St. Francis Institute of Technology, Mumbai-400 103

**Department Of Information Technology**

A.Y. 2024-2025

Class: TE-ITA/B, Semester: V

Subject: **DevOps Lab**

**Experiment – 7: a. To build pipeline of jobs in Jenkins, create a pipeline script to test and deploy an application.**

**b. To automatically build a job in Jenkins using webhooks (Topic Beyond Syllabus)**

1. **Aim:** To build pipeline of jobs in Jenkins, create a pipeline script to test and deploy an application
2. **Objectives:** Aim of this experiment is that, the students will be able

* To build pipeline of jobs in Jenkins, create a pipeline script to test and deploy an application

1. **Outcomes:** After study of this experiment, the students will be able

* To understand the importance of Jenkins to Build and deploy Software

Applications on server environment.

1. **Prerequisite:** Knowledge of software engineering concept of integration and deployment
2. **Requirements:** Jenkins,JDK, python,Personal Computer, Windows operating system, browser, Internet Connection, Microsoft Word.
3. **Pre-Experiment Exercise:**

**Brief Theory:** Refer shared material

1. **Laboratory Exercise**
   * + 1. **Procedure:**

**a. Answer the following:**

* What is Jenkins pipeline?
* What are the different ways to write a Jenkins pipeline?

**b**. **Execute following (Refer the shared material) and attach screenshots:**

* Create and build pipeline project with Git
* Create and build pipeline project with pipeline script
* Create and automatically build a pipeline project using webhooks

1. **Post-Experiments Exercise**
2. **Extended Theory:**

Nil

1. **Questions:**

* Explain the types of agents in a Jenkinsfile?
* What are webhooks?

1. **Conclusion:**

* Write what was performed in the experiment.
* Write the significance of the topic studied in the experiment.

1. **References:**

<https://jenkins.io/doc/>

<https://www.jenkins.io/doc/book/pipeline/syntax/>

<https://www.edureka.co/blog/jenkins-pipeline-tutorial-continuous-delivery>

<https://www.slideshare.net/abediaz/introduction-to-jenkins>

<https://www.slideshare.net/jph98/jenkins-ci-presentation>

* What is Jenkins pipeline?

A **Jenkins pipeline** is a suite of plugins that allows implementing and integrating continuous integration and continuous delivery (CI/CD) workflows in Jenkins. It enables automation of building, testing, and deploying software applications. The pipeline defines a series of stages that represent various steps in the software delivery process, which can range from code compilation to deployment.

### Key Concepts in Jenkins Pipeline:

1. **Pipeline**: A user-defined model of a continuous delivery process.
2. **Stages**: The distinct phases in the pipeline, such as Build, Test, and Deploy.
3. **Steps**: Specific tasks performed at each stage, like running shell commands or invoking test scripts.
4. **Nodes**: Machines where Jenkins runs parts of the pipeline.
5. **Agent**: A machine or environment where the pipeline or its stages run.

### Benefits:

* **Automation**: Automates repetitive tasks in software development.
* **Flexibility**: Can be customized based on the needs of the project.
* **Visibility**: Provides insight into the progress and status of each stage.
* **Error handling**: Supports advanced error handling and notifications in case of failures.
* What are the different ways to write a Jenkins pipeline?

### 1. Declarative Pipeline

* **Syntax**: Structured and user-friendly.
* **Use case**: Preferred for most use cases as it is easier to read, maintain, and enforce best practices.
* **Features**: It enforces a more structured approach with stages and steps.

**Example**:

groovy

Copy code

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building...'

}

}

stage('Test') {

steps {

echo 'Testing...'

}

}

stage('Deploy') {

steps {

echo 'Deploying...'

}

}

}

}

#### Key Elements of a Declarative Pipeline:

* **pipeline block**: The main block that contains the entire pipeline definition.
* **agent**: Specifies where the pipeline will run (e.g., any machine, specific label).
* **stages**: A series of phases in the pipeline, such as Build, Test, and Deploy.
* **steps**: The tasks executed inside each stage.
* **post**: Block for defining actions after pipeline completion, such as notifications or cleanup.

### 2. Scripted Pipeline

* **Syntax**: Based on Groovy scripting and more flexible.
* **Use case**: Used when more complex logic or dynamic behavior is needed in the pipeline.
* **Features**: Provides maximum flexibility, but it's harder to maintain and prone to errors if not properly managed.

**Example**:

groovy

Copy code

node {

stage('Build') {

echo 'Building...'

}

stage('Test') {

echo 'Testing...'

}

stage('Deploy') {

echo 'Deploying...'

}

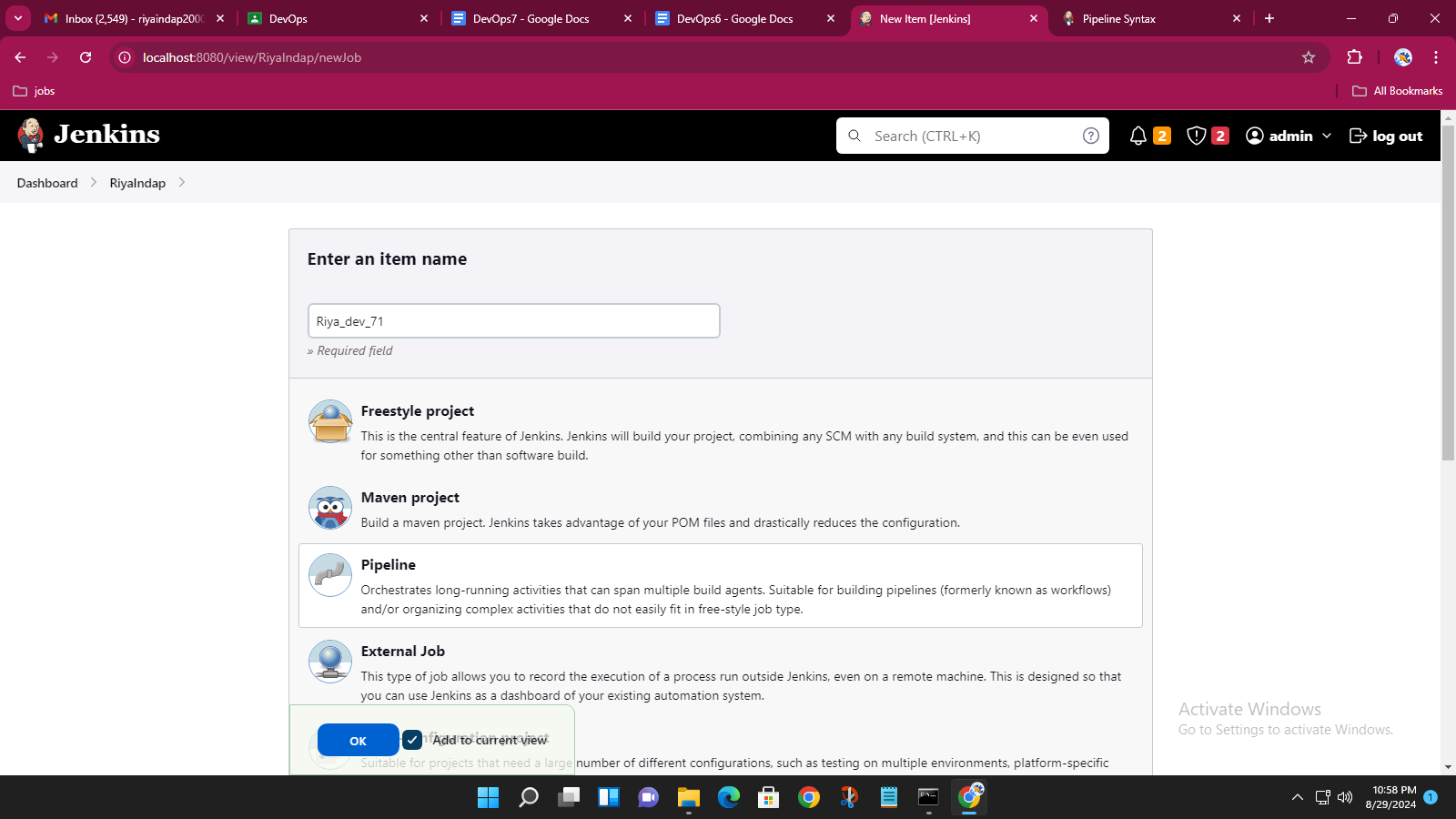
}

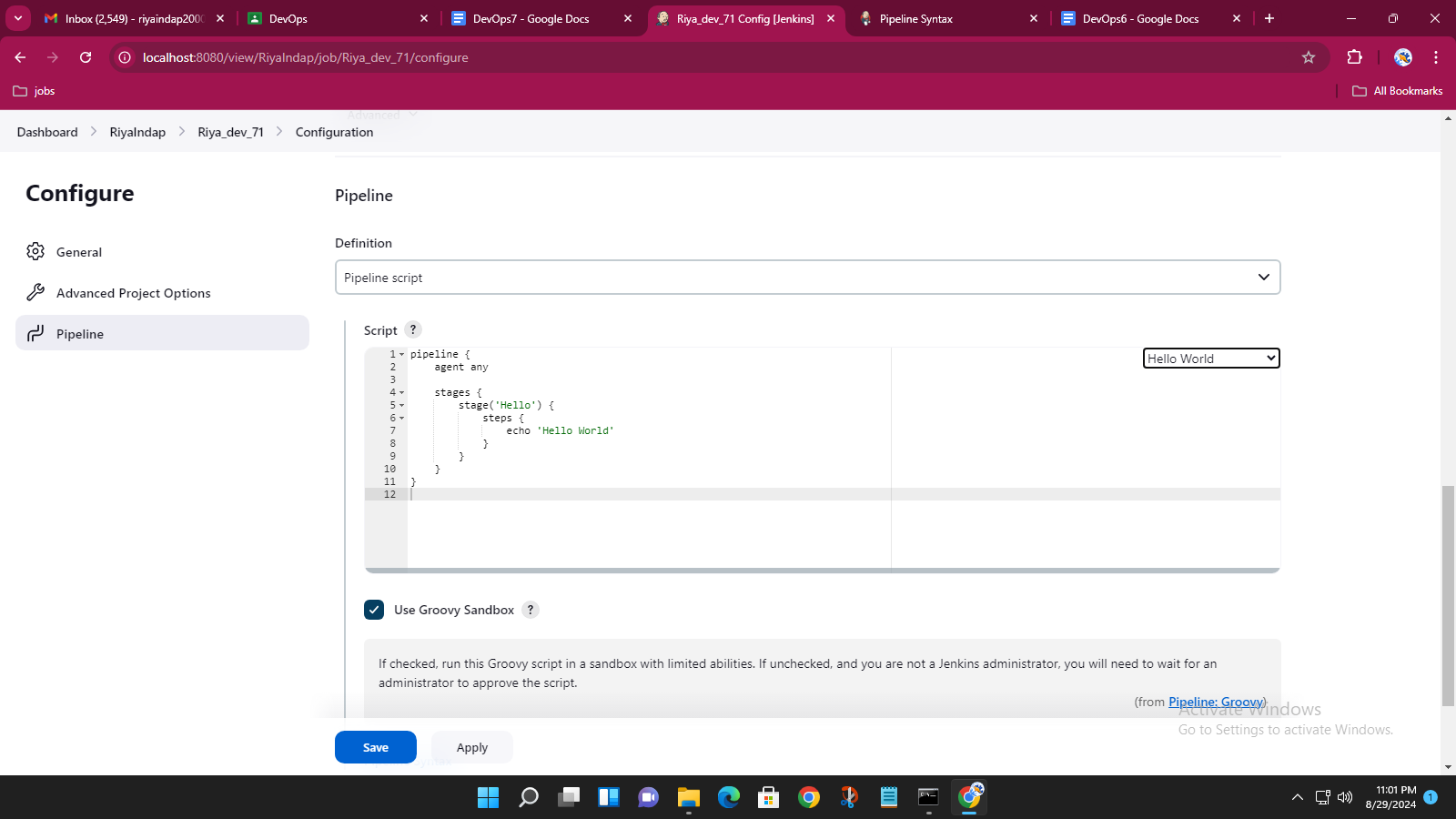
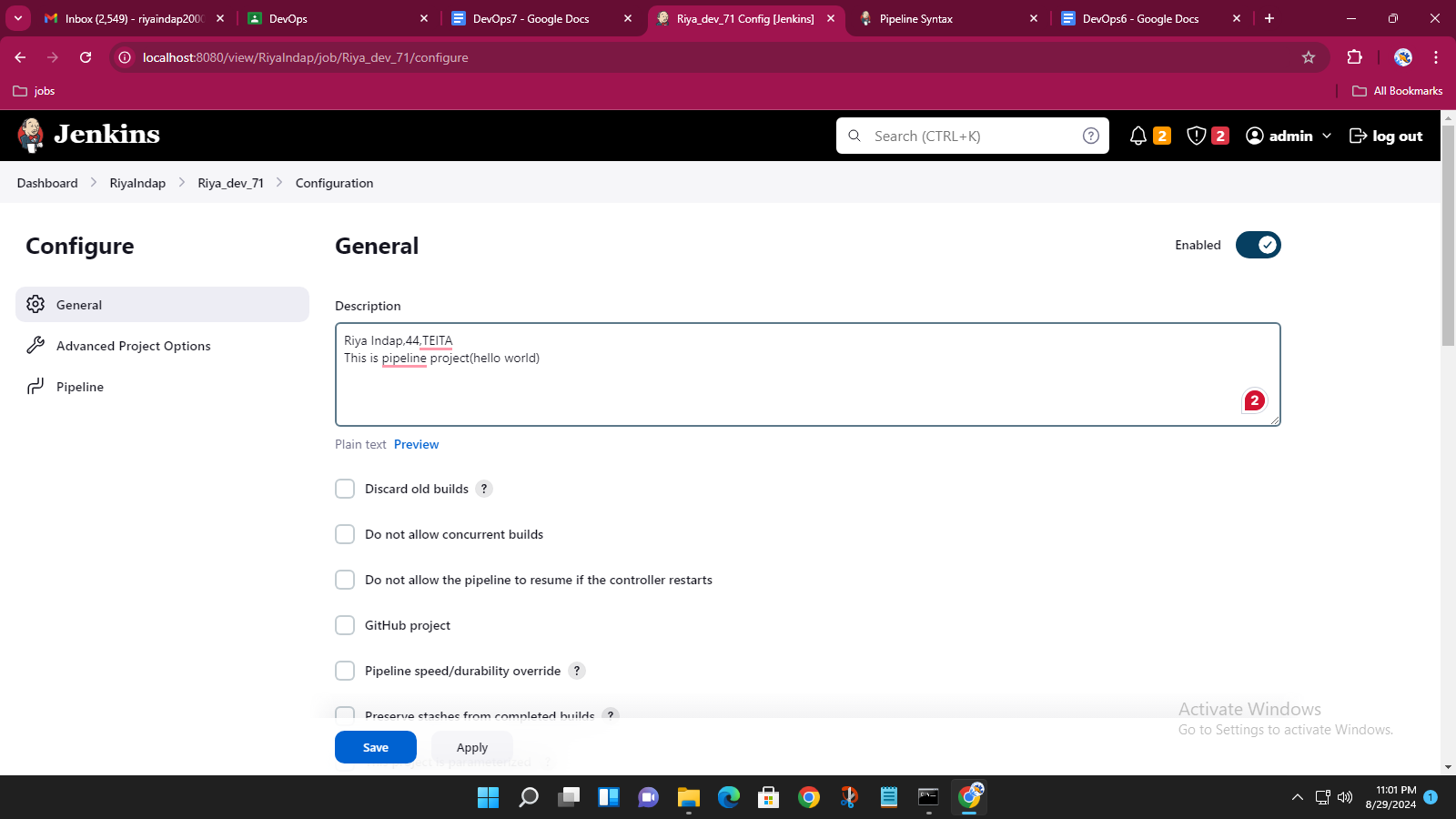
#### Key Elements of a Scripted Pipeline:

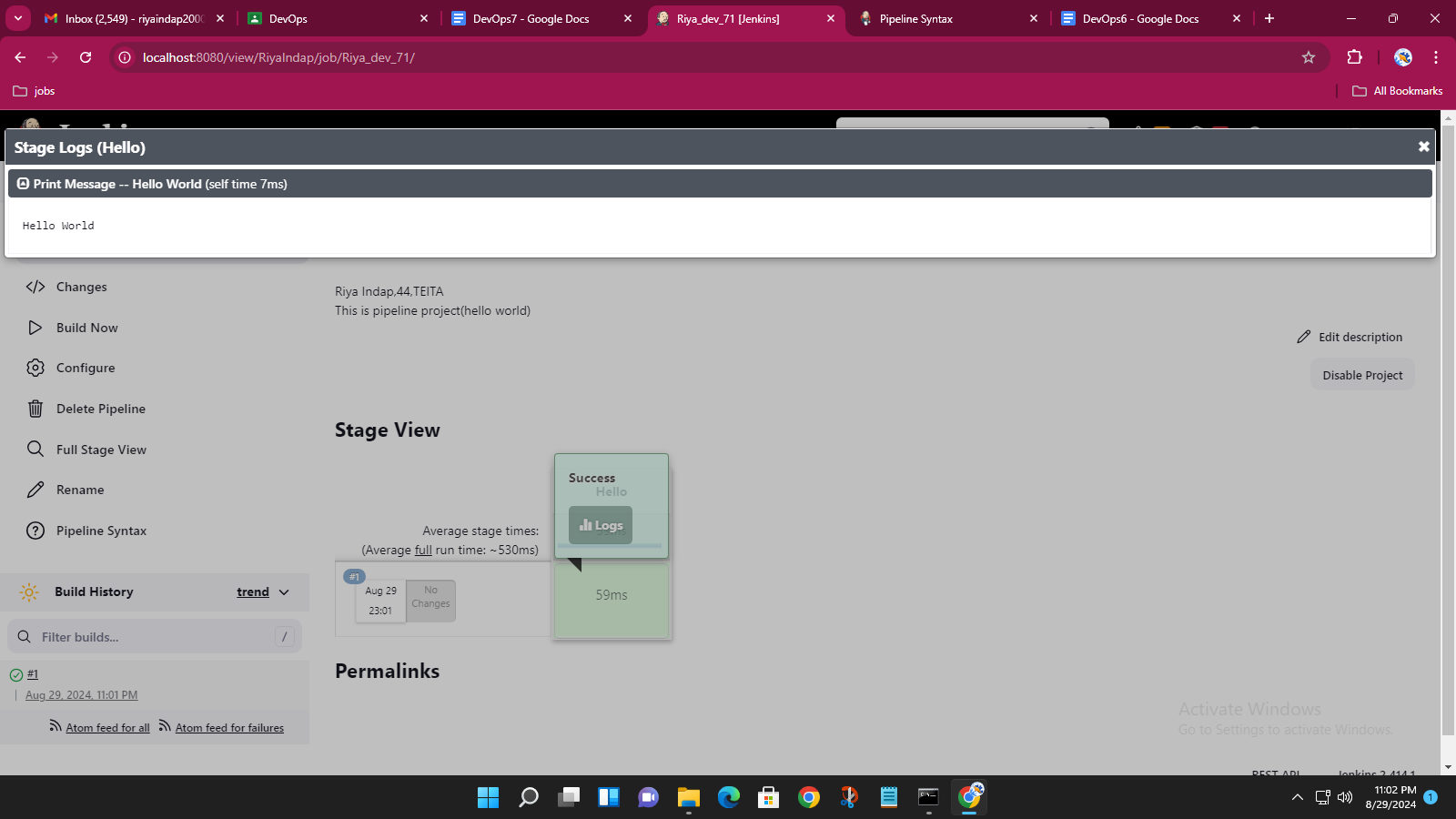
* **node**: Specifies the Jenkins agent (worker node) on which the pipeline or a part of it runs.
* **stage**: Represents each distinct phase of the pipeline.
* **Groovy scripting**: Allows using loops, conditionals, and other advanced logic for dynamic pipeline behavior.

1. Create and build pipeline project with pipeline script

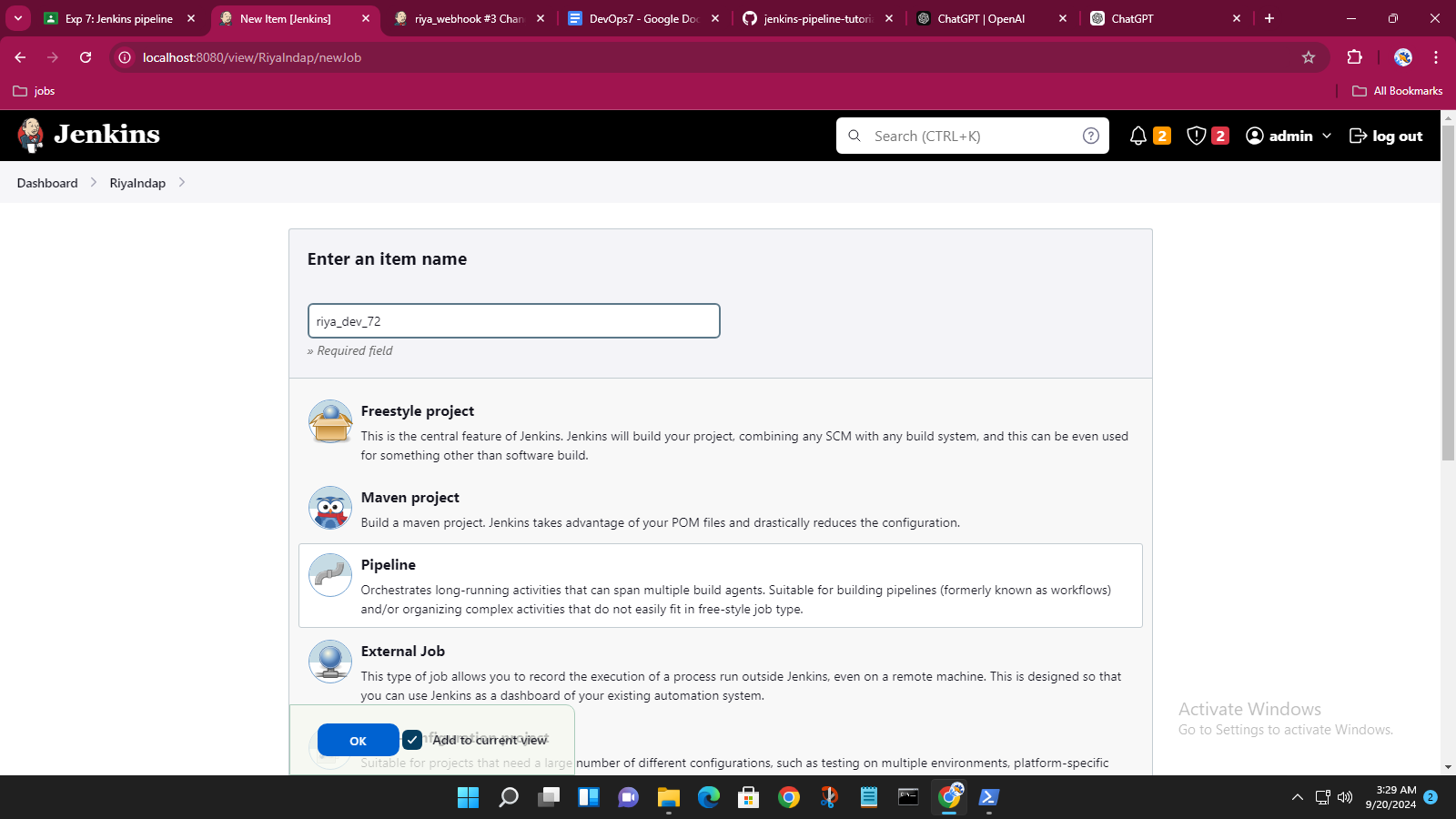
• Project with Hello World pipeline script

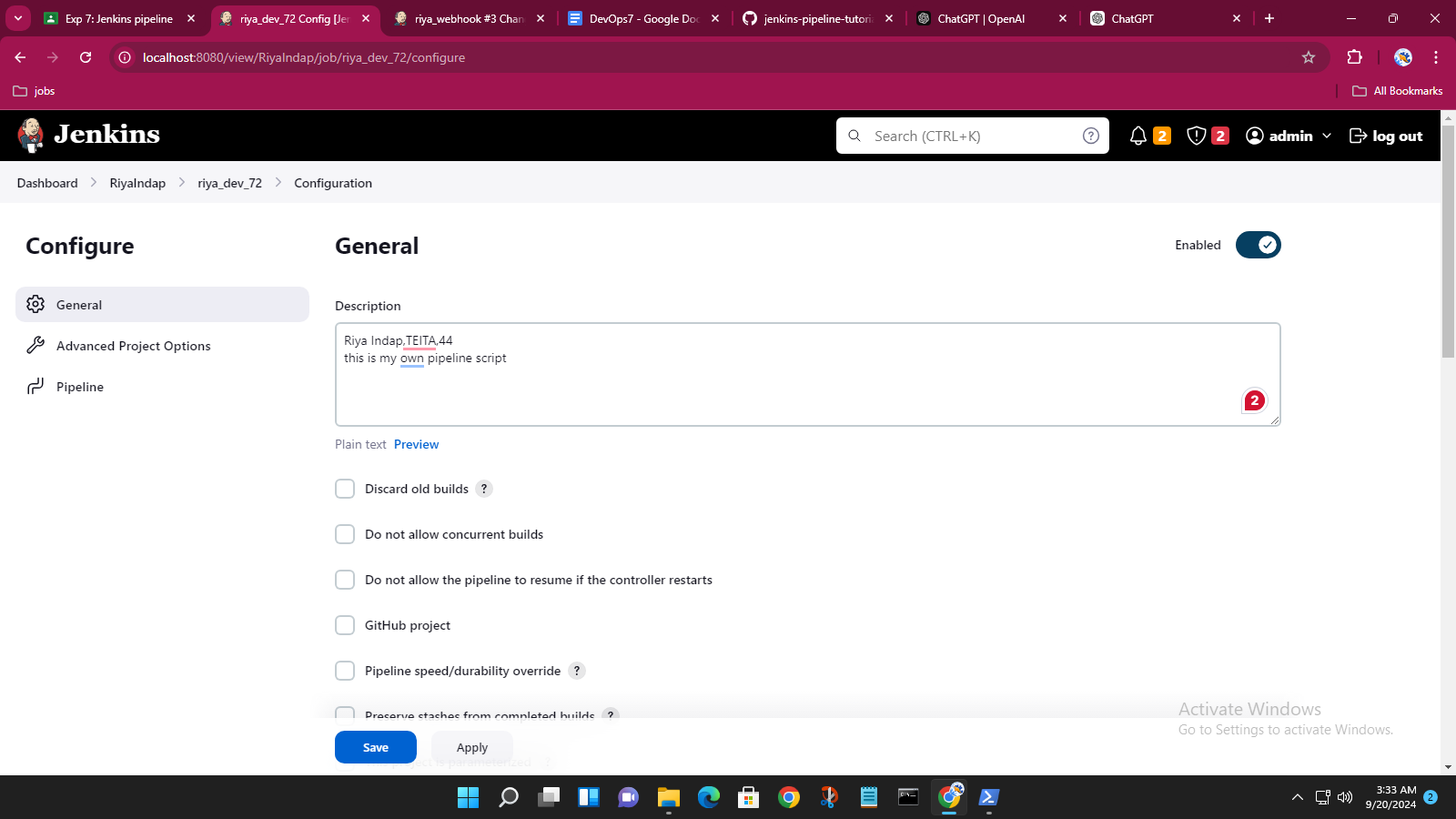


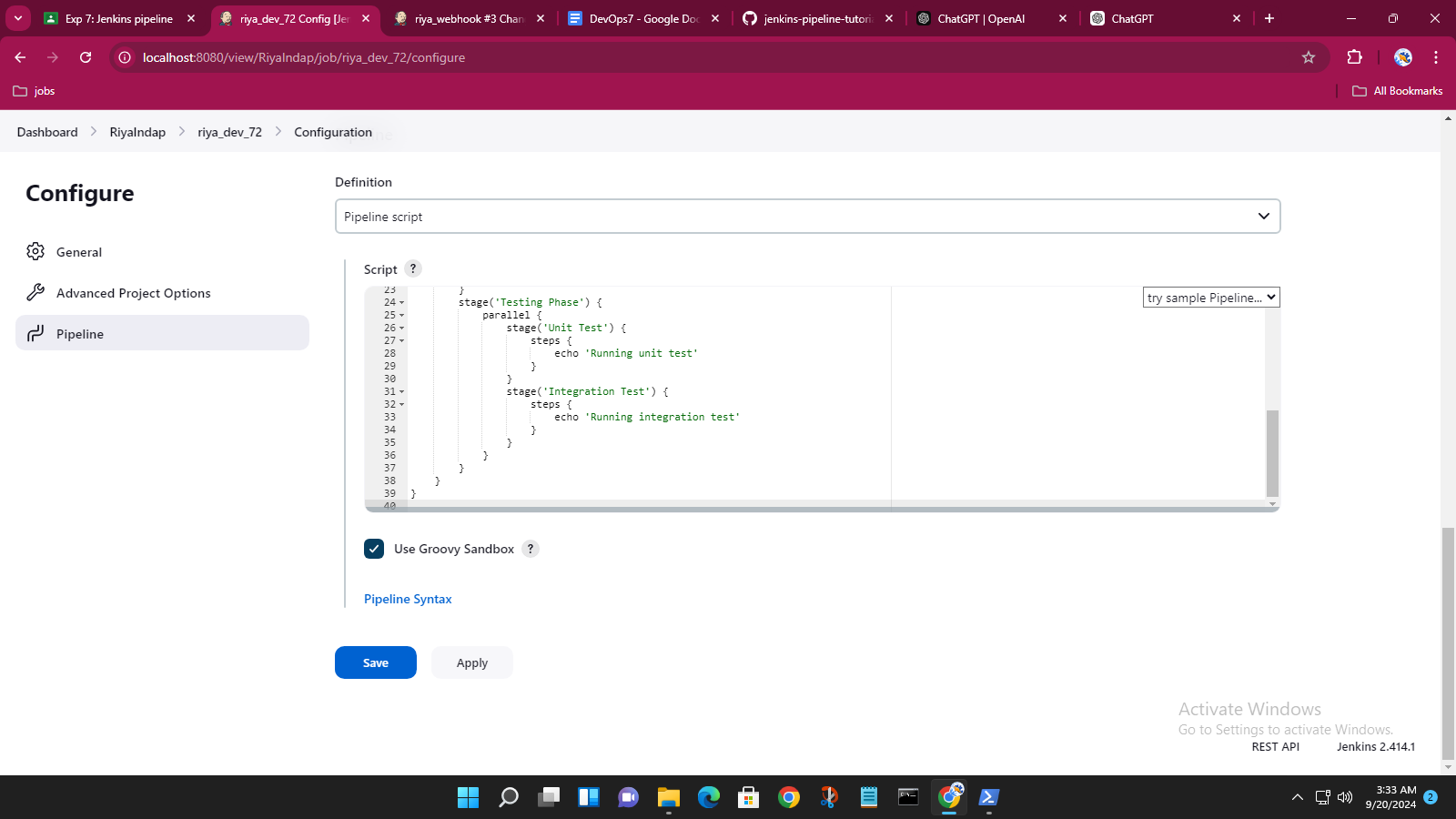


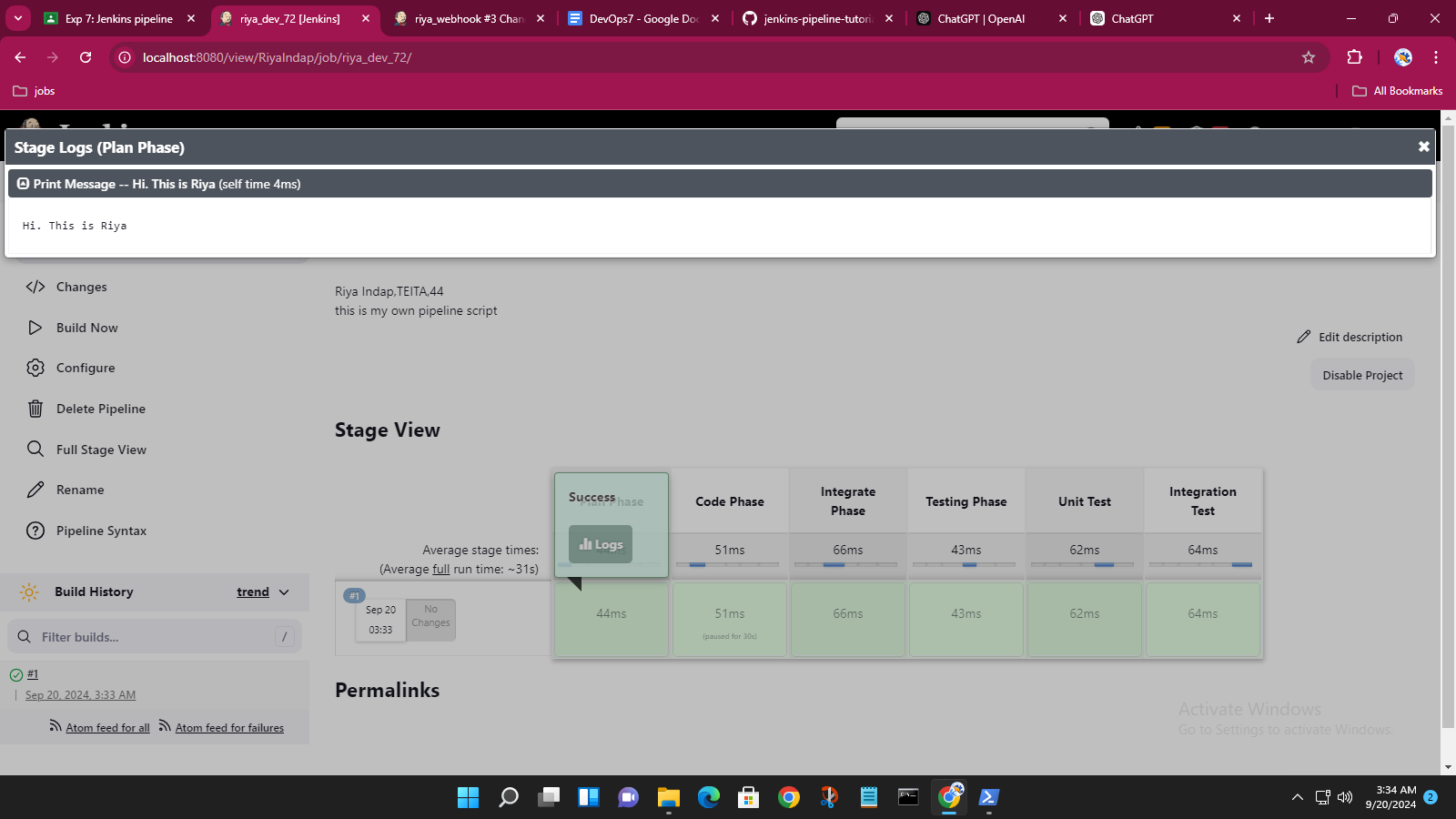


• Project with your own pipeline script



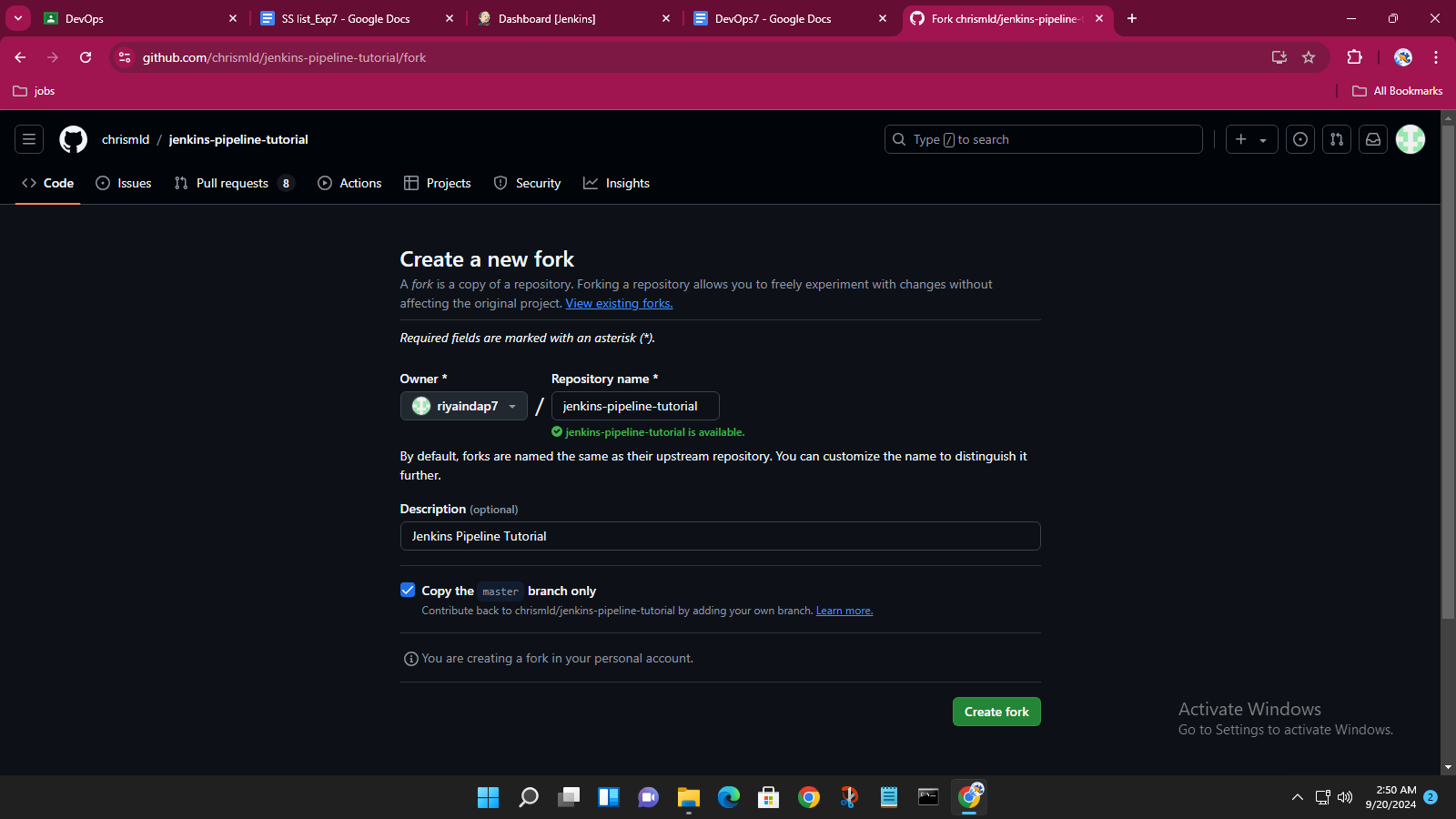


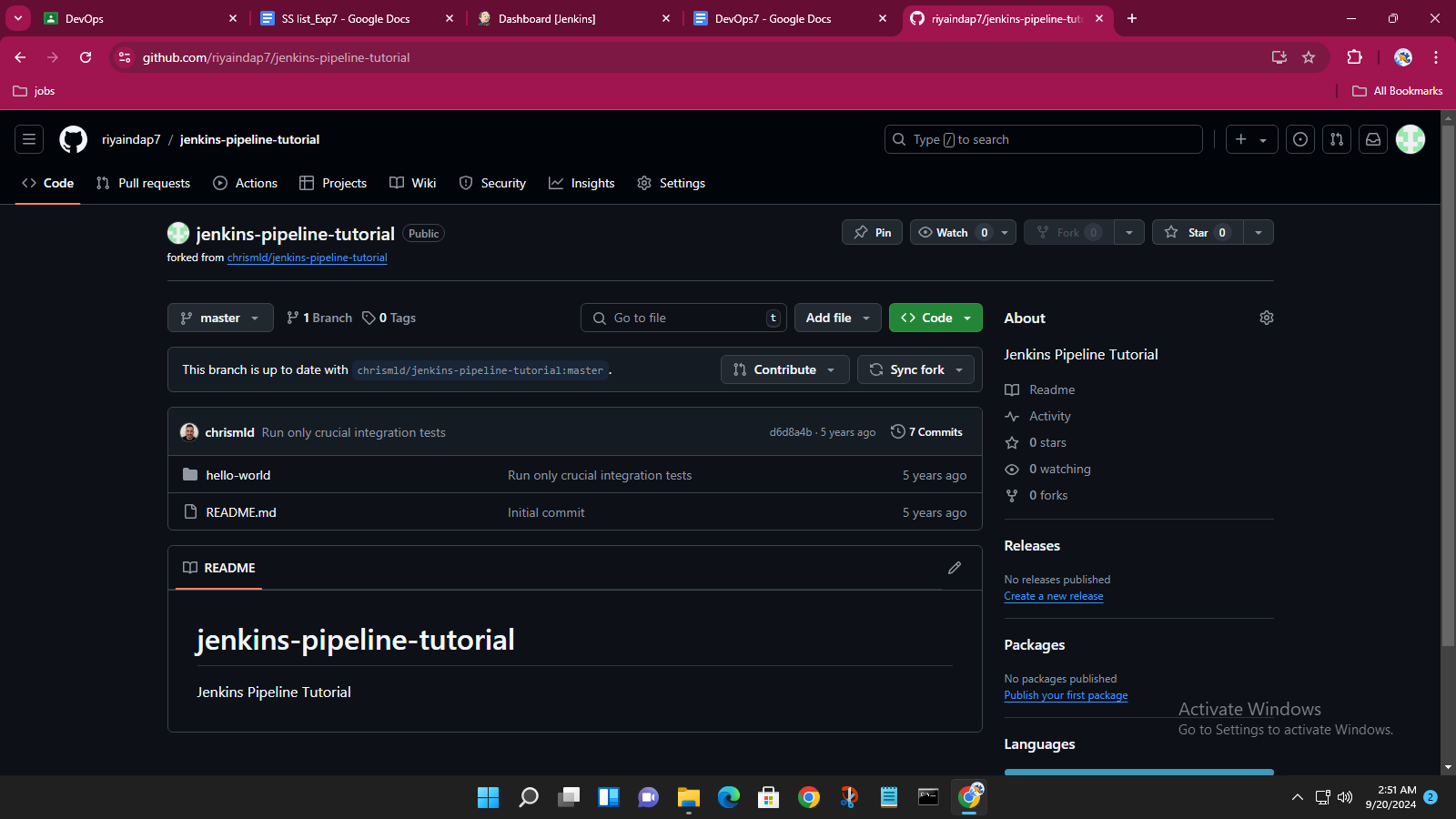


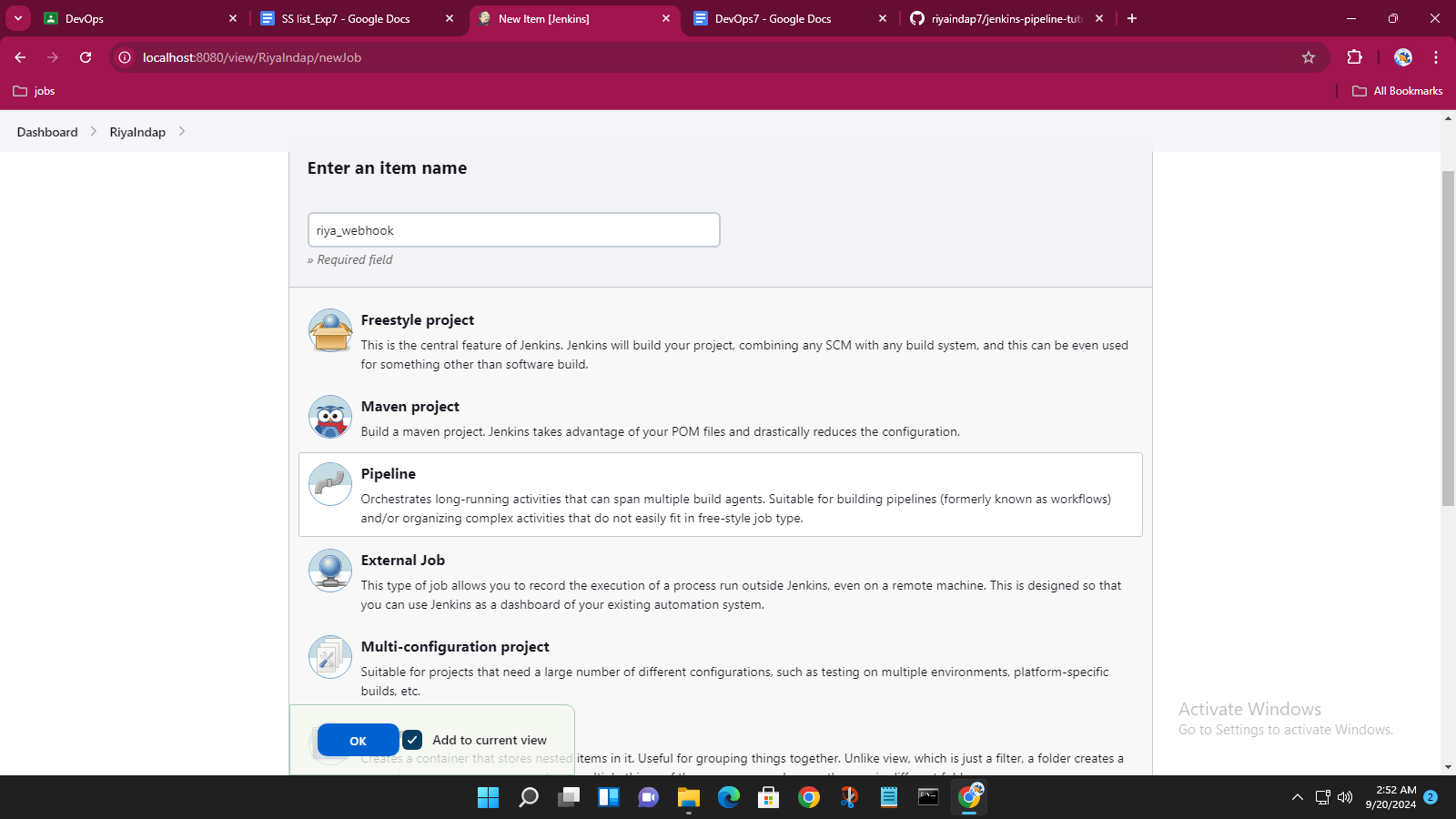


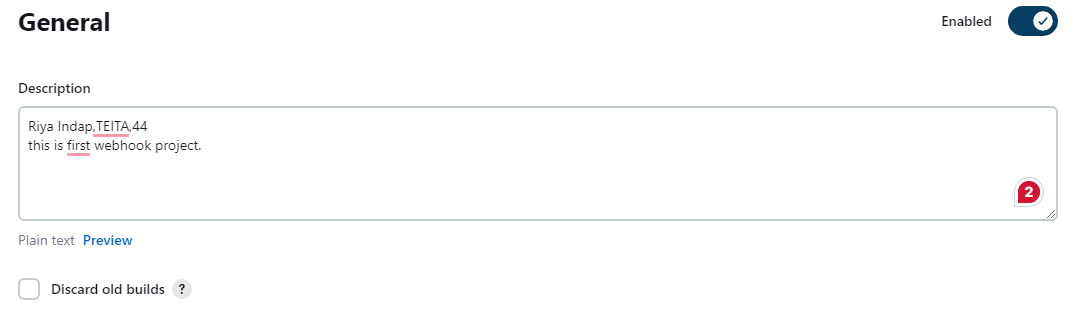
2. Create and build pipeline project with Git

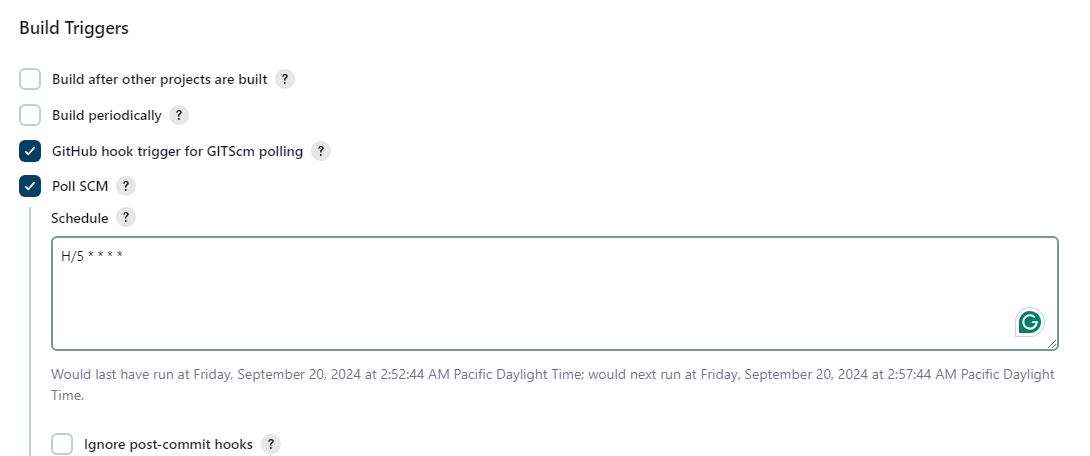
• Fork repository on GitHub

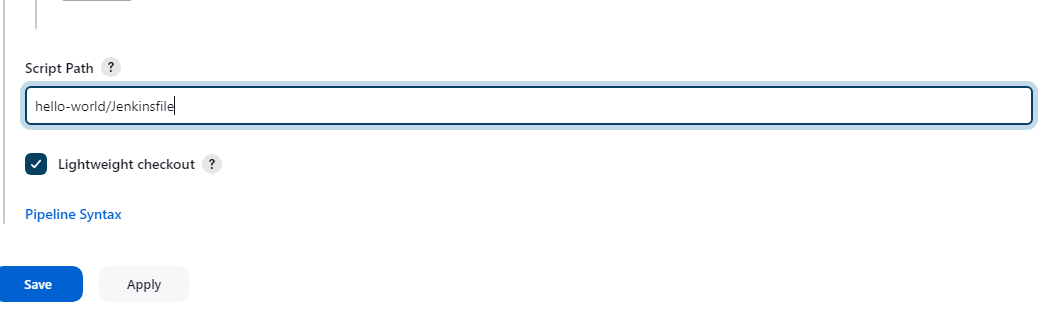


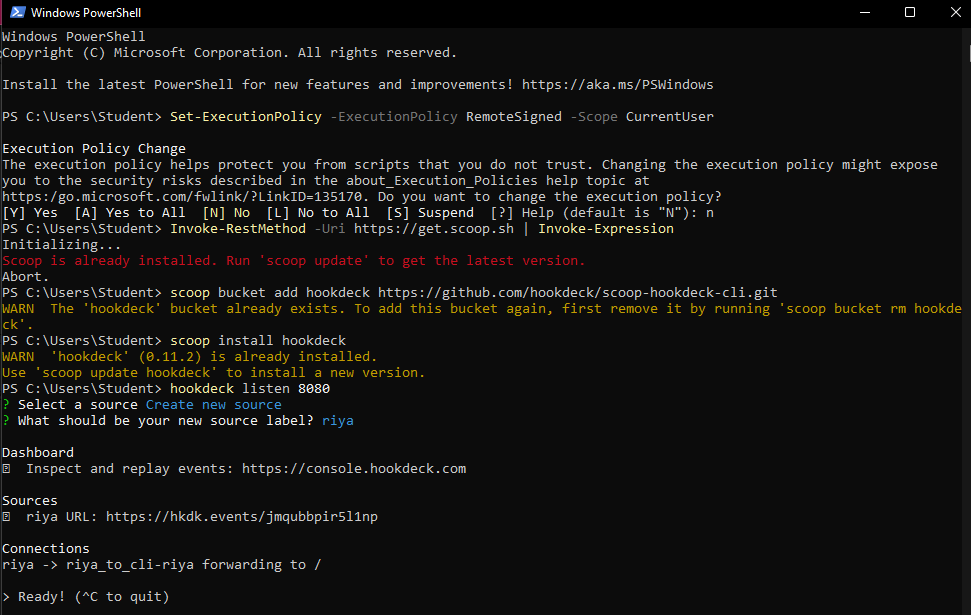


• Build pipeline project

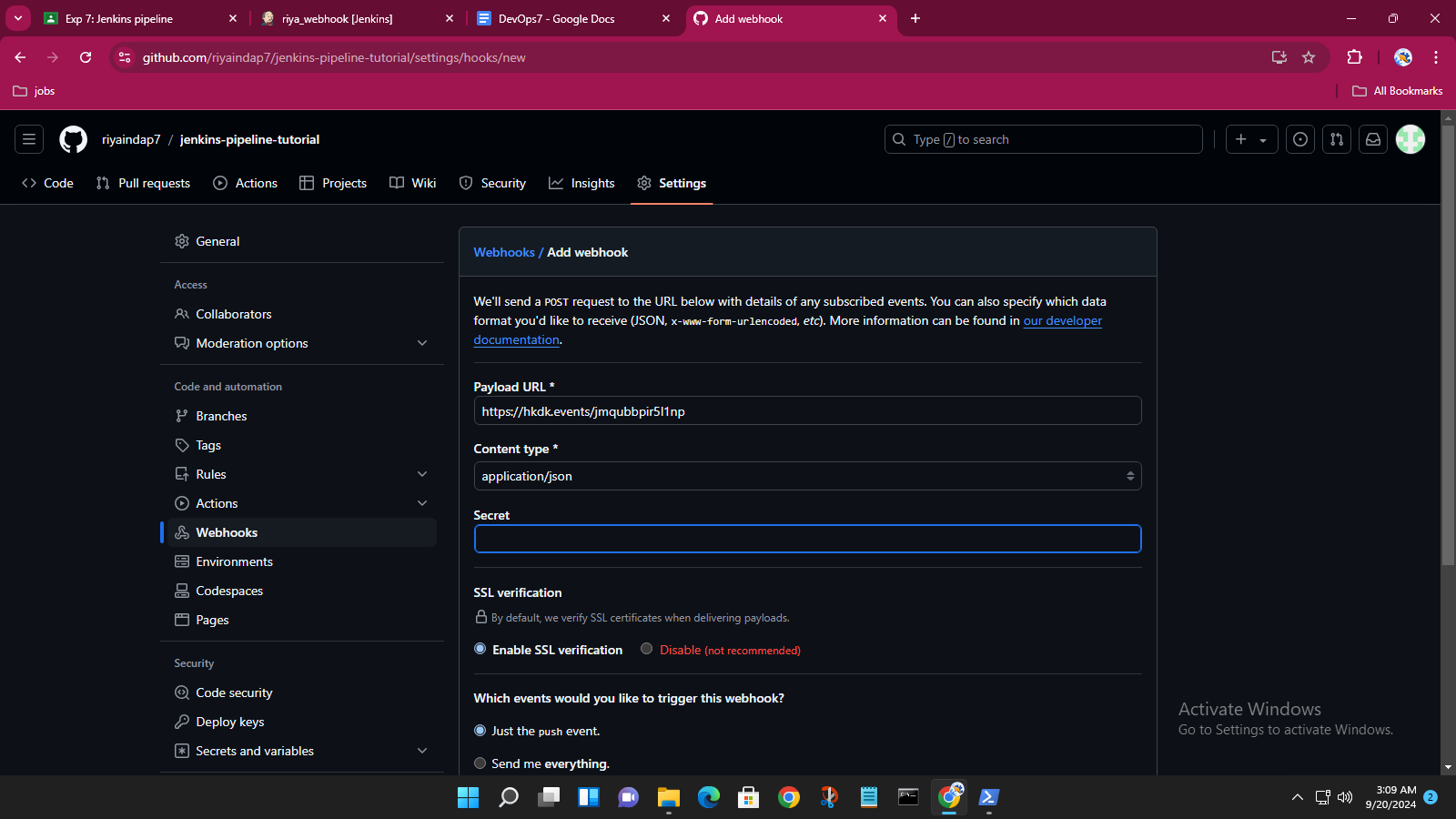




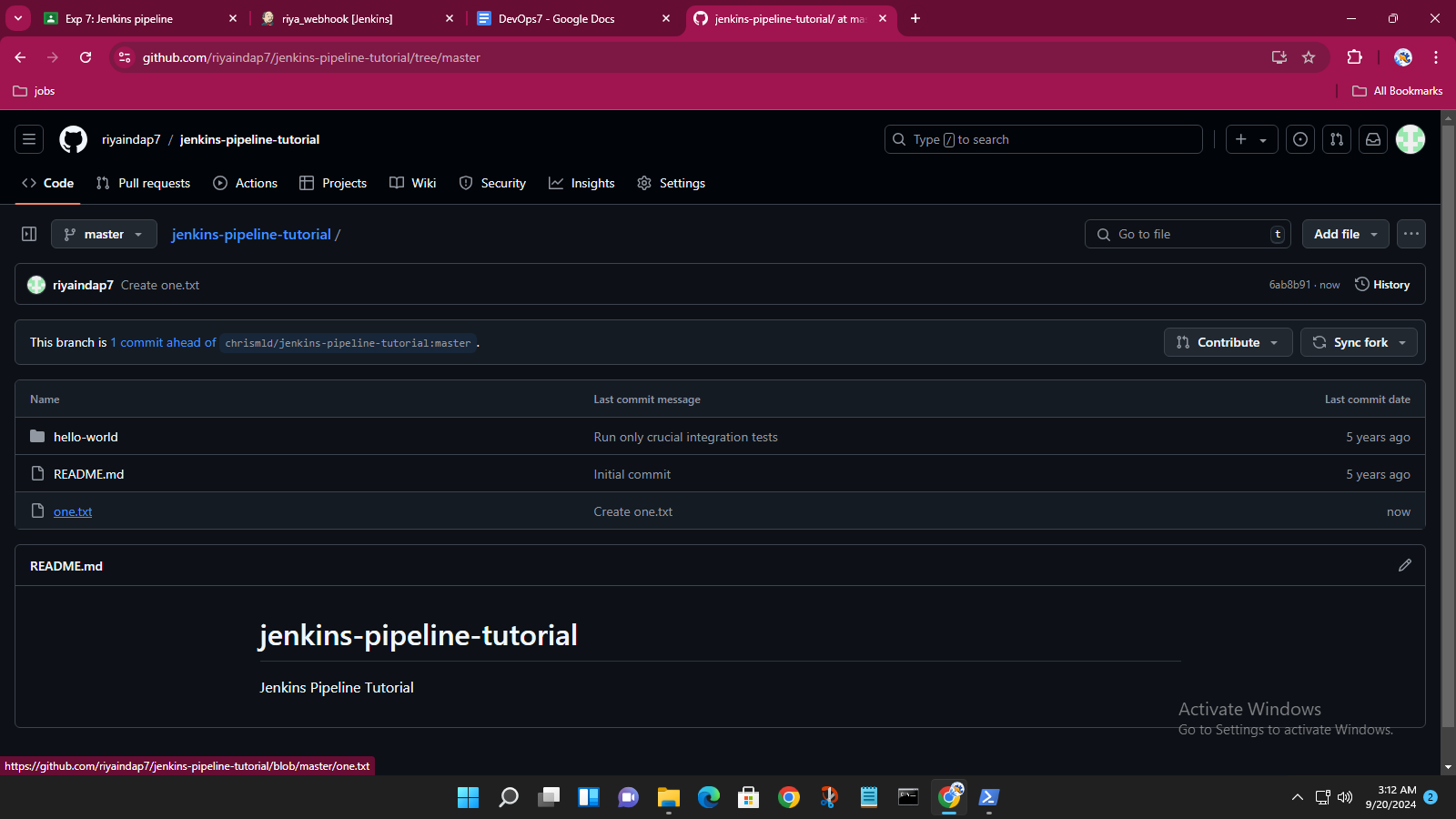
Create pipeline project with pipeline script from SCM

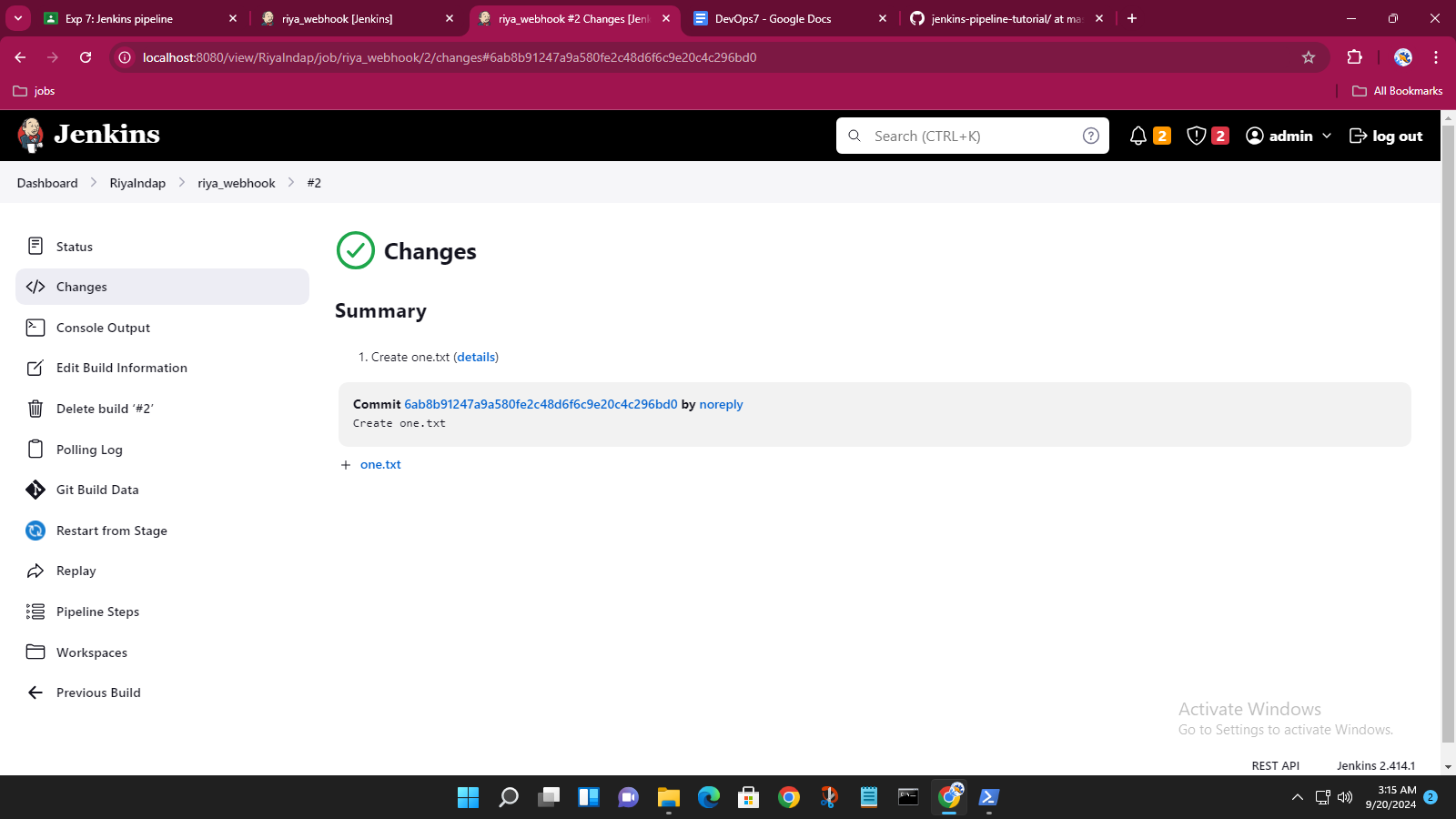


• Add webhooks to the forked repository



• Add file to forked repository and observe the automated build





• Make changes to Jenkins file on forked repository and observe the automated build

