**Lab Experiment 2: Creating a Jenkins Pipeline with a Jenkinsfile**

**Objective: Create a Jenkins pipeline using a Jenkinsfile that builds a simple project, runs tests, and deploys the project to a designated environment.**

**Prerequisites:**

1. Jenkins server up and running.
2. A sample project hosted in a version control repository (e.g., Git).

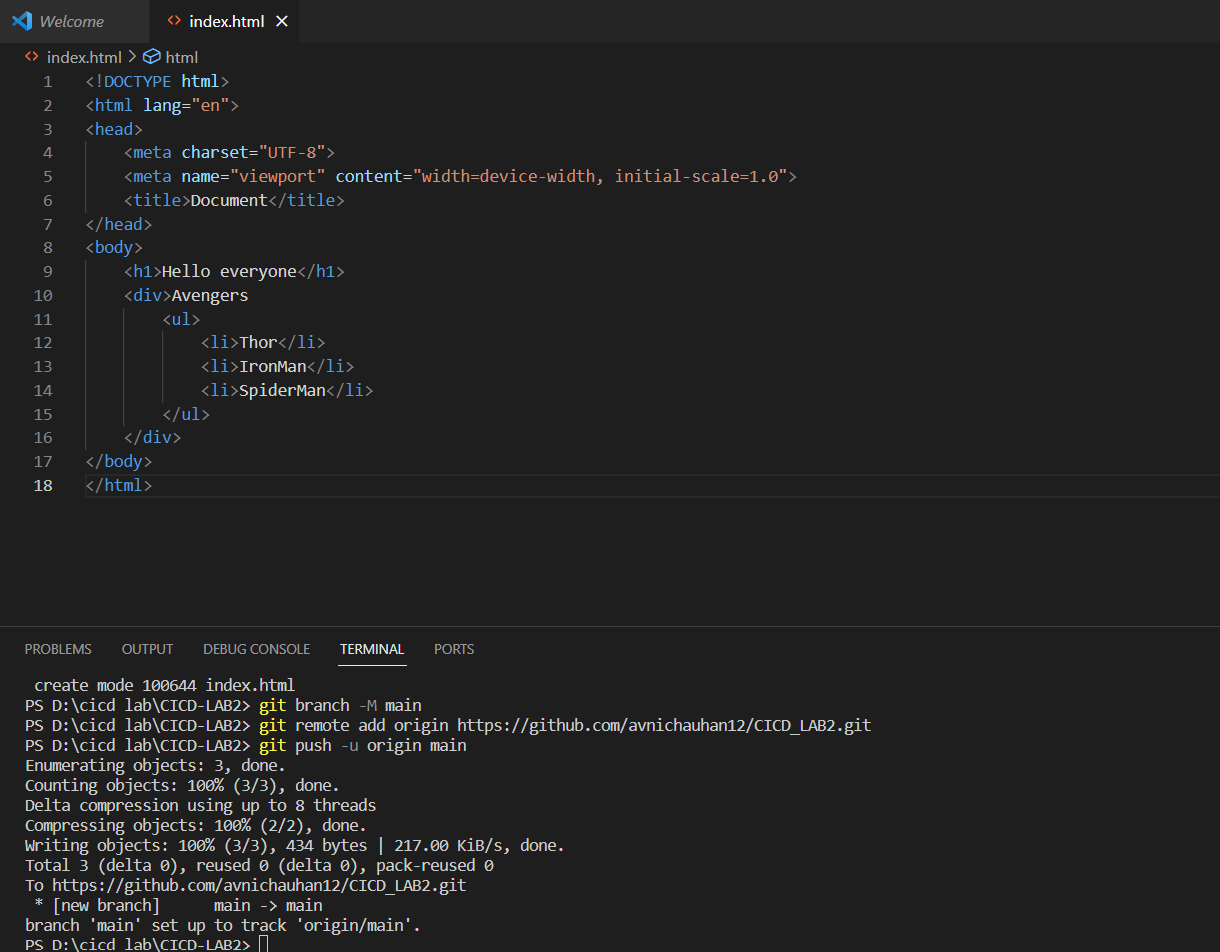
**Steps:**

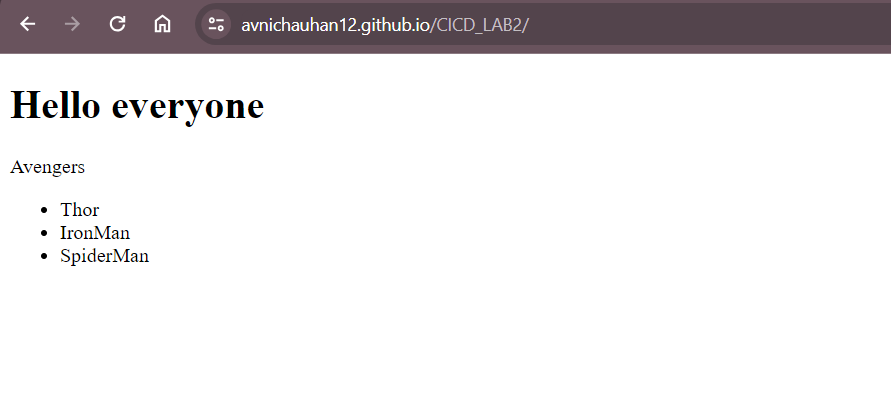
**Jenkins Configuration:**

* Ensure that Jenkins is installed and accessible.
* Install necessary plugins: Pipeline and any plugins specific to your version control system (e.g., Git Plugin).

**Setting Up the Project:**

* Create a sample project (e.g., a simple web application) and host it on a version control repository (e.g., GitHub).





Link of website - <https://avnichauhan12.github.io/CICD_LAB2/>

**Creating a Jenkinsfile:**

In the root of your project repository, create a file named Jenkinsfile.

**Defining the Pipeline:**

Open the Jenkinsfile and define the pipeline stages using the declarative pipeline syntax.

Here's an example Jenkinsfile with basic stages:

pipeline {

agent any

stages {

stage('Checkout') {

steps {

checkout scm

}

}

stage('Build') {

steps {

echo 'build'

}

}

stage('Test') {

steps {

echo 'No tests defined for HTML content'

}

}

}

post {

success {

echo 'Pipeline succeeded! Project built and deployed.'

}

failure {

echo 'Pipeline failed! Check logs for details.'

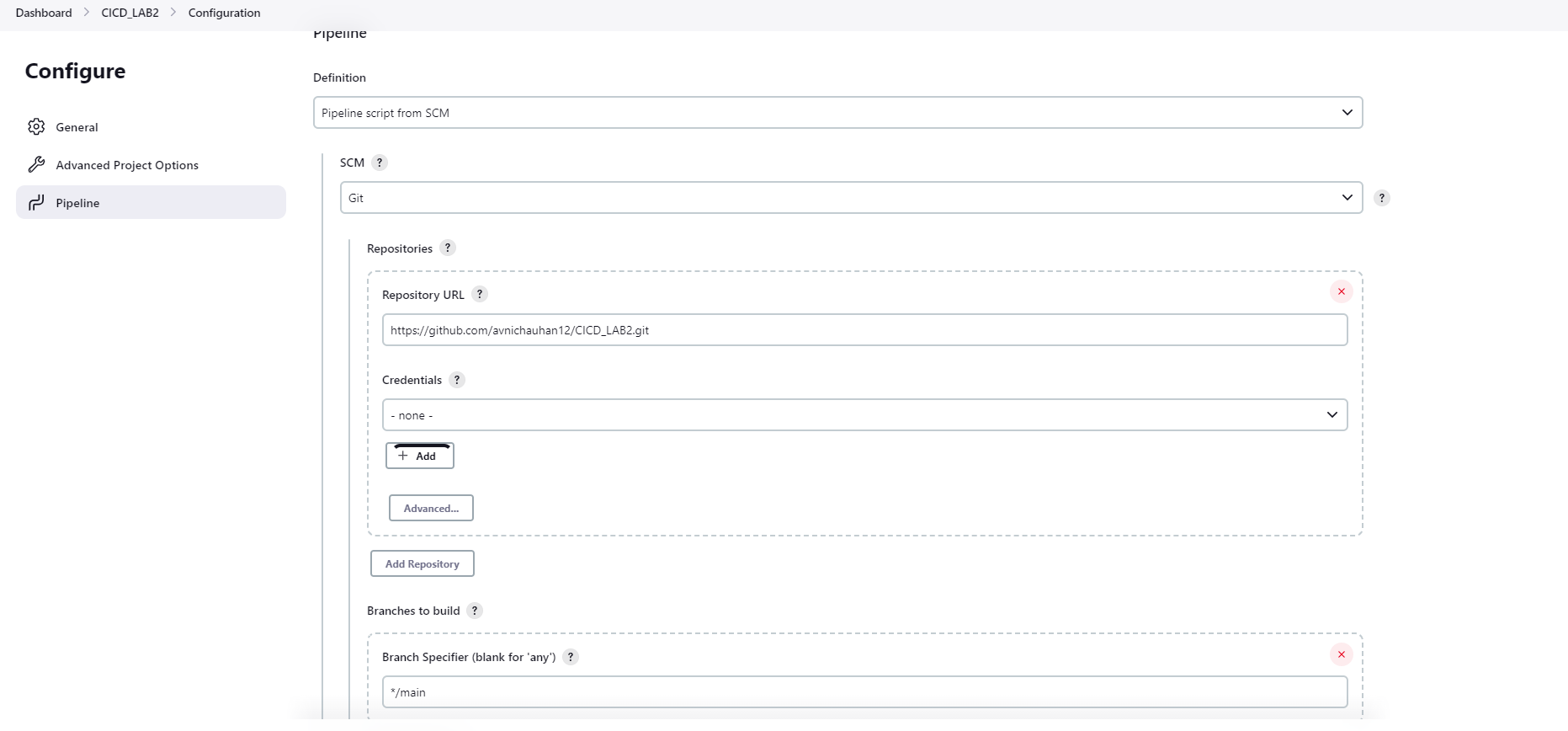
}

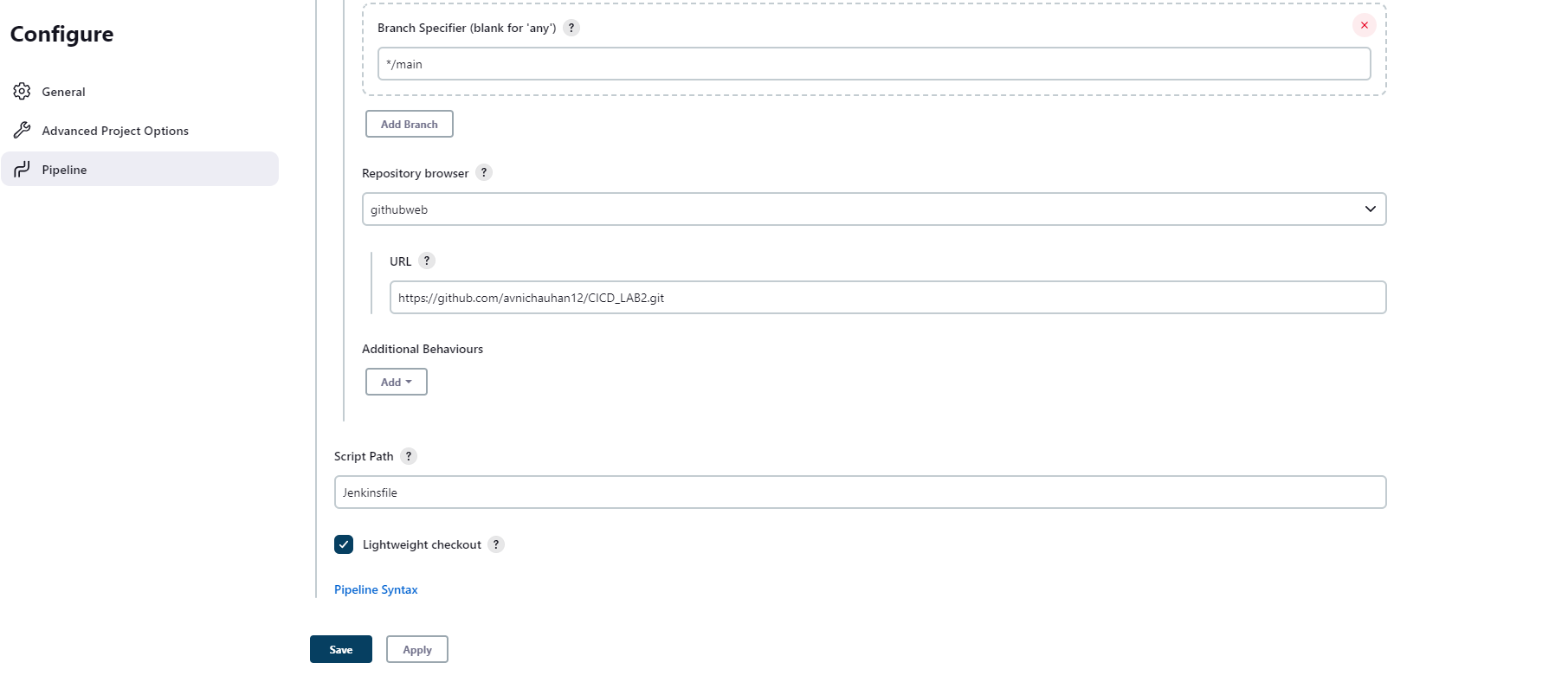
}

}

**Configuring the Pipeline in Jenkins:**

* In Jenkins, create a new pipeline job.
* Link the job to your version control repository (e.g., provide the repository URL).
* Choose the option to use a Jenkinsfile from the repository and specify the path to your Jenkinsfile (usually the root directory).

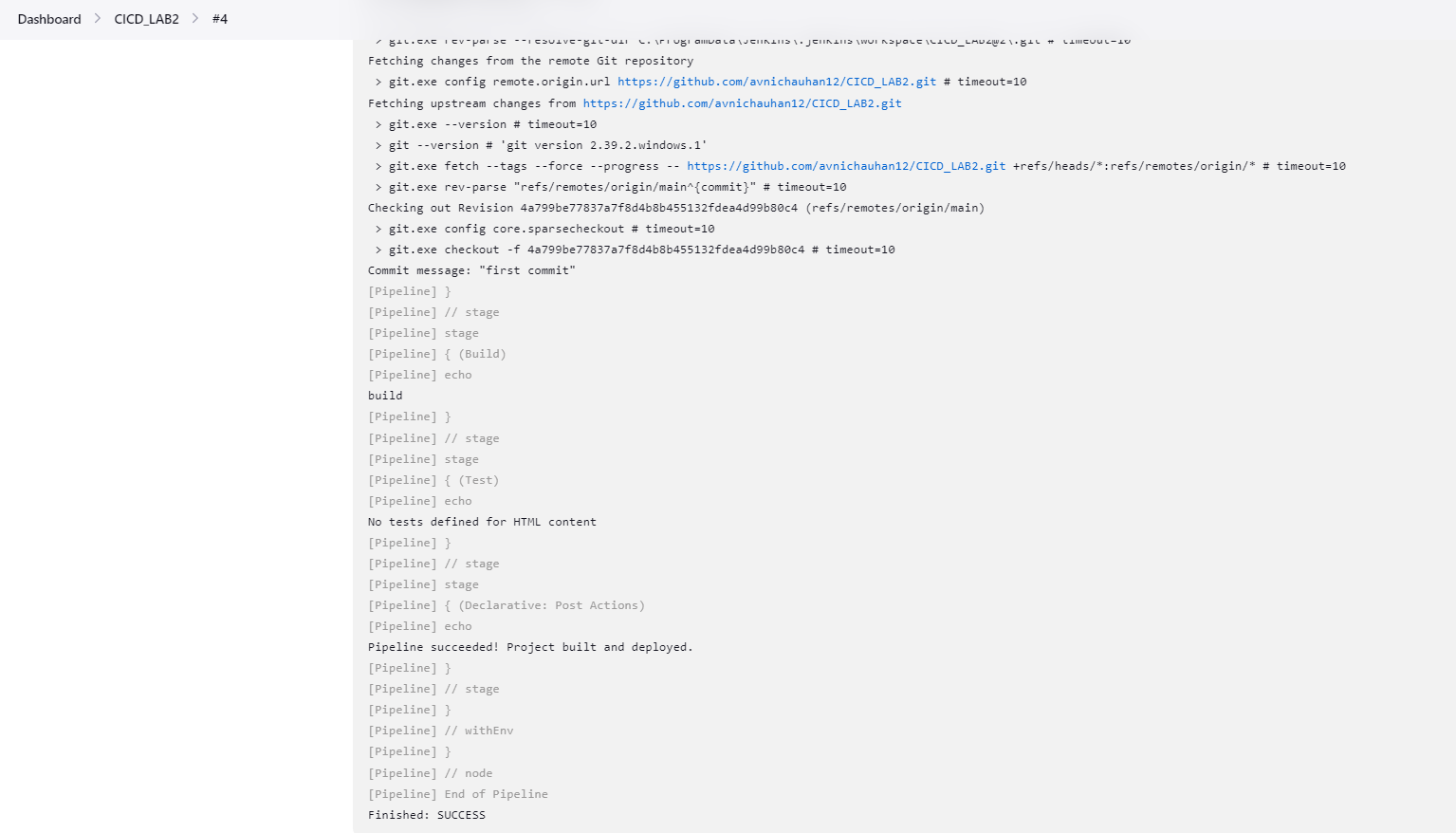


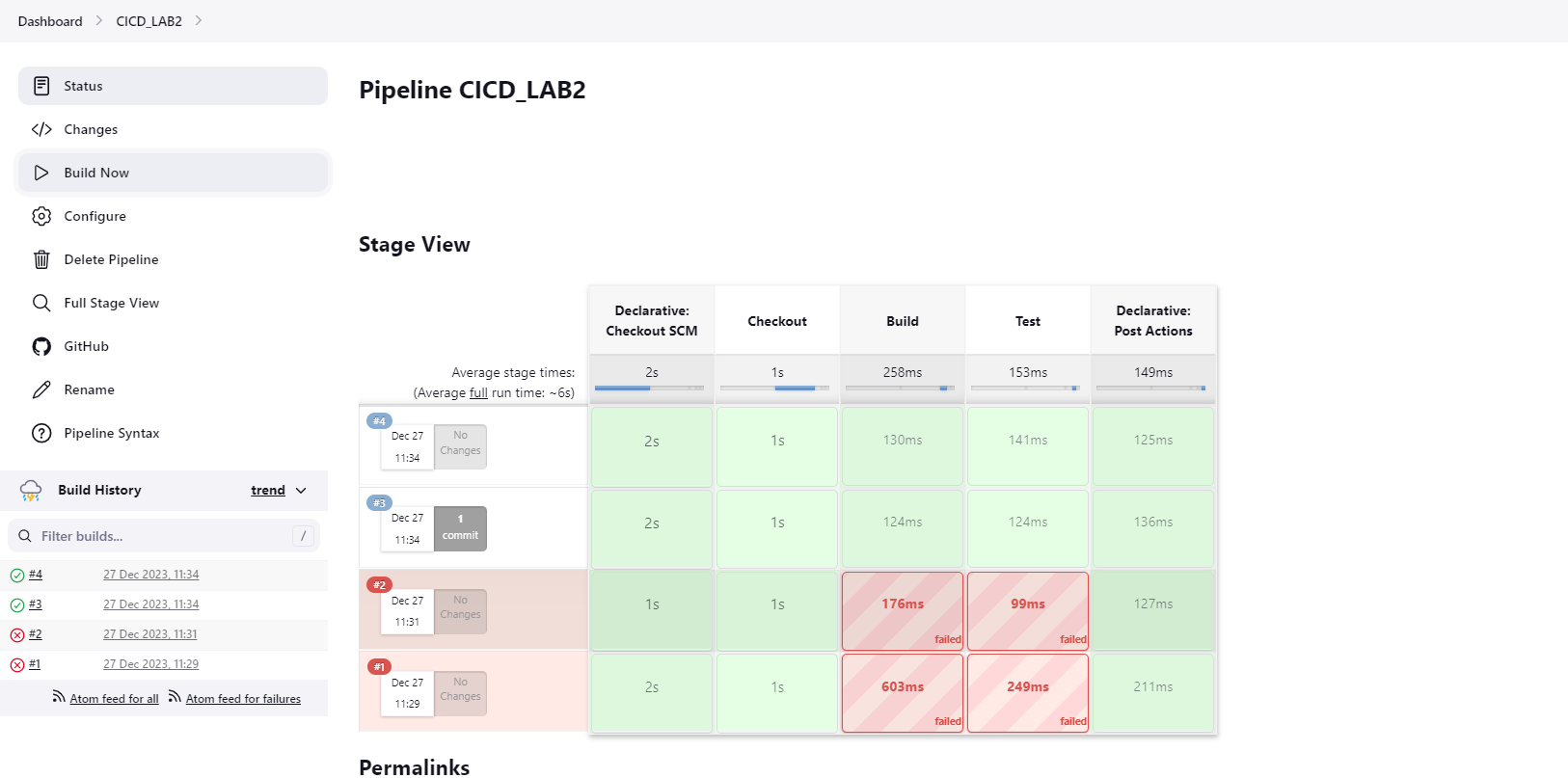


**Running the Pipeline:**

* Trigger the pipeline manually or set up a webhook to trigger it automatically on repository changes.

**Observing the Results:**

****

****