Introduction to Automated Builds & Pipeline Commits

Updated 7/15/2022

Overview

This lab will help the learner understand how to configure a Jenkins pipeline, connect it to a source code repository (GitHub), and trigger an automated build flow when changes are made to the web application or the pipeline itself.

We will be using a web app called “Farm to Front Door” throughout this course. The idea is that “Development” is working on this application and needs our help to create a continuous integration pipeline to build it. We will do more than just build it as we complete this course but for starters, this is just to get the application pipeline stood-up so we can build out a complete pipeline by the end of the course.

Estimated time: 40 min

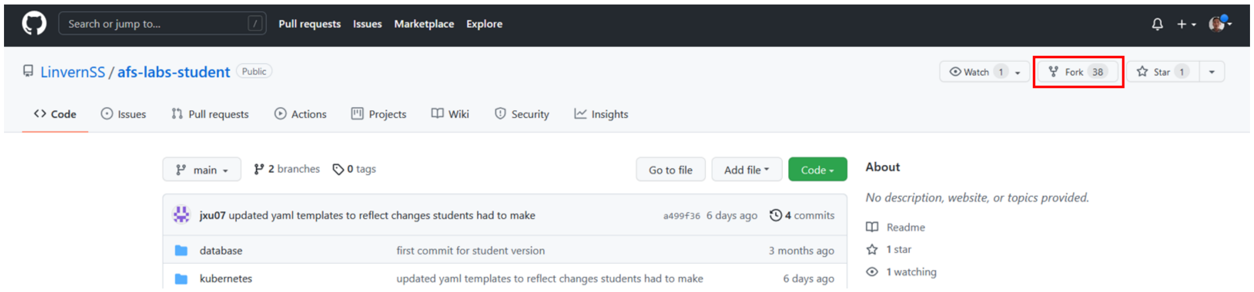
Learning Objectives

1. Create and configure a Jenkins pipeline
2. Understand and utilize GitHub webhooks
3. Test the pipeline is working

Instructions

Set up the Project Repository

1. Log in to your https://github.com/ account. If you do not have an account, please create one.
2. Navigate to the application repository, Farm to Front Door: https://github.com/LinvernSS/afs-labs-student
3. Fork this repository by clicking on the “Fork” button, which can be found in the upper right corner of the GitHub screen



jxu07 updated yaml templates to reflect changes students had to make a499f36 6 days ago 4 commits

database	first commit for student version	3 months ago
kubernetes	updated yaml templates to reflect changes students had to make	6 days ago

4. Login to your EC2 and enter your lab environment container via VSCode

> **Note:** VSCode will remember your most recent window opened. If you closed VSCode while working inside your container, VSCode will automatically attempt to reconnect to your container when you reopen VSCode (you'll be asked to enter your password again).
>
> Otherwise, you can reconnect to your container with VSCode under "File" → "Open Recent" → "root/dso-bootcamp-home [Container]".
>
> If none of the above works, then follow the steps below.

a. Open a new VSCode window

b. Under "Remote Explorer", select "SSH Targets" from the drop-down menu

c. Click on your EC2's IP address to SSH into

d. Under "Remote Explorer", select "Containers" from the drop-down menu

e. Click on "Refresh" to see a list of running containers

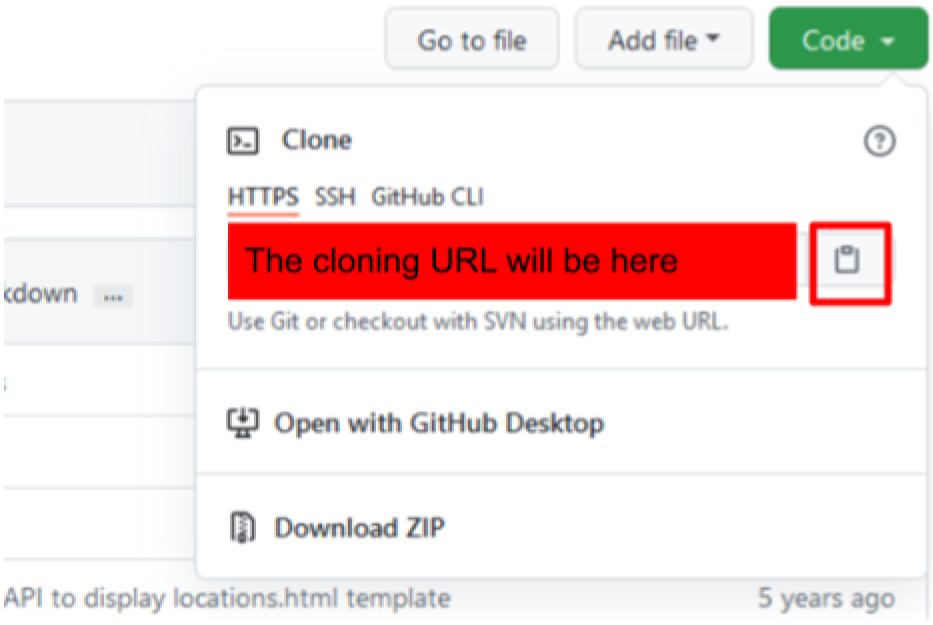
f. Right-click on "lab-env:latest" and "Attach to Container" to enter your lab environment

5. Clone your GitHub repository (that you just forked) into your container by completing the following steps:

a. First, click the green “Code” button on the GitHub repository

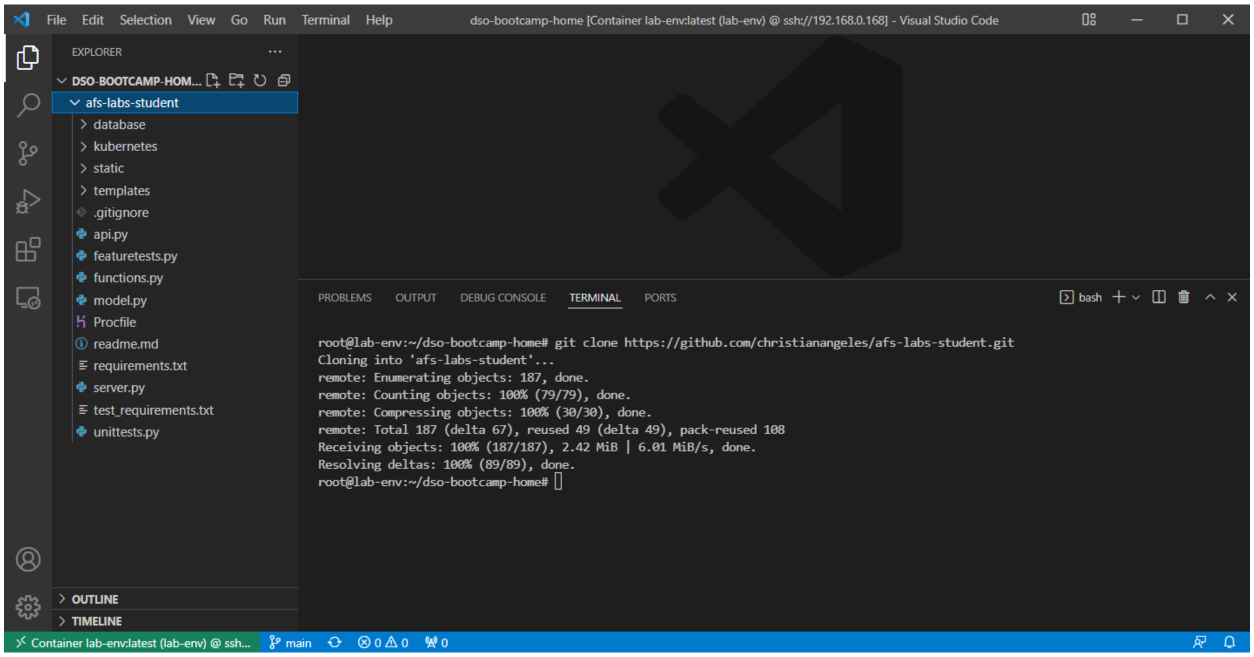
b. Copy the repository URL shown by clicking on the button to the right of it

> **Tip:** Make sure you're cloning with the *HTTPS*
>
> The URL should be something like `https://github.com/<YOUR GITHUB USERNAME>/afs-labs-student.git`


Go to file
Add file
Code
Clone
HTTPS
SSH
GitHub CLI
The cloning URL will be here
Use Git or checkout with SVN using the web URL.
Open with GitHub Desktop
Download ZIP
API to display locations.html template
5 years ago

c. Run the following command in your VSCode terminal to clone your repository to your container:

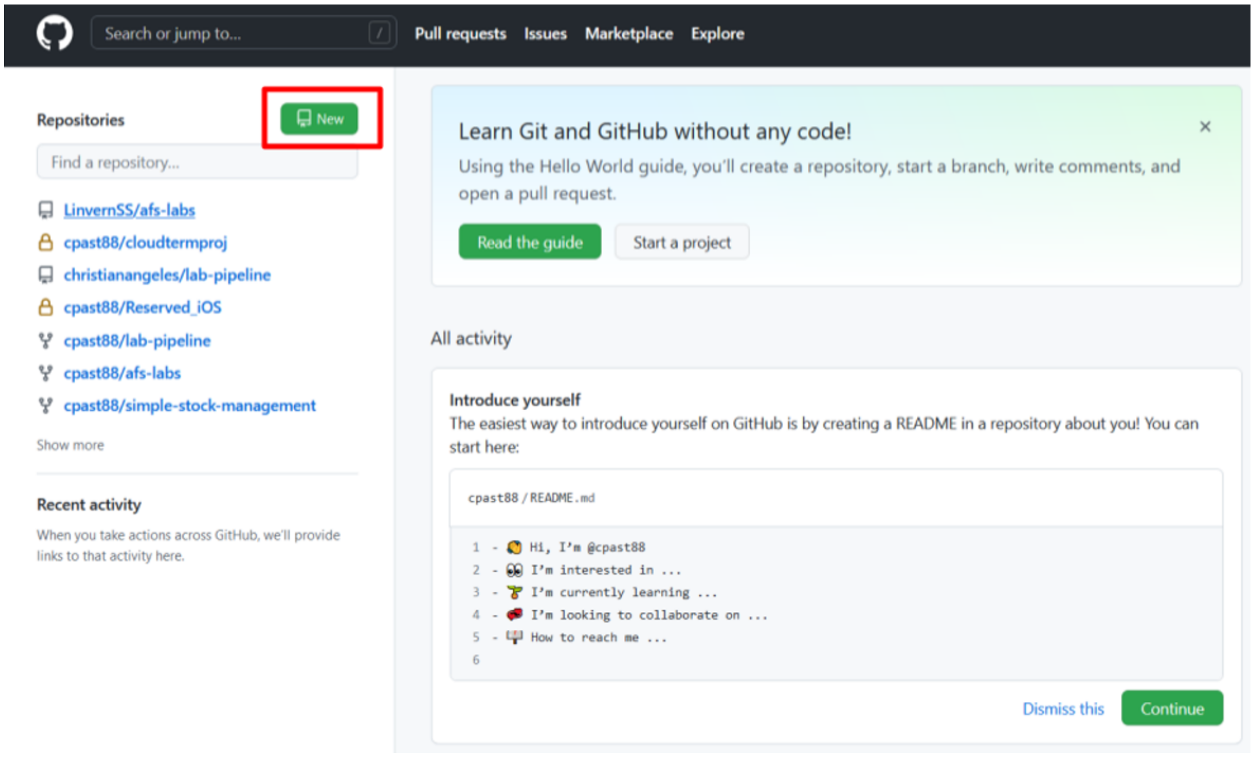
```
git clone <YOUR REPOSITORY URL>
```


File Edit Selection View Go Run Terminal Help
dso-bootcamp-home [Container lab-env:latest (lab-env) @ ssh://192.168.0.168] - Visual Studio Code
EXPLORER
DSO-BOOTCAMP-HOM...
afs-labs-student
database
kubernetes
static
templates
.gitignore
api.py
featuretests.py
functions.py
model.py
Procfile
readme.md
requirements.txt
server.py
test_requirements.txt
unittests.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
Container lab-env:latest (lab-env) @ ssh...
main

```
root@lab-env:~/dso-bootcamp-home# git clone https://github.com/christianangeles/afs-labs-student.git
Cloning into 'afs-labs-student'...
remote: Enumerating objects: 187, done.
remote: Counting objects: 100% (79/79), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 187 (delta 67), reused 49 (delta 49), pack-reused 108
Receiving objects: 100% (187/187), 2.42 MiB | 6.01 MiB/s, done.
Resolving deltas: 100% (89/89), done.
root@lab-env:~/dso-bootcamp-home# 
```

Create a Pipeline Repository

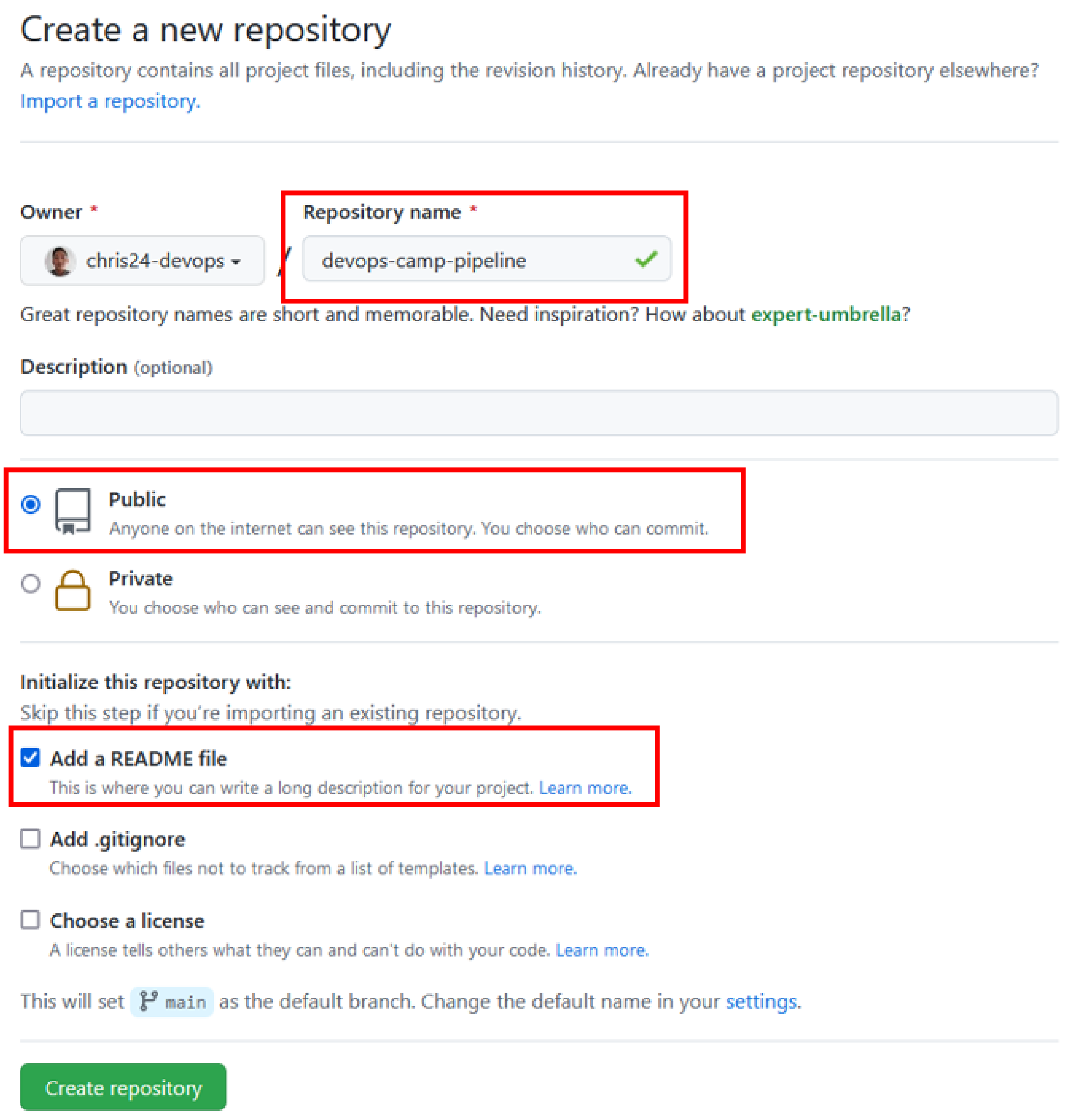
1. On https://github.com/, create a new repository by clicking on the green “New” button in the top left corner


Search or jump to...
Pull requests
Issues
Marketplace
Explore
Repositories
New
Find a repository...
LinvernSS/afs-labs
cpast88/cloudtermproj
christianangeles/lab-pipeline
cpast88/Reserved_iOS
cpast88/lab-pipeline
cpast88/afs-labs
cpast88/simple-stock-management
Show more
Recent activity
When you take actions across GitHub, we'll provide links to that activity here.
Learn Git and GitHub without any code!
Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.
Read the guide
Start a project
All activity
Introduce yourself
The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:
cpast88 / README.md
Dismiss this
Continue

```
- 👋 Hi, I’m @cpast88
- 👀 I’m interested in ...
- 🌱 I’m currently learning ...
- 💞️ I’m looking to collaborate on ...
- 📫 How to reach me ...
```

2. Name the new repository “devops-camp-pipeline”

   a. Make sure "Public" and "Add a README file" are selected


Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * chris24-devops / Repository name * devops-camp-pipeline

Great repository names are short and memorable. Need inspiration? How about expert-umbrella?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

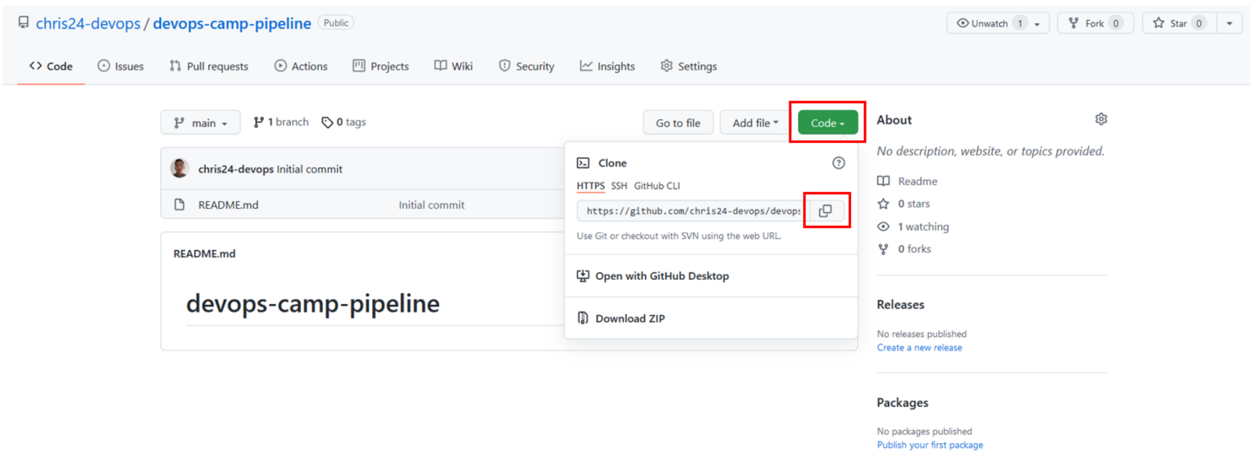
[✓] Add a README file
This is where you can write a long description for your project. Learn more.

[ ] Add .gitignore
Choose which files not to track from a list of templates. Learn more.

[ ] Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

3. Click “Create repository” to complete the process
4. Once you’ve created the repository, you should be shown a screen that looks like the one below; click on the highlighted buttons to copy the repository URL



5. In VSCode, clone your new repository into your container by running the following command in the terminal:

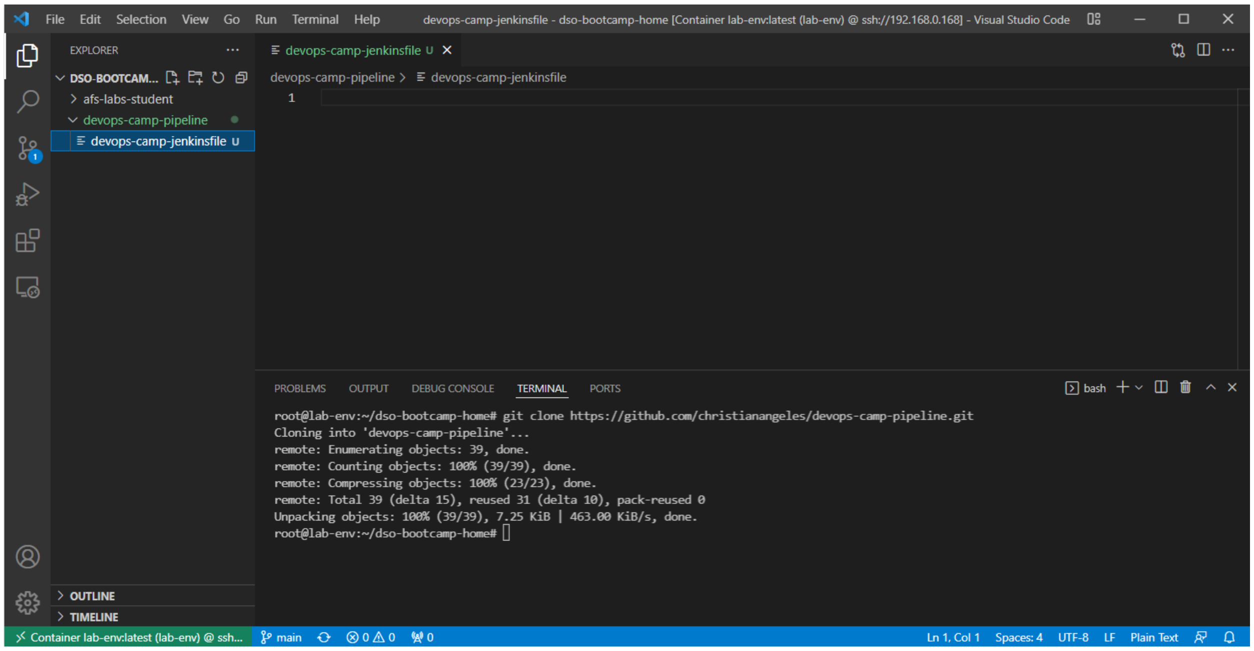
```
git clone <URL>
```

> **Note:** If you get a warning about cloning an empty repository, you can ignore it.

6. Create a Jenkinsfile for your pipeline

   a. Under "Explorer", right-click on the `devops-camp-pipeline` folder and select "New File"

   b. Name the new file `devops-camp-jenkinsfile`


File Edit Selection View Go Run Terminal Help
devops-camp-jenkinsfile - dso-bootcamp-home [Container lab-env:latest (lab-env) @ ssh://192.168.0.168] - Visual Studio Code
EXPLORER
DSO-BOOTCAM...
afs-labs-student
devops-camp-pipeline
devops-camp-jenkinsfile U
devops-camp-pipeline > devops-camp-jenkinsfile
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash
OUTLINE
TIMELINE
Container lab-env:latest (lab-env) @ ssh...
main
Ln 1, Col 1
Spaces: 4
UTF-8
LF
Plain Text

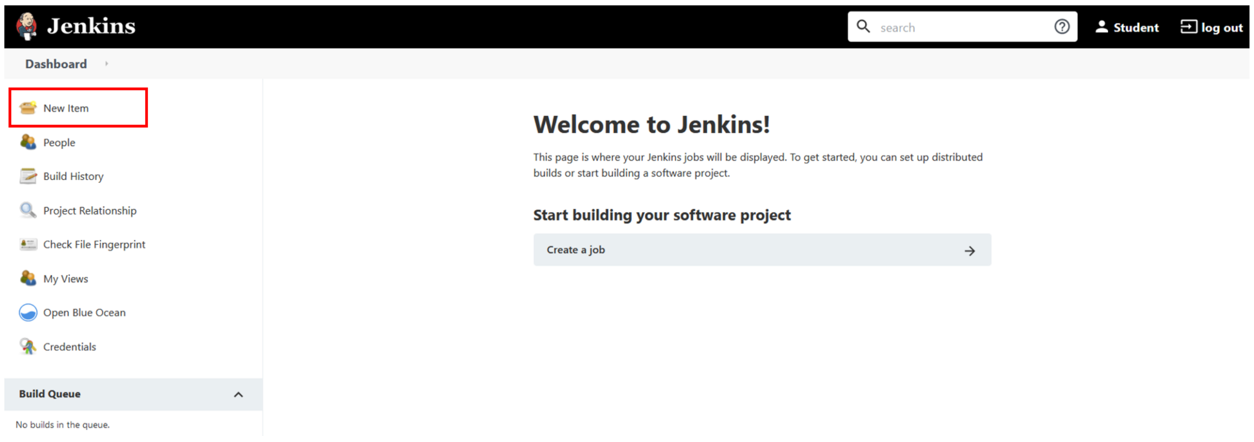
```
root@lab-env:~/dso-bootcamp-home# git clone https://github.com/christianangeles/devops-camp-pipeline.git
Cloning into 'devops-camp-pipeline'...
remote: Enumerating objects: 39, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 39 (delta 15), reused 31 (delta 10), pack-reused 0
Unpacking objects: 100% (39/39), 7.25 KiB | 463.00 KiB/s, done.
root@lab-env:~/dso-bootcamp-home# 
```

Create and Configure a Jenkins Pipeline

1. Navigate to your Jenkins environment: https://jenkins.dev.afsmtddso.com/

   a. Login with the credentials that were already provided

2. Create a new pipeline by clicking on "New Item"



Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

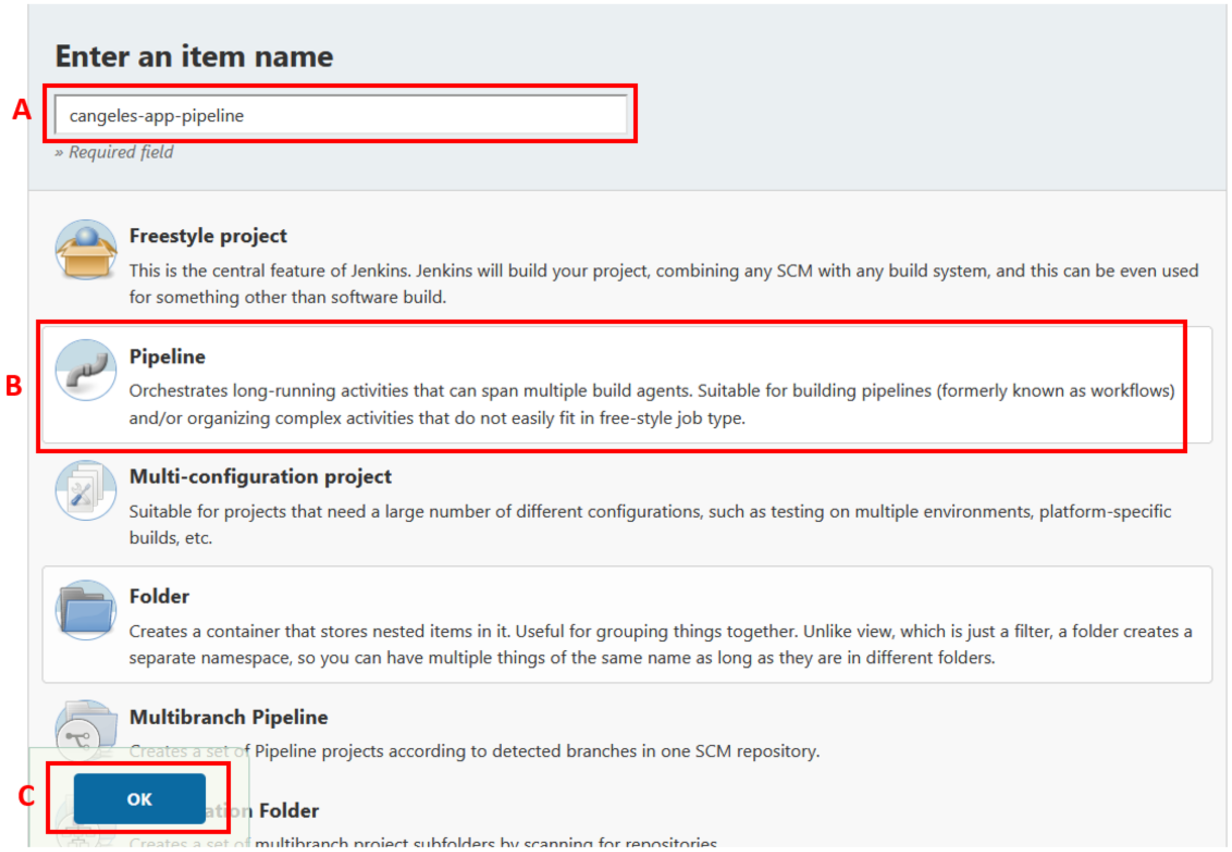
No builds in the queue.

3. You should now see a screen with options for your new item

   a. Name your new item `<YOUR FIRST INITIAL + LAST NAME>-app-pipeline`

   b. Select "Pipeline" as the type of project

   c. Press "OK" to proceed



Enter an item name

A
cangeles-app-pipeline
» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

B
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

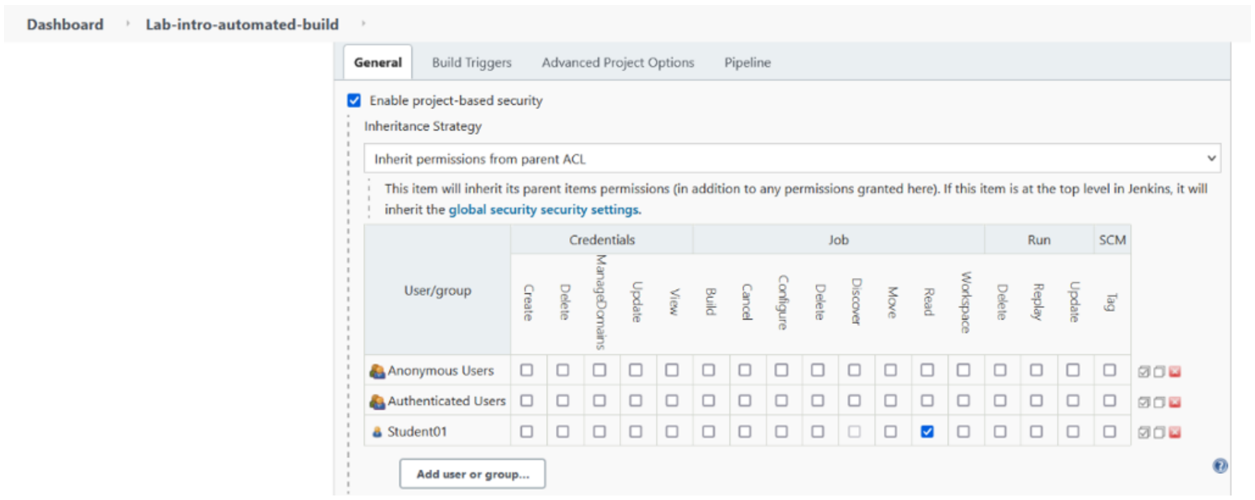
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

C
OK
Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories

4. You should now see a screen with configuration options to enable

   a. “Project-based security” will be enabled by default, leave the settings as is



Dashboard › Lab-intro-automated-build ›

General | Build Triggers | Advanced Project Options | Pipeline

☑ Enable project-based security

Inheritance Strategy

Inherit permissions from parent ACL

This item will inherit its parent items permissions (in addition to any permissions granted here). If this item is at the top level in Jenkins, it will inherit the global security security settings.

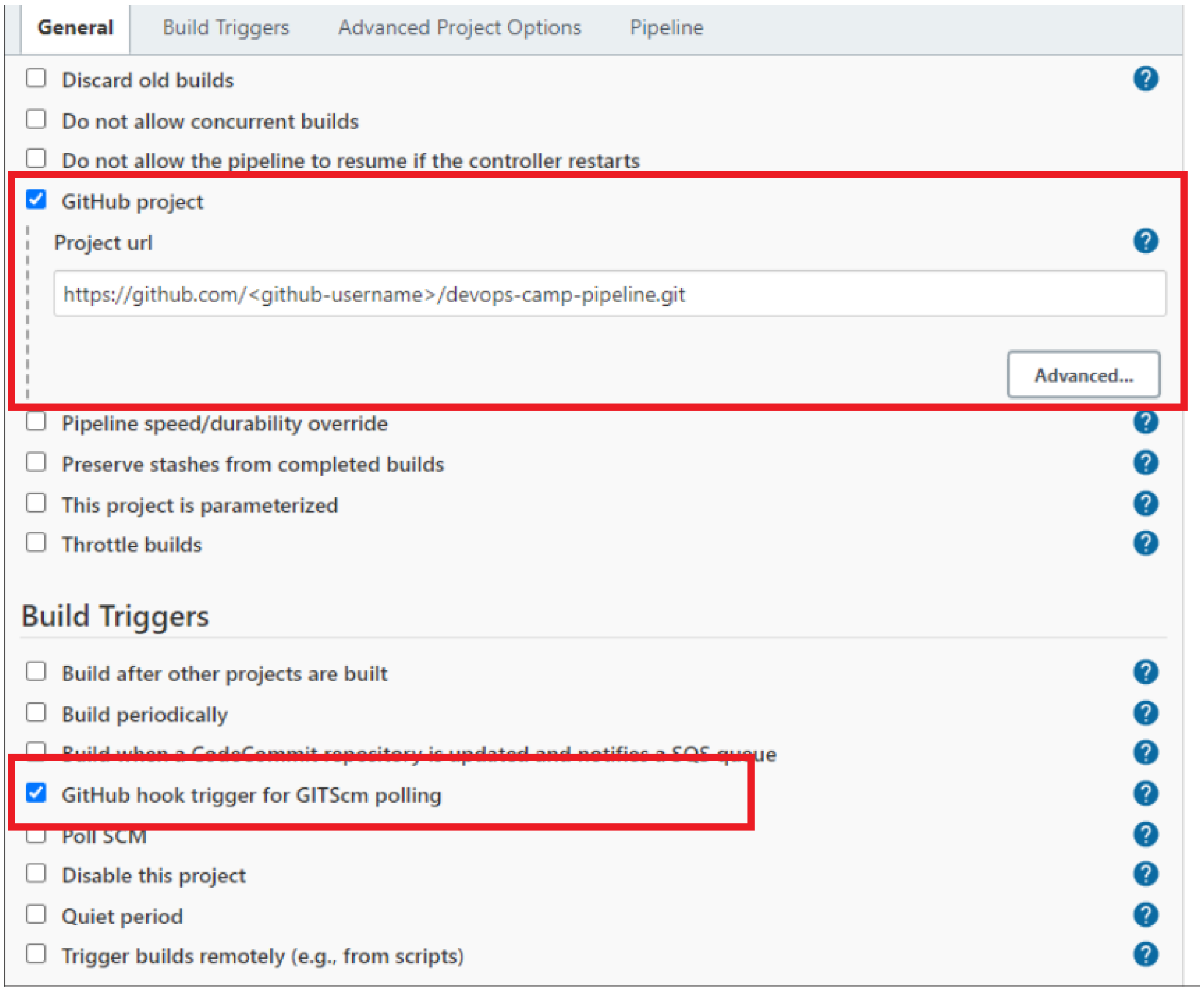
User/group	Credentials					Job								Run			SCM
	Create	Delete	ManageDomains	Update	View	Build	Cancel	Configure	Delete	Discover	Move	Read	Workspace	Delete	Replay	Update	Tag
Anonymous Users	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Authenticated Users	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Student01	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐

Add user or group...

   b. A little bit underneath that, make sure "GitHub Project" is checked

   c. In the input box below "Github Project," paste in the GitHub URL of your `devops-camp-pipeline` repository

d. Scroll down to the subsection "Build Triggers" and check "GitHub hook trigger for GITScm polling"



- [ ] Discard old builds
- [ ] Do not allow concurrent builds
- [ ] Do not allow the pipeline to resume if the controller restarts
- [x] GitHub project

Project url

https://github.com/<github-username>/devops-camp-pipeline.git

- [ ] Pipeline speed/durability override
- [ ] Preserve stashes from completed builds
- [ ] This project is parameterized
- [ ] Throttle builds

Build Triggers

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] Build when a CodeCommit repository is updated and notifies a SQS queue
- [x] GitHub hook trigger for GITScm polling
- [ ] Poll SCM
- [ ] Disable this project
- [ ] Quiet period
- [ ] Trigger builds remotely (e.g., from scripts)

5. Select the "Pipeline" tab and configure for your SCM (Source Code Management)

a. Under "Definition," click "Pipeline script" to open the dropdown menu

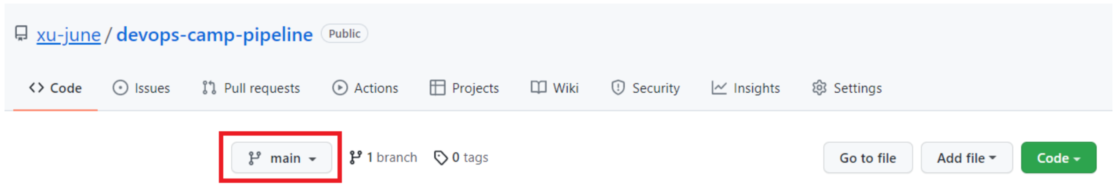
i. Select “Pipeline script from SCM”

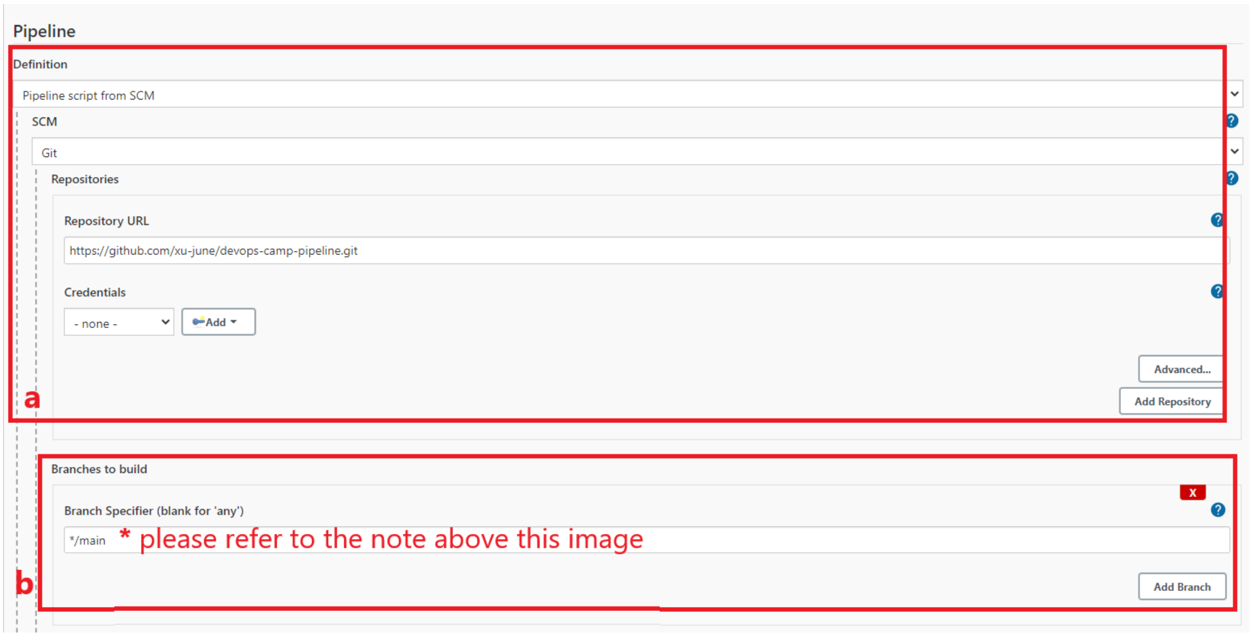
ii. In the SCM dropdown menu, select "Git"

iii. In the "Repositories" section, add your `devops-camp-pipeline` repository URL

b. On "Branches to build," make sure you put your main branch

> **Tip:** This branch could be called either `main` or `master`, check your repository to see what yours is called by looking here:




Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/xu-june/devops-camp-pipeline.git

Credentials: - none -

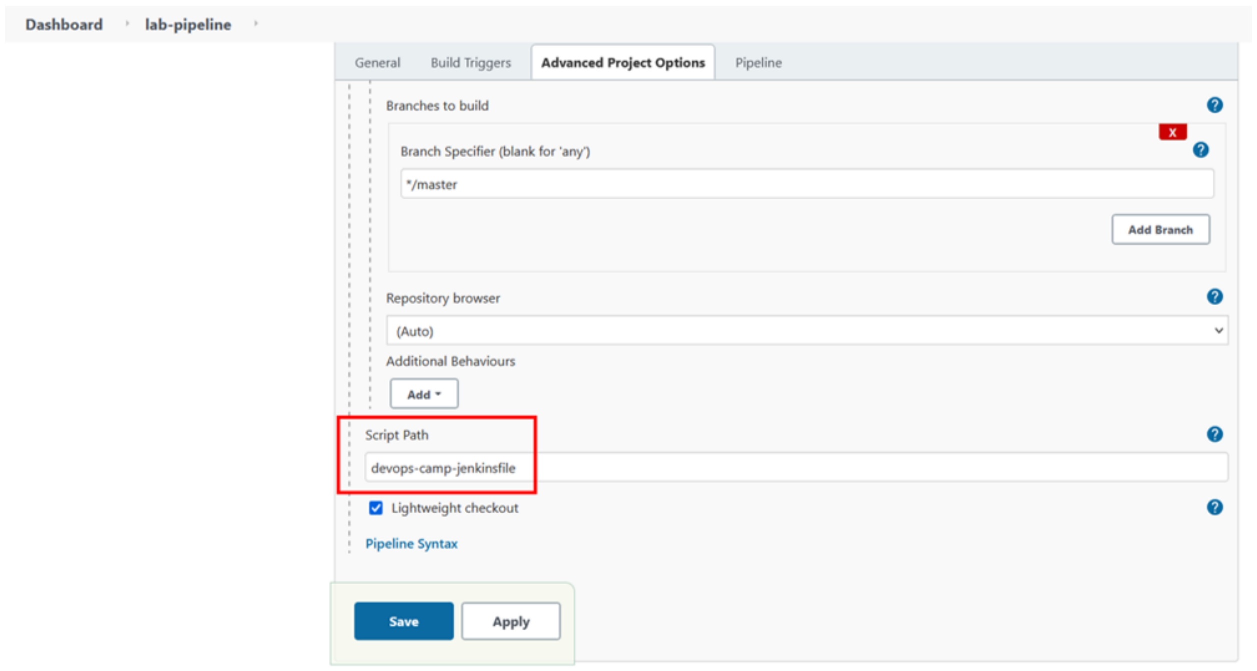
Branches to build

Branch Specifier (blank for 'any'): */main

6. Scroll to the bottom of the options and look for "Script Path"

   a. In the Script Path, change the default value `Jenkinsfile` to `devops-camp-jenkinsfile`

> **Note:** By default, Jenkins will look for a file named “Jenkinsfile” to build the pipeline with. Since ours is named something different, we must specify it in the "Script Path".



Branches to build

Branch Specifier (blank for 'any')

*/master

Repository browser

(Auto)

Additional Behaviours

Script Path

devops-camp-jenkinsfile

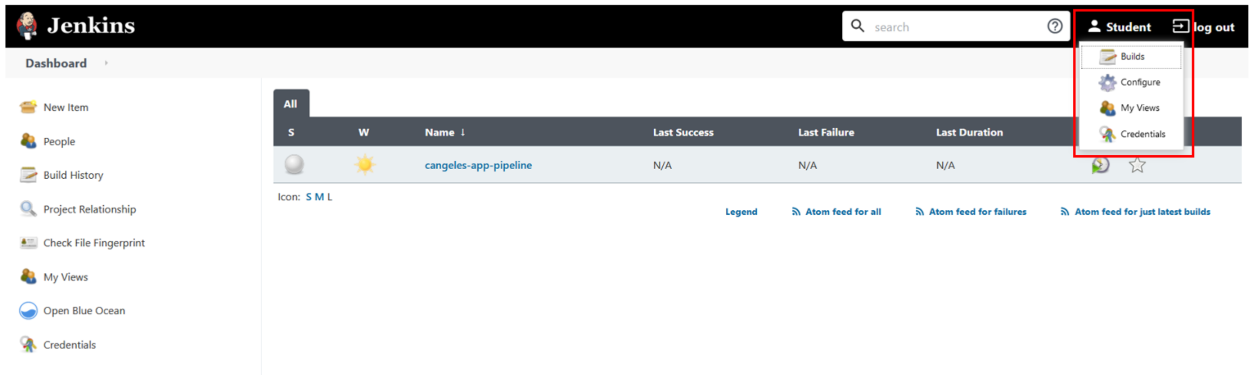
Lightweight checkout

7. Click the “Save” button at the bottom of the page to create the pipeline

Create Jenkins API Token

API tokens will allow one of your GitHub webhooks to authenticate with the Jenkins server and send notifications to trigger a pipeline build. Follow the instructions below to generate an API token for your Jenkins account.

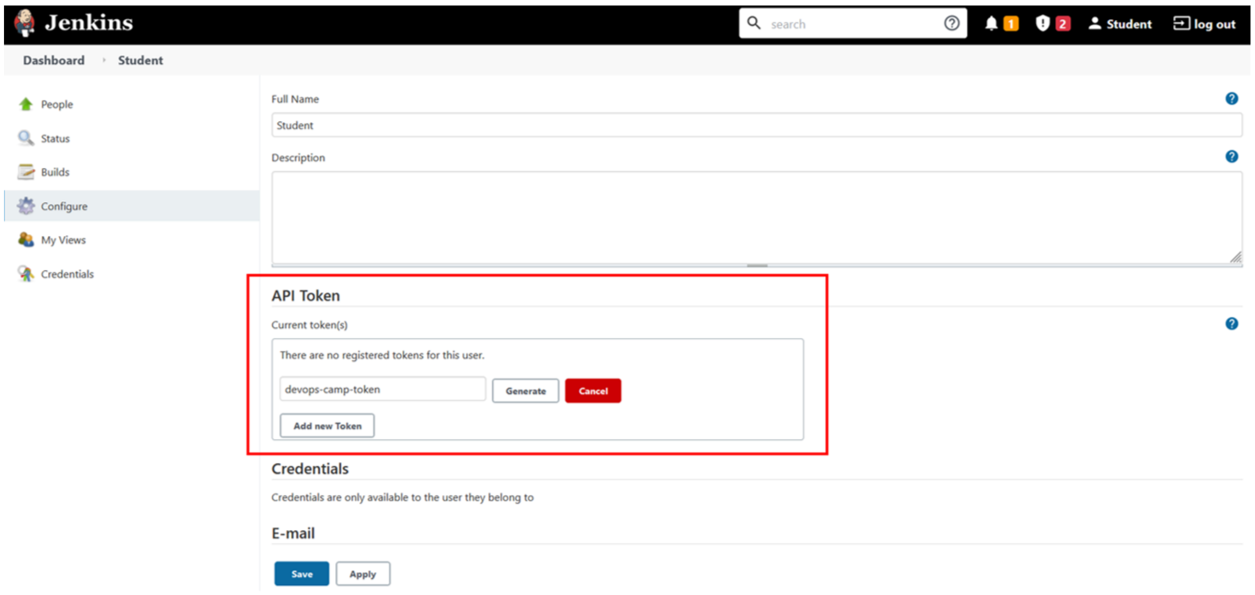
1. On the drop-down menu of your account name, click on “Configure”



S	W	Name ↓	Last Success	Last Failure	Last Duration
		cangeles-app-pipeline	N/A	N/A	N/A

2. Within the API Token section, click on “Add new Token”

3. Name your token `devops-camp-token` and click on “Generate”



Full Name

Student

Description

API Token

Current token(s)

There are no registered tokens for this user.

devops-camp-token Generate Cancel

Add new Token

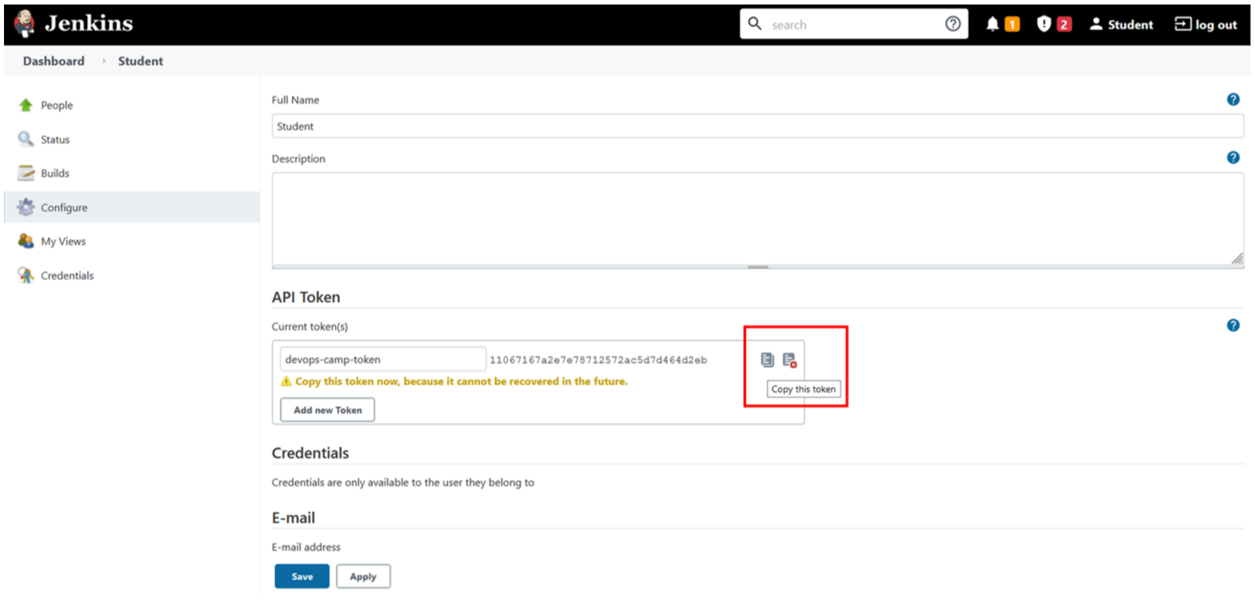
Credentials

Credentials are only available to the user they belong to

E-mail

Save Apply

4. Copy the generated token



Full Name

Student

Description

API Token

Current token(s)

devops-camp-token 11067167a2e7e78712572ac5d7d464d2eb

⚠ Copy this token now, because it cannot be recovered in the future.

Copy this token

Add new Token

Credentials

Credentials are only available to the user they belong to

E-mail

E-mail address

Save Apply

> **Tip:** *Paste this token into the cheat sheet we provided for you*, because it will be used as part of the "Payload URL" of a GitHub webhook.

5. Click on the “Save” button at the bottom of the page once you’ve copied your token

Create GitHub Webhooks

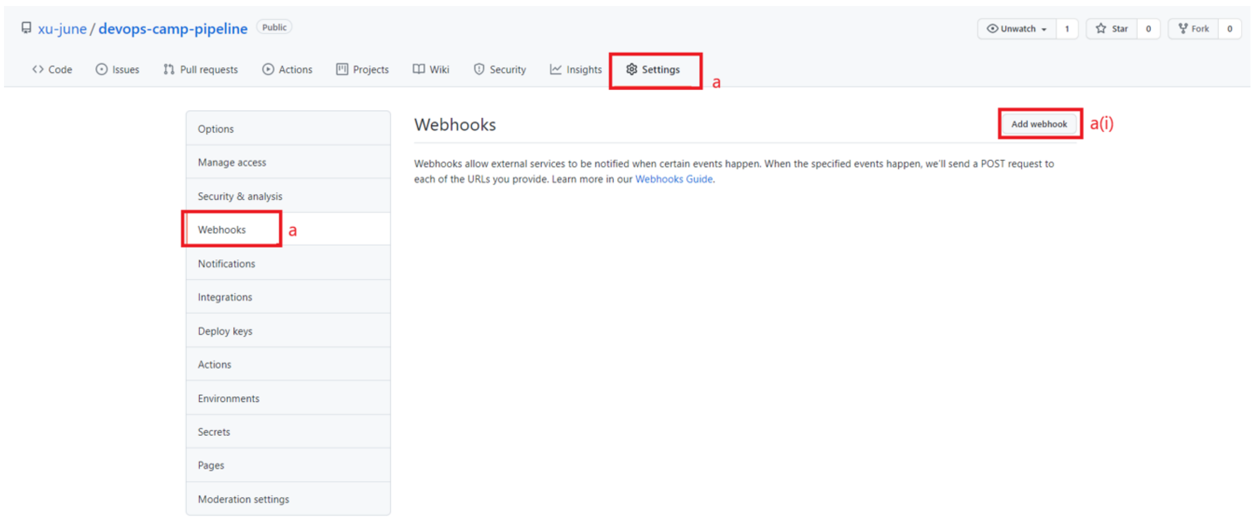
Webhooks will automate our build process by sending notifications to the Jenkins server when commit changes are pushed to your GitHub repositories. You can learn more from: https://plugins.jenkins.io/github/

In this section, you will create two webhooks—one for the `devops-camp-pipeline` repository and the other for the `afs-labs-student` repository.

1. Navigate to your `devops-camp-pipeline` repository on the GitHub website and follow the instructions below to add a webhook

   a. Under Settings → Webhooks

      i. Click on "Add webhook"

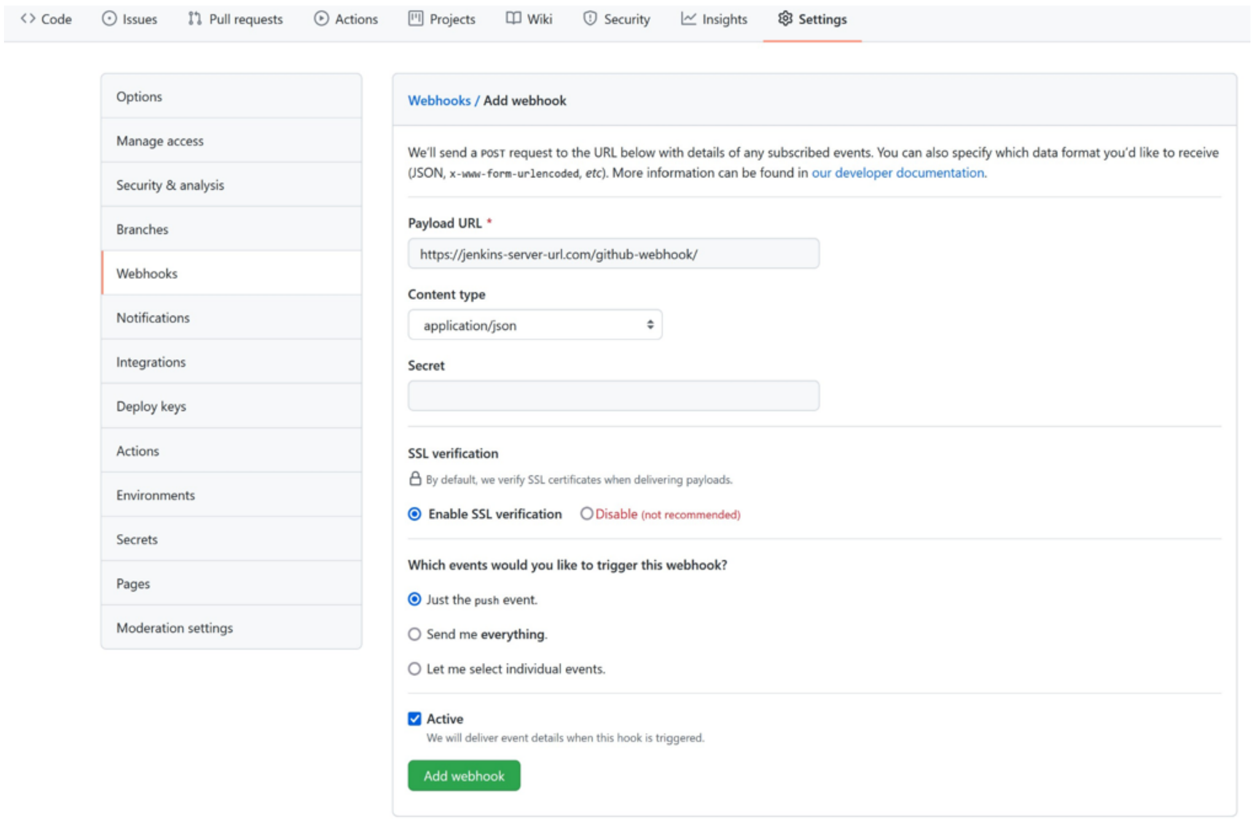


Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

   b. Change the "Content type" to "application/json"

   c. For the "Payload URL," paste your Jenkins server URL followed by "github-webhook/"; it should be:

```
https://jenkins.dev.afsmtddso.com/github-webhook/
```



We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

Payload URL *: https://jenkins-server-url.com/github-webhook/

Content type: application/json

Secret:

SSL verification
By default, we verify SSL certificates when delivering payloads.
[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?
[✓ Just the push event.]
Send me everything.
Let me select individual events.

[✓ Active]
We will deliver event details when this hook is triggered.

d. Click “Add webhook” to create this webhook

2. Navigate to your GitHub `afs-labs-student` repository

a. Under Settings → Webhooks, click on “Add webhook”

b. In the “Payload URL” box, paste in the following URL filled in with your information:

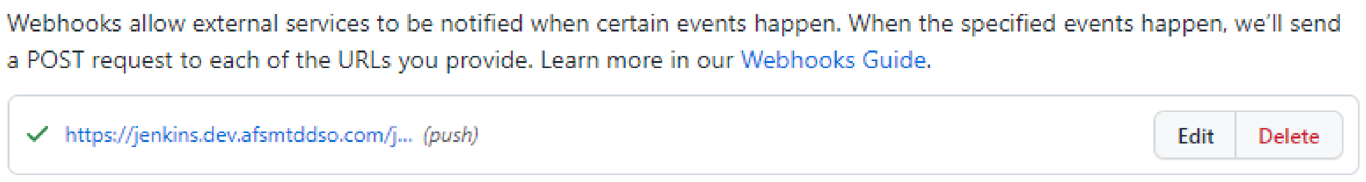
```
https://<JENKINS USERNAME>:<API TOKEN>@jenkins.dev.afsmtddso.com/job/<JENKINS PIPEL
```

> **Tip:** Your API token is the token that you created and saved in the previous section, "Create Jenkins API Token". Your Jenkins pipeline name should be `<YOUR FIRST INITIAL + LAST NAME>-app-pipeline`

c. Make sure the “Content type” is "application/json"

d. Click on “Add webhook” to create the webhook

e. Refresh the page and if you see a green check mark next to a link, you have successfully created your webhook.


Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ https://jenkins.dev.afsmtddso.com/j... (push)

> **Explanation**
>
> The `devops-camp-pipeline` webhook is simple because you have already linked that repository to your Jenkins pipeline. In contrast, you need to need to specify the API token and pipeline name for the `afs-labs-student` webhook because this link has not already been established and so you have to tell this information exactly where to go.

> **Note:** You can learn more about Jenkins API from: https://www.jenkins.io/doc/book/using/remote-access-api/

Test Pipeline

In the previous section, you configured a Jenkins pipeline to monitor changes to a source code repository (GitHub) and created a webhook to trigger an automated build when changes are made to the web application or the pipeline itself.

In this section, you will create a Jenkinsfile that contains the definition of how your pipeline will run. You can learn more about Jenkins Pipelines from: https://www.jenkins.io/doc/book/pipeline/

1. In VSCode, open your `devops-camp-jenkinsfile`

   a. Copy and paste the template below to your `devops-camp-jenkinsfile`

   b. *Replace* the section `<YOUR GITHUB REPOSITORY URL>`, with your `afs-labs-student` GitHub URL

   > **Tip:** *Make sure to include* the `.git` portion of the URL.

   c. Save the changes you made to the `devops-camp-jenkinsfile`

```
pipeline {
    agent {
        label 'jenkins-agent'
    }
    //TODO("Lab 4 - Artifact Management"): add environment variables
    stages {
        stage('Application repository') {
            steps {
                echo "Cloning application repository"
                //TODO("Lab 3 - Introduction to Automated Builds"): add github url
                sh 'git clone <YOUR GITHUB REPOSITORY URL>'
                //TODO("Lab 4 - Artifact Management"): add COMMIT_HASH variable
            }
        }
        stage('Application docker build') {
            steps {
                echo "Building application image"
                //TODO("Lab 4 - Artifact Management"): build the docker app image
            }
            //TODO("Lab 4 - Artifact Management"): clean local docker app image
        }
        stage('Database docker build') {
            steps {
                echo "Building database image"
                //TODO("Lab 4 - Artifact Management"): build the docker db image
            }
            //TODO("Lab 4 - Artifact Management"): clean local docker db image
        }
        //TODO("Lab 7 - Automated Security Scans"): add a stage
        stage('Deploy') {
            steps {
                echo "Deployment stage"
                //TODO("Lab 5 - Deploy to EKS"): deploy docker images
            }
        }
    }
    post {
        cleanup {
            echo "Clean workspace"
            sh 'rm -rf .git ./*'
        }
    }
}
```

2. In the VSCode terminal, within your lab environment container, navigate to your `devops-camp-pipeline` folder

```
cd devops-camp-pipeline
```

3. Configure your GitHub account in your "lab-env" container.

   a. You can do this by running the following commands in your terminal

```
git config --global user.name "<YOUR GITHUB USERNAME>"
```

```
git config --global user.email <YOUR EMAIL>
```

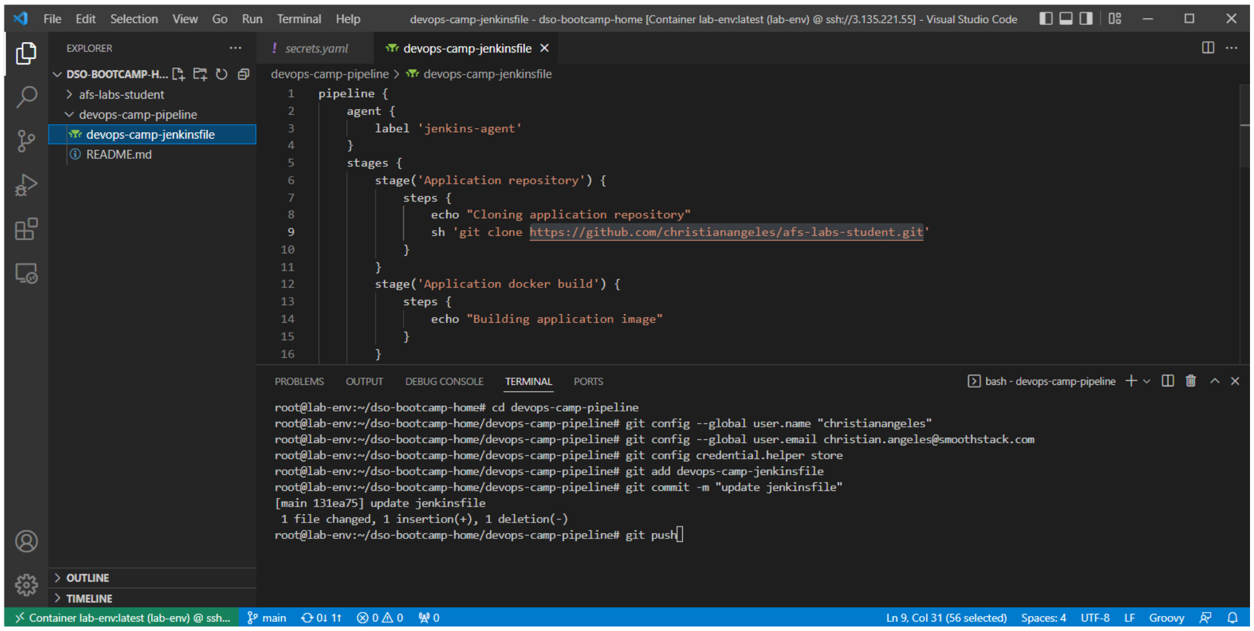
```
git config credential.helper store
```

> **Note:** The credential helper will store your GitHub PAT so you won't have to re-enter it every time you fetch or push to your repository.

4. Commit and push your changes

   a. You'll need to add your file to git by running `git add devops-camp-jenkinsfile`

   b. Run `git commit -m "updated jenkinsfile"` to commit to your changes.



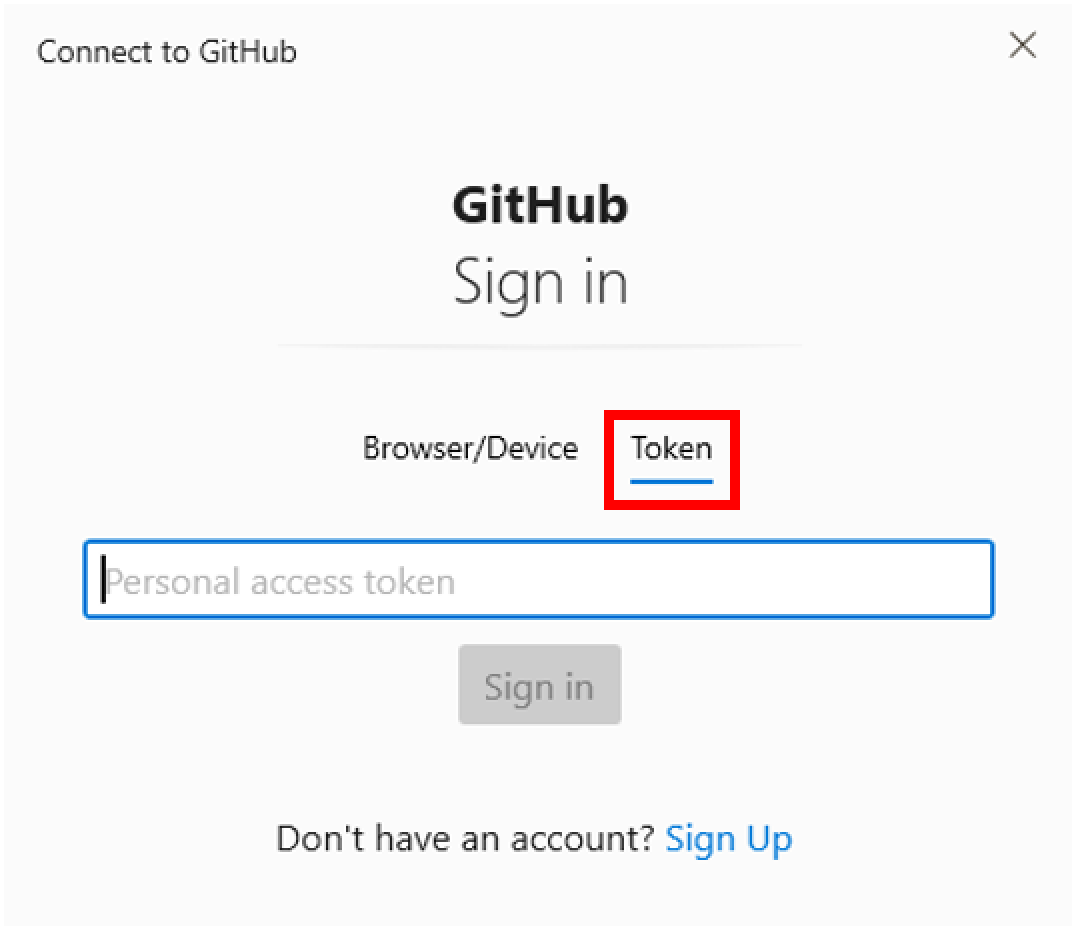
```
pipeline {
    agent {
        label 'jenkins-agent'
    }
    stages {
        stage('Application repository') {
            steps {
                echo "Cloning application repository"
                sh 'git clone https://github.com/christianangeles/afs-labs-student.git'
            }
        }
        stage('Application docker build') {
            steps {
                echo "Building application image"
            }
        }
```

```
root@lab-env:~/dso-bootcamp-home# cd devops-camp-pipeline
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git config --global user.name "christianangeles"
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git config --global user.email christian.angeles@smoothstack.com
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git config credential.helper store
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git add devops-camp-jenkinsfile
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git commit -m "update jenkinsfile"
[main 131ea75] update jenkinsfile
 1 file changed, 1 insertion(+), 1 deletion(-)
root@lab-env:~/dso-bootcamp-home/devops-camp-pipeline# git push
```

File Edit Selection View Go Run Terminal Help
devops-camp-jenkinsfile - dso-bootcamp-home [Container lab-env:latest (lab-env) @ ssh://3.135.221.55] - Visual Studio Code
EXPLORER
DSO-BOOTCAMP-H...
afs-labs-student
devops-camp-pipeline
devops-camp-jenkinsfile
README.md
secrets.yaml
devops-camp-pipeline > devops-camp-jenkinsfile
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash - devops-camp-pipeline
OUTLINE
TIMELINE
Container lab-env:latest (lab-env) @ ssh...
main
Ln 9, Col 31 (56 selected) Spaces: 4 UTF-8 LF Groovy

   c. Run `git push` in order to sync changes from your local repository over to the remote repository

> **Tip:** You'll either be prompted to enter your GitHub PAT again or authorize VS Code for additional permissions.

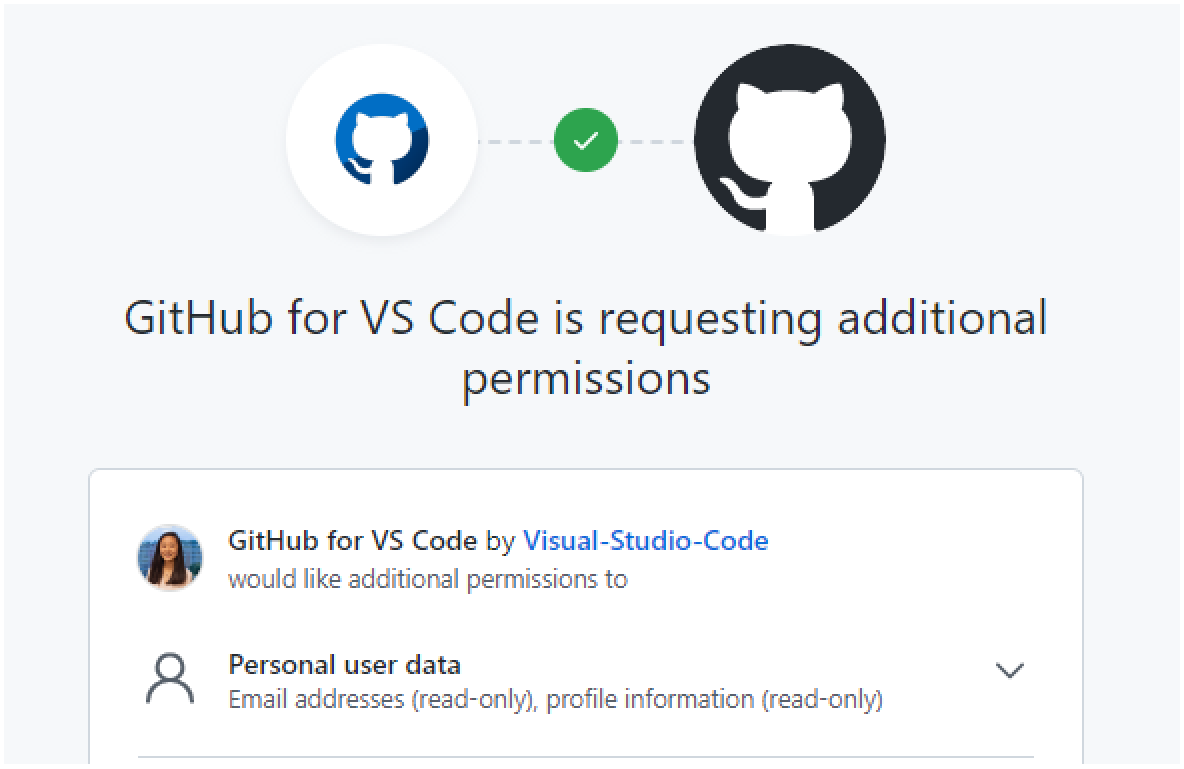


Connect to GitHub

GitHub

Sign in

Don't have an account? Sign Up



GitHub for VS Code is requesting additional permissions

GitHub for VS Code by Visual-Studio-Code would like additional permissions to

Personal user data

Email addresses (read-only), profile information (read-only)