Quick Start Guide

DevOps Center

Pilot, Summer ’21

@salesforcedocs

Last updated: May 14, 2021

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

DevOps Center Basics
- DevOps Center Projects
- One Collaboration Tool to Support a Variety of Roles
- Get Set Up with GitHub and the Repository

Typical Workflow
- Open DevOps Center
- Open a Project Work Item
- Start on a Work Item
- Pull Changes from the Development Environment
- Commit Changes to the Project Repository
- Share Your Changes for Team Member Review
- Approve Work Items
- Promote Work Items Through Your Pipeline
    - Background Information on Pipelines (please READ)
    - Promote Individual Work Items
    - Promote as Work Item Bundle
    - Promote Work Item Bundle
- Conflict Detection and Resolution
    - Multiple Work Items That Modify the Same Source File
    - Unresolvable Merge Conflict in Two Branches
    - Your Component Version Is Different from Merged Component
- View Promotion Status
- Validate Changes

Manage Work Items
- Work Item Stages
- Create Work Items
- Edit Work Items

TROUBLESHOOTING
- You're seeing the error "The request was invalid. URL does not reference a valid SFDX project. projectUrl"
- You're seeing the error "There was an error. Please reload the page or contact your system administrator if the problem persists."

DevOps Center Basics

Welcome to the DevOps Center pilot.

Your team manager has set up DevOps Center according to the steps in the *DevOps Center Setup Guide.* We assume that you've been notified by the team manager that the DevOps Center org is ready for you to log in.

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 development environments are used to make changes

During configuration, the team manager set up the project and associated it with a source control repository. For pilot, we support only GitHub for project repositories. You don't have to be a GitHub expert to use DevOps Center.

One Collaboration Tool to Support a Variety of Roles

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

- Team Manager/Project Manager
- Org Admin
- Admin/Declarative Developer
- Pro-code Developer
- Release Manager
- Business Owner
- Environments Manager
- Quality Assurance Specialist

To ensure that team members have access to the features they need to do their work, we provide several DevOps Center permission sets that your Salesforce admin (or team manager) can use to assign you appropriate permissions to access DevOps Center functionality.

The team also needs a Salesforce admin to create and manage DevOps Center users, including assigning the permission sets. This user could also be the team manager user. If you perform

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

For more information about DevOps Center permission sets, see the *DevOps Center Setup Guide*.

Xxx

xxx

Get Set Up with GitHub and the Repository

You *must* have a cloud-based, GitHub-hosted, github.com account to use the DevOps Center in this pilot release. You can create a new (free) account or use an existing account. We aren't supporting any kind of on-premise, enterprise, locally-hosted GitHub, or any other Git-based or other source control provider system in this release.

1. 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.
2. Send your GitHub username to the team manager or Salesforce admin who's setting up DevOps Center so they can add you as a collaborator in the repository.
3. Accept the invitation to collaborate in the GitHub repository. If you haven't received an email invitation, contact the person who set up DevOps Center.

If you're new to GitHub concepts, we recommend working through the Git and GitHub Basics module and the Work with the GitHub Workflow unit in Trailhead.

Basic Workflow

1. Open DevOps Center
2. Start on a Work Item
3. Pull Changes from the development environment for Your Review
4. Commit Changes to the Project Repository
5. Share Your Changes for Team Member Review
6. Approve Work Items
7. Promote Work Items
8. Resolve conflicts (pointer to conflict detection and resolution)

Reach out to your team manager if you haven't received instructions regarding the tasks to perform to complete setup. We assume that you already completed these tasks before you start working with DevOps Center.

Open DevOps Center

Before you start: Locate the *Welcome to Salesforce* email generated when your team manager added you as a user to the DevOps Center org.

1. Use the link in the *Welcome to Salesforce* email you received to log in to the DevOps Center org.
2. From the App Launcher, find and select DevOps Center.

DevOps Center opens to the Projects page, which lists current projects. Your team manager created at least one project as part of the setup process.

Open a Project Work Item

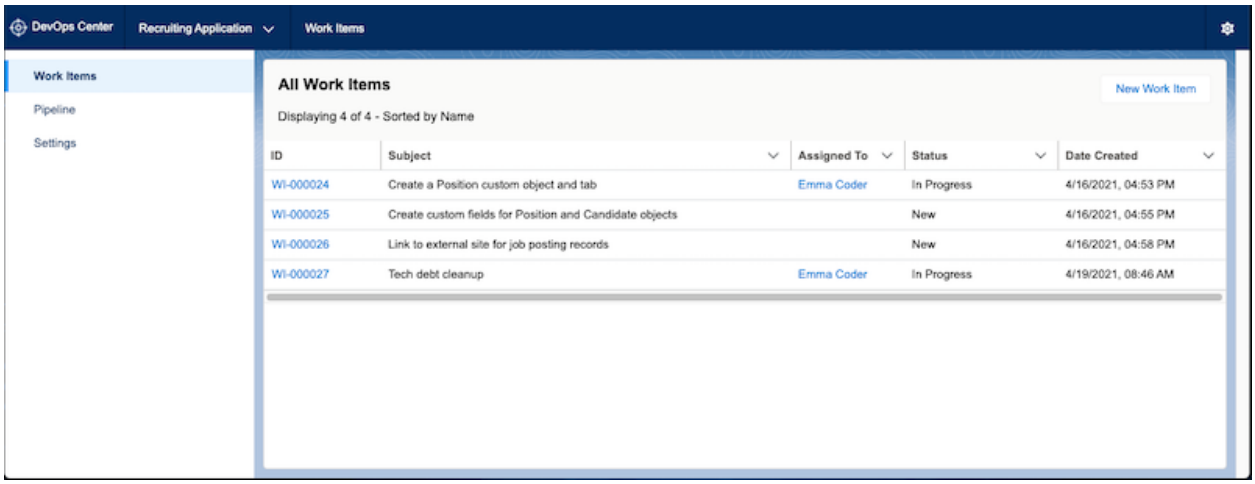
A team uses *work items* to track the progress of changes created to achieve a specific objective or task, such as enabling a user story, creating a new feature, or addressing a bug. Managing releases through work items makes it easier to track the progress of the overall release and identify areas of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

When you open a project for the first time, you might see one or more work items assigned to you.

Before you start: Be sure you've accepted the emailed invitation from GitHub to collaborate.

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

   The project opens to its Work Items page. If your team manager created work items, you'll see them listed here. Both you and other team members can create additional work items as the project progresses.

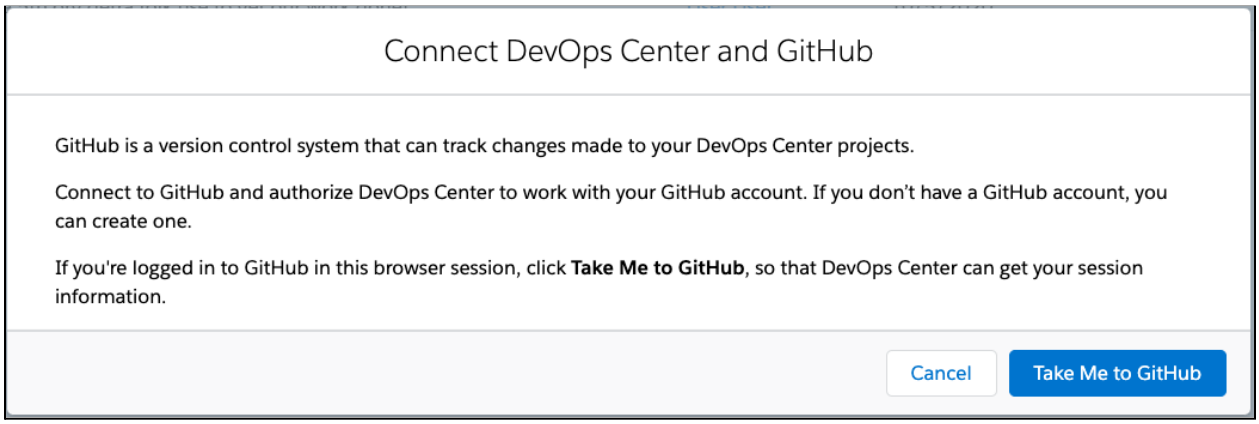


2. Click an unassigned work item or a work item assigned to you.

   If you're currently logged in to GitHub, DevOps Center retrieves your session information and returns you to the work item. Skip to Start on a Work Item.

   If you're not currently logged in to GitHub, you're asked to log in to GitHub and authorize access for DevOps Center. After you complete the authentication process, we can make changes on your behalf in the project's GitHub repository.

   Click **Take Me to GitHub**.



   If the authorization is successful, you're returned to the work item. DevOps Center can now update the project repository in GitHub.

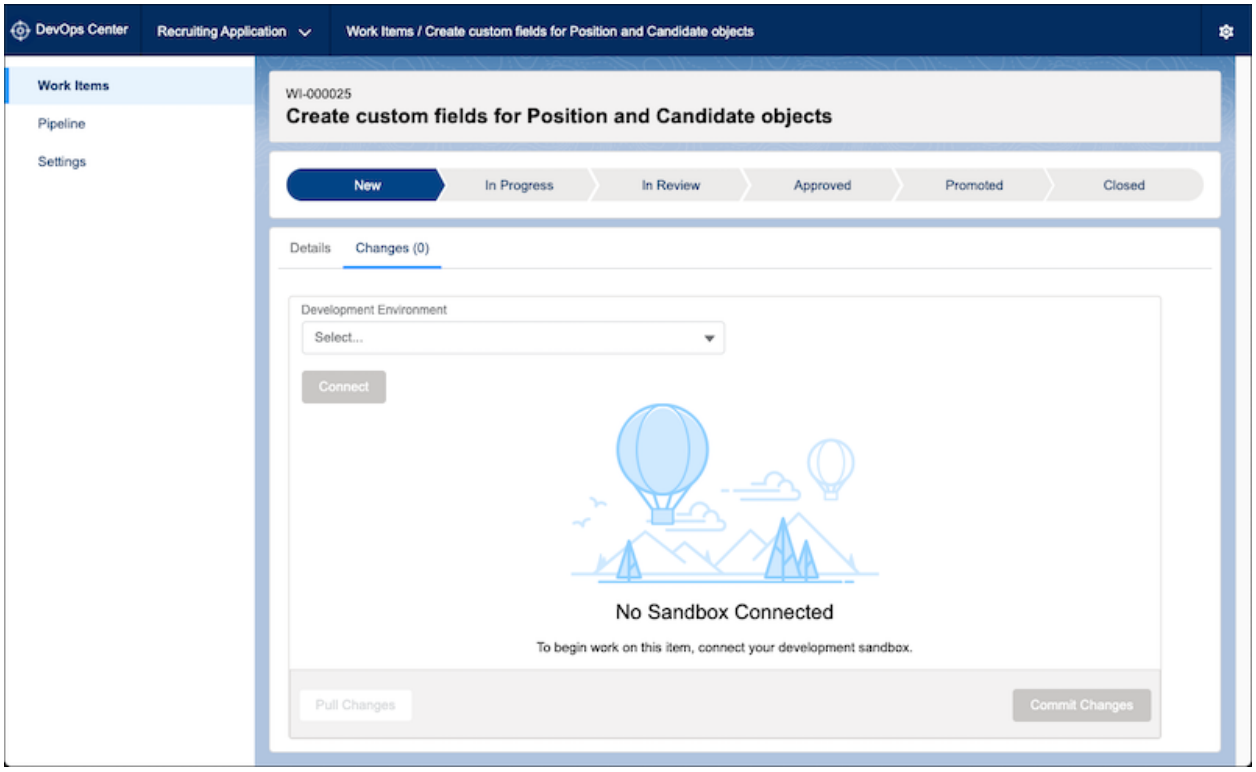
Start on a Work Item

When you're ready to begin work, connect to the environment you want to use for this work item. For background information on work items and work item stages, see Manage Work Items.

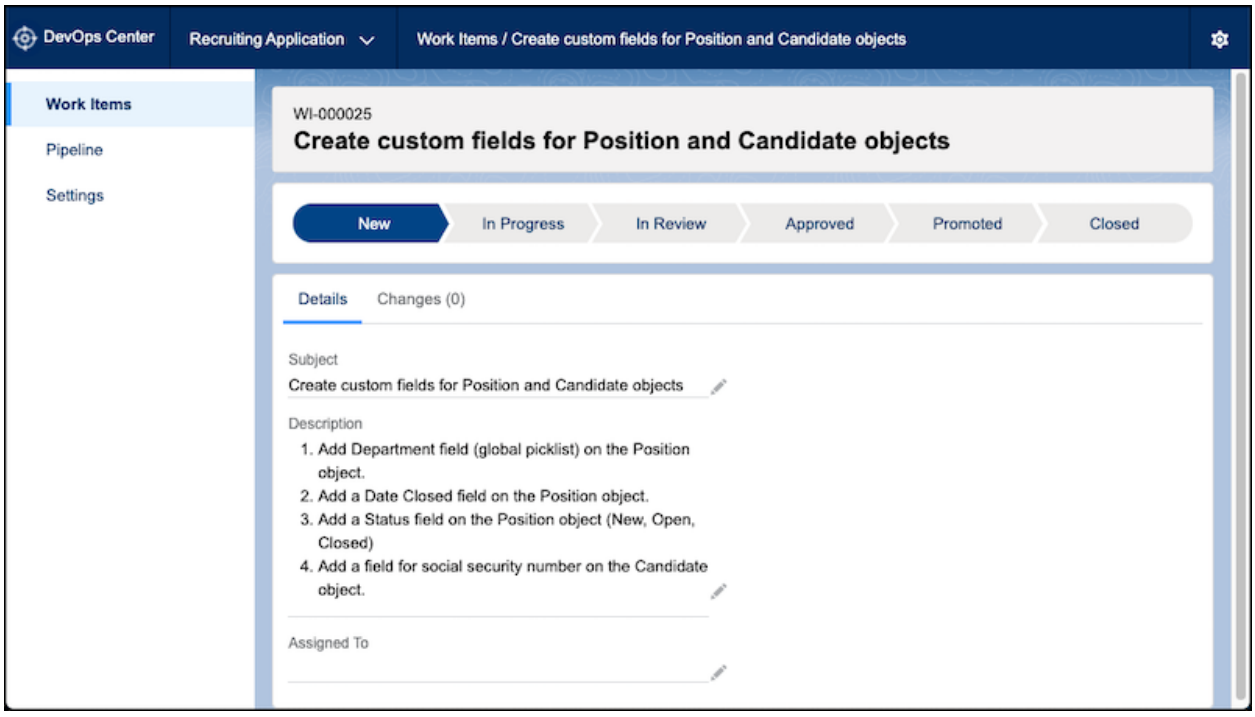
The development environments are Developer or Developer Pro sandboxes with the source tracking in sandboxes feature enabled. The environments were added to the project during setup to make them available for team members. During setup, the team manager gave each environment a nickname to make them easier to recognize in this menu.

Note: Go to the Settings page in DevOps Center to see the nickname and URL for each environment to confirm you're connecting to the correct environment.

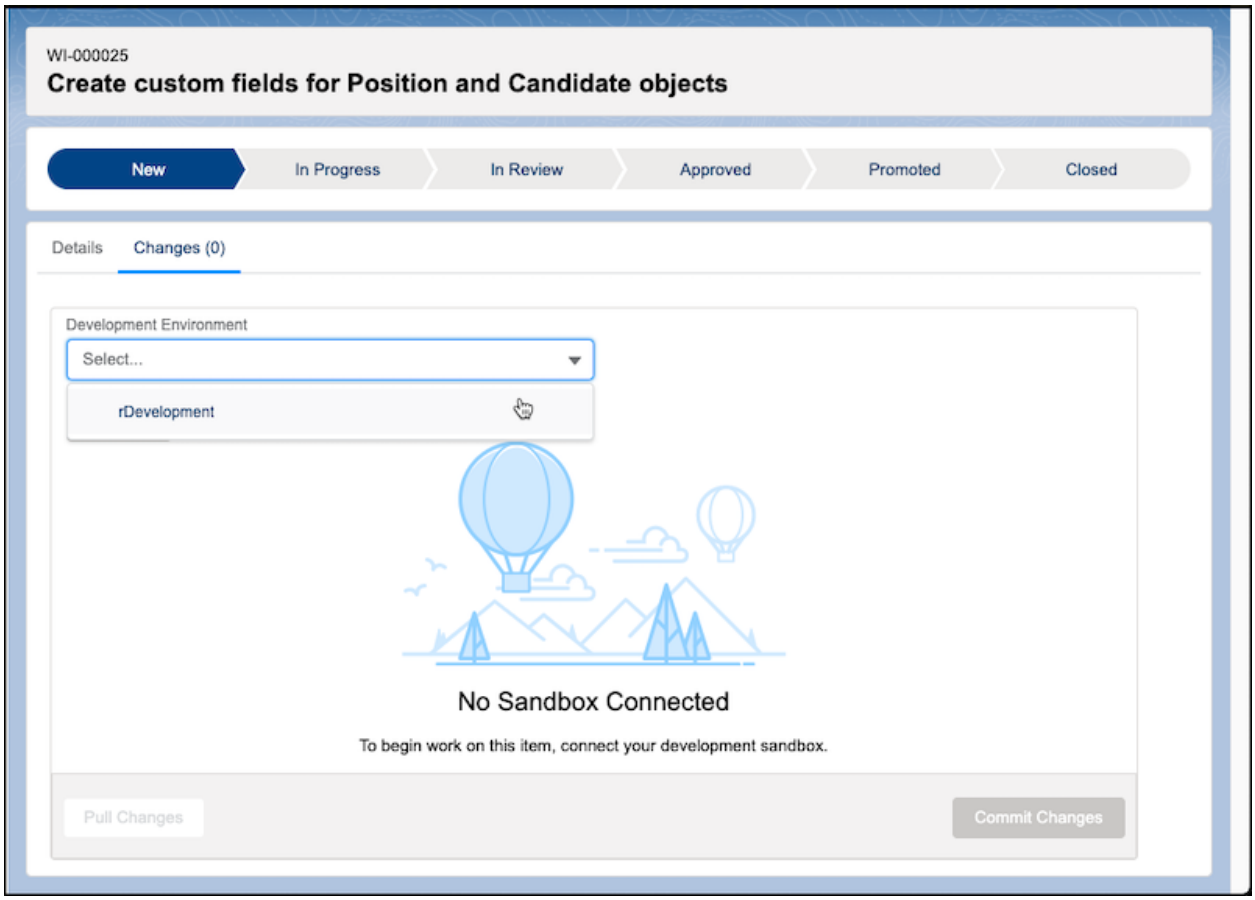
A work item has a progress bar that shows which stage of the development process it's in. The work item starts in the New stage, meaning that you haven't made any changes in support of the bug or user story that the work item describes. (We'll go over work item stages in more detail a little later.)



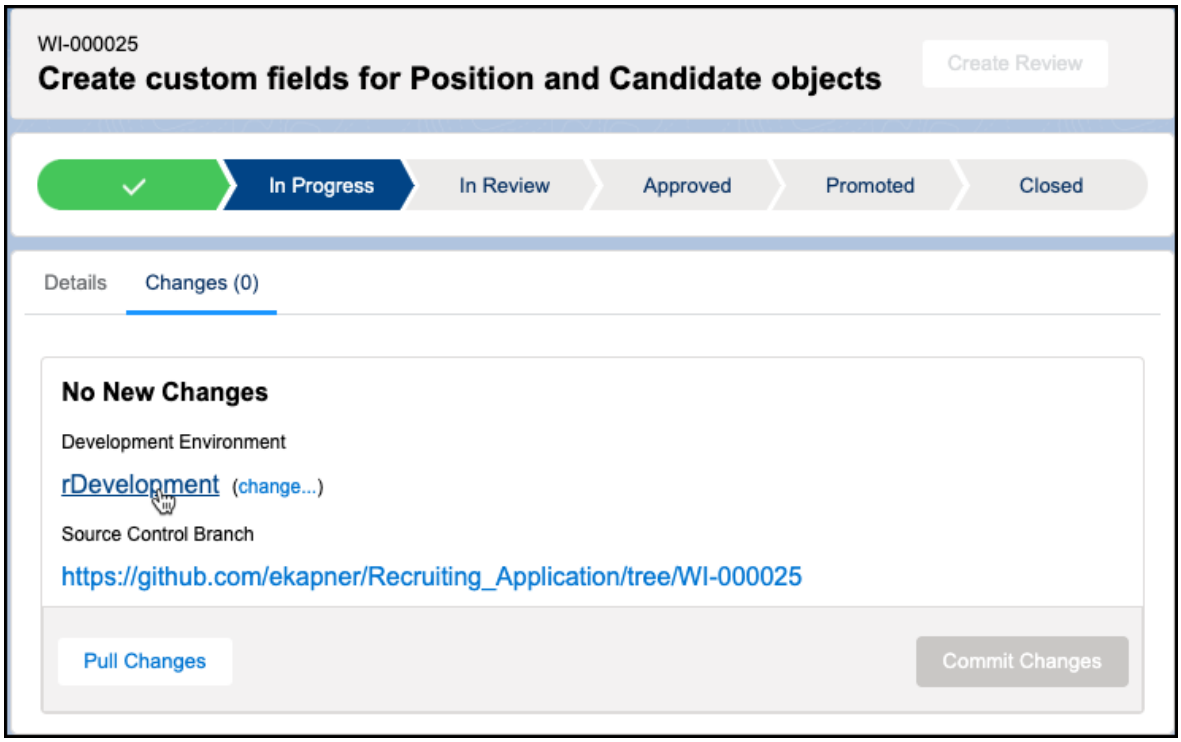
1. Click the Details tab to view the description for the work item scope, or to assign it to yourself.



2. Back in the Changes tab, select the development environment to use for this work item.



3. Click **Connect** to log in. After you connect to a development environment, the work item moves to the In Progress stage.
4. Click the link for the Development Environment name to open the development environment and begin making your changes.



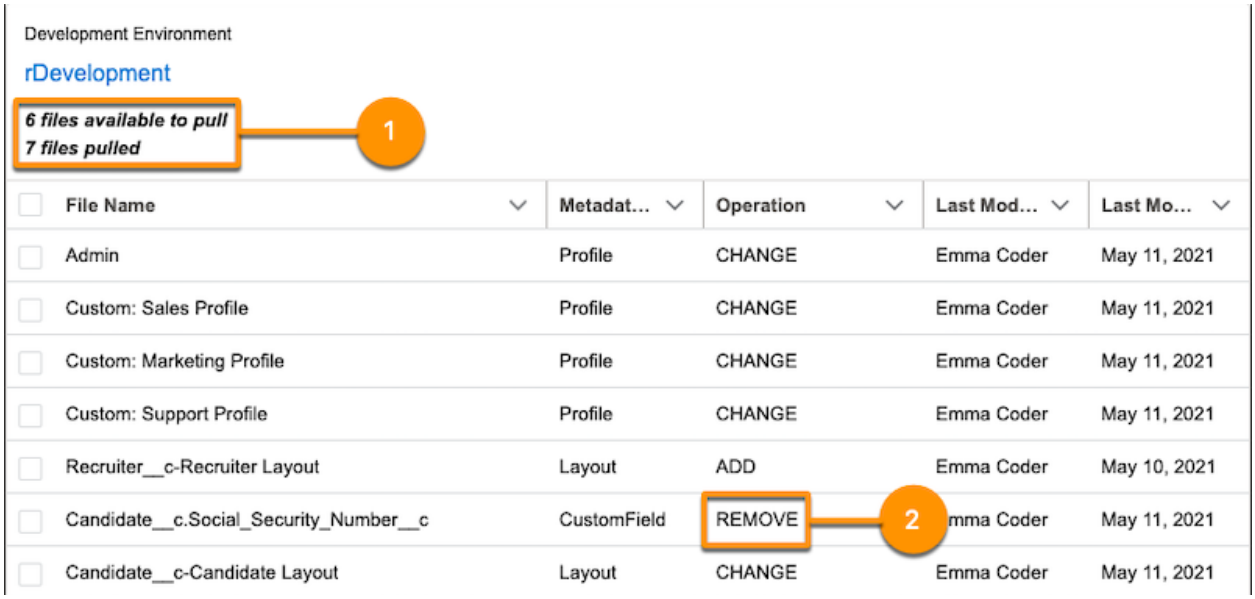
Pull Changes from the Development Environment

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

After making and testing your changes, pull them from your development environment, so you can see the changes before committing them to the repository for review.

The Changes tab lists see how many files are available to pull (new changes from the dev environment), and how many files are already pulled but not yet committed in the source control repo (1).

Warning: DevOps Center doesn't currently support committing files marked as REMOVE (2). The commit fails and the file won't be removed from downstream pipeline stages.



1. Click **Pull Changes**.

   A list of the added, changed, and removed component files is generated from the source tracking in your development environment.

2. Select the changes to commit to the source control repository.

   You can click the headings in the list of files to sort the list. If the list shows a file that you don't want to commit, leave its checkbox unselected. For example, if you found a Profile component file in the list that wasn't part of your intended changes for the work item, don't select it to leave it out of your commit.

   &&& screen shot showing what you've selected?

   Note: When you don't include files, they are not included in your commit, but still exist in the development environment. Any changes that you pull, but don't commit, are indicated as pulled in all uncommitted work items.

Commit Changes to the Project Repository

After you are satisfied with the contents of the change request, commit the changes to the project repository. When you commit changes, you store them in the project repository branch in GitHub that was created when the work item moved to the In Progress stage. The branch is named after the work item, so it's clear where the changes it contains came from. For example, the branch for a work item WI-12345 might be similar to:

`https://github.com/<account>/<repo-name>/tree/WI-12345`

Important: If you're not the only person making changes in the development environment, don't let too much time go by between pulling changes and committing changes. It's possible to have someone modify one or more files that you plan to commit after you pull changes. When you commit changes to the project repository, you're committing the selected files as they currently are in the development environment, not as they were at the time you pulled the changes, if the two states are different. If you commit changes with an out-of-date understanding of what the changes are, you might commit a change you didn't intend or create a potential merge conflict.

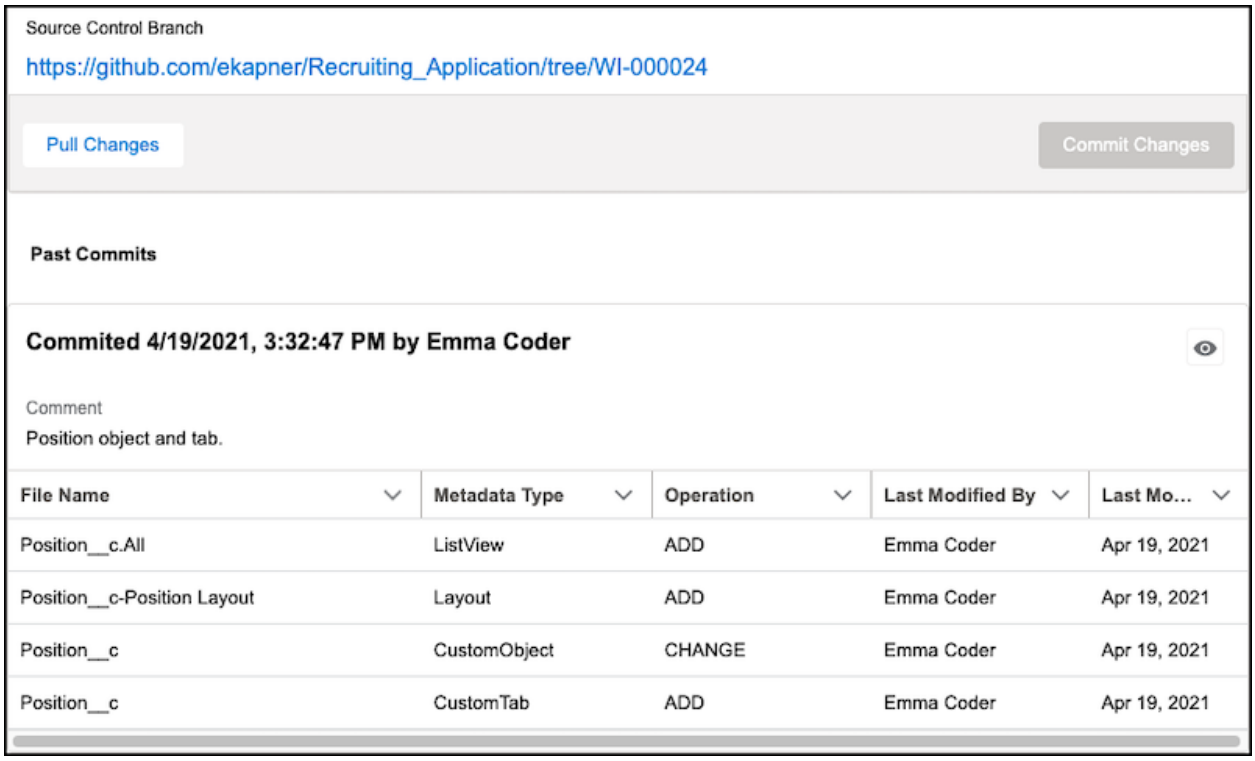
1. In the Changes tab, verify that you selected all the components you plan to commit.
2. Add a comment about this particular commit.

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

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

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

   After the page refreshes, the work item's committed changes are listed on the work item's Changes tab, most recent to oldest if there's more than one.



4. (Optional) To see the branch in GitHub, click the Source Control Branch URL.

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

Share Your Changes for Team Member Review

When changes for an In Progress work item have been committed and are ready to share with reviewers, creating a Review opens a change request, called a pull request in GitHub.

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

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

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

2. Click **View Change Request** to open a browser tab showing the work item branch in the source control repository, where reviewers can see the file changes.

Approve Work Items

After a team member reviews the work item, the work item can be marked as approved. Work items are approved individually by clicking the **Approve?** toggle. Click the toggle again to withdraw the approval. You can revoke the approval until the work item is promoted.

Promote Work Items Through Your Pipeline

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

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

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

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

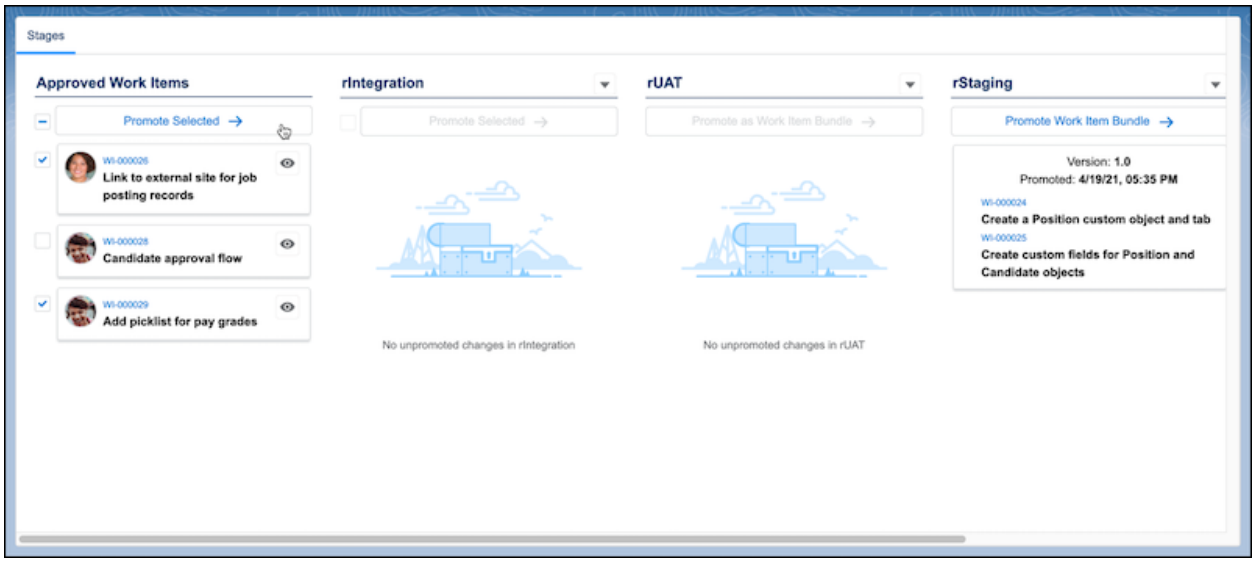
- Allow you to move work items individually through the entire pipeline.
- Allow you to move work items individually in early stages of the pipeline, and as a work item bundle in later stages. The point in the pipeline where you transition from the more flexible/individually-selectable work item promotion to the more predictable/versioned promotion is referred to as the bundling stage. 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 work item bundle continues to be promoted as a consistent unit through subsequent stages when you perform a promotion.

See also
This guide: Conflict Detection and Resolution
DevOps Center Setup Guide: Configure Your Pipeline

Promote Individual Work Items

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

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



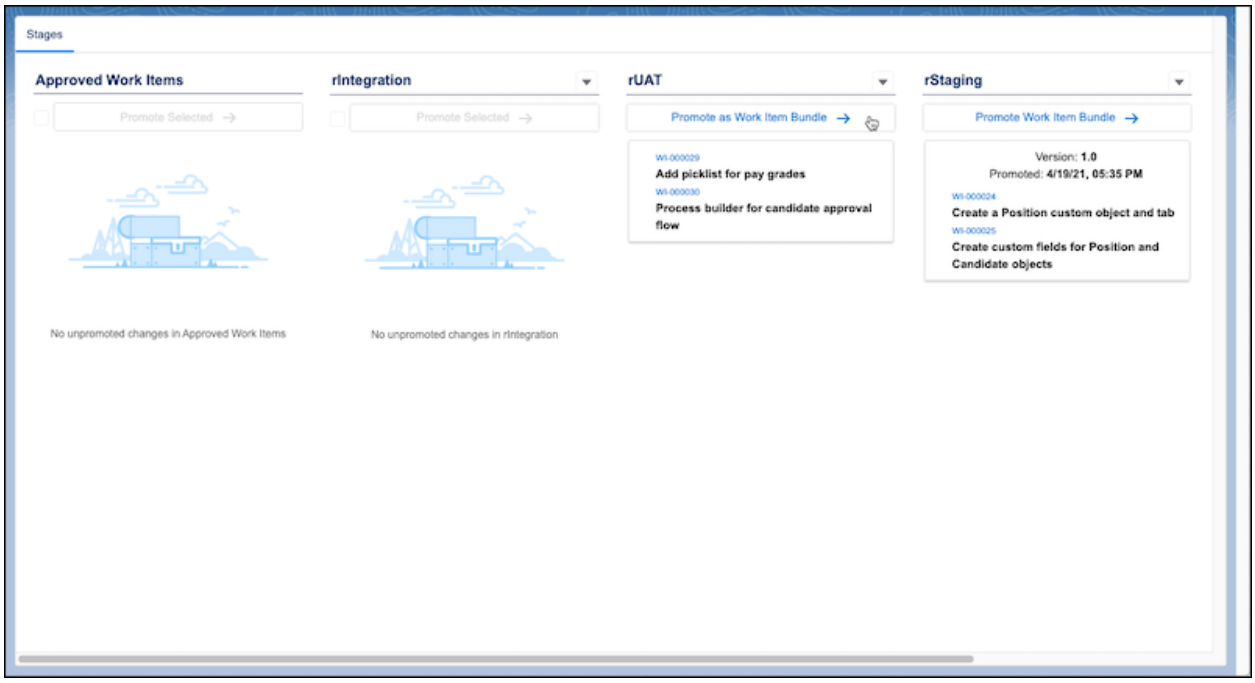
4. If this is your first time promoting to this stage, click **Connect** to log in to the stage's environment, then repeat steps 2-3.
5. In the Promotion Options dialog, select whether to run all Apex tests.
6. Click **Promote**. The selected work items are merged and deployed to the environment associated with the next stage.

Promote as Work Item Bundle

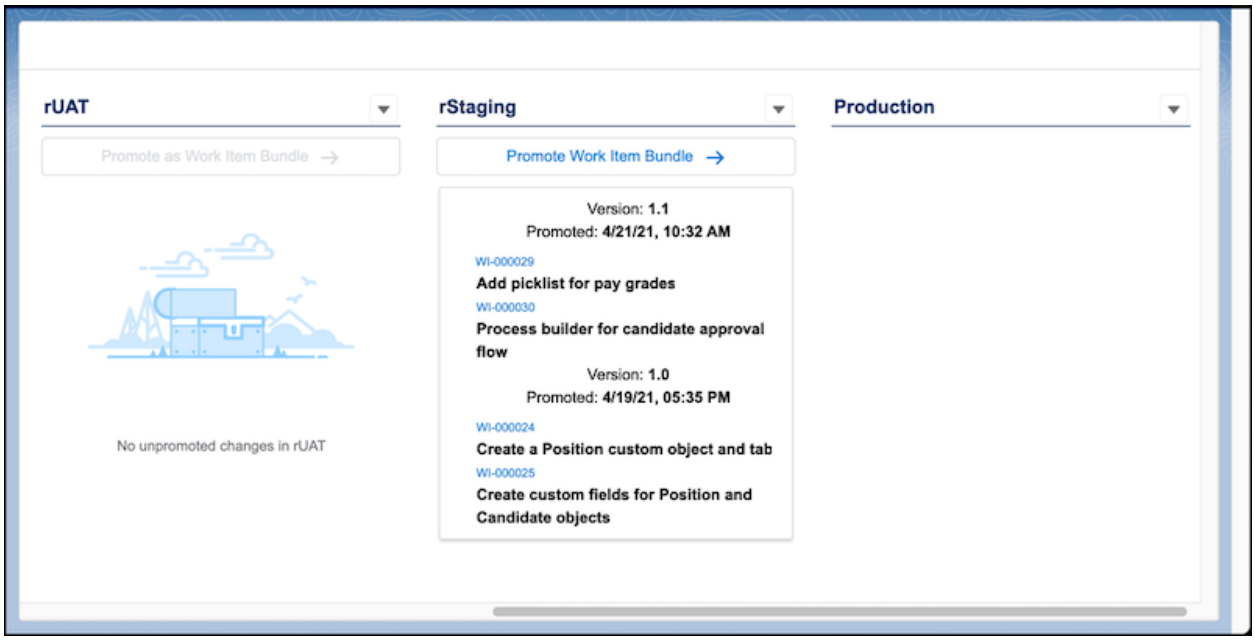
Promotions in this stage result in the creation of a work item bundle. You see a list of the items you promoted individually from the previous stage. All unpromoted work items in this stage become part of the bundle.

After promotion, the work item bundle is versioned.

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



2. If this is your first time promoting to this stage, click **Connect** to log in to the stage's environment, then repeat step 2.
3. In the Promotion Options dialog, enter a unique version identifier. Indicate the version as an alphanumeric string of your choice.
4. Select whether to run all Apex tests.
5. Click **Promote**. The work item bundle is merged and deployed to the environment associated with the next stage.



Promote Work Item Bundle

When you promote a work item bundle, all unpromoted versioned bundles are released together to the next stage. The unpromoted work item bundles and corresponding work items are listed here.

1. In the Pipeline view, in a versioned stage, click **Promote Work Item Bundle**.
2. If this is your first time promoting to this stage, click **Connect** to log in to the stage's environment, then repeat step 1.
3. In the Promotion Options dialog, select a version.
4. Select whether to run all Apex tests.
5. Click **Promote**. All versioned work item bundles are merged and deployed to the environment associated with the next stage although only the last work item bundle version is displayed.

Conflict Detection and Resolution

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

Here are some of the common types of conflicts.

Multiple Work Items That Modify the Same Source File

If you have multiple work items that modify the same source file (particularly in cases where multiple developers are touching the same source file in their own respective development environments), the changes may conflict or potentially overwrite each other when they are merged into the first integrated pipeline stage. DevOps Center warns you about the possibility of conflict when you promote multiple work items that contain the same source file. You can then choose to continue with the promotion or cancel to further analyze the potential conflict, or promote the work items one at a time.

Unresolvable Merge Conflict in Two Branches

When a work item is promoted from one stage to the next, the source files contained in the work item are merged from the first stage's source control branch to the next stage's source control branch. Sometimes this merge can fail due to unresolvable merge conflicts between the two branches. When this happens, you can see more information on the merge conflict in the error dialog in DevOps Center, and you can view additional details of the conflict in the pull request in the source control system. You can resolve the conflict manually directly in the source control system, and then you can reattempt the promotion in DevOps Center.

Your Component Version Is Different from Merged Component

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

Resolving Conflicts in Source Control System

- Use the View Change Request menu pick to open in source control
- View the conflict there
- Manually resolve
- Commit the Merge
- (Don't merge the PR).
- Come back to DOC and promote again

View Promotion Status

If the promotion succeeded, congrats!

If you need to validate promotion or dig into errors:

- Open Environment
- Open Branch in Source Control
- View Last Commit in Source Control

Validate Changes

When promotion completes you'll either see a success banner or a failure banner where you can see details of the error.

- You can also open the environment, view the branch in GH, and see the last commit in GH, via stage menu.

- If you want to view the results of the promotion in the org you open it to verify the changes and/or view the Deployment Status.

If you want to look at the deployment status in the org, you can do it

1. From the stage dropdown menu, select **Open Environment**.
2. From Setup, enter `Deployment Status` in Quick Find box, then select **Deployment Status**.
3. From the list of Succeeded or Failed deployments, click View Details.
4. View and verify the actual changes in the org.

Manage Work Items

In DevOps Center, a team uses *work items* to track the progress of changes created to achieve a specific objective or task, such as enabling a user story or addressing a bug. Managing releases through work items makes it easier to track the progress of the overall release and identify areas of concern.

A work item encapsulates information about an objective or task while it's being worked on. After the changes a work item tracks are completed, tested, and deployed, the work item is a record the team can come back to if a question arises about those changes. Information in a work item includes:

- **ID:** An automatically generated unique identifier for a work item that has a **WI-** prefix (for example, WI-12345). The ID helps DevOps Center users distinguish between similar-sounding work items. It's also used in the project repository to identify the branch where changes for the work item are stored.

- **Subject**: Required. A brief description of the task to be tracked.

- **Description**: A more detailed description of the task to be tracked. If the work item is a bug, include an example of the problem to be solved. If the work item is a user story, include a hypothetical description of how the new functionality would benefit users.

It's a good idea to include acceptance criteria that identifies what the successful completion of this work would look like. Make sure the person assigned this work item understands what's required or you'll lose time to iterations that could have been avoided.

- **Assigned To**: The DevOps Center user currently responsible for the completion of the task or objective tracked by this work item.

Anyone assigned the **DevOps Center** permission set can create work items and assign them to other DevOps Center users. This approach enables anyone who finds a bug to report it and anyone who wants to suggest an enhancement to share their idea for the rest of the team's consideration.

Work Item Stages

As work items move through the release management cycle, use the work item stages to indicate to the rest of the team how work is progressing.

[screenshot detail of progress bar]

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

- **New:** The initial status of a work item upon creation. The New status doesn't necessarily mean that no related work has occurred. Your team might be planning, sizing, scheduling, designing, and so on, in support of this work item. When it's time to start building customizations, however, select a development environment, which moves the work item to In Progress, and enables you to launch the environment directly from the work item.
- **In Progress:** Work that the listed assignee is actively pursuing in the development environment. Once in this stage, another team member can't assign this item to themselves or start working on it. DevOps Center creates a branch directory for it in the project repository.
- **In Review:** Work that's ready for your team members' review. When you click **Create Review**, the work item is moved to the In Review stage, and a change request is created automatically. (See Share Your Changes for Team Member Review for more about pull requests.)
- **Promoted:** This work item has been promoted to a pipeline stage. When it's promoted to the last pipeline stage, it's Closed.
- **Closed:** Work that's complete, reviewed, fully tested and verified, and promoted through the entire pipeline.

Create Work Items

Work items are organized by project and belong to the project where you created them.

1. From the Projects page, click the name of the project for which you're creating work items to open that project.
2. Click **New**.
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.

   The first 255 characters are used to create the description of the pull request associated with the work item.
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. All DevOps Center users can create additional work items as the project progresses.

Edit Work Items

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

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

   When you open a project, the Work Items tab is shown by default and lists all the current work items.
2. Locate the work item you want to edit, then click its ID to open it.
3. Edit the fields as needed.

   Note: You can choose a different work environment only if no changes have been committed for the work item from the current work environment.
4. Click **Save**.

TROUBLESHOOTING

You're seeing the error "The request was invalid. URL does not reference a valid SFDX project. *projectUrl*"

Cause: A project repository must contain a Salesforce DX project to work with DevOps Center. The easiest way to create a repository is to use a template we provide in GitHub to create your project repository.

Action: Log in to GitHub and use the provided repository template at github.com/forcedotcom/dx-empty to create your project repository. For details, see Create a Project Repository in GitHub.

You're seeing the error "There was an error. Please reload the page or contact your system administrator if the problem persists."

Cause: A likely cause is that you're using a sandbox where source tracking isn't enabled. Sandboxes must be your own Developer or Developer Pro sandboxes with source tracking enabled. For details, see Create Sandboxes to Use with DevOps Center.

Action: If your sandbox doesn't have source tracking enabled, follow the steps in Enable Source Tracking in Sandboxes (Beta).