center app when creating work items. You can alternatively access the DevOps Center app using App Launcher.

1. In Setup, find and select **Object Manager**.
2. Find and click **Work Item** (sf_devops_Work_Item__c).
3. From Page Layouts, click **Work Item Layout**.
4. From the side panel, click **Mobile & Lightning Actions**, then locate the View in DevOps Center button tile.



5. Under Salesforce Mobile and Lightning Experience Actions, drop the View in DevOps Center button tile at the beginning of the list.

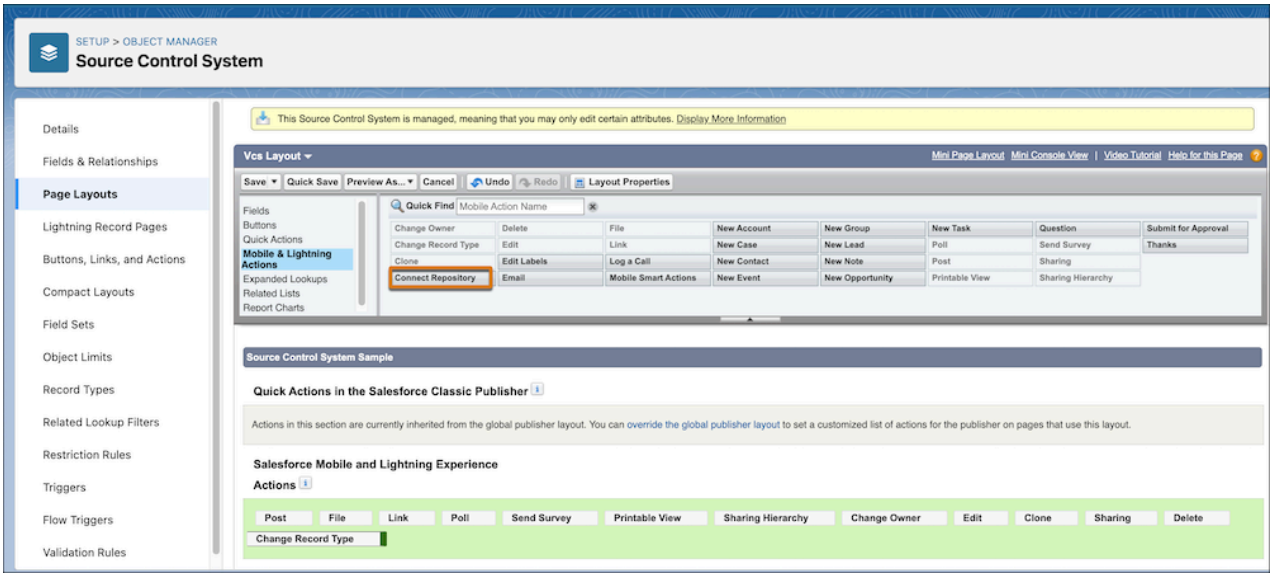


6. Save your changes.

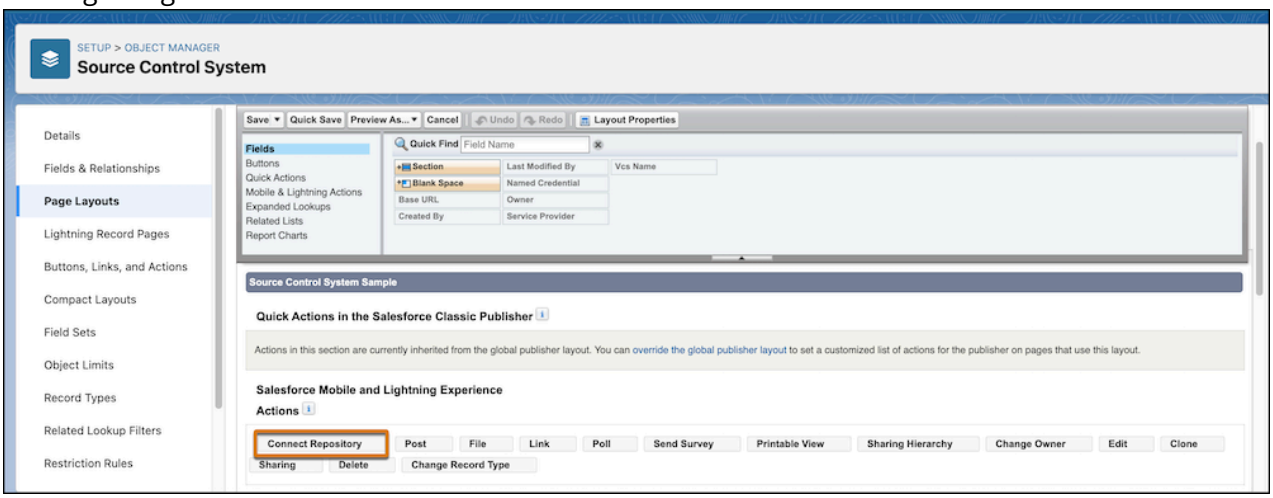
Add Connect Repository Button to VCS Layout

If you upgraded from a previous version of DevOps Center, the Connect Repository button doesn't always show up in the Source Control System (VCS) page layout. This button is required when initially setting up a source control system vendor, such as Bitbucket.

1. In Setup, find and select **Object Manager**.
2. Find and click **Source Control System** (sf_devops_Vcs__c).
3. From Page Layouts, click **Vcs Layout**.
4. From the side panel, click **Mobile & Lightning Actions**, then locate the Connect Repository button tile.



5. Under Salesforce Mobile and Lightning Experience Actions, drop the Connect Repository button tile at the beginning of the list.



If you don’t see a list or the red No Symbol icon appears when you attempt to drag the tile, click the **Override the Predefined Actions** link to make the section editable. The text link is “override the global publisher layout” when the page layout is editable.

6. Save your changes.

Set Up DevOps Center for GitHub

To use GitHub with DevOps Center as your source control repository, some tasks are performed within GitHub, such as creating a repository and adding team members. And some tasks are performed in DevOps Center, such as setting up the associated DevOps Center project, adding development and pipeline environments, configuring your pipeline, and creating and assigning work items to team members.

DevOps Center can be used with only GitHub.com Cloud-based plans, including GitHub Enterprise

Cloud.

Here's a sneak-peak at the overall process:

- Set up a GitHub project repository and add team members as collaborators.
- Identify your development and pipeline environments.
- Create a GitHub Project.
- Plan, build, and activate your pipeline.
- Create and assign project work items.
- Complete any optional setup tasks.

Set Up Your GitHub Project Repositories

A GitHub repository (sometimes called a repo for short) stores project work files – code, text, images, and so on. Each DevOps Center project needs its own repository for storing project changes. While you're working on the project, the repository is the team's centralized source of truth to manage changes.

Add Team Members to the GitHub Repo As Collaborators

In GitHub, add team members as repository collaborators. A collaborator is someone who has been granted write access to the project repository. Add everyone who creates customizations or code for your project to the project repository as a collaborator. That way, they have access to the repository through DevOps Center.

If an Organization Owns the GitHub Repo

GitHub repos owned by an organization aren't visible in DevOps Center until an organization account owner provides access.

If the GitHub Organization Has an IP Allowlist

If you have restricted access to the GitHub organization assets by configuring an allowlist, add the domains for the DevOps Center app.

Connect to a Different GitHub Account

When creating or opening a project, all team members log in to GitHub using the account that has access to the repos that your team wants to use with DevOps Center. If you or a team member logged in with the incorrect GitHub account, a team member with Salesforce admin privileges can delete the authorization information so you or the team member can log in using the proper credentials.

Set Up Your GitHub Project Repositories

A GitHub repository (sometimes called a repo for short) stores project work files – code, text, images, and so on. Each DevOps Center project needs its own repository for storing project changes. While you're working on the project, the repository is the team's centralized source of truth to manage changes.

DevOps Center can be used only with GitHub.com Cloud-based plans, including GitHub Enterprise Cloud. All users must have their own cloud-based GitHub.com account to work in this version of DevOps Center. Locally hosted versions of GitHub, including GitHub Enterprise Server, aren't currently supported. We've designed DevOps Center to eventually integrate with multiple third-party source control systems, such as Bitbucket (in beta), GitLab, and other GitHub versions.

Communication Protocols

DevOps Center uses the GitHub REST API for all communications to and from GitHub.

DevOps Center uses the OAuth 2.0 open protocol to establish access to both your GitHub repository (hosted by GitHub) and your work environments (hosted by Salesforce). OAuth allows you to delegate a client application (DevOps Center) to access data from a protected resource (your project repository, for example) through the exchange of tokens, instead of exchanging security credentials. For details, see Authorize Apps with OAuth in Salesforce Help.

Create a GitHub Account

If you already have a GitHub account, great! If you don't have a GitHub account yet, it's easy (and free) to sign up for one.

New to GitHub or to Source Control?

We've designed DevOps Center to make it easy to take advantage of a source control system like GitHub even if you aren't yet familiar with it. If you want to learn more about Git and GitHub concepts and terminology as you dive in to using DevOps Center, look at the Git and GitHub Basics Trailhead module (estimated time: less than two hours), which covers why source control is so key to successful team collaboration and what to expect in a typical GitHub workflow.

Methods to Create GitHub Project Repositories

A DevOps Center project repository must contain a Salesforce DX project. You can create a DevOps Center project repository using any of these methods:

- When you create your project within DevOps Center, let it create a Github repository that uses the Salesforce DX project structure. Skip to Manage Environments.
- Use an existing Github repository.
- Create a different GitHub repository.

Note If you plan to use an existing GitHub repository, it must contain an `sfdx-project.json` file in the root directory, a file that identifies the repo as a Salesforce DX project.

Create a New Repository

Creating a repository from our template ensures that the repository has the right structure and configuration to work with DevOps Center, that is, the structure and configuration of a Salesforce DX project.

If you don't already have a source control repo, a corresponding project repository with the proper DX structure is created when you define a DevOps Center project. Alternately, you can log in to GitHub and

use the repository template at https://github.com/forcedotcom/dx-empty to create your project repository. Your project repository can be either public or private. You don't need to include all branches. For details about how to create a repository from a template, see the GitHub help.

After the repository is created, you can seed it by adding metadata files from your production org or another source org.

Add Team Members to the GitHub Repo As Collaborators

In GitHub, add team members as repository collaborators. A collaborator is someone who has been granted write access to the project repository. Add everyone who creates customizations or code for your project to the project repository as a collaborator. That way, they have access to the repository through DevOps Center.

To add a team member as a collaborator in the project repository:

1. Ask each of your team members to create a GitHub account (if they don't have one already) 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.

   GitHub sends an email to each team member you invited, asking them to accept the invitation.

Tip Follow up with your team members to make sure they accept the invitation from GitHub to avoid access problems when the team members begin using DevOps Center.

If an Organization Owns the GitHub Repo

GitHub repos owned by an organization aren't visible in DevOps Center until an organization account owner provides access.

1. Authenticate to GitHub through DevOps Center.
2. In GitHub, provide access to both Salesforce Integration applications.

   Each integration application corresponds to the login mechanism used to authenticate to the org: `test.salesforce.com` or `login.salesforce.com`. The integration app for `test.salesforce.com` provides integration across your GitHub repo, DevOps Center, and development environments (scratch orgs and sandboxes). The integration app for `login.salesforce.com` provides integration across your GitHub repo, DevOps Center, and non-sandbox orgs (production and Developer Edition orgs).

   If you see only one Salesforce Integration application, you have connected to environments that use only one of the login mechanisms.

   a. In your personal GitHub account, go to **Settings**.

   b. Click **Applications**, then select **Authorized OAuth Apps**.
   c. Click **Salesforce Integration Application**.
   d. Find the organization that owns your repo, then click **Request**.
   e. (If applicable) Repeat steps for the other Salesforce Integration Application.
3. After the GitHub repository owner approves your request for access, open the DevOps Center org.
   a. If you're in DevOps Center, click the Home icon to get to the org's home page.
   b. Click your profile icon, then select **Settings**.
   c. Click **Authentication Settings for External Systems**.
   d. Delete **DevOps Center GitHub**.
4. Now, you can proceed to Create a GitHub Project.

If the GitHub Organization Has an IP Allowlist

If you have restricted access to the GitHub organization assets by configuring an allowlist, add the domains for the DevOps Center app.

1. Add either these domains or IP addresses to your GitHub organization's allowlist:

```
http://devops-center.staging.repo-ops.sfdc.sh
https://devops-center.staging.repo-ops.sfdc.sh
http://devops-center.prod.repo-ops.sfdc.sh
https://devops-center.prod.repo-ops.sfdc.sh
```

```
44.225.154.211
100.21.209.24
52.43.33.118
44.230.180.227
```

2. Add these Salesforce domains.

   See Salesforce Help: Allow the Required Domains.

See Also

Salesforce Core Services Article: IP Addresses and Domains to Allow
GitHub Docs: Managing allowed IP addresses for your organization

Connect to a Different GitHub Account

When creating or opening a project, all team members log in to GitHub using the account that has access to the repos that your team wants to use with DevOps Center. If you or a team member logged in with the incorrect GitHub account, a team member with Salesforce admin privileges can delete the authorization information so you or the team member can log in using the proper credentials.

If you're using DevOps Center and don't have a Lightning Experience tab opened, click the Home icon.

1. From Setup, enter *Named Credentials* in the Quick Find box, then select **Named Credentials**.
2. Click **DevOps Center GitHub**.
3. Under External Data User Authentications, delete the record for the affected user.

When you or the team member returns to DevOps Center and opens the project, this message appears:

```
Background sync is attempting to run but requires you to log in to the source
control
system to continue. Log in...
```

Click **Log in**, click **Connect to GitHub**, and then proceed to log in with the proper GitHub account.

Set Up DevOps Center for Bitbucket (Beta)

Set up DevOps Center to use a Bitbucket source control repository. Currently, we support only Bitbucket Cloud.

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.

A repository stores project work files – code, text, images, and so on. Each DevOps Center project needs its own repository for storing project changes. While you're working on the project, the repository is the team's centralized source of truth to manage changes.

If you're familiar with using DevOps Center with GitHub, some of the steps are different for setting up Bitbucket. As part of our extensibility initiatives, we're transforming our user interface, where DevOps Center functionality is moving to Lightning Experience so you have more control and flexibility in customizing the user experience. During this transition period, some Bitbucket setup and configuration tasks are performed in Lightning Experience and some are performed in the DevOps Center app. Menu options and buttons help you smoothly transition between the two.

Bitbucket Workflow

- Create a Bitbucket Cloud account. All team members must have their own Bitbucket Cloud account.
- If your organization doesn't already use Bitbucket, follow the Bitbucket documentation to set up:
  - A workspace
  - A project
  - At least one repository

Bitbucket Repository Configuration

If you have an existing Bitbucket repo, you can use it as long as it follows the Salesforce DX project structure. If it isn't a Salesforce DX repo, you can add the necessary DX project file.

- In Bitbucket, add team members to the repo as collaborators.
- In Lightning Experience, connect the repository in DevOps Center.
- In Lightning Experience, create the DevOps Center project.
- Complete any optional configuration tasks.
- In the DevOps Center app, add your project environments, build your pipeline, and create and assign work items. All user tasks are performed in the DevOps Center app.

Add Salesforce DX Project File to Bitbucket Repository

To support DevOps Center features, the Bitbucket repository must follow the Salesforce DX project structure and contain a Salesforce DX project file.

Add Team Members to the Bitbucket Repo As Collaborators

In Bitbucket, add team members as repository collaborators. A collaborator is someone who has been granted write access to the project repository. Add everyone who creates customizations or code for your project to the project repository as a collaborator. That way, they have access to the repository through DevOps Center.

Connect the Bitbucket Repo in DevOps Center

Each DevOps Center project has a corresponding project repository. By connecting the Bitbucket repo to DevOps Center, DevOps Center can interact with the repo to perform operations such as creating feature branches, creating change requests, and committing and merging changes to each pipeline branch (stage).

See Also

Set Up DevOps Center for GitHub

Add Salesforce DX Project File to Bitbucket Repository

To support DevOps Center features, the Bitbucket repository must follow the Salesforce DX project structure and contain a Salesforce DX project file.

For Bitbucket beta, we aren't providing a Salesforce DX project repository template. If the repository you plan to use with DevOps Center doesn't contain an `sfdx-project.json` file, you must add one.

1. In Bitbucket, navigate to your repository.
2. From the context menu, select **Add File**.
3. Enter **sfdx-project.json** for the filename.
4. Copy these contents, then click **Commit**.

```
{
  "packageDirectories": [
    {
```

```
      "path": "force-app",
      "default": true,
      "versionName": "Winter '25",
      "versionNumber": "1.0.0.NEXT"
    }
  ],
  "namespace": "",
  "sourceApiVersion": "61.0"
}
```

You can indicate a different directory for the `path`, which indicates where your source is located. If that directory doesn't currently exist, you must create it.

Accept for update the commit message, then click **Commit**.

5. (Optional) If the path indicated in the sfdx-project.json file doesn't exist, create it.
   a. Go back to the repository view.
   b. From the context menu, select **Add File**.
   c. For the filename, indicate **force-app/text.txt**.
      To create the directory, you must also create a file. Add a comment to the text file so you can commit it.
   d. Click **Commit**, accept or update the commit comment, then click **Commit** again.

Next: add team members as collaborators. You can then go to DevOps Center to connect the repository and then create a DevOps Center Bitbucket project.

Add Team Members to the Bitbucket Repo As Collaborators

In Bitbucket, add team members as repository collaborators. A collaborator is someone who has been granted write access to the project repository. Add everyone who creates customizations or code for your project to the project repository as a collaborator. That way, they have access to the repository through DevOps Center.

1. In the Bitbucket repository, click **Invite**.
2. Click **Add Users or Groups**.
3. Search for team members to invite, provide the appropriate privileges, and then click **Confirm**.

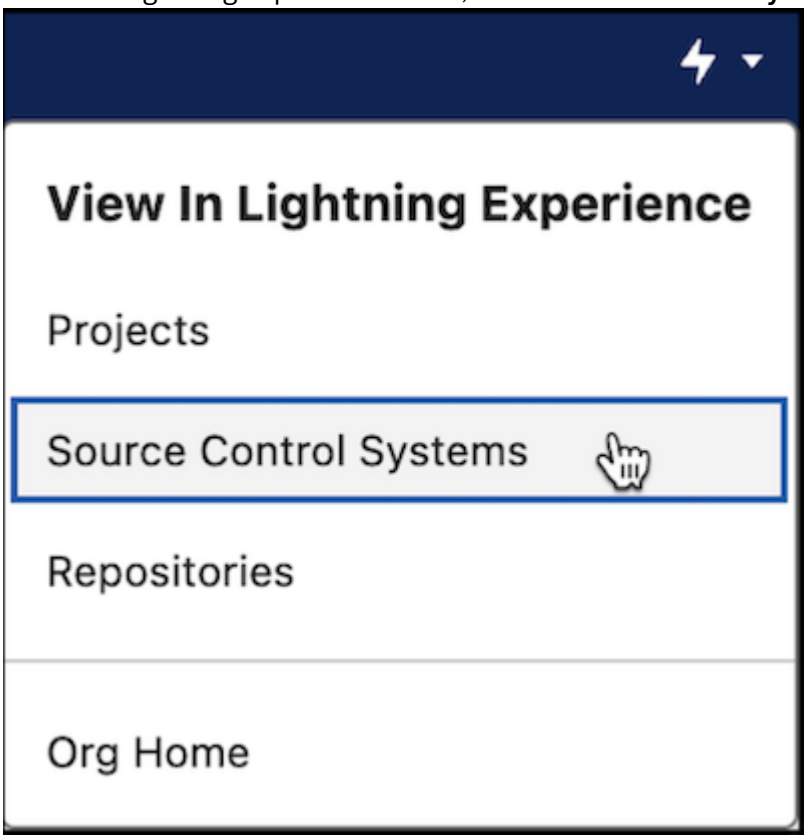
Follow up with your team members to make sure they accept the invitation to avoid access problems when the team members begin using DevOps Center.

Connect the Bitbucket Repo in DevOps Center

Each DevOps Center project has a corresponding project repository. By connecting the Bitbucket repo to DevOps Center, DevOps Center can interact with the repo to perform operations such as creating feature

branches, creating change requests, and committing and merging changes to each pipeline branch (stage).

1. Launch the DevOps Center app.
2. From the Lightning Experience menu, select **Source Control Systems**.



3. Select **All** in the Source Control Systems list view.
4. Select **Bitbucket**.
5. Click **Connect Repository**.

   The first time you connect a Bitbucket repo, you're prompted to grant access to Bitbucket, so DevOps Center can work with your Bitbucket account. After you go through the authentication process, DevOps Center can make changes on your behalf in your project's repository.

   If you upgraded from a previous version of DevOps Center, it's possible that the Connect Repository button is missing. Add it to the page layout to provide you the ability to connect to the repository.

6. In the Repository Path box, paste the Bitbucket repo URL, then click **Connect**.

Next: Create the DevOps Center project associated with the Bitbucket repo.

Manage DevOps Center Projects

Your team's central arena for work in DevOps Center is the project. The purpose of a project is to help you and your team 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 work environments are used to make changes
- 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

Projects in DevOps Center require a source control repository that uses the Salesforce DX project structure. The repository is used to store project changes. You can use the same source control repository across projects. However, use unique branches for pipeline stages across DevOps Center projects, except for the release branch (often called main), when building the release pipeline.

Create a GitHub Project
Create a GitHub project to help you and your team manage changes for a particular application or customization.

Create a Bitbucket Project (Beta)
After you connect the Bitbucket repository in DevOps Center, create the DevOps Center project associated with this source code repository to help you and your team manage changes for a particular application or customization.

Update Project Details
Update project details for a DevOps Center project. You can change the name and description for a project after you define the release environment.

Hide a Project
Hide a DevOps Center project so that it doesn't show up on the Project page. The project still exists but is removed from the Projects list.

Create a GitHub Project

Create a GitHub project to help you and your team manage changes for a particular application or customization.

To create a Bitbucket project, see Set Up DevOps Center for Bitbucket (Beta).

Projects in DevOps Center require an associated source control repository that uses the Salesforce DX project structure. The repository is used to store project changes.

1. From App Launcher, find and select **DevOps Center**.

   DevOps Center opens to the All Projects page. If you're logging in for the first time, the list is empty.

2. Click **New Project**.

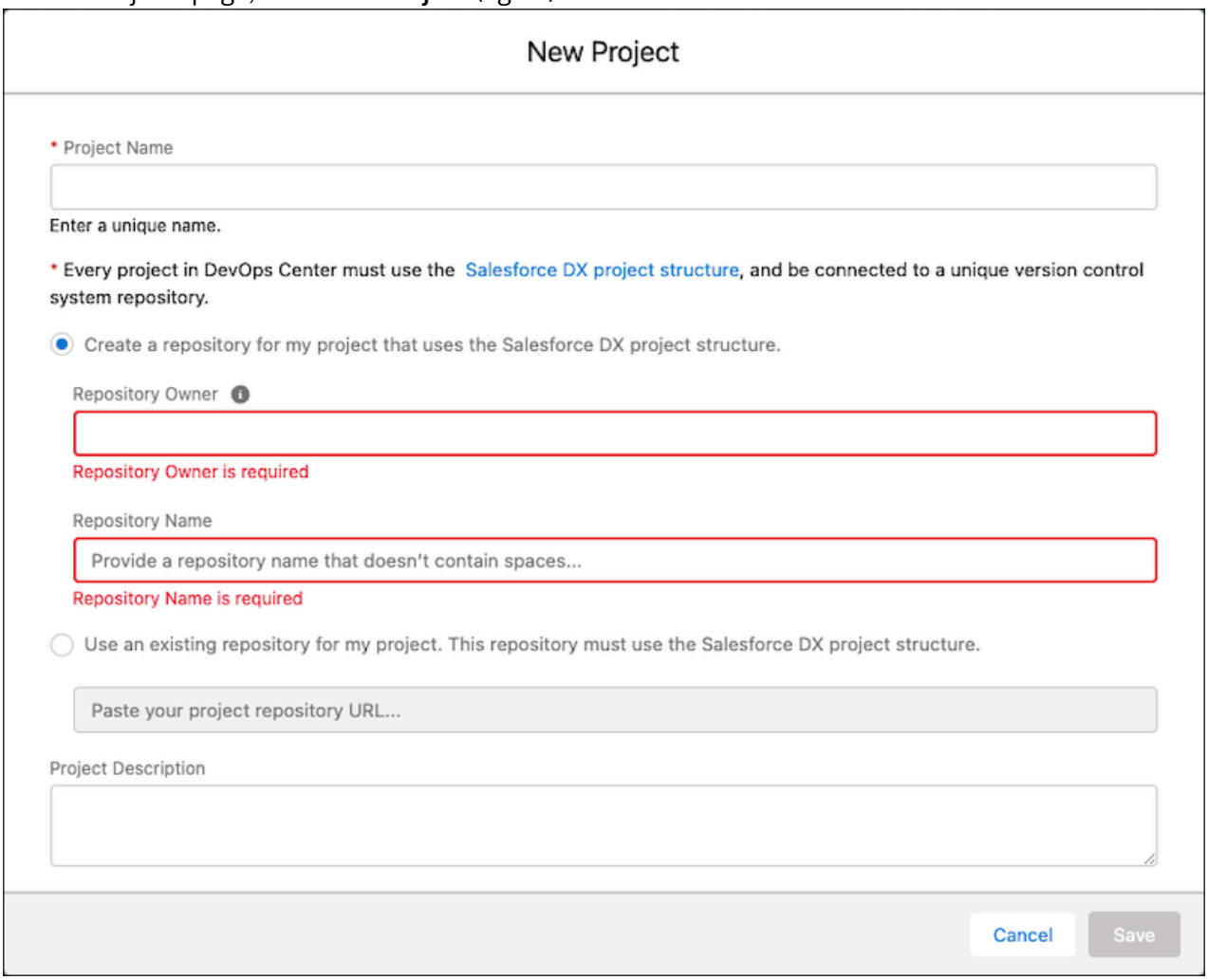
   The first time you create a project, you're prompted to log in to GitHub, so you can authorize DevOps Center to work with your GitHub account. After you go through the authentication process, DevOps

Center can make changes on your behalf in your project's GitHub repository.

3. Click **Connect to GitHub**.

Is your GitHub repo owned by an individual or an organization?	How to proceed
Individual account	Authorize access so DevOps Center can make changes in GitHub on your behalf. After authentication, you're returned to the DevOps Center Projects page.
Organization account	Sometimes, GitHub repos owned by an organization aren't visible in DevOps Center without specifically providing access via OAuth. After you follow the instructions in If an Organization Owns the GitHub Repo, and the repo owner approves the request for access, you can create the project.

4. On the Projects page, click **New Project** (again).



5. Enter a unique name for your project.
6. Create a repository or use an existing one.
   - If you create a repo, we base the repository name on the project name. However, you can change the name as long as it doesn't contain any spaces.
   - If you use an existing repository, enter the URL of the GitHub repository you want to use for the project. For example, https://github.com/mygithubusername/Myrepo. If using the same source control repository as another DevOps Center project, use unique branches for each pipeline stage, except for the release branch (often called main), when building the release pipeline.
7. (Optional) Enter a description to identify the purpose of the project.
8. Click **Save**.

   Your project is created and added to the Projects page. Different projects can be in different phases of project setup.

9. Under Release Environment, click **Click to Create Pipeline** to build your pipeline and add the associated environments.

See Also

Salesforce Help: Open a Work Item
Connect to a Different GitHub Account

Create a Bitbucket Project (Beta)

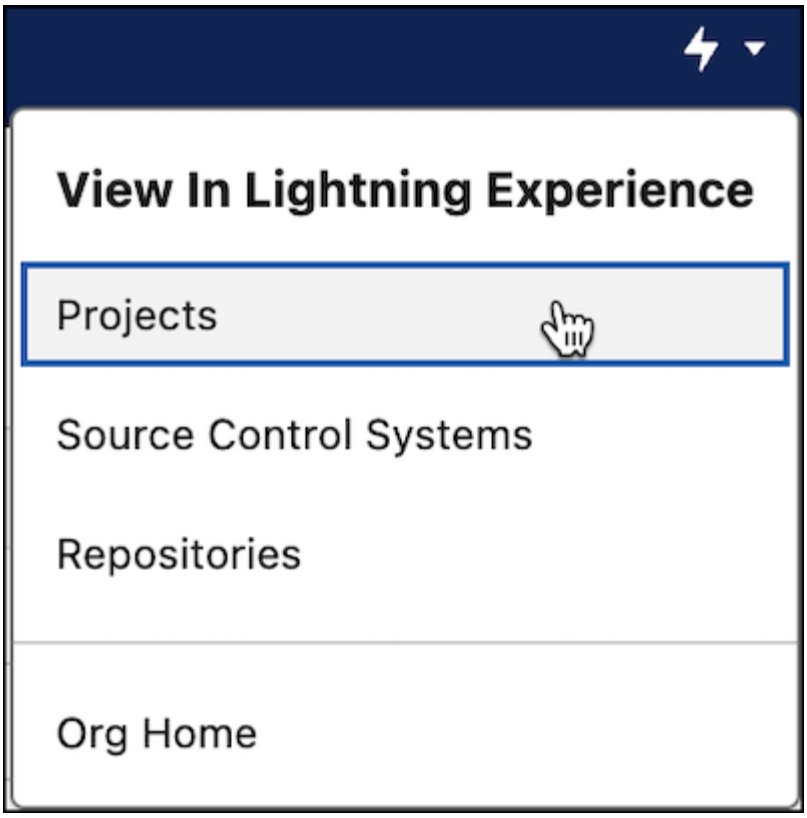
After you connect the Bitbucket repository in DevOps Center, create the DevOps Center project associated with this source code repository to help you and your team manage changes for a particular application or customization.

Before you continue, complete the steps in Set Up DevOps Center for Bitbucket (Beta).

Note DevOps Center Bitbucket 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.

As part of our extensibility initiatives, we're transforming our user interface, where DevOps Center functionality is moving to Lightning Experience so you have more control and flexibility in customizing the user experience. During this transition period, some Bitbucket setup and configuration tasks, such as creating a DevOps Center project for Bitbucket, are performed in Lightning Experience instead of directly in the DevOps Center app.

1. In the DevOps Center app, from the Lightning Experience menu, select **Projects**.



2. (Optional) Select **All** to view all DevOps Center projects in the Projects list view.
3. Click **New**.
4. In the New Projects dialog, enter a unique project name and optional description.
5. In the Platform Repository field, enter or search for the appropriate Bitbucket repository.

   Copy the full repository name, or search by entering the starting string.

6. Click **Save**.

   DevOps Center opens the project's record page.

7. Click **View in DevOps Center** to continue project configuration in the DevOps Center app.

   If you upgraded from a previous version of DevOps Center, it's possible that the View in DevOps Center button is missing. We recommend that you add it to the page layout; however, you can also launch the DevOps Center app from App Launcher.

In the DevOps Center app, add your project environments, build your pipeline, and create and assign work items.

See Also

- Manage Environments
- Plan Your Pipeline
- Create and Assign Project Work Items

Update Project Details

Update project details for a DevOps Center project. You can change the name and description for a project after you define the release environment.

1. From the Projects page, click the project name.
2. Click **Settings**.
3. Change the project name or description, or both.
4. Click **Update**.

Hide a Project

Hide a DevOps Center project so that it doesn't show up on the Project page. The project still exists but is removed from the Projects list.

1. In DevOps Center, in the org URL, delete `sf_devops/DevOpsCenter.app` and replace it with `lightning/o/sf_devops__Project__c/list`.

   ```
   https://MyDomainName.lightning.force.com/lightning/o/sf_devops__Project__c/list
   ```

2. From the Projects list dropdown, select **All Projects**.

   If the Projects list is empty and you don't see an option to view all projects, create a Projects list view.

   a. From the List View Controls dropdown, select **New**.
   b. Enter the list name, such as *Project List*.
   c. Click **Save**.
3. In the list, click the link for the project you want to hide.
4. In the Project Details page, select **Hidden**, then click **Save**.

If you change your mind, you can unhide the project.

Complete Optional Setup Tasks

Complete these optional setup tasks, as needed.

- Create a `.forceignore` file to manage which files and directories are ignored during metadata deployments and retrievals. Some compelling reasons to create this file include:
  - Preventing unnecessary files from deploying
  - Minimizing the risk for errors and conflicts
  - Protecting sensitive information or configuration files you don't want to deploy to the production environment
- Install the DevOps Center CLI plugin to automate deployments through your pipeline using Salesforce

CLI on the command line or in scripts.
- Seed a source control repository with metadata from an org. If you don't currently use source control, these steps guide you through retrieving your org metadata to a Salesforce DX project. Although these steps were designed and tested using a GitHub repository, they likely work using any Git-based repository.

Determine What Metadata Types to Ignore During Development
Your Salesforce DX project repo contains a `.forceignore` file that determines what metadata and files to exclude when retrieving files from development environments and deploying changes to environments in your release pipeline.

(Optional) Install Salesforce CLI DevOps Center Plugin (Beta)
Install the Salesforce CLI DevOps Center plugin to perform DevOps Center actions on the command line for developers who plan to work outside of DevOps Center or want to automate tasks. We plan to roll out new commands over time. The first set of commands support the ability to deploy changes to a pipeline environment and perform a validate-only deployment.

(Optional) Seed a Source Control Repository with Org Metadata
If you're currently using change sets to move changes to environments in your release pipeline, it's likely that you don't have a source control repository that contains your production org's metadata. Although you can approach this task in any number of ways, this method uses the Salesforce (Developer Experience) DX tools that many of you're already familiar with. However, those of you who don't have any experience using DX developer tools can easily follow along.

Determine What Metadata Types to Ignore During Development

Your Salesforce DX project repo contains a `.forceignore` file that determines what metadata and files to exclude when retrieving files from development environments and deploying changes to environments in your release pipeline.

The `.forceignore` file structure mimics the `.gitignore` structure. Each line in `.forceignore` specifies a pattern that corresponds to one or more files. The files typically represent metadata components, but can be any files you want to exclude, such as LWC configuration JSON files or tests. Create or edit the file directly in the source control repository.

Considerations When Initially Creating the .forceignore File

If you used our template to create your repo, the `.forceignore` file likely requires updating to reflect your specific project needs. If you manually create `.forceignore` in an existing repo, be sure the file is located in the root folder of the Salesforce DX project directory.

When you initially build your release pipeline, all pipeline stage branches contain the version of `.forceignore` that's in the main branch associated with your release environment. When work items are created, their associated feature branches use the version of the file that's in your first pipeline stage's branch. After you activate your pipeline, DevOps Center becomes aware of the `.forceignore` file and honors it when pulling or promoting changes.

Sample .forceignore Syntax

This sample file contains many of the common files and folders that developers want to ignore. The syntax is different for the promote (deploy) and pull (retrieve) operations. For example, if you want to exclude all profiles when promoting changes, you indicate `**/profiles/**`. If you want to exclude all profiles when pulling changes, you indicate `*.profile`.

```
# List files or folders below to ignore them when deploying and retrieving cha
nges

# Standard metadata
package.xml

# These metadata files are ignored when promoting (deploying)
**/appMenus/**
**/appSwitcher/**
**/fieldRestrictionRules/**
**/objectTranslations/**
**/profiles/**
**/profilePasswordPolicy/**
**/profileSessionSetting/**
**/settings/**
**/sharingRules/**
**/AuthProvider/**

# These metadata files are ignored when pulling (retrieving)
*.settings
*.appMenu
*.appSwitcher
*.rule
*.objectTranslation
*.profile
*.profilePasswordPolicy
*.profileSessionSetting
*.sharingRules
*.AuthProvider

# LWC configuration files
**/jsconfig.json
**/.eslintrc.json

# LWC Jest
**/__tests__/**
```

Considerations When Updating the .forceignore File

If changes are required, update the `.forceignore` file in the main branch of your project repo. If you have multiple DevOps Center projects, manually update the file in each project repository. However, the changes don't automatically propagate to existing branches in the source control system. To apply these changes to any existing work items or future promotions, copy and commit the changes to each work item feature branch and all pipeline stage branches in all source control project repositories. After the changes are pushed from the source control repository to DevOps Center, all future work item branches contain the updated file and all future promotions honor the changes in the file.

(Optional) Install Salesforce CLI DevOps Center Plugin (Beta)

Install the Salesforce CLI DevOps Center plugin to perform DevOps Center actions on the command line for developers who plan to work outside of DevOps Center or want to automate tasks. We plan to roll out new commands over time. The first set of commands support the ability to deploy changes to a pipeline environment and perform a validate-only deployment.

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. 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.

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. See Assign the DevOps Center Permission Sets for details.
- 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.

To install and configure Salesforce CLI, see Salesforce CLI Setup Guide. After you install Salesforce CLI, the DevOps Center plugin is installed the first time you run a DevOps Center CLI command. If you already have the CLI installed or want to proactively install the plugin, run this command:

```
sf plugins install @salesforce/plugin-devops-center
```

See Also

Salesforce CLI Command Reference: sf project deploy pipeline
Salesforce Help: Deploy Changes Using Salesforce CLI

(Optional) Seed a Source Control Repository with Org Metadata

If you're currently using change sets to move changes to environments in your release pipeline, it's likely that you don't have a source control repository that contains your production org's metadata. Although you can approach this task in any number of ways, this method uses the Salesforce (Developer Experience) DX tools that many of you're already familiar with. However, those of you who don't have any experience using DX developer tools can easily follow along.

Note If you already use source control and have your source control system repositories set up, you can skip this task.

Before you begin, install DevOps Center and the required developer tools. These steps were developed and tested using a GitHub repo.

- Install DevOps Center.
- Install Salesforce Extension for VS Code, Salesforce CLI, and Java Platform, Standard Edition Development Kit.
- Install Git.

After you create the DevOps Center project repository, you'll retrieve your metadata from a source org and then add those files to the repository. From now on, the source control repository's main branch now becomes your source of truth for all your metadata and changes.

This method assumes:

- You have limited experience with Salesforce DX developer tools.
- You don't currently use source control.
- You want to retrieve all (or most) your metadata from a source org, such as your release environment, to seed the source control repo.
- You haven't customized the user interface or Git functionality in VS Code.

After you complete the tasks to seed the repo, you have a source control repository with a main branch. Continue configuring DevOps Center, then add environments and build your release pipeline, which creates the associated pipeline stage branches.

1. **Create a DevOps Center Project Repository**
   Create a project in DevOps Center, which creates a corresponding GitHub repository with the required Salesforce DX project structure.
2. **Authorize the Source or Production Org**
   In Salesforce Extensions for VS Code, connect the release (production) org or source org that contains the metadata that you want to add to the source control repository.
3. **Generate a Manifest and Retrieve Metadata From the Org**
   In Salesforce Extensions for VS Code, use Salesforce CLI to generate the manifest that contains the metadata types you want to retrieve from the production org.
4. **Add the Local DX Project Files to the GitHub Repository**
   In Salesforce Extensions for VS Code, add the files you retrieved from your org to the DevOps Center

project's GitHub repository. The GitHub repository has the same name as the DevOps Center project name. After you add the files to the source control repository, the process creates the main branch, which you'll associate with your release environment. Other branches for pipeline stages are created when you build your pipeline in DevOps Center.

Create a DevOps Center Project Repository

Create a project in DevOps Center, which creates a corresponding GitHub repository with the required Salesforce DX project structure.

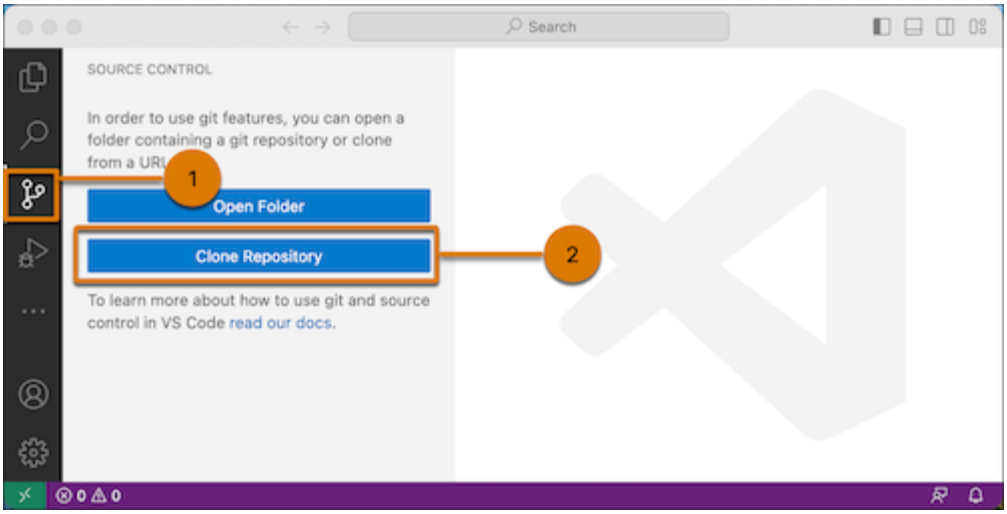
1. In DevOps Center, create a DevOps Center project.

   Select **Create a repository for my project that uses the Salesforce DX project structure**.

2. To launch GitHub, click the **Repository URL** link.

   Under Code, decide how you're going to clone the repo. Click the Code button dropdown menu, then copy the HTTPS or SSH command, depending on your company's security requirements.

3. Open VS Code then click the Code Fork icon (1), then click **Clone Repository** (2).



   a. Paste the repository URL, then press **Enter**.
   b. Navigate to the desired location for the repository folder on your local file system, then click **Select as Repository Destination**.
   c. To view the files in VS Code Explorer, click **Open**.

4. Click the `.gitignore` file to open it in the editor.

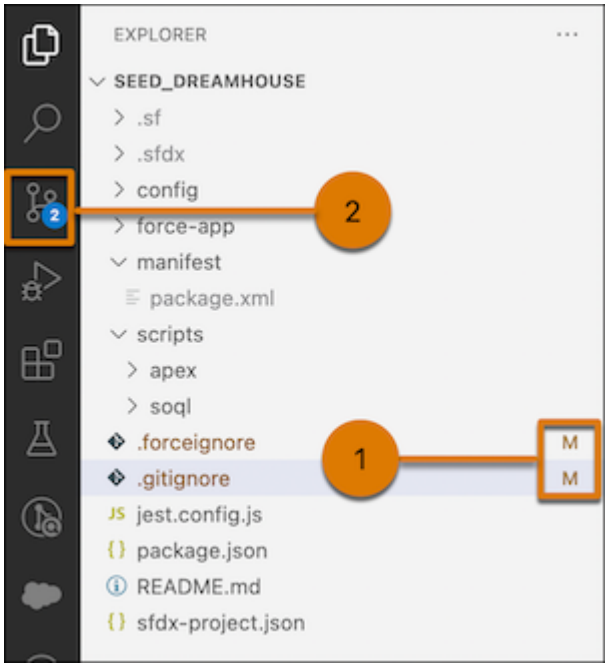
   This file specifies which files and file types you don't want to save in the repo. Later, you're going to create a manifest file for seeding the repo. In most cases, you don't need to keep this file. If you'd like to keep the file on your local file system but not store it in the repo, add `package.xml` to the `.gitignore` file, then save the file.

5. Click the `.forceignore` file to open it in the editor.

   This file specifies file and metadata types that you don't want to deploy to or retrieve from your org. If

you plan to add items to this file, we recommend doing it now so that all source control branches created in DevOps Center later contain the correct version of this file. See Determine What Metadata Types to Ignore During Development for recommendations on which metadata types and files to ignore. Updating the file later can be cumbersome. Don't forget to save the file.

If you update any files, VS Code denotes modified files with an M (1), and updates the badge for the Code Fork icon (2).



Next: Authorize the org that contains the metadata to add to the source control repository.

Authorize the Source or Production Org

In Salesforce Extensions for VS Code, connect the release (production) org or source org that contains the metadata that you want to add to the source control repository.

Before you begin, create a DevOps Center project with a corresponding GitHub source control repository.

1. In VS Code, at the bottom of the window, click **No Default Org Set**.
2. Click **SFDX: Authorize an Org**, then select the way to log in for the specific org.

   In most cases, the source org is your release (production) org. If so, select **Production**.

3. Enter an alias (nickname) for the org.
4. Log in with your credentials.

Next: Create the manifest (`package.xml`), then retrieve the org's metadata to your local DX project folder.

Generate a Manifest and Retrieve Metadata From the Org

In Salesforce Extensions for VS Code, use Salesforce CLI to generate the manifest that contains the metadata types you want to retrieve from the production org.

Before you begin, authorize the source org in Salesforce Extensions for VS Code.

1. In VS Code, open the Terminal.
2. Use Salesforce CLI to generate a comprehensive manifest file that lists all the metadata types that the source org contains.

   ```
   sf project generate manifest --output-dir ./manifest --from-org <orgname/alias>
   ```

   The `--output-dir` flag creates the `package.xml` file in a new directory called `manifest`. The `manifest` directory is where Salesforce Extensions for VS Code expects to find the `package.xml` file, so it provides a logical list of commands in the context menu.

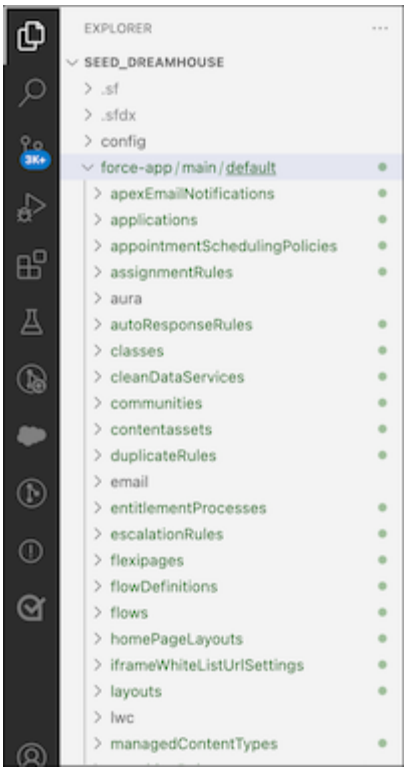
3. In VS Code Explorer, in the manifest folder, right-click the `package.xml` file, then select **SFDX: Retrieve Source in Manifest From Org**.

   If the retrieve operation fails, you're trying to retrieve too many files at once and you've exceeded the file limits for the retrieval. Split your `package.xml` into smaller files, and then run this command for each manifest file. For more information on retrieval limits, see Metadata API Guide: retrieve().

   After the files are retrieved, the project files are now in the DX project folder on your local machine.

4. In VS Code Explorer, expand the `force-app` folder to see the org metadata in your local DX project.

   If you're using the default VS Code UI configuration, new files appear as green in the Explorer.



Next: Add and commit the metadata files to the DevOps Center project's GitHub repository.

See Also

Metadata API Developer Guide: Sample package.xml Manifest Files
Metadata API Developer Guide: Metadata Types

Add the Local DX Project Files to the GitHub Repository

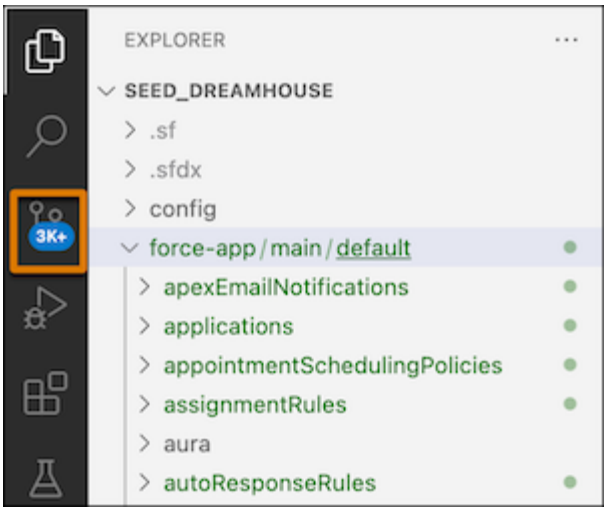
In Salesforce Extensions for VS Code, add the files you retrieved from your org to the DevOps Center project's GitHub repository. The GitHub repository has the same name as the DevOps Center project name. After you add the files to the source control repository, the process creates the main branch, which you'll associate with your release environment. Other branches for pipeline stages are created when you build your pipeline in DevOps Center.

Before you begin, retrieve metadata from the production org to your local DX project. Use the same public GitHub account that you created for DevOps Center.

If your project contains fewer than 100 files, you can upload them directly in GitHub, then commit and merge them into the main branch. This method is straight-forward for small repositories. See GitHub Docs: Adding a file to a repository.

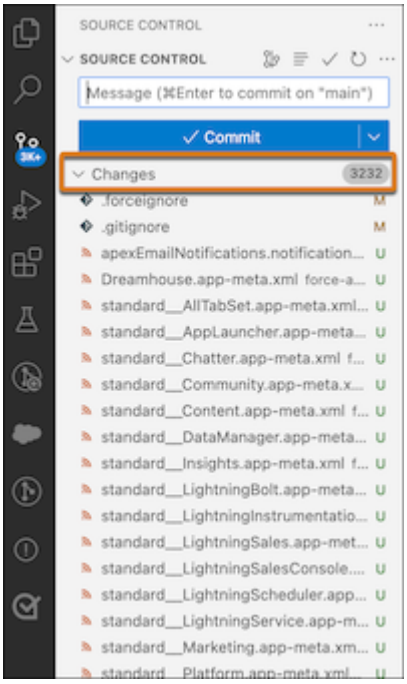
If your project contains more than 100 files, use this method to populate the source control repository using the click-based Git functionality in VS Code. After you become comfortable using Git within VS Code, you can use this method for any number of files.

1. In VS Code, open the project, then click the Code Fork icon.

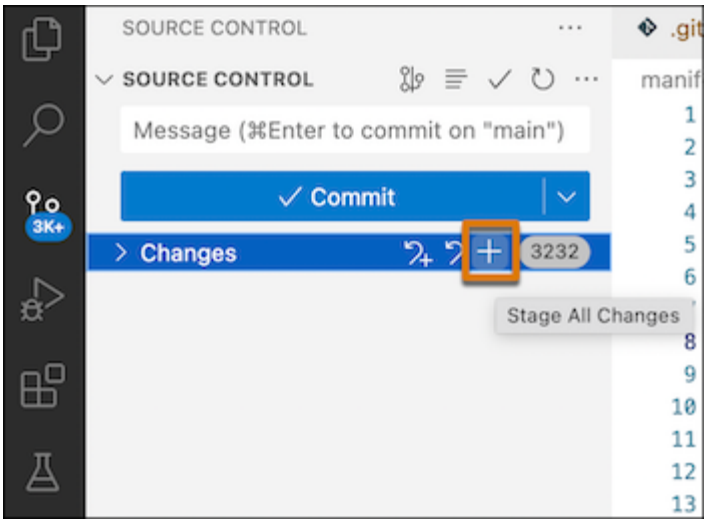


2. Under Changes, see the metadata that you retrieved from your org.

   Each file is tagged as Untracked (U), which means that GitHub is now aware of it as a new file but it isn't yet synched with the repository.

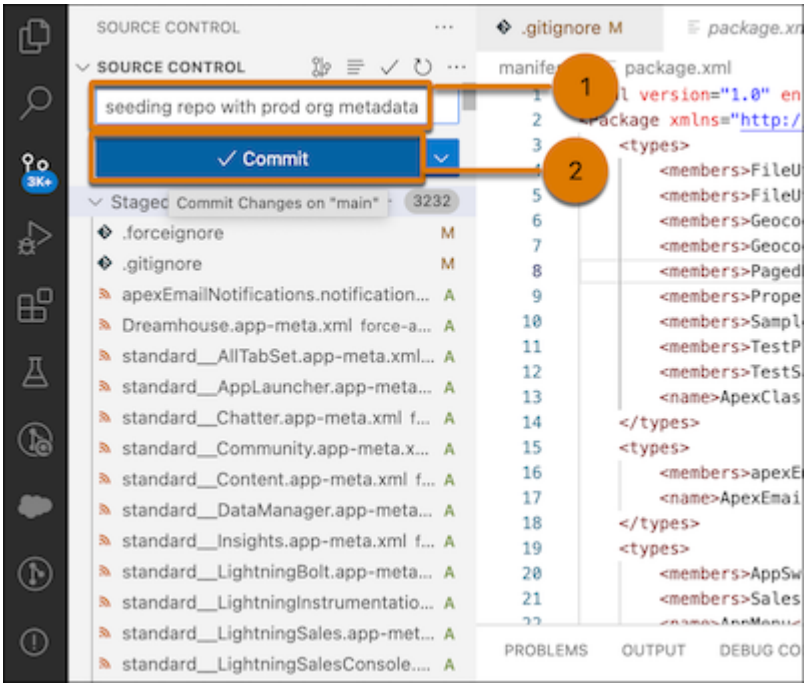


3. Click the **Changes** bar, then click **+** to stage all changes, which creates a snapshot of the files so you can commit them.



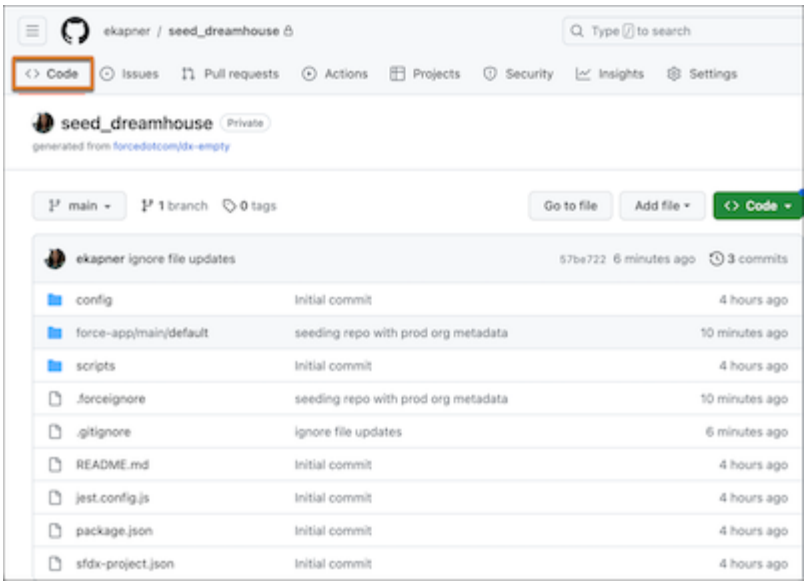
VS Code denotes staged files with an A.

4. Enter a commit message (1), then click **Commit** (2), which commits and pushes the files to the main branch.



5. (Optional) If you modified the `.gitignore` or `.forceignore` files, click **Sync Changes** to push them to the main branch in the repository.
6. Go to GitHub to view your seeded project repository.

   In GitHub, find the repository, or click the Repository URL link from within DevOps Center. To see your files listed in the main branch, click the **Code** tab.



You have successfully added files from your production org to a source control repository and created the main branch. When you build your release pipeline within DevOps Center, you create the rest of your branches when you define each pipeline stage.

Next: In DevOps Center, continue configuring your DevOps Center project. Then add the release environment (production org) and then build your release pipeline.

Manage Environments

When first configuring DevOps Center, identify the environments that you're going to use for each project. After environments are defined in the Pipeline Environments tab, you can manage each environment from its drop-down menu.

Identify Your Environments

Have the login credentials handy for the environments you plan to use for your DevOps Center project. Before you proceed, make sure all the dev and pipeline environments for this project are created. Enable Source Tracking for Sandboxes for your developer sandboxes. Scratch orgs have source tracking enabled by default. The synchronize development environments feature keeps dev environments in sync with the source control repository.

- You can enable source tracking in production for all Developer and Developer Pro sandboxes. After you enable this feature, you must refresh any developer sandboxes before proceeding. All newly created or refreshed sandboxes have source tracking enabled.
- You can enable source tracking in a specific sandbox in the Sandbox Settings page. Refreshing the sandbox isn't required for this option.

See Salesforce Help: Enable Source Tracking in Sandboxes for more information.

Before you continue, be sure to:

- Create all the necessary source-tracked Developer or Developer Pro sandboxes you need for the project.
- Create sandboxes (as needed) for pipeline stages, for example, integration, user acceptance testing, and staging.
- Gather the usernames and passwords for all environments, including the final release environment, such as production.
- Add team members as users to every environment that they require access to.
- Update the IP allow list for the DevOps Center app (if your security settings require it).

What's Special About Development Environments?

Development environments require source tracking so they can automatically track changes as they are made. Use Developer or Developer Pro sandboxes created from your production org.

Ideally, everyone contributing to a DevOps Center project is assigned their own Developer sandbox.

Add the Release Environment for the Project

At a minimum, projects require a release environment (often your production org) and one developer environment. Be sure all team members are users in this environment if they require access to it to perform deployments to it from within DevOps Center.

Add Development Environments to the Project

Add the source-tracked development environment that you created for each developer working on the project.

Update IP Allowlist for DevOps Center App

If your security settings require it, add the necessary domains for all environments that have allowlisting enabled.

Environment Management Options

You have several options to manage your environments from the environment's menu in the Pipeline Environments tab. Which menu options are available is determined by the environment type and whether the pipeline is activated.

See Also

Salesforce DX Developer Guide: Enable Source Tracking in Sandboxes
Salesforce DX Developer Guide: Track Changes Between Your Project and Org
Salesforce Help: Are You Sharing a Development Environment?
Salesforce Help: Synchronize Your Development Environment

Add the Release Environment for the Project

At a minimum, projects require a release environment (often your production org) and one developer environment. Be sure all team members are users in this environment if they require access to it to perform deployments to it from within DevOps Center.

1. On the All Projects page, under Release Environment, click **Click to Create Pipeline**.
2. In the Pipeline Environments page, find the Production stage, then click **Connect Environment**.
3. Provide a unique name for the production environment and indicate the environment type.

   By default, DevOps Center populates this field with the project name + `Production`. Environment names must be unique across projects.

4. Click **Log In**.
5. Log in to the environment.
6. To allow this environment to connect to DevOps Center, click **Confirm**.

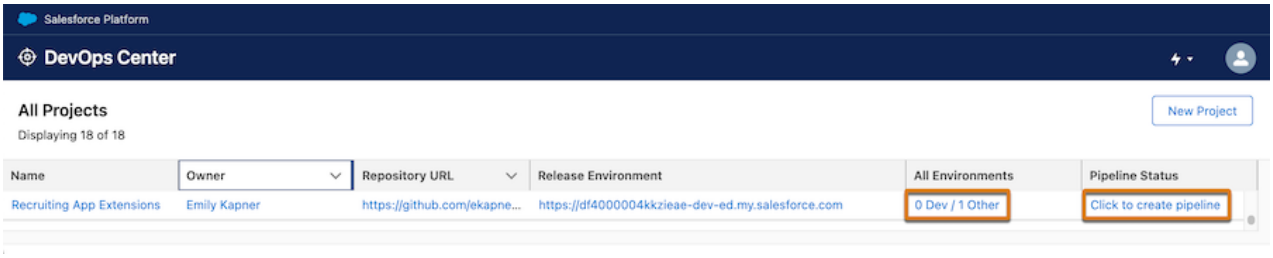
   DevOps Center uses an OAuth process to remember your credentials. The release environment URL is now listed on the Projects page. Now define the development environments for this project.

Add Development Environments to the Project

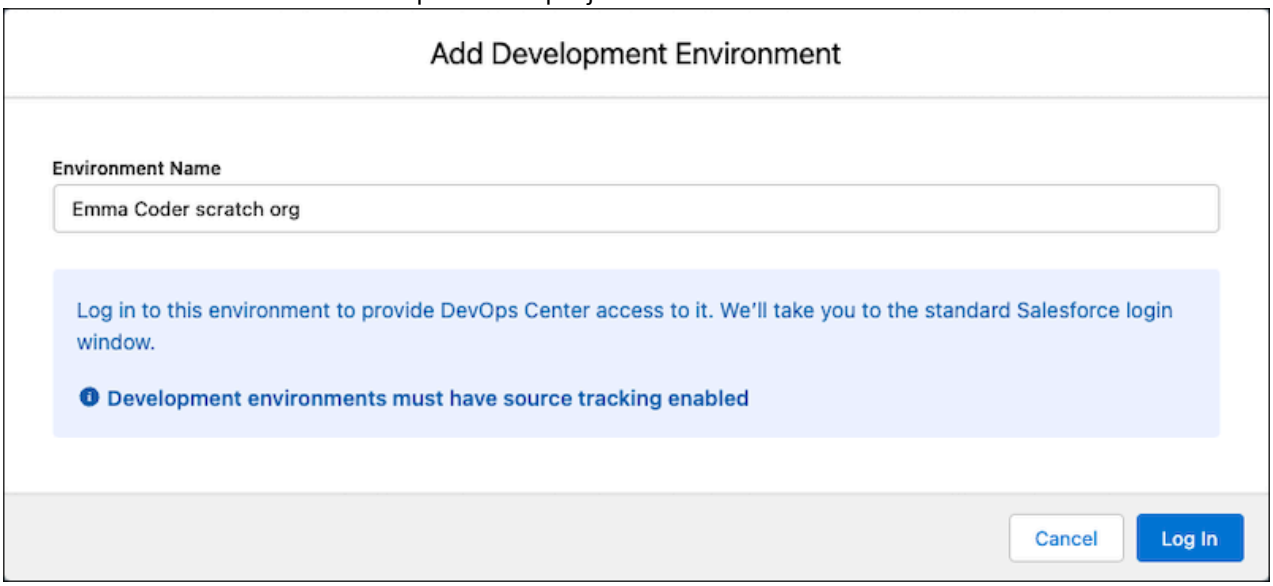
Add the source-tracked development environment that you created for each developer working on the project.

Important When adding a sandbox, be sure it's a Developer or Developer Pro sandbox with source tracking enabled. Other sandbox types don't support source tracking, which is a requirement for development environments.

1. Navigate to the Pipeline Environments page.
   - If you just defined your release environment, you're already there.
   - From the the Project page, click the hot link text under All Environments or click **Click to Create Pipeline**.



2. In the Pipelines Environments page, click **Add Development Environment**.
3. Enter a unique name for the development environment.
   Environment names must be unique across projects.



4. Click **Log In**.

   DevOps Center takes you to `test.salesforce.com` to log in.

5. To allow this environment to connect to DevOps Center, click **Confirm**.

   DevOps Center uses an OAuth process to remember your credentials.

6. Repeat steps to add more development environments.

   Now, when team members select or are assigned a work item, they can easily connect to their development environments from within the work item. Users are asked to authenticate to the environment with their own credentials to authorize their use from DevOps Center.

Important You can remove a development environment up until the developer commits a change in a work item.

Next, determine your pipeline configuration (how many stages), then add the other environments associated with your pipeline stages.

See Also

Plan Your Pipeline

Salesforce DX Developer Guide: Enable Source Tracking in Sandboxes

Update IP Allowlist for DevOps Center App

If your security settings require it, add the necessary domains for all environments that have allowlisting enabled.

1. Add either these domains or IP addresses to your allowlist for the DevOps Center app:

```
http://devops-center.staging.repo-ops.sfdc.sh
https://devops-center.staging.repo-ops.sfdc.sh
http://devops-center.prod.repo-ops.sfdc.sh
https://devops-center.prod.repo-ops.sfdc.sh
```

```
44.225.154.211
100.21.209.24
52.43.33.118
44.230.180.227
```

2. Add these Salesforce domains.

   See Salesforce Help: Allow the Required Domains.

See Also

Salesforce Core Services Article: IP Addresses and Domains to Allow
Plan Your Pipeline

Environment Management Options

You have several options to manage your environments from the environment's menu in the Pipeline Environments tab. Which menu options are available is determined by the environment type and whether the pipeline is activated.

Swap Environment

Swap a development or pipeline environment with another after the pipeline is activated. Use this operation after a sandbox has been refreshed to re-establish the connection with DevOps Center.

- Development environments–if DevOps Center can determine that the environment was descended from an environment in the project pipeline, all unpromoted work items that are connected to the current dev environment are moved to the new dev environment. If DevOps Center can't determine the environment's ancestry, DevOps Center deploys all changes from the first pipeline stage's branch to the new dev environment.
- Pipeline environments–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 the environment's ancestry, DevOps Center deploys all changes from the pipeline stage's branch to the new pipeline environment.

Remove Environment

Remove an unused dev environment from the list. If you change your mind, you can add a removed environment again. However, you can't remove an environment after it has an activity history associated with it. After an environment has an activity history, your options are to disconnect it or swap it.

Disconnect Environment

Disconnect a dev environment that is expired or you no longer plan to use to remove it from the list. You can disconnect an environment only if no active work items are associated with it. After it's disconnected, you can still view information about the environment in the Activity History but you can't use it again in the DevOps Center project. You can, however, use the environment in a different DevOps Center project.

Change Environment

Select a different environment for a pipeline stage, if the pipeline isn't yet activated.

See Also

Salesforce Help: Swap a Development Environment
Salesforce Help: Swap a Pipeline Environment

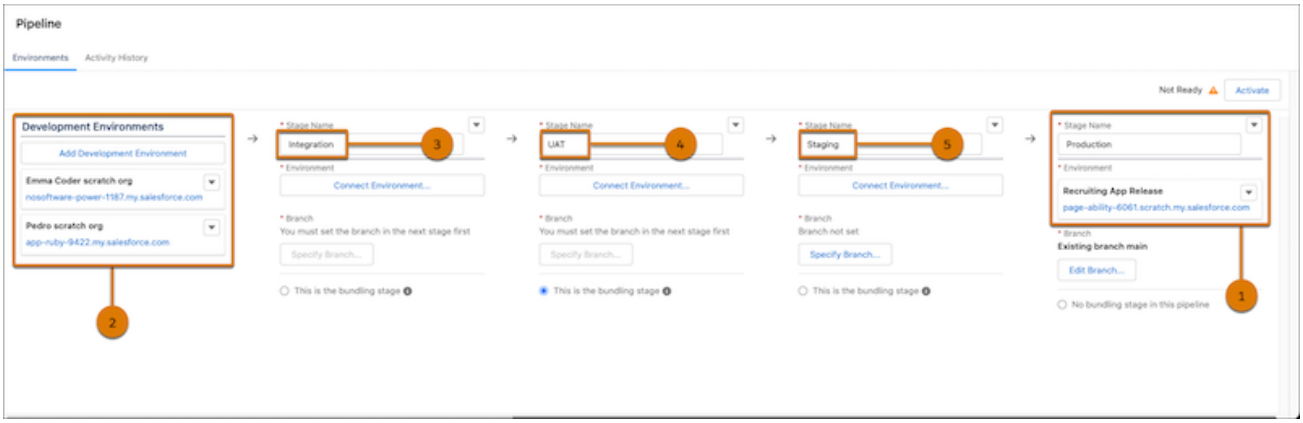
Plan Your Pipeline

A pipeline defines the sequence of stages that work items progress as they go through the release lifecycle from development through to production (or some other final release stage).

The pipeline consists of pipeline stages. Each pipeline stage corresponds to an environment (currently a Salesforce org), and a branch in the source control repository. The pipeline can't be modified after any changes have been promoted through it.

How Many Pipeline Stages Do I Need?

Your pipeline can contain any number of pipeline stages. The configuration of your pipeline is entirely up to you, and is based on the development and business processes that you have in place.



A pipeline requires only a *final release environment* (1). However, if your team performs some of its development work in sandboxes and scratch orgs, you can connect the *development environments* that you plan to use for this project when building the pipeline. (2). We recommend that you have at least one test stage besides your development and production stages. Each pipeline stage has an associated environment (org) and a corresponding branch in the source control repository.

To assist you with building a robust pipeline, which typically includes 2–3 test stages, we provide a pipeline template. This pipeline template includes these recommended stages:

Stage	Description
Integration	The first test stage (after the development stage) is an integration stage (3) where all changes from the various development environments come together for the first time and can be tested in an integrated environment. This stage is where you identify conflicts and resolve them before you move forward.
UAT	You can have one or more stages in between “integration” and “staging” (4) where you can perform additional testing, including by business stakeholders, often called user acceptance testing (UAT). We indicate this stage as the bundling stage, where you version a group of related changes and move them together through the pipeline.
Staging	This last test stage (before production) is used for final validation or “staging” (5) before you promote the changes to production.
Production (Release)	The final destination for your changes (1). The org for this stage is your live Salesforce instance.

Pipeline Stage Branches Must Be Unique Across Projects

You can use the same source control repository across projects. However, branches must be unique for each pipeline stage, except for the release branch (often called main), to ensure that conflicts don’t get introduced and changes don’t get overwritten.

Pipeline Configuration Options

Building and activating the pipeline are the last steps to complete configuration of your DevOps Center project.

Changes move through the pipeline when team members promote work items or work item bundles (a versioned group of changes that get promoted together). Upon promotion, changes are merged from the current stage branch (or feature branch) to the next stage branch, and then are deployed to the next stage org.

You can configure your pipeline in one of two ways:

- Allow team members to move work items individually through the entire pipeline.
- Allow team members to move work items individually in early stages of the pipeline, and as a versioned group of changes (work item bundle) in later stages. Keep reading to learn all about the benefits of work item bundles.

Decide How to Build Your Pipeline

After you define development environments and a release (production) org, you have a basic pipeline. However, it's not recommended that you deploy directly to production. You can use our template to build the pipeline, you can add more stages to the pipeline template, or you can build your own.

Best Practices When Naming Branches

Each pipeline stage has an associated branch in the source control repository, which contains the changes associated with your team's work items. When building your pipeline, you can specify either an existing branch in the repository, or allow DevOps Center to create one for you. If you allow DevOps Center to create it for you, indicate a unique alphanumeric string.

Build Your Pipeline Using the Template

When you specified the release target (production) environment, it automatically became the last pipeline stage. DevOps Center also associates it with the default branch in the source control repository, which is often called `main` by default. We don't recommend that you edit the branch for this final pipeline stage.

Build Your Own Pipeline

Quickly build your own pipeline by modifying the existing template pipeline.

Activate the Pipeline

A pipeline is ready for your team to use after you activate it.

What's The Difference Between the Stages and Environments Tabs?

After the pipeline is activated, the Stages tab appears as a first tab on the Pipeline page.

To Bundle or Not to Bundle, That's a Great Question

The process and mechanism for promoting changes from one stage to the next can vary as you move from "left to right" in the pipeline. In the earlier (left) stages of the pipeline, you often want more flexibility to promote individual work items from one stage to the next. As you move to the later (right) stages of the pipeline, it's often desirable to have more predictability and ability to version the sets of changes that are promoted and ultimately released.

See Also

To Bundle or Not to Bundle, That's a Great Question
Salesforce Help: Promote Work Items Through Your Pipeline
Apex Developer Guide: Testing and Code Coverage

Decide How to Build Your Pipeline

After you define development environments and a release (production) org, you have a basic pipeline. However, it's not recommended that you deploy directly to production. You can use our template to build the pipeline, you can add more stages to the pipeline template, or you can build your own.

While you're working on building the pipeline, your changes persist but don't get saved to the project until you activate the pipeline.

Complete the creation of the environments associated with each pipeline stage.

Important Although you can use the same source control repository across DevOps Center projects, be sure the branches for each project are unique. If you re-use branches across projects, you risk overwriting changes and introducing conflicts.

1. Access the Pipeline page.
   a. From within the project, click **Pipeline**.
   b. From the All Projects page, click **Click to Create Pipeline**.

   In the Pipeline page, you see the pre-defined pipeline template.

2. Before you begin, it's important to understand how to properly specify the branch for each pipeline stage.

   We recommend that you don't share branches across projects.

3. Complete your pipeline configuration using the preferred method.
   - Use the provided template to build your pipeline.
   - Build your own pipeline configuration.

For best results:

- Stage names can't contain backslashes or double quotes.
- Follow recommended practices when naming the branches associated with each pipeline stage.
- If you're using the same source control repository as another DevOps Center project, branches must be unique for each pipeline stage, except for the release branch (often called main). DevOps Center can't track changes across projects so unique branches ensure that conflicts don't get introduced and changes don't get overwritten.

See Also

Build Your Pipeline Using the Template
Build Your Own Pipeline

Best Practices When Naming Branches

Best Practices When Naming Branches

Each pipeline stage has an associated branch in the source control repository, which contains the changes associated with your team's work items. When building your pipeline, you can specify either an existing branch in the repository, or allow DevOps Center to create one for you. If you allow DevOps Center to create it for you, indicate a unique alphanumeric string.

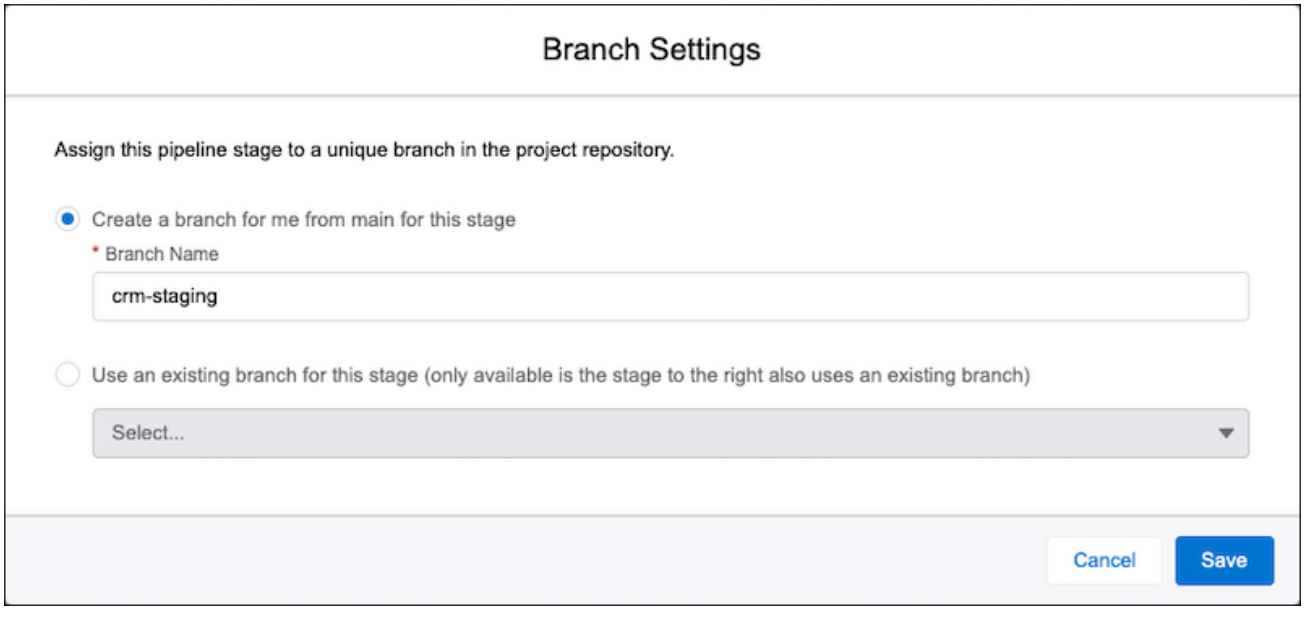
We also recommend that branch names use:

- All lowercase letters
- 60 characters or fewer
- Hyphens or underscores as separators (no spaces)
- Both letters and numbers, but not all numbers

Important If you share branches across projects you can experience unexpected behavior particularly if changes are made to the branches outside of DevOps Center.

A branch must be sourced from the next stage's branch (right to left). For example, let's use our template pipeline structure to clarify what we mean. If your release environment's branch is main, the branch for the pipeline stage to the left of it, `staging`, must be created from `main`. The branch for the pipeline stage to the left of Staging, called `uat`, must be created from the `staging`.

We handle this process for you if DevOps Center creates the branch. In this example, we're creating a new branch for Staging from main. You provide a unique branch name, for example, `crm-staging`.



Build Your Pipeline Using the Template

When you specified the release target (production) environment, it automatically became the last pipeline stage. DevOps Center also associates it with the default branch in the source control repository, which is often called `main` by default. We don't recommend that you edit the branch for this final pipeline stage.

Important Although you can use the same source control repository across DevOps Center projects, be sure the branches for each project are unique. If you re-use branches across projects, you risk overwriting changes and introducing conflicts.

1. From the Pipeline page, start with the Staging stage.
2. Enter the **Stage Name**.

   Stage names can't contain backslashes or double quotes.

3. Click **Connect Environment** to log into the environment associated with this stage.
4. Next, click **Specify Branch** to indicate the branch associated with this stage.

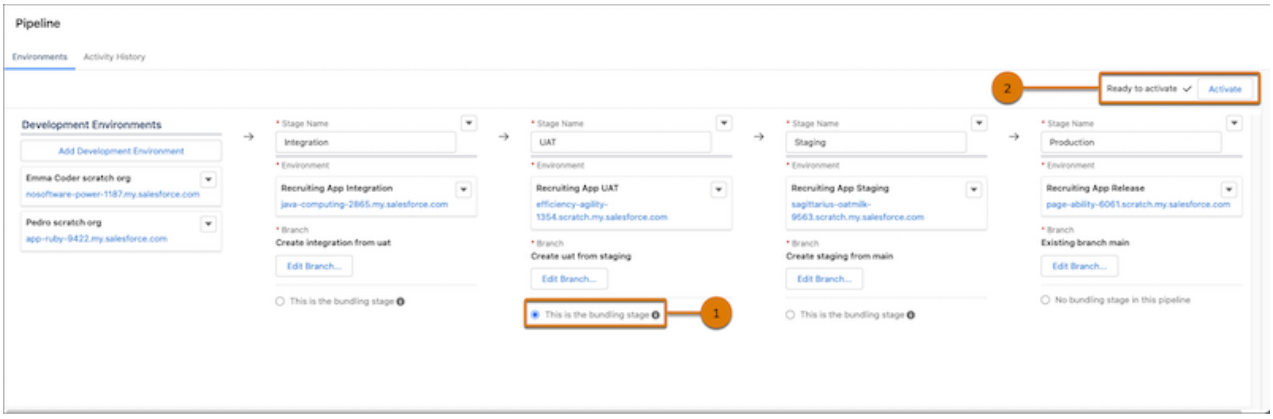
   You can create a branch for this stage, or select an existing one. If you're using the same source control repository as another DevOps Center project, branches must be unique for each pipeline stage, except for the release branch (often called main). Select an existing branch only if it isn't being used in another DevOps Center project. DevOps Center can't track changes across projects so unique branches ensure that conflicts don't get introduced and changes don't get overwritten.

   The Branch Settings dialog can display a maximum of 500 branches. If you don't see the branch you want to select in the dropdown, delete any obsolete branches in the source control repository and try again.

   When you promote changes to this stage, the changes are merged to this branch, and DevOps Center knows exactly what to deploy to the associated pipeline environment.

5. Indicate which stage is the bundling stage, the stage where changes are grouped, versioned, and promoted together in downstream stages (1).

   In the template, the UAT stage is the bundling stage. The items you promote from integration are released together in a work item bundle in the next stage, UAT, and for all future stages.



See To Bundle or Not to Bundle, That's a Great Question for why we recommend a bundling stage.

6. Repeat this process for the rest of the stages.
7. Activate the Pipeline (2).

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

Build Your Own Pipeline

Quickly build your own pipeline by modifying the existing template pipeline.

- From the Stage dropdown, move stages left or right, or remove a stage.
- Change the stage name.

Important Although you can use the same source control repository across DevOps Center projects, be sure the branches for each project are unique. If you re-use branches across projects, you risk overwriting changes and introducing conflicts.

1. Starting with the last stage before your release stage, enter the **Stage Name**.

   Stage names can't contain backslashes or double quotes.

2. Click **Connect Environment** to log into the environment associated with this stage.
3. Next, click **Specify Branch** to indicate the branch associated with this stage.

   You can create a branch for this stage, or select an existing one. If you're using the same source control repository as another DevOps Center project, branches must be unique for each pipeline stage, except for the release branch (often called main). Select an existing branch only if it isn't being used in another DevOps Center project. DevOps Center can't track changes across projects so unique branches ensure that conflicts don't get introduced and changes don't get overwritten.

   The Branch Settings dialog can display a maximum of 100 branches. If you don't see the branch you want to select in the dropdown, delete any obsolete branches in the source control repository and try again.

When you promote changes to this stage, the changes are merged to this branch, and DevOps Center knows exactly what to deploy to the associated pipeline environment.

4. Repeat for the rest of the stages.
5. Indicate which stage is the bundling stage, the stage where changes are grouped, versioned, and promoted together in downstream stages.

   See To Bundle or Not to Bundle, That's a Great Question for why we recommended a bundling stage.

6. Activate the Pipeline.

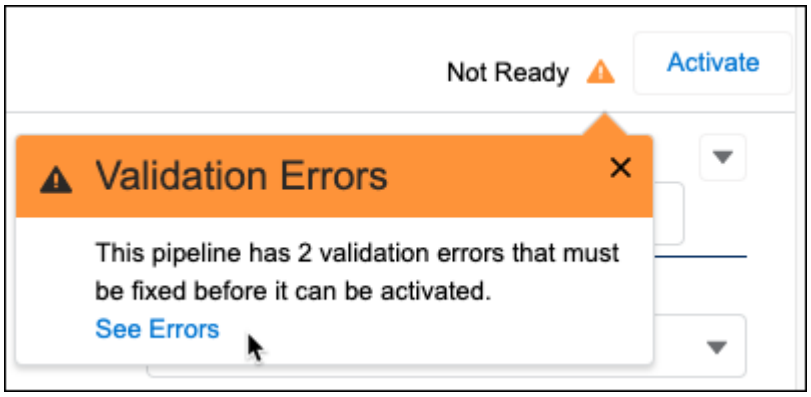
Activate the Pipeline

A pipeline is ready for your team to use after you activate it.

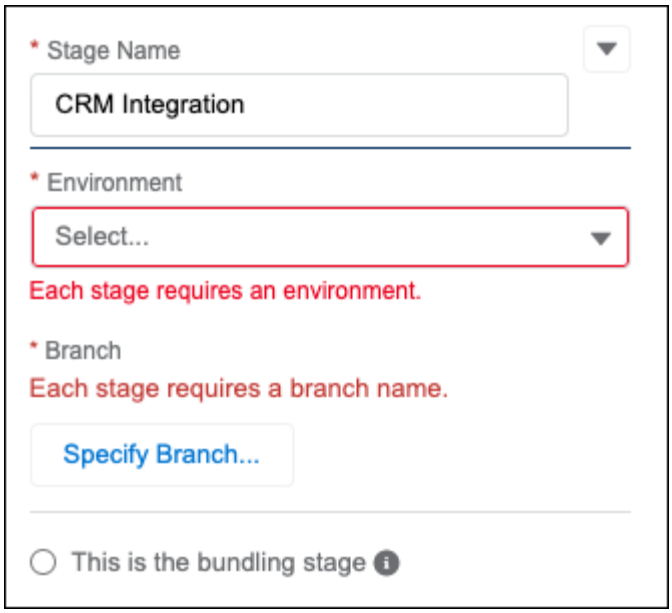
Here's some important information about pipeline activation:

- You and your team can create work items after a pipeline is activated.
- You can't inactivate a pipeline after changes have been promoted through it.
- You can edit only inactive pipelines. If you haven't yet promoted changes, you can deactivate it, make the changes, and then reactivate it.
- You can remove pipeline environments only if the pipeline is inactive.

You can activate the pipeline if the status is `Ready to activate`. If there are validation errors with the pipeline, a `Not Ready` warning appears next to the Activate button with a link to see the errors to resolve.



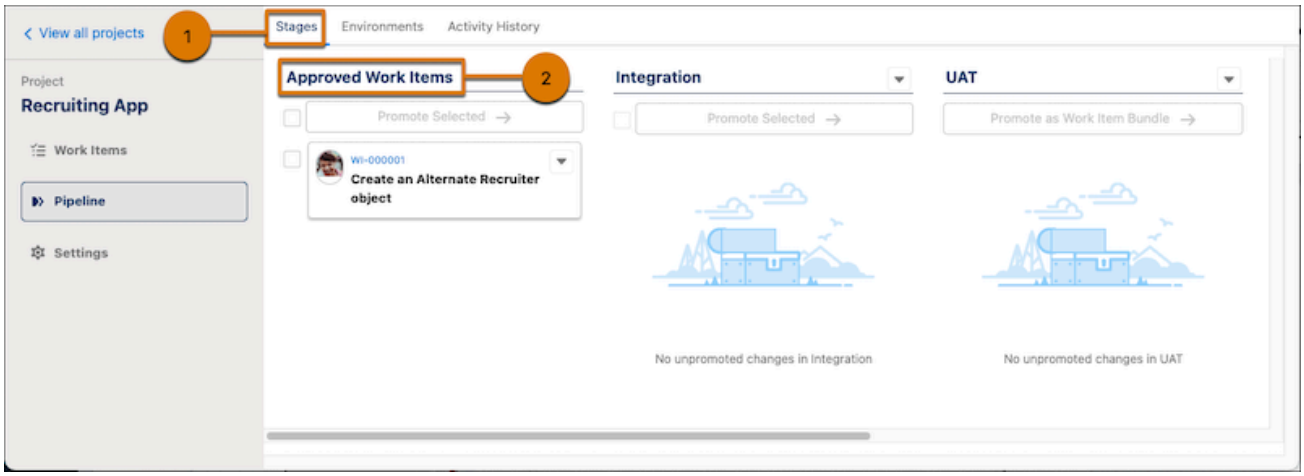
The errors appear in context, so you know exactly what to address.



What's The Difference Between the Stages and Environments Tabs?

After the pipeline is activated, the Stages tab appears as a first tab on the Pipeline page.

The Stages tab (1) is where you promote work items through the pipeline. Work items appear in the Approved Work Items column (2) after they have been marked `Ready to Promote`.

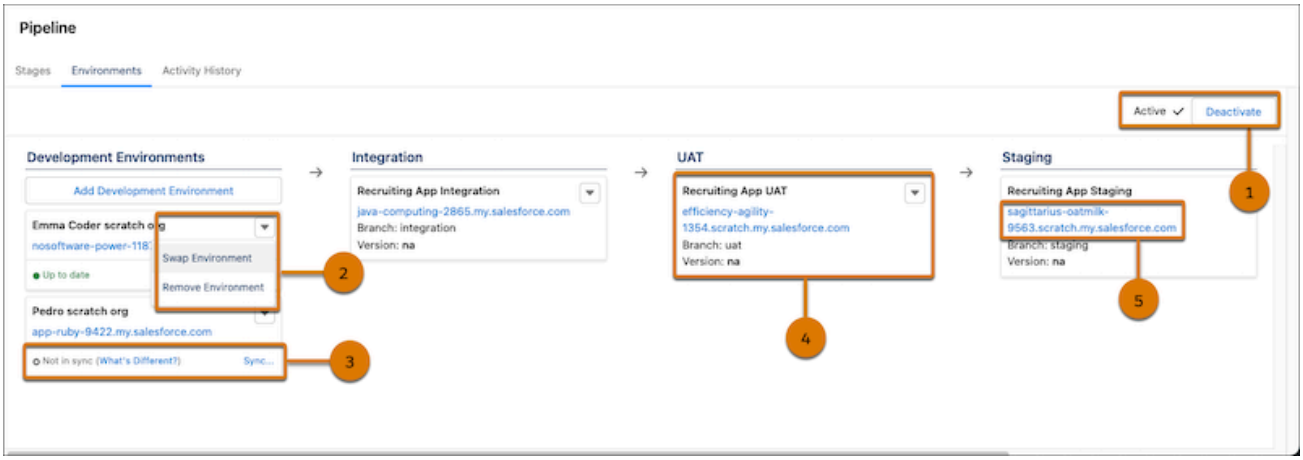


In the Environments tab, you can:

- Continue to configure the pipeline stages, if the pipeline is not active.
- Deactivate your pipeline (1), if you haven't promoted any work items. After a work item has been promoted, you can no longer deactivate it.
- Remove or swap an environment (2). You can remove a dev environment only if changes haven't been committed in a work item. You can remove a pipeline environment only if the pipeline isn't activated.
- Initiate a synchronization process if a development environment is out sync with the next pipeline

stage (3). Available after pipeline is activated.

- View version information about each stage (4). This information provides a quick visual comparison regarding what version exists in each pipeline stage relative to the other stages. In the example, the version for each pipeline is “na” because nothing has been promoted yet.
- Access environments directly with a click (5).



See Also

Create and Assign Project Work Items

To Bundle or Not to Bundle, That’s a Great Question

The process and mechanism for promoting changes from one stage to the next can vary as you move from “left to right” in the pipeline. In the earlier (left) stages of the pipeline, you often want more flexibility to promote individual work items from one stage to the next. As you move to the later (right) stages of the pipeline, it’s often desirable to have more predictability and ability to version the sets of changes that are promoted and ultimately released.

DevOps Center allows you to define what this overall model of promotion looks like. 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*. The bundling stage is where changes come together to be bundled into a *work item bundle* that can be versioned and promoted as a unit through the subsequent stages. When changes are promoted from the bundling stage to the next stage, all work items that have not yet been promoted are included in the versioned work item bundle and promoted as a unit. This versioned bundle continues to be promoted as a consistent unit through subsequent stages when you perform a promotion.

The bundling stage is defined when you configure the pipeline. All stages to the left of this stage allow for individual work item promotion, and all stages to the right of this stage allow for versioned work item bundle promotion.

Work Item Bundles Reduce Merge Conflicts

In the stages to the left of the bundling stage, you have increased flexibility regarding which work items

can be promoted and when. However, this flexibility comes with the tradeoff of increased risk of conflicts and unanticipated behavior because the combination of changes in each stage can be inconsistent. The versioned work item bundle that is created in the bundling stage and promoted through subsequent stages provides improved consistency because the changes contained in it have been merged into a unit that you can promote consistently and predictably from stage to stage.

See Also

Salesforce Help: Conflict Detection and Resolution

Create and Assign Project Work Items

Create work items so that when your team members open DevOps Center for the first time, project work is already identified and assigned to them.

Before you can create and assign work items, you must create and active your project pipeline.

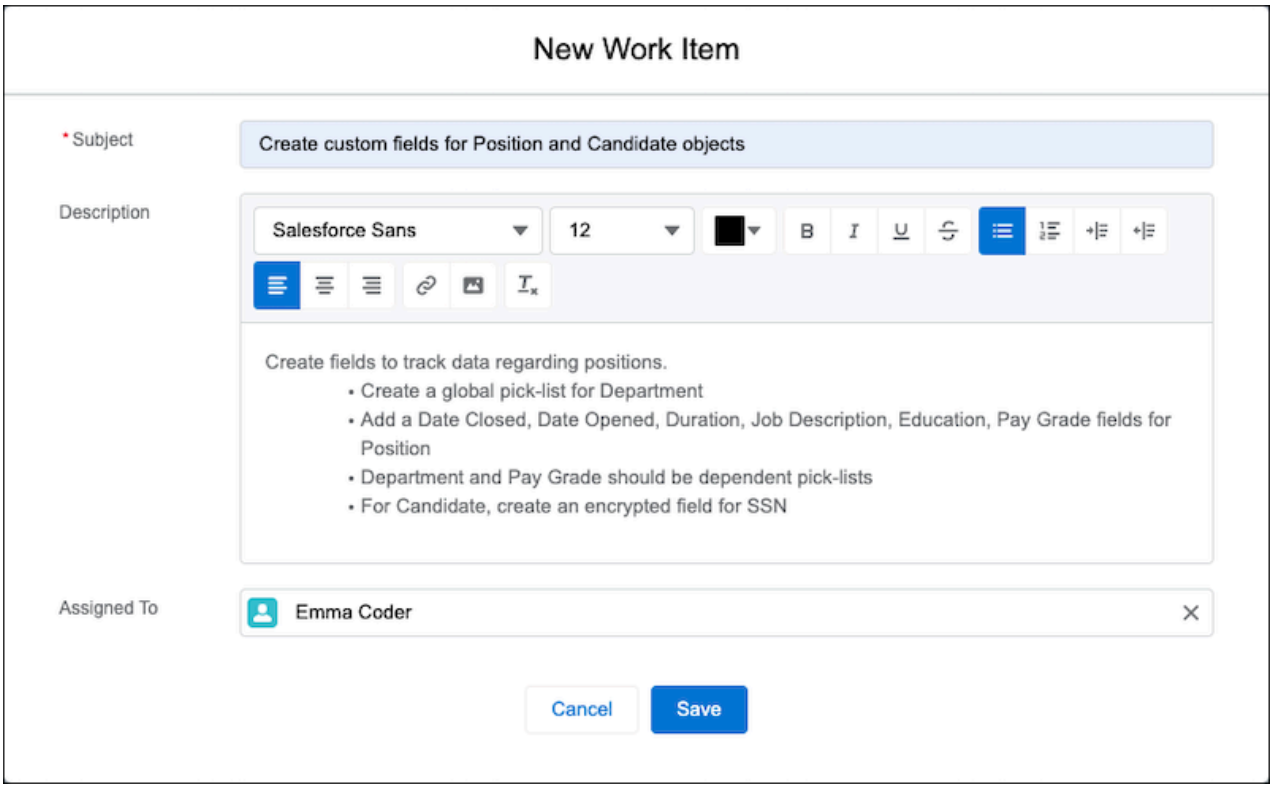
In DevOps Center, a team uses work items to track the progress of changes created to achieve a specific objective, such as enabling a user story or addressing a bug. Work items help a team manage a release by making it easier to identify the status and manage the progress of related changes.

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.

1. From the Projects page, click the name of the project for which you're creating work items.
2. From the Work Items tab, click **New Work Item**.
3. Specify the objective or the problem to be addressed in the Subject field.
4. Use the Description field to provide additional information, including details about the changes and acceptance criteria.

   We use the first 255 characters of the description to help identify changes for this work item in the source control repository.

5. (Optional) Assign the work item to a team member.



6. Click **Save**.

   The work item is displayed in the Work Items tab.

7. Repeat this procedure as needed to track and assign project work.

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

Now that you've set up a DevOps Center, it's time to announce that DevOps Center is ready to use. See Manage and Release Changes Easily and Collaboratively with DevOps Center in Salesforce Help to get started.

See Also

Salesforce Help: Basic Development Workflow
Salesforce Help: Open a Work Item

Build an Extension Package for DevOps Center

Salesforce partners and ISVs can use scratch orgs as their dev environments when building managed second-generation (2GP) extension packages that expand the functionality of DevOps Center. An extension is any package, component, or set of components that adds to the functionality of a managed package. An extension requires that the base managed package is installed in the org.

REQUIRED EDITIONS

USER PERMISSIONS NEEDED	
To install and configure DevOps Center:	“Download AppExchange Packages”
To create packages:	“Create AppExchange Packages”
To upload packages:	“Upload AppExchange Packages”

Use this high-level work flow as a guide for using a scratch org as your development environment when building an extension package.

1. Enable DevOps Center in your Dev Hub org.
2. Update the scratch org definition file to enable DevOps Center, and then use this definition file when creating scratch orgs and package versions.

```
{
    "orgName": "Acme",
    "edition": "Enterprise",
    "features": ["DevOpsCenter"],
    "settings": {
        "devHubSettings": {
            "enableDevOpsCenterGA": true
            }
        }
    }
```

   If creating a scratch org based on an org shape, you still have to include the DevOps Center feature and settings in the scratch org definition file for legal reasons as part of the DevOps Center terms and conditions.

```
{
    "orgName": "Acme",
    "sourceOrg": "00DB1230400Ifx5",
    "features": ["DevOpsCenter"],
    "settings": {
        "devHubSettings": {
            "enableDevOpsCenterGA": true
            }
        }
    }
```

3. Update the `sfdx_project.json` file to include the DevOps Center base package as a dependency.

   You can also reference the definition file or indicate it on the command line later during package version creation. See the example files in Project Configuration File for a Second-Generation Managed

Package in the *Second-Generation Managed Packaging Developer Guide*.

4. In the scratch org, from the DevOps Center Setup page, install the DevOps Center package.

   To use Salesforce CLI to install the package in additional scratch orgs, note the package ID, which starts with `04t`.

5. After you have finished developing your extension package, use Salesforce CLI to create a package.
6. When you're ready to test or release the package, create a package version.

   If you didn't include the definition file in the `sfdx_project.json` project file, be sure to indicate it on the command line. For example:

```
sf package version create --package "Agile Addon " --definition-file config/
project-scratch-def.json \
--installation-key test1234 --wait 10
```

See Also

DevOps Center Developer Guide
First-Generation Managed Packaging Developer Guide: Publishing Extensions to Managed Packages
Salesforce DX Developer Guide: Scratch Orgs
Second-Generation Managed Packaging Developer Guide: Second-Generation Managed Packages

Uninstall DevOps Center

You can uninstall DevOps Center, if necessary. If you plan to reinstall DevOps Center, don't delete the DevOps Center auth providers. However, if you deleted the auth providers, you can recreate them by re-enabling the DevOps Center preference in Setup.

Important Uninstalling DevOps Center also removes all associated data including, but not limited to, work items, pipeline configuration, and activity history.

It's important to follow the tasks in order to properly uninstall the DevOps Center package. You must remove some configuration dependencies before you can successfully uninstall the package.

Remove Permission Set Assignments

Before you uninstall the DevOps Center package, remove permission set assignments, then delete DevOps Center permissions sets.

Uninstall the DevOps Center Package

After you remove the permission set and permission set assignments, you can uninstall the DevOps Center package, which removes the application and all its associated data and objects.

Delete DevOps Center Named Credentials

Remove any named credentials associated with DevOps Center, which manage authentication to your development and pipeline environments.

Delete DevOps Center Connected App

Delete the DevOps Center connected app so that the app doesn't appear on App Launcher.

Delete DevOps Center Authentication Providers

Delete the auth providers only if you don't plan to reinstall DevOps Center. After you're done with your evaluation, you can delete the DevOps Center auth providers as the last step to remove all installation artifacts from your org. But proceed with caution. You can't manually recreate them if you change your mind.

Disable DevOps Center Preference

When DevOps Center was installed, a Salesforce admin first enabled DevOps Center. To ensure that DevOps Center can't be installed or used, you can disable the DevOps Center preference.

See Also

Salesforce Security Guide: Remove User Assignments from a Permission Set

Remove Permission Set Assignments

Before you uninstall the DevOps Center package, remove permission set assignments, then delete DevOps Center permissions sets.

First, remove the DevOps Center permission set assignments from all DevOps Center users.

- DevOps Center
- DevOps Center Manager
- DevOps Center Release Manager
- sf_devops_InitializeEnvironments
- sf_devops_NamedCredentials

Next, delete the DevOps Center permission sets.

1. From Setup, enter *Permission Sets* in the Quick Find box, then select **Permission Sets**.
2. Delete these two permission sets: `sf_devops_InitializeEnvironments` and `sf_devops_NamedCredentials`.

Uninstall the DevOps Center Package

After you remove the permission set and permission set assignments, you can uninstall the DevOps Center package, which removes the application and all its associated data and objects.

Before you uninstall the package:

- Remove DevOps Center permission Set assignments.
- Delete DevOps Center permission sets.

1. From Setup, enter *Installed Packages* in the Quick Find box, then select **Installed Packages**.
2. Click **Uninstall**, then scroll to the bottom of the page to choose whether to save and export a copy of

the package's data.
3. Select **Yes, I want to uninstall this package and permanently delete all associated components**.
4. Click **Uninstall**.

Delete DevOps Center Named Credentials

Remove any named credentials associated with DevOps Center, which manage authentication to your development and pipeline environments.

1. From Setup, enter *Named Credentials* in the Quick Find box, then select **Named Credentials**.
2. Delete any named credentials associated with DevOps Center environments.

   Look for the format `<ID>_<environment-name>_<number>`.

Delete DevOps Center Connected App

Delete the DevOps Center connected app so that the app doesn't appear on App Launcher.

1. From Setup, enter App Manager in the Quick Find box, then select **App Manager**.
2. From the dropdown menu for DevOps Center, select **View**.
3. Click **Delete**.
4. Confirm that you want to delete the connected app.

Delete DevOps Center Authentication Providers

Delete the auth providers only if you don't plan to reinstall DevOps Center. After you're done with your evaluation, you can delete the DevOps Center auth providers as the last step to remove all installation artifacts from your org. But proceed with caution. You can't manually recreate them if you change your mind.

Important If you plan to reinstall DevOps Center, don't delete the DevOps Center auth providers and skip this task. We created the auth providers when you enabled the DevOps Center preference in Setup. However, if you delete the auth providers, you can recreate them by re-enabling the DevOps Center preference. If it's still enabled, just disable then re-enable it.

1. From Setup, find and select **Auth. Providers**.
2. Delete the DevOps center authentication providers.
   - DevOps Center Bitbucket
   - DevOps Center GitHub
   - DevOps Center Prod
   - DevOps Center Test

See Also

Enable and Install DevOps Center in the Org

Disable DevOps Center Preference

Disable DevOps Center Preference

When DevOps Center was installed, a Salesforce admin first enabled DevOps Center. To ensure that DevOps Center can't be installed or used, you can disable the DevOps Center preference.

1. From Setup, enter *DevOps Center* in the Quick Find box, then select **DevOps Center**.
2. Disable **Enable DevOps Center**.

You no longer can install or upgrade the DevOps Center app, and users see `DevOps Center has not been enabled for your org` when they try to use it.

Troubleshoot DevOps Center Configuration

Here are some tips if you encounter issues when installing or configuring DevOps Center.

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.

Issue: Auto-generated work item numbers aren't consecutive

Sometimes, auto-generated record numbering, including work item numbers, aren't sequential or consecutive.

Cause: Record numbers can skip when multiple operations are running concurrently.

Action: No action required. See Knowledge Article: Auto numbers that are generated are not sequential all the time for more information.

Issue: Can't add or log in to a sandbox

When attempting to add a sandbox as a development or pipeline environment, or to log in to a sandbox, DevOps Center launches `login.salesforce.com` so you can't log in to it.

Cause: Often, the cause is that the required auth provider, DevOps Center Test, has been deleted.

Action: Toggle the DevOps Center Setup option to recreate all DevOps Center auth providers.

- In Setup, find and select **DevOps Center**.
- Click to disable DevOps Center and then click again to enable it.

- In Setup, find and select **Auth. Providers**.
- Verify that the list contains DevOps Center auth providers, including `DevOps Center Test`.

After you re-add the DevOps Center Test auth provider, you're directed to the expected login URL, `test.salesforce.com`.

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.

Issue: Can't connect a release environment to a project or perform common operations

You see this error, `Error: There was an error. Please reload the page`, when you try to connect a release environment to a project.

Cause: If you have the API Access Control Setting `For admin-approved users, limit API access to only allowlisted connected apps` enabled, it blocks DevOps Center from interacting with required APIs. The behavior you see depends on if it was enabled before or after you installed and configured DevOps Center.

- If this setting was enabled before DevOps Center was installed, you can't add a release environment to a project.
- If this setting is enabled after DevOps Center was installed, you can't perform common operations, such as commits and promotions.

Action: Disable the setting. From Setup, in the Quick Find box, enter *`API Access Control`*, then select **API Access Control**.

Issue: DevOps Center doesn't appear in the App Launcher

Cause: If you miss some of the post-installation configuration steps, DevOps Center doesn't appear in the App Launcher.

Action: Try these troubleshooting steps in order:

- Did you create the connected app? Be sure to complete all steps.
- If you created the connected app, did you assign the `sf_devops_NamedCredentials` permission set to it?
- Does the DevOps Center app Start URL appear in App Manager?

- Go to Setup > App Manager.
- Find DevOps Center and select **Manage** from the menu.
- Click **Edit Policies**.
- In the Start URL field, enter `/sf_devops/DevOpsCenter.app`, then click **Save**.

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**.

Issue: You don't see an existing branch in the dropdown when building your pipeline

Cause: When using an existing source control repository for the DevOps Center project, you don't see the branch you want to select in the dropdown when building the pipeline. The Branch Settings dialog dropdown can display a maximum of 500 branches.

Action: As a workaround, delete any obsolete branches in the source control repository and try again.

Issue: Logged in to GitHub using the incorrect account

When creating or opening a project, all team members log in to GitHub using the account that has access to the repos that your team wants to use with DevOps Center. If someone logs in with the incorrect GitHub account, a team member with Salesforce admin privileges can delete the authorization information so you or the team member can log in using the proper credentials.

Action: See Connect to a Different GitHub Account for instructions.

Error: Did You Log In to the Correct Environment?

Cause: DevOps Center acts like you didn't log into the correct sandbox. In Winter '24, Salesforce enforced 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. If you set up DevOps Center when enhanced domains weren't deployed, the named credential requires an update.

This issue surfaces when a DevOps Center user tries 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: Users with Salesforce admin privileges or the “Customize Application” user permission can update named credentials.

- From Setup, in the Quick Find box, enter `Named Credentials` in the Quick Find box, then select **Named Credentials**.
- Locate the named credential records that correspond to sandboxes used as development environments or in pipeline stages.
  Look for the environment’s DevOps Center record ID in the Label field. DevOps Center uses this pattern for creating Named Credential records:
  Release environments:
  environmentRecord ID_projectName_environmentName_sequentialNumber
  Other environments:
  environmentRecord ID_environmentName_sequentialNumber
- From the Actions menu, select **Edit**.
- In the URL field, replace your old My Domain login URL with your new My Domain login URL.
  For example, replace:

  ```
  https://MyDomainName--SandboxName.my.salesforce.com/
  ```

  with

  ```
  https://MyDomainName--SandboxName.sandbox.my.salesforce.com
  ```

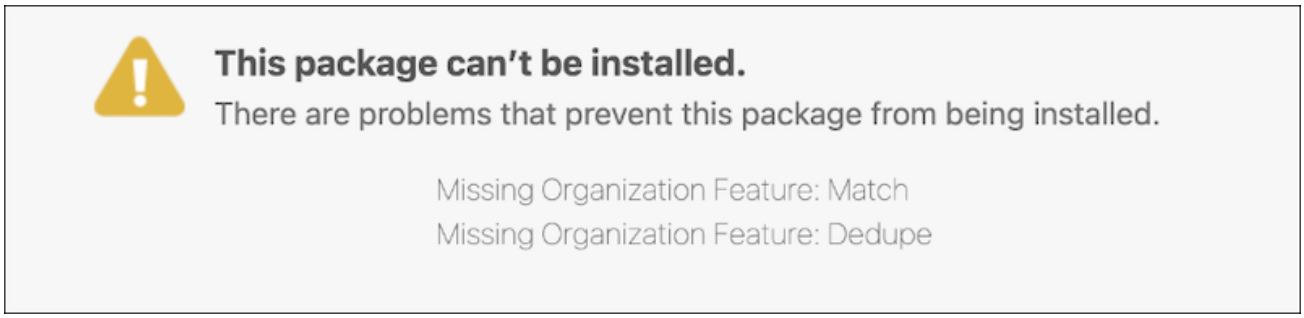
- Deselect **Start Authentication Flow on Save** and save your changes.

Issue: DevOps Center Setup page doesn't appear in the org

Cause: DevOps Center is available in Enterprise, Performance, Professional, Unlimited, and Developer editions, as well as scratch orgs and Trailhead playgrounds. It's not available in sandboxes or other unlisted org editions.

Action: To evaluate DevOps Center, we suggest using a Developer Edition org or scratch org. Scratch orgs don't live forever, so we suggest indicating a duration of 30 days so you have ample time to try out DevOps Center.

Error: Package can't be installed because Duplicate Rules and Matching Rules features are missing.



Cause: DevOps Center is available in Enterprise, Performance, Professional, Unlimited, and Developer editions, as well as scratch orgs and Trailhead playgrounds. It's not available in sandboxes or other unlisted org editions.

Action: To evaluate DevOps Center, we suggest using a Developer Edition org or scratch org. Scratch orgs don't live forever, so we suggest indicating a duration of 30 days so you have ample time to try out DevOps Center.

Error: “You don’t have permission to view this data, enter credentials for the named credential in your personal settings. Or ask your Salesforce admin for help.”

Cause: After the DevOps Center package is upgraded to the current version, a user sees this error when they try to open a project. This error is likely to occur when a user isn't logged into GitHub or doesn't have access to the project’s GitHub repo.

Action: To work around this issue, click **New Project**, which prompts you to log in to GitHub.

Error: “(DevOps_Center_GitHub) In field: authProvider - no AuthProvider named DEVOPS_CENTER_GITHUB found DevOps_Center_GitHub: In field: authProvider - no AuthProvider named DEVOPS_CENTER_GITHUB found”

Cause: It’s highly likely that you didn't enable DevOps Center in your org.

Action: Enable DevOps Center and accept the license terms in the DevOps Center Setup page. From Setup, enter `DevOps Center` in the Quick Find box, then select **DevOps Center**.

Error: “The request was invalid. Resource protected by organization SAML enforcement.”

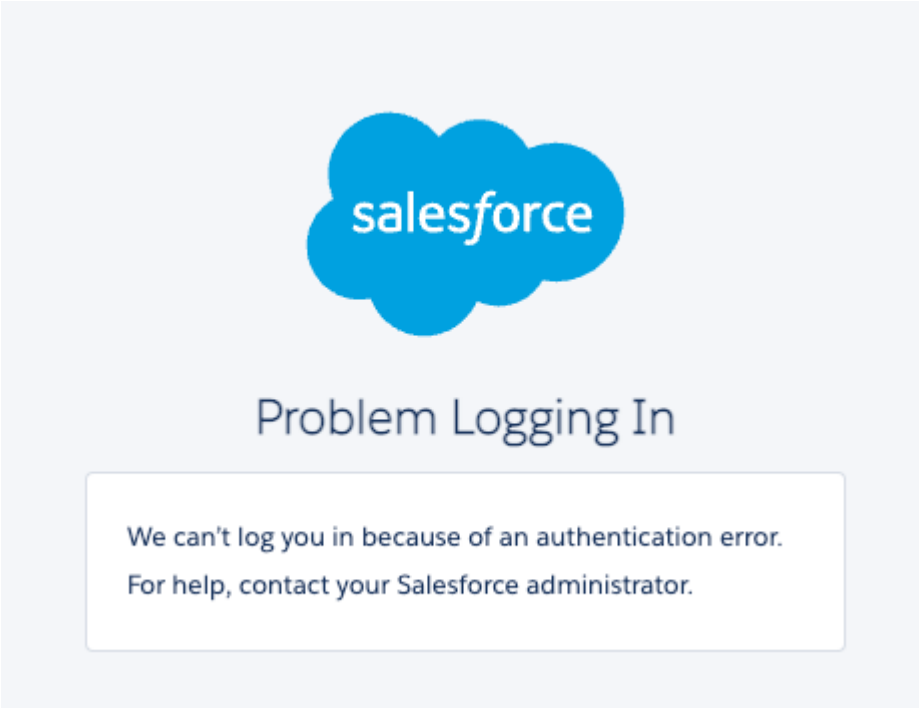
Cause: GitHub repos owned by an organization aren’t visible in DevOps Center until an organization

account owner provides access.

Action: See If an Organization Owns the GitHub Repo for instructions on fixing this issue.

Error: Problem Logging In Due to Login IP Restrictions

Cause: The org or org user attempting to connect has login IP restrictions. DevOps Center uses named credentials and OAuth to facilitate a trusted connection between itself and the other environments. When you attempt to log in, you see an error popup and a URL format that conveys the existence of IP restrictions.



In the address bar, you see a URL that ends with something like this:

```
/AuthorizationError?ErrorCode=No_Oauth_Token&ErrorDescription=invalid_grant+ip+restricted
```

Action: Here are some options to work around this error:

- Create an integration user in the target org, with a profile that doesn't have IP restrictions.
- Add Salesforce IPs and DevOps Center service IPs to the list of trusted IPs in the target user's profile. See Salesforce IP Address and Domains to Allow and Update IP Allow List for DevOps Center App.
- Completely disable IP allow-listing in the target org.

Error: Commit or Promotion Failed Due to API Version Mismatch

Action: To resolve this error, update the source API version. For more information, see Update the

Project's Source API Version Each Salesforce Release.