Module : 06

Jenkins Exercise

Name : Anubhav Ranjan ***Registration No : 774***

College : IIIT kalyani ***Roll No : ECE/21114***

Exercise 4: Pipeline Job - End-to-End CI/CD for a React Application

1. Objective: Implement a CI/CD pipeline for a React web application.

2. Steps:

o Create a Pipeline Job.

o Write a Jenkinsfile to:

▪ Clone the React project from GitHub.

▪ Install dependencies (npm install).

▪ Run tests (npm test).

▪ Build the project (npm run build).

▪ Deploy the application by copying the build folder to a deployment directory.

o Add stages for:

▪ Linting using ESLint.

▪ Post-deployment testing using curl to verify the app is running.

3. Task: Run the pipeline and validate each stage's output.

React Project GitHub Repo Link:

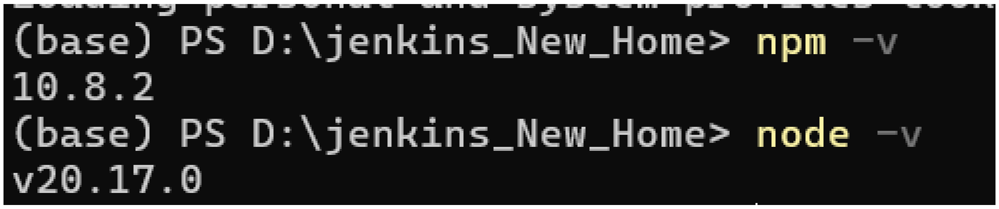
https://github.com/feelinganubhav/React-ToDo-App.git

Let's complete this exercise step by step of End-to-End CI/CD for a React Application using Jenkins.

Step 1: Prerequisites

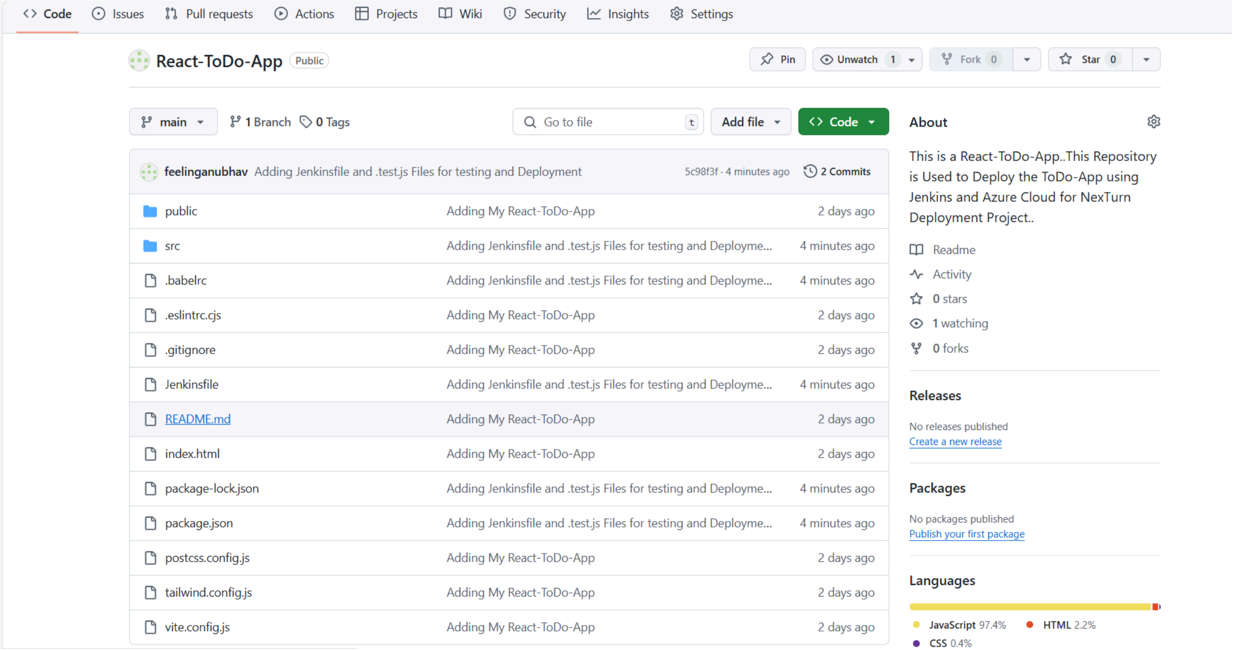
Before proceeding, ensure the following:

1. **Jenkins is installed and running.**
2. **Node.js and npm are installed** on the Jenkins server. Verify by running:
   `node -v & npm -v`



```
(base) PS D:\jenkins_New_Home> npm -v
10.8.2
(base) PS D:\jenkins_New_Home> node -v
v20.17.0
```

3. A **React project repository** (a React ToDo app) is available on GitHub.



feelinganubhav Adding Jenkinsfile and .test.js Files for testing and Deployment 5c98f3f · 4 minutes ago 2 Commits

public	Adding My React-ToDo-App	2 days ago
src	Adding Jenkinsfile and .test.js Files for testing and Deployme...	4 minutes ago
.babelrc	Adding Jenkinsfile and .test.js Files for testing and Deployme...	4 minutes ago
.eslintrc.cjs	Adding My React-ToDo-App	2 days ago
.gitignore	Adding My React-ToDo-App	2 days ago
Jenkinsfile	Adding Jenkinsfile and .test.js Files for testing and Deployme...	4 minutes ago
README.md	Adding My React-ToDo-App	2 days ago
index.html	Adding My React-ToDo-App	2 days ago
package-lock.json	Adding Jenkinsfile and .test.js Files for testing and Deployme...	4 minutes ago
package.json	Adding Jenkinsfile and .test.js Files for testing and Deployme...	4 minutes ago
postcss.config.js	Adding My React-ToDo-App	2 days ago
tailwind.config.js	Adding My React-ToDo-App	2 days ago
vite.config.js	Adding My React-ToDo-App	2 days ago

About

This is a React-ToDo-App..This Repository is Used to Deploy the ToDo-App using Jenkins and Azure Cloud for NexTurn Deployment Project..

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

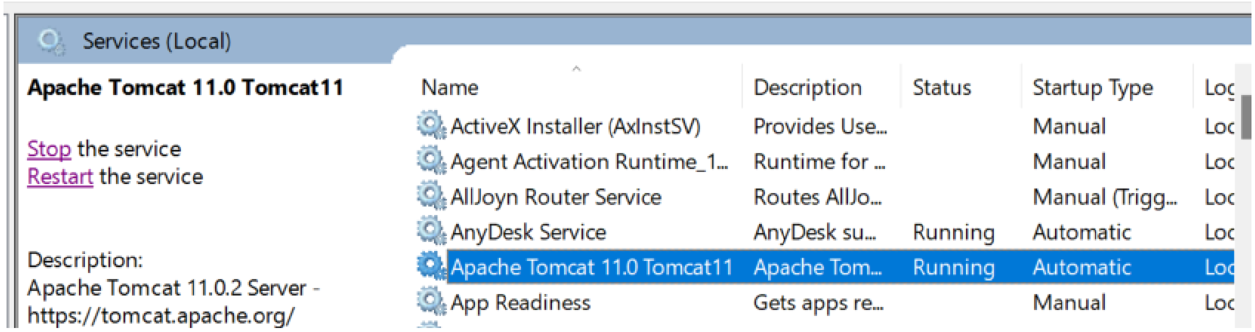
Languages

- JavaScript 97.4%
- HTML 2.2%
- CSS 0.4%

4. The server has access to the deployment directory.

Step 2: Create a Freestyle Job in Jenkins

1. **Open Jenkins:**
   - Start the Apache Tomcat Server from Services on your Windows



Services (Local)

Apache Tomcat 11.0 Tomcat11

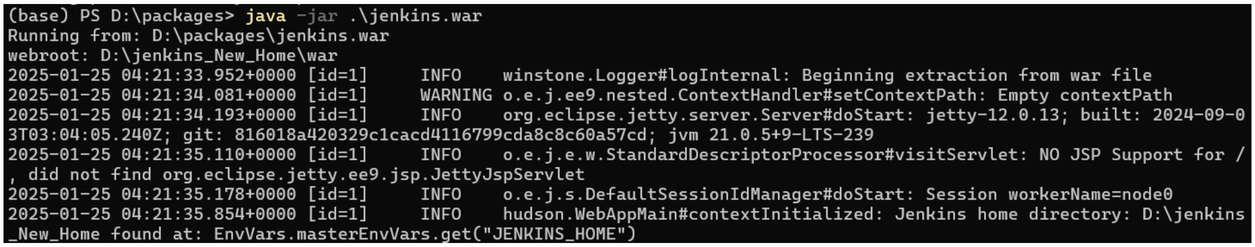
Stop the service
Restart the service

Description:
Apache Tomcat 11.0.2 Server - https://tomcat.apache.org/

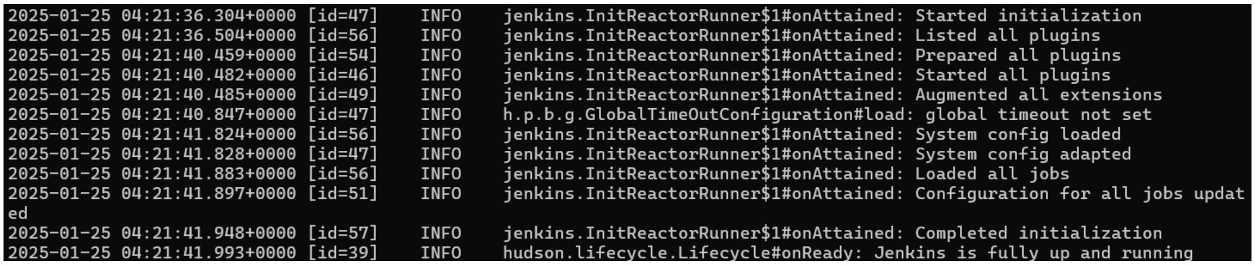
Name	Description	Status	Startup Type	Log
ActiveX Installer (AxInstSV)	Provides Use...		Manual	Loc
Agent Activation Runtime_1...	Runtime for ...		Manual	Loc
AllJoyn Router Service	Routes AllJo...		Manual (Trigg...	Loc
AnyDesk Service	AnyDesk su...	Running	Automatic	Loc
Apache Tomcat 11.0 Tomcat11	Apache Tom...	Running	Automatic	Loc
App Readiness	Gets apps re...		Manual	Loc

   - Or Open the Terminal..Navigate to the directory where jenkins.war file is present and Run the command:

     `java -jar .\jenkins.war`

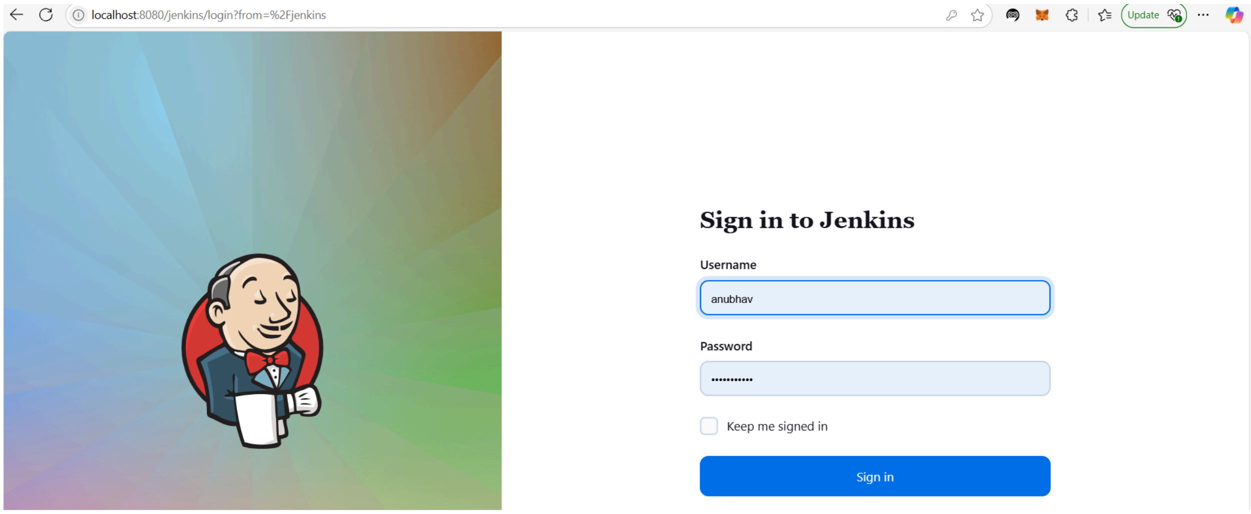


```
(base) PS D:\packages> java -jar .\jenkins.war
Running from: D:\packages\jenkins.war
webroot: D:\jenkins_New_Home\war
2025-01-25 04:21:33.952+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2025-01-25 04:21:34.081+0000 [id=1]     WARNING o.e.j.ee9.nested.ContextHandler#setContextPath: Empty contextPath
2025-01-25 04:21:34.193+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-12.0.13; built: 2024-09-0
3T03:04:05.240Z; git: 816018a420329c1cacd4116799cda8c8c60a57cd; jvm 21.0.5+9-LTS-239
2025-01-25 04:21:35.110+0000 [id=1]     INFO    o.e.j.e.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /
, did not find org.eclipse.jetty.ee9.jsp.JettyJspServlet
2025-01-25 04:21:35.178+0000 [id=1]     INFO    o.e.j.s.DefaultSessionIdManager#doStart: Session workerName=node0
2025-01-25 04:21:35.854+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: D:\jenkins
_New_Home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
```



```
2025-01-25 04:21:36.304+0000 [id=47]    INFO    jenkins.InitReactorRunner$1#onAttained: Started initialization
2025-01-25 04:21:36.504+0000 [id=56]    INFO    jenkins.InitReactorRunner$1#onAttained: Listed all plugins
2025-01-25 04:21:40.459+0000 [id=54]    INFO    jenkins.InitReactorRunner$1#onAttained: Prepared all plugins
2025-01-25 04:21:40.482+0000 [id=46]    INFO    jenkins.InitReactorRunner$1#onAttained: Started all plugins
2025-01-25 04:21:40.485+0000 [id=49]    INFO    jenkins.InitReactorRunner$1#onAttained: Augmented all extensions
2025-01-25 04:21:40.847+0000 [id=47]    INFO    h.p.b.g.GlobalTimeOutConfiguration#load: global timeout not set
2025-01-25 04:21:41.824+0000 [id=56]    INFO    jenkins.InitReactorRunner$1#onAttained: System config loaded
2025-01-25 04:21:41.828+0000 [id=47]    INFO    jenkins.InitReactorRunner$1#onAttained: System config adapted
2025-01-25 04:21:41.883+0000 [id=56]    INFO    jenkins.InitReactorRunner$1#onAttained: Loaded all jobs
2025-01-25 04:21:41.897+0000 [id=51]    INFO    jenkins.InitReactorRunner$1#onAttained: Configuration for all jobs updat
ed
2025-01-25 04:21:41.948+0000 [id=57]    INFO    jenkins.InitReactorRunner$1#onAttained: Completed initialization
2025-01-25 04:21:41.993+0000 [id=39]    INFO    hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
```

   - Launch Jenkins in your browser (usually `http://localhost:8080 or http://localhost:8080/jenkins`).
   - Log in with your credentials.



localhost:8080/jenkins/login?from=%2Fjenkins

Sign in to Jenkins

Username

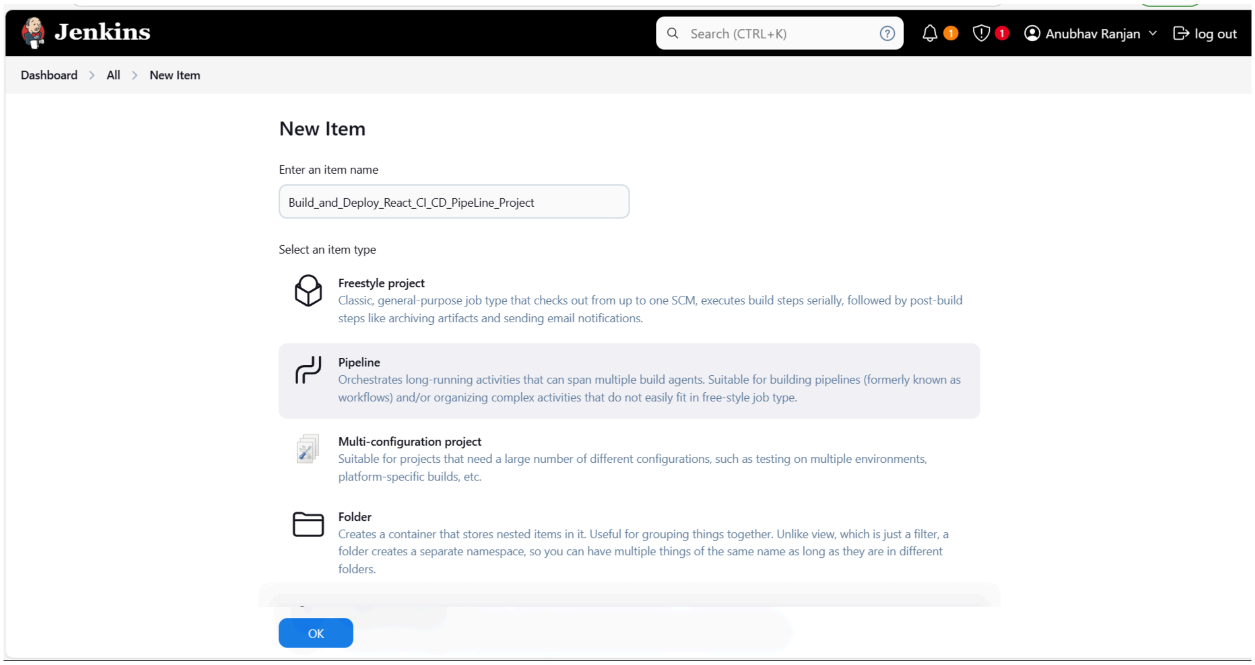
anubhav

Password

••••••••••

□ Keep me signed in

2. **Create a New Pipeline Job:**
   - Click on **“New Item”** from the Jenkins dashboard.
   - Enter a name for the job, i.e. `Build_and_Deploy_React_CI_CD_PipeLine_Project`
   - Select **“Pipeline”** and click **OK**.



Dashboard > All > New Item

New Item

Enter an item name

Build_and_Deploy_React_CI_CD_PipeLine_Project

Select an item type

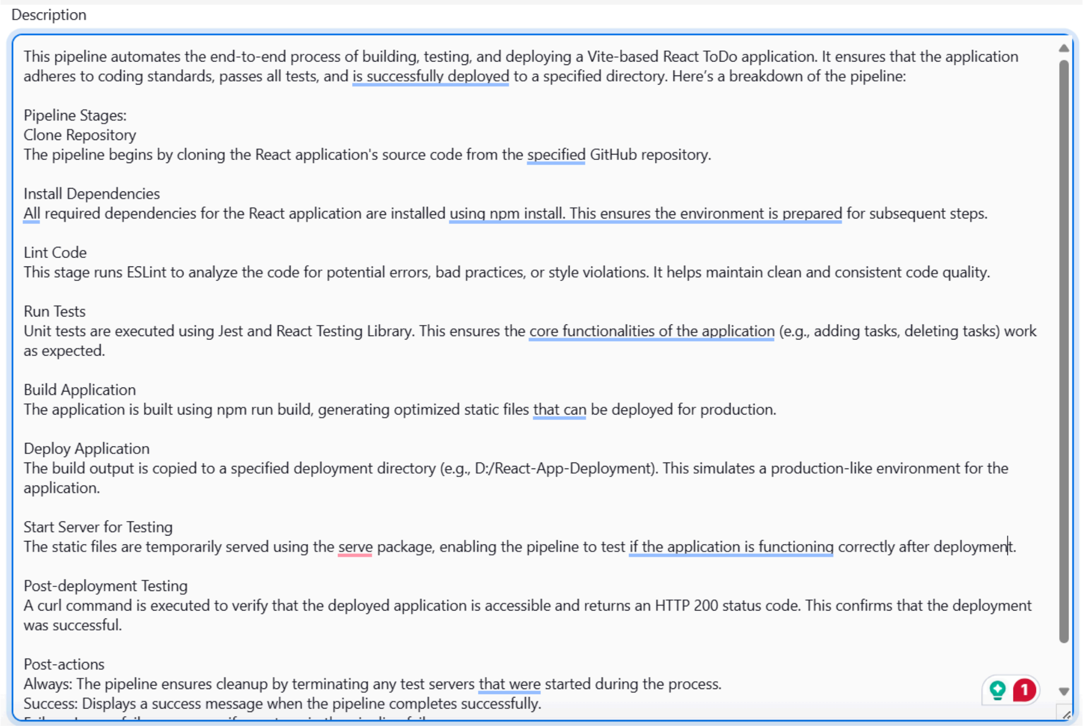
Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

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.

3. **Add a Description :**
   - On the job configuration page, add a brief description such as:



Description

This pipeline automates the end-to-end process of building, testing, and deploying a Vite-based React ToDo application. It ensures that the application adheres to coding standards, passes all tests, and is successfully deployed to a specified directory. Here's a breakdown of the pipeline:

Pipeline Stages:
Clone Repository
The pipeline begins by cloning the React application's source code from the specified GitHub repository.

Install Dependencies
All required dependencies for the React application are installed using npm install. This ensures the environment is prepared for subsequent steps.

Lint Code
This stage runs ESLint to analyze the code for potential errors, bad practices, or style violations. It helps maintain clean and consistent code quality.

Run Tests
Unit tests are executed using Jest and React Testing Library. This ensures the core functionalities of the application (e.g., adding tasks, deleting tasks) work as expected.

Build Application
The application is built using npm run build, generating optimized static files that can be deployed for production.

Deploy Application
The build output is copied to a specified deployment directory (e.g., D:/React-App-Deployment). This simulates a production-like environment for the application.

Start Server for Testing
The static files are temporarily served using the serve package, enabling the pipeline to test if the application is functioning correctly after deployment.

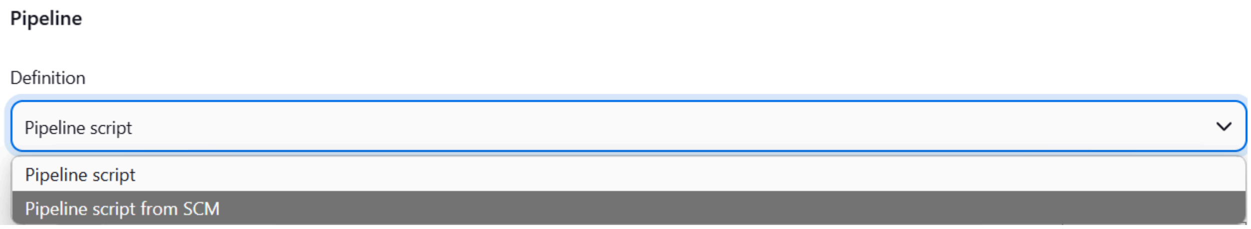
Post-deployment Testing
A curl command is executed to verify that the deployed application is accessible and returns an HTTP 200 status code. This confirms that the deployment was successful.

Post-actions
Always: The pipeline ensures cleanup by terminating any test servers that were started during the process.
Success: Displays a success message when the pipeline completes successfully.

Step 3: Configure the Job

Step 3.1: Import the Jenkinsfile from GitHub Repository

1. In the job configuration page, go to the **Pipeline** section.
2. Choose **Pipeline script from SCM** if your `Jenkinsfile` is stored in your GitHub repo, or use **Pipeline script** to write it directly in Jenkins.

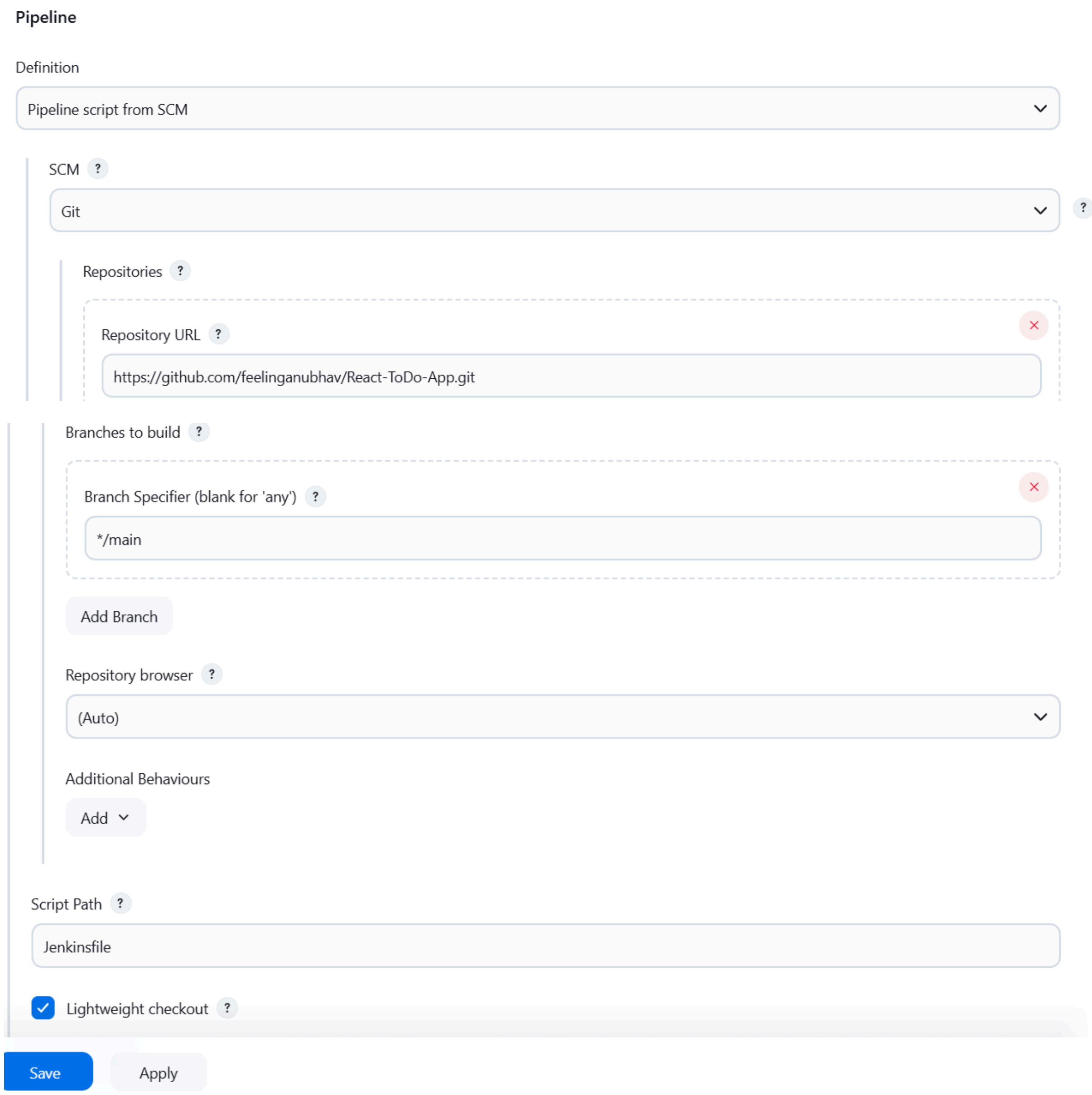


Pipeline

Definition

Pipeline script

3. Select **Git** and enter the repository URL (i.e., `https://github.com/feelinganubhav/React-ToDo-App.git`).
4. (Optional) If the repository is private, add your credentials.
5. **Save Your Job:** Scroll to the bottom and click "Save".



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/feelinganubhav/React-ToDo-App.git

Branches to build

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

Repository browser: (Auto)

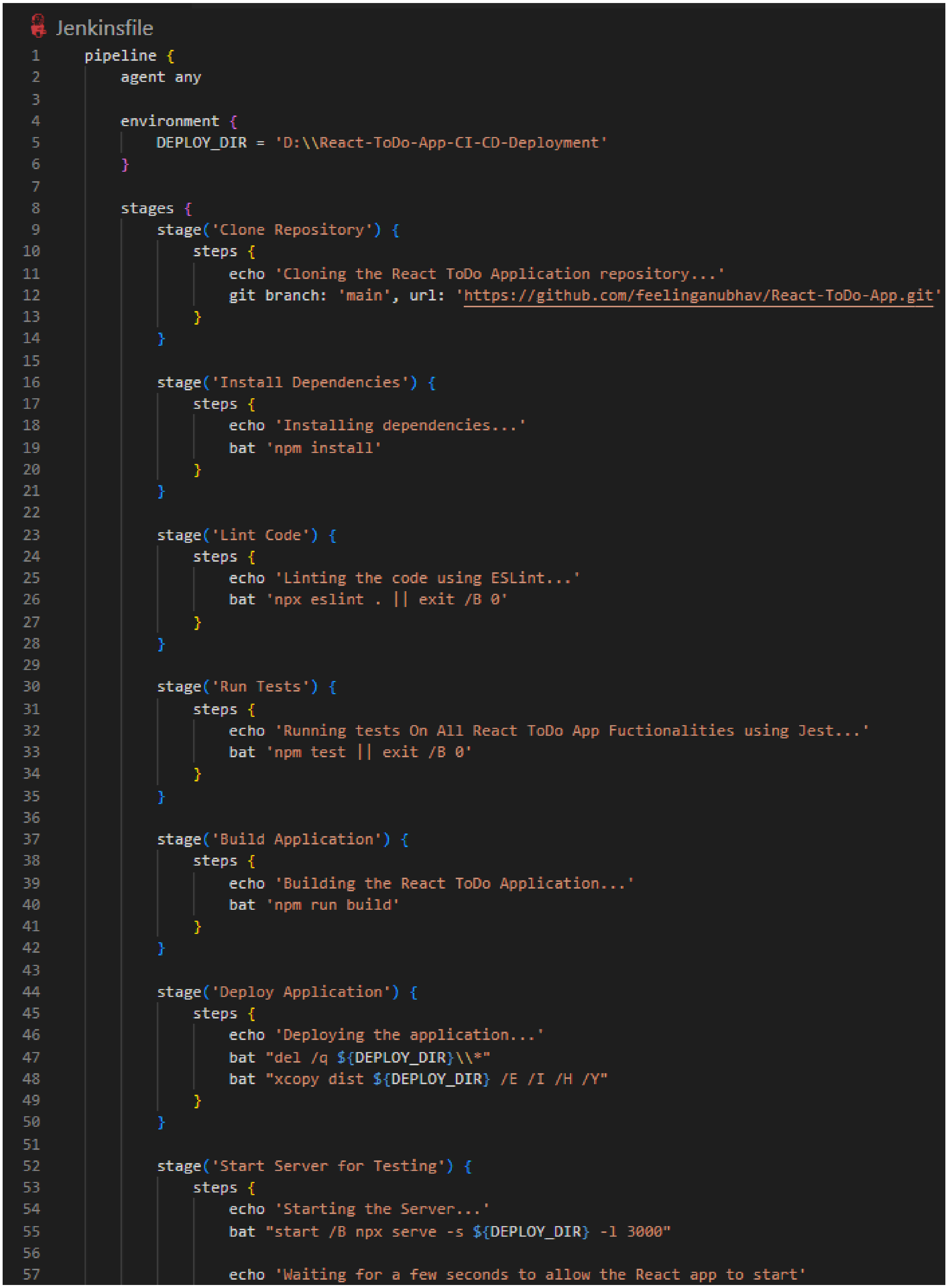
Additional Behaviours

Script Path: Jenkinsfile

☑ Lightweight checkout

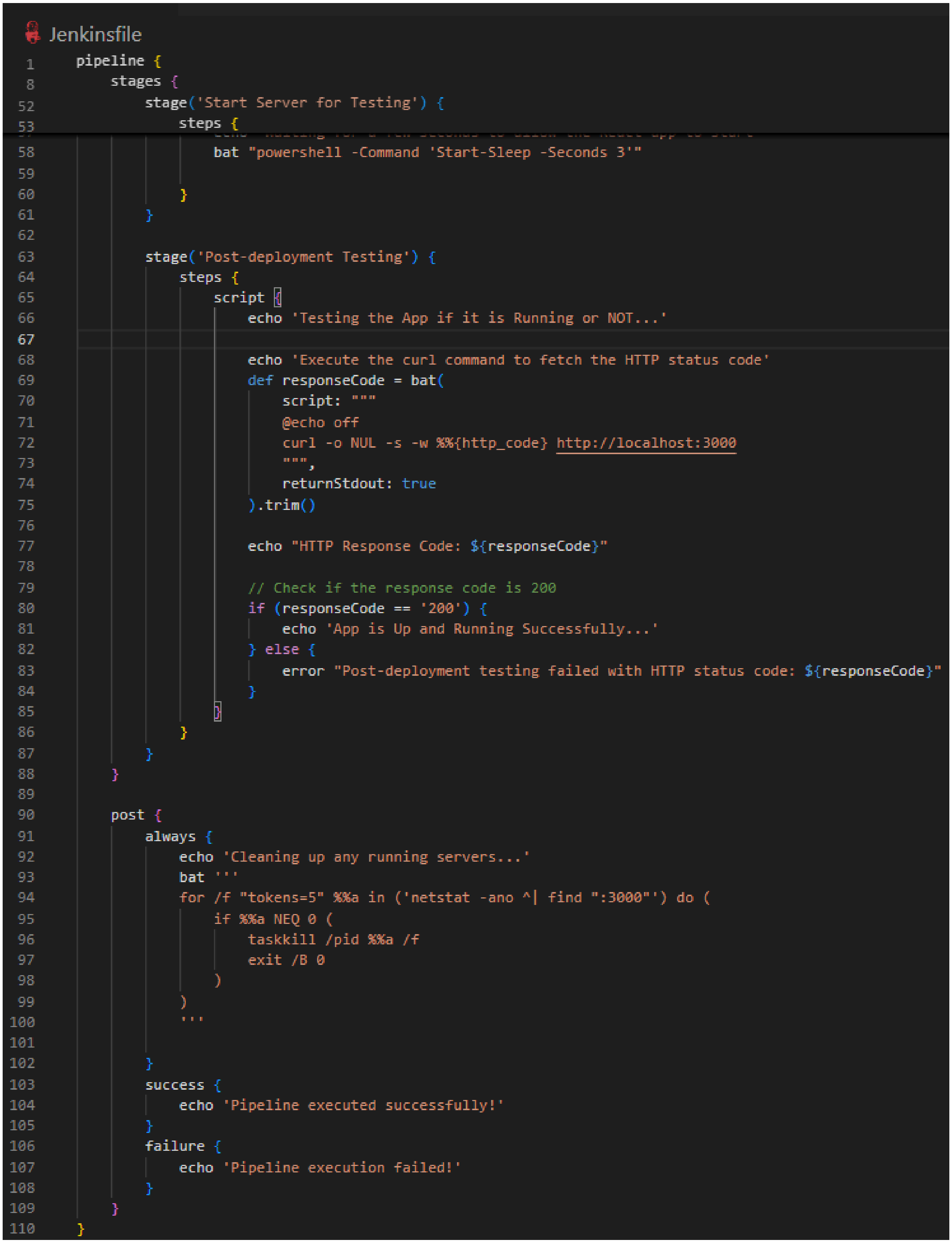
Step 3.2: Write the Jenkinsfile

1. Open **VS Code**, and Add the File **“Jenkinsfile”** in your root directory.
2. Below is the `Jenkinsfile` for your pipeline. This file defines all stages of the CI/CD process:



Jenkinsfile

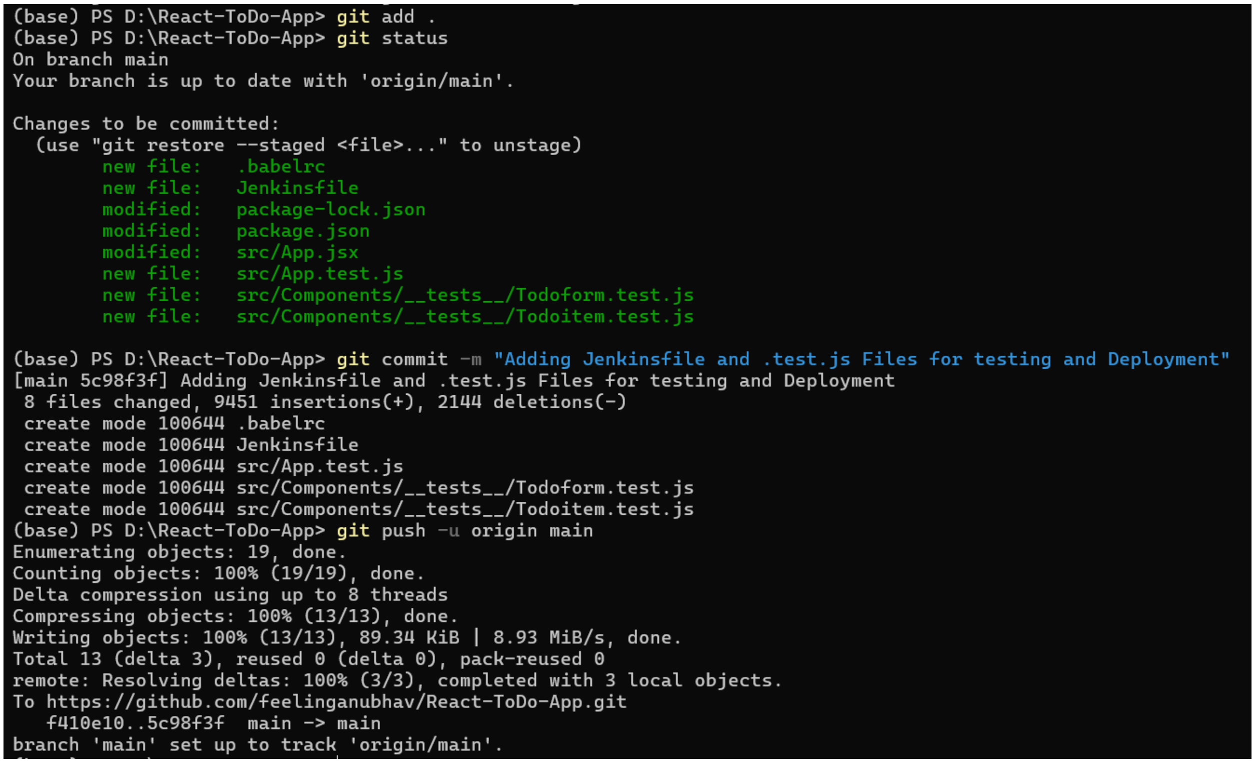
```
pipeline {
    agent any

    environment {
        DEPLOY_DIR = 'D:\\React-ToDo-App-CI-CD-Deployment'
    }

    stages {
        stage('Clone Repository') {
            steps {
                echo 'Cloning the React ToDo Application repository...'
                git branch: 'main', url: 'https://github.com/feelinganubhav/React-ToDo-App.git'
            }
        }

        stage('Install Dependencies') {
            steps {
                echo 'Installing dependencies...'
                bat 'npm install'
            }
        }

        stage('Lint Code') {
            steps {
                echo 'Linting the code using ESLint...'
                bat 'npx eslint . || exit /B 0'
            }
        }

        stage('Run Tests') {
            steps {
                echo 'Running tests On All React ToDo App Fuctionalities using Jest...'
                bat 'npm test || exit /B 0'
            }
        }

        stage('Build Application') {
            steps {
                echo 'Building the React ToDo Application...'
                bat 'npm run build'
            }
        }

        stage('Deploy Application') {
            steps {
                echo 'Deploying the application...'
                bat "del /q ${DEPLOY_DIR}\\*"
                bat "xcopy dist ${DEPLOY_DIR} /E /I /H /Y"
            }
        }

        stage('Start Server for Testing') {
            steps {
                echo 'Starting the Server...'
                bat "start /B npx serve -s ${DEPLOY_DIR} -l 3000"

                echo 'Waiting for a few seconds to allow the React app to start'
```



```
pipeline {
    stages {
        stage('Start Server for Testing') {
            steps {
                bat "powershell -Command 'Start-Sleep -Seconds 3'"

            }
        }

        stage('Post-deployment Testing') {
            steps {
                script {
                    echo 'Testing the App if it is Running or NOT...'

                    echo 'Execute the curl command to fetch the HTTP status code'
                    def responseCode = bat(
                        script: """
                        @echo off
                        curl -o NUL -s -w %%{http_code} http://localhost:3000
                        """,
                        returnStdout: true
                    ).trim()

                    echo "HTTP Response Code: ${responseCode}"

                    // Check if the response code is 200
                    if (responseCode == '200') {
                        echo 'App is Up and Running Successfully...'
                    } else {
                        error "Post-deployment testing failed with HTTP status code: ${responseCode}"
                    }
                }
            }
        }
    }

    post {
        always {
            echo 'Cleaning up any running servers...'
            bat '''
            for /f "tokens=5" %%a in ('netstat -ano ^| find ":3000"') do (
                if %%a NEQ 0 (
                    taskkill /pid %%a /f
                    exit /B 0
                )
            )
            '''

        }
        success {
            echo 'Pipeline executed successfully!'
        }
        failure {
            echo 'Pipeline execution failed!'
        }
    }
}
```

Step 3.3: Save the Jenkinsfile

1. Save the Jenkinsfile along with .test.js Files for Testing using Jest and Push it to your GitHub Repository.



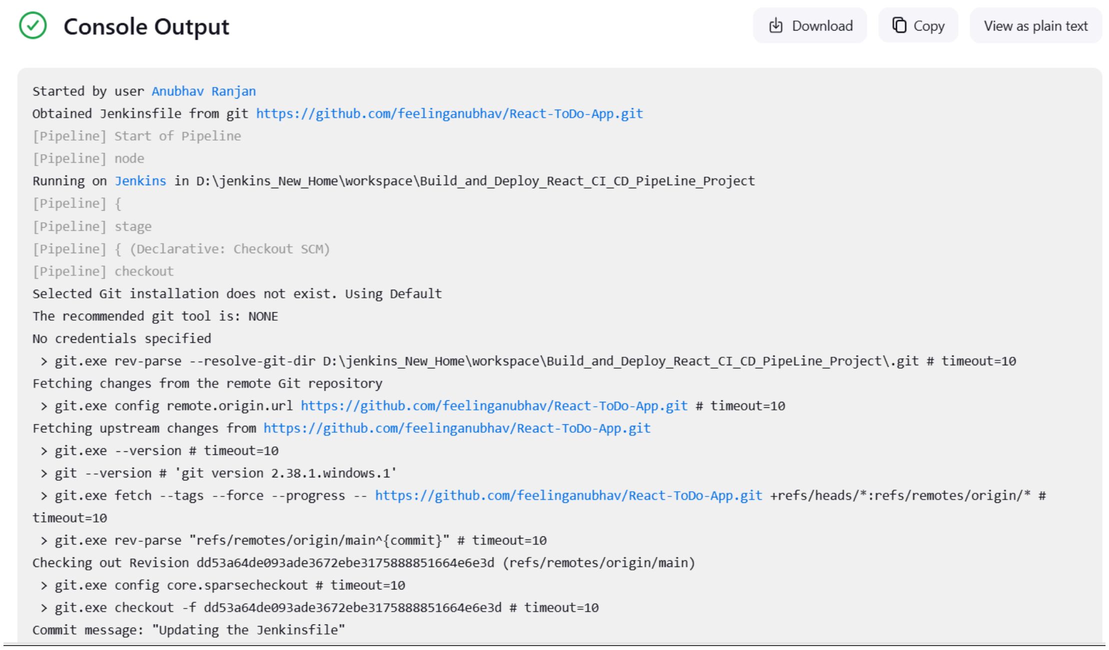
```
(base) PS D:\React-ToDo-App> git add .
(base) PS D:\React-ToDo-App> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .babelrc
        new file:   Jenkinsfile
        modified:   package-lock.json
        modified:   package.json
        modified:   src/App.jsx
        new file:   src/App.test.js
        new file:   src/Components/__tests__/Todoform.test.js
        new file:   src/Components/__tests__/Todoitem.test.js

(base) PS D:\React-ToDo-App> git commit -m "Adding Jenkinsfile and .test.js Files for testing and Deployment"
[main 5c98f3f] Adding Jenkinsfile and .test.js Files for testing and Deployment
 8 files changed, 9451 insertions(+), 2144 deletions(-)
 create mode 100644 .babelrc
 create mode 100644 Jenkinsfile
 create mode 100644 src/App.test.js
 create mode 100644 src/Components/__tests__/Todoform.test.js
 create mode 100644 src/Components/__tests__/Todoitem.test.js
(base) PS D:\React-ToDo-App> git push -u origin main
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 8 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (13/13), 89.34 KiB | 8.93 MiB/s, done.
Total 13 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/feelinganubhav/React-ToDo-App.git
   f410e10..5c98f3f  main -> main
branch 'main' set up to track 'origin/main'.
```

2. If your repository contains the `Jenkinsfile`, save it in the root of the project.
3. Otherwise, paste it into the **Pipeline script** section in Jenkins.

Step 5: Run and Test the Job

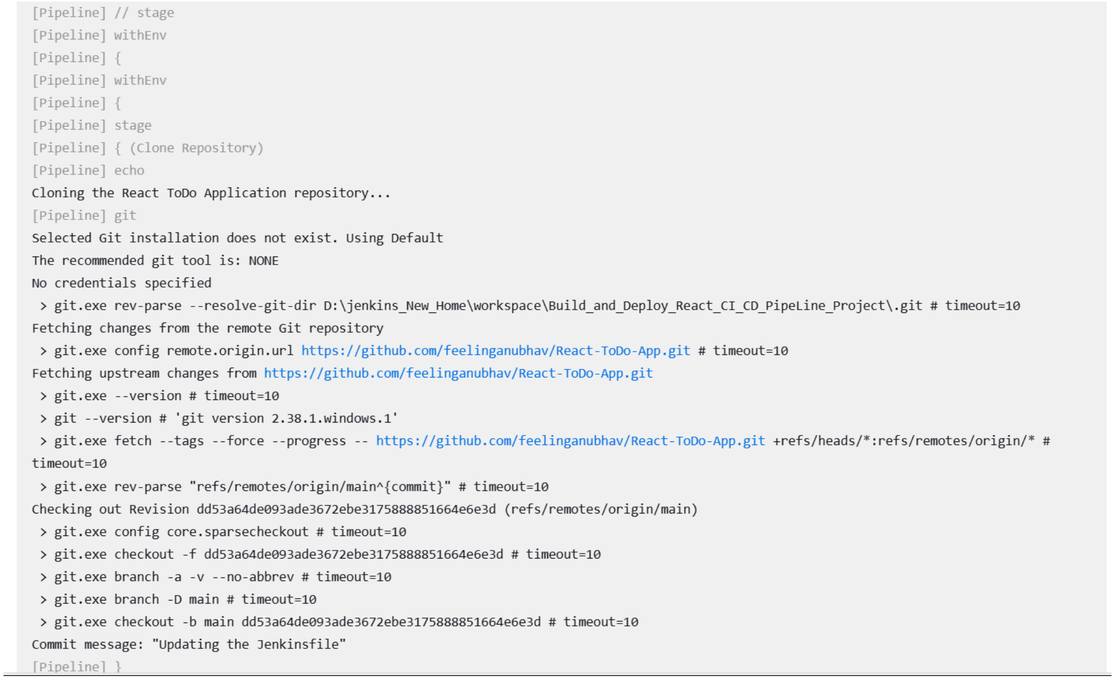
1. **Run the Job:**
    - Go to the job's dashboard.
    - Click “Build Now” to execute the **Pipeline** Job.
2. **Verify the Output:**
    - Watch the logs to monitor the progress of each stage.

- Validate that each stage completes successfully:
    - Import the Jenkinsfile from the Repo.



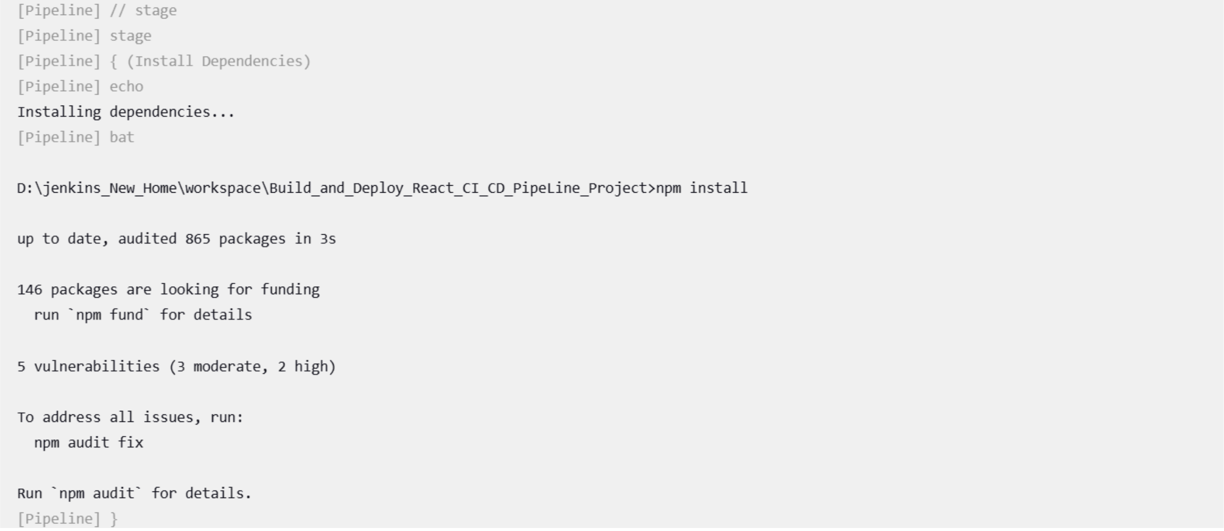
```
Started by user Anubhav Ranjan
Obtained Jenkinsfile from git https://github.com/feelinganubhav/React-ToDo-App.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/feelinganubhav/React-ToDo-App.git # timeout=10
Fetching upstream changes from https://github.com/feelinganubhav/React-ToDo-App.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.38.1.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/feelinganubhav/React-ToDo-App.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision dd53a64de093ade3672ebe3175888851664e6e3d (refs/remotes/origin/main)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f dd53a64de093ade3672ebe3175888851664e6e3d # timeout=10
Commit message: "Updating the Jenkinsfile"
```

- Clone the repository.



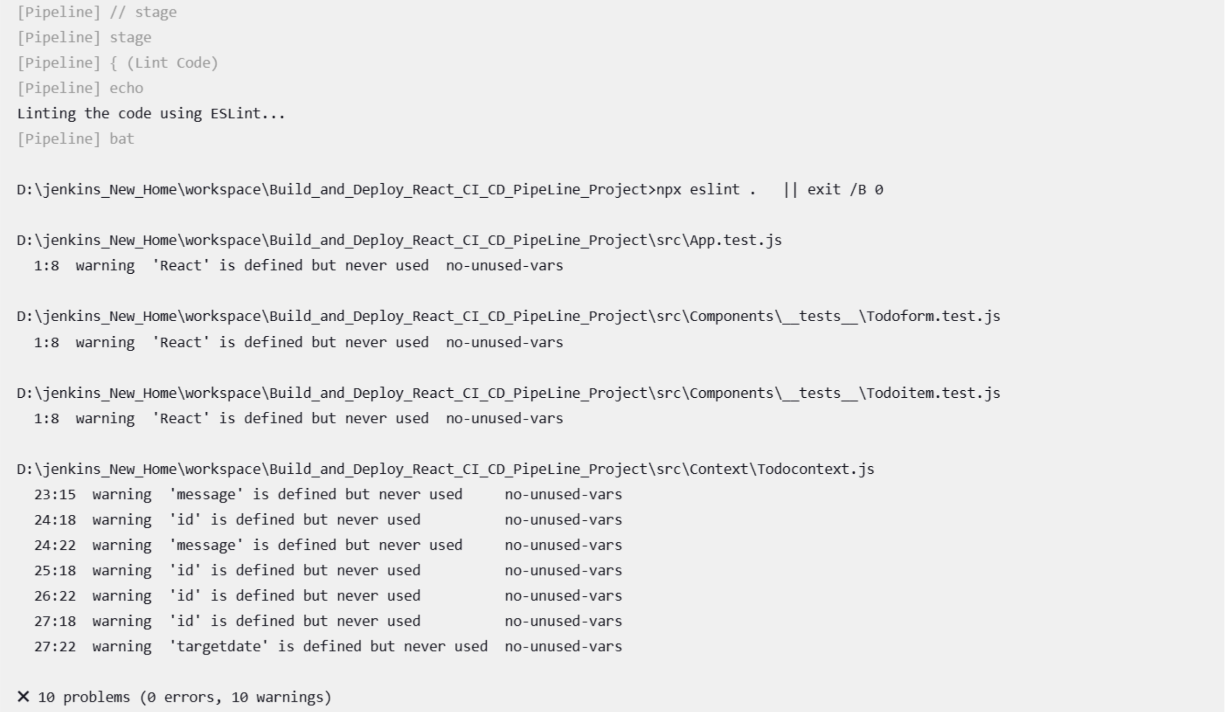
```
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Clone Repository)
[Pipeline] echo
Cloning the React ToDo Application repository...
[Pipeline] git
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/feelinganubhav/React-ToDo-App.git # timeout=10
Fetching upstream changes from https://github.com/feelinganubhav/React-ToDo-App.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.38.1.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/feelinganubhav/React-ToDo-App.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision dd53a64de093ade3672ebe3175888851664e6e3d (refs/remotes/origin/main)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f dd53a64de093ade3672ebe3175888851664e6e3d # timeout=10
 > git.exe branch -a -v --no-abbrev # timeout=10
 > git.exe branch -D main # timeout=10
 > git.exe checkout -b main dd53a64de093ade3672ebe3175888851664e6e3d # timeout=10
Commit message: "Updating the Jenkinsfile"
[Pipeline] }
```

- Install dependencies.



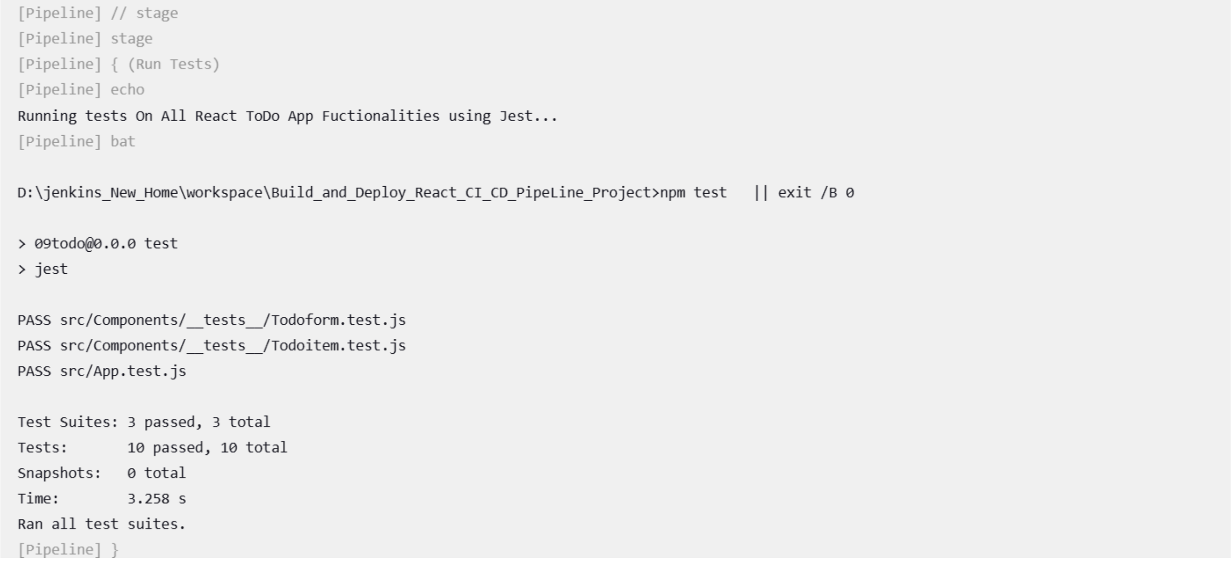
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Install Dependencies)
[Pipeline] echo
Installing dependencies...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>npm install

up to date, audited 865 packages in 3s

146 packages are looking for funding
  run `npm fund` for details

5 vulnerabilities (3 moderate, 2 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
[Pipeline] }
```

- Run ESLint (verify linting errors/warnings in the logs).



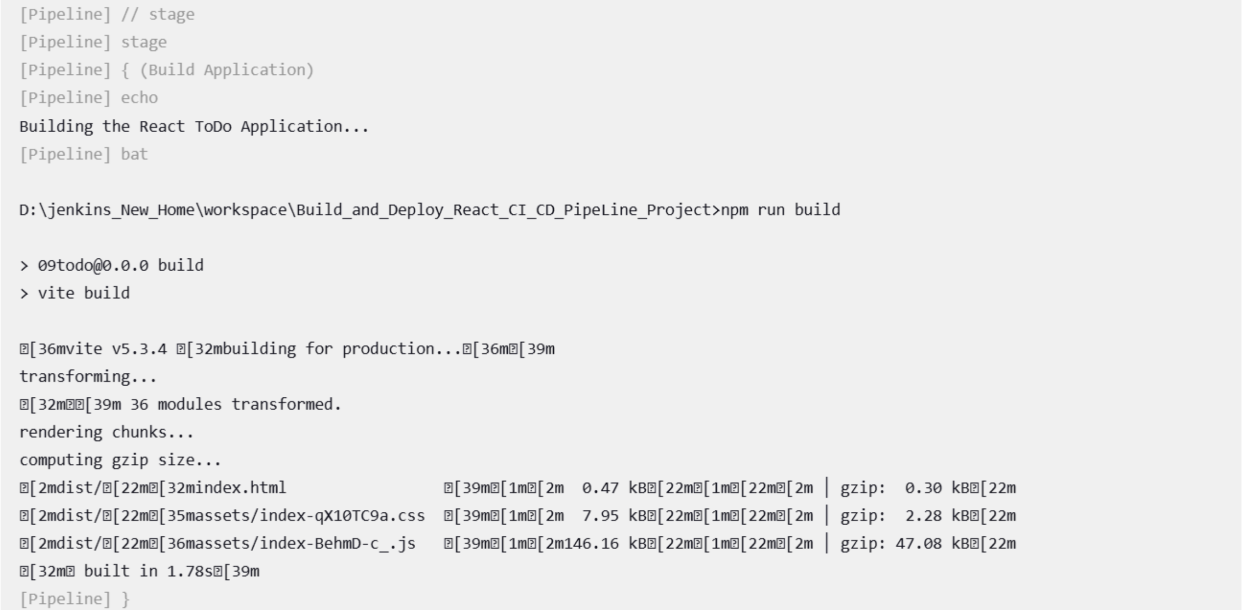
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Lint Code)
[Pipeline] echo
Linting the code using ESLint...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>npx eslint .   || exit /B 0

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\src\App.test.js
  1:8  warning  'React' is defined but never used  no-unused-vars

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\src\Components\__tests__\Todoform.test.js
  1:8  warning  'React' is defined but never used  no-unused-vars

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\src\Components\__tests__\Todoitem.test.js
  1:8  warning  'React' is defined but never used  no-unused-vars

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project\src\Context\Todocontext.js
  23:15  warning  'message' is defined but never used     no-unused-vars
  24:18  warning  'id' is defined but never used          no-unused-vars
  24:22  warning  'message' is defined but never used     no-unused-vars
  25:18  warning  'id' is defined but never used          no-unused-vars
  26:22  warning  'id' is defined but never used          no-unused-vars
  27:18  warning  'id' is defined but never used          no-unused-vars
  27:22  warning  'targetdate' is defined but never used  no-unused-vars

✖ 10 problems (0 errors, 10 warnings)
```

- Run tests on All the Functions of React ToDo App using the Jest Testing Framework(all tests should pass).



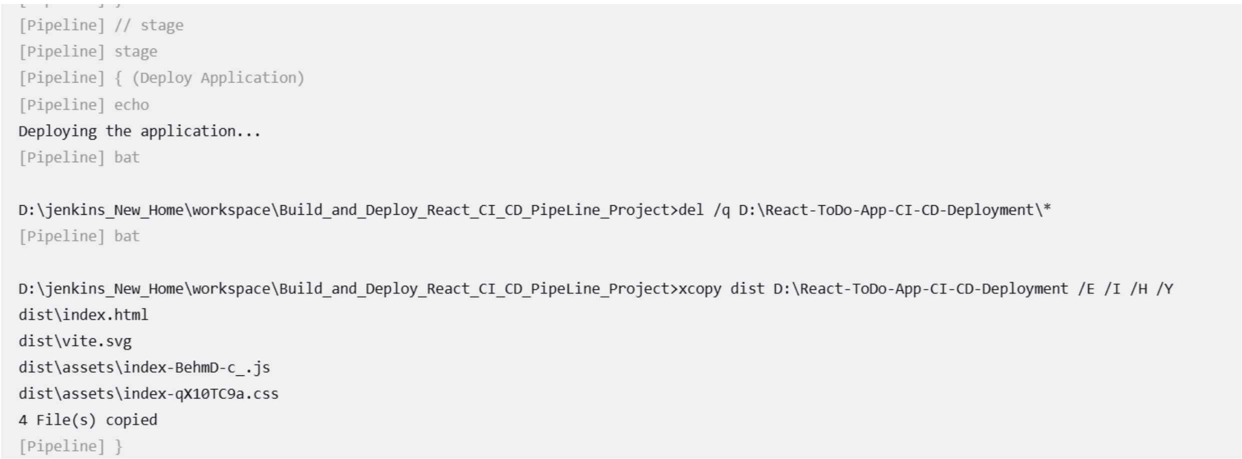
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Run Tests)
[Pipeline] echo
Running tests On All React ToDo App Fuctionalities using Jest...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>npm test   || exit /B 0

> 09todo@0.0.0 test
> jest

PASS src/Components/__tests__/Todoform.test.js
PASS src/Components/__tests__/Todoitem.test.js
PASS src/App.test.js

Test Suites: 3 passed, 3 total
Tests:       10 passed, 10 total
Snapshots:   0 total
Time:        3.258 s
Ran all test suites.
[Pipeline] }
```

- Build the project.



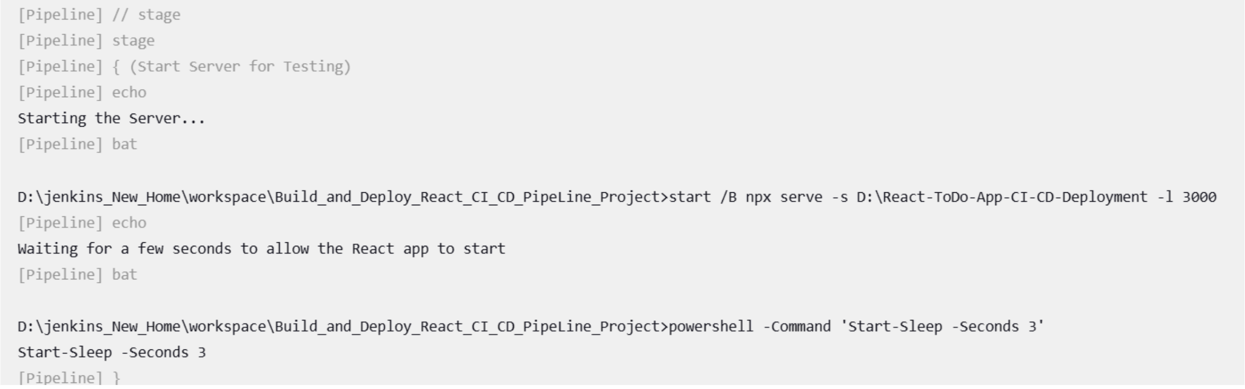
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Application)
[Pipeline] echo
Building the React ToDo Application...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>npm run build

> 09todo@0.0.0 build
> vite build

⍰[36mvite v5.3.4 ⍰[32mbuilding for production...⍰[36m⍰[39m
transforming...
⍰[32m⍰⍰[39m 36 modules transformed.
rendering chunks...
computing gzip size...
⍰[2mdist/⍰[22m⍰[32mindex.html                 ⍰[39m⍰[1m⍰[2m  0.47 kB⍰[22m⍰[1m⍰[22m⍰[2m │ gzip:  0.30 kB⍰[22m
⍰[2mdist/⍰[22m⍰[35massets/index-qX10TC9a.css  ⍰[39m⍰[1m⍰[2m  7.95 kB⍰[22m⍰[1m⍰[22m⍰[2m │ gzip:  2.28 kB⍰[22m
⍰[2mdist/⍰[22m⍰[36massets/index-BehmD-c_.js   ⍰[39m⍰[1m⍰[2m146.16 kB⍰[22m⍰[1m⍰[22m⍰[2m │ gzip: 47.08 kB⍰[22m
⍰[32m⍰ built in 1.78s⍰[39m
[Pipeline] }
```

- Deploy the app to the deployment directory.



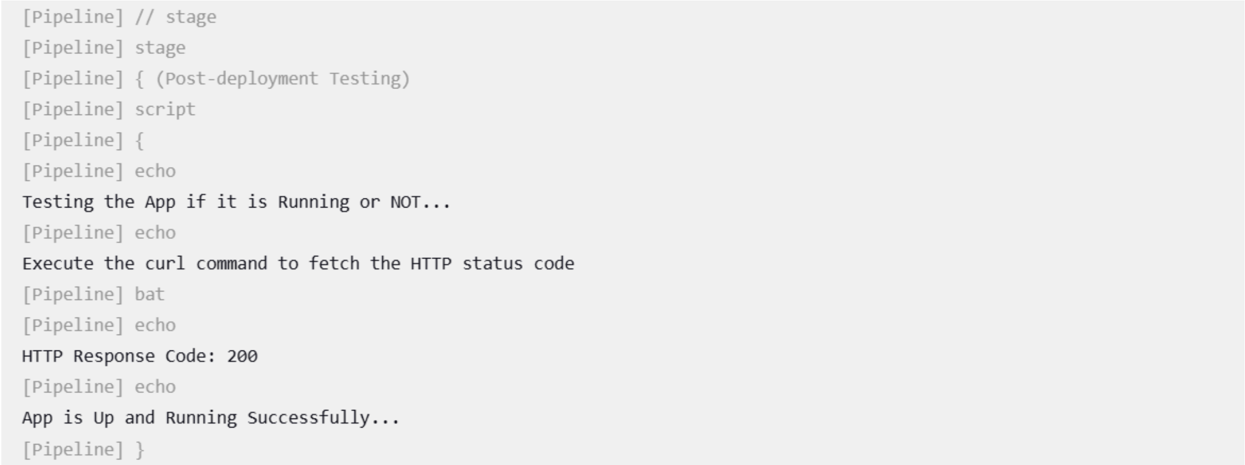
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy Application)
[Pipeline] echo
Deploying the application...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>del /q D:\React-ToDo-App-CI-CD-Deployment\*
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>xcopy dist D:\React-ToDo-App-CI-CD-Deployment /E /I /H /Y
dist\index.html
dist\vite.svg
dist\assets\index-BehmD-c_.js
dist\assets\index-qX10TC9a.css
4 File(s) copied
[Pipeline] }
```

- Start the Server for Deployment Testing.



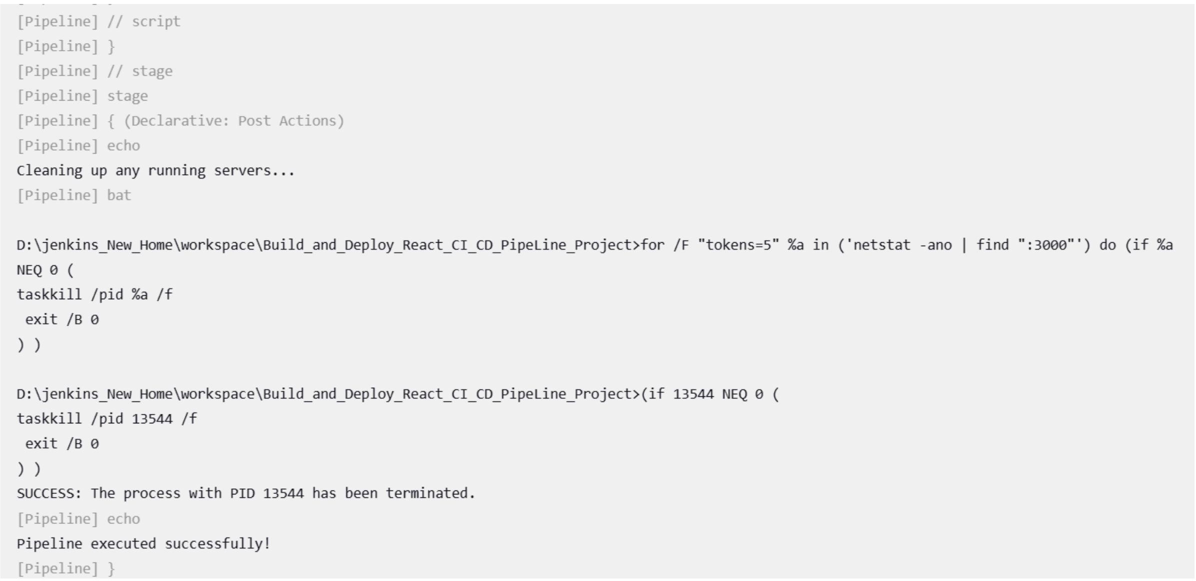
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Start Server for Testing)
[Pipeline] echo
Starting the Server...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>start /B npx serve -s D:\React-ToDo-App-CI-CD-Deployment -l 3000
[Pipeline] echo
Waiting for a few seconds to allow the React app to start
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>powershell -Command 'Start-Sleep -Seconds 3'
Start-Sleep -Seconds 3
[Pipeline] }
```

- Post Deployment Testing: Use `curl` to verify the app is running.



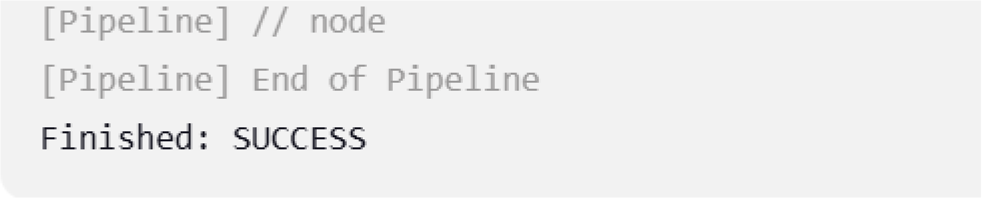
```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Post-deployment Testing)
[Pipeline] script
[Pipeline] {
[Pipeline] echo
Testing the App if it is Running or NOT...
[Pipeline] echo
Execute the curl command to fetch the HTTP status code
[Pipeline] bat
[Pipeline] echo
HTTP Response Code: 200
[Pipeline] echo
App is Up and Running Successfully...
[Pipeline] }
```

- Clean up the Running Servers..



```
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Cleaning up any running servers...
[Pipeline] bat

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>for /F "tokens=5" %a in ('netstat -ano | find ":3000"') do (if %a
NEQ 0 (
taskkill /pid %a /f
 exit /B 0
) )

D:\jenkins_New_Home\workspace\Build_and_Deploy_React_CI_CD_PipeLine_Project>(if 13544 NEQ 0 (
taskkill /pid 13544 /f
 exit /B 0
) )
SUCCESS: The process with PID 13544 has been terminated.
[Pipeline] echo
Pipeline executed successfully!
[Pipeline] }
```

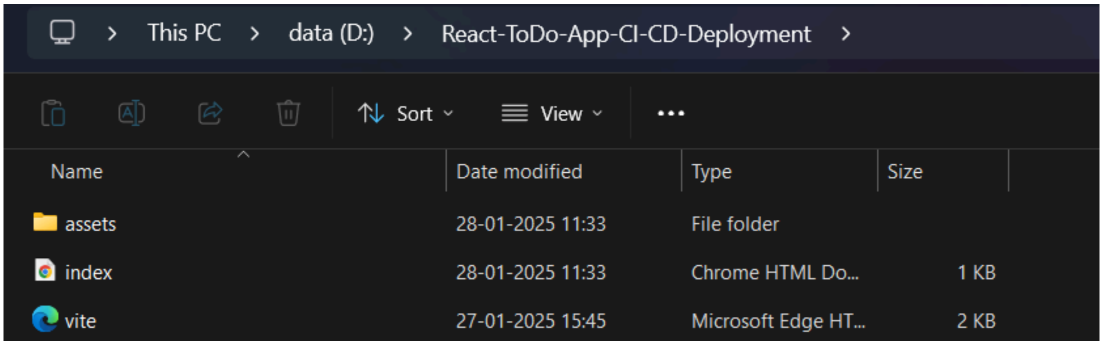
- Check the Console Output in Jenkins for the success message:



```
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Step 6: Verify the Deployment

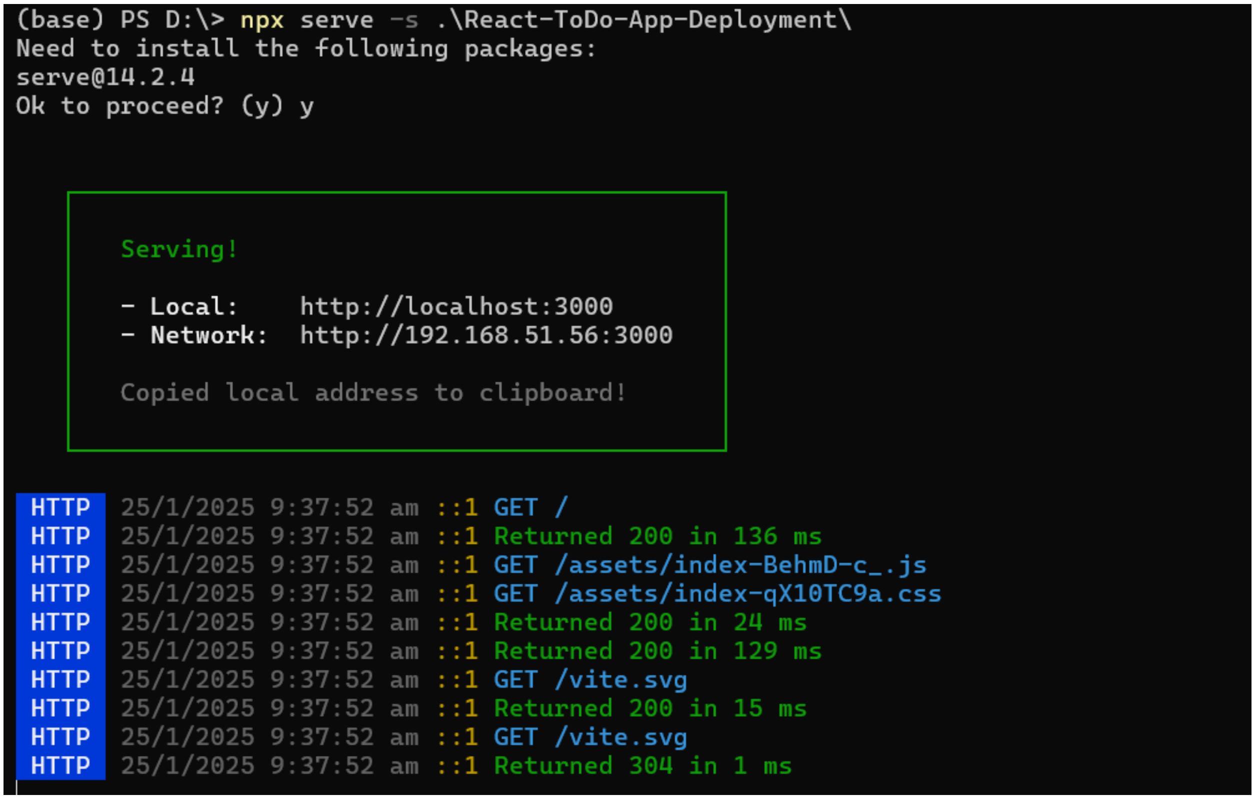
1. Navigate to the deployment directory: `"D:\React-ToDo-App-CI-CD-Deployment"`
2. Check if the `React-ToDo-App-CI-CD-Deployment` folder contains the files.



Name	Date modified	Type	Size
assets	28-01-2025 11:33	File folder	
index	28-01-2025 11:33	Chrome HTML Do...	1 KB
vite	27-01-2025 15:45	Microsoft Edge HT...	2 KB

3. Host the `React-ToDo-App-CI-CD-Deployment` folder using a simple HTTP server :

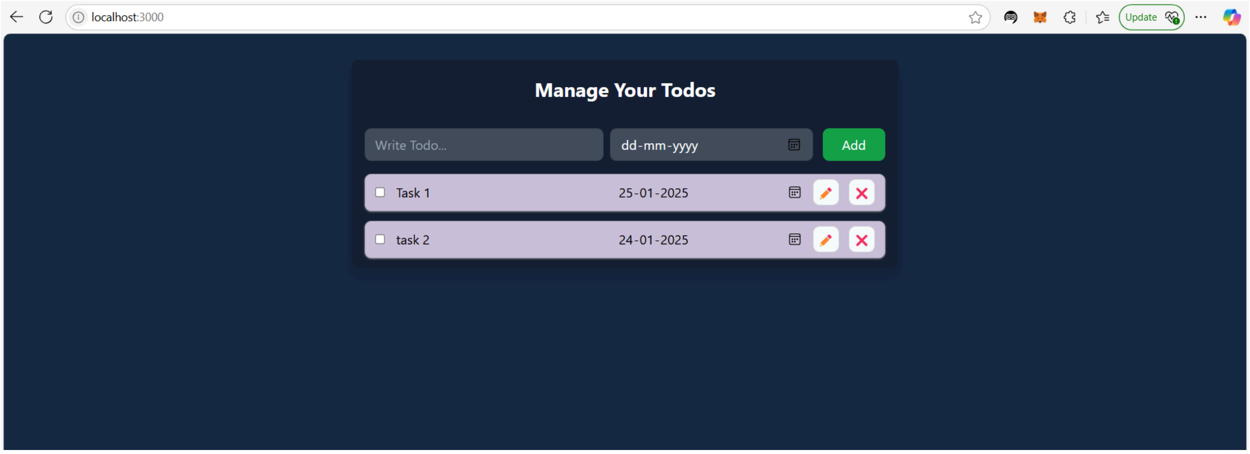
   ```
   npx serve -s React-ToDo-App-CI-CD-Deployment
   ```



```
(base) PS D:\> npx serve -s .\React-ToDo-App-Deployment\
Need to install the following packages:
serve@14.2.4
Ok to proceed? (y) y


   ┌──────────────────────────────────────────┐
   │                                          │
   │   Serving!                               │
   │                                          │
   │   - Local:    http://localhost:3000      │
   │   - Network:  http://192.168.51.56:3000  │
   │                                          │
   │   Copied local address to clipboard!     │
   │                                          │
   └──────────────────────────────────────────┘

 HTTP  25/1/2025 9:37:52 am ::1 GET /
 HTTP  25/1/2025 9:37:52 am ::1 Returned 200 in 136 ms
 HTTP  25/1/2025 9:37:52 am ::1 GET /assets/index-BehmD-c_.js
 HTTP  25/1/2025 9:37:52 am ::1 GET /assets/index-qX10TC9a.css
 HTTP  25/1/2025 9:37:52 am ::1 Returned 200 in 24 ms
 HTTP  25/1/2025 9:37:52 am ::1 Returned 200 in 129 ms
 HTTP  25/1/2025 9:37:52 am ::1 GET /vite.svg
 HTTP  25/1/2025 9:37:52 am ::1 Returned 200 in 15 ms
 HTTP  25/1/2025 9:37:52 am ::1 GET /vite.svg
 HTTP  25/1/2025 9:37:52 am ::1 Returned 304 in 1 ms
```

4. Access the web application in your browser at:

   ```
   http://<server-ip>:<port>
   ```

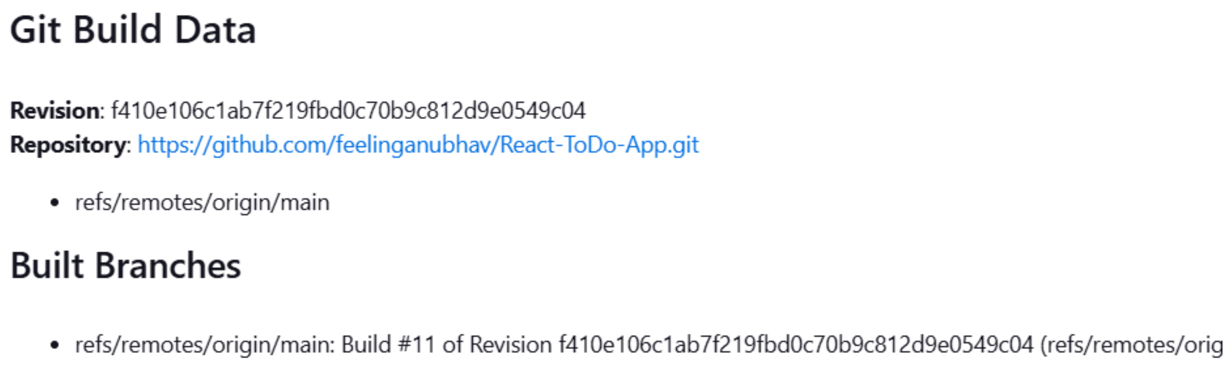


localhost:3000

Manage Your Todos

□ Task 1 25-01-2025

□ task 2 24-01-2025



Git Build Data

Revision: f410e106c1ab7f219fbd0c70b9c812d9e0549c04
Repository: https://github.com/feelinganubhav/React-ToDo-App.git

- refs/remotes/origin/main

Built Branches

- refs/remotes/origin/main: Build #11 of Revision f410e106c1ab7f219fbd0c70b9c812d9e0549c04 (refs/remotes/orig