Lab 2: Develop using Git

The purpose of this lab is to show how you can coordinate changes between your environment and Git repository. The primary scenario will show how to make changes in your repository source code and sync it into your environment. This lab will also cover collaborating with other developers using two different environments synched to the same repository. This lab will also cover basic conflict management strategies.

Prerequisite for this lab is to have two different environments, with one connected to a Git repository. Complete Lab 1 – Setup and Configure then proceed with this lab.

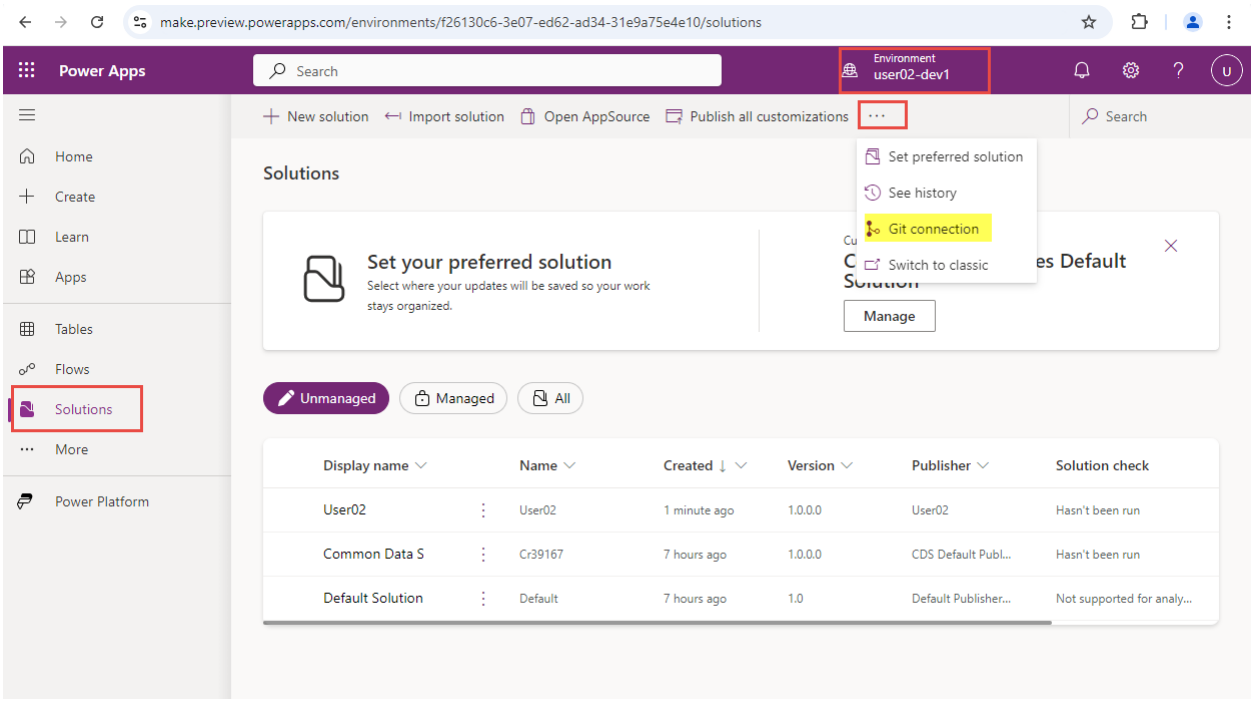
Lab 2 – Tasks

- Making changes in Git, then syncing into your environment
- Resolving conflicts with changes in Git and in your environment
- Connecting two different environments to Git
- Syncing changes between two environments using Git

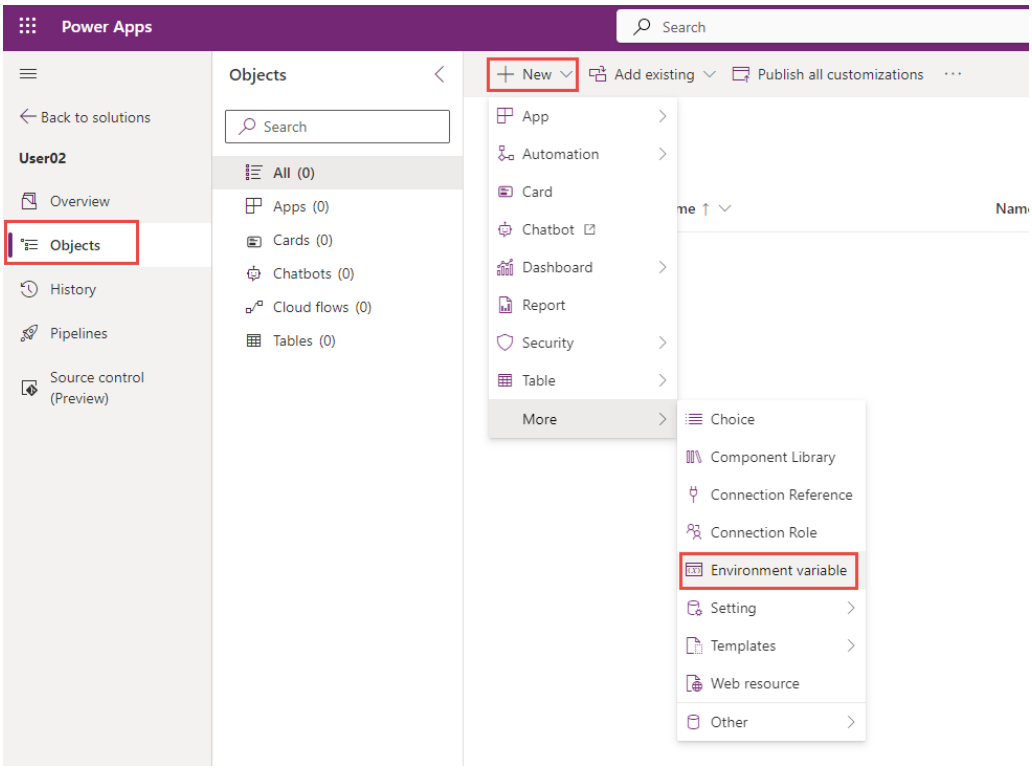
Task 1:

Let's build a very simple Canvas app.

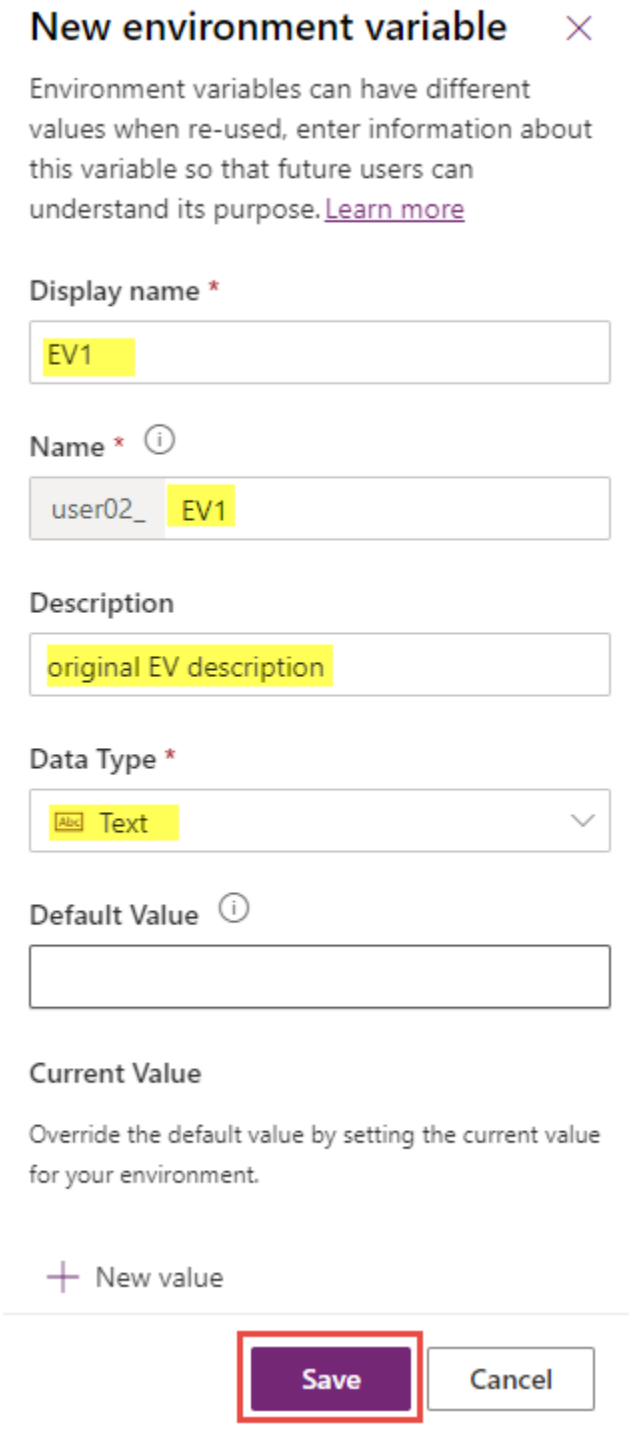
Navigate to maker portal, select the first development environment you connected to Git in Lab 1, then select Solutions from the left navigation. Ensure that the Git feature enablement is on your querystring, or that you can still see the Git connection option in the menu. If you do not see the Git menu option, simply add this to your url: ?powerappsAlmSourceControl.enableAlmSourceControl=true



We will use the solution that you set up in Lab 1. It is an empty solution. Click your solution name to edit it, then from that screen select **+ New** | **More** | **Environment Variable** to create a new environment variable.

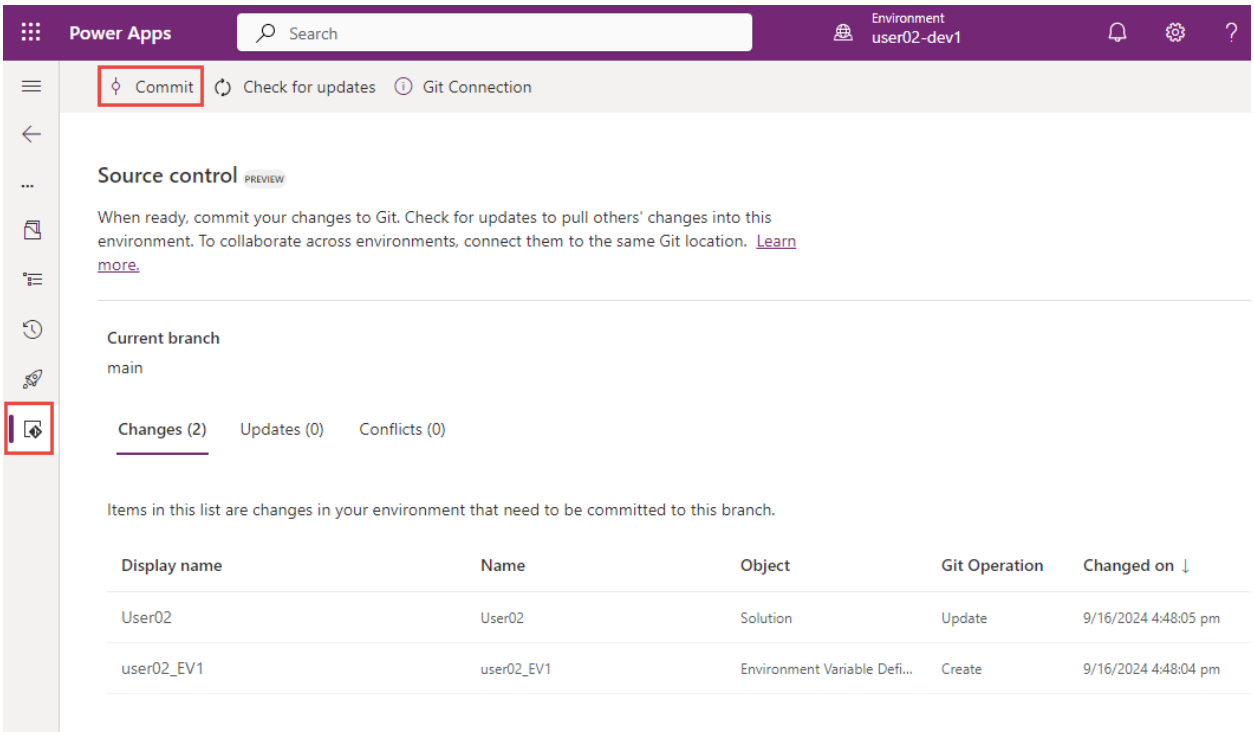


Give the environment variable a unique name, and enter a description you will recognize, select **Text** as a data type and click **Save**.

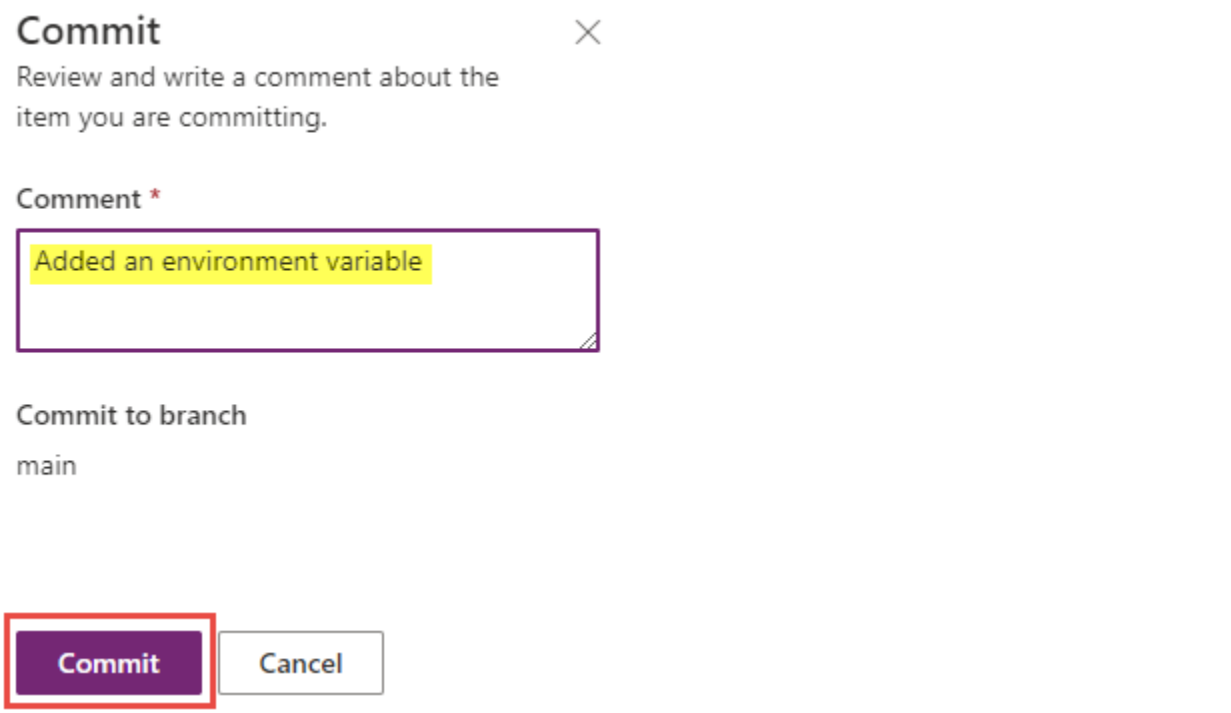


Click the Skip button to start with an empty canvas.

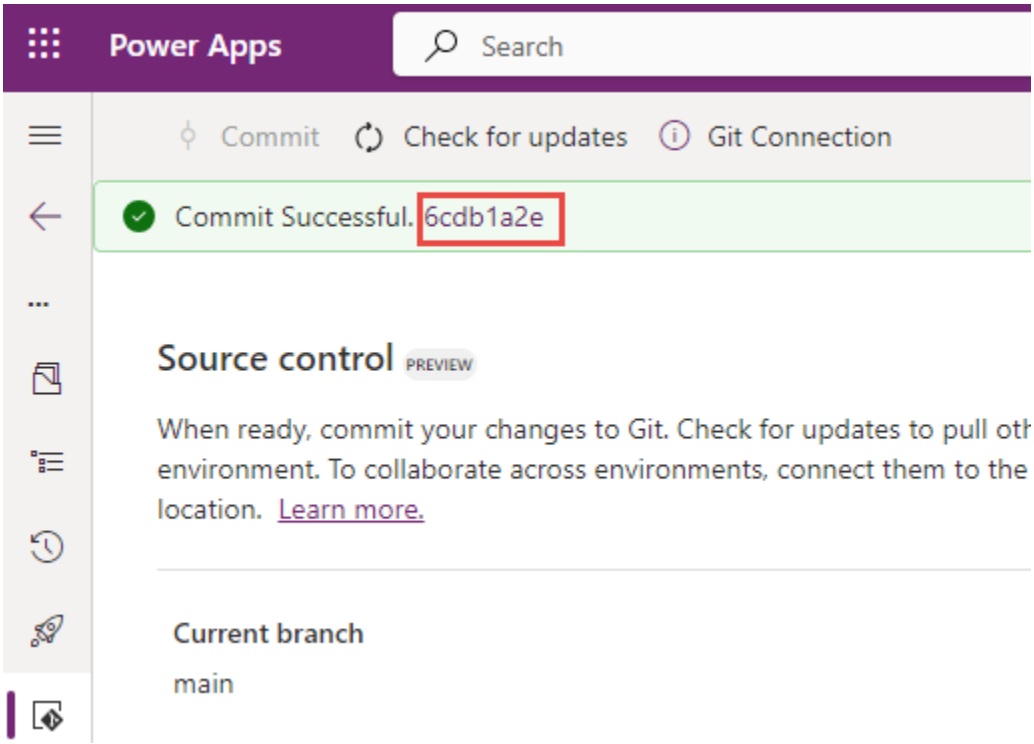
Back on the solution explorer, switch to the Source Control tab using the left navigation and review the changes. You should see the new environment variable to be created and your solution to be updated. Click Commit.



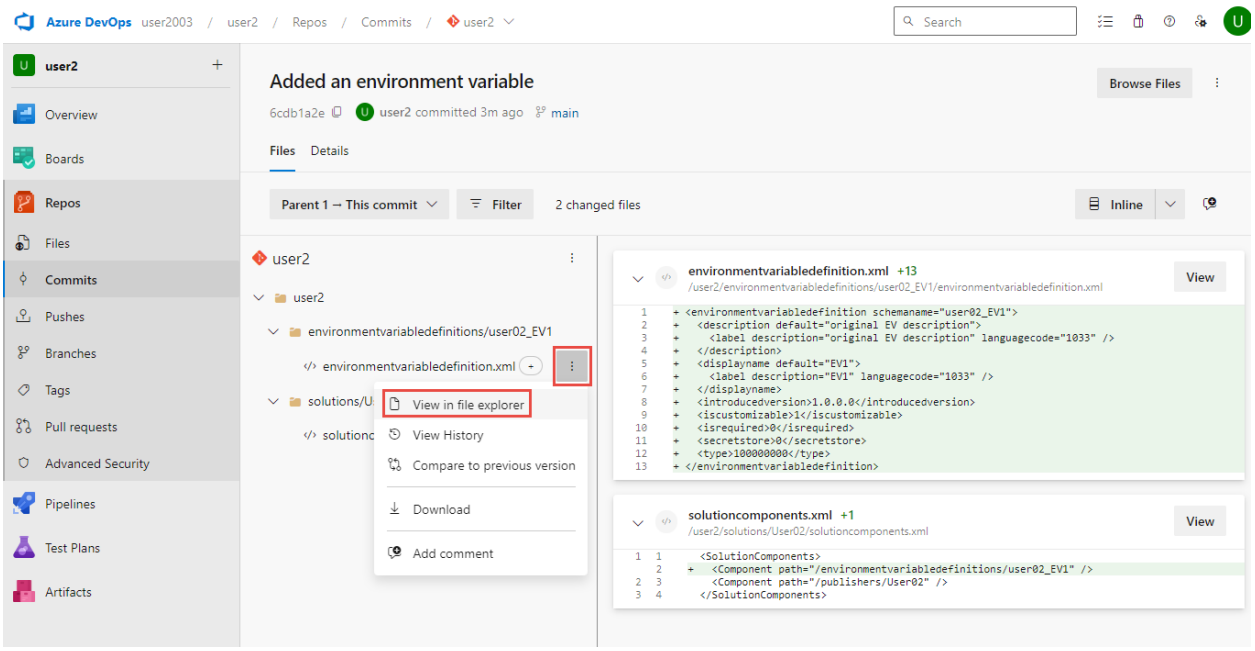
Enter a commit message and click **Commit**.



You will shortly see an information bar near the top of the screen that will include the commit id. Click that commit link to open a new tab to your repository.

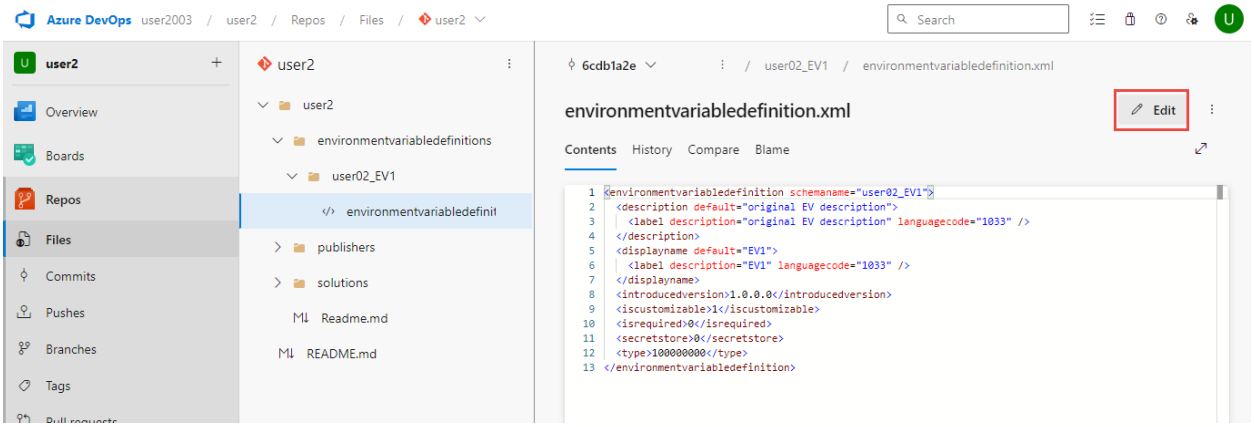


Locate the environment variable definition from within the files included in your commit. Hover over that to expose the command bar, and then select **View in file explorer** to go directly to the file.



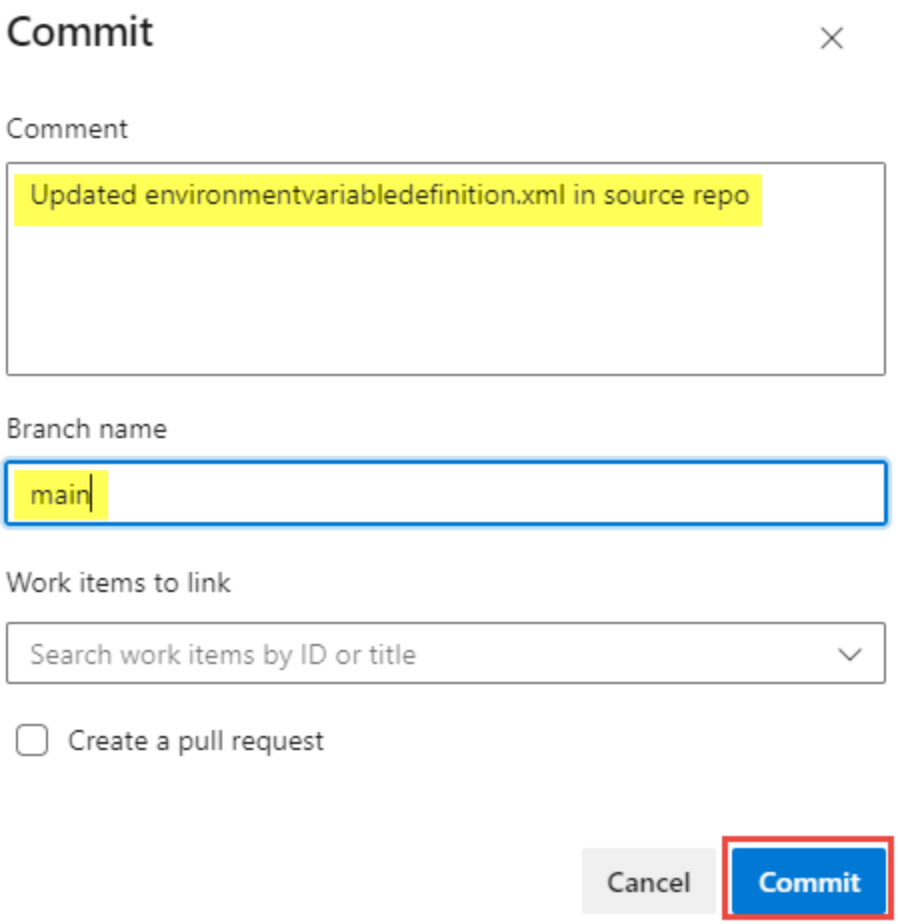
Task 2:

We want to demonstrate how you can change the source code in your repository and sync it within your environment. Using the file you located in the last task, click the **Edit** button to make a direct edit in your repository.

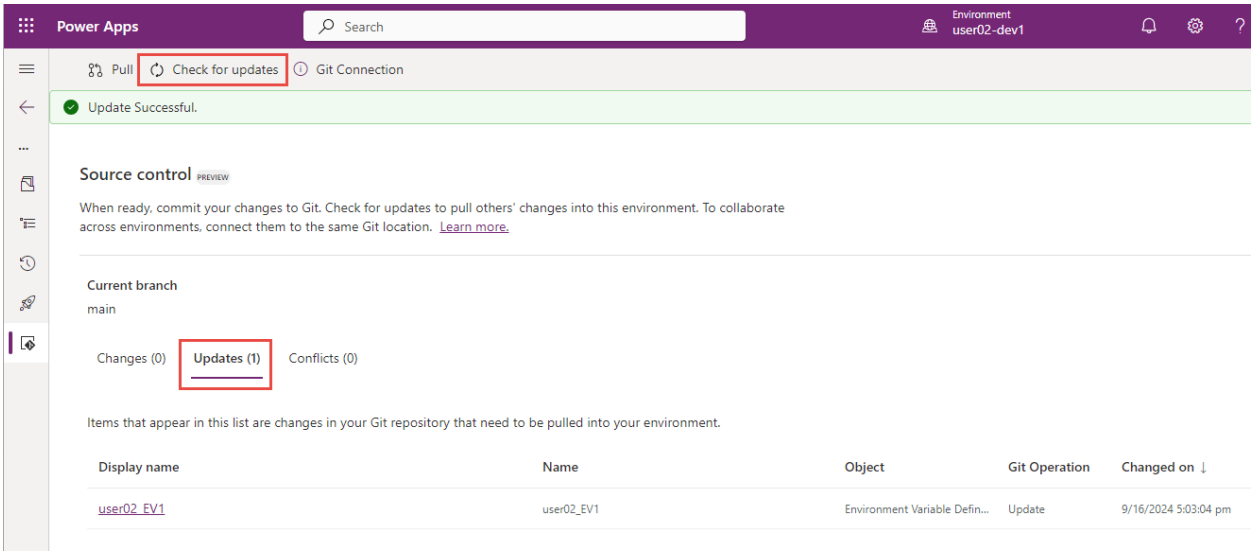


Change the text to a different value and then click **Commit**.

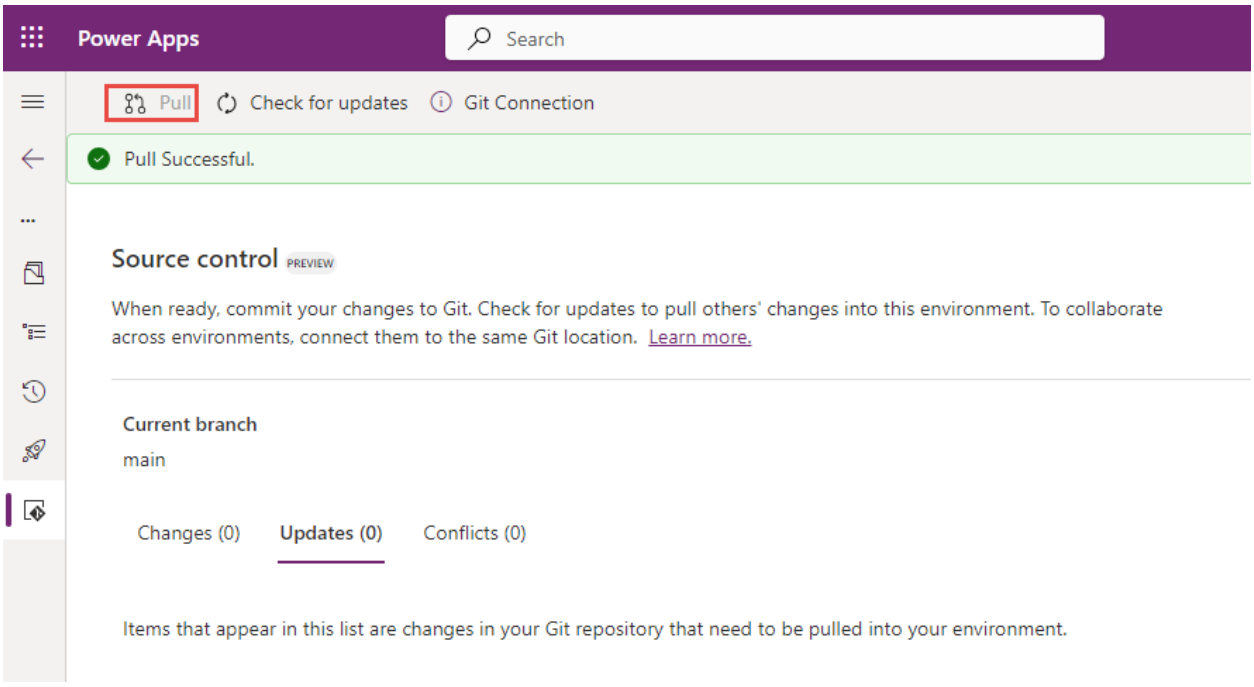
Enter a comment for the commit, use your branch name (use the same branch as you used to bind your environment to Git to avoid having to also use and complete a pull request). Click **Commit** to finish.



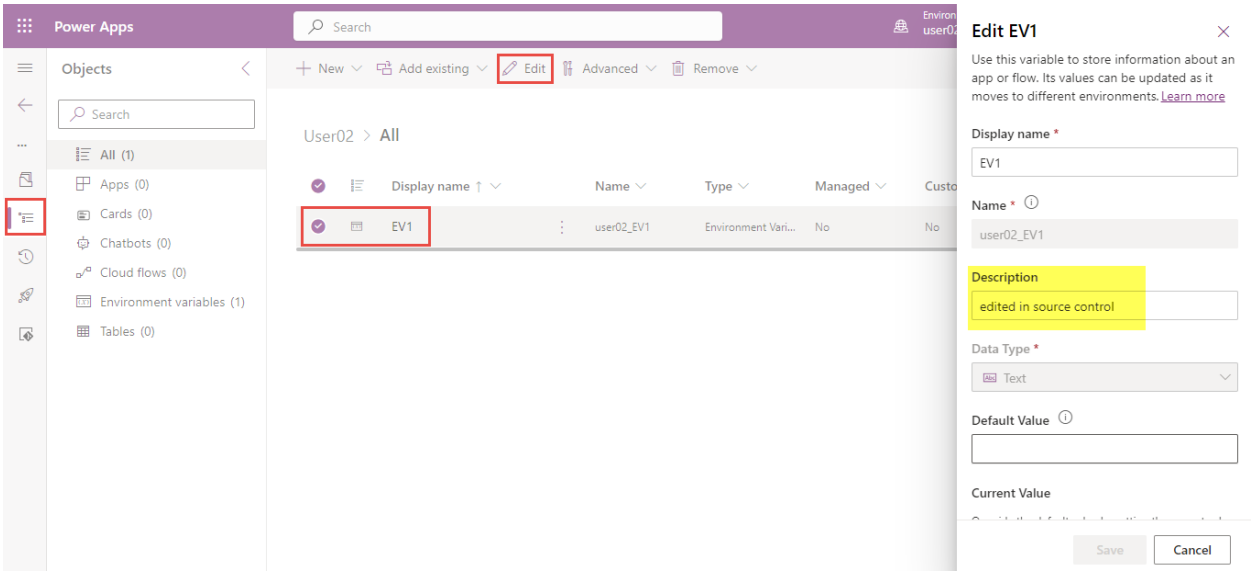
Switch back to your Power Apps tab in the browser and click **Check for updates**. You should now see one available update. Click the Updates tab to review.



Since there are no conflicts, you can now click the Pull button in the command bar to initiate the import of the environment variable change into your environment.



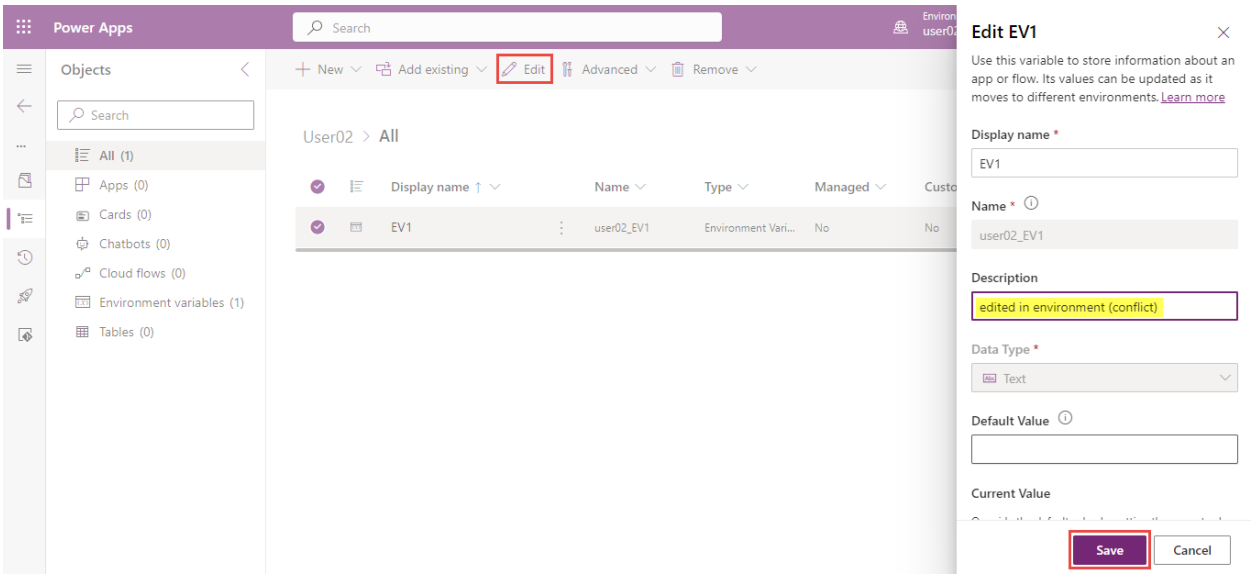
After your pull is completed, switch to the solution objects view using the left navigation, locate your environment variable and edit it.



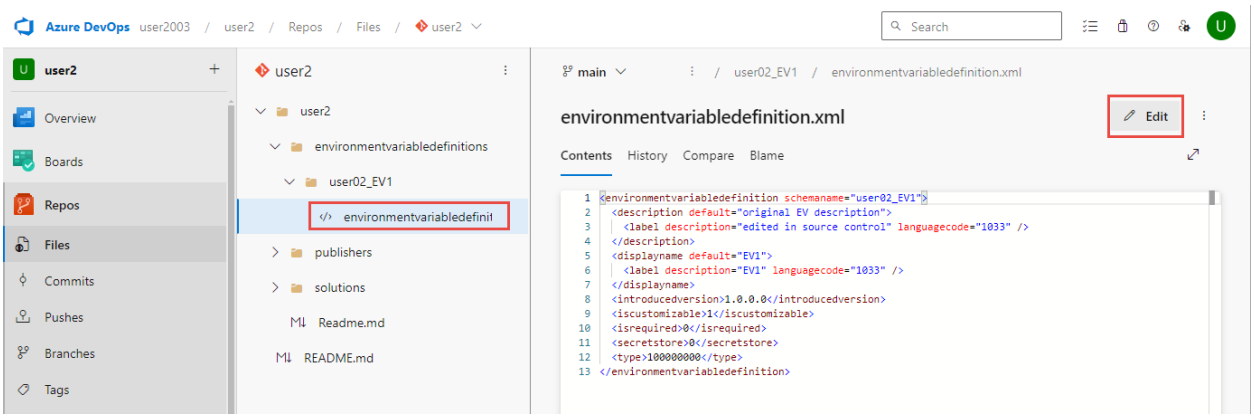
Notice that edits that you made in your source control system have been brought into your environment.

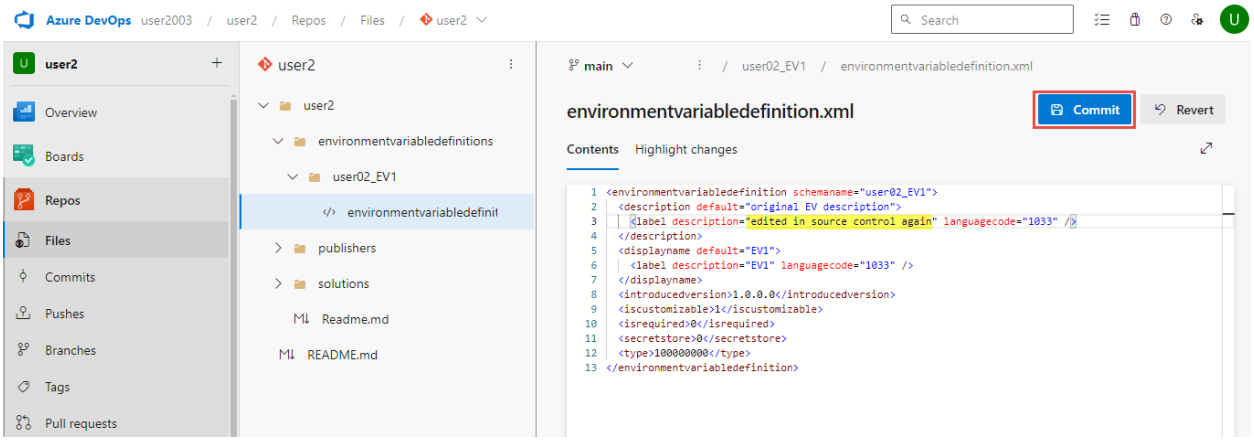
Task 3:

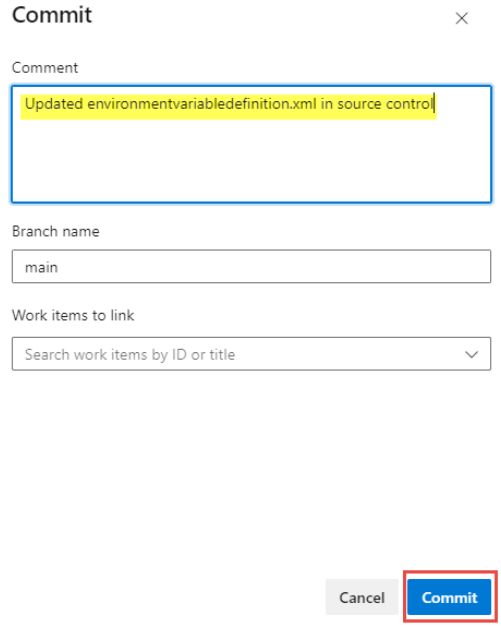
Let's create a conflict by making changes in source code and the environment and then syncing. Edit the existing environment variable and change the description to another new value.



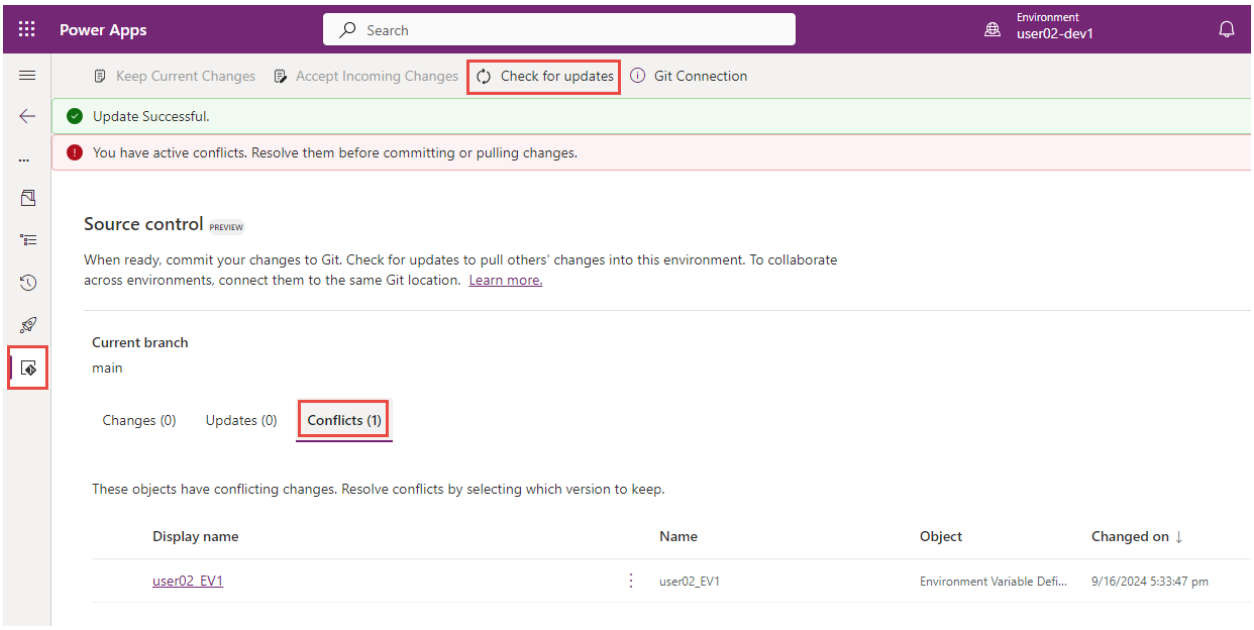
Locate the file in your source code again and commit a new PR with another value that is different than your environment.


Azure DevOps user2003 / user2 / Repos / Files / user2
Search
user2
Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
user2
environmentvariabledefinitions
user02_EV1
environmentvariabledefinit
publishers
solutions
Readme.md
README.md
main / user02_EV1 / environmentvariabledefinition.xml
environmentvariabledefinition.xml
Edit
Contents History Compare Blame
<environmentvariabledefinition schemaname="user02_EV1">
<description default="original EV description">
<label description="edited in source control" languagecode="1033" />
</description>
<displayname default="EV1">
<label description="EV1" languagecode="1033" />
</displayname>
<introducedversion>1.0.0.0</introducedversion>
<iscustomizable>1</iscustomizable>
<isrequired>0</isrequired>
<secretstore>0</secretstore>
<type>100000000</type>
</environmentvariabledefinition>

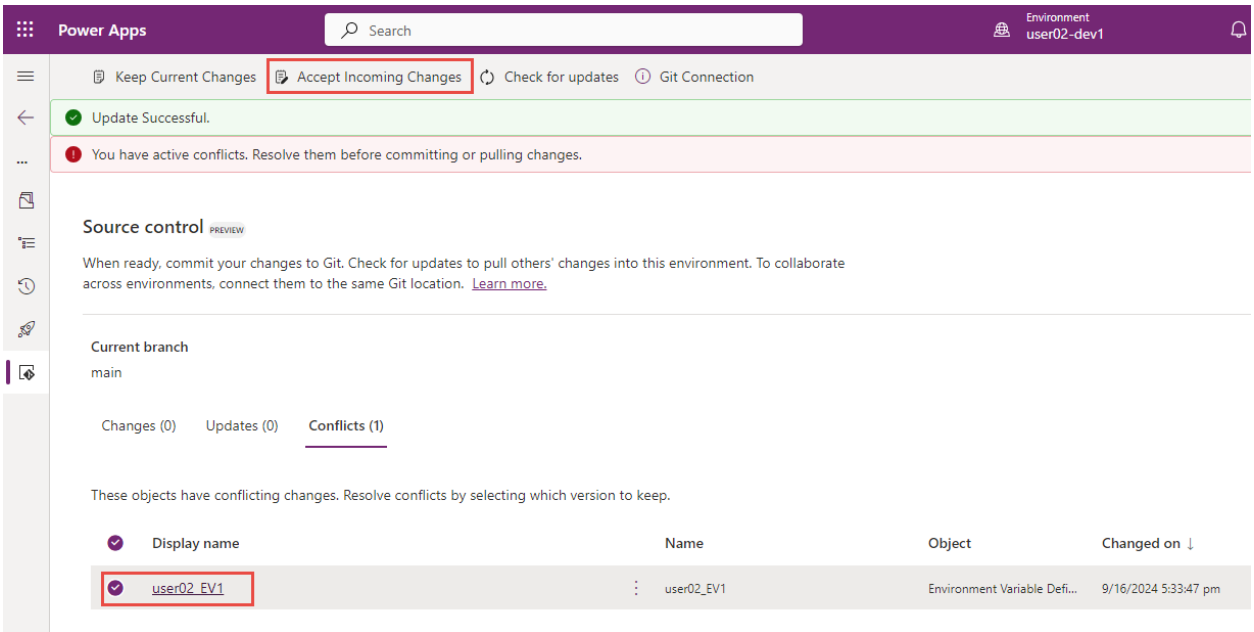

Azure DevOps user2003 / user2 / Repos / Files / user2
Search
user2
Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
Pull requests
user2
environmentvariabledefinitions
user02_EV1
environmentvariabledefinit
publishers
solutions
Readme.md
README.md
main / user02_EV1 / environmentvariabledefinition.xml
environmentvariabledefinition.xml
Commit
Revert
Contents Highlight changes
<environmentvariabledefinition schemaname="user02_EV1">
<description default="original EV description">
<label description="edited in source control again" languagecode="1033" />
</description>
<displayname default="EV1">
<label description="EV1" languagecode="1033" />
</displayname>
<introducedversion>1.0.0.0</introducedversion>
<iscustomizable>1</iscustomizable>
<isrequired>0</isrequired>
<secretstore>0</secretstore>
<type>100000000</type>
</environmentvariabledefinition>


Commit
Comment
Updated environmentvariabledefinition.xml in source control
Branch name
main
Work items to link
Search work items by ID or title
Cancel
Commit

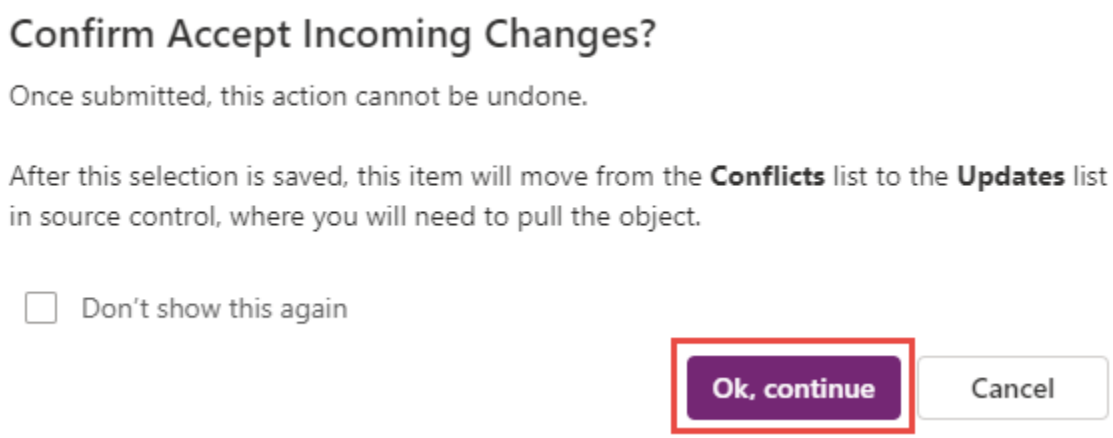
Switch back to the maker portal and select the source code tab. Click **Check for updates** and the system will detect a conflict and give you a warning message about it. Click on the Conflicts tab to review the conflicts. You will see that your environment variable is listed.



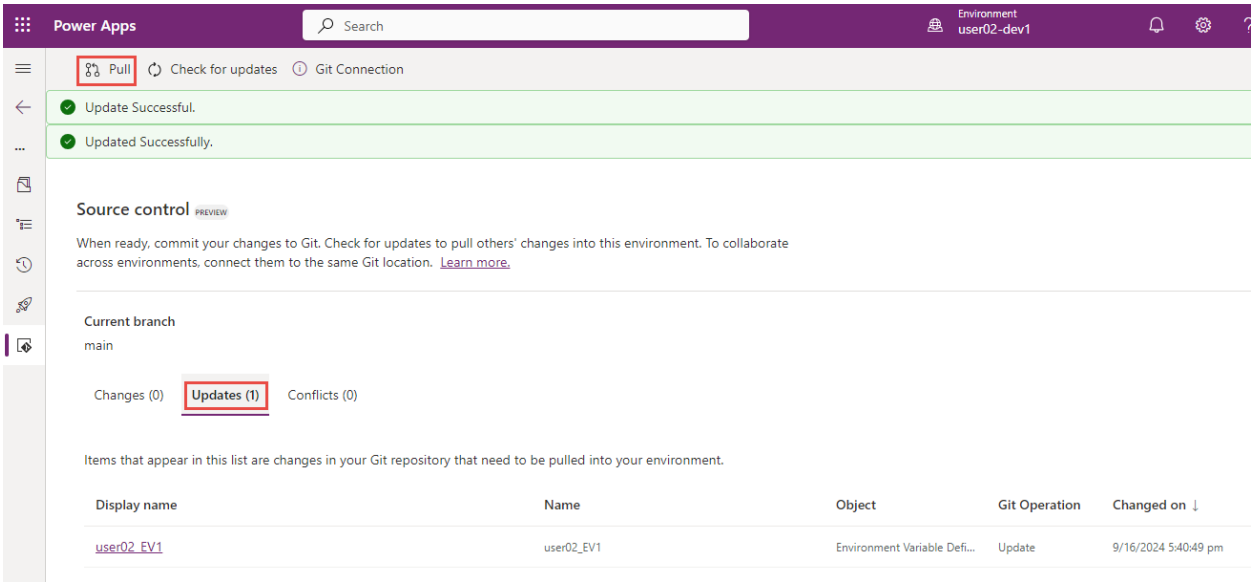
Select the environment variable, then choose which version to keep by clicking on the menu options **Keep Current Changes** to keep the changes in your environment, or **Accept Incoming Changes** to indicate that you want to accept the changes that were made in your source repository. For this task, we are going to select to accept the incoming changes.



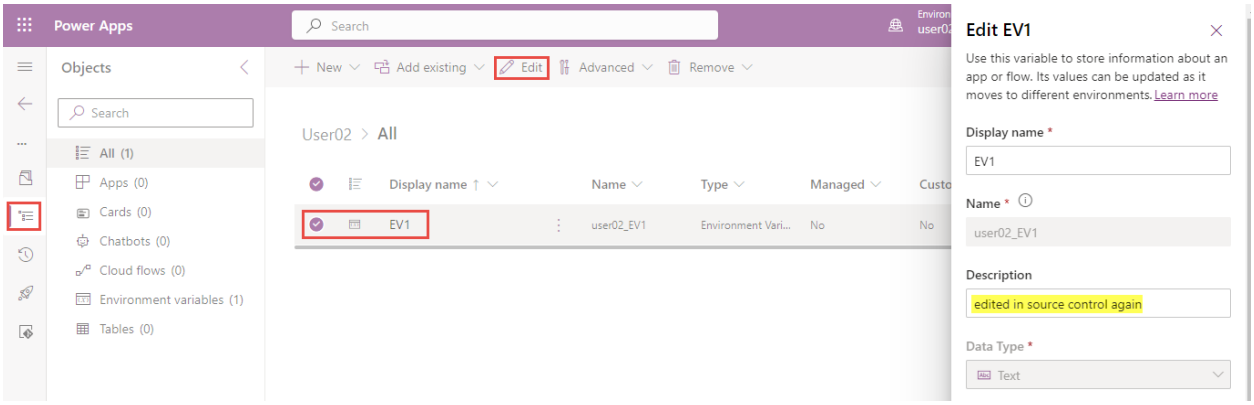
This will present a confirmation dialog of which you can accept.



At this point, the change strategy is remembered, but the data has not been synchronized with source code yet. The system will now track this as an update that is available to pull into your environment. Click the **Updates** tab to review it, then click **Pull** to complete the operation and bring that change into your environment.



Using the left navigation, switch to the **Objects** panel, select your environment variable, then click **Edit** to view it. Confirm that the description is consistent with the value you checked into source control.

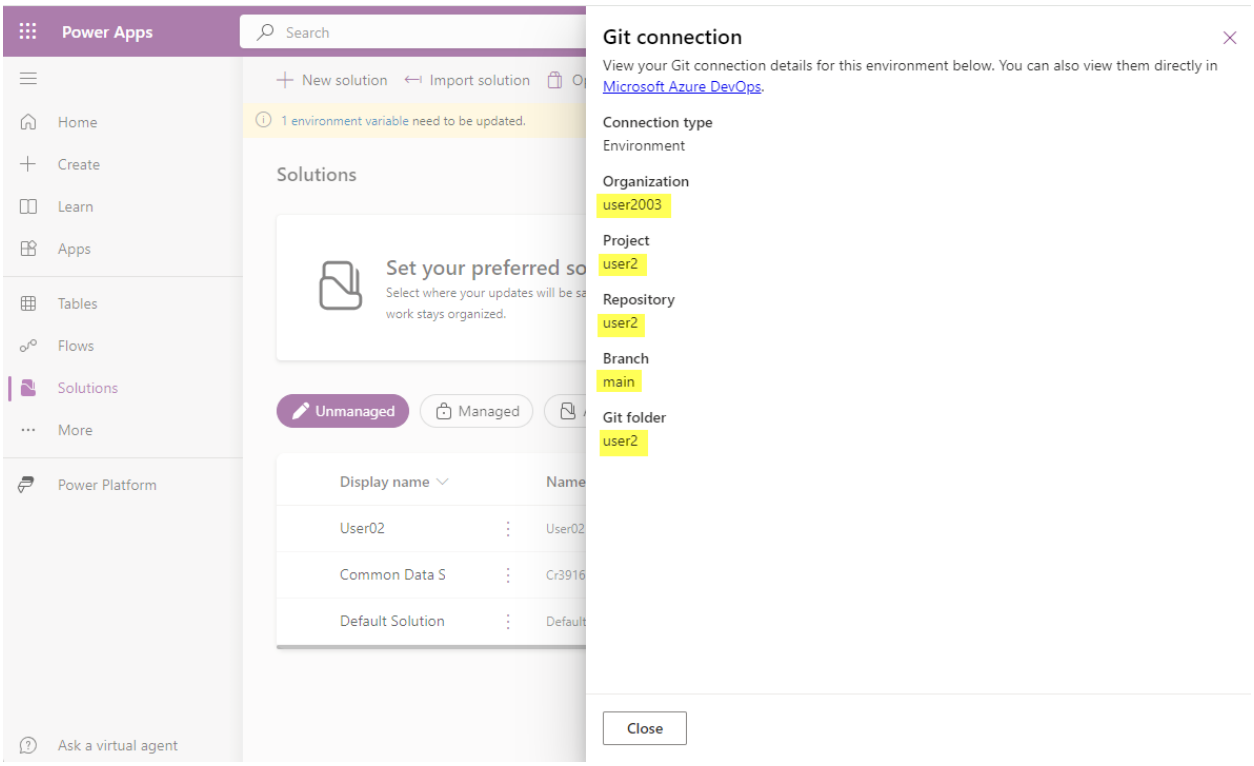


Task 3

This task will use source control integration to enable development across two environments, providing a level of isolation for makers.

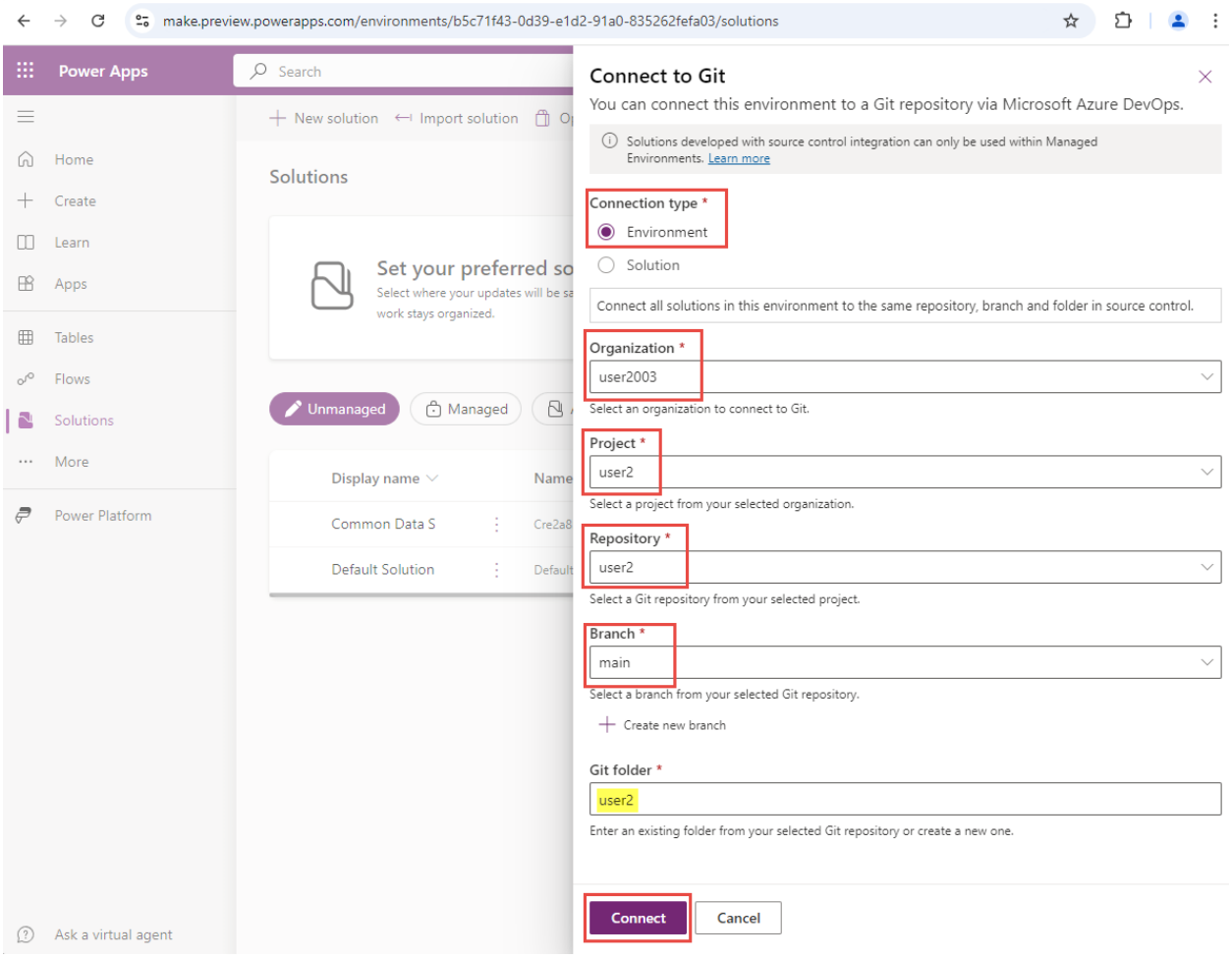
This task will require that you have a second development environment. As per the instructions in Lab 1, if you have not already created one, create a second development environment. For this lab, we recommend duplicating your tab and selecting the appropriate environment for each tab, making it easy to switch back and forth between the environments.

We need to connect the environment to the same source location. It is probably a good idea to duplicate this tab and select your first development environment. From there you can then click **Solutions**, then **Git Connection** to view the current settings.



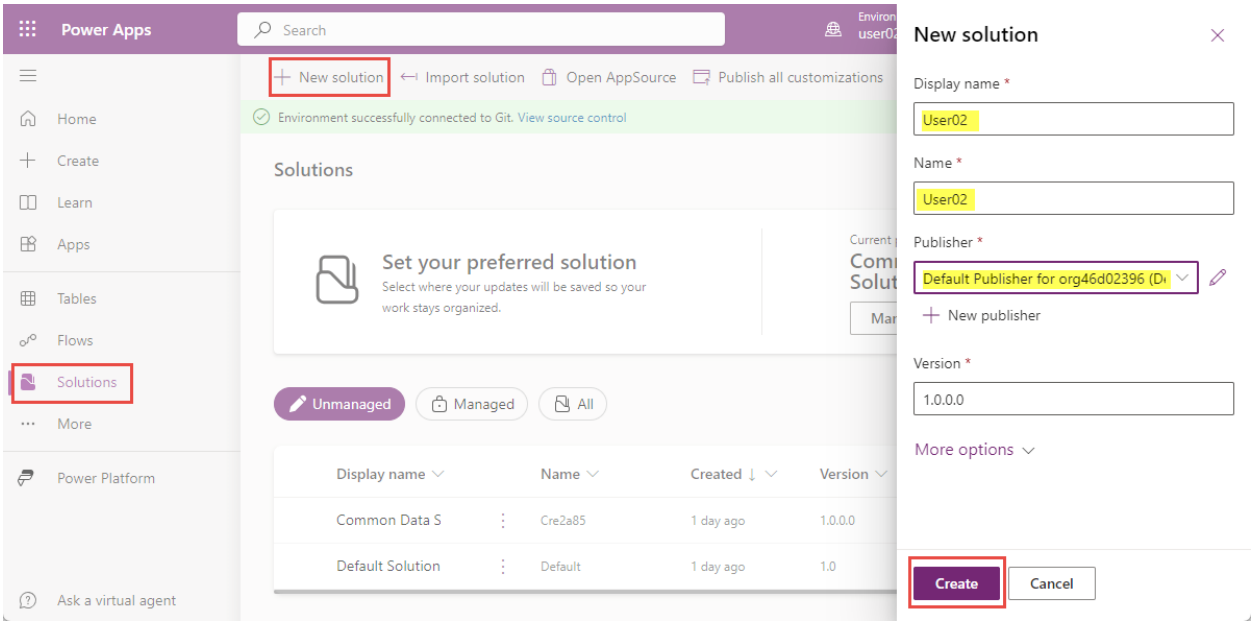
Switch back to the tab for your second development environment, select **Solutions** and then click **Connect to Git**. In the connection dialog, select the same connection type (environment), organization, project, repository, branch, and enter the same folder name.

Please double check ALL of the values as the present version of our feature does not have the ability to disconnect from Git or to reconnect to a different location. If you make a mistake, you will have to delete your environment and recreate it.

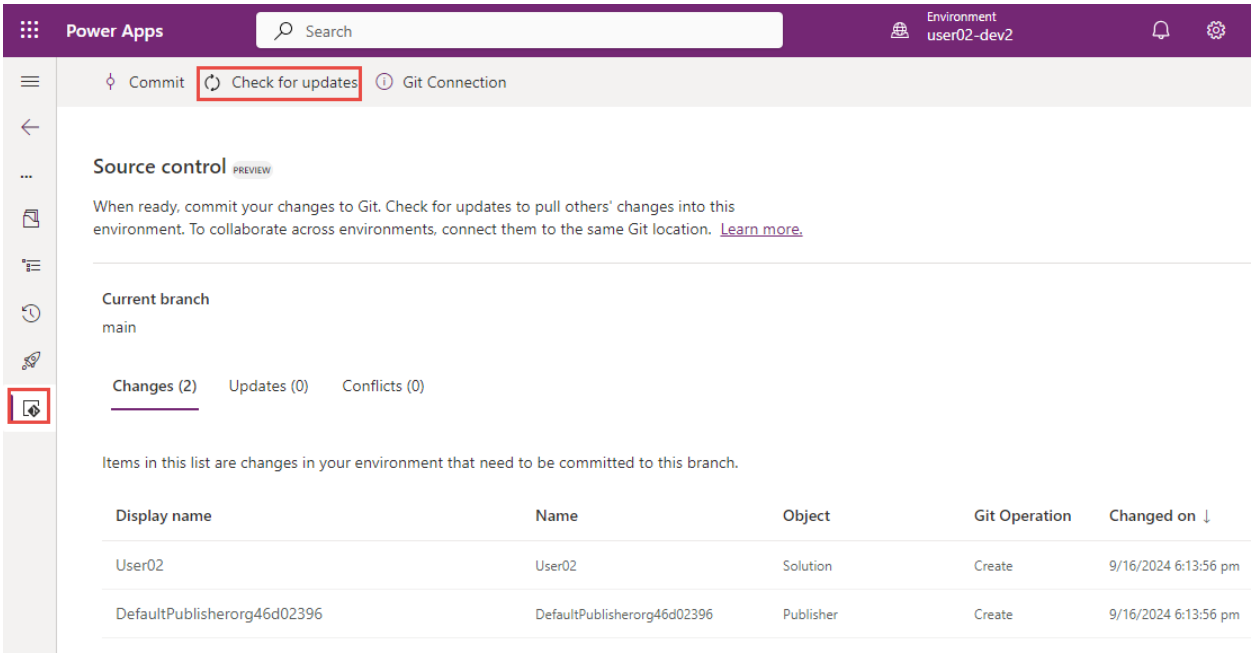


The system will only synchronize existing solutions in the environment, so we need to bootstrap the sync process by either importing a fresh solution that was exported from the other environment, or simply create a new solution with the same unique name. Switch to the Solutions area using the left navigation and click **+ New Solution** from the menu bar. Enter the same Name of the solution that you used in your first development environment. If you were following the instructions closely, it would be your username. You can switch to the appropriate tab and validate that name if you need to. You can simply use the

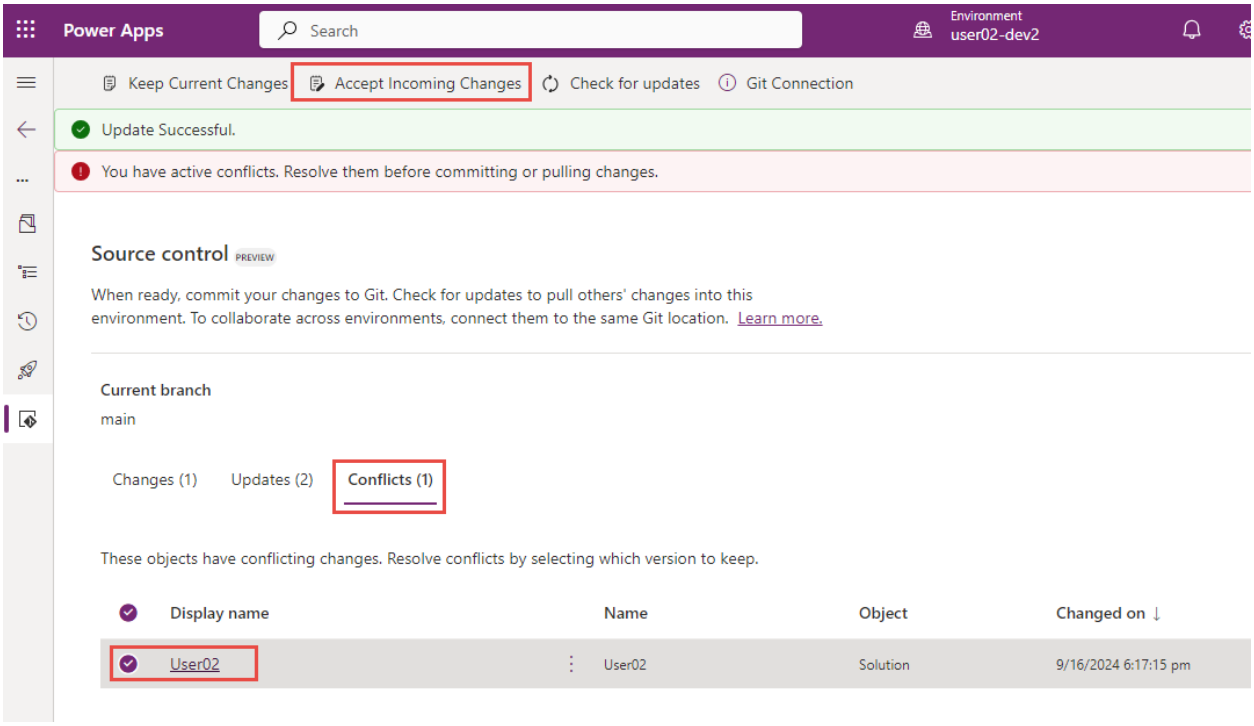
default publisher for your organization, or create a custom publisher that is the same as your other solution and then click **Create** to create the solution.



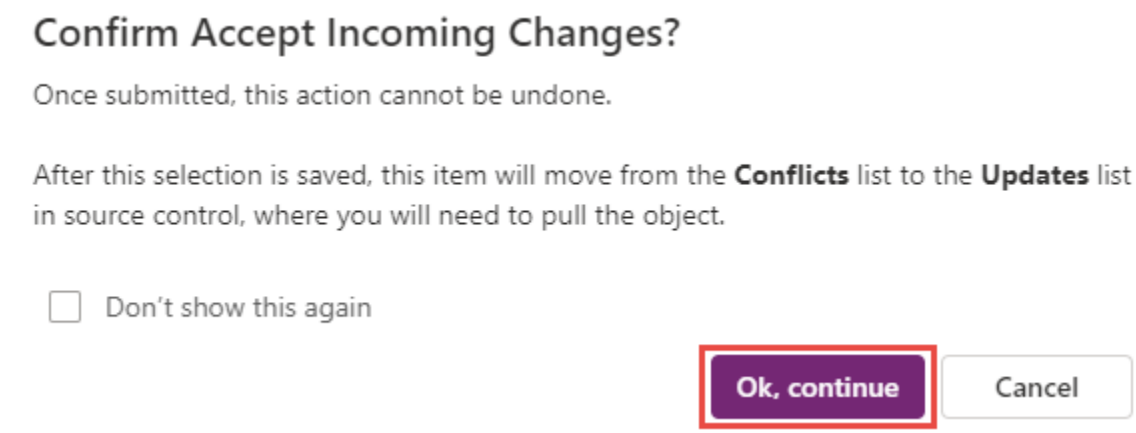
Once the solution is created, it will be available for synchronizing with your source repository. When the solution is created, switch to the source code tab in the left navigation. You will see that your solution and publisher are there. You don't want to check in the solution in its current state because it is empty. Click the **Check for updates button** and you will see the system detect a conflict.



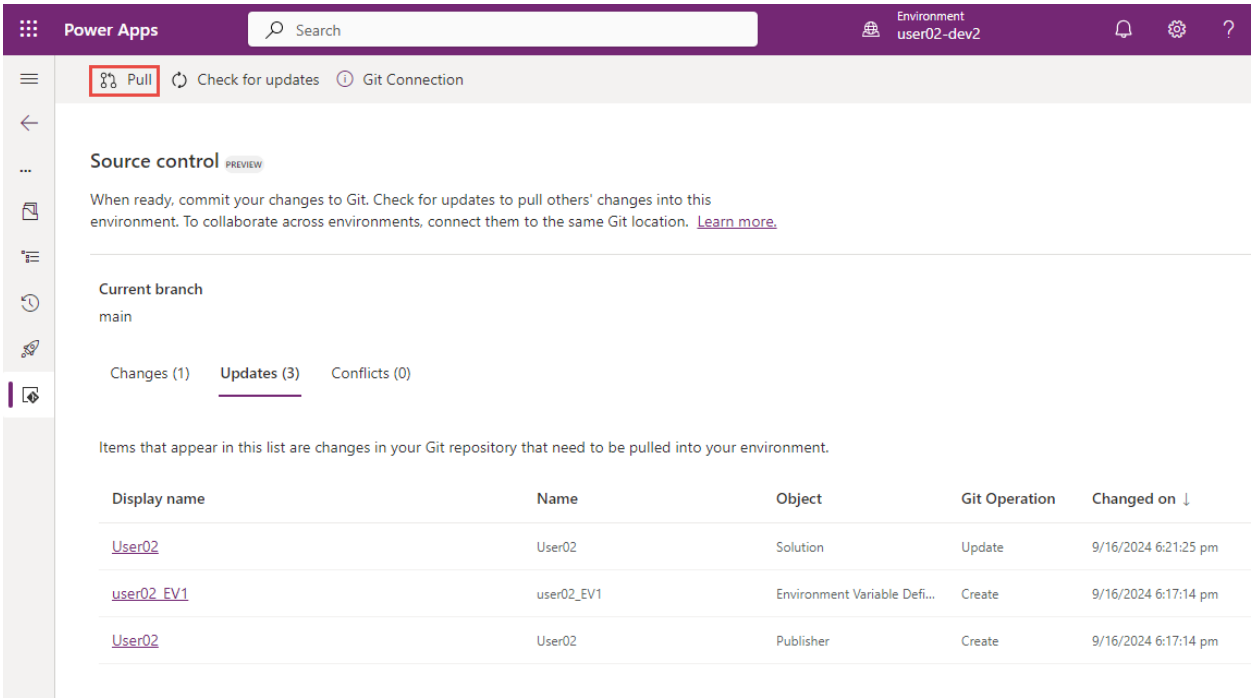
Select the **Conflicts** tab, then select the solution record from that list. Click the **Accept Incoming Changes** menu option to choose to accept the solution file from your source repository.



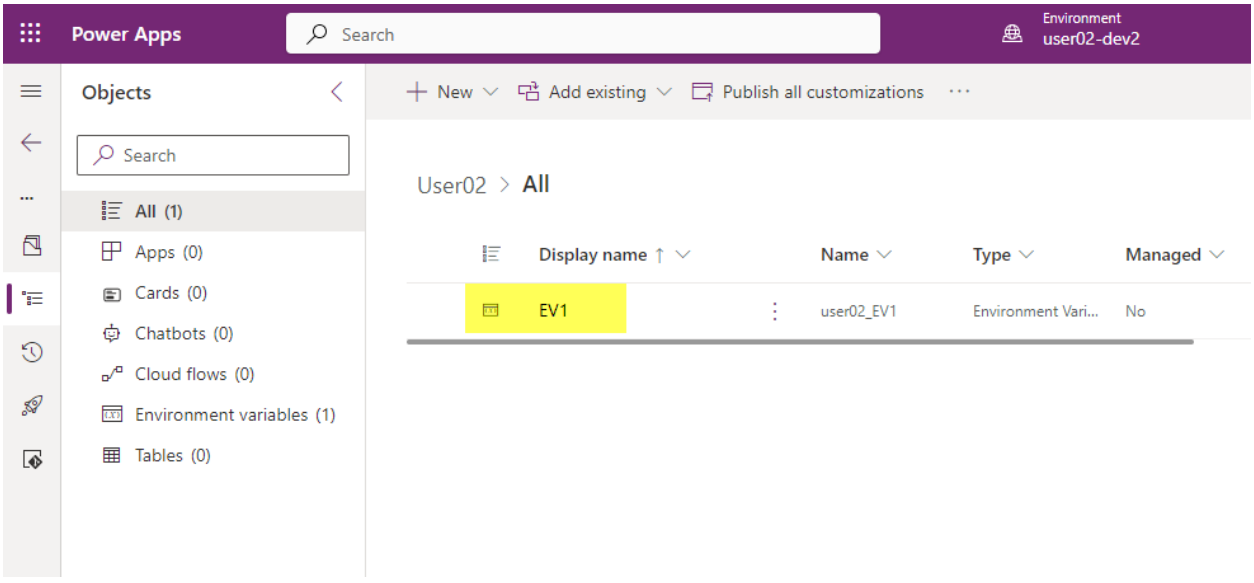
Confirm your selection in the confirmation dialog.



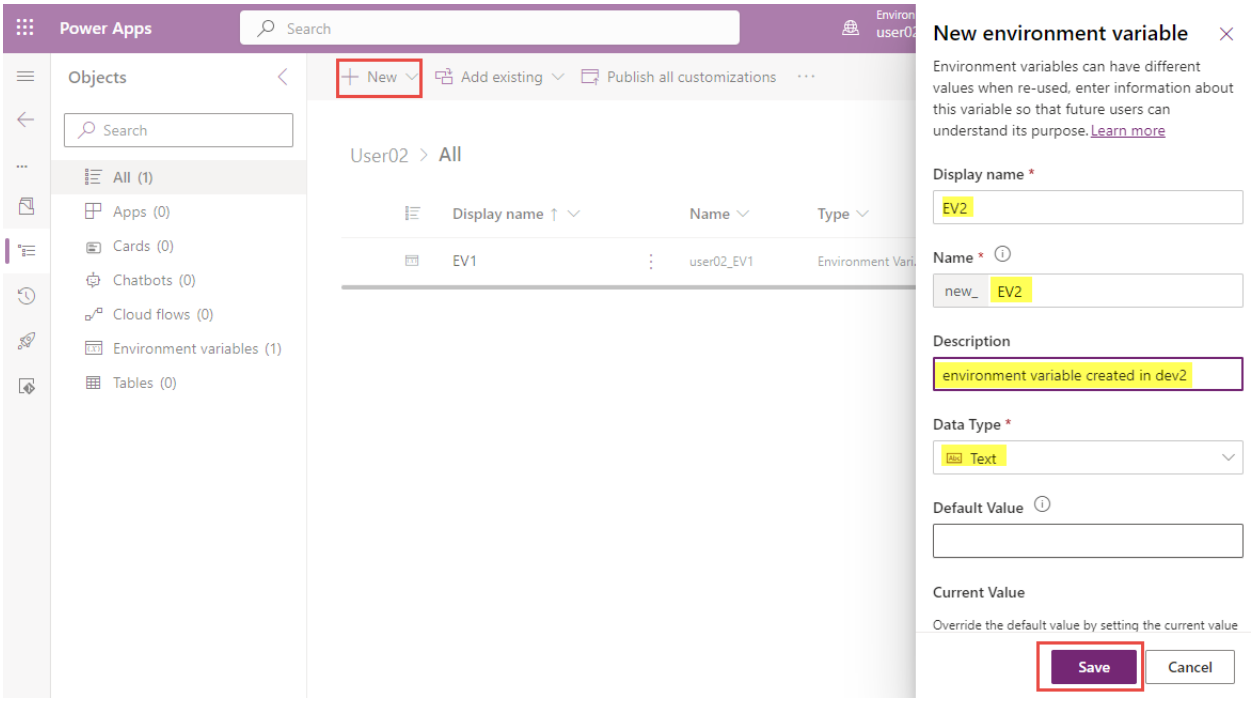
Switch to the **Updates** tab and review the changes. You should see your solution, publisher and environment variable component as available. Click the **Pull** button in the command bar to sync these into your environment.



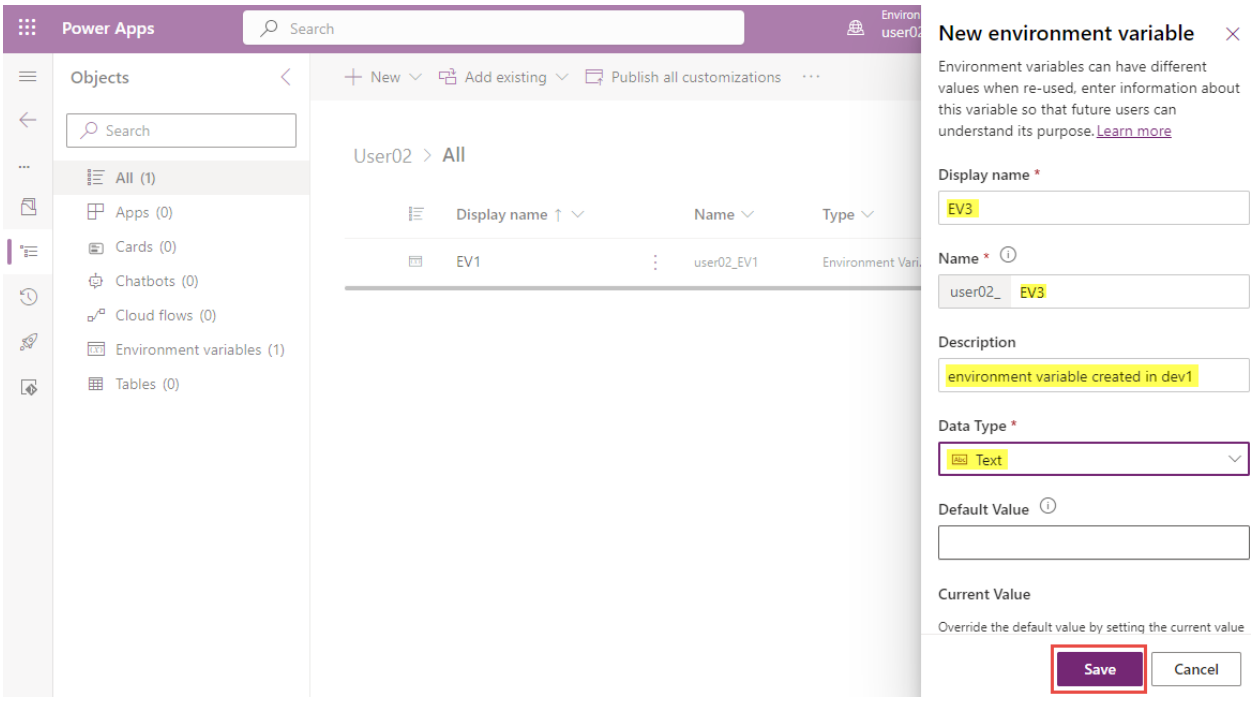
Switch to the **Objects** view using the left navigation and review that your solution objects are present.



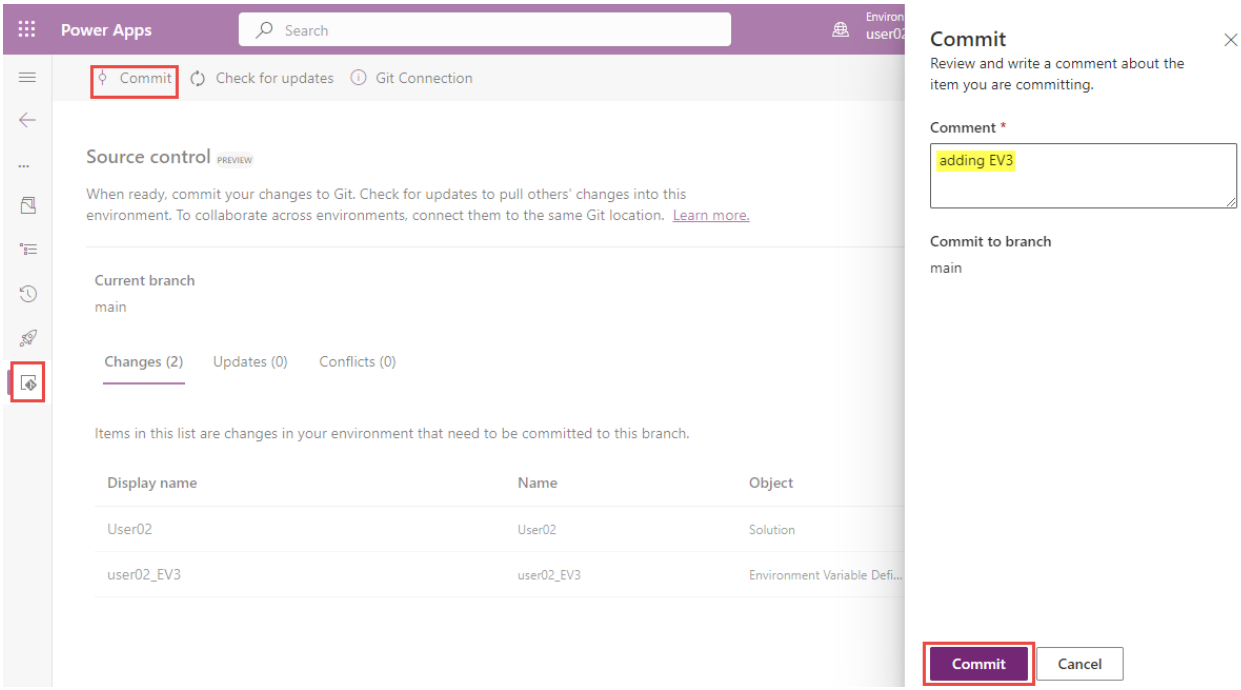
Let's create a new object in this environment. Click **+ New** | **More** | **Environment Variable** to create a new environment variable. Create a new environment called EV2.



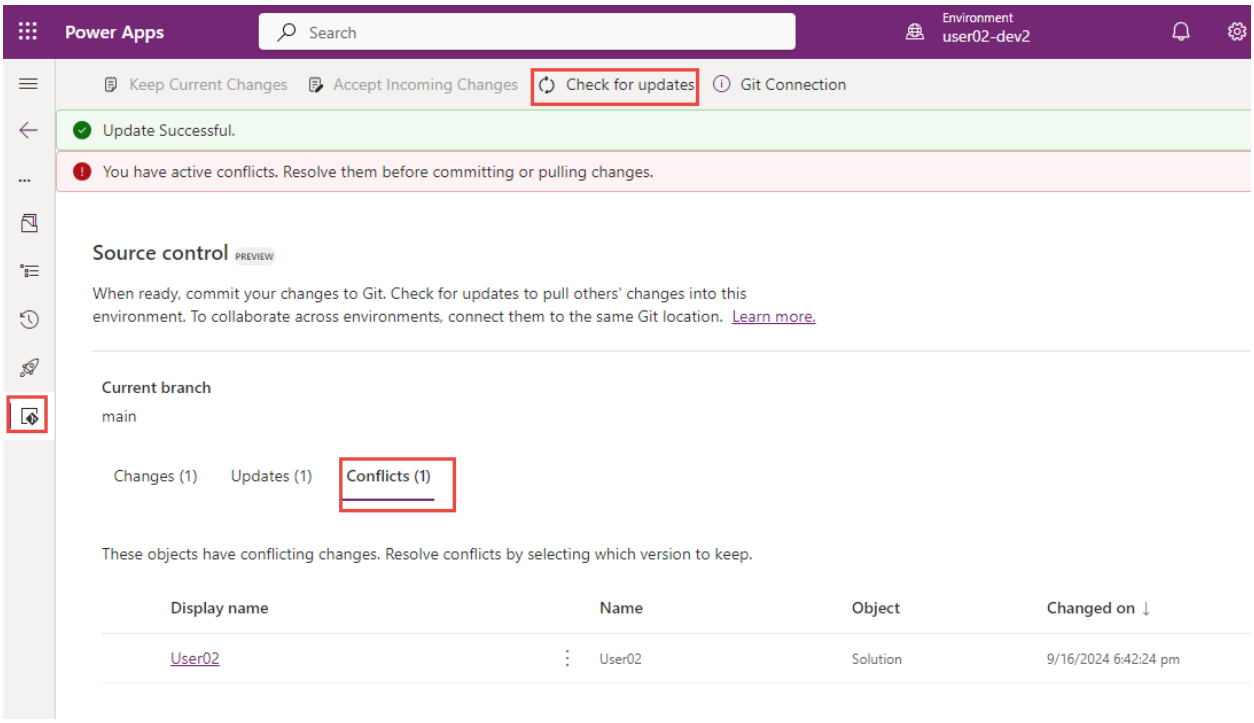
We want to show that changes can be made in either environment, so switch back to your first environment's tab in your browser and create another environment variable there, calling it EV3. If you closed that tab, you can simply select the environment from the environment list at the top of the maker portal.



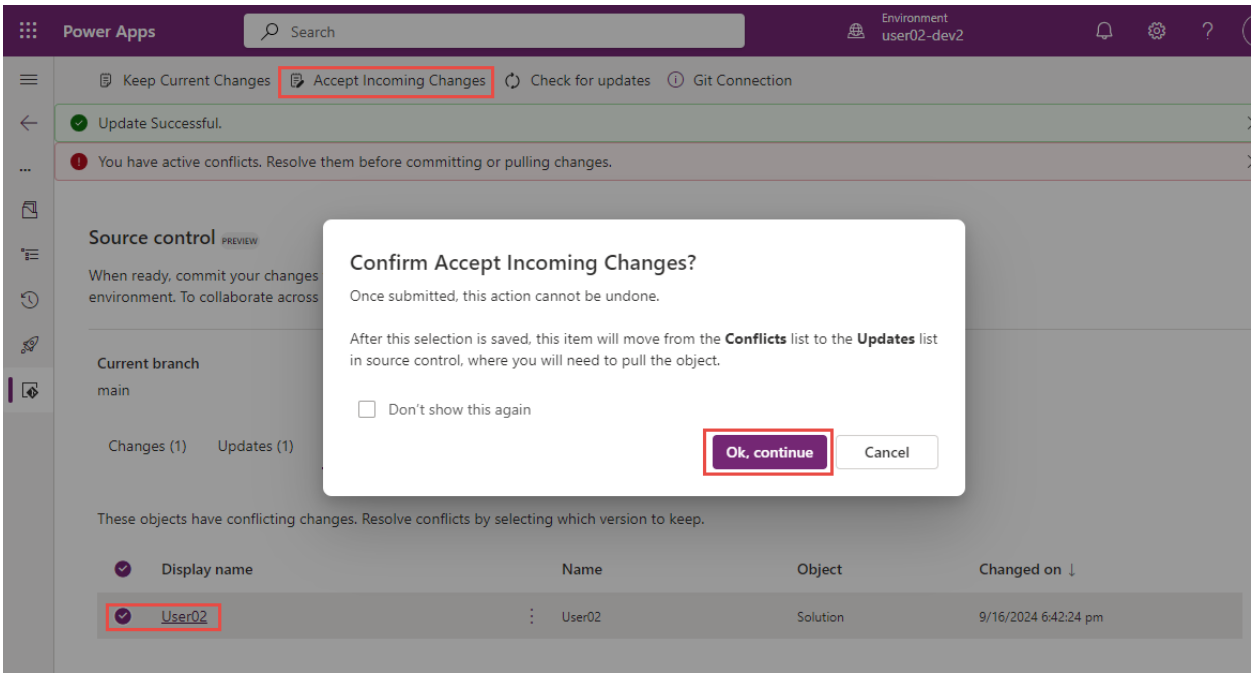
At this point, in the dev1 environment, you should have the original EV1 as well as EV3 in your solution and in dev2 environment you should have the original EV1 as well as EV2. In the dev1 environment, switch to the **Source Control** tab. Review the changes and click **Commit**. Enter your commit comment and click **Commit**.



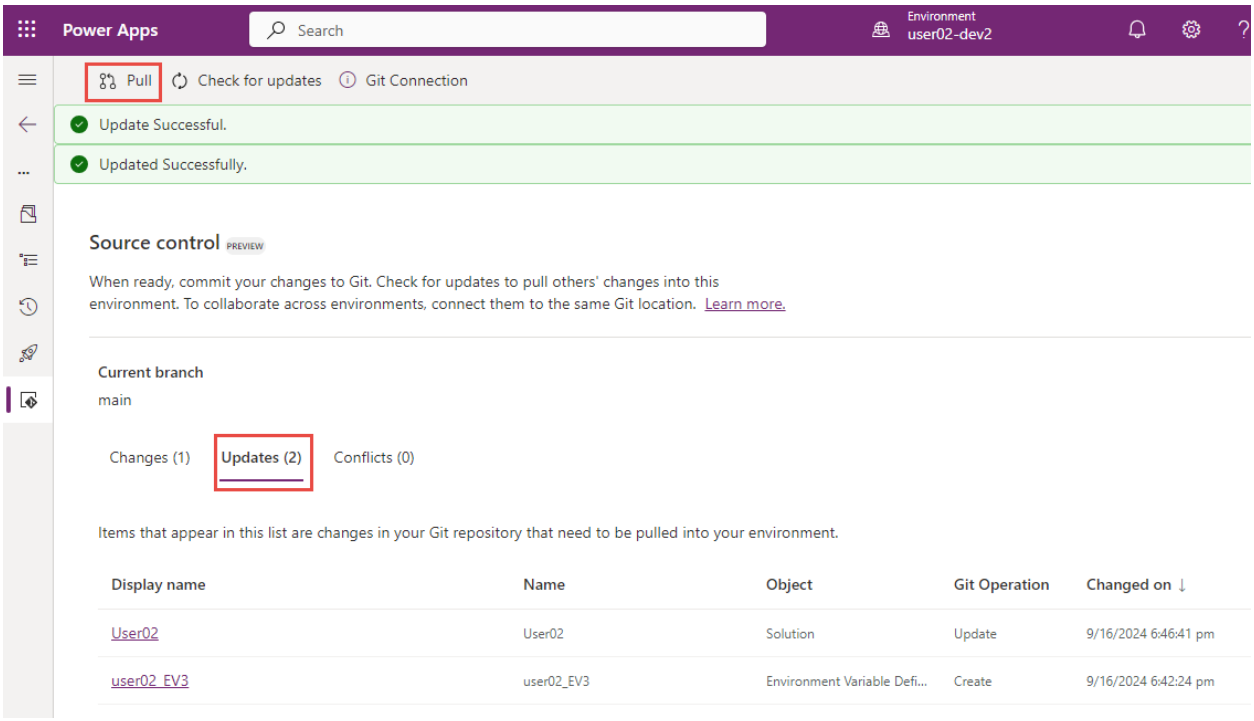
Switch to your browser tab for your first development environment, then click the **Source Control** tab in the left navigation. Click the **Check for updates** button and then click the **Conflicts** tab once the conflict is identified.



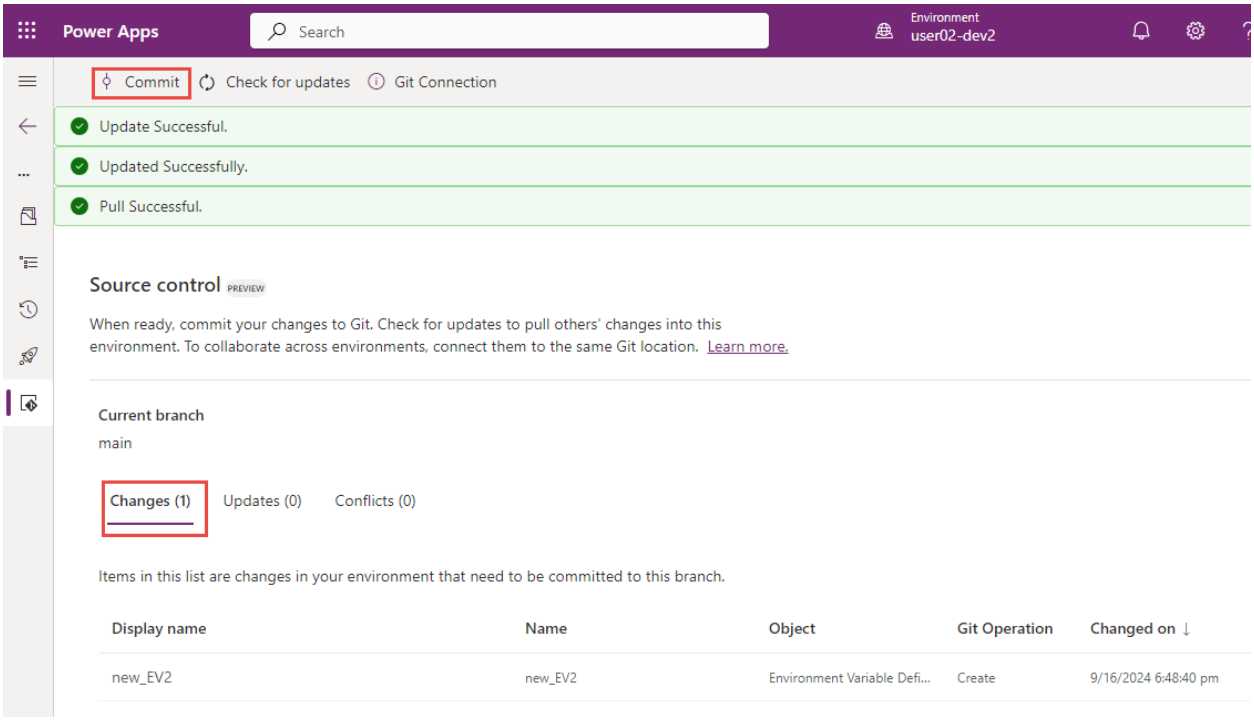
Select the solution conflict and then click the **Accept Incoming Changes** button on the command bar and then confirm the choice.



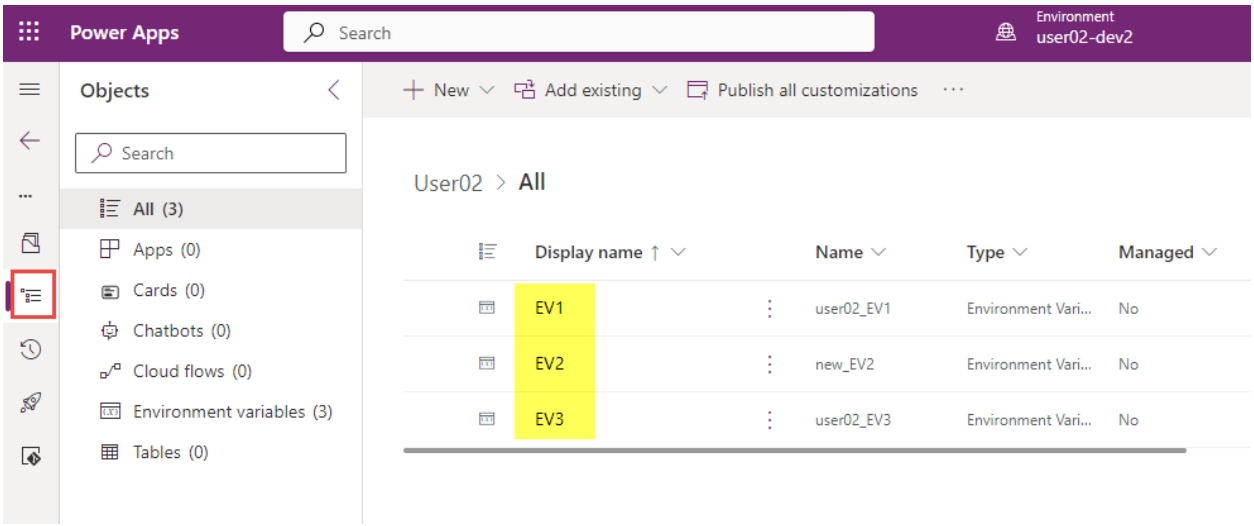
As the best practice is to pull before you push, switch to the **Updates** tab, review the incoming changes and click **Pull** to bring these into your environment.



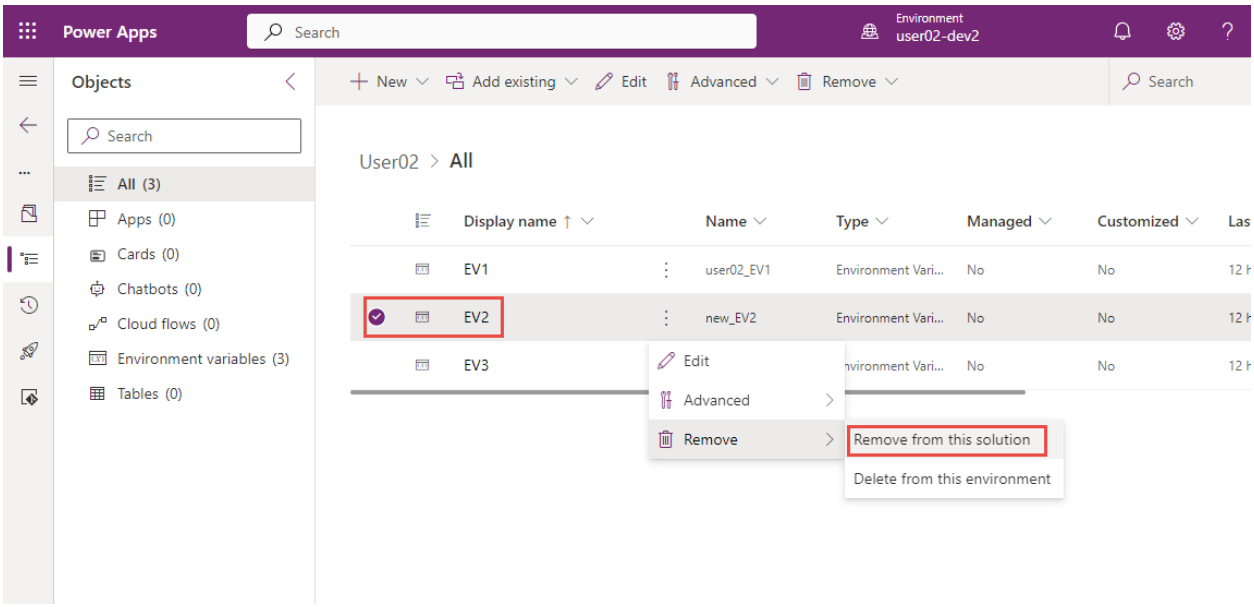
Select the **Changes** tab, review the new environment variable that is available to commit. Notice that the solution is not tagged as different. This is a small issue that we will fix shortly.



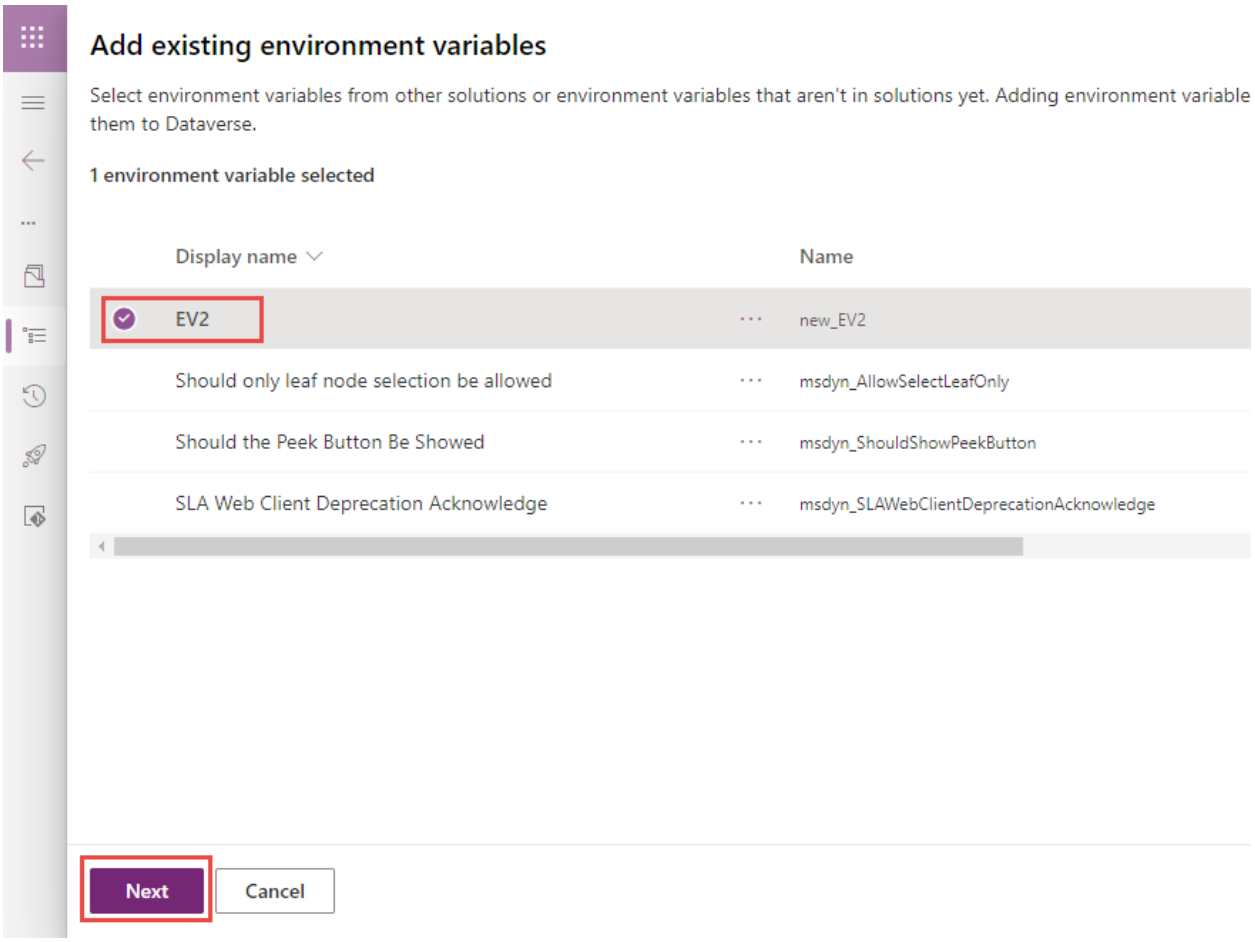
To fix this issue, we must change the solution. The simplest thing is to remove and re-add the component to the solution, which will mark the solution file as dirty. Switch to the objects tab and confirm that this environment has all three environment variables.



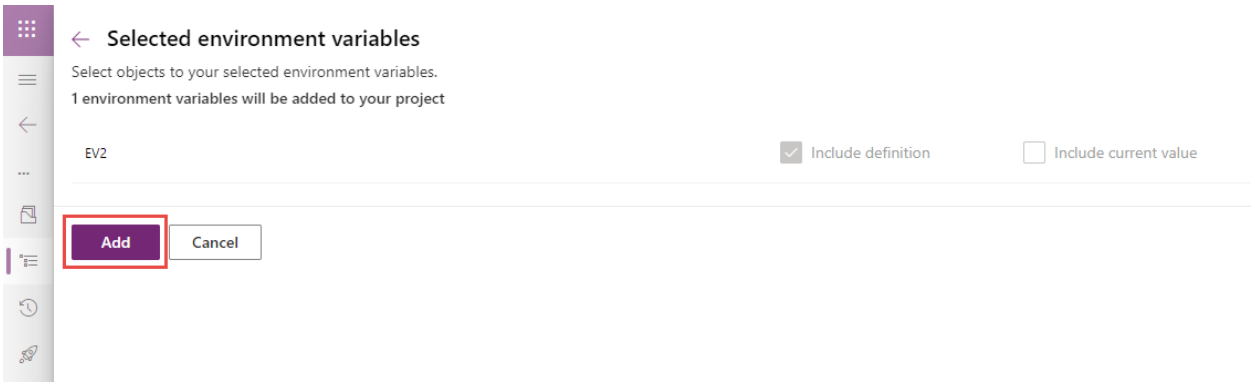
Select EV2, and remove it from the solution (but don't delete it)



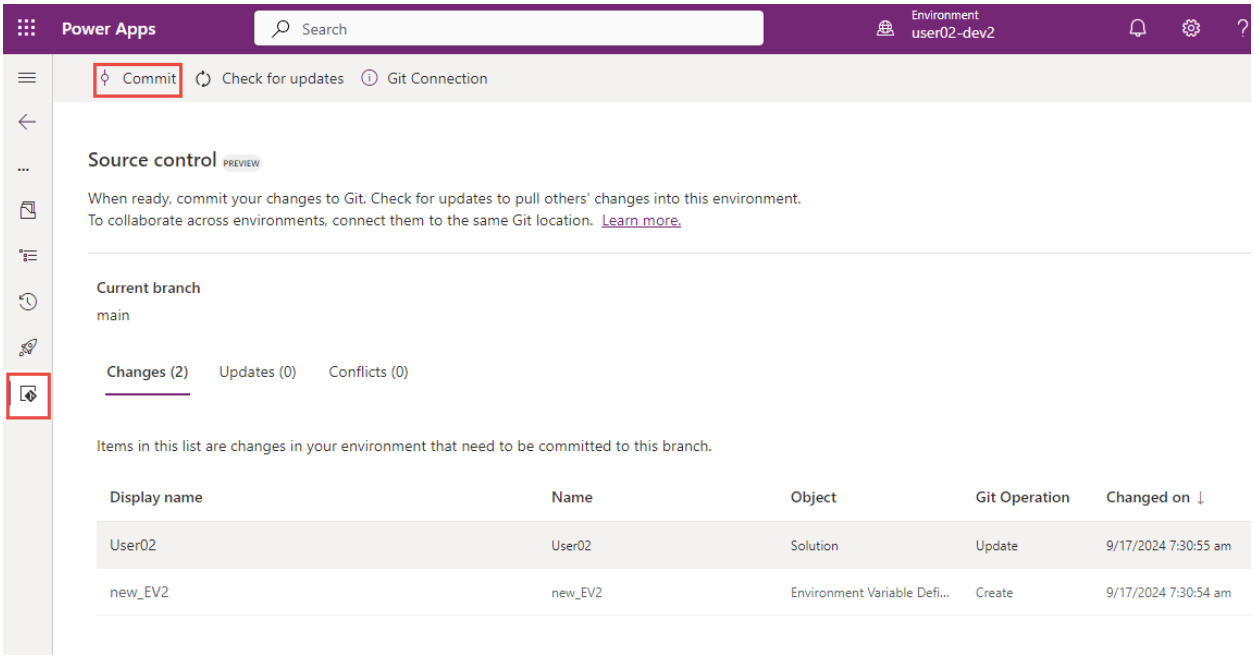
Then select **Add Existing** | **More** | **Environment variable** and select EV2 from the list.



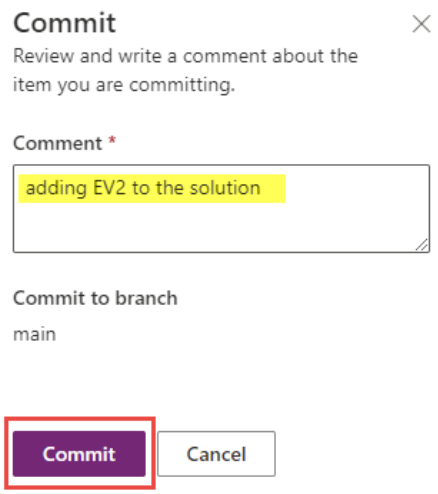
Confirm on the next page to add it back to the solution.



Switch to the **Source control** panel and review that you have the new EV2 and the solution update available to commit. Click the **Commit** button to initiate that action.

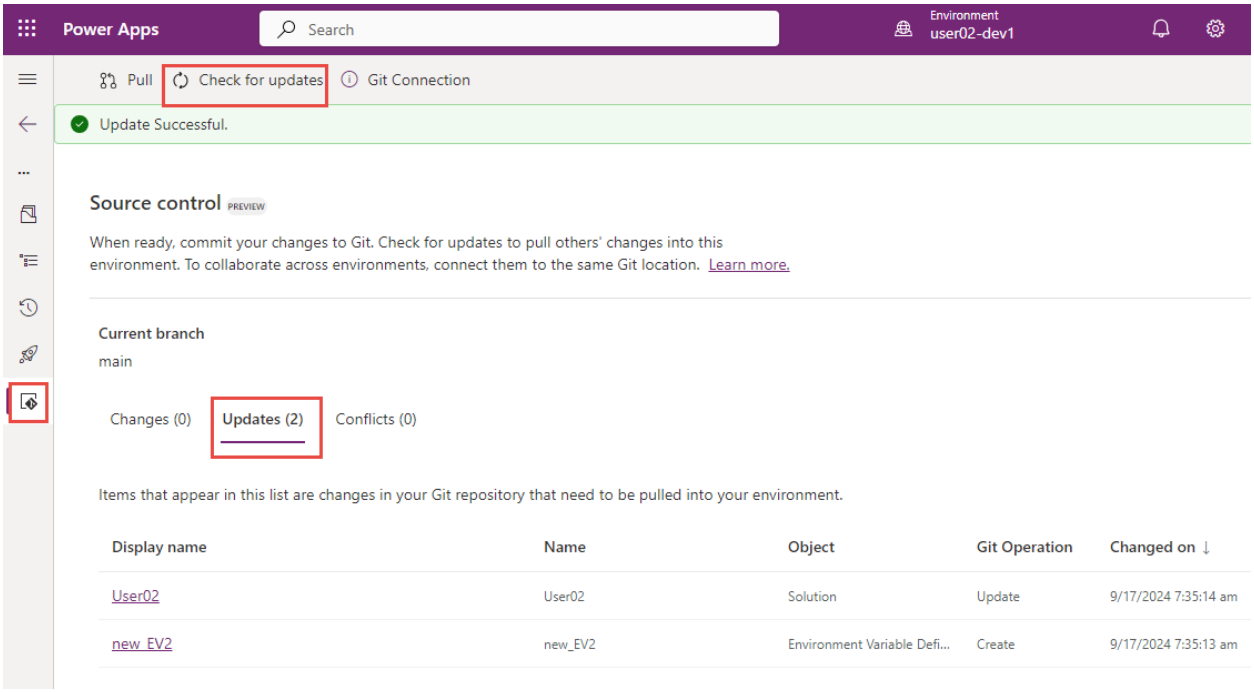


Enter a commit comment and complete the commit.

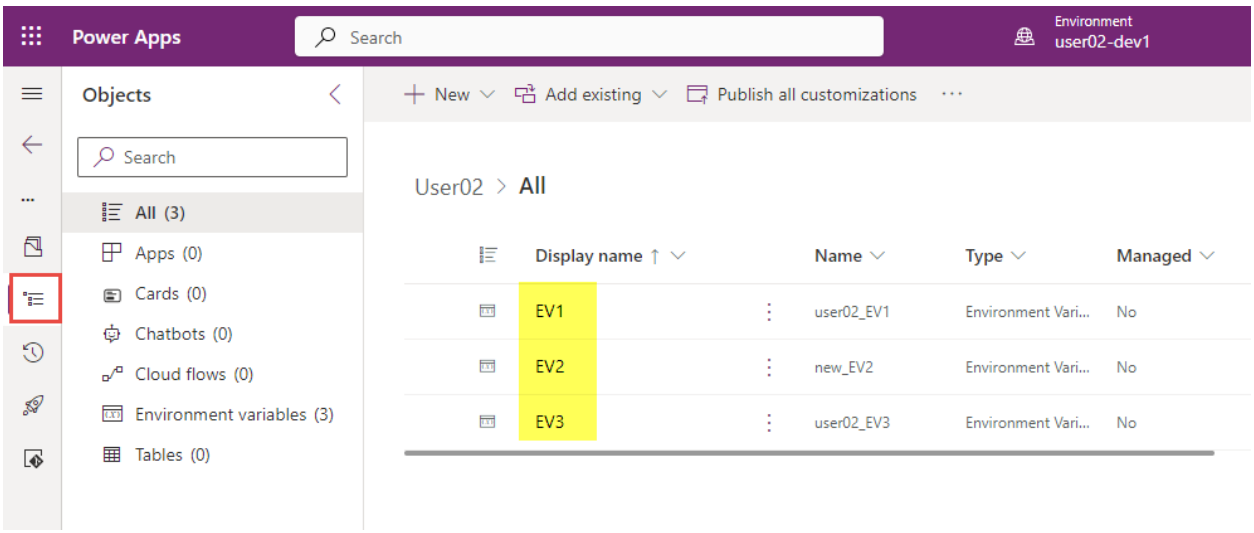


At this point, your second development environment is fully synchronized with source control and the changes that were made in the first development environment, but the first development environment doesn't have the new changes you just committed. To complete the process, simply switch back to the browser tab for your first development environment, navigate to the **Source Control** view, then click **Check for Updates**. The

system will detect 2 available updates. Click on the **Updates** tab to review the updates, which should include the solution and EV2.



Click the Pull button on the command bar to bring these into your environment. Then switch to the objects view and confirm all three environment variables are here too. You may have to wait for the spinner to complete as the view will refresh its contents.



At the end of this process, you will have successfully made changes in two separate environments using your source code repository as a source of truth. The development in each environment was done in complete isolation of the other environment.