

## Lesson 10 Demo 03

### Creating an Event-Based Workflow

**Objective:** To create an event-based workflow using the GitHub Actions trigger to automatically initiate continuous integration processes whenever new code is pushed to the repository

**Tools required:** GitHub Actions

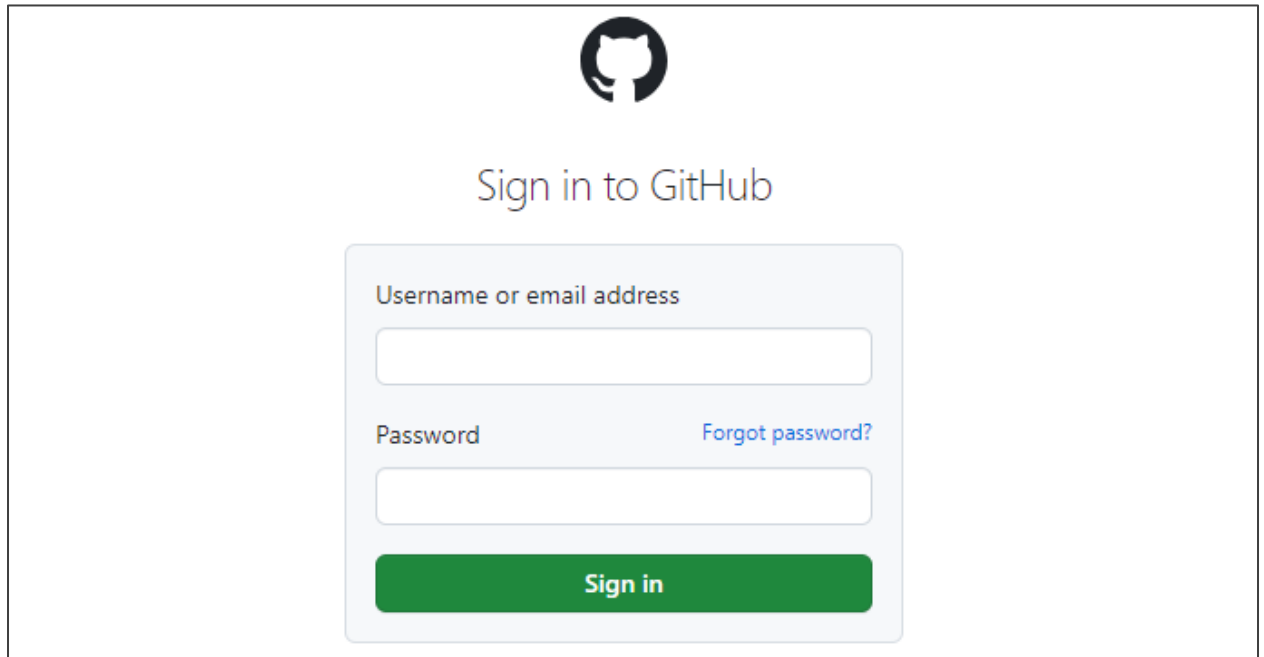
**Prerequisites:** None

Steps to be followed:

1. Create a new GitHub repository
2. Create and execute a new workflow file using the GitHub Actions trigger

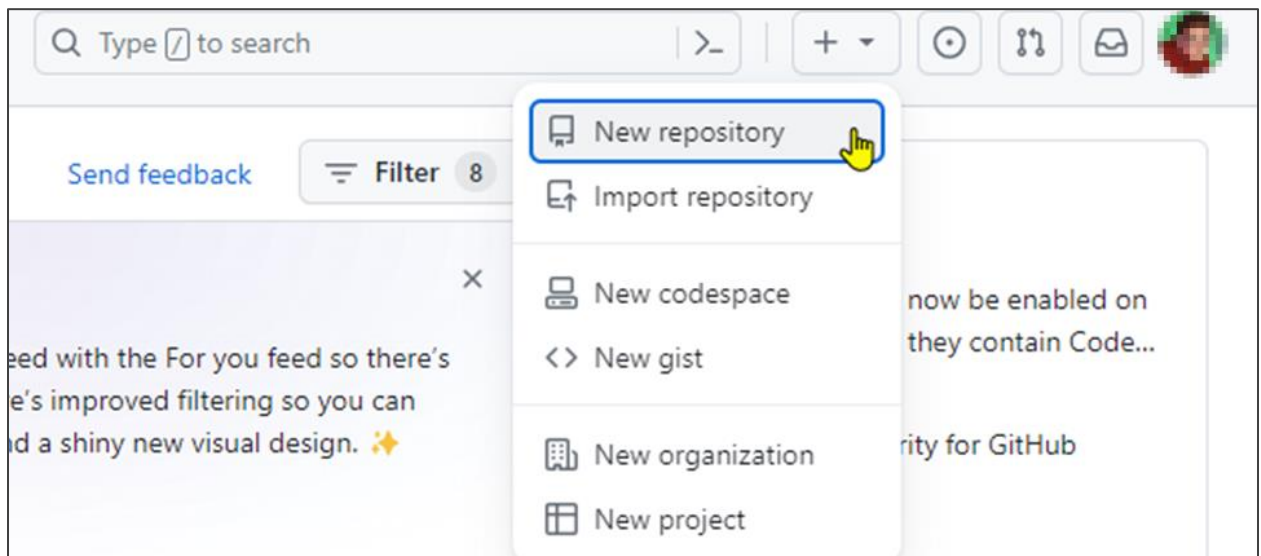
#### Step 1: Create a new GitHub repository

1.1 Open the browser in your lab, go to **github.com**, and log in to your account

A screenshot of the GitHub login page. At the top center is the GitHub logo (an octocat). Below it, the text "Sign in to GitHub" is displayed. Underneath is a light blue rounded rectangle containing the login form. The form has two input fields: "Username or email address" and "Password". To the right of the password field is a blue link that says "Forgot password?". At the bottom of the form is a green button with the text "Sign in" in white.

**Note:** If you do not have a GitHub account, visit the official website at <https://github.com/signup> and create a new account

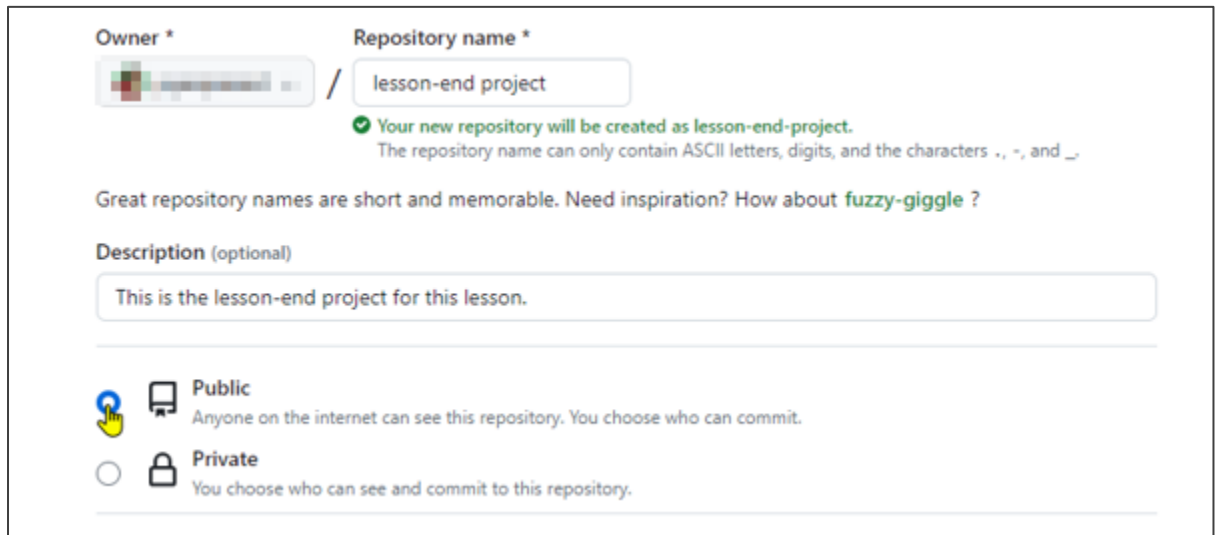
- 1.2 Click on the + icon from the upper-right corner of the page and select **New repository** from the drop-down menu




- 1.3 Enter the name and description of the GitHub repository

A screenshot of the 'Create a new repository' form on GitHub. The form has a title 'Create a new repository' and a subtitle 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. Below this is a note: 'Required fields are marked with an asterisk (\*)'. The form contains two main sections: 'Owner \*' and 'Repository name \*'. The 'Owner' field is a dropdown menu showing a user profile picture and the name 'lesson-end project'. The 'Repository name' field is a text input box containing 'lesson-end project'. Below the 'Repository name' field is a green checkmark icon and the text: 'Your new repository will be created as lesson-end-project. The repository name can only contain ASCII letters, digits, and the characters -, ., and \_'. Below this is a line of text: 'Great repository names are short and memorable. Need inspiration? How about fuzzy-giggle ?'. At the bottom, there is a 'Description (optional)' section with a text input box containing 'This is the lesson-end project for this lesson.'.

#### 1.4 Choose **Public** for the repository type





Owner \*  / Repository name \*

✓ Your new repository will be created as lesson-end-project.  
The repository name can only contain ASCII letters, digits, and the characters -, ., and \_.

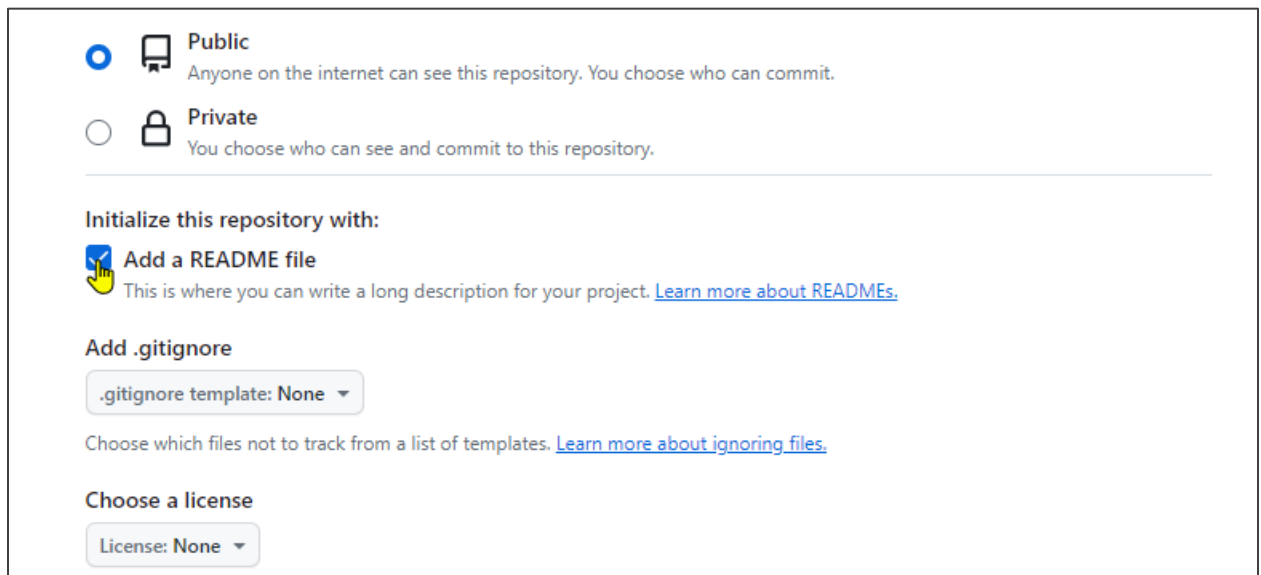
Great repository names are short and memorable. Need inspiration? How about [fuzzy-giggle](#) ?


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.


#### 1.5 Select **Add a README file** to include a README file for the repository



☒  **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:

☒  **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

## 1.6 Click on the **Create repository** button

Initialize this repository with:

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

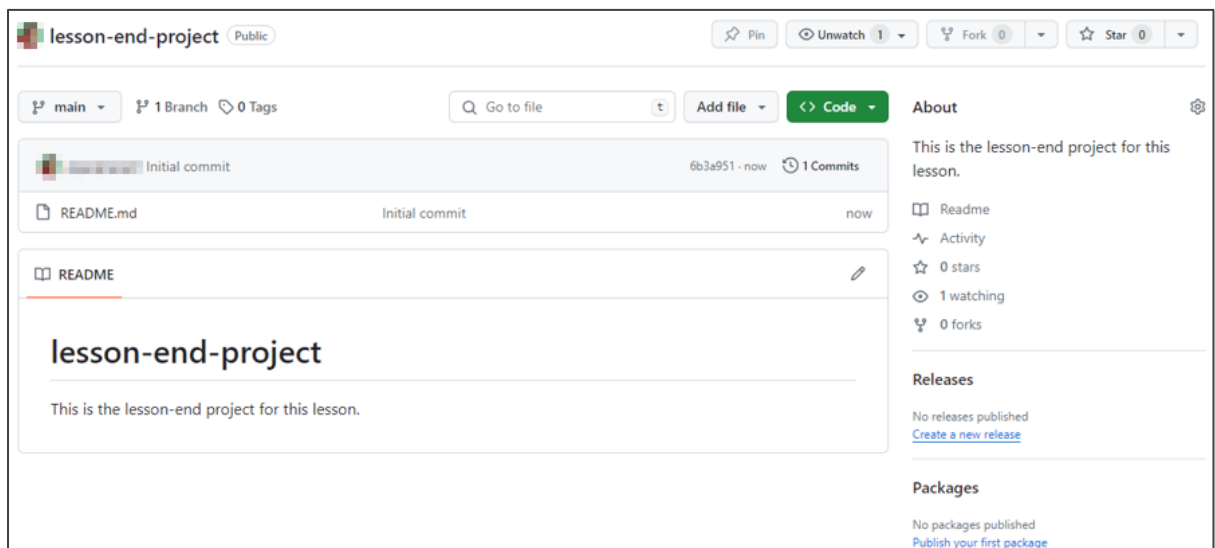
**Add .gitignore**  
.gitignore template: **None** ▾  
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

**Choose a license**  
License: **None** ▾  
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set **main** as the default branch. Change the default name in your [settings](#).

You are creating a public repository in your personal account.

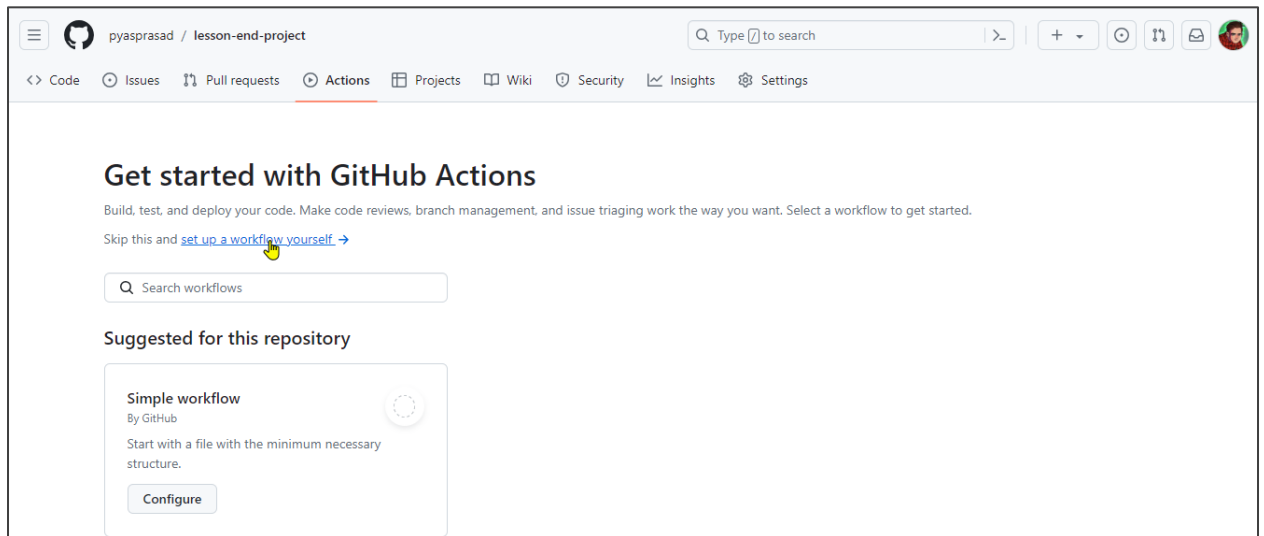
**Create repository**



The remote GitHub repository is created.

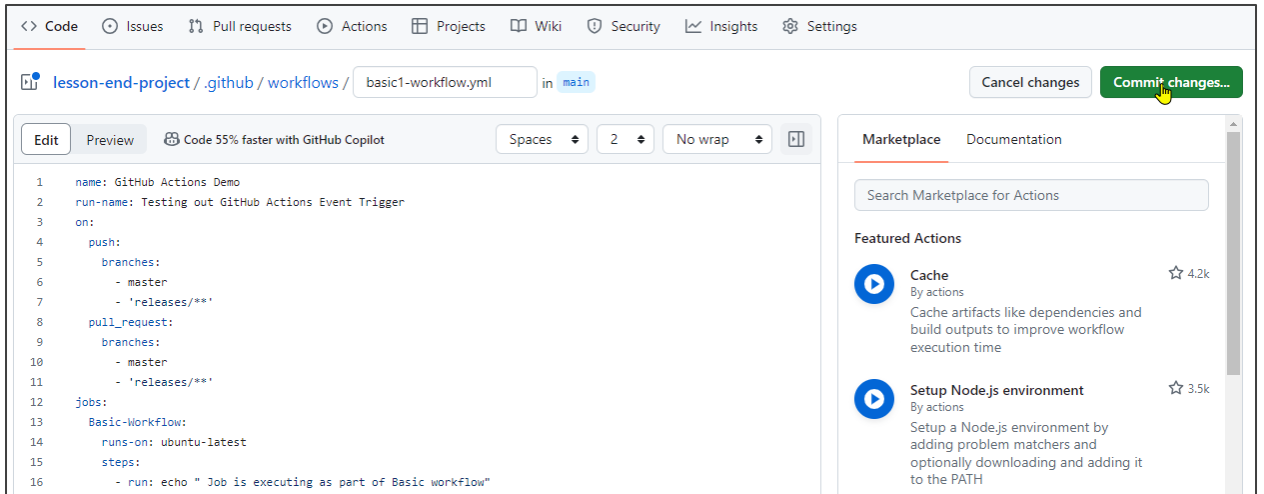
## Step 2: Create and execute a new workflow file using the GitHub Actions trigger

2.1 Navigate to the **Actions** tab and click on **set up a workflow yourself** to create a **.github/workflows** directory

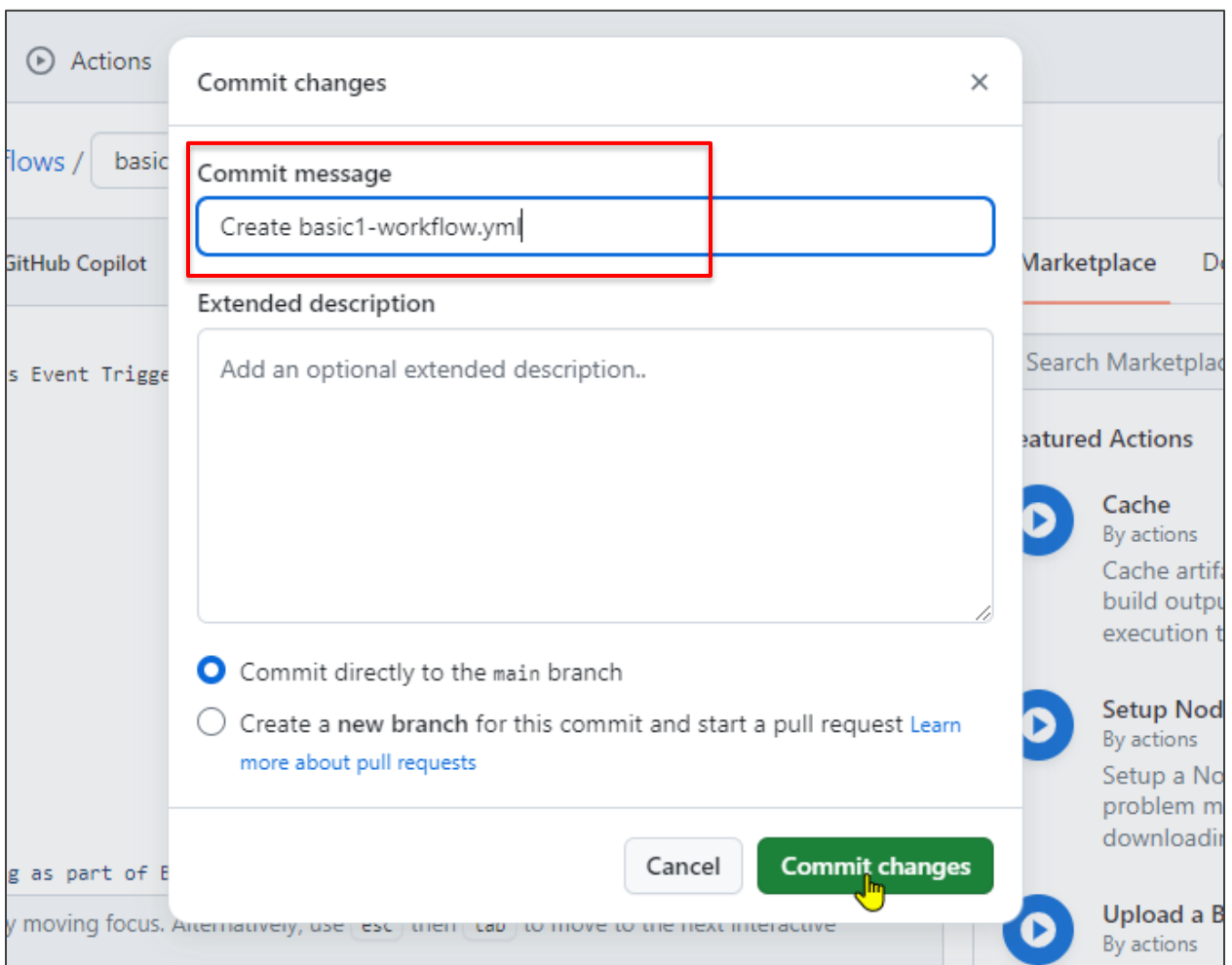


2.2 Create a new workflow file **basic1-workflow.yml** with the below code, and then click on **Commit changes**:

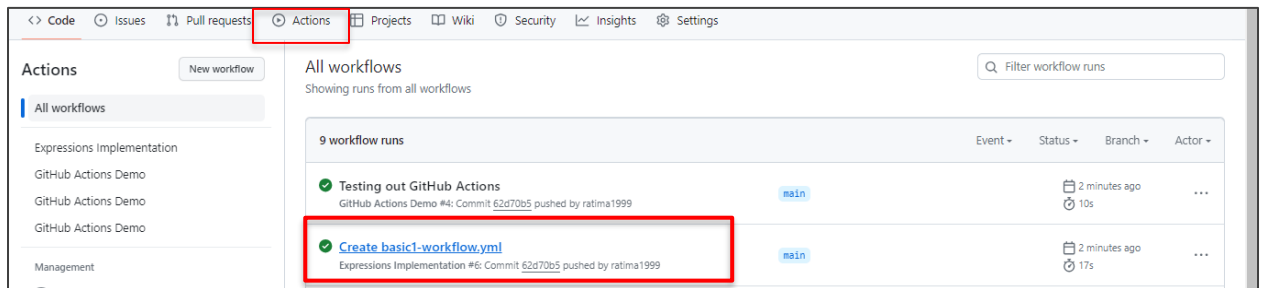
```
name: GitHub Actions Demo
run-name: Testing out GitHub Actions Event Trigger
on:
  push:
    branches:
      - master
      - 'releases/**'
  pull_request:
    branches:
      - master
      - 'releases/**'
jobs:
  Basic-Workflow:
    runs-on: ubuntu-latest
    steps:
      - run: echo " Job is executing as part of Basic workflow"
```



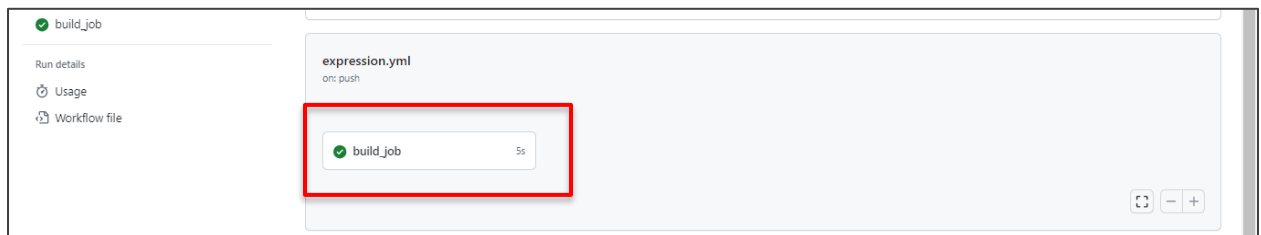
2.3 Add **Commit message** as **Create basic1-workflow.yml** and click on **Commit changes** to save the workflow file in the code repository



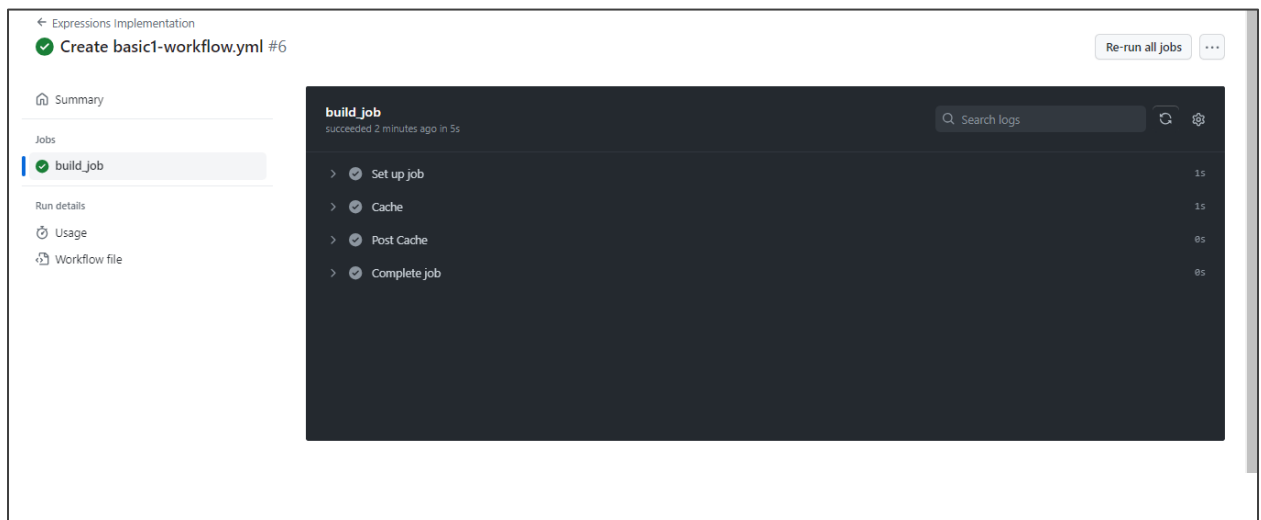
2.4 Navigate to the **Actions** tab, and under **All workflows**, click on the **Create basic1-workflow.yml** workflow to execute the workflow



2.5 Click on the **build\_job** to see the output



The following **build\_job** screen will appear:



## 2.6 Expand the **output** located under **build\_job**

← Expressions Implementation  
✓ Create basic1-workflow.yml #6 Re-run all jobs ...

Summary

Jobs

- ✓ build\_job

Run details

Usage

Workflow file

**build\_job**  
succeeded 3 minutes ago in 5s

Search logs

✓ Set up job 1s

- 1 Current runner version: '2.311.0'
- 2 ▶ Operating System
- 3 ▶ Runner Image
- 4 ▶ Runner Image Provisioner
- 5 ▶ Runner Image Provisioner
- 6 ▶ GitHub\_TOKEN Permissions
- 7 Secret source: Actions
- 8 Prepare workflow directory
- 9 Prepare all required actions
- 10 Getting action download info
- 11 Download action repository 'actions/cache@v1.2.1' (SHA:f5ce41475b483ad7581884324a6eca9f48f8dc7)
- 12 Complete job name: build\_job

✓ Cache 1s

- 1 ▶ Run actions/cache@v1.2.1
- 2 Cache Size: ~0 MB (84 B)
- 3 C:\Windows\System32\tar.exe -xz -f D:/a/\_temp/a8ebc3b9-511b-4973-9be1-cfb01413495f/cache.tgz -C D:/a/GitHubActionsDemos/GitHubActionsDemos
- 4 Cache restored from key: artifact

Summary

Jobs

- ✓ build\_job

Run details

Usage

Workflow file

**build\_job**  
succeeded 3 minutes ago in 5s

Search logs

✓ Set up job 1s

- 2 ▶ Operating System
- 3 ▶ Runner Image
- 4 ▶ Runner Image Provisioner
- 5 ▶ Runner Image Provisioner
- 6 ▶ GitHub\_TOKEN Permissions
- 7 Secret source: Actions
- 8 Prepare workflow directory
- 9 Prepare all required actions
- 10 Getting action download info
- 11 Download action repository 'actions/cache@v1.2.1' (SHA:f5ce41475b483ad7581884324a6eca9f48f8dc7)
- 12 Complete job name: build\_job

✓ Cache 1s

- 1 ▶ Run actions/cache@v1.2.1
- 2 Cache Size: ~0 MB (84 B)
- 3 C:\Windows\System32\tar.exe -xz -f D:/a/\_temp/a8ebc3b9-511b-4973-9be1-cfb01413495f/cache.tgz -C D:/a/GitHubActionsDemos/GitHubActionsDemos
- 4 Cache restored from key: artifact

✓ Post Cache 0s

- 1 Post job cleanup.
- 2 Cache hit occurred on the primary key artifact, not saving cache.

✓ Complete job 0s

- 1 Cleaning up orphan processes

By following these steps, you have successfully created an event-based workflow using the GitHub Actions trigger to automatically initiate continuous integration processes whenever new code is pushed to the repository.