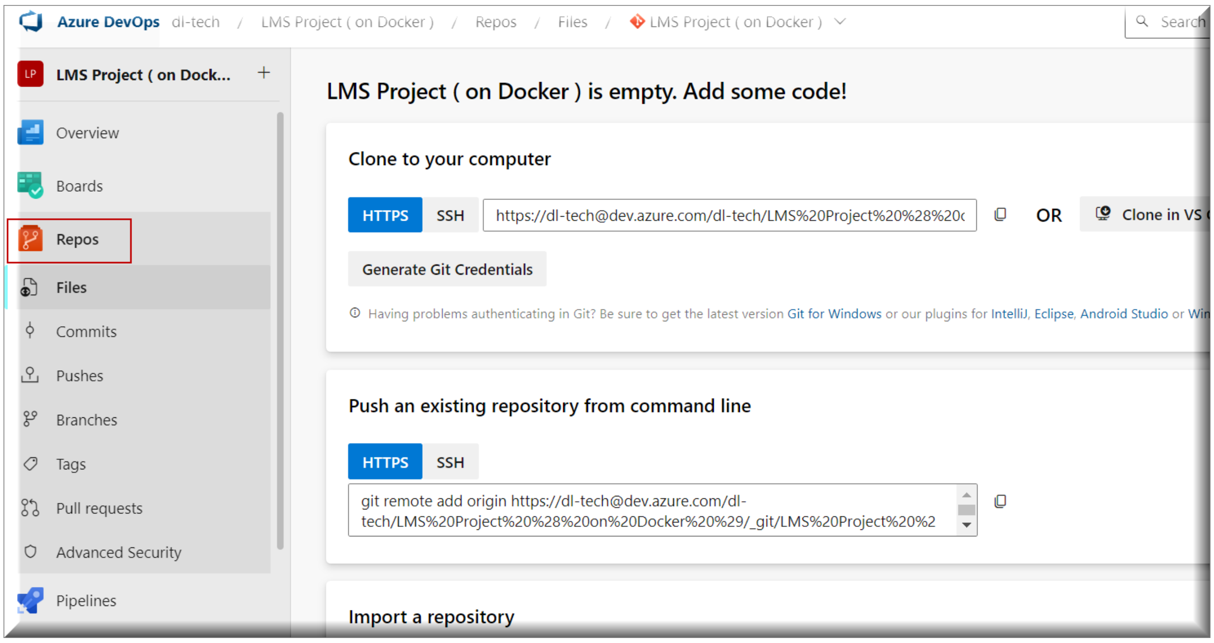
Azure Repos

When you create a new project in Azure DevOps, you get a new git repository with the same name as your project. You can see this when you click on the side menu item **Repos** as shown in Figure 61. As

depicted in the figure, there are several ways you can store your source code and other resources on a remote git repository.



LMS Project (on Docker) is empty. Add some code!

Clone to your computer

https://dl-tech@dev.azure.com/dl-tech/LMS%20Project%20%28%20c

Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Wi

Push an existing repository from command line

```
git remote add origin https://dl-tech@dev.azure.com/dl-
tech/LMS%20Project%20%28%20on%20Docker%20%29/_git/LMS%20Project%20%2
```

Import a repository

1. **Clone to your computer. (From Azure Repos to your computer)**

Using this method, you can take a copy of the repository and download it to a directory in your computer using the following command, or you can directly clone it to Visual Studio Code.

However, in this case it will be an empty repository.

git clone {{your repository URL}}

1. **Push an existing repository to Azure Repos (From your computer to Azure Repos)**

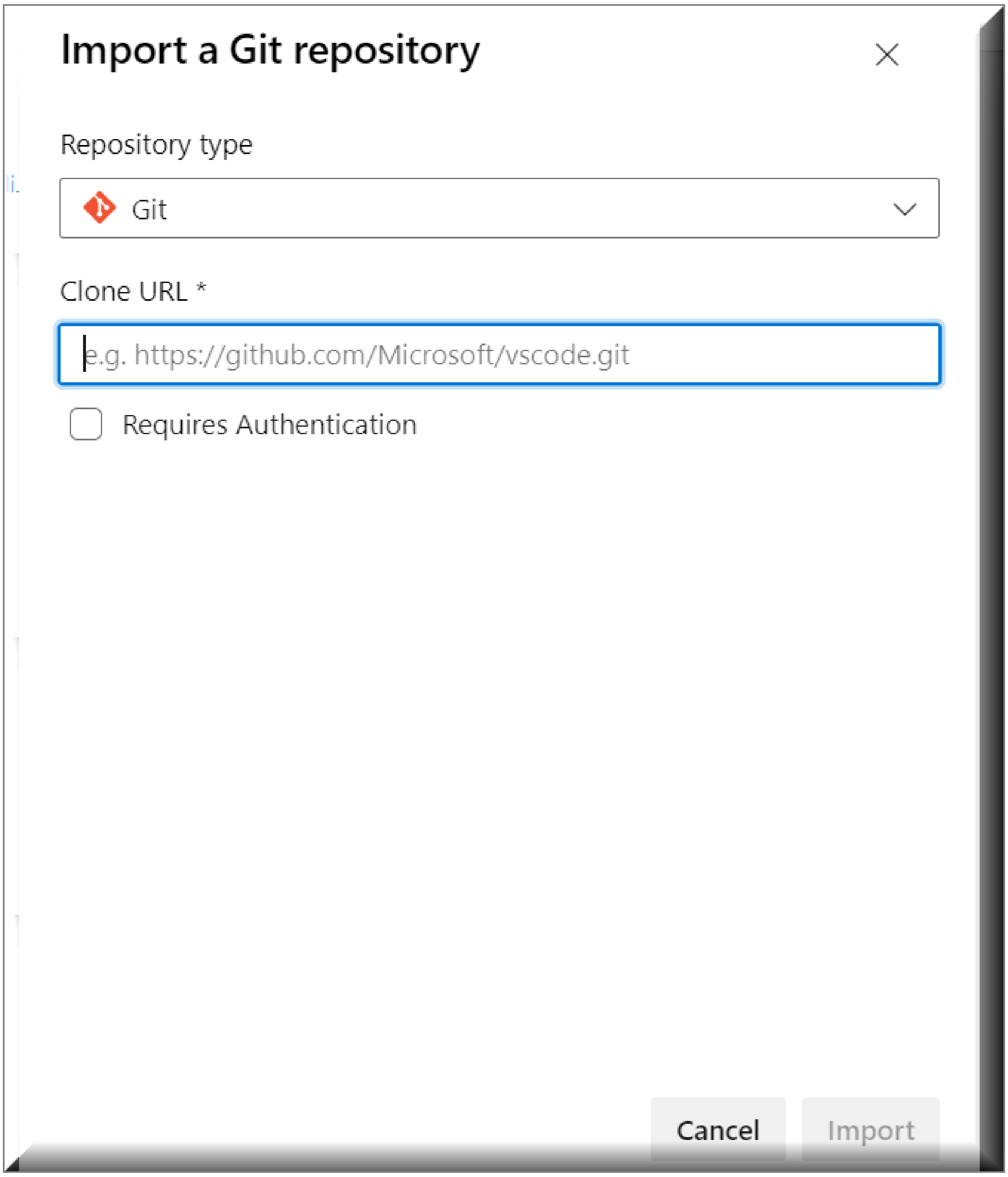
You can use this option if you already have your project files inside a folder in your computer. Then it is a matter of just transferring your files from your computer to Azure Repos. The following two commands must be run from your project folder.

git remote add origin {{your repository URL}} git push -u origin –all

1. **Import an existing repository (From another Git Repo to Azure Repos)**

If you already have a git repository in GitHub, Bitbucket, GitLab or any other location, then you can use this option to import that repository to Azure Repos as shown in Figure 62.

To issue git commands, you will need to download and install git software to your computer. For windows, download it from https://git-scm.com/download/win



Import a Git repository

Repository type

Git

Clone URL *

e.g. https://github.com/Microsoft/vscode.git

Requires Authentication

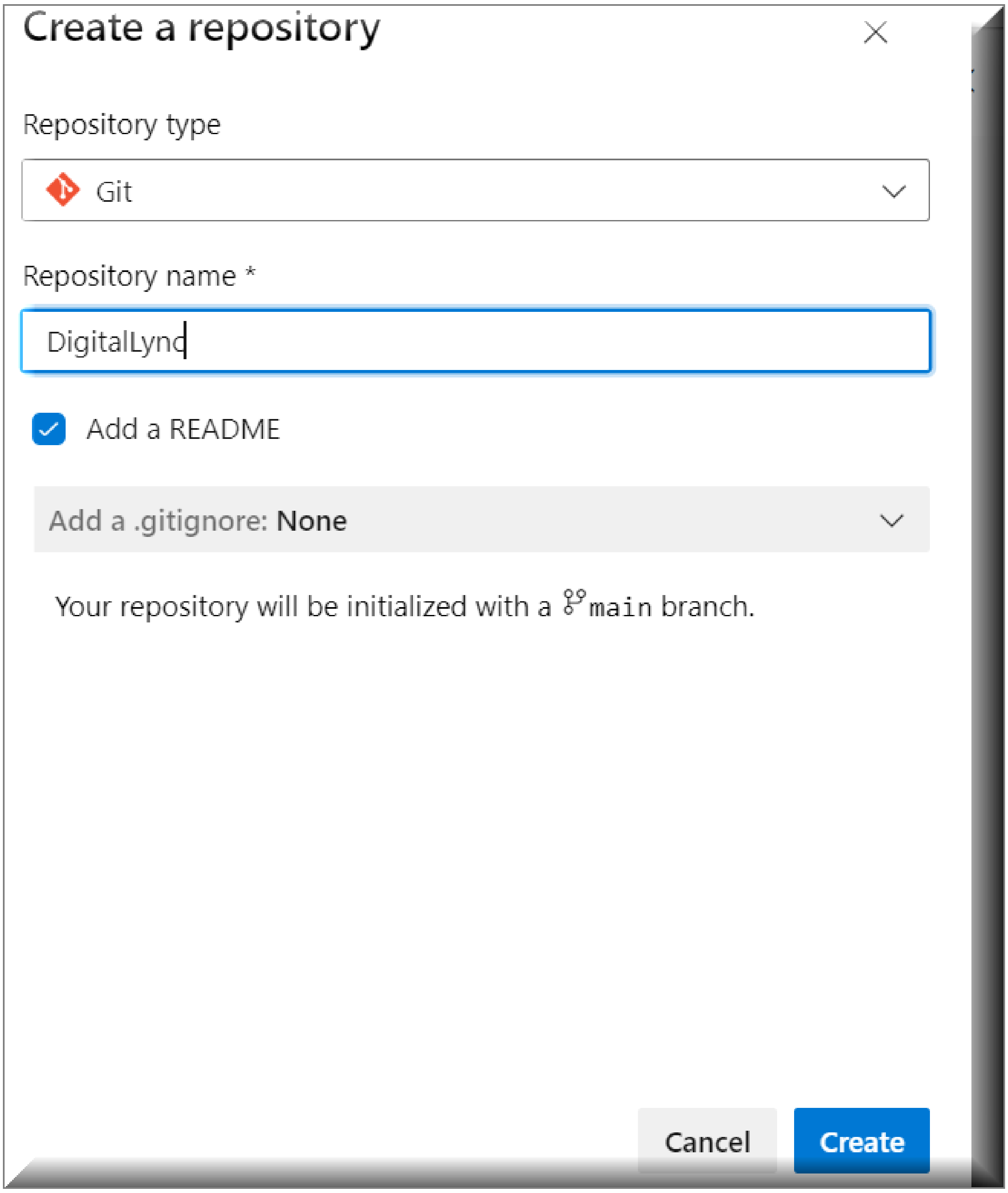
. Remember to uncheck the checkbox

Add a README

and click on the

Create

button.


Create a repository

Repository type

Git

Repository name *

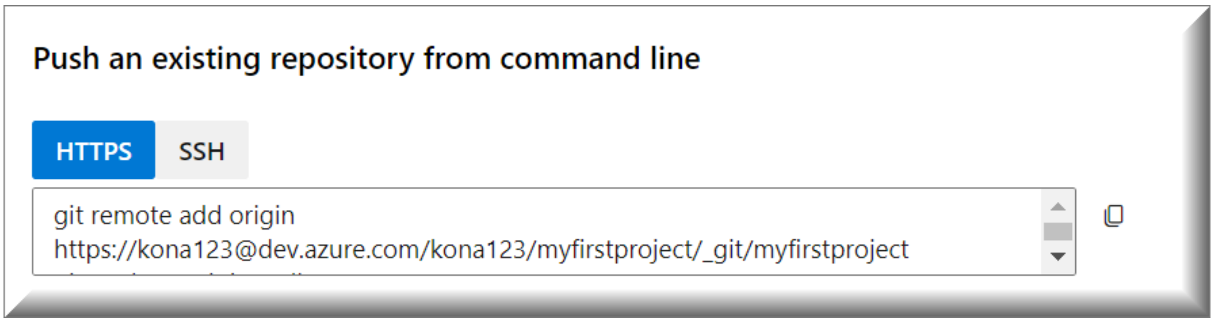
DigitalLync

✓ Add a README

Add a .gitignore: None

Your repository will be initialized with a `main` branch.

In order to push our application to the Azure DevOps remote repository, use the commands we discussed in the second option at the beginning of this chapter.


Push an existing repository from command line

```
git remote add origin
https://kona123@dev.azure.com/kona123/myfirstproject/_git/myfirstproject
```

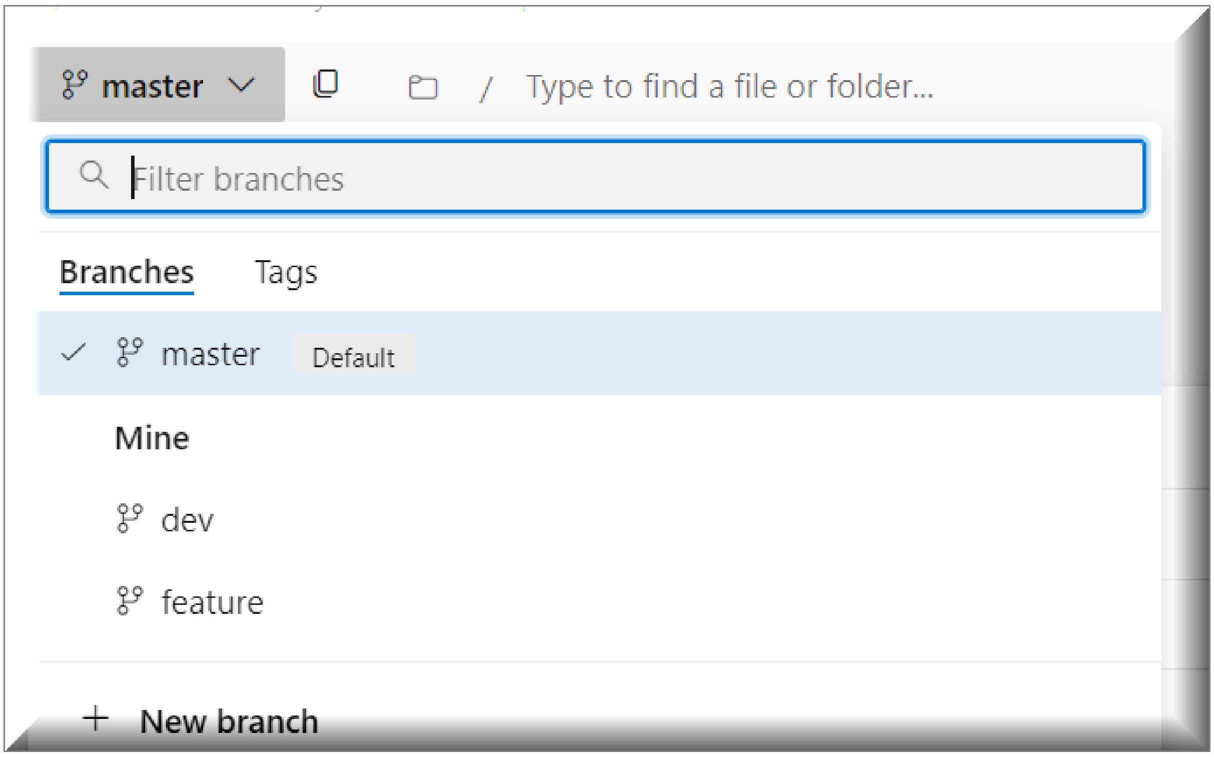
If you have not opened a Git Bash under the **my-quiz-ui** folder, then open it and paste these two commands you copied earlier one after the other. You have to probably authenticate yourself if you have not done so yet.

If everything goes well, you will see a different page in your Azure DevOps repos page, once you do a refresh. This is shown in Figure 76.

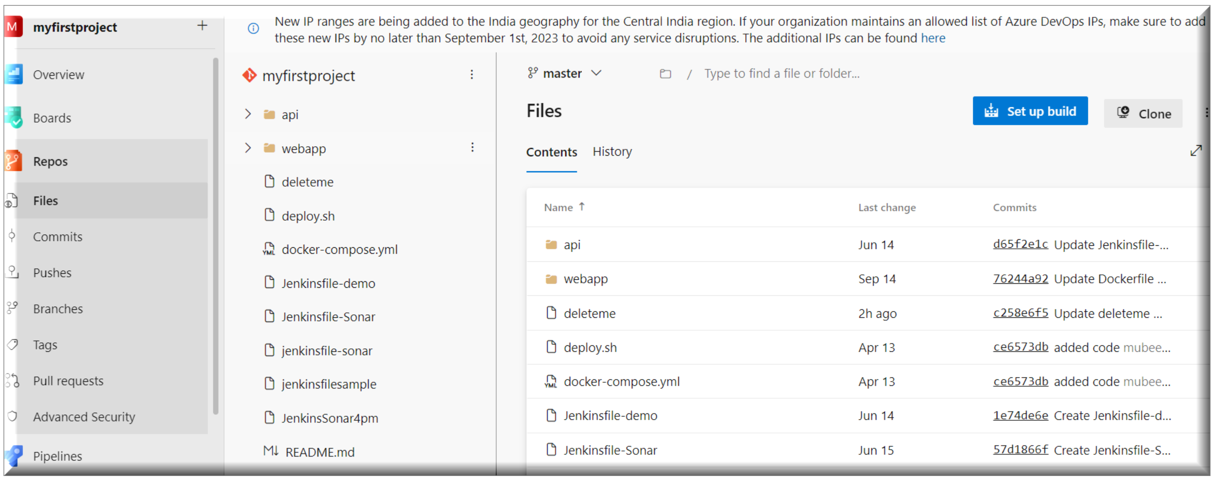
Remember that all your changes are pushed to the **master** branch of your repository. So, now we have successfully pushed our code to Azure DevOps Repos. Let us now look closer into each sub menu under **Repos** menu in Azure DevOps.

Files

Here, you can see the name of the project and all the files in a tree structure under that. You can also filter the results by the branch you need to see as shown in Figure 77. In addition to that, you can search for a specific file or a folder.



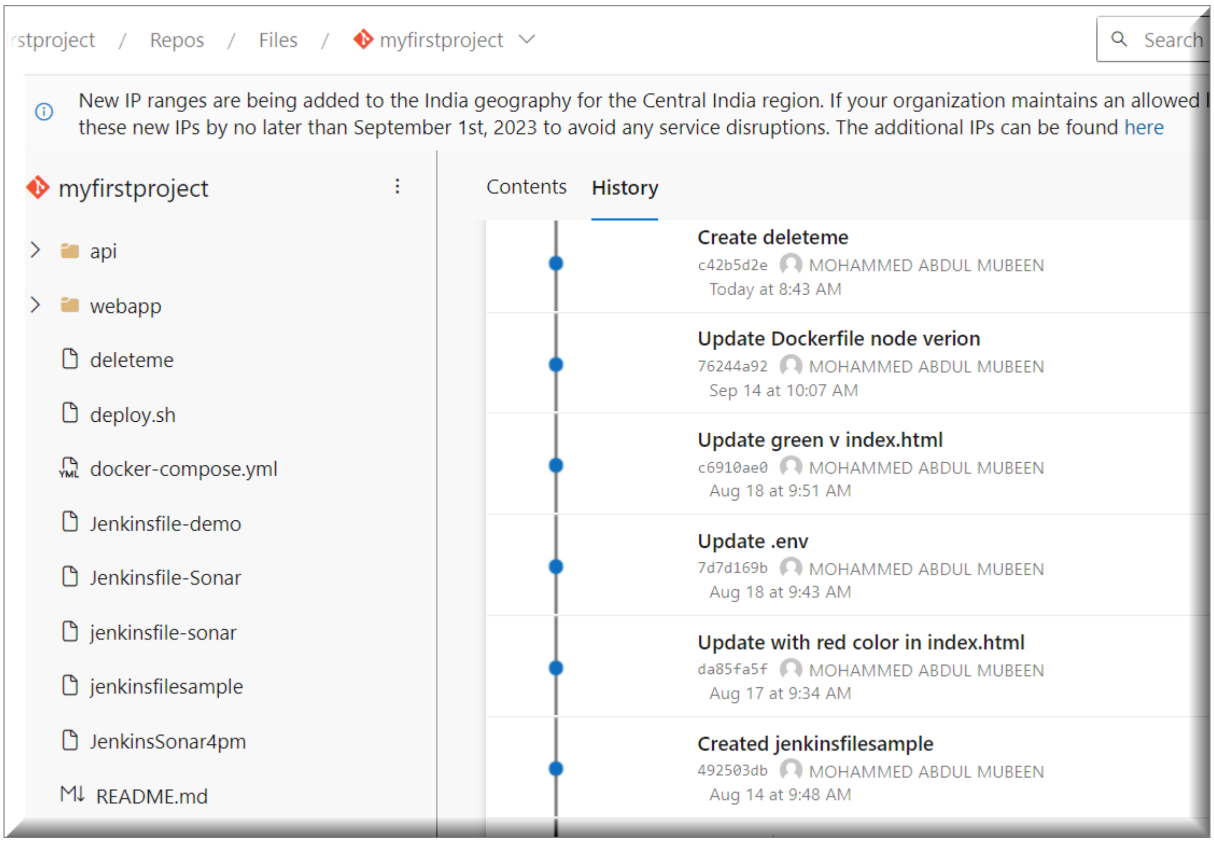
Contents tab is selected by default and you see the same file structure in the right-hand side as shown in Figure 78. You can also see the commits and when you did the last change.



New IP ranges are being added to the India geography for the Central India region. If your organization maintains an allowed list of Azure DevOps IPs, make sure to add these new IPs by no later than September 1st, 2023 to avoid any service disruptions. The additional IPs can be found here

Name ↑	Last change	Commits
api	Jun 14	d65f2e1c Update Jenkinsfile-...
webapp	Sep 14	76244a92 Update Dockerfile ...
deleteme	2h ago	c258e6f5 Update deleteme ...
deploy.sh	Apr 13	ce6573db added code mubee...
docker-compose.yml	Apr 13	ce6573db added code mubee...
Jenkinsfile-demo	Jun 14	1e74de6e Create Jenkinsfile-d...
Jenkinsfile-Sonar	Jun 15	57d1866f Create Jenkinsfile-S...

In the **History** tab, you can see all your commits to the repository. For example, you can see the last commit we did for the task 12.



rstproject / Repos / Files / myfirstproject

Search

New IP ranges are being added to the India geography for the Central India region. If your organization maintains an allowed l
these new IPs by no later than September 1st, 2023 to avoid any service disruptions. The additional IPs can be found here

myfirstproject

- api
- webapp
- deleteme
- deploy.sh
- docker-compose.yml
- Jenkinsfile-demo
- Jenkinsfile-Sonar
- jenkinsfile-sonar
- jenkinsfilesample
- JenkinsSonar4pm
- README.md

Contents History

Create deleteme
c42b5d2e MOHAMMED ABDUL MUBEEN
Today at 8:43 AM

Update Dockerfile node verion
76244a92 MOHAMMED ABDUL MUBEEN
Sep 14 at 10:07 AM

Update green v index.html
c6910ae0 MOHAMMED ABDUL MUBEEN
Aug 18 at 9:51 AM

Update .env
7d7d169b MOHAMMED ABDUL MUBEEN
Aug 18 at 9:43 AM

Update with red color in index.html
da85fa5f MOHAMMED ABDUL MUBEEN
Aug 17 at 9:34 AM

Created jenkinsfilesample
492503db MOHAMMED ABDUL MUBEEN
Aug 14 at 9:48 AM

Figure 79: Commits history

If you remember when we committed the change, we added **#12** at the beginning of the commit message. Now, if you navigate to Azure Boards and navigate to task number 12, you will find a link to the commit. This can be very useful to link your code changes to the tasks you are working on.

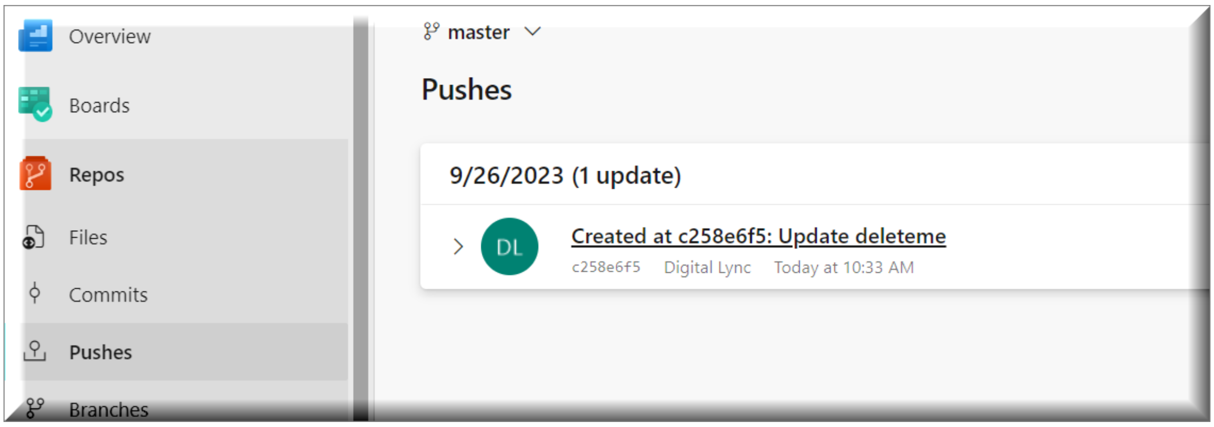
Commits

Figure 80: Link to the git commit related to the task

Under the commits section, you can see all the commits done to the whole repository. It shows a graphical view in addition to the commit messages as shown in Figure 81.

Pushes

Pushes section under Repos shows all the pushes you have done to the repository. If you expand a specific push, you can see all the commits related to that push.



Pushes

9/26/2023 (1 update)

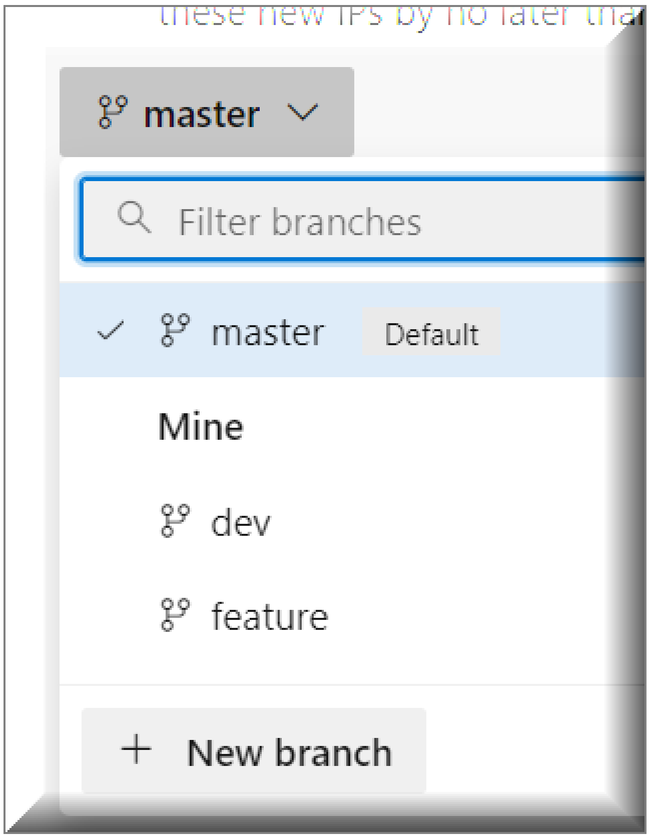
Created at c258e6f5: Update deleteme
c258e6f5 Digital Lync Today at 10:33 AM

Branches

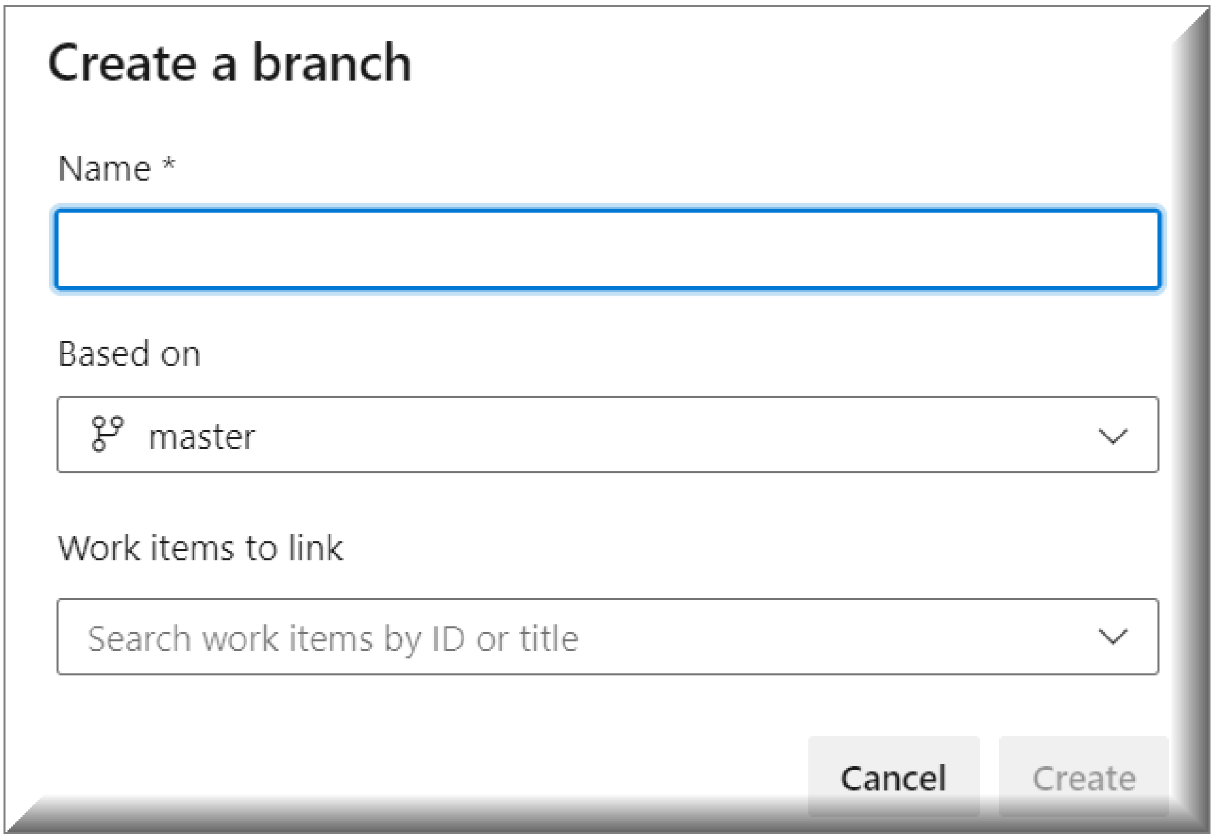
When we are working in a team, we have to collaborate with other team members and share the code with each other. That is why we are using Azure Repos through Git repositories. Branches in Azure Repos provide a smooth way to achieve this.

We have already worked with the **master** branch in our previous examples. However, master branch itself is not enough for a better collaboration.

Therefore, we need to create branches off the master branch to work on different work item tasks assigned to us through Azure Boards. To create a branch from the master, you can navigate to **Repos** --> **Branches** and click on the more icon on the right to get the context menu. Click on the link **New Branch**.



In the following modal dialog, give a **name** to your new branch, and select **master** as the “**Based on**” option. In addition to that, you can link a work item to this branch.



Create a branch

Name *

Based on

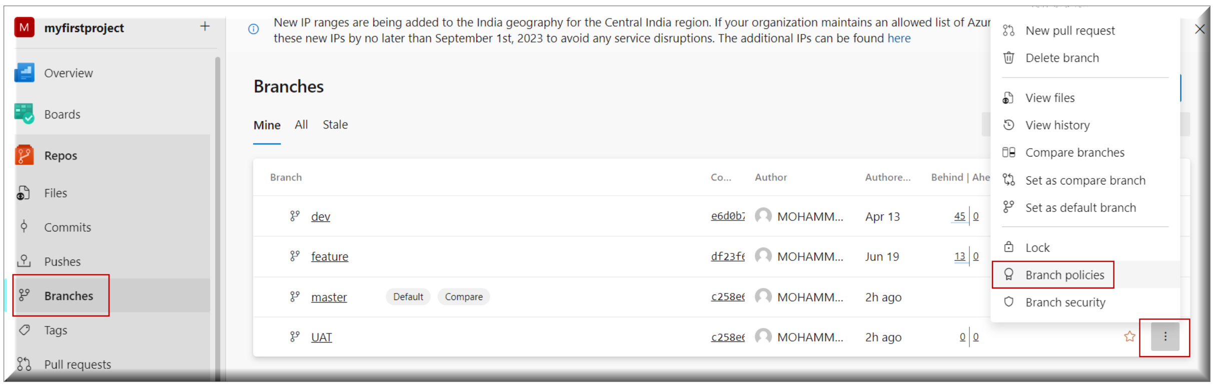
master

Work items to link

Search work items by ID or title

Figure 88: Select branch policies

1. In the master branch, click on the more icon on the right-hand corner and select **branch policies**.
2. Create at least one branch policy (For example, *Require a minimum number of reviewers*) in the following page so that you can prevent members directly pushing changes to the master branch.

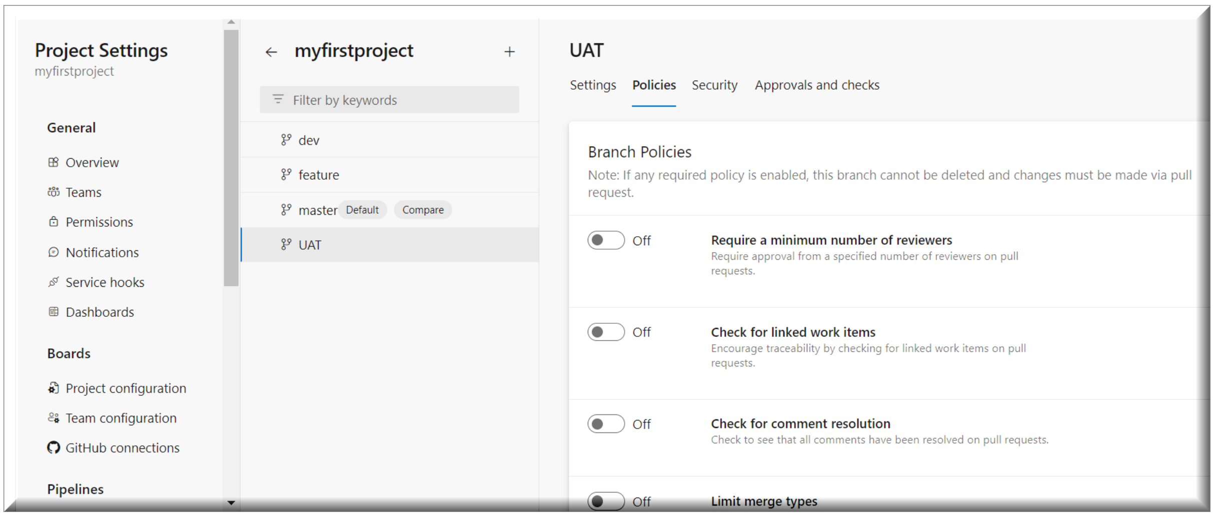


New IP ranges are being added to the India geography for the Central India region. If your organization maintains an allowed list of Azur
these new IPs by no later than September 1st, 2023 to avoid any service disruptions. The additional IPs can be found here

Branches

Mine All Stale

Branch	Co...	Author	Authore...	Behind | Ahe
dev	e6d0b7	MOHAMM...	Apr 13	45 | 0
feature	df23f6	MOHAMM...	Jun 19	13 | 0
master Default Compare	c258e6	MOHAMM...	2h ago	
UAT	c258e6	MOHAMM...	2h ago	0 | 0


Project Settings
myfirstproject

General
- Overview
- Teams
- Permissions
- Notifications
- Service hooks
- Dashboards

Boards
- Project configuration
- Team configuration
- GitHub connections

Pipelines

← myfirstproject +

Filter by keywords
- dev
- feature
- master Default Compare
- UAT

UAT

Settings Policies Security Approvals and checks

Branch Policies

Note: If any required policy is enabled, this branch cannot be deleted and changes must be made via pull request.

Off	**Require a minimum number of reviewers** Require approval from a specified number of reviewers on pull requests.
Off	**Check for linked work items** Encourage traceability by checking for linked work items on pull requests.
Off	**Check for comment resolution** Check to see that all comments have been resolved on pull requests.
Off	**Limit merge types**

Git flow

Git flow uses a set of long running branches to represent different stages of the development cycle. The **master** branch always contains the stable code that is deployed (or will be deployed) to production. In addition to the master branch, there is a parallel branch called **develop** that is used by developers to work from. Developers can create their **feature** branches from the **develop** branch. Once the develop branch comes to a stable point, you can merge it to the master branch for the next release. This can be done through a release branch and the bug fixing on the release branch has to be continuously merged back into the develop branch. Once you are satisfied with the release branch, you can merge it to the master branch for the next release. Hotfixes to the current version can be done on a hotfix branch from master and merged back to both master and develop. This is shown in Figure 92.

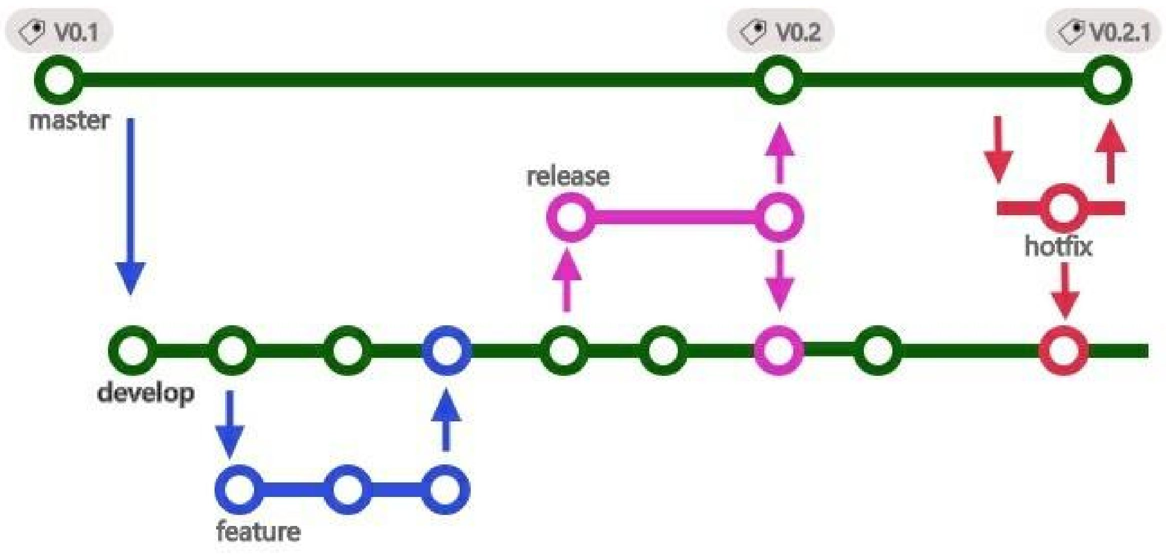


Figure 92: git flow branching strategy

As explained in the previous section, you can use pull requests to merge changes to **develop**, **release** and **master** branches.

Tags

Git tags are used to mark a specific commit as an important point in the history. Usually, this is used to mark a release point, at which commit a certain version of the code was released. However, you do not need to create tags if you are using release branches to manage your releases.

The easiest way to create a tag is by navigating to the **Commits** sub menu. Here, you go to a specific comment and click on the more icon on the right- hand corner.

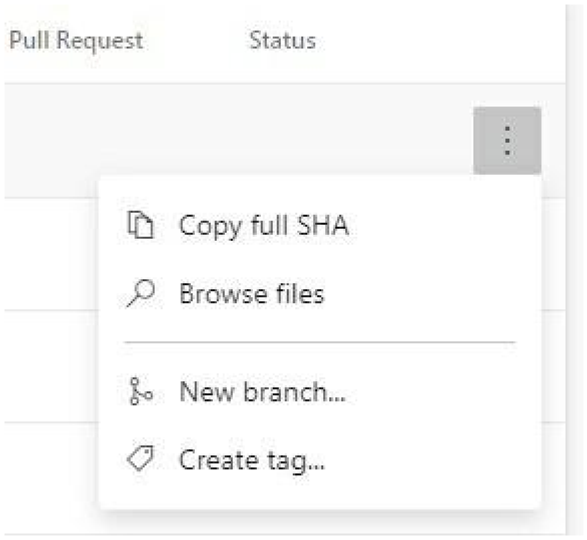
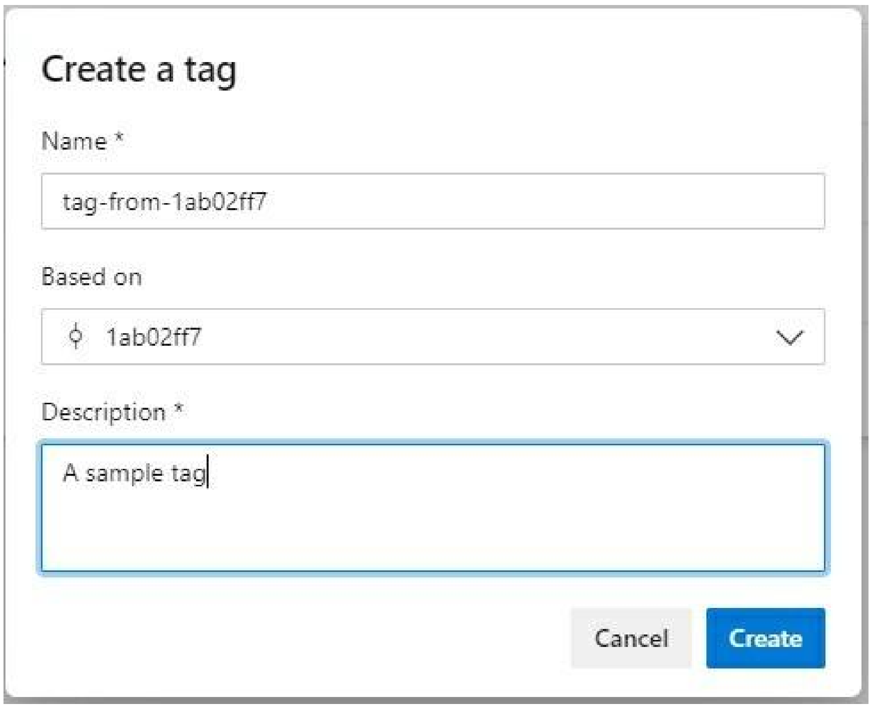


Figure 93: Create tag from the context menu

Now, click on the **Create tag** link. In the modal dialog, you can write a **name**

and a **description** for your tag and click on the **Create** button.



Create a tag

Name *

tag-from-1ab02ff7

Based on

1ab02ff7

Description *

A sample tag

Figure 94: Modal dialog for creating a tag

Now, you will see a new tag is created with the whole source code of the project.

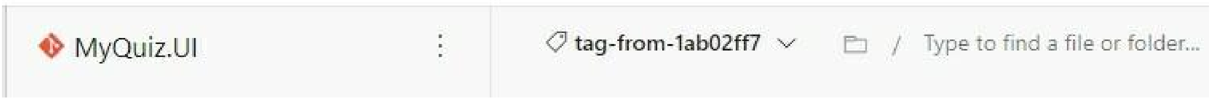
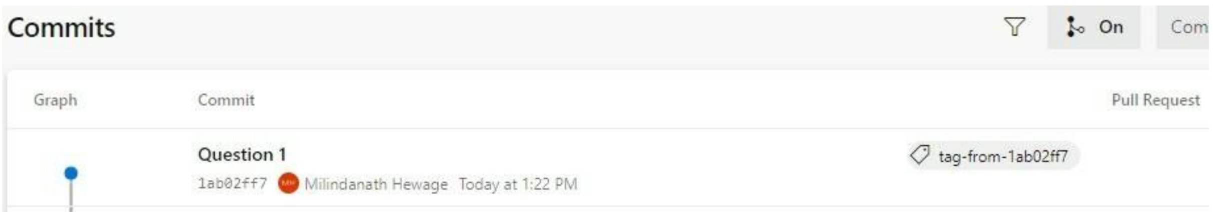


Figure 95: Tag created

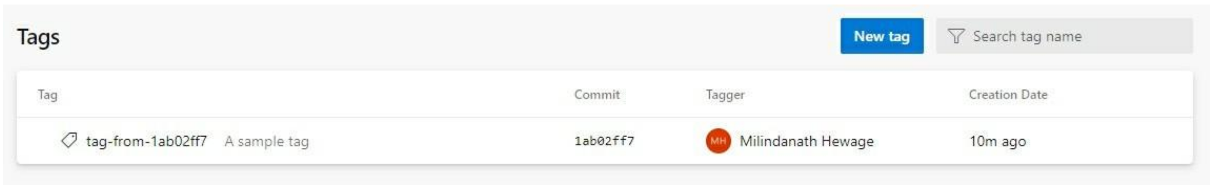
Moreover, you will see a label attached to the commit you created the tag for.



Graph	Commit		Pull Request
	Question 1 1ab02ff7 Milindanath Hewage Today at 1:22 PM	tag-from-1ab02ff7	

Figure 96: commit related to the tag

You can see all the tags created for your project by navigating to the **Tags** sub menu.



Tags

New tag
Search tag name

Tag	Commit	Tagger	Creation Date
tag-from-1ab02ff7 A sample tag	1ab02ff7	Milindanath Hewage	10m ago

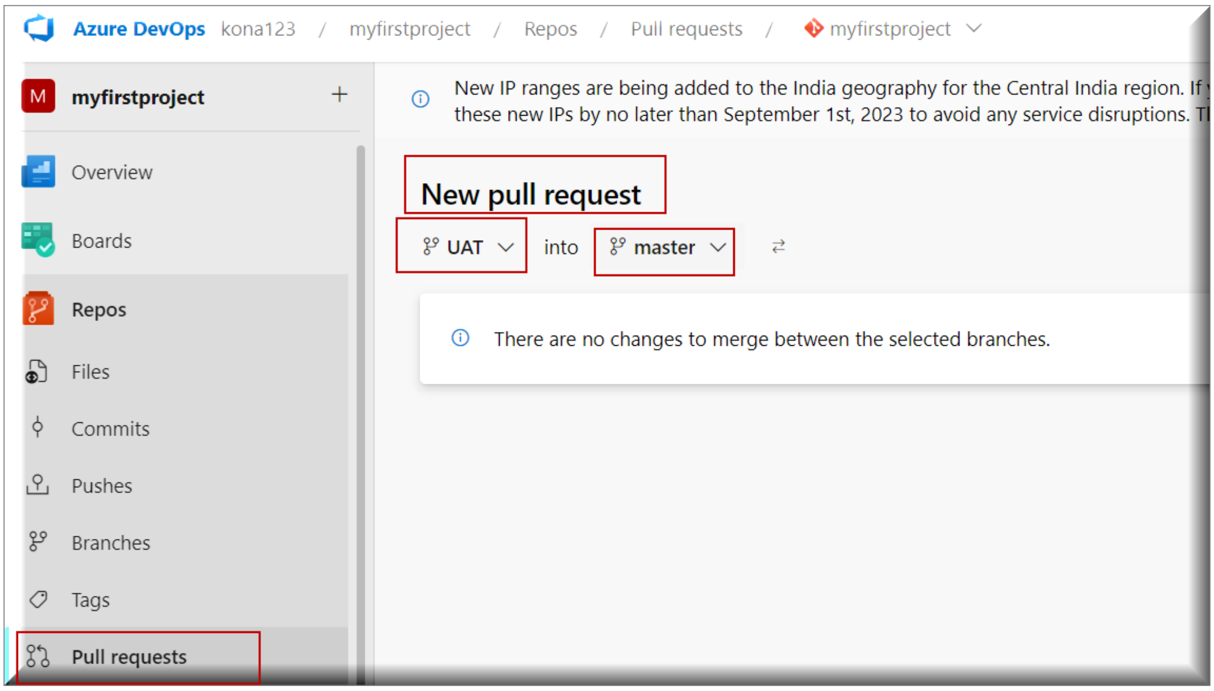
Figure 97: Tags list page

In order to edit the source code in a tagged version, you have to create a branch from the tag and then do a pull request to bring the changes back to the master.

Pull requests

Pull requests is a very good way of maintaining a high quality in your code. This allows you to discuss, review and quality assure your code changes before they get merged into your base branch. Pull requests functionality can be enabled to branches by setting branch policies. Let us see how we can use pull requests to our master branch. Open branch policies page for the master

branch.



Azure DevOps kona123 / myfirstproject / Repos / Pull requests / myfirstproject
myfirstproject
Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
Pull requests

New IP ranges are being added to the India geography for the Central India region. If
these new IPs by no later than September 1st, 2023 to avoid any service disruptions. T

New pull request

UAT into master

There are no changes to merge between the selected branches.

Figure 98: Set branch policies for the master branch

Here, you can add some restrictions and checks before a certain pull request can be accepted. For example,

1. Specify the number of reviewers who will review the code. You can also add an automatic code reviewer.
2. The application has to be built successfully in order to complete

the pull request.

1. At least one work item has to be linked to the pull request.

Suppose we have set up our branch policy as shown in Figure 98. Now, let us try to work on a task and do some code changes.

1. Create a branch from your master branch with the name

feature/13

1. Add some changes to your HelloWorld.vue file.
2. Commit and push those changes to the Azure Repos branch
3. Now, you can go to the Repos and to your branch **feature/13**. Then you will see the following message.

Figure 99: Create a pull request message

1. Click on the **Create a pull request** button to create the pull request
2. In the following page you can see that the pull request is from **feature/13** into **master** branch. You can also provide a suitable title and description for the pull request. Then add who is going to review your code. If you have not linked a task in your commit message, then do it here. After you fill all the required information, click on the Create button to start the pull request.
3. If you do not specify a reviewer and a work item, then you will see that you have violated the branch policies as below.

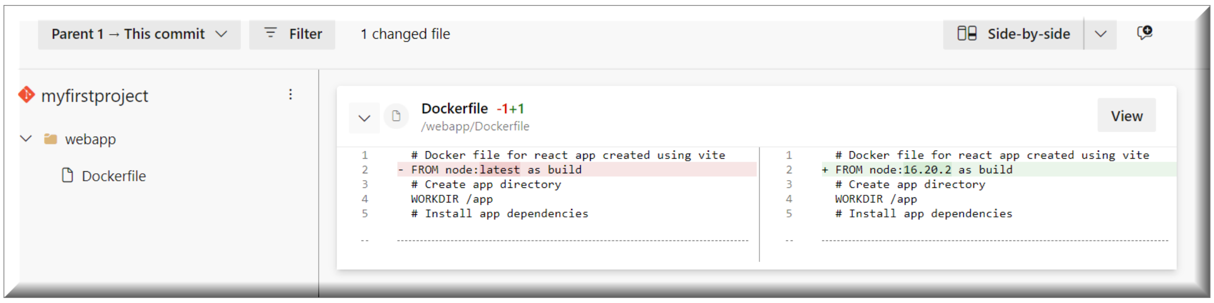
Figure 101: Branch policies not fulfilled yet

1. Even here you can add that information by clicking on the **+** sign.
2. In addition to that, the reviewers can start a conversation with the developer by adding comments in the comments section.

Figure 102: Comments section in the pull request

1. Once you click on the **Files** tab, you can see the

changes related to the pull request. Here, the reviewer can add comments at specific lines in the file.



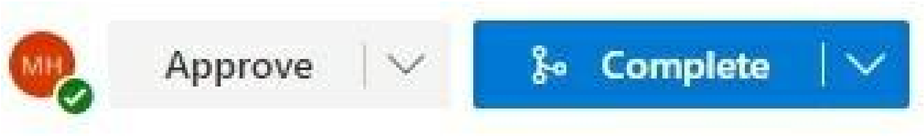
```
  # Docker file for react app created using vite
- FROM node:latest as build
  # Create app directory
  WORKDIR /app
  # Install app dependencies
```

```
  # Docker file for react app created using vite
+ FROM node:16.20.2 as build
  # Create app directory
  WORKDIR /app
  # Install app dependencies
```

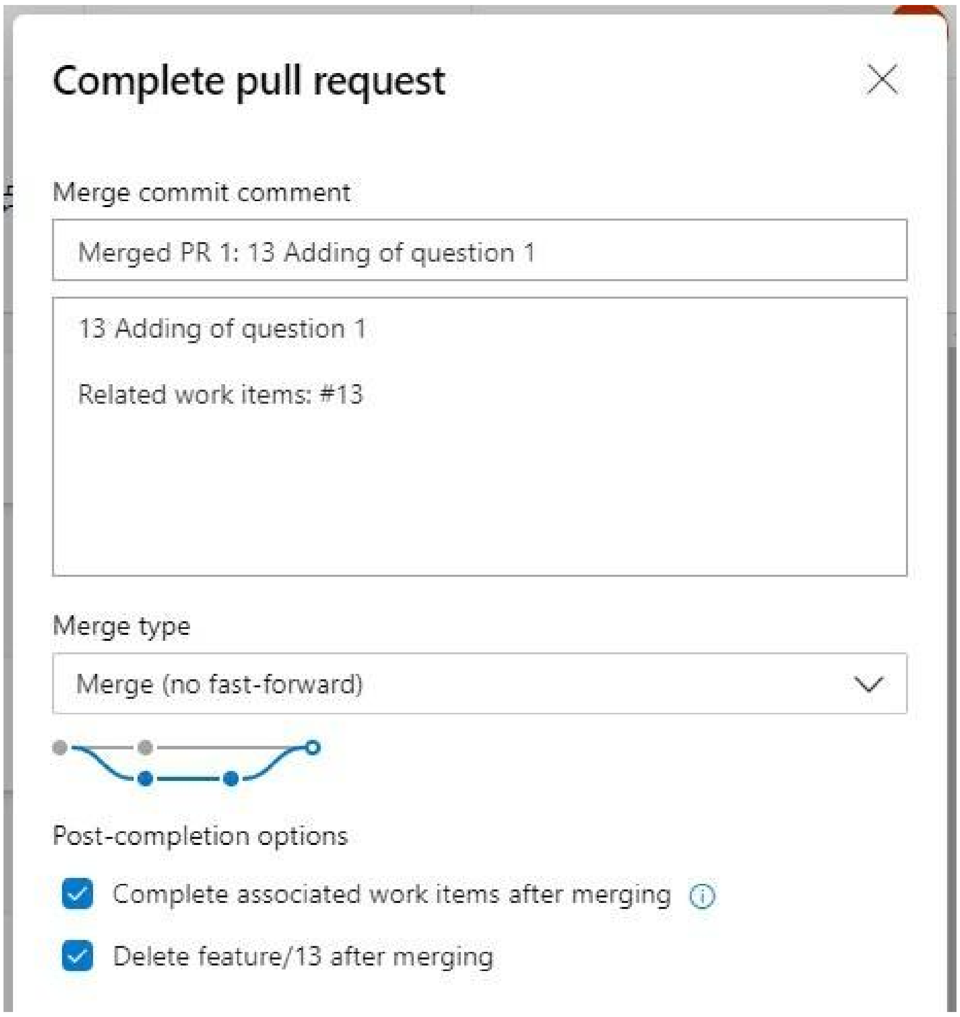
Figure 103: Add comment to a file

1. Then you can start the conversation with the developer mentioning your concerns about the code. In this way, the team members can communicate back and forth to produce a high- quality code.
2. Then, the developer can fix issues mentioned in the comment on the same branch and push the changes back to the server. Then, the new commit will appear under the pull request. You can also **Resolve** the comment added by the reviewer.
3. Once both the parties have agreed on the changes, the reviewer can approve the change by clicking on the **Approve** button.

4. Now, you will see that all the required branch policies are fulfilled. Click on the **Complete** button to finally start **merging** the code changes to master branch.



1. You can add a comment if you wish and set which merge type you want to merge the changes. Also, you can delete the feature branch after merge and set the work item to **Done** state.


Complete pull request

Merge commit comment

Merged PR 1: 13 Adding of question 1

13 Adding of question 1

Related work items: #13

Merge type

Merge (no fast-forward)

Post-completion options

✓ Complete associated work items after merging

✓ Delete feature/13 after merging

Figure 106: Complete pull request dialog

1. You will see a message saying that you have completed the pull request.
2. You can verify that your changes are committed to the master branch, by inspecting the commits to the master.

Summary

In this chapter, we learned about creating an application and moving its source code to a git repository located in Azure Repos. We also looked into different methods of creating a git repository in Azure Repos. Moreover, we learned about different parts of Azure Repos, such as Files, Commits, Pushes, Branches, Tags and pull requests.