

**NAME:** K. Manoj Kumar

**REGISTER NUMBER:** 23MIS0159

**COURSE NAME:** Agile Development Process and DevOps Lab

**COURSE CODE:** ISWE406P

**SLOT:** L51+L52

**TASK 1: Jenkins Familiarization Objective:**

**Understand Jenkins UI and basic navigation**

**Tasks:**

**1. Open Jenkins Dashboard in browser**

**2. Identify:**

**o Dashboard**

**o Manage Jenkins**

**o New Item**

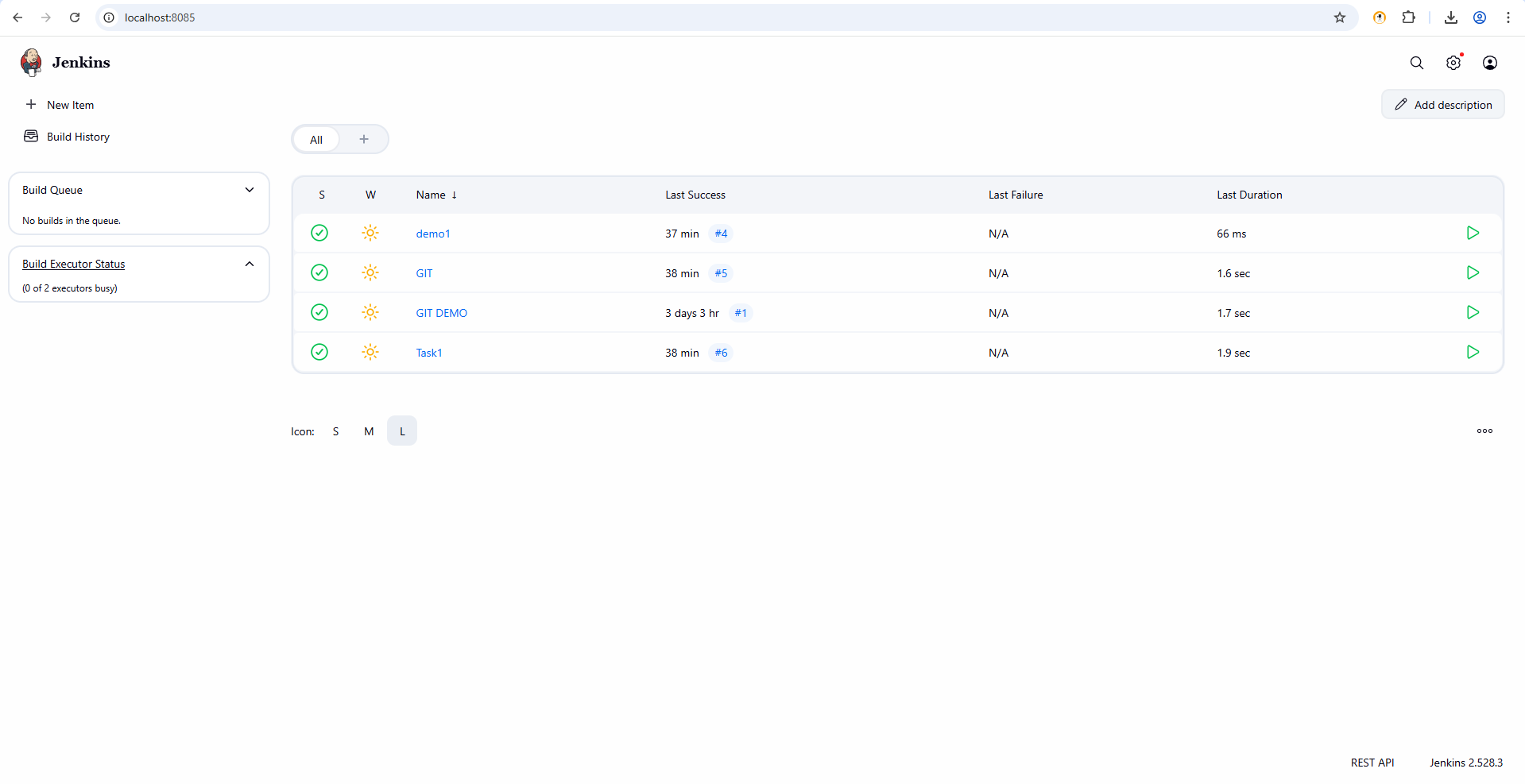
**o Build History**

**3. Check Jenkins version**

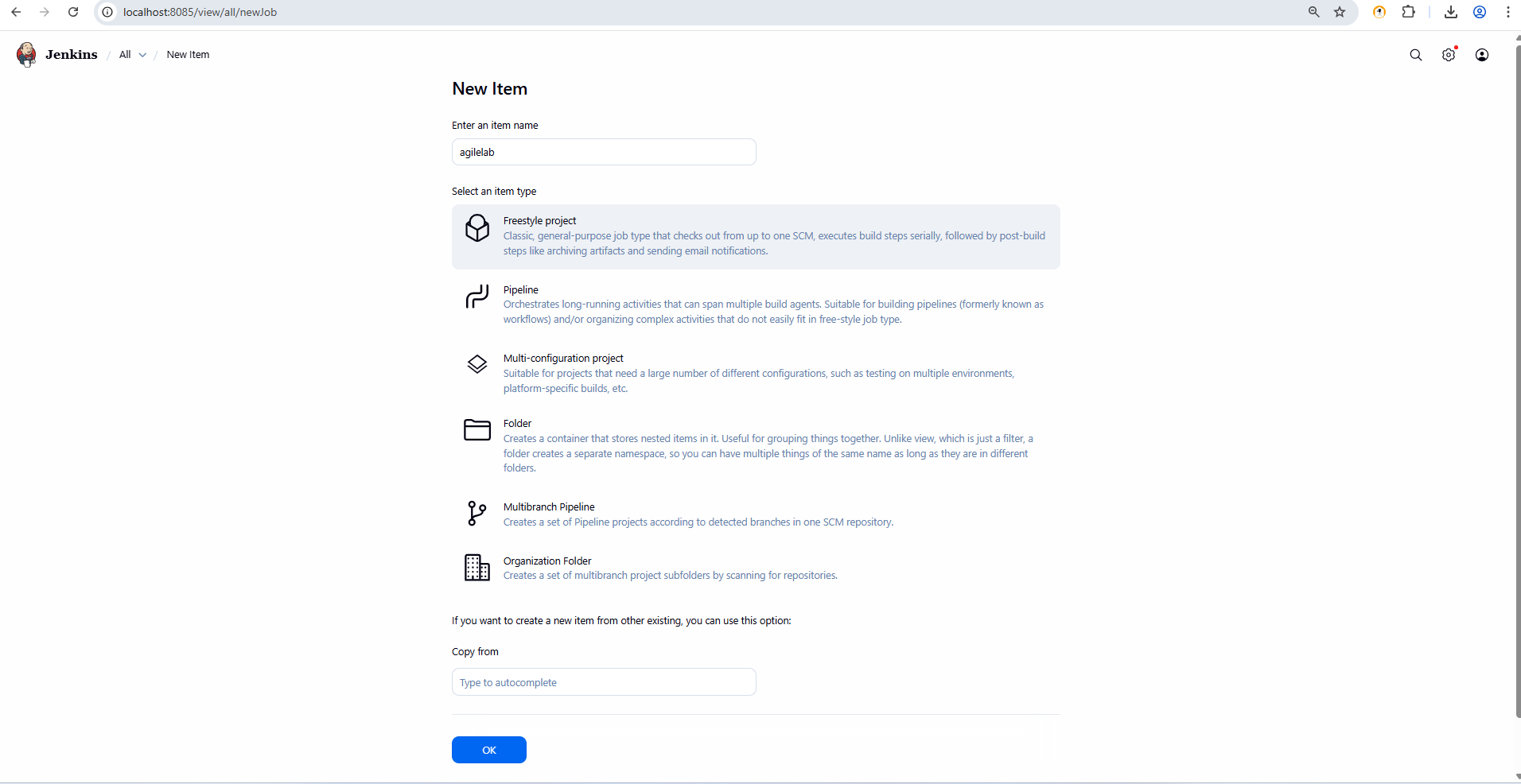
**Expected Output:**

**Screenshot or note of Jenkins version**

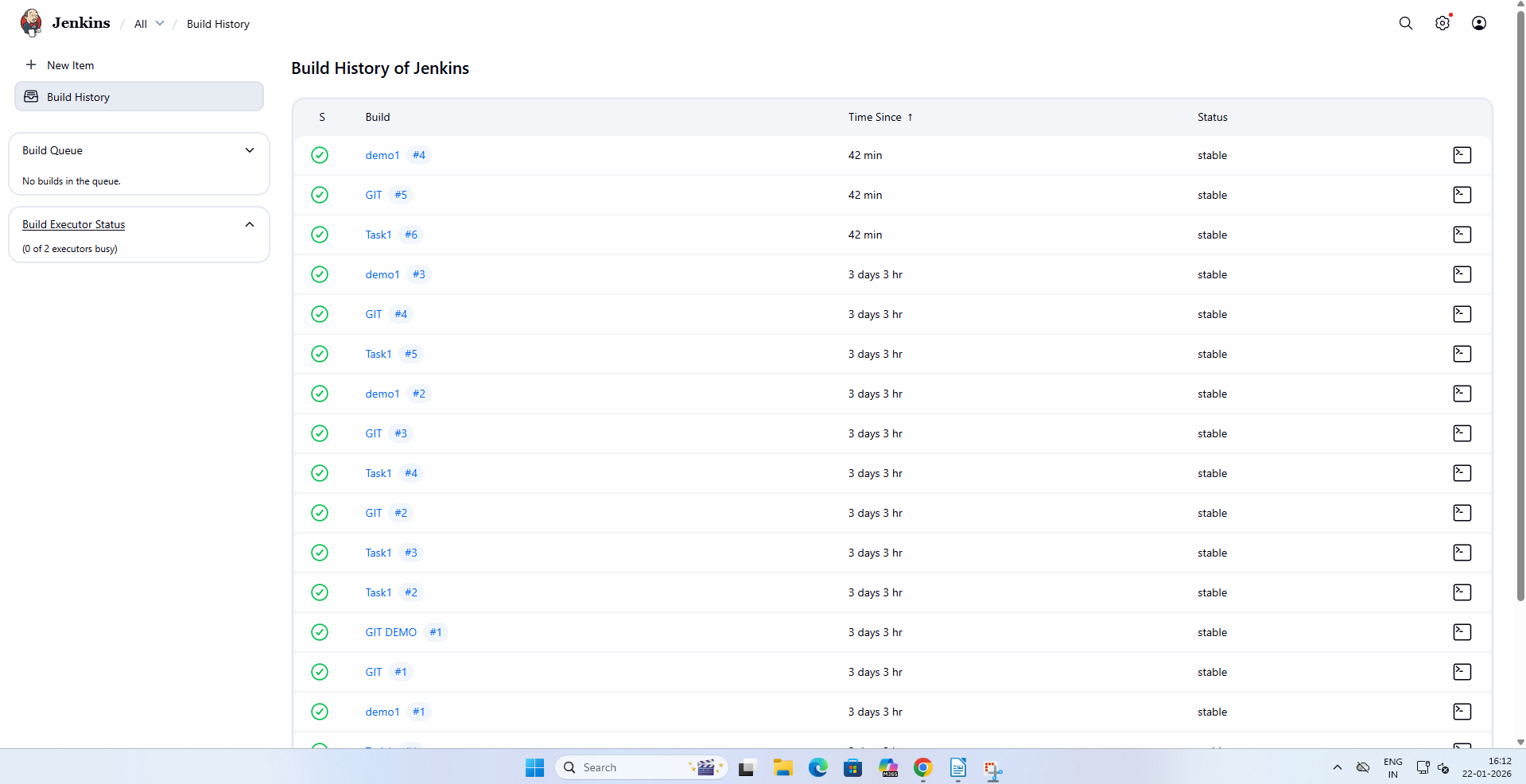
**DASHBOARD, Jenkins Version:**



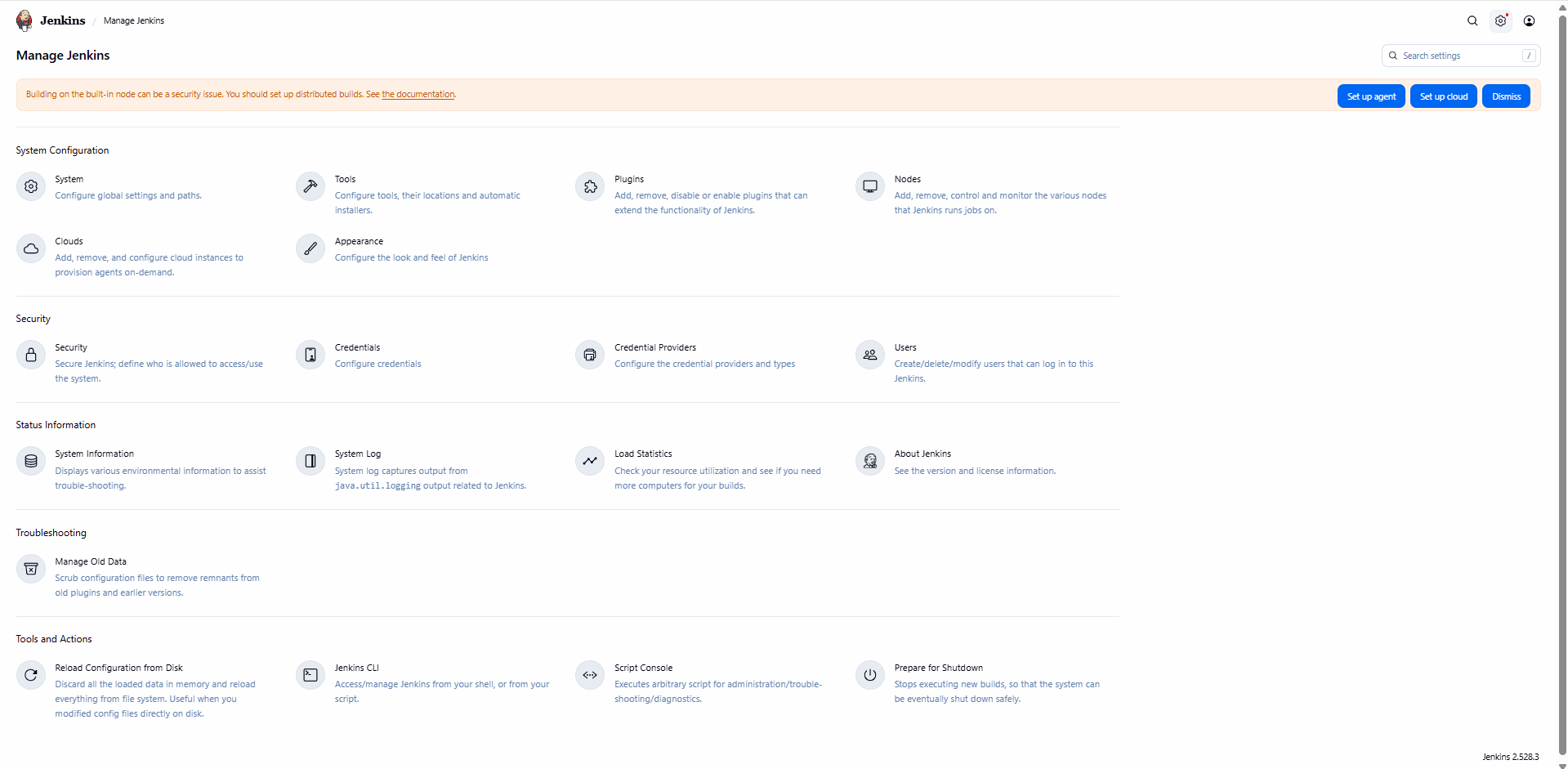
**NEWITEMS:**



**Build History:**



**MANAGE JENKINS:**



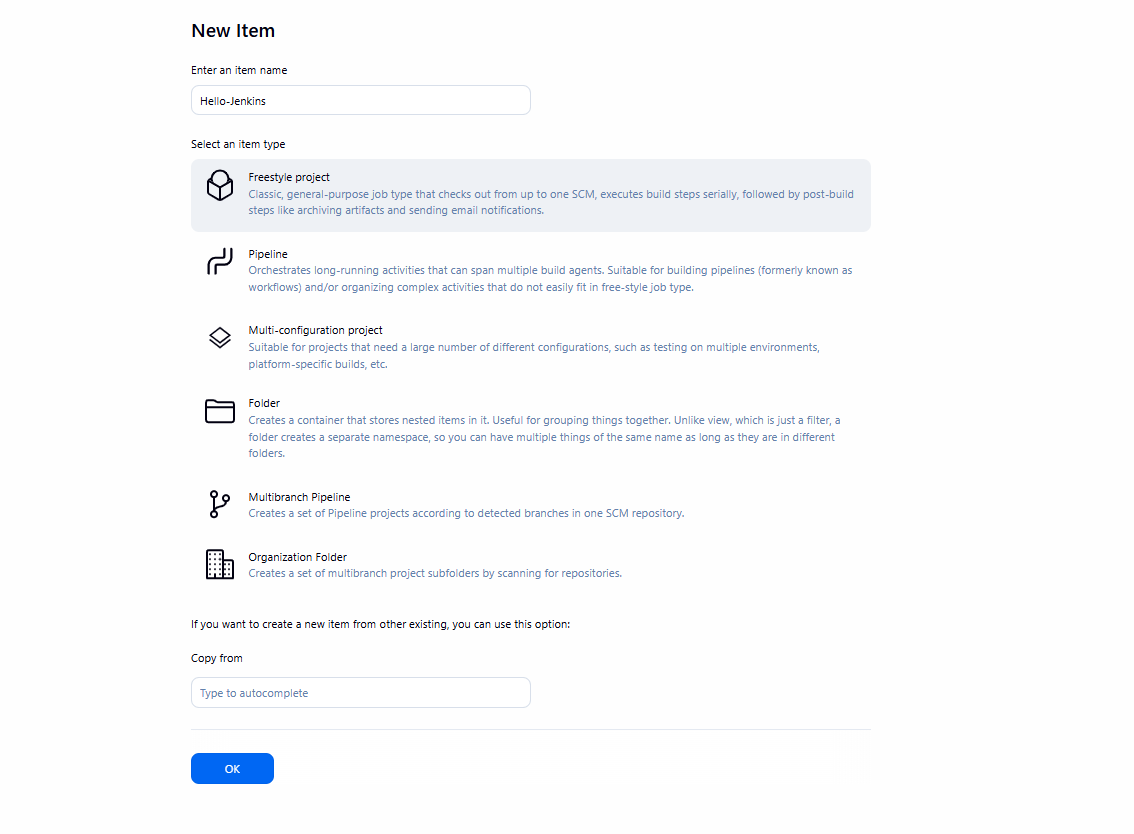
**TASK 2: Create First Freestyle Job**

**Objective:**

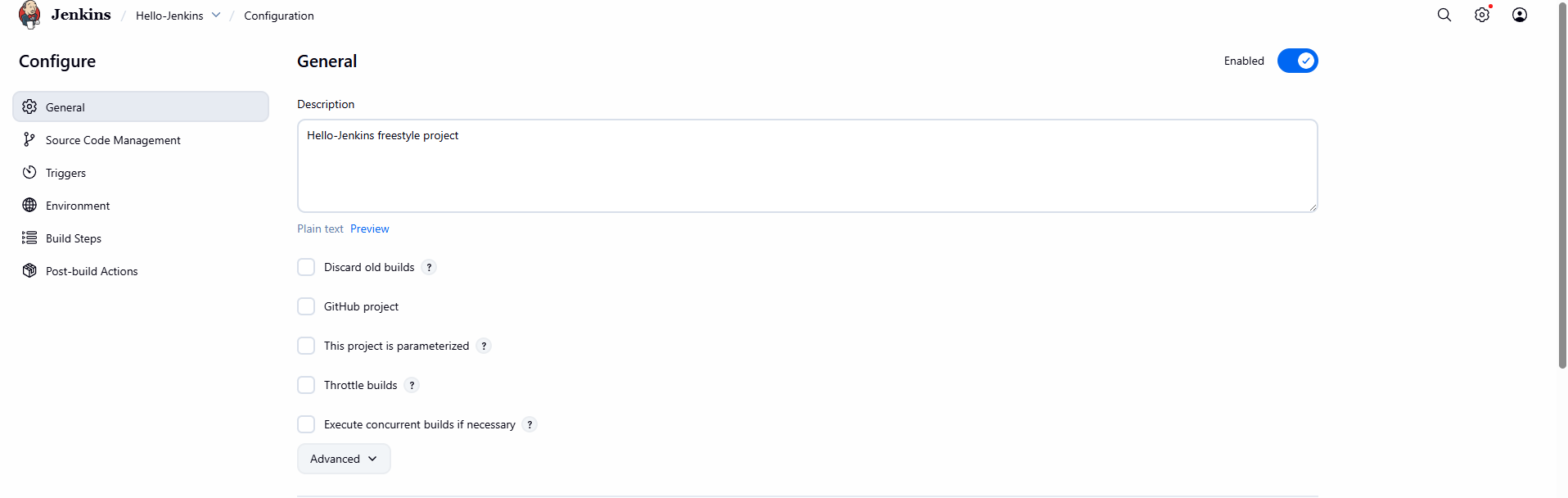
**Create and run a Jenkins job**

**Tasks:**

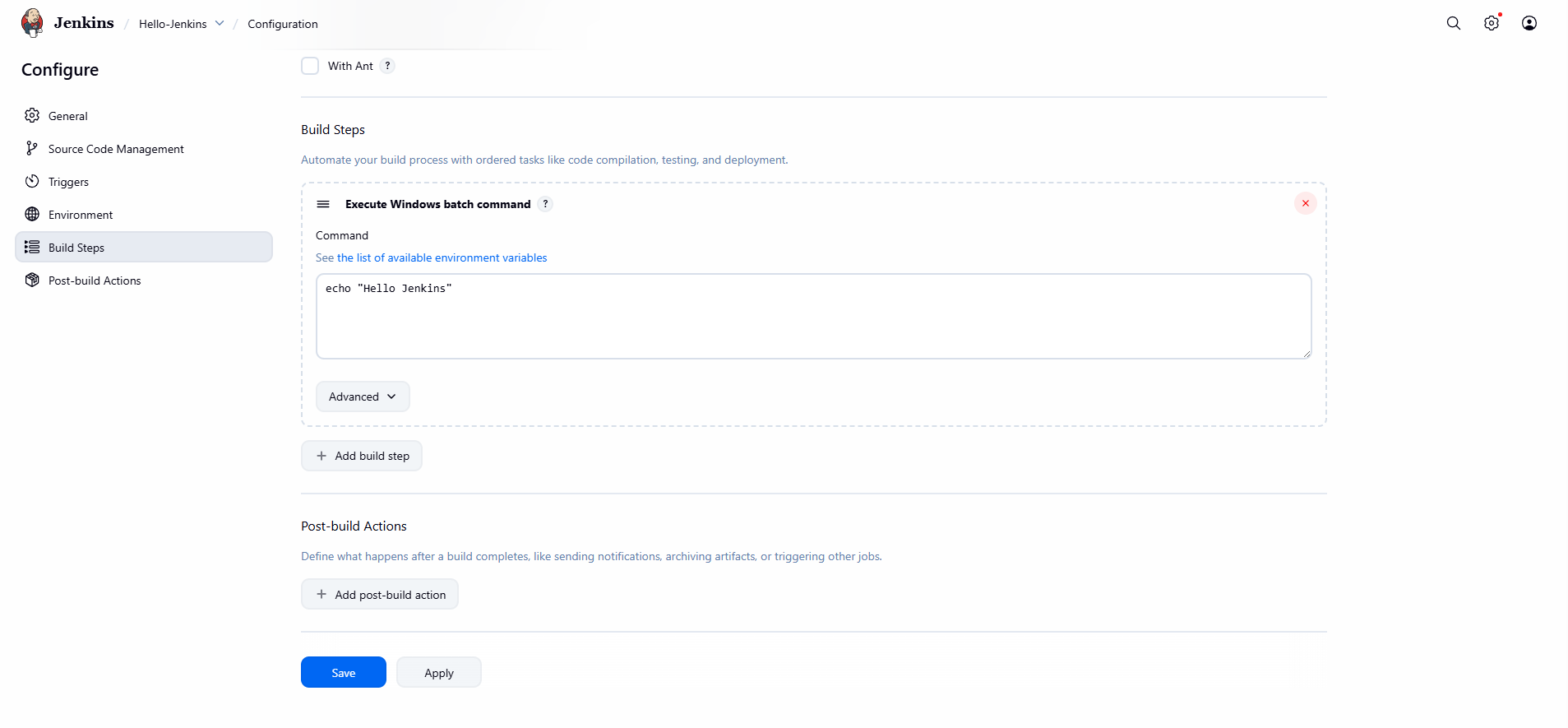
1. **Create a Freestyle project named Hello-Jenkins**



**2. Add a description**



**3. Add build step: o Execute shell / Windows batch command o Print "Hello Jenkins"**



**4. Build the job manually**

**Expected Output: Console output showing message**



**TASK 3: Jenkins Workspace & Commands**

**Objective: Understand workspace usage**

**Tasks:**

1. **Navigate to job workspace**



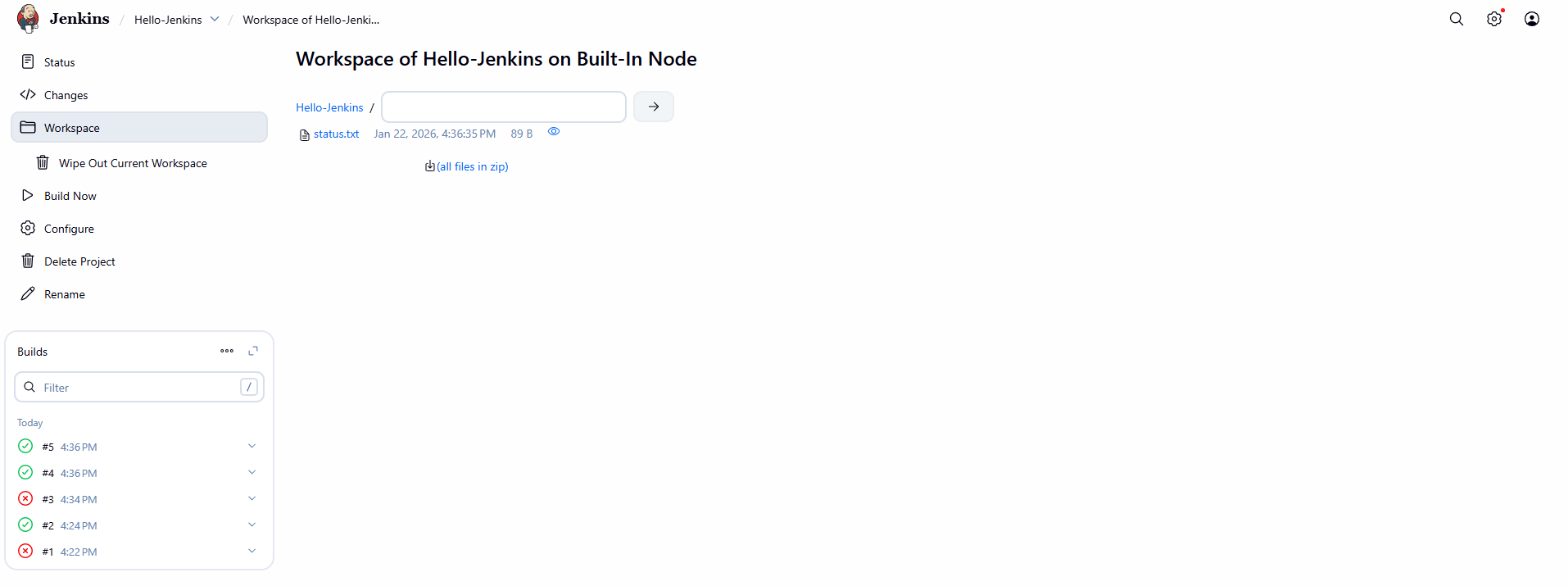
**2. Create a text file using build step**



**3. Display file contents in console**



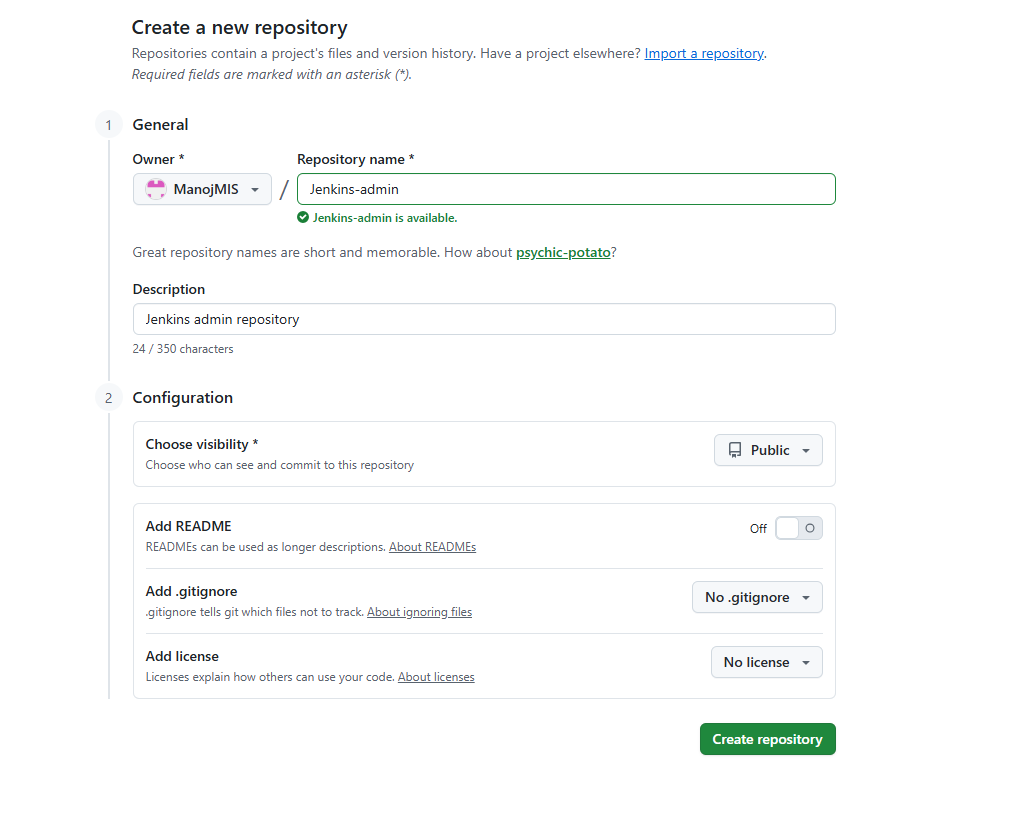
**Expected Output: File created inside workspace**

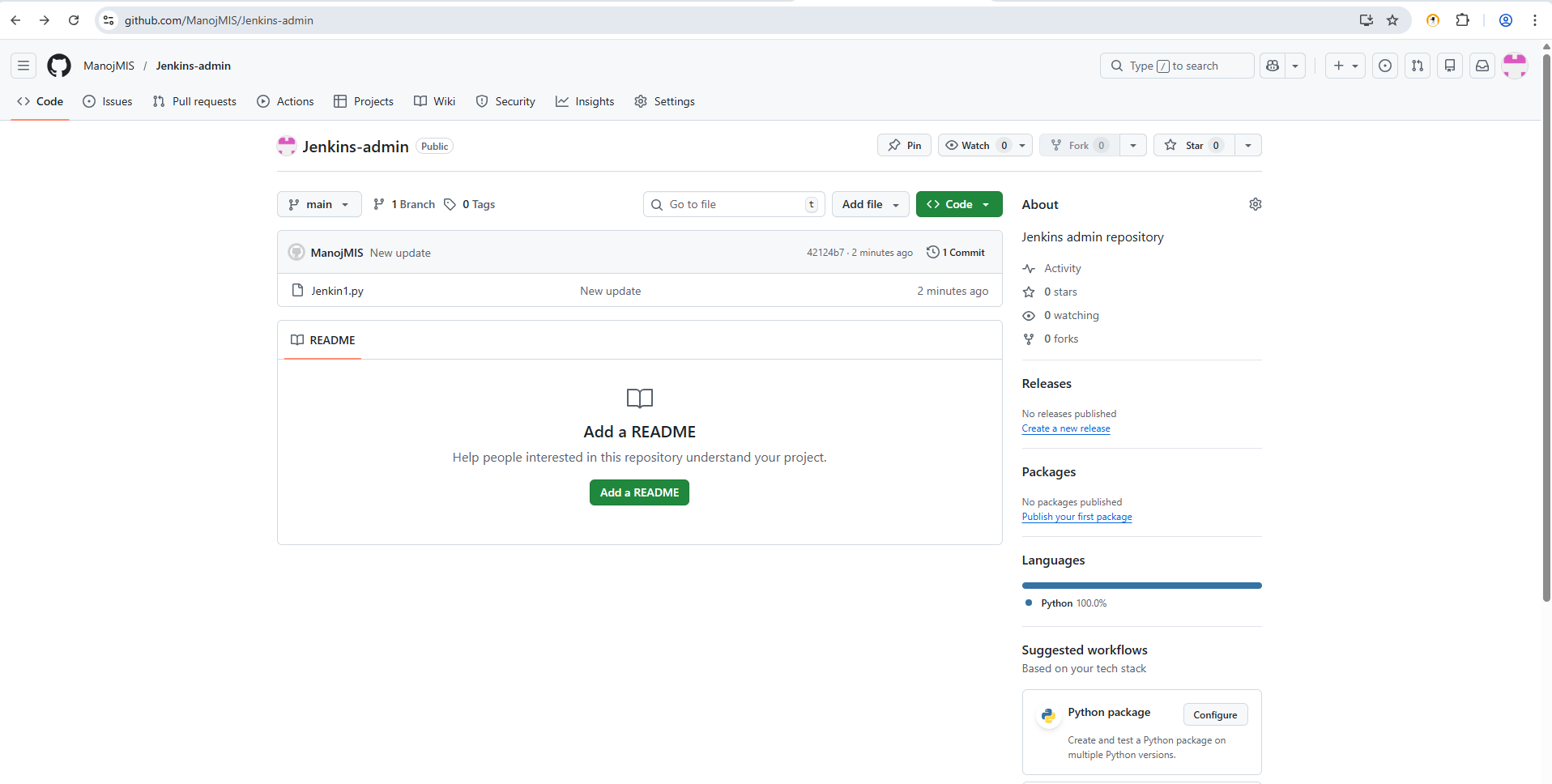


**TASK 4: Git Integration**

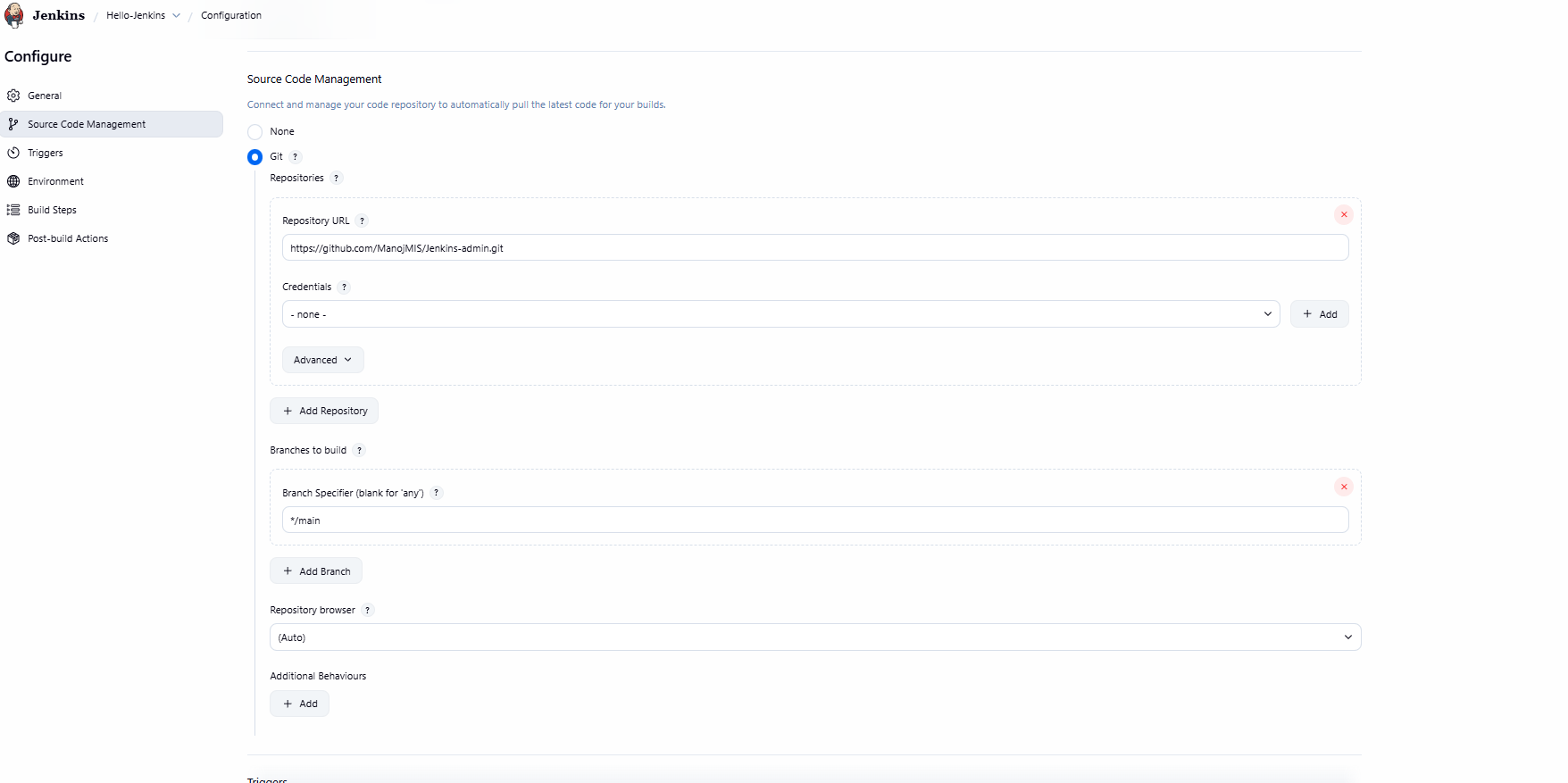
**Objective: Integrate Jenkins with GitHub**

**Tasks: 1. Create a GitHub repository with sample code**

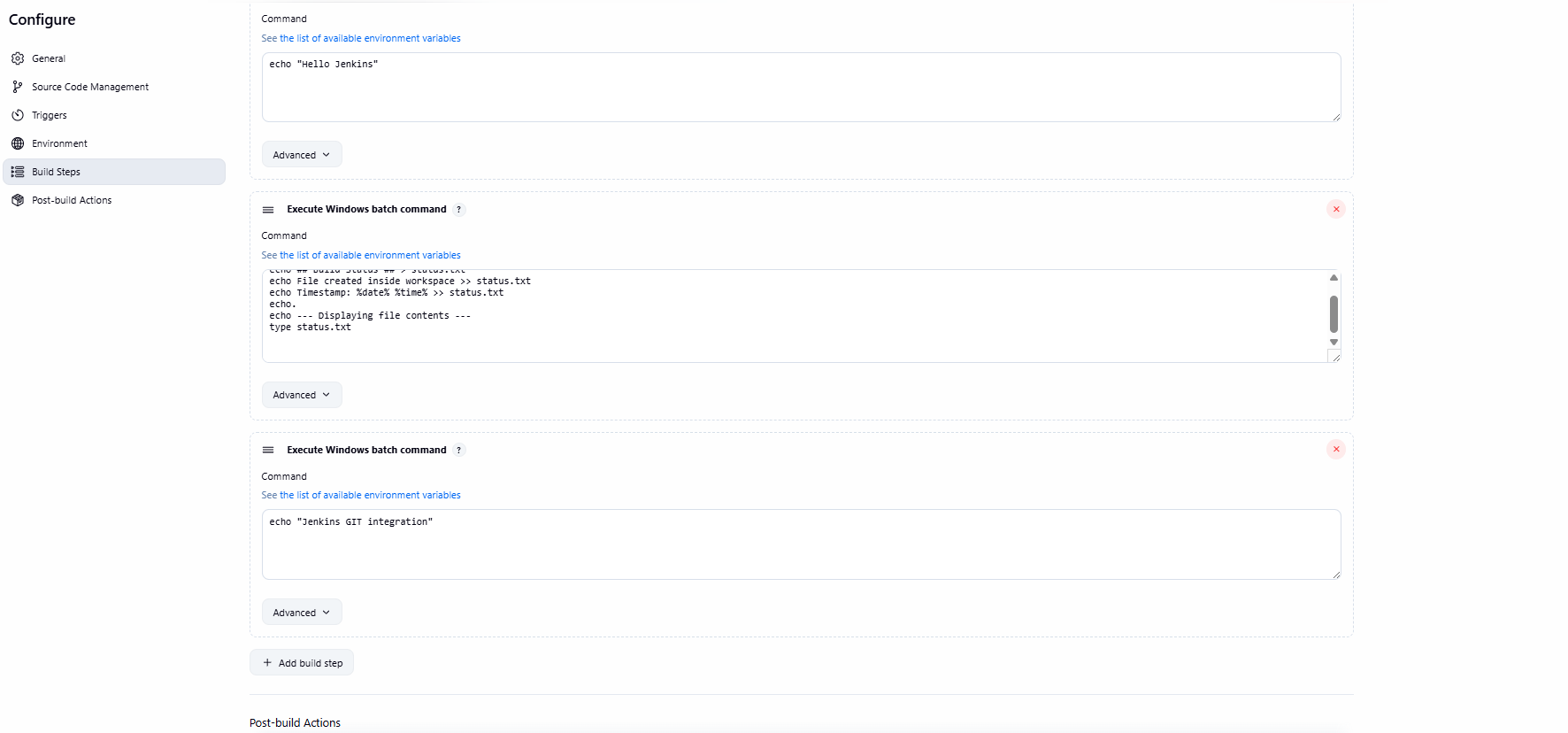




**2. Configure Git in Jenkins**

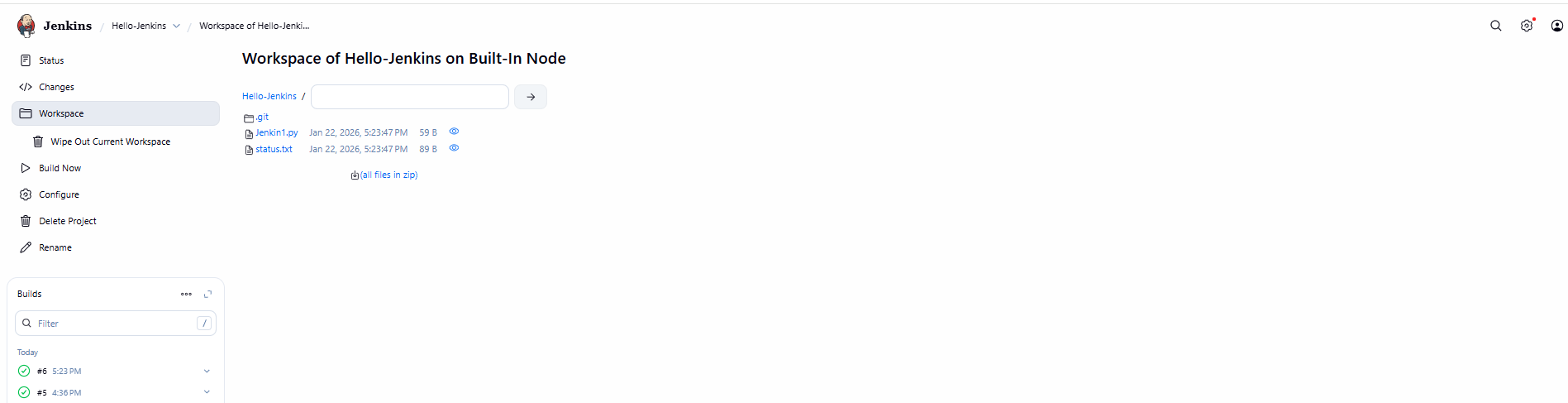


**3. Add Git repository URL in job**



**4. Build and verify code checkout**

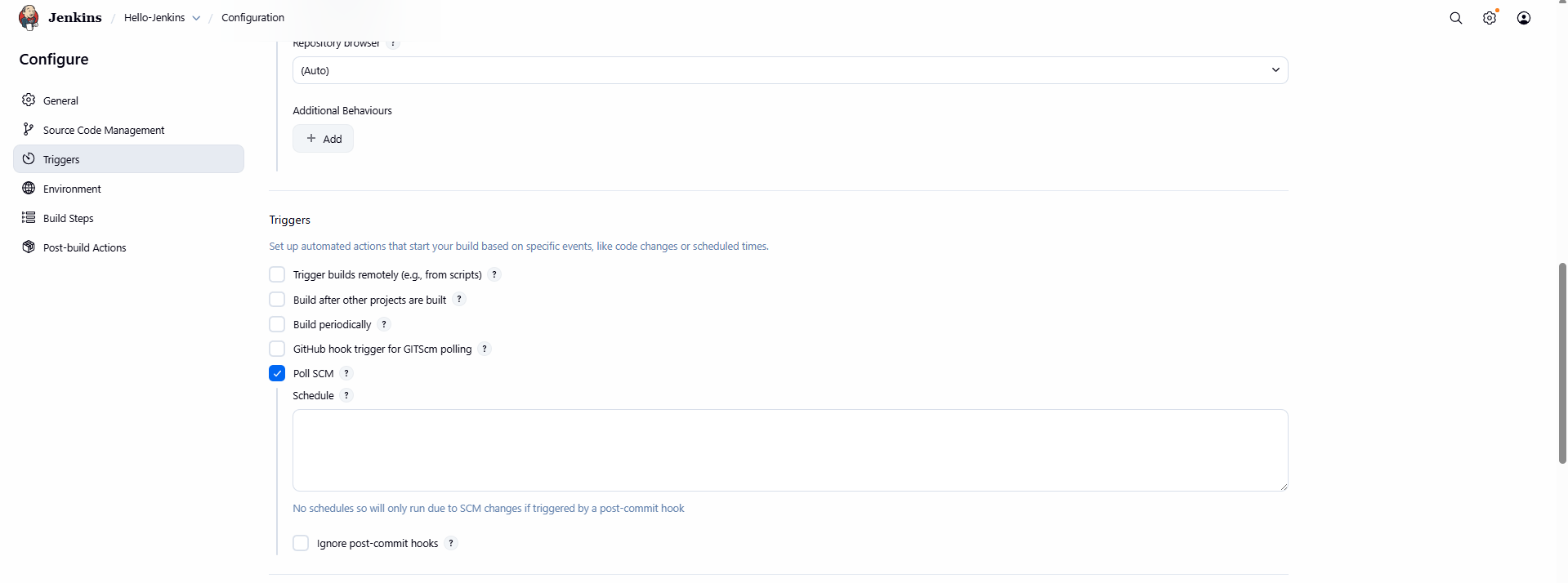
**Expected Output: Source code visible in workspace**

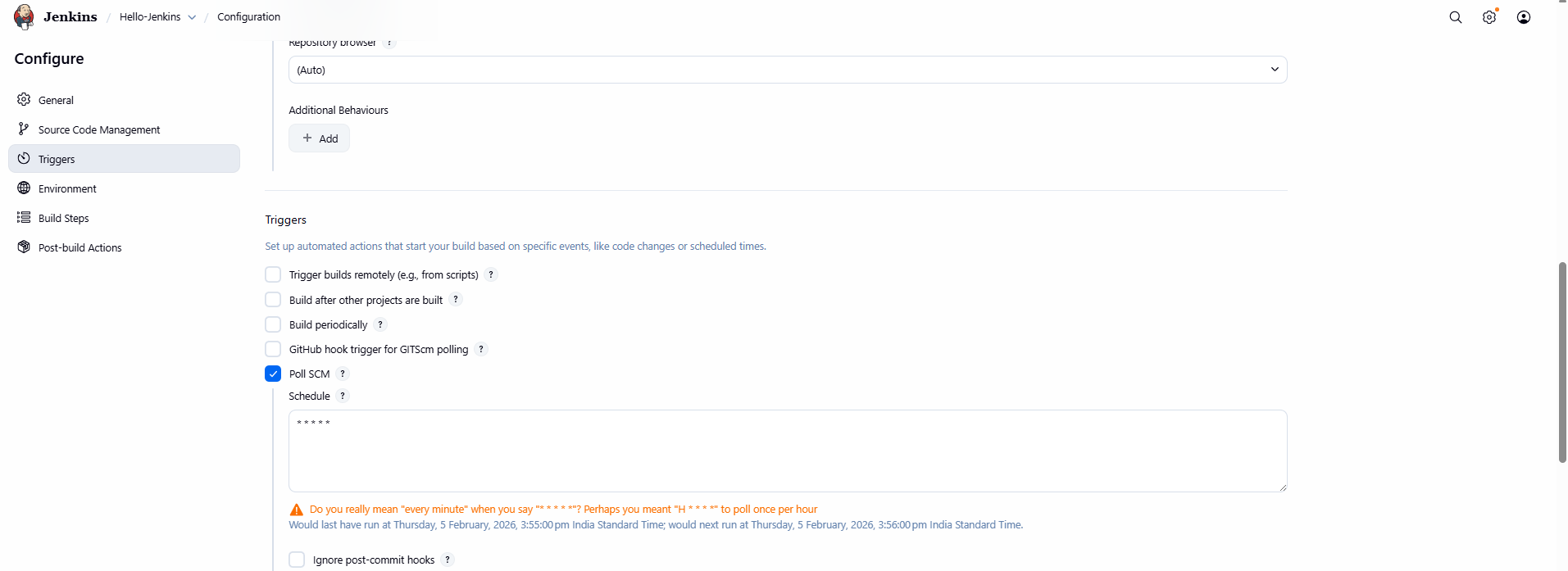


**TASK 5: Poll SCM Trigger**

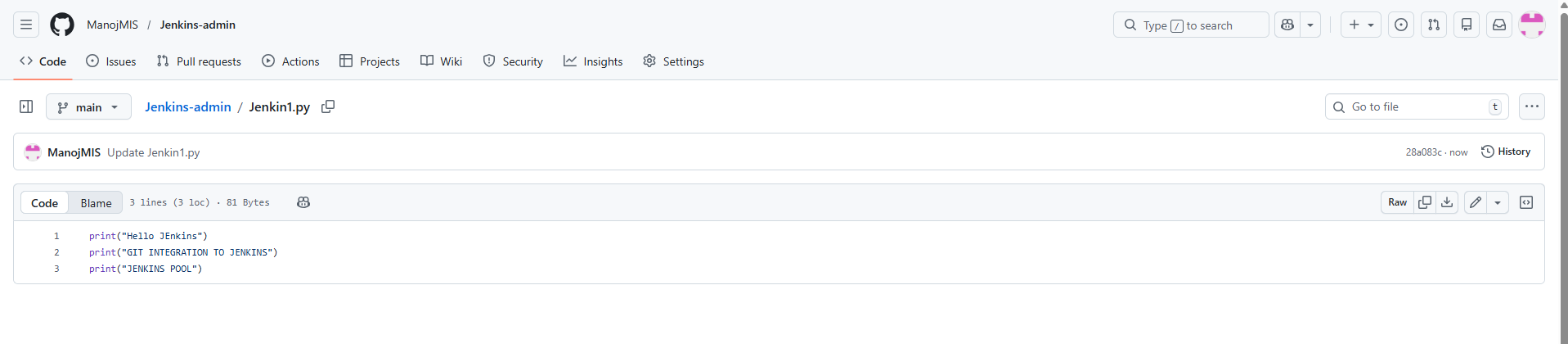
**Objective: Automatically trigger builds on code change**

**Tasks: 1. Enable Poll SCM**

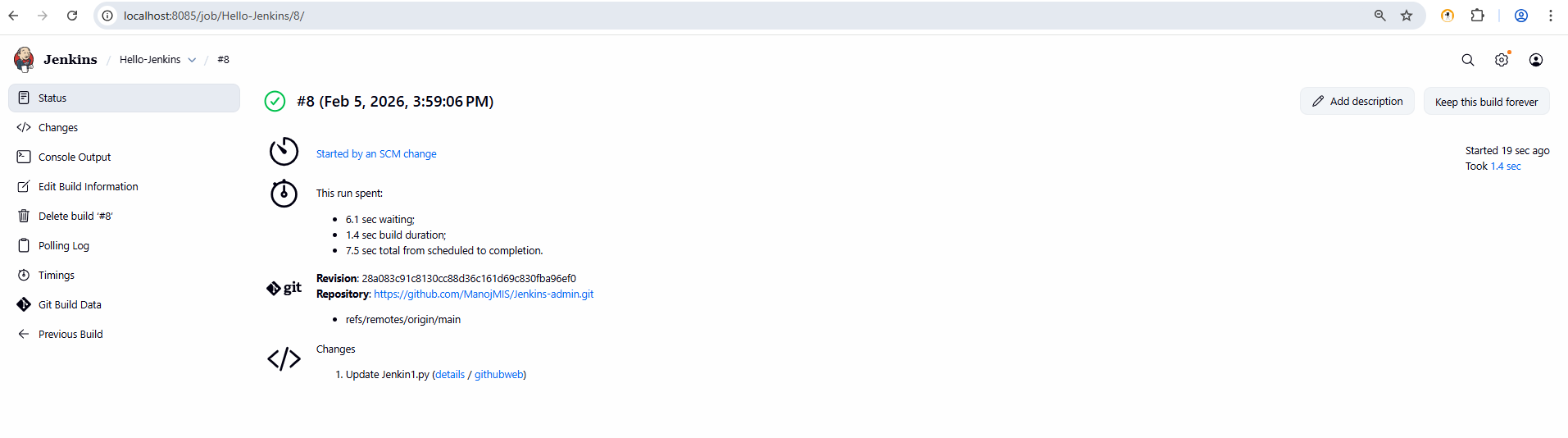
**2. Set schedule: \* \* \* \* \***



**3. Modify GitHub file and commit**



**4. Observe automatic build Expected Output:  Build triggered without manual action**

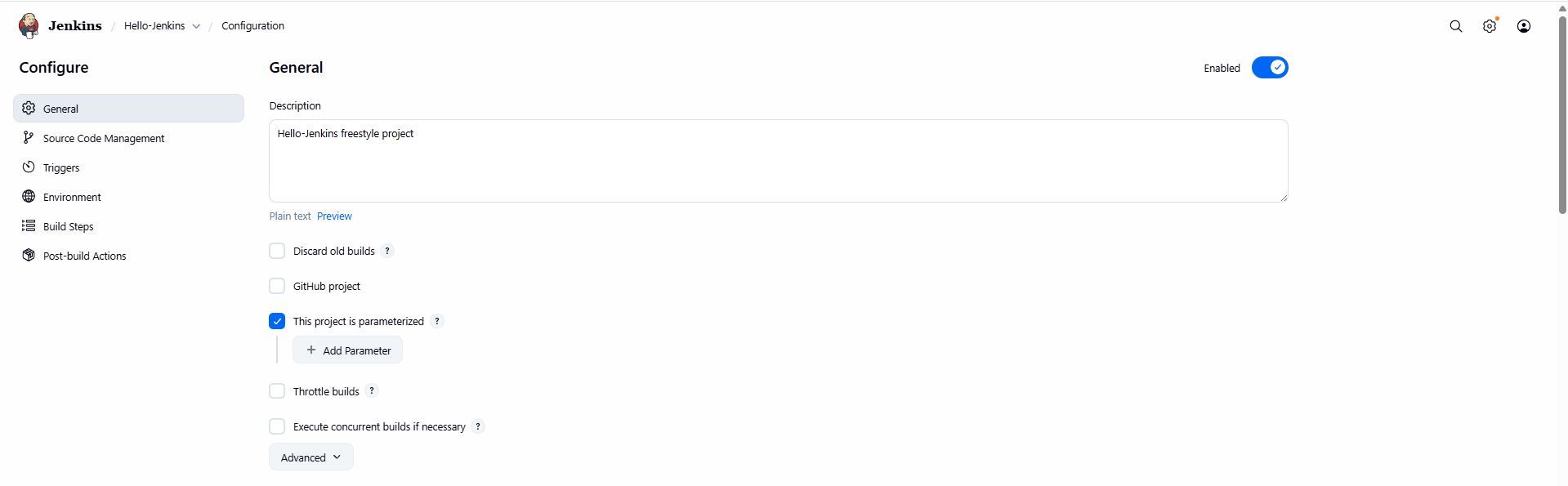




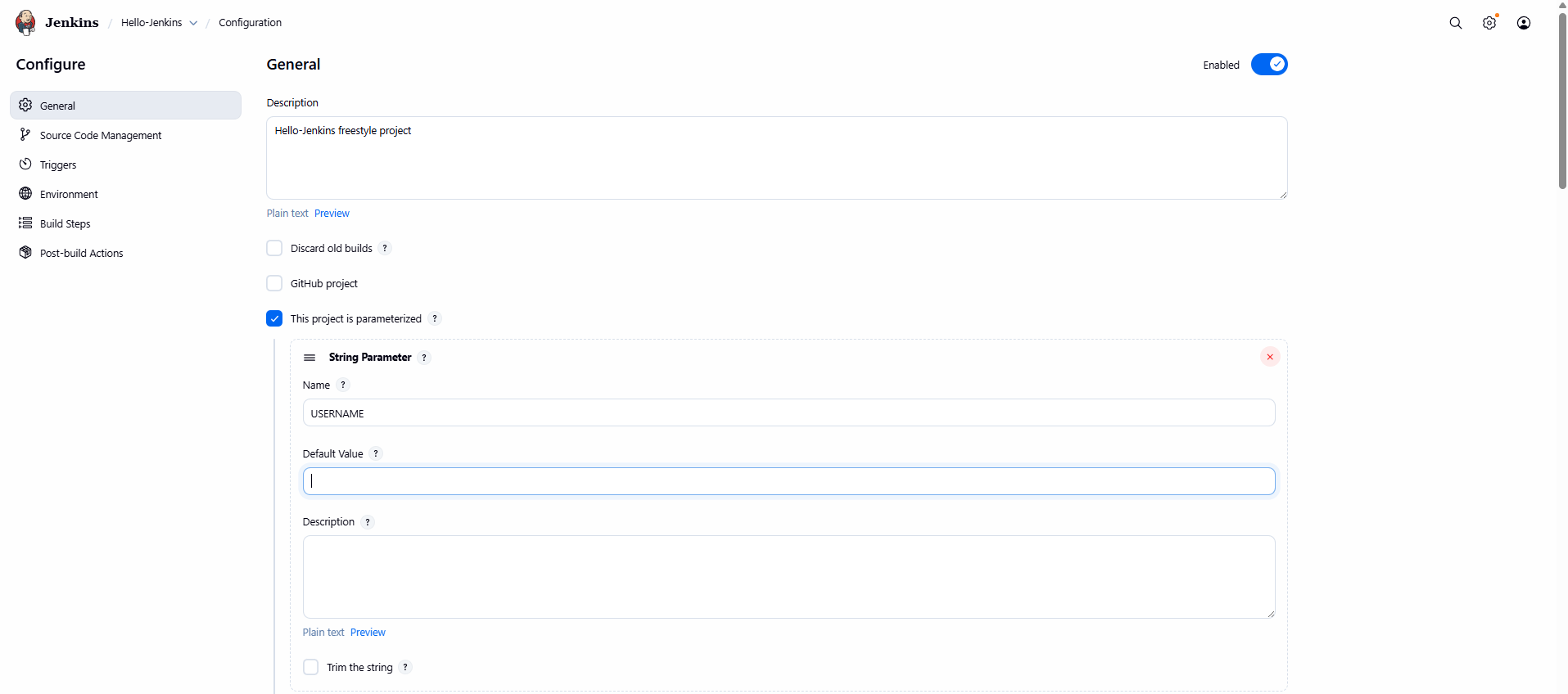
**TASK 6: Parameterized Build**

**Objective: Use parameters in Jenkins job**

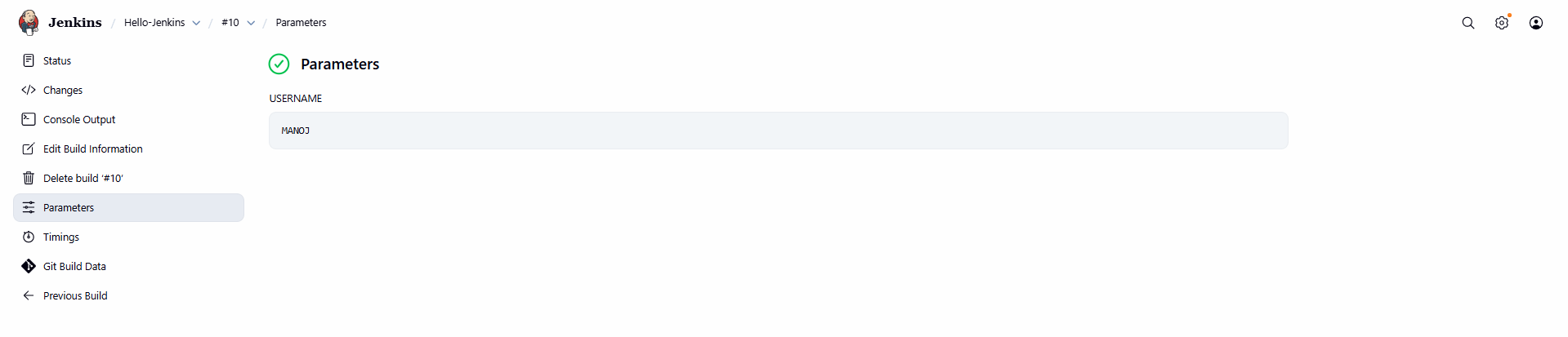
**Tasks: 1. Enable parameterized build**



**2. Add String parameter USERNAME**



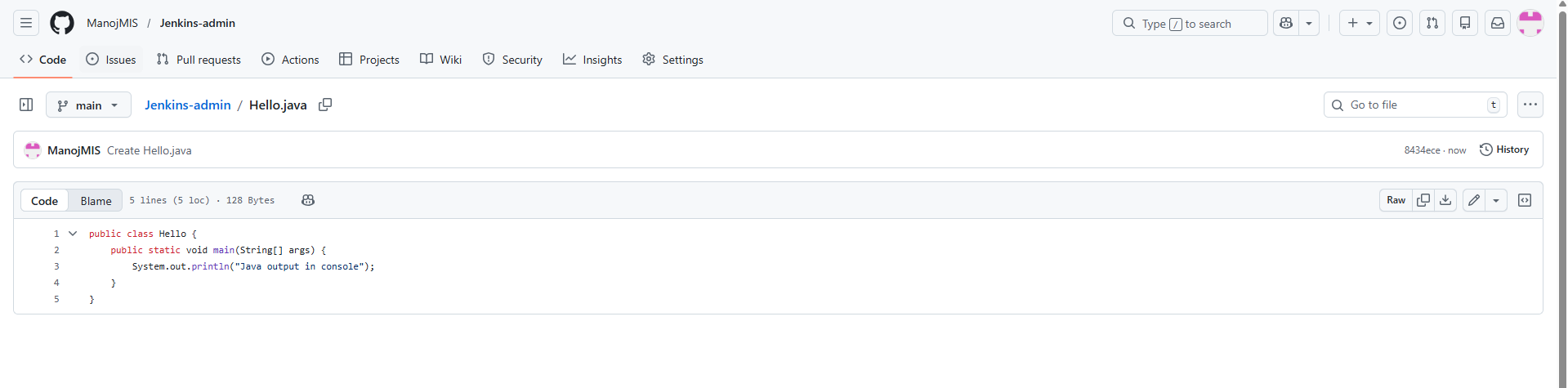
**3. Print parameter value in build step Expected Output:  Console output showing parameter value**

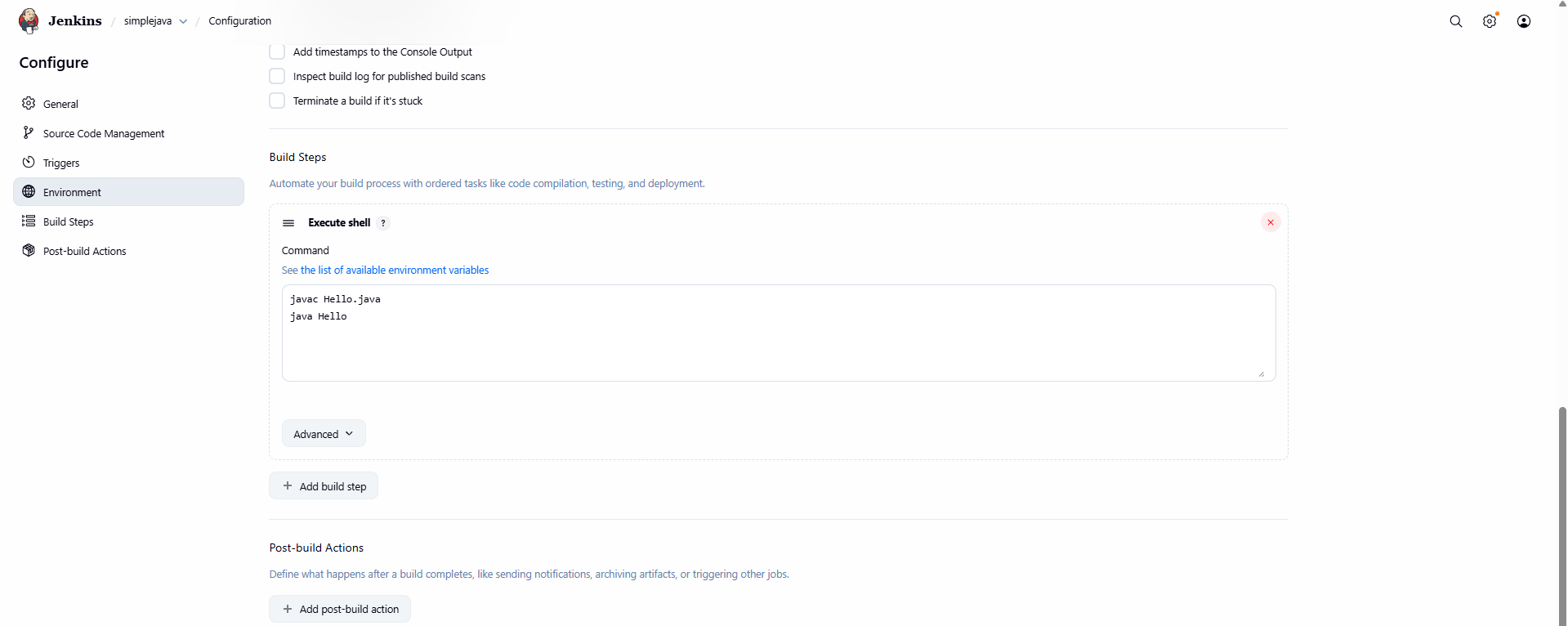


**TASK 7: Java Build Using Jenkins**

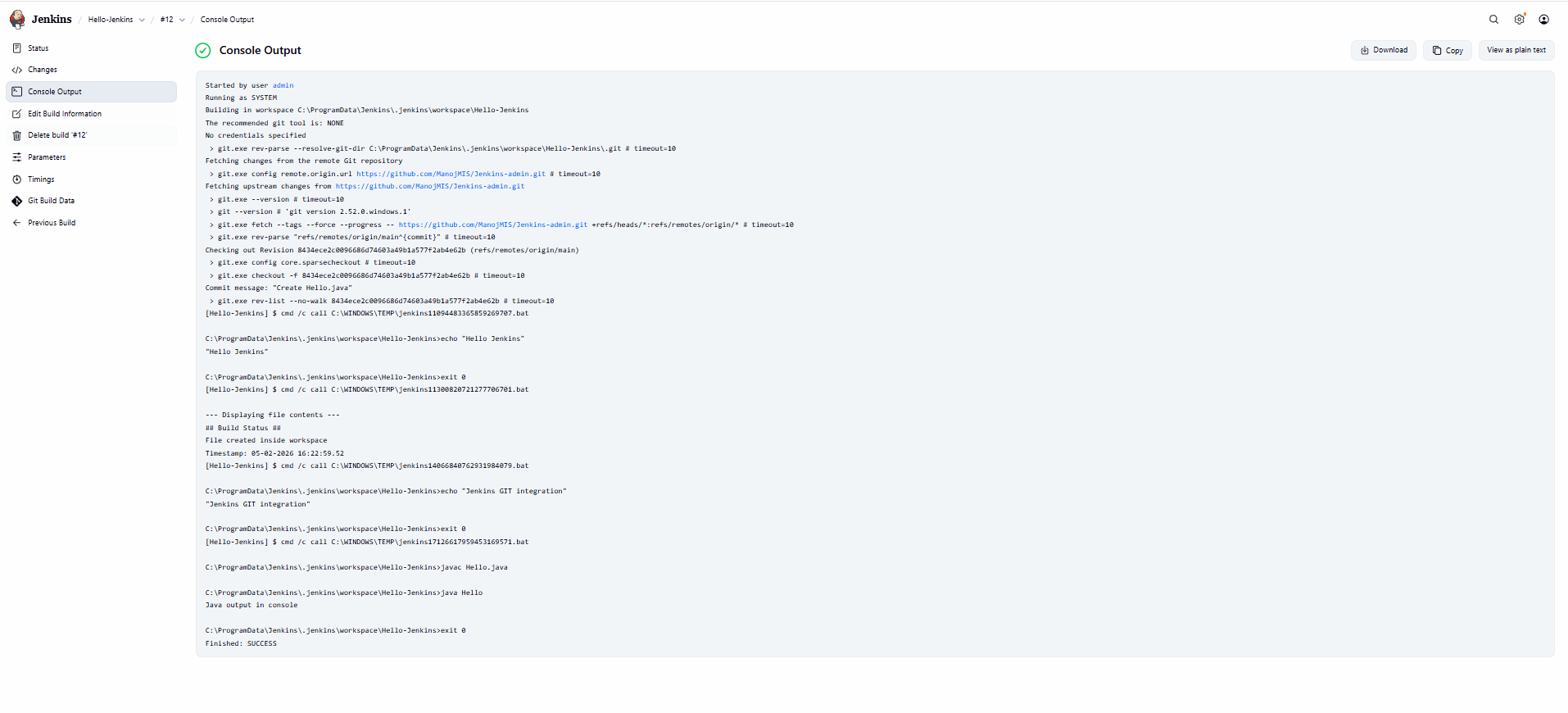
**Objective: Compile Java program using Jenkins**

**Tasks: 1. Create simple Hello.java**

**2. Compile using javac**



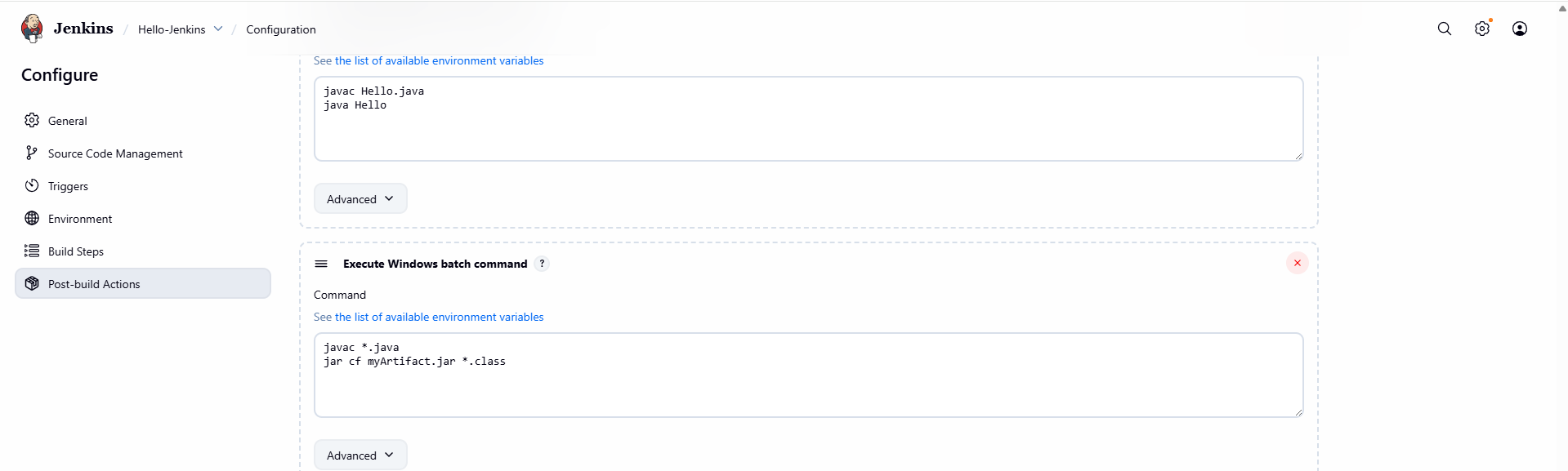
**3. Run Java program Expected Output:  Java output in console**



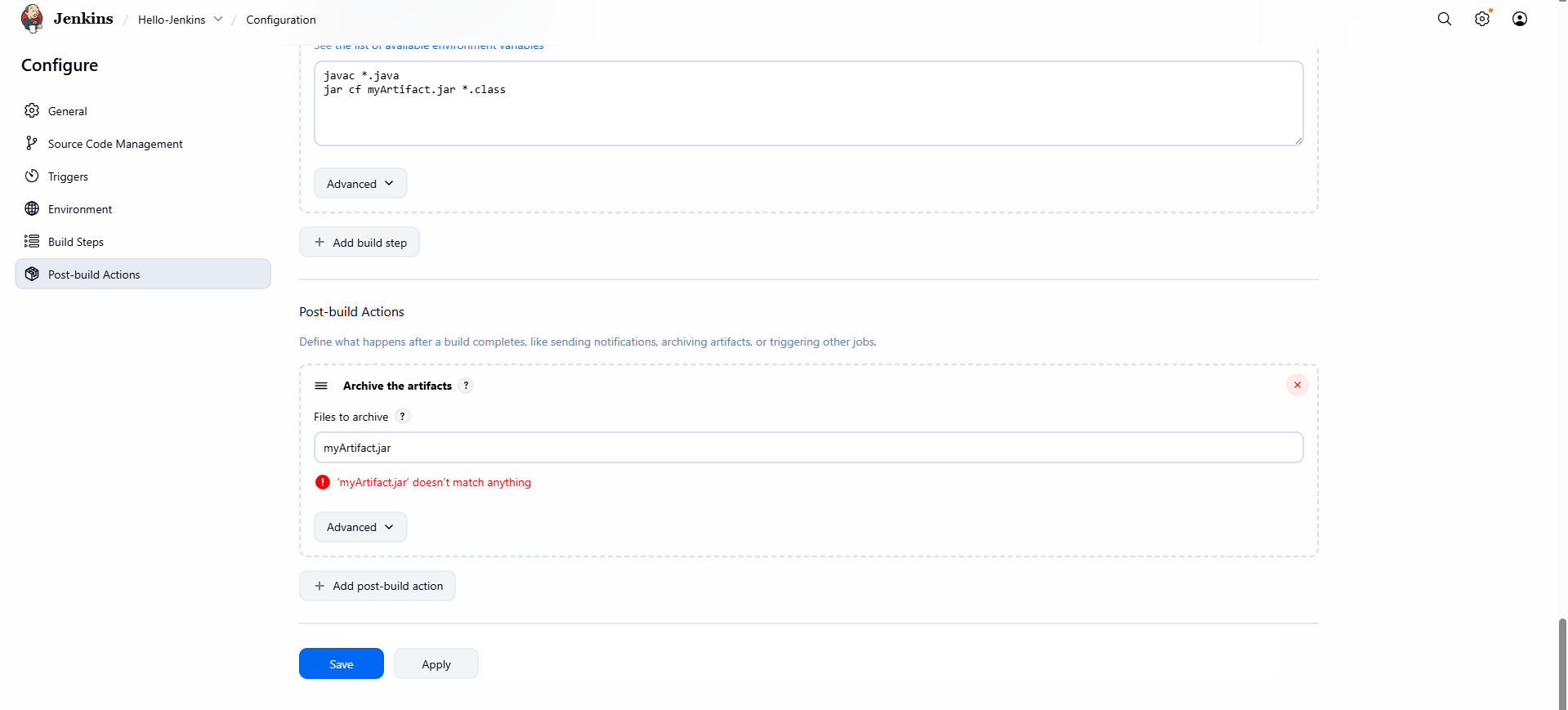
**TASK 8: Archive Artifacts**

**Objective: Store build outputs**

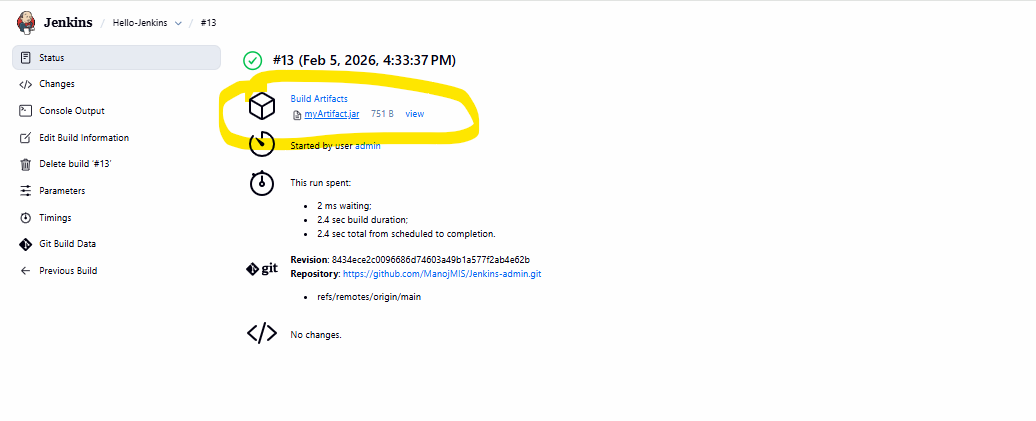
**Tasks: 1. Generate .class or .jar file**



**2. Archive artifacts in post-build action**



**3. Download artifact from Jenkins UI Expected Output:  Artifact available for download**



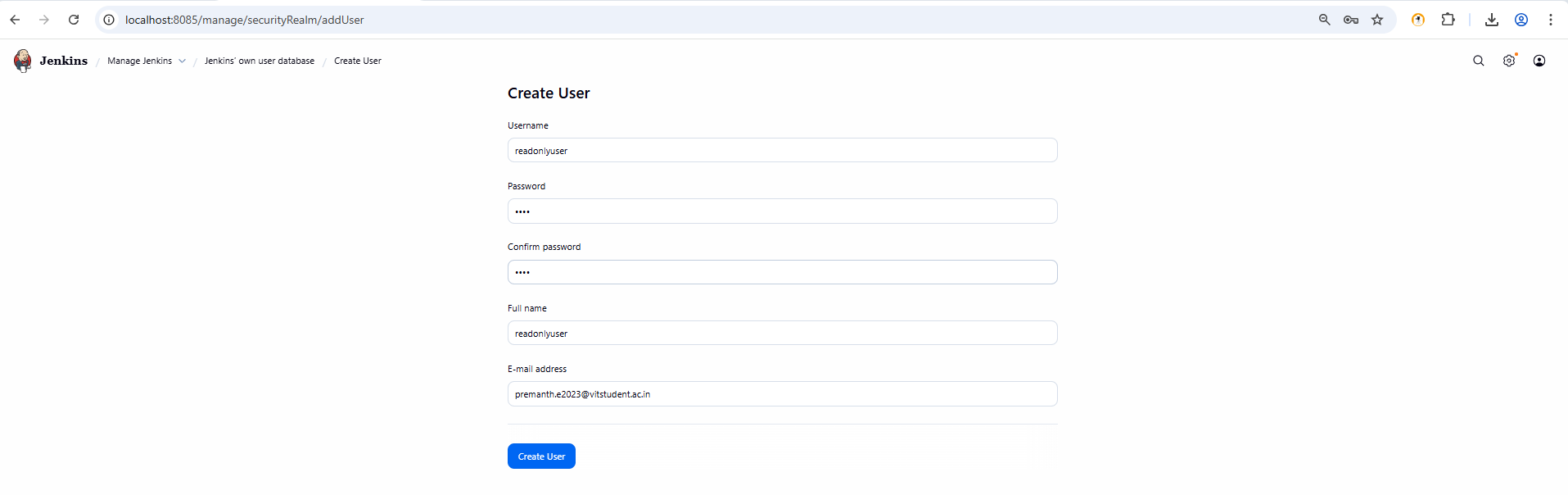
**TASK 9: Users & Roles**

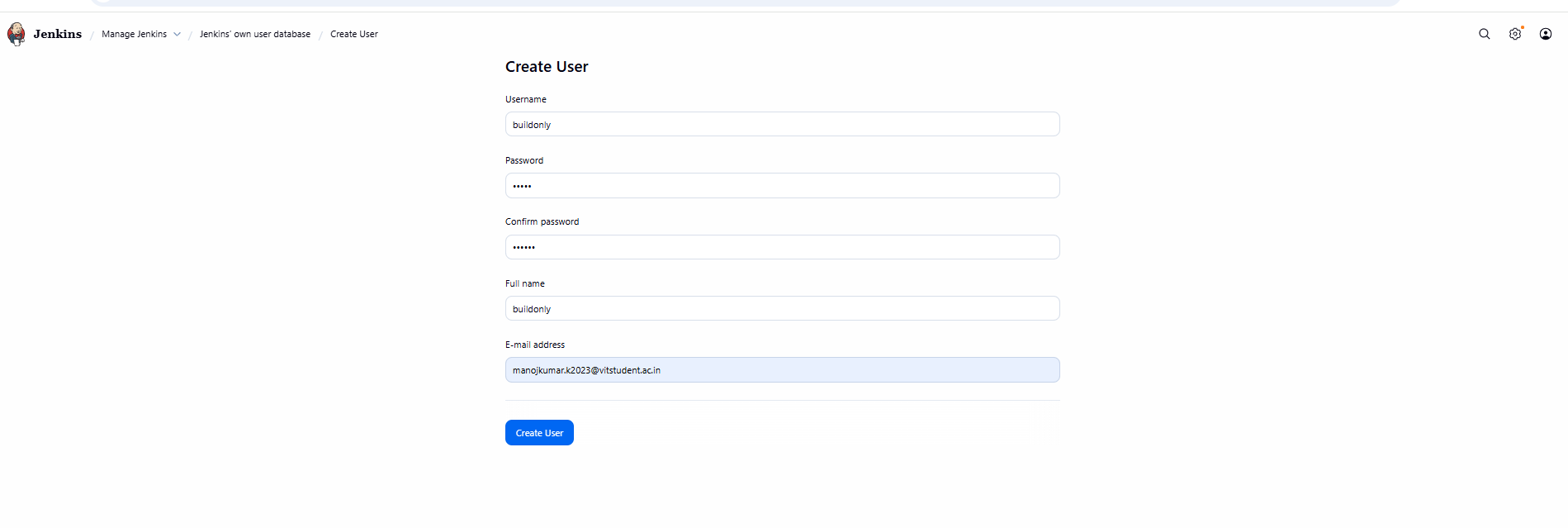
**Objective: Manage Jenkins users**

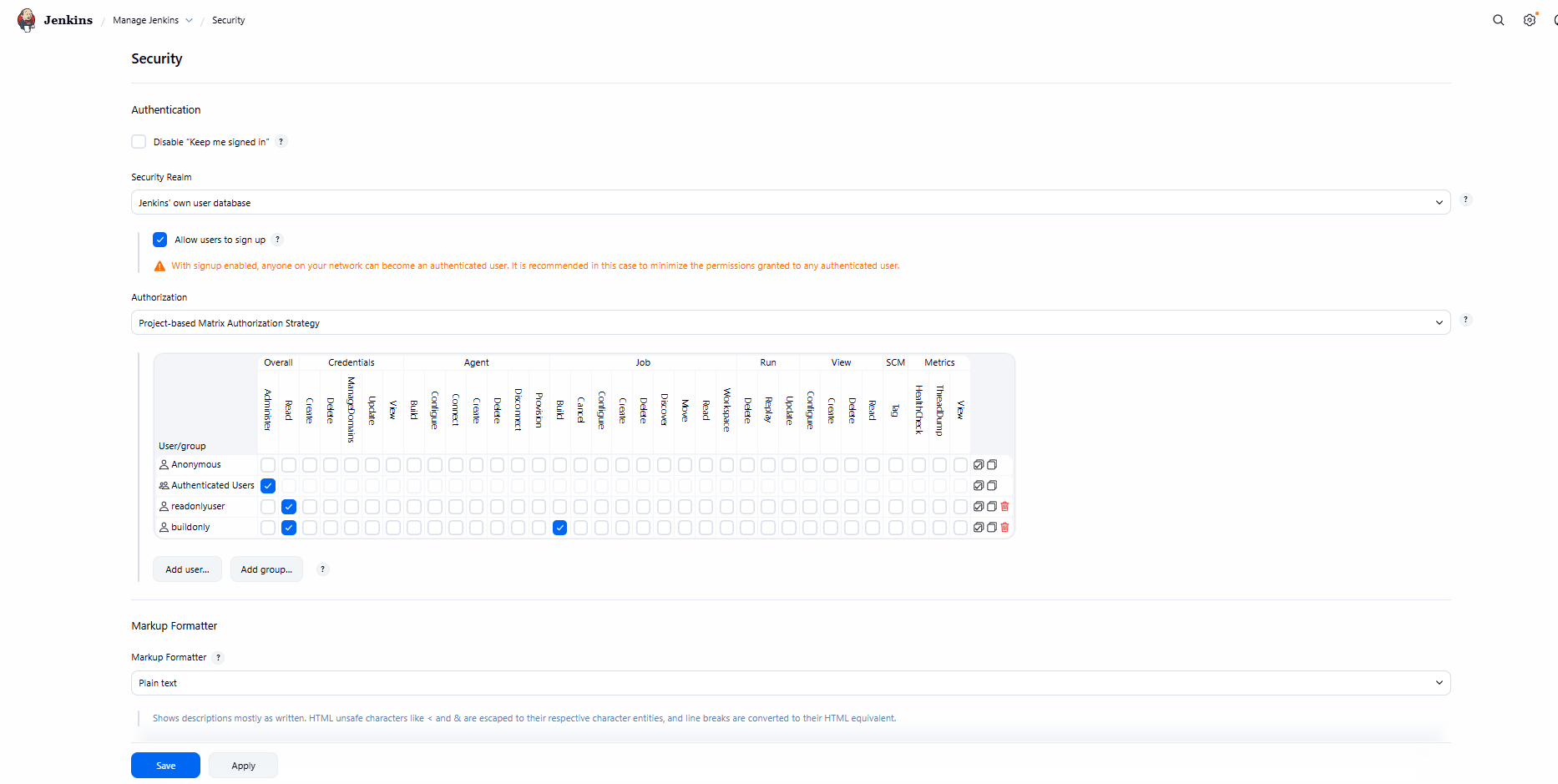
**Tasks: 1. Create two users**

**2. Assign read-only permission to one user**

**3. Assign build permission to another user Expected Output:  Permission differences verified**









**DESCRIPTION:PIPELINES WERE NOT TAUGHT SO MAM ANNOUNCED TO DO TASKS TILL 9 and from 10 to 15 related to pipeline not required**

**TASK 10: Simple Jenkins Pipeline**

**Objective: Create basic pipeline**

**Tasks: 1. Create Pipeline job**

**2. Write pipeline with stages: o Checkout o Build o Test**

**3. Run pipeline Expected Output:  Pipeline stage view**

**TASK 11: Jenkinsfile from Git**

**Objective: Pipeline as Code**

**Tasks: 1. Create Jenkinsfile in Git repo**

**2. Configure pipeline from SCM**

**3. Trigger build Expected Output:  Pipeline executed from Git**

**TASK 12: Post-Build Actions**

**Objective: Handle build result**

**Tasks: 1. Add post section**

1. **Print message on success/failure Expected Output:**  **Appropriate message displayed**

**TASK 13: Trigger Job from Another Job**

**Objective: Job chaining**

**Tasks: 1. Create Job-A and Job-B**

1. **Configure Job-B to trigger after Job-A Expected Output:**  **Job-B triggered automatically**

**TASK 14: Workspace Cleanup**

**Objective: Manage disk usage**

**Tasks: 1. Install Workspace Cleanup plugin**

1. **Clean workspace before build Expected Output:**  **Workspace cleared before execution TASK**

**15: Mini CI Project**

**Objective: Implement basic CI flow**

**Tasks: 1. Git commit → Jenkins build**

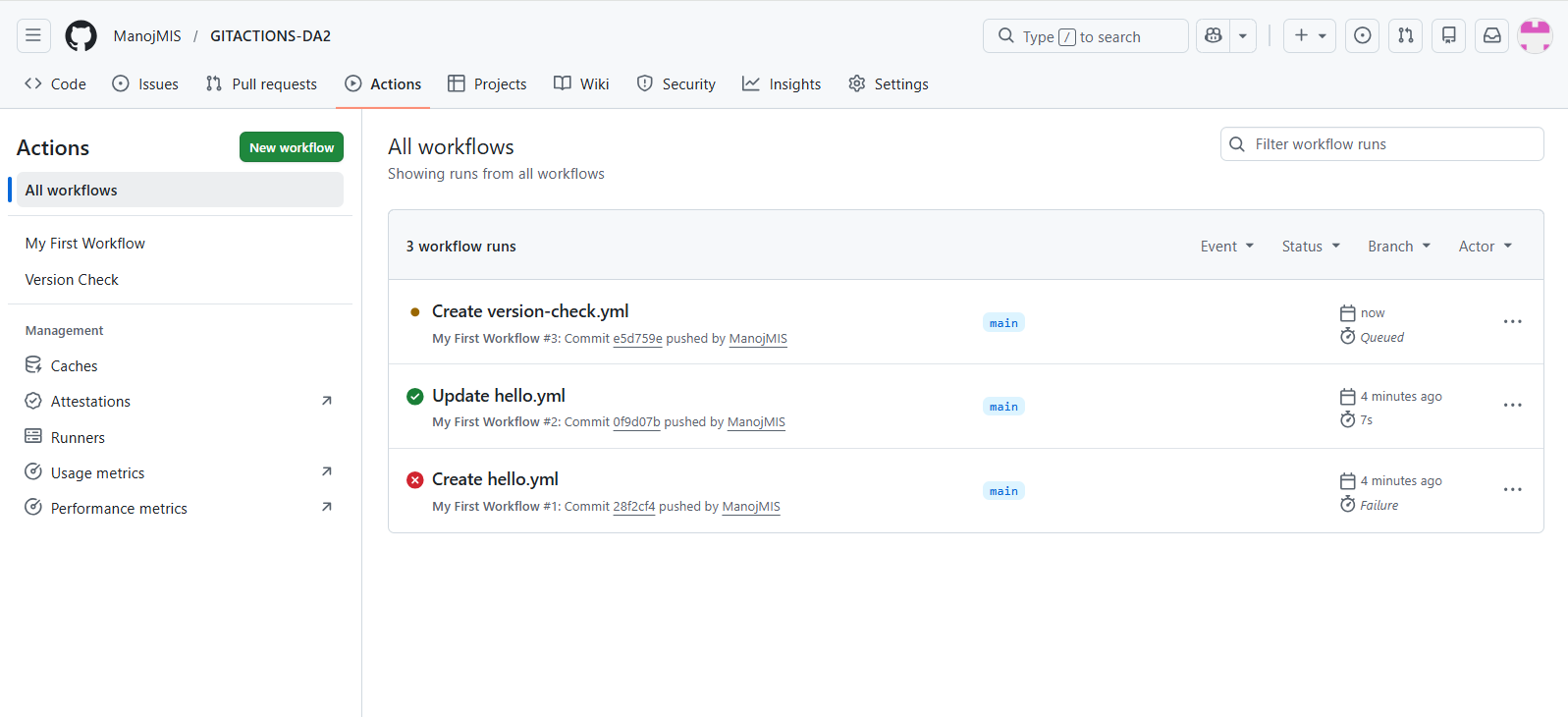
**2. Compile code**

**3. Archive artifacts**

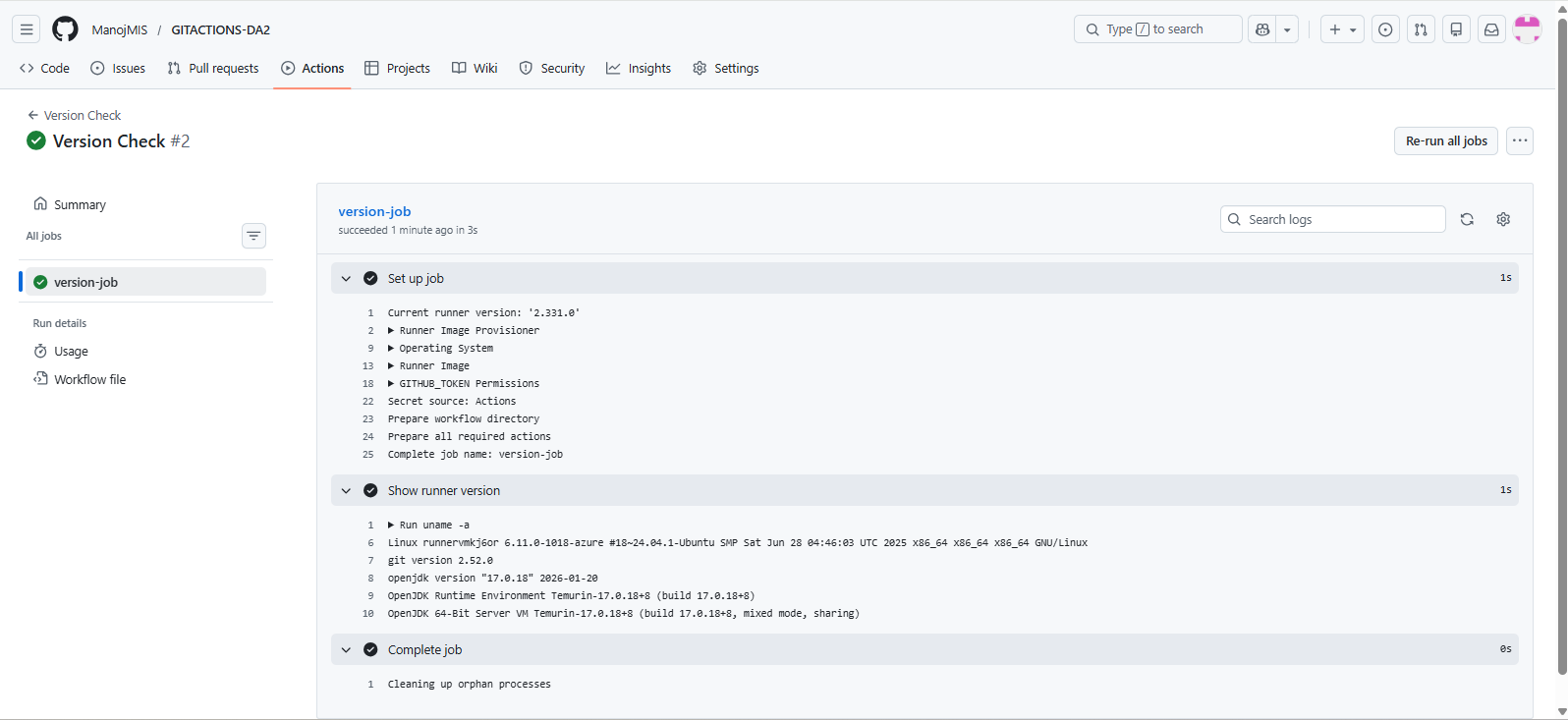
**4. Fail build on error Expected Output:**  **Automated CI pipeline**

**TOOL:GIT ACTIONS**

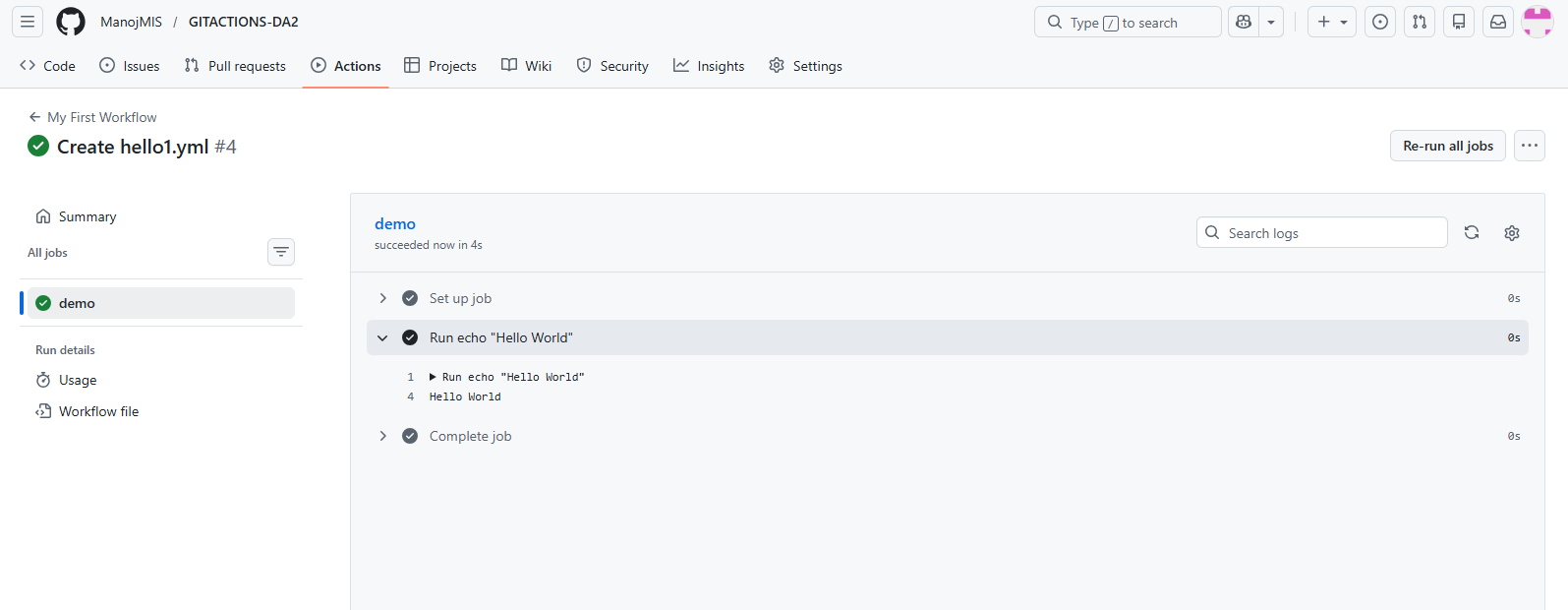
**1.)GIT ACTIONS DASBOARD:**



**VERSION AND OUTPUT:**



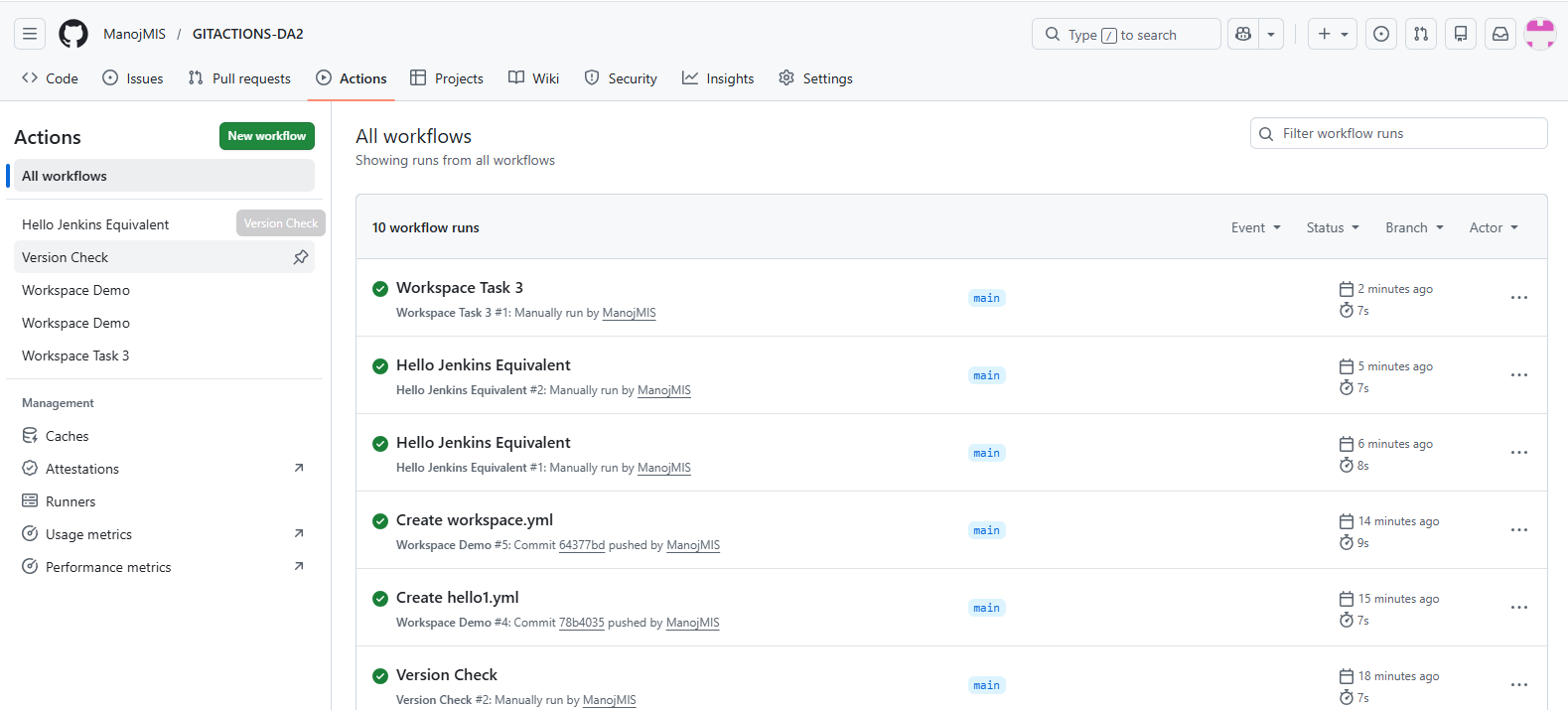
**2.)First Freestyle job created and Output:**



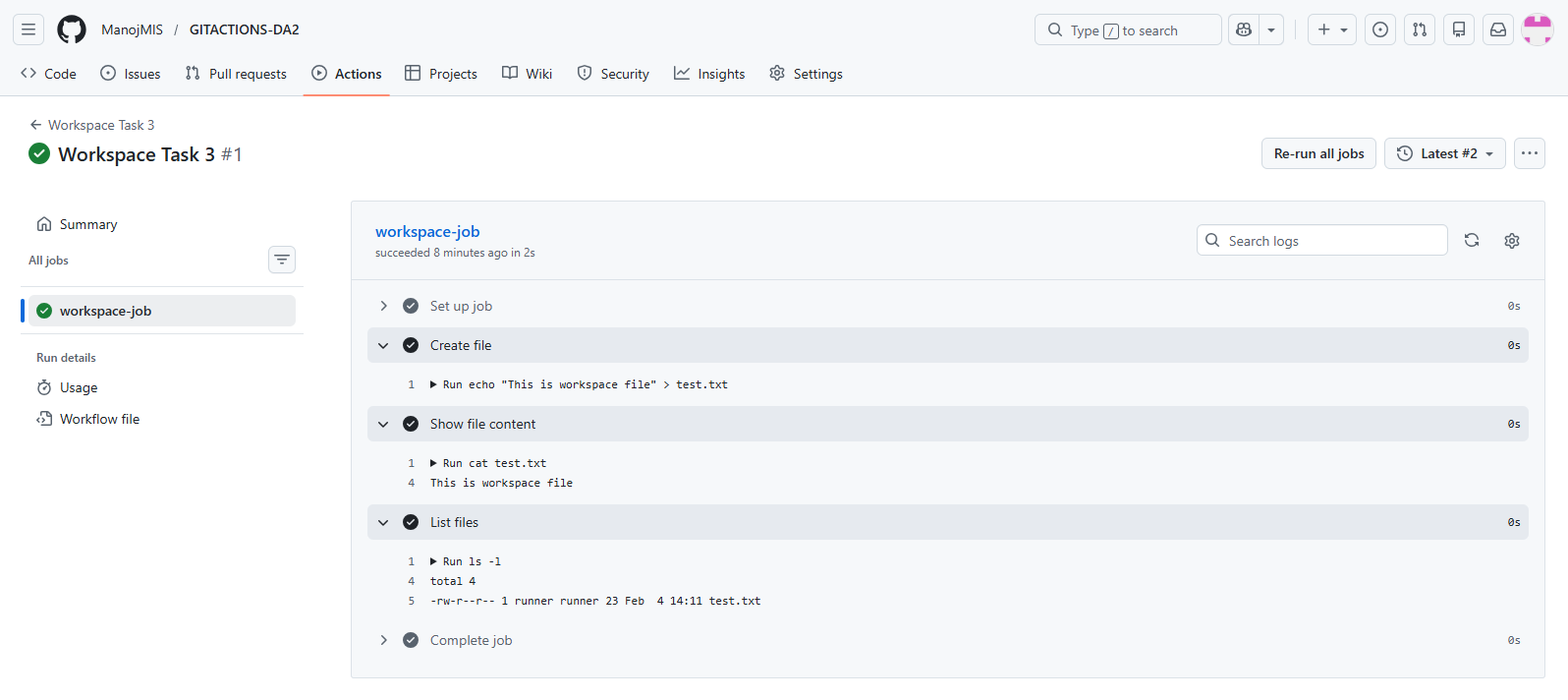
3.)**Text File Created:**



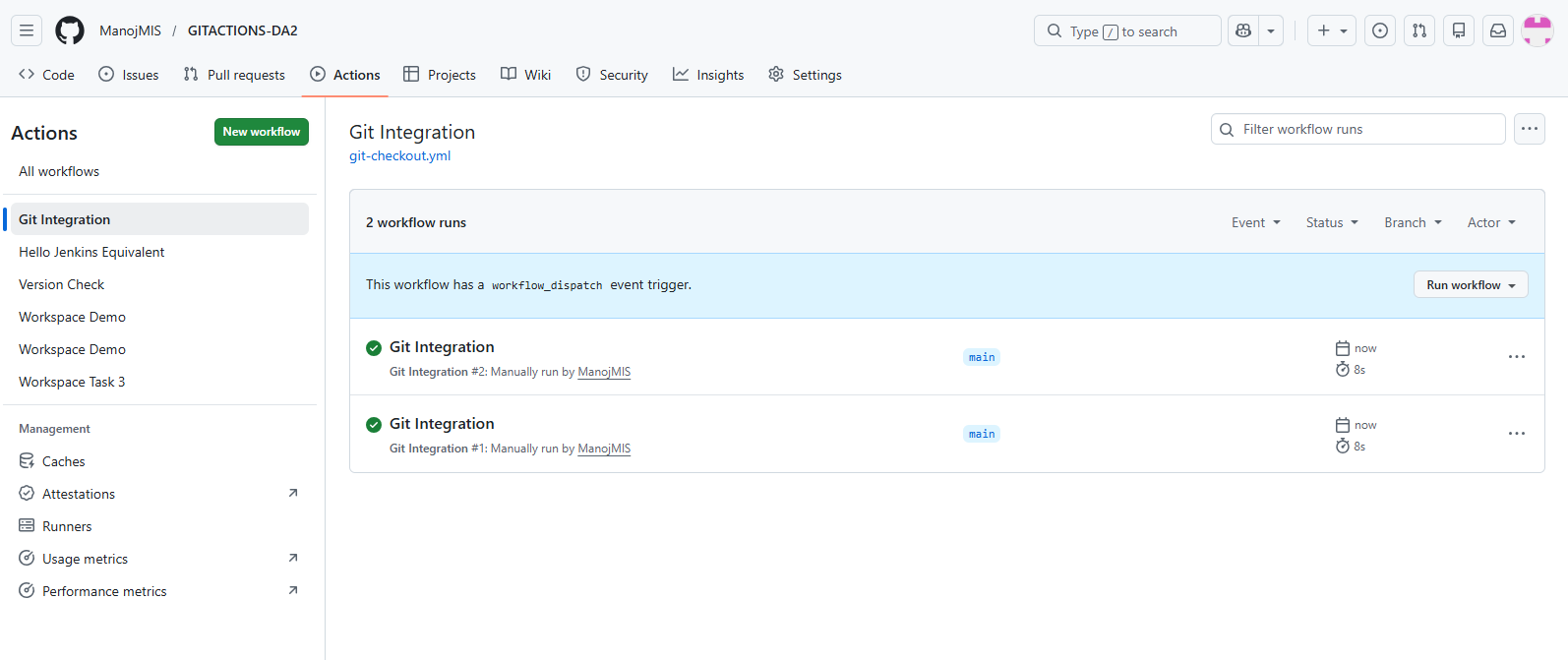
**Job Workspace:**



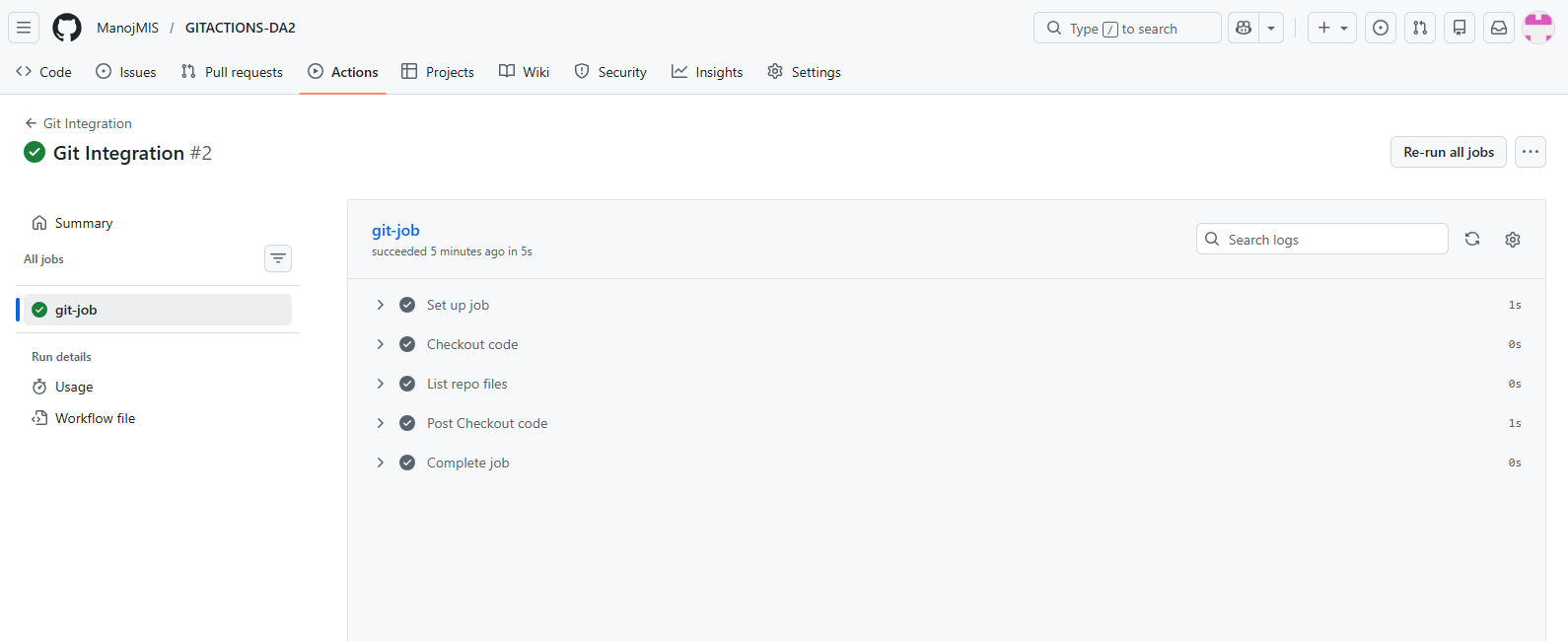
**OUTPUT:**

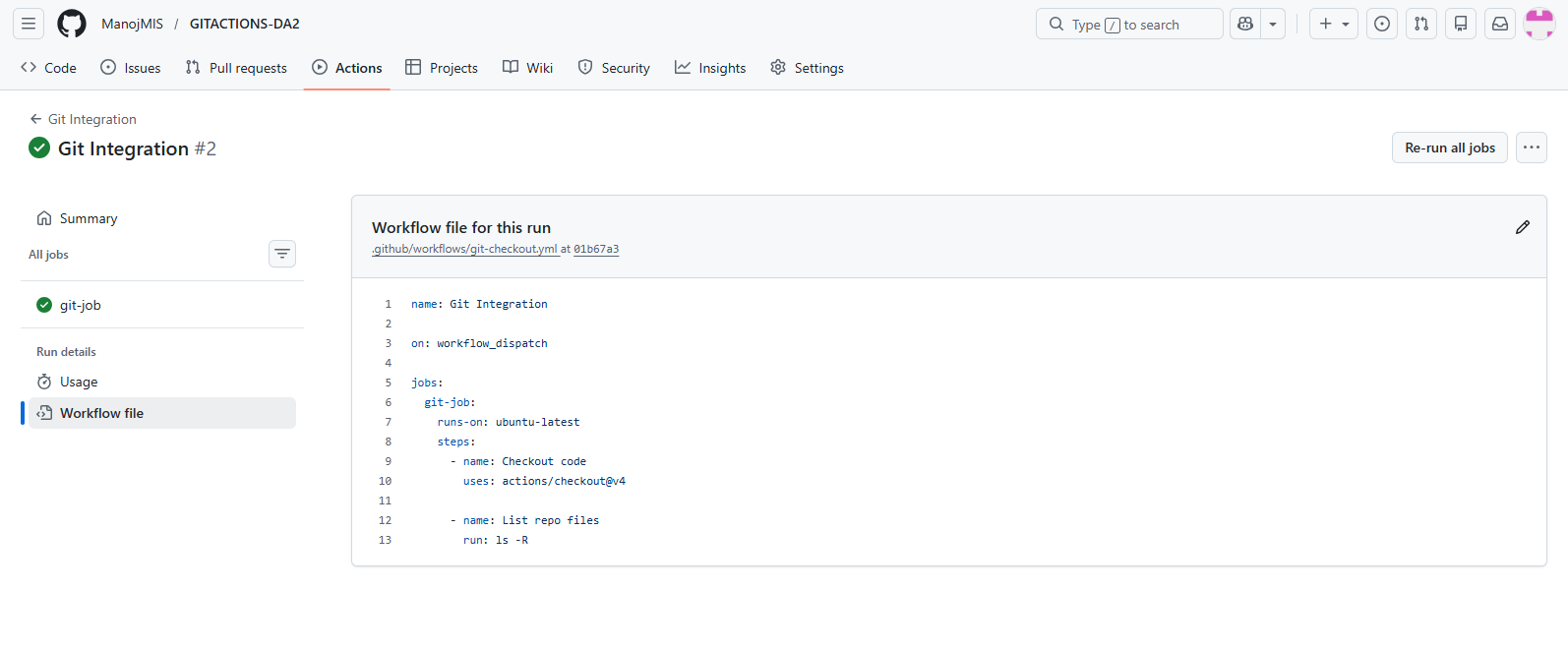


4.)**Git Integration Workflow created:**

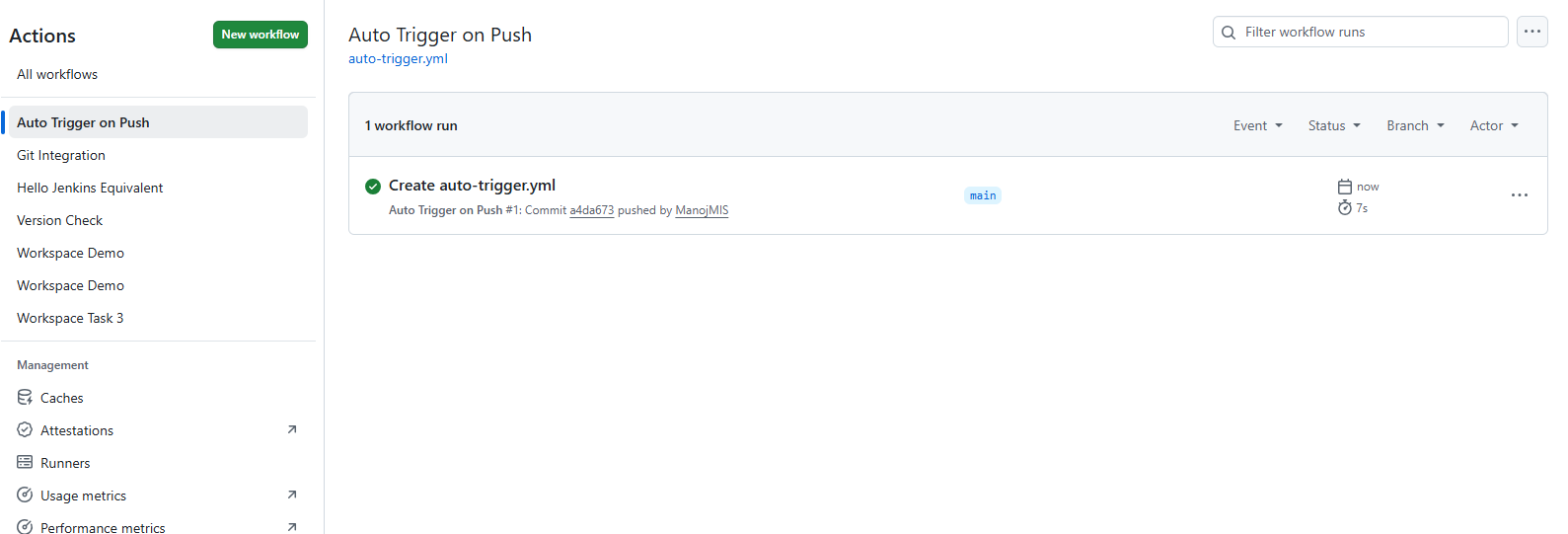


**OUTPUT:**

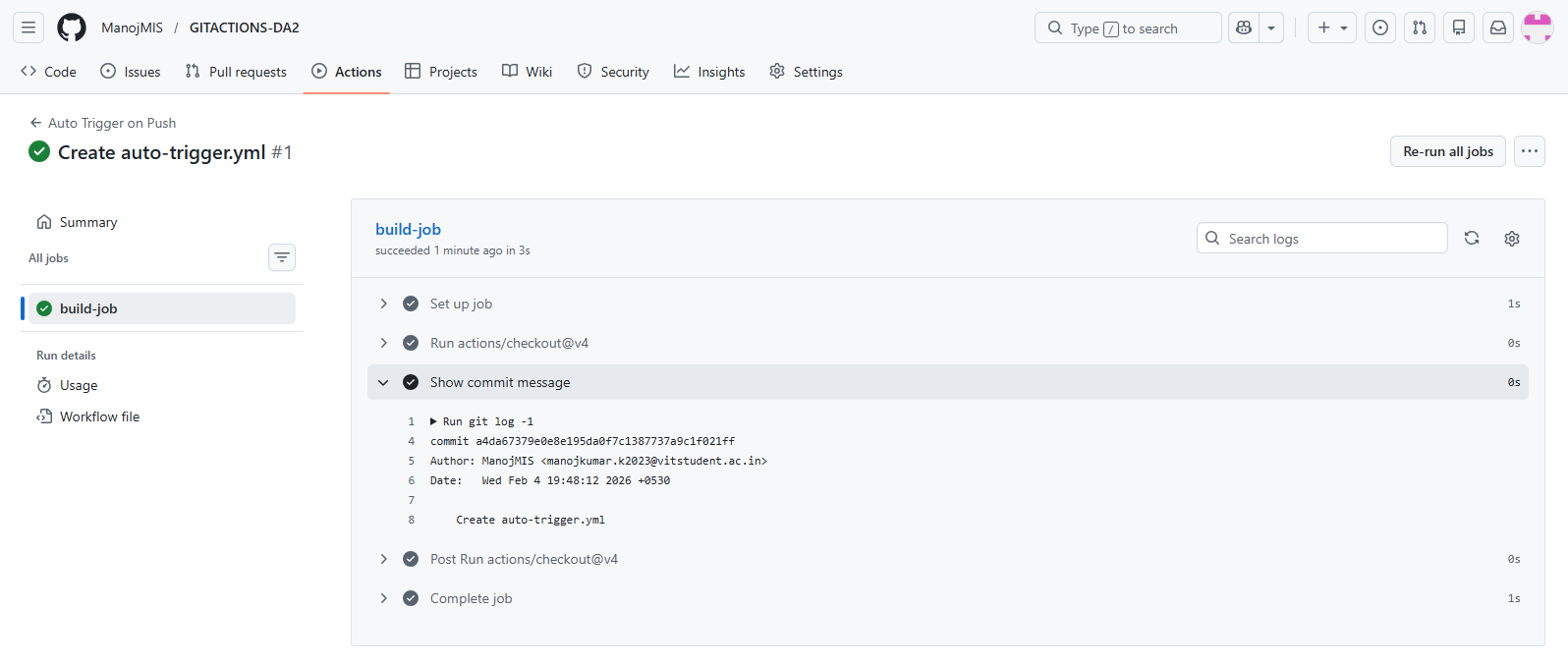




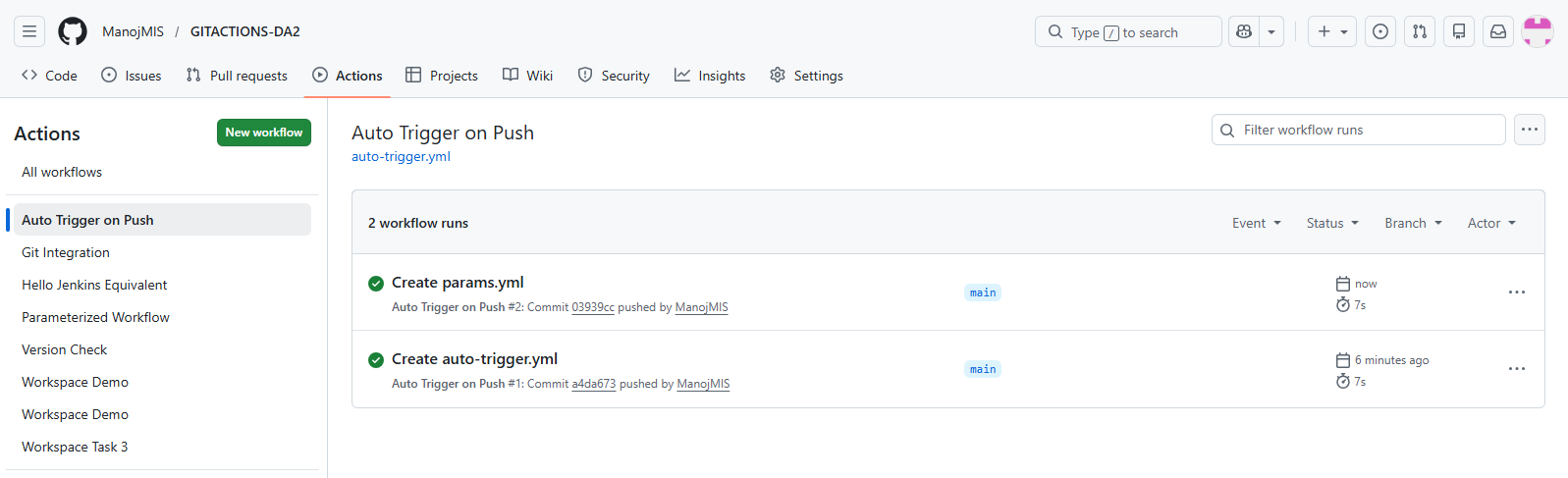
5.)**Auto Trigger Workflow created:**



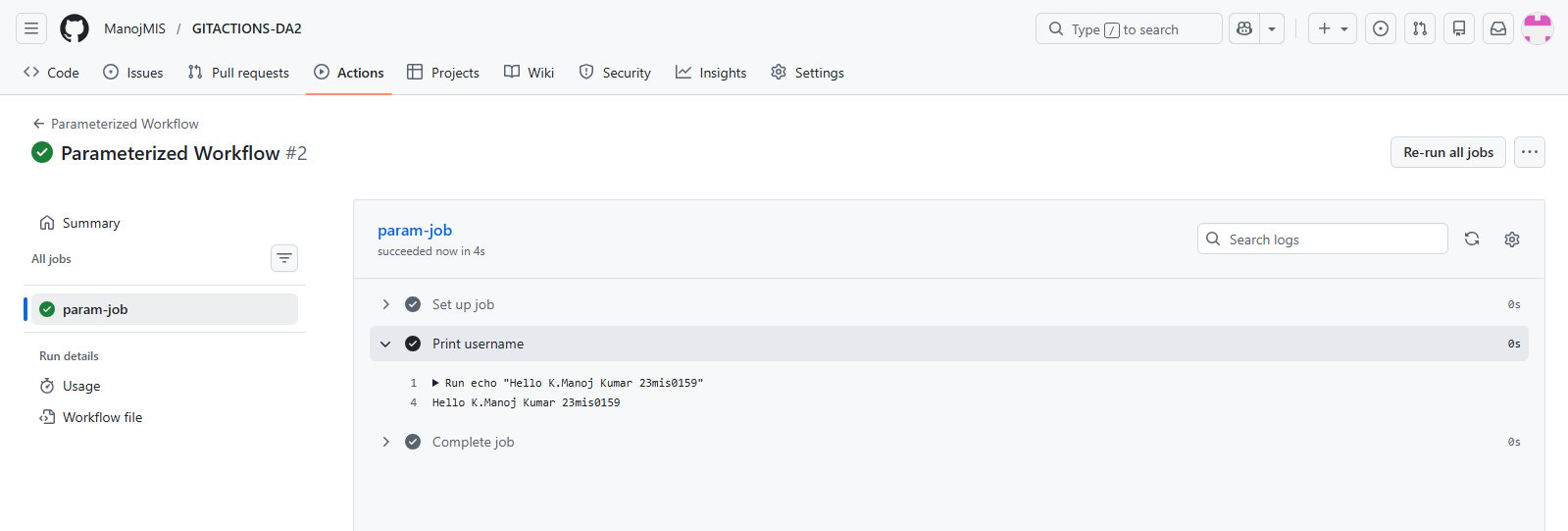
**OUTPUT:**



6.)**Parameterized Workflow Created:**

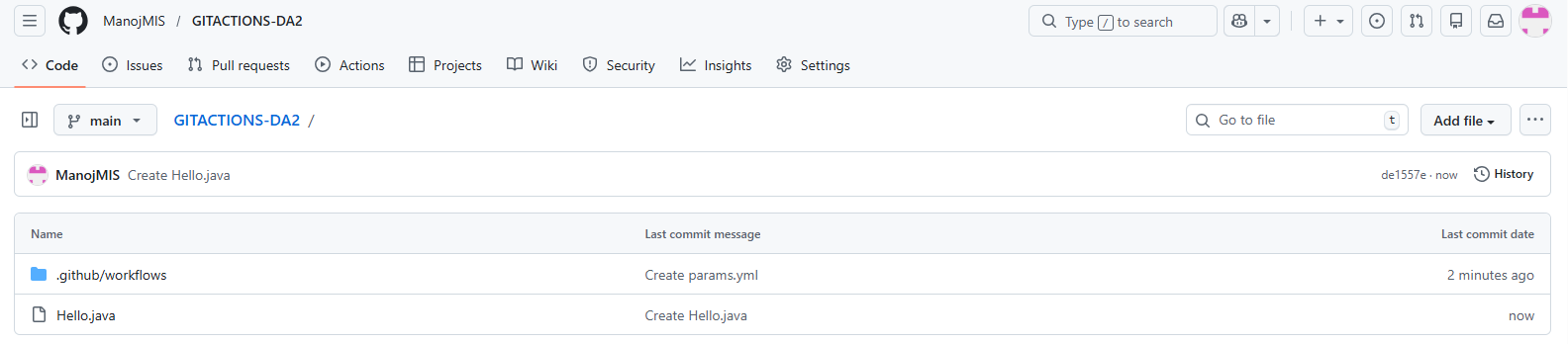


**OUTPUT:**

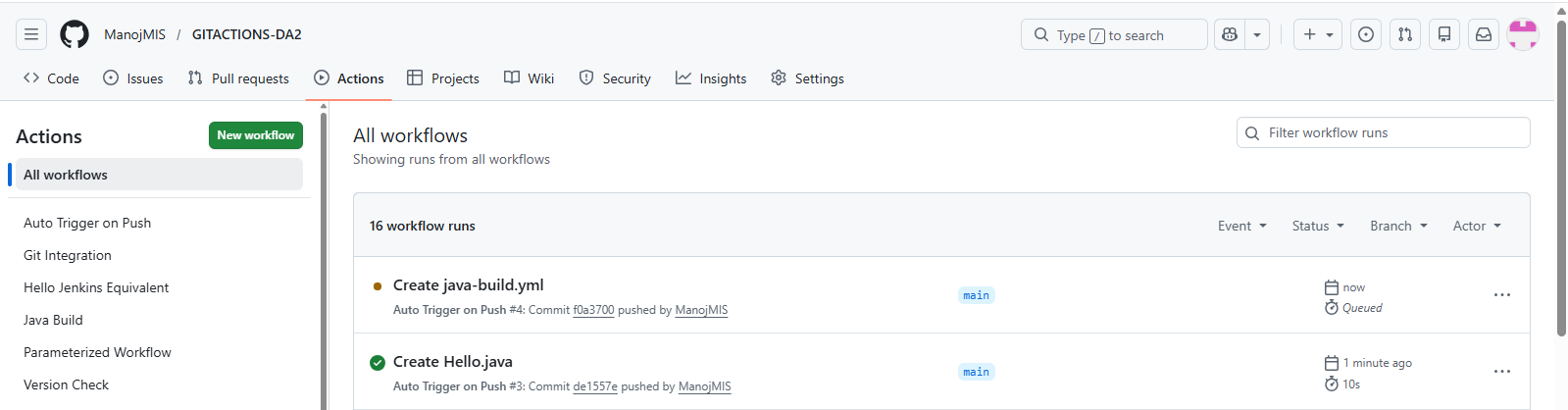


7

7.)**Hello.java FILE:**

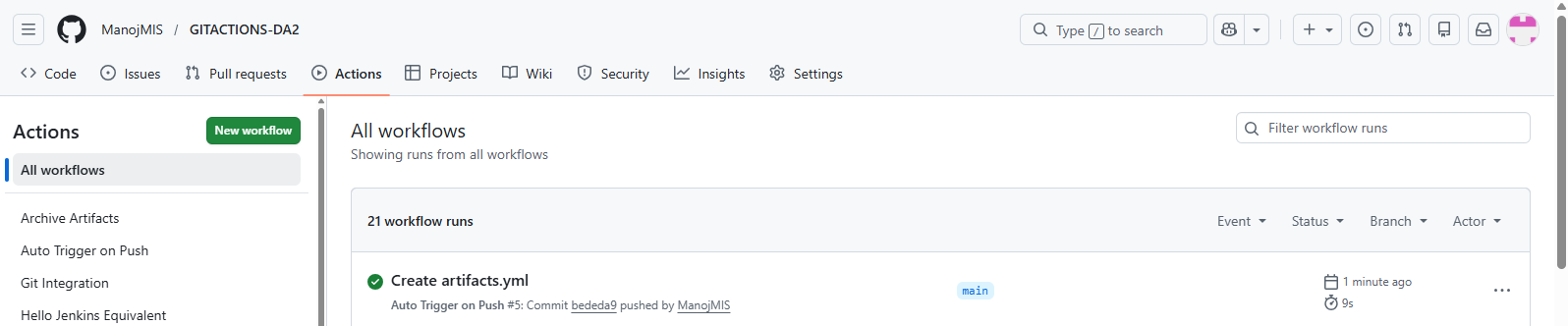


**JAVA workflow created:**

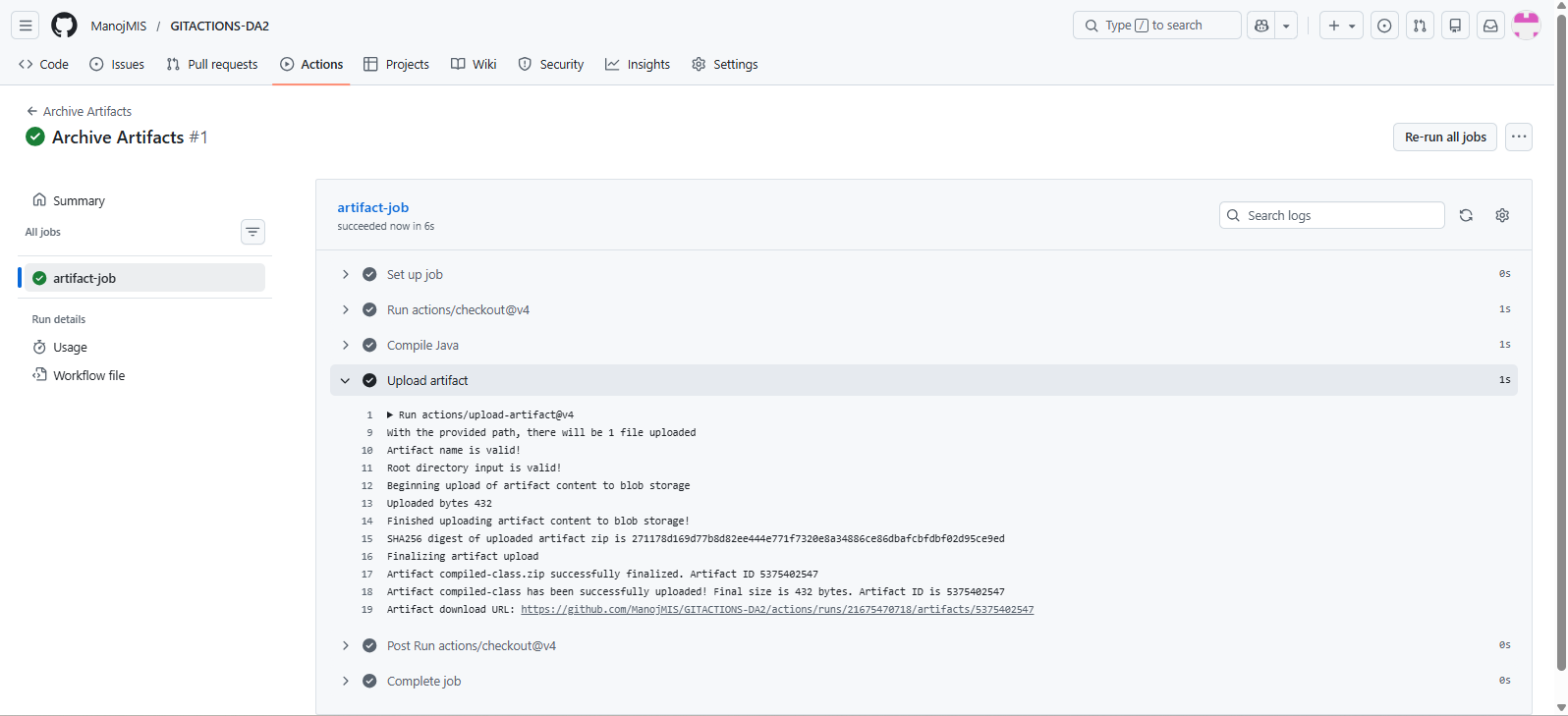


**OUTPUT:**  

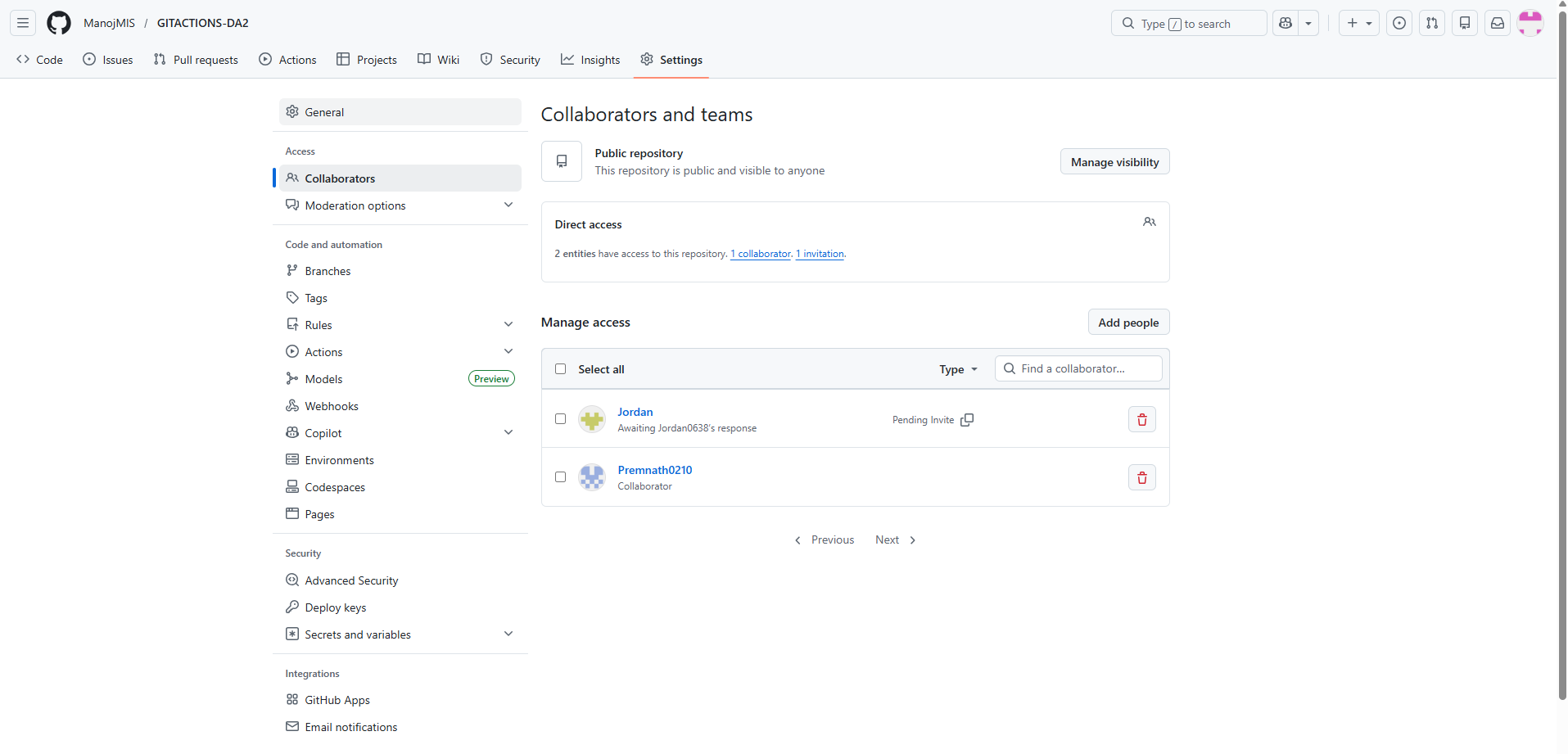

**8.)Artifacts workflow Created:**



**OUTPUT:**

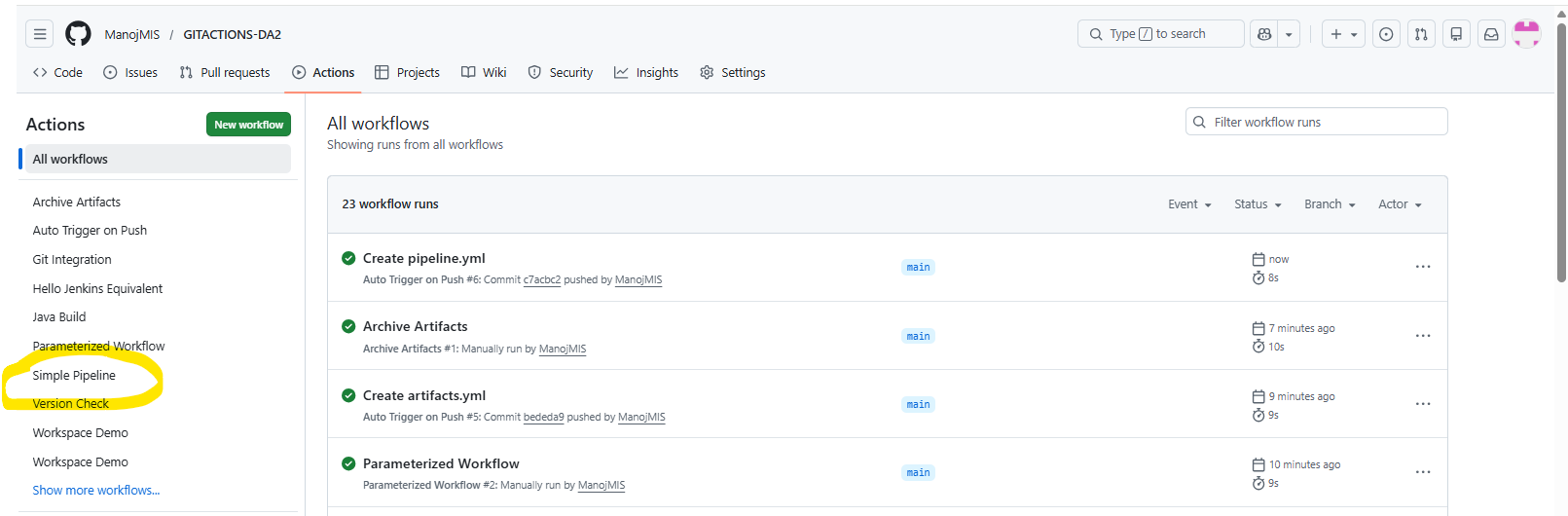


**9.)GIT USERS:**

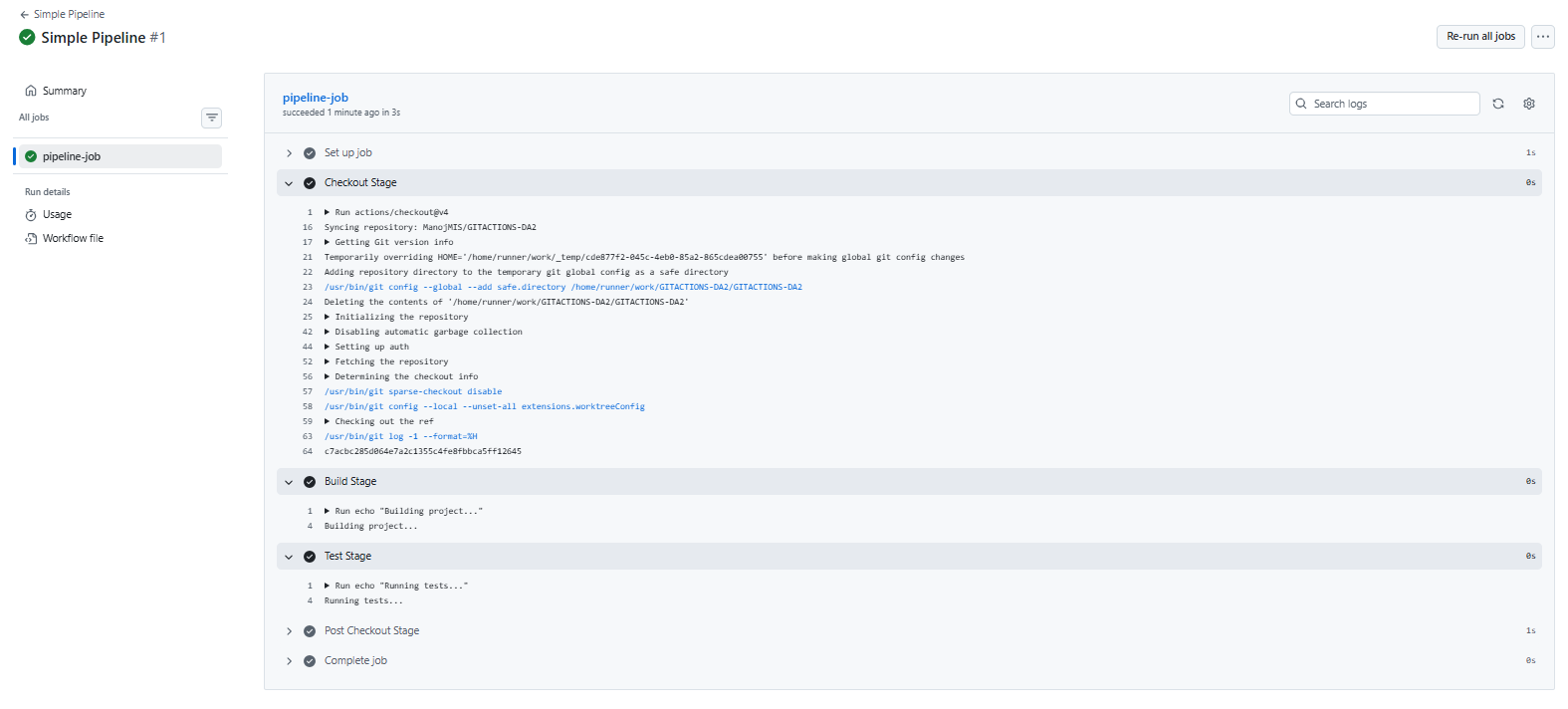


10.)

**Pipeline workflow created:**



**OUTPUT:**

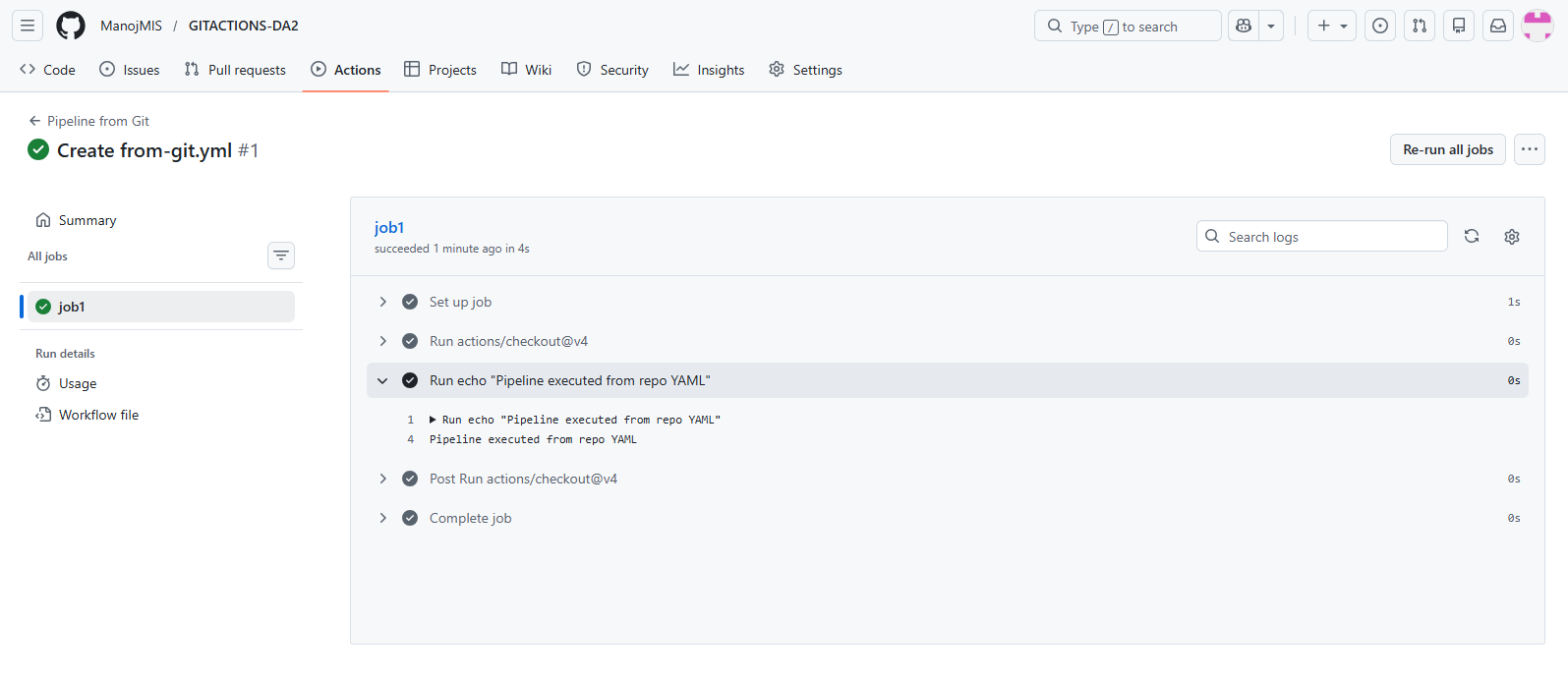


11.)

**Pipeline from GIT workflow created:**

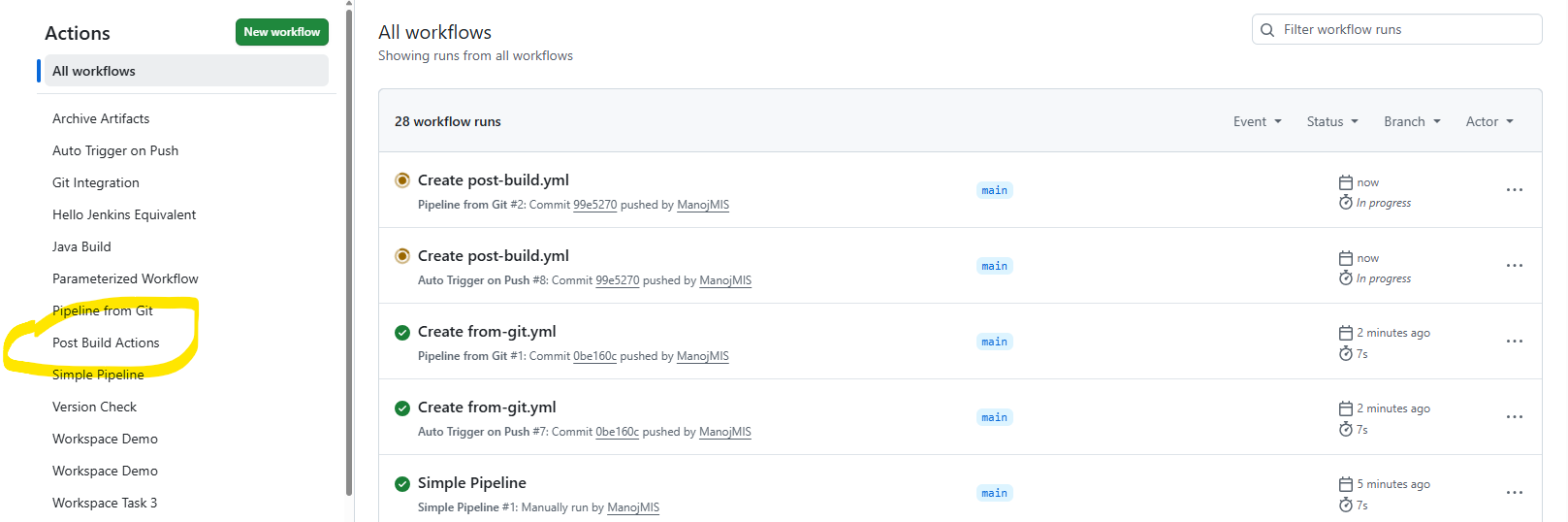


**OUTPUT:**



12.)

**Post-build workflow created:**

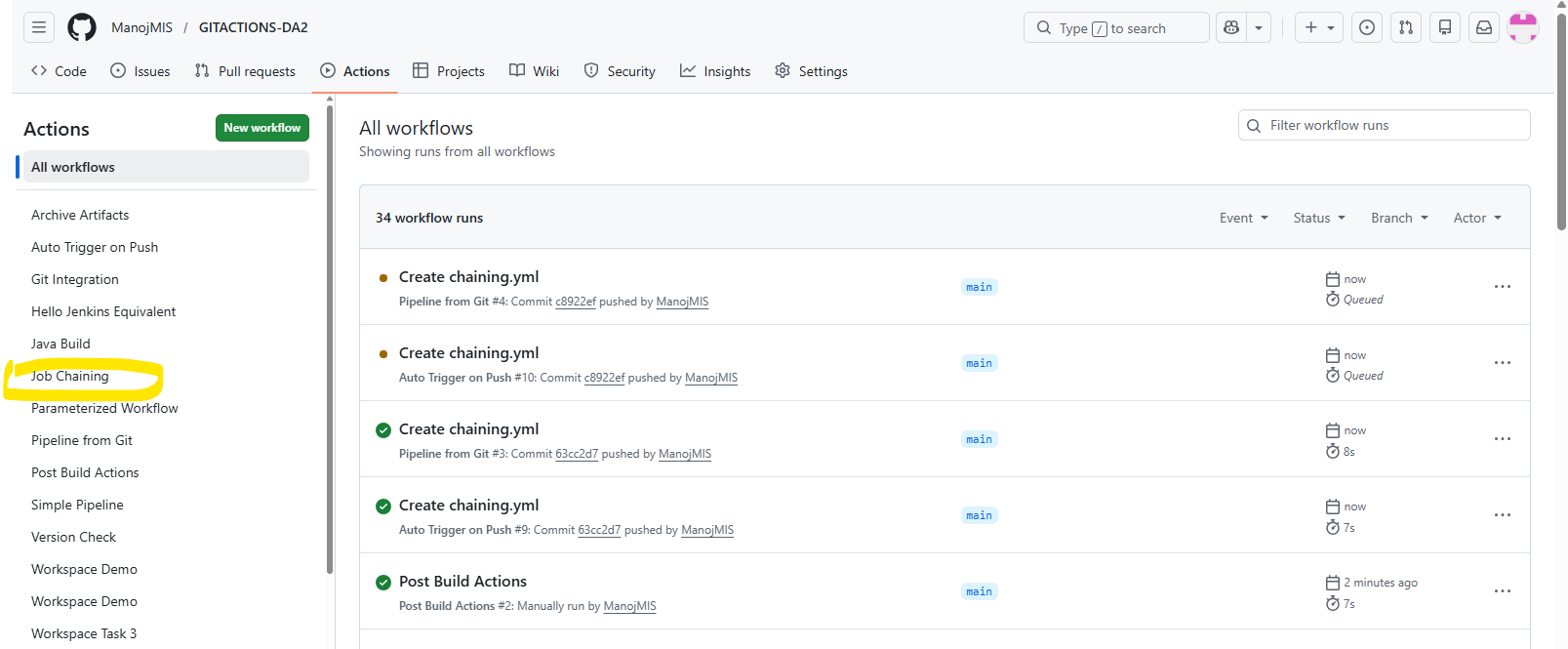


**OUTPUT:**



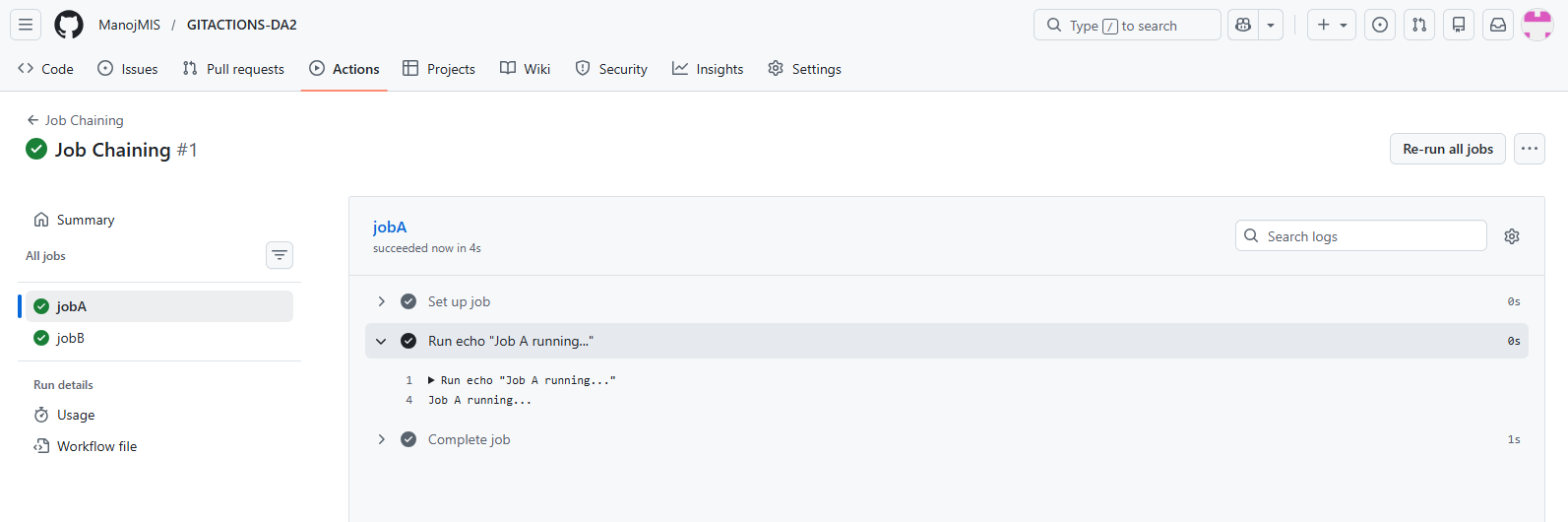
13.)

**Chaining workflow created:**

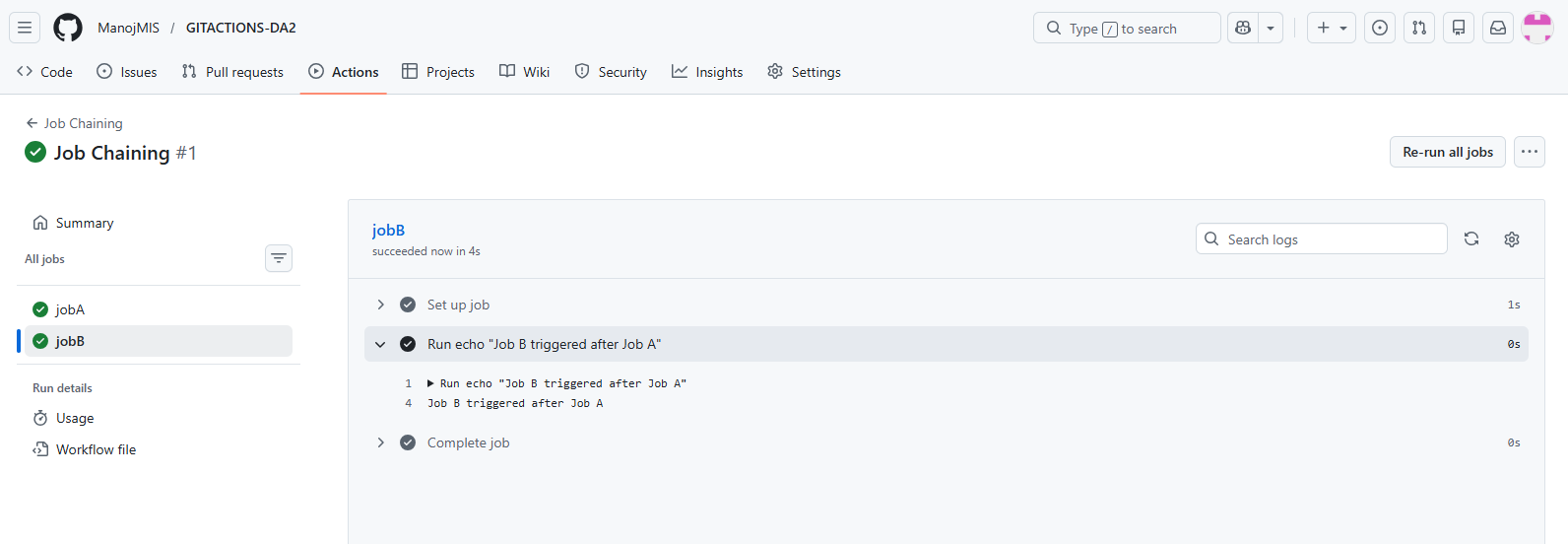


**OUTPUT:**

**JOB-A:**

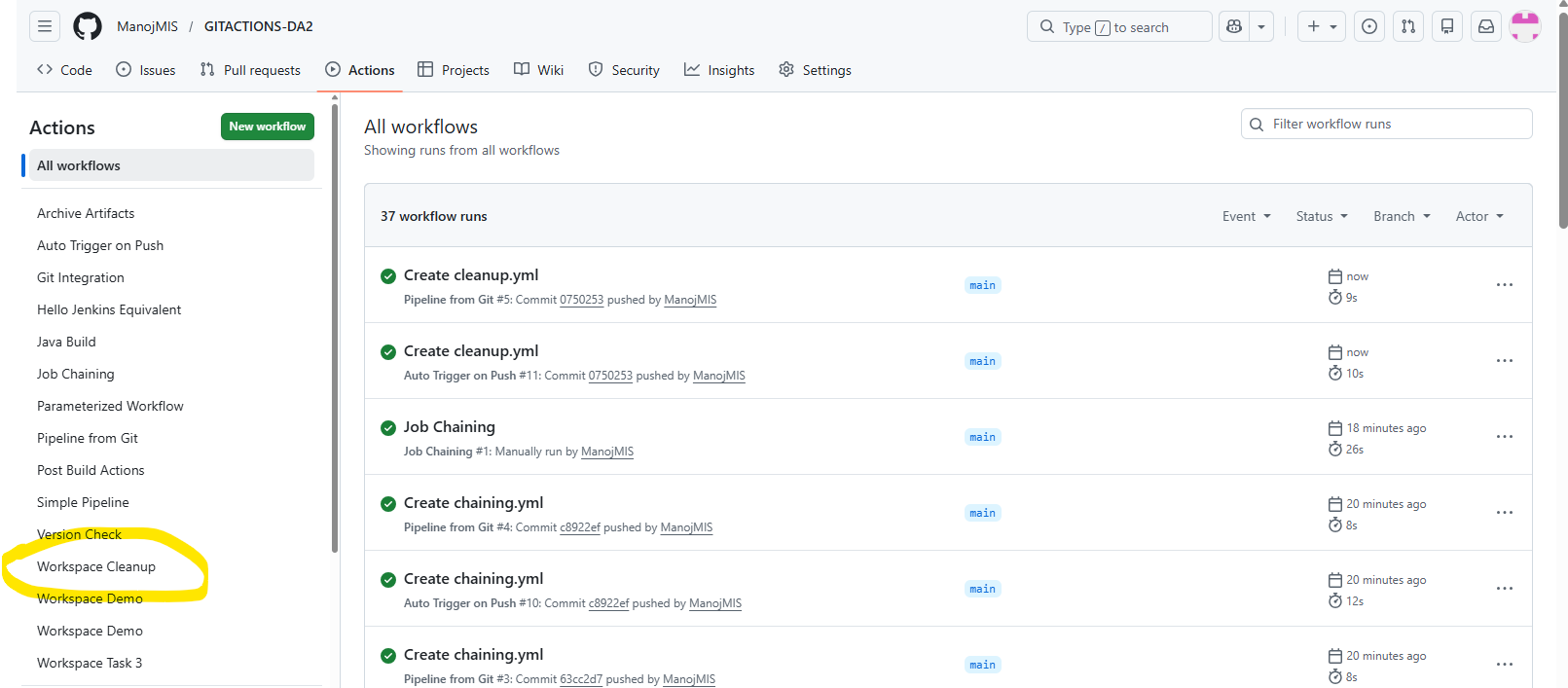


**JOB-B:**

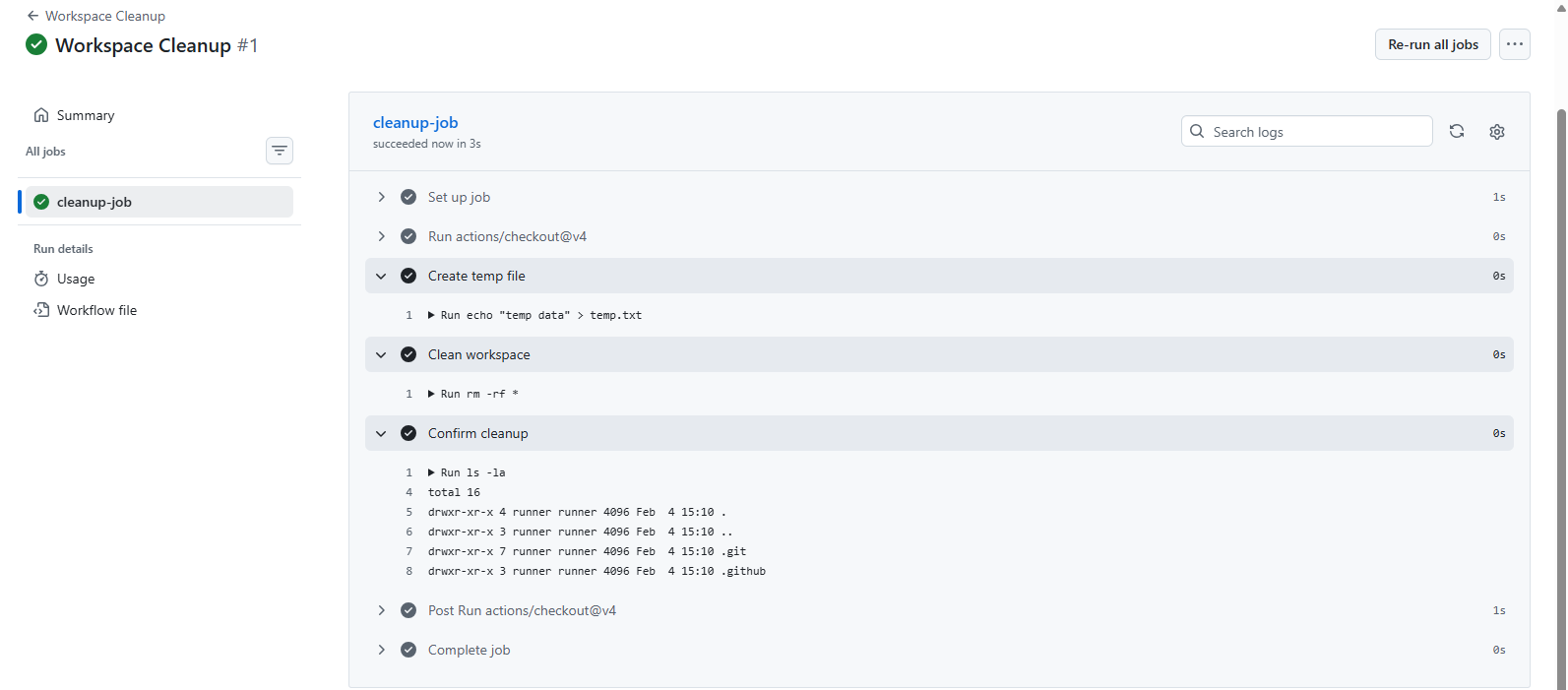


14.)

**Cleanup Workflow created:**

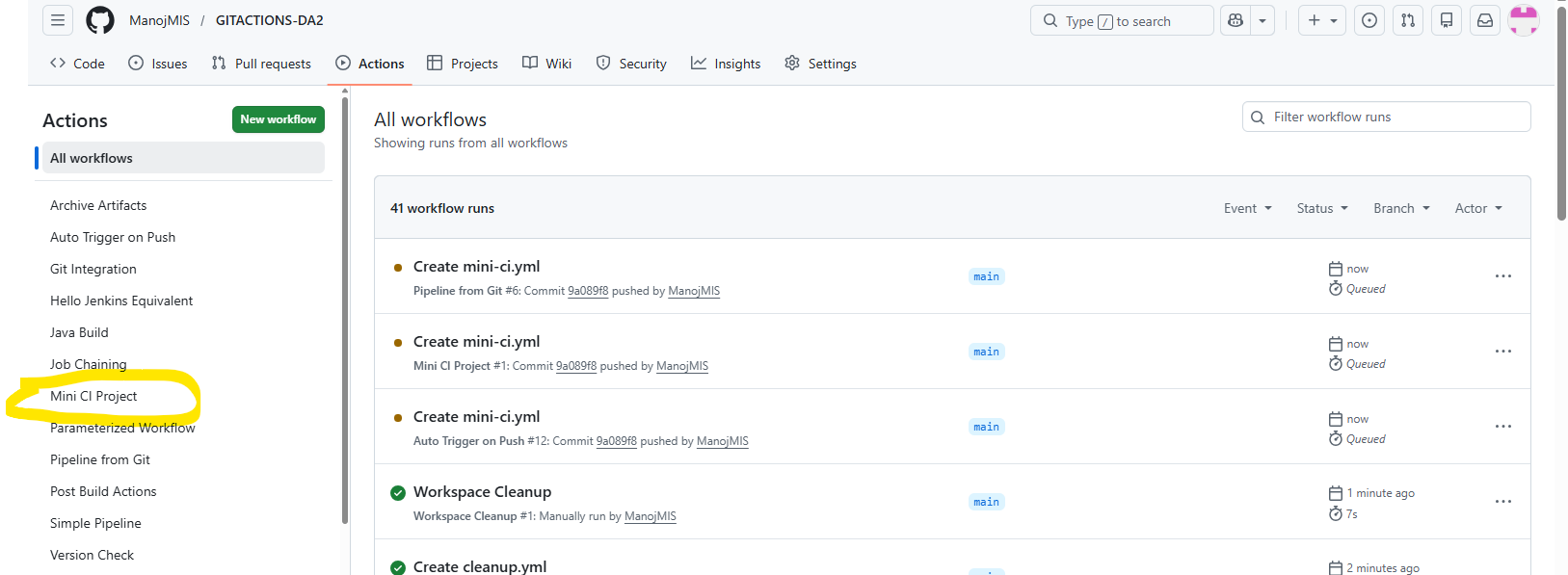


**OUTPUT:**

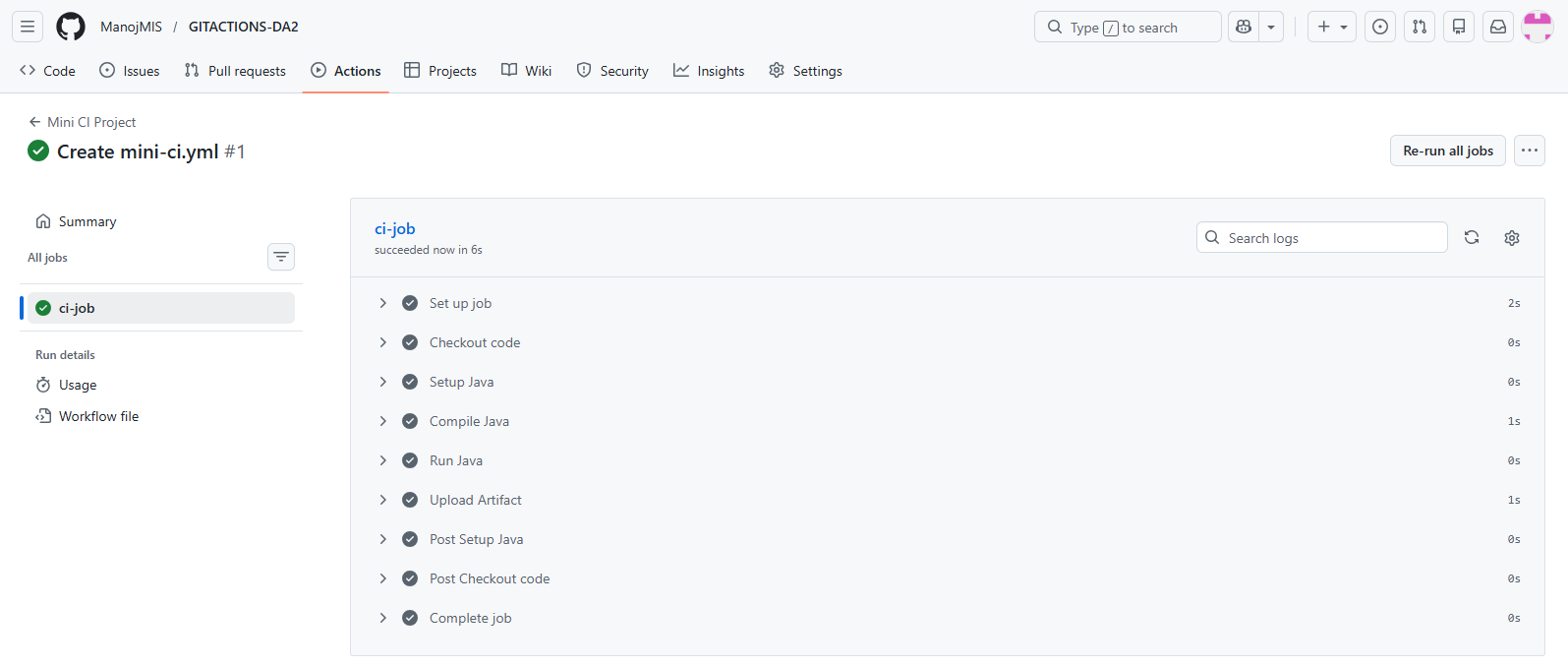


15.)

**CI workflow created:**



**OUTPUT:**



**ALL THE WORKFLOWS CREATED:**

