

## Lesson-End Project

### Executing CI/CD with GitHub Actions

**Project agenda:** To create an event-based workflow using Git and GitHub Actions for efficient project automation and version control

**Description:** As a developer at a tech company embarking on a significant project, the success of the team depends on seamless collaboration and rapid delivery of high-quality code. To supercharge the development process and maintain code integrity, the focus is on diving into continuous integration (CI) and continuous deployment (CD) workflows using Java CI with Maven. The aim is to revolutionize the workflow and propel the project to new heights of efficiency and excellence.

**Tools required:** Git and GitHub

**Prerequisites:** You must have Git installed in the lab to proceed.

**Expected deliverables:** A GitHub Actions CI/CD workflow to create automated Maven builds

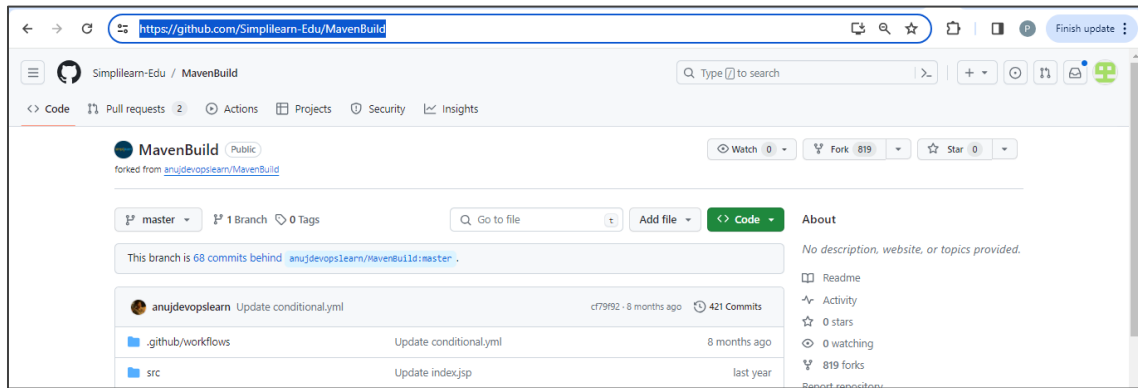
Steps to be followed:

1. Log in to GitHub.com and fork the repository
2. Create a new workflow file
3. Execute the GitHub Actions workflow

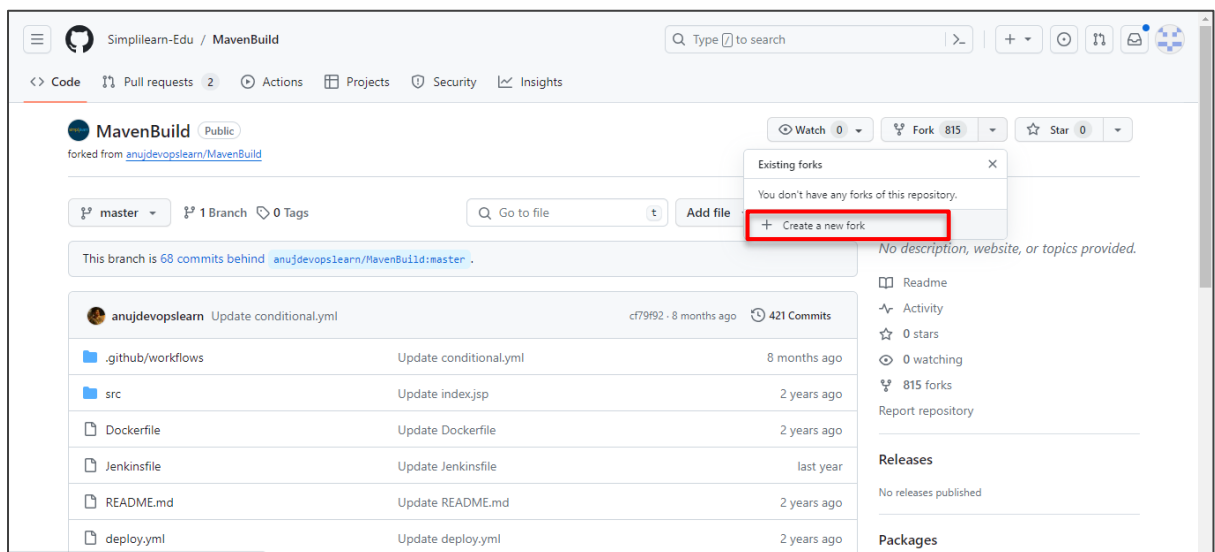
## Step 1: Log in to GitHub.com and fork the repository

1.1 Log in to your GitHub account and use the following link to fork the repository into your GitHub account:

**<https://github.com/Simplilearn-Edu/MavenBuild>**



1.2 Click on the **Fork** tab and select **Create a new fork**



## 1.2 Enter the **Repository name** as **Maven-Build** and click on **Create Fork**

### Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. [View existing forks.](#)

*Required fields are marked with an asterisk (\*).*

Owner \*

GithubResources1

Repository name \*

Maven-Build

✓ Maven-Build is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

☒ Copy the **master** branch only

Contribute back to Simplilearn-Edu/MavenBuild by adding your own branch. [Learn more.](#)

📘 You are creating a fork in your personal account.

Create fork

## Step 2: Create a new workflow file

### 2.1 Navigate to the **Actions** tab and click on **set up a workflow yourself** to create a workflow directory

<> Code Pull requests **Actions** Projects Wiki Security Insights Settings

## Choose a workflow

Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and **set up a workflow yourself** →

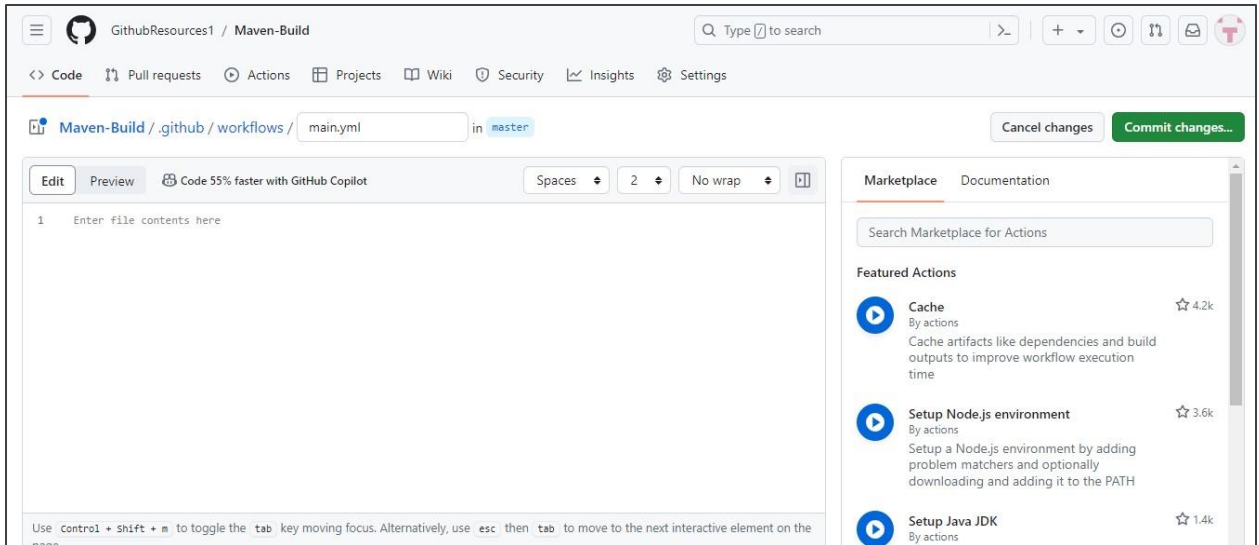
Search workflows

### Suggested for this repository

**Docker image**  
By GitHub Actions  
Build a Docker image to deploy, run, or push to a registry.  
Configure Dockerfile

**Android CI**  
By GitHub Actions  
Build an Android project with Gradle.  
Configure Java

**Java with Ant**  
By GitHub Actions  
Build and test a Java project with Apache Ant.  
Configure Java



The above screen will appear.

2.2 Create a new workflow file named **maven-cache.yml** using the below code and click on **Commit changes:**

**name: Java CI with Maven**

**on:**

**push:**

**concurrency:**

**group: environment-`{{ github.ref }}`**

**cancel-in-progress: true**

**jobs:**

**maven\_build:**

**strategy:**

**matrix:**

**version: [11, 8]**

**runs-on: ubuntu-latest**

**steps:**

**- uses: actions/checkout@v3**

**- name: Set up JDK `"{{ matrix.version }}"`**

**uses: actions/setup-java@v3**

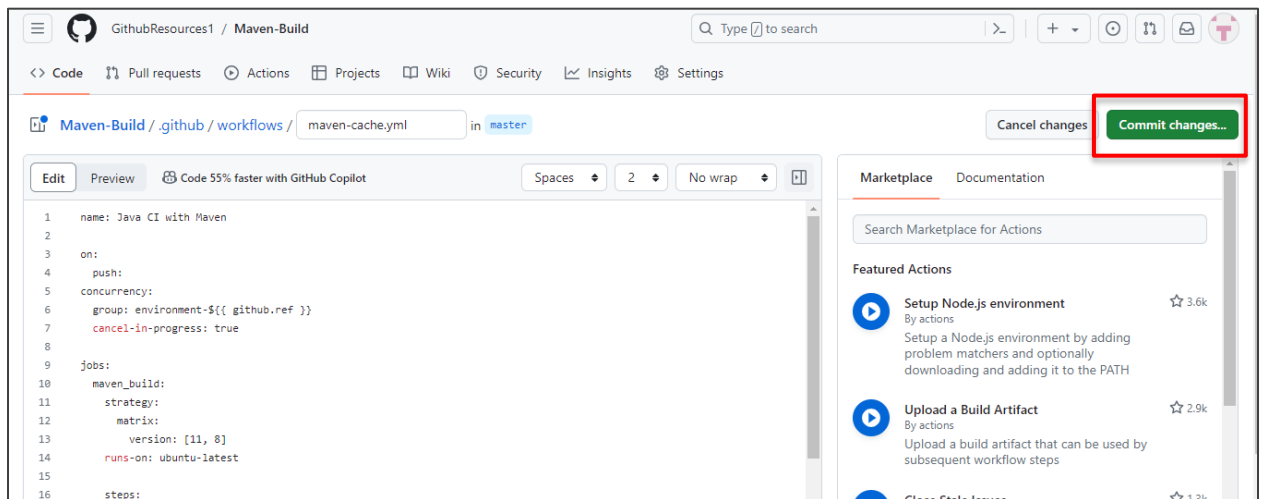
**with:**

**java-version: `"{{ matrix.version }}"`**

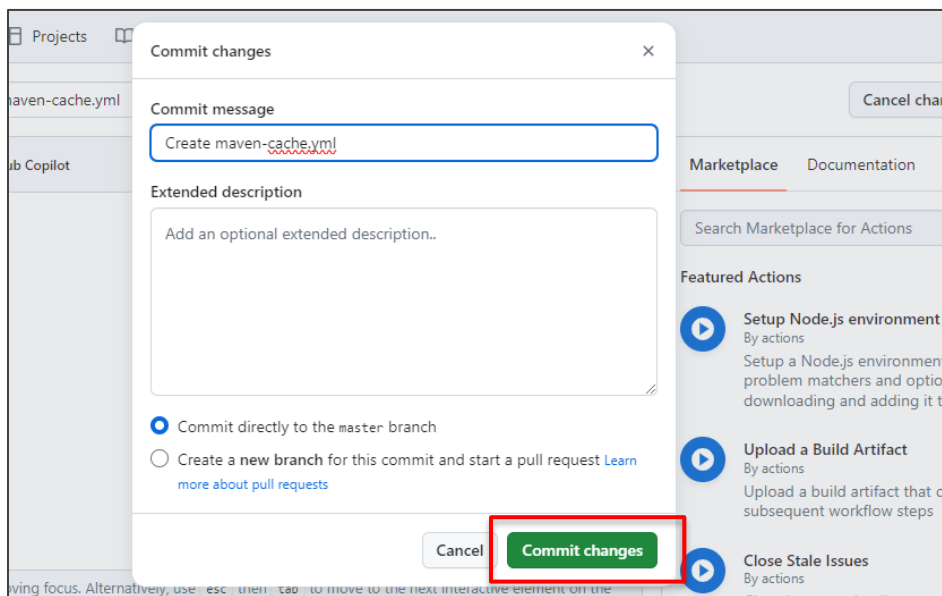
**distribution: 'temurin'**

cache: maven

- name: Cache Maven Dependencies  
uses: actions/cache@v3  
with:  
  path: ~/.m2  
  key: \${{ runner.os }}-cache
- name: Build with Maven  
run: mvn -B package --file pom.xml

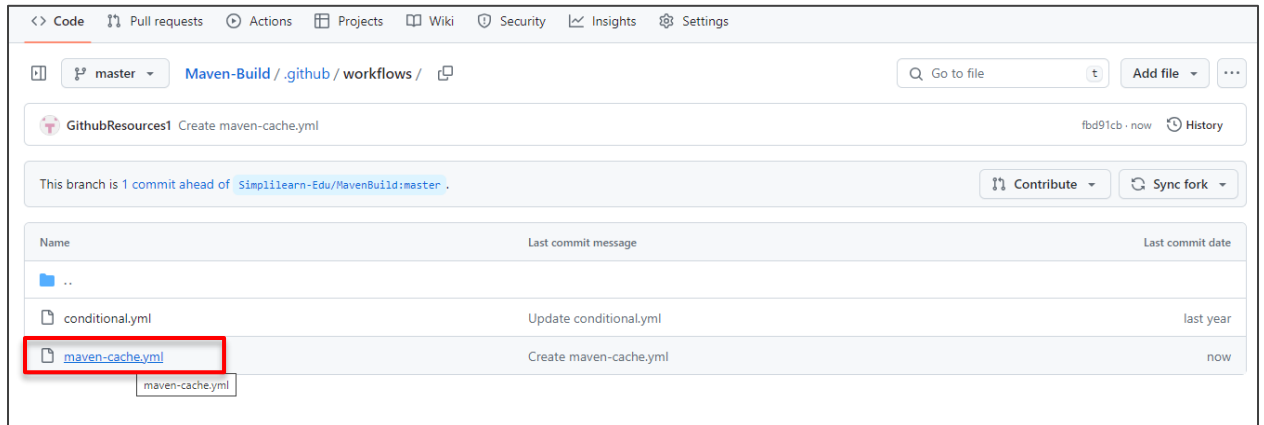


2.3 Enter a **Commit message** and then click on **Commit changes** to save the workflow file in the code repository

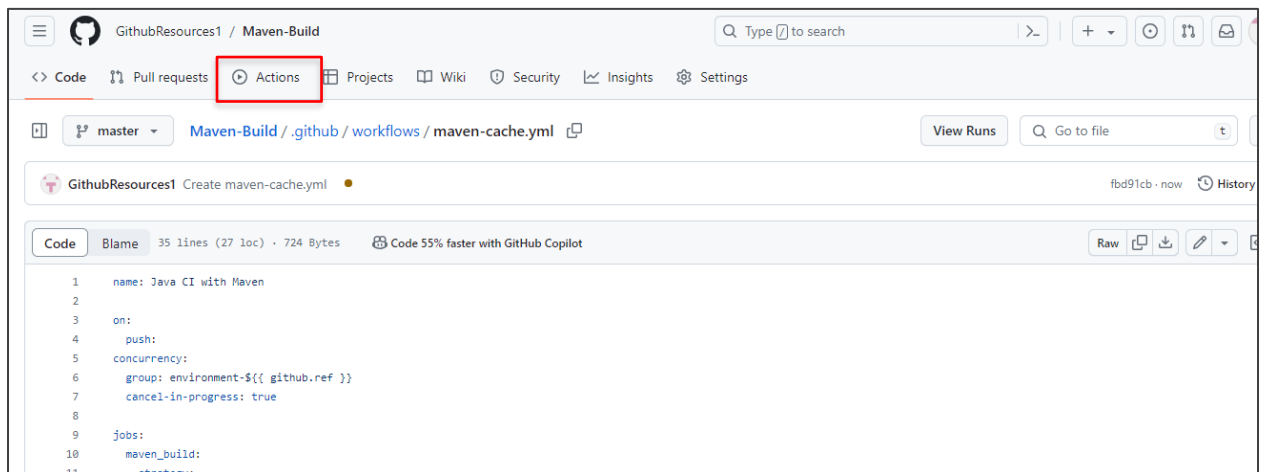


## Step 3: Execute the GitHub Actions workflow

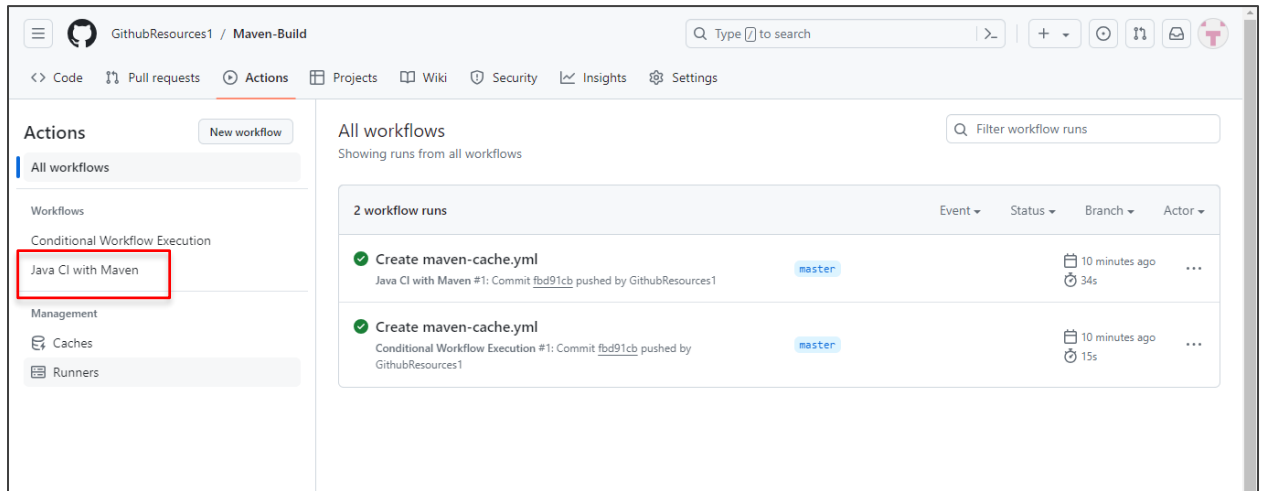
### 3.1 Click on the **maven-cache.yml** workflow file in the main repository page



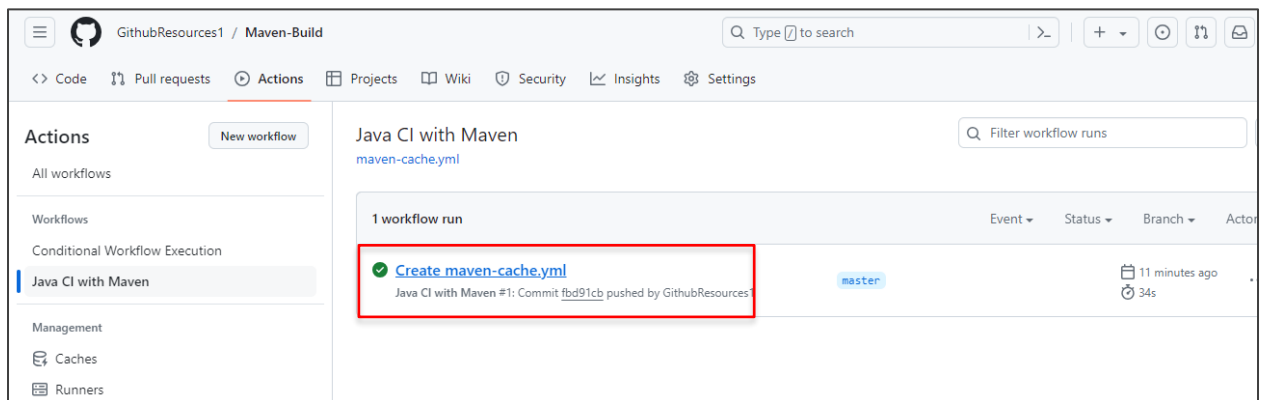
### 3.2 Navigate back to the **Actions** tab in the repository to access the workflow execution



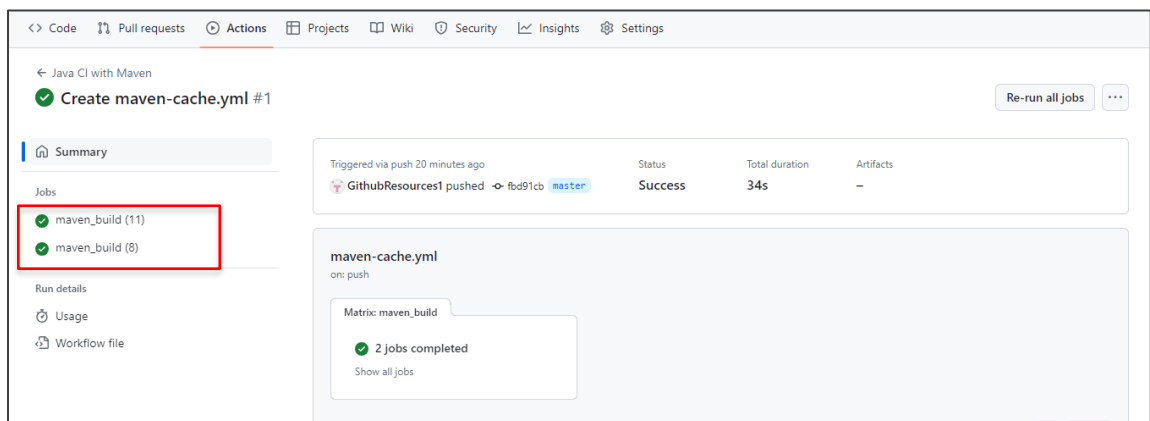
### 3.2 Select Java CI with Maven under All workflows



### 3.3 Then, click on Create maven-cache.yml



### 3.4 Under Jobs, select maven\_build (11) and maven\_build (8) to view all the job logs



This screenshot shows the GitHub Actions interface for a workflow named "Create maven-cache.yml #1". The left sidebar contains a "Jobs" section with two items: "maven\_build (11)" and "maven\_build (8)". The "maven\_build (11)" job is selected, and its details are displayed in the main panel. The job status is "succeeded 17 minutes ago in 22s". The job steps are listed as follows:

Step	Duration
Set up job	10s
Run actions/checkout@v3	1s
Set up JDK "11"	1s
Cache Maven Dependencies	1s
Build with Maven	7s
Post Cache Maven Dependencies	0s
Post Set up JDK "11"	0s
Post Run actions/checkout@v3	0s
Complete job	0s

This screenshot shows the GitHub Actions interface for the same workflow "Create maven-cache.yml #1". The "Jobs" section in the left sidebar now shows "maven\_build (11)" and "maven\_build (8)". The "maven\_build (8)" job is selected, and its details are displayed in the main panel. The job status is "succeeded 18 minutes ago in 15s". The job steps are listed as follows:

Step	Duration
Set up job	1s
Run actions/checkout@v3	1s
Set up JDK "8"	2s
Cache Maven Dependencies	0s
Build with Maven	7s
Post Cache Maven Dependencies	2s
Post Set up JDK "8"	2s
Post Run actions/checkout@v3	0s
Complete job	0s

The above screenshots are of the output of the task performed.

By following these steps, you have successfully created an event-based workflow using the GitHub Actions trigger to initiate the workflow execution. You have also verified the workflow execution and viewed the output of the Maven build.