



Setup Guide

DevOps Center

Pilot, Summer '21



@salesforcedocs

Last updated: July 26, 2021

© Copyright 2000–2021 salesforce.com, inc. All rights reserved. Salesforce is a registered trademark of salesforce.com, inc., as are other names and marks. Other marks appearing herein may be trademarks of their respective owners.



Important: This feature is not generally available and is being piloted with certain Customers subject to additional terms and conditions. It is not part of your purchased Services. This feature is subject to change, may be discontinued with no notice at any time in SFDC's sole discretion, and SFDC may never make this feature generally available. Make your purchase decisions only on the basis of generally available products and features. This feature is made available on an AS IS basis and use of this feature is at your sole risk.

CONTENTS

[Welcome To The DevOps Center Pilot](#)

[A Reminder About Pilot Software](#)

[Updates to DevOps Center](#)

[Setup Workflow](#)

[GitHub Project Repositories](#)

[Create a GitHub Account](#)

[New to GitHub or to Source Control?](#)

[Methods to Create GitHub Project Repositories](#)

[Create a New Repository](#)

[Identify Your Environments](#)

[What's Special About Development Environments?](#)

[Open DevOps Center](#)

[Create a Project](#)

[If an Organization Owns the GitHub Repo](#)

[Connect to the Release Environment](#)

[Add Development Environments](#)

[Plan Your Pipeline](#)

[How Many Pipeline Stages Do I Need?](#)

[Add Other Pipeline Environments](#)

[Pipeline Configuration Options](#)

[Build Your Pipeline](#)

[What You Need to Know About Branches](#)

[Build Your Pipeline Using the Template](#)

[Build Your Own Pipeline](#)

[Activate The Pipeline](#)

[What's The Difference Between the Environments and Stages Tabs?](#)

[To Bundle or Not to Bundle, That's a Great Question](#)

[Work Item Bundles Reduce Merge Conflicts](#)

[Set Up Team Members in the DevOps Center Org](#)

[Add Team Members as Repository Collaborators](#)

[Add Team Members as Users in the DevOps Center Org](#)

[Assign the DevOps Center Permission Sets](#)

[DevOps Center Permission Sets](#)

[Assign These Permission Sets to All DevOps Center Users](#)

[Assign Additional Permission Sets to DevOps Center Managers](#)

[Create and Assign Project Work Items](#)

[Next: Bring Team Members into DevOps Center](#)

[Known Issues](#)

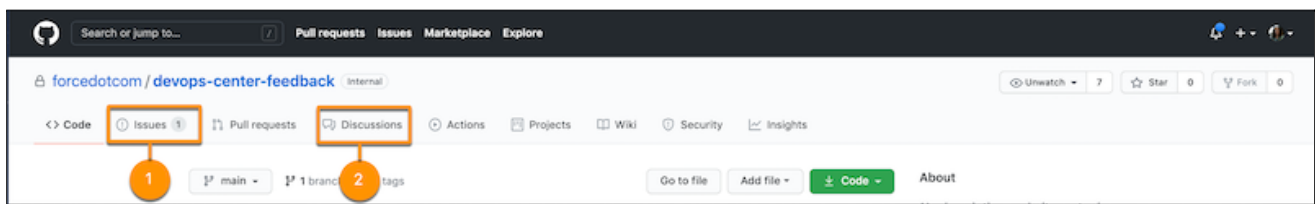
[DevOps Center Can't Find Existing Repo When Creating a Project](#)

Welcome To The DevOps Center Pilot

Congratulations! Your team has been nominated to participate in the DevOps Center pilot. We thank you for taking the time to use DevOps Center and provide feedback. Your input will be instrumental in improving the product and user experience.

Ask questions and post feedback in the [DevOps Center GitHub repo](#). See [readme.md](#) for all the details.

This private shared repository is the central location for sharing information and collecting feedback on the pilot. Use the [Issues tab](#) (1) at the top of the repo to capture bugs, feature requests, and general feedback. Use the [Discussions tab](#) (2) for general discussion or questions. Access to this repository is limited to participants of the pilot and members of the Salesforce DevOps Center product team.



Share [this form](#) with any users on your team who are participating in this pilot so that we can provide them with access to this repository.

A Reminder About Pilot Software

This is a pilot release. The software isn't yet feature complete, and is at an early stage of development. You'll be able to use the product to evaluate how it might work for your team's situation and environment, and to provide us with feedback on the product and your experience with it as we continue through our development process.

Updates to DevOps Center

DevOps Center is delivered as a managed package. We've already installed the package for you in the trial org you've received for pilot. We'll occasionally push updates to the managed package to deliver ongoing improvements to the product. Whenever possible, these updates won't disrupt you. If you need to take any action after an update, we'll reach out to let you know.

Setup Workflow

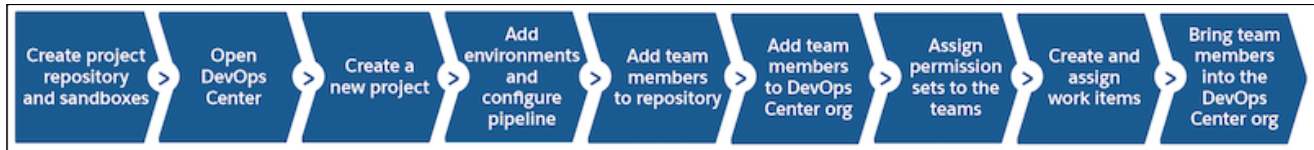
We've added one primary System Administrator to the org and assigned the appropriate permissions. As the administrator, you can add team members, set up projects, add environments, configure your pipeline, and create work items.

In addition to the System Administrator user we've created in your trial org, we include a second Salesforce Admin user. If issues arise that require debugging or configuration changes, we can use this account to log in and troubleshoot, or to make updates to the DevOps Center application.



Warning: Don't change the subdomain name of the DevOps Center org assigned to you.

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



1. [Decide which method you'll use to create a GitHub project repository.](#)
2. [Identify and create your environments.](#)
3. [Open DevOps Center.](#)
4. [Create a project.](#)
5. [Connect to the release environment.](#)
6. [Add development environments.](#)
7. [Add your project team members as collaborators in the project's GitHub repository.](#)
8. [Add your project team members as users in the DevOps Center org](#) that Salesforce created for you.
9. [Create and assign project work items.](#)
10. [Assign the team members the permission sets](#) that allow them to address project work items.
11. [Bring your team members into the DevOps Center org.](#)

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.



Note: 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.

We've designed DevOps Center to eventually integrate with multiple third-party source control systems, such as Bitbucket, GitLab, and GitHub. For the pilot, all participants use GitHub as the source control system for DevOps Center projects. All participants need their own *GitHub-hosted cloud-based GitHub.com* account to work in this version of DevOps Center. Enterprise or locally hosted versions of GitHub aren't currently supported at this time.

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 [Identify Your Environments](#).
- Use an existing Github repository.
- Create a new 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](#).

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.

Identify Your Environments

You'll want to have the login credentials handy for the environments you plan to use for your DevOps Center project. Before you proceed, make sure all the environments you need for this project are created.

First, be sure that Source Tracking for Sandboxes is enabled in any orgs from which you plan to create your developer sandboxes.



Tip: If you have any existing Developer sandboxes that were created before source tracking

was enabled, enable Source Tracking for Sandboxes in the production org, then refresh those sandboxes before proceeding.

Did you...

- 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'll need access to.



Important: Because the product is still in a relatively early stage of development, we advise you to carefully consider how you use it in your production environments for now. As a safeguard, you can create a sandbox to use for this purpose.

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 own production org.

Ideally, everyone contributing customizations to a DevOps Center project is assigned their own Developer sandbox. See your Salesforce Admin about allocating sandboxes for your team's participation in this pilot. For more information about enabling source tracking in sandboxes, see the [Salesforce DX Developer Guide](#).

Open DevOps Center

Before you start: Locate the *Finish Resetting Your Salesforce Password* or *Welcome to Salesforce* email generated when Salesforce added you as a user to the DevOps Center org you're using for Developer Preview.

1. Use the link in the *Finish Resetting Your Salesforce Password* or *Welcome to Salesforce* email you received to reset the password and log in to the DevOps Center org.
2. From the App Launcher, find and select **DevOps Center**.

DevOps Center opens to the Projects page. You've just logged in, so there aren't any projects yet.

Create a Project

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 must use the [Salesforce DX project structure](#), and be associated with a GitHub repository. The repository is used to store project changes.

1. From the DevOps Center Projects page, click **New Project**.

The first time you create a new 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.

Connect DevOps Center and GitHub

GitHub is a version control system that can track changes made to your DevOps Center projects.

Connect to GitHub and authorize DevOps Center to work with your GitHub account. If you don't have a GitHub account, you can create one.

If you're logged in to GitHub in this browser session, click **Connect to GitHub**, so that DevOps Center can get your session information. You won't be prompted to log in but the authorized connection is made.

Cancel
Connect to GitHub

2. 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. Continue to step 3.
Organization account	In pilot, 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.

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

New Project

* Project Name

Enter a unique name.

* Every project in DevOps Center must use the [Salesforce DX project structure](#), and be connected to a unique version control system repository.

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

Repository Owner ⓘ

Repository Name

Provide a repository name that doesn't contain spaces...

Repository Owner is required

Repository Name is required

☐ Use an existing repository for my project. This repository must use the Salesforce DX project structure.

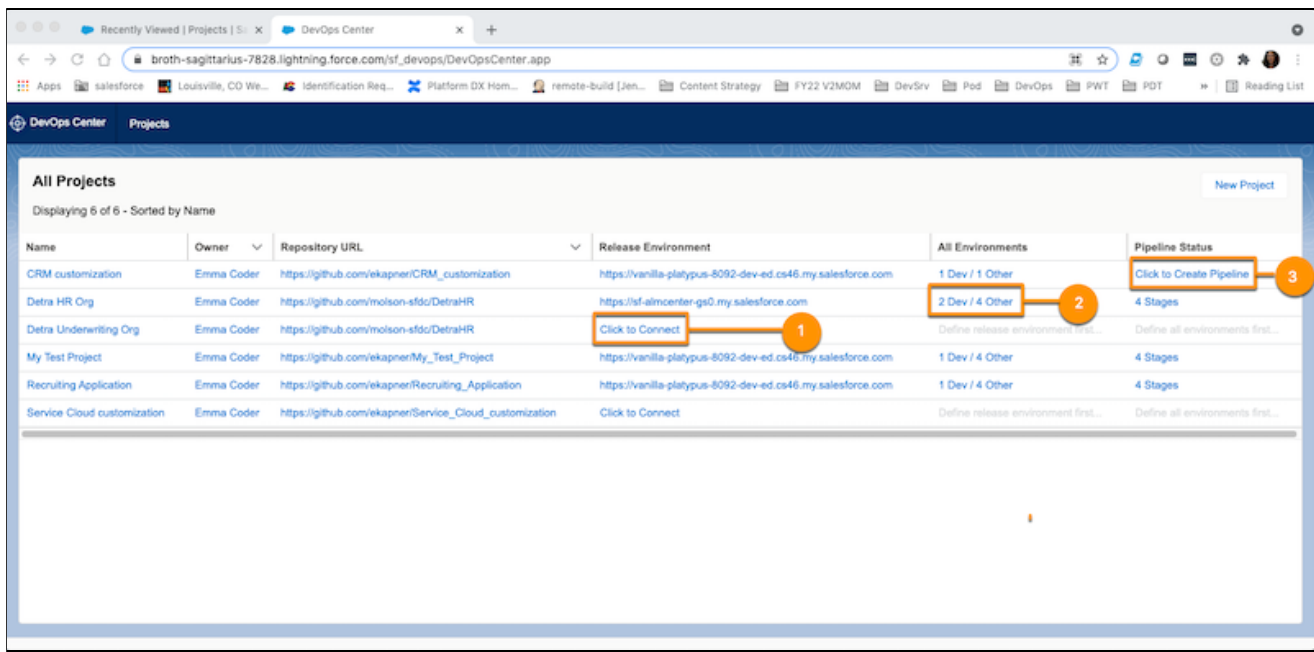
Paste your project repository URL...

Project Description

Cancel Save

4. Enter a unique name for your project.
5. Create a new repository or use an existing one.
 - If you create a new 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>.
6. (Optional) Enter a description to identify the purpose of the project.
7. Click **Save**.

Your project is created and added to the Projects page. The Projects list view helps guide you through the project configuration process. Different projects can be in different phases of project setup.



The project setup workflow is:

- Define and connect to the final release environment (1).
- Add development and other environments (2) that you'll use for this project.
- Configure your pipeline (3).

DevOps Center provides the “hot link” only after you complete each step, so you don’t accidentally jump ahead.

If an Organization Owns the GitHub Repo

Due to a limitation in pilot, 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, in your personal account, go to **Settings**.
 - a. Click **Applications**, then select **Authorized OAuth Apps**.
 - b. Click **Salesforce Integration Application**.
 - c. Find the organization that owns your repo, then click **Request**.
3. Once the GitHub repository owner approves your request for access, open DevOps Center.
 - a. Click your profile icon, then select **Settings**.
 - b. Click **Authentication Settings for External Systems**.
 - c. Delete **DevOps Center GitHub**.
4. Start with step 1 again in [Create a Project](#).

Connect to the Release Environment

At a minimum, projects require a release environment (and one or more developer environments). Be sure all team members are users in this environment if they require access to open this environment and perform deployments to it from DevOps Center.

1. On the All Projects page, under Release Environment, click **Click to Connect**.
2. Provide a unique name for the production environment and indicate the environment type. By default, DevOps Center populates this field with the project name + Release.

Add Release Environment

*** Environment Name**

*** What kind of environment is this?**

☐ Production Org

☒ Sandbox/Scratch Org

Log in to this environment to provide DevOps Center access to it. We'll take you to the standard Salesforce login window.

[Cancel](#) [Log In](#)

3. Click **Log In**.
4. Log in to the environment. After you authenticate successfully, allow DevOps Center to use 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

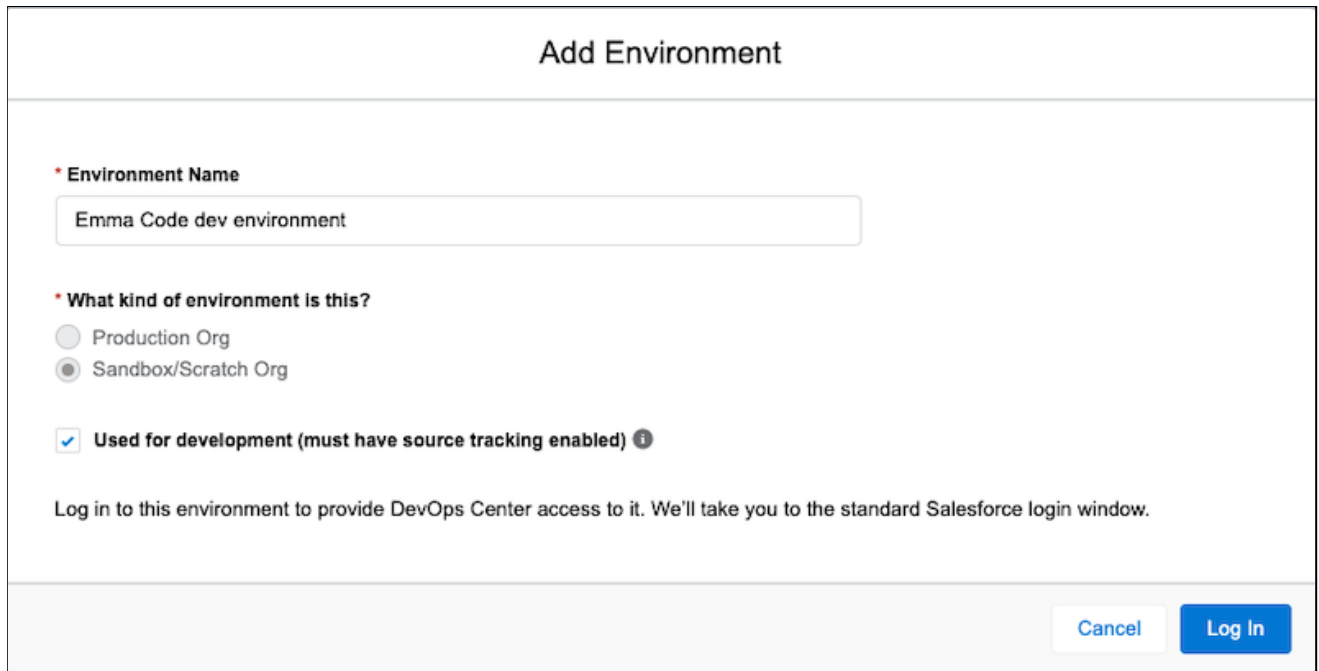
Add the source-tracked development environment that you created for each developer working on the project.

1. In the Project page, click the hot link text under All Environments.

All Projects						New Project
Displaying 3 of 3 - Sorted by Name						
Name	Owner	Repository URL	Release Environment	All Environments	Pipeline Status	
CRM customizations	User User	https://github.com/ekapner/CRM_customizations	https://velocity-connect-9064-dev-ed.cs79.my.salesforce.com	0 Dev / 1 Other	Define all environments first...	

2. In the Settings page, click **Add**.

3. Select **Sandbox/Scratch Org**.
4. Be sure to select the checkbox, **Used for Development**.



The screenshot shows a dialog box titled "Add Environment". It contains the following fields and options:

- * Environment Name**: A text input field containing "Emma Code dev environment".
- * What kind of environment is this?**: Two radio button options: "Production Org" (unselected) and "Sandbox/Scratch Org" (selected).
- ☒ **Used for development (must have source tracking enabled)**: A checked checkbox with an information icon.
- A message: "Log in to this environment to provide DevOps Center access to it. We'll take you to the standard Salesforce login window."
- At the bottom right, there are two buttons: "Cancel" and "Log In".

5. Click **Log In**.

DevOps Center takes you to `test.salesforce.com` to log in. After you authenticate successfully, allow DevOps Center to use an OAuth process to remember your credentials.

6. Repeat steps 2-5 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.

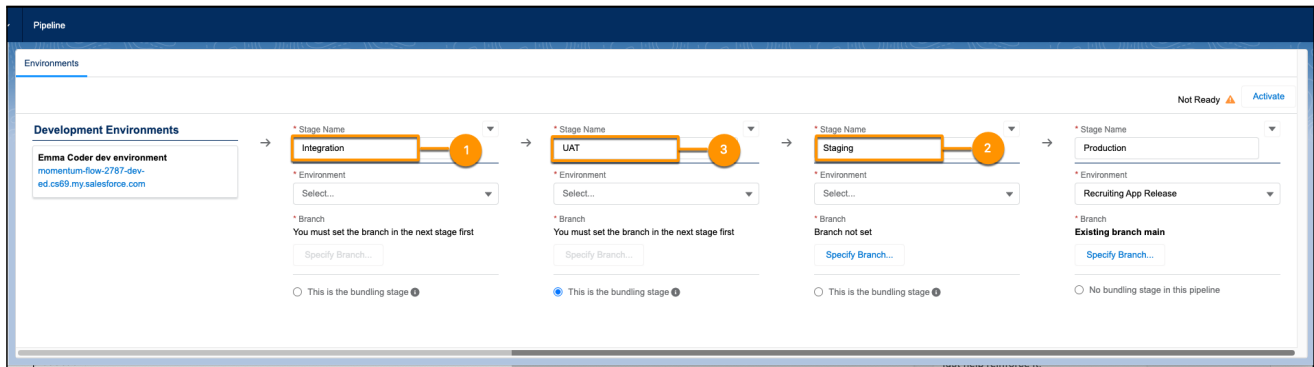
Next step: Determine your pipeline configuration ([how many stages](#)), then [add the other environments](#) associated with your pipeline stages.

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. At a minimum, a pipeline requires a final release environment and one development environment. We recommend that you have at least one test stage in addition to your development and production stages.



To assist you with building a robust pipeline, which typically includes 2-3 test stages, we provide a pipeline template:

- The first test stage (after the development stage) is an integration stage (1) where all changes from the various development environments come together for the first time and can be tested in an integrated environment.
- The final test stage (before production) is used for final validation or “staging” (2) before release to production.
- You can also have one or more stages in between “integration” and “staging” (3) where you can perform additional testing, including by business stakeholders, often called user acceptance testing (UAT).

The configuration of your pipeline is entirely up to you, and is based on the development and business processes that you have in place.

Add Other Pipeline Environments

Once you determine the number of stages in your pipeline, you’re ready to add the environments associated with each pipeline stage that occurs before the final release stage. The final release stage is often your production org, which is already connected. If team members require access to open the environments and perform deployments (promotions) to them using DevOps Center, be sure these team members are users in the environments.



Tip: You can add a new environment at any time. However, to expedite building your pipeline, we recommend defining all pipeline environments first before building the pipeline.

1. From the Setting page, click **Add**.

Add Environment

* **Environment Name**

User Acceptance Testing

* **What kind of environment is this?**

☐ Production Org

☒ Sandbox/Scratch Org

☐ **Used for development (must have source tracking enabled)** ⓘ

Log in to this environment to provide DevOps Center access to it. We'll take you to the standard Salesforce login window.

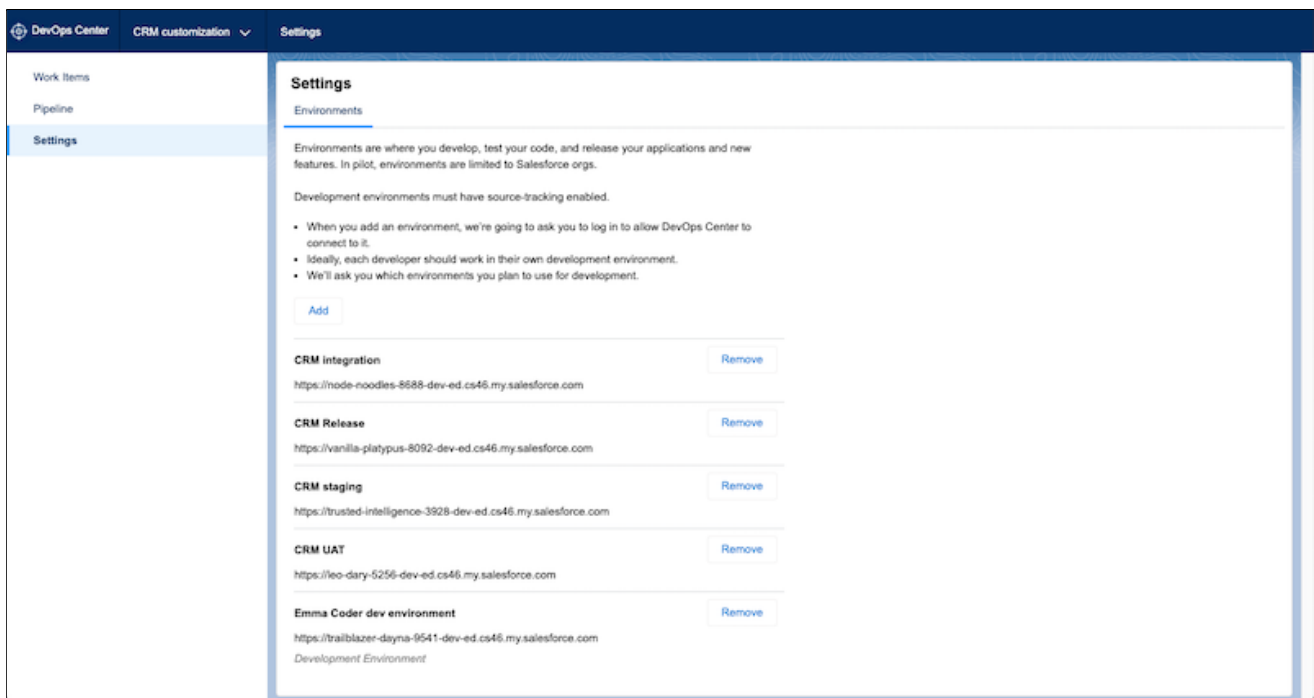
[Cancel](#) [Log In](#)

2. Select the environment type. Skip the **Used for development** checkbox.
3. Click **Log In**.

DevOps Center takes you to the Salesforce login screen. After you authenticate successfully, allow DevOps Center to remember your credentials.

4. Repeat steps to add another pipeline environment.

Once you've added all your pipeline environments, you'll see them listed. Now you're ready to build your pipeline.



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:

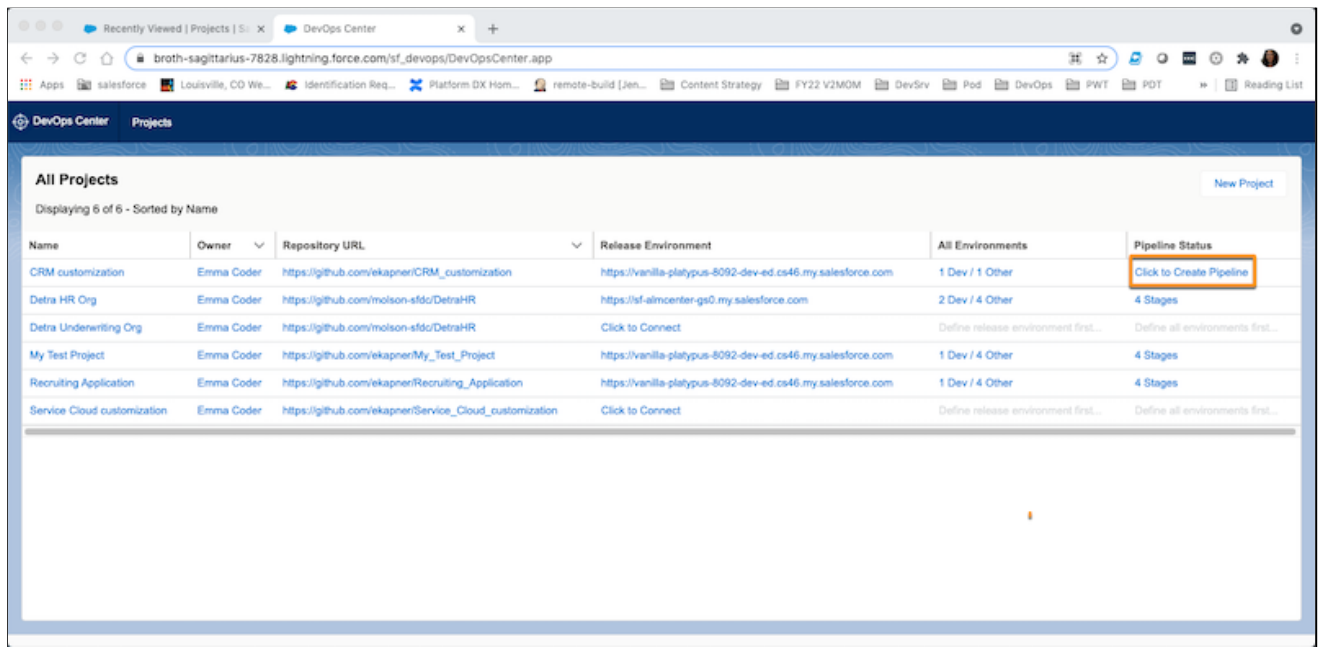
- Allow team members to move work items individually through the entire pipeline.
- Allow team members to move work items individually in early stages of the pipeline, and as a versioned group of changes (work item bundle) in later stages. Keep reading to learn all about the benefits of [work item bundles](#).

Build Your Pipeline

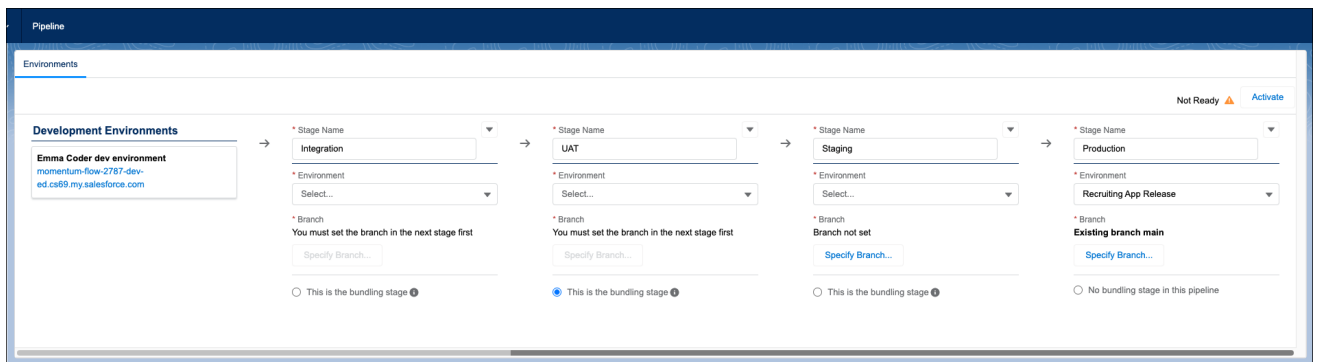
Once you define development environments and a release (production) org, you have a basic pipeline. However, it's not recommended that you deploy directly to production. You can use our template to build the pipeline, you can add more stages to the pipeline template, or you can build your own.

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

1. On the Projects page, under Pipeline Status, click **Click to Create Pipeline**, which takes you to the Environments tab. You can also click **Pipeline** on the left navigation bar.



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



- Before you begin, it's important to understand how to properly [specify the branch for each pipeline stage](#).
- Complete your pipeline configuration using the preferred method.
 - If you want to use the template to build your pipeline, see [Build Your Pipeline Using the Template](#).
 - If you want to build your own pipeline configuration, skip to [Build Your Own Pipeline](#).

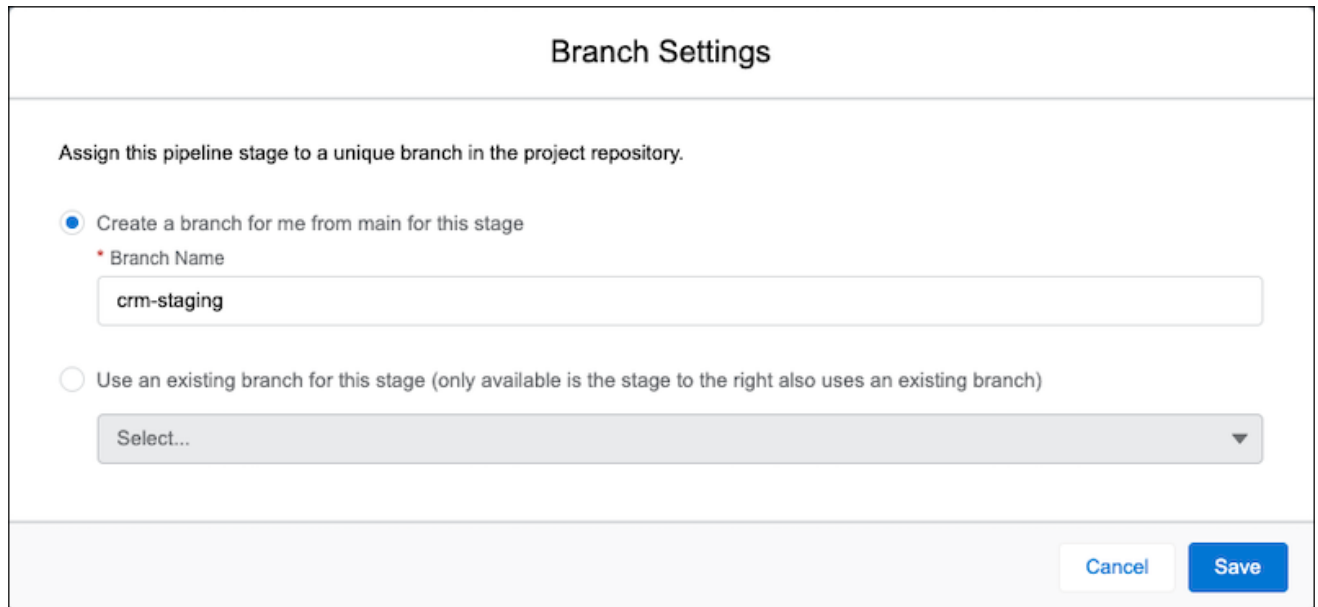
What You Need to Know About Branches

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

The branch must be sourced from the next stage 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.



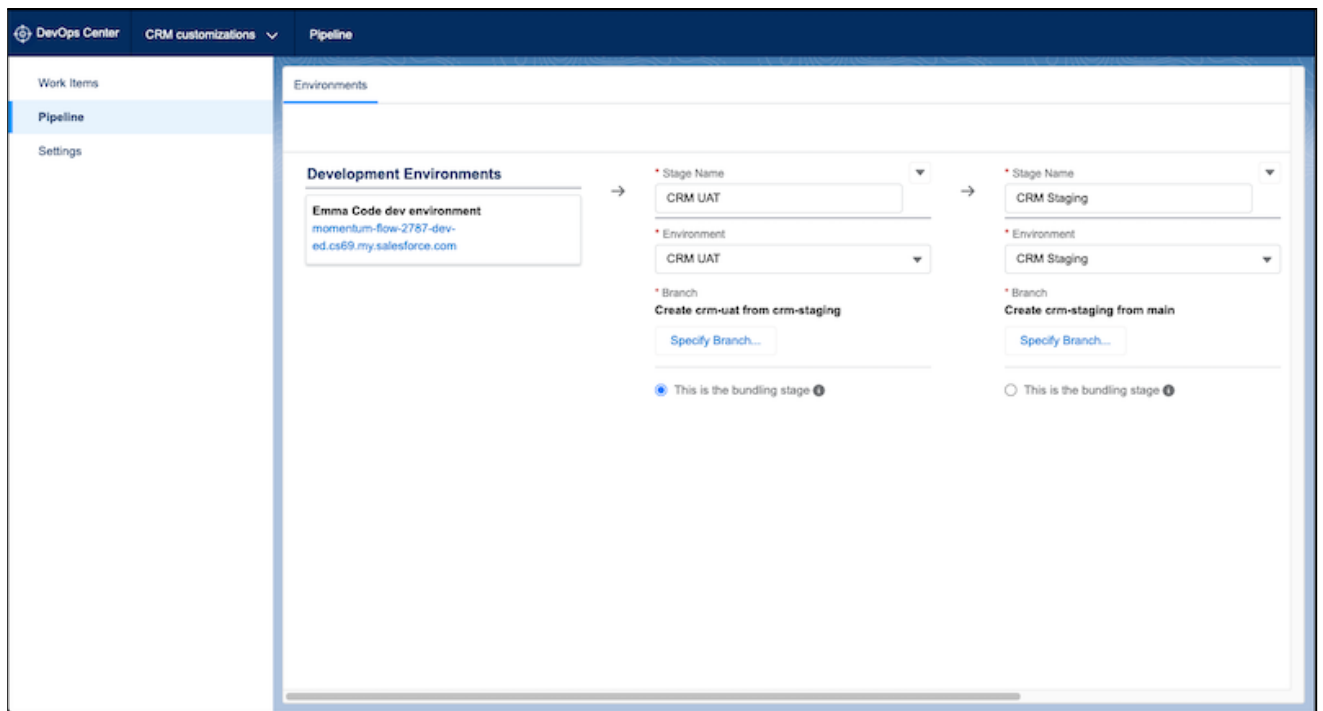
The image shows a 'Branch Settings' dialog box. At the top, it says 'Assign this pipeline stage to a unique branch in the project repository.' There are two radio button options. The first option, 'Create a branch for me from main for this stage', is selected. Below it is a text input field labeled '* Branch Name' with the value 'crm-staging'. The second option, 'Use an existing branch for this stage (only available is the stage to the right also uses an existing branch)', is unselected. Below it is a dropdown menu with the text 'Select...'. At the bottom right, there are 'Cancel' and 'Save' buttons.

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 main in GitHub. We don't recommend that you edit the branch for this stage.

1. From the Pipelines page, start with the Staging stage.
2. Select the environment associated with this stage. If you don't see the environment you want to select, go to Settings and add it there first.
3. Next, specify the branch associated with this stage.
4. Indicate which stage is the bundling stage, the stage where changes are grouped together, versioned, and promoted together in downstream stages.

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 recommended a bundling stage.

5. Repeat this process for the rest of the stages.
6. [Activate the pipeline.](#)

Build Your Own Pipeline

You can 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.
1. Starting with the last stage before your release stage, select the environment associated with this stage. If you don't see the environment you want to select, go to Settings and add it there first.
 2. Next, specify the branch associated with this stage.
 3. Repeat for the rest of the stages.
 4. Indicate which stage is the bundling stage, the stage where changes are grouped together, 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.
 5. [Activate the pipeline.](#)

Activate The Pipeline

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

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

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.

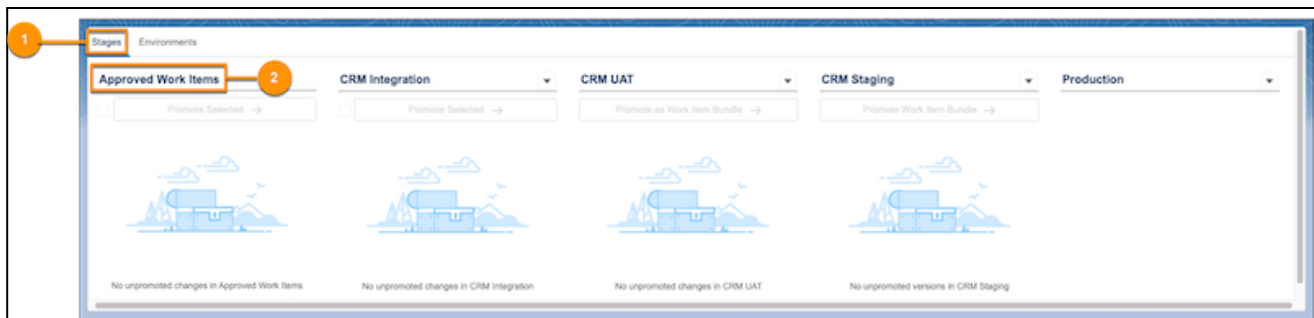
 A screenshot of a stage configuration form. It has three main sections:

- * Stage Name:** A text input field containing 'CRM Integration'.
- * Environment:** A dropdown menu with 'Select...' as the current selection. Below it, a red error message reads: 'Each stage requires an environment.'
- * Branch:** A section with a red error message: 'Each stage requires a branch name.' Below this is a blue button labeled 'Specify Branch...'.

 At the bottom of the form, there is a radio button option labeled 'This is the bundling stage' with an information icon.

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

After the pipeline is activated, the Stages tab (1) appears as a second tab in the Pipeline view.

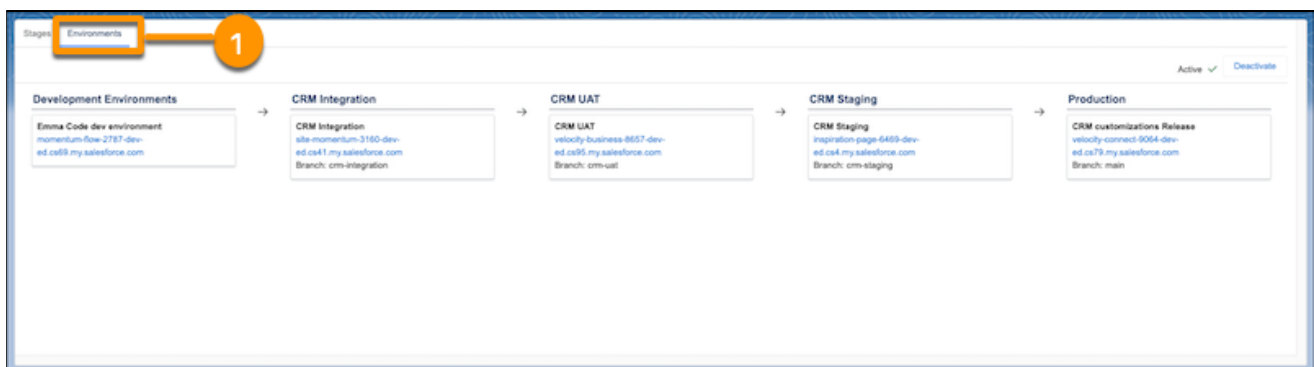


The Stages tab is where you promote work items through the pipeline. Work items appear in the Approved Work Items column (2) after they have been marked Approved. See [Create and Assign Project Work Items](#) for more information.

If you need to modify your pipeline configuration, go to the Environments tab and deactivate it. However, if any work items have been promoted through the pipeline, you can no longer deactivate it.

The Environments tab (1) provides the following functionality:

- When the pipeline is not active, configure the pipeline stages here.
- When the pipeline is active, you can see the configuration of each pipeline stage, including the environment and branch associated with the stage, and the latest bundle version installed to the stage.



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’ll 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 where you transition from the more flexible/individually-selectable work item promotion to the more predictable/versioned promotion is referred to as the *bundling stage*. This is the stage in the pipeline 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 unexpected behavior because the combination of changes in each stage may not be consistent. The versioned work item bundle that is created in the bundling stage and promoted through subsequent stages provides enhanced consistency because the changes contained in it have been merged into a unit that you can promote consistently and predictably from stage to stage.

Set Up Team Members in the DevOps Center Org

When your project is connected to its GitHub repository and you've set up the environments and pipeline, it's time to start adding the people on your project team to DevOps Center. Add team members as collaborators in your GitHub repository, create user accounts for them in the DevOps Center org, and assign them the permission sets they need to work in DevOps Center.

Add Team Members as Repository Collaborators

In GitHub, 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, or they won't be able to access 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.

Add Team Members as Users in the DevOps Center Org

When you add team members, be sure to specify the appropriate license based on their role.

- When you add a project team member as a user in the DevOps Center org, use one of the twenty Developer licenses provided to pilot participants. If you need more Developer licenses, please reach out to sf-devopscenter-preview@salesforce.com.
- If you plan to create a second user on your team who has the permissions needed to be a team/project manager, like yourself, create their user account in the DevOps Center Org with a Salesforce User license. The Developer license doesn't provide the appropriate user permissions.



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 that they should wait until they hear from you that you're done with setup 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**.
4. Enter each user's name and email address, nickname and a unique username in the form of an email address. By default, the username is the same as the email address.

A Salesforce username must be unique across all Salesforce orgs. The username must be in the format of an email address, for example, jane@salesforce.com. The email used for your username doesn't have to function. You can have the same email address associated with your account across multiple orgs.

5. Select the **Developer** user license for team members, or the **Salesforce User** license for team/project managers.
6. Select the **Developer** profile for team members, or the **Standard User** profile for team/project managers.
7. Select the **Generate passwords and notify user via email** checkbox.
8. Click **Save**.

Repeat as needed to create user accounts for all the team members.

Assign the DevOps Center Permission Sets

Assign permission sets to everyone working on your project in DevOps Center. Consider who might need to change project-level settings (such as adding another work environment) and who will be concerned only with work items.

DevOps Center Permission Sets

Permission Set	Description
DevOps Center	<p>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 environments and pipelines.</p> <p>Assign to: all DevOps Center users, including those also assigned the DevOps Center Manager permission set. The two permission sets don't overlap.</p>
DevOps Center Manager	<p>Provides the data access and permissions needed to set up DevOps Center projects, environments, and users.</p> <p>Assign to: team/project managers</p>
DevOps Center Release Manager	<p>Provides permissions to perform promotions through the pipeline.</p> <p>Assign to: release manager</p>
DevOps Center DevOps	<p>Allows the DevOps engineer to add environments, create pipeline stages, and build the pipeline.</p> <p>Assign to: DevOps engineer</p>
sf_devops_InitializeEnvironments	<p>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 new NamedCredential records.</p> <p>Assign to: team/project managers</p>
sf_devops_NamedCredentials	<p>Grants access to the named credentials needed to authenticate to environments. Created and maintained automatically by DevOps Center.</p> <p>Assign to: all DevOps Center users</p>

See also

Salesforce Help: [Permission Sets](#)

Salesforce Help: [Named Credentials](#)

Assign These Permission Sets 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.

Assign Additional Permission Sets to DevOps Center Managers

Add the **DevOps Center Manager**, **DevOps Center Release Manager**, and **DevOps Center DevOps** permission sets to team members who need permissions to configure projects, build pipelines, and promote changes through the pipeline.

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.

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.

1. From the Projects page, click the name of the project for which you're creating work items.



Tip: Did you already [configure your pipeline](#)? If not, DevOps Center walks you through specifying a release environment first before you can continue.

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. If more details would be helpful to the assignee, use the Description field to provide additional information. We'll use the first 255 characters of the description to help identify changes for this work item in GitHub.
5. (Optional) Assign the work item to a team member.

New Work Item

*** Subject**

Description

Assigned To

Create custom fields for Position and Candidate objects

Salesforce Sans ▼
12 ▼
■ ▼
B I U

Create fields to track data regarding positions.

- Create a global pick-list for Department
- Add a Date Closed, Date Opened, Duration, Job Description, Education, Pay Grade fields for Position
- Department and Pay Grade should be dependent pick-lists
- For Candidate, create an encrypted field for SSN

Emma Coder
✕

Cancel
Save

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.

Next: Bring Team Members into DevOps Center

Now that you've set up a DevOps Center project and set up your team members to work in DevOps Center, it's time to bring your team members into the DevOps Center org. Refer them to the *DevOps Center Quick Start Guide*, where they can learn how to use the main features of DevOps Center, including how to manage project work items, review changes they've made in a work environment, commit those changes to the project repository, and promote the work items through the pipeline.

Known Issues

Here are some tips if you encounter these known issues.

DevOps Center Can't Find Existing Repo When Creating a Project

Cause: Due to a limitation in pilot, 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 working around this issue.