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.

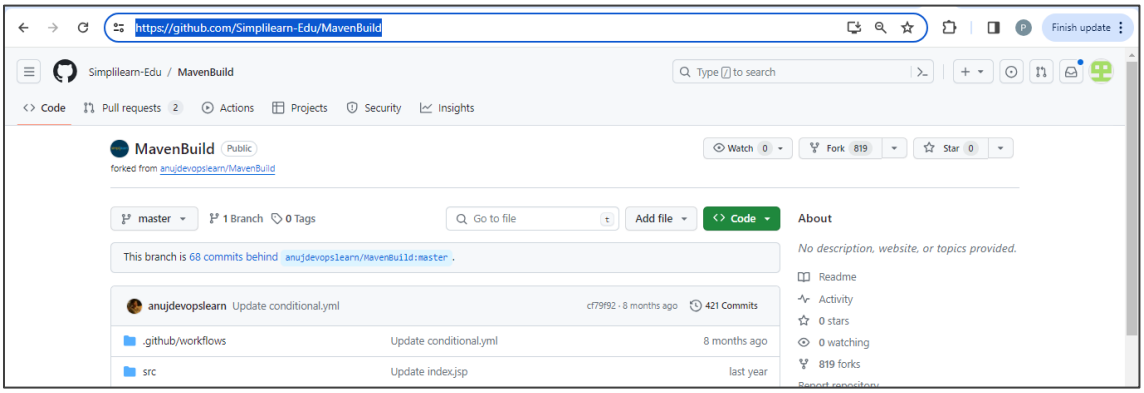
Expected deliverables: A GitHub Actions CI/CD workflow to create automated Maven builds

Steps to be followed:

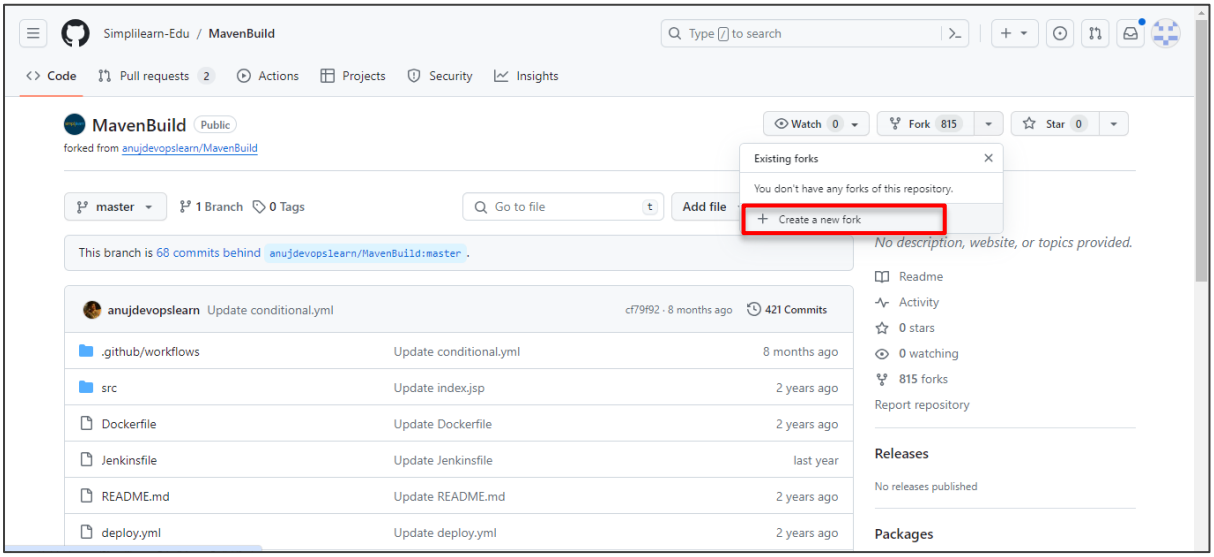
1. Log in to GitHub.com and fork the repository
2. Create a new workflow file
3. Execute the GitHub Actions workflow

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

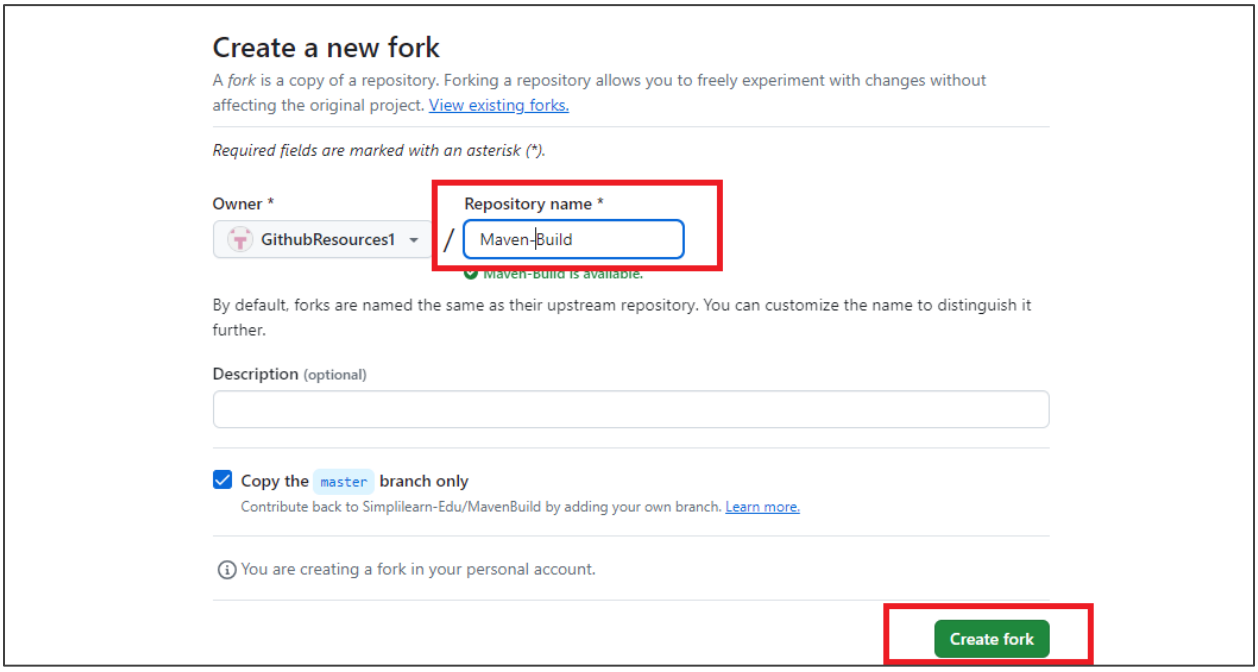
1.1 Log in to your GitHub account and use the following link to fork the repository into your GitHub account:
https://github.com/Simplilearn-Edu/MavenBuild



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

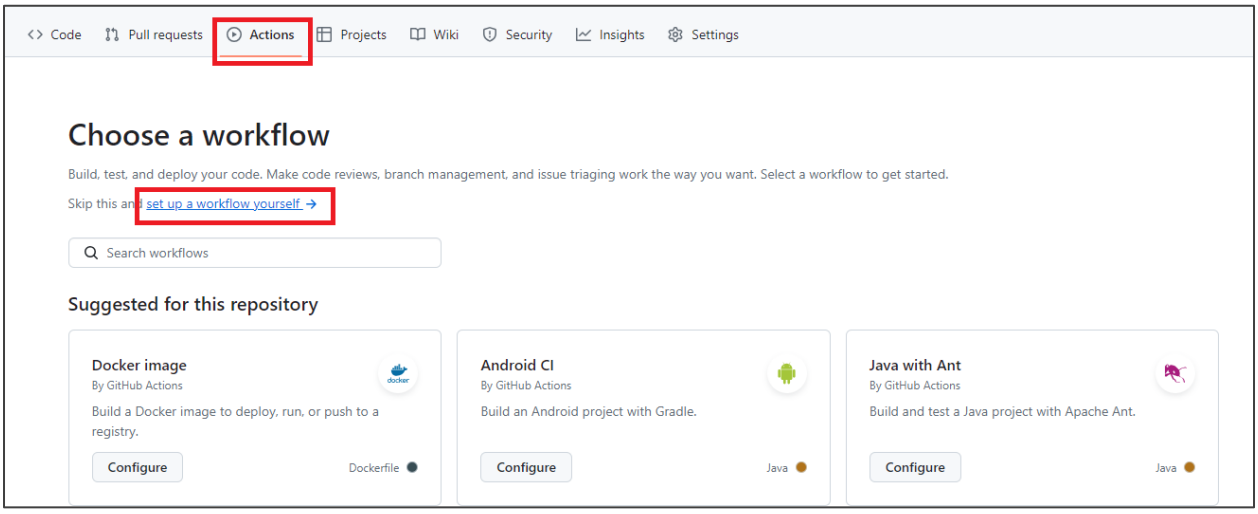


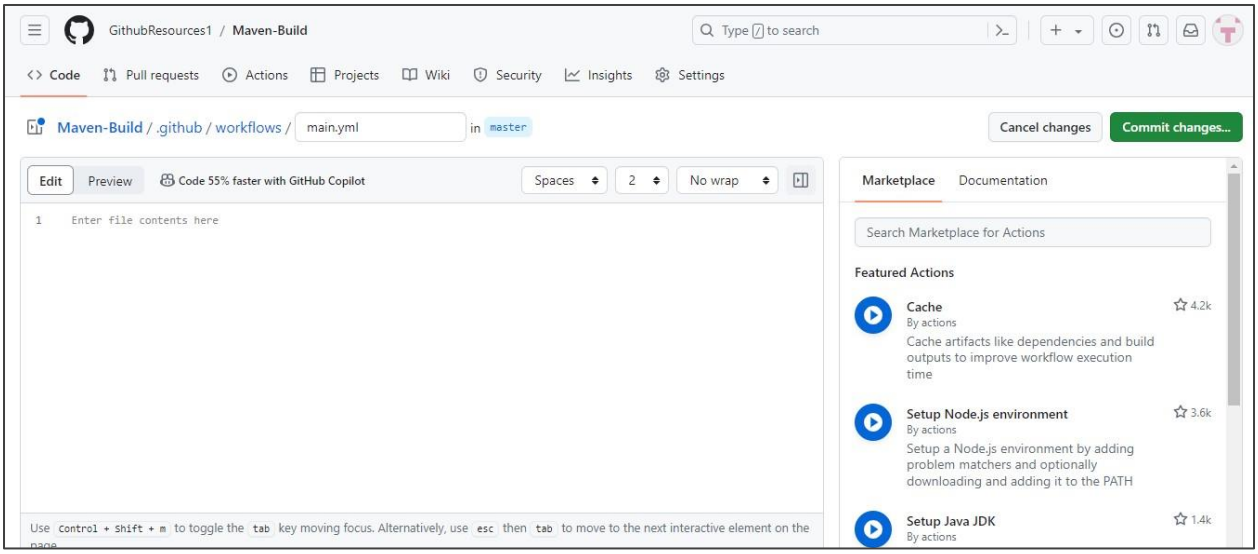
1.2 Enter the **Repository name** as **Maven-Build** and click on **Create Fork**



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



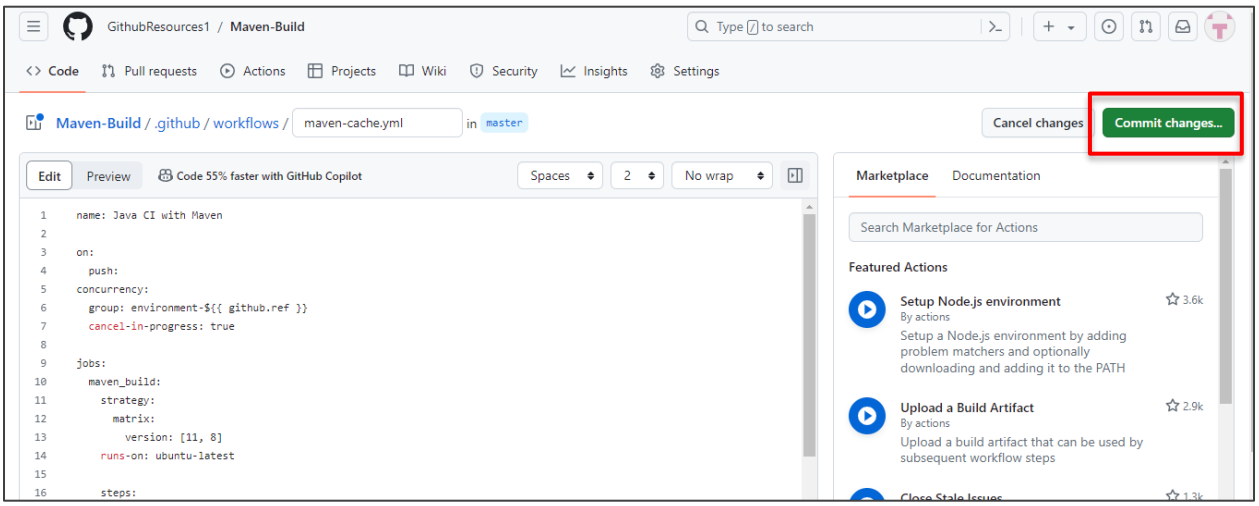


The above screen will appear.

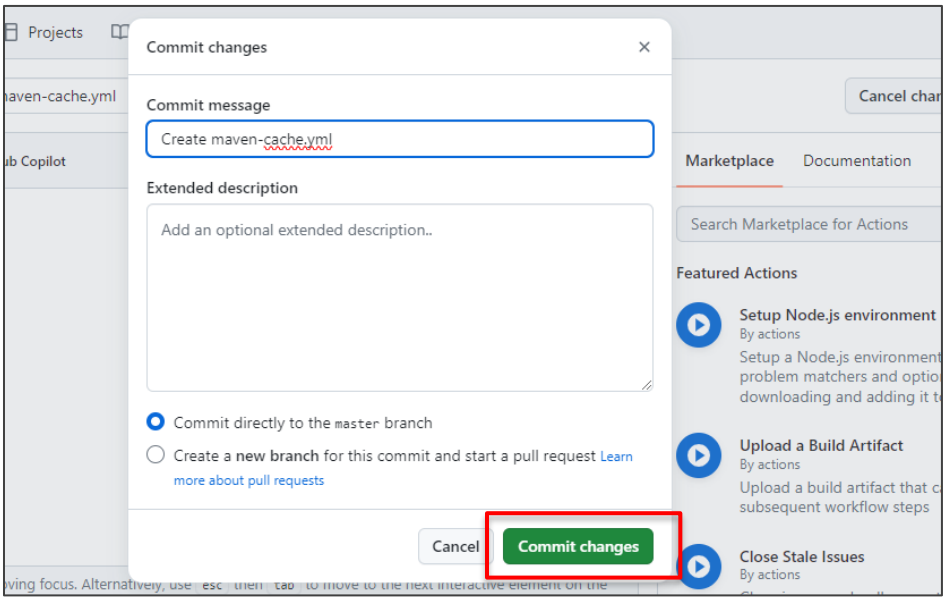
2.2 Create a new workflow file named **maven-cache.yml** using the below code and click on **Commit changes:**

```
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
```

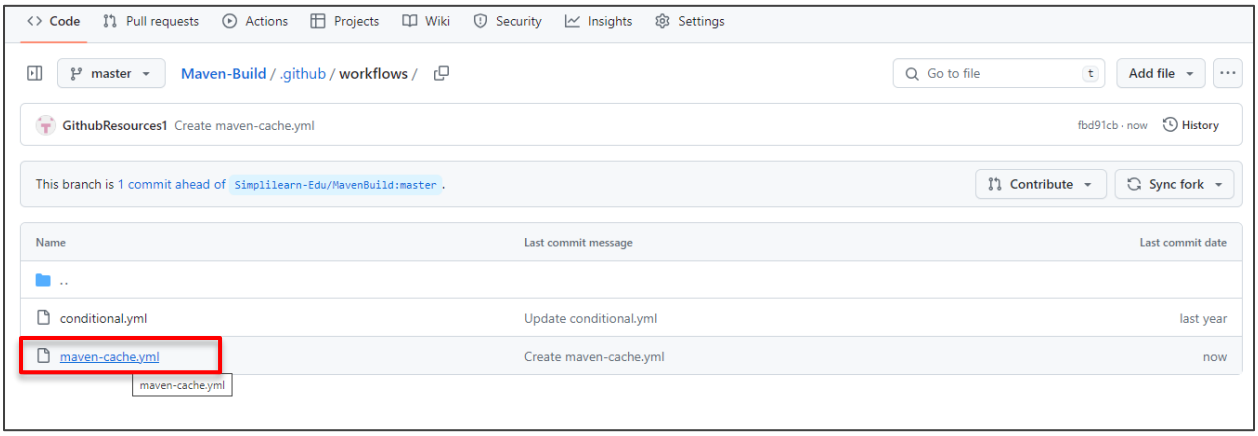


2.3 Enter a **Commit message** and then click on **Commit changes** to save the workflow file in the code repository

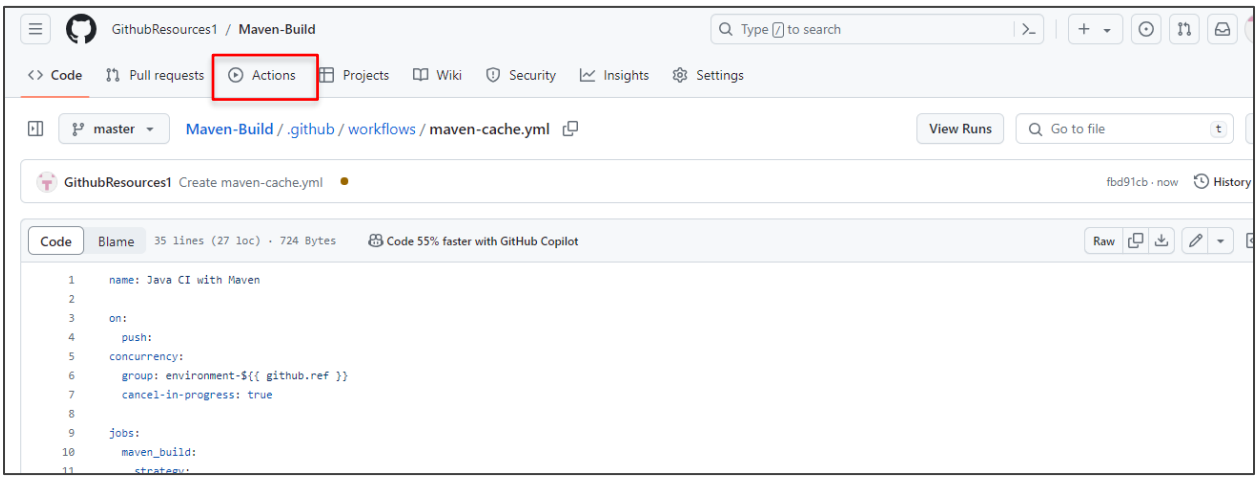


Step 3: Execute the GitHub Actions workflow

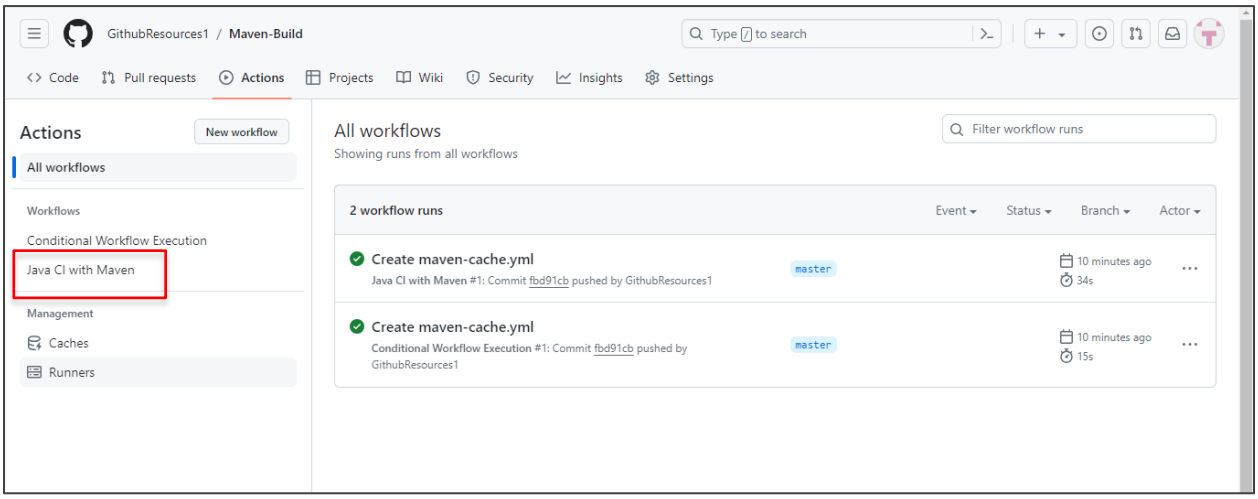
3.1 Click on the **maven-cache.yml** workflow file in the main repository page



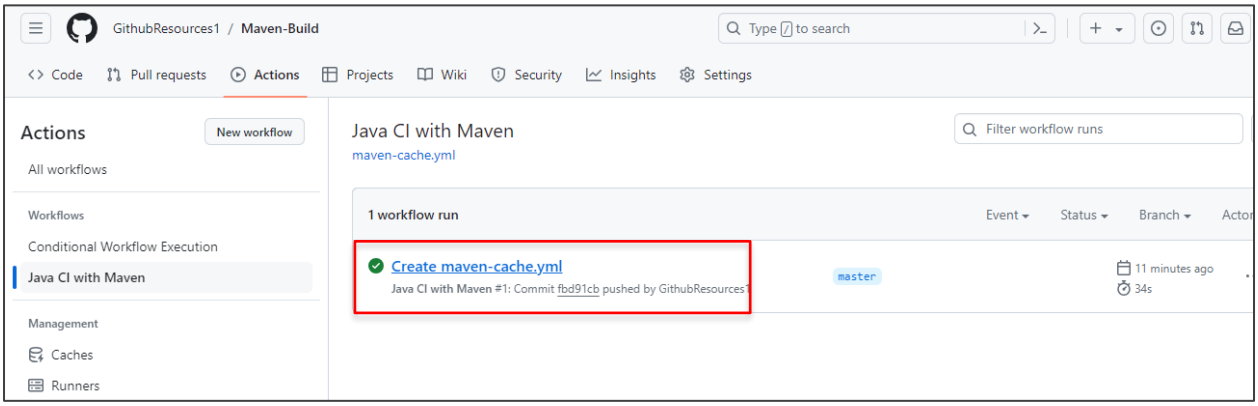
3.2 Navigate back to the **Actions** tab in the repository to access the workflow execution



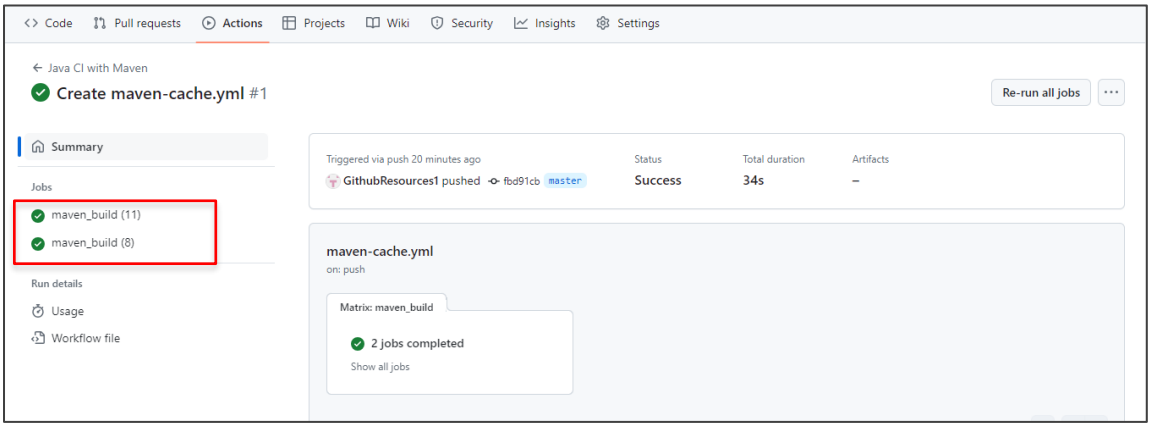
3.2 Select **Java CI with Maven** under **All workflows**

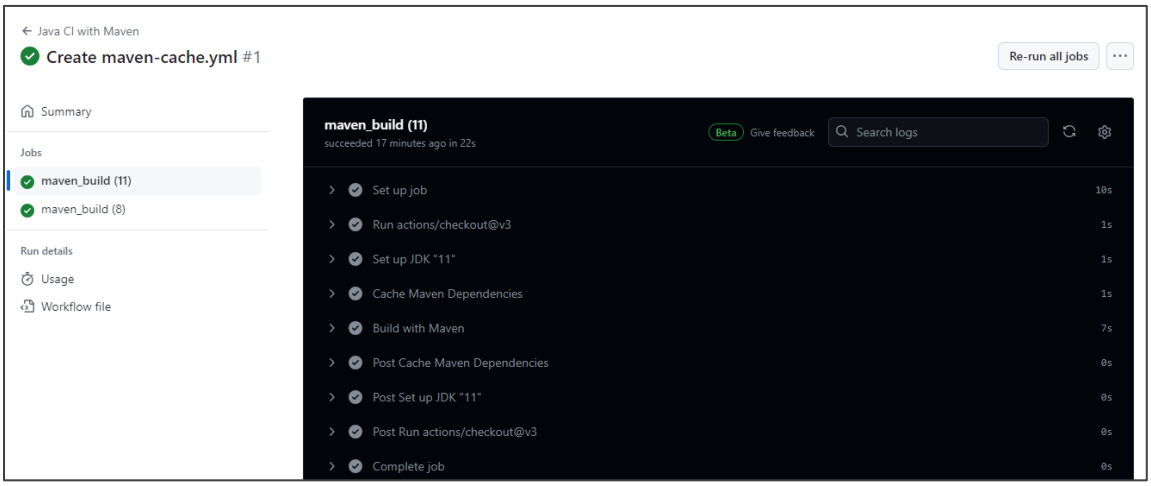


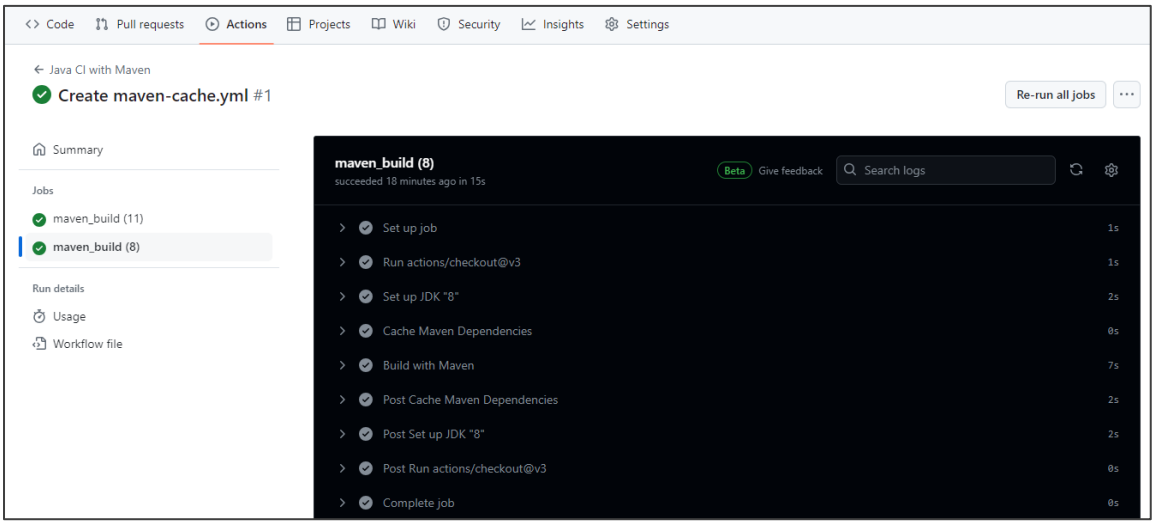
3.3 Then, click on **Create maven-cache.yml**



3.4 Under **Jobs**, select **maven_build (11)** and **maven_build (8)** to view all the job logs







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.